



開發人員指南

Amazon Simple Storage Service



API 版本 2006-03-01

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Simple Storage Service: 開發人員指南

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商標和商業外觀不得用於任何非 Amazon 的產品或服務，也不能以任何可能造成客戶混淆、任何貶低或使 Amazon 名譽受損的方式使用 Amazon 的商標和商業外觀。所有其他非 Amazon 擁有的商標均為其各自擁有者的財產，這些擁有者可能隸屬於 Amazon，或與 Amazon 有合作關係，亦或受到 Amazon 贊助。

Table of Contents

什麼是 Amazon S3 ?	1
API 呼叫建議	2
進行直接 REST API 呼叫	2
許可	3
使用 Amazon S3 進行開發	4
提出要求	4
關於存取金鑰	4
要求端點	6
透過 IPv6 提出請求	6
使用 AWS SDKs 提出請求	16
使用 REST API 提出要求	52
使用 AWS CLI	63
進一步了解 AWS CLI	64
使用 AWS SDKs 進行開發	64
SDK 程式設計界面	66
在請求身分驗證中指定簽章版本	66
取得的 Amazon S3 請求 IDs 支援	74
使用 AWS CLI 取得請求 IDs	74
使用 Windows PowerShell 取得請求識別碼	74
使用 AWS CloudTrail 資料事件來取得請求 IDs	75
使用 S3 伺服器存取記錄日誌來取得請求 ID	75
使用 HTTP 取得請求 ID	75
使用 Web 瀏覽器取得請求 ID	75
使用 AWS SDKs 取得請求 IDs	76
S3 資料表支援的 S3 物件層級 API 操作	80
程式碼範例	86
Amazon S3	92
基本概念	97
案例	781
無伺服器範例	1230
Amazon S3 Control	1242
基本概念	1243
S3 目錄儲存貯體	1322
基本概念	1324

案例	1447
驗證請求AWS（簽章第 4 版）	1451
身分驗證方法	1452
簽署請求簡介	1453
使用授權標頭	1454
概觀	1454
簽章計算：在單一區塊中傳輸承載	1457
簽章計算：以多個區塊傳輸承載	1475
簽章計算：包含追蹤標頭	1487
使用查詢參數	1493
計算簽章	1495
範例	1499
範例 2	1501
範例：簽章計算	1502
使用 Java 的簽章計算範例	1502
使用 C# 的簽章計算範例	1503
驗證 HTTP POST 請求	1504
計算簽章	1506
Amazon S3 Signature 第 4 版身分驗證特定政策金鑰	1508
使用 Signature 第 4 版相關條件索引鍵的儲存貯體政策範例	1510
使用 POST 的瀏覽器型上傳	1513
POST 物件	1514
說明	1514
版本控制	1515
要求	1515
範例	1530
相關資源	1531
POST 物件還原	1532
說明	1532
使用選取請求查詢封存	1532
還原封存	1533
要求	1534
回應	1549
範例	1550
詳細資訊	1552
使用 HTTP POST 的瀏覽器型上傳	1552

計算簽章	1554
建立 HTML 表單	1555
HTML 表單宣告	1556
HTML 表單欄位	1556
POST 政策	1562
Expiration	1563
條件比對	1563
條件	1564
字元逸出	1568
POST 上傳範例	1570
使用 HTTP POST 將檔案上傳至 Amazon S3	1570
使用 AWS Amplify 的瀏覽器型上傳	1572
使用 AWS Amplify JavaScript 程式庫將檔案上傳至 Amazon S3	1573
詳細資訊	1574
常見請求標頭	1575
常見回應標頭	1578
錯誤回應	1582
REST 錯誤回應	1583
錯誤代碼清單	1584
SELECT 物件內容錯誤代碼的清單	1607
複寫相關錯誤代碼的清單	1618
標記相關錯誤代碼的清單	1619
Amazon S3 註釋相關錯誤代碼的清單	1620
Amazon S3 on Outposts 錯誤代碼清單	1623
Amazon S3 Storage Lens 錯誤代碼清單	1624
Amazon S3 Object Lambda 錯誤代碼清單	1630
Amazon S3 非同步錯誤代碼清單	1633
Amazon S3 Access Grants 錯誤代碼清單	1634
Amazon FSx 相關錯誤代碼的清單	1637
Amazon S3 Tables 錯誤代碼清單	1638
Amazon S3 錯誤的最佳實務	1638
重試 InternalError	1639
針對重複出現的 SlowDown 錯誤微調應用程式	1639
隔離錯誤	1639
AWS 詞彙表	1641
Resources	1642

附錄	1644
附錄：SelectObjectContent 回應	1645
說明	1645
回應	1645
相關資源	1654
附錄：OPONS 物件	1655
說明	1655
要求	1655
回應	1656
範例	1657
相關資源	1658
附錄：SOAP API	1658
服務上的操作 (SOAP API)	1659
儲存貯體上的操作 (SOAP API)	1661
物件上的操作 (SOAP API)	1673
對 SOAP 要求進行身分驗證	1696
使用 SOAP 設定存取原則	1698
常見元素	1699
SOAP 錯誤回應	1699
附錄：驗證請求 (AWS 簽章第 2 版)	1701
使用 REST API 驗證請求 (AWS 簽章版本 2)	1703
簽署和驗證 REST 請求 (AWS 簽章版本 2)	1706
使用 POST 的瀏覽器型上傳 (AWS 簽章版本 2)	1719
附錄：生命週期組態 APIs (已棄用)	1742
PUT 儲存貯體生命週期 (已棄用)	1743
GET 儲存貯體生命週期 (已棄用)	1759
文件歷史紀錄	1770
.....	mdcclxxi

什麼是 Amazon S3 ?

Amazon Simple Storage Service (Amazon S3) 是一項物件儲存服務，提供領先業界的可擴展性、資料可用性、安全性和效能。所有規模與產業的客戶都可使用 Amazon S3 來存放和保護適用於各種使用案例的任意資料量，例如資料湖、網站、行動應用程式、備份與還原、封存、企業應用程式、IoT 裝置和大數據分析。Amazon S3 提供管理功能，讓您可以最佳化、組織和設定對資料的存取，從而滿足您的特定業務、組織和合規要求。

本指南提供有關使用 Amazon S3 REST API 來建置在雲端中存放和擷取資料的應用程式，以開發人員為中心的資訊。

您可以使用支援 HTTP 的任何工具組來使用 REST API。您甚至可以使用瀏覽器來擷取物件，只要物件是可匿名讀取即可。

REST API 使用標準 HTTP 標頭與狀態碼，因此標準瀏覽器與工具組會如預期般運作。在某些區域中，我們新增了 HTTP 的功能（例如，我們新增了標頭以支援存取控制）。在這些情況下，我們新增了符合標準 HTTP 用量樣式的新功能。

Note

透過 HTTP 的 SOAP 支援已棄用，但仍可透過 HTTPS 使用。不過，SOAP 不支援新的 Amazon S3 功能。建議您透過以下連結使用此 REST API 或 AWS SDKs：

<https://aws.amazon.com/developer/tools/>

此開發人員指南包括：

- [常見請求標頭](#) 和 [常見回應標頭](#)
- [錯誤回應](#)
- [使用 POST 的瀏覽器型上傳AWS（簽章第 4 版）](#)

Important

在前往特定 API 主題之前，請閱讀以下有關身分驗證和存取控制的資訊。

對 Amazon S3 的請求可以進行身分驗證或匿名。已驗證的存取需要 AWS 登入資料，可用來驗證您的請求。

API 呼叫建議

直接從程式碼進行 REST API 呼叫可能會很麻煩。它要求您撰寫必要的程式碼來計算有效的簽章，以驗證您的請求。我們建議您改用下列替代方法：

- 使用 [AWS SDKs](#) 傳送您的請求。

此外，請參閱[範例程式碼和程式庫](#)。

如果您使用 SDKs，則不需要編寫程式碼來計算請求身分驗證的簽章，因為 SDK 用戶端會使用您提供的存取金鑰來驗證您的請求。除非您有充分理由不這麼做，否則您應該一律使用 AWS SDKs。

- 使用 AWS CLI 進行 Amazon S3 API 呼叫。如需設定 CLI 和範例 Amazon S3 命令的相關資訊，AWS 請參閱下列主題：

《Amazon Simple Storage Service [AWS 使用者指南](#)》中的設定 CLI。

AWS Command Line Interface 《使用者指南》中的[將 Amazon S3 與 AWS 命令列界面](#)搭配使用。

進行直接 REST API 呼叫

Note

PUT 要求標頭的大小限制為 8 KB。在 PUT 要求標頭中，系統定義中繼資料的大小限制為 2 KB。測量系統定義中繼資料大小的方式，是透過計算每個索引鍵和值之 US-ASCII 編碼中的位元組數目加總。

如果您想要進行自己的 REST API 呼叫，而不是使用上述其中一個替代方案，請謹記一些事項。

- 若要從您的程式碼進行直接 REST API 呼叫，請使用有效的登入資料建立簽章，並在請求中包含簽章。如需各種身分驗證方法和簽章計算的資訊，請參閱 [驗證請求AWS \(簽章第 4 版\)](#)。
- REST API 使用標準 HTTP 標頭和狀態碼，因此標準瀏覽器和工具組可如預期運作。在某些區域中，我們新增了 HTTP 的功能（例如，我們新增了標頭以支援存取控制）。在這些情況下，我們新增了符合標準 HTTP 用量樣式的新功能。如需提出請求的詳細資訊，請參閱[提出要求](#)。

許可

您可以擁有有效的登入資料來驗證請求，但除非您擁有帳戶擁有者或儲存貯體擁有者的 S3 許可，否則無法建立或存取 Amazon S3 資源。這些許可通常透過 AWS Identity and Access Management (IAM) [政策](#) 授予，例如儲存貯體政策。例如，您必須擁有建立 S3 儲存貯體或取得儲存貯體中物件的許可。如需 S3 許可的完整清單，請參閱 [Amazon S3 的動作、資源和條件索引鍵](#)。

如需 S3 資源類型對 S3 API 操作之許可的詳細資訊，請參閱《Amazon Simple Storage Service [使用者指南](#)》中的 [Amazon S3 API 操作的必要許可](#)。

如果您使用的根使用者登入資料 AWS 帳戶，則您擁有所有許可。不過，不建議使用根使用者登入資料。反之，我們建議您在帳戶中建立 AWS Identity and Access Management (IAM) 角色並管理使用者許可。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [存取管理](#)。

使用 Amazon S3 進行開發

本節涵蓋使用 Amazon S3 的開發人員相關主題。如需詳細資訊，請檢閱下列主題。

主題

- [提出要求](#)
- [使用 使用 Amazon S3 進行開發 AWS CLI](#)
- [使用 AWS SDKs 使用 Amazon S3 進行開發](#)
- [取得的 Amazon S3 請求 IDs AWS 支援](#)
- [SAmazon S3 S3 物件層級 API 操作](#)

提出要求

Amazon S3 是一個 REST 服務。您可以使用 REST API 或包裝基礎 Amazon S3 REST API 的 AWS SDK（請參閱[範例程式碼和程式庫](#)）包裝函式程式庫，將請求傳送至 Amazon S3，簡化您的程式設計任務。

與 Amazon S3 的每次互動，可以經過驗證身分或是匿名進行。身分驗證是確認嘗試存取 Amazon Web Services (AWS) 產品之申請者身分的過程。經過身分驗證的要求，必須包含能驗證要求傳送者身分的簽章值。簽章值部分是從請求者的 AWS 存取金鑰（存取金鑰 ID 和私密存取金鑰）產生的。如需取得存取金鑰的詳細資訊，請參閱《AWS 一般參考》中的[如何取得安全憑證？](#)。

如果您使用 AWS SDK，程式庫會從您提供的金鑰計算簽章。但如果直接在應用程式中呼叫 REST API，您必須撰寫程式碼來運算簽章，並將其新增至要求。

主題

- [關於存取金鑰](#)
- [要求端點](#)
- [透過 IPv6 向 Amazon S3 提出請求](#)
- [使用 AWS SDKs 提出請求](#)
- [使用 REST API 提出要求](#)

關於存取金鑰

以下各節會檢閱您可用於驗證要求的存取金鑰類型。

AWS 帳戶 存取金鑰

帳戶存取金鑰提供帳戶所擁有 AWS 資源的完整存取權。以下為存取金鑰範例：

- 存取金鑰 ID (20 個字元的英數字串)。例如：AKIAIOSFODNN7EXAMPLE
- 私密存取金鑰 (40 個字元的字串)。例如：wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

存取金鑰 ID 可唯一識別 AWS 帳戶。您可以使用這些存取金鑰，將驗證請求傳送至 Amazon S3。

IAM 使用者存取金鑰

您可以 AWS 帳戶 為公司建立一個；不過，組織中可能有多名員工需要存取組織的 AWS 資源。共用您的 AWS 帳戶 存取金鑰可降低安全性，而且 AWS 帳戶 為每個員工建立個人可能不切實際。而且您也無法輕易地分享像是儲存貯體與物件等資源，因為它們皆屬不同帳戶所擁有。若要分享資源，您必須授予許可，這是額外進行的工作。

在這類情況下，您可以使用 AWS Identity and Access Management (IAM) 以自己的存取金鑰在 AWS 帳戶 下建立使用者，並連接將適當資源存取許可授予這些使用者的 IAM 使用者政策。為加強管理這些使用者，您可利用 IAM 建立使用者群組，並授予會套用到該群組內所有使用者的群組層級許可。

這些使用者稱為您在其中建立和管理的 IAM 使用者 AWS。父帳戶能控制使用者存取 的能力 AWS IAM 使用者建立的任何資源，都由父 AWS 帳戶 控制，且由其付費。這些 IAM 使用者會使用自己的安全登入資料，將經過身分驗證的請求傳送至 Amazon S3。如需在 下建立和管理使用者的詳細資訊 AWS 帳戶，請前往[AWS Identity and Access Management 產品詳細資訊頁面](#)。

暫時安全憑證

除了使用自己的存取金鑰建立 IAM 使用者之外，IAM 也可讓您將臨時安全登入資料（臨時存取金鑰和安全字符）授予任何 IAM 使用者，讓他們可以存取您的 AWS 服務和資源。您也可以 AWS 之外，於自己的系統中管理使用者。這些稱為聯合身分使用者。此外，使用者可以是您建立以存取 AWS 資源的應用程式。

IAM 提供 AWS Security Token Service API 供您請求臨時安全登入資料。您可以使用 AWS STS API 或 AWS SDK 來請求這些登入資料。API 會傳回暫時性安全登入資料 (存取金鑰 ID 與私密存取金鑰) 及安全權杖。這些登入資料只有在您申請時所指定的期間內才有效。使用存取金鑰 ID 和私密金鑰的方式，與您利用 AWS 帳戶 或 IAM 使用者存取金鑰來傳送請求時的使用方式一樣。此外，傳送至 Amazon S3 的每個請求中都必要有權杖。

IAM 使用者可以申請這些暫時性安全登入資料供其本身使用，或將它們提供給聯合身分使用者或應用程式。為聯合身分使用者申請暫時性安全登入資料時，您必須提供使用者名稱且定義您希望與這些暫時

性安全登入資料相關聯性之許可的 IAM 原則。聯合身分使用者取得的許可數目，不得超過已申請暫時性登入資料的父 IAM 使用者。

您可使用這些暫時性安全登入資料，對 Amazon S3 提出請求。API 程式庫會使用這些登入資料來運算出必要的簽章值，以驗證您的要求。若使用過期的登入資料傳送請求，Amazon S3 會拒絕該請求。

如需使用 REST API 要求中暫時性安全登入資料來簽署要求的資訊，請參閱「[簽署和驗證 REST 請求 \(AWS 簽章版本 2\)](#)」。如需使用 AWS SDKs 傳送請求的資訊，請參閱 [使用 AWS SDKs 提出請求](#)。

如需 IAM 的臨時安全登入資料支援的詳細資訊，請參閱《IAM 使用者指南》中的[臨時安全登入資料](#)。

對於新增的安全來說，您可以在存取 Amazon S3 資源時配置儲存貯體原則，請求進行多重驗證 (MFA)。如需詳細資訊，請參閱[範例儲存貯體政策：需要 MFA](#)。在您需要 MFA 才可存取 Amazon S3 資源後，可以存取這些資源唯一的方法是提供以 MFA 金鑰建立的暫時性登入資料。如需詳細資訊，請參閱《IAM 使用者指南》中的 [AWS 多重要素驗證](#) 詳細資訊頁面與 [設定受 MFA 保護的 API 存取權限](#)。

要求端點

您要將 REST 要求傳送到服務的預先定義端點。如需所有 AWS 服務及其對應端點的清單，請前往 [中的區域和端點](#) AWS 一般參考。

透過 IPv6 向 Amazon S3 提出請求

Amazon Simple Storage Service (Amazon S3) 除了 IPv4 通訊協定之外，還支援使用網際網路通訊協定第 6 版 (IPv6) 存取 S3 儲存貯體的能力。Amazon S3 雙堆疊端點支援透過 IPv6 與 IPv4 的 S3 儲存貯體要求。透過 IPv6 存取 Amazon S3 不另行收費。如需定價的詳細資訊，請參閱 [Amazon S3 定價](#)。

Note

下列資訊適用於公有 Amazon S3 IPv6 端點。如需有關 AWS Virtual Private Cloud (VPC) 端點的資訊，請參閱適用於 [AWS Amazon S3 的 PrivateLink](#)。

主題

- [透過 IPv6 提出請求的入門](#)
- [在 IAM 原則中使用 IPv6 地址](#)
- [測試 IP 地址相容性](#)

- [使用 Amazon S3 雙堆疊端點](#)

透過 IPv6 提出請求的入門

若要透過 IPv6 向 S3 儲存貯體提出請求，您需要使用雙堆疊端點。下節說明如何使用雙堆疊端點透過 IPv6 提出請求。

以下是透過 IPv6 嘗試存取儲存貯體前的須知事項：

- 存取儲存貯體的用戶端與網路必須啟用才能使用 IPv6。
- 虛擬託管式與路徑式請求都支援 IPv6 存取。如需詳細資訊，請參閱[Amazon S3 雙堆疊端點](#)。
- 如果您在 AWS Identity and Access Management (IAM) 使用者或儲存貯體政策中使用來源 IP 地址篩選，則需要更新政策以包含 IPv6 地址範圍。如需詳細資訊，請參閱[在 IAM 原則中使用 IPv6 地址](#)。
- 使用 IPv6 時，伺服器會存取 IPv6 格式的日誌檔案輸出 IP 地址。您需要更新用來剖析 Amazon S3 日誌檔案的現有工具、指令碼與軟體，以便它們可以剖析 IPv6 格式的 Remote IP 地址。如需詳細資訊，請參閱[使用伺服器存取記錄記錄請求](#)。

Note

如果碰到與日誌檔案中 IPv6 地址相關的問題，請聯絡 [AWS 支援](#)。

使用雙重堆疊端點透過 IPv6 提出請求

您使用雙重堆疊端點透過 IPv6 提出具癒 Amazon S3 API 呼叫的請求。無論您是透過 IPv6 還是透過 IPv4 存取 Amazon S3，Amazon S3 API 操作都會以相同的方式運作。效能也應該是相同。

使用 REST API 時，您會直接存取雙堆疊端點。如需詳細資訊，請參閱[雙堆疊端點](#)。

使用 AWS Command Line Interface (AWS CLI) 和 AWS SDKs 時，您可以使用參數或旗標來變更為雙堆疊端點。您也可以直接指定雙堆疊端點來覆寫設定檔中的 Amazon S3 端點。

您可以使用雙堆疊端點，從下列任一位置透過 IPv6 存取儲存貯體：

- AWS CLI，請參閱 [從使用雙堆疊端點 AWS CLI](#)。
- AWS SDKs 請參閱 [從 AWS SDKs 使用雙堆疊端點](#)。

- REST API，請參閱「[使用 REST API 提出雙重堆疊端點要求](#)」。

無法透過 IPv6 提供的功能

目前，透過 IPv6 存取 S3 儲存貯體時，不支援下列功能：從 S3 儲存貯體託管靜態網站。

在 IAM 原則中使用 IPv6 地址

嘗試使用 IPv6 存取儲存貯體之前，您必須確保用於 IP 地址篩選條件的所有 IAM 使用者或 S3 儲存貯體原則，皆已更新包含 IPv6 地址範圍。未針對處理 IPv6 地址更新的 IP 地址篩選條件原則，可能會導致用戶端在開始使用 IPv6 時，錯誤遺失或取得儲存貯體的存取許可。如需使用 IAM 管理存取權的詳細資訊，請參閱 [Amazon S3 的 Identity and Access Management](#)。

篩選 IP 地址的 IAM 原則使用 [IP 地址條件運算子](#)。以下儲存貯體原則會使用 IP 地址條件運算子找出允許之 IPv4 地址的 54.240.143.* 範圍。拒絕此範圍外的任何 IP 地址存取儲存貯體 (examplebucket)。因為所有的 IPv6 地址都在允許的範圍外，所以此原則可避免 IPv6 地址得以存取 examplebucket。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::examplebucket/*",
      "Condition": {
        "IpAddress": {"aws:SourceIp": "54.240.143.0/24"}
      }
    }
  ]
}
```

您可以修改儲存貯體政策的 Condition 元素，允許 IPv4 (54.240.143.0/24) 與 IPv6 (2001:DB8:1234:5678::/64) 地址範圍，如下例所示。您可使用本例所示的同類型 Condition 區塊，更新您的 IAM 使用者與儲存貯體原則。

```
"Condition": {
  "IpAddress": {
    "aws:SourceIp": [
      "54.240.143.0/24",
      "2001:DB8:1234:5678::/64"
    ]
  }
}
```

在使用 IPv6 之前，您必須更新使用 IP 地址篩選的所有相關 IAM 使用者和儲存貯體政策。避免在儲存貯體政策中使用 IP 地址篩選。

您可以使用 <https://console.aws.amazon.com/iam/> 的 IAM 主控台來檢閱 IAM 使用者原則。如需 IAM 的詳細資訊，請參閱《[IAM 使用者指南](#)》。如需有關編輯 S3 儲存貯體政策的資訊，請參閱[新增儲存貯體政策](#)。

測試 IP 地址相容性

如果使用 Linux/Unix 或 Mac OS X，您可以使用 `curl` 命令測試能否透過 IPv6 存取雙堆疊端點，如下例所示：

Example

```
curl -v http://s3.dualstack.us-west-2.amazonaws.com/
```

您會收到類似下例的資訊。如果您是透過 IPv6 連線，則連線的 IP 地址就會是 IPv6 地址。

```
* About to connect() to s3-us-west-2.amazonaws.com port 80 (#0)
* Trying IPv6 address... connected
* Connected to s3.dualstack.us-west-2.amazonaws.com (IPv6 address) port 80 (#0)
> GET / HTTP/1.1
> User-Agent: curl/7.18.1 (x86_64-unknown-linux-gnu) libcurl/7.18.1 OpenSSL/1.0.1t
  zlib/1.2.3
> Host: s3.dualstack.us-west-2.amazonaws.com
```

如果您使用的是 Microsoft 7 或 10，您可以使用 `ping` 命令來測試是否可以透過 IPv6 或 IPv4 存取雙堆疊端點，如下列範例所示。

```
ping ipv6.s3.dualstack.us-west-2.amazonaws.com
```

使用 Amazon S3 雙堆疊端點

Amazon S3 雙堆疊端點支援透過 IPv6 與 IPv4 的 S3 儲存貯體要求。本節說明如何使用雙堆疊端點。

主題

- [Amazon S3 雙堆疊端點](#)
- [從 使用雙堆疊端點 AWS CLI](#)
- [從 AWS SDKs 使用雙堆疊端點](#)
- [從 REST API 使用雙堆疊端點](#)

Amazon S3 雙堆疊端點

當您請求雙堆疊端點時，儲存貯體 URL 會解析為 IPv6 或 IPv4 地址。如需如何存取 IPv6 儲存貯體的詳細資訊，請參閱 [透過 IPv6 向 Amazon S3 提出請求](#)。

使用 REST API 等同於使用端點名稱 (URI) 直接存取 Amazon S3 端點。您可以使用虛擬代管型或路徑型的端點名稱，透過雙堆疊端點存取 S3 儲存貯體。Amazon S3 僅支援區域雙堆疊端點名稱，亦即，指定的名稱必須包含區域。

請使用下列命名慣例，為雙堆疊虛擬代管型及路徑型的端點名稱命名：

- 虛擬代管型雙堆疊端點：

bucketname.s3.dualstack.*aws-region*.amazonaws.com

- 路徑型雙堆疊端點：

s3.dualstack.*aws-region*.amazonaws.com/*bucketname*

如需端點名稱樣式的詳細資訊，請參閱[存取和列出 Amazon S3 儲存貯體](#)。如需 Amazon S3 端點的清單，請參閱《AWS 一般參考》中的[區域與端點](#)。

Important

您可以搭配使用 Transfer Acceleration 與雙堆疊端點。如需詳細資訊，請參閱[開始使用 Amazon S3 Transfer Acceleration](#)。

 Note

存取 Amazon S3 的兩種虛擬私有雲端 (VPC) 端點類型 (介面 VPC 端點和閘道 VPC 端點) 現在具有雙堆疊支援。如需 Amazon S3 VPC 端點的詳細資訊，請參閱適用於 [AWS Amazon S3 的 PrivateLink](#)。

使用 AWS Command Line Interface (AWS CLI) 和 AWS SDKs 時，您可以使用參數或旗標來變更為雙堆疊端點。您也可以直接指定雙堆疊端點來覆寫設定檔中的 Amazon S3 端點。下列各節說明如何從 AWS CLI 和 AWS SDKs 使用雙堆疊端點。

從 使用雙堆疊端點 AWS CLI

本節提供用於向雙堆疊端點提出請求的 AWS CLI 命令範例。如需設定的指示 AWS CLI，請參閱 [使用 Amazon S3 進行開發 AWS CLI](#)。

您可以在 AWS Config 檔案的設定檔 `use_dualstack_endpoint` 中將組態值設定為 `true`，將 `s3` 和 `s3api` AWS CLI 命令提出的所有 Amazon S3 請求導向至指定區域的雙堆疊端點。您可以在設定檔或命令中使用 `--region` 選項指定區域。

搭配 使用雙堆疊端點時 AWS CLI，同時支援 `path` 和 `virtual` 定址樣式。設定檔中設定的定址樣式控制儲存貯體名稱應包含主機名稱中，或是包含在 URL 中。根據預設，CLI 會盡可能地嘗試使用虛擬樣式，但會在必要時切換回路徑樣式。如需詳細資訊，請參閱 [AWS CLI Amazon S3 組態](#)。

您也可以使用命令變更組態，將預設設定檔中的 `use_dualstack_endpoint` 設為 `true`，以及將 `addressing_style` 設為 `virtual`，如下所示。

```
$ aws configure set default.s3.use_dualstack_endpoint true
$ aws configure set default.s3.addressing_style virtual
```

如果只想將雙堆疊端點用於指定的 AWS CLI 命令（不是所有命令），您可以使用下列其中一種方法：

- 您可以將任何 `--endpoint-url` 或 `https://s3.dualstack.aws-region.amazonaws.com` 命令的 `http://s3.dualstack.aws-region.amazonaws.com` 參數設成 `s3` 或 `s3api`，以在個別的命令中使用雙堆疊端點。

```
$ aws s3api list-objects --bucket bucketname --endpoint-url https://s3.dualstack.aws-region.amazonaws.com
```

- 您可以在 AWS Config 檔案中設定個別的設定檔。例如，您可以建立一個設定檔，將 `use_dualstack_endpoint` 設為 `true`，再建立另一個設定檔不設定 `use_dualstack_endpoint`。當您執行命令時，必須依據是否要使用雙堆疊端點來指定所要使用的設定檔。

Note

使用時 AWS CLI，您目前無法搭配雙堆疊端點使用傳輸加速。不過，的支援 AWS CLI 即將推出。如需詳細資訊，請參閱[啟用和使用 S3 Transfer Acceleration](#)。

從 AWS SDKs 使用雙堆疊端點

本節提供如何使用 AWS SDKs 存取雙堆疊端點的範例。

適用於 Java 的 AWS SDK 雙堆疊端點範例

下列範例示範如何在使用適用於 Java 的 AWS SDK 建立的 Amazon S3 用戶端狀態下，啟用雙堆疊端點。

如需建立和測試工作 Java 範例的說明，請參閱《適用於 Java 的 AWS SDK 開發人員指南》中的[入門](#)。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;

public class DualStackEndpoints {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            // Create an Amazon S3 client with dual-stack endpoints enabled.
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                .withCredentials(new ProfileCredentialsProvider())
```

```
        .withRegion(clientRegion)
        .withDualstackEnabled(true)
        .build();

    s3Client.listObjects(bucketName);
} catch (AmazonServiceException e) {
    // The call was transmitted successfully, but Amazon S3 couldn't process
    // it, so it returned an error response.
    e.printStackTrace();
} catch (SdkClientException e) {
    // Amazon S3 couldn't be contacted for a response, or the client
    // couldn't parse the response from Amazon S3.
    e.printStackTrace();
}
}
```

如果您在 Windows 適用於 Java 的 AWS SDK 上使用，您可能需要設定下列 Java 虛擬機器 (JVM) 屬性：

```
java.net.preferIPv6Addresses=true
```

AWS .NET SDK 雙堆疊端點範例

使用適用於 .NET 的 AWS SDK 時，您可以使用 `AmazonS3Config` 類別來啟用雙堆疊端點的使用，如下列範例所示。

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DualStackEndpointTest
    {
        private const string bucketName = "**** bucket name ****";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;
```

```
public static void Main()
{
    var config = new AmazonS3Config
    {
        UseDualstackEndpoint = true,
        RegionEndpoint = bucketRegion
    };
    client = new AmazonS3Client(config);
    Console.WriteLine("Listing objects stored in a bucket");
    ListingObjectsAsync().Wait();
}

private static async Task ListingObjectsAsync()
{
    try
    {
        var request = new ListObjectsV2Request
        {
            BucketName = bucketName,
            MaxKeys = 10
        };
        ListObjectsV2Response response;
        do
        {
            response = await client.ListObjectsV2Async(request);

            // Process the response.
            foreach (S3Object entry in response.S3Objects)
            {
                Console.WriteLine("key = {0} size = {1}",
                    entry.Key, entry.Size);
            }
            Console.WriteLine("Next Continuation Token: {0}",
response.NextContinuationToken);
            request.ContinuationToken = response.NextContinuationToken;
        } while (response.IsTruncated == true);
    }
    catch (AmazonS3Exception amazonS3Exception)
    {
        Console.WriteLine("An AmazonS3Exception was thrown. Exception: " +
amazonS3Exception.ToString());
    }
    catch (Exception e)
    {
    }
```



```
        Console.WriteLine("Exception: " + e.ToString());
    }
}
}
```

如需有關設定和執行程式碼範例的資訊，請參閱《[適用於 .NET 的 AWS SDK 開發人員指南](#)》中的[適用於 .NET 的 SDK 入門](#)。AWS

從 REST API 使用雙堆疊端點

如需如何使用 REST API 要雙堆疊端點的資訊，請參閱「[使用 REST API 提出雙重堆疊端點要求](#)」。

使用 AWS SDKs 提出請求

主題

- [使用 AWS 帳戶 或 IAM 使用者登入資料提出請求](#)
- [使用 IAM 使用者暫時性登入資料提出請求](#)
- [使用聯合身分使用者暫時登入資料提出要求](#)

您可以使用 AWS SDK 或直接在應用程式中進行 REST API 呼叫，將已驗證的請求傳送至 Amazon S3。AWS SDK API 會使用您提供的登入資料來計算簽章以進行身分驗證。若直接在應用程式中使用 REST API，您必須撰寫必要的程式碼，來運算簽章以驗證您的要求。如需可用 AWS SDKs 的清單，請前往[範例程式碼和程式庫](#)。

使用 AWS 帳戶 或 IAM 使用者登入資料提出請求

您可以使用 AWS 帳戶 或 IAM 使用者安全登入資料，將已驗證的請求傳送至 Amazon S3。本節提供如何使用 適用於 Java 的 AWS SDK 適用於 .NET 的 AWS SDK、和 傳送已驗證請求的範例 適用於 PHP 的 AWS SDK。如需 AWS SDKs 的清單，請前往[範例程式碼和程式庫](#)。

每個這些 AWS SDKs 都使用開發套件特定的登入資料提供者鏈結來尋找和使用登入資料，並代表登入資料擁有者執行動作。所有這些登入資料提供者鏈結都有的共同之處是，它們都會尋找您的本機 AWS 登入資料檔案。

如需詳細資訊，請參閱下列主題。

主題

- [建立本機 AWS 登入資料檔案](#)
- [使用 AWS SDKs 傳送已驗證的請求](#)
- [相關資源](#)

建立本機 AWS 登入資料檔案

為 AWS SDKs 設定登入資料最簡單的方法是使用 AWS 登入資料檔案。如果您使用 AWS Command Line Interface (AWS CLI)，您可能已設定本機 AWS 登入資料檔案。若非如此，請使用下列程序來設定登入資料檔案：

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/iam/> 的 IAM 主控台。

2. 建立新的使用者，使其許可受限在希望程式碼可以存取的服務與動作。如需建立新使用者的詳細資訊，請參閱[建立 IAM 使用者 \(主控台\)](#)，並遵循直到步驟 8 的指示執行作業。
3. 選擇 Download .csv (下載 .csv) 並在本機儲存 AWS 登入資料複本。
4. 在您的電腦上，瀏覽至主目錄並建立 .aws 目錄。在 Unix 系統上 (例如 Linux 或 OS X)，其會是下列位置：

```
~/ .aws
```

在 Windows 上，其會是下列位置：

```
%HOMEPATH%\ .aws
```

5. 在 .aws 目錄中，建立稱為 credentials 的新檔案。
6. 開啟您從 IAM 主控台下載的憑證 .csv 檔案，並使用下列格式將其內容複製到 credentials 檔案：

```
[default]
aws_access_key_id = your_access_key_id
aws_secret_access_key = your_secret_access_key
```

7. 儲存 credentials 檔案，然後刪除在步驟 3 中所下載的 .csv 檔案。

您共用的登入資料檔案現在已在本機電腦上設定，並且可以與 AWS SDKs 搭配使用。

使用 AWS SDKs 傳送已驗證的請求

使用 AWS SDKs 傳送已驗證的請求。如需傳送已驗證請求的詳細資訊，請參閱 [AWS 安全憑證](#) 或 [IAM Identity Center 驗證](#)。

Java

如需有關使用適用於 Java 的 AWS SDK 驗證請求的資訊，請參閱 [《AWS SDKs 和工具參考指南》中的使用共用組態和登入資料檔案來全域設定 SDK 和工具](#) 以及 [使用 AWS SDKs 和工具進行身分驗證和存取](#)。AWS SDKs

.NET

若要使用 AWS 帳戶 或 IAM 使用者登入資料傳送已驗證的請求：

- 建立 AmazonS3Client 類別的執行個體。

- 執行其中一個 `AmazonS3Client` 方法，將請求傳送至 Amazon S3。用戶端會從您所提供的登入資料產生必要的簽章，並在傳送給 Amazon S3 的請求中包含此簽章。

如需詳細資訊，請參閱[使用 AWS 帳戶或 IAM 使用者登入資料提出請求](#)。

Note

- 您可以建立 `AmazonS3Client` 用戶端，而無須提供您的安全登入資料。使用此用戶端傳送的要求，是沒有簽章的匿名要求。若針對非公有存取的資源傳送匿名要求，Amazon S3 會傳回錯誤。
- 您可以建立 AWS 帳戶 並建立所需的使用者。您也可以管理這些使用者的登入資料。您需要這些登入資料才能執行下列範例中的任務。如需詳細資訊，請參閱《適用於 .NET 的 SDK 開發人員指南》中的[設定 AWS 憑證](#)。

然後，您也可以設定應用程式主動擷取設定檔和登入資料，然後在建立 AWS 服務用戶端時明確使用這些登入資料。如需詳細資訊，請參閱《適用於 .NET 的 SDK 開發人員指南》中的[在應用程式中存取登入資料和設定檔](#)。

下列 C# 範例會說明如何執行前述作業。如需有關設定和執行程式碼範例的資訊，請參閱《[適用於 .NET 的 AWS SDK 開發人員指南](#)》中的[適用於 .NET 的 SDK 入門](#)。AWS

Example

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class MakeS3RequestTest
    {
        private const string bucketName = "**** bucket name ****";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
        RegionEndpoint.USWest2;
        private static IAmazonS3 client;
```

```
public static void Main()
{
    using (client = new AmazonS3Client(bucketRegion))
    {
        Console.WriteLine("Listing objects stored in a bucket");
        ListingObjectsAsync().Wait();
    }
}

static async Task ListingObjectsAsync()
{
    try
    {
        ListObjectsRequest request = new ListObjectsRequest
        {
            BucketName = bucketName,
            MaxKeys = 2
        };
        do
        {
            ListObjectsResponse response = await
client.ListObjectsAsync(request);
            // Process the response.
            foreach (S3Object entry in response.S3Objects)
            {
                Console.WriteLine("key = {0} size = {1}",
                    entry.Key, entry.Size);
            }

            // If the response is truncated, set the marker to get the next
            // set of keys.
            if (response.IsTruncated)
            {
                request.Marker = response.NextMarker;
            }
            else
            {
                request = null;
            }
        } while (request != null);
    }
    catch (AmazonS3Exception e)
    {
    }
```

```
        Console.WriteLine("Error encountered on server. Message:'{0}' when  
writing an object", e.Message);  
    }  
    catch (Exception e)  
    {  
        Console.WriteLine("Unknown encountered on server. Message:'{0}' when  
writing an object", e.Message);  
    }  
}  
}
```

PHP

本節說明如何使用 第 3 版的類別 適用於 PHP 的 AWS SDK，使用您的 AWS 帳戶 或 IAM 使用者 登入資料傳送已驗證的請求。如需適用於 Ruby 的 AWS SDK API 的詳細資訊，請前往[AWS 適用於 Ruby 的 SDK - 第 2 版](#)。

下列 PHP 範例說明用戶端如何使用您的安全登入資料提出要求，以列出您帳戶的所有儲存貯體。

Example

```
require 'vendor/autoload.php';  
  
use Aws\S3\Exception\S3Exception;  
use Aws\S3\S3Client;  
  
$bucket = '*** Your Bucket Name ***';  
  
$s3 = new S3Client([  
    'region' => 'us-east-1',  
    'version' => 'latest',  
]);  
  
// Retrieve the list of buckets.  
$result = $s3->listBuckets();  
  
try {  
    // Retrieve a paginator for listing objects.  
    $objects = $s3->getPaginator('ListObjects', [  
        'Bucket' => $bucket  
    ]);
```

```
echo "Keys retrieved!" . PHP_EOL;

// Print the list of objects to the page.
foreach ($objects as $object) {
    echo $object['Key'] . PHP_EOL;
}
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Note

您可以建立 S3Client 用戶端，而無須提供您的安全登入資料。使用此用戶端傳送的要求，是沒有簽章的匿名要求。若針對非公有存取的資源傳送匿名要求，Amazon S3 會傳回錯誤。如需詳細資訊，請參閱 [適用於 PHP 的 AWS SDK 文件](#) 中的 [建立匿名用戶端](#)。

Ruby

您必須先設定 SDK AWS 用來驗證儲存貯體和物件存取權的存取憑證，才能使用第 3 版來適用於 Ruby 的 AWS SDK 呼叫 Amazon S3。如果您已在本機系統的登入資料設定檔中設定共用 AWS 登入資料，則適用於 Ruby 的 SDK 第 3 版可以使用這些登入資料，而不必在程式碼中宣告這些登入資料。如需設定共用登入資料的詳細資訊，請參閱 [使用 AWS 帳戶或 IAM 使用者登入資料提出請求](#)。

下列 Ruby 程式碼片段使用本機電腦上共用登入資料檔案中的 AWS 登入資料來驗證請求，以取得特定儲存貯體中的所有物件金鑰名稱。它會執行以下動作：

1. 建立 `Aws::S3::Client` 類別的執行個體。
2. 使用 `list_objects_v2` 的 `Aws::S3::Client` 方法列舉儲存貯體中的物件，對 Amazon S3 提出要求。用戶端會從您電腦上登入資料檔案中的 AWS 登入資料產生必要的簽章值，並將其包含在傳送至 Amazon S3 的請求中。
3. 將物件金鑰名稱陣列，列印至終端機。

Example

```
# Prerequisites:
```

```
# - An existing Amazon S3 bucket.

require 'aws-sdk-s3'

# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if all operations succeed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_bucket_objects?(s3_client, 'amzn-s3-demo-bucket')
def list_bucket_objects?(s3_client, bucket_name)
  puts "Accessing the bucket named '#{bucket_name}'..."
  objects = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if objects.count.positive?
    puts 'The object keys in this bucket are (first 50 objects):'
    objects.contents.each do |object|
      puts object.key
    end
  else
    puts 'No objects found in this bucket.'
  end

  true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
  false
end

# Example usage:
def run_me
  region = 'us-west-2'
  bucket_name = 'BUCKET_NAME'
  s3_client = Aws::S3::Client.new(region: region)

  exit 1 unless list_bucket_objects?(s3_client, bucket_name)
end

run_me if $PROGRAM_NAME == __FILE__
```


如果您沒有本機 AWS 登入資料檔案，您仍然可以建立 `Aws::S3::Client` 資源，並針對 Amazon S3 儲存貯體和物件執行程式碼。使用適用於 Ruby 的開發套件第 3 版傳送請求，預設是沒有簽章的匿名請求。若針對非公有存取的資源傳送匿名請求，Amazon S3 會傳回錯誤。

您可以針對適用於 Ruby 的開發套件應用程式使用並展開上述程式碼片段，如下列更強大的範例所示。

```
# Prerequisites:
# - An existing Amazon S3 bucket.

require 'aws-sdk-s3'

# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if all operations succeed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_bucket_objects?(s3_client, 'amzn-s3-demo-bucket')
def list_bucket_objects?(s3_client, bucket_name)
  puts "Accessing the bucket named '#{bucket_name}'..."
  objects = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if objects.count.positive?
    puts 'The object keys in this bucket are (first 50 objects):'
    objects.contents.each do |object|
      puts object.key
    end
  else
    puts 'No objects found in this bucket.'
  end

  true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
  false
end

# Example usage:
def run_me
```

```
region = 'us-west-2'
bucket_name = 'BUCKET_NAME'
s3_client = Aws::S3::Client.new(region: region)

exit 1 unless list_bucket_objects?(s3_client, bucket_name)
end

run_me if $PROGRAM_NAME == __FILE__
```

Go

Example

下列範例使用 SDK for Go 從共用 AWS 登入資料檔案自動載入的登入資料。

```
package main

import (
    "context"
    "errors"
    "fmt"

    "github.com/aws/aws-sdk-go-v2/config"
    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/smithy-go"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Storage Service
// (Amazon S3) client and list up to 10 buckets in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
    ctx := context.Background()
    sdkConfig, err := config.LoadDefaultConfig(ctx)
    if err != nil {
        fmt.Println("Couldn't load default configuration. Have you set up your AWS
account?")
        fmt.Println(err)
        return
    }
    s3Client := s3.NewFromConfig(sdkConfig)
    count := 10
```

```
fmt.Printf("Let's list up to %v buckets for your account.\n", count)
result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
if err != nil {
    var ae smithy.APIError
    if errors.As(err, &ae) && ae.ErrorCode() == "AccessDenied" {
        fmt.Println("You don't have permission to list buckets for this account.")
    } else {
        fmt.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
    }
    return
}
if len(result.Buckets) == 0 {
    fmt.Println("You don't have any buckets!")
} else {
    if count > len(result.Buckets) {
        count = len(result.Buckets)
    }
    for _, bucket := range result.Buckets[:count] {
        fmt.Printf("\t%v\n", *bucket.Name)
    }
}
}
```

相關資源

- [使用 AWS SDKs 使用 Amazon S3 進行開發](#)
- [適用於 PHP 的 AWS SDK 適用於 Amazon S3 Aws\S3\S3Client 類別](#)
- [適用於 PHP 的 AWS SDK 文件](#)

使用 IAM 使用者暫時性登入資料提出請求

AWS 帳戶 或 IAM 使用者可以請求臨時安全登入資料，並使用它們將已驗證的請求傳送至 Amazon S3。本節提供的範例是如何使用 適用於 Java 的 AWS SDK、.NET 與 PHP 來取得暫時性安全登入資料，然後使用這些登入資料來為對 Amazon S3 的請求進行驗證。

Java

IAM 使用者或 AWS 帳戶 可以使用 請求臨時安全登入資料 (請參閱 [提出要求](#)) 適用於 Java 的 AWS SDK，並使用它們來存取 Amazon S3。這些登入資料會在指定的工作階段使用期限之後到期。

工作階段使用期限預設是一小時。如果您使用 IAM 使用者登入資料，可以在請求暫時性安全登入資料時，為角色指定持續時間，從 15 分鐘到最大的工作階段持續時間。如需暫時安全登入資料的詳細資訊，請參閱《IAM 使用者指南》中的 [暫時安全登入資料](#)。如需提出請求的詳細資訊，請參閱[提出要求](#)。

取得暫時性安全登入資料並存取 Amazon S3

1. 建立 `AWSecurityTokenService` 類別的執行個體。
2. 呼叫安全性字服務 (STS) 用戶端的 `assumeRole()` 方法，擷取所需角色的臨時安全登入資料。
3. 將暫時安全登入資料封裝至 `BasicSessionCredentials` 物件。您可以使用此物件，將暫時性安全登入資料提供給 Amazon S3 用戶端。
4. 使用暫時性安全登入資料建立 `AmazonS3Client` 類別的執行個體。您可以使用此用戶端，將請求傳送給 Amazon S3。如果使用過期的登入資料傳送請求，Amazon S3 將會傳回錯誤。

Note

如果您使用 AWS 帳戶 安全登入資料來取得暫時性安全登入資料，則暫時性登入資料只會作用一個小時。只有在您使用 IAM 使用者登入資料來請求工作階段時，才能指定工作階段使用期限。

下列範例會列出所指定儲存貯體中的一組物件金鑰。此範例會取得工作階段的臨時安全登入資料，然後使用這些登入資料將透過身分驗證的請求傳送給 Amazon S3。

如果您想要使用 IAM 使用者憑證來測試範例，則必須透過 AWS 帳戶建立 IAM 使用者。如需如何建立 IAM 使用者的詳細資訊，請參閱《IAM 使用者指南》中的[建立第一個 IAM 使用者與管理員群組](#)。

如需建立和測試工作範例的說明，請參閱《適用於 Java 的 AWS SDK 開發人員指南》中的[入門](#)。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.AWSSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.securitytoken.AWSSecurityTokenService;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClientBuilder;
import com.amazonaws.services.securitytoken.model.AssumeRoleRequest;
import com.amazonaws.services.securitytoken.model.AssumeRoleResult;
import com.amazonaws.services.securitytoken.model.Credentials;

public class MakingRequestsWithIAMTempCredentials {
    public static void main(String[] args) {
        String clientRegion = "**** Client region ****";
        String roleARN = "**** ARN for role to be assumed ****";
        String roleSessionName = "**** Role session name ****";
        String bucketName = "**** Bucket name ****";

        try {
            // Creating the STS client is part of your trusted code. It has
            // the security credentials you use to obtain temporary security
            credentials.
            AWSSecurityTokenService stsClient =
            AWSSecurityTokenServiceClientBuilder.standard()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(clientRegion)
                .build();

            // Obtain credentials for the IAM role. Note that you cannot assume the
            role of
            // an AWS root account;
            // Amazon S3 will deny access. You must use credentials for an IAM user
            or an
```

```

        // IAM role.
        AssumeRoleRequest roleRequest = new AssumeRoleRequest()
            .withRoleArn(roleARN)
            .withRoleSessionName(roleSessionName);
        AssumeRoleResult roleResponse = stsClient.assumeRole(roleRequest);
        Credentials sessionCredentials = roleResponse.getCredentials();

        // Create a BasicSessionCredentials object that contains the credentials
you
        // just retrieved.
        BasicSessionCredentials awsCredentials = new BasicSessionCredentials(
            sessionCredentials.getAccessKeyId(),
            sessionCredentials.getSecretAccessKey(),
            sessionCredentials.getSessionToken());

        // Provide temporary security credentials so that the Amazon S3 client
        // can send authenticated requests to Amazon S3. You create the client
        // using the sessionCredentials object.
        AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
            .withCredentials(new
AWSStaticCredentialsProvider(awsCredentials))
            .withRegion(clientRegion)
            .build();

        // Verify that assuming the role worked and the permissions are set
correctly
        // by getting a set of object keys from the bucket.
        ObjectListing objects = s3Client.listObjects(bucketName);
        System.out.println("No. of Objects: " +
objects.getObjectSummaries().size());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}

```

.NET

IAM 使用者或 AWS 帳戶 可以使用 請求臨時安全登入資料 適用於 .NET 的 AWS SDK，並使用它們來存取 Amazon S3。這些登入資料會在工作階段使用期限之後到期。

工作階段使用期限預設是一小時。如果您使用 IAM 使用者登入資料，可以在請求暫時性安全登入資料時，為角色指定持續時間，從 15 分鐘到最大的工作階段持續時間。如需暫時安全登入資料的詳細資訊，請參閱《IAM 使用者指南》中的[暫時安全登入資料](#)。如需提出請求的詳細資訊，請參閱[提出要求](#)。

取得暫時性安全登入資料並存取 Amazon S3

1. 建立 AWS Security Token Service 用戶端的執行個體 `AmazonSecurityTokenServiceClient`。
2. 呼叫您在上一個步驟中所建立之 STS 用戶端的 `GetSessionToken` 方法，以啟動工作階段。您可以使用 `GetSessionTokenRequest` 物件，將工作階段資訊提供給此方法。

此方法會傳回暫時性安全登入資料。

3. 將暫時安全登入資料封裝至 `SessionAWSCredentials` 物件執行個體。您可以使用此物件，將暫時性安全登入資料提供給 Amazon S3 用戶端。
4. 傳入暫時性安全登入資料，以建立 `AmazonS3Client` 類別執行個體。您可以使用此用戶端，將請求傳送給 Amazon S3。如果使用過期的登入資料傳送請求，Amazon S3 會傳回錯誤。

Note

如果您使用 AWS 帳戶 安全登入資料來取得暫時性安全登入資料，則該些登入資料只會作用一個小時。您只能在使用 IAM 使用者登入資料請求工作階段時，指定工作階段使用期限。

下列 C# 範例會列出指定儲存貯體中的物件金鑰。為了方便說明，該範例會取得預設一小時工作階段的暫時性安全登入資料，然後使用這些登入資料將透過身分驗證的請求傳送給 Amazon S3。

如果您想要使用 IAM 使用者憑證來測試範例，則必須透過 AWS 帳戶建立 IAM 使用者。如需如何建立 IAM 使用者的詳細資訊，請參閱《IAM 使用者指南》中的[建立第一個 IAM 使用者與管理員群組](#)。如需提出請求的詳細資訊，請參閱[提出要求](#)。

如需有關設定和執行程式碼範例的資訊，請參閱 [《適用於 .NET 的 AWS SDK 開發人員指南》](#) 中的 [適用於 .NET 的 SDK 入門](#)。AWS

```
using Amazon;
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TempCredExplicitSessionStartTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            ListObjectsAsync().Wait();
        }

        private static async Task ListObjectsAsync()
        {
            try
            {
                // Credentials use the default AWS SDK for .NET credential search
chain.
                // On local development machines, this is your default profile.
                Console.WriteLine("Listing objects stored in a bucket");
                SessionAWSCredentials tempCredentials = await
GetTemporaryCredentialsAsync();

                // Create a client by providing temporary security credentials.
                using (s3Client = new AmazonS3Client(tempCredentials, bucketRegion))
                {
                    var listObjectRequest = new ListObjectsRequest
                    {
```



```
        BucketName = bucketName
    };
    // Send request to Amazon S3.
    ListObjectsResponse response = await
s3Client.ListObjectsAsync(listObjectRequest);
    List<S3Object> objects = response.S3Objects;
    Console.WriteLine("Object count = {0}", objects.Count);
    }
}
catch (AmazonS3Exception s3Exception)
{
    Console.WriteLine(s3Exception.Message, s3Exception.InnerException);
}
catch (AmazonSecurityTokenServiceException stsException)
{
    Console.WriteLine(stsException.Message,
stsException.InnerException);
}
}

private static async Task<SessionAWSCredentials>
GetTemporaryCredentialsAsync()
{
    using (var stsClient = new AmazonSecurityTokenServiceClient())
    {
        var getSessionTokenRequest = new GetSessionTokenRequest
        {
            DurationSeconds = 7200 // seconds
        };

        GetSessionTokenResponse sessionTokenResponse =
            await
stsClient.GetSessionTokenAsync(getSessionTokenRequest);

        Credentials credentials = sessionTokenResponse.Credentials;

        var sessionCredentials =
            new SessionAWSCredentials(credentials.AccessKeyId,
                                       credentials.SecretAccessKey,
                                       credentials.SessionToken);

        return sessionCredentials;
    }
}
}
```

```
}
```

PHP

如需適用於 Ruby 的 AWS SDK API 的詳細資訊，請前往[AWS 適用於 Ruby 的 SDK - 第 2 版](#)。

IAM 使用者或 AWS 帳戶 可以使用第 3 版請求臨時安全登入資料 適用於 PHP 的 AWS SDK。然後，可以使用該暫時性登入資料來存取 Amazon S3。該登入資料會在工作階段使用期限到期時到期。

工作階段使用期限預設是一小時。如果您使用 IAM 使用者登入資料，可以在請求暫時性安全登入資料時，為角色指定持續時間，從 15 分鐘到最大的工作階段持續時間。如需暫時安全登入資料的詳細資訊，請參閱《IAM 使用者指南》中的[暫時安全登入資料](#)。如需提出請求的詳細資訊，請參閱[提出要求](#)。

Note

如果您使用 AWS 帳戶 安全登入資料來取得暫時性安全登入資料，則暫時性安全登入資料只會作用一個小時。只有在您使用 IAM 使用者登入資料來請求工作階段時，才能指定工作階段使用期限。

Example

下列 PHP 範例會使用暫時性安全登入資料來列出指定儲存貯體中的物件金鑰。此範例會取得預設一小時工作階段的暫時性安全登入資料，然後使用這些登入資料將透過身分驗證的請求傳送給 Amazon S3。如需適用於 Ruby 的 AWS SDK API 的詳細資訊，請前往[AWS 適用於 Ruby 的 SDK - 第 2 版](#)。

如果您想要使用 IAM 使用者憑證來測試範例，則需要透過 AWS 帳戶建立 IAM 使用者。如需如何建立 IAM 使用者的資訊，請參閱《IAM 使用者指南》中的[建立第一個 IAM 使用者與管理員群組](#)。如需使用 IAM 使用者登入資料要求工作階段時設定工作階段使用期限的範例，請參閱「[使用 IAM 使用者暫時性登入資料提出請求](#)」。

```
require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;

$bucket = '*** Your Bucket Name ***';
```

```

$sts = new StsClient([
    'version' => 'latest',
    'region' => 'us-east-1'
]);

$sessionToken = $sts->getSessionToken();

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
    'credentials' => [
        'key' => $sessionToken['Credentials']['AccessKeyId'],
        'secret' => $sessionToken['Credentials']['SecretAccessKey'],
        'token' => $sessionToken['Credentials']['SessionToken']
    ]
]);

$result = $s3->listBuckets();

try {
    // Retrieve a paginator for listing objects.
    $objects = $s3->getPaginator('ListObjects', [
        'Bucket' => $bucket
    ]);

    echo "Keys retrieved!" . PHP_EOL;

    // List objects
    foreach ($objects as $object) {
        echo $object['Key'] . PHP_EOL;
    }
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}

```

Ruby

IAM 使用者或 AWS 帳戶 可以使用 請求臨時安全登入資料 適用於 Ruby 的 AWS SDK，並使用它們來存取 Amazon S3。這些登入資料會在工作階段使用期限之後到期。

工作階段使用期限預設是一小時。如果您使用 IAM 使用者登入資料，可以在請求暫時性安全登入資料時，為角色指定持續時間，從 15 分鐘到最大的工作階段持續時間。如需暫時安全登入資料的詳

細資訊，請參閱《IAM 使用者指南》中的[暫時安全登入資料](#)。如需提出請求的詳細資訊，請參閱[提出要求](#)。

 Note

如果您使用 AWS 帳戶 安全登入資料來取得暫時性安全登入資料，則暫時性安全登入資料只會作用一個小時。只有在您使用 IAM 使用者登入資料請求工作階段時，才能指定工作階段使用期限。

下列 Ruby 範例會建立暫時使用者有一小時的時間可列出指定儲存貯體中的項目。若要使用此範例，您必須擁有 AWS 具有必要許可的登入資料，才能建立新的 AWS Security Token Service (AWS STS) 用戶端，並列出 Amazon S3 儲存貯體。

```
# Prerequisites:
# - A user in AWS Identity and Access Management (IAM). This user must
#   be able to assume the following IAM role. You must run this code example
#   within the context of this user.
# - An existing role in IAM that allows all of the Amazon S3 actions for all of the
#   resources in this code example. This role must also trust the preceding IAM
#   user.
# - An existing S3 bucket.

require 'aws-sdk-core'
require 'aws-sdk-s3'
require 'aws-sdk-iam'

# Checks whether a user exists in IAM.
#
# @param iam [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [Boolean] true if the user exists; otherwise, false.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   exit 1 unless user_exists?(iam_client, 'my-user')
def user_exists?(iam_client, user_name)
  response = iam_client.get_user(user_name: user_name)
  return true if response.user.user_name
rescue Aws::IAM::Errors::NoSuchEntity
  # User doesn't exist.
rescue StandardError => e
```

```

    puts 'Error while determining whether the user ' \
        "'#{user_name}' exists: #{e.message}"
end

# Creates a user in IAM.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [AWS::IAM::Types::User] The new user.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   user = create_user(iam_client, 'my-user')
#   exit 1 unless user.user_name
def create_user(iam_client, user_name)
  response = iam_client.create_user(user_name: user_name)
  response.user
rescue StandardError => e
  puts "Error while creating the user '#{user_name}': #{e.message}"
end

# Gets a user in IAM.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [AWS::IAM::Types::User] The existing user.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   user = get_user(iam_client, 'my-user')
#   exit 1 unless user.user_name
def get_user(iam_client, user_name)
  response = iam_client.get_user(user_name: user_name)
  response.user
rescue StandardError => e
  puts "Error while getting the user '#{user_name}': #{e.message}"
end

# Checks whether a role exists in IAM.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param role_name [String] The role's name.
# @return [Boolean] true if the role exists; otherwise, false.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   exit 1 unless role_exists?(iam_client, 'my-role')

```

```

def role_exists?(iam_client, role_name)
  response = iam_client.get_role(role_name: role_name)
  return true if response.role.role_name
rescue StandardError => e
  puts 'Error while determining whether the role ' \
    "'#{role_name}' exists: #{e.message}"
end

# Gets credentials for a role in IAM.
#
# @param sts_client [Aws::STS::Client] An initialized AWS STS client.
# @param role_arn [String] The role's Amazon Resource Name (ARN).
# @param role_session_name [String] A name for this role's session.
# @param duration_seconds [Integer] The number of seconds this session is valid.
# @return [AWS::AssumeRoleCredentials] The credentials.
# @example
#   sts_client = Aws::STS::Client.new(region: 'us-west-2')
#   credentials = get_credentials(
#     sts_client,
#     'arn:aws:iam::123456789012:role/AmazonS3ReadOnly',
#     'ReadAmazonS3Bucket',
#     3600
#   )
#   exit 1 if credentials.nil?
def get_credentials(sts_client, role_arn, role_session_name, duration_seconds)
  Aws::AssumeRoleCredentials.new(
    client: sts_client,
    role_arn: role_arn,
    role_session_name: role_session_name,
    duration_seconds: duration_seconds
  )
rescue StandardError => e
  puts "Error while getting credentials: #{e.message}"
end

# Checks whether a bucket exists in Amazon S3.
#
# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The name of the bucket.
# @return [Boolean] true if the bucket exists; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless bucket_exists?(s3_client, 'amzn-s3-demo-bucket')
def bucket_exists?(s3_client, bucket_name)

```

```

    response = s3_client.list_buckets
    response.buckets.each do |bucket|
      return true if bucket.name == bucket_name
    end
  rescue StandardError => e
    puts "Error while checking whether the bucket '#{bucket_name}' " \
      "exists: #{e.message}"
  end

# Lists the keys and ETags for the objects in an Amazon S3 bucket.
#
# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if the objects were listed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_objects_in_bucket?(s3_client, 'amzn-s3-demo-bucket')
def list_objects_in_bucket?(s3_client, bucket_name)
  puts "Accessing the contents of the bucket named '#{bucket_name}'..."
  response = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if response.count.positive?
    puts "Contents of the bucket named '#{bucket_name}' (first 50 objects):"
    puts 'Name => ETag'
    response.contents.each do |obj|
      puts "#{obj.key} => #{obj.etag}"
    end
  else
    puts "No objects in the bucket named '#{bucket_name}'."
  end
  true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
end

```

使用聯合身分使用者暫時登入資料提出要求

您可以請求臨時安全登入資料，並將其提供給需要存取 AWS 資源的聯合身分使用者或應用程式。本節提供如何使用 AWS SDK 來取得聯合身分使用者或應用程式的臨時安全登入資料，以及使用這些登入資料將已驗證的請求傳送至 Amazon S3 的範例。如需 AWS SDKs 的清單，請參閱[範例程式碼和程式庫](#)。

Note

AWS 帳戶 和 IAM 使用者可以為聯合身分使用者請求臨時安全登入資料。不過，為了提高安全，只有具有必要許可的 IAM 使用者才應該請求這些暫時登入資料，確保聯合身分使用者善加利用請求 IAM 使用者的許可。在一些應用程式中，您可能會發現一些適合建立 IAM 的使用者，其具有唯一目的為將暫時安全登入資料授予聯合身分使用者與應用程式的特定許可。

Java

您可以為聯合身分使用者和應用程式提供臨時安全登入資料，以便他們可以傳送已驗證的請求來存取您的 AWS 資源。請求這些暫時登入資料時，您必須提供使用者名稱，以及說明您想要授予之資源許可的 IAM 原則。工作階段使用期限預設是一小時。要求聯合身分使用者與應用程式的暫時安全登入資料時，您可以明確地設定不同的持續時間值。

Note

在請求用於聯合身分使用者與應用程式得暫時安全登入資料時，為了提高安全，我們建議您使用專用 IAM 使用者，只進行有必要的存取許可。您所建立暫時使用者的許可絕不可高於已請求暫時安全登入資料的 IAM 使用者。如需詳細資訊，請參閱 [AWS Identity and Access Management 常見問答集](#)。

為提供安全登入資料和傳送經過驗證的要求以存取資源，請執行以下動作：

- 建立 `AWSSecurityTokenServiceClient` 類別的執行個體。
- 透過呼叫安全字服務 (STS) 用戶端的 `getFederationToken()` 方法，開始工作階段。請您提供工作階段資訊，包括您要連線至暫時性登入資料的使用者名稱與 IAM 原則。您可以提供選用的工作階段持續時間。此方法會傳回暫時安全登入資料。
- 將暫時安全登入資料封裝至 `BasicSessionCredentials` 物件執行個體。您可以使用此物件，將暫時性安全登入資料提供給 Amazon S3 用戶端。

- 使用暫時性安全登入資料建立 AmazonS3Client 類別的執行個體。您可以使用此用戶端，將請求傳送給 Amazon S3。如果使用過期的登入資料傳送請求，Amazon S3 會傳回錯誤。

Example

此範例會列出指定 S3 儲存貯體中的金鑰。在範例中，您會先為聯合身分使用者，取得兩個小時的暫時安全登入資料工作階段，並使用它們將經過身分驗證的請求傳送給 Amazon S3。若要執行此範例，您需要建立具有連接政策的 IAM 使用者，以允許使用者請求臨時安全登入資料並列出您的 AWS 資源。以下政策可以完成這個情況：

```
{
  "Statement": [{
    "Action": ["s3:ListBucket",
      "sts:GetFederationToken"],
    "Effect": "Allow",
    "Resource": "*"
  }]
}
```

如需如何建立 IAM 使用者的詳細資訊，請參閱《IAM 使用者指南》中的[建立第一個 IAM 使用者與管理員群組](#)。

在建立 IAM 使用者和附加先前的原則，您可以執行下列的範例。如需建立和測試工作範例的說明，請參閱《適用於 Java 的 AWS SDK 開發人員指南》中的[入門](#)。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.AWSSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Resource;
import com.amazonaws.auth.policy.Statement;
import com.amazonaws.auth.policy.Statement.Effect;
import com.amazonaws.auth.policy.actions.S3Actions;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
```

```
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.securitytoken.AWSSecurityTokenService;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClientBuilder;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;

import java.io.IOException;

public class MakingRequestsWithFederatedTempCredentials {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "**** Specify bucket name ****";
        String federatedUser = "**** Federated user name ****";
        String resourceARN = "arn:aws:s3:::" + bucketName;

        try {
            AWSSecurityTokenService stsClient = AWSSecurityTokenServiceClientBuilder
                .standard()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(clientRegion)
                .build();

            GetFederationTokenRequest getFederationTokenRequest = new
            GetFederationTokenRequest();
            getFederationTokenRequest.setDurationSeconds(7200);
            getFederationTokenRequest.setName(federatedUser);

            // Define the policy and add it to the request.
            Policy policy = new Policy();
            policy.withStatements(new Statement(Effect.Allow)
                .withActions(S3Actions.ListObjects)
                .withResources(new Resource(resourceARN)));
            getFederationTokenRequest.setPolicy(policy.toJson());

            // Get the temporary security credentials.
            GetFederationTokenResult federationTokenResult =
            stsClient.getFederationToken(getFederationTokenRequest);
            Credentials sessionCredentials = federationTokenResult.getCredentials();

            // Package the session credentials as a BasicSessionCredentials
            // object for an Amazon S3 client object to use.
```

```

        BasicSessionCredentials basicSessionCredentials = new
BasicSessionCredentials(
            sessionCredentials.getAccessKeyId(),
            sessionCredentials.getSecretAccessKey(),
            sessionCredentials.getSessionToken());
    AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
        .withCredentials(new
AWSStaticCredentialsProvider(basicSessionCredentials))
        .withRegion(clientRegion)
        .build();

    // To verify that the client works, send a listObjects request using
    // the temporary security credentials.
    ObjectListing objects = s3Client.listObjects(bucketName);
    System.out.println("No. of Objects = " +
objects.getObjectSummaries().size());
    } catch (AmazonServiceException e) {
        // The call was transmitted successfully, but Amazon S3 couldn't process
        // it, so it returned an error response.
        e.printStackTrace();
    } catch (SdkClientException e) {
        // Amazon S3 couldn't be contacted for a response, or the client
        // couldn't parse the response from Amazon S3.
        e.printStackTrace();
    }
}
}
}

```

.NET

您可以為聯合身分使用者和應用程式提供臨時安全登入資料，以便他們可以傳送已驗證的請求來存取您的 AWS 資源。請求這些暫時登入資料時，您必須提供使用者名稱，以及說明您想要授予之資源許可的 IAM 原則。預設的工作階段使用期限為一小時。要求聯合身分使用者與應用程式的暫時安全登入資料時，您可以明確地設定不同的持續時間值。如需有關傳送已驗證要求的詳細資訊，請參閱[提出要求](#)。

Note

為聯合身分使用者和應用程式請求臨時安全登入資料時，为了提高安全性，請使用僅具有必要存取許可的專用 IAM 使用者。您所建立暫時使用者的許可絕不可高於已請求暫時安全登

入資料的 IAM 使用者。如需詳細資訊，請參閱 [AWS Identity and Access Management 常見問答集](#)。

您可執行下列項目：

- 建立 AWS Security Token Service 用戶端 `AmazonSecurityTokenServiceClient` 類別的執行個體。
- 呼叫 STS 用戶端的 `GetFederationToken` 方法，以啟動工作階段。您需要提供工作階段資訊，包括您要連線至暫時性登入資料的使用者名稱與 IAM 原則。您可以選擇性提供工作階段持續的時間。此方法會傳回暫時安全登入資料。
- 將暫時安全登入資料封裝至 `SessionAWSCredentials` 物件執行個體。您可以使用此物件，將暫時性安全登入資料提供給 Amazon S3 用戶端。
- 請傳遞暫時性安全登入資料，以建立 `AmazonS3Client` 類別的執行個體。您可以使用此用戶端，傳送請求給 Amazon S3。如果使用過期的登入資料傳送請求，Amazon S3 會傳回錯誤。

Example

下列 C# 範例會列出所指定儲存貯體中的金鑰。在範例中，您會先以聯合身分使用者(User1)，取得兩個小時的暫時安全登入資料工作階段，並使用它們將經過身分驗證的請求傳送給 Amazon S3。

- 在此範例中，請您建立 IAM 使用者，並使其擁有最少許可。若使用 IAM 使用者的登入資料，請您為其他請求暫時性登入資料。此範例僅會列出特定儲存貯體中的此物件。請使用下列附加原則來建立 IAM 使用者：

```
{
  "Statement": [{
    "Action": ["s3:ListBucket",
      "sts:GetFederationToken*"],
    "Effect": "Allow",
    "Resource": "*"
  }]
}
```

此政策允許 IAM 使用者請求暫時安全登入資料，以及僅列出 AWS 資源的存取許可。如需如何建立 IAM 使用者的詳細資訊，請參閱《IAM 使用者指南》中的[建立 IAM 使用者與管理員群組](#)。

- 使用 IAM 使用者安全登入資料來測試下列範例。此範例會使用暫時安全登入資料將經過身分驗證的請求傳送給 Amazon S3。此範例會在要求聯合身分使用者 (使用者 1) 的暫時性安全登入資料時指定下列政策，以限制存取指定儲存貯體 (YourBucketName) 中列出的物件。您必須更新政策，並提供自己的現有儲存貯體名稱。

```
{
  "Statement": [
    {
      "Sid": "1",
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::YourBucketName"
    }
  ]
}
```

- Example

更新下列範例，並提供在上一個聯合身分使用者存取原則中所指定的儲存貯體名稱。如需有關設定和執行程式碼範例的資訊，請參閱 [《適用於 .NET 的 AWS SDK 開發人員指南》](#) 中的 [適用於 .NET 的 SDK 入門](#)。AWS

```
using Amazon;
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TempFederatedCredentialsTest
    {
        private const string bucketName = "**** bucket name ****";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
            RegionEndpoint.USWest2;
        private static IAmazonS3 client;
```

```
public static void Main()
{
    ListObjectsAsync().Wait();
}

private static async Task ListObjectsAsync()
{
    try
    {
        Console.WriteLine("Listing objects stored in a bucket");
        // Credentials use the default AWS SDK for .NET credential search
chain.
        // On local development machines, this is your default profile.
        SessionAWSCredentials tempCredentials =
            await GetTemporaryFederatedCredentialsAsync();

        // Create a client by providing temporary security credentials.
        using (client = new AmazonS3Client(bucketRegion))
        {
            ListObjectsRequest listObjectRequest = new
ListObjectsRequest();
            listObjectRequest.BucketName = bucketName;

            ListObjectsResponse response = await
client.ListObjectsAsync(listObjectRequest);
            List<S3Object> objects = response.S3Objects;
            Console.WriteLine("Object count = {0}", objects.Count);

            Console.WriteLine("Press any key to continue...");
            Console.ReadKey();
        }
    }
    catch (AmazonS3Exception e)
    {
        Console.WriteLine("Error encountered ***. Message:'{0}' when
writing an object", e.Message);
    }
    catch (Exception e)
    {
        Console.WriteLine("Unknown encountered on server. Message:'{0}'
when writing an object", e.Message);
    }
}
```

```

        private static async Task<SessionAWSCredentials>
GetTemporaryFederatedCredentialsAsync()
    {
        AmazonSecurityTokenServiceConfig config = new
AmazonSecurityTokenServiceConfig();
        AmazonSecurityTokenServiceClient stsClient =
            new AmazonSecurityTokenServiceClient(
                config);

        GetFederationTokenRequest federationTokenRequest =
            new GetFederationTokenRequest();
        federationTokenRequest.DurationSeconds = 7200;
        federationTokenRequest.Name = "User1";
        federationTokenRequest.Policy = @"{
            ""Statement"":
            [
                {
                    ""Sid"": ""Stmt1311212314284"",
                    ""Action"": [""s3:ListBucket""],
                    ""Effect"": ""Allow"",
                    ""Resource"": ""arn:aws:s3:::" + bucketName + @""
                }
            ]
        }
";

        GetFederationTokenResponse federationTokenResponse =
            await
stsClient.GetFederationTokenAsync(federationTokenRequest);
        Credentials credentials = federationTokenResponse.Credentials;

        SessionAWSCredentials sessionCredentials =
            new SessionAWSCredentials(credentials.AccessKeyId,
                credentials.SecretAccessKey,
                credentials.SessionToken);

        return sessionCredentials;
    }
}

```

PHP

本主題說明如何使用第 3 版的類別適用於 PHP 的 AWS SDK 來請求聯合身分使用者和應用程式的臨時安全登入資料，並使用它們來存取存放在 Amazon S3 中的資源。如需適用於 Ruby 的 AWS SDK API 的詳細資訊，請前往[AWS 適用於 Ruby 的 SDK - 第 2 版](#)。

您可以提供臨時安全登入資料給您的聯合身分使用者和應用程式，讓他們可以傳送已驗證的請求來存取您的 AWS 資源。請求這些暫時登入資料時，您必須提供使用者名稱，以及說明您想要授予之資源許可的 IAM 原則。這些登入資料會在工作階段使用期限到期時到期。工作階段使用期限預設是一小時。在要求用於聯合身分使用者與應用程式的暫時安全登入資料時，您可以明確地為持續時間設定不同的值。如需暫時安全登入資料的詳細資訊，請參閱《IAM 使用者指南》中的[暫時安全登入資料](#)。如需有關提供暫時安全登入資料用於聯合身分使用者與應用程式之詳細資料，請參閱[提出要求](#)。

在請求用於聯合身分使用者與應用程式得暫時安全登入資料時，為了提高安全，我們建議請使用專用 IAM 使用者，只進行有必要的存取許可。您所建立暫時使用者的許可絕不可高於已請求暫時安全登入資料的 IAM 使用者。如需關於聯合身分的資訊，請參閱[AWS Identity and Access Management 常見問答集](#)。

如需適用於 Ruby 的 AWS SDK API 的詳細資訊，請前往[AWS 適用於 Ruby 的 SDK - 第 2 版](#)。

Example

下列 PHP 範例會列出所指定儲存貯體中的金鑰。在範例中，您會為聯合身分使用者 (User1) 的身分，取得 1 小時的暫時安全登入資料工作階段。然後，請您使用暫時安全登入資料傳送驗證請求給 Amazon S3。

為了提高安全，當請求其他人的暫時登入資料時，使用安全登入資料的 IAM 使用者，有權請求暫時性安全登入資料。為確保 IAM 使用者只授予最有應用程式特定許可給聯合身分使用者，您也可以限制此 IAM 使用者的存取許可。此範例僅會列出特定儲存貯體中的物件。請使用下列附加原則來建立 IAM 使用者：

```
{
  "Statement": [{
    "Action": ["s3:ListBucket",
      "sts:GetFederationToken"],
    "Effect": "Allow",
    "Resource": "*"
  }]
}
```



```
}
```

此政策允許 IAM 使用者請求暫時安全登入資料，以及僅列出 AWS 資源的存取許可。如需如何建立 IAM 使用者的詳細資訊，請參閱《IAM 使用者指南》中的[建立第一個 IAM 使用者與管理員群組](#)。

您現在可以使用 IAM 使用者安全登入資料來測試下列範例。此範例會使用暫時安全登入資料將經過身分驗證的請求傳送給 Amazon S3。在要求用於聯合身分使用者 (User1) 的暫時安全登入資料時，此範例指定下列政策，以限制列出特定儲存貯體中物件的存取。使用您儲存貯體名稱更新政策。

```
{
  "Statement": [
    {
      "Sid": "1",
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::YourBucketName"
    }
  ]
}
```

在下列範例中，您必須在指定政策資源時，將 `YourBucketName` 取代為您的儲存貯體名稱。

```
require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;

$bucket = '*** Your Bucket Name ***';

// In real applications, the following code is part of your trusted code. It has
// the security credentials that you use to obtain temporary security credentials.
$sts = new StsClient([
    'version' => 'latest',
    'region' => 'us-east-1'
]);

// Fetch the federated credentials.
$sessionToken = $sts->getFederationToken([
    'Name' => 'User1',
    'DurationSeconds' => '3600',
    'Policy' => json_encode([
```

```

        'Statement' => [
            'Sid'           => 'randomstatementid' . time(),
            'Action'         => ['s3:ListBucket'],
            'Effect'         => 'Allow',
            'Resource'       => 'arn:aws:s3:::' . $bucket
        ]
    ])
});

// The following will be part of your less trusted code. You provide temporary
// security credentials so the code can send authenticated requests to Amazon S3.

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
    'credentials' => [
        'key' => $sessionToken['Credentials']['AccessKeyId'],
        'secret' => $sessionToken['Credentials']['SecretAccessKey'],
        'token' => $sessionToken['Credentials']['SessionToken']
    ]
]);

try {
    $result = $s3->listObjects([
        'Bucket' => $bucket
    ]);
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}

```

Ruby

您可以為聯合身分使用者和應用程式提供臨時安全登入資料，以便他們可以傳送已驗證的請求來存取您的 AWS 資源。向 IAM 服務請求這些暫時登入資料時，您必須提供使用者名稱，以及說明您想要授予之資源許可的 IAM 原則。工作階段使用期限預設是一小時。不過，如果您要使用 IAM 使用者登入資料來請求暫時登入資料，則可以在請求聯合身分使用者與應用程式的暫時安全登入資料時明確地設定不同的持續時間值。如需更多有關聯合身分使用者的暫時安全登入資料與應用程式，請參閱 [提出要求](#)。

 Note

為了提高安全，請求聯合身分使用者與應用程式的暫時安全登入資料，您可能會想要使用只具有必要存取許可的專用 IAM 使用者。您所建立暫時使用者的許可絕不可高於已請求暫時安全登入資料的 IAM 使用者。如需詳細資訊，請參閱 [AWS Identity and Access Management 常見問答集](#)。

Example

下列 Ruby 程式碼範例允許聯合身分使用者具有有限許可，以設定在指定儲存貯體中的金鑰清單。

```
# Prerequisites:
# - An existing Amazon S3 bucket.

require 'aws-sdk-s3'
require 'aws-sdk-iam'
require 'json'

# Checks to see whether a user exists in IAM; otherwise,
# creates the user.
#
# @param iam [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [Aws::IAM::Types::User] The existing or new user.
# @example
#   iam = Aws::IAM::Client.new(region: 'us-west-2')
#   user = get_user(iam, 'my-user')
#   exit 1 unless user.user_name
#   puts "User's name: #{user.user_name}"
def get_user(iam, user_name)
  puts "Checking for a user with the name '#{user_name}'..."
  response = iam.get_user(user_name: user_name)
  puts "A user with the name '#{user_name}' already exists."
  response.user
# If the user doesn't exist, create them.
rescue Aws::IAM::Errors::NoSuchEntity
  puts "A user with the name '#{user_name}' doesn't exist. Creating this user..."
  response = iam.create_user(user_name: user_name)
  iam.wait_until(:user_exists, user_name: user_name)
  puts "Created user with the name '#{user_name}'."
  response.user
```

```

rescue StandardError => e
  puts "Error while accessing or creating the user named '#{user_name}':
    #{e.message}"
end

# Gets temporary AWS credentials for an IAM user with the specified permissions.
#
# @param sts [Aws::STS::Client] An initialized AWS STS client.
# @param duration_seconds [Integer] The number of seconds for valid credentials.
# @param user_name [String] The user's name.
# @param policy [Hash] The access policy.
# @return [Aws::STS::Types::Credentials] AWS credentials for API authentication.
# @example
#   sts = Aws::STS::Client.new(region: 'us-west-2')
#   credentials = get_temporary_credentials(sts, duration_seconds, user_name,
#     {
#       'Version' => '2012-10-17',
#       'Statement' => [
#         'Sid' => 'Stmt1',
#         'Effect' => 'Allow',
#         'Action' => 's3:ListBucket',
#         'Resource' => 'arn:aws:s3:::amzn-s3-demo-bucket'
#       ]
#     }
#   )
#   exit 1 unless credentials.access_key_id
#   puts "Access key ID: #{credentials.access_key_id}"
def get_temporary_credentials(sts, duration_seconds, user_name, policy)
  response = sts.get_federation_token(
    duration_seconds: duration_seconds,
    name: user_name,
    policy: policy.to_json
  )
  response.credentials
rescue StandardError => e
  puts "Error while getting federation token: #{e.message}"
end

# Lists the keys and ETags for the objects in an Amazon S3 bucket.
#
# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if the objects were listed; otherwise, false.
# @example

```

```

# s3_client = Aws::S3::Client.new(region: 'us-west-2')
# exit 1 unless list_objects_in_bucket?(s3_client, 'amzn-s3-demo-bucket')
def list_objects_in_bucket?(s3_client, bucket_name)
  puts "Accessing the contents of the bucket named '#{bucket_name}'..."
  response = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if response.count.positive?
    puts "Contents of the bucket named '#{bucket_name}' (first 50 objects):"
    puts 'Name => ETag'
    response.contents.each do |obj|
      puts "#{obj.key} => #{obj.etag}"
    end
  else
    puts "No objects in the bucket named '#{bucket_name}'."
  end
  true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
end

# Example usage:
def run_me
  region = "us-west-2"
  user_name = "my-user"
  bucket_name = "amzn-s3-demo-bucket"

  iam = Aws::IAM::Client.new(region: region)
  user = get_user(iam, user_name)

  exit 1 unless user.user_name

  puts "User's name: #{user.user_name}"
  sts = Aws::STS::Client.new(region: region)
  credentials = get_temporary_credentials(sts, 3600, user_name,
    {
      'Version' => '2012-10-17',
      'Statement' => [
        'Sid' => 'Stmnt1',
        'Effect' => 'Allow',
        'Action' => 's3:ListBucket',

```

```
                                'Resource' =>
"arn:aws:s3:::#{bucket_name}"
                                ]
                                })

exit 1 unless credentials.access_key_id

puts "Access key ID: #{credentials.access_key_id}"
s3_client = Aws::S3::Client.new(region: region, credentials: credentials)

exit 1 unless list_objects_in_bucket?(s3_client, bucket_name)
end

run_me if $PROGRAM_NAME == __FILE__
```

使用 REST API 提出要求

本節包含如何使用 REST API 向 Amazon S3 端點提出請求的資訊。如需 Amazon S3 端點的清單，請參閱《AWS 一般參考》中的[區域與端點](#)。

為 REST API 請求建構 S3 主機名稱

Amazon S3 端點遵循如下所示的結構：

```
s3.Region.amazonaws.com
```

Amazon S3 存取點端點和雙堆疊端點也遵循標準結構：

- Amazon S3 存取點 -s3-accesspoint.*Region*.amazonaws.com
- 雙堆疊 - s3.dualstack.*Region*.amazonaws.com

如需 Amazon S3 區域和端點的完整清單，請參閱《Amazon Web Services 一般參考》中的[Amazon S3 端點和配額](#)。

虛擬主辦式和路徑式請求

使用 REST API 提出請求時，可以使用 Amazon S3 端點的虛擬託管式或路徑式 URI。如需詳細資訊，請參閱[路徑樣式請求](#)。

Example 虛擬託管式請求

下例是從美國西部 (奧勒岡) 區域中名為 puppy.jpg 的儲存貯體中刪除 examplebucket 檔案的虛擬託管式請求。如需虛擬託管樣式請求的詳細資訊，請參閱[路徑樣式請求](#)。

```
DELETE /puppy.jpg HTTP/1.1
Host: examplebucket.s3.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Example 路徑型要求

下例是相同要求的路徑型版本。

```
DELETE /examplebucket/puppy.jpg HTTP/1.1
Host: s3.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Amazon S3 目前支援所有 AWS 區域中的虛擬託管樣式 URL 與路徑樣式 URL 存取權。但是，將來會停產路徑樣式 URL。如需詳細資訊，請參閱下列 Important (重要) 注意事項。

如需路徑樣式請求的詳細資訊，請參閱[路徑樣式請求](#)。

Important

更新 (2020 年 9 月 23 日) – 我們決定將棄用路徑樣式 URL 的日期延後，以確保客戶有足夠時間轉換為虛擬託管樣式 URL。如需詳細資訊，請參閱 AWS 新聞部落格中的 [Amazon S3 路徑廢除計畫 - 其他故事](#)。

使用 REST API 提出雙重堆疊端點要求

使用 REST API 時，可以使用虛擬託管型或路徑型端點名稱 (URI)，直接存取雙重堆疊端點。所有 Amazon S3 雙重堆疊端點名稱，在名稱中都包含區域。與標準型僅 IPv4 端點不同，虛擬託管型和路徑型端點都會使用區域專屬的端點名稱。

Example 虛擬託管式雙重堆疊端點請求

您可在 REST 請求中使用虛擬託管式端點 (如下例所示)，從美國西部 (奧勒岡) 區域中名為 puppy.jpg 的儲存貯體擷取 examplebucket 物件。

```
GET /puppy.jpg HTTP/1.1
Host: examplebucket.s3.dualstack.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Example 路徑型雙重堆疊端點要求

或者，您可在要求中使用路徑型端點，如下例所示。

```
GET /examplebucket/puppy.jpg HTTP/1.1
Host: s3.dualstack.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

如需雙重堆疊端點的詳細資訊，請參閱「[使用 Amazon S3 雙堆疊端點](#)」。

如需使用 REST API 提出請求的詳細資訊，請參閱下列主題。

主題

- [要求重新導向與 REST API](#)
- [要求路由](#)

要求重新導向與 REST API

主題

- [重新導向與 HTTP 使用者代理程式](#)
- [重新導向與 100-Continue](#)
- [重新導向範例](#)

本節描述如何使用 Amazon S3 REST API 處理 HTTP 重新導向。如需 Amazon S3 重新導向的一般資訊，請參閱《Amazon Simple Storage Service API 參考》中的「[提出要求](#)」。

重新導向與 HTTP 使用者代理程式

使用 Amazon S3 REST API 的程式應該會在應用程式層級或 HTTP 層級處理重新導向。許多 HTTP 用戶端程式庫與使用者代理程式會自動設定以正確處理重新導向；不過，還有許多其他重新導向實作不正確或不完整。

在您依賴程式庫完成重新導向需求之前，請測試下列案例：

- 確認包含在重新導向要求 (收到重新導向後的第二個要求) 中的所有 HTTP 要求標頭正確，包含 Authorization 與 Date 等 HTTP 標準。
- 確認非 GET 重新導向 (例如 PUT 與 DELETE) 運作正常。
- 確認大型 PUT 要求正確遵循重新導向。
- 如果 100-continue 回應花很長的時間抵達，請確認 PUT 要求正確遵循重新導向。

當 HTTP 要求方法不是 GET 或 HEAD 時，嚴格遵守 RFC 2616 的 HTTP 使用者代理程式可能需要明確確認，才能遵循重新導向。遵循由 Amazon S3 自動產生的重新導向通常很安全，因為系統只會對 amazonaws.com 網域中的主機發出重新導向，因此重新導向請求的效果會與原始請求相同。

重新導向與 100-Continue

為簡化重新導向處理、提高效率，並避免與傳送重新導向要求本文兩次相關聯的成本，請設定您的應用程式，使用 100-continue 進行 PUT 操作。當您的應用程式使用 100-continue 時，只有在收到確認才會傳送要求本文。如果根據標頭拒絕了訊息，則不會傳送訊息本文。如需 100-continue 的詳細資訊，請前往 [RFC 2616 第 8.2.3 節](#)

Note

根據 RFC 2616，對不明 HTTP 伺服器使用 Expect: Continue 時，您應該不會無限期地等候傳送要求本文。這是因為雖然某些 HTTP 伺服器無法識別 100-continue，但 Amazon S3 會識別您的請求是否包含 Expect: Continue，並回應暫時性 100-continue 狀態或最終狀態碼。此外，收到暫時性 100 continue go-ahead 之後，不會發生重新導向錯誤。這有助於您避免仍在撰寫要求本文時收到重新導向回應。

重新導向範例

本節提供使用 HTTP 重新導向與 100-continue 的主從互動範例。

以下為 `quotes.s3.amazonaws.com` 儲存貯體的 PUT 範例。

```
PUT /nelson.txt HTTP/1.1
Host: quotes.s3.amazonaws.com
Date: Mon, 15 Oct 2007 22:18:46 +0000
```

```
Content-Length: 6
Expect: 100-continue
```

Amazon S3 傳回下列回應：

```
HTTP/1.1 307 Temporary Redirect
Location: http://quotes.s3-4c25d83b.amazonaws.com/nelson.txt?rk=8d47490b
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Mon, 15 Oct 2007 22:18:46 GMT
```

```
Server: AmazonS3
```

```
<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>TemporaryRedirect</Code>
  <Message>Please re-send this request to the
    specified temporary endpoint. Continue to use the
    original request endpoint for future requests.
  </Message>
  <Endpoint>quotes.s3-4c25d83b.amazonaws.com</Endpoint>
  <Bucket>quotes</Bucket>
</Error>
```

用戶端遵循重新導向回應，並對 `quotes.s3-4c25d83b.amazonaws.com` 暫存端點發出新的要求。

```
PUT /nelson.txt?rk=8d47490b HTTP/1.1
Host: quotes.s3-4c25d83b.amazonaws.com
Date: Mon, 15 Oct 2007 22:18:46 +0000
```

```
Content-Length: 6
Expect: 100-continue
```

Amazon S3 傳回 100-continue，指出用戶端應該繼續傳送請求本文。

```
HTTP/1.1 100 Continue
```

用戶端傳送要求本文。

```
ha ha\n
```

Amazon S3 傳回最終回應。

```
HTTP/1.1 200 OK
Date: Mon, 15 Oct 2007 22:18:48 GMT

ETag: "a2c8d6b872054293afd41061e93bc289"
Content-Length: 0
Server: AmazonS3
```

要求路由

對使用 [CreateBucket](#) API (包含 [CreateBucketConfiguration](#)) 建立的儲存貯體提出請求的程式必須支援重新導向。此外，未採用 DNS TTL 的某些用戶端可能會遇到問題。

本節說明設計服務或應用程式以搭配 Amazon S3 使用時須考量的路由與 DNS 問題。

要求重新導向與 REST API

Amazon S3 會使用網域名稱系統 (DNS) 將請求路由到能夠處理請求的設備。此系統有效率執行作業，但是也可能會發生暫時性的路由錯誤。若請求抵達錯誤的 Amazon S3 位置，則 Amazon S3 會以暫時重新導向回應，指示申請者將請求重新傳送到新端點。若請求的格式錯誤，Amazon S3 會使用永久重新導向來指示如何正確執行請求。

Important

您必須擁有可以處理 Amazon S3 重新導向回應的應用程式，才可以使用此功能。此應用程式唯一的例外是專為不使用 `<CreateBucketConfiguration>` 建立儲存貯體。如需位置限制的詳細資訊，請參閱[存取和列出 Amazon S3 儲存貯體](#)。

如果區域是在 2019 年 3 月 20 日後才推出，則當請求抵達錯誤的 Amazon S3 位置時，Amazon S3 會傳回「HTTP 400 錯誤的請求」錯誤。

如需啟用或停用的詳細資訊 AWS 區域，請參閱中的 [AWS 區域和端點](#) AWS 一般參考。

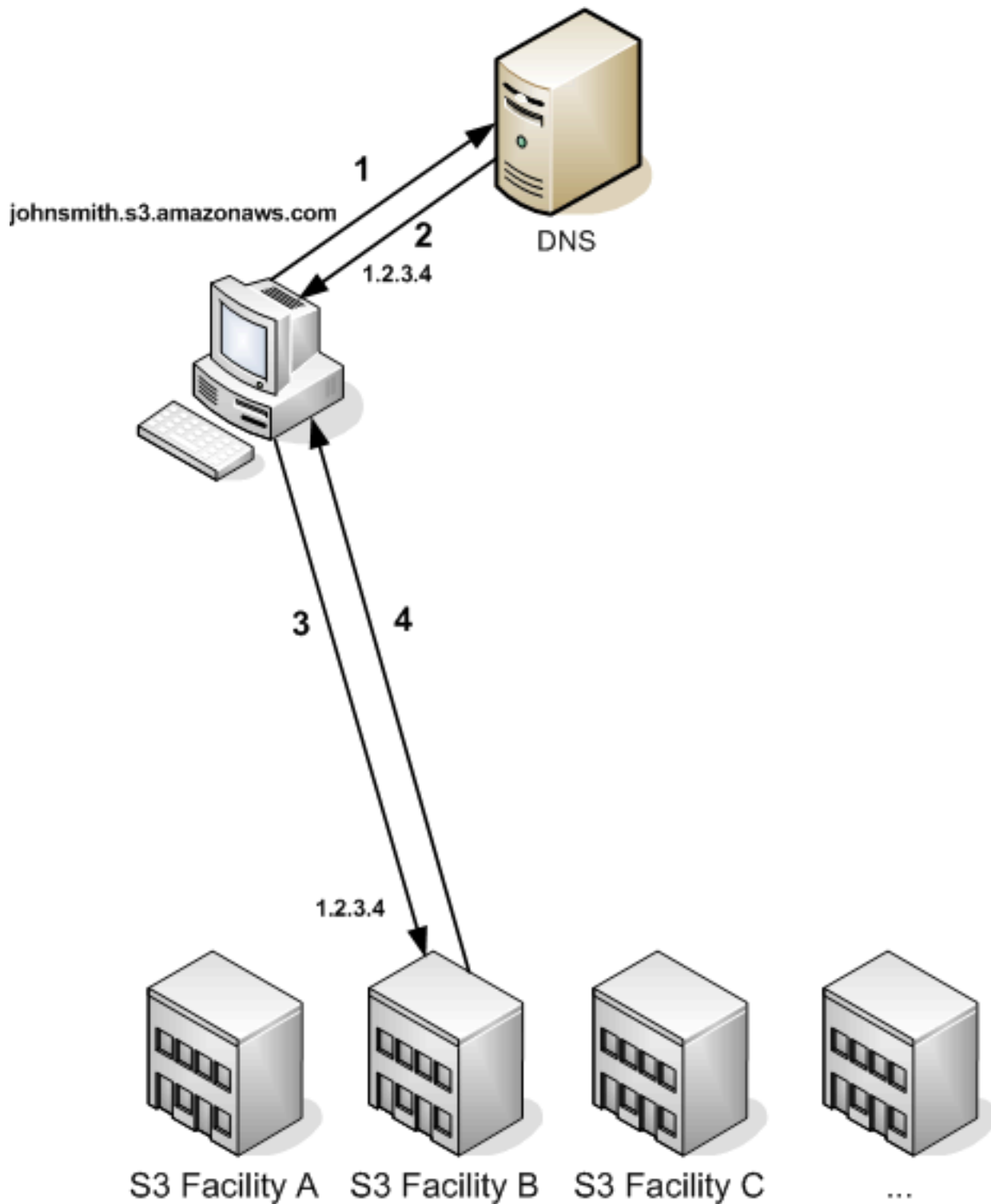
主題

- [DNS 路由](#)

- [暫時要求重新導向](#)
- [永久性要求重新導向](#)
- [要求重新導向範例](#)

DNS 路由

DNS 路由會將請求路由到合適的 Amazon S3 設備。下圖範例顯示 DNS 路由過程。



DNS 路由要求步驟

1. 用戶端提出 DNS 請求，以取得存放在 Amazon S3 上的物件。
2. 用戶端收到一或多個可處理要求的設備 IP 位址。在此範例中，IP 地址為 B 設施。

3. 用戶端向 Amazon S3 設備 B 提出請求。
4. 設備 B 傳回物件的複本至客戶端。

暫時要求重新導向

暫時性重新導向是一種錯誤回應，會向申請者發出訊息，要求其必須將要求重新傳送至其他端點。Amazon S3 的分散特性可能會使請求暫時路由至錯誤的設備。這是相當少見的情況，最可能在建立儲存貯體之後立即發生。

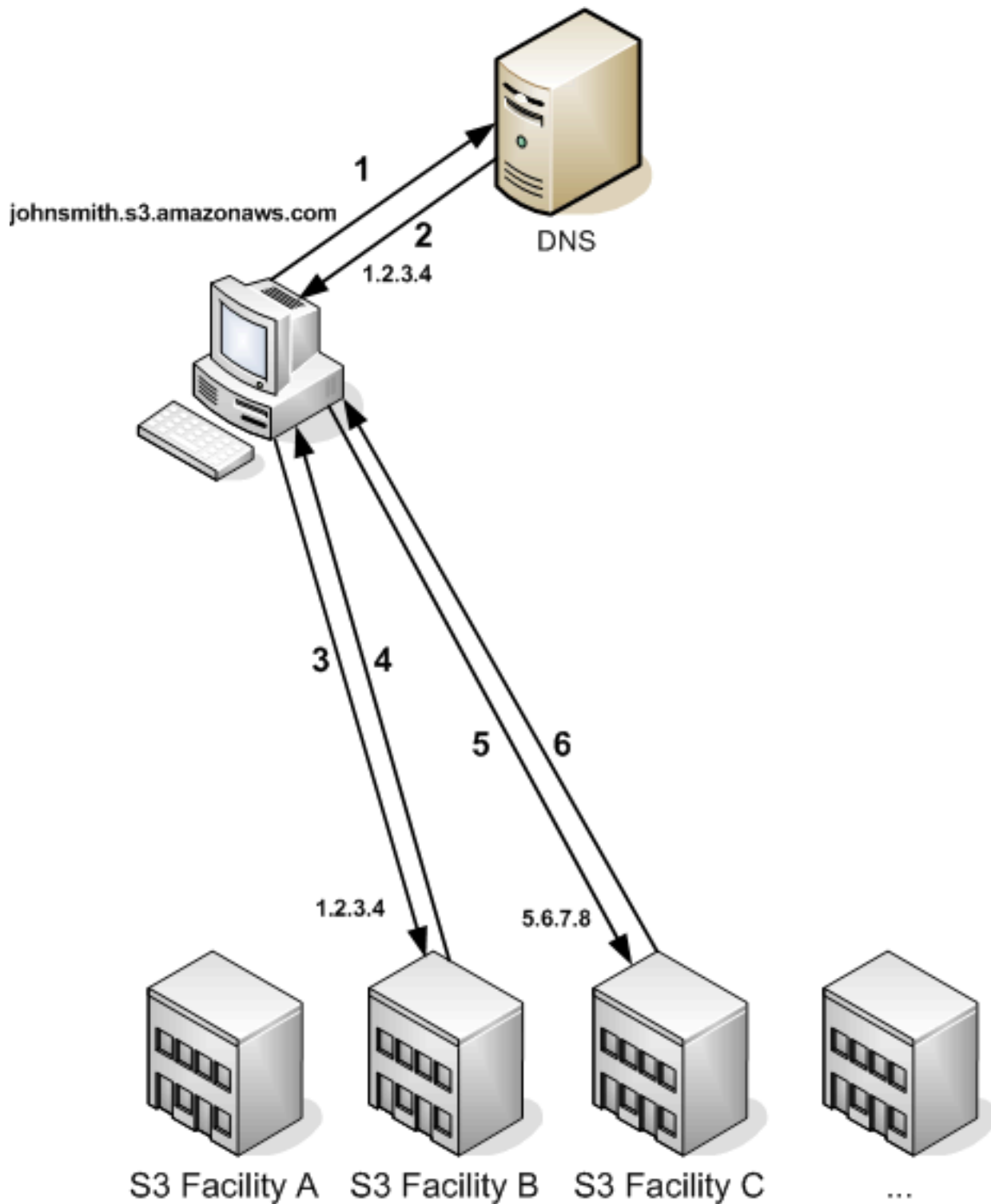
例如，若您在建立新儲存貯體後立即向儲存貯體提出要求，就可能會收到暫時性重新導向，這會視儲存貯體的位置限制條件而定。如果您在美國東部（維吉尼亞北部）建立儲存貯體 AWS 區域，則不會看到重新導向，因為這也是預設的 Amazon S3 端點。

但是，若在任何其他區域建立儲存貯體，則任何對儲存貯體提出的要求都會前往預設端點，而儲存貯體的 DNS 項目會傳播。預設端點會要求重新導向到正確的端點，並傳回 HTTP 302 回應。暫時性重新導向會包含正確設備的 URI，您可使用來立即重新傳送要求。

Important

請勿重複使用前述重新導向回應提供的端點。該端點可能看似運作正常 (即使是長時間運作)，但其可能會提供無法預測的結果，最終導致失敗且不會產生任何通知。

下圖過程顯示暫時性重新導向路由範例。



暫時要求重新導向步驟

1. 用戶端提出 DNS 請求，以取得存放在 Amazon S3 上的物件。
2. 用戶端收到一或多個可處理要求的設備 IP 位址。

3. 用戶端向 Amazon S3 設備 B 提出請求。
4. 設備 B 傳回重新導向，表示可從位置 C 使用該物件。
5. 用戶端將要求重新傳送至設備 C。
6. 設備 C 傳回物件的複本。

永久性要求重新導向

永久性重新導向表示要求設定的資源位置錯誤。例如，若您使用路徑型要求來存取使用 `<CreateBucketConfiguration>` 建立的儲存貯體，就會發生永久性重新導向。如需詳細資訊，請參閱[存取和列出 Amazon S3 儲存貯體](#)。

為協助您在部署期間找出這些錯誤，這類重新導向都不會包含可讓您自動跟隨要求前往正確位置的位置 HTTP 標頭。如需使用正確 Amazon S3 端點方面的說明，請參閱產生的 XML 錯誤文件。

要求重新導向範例

下圖為暫時重新導回應範例。

REST API 暫時要求重新導向回應

```
HTTP/1.1 307 Temporary Redirect
Location: http://awsexamplebucket1.s3-gztb4pa9sq.amazonaws.com/photos/puppy.jpg?rk=e2c69a31
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Fri, 12 Oct 2007 01:12:56 GMT
Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>TemporaryRedirect</Code>
  <Message>Please re-send this request to the specified temporary endpoint. Continue to use the original request endpoint for future requests.</Message>
  <Endpoint>awsexamplebucket1.s3-gztb4pa9sq.amazonaws.com</Endpoint>
</Error>
```


SOAP API 暫時要求重新導向回應

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

```
<soapenv:Body>
  <soapenv:Fault>
    <Faultcode>soapenv:Client.TemporaryRedirect</Faultcode>
    <Faultstring>Please re-send this request to the specified temporary endpoint.
    Continue to use the original request endpoint for future requests.</Faultstring>
    <Detail>
      <Bucket>images</Bucket>
      <Endpoint>s3-gztb4pa9sq.amazonaws.com</Endpoint>
    </Detail>
  </soapenv:Fault>
</soapenv:Body>
```

DNS 考量

Amazon S3 的設計要求之一是極高的可用性。符合此要求的其中一種方式是視需要更新與 DNS 中 Amazon S3 端點相關聯的 IP 地址。這些變更會自動反映至短期用戶端，而不是一些長期用戶端。長期用戶端需要定期採取特殊動作來重新解析 Amazon S3 端點，以從這些變更受益。如需虛擬機器 (VM) 的詳細資訊，請參閱下列項目：

- 針對 Java，根據預設，Sun 的 JVM 會永久快取 DNS 查閱；如需如何變更此行為的資訊，請前往 [InetAddress 文件](#) 的「InetAddress 快取」一節。
- 針對 PHP，除非重新啟動 VM，否則最常見部署組態中執行的持久性 PHP VM 會快取 DNS 查閱。前往 [getHostByName PHP 文件](#)。

使用 使用 Amazon S3 進行開發 AWS CLI

Amazon S3 AWS CLI 命令會在 AWS CLI 命令參考中組織成不同的集合，每個命令都有自己的可用命令。如果您在一組中找不到要尋找的命令，請檢查其他一組。不同的集合如下所示：

- s3 – 描述使用物件和儲存貯體的高階命令。其中包括複製、列出和移動動作。如需此集中命令的完整清單，請參閱《AWS CLI 命令參考》中的 [s3](#)。

- `s3api` – 描述管理儲存貯體、物件、工作階段和分段上傳等 S3 資源的低階命令。這些命令對應至 [Amazon S3 API](#) 操作。如需此集中 CLI 命令的完整清單，請參閱《AWS CLI 命令參考》中的 [s3api](#)。
- `s3control` – 描述管理所有其他 S3 資源的低階命令，例如 Access Grants、Storage Lens 群組和 Amazon S3 on Outposts 儲存貯體。這些命令對應至 [Amazon S3 Control](#) API 操作。如需此集中 CLI 命令的完整清單，請參閱《AWS CLI 命令參考》中的 [s3control](#)。

如果您在設定時遇到問題 AWS CLI，請參閱 AWS re:Post 知識中心中的[如何解決 Amazon S3 中的「找不到登入資料」錯誤](#)。

Note

中的服務 AWS，例如 Amazon S3，需要您在存取憑證時提供憑證。接著服務可以判斷您是否擁有存取其資源的許可。主控台需要您的密碼。您可以為 AWS 帳戶建立存取金鑰，以存取 AWS CLI 或 API。不過，請勿使用的登入 AWS 資料來存取 AWS 帳戶。反之，我們建議您使用 AWS Identity and Access Management (IAM)。建立 IAM 使用者，並將使用者新增至擁有管理許可的 IAM 群組，然後將管理許可授予您建立的 IAM 使用者。之後，您可以使用特殊的 URL 和 IAM 使用者的憑證來存取 AWS。如需說明，請前往 IAM 使用者指南中的[建立第一個 IAM 使用者與管理員群組](#)。

進一步了解 AWS CLI

若要進一步了解 AWS，請參閱下列資源：

- [AWS Command Line Interface](#)
- [AWS Command Line Interface 第 2 版使用者指南](#)

使用 AWS SDKs 使用 Amazon S3 進行開發

AWS 軟體開發套件 (SDKs) 適用於許多熱門的程式設計語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

 Note

您可以使用 AWS Amplify 進行 end-to-end 完整堆疊開發。Amplify Storage 無縫地將檔案儲存和管理功能整合到在 Amazon S3 上建置的前端 Web 和行動應用程式。如需詳細資訊，請參閱 Amplify 使用者指南中的[儲存](#)。

SDK 文件	代碼範例
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK 程式碼範例
AWS CLI	AWS CLI 程式碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK 程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK 程式碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK 程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK 程式碼範例
AWS Tools for PowerShell	適用於 PowerShell 的工具程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK 程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK 程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK 程式碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK 程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK 程式碼範例

如需特定範例，請參閱 [Amazon S3 AWS SDKs 程式碼範例](#)。

SDK 程式設計界面

每個 AWS SDK 提供一或多個程式設計界面，可用於 Amazon S3。每個 SDK 都為 Amazon S3 提供低階介面，方法與 API 操作非常相似。有些 SDKs 為 Amazon S3 提供高階界面，這些界面旨在簡化常見的使用案例。

例如，當您使用低階 API 操作執行分段上傳時，您可以使用 `upload` 操作來啟動上傳、另一個操作來上傳組件，以及最終操作來完成上傳。高階分段上傳 API 操作可讓您在單一 API 呼叫中執行上傳所需的所有操作。如需範例，請參閱 [《Amazon S3 使用者指南》中的使用分段上傳上傳物件](#)。Amazon S3

低階 API 操作可更好地控制上傳。如果您需要暫停和繼續上傳、在上傳期間改變部分大小，或是在您事先不知道資料大小時開始上傳，建議您使用低階 API 操作。

在請求身分驗證中指定簽章版本

Amazon S3 在大多數情況下僅支援 AWS Signature 第 4 版 AWS 區域。在某些較舊版本中 AWS 區域，Amazon S3 同時支援 Signature 第 4 版和 Signature 第 2 版。然而，Signature 第 2 版已結束 (已廢除)。如需 Signature 第 2 版終止支援的詳細資訊，請參閱 [AWS Amazon S3 的 Signature 第 2 版已關閉 \(已棄用\)](#)。

如需所有 Amazon S3 區域及其支援之簽章版本的清單，請參閱《AWS 一般參考》中的 [區域與端點](#)。

對於所有 AWS 區域，AWS SDKs 預設使用 Signature 第 4 版來驗證請求。使用 2016 年 5 月之前發行 AWS SDKs 時，您可能需要請求 Signature 第 4 版，如下表所示。

SDK	要求 Signature 第 4 版進行要求身分驗證
AWS CLI	針對預設描述檔，執行下列命令： <pre>\$ aws configure set default.s3.signature_version s3v4</pre> 針對自訂描述檔，執行下列命令： <pre>\$ aws configure set profile.your_profile_name.s3.signature_version s3v4</pre>
Java 開發套件	在程式碼中新增下列程式碼：

SDK	要求 Signature 第 4 版進行要求身分驗證 <pre>System.setProperty(SDKGlobalConfiguration.ENABLE_S3_SIGV4_SYSTEM_PROPERTY, "true");</pre> 或者，在命令列指定下列項目： <pre>-Dcom.amazonaws.services.s3.enableV4</pre>
PHP SDK	針對 PHP SDK v2 建構 Amazon S3 服務用戶端時，請將 <code>signature</code> 參數設為 <code>v4</code>： <pre><?php \$client = S3Client::factory(['region' => 'YOUR-REGION', 'version' => 'latest', 'signature' => 'v4']);</pre> 使用 PHP SDK 第 3 版時，請在建構 Amazon S3 服務用戶端時，將 <code>signature_version</code> 參數設為 <code>v4</code>： <pre><?php \$s3 = new Aws\S3\S3Client(['version' => '2006-03-01', 'region' => 'YOUR-REGION', 'signature_version' => 'v4']);</pre>
Python-Boto SDK	在 boto 預設組態檔中，指定以下內容： <pre>[s3] use-sigv4 = True</pre>

SDK	要求 Signature 第 4 版進行要求身分驗證
Ruby 開發套件	Ruby 開發套件 - 第 1 版：建構用戶端時，請將 <code>:s3_signature_version</code> 參數設定為 <code>:v4</code>： <pre>s3 = AWS::S3::Client.new(:s3_signature_version => :v4)</pre> Ruby 開發套件 - 第 3 版：建構用戶端時，請將 <code>signature_version</code> 參數設定為 <code>v4</code>： <pre>s3 = Aws::S3::Client.new(signature_version: 'v4')</pre>
.NET 開發套件	在建立 Amazon S3 用戶端之前，將以下內容新增至程式碼： <pre>AWSConfigsS3.UseSignatureVersion4 = true;</pre> 或者，將以下內容新增至組態檔： <pre><appSettings> <add key="AWS.S3.UseSignatureVersion4" value="true" /> </appSettings></pre>

AWS Amazon S3 的 Signature 第 2 版已關閉（已棄用）

Amazon S3 中的 Signature 第 2 版已結束（已廢除）。Amazon S3 將僅接受使用 Signature 第 4 版簽署的 API 請求。

本節提供有關 Signature 第 2 版終止支援的常見問答。

什麼是 Signature 第 2/4 版，而簽署請求又表示什麼意思？

Signature 第 2 版或 Signature 第 4 版簽署程序用於驗證您的 Amazon S3 API 請求。簽署請求可讓 Amazon S3 識別傳送請求的人員，並保護請求免於受到不良執行者的破壞。

如需簽署 AWS 請求的詳細資訊，請參閱《》中的[簽署 AWS API 請求](#) AWS 一般參考。

更新包含什麼內容？

我們目前支援使用 Signature 第 2 版和 Signature 第 4 版程序簽署的 Amazon S3 API 請求。在此之後，Amazon S3 將僅接受使用 Signature 第 4 版簽署的請求。

如需簽署 AWS 請求的詳細資訊，請參閱 [《》中的簽章版本 4](#) 的變更AWS 一般參考。

更新的原因？

Signature 第 4 版使用簽署金鑰 (而不使用私密存取金鑰) 來提高安全性。所有 目前都支援 Signature 第 4 版 AWS 區域，而 Signature 第 2 版僅支援 2014 年 1 月之前啟動的 區域。這項更新可讓我們在所有區域中提供更一致的體驗。

我要如何確定正在使用的是 Signature 第 4 版，以及我需要進行哪些更新？

用於簽署請求的 Signature 版本，其常見的設定方法為用戶端上的工具或 SDK。根據預設，最新版本的 AWS SDKs 會使用 Signature 第 4 版。對於第三方軟體，請聯絡適當的軟體支援團隊以確認您需要的版本。如果要向 Amazon S3 傳送直接 REST 呼叫，必須修改應用程式以使用 Signature 第 4 版簽署程序。

如需有關移動到 Signature 第 4 版時要使用的 AWS SDKs 版本的資訊，請參閱 [從 Signature 第 2 版遷移至 Signature 第 4 版](#)。

如需有關在 Amazon S3 REST API 中使用 Signature 第 4 版的詳細資訊，請參閱《Amazon Simple Storage Service API 參考》中的 [身分驗證請求 \(AWS Signature 第 4 版\)](#)。

如果不更新會發生什麼情況？

在此之後使用 Signature 第 2 版簽署的請求，將無法向 Amazon S3 進行身分驗證。請求者將看到系統顯示錯誤，表示必須使用 Signature 第 4 版簽署請求。

即使我使用的預先簽章 URL 要求大於 7 天的簽署時間，是否也需要進行更新？

如果您使用的預先簽章 URL 要求大於 7 天的簽署時間，則目前無需採取任何動作。您可以繼續使用 AWS Signature 第 2 版來簽署和驗證預先簽章的 URL。如需如何針對預先簽章的 URL 案例遷移至 Signature 第 4 版的詳細資訊，請參閱 Amazon S3 文件。

詳細資訊

- 如需使用 Signature 第 4 版的詳細資訊，請參閱[簽署 AWS API 請求](#)。

- 在 [Signature 第 4 版的變更](#) 中檢視 Signature 第 2 版和 Signature 第 4 版之間的變更清單。
- 檢視 [AWS Signature 第 4 版文章](#)，取代 AWS Signature 第 2 版，以在論壇中簽署 Amazon S3 API 請求。AWS
- 如果您有其他問題或疑慮，請聯絡 [支援](#)。

從 Signature 第 2 版遷移至 Signature 第 4 版

如果您目前使用 Signature 第 2 版進行 Amazon S3 API 請求身分驗證，應改為使用 Signature 第 4 版。系統即將終止對 Signature 第 2 版的支援，如 [AWS Amazon S3 的 Signature 第 2 版已關閉（已棄用）](#) 中所述。

如需有關在 Amazon S3 REST API 中使用 Signature 第 4 版的詳細資訊，請參閱《Amazon Simple Storage Service API 參考》中的 [身分驗證請求 \(AWS Signature 第 4 版\)](#)。

下表列出使用 Signature 第 4 版 (SigV4) 所需的最低軟體開發套件版本。如果您使用預先簽章URLs 搭配 AWS Java、JavaScript (Node.js) 或 Python (Boto/CLI) SDKs，您必須在用戶端組態中設定正確的 AWS 區域 簽章版本 4。如需在用戶端組態中設定 SigV4 的資訊，請參閱 [在請求身分驗證中指定簽章版本](#)。

如果您使用此軟體開發套件/產品	升級至此軟體開發套件版本	用戶端使用 Sigv4 時需要變更程式碼嗎？	軟體開發套件文件的連結
適用於 Java 的 AWS SDK v1	升級至 Java 1.11.201+ 或 v2。	是	在請求身分驗證中指定簽章版本
適用於 Java 的 AWS SDK v2	無需升級軟體開發套件。	否	適用於 Java 的 AWS SDK
適用於 .NET 的 AWS SDK v1	升級至 3.1.10 或更新版本。	是	適用於 .NET 的 AWS SDK

如果您使用此軟體開發套件/產品	升級至此軟體開發套件版本	用戶端使用 Sigv4 時需要變更程式碼嗎？	軟體開發套件文件的連結
適用於 .NET 的 AWS SDK v2	升級至 3.1.10 或更新版本。	否	適用於 .NET 的 AWS SDK v2
適用於 .NET 的 AWS SDK v3	升級至 3.3.0.0 或更新版本。	是	適用於 .NET 的 AWS SDK v3
適用於 JavaScript 的 AWS SDK v1	升級至 2.68.0 或更新版本。	是	適用於 JavaScript 的 AWS SDK
適用於 JavaScript 的 AWS SDK v2	升級至 2.68.0 或更新版本。	是	適用於 JavaScript 的 AWS SDK
適用於 JavaScript 的 AWS SDK v3	目前無需採取動作。在 2019 年第 3 季升級至主要版本 V3。	否	適用於 JavaScript 的 AWS SDK
適用於 PHP 的 AWS SDK v1	建議升級至 PHP 的最新版本，或至少升級至 v2.7.4，其中 S3 用戶端組態中的簽署參數應設為 v4。	是	適用於 PHP 的 AWS SDK

如果您使用此軟體開發套件/產品	升級至此軟體開發套件版本	用戶端使用 Sigv4 時需要變更程式碼嗎？	軟體開發套件文件的連結
適用於 PHP 的 AWS SDK v2	建議升級至 PHP 的最新版本，或至少升級至 v2.7.4，其中 S3 用戶端組態中的簽署參數應設為 v4。	否	適用於 PHP 的 AWS SDK
適用於 PHP 的 AWS SDK v3	無需升級軟體開發套件。	否	適用於 PHP 的 AWS SDK
Boto2	升級至 Boto2 v2.49.0。	是	Boto 2 升級
Boto3	升級至 1.5.71 (Botocore)、1.4.6 (Boto3)。	是	適用於 Python 的 Boto 3 - AWS SDK
AWS CLI	升級至 1.11.108。	是	AWS Command Line Interface
AWS CLI v2 (預覽版)	無需升級軟體開發套件。	否	AWS Command Line Interface 第 2 版
適用於 Ruby 的 AWS SDK v1	升級至 Ruby V3。	是	適用於 AWS 的 Ruby V3

如果您使用此軟體開發套件/產品	升級至此軟體開發套件版本	用戶端使用 Sigv4 時需要變更程式碼嗎？	軟體開發套件文件的連結
適用於 Ruby 的 AWS SDK v2	升級至 Ruby V3。	是	適用於 AWS 的 Ruby V3
適用於 Ruby 的 AWS SDK v3	無需升級軟體開發套件。	否	適用於 AWS 的 Ruby V3
Go	無需升級軟體開發套件。	否	適用於 Go 的 AWS SDK
C++	無需升級軟體開發套件。	否	適用於 C++ 的 AWS SDK

AWS Tools for Windows PowerShell 或 AWS Tools for PowerShell Core

如果您使用的模組版本早於 3.3.0.0，則必須升級到 3.3.0.0。

如需取得版本資訊，請使用 Get-Module cmdlet：

```
Get-Module -Name AWSPowershell
Get-Module -Name AWSPowershell.NetCore
```

若要更新 3.3.0.0 版，請使用 Update-Module cmdlet：

```
Update-Module -Name AWSPowershell
Update-Module -Name AWSPowershell.NetCore
```

您可以使用有效期超過 7 天的預先簽署 URL，您將使用該 URL 傳送 Signature 第 2 版流量。

取得的 Amazon S3 請求 IDs AWS 支援

每當您 AWS 支援 因為在 Amazon S3 中遇到錯誤或意外行為而聯絡 時，您必須提供與失敗動作相關聯的請求 IDs。AWS 支援 會使用這些請求 IDs 來協助解決您遇到的問題。

請求 ID 成對出現、在 Amazon S3 處理的每個回應中傳回 (甚至是錯誤回應)，而且可透過詳細日誌取得。取得請求 IDs 的方法有很多種，包括 S3 伺服器存取日誌和 AWS CloudTrail 事件或資料事件。

復原這些日誌之後，請複製並保留這兩個值，因為您在聯絡時需要它們 AWS 支援。如需聯絡的相關資訊 AWS 支援，請參閱[聯絡 AWS](#)或 [AWS 支援 文件](#)。

主題

- [使用 AWS CLI 取得請求 IDs](#)
- [使用 Windows PowerShell 取得請求識別碼](#)
- [使用 AWS CloudTrail 資料事件來取得請求 IDs](#)
- [使用 S3 伺服器存取記錄日誌來取得請求 ID](#)
- [使用 HTTP 取得請求 ID](#)
- [使用 Web 瀏覽器取得請求 ID](#)
- [使用 AWS SDKs 取得請求 IDs](#)

使用 AWS CLI 取得請求 IDs

若要在使用 AWS Command Line Interface (AWS CLI) 時取得您的請求 IDs，請將 `--debug` 新增至您的命令。例如，您應該會在偵錯日誌 `x-amz-id-2` 中看到 `x-amz-request-id` 和 `x-amz-id-1`，如下所示：

```
...
2025-04-30 14:35:28,572 - MainThread - botocore.parsers - DEBUG - Response headers:
{'x-amz-id-2': 'a+zm50vDJH3LLmCiXvwEo0u0PtPS/qCJaBvB2ZMH9dzyzTiJhiLZkBFFoRfsPf0KztUKT/
garCI=', 'x-amz-request-id': 'N4NMN0MJ4VDFZMX9', 'Date': 'Wed, 30 Apr 2025 21:35:29
GMT', 'Content-Type': 'application/xml', 'Transfer-Encoding': 'chunked', 'Server':
'AmazonS3'}
...
```

使用 Windows PowerShell 取得請求識別碼

如需使用 Windows PowerShell 復原日誌的資訊，請參閱《AWS 開發人員工具部落格》中的 [中的回應記錄 AWS Tools for Windows PowerShell](#)。

使用 AWS CloudTrail 資料事件來取得請求 IDs

使用 CloudTrail 資料事件設定的 Amazon S3 儲存貯體，可記錄 S3 物件層級 API 操作，提供有關使用者、角色或 Amazon S3 中 AWS 服務所採取動作的詳細資訊。您可以透過使用 Athena 查詢 CloudTrail 事件來識別 S3 請求 ID。

如需詳細資訊，請參閱 [《Amazon Simple Storage Service 使用者指南》中的使用 CloudTrail 識別 Amazon S3 請求 CloudTrail](#)。

使用 S3 伺服器存取記錄日誌來取得請求 ID

為 S3 伺服器存取記錄設定的 Amazon S3 儲存貯體，針對向儲存貯體提出的請求，提供詳細的記錄。您可以使用 Athena 查詢伺服器存取日誌來識別 S3 請求 ID。

如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [使用 查詢請求的存取日誌 Amazon Athena](#)。

使用 HTTP 取得請求 ID

您可以在 HTTP 要求傳抵目標應用程式之前，先記錄 HTTP 要求的位元數，從而取得自己的要求 ID x-amz-request-id 與 x-amz-id-2。您有多種第三方工具可供您用來復原 HTTP 要求的詳細日誌。請選擇您信任的工具來執行，當您送出另一個 Amazon S3 HTTP 請求時，接聽 Amazon S3 流量行經的連接埠。

HTTP 請求的請求 ID 對類似下列範例所示：

```
x-amz-request-id: 79104EXAMPLEB723
x-amz-id-2: IOwQ4fDEXAMPLEQM+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK+Jd1vEXAMPLEa3Km
```

Note

在大部分的封包擷取中，多會加密及隱藏 HTTPS 要求。

使用 Web 瀏覽器取得請求 ID

大多數 Web 瀏覽器都會提供開發人員工具，您可用來檢視請求標頭。

傳回錯誤之 Web 瀏覽器要求的要求 ID 對類似下列範例所示。

```
<Error><Code>AccessDenied</Code><Message>Access Denied</Message>
```

```
<RequestId>79104EXAMPLEB723</RequestId><HostId>IOWQ4fDEXAMPLEQM  
+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK+Jd1vEXAMPLEa3Km</HostId></Error>
```

若要從成功的請求取得請求 ID 對，請使用瀏覽器的開發人員工具檢視 HTTP 回應標頭。

使用 AWS SDKs 取得請求 IDs

下列各節包含使用不同 AWS SDKs 設定記錄的資訊。

Note

雖然您可以在每個請求和回應上啟用詳細記錄，但不建議在生產系統中啟用記錄，因為大型請求或回應可能會大幅降低應用程式的速度。

對於 AWS SDK 請求，一對請求 IDs 會如下所示：

```
Status Code: 403, AWS Service: Amazon S3, AWS Request ID: 79104EXAMPLEB723  
AWS Error Code: AccessDenied AWS Error Message: Access Denied  
S3 Extended Request ID: IOWQ4fDEXAMPLEQM+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK  
+Jd1vEXAMPLEa3Km
```

C++

記錄器的類型和詳細程度會在 SDKOptions 引數中的 SDK 初始化期間指定。下列範例會將詳細程度指定為 LogLevel::Debug。

預設記錄器會寫入檔案系統，並使用下列慣例將檔案命名為 aws_sdk_YYYY-MM-DD-HH.log。記錄器會在該小時建立新的檔案。

```
Aws::SDKOptions options;  
options.loggingOptions.logLevel = Aws::Utils::Logging::LogLevel::Debug;  
Aws::InitAPI(options);  
// ...  
Aws::ShutdownAPI(options);
```

如需詳細資訊，請參閱[如何開啟記錄功能？](#) 適用於 AWS GitHub 上 C++ wiki 的 SDK。

Go

您可以使用 SDK for Go 來設定記錄。如需詳細資訊，請參閱《適用於 Go 的 AWS SDK v2 開發人員指南》中的[記錄](#)。

Java

您可以啟用特定請求或回應的記錄，以僅擷取和傳回相關標頭。若要執行此作業，請匯入 `com.amazonaws.services.s3.S3ResponseMetadata` 類別。如此一來，您就能將請求存放在變數中，然後再執行實際的要求。若要取得所記錄的請求或回應，請呼叫 `getCachedResponseMetadata(AmazonWebServiceRequest request).getRequestID()`。

```
PutObjectRequest req = new PutObjectRequest(bucketName, key, createSampleFile());
s3.putObject(req);
S3ResponseMetadata md = s3.getCachedResponseMetadata(req);
System.out.println("Host ID: " + md.getHostId() + " RequestID: " +
    md.getRequestId());
```

或者，您可以對每一個 Java 要求與回應使用詳細記錄功能。如需詳細資訊，請參閱《適用於 Java 的 AWS SDK 開發人員指南》中的[詳細連線記錄](#)。

JavaScript

適用於 JavaScript 的 AWS SDK 具有內建記錄器，因此您可以記錄使用它進行的 API 呼叫。若要在主控台中開啟記錄器並列印日誌項目，請將下列陳述式新增至您的程式碼：

```
AWS.config.logger = console;
```

如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》中的[記錄 適用於 JavaScript 的 AWS SDK 通話](#)。

Kotlin

使用適用於 Kotlin 的 AWS SDK，您可以使用程式碼或環境設定來指定線路層級訊息的日誌模式。您可以設定 HTTP 請求和 HTTP 回應的日誌模式。

若要選擇加入其他記錄，請在建構服務用戶端時設定 `logMode` 屬性：

```
import aws.smithy.kotlin.runtime.client.LogMode

// ...

val client = S3Client {
    // ...
    logMode = LogMode.LogRequestWithBody + LogMode.LogResponse
}
```

或者，您可以使用 環境變數來設定日誌模式：

```
export SDK_LOG_MODE=LogRequestWithBody|LogResponse
```

如需詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK 開發人員指南》中的[記錄](#)。

.NET

您可以使用內建記錄工具適用於 .NET 的 SDK，使用 設定System.Diagnostics記錄。如需詳細資訊，請參閱使用AWS 開發人員部落格文章[記錄 適用於 .NET 的 SDK](#)。

Note

傳回的記錄預設只會包含錯誤資訊。若要取得請求 ID，必須在設定檔中新增 AWSLogMetrics (以及選用的 AWSResponseLogging)。

PHP

您可以在用戶端建構函數true中將偵錯選項設定為，以取得偵錯資訊，包括透過線路傳送的資料。

```
$s3Client = new Aws\S3\S3Client([
    'region' => 'us-standard',
    'version' => '2006-03-01',
    'debug'   => true
]);
```

如需詳細資訊，請參閱適用於《適用於 PHP 的 AWS SDK 開發人員指南》中的[如何得知網路上傳送哪些資料？](#)。

Python (Boto3)

使用 適用於 Python (Boto3) 的 AWS SDK，您可以記錄特定回應。您可以使用此功能僅擷取相關標頭。下列程式碼顯示如何將回應的部分記錄到檔案中：

```
import logging
import boto3
logging.basicConfig(filename='logfile.txt', level=logging.INFO)
logger = logging.getLogger(__name__)
s3 = boto3.resource('s3')
response = s3.Bucket(bucket_name).Object(object_key).put()
```



```
logger.info("HTTPStatusCode: %s", response['ResponseMetadata']['HTTPStatusCode'])
logger.info("RequestId: %s", response['ResponseMetadata']['RequestId'])
logger.info("HostId: %s", response['ResponseMetadata']['HostId'])
logger.info("Date: %s", response['ResponseMetadata']['HTTPHeaders']['date'])
```

您也可以擷取例外狀況，並在引發例外狀況時記錄相關資訊。如需詳細資訊，請參閱適用於 Python (Boto) 的 AWS 開發套件 API 參考中的[從錯誤回應中辨別實用的資訊](#)。

此外，您可以使用下列程式碼，配置 Boto3 輸出詳細資訊偵錯記錄檔：

```
import logging
import boto3
boto3.set_stream_logger('', logging.DEBUG)
```

如需詳細資訊，請參閱適用於 Python (Boto) 的 AWS 開發套件 API 參考中的[set_stream_logger](#)。

Ruby

您可以使用適用於 Ruby IDs。您可以使用下列程式碼在用戶端中啟用 HTTP 線路記錄：

```
s3 = Aws::S3::Client.new(http_wire_trace: true)
```

Note

線路記錄可以包含敏感資訊，例如您的存取金鑰 ID。在與 AWS Support 共用敏感資訊之前，應對其進行淨化。

您也可以在請求回應或錯誤中找到請求內容物件的請求 ID：

```
# Finding the request ID from an error:
begin
  s3.put_object(bucket: 'bucket', key: 'key', body: 'test')
rescue Aws::S3::Errors::ServiceError => e
  puts e.context[:request_id]
  puts e.context[:s3_host_id]
end

# Finding the request ID from a successful call:
resp = s3.put_object(bucket: 'bucket', key: 'key', body: 'test')
puts resp.context[:request_id]
puts resp.context[:s3_host_id]
```

如需詳細資訊，請參閱《適用於 Ruby 的 AWS SDK 開發人員指南[適用於 Ruby 的 AWS SDK](#)》中的[使用用戶端的線路追蹤資訊進行偵錯](#)。

Rust

若要啟用記錄，請新增tracing-subscriber木箱並在 Rust 應用程式中初始化木箱。

將追蹤程式庫新增至您的 Cargo.toml 檔案：

```
tracing-subscriber = { version = "0.3", features = ["env-filter"] }
```

然後，在您的 Rust 程式碼中，先在主要函數中初始化記錄器，再呼叫任何 SDK 操作：

```
tracing_subscriber::fmt::init();
```

如需詳細資訊，請參閱《適用於 Rust 的 AWS SDK 開發人員指南》中的在 [中設定和使用記錄 適用於 Rust 的 AWS SDK](#)。

SAmazon S3 S3 物件層級 API 操作

下表包含支援的 S3 物件層級 API 操作，以及 S3 Tables 的對應標頭。如需 Amazon S3 Tables APIs 的清單，請參閱 [Amazon S3 Tables](#)。如需 Amazon S3 資料表的詳細資訊，請參閱《[Amazon S3 使用者指南](#)》中的[使用 Amazon S3 資料表和資料表儲存貯體](#)。Amazon S3

物件層級 API	支援的標頭	備註	
AbortMultipartUpload	x-amz-expected-bucket-owner	無	
CompleteMultipartUpload	x-amz-checksum-crc32	無	
	x-amz-checksum-crc32c	無	
	x-amz-checksum-sha1	無	

物件層級 API	支援的標頭	備註	
	x-amz-checksum-sha256	無	
	x-amz-expected-bucket-owner	無	
	If-Match	無	
	If-None-Match	無	
CreateMultipartUpload	x-amz-acl: ACL	對於 S3 Tables , 預設值為 bucket-owner-full-control 且無法變更。	
	Cache-Control	無	
	Content-Disposition	無	
	Content-Encoding	無	
	Content-Language	無	
	Content-Type	無	
	Expires	無	
	x-amz-checksum-algorithm	無	
	x-amz-storage-class	對於 S3 Tables , 預設值為 STANDARD且無法變更。	

物件層級 API	支援的標頭	備註	
	x-amz-server-side-encryption	對於 S3 Tables , 預設值為 (SSE-S3) (AES256) , 且無法變更。	
GetObject	If-Match	無	
	If-Modified-Since	無	
	If-None-Match	無	
	If-Unmodified-Since	無	
	Range	無	
	x-amz-expected-bucket-owner	無	
	x-amz-checksum-mode	無	
HeadObject	If-Match	無	
	If-Modified-Since	無	
	If-None-Match	無	
	If-Unmodified-Since	無	
	Range	無	
	x-amz-checksum-mode	無	

物件層級 API	支援的標頭	備註	
	x-amz-expected-bucket-owner	無	
ListParts	x-amz-expected-bucket-owner	無	
PutObject	x-amz-acl: ACL	對於 S3 Tables , 預設值為 bucket-owner-full-control 且無法變更。	
	Cache-Control	無	
	Content-Disposition	無	
	Content-Encoding	無	
	Content-Language	無	
	Content-Length	無	
	Content-MD5	無	
	Content-Type	無	
	x-amz-sdk-checksum-algorithm	無	
	x-amz-checksum-crc32	無	
	x-amz-checksum-crc32c	無	

物件層級 API	支援的標頭	備註	
	x-amz-checksum-sha1	無	
	x-amz-checksum-sha256	無	
	Expires	無	
	If-None-Match	無	
	x-amz-expected-bucket-owner	無	
	x-amz-storage-class	對於 S3 Tables，預設值為 STANDARD 且無法變更。	
	x-amz-server-side-encryption	對於 S3 Tables，預設值為 (SSE-S3) (AES256)，且無法變更。	
UploadPart	Content-Length	無	
	Content-MD5	無	
	x-amz-checksum-crc32	無	
	x-amz-checksum-crc32c	無	
	x-amz-checksum-sha1	無	
	x-amz-checksum-sha256	無	

物件層級 API	支援的標頭	備註	
	x-amz-expected-bucket-owner	無	

Amazon S3 AWS SDKs程式碼範例

下列程式碼範例示範如何使用 Amazon S3 搭配 AWS 軟體開發套件 (SDK)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

程式碼範例

- [Amazon S3 AWS SDKs程式碼範例](#)
 - [Amazon S3 AWS SDKs的基本範例](#)
 - [您好 Amazon S3](#)
 - [使用 AWS SDK 了解 Amazon S3 的基本概念](#)
 - [Amazon S3 AWS SDKs的動作](#)
 - [AbortMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
 - [CompleteMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
 - [CopyObject 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateBucket 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateMultiRegionAccessPoint 搭配 AWS SDK 使用](#)
 - [CreateMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
 - [CreatePresignedPost 搭配 AWS SDK 使用](#)
 - [DeleteBucket 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeleteBucketAnalyticsConfiguration 與 CLI](#)
 - [DeleteBucketCors 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeleteBucketEncryption 與 CLI](#)
 - [搭配使用 DeleteBucketInventoryConfiguration 與 CLI](#)
 - [DeleteBucketLifecycle 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeleteBucketMetricsConfiguration 與 CLI](#)
 - [DeleteBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeleteBucketReplication 與 CLI](#)
 - [搭配使用 DeleteBucketTagging 與 CLI](#)
 - [DeleteBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteObject 搭配 AWS SDK 或 CLI 使用](#)

- [搭配使用 DeleteObjectTagging 與 CLI](#)
- [DeleteObjects 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeletePublicAccessBlock 與 CLI](#)
- [搭配使用 GetBucketAccelerateConfiguration 與 CLI](#)
- [GetBucketAcl 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketAnalyticsConfiguration 與 CLI](#)
- [GetBucketCors 搭配 AWS SDK 或 CLI 使用](#)
- [GetBucketEncryption 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketInventoryConfiguration 與 CLI](#)
- [GetBucketLifecycleConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [GetBucketLocation 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketLogging 與 CLI](#)
- [搭配使用 GetBucketMetricsConfiguration 與 CLI](#)
- [搭配使用 GetBucketNotification 與 CLI](#)
- [GetBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketPolicyStatus 與 CLI](#)
- [GetBucketReplication 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketRequestPayment 與 CLI](#)
- [搭配使用 GetBucketTagging 與 CLI](#)
- [搭配使用 GetBucketVersioning 與 CLI](#)
- [GetBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
- [GetObject 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectAcl 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectAttributes 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectLegalHold 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectLockConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectRetention 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetObjectTagging 與 CLI](#)
- [搭配使用 GetPublicAccessBlock 與 CLI](#)
- [HeadBucket 搭配 AWS SDK 或 CLI 使用](#)

- [HeadObject 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 ListBucketAnalyticsConfigurations 與 CLI](#)
- [搭配使用 ListBucketInventoryConfigurations 與 CLI](#)
- [ListBuckets 搭配 AWS SDK 或 CLI 使用](#)
- [ListMultipartUploads 搭配 AWS SDK 或 CLI 使用](#)
- [ListObjectVersions 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 ListObjects 與 CLI](#)
- [ListObjectsV2 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketAccelerateConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketAcl 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketCors 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketEncryption 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketLifecycleConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketLogging 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 PutBucketNotification 與 CLI](#)
- [PutBucketNotificationConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketReplication 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 PutBucketRequestPayment 與 CLI](#)
- [搭配使用 PutBucketTagging 與 CLI](#)
- [PutBucketVersioning 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
- [PutObject 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectAcl 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectLegalHold 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectLockConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectRetention 搭配 AWS SDK 或 CLI 使用](#)
- [RestoreObject 搭配 AWS SDK 或 CLI 使用](#)
- [SelectObjectContent 搭配 AWS SDK 或 CLI 使用](#)
- [UploadPart 搭配 AWS SDK 或 CLI 使用](#)

- [UploadPartCopy 搭配 AWS SDK 或 CLI 使用](#)
- [Amazon S3 AWS SDKs 案例](#)
 - [檢查儲存貯體是否存在](#)
 - [檢查物件是否存在](#)
 - [使用 AWS SDK 將文字轉換為語音並返回文字](#)
 - [使用 AWS SDK 建立 Amazon S3 的預先簽章 URL](#)
 - [建立相片資產管理應用程式，讓使用者以標籤管理相片](#)
 - [使用 AWS SDK 列出 Amazon S3 物件的網頁](#)
 - [建立 Amazon Textract Explorer 應用程式](#)
 - [使用 AWS SDK 刪除指定 Amazon S3 儲存貯體中的所有物件](#)
 - [使用 AWS SDK 刪除不完整的分段上傳至 Amazon S3](#)
 - [使用 AWS SDK 透過 Amazon Rekognition 偵測映像中的個人防護裝備](#)
 - [使用 AWS SDK 偵測從映像擷取的文字中的實體](#)
 - [使用 AWS SDK 偵測映像中的臉部](#)
 - [使用 AWS SDK 透過 Amazon Rekognition 偵測映像中的物件](#)
 - [使用 AWS SDK 透過 Amazon Rekognition 偵測影片中的人物和物件](#)
 - [從 Amazon Simple Storage Service \(Amazon S3\) 儲存貯體下載 S3 'directories'](#)
 - [將 Amazon Simple Storage Service \(Amazon S3\) 儲存貯體中的所有物件下載至本機目錄](#)
 - [使用 AWS SDK 從 Amazon S3 物件下載未知大小的串流](#)
 - [使用 SDK 從多區域存取點取得 Amazon S3 物件 AWS](#)
 - [使用 AWS SDK 從 Amazon S3 儲存貯體取得物件，指定 If-Modified-Since 標頭](#)
 - [使用 AWS SDK 開始加密 Amazon S3 物件](#)
 - [使用 AWS SDK 開始使用 Amazon S3 物件的標籤](#)
 - [大數據處理叢集入門](#)
 - [組態管理入門](#)
 - [文件文字擷取入門](#)
 - [機器學習功能存放區的入門](#)
 - [物件儲存入門](#)
 - [查詢分析入門](#)
 - [使用 AWS SDK 使用 Amazon S3 物件鎖定功能](#)

- [使用 SDK 提出 Amazon S3 條件式請求 AWS](#)
- [使用 AWS SDK 管理 Amazon S3 儲存貯體的存取控制清單 \(ACLs\)](#)
- [透過 SDK 使用 Amazon S3 管理大型 Amazon SQS 訊息 Amazon S3 AWS](#)
- [使用 SDK 使用 Lambda 函數分批管理版本控制的 Amazon S3 物件 AWS](#)
- [使用 AWS SDK 剖析 Amazon S3 URIs](#)
- [使用 AWS SDK 執行 Amazon S3 物件的分段複製](#)
- [使用 SDK 接收和處理 Amazon S3 事件通知 AWS](#)
- [使用 AWS SDK 儲存 EXIF 和其他影像資訊](#)
- [使用 AWS SDK 將 S3 事件通知傳送至 Amazon EventBridge](#)
- [使用 SDK 追蹤 Amazon S3 物件上傳或下載 AWS](#)
- [使用 S3 Object Lambda 轉換應用程式的資料](#)
- [使用 AWS SDK 進行單元和整合測試的範例方法](#)
- [以遞迴的方式將本機目錄上傳至 Amazon Simple Storage Service \(Amazon S3\) 儲存貯體](#)
- [使用 AWS SDK 在 Amazon S3 之間上傳或下載大型檔案](#)
- [使用 AWS SDK 將大小不明的串流上傳至 Amazon S3 物件](#)
- [使用檢查總和來使用 AWS SDK 處理 Amazon S3 物件](#)
- [使用 AWS SDK 使用 Amazon S3 物件完整性功能](#)
- [使用 AWS SDK 處理 Amazon S3 版本控制的物件](#)
- [Amazon S3 的無伺服器範例](#)
 - [使用 Amazon S3 觸發條件調用 Lambda 函數](#)
- [使用 AWS SDKs 的 Amazon S3 控制程式碼範例](#)
- [使用 AWS SDKs 的 Amazon S3 控制基本範例](#)
 - [Hello Amazon S3 Control](#)
 - [使用 AWS SDK 了解 Amazon S3 控制的基本概念](#)
 - [使用 AWS SDKs 的 Amazon S3 控制動作](#)
 - [CreateJob 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteJobTagging 搭配 AWS SDK 使用](#)
 - [DescribeJob 搭配 AWS SDK 或 CLI 使用](#)
 - [GetJobTagging 搭配 AWS SDK 使用](#)
 - [PutJobTagging 搭配 AWS SDK 使用](#)

- [UpdateJobPriority 搭配 AWS SDK 或 CLI 使用](#)
- [UpdateJobStatus 搭配 AWS SDK 或 CLI 使用](#)
- [使用 SDK 的 S3 目錄儲存貯體程式碼範例 AWS SDKs](#)
 - [使用 AWS SDKs 的 S3 目錄儲存貯體基本範例](#)
 - [Hello Amazon S3 目錄儲存貯體](#)
 - [使用 AWS SDK 了解 S3 目錄儲存貯體的基本概念](#)
 - [使用 AWS SDKs 的 S3 目錄儲存貯體動作](#)
 - [AbortMultipartUpload 搭配 AWS SDK 使用](#)
 - [CompleteMultipartUpload 搭配 AWS SDK 使用](#)
 - [CopyObject 搭配 AWS SDK 使用](#)
 - [CreateBucket 搭配 AWS SDK 使用](#)
 - [CreateMultipartUpload 搭配 AWS SDK 使用](#)
 - [CreateSession 搭配 AWS SDK 使用](#)
 - [DeleteBucket 搭配 AWS SDK 使用](#)
 - [DeleteBucketEncryption 搭配 AWS SDK 使用](#)
 - [DeleteBucketPolicy 搭配 AWS SDK 使用](#)
 - [DeleteObject 搭配 AWS SDK 使用](#)
 - [DeleteObjects 搭配 AWS SDK 使用](#)
 - [GetBucketEncryption 搭配 AWS SDK 使用](#)
 - [GetBucketPolicy 搭配 AWS SDK 使用](#)
 - [GetObject 搭配 AWS SDK 使用](#)
 - [GetObjectAttributes 搭配 AWS SDK 使用](#)
 - [HeadBucket 搭配 AWS SDK 使用](#)
 - [HeadObject 搭配 AWS SDK 使用](#)
 - [ListDirectoryBuckets 搭配 AWS SDK 使用](#)
 - [ListMultipartUploads 搭配 AWS SDK 使用](#)
 - [ListObjectsV2 搭配 AWS SDK 使用](#)
 - [ListParts 搭配 AWS SDK 使用](#)
 - [PutBucketEncryption 搭配 AWS SDK 使用](#)
 - [PutBucketPolicy 搭配 AWS SDK 使用](#)

- [PutObject 搭配 AWS SDK 使用](#)
- [UploadPart 搭配 AWS SDK 使用](#)
- [UploadPartCopy 搭配 AWS SDK 使用](#)
- [使用 AWS SDKs S3 目錄儲存貯體案例](#)
 - [為 Amazon S3 目錄儲存貯體建立預先簽章的 URL，以使用 AWS SDK 取得物件](#)

Amazon S3 AWS SDKs程式碼範例

下列程式碼範例示範如何使用 Amazon S3 搭配 AWS 軟體開發套件 (SDK)。

基本概念是程式碼範例，這些範例說明如何在服務內執行基本操作。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

案例是向您展示如何呼叫服務中的多個函數或與其他 AWS 服務組合來完成特定任務的程式碼範例。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Amazon S3

- [Amazon S3 AWS SDKs的基本範例](#)
 - [您好 Amazon S3](#)
 - [使用 AWS SDK 了解 Amazon S3 的基本概念](#)
 - [Amazon S3 AWS SDKs的動作](#)
 - [AbortMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
 - [CompleteMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
 - [CopyObject 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateBucket 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateMultiRegionAccessPoint 搭配 AWS SDK 使用](#)
 - [CreateMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
 - [CreatePresignedPost 搭配 AWS SDK 使用](#)
 - [DeleteBucket 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeleteBucketAnalyticsConfiguration 與 CLI](#)
 - [DeleteBucketCors 搭配 AWS SDK 或 CLI 使用](#)

- [搭配使用 DeleteBucketEncryption 與 CLI](#)
- [搭配使用 DeleteBucketInventoryConfiguration 與 CLI](#)
- [DeleteBucketLifecycle 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeleteBucketMetricsConfiguration 與 CLI](#)
- [DeleteBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeleteBucketReplication 與 CLI](#)
- [搭配使用 DeleteBucketTagging 與 CLI](#)
- [DeleteBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
- [DeleteObject 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeleteObjectTagging 與 CLI](#)
- [DeleteObjects 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeletePublicAccessBlock 與 CLI](#)
- [搭配使用 GetBucketAccelerateConfiguration 與 CLI](#)
- [GetBucketAcl 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketAnalyticsConfiguration 與 CLI](#)
- [GetBucketCors 搭配 AWS SDK 或 CLI 使用](#)
- [GetBucketEncryption 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketInventoryConfiguration 與 CLI](#)
- [GetBucketLifecycleConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [GetBucketLocation 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketLogging 與 CLI](#)
- [搭配使用 GetBucketMetricsConfiguration 與 CLI](#)
- [搭配使用 GetBucketNotification 與 CLI](#)
- [GetBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketPolicyStatus 與 CLI](#)
- [GetBucketReplication 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketRequestPayment 與 CLI](#)
- [搭配使用 GetBucketTagging 與 CLI](#)
- [搭配使用 GetBucketVersioning 與 CLI](#)
- [GetBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)

- [GetObject 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectAcl 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectAttributes 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectLegalHold 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectLockConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectRetention 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetObjectTagging 與 CLI](#)
- [搭配使用 GetPublicAccessBlock 與 CLI](#)
- [HeadBucket 搭配 AWS SDK 或 CLI 使用](#)
- [HeadObject 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 ListBucketAnalyticsConfigurations 與 CLI](#)
- [搭配使用 ListBucketInventoryConfigurations 與 CLI](#)
- [ListBuckets 搭配 AWS SDK 或 CLI 使用](#)
- [ListMultipartUploads 搭配 AWS SDK 或 CLI 使用](#)
- [ListObjectVersions 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 ListObjects 與 CLI](#)
- [ListObjectsV2 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketAccelerateConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketAcl 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketCors 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketEncryption 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketLifecycleConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketLogging 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 PutBucketNotification 與 CLI](#)
- [PutBucketNotificationConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketReplication 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 PutBucketRequestPayment 與 CLI](#)
- [搭配使用 PutBucketTagging 與 CLI](#)
- [PutBucketVersioning 搭配 AWS SDK 或 CLI 使用](#)

- [PutBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
- [PutObject 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectAcl 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectLegalHold 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectLockConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectRetention 搭配 AWS SDK 或 CLI 使用](#)
- [RestoreObject 搭配 AWS SDK 或 CLI 使用](#)
- [SelectObjectContent 搭配 AWS SDK 或 CLI 使用](#)
- [UploadPart 搭配 AWS SDK 或 CLI 使用](#)
- [UploadPartCopy 搭配 AWS SDK 或 CLI 使用](#)
- [Amazon S3 AWS SDKs 案例](#)
 - [檢查儲存貯體是否存在](#)
 - [檢查物件是否存在](#)
 - [使用 AWS SDK 將文字轉換為語音並返回文字](#)
 - [使用 AWS SDK 建立 Amazon S3 的預先簽章 URL](#)
 - [建立相片資產管理應用程式，讓使用者以標籤管理相片](#)
 - [使用 AWS SDK 列出 Amazon S3 物件的網頁](#)
 - [建立 Amazon Textract Explorer 應用程式](#)
 - [使用 AWS SDK 刪除指定 Amazon S3 儲存貯體中的所有物件](#)
 - [使用 AWS SDK 刪除不完整的分段上傳至 Amazon S3](#)
 - [使用 AWS SDK 透過 Amazon Rekognition 偵測映像中的個人防護裝備](#)
 - [使用 AWS SDK 偵測從映像擷取的文字中的實體](#)
 - [使用 AWS SDK 偵測映像中的臉部](#)
 - [使用 AWS SDK 透過 Amazon Rekognition 偵測映像中的物件](#)
 - [使用 AWS SDK 透過 Amazon Rekognition 偵測影片中的人物和物件](#)
 - [從 Amazon Simple Storage Service \(Amazon S3\) 儲存貯體下載 S3 'directories'](#)
 - [將 Amazon Simple Storage Service \(Amazon S3\) 儲存貯體中的所有物件下載至本機目錄](#)
 - [使用 AWS SDK 從 Amazon S3 物件下載未知大小的串流](#)
 - [使用 SDK 從多區域存取點取得 Amazon S3 物件 AWS](#)
- [使用 AWS SDK 從 Amazon S3 儲存貯體取得物件，指定 If-Modified-Since 標頭](#)

- [使用 AWS SDK 開始加密 Amazon S3 物件](#)
- [使用 AWS SDK 開始使用 Amazon S3 物件的標籤](#)
- [大數據處理叢集入門](#)
- [組態管理入門](#)
- [文件文字擷取入門](#)
- [機器學習功能存放區的入門](#)
- [物件儲存入門](#)
- [查詢分析入門](#)
- [使用 AWS SDK 使用 Amazon S3 物件鎖定功能](#)
- [使用 SDK 提出 Amazon S3 條件式請求 AWS](#)
- [使用 AWS SDK 管理 Amazon S3 儲存貯體的存取控制清單 \(ACLs\)](#)
- [透過 SDK 使用 Amazon S3 管理大型 Amazon SQS 訊息 Amazon S3 AWS](#)
- [使用 SDK 使用 Lambda 函數分批管理版本控制的 Amazon S3 物件 AWS](#)
- [使用 AWS SDK 剖析 Amazon S3 URIs](#)
- [使用 AWS SDK 執行 Amazon S3 物件的分段複製](#)
- [使用 SDK 接收和處理 Amazon S3 事件通知 AWS](#)
- [使用 AWS SDK 儲存 EXIF 和其他影像資訊](#)
- [使用 AWS SDK 將 S3 事件通知傳送至 Amazon EventBridge](#)
- [使用 SDK 追蹤 Amazon S3 物件上傳或下載 AWS](#)
- [使用 S3 Object Lambda 轉換應用程式的資料](#)
- [使用 AWS SDK 進行單元和整合測試的範例方法](#)
- [以遞迴的方式將本機目錄上傳至 Amazon Simple Storage Service \(Amazon S3\) 儲存貯體](#)
- [使用 AWS SDK 在 Amazon S3 之間上傳或下載大型檔案](#)
- [使用 AWS SDK 將大小不明的串流上傳至 Amazon S3 物件](#)
- [使用檢查總和來使用 AWS SDK 處理 Amazon S3 物件](#)
- [使用 AWS SDK 使用 Amazon S3 物件完整性功能](#)
- [使用 AWS SDK 處理 Amazon S3 版本控制的物件](#)
- [Amazon S3 的無伺服器範例](#)
 - [使用 Amazon S3 觸發條件調用 Lambda 函數](#)

Amazon S3 AWS SDKs的基本範例

下列程式碼範例示範如何搭配 AWS SDK 使用 Amazon Simple Storage Service 的基本功能。

範例

- [您好 Amazon S3](#)
- [使用 AWS SDK 了解 Amazon S3 的基本概念](#)
- [Amazon S3 AWS SDKs的動作](#)
 - [AbortMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
 - [CompleteMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
 - [CopyObject 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateBucket 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateMultiRegionAccessPoint 搭配 AWS SDK 使用](#)
 - [CreateMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
 - [CreatePresignedPost 搭配 AWS SDK 使用](#)
 - [DeleteBucket 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeleteBucketAnalyticsConfiguration 與 CLI](#)
 - [DeleteBucketCors 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeleteBucketEncryption 與 CLI](#)
 - [搭配使用 DeleteBucketInventoryConfiguration 與 CLI](#)
 - [DeleteBucketLifecycle 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeleteBucketMetricsConfiguration 與 CLI](#)
 - [DeleteBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeleteBucketReplication 與 CLI](#)
 - [搭配使用 DeleteBucketTagging 與 CLI](#)
 - [DeleteBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteObject 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeleteObjectTagging 與 CLI](#)
 - [DeleteObjects 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 DeletePublicAccessBlock 與 CLI](#)
 - [搭配使用 GetBucketAccelerateConfiguration 與 CLI](#)
 - [GetBucketAcl 搭配 AWS SDK 或 CLI 使用](#)

- [搭配使用 GetBucketAnalyticsConfiguration 與 CLI](#)
- [GetBucketCors 搭配 AWS SDK 或 CLI 使用](#)
- [GetBucketEncryption 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketInventoryConfiguration 與 CLI](#)
- [GetBucketLifecycleConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [GetBucketLocation 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketLogging 與 CLI](#)
- [搭配使用 GetBucketMetricsConfiguration 與 CLI](#)
- [搭配使用 GetBucketNotification 與 CLI](#)
- [GetBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketPolicyStatus 與 CLI](#)
- [GetBucketReplication 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketRequestPayment 與 CLI](#)
- [搭配使用 GetBucketTagging 與 CLI](#)
- [搭配使用 GetBucketVersioning 與 CLI](#)
- [GetBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
- [GetObject 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectAcl 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectAttributes 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectLegalHold 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectLockConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectRetention 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetObjectTagging 與 CLI](#)
- [搭配使用 GetPublicAccessBlock 與 CLI](#)
- [HeadBucket 搭配 AWS SDK 或 CLI 使用](#)
- [HeadObject 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 ListBucketAnalyticsConfigurations 與 CLI](#)
- [搭配使用 ListBucketInventoryConfigurations 與 CLI](#)
- [ListBuckets 搭配 AWS SDK 或 CLI 使用](#)
- [ListMultipartUploads 搭配 AWS SDK 或 CLI 使用](#)

- [ListObjectVersions 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 ListObjects 與 CLI](#)
- [ListObjectsV2 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketAccelerateConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketAcl 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketCors 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketEncryption 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketLifecycleConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketLogging 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 PutBucketNotification 與 CLI](#)
- [PutBucketNotificationConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketReplication 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 PutBucketRequestPayment 與 CLI](#)
- [搭配使用 PutBucketTagging 與 CLI](#)
- [PutBucketVersioning 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
- [PutObject 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectAcl 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectLegalHold 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectLockConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectRetention 搭配 AWS SDK 或 CLI 使用](#)
- [RestoreObject 搭配 AWS SDK 或 CLI 使用](#)
- [SelectObjectContent 搭配 AWS SDK 或 CLI 使用](#)
- [UploadPart 搭配 AWS SDK 或 CLI 使用](#)
- [UploadPartCopy 搭配 AWS SDK 或 CLI 使用](#)

您好 Amazon S3

下列程式碼範例示範如何開始使用 Amazon S3。

基本概念

.NET

適用於 .NET 的 SDK (v4)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Hello Amazon Simple Storage Service
/// (Amazon S3) example.
/// </summary>
public class HelloS3
{
    /// <summary>
    /// Main method to run the Hello S3 example.
    /// </summary>
    /// <param name="args">Command line arguments.</param>
    /// <returns>A Task object.</returns>
    public static async Task Main(string[] args)
    {
        var s3Client = new AmazonS3Client();

        try
        {
            Console.WriteLine("Hello Amazon S3! Let's list your buckets:");
            Console.WriteLine(new string('-', 80));

            // Use the built-in paginator to list buckets
            var request = new ListBucketsRequest();
            var paginator = s3Client.Paginators.ListBuckets(request);

            var buckets = new List<S3Bucket>();

            await foreach (var response in paginator.Responses)
            {
                buckets.AddRange(response.Buckets);
            }

            if (buckets.Any())
```

```
        {
            Console.WriteLine($"Found {buckets.Count} S3 buckets:");
            Console.WriteLine();

            foreach (var bucket in buckets)
            {
                Console.WriteLine($"- Bucket Name: {bucket.BucketName}");
                Console.WriteLine($"  Creation Date:
{bucket.CreationDate:yyyy-MM-dd HH:mm:ss UTC}");
                Console.WriteLine();
            }
        }
        else
        {
            Console.WriteLine("No S3 buckets found in your account.");
        }

        Console.WriteLine("Hello S3 completed successfully.");
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"S3 service error occurred: {ex.Message}");
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Couldn't list S3 buckets. Here's why:
{ex.Message}");
    }
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [ListBuckets](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

CMakeLists.txt CMake 檔案的程式碼。

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS s3)

# Set this project's name.
project("hello_s3")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
    libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
        "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
    for running and debugging.

    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
    may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
        ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
    hello_s3.cpp)

target_link_libraries(${PROJECT_NAME}
```



```
${AWSSDK_LINK_LIBRARIES})
```

hello_s3.cpp 來源檔案的程式碼。

```
#include <aws/core/Aws.h>
#include <aws/s3/S3Client.h>
#include <iostream>
#include <aws/core/auth/AWSCredentialsProviderChain.h>
using namespace Aws;
using namespace Aws::Auth;

/*
 * A "Hello S3" starter application which initializes an Amazon Simple Storage
 * Service (Amazon S3) client
 * and lists the Amazon S3 buckets in the selected region.
 *
 * main function
 *
 * Usage: 'hello_s3'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
    // options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        // You don't normally have to test that you are authenticated. But the
        // S3 service permits anonymous requests, thus the s3Client will return "success"
        // and 0 buckets even if you are unauthenticated, which can be confusing to a new
        // user.

        auto provider =
            Aws::MakeShared<DefaultAWSCredentialsProviderChain>("alloc-tag");
        auto creds = provider->GetAWSCredentials();
        if (creds.IsEmpty()) {
            std::cerr << "Failed authentication" << std::endl;
        }
    }
}
```

```

    }

    Aws::S3::S3Client s3Client(clientConfig);
    auto outcome = s3Client.ListBuckets();

    if (!outcome.IsSuccess()) {
        std::cerr << "Failed with error: " << outcome.GetError() <<
std::endl;
        result = 1;
    } else {
        std::cout << "Found " << outcome.GetResult().GetBuckets().size()
            << " buckets\n";
        for (auto &bucket: outcome.GetResult().GetBuckets()) {
            std::cout << bucket.GetName() << std::endl;
        }
    }
}

Aws::ShutdownAPI(options); // Should only be called once.
return result;
}

```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [ListBuckets](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

package main

import (
    "context"
    "errors"
    "fmt"

```

```
"github.com/aws/aws-sdk-go-v2/config"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/aws/smithy-go"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Storage Service
// (Amazon S3) client and list up to 10 buckets in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
    ctx := context.Background()
    sdkConfig, err := config.LoadDefaultConfig(ctx)
    if err != nil {
        fmt.Println("Couldn't load default configuration. Have you set up your AWS
account?")
        fmt.Println(err)
        return
    }
    s3Client := s3.NewFromConfig(sdkConfig)
    count := 10
    fmt.Printf("Let's list up to %v buckets for your account.\n", count)
    result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
    if err != nil {
        var ae smithy.APIError
        if errors.As(err, &ae) && ae.ErrorCode() == "AccessDenied" {
            fmt.Println("You don't have permission to list buckets for this account.")
        } else {
            fmt.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
        }
        return
    }
    if len(result.Buckets) == 0 {
        fmt.Println("You don't have any buckets!")
    } else {
        if count > len(result.Buckets) {
            count = len(result.Buckets)
        }
        for _, bucket := range result.Buckets[:count] {
            fmt.Printf("\t%v\n", *bucket.Name)
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [ListBuckets](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.Bucket;
import software.amazon.awssdk.services.s3.model.ListBucketsResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class HelloS3 {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        listBuckets(s3);
    }

    /**
```

```

    * Lists all the S3 buckets associated with the provided AWS S3 client.
    *
    * @param s3 the S3Client instance used to interact with the AWS S3 service
    */
    public static void listBuckets(S3Client s3) {
        try {
            ListBucketsResponse response = s3.listBuckets();
            List<Bucket> bucketList = response.buckets();
            bucketList.forEach(bucket -> {
                System.out.println("Bucket Name: " + bucket.name());
            });

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListBuckets](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

import {
    paginateListBuckets,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * List the S3 buckets in your configured AWS account.
 */
export const helloS3 = async () => {

```

```
// When no region or credentials are provided, the SDK will use the
// region and credentials from the local AWS config.
const client = new S3Client({});

try {
  /**
   * @type { import("@aws-sdk/client-s3").Bucket[] }
   */
  const buckets = [];

  for await (const page of paginateListBuckets({ client }, {})) {
    buckets.push(...page.Buckets);
  }
  console.log("Buckets: ");
  console.log(buckets.map((bucket) => bucket.Name).join("\n"));
  return buckets;
} catch (caught) {
  // ListBuckets does not throw any modeled errors. Any error caught
  // here will be something generic like `AccessDenied`.
  if (caught instanceof S3ServiceException) {
    console.error(`${caught.name}: ${caught.message}`);
  } else {
    // Something besides S3 failed.
    throw caught;
  }
}
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [ListBuckets](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
use Aws\S3\S3Client;
```

```
$client = new S3Client(['region' => 'us-west-2']);  
$results = $client->listBuckets();  
var_dump($results);
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [ListBuckets](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import boto3  
  
def hello_s3():  
    """  
    Use the AWS SDK for Python (Boto3) to create an Amazon Simple Storage Service  
    (Amazon S3) client and list the buckets in your account.  
    This example uses the default settings specified in your shared credentials  
    and config files.  
    """  
  
    # Create an S3 client.  
    s3_client = boto3.client("s3")  
  
    print("Hello, Amazon S3! Let's list your buckets:")  
  
    # Create a paginator for the list_buckets operation.  
    paginator = s3_client.get_paginator("list_buckets")  
  
    # Use the paginator to get a list of all buckets.  
    response_iterator = paginator.paginate(  
        PaginationConfig={  
            "PageSize": 50, # Adjust PageSize as needed.  
            "StartingToken": None,
```

```
    }
)

# Iterate through the pages of the response.
buckets_found = False
for page in response_iterator:
    if "Buckets" in page and page["Buckets"]:
        buckets_found = True
        for bucket in page["Buckets"]:
            print(f"\t{bucket['Name']}")

if not buckets_found:
    print("No buckets found!")

if __name__ == "__main__":
    hello_s3()
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [ListBuckets](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
# frozen_string_literal: true

# S3Manager is a class responsible for managing S3 operations
# such as listing all S3 buckets in the current AWS account.
class S3Manager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
```



```
end

# Lists and prints all S3 buckets in the current AWS account.
def list_buckets
  @logger.info('Here are the buckets in your account:')

  response = @client.list_buckets

  if response.buckets.empty?
    @logger.info("You don't have any S3 buckets yet.")
  else
    response.buckets.each do |bucket|
      @logger.info("- #{bucket.name}")
    end
  end
end

rescue Aws::Errors::ServiceError => e
  @logger.error("Encountered an error while listing buckets: #{e.message}")
end

end

if $PROGRAM_NAME == __FILE__
  s3_client = Aws::S3::Client.new
  manager = S3Manager.new(s3_client)
  manager.list_buckets
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [ListBuckets](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// S3 Hello World Example using the AWS SDK for Rust.
///
```

```

/// This example lists the objects in a bucket, uploads an object to that bucket,
/// and then retrieves the object and prints some S3 information about the
/// object.
/// This shows a number of S3 features, including how to use built-in paginators
/// for large data sets.
///
/// # Arguments
///
/// * `client` - an S3 client configured appropriately for the environment.
/// * `bucket` - the bucket name that the object will be uploaded to. Must be
///   present in the region the `client` is configured to use.
/// * `filename` - a reference to a path that will be read and uploaded to S3.
/// * `key` - the string key that the object will be uploaded as inside the
///   bucket.
async fn list_bucket_and_upload_object(
    client: &aws_sdk_s3::Client,
    bucket: &str,
    filepath: &Path,
    key: &str,
) -> Result<(), S3ExampleError> {
    // List the buckets in this account
    let mut objects = client
        .list_objects_v2()
        .bucket(bucket)
        .into_paginator()
        .send();

    println!("key\tetag\tlast_modified\tstorage_class");
    while let Some(Ok(object)) = objects.next().await {
        for item in object.contents() {
            println!(
                "{}\t{}\t{}\t{}",
                item.key().unwrap_or_default(),
                item.e_tag().unwrap_or_default(),
                item.last_modified()
                    .map(|lm| format!("{lm}"))
                    .unwrap_or_default(),
                item.storage_class()
                    .map(|sc| format!("{sc}"))
                    .unwrap_or_default()
            );
        }
    }
}

```

```

    // Prepare a ByteStream around the file, and upload the object using that
    ByteStream.
    let body = aws_sdk_s3::primitives::ByteStream::from_path(filepath)
        .await
        .map_err(|err| {
            S3ExampleError::new(format!(
                "Failed to create bytestream for {filepath:?} ({err:?})"
            ))
        })?;
    let resp = client
        .put_object()
        .bucket(bucket)
        .key(key)
        .body(body)
        .send()
        .await?;

    println!(
        "Upload success. Version: {:?}" ,
        resp.version_id()
        .expect("S3 Object upload missing version ID")
    );

    // Retrieve the just-uploaded object.
    let resp = client.get_object().bucket(bucket).key(key).send().await?;
    println!("etag: {}", resp.e_tag().unwrap_or("(missing)"));
    println!("version: {}", resp.version_id().unwrap_or("(missing)"));

    Ok(())
}

```

S3ExampleError 公用程式。

```

/// S3ExampleError provides a From<T: ProvideErrorMetadata> impl to extract
/// client-specific error details. This serves as a consistent backup to handling
/// specific service errors, depending on what is needed by the scenario.
/// It is used throughout the code examples for the AWS SDK for Rust.
#[derive(Debug)]
pub struct S3ExampleError(String);
impl S3ExampleError {
    pub fn new(value: impl Into<String>) -> Self {
        S3ExampleError(value.into())
    }
}

```

```

    }

    pub fn add_message(self, message: impl Into<String>) -> Self {
        S3ExampleError(format!("{}", message.into()), self.0))
    }
}

impl<T: aws_sdk_s3::error::ProvideErrorMetadata> From<T> for S3ExampleError {
    fn from(value: T) -> Self {
        S3ExampleError(format!(
            "{}: {}",
            value
                .code()
                .map(String::from)
                .unwrap_or("unknown code".into()),
            value
                .message()
                .map(String::from)
                .unwrap_or("missing reason".into()),
        ))
    }
}

impl std::error::Error for S3ExampleError {}

impl std::fmt::Display for S3ExampleError {
    fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result {
        write!(f, "{}", self.0)
    }
}
}

```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [ListBuckets](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 了解 Amazon S3 的基本概念

下列程式碼範例示範如何：

- 建立儲存貯體並上傳檔案到該儲存貯體。

- 從儲存貯體下載物件。
- 將物件複製至儲存貯體中的子文件夾。
- 列出儲存貯體中的物件。
- 刪除儲存貯體物件和該儲存貯體。

.NET

適用於 .NET 的 SDK (v4)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

執行展示 Amazon S3 功能的互動式情境。

```
public class S3_Basics
{
    public static bool IsInteractive = true;
    public static string BucketName = null!;
    public static string TempFilePath = null!;
    public static S3Wrapper _s3Wrapper = null!;
    public static ILogger<S3_Basics> _logger = null!;

    public static async Task Main(string[] args)
    {
        // Set up dependency injection for the Amazon service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureServices((_, services) =>
            {
                services.AddAWSService<IAmazonS3>()
                    .AddTransient<S3Wrapper>()
                    .AddLogging(builder => builder.AddConsole());
            })
            .Build();

        _logger = LoggerFactory.Create(builder => builder.AddConsole())
            .CreateLogger<S3_Basics>();

        _s3Wrapper = host.Services.GetRequiredService<S3Wrapper>();

        var sepBar = new string('-', 45);
```

```

        Console.WriteLine(sepBar);
        Console.WriteLine("Amazon Simple Storage Service (Amazon S3) basic");
        Console.WriteLine("procedures. This application will:");
        Console.WriteLine("\n\t1. Create a bucket");
        Console.WriteLine("\n\t2. Upload an object to the new bucket");
        Console.WriteLine("\n\t3. Copy the uploaded object to a folder in the
bucket");
        Console.WriteLine("\n\t4. List the items in the new bucket");
        Console.WriteLine("\n\t5. Delete all the items in the bucket");
        Console.WriteLine("\n\t6. Delete the bucket");
        Console.WriteLine(sepBar);

        await RunScenario(_s3Wrapper, _logger);

        Console.WriteLine(sepBar);
        Console.WriteLine("The Amazon S3 scenario has successfully completed.");
        Console.WriteLine(sepBar);
    }

    /// <summary>
    /// Run the S3 Basics scenario with injected dependencies.
    /// </summary>
    /// <param name="s3Wrapper">The S3 wrapper instance.</param>
    /// <param name="scenarioLogger">The logger instance.</param>
    /// <returns>A Task object.</returns>
    public static async Task RunScenario(S3Wrapper s3Wrapper, ILogger<S3_Basics>
scenarioLogger)
    {
        string bucketName = BucketName;
        string filePath = TempFilePath;
        string keyName = string.Empty;

        var sepBar = new string('-', 45);

        try
        {
            // Create a bucket.
            Console.WriteLine($"{sepBar}");
            Console.WriteLine("\nCreate a new Amazon S3 bucket.\n");
            Console.WriteLine(sepBar);

            if (IsInteractive)
            {

```

```
        Console.WriteLine("Please enter a name for the new bucket: ");
        bucketName = Console.ReadLine();
    }
    else
    {
        Console.WriteLine($"Using bucket name: {bucketName}");
    }

    var success = await s3Wrapper.CreateBucketAsync(bucketName);
    if (success)
    {
        Console.WriteLine($"Successfully created bucket: {bucketName}.
\n");
    }
    else
    {
        Console.WriteLine($"Could not create bucket: {bucketName}.\n");
    }

    Console.WriteLine(sepBar);
    Console.WriteLine("Upload a file to the new bucket.");
    Console.WriteLine(sepBar);

    if (IsInteractive)
    {
        // Get the local path and filename for the file to upload.
        while (string.IsNullOrEmpty(filePath))
        {
            Console.WriteLine("Please enter the path and filename of the file
to upload: ");
            filePath = Console.ReadLine();

            // Confirm that the file exists on the local computer.
            if (!File.Exists(filePath))
            {
                Console.WriteLine($"Couldn't find {filePath}. Try again.
\n");
                filePath = string.Empty;
            }
        }
    }
    else
    {
        // Use the public variable if set, otherwise create a temp file
```

```
        if (!string.IsNullOrEmpty(TempFilePath))
        {
            filePath = TempFilePath;
            Console.WriteLine($"Using provided test file: {filePath}");
        }
        else
        {
            // Create a temporary test file for non-interactive mode
            filePath = Path.GetTempFileName();
            var testContent = "This is a test file for S3 basics
scenario.\nGenerated on: " + DateTime.UtcNow.ToString("yyyy-MM-dd HH:mm:ss
UTC");

            await File.WriteAllTextAsync(filePath, testContent);
            Console.WriteLine($"Created temporary test file:
{filePath}");
        }
    }

    // Get the file name from the full path.
    keyName = Path.GetFileName(filePath);

    success = await s3Wrapper.UploadFileAsync(bucketName, keyName,
filePath);

    if (success)
    {
        Console.WriteLine($"Successfully uploaded {keyName} from
{filePath} to {bucketName}.\n");
    }
    else
    {
        Console.WriteLine($"Could not upload {keyName}.\n");
    }

    // Set up download path
    string downloadPath = string.Empty;

    if (IsInteractive)
    {
        // Now get a new location where we can save the file.
        while (string.IsNullOrEmpty(downloadPath))
        {
            // First get the path to which the file will be downloaded.
```



```
        Console.WriteLine("Please enter the path where the file will be
downloaded: ");
        downloadPath = Console.ReadLine();

        // Confirm that the file doesn't already exist on the local
computer.
        if (File.Exists($"{downloadPath}\\{keyName}"))
        {
            Console.WriteLine($"Sorry, the file already exists in
that location.\n");
            downloadPath = string.Empty;
        }
    }
    else
    {
        downloadPath = Path.GetTempPath();
        var downloadFile = Path.Combine(downloadPath, keyName);
        if (File.Exists(downloadFile))
        {
            File.Delete(downloadFile);
        }

        Console.WriteLine($"Using download path: {downloadPath}");
    }

    // Download an object from a bucket.
    success = await s3Wrapper.DownloadObjectFromBucketAsync(bucketName,
keyName, downloadPath);

    if (success)
    {
        Console.WriteLine($"Successfully downloaded {keyName}.\n");
    }
    else
    {
        Console.WriteLine($"Sorry, could not download {keyName}.\n");
    }

    // Copy the object to a different folder in the bucket.
    string folderName = string.Empty;

    if (IsInteractive)
    {
```

```
        while (string.IsNullOrEmpty(folderName))
        {
            Console.WriteLine("Please enter the name of the folder to copy
your object to: ");
            folderName = Console.ReadLine();
        }
    }
    else
    {
        folderName = "test-folder";
        Console.WriteLine($"Using folder name: {folderName}");
    }

    await s3Wrapper.CopyObjectInBucketAsync(bucketName, keyName,
folderName);

    // List the objects in the bucket.
    await s3Wrapper.ListBucketContentsAsync(bucketName);

    // Delete the contents of the bucket.
    if (IsInteractive)
    {
        Console.WriteLine("Press <Enter> when you are ready to delete the
bucket contents.");
        _ = Console.ReadLine();
    }

    var deleteContentsSuccess = await
s3Wrapper.DeleteBucketContentsAsync(bucketName);
    if (deleteContentsSuccess)
    {
        Console.WriteLine($"Successfully deleted contents of
{bucketName}.\n");
    }
    else
    {
        Console.WriteLine($"Sorry, could not delete contents of
{bucketName}.\n");
    }

    if (IsInteractive)
    {
        // Deleting the bucket too quickly after separately deleting its
contents can
```

```
        // cause an error that the bucket isn't empty. To delete contents
and bucket in one
        // operation, use AmazonS3Util.DeleteS3BucketWithObjectsAsync
        Console.WriteLine("Press <Enter> when you are ready to delete the
bucket.");
        _ = Console.ReadLine();
    }
    else
    {
        // Add a small delay for non-interactive mode to ensure objects
are fully deleted.
        Console.WriteLine("Waiting a moment for objects to be fully
deleted...");
        await Task.Delay(2000);
    }

    // Delete the bucket.
    var deleteSuccess = await s3Wrapper.DeleteBucketAsync(bucketName);
    if (deleteSuccess)
    {
        Console.WriteLine($"Successfully deleted {bucketName}.\n");
    }
    else
    {
        Console.WriteLine($"Sorry, could not delete {bucketName}.\n");
    }

    // Clean up temporary files in non-interactive mode
    if (!IsInteractive)
    {
        try
        {
            if (File.Exists(filePath))
            {
                File.Delete(filePath);
                Console.WriteLine("Cleaned up temporary test file.");
            }

            var downloadFile = Path.Combine(downloadPath, keyName);
            if (File.Exists(downloadFile))
            {
                File.Delete(downloadFile);
                Console.WriteLine("Cleaned up downloaded test file.");
            }
        }
    }
}
```

```
        }
        catch (Exception ex)
        {
            scenarioLogger.LogWarning(ex, "Failed to clean up temporary
files.");
        }
    }
    catch (Exception ex)
    {
        scenarioLogger.LogError(ex, "An error occurred during the S3 scenario
execution.");

        // Clean up on error - delete bucket if it exists
        try
        {
            if (!string.IsNullOrEmpty(bucketName))
            {
                await s3Wrapper.DeleteBucketContentsAsync(bucketName);
                await s3Wrapper.DeleteBucketAsync(bucketName);
            }
        }
        catch (Exception cleanupEx)
        {
            scenarioLogger.LogError(cleanupEx, "Error during cleanup.");
        }

        // Clean up temporary files in non-interactive mode
        if (!IsInteractive)
        {
            try
            {
                if (!string.IsNullOrEmpty(filePath) && File.Exists(filePath))
                {
                    File.Delete(filePath);
                }
            }
            catch (Exception fileCleanupEx)
            {
                scenarioLogger.LogWarning(fileCleanupEx, "Failed to clean up
temporary files during error handling.");
            }
        }
    }
}
```

```
        throw;  
    }  
}  
}
```

Amazon S3 SDK 方法的包裝函式類別。

```
using Amazon.S3;  
using Amazon.S3.Model;  
  
namespace S3_Actions;  
  
/// <summary>  
/// This class contains all of the methods for working with Amazon Simple  
/// Storage Service (Amazon S3) buckets.  
/// </summary>  
public class S3Wrapper  
{  
    private readonly IAmazonS3 _amazonS3;  
  
    /// <summary>  
    /// Initializes a new instance of the <see cref="S3Wrapper"/> class.  
    /// </summary>  
    /// <param name="amazonS3">An initialized Amazon S3 client object.</param>  
    public S3Wrapper(IAmazonS3 amazonS3)  
    {  
        _amazonS3 = amazonS3;  
    }  
  
    /// <summary>  
    /// Shows how to create a new Amazon S3 bucket.  
    /// </summary>  
    /// <param name="bucketName">The name of the bucket to create.</param>  
    /// <returns>A boolean value representing the success or failure of  
    /// the bucket creation process.</returns>  
    public async Task<bool> CreateBucketAsync(string bucketName)  
    {  
        try  
        {  
            var request = new PutBucketRequest
```

```
        {
            BucketName = bucketName,
            UseClientRegion = true,
        };

        var response = await _amazonS3.PutBucketAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error creating bucket: '{ex.Message}'");
        return false;
    }
}
```

```
/// <summary>
/// Shows how to upload a file from the local computer to an Amazon S3
/// bucket.
/// </summary>
/// <param name="bucketName">The Amazon S3 bucket to which the object
/// will be uploaded.</param>
/// <param name="objectName">The object to upload.</param>
/// <param name="filePath">The path, including file name, of the object
/// on the local computer to upload.</param>
/// <returns>A boolean value indicating the success or failure of the
/// upload procedure.</returns>
public async Task<bool> UploadFileAsync(
    string bucketName,
    string objectName,
    string filePath)
{
    try
    {
        var request = new PutObjectRequest
        {
            BucketName = bucketName,
            Key = objectName,
            FilePath = filePath,
        };

        var response = await _amazonS3.PutObjectAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch { }
}
```

```
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error uploading {objectName}: {ex.Message}");
        return false;
    }
}

/// <summary>
/// Shows how to download an object from an Amazon S3 bucket to the
/// local computer.
/// </summary>
/// <param name="bucketName">The name of the bucket where the object is
/// currently stored.</param>
/// <param name="objectName">The name of the object to download.</param>
/// <param name="filePath">The path, including filename, where the
/// downloaded object will be stored.</param>
/// <returns>A boolean value indicating the success or failure of the
/// download process.</returns>
public async Task<bool> DownloadObjectFromBucketAsync(
    string bucketName,
    string objectName,
    string filePath)
{
    var request = new GetObjectRequest
    {
        BucketName = bucketName,
        Key = objectName,
    };

    using GetObjectResponse response = await
        _amazonS3.GetObjectAsync(request);

    try
    {
        // Save object to local file
        await response.WriteResponseStreamToFileAsync($"{filePath}\
        \{objectName}", true, CancellationTokens.None);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {

```

```
        Console.WriteLine($"Error saving {objectName}: {ex.Message}");
        return false;
    }
}

/// <summary>
/// Copies an object in an Amazon S3 bucket to a folder within the
/// same bucket.
/// </summary>
/// <param name="bucketName">The name of the Amazon S3 bucket where the
/// object to copy is located.</param>
/// <param name="objectName">The object to be copied.</param>
/// <param name="folderName">The folder to which the object will
/// be copied.</param>
/// <returns>A boolean value that indicates the success or failure of
/// the copy operation.</returns>
public async Task<bool> CopyObjectInBucketAsync(
    string bucketName,
    string objectName,
    string folderName)
{
    try
    {
        var request = new CopyObjectRequest
        {
            SourceBucket = bucketName,
            SourceKey = objectName,
            DestinationBucket = bucketName,
            DestinationKey = $"{folderName}\\{objectName}",
        };
        var response = await _amazonS3.CopyObjectAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error copying object: '{ex.Message}'");
        return false;
    }
}
```



```
/// <summary>
/// Shows how to list the objects in an Amazon S3 bucket.
/// </summary>
/// <param name="bucketName">The name of the bucket for which to list.
/// <param name="printList">True to print out the list.
/// <returns>The collection of objects.</returns>
public async Task<List<S3Object>?> ListBucketContentsAsync(string bucketName,
bool printList = true)
{
    try
    {
        var request = new ListObjectsV2Request
        {
            BucketName = bucketName,
            MaxKeys = 5,
        };

        if (printList)
        {
            Console.WriteLine("-----");
            Console.WriteLine($"Listing the contents of {bucketName}:");
            Console.WriteLine("-----");
        }

        var listObjectsV2Paginator = _amazonS3.Paginators.ListObjectsV2(new
ListObjectsV2Request
        {
            BucketName = bucketName,
        });
        var s3objects = new List<S3Object>();
        await foreach (var response in listObjectsV2Paginator.Responses)
        {
            if (response.S3Objects != null)
            {
                s3objects.AddRange(response.S3Objects);
            }
        }

        if (printList)
        {
            Console.WriteLine($"Number of Objects: {s3objects.Count}");
            foreach (var entry in s3objects)
            {
                Console.WriteLine($"Key = {entry.Key} Size = {entry.Size}");
            }
        }
    }
}
```

```

        }
    }

    return s3objects;
}
catch (AmazonS3Exception ex)
{
    Console.WriteLine($"Error encountered on server.
Message: '{ex.Message}' getting list of objects.");
    return null;
}
}

/// <summary>
/// Delete all of the objects stored in an existing Amazon S3 bucket.
/// </summary>
/// <param name="bucketName">The name of the bucket from which the
/// contents will be deleted.</param>
/// <returns>A boolean value that represents the success or failure of
/// deleting all of the objects in the bucket.</returns>
public async Task<bool> DeleteBucketContentsAsync(string bucketName)
{
    // Iterate over the contents of the bucket and delete all objects.
    try
    {
        // Delete all objects in the bucket.
        var deleteList = await ListBucketContentsAsync(bucketName, false);
        if (deleteList != null && deleteList.Any())
        {
            await _amazonS3.DeleteObjectsAsync(new DeleteObjectsRequest()
            {
                BucketName = bucketName,
                Objects = deleteList.Select(o => new KeyVersion { Key =
o.Key }).ToList(),
            });
        }

        return true;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error deleting objects: {ex.Message}");
    }
}

```

```
        return false;
    }
}

/// <summary>
/// Shows how to delete an Amazon S3 bucket.
/// </summary>
/// <param name="bucketName">The name of the Amazon S3 bucket to delete.</
param>
/// <returns>A boolean value that represents the success or failure of
/// the delete operation.</returns>
public async Task<bool> DeleteBucketAsync(string bucketName)
{
    try
    {
        var request = new DeleteBucketRequest { BucketName = bucketName, };

        await _amazonS3.DeleteBucketAsync(request);
        return true;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error deleting bucket: {ex.Message}");
        return false;
    }
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的下列主題。

- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObjects](#)
- [GetObject](#)
- [ListObjectsV2](#)

- [PutObject](#)

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
#####
# function s3_getting_started
#
# This function creates, copies, and deletes S3 buckets and objects.
#
# Returns:
#     0 - If successful.
#     1 - If an error occurred.
#####
function s3_getting_started() {
{
    if [ "$BUCKET_OPERATIONS_SOURCED" != "True" ]; then
        cd bucket-lifecycle-operations || exit

        source ./bucket_operations.sh
        cd ..
    fi
}

echo_repeat "*" 88
echo "Welcome to the Amazon S3 getting started demo."
echo_repeat "*" 88
    echo "A unique bucket will be created by appending a Universally Unique
Identifier to a bucket name prefix."
    echo -n "Enter a prefix for the S3 bucket that will be used in this demo: "
    get_input
    bucket_name_prefix=$get_input_result
local bucket_name
bucket_name=$(generate_random_name "$bucket_name_prefix")
```

```
local region_code
region_code=$(aws configure get region)

if create_bucket -b "$bucket_name" -r "$region_code"; then
    echo "Created demo bucket named $bucket_name"
else
    errecho "The bucket failed to create. This demo will exit."
    return 1
fi

local file_name
while [ -z "$file_name" ]; do
    echo -n "Enter a file you want to upload to your bucket: "
    get_input
    file_name=$get_input_result

    if [ ! -f "$file_name" ]; then
        echo "Could not find file $file_name. Are you sure it exists?"
        file_name=""
    fi
done

local key
key="$(basename "$file_name")"

local result=0
if copy_file_to_bucket "$bucket_name" "$file_name" "$key"; then
    echo "Uploaded file $file_name into bucket $bucket_name with key $key."
else
    result=1
fi

local destination_file
destination_file="$file_name.download"
if yes_no_input "Would you like to download $key to the file $destination_file?
(y/n) "; then
    if download_object_from_bucket "$bucket_name" "$destination_file" "$key";
then
        echo "Downloaded $key in the bucket $bucket_name to the file
$destination_file."
    else
        result=1
    fi
fi
```

```
fi

if yes_no_input "Would you like to copy $key a new object key in your bucket?
(y/n) "; then
    local to_key
    to_key="demo/$key"
    if copy_item_in_bucket "$bucket_name" "$key" "$to_key"; then
        echo "Copied $key in the bucket $bucket_name to the $to_key."
    else
        result=1
    fi
fi

local bucket_items
bucket_items=$(list_items_in_bucket "$bucket_name")

# shellcheck disable=SC2181
if [[ $? -ne 0 ]]; then
    result=1
fi

echo "Your bucket contains the following items."
echo -e "Name\t\tSize"
echo "$bucket_items"

if yes_no_input "Delete the bucket, $bucket_name, as well as the objects in it?
(y/n) "; then
    bucket_items=$(echo "$bucket_items" | cut -f 1)

    if delete_items_in_bucket "$bucket_name" "$bucket_items"; then
        echo "The following items were deleted from the bucket $bucket_name"
        echo "$bucket_items"
    else
        result=1
    fi

    if delete_bucket "$bucket_name"; then
        echo "Deleted the bucket $bucket_name"
    else
        result=1
    fi
fi

return $result
```

```
}
```

此案例中使用的 Amazon S3 函數。

```
#####
# function create-bucket
#
# This function creates the specified bucket in the specified AWS Region, unless
# it already exists.
#
# Parameters:
#     -b bucket_name  -- The name of the bucket to create.
#     -r region_code  -- The code for an AWS Region in which to
#                        create the bucket.
#
# Returns:
#     The URL of the bucket that was created.
#     And:
#     0 - If successful.
#     1 - If it fails.
#####
function create_bucket() {
    local bucket_name region_code response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function create_bucket"
        echo "Creates an Amazon S3 bucket. You must supply a bucket name:"
        echo "  -b bucket_name    The name of the bucket. It must be globally
unique."
        echo "  [-r region_code]  The code for an AWS Region in which the bucket is
created."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "b:r:h" option; do
        case "${option}" in
            b) bucket_name="${OPTARG}" ;;
            r) region_code="${OPTARG}" ;;
            h)

```

```

        usage
        return 0
    ;;
\?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
esac
done

if [[ -z "$bucket_name" ]]; then
    errecho "ERROR: You must provide a bucket name with the -b parameter."
    usage
    return 1
fi

local bucket_config_arg
# A location constraint for "us-east-1" returns an error.
if [[ -n "$region_code" ]] && [[ "$region_code" != "us-east-1" ]]; then
    bucket_config_arg="--create-bucket-configuration LocationConstraint=
$region_code"
fi

iecho "Parameters:\n"
iecho "    Bucket name:  $bucket_name"
iecho "    Region code:  $region_code"
iecho ""

# If the bucket already exists, we don't want to try to create it.
if (bucket_exists "$bucket_name"); then
    errecho "ERROR: A bucket with that name already exists. Try again."
    return 1
fi

# shellcheck disable=SC2086
response=$(aws s3api create-bucket \
    --bucket "$bucket_name" \
    $bucket_config_arg)

# shellcheck disable=SC2181
if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports create-bucket operation failed.\n$response"
    return 1

```



```

    fi
}

#####
# function copy_file_to_bucket
#
# This function creates a file in the specified bucket.
#
# Parameters:
#     $1 - The name of the bucket to copy the file to.
#     $2 - The path and file name of the local file to copy to the bucket.
#     $3 - The key (name) to call the copy of the file in the bucket.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function copy_file_to_bucket() {
    local response bucket_name source_file destination_file_name
    bucket_name=$1
    source_file=$2
    destination_file_name=$3

    response=$(aws s3api put-object \
        --bucket "$bucket_name" \
        --body "$source_file" \
        --key "$destination_file_name")

    # shellcheck disable=SC2181
    if [[ ${?} -ne 0 ]]; then
        errecho "ERROR: AWS reports put-object operation failed.\n$response"
        return 1
    fi
}

#####
# function download_object_from_bucket
#
# This function downloads an object in a bucket to a file.
#
# Parameters:
#     $1 - The name of the bucket to download the object from.
#     $2 - The path and file name to store the downloaded bucket.
#     $3 - The key (name) of the object in the bucket.

```

```

#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function download_object_from_bucket() {
    local bucket_name=$1
    local destination_file_name=$2
    local object_name=$3
    local response

    response=$(aws s3api get-object \
        --bucket "$bucket_name" \
        --key "$object_name" \
        "$destination_file_name")

    # shellcheck disable=SC2181
    if [[ ${?} -ne 0 ]]; then
        errecho "ERROR: AWS reports put-object operation failed.\n$response"
        return 1
    fi
}

#####
# function copy_item_in_bucket
#
# This function creates a copy of the specified file in the same bucket.
#
# Parameters:
#     $1 - The name of the bucket to copy the file from and to.
#     $2 - The key of the source file to copy.
#     $3 - The key of the destination file.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function copy_item_in_bucket() {
    local bucket_name=$1
    local source_key=$2
    local destination_key=$3
    local response

    response=$(aws s3api copy-object \

```

```

    --bucket "$bucket_name" \
    --copy-source "$bucket_name/$source_key" \
    --key "$destination_key")

# shellcheck disable=SC2181
if [[ $? -ne 0 ]]; then
    errecho "ERROR: AWS reports s3api copy-object operation failed.\n$response"
    return 1
fi
}

#####
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#     $1 - The name of the bucket.
#
# Returns:
#     The list of files in text format.
#     And:
#     0 - If successful.
#     1 - If it fails.
#####
function list_items_in_bucket() {
    local bucket_name=$1
    local response

    response=$(aws s3api list-objects \
        --bucket "$bucket_name" \
        --output text \
        --query 'Contents[].{Key: Key, Size: Size}')

# shellcheck disable=SC2181
if [[ ${?} -eq 0 ]]; then
    echo "$response"
else
    errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
    return 1
fi
}

```

```
#####
# function delete_items_in_bucket
#
# This function deletes the specified list of keys from the specified bucket.
#
# Parameters:
#     $1 - The name of the bucket.
#     $2 - A list of keys in the bucket to delete.

# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function delete_items_in_bucket() {
    local bucket_name=$1
    local keys=$2
    local response

    # Create the JSON for the items to delete.
    local delete_items
    delete_items="{\"Objects\":["
    for key in $keys; do
        delete_items="$delete_items{\"Key\": \"$key\"},"
    done
    delete_items=${delete_items%?} # Remove the final comma.
    delete_items="$delete_items]"

    response=$(aws s3api delete-objects \
        --bucket "$bucket_name" \
        --delete "$delete_items")

    # shellcheck disable=SC2181
    if [[ $? -ne 0 ]]; then
        errecho "ERROR: AWS reports s3api delete-object operation failed.\n
$response"
        return 1
    fi
}

#####
# function delete_bucket
#
# This function deletes the specified bucket.
```

```
#
# Parameters:
#     $1 - The name of the bucket.

# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function delete_bucket() {
    local bucket_name=$1
    local response

    response=$(aws s3api delete-bucket \
        --bucket "$bucket_name")

    # shellcheck disable=SC2181
    if [[ $? -ne 0 ]]; then
        errecho "ERROR: AWS reports s3api delete-bucket failed.\n$response"
        return 1
    fi
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
 - [CopyObject](#)
 - [CreateBucket](#)
 - [DeleteBucket](#)
 - [DeleteObjects](#)
 - [GetObject](#)
 - [ListObjectsV2](#)
 - [PutObject](#)

C++

適用於 C++ 的開發套件

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
#include <iostream>
#include <aws/core/Aws.h>
#include <aws/s3/S3Client.h>
#include <aws/s3/model/CopyObjectRequest.h>
#include <aws/s3/model/CreateBucketRequest.h>
#include <aws/s3/model/DeleteBucketRequest.h>
#include <aws/s3/model/DeleteObjectRequest.h>
#include <aws/s3/model/GetObjectRequest.h>
#include <aws/s3/model/ListObjectsV2Request.h>
#include <aws/s3/model/PutObjectRequest.h>
#include <aws/s3/model/BucketLocationConstraint.h>
#include <aws/s3/model/CreateBucketConfiguration.h>
#include <aws/core/utils/UUID.h>
#include <aws/core/utils/StringUtils.h>
#include <aws/core/utils/memory/stl/AWSAllocator.h>
#include <fstream>
#include "s3_examples.h"

namespace AwsDoc {
    namespace S3 {

        //! Delete an S3 bucket.
        /*!
         * \param bucketName: The S3 bucket's name.
         * \param client: An S3 client.
         * \return bool: Function succeeded.
         */
        static bool
        deleteBucket(const Aws::String &bucketName, Aws::S3::S3Client &client);

        //! Delete an object in an S3 bucket.
        /*!
```

```

        \param bucketName: The S3 bucket's name.
        \param key: The key for the object in the S3 bucket.
        \param client: An S3 client.
        \return bool: Function succeeded.
    */
    static bool
    deleteObjectFromBucket(const Aws::String &bucketName, const Aws::String
&key,
                           Aws::S3::S3Client &client);
    }
}

//! Scenario to create, copy, and delete S3 buckets and objects.
/*!
    \param bucketNamePrefix: A prefix for a bucket name.
    \param uploadFilePath: Path to file to upload to an Amazon S3 bucket.
    \param saveFilePath: Path for saving a downloaded S3 object.
    \param clientConfig: Aws client configuration.
    \return bool: Function succeeded.
*/
bool AwsDoc::S3::S3_GettingStartedScenario(const Aws::String &bucketNamePrefix,
      const Aws::String &uploadFilePath,
      const Aws::String &saveFilePath,
      const Aws::Client::ClientConfiguration
&clientConfig) {

    Aws::S3::S3Client client(clientConfig);

    // Create a unique bucket name which is only temporary and will be deleted.
    // Format: <bucketNamePrefix> + "-" + lowercase UUID.
    Aws::String uuid = Aws::Utils::UUID::RandomUUID();
    Aws::String bucketName = bucketNamePrefix +
        Aws::Utils::StringUtils::ToLower(uuid.c_str());

    // 1. Create a bucket.
    {
        Aws::S3::Model::CreateBucketRequest request;
        request.SetBucket(bucketName);

        if (clientConfig.region != Aws::Region::US_EAST_1) {
            Aws::S3::Model::CreateBucketConfiguration createBucketConfiguration;
            createBucketConfiguration.WithLocationConstraint(

```

```

    Aws::S3::Model::BucketLocationConstraintMapper::GetBucketLocationConstraintForName(
        clientConfig.region));
    request.WithCreateBucketConfiguration(createBucketConfiguration);
}

    Aws::S3::Model::CreateBucketOutcome outcome =
client.CreateBucket(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: createBucket: " <<
            err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
        return false;
    } else {
        std::cout << "Created the bucket, '" << bucketName <<
            "', in the region, '" << clientConfig.region << "'." <<
std::endl;
    }
}

// 2. Upload a local file to the bucket.
Aws::String key = "key-for-test";
{
    Aws::S3::Model::PutObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(key);

    std::shared_ptr<Aws::FStream> input_data =
        Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
            uploadFilePath,
            std::ios_base::in |
            std::ios_base::binary);

    if (!input_data->is_open()) {
        std::cerr << "Error: unable to open file, '" << uploadFilePath <<
            "'." << std::endl;
        AwsDoc::S3::deleteBucket(bucketName, client);
        return false;
    }

    request.SetBody(input_data);

```



```

    Aws::S3::Model::PutObjectOutcome outcome =
        client.PutObject(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: putObject: " <<
            outcome.GetError().GetMessage() << std::endl;
        AwsDoc::S3::deleteObjectFromBucket(bucketName, key, client);
        AwsDoc::S3::deleteBucket(bucketName, client);
        return false;
    } else {
        std::cout << "Added the object with the key, '" << key
            << "', to the bucket, '"
            << bucketName << "'." << std::endl;
    }
}

// 3. Download the object to a local file.
{
    Aws::S3::Model::GetObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(key);

    Aws::S3::Model::GetObjectOutcome outcome =
        client.GetObject(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: getObject: " <<
            err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
    } else {
        std::cout << "Downloaded the object with the key, '" << key
            << "', in the bucket, '"
            << bucketName << "'." << std::endl;

        Aws::IOStream &ioStream = outcome.GetResultWithOwnership().
            GetBody();
        Aws::OFStream outStream(saveFilePath,
                                std::ios_base::out | std::ios_base::binary);
        if (!outStream.is_open()) {
            std::cout << "Error: unable to open file, '" << saveFilePath <<
                "'."
                << std::endl;

```

```

        } else {
            outputStream << ioStream.rdbuf();
            std::cout << "Wrote the downloaded object to the file '"
                      << saveFilePath << "'." << std::endl;
        }
    }
}

// 4. Copy the object to a different "folder" in the bucket.
Aws::String copiedToKey = "test-folder/" + key;
{
    Aws::S3::Model::CopyObjectRequest request;
    request.WithBucket(bucketName)
           .WithKey(copiedToKey)
           .WithCopySource(bucketName + "/" + key);

    Aws::S3::Model::CopyObjectOutcome outcome =
        client.CopyObject(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error: copyObject: " <<
                  outcome.GetError().GetMessage() << std::endl;
    } else {
        std::cout << "Copied the object with the key, '" << key
                  << "', to the key, '" << copiedToKey
                  << "', in the bucket, '" << bucketName << "'." << std::endl;
    }
}

// 5. List objects in the bucket.
{
    Aws::S3::Model::ListObjectsV2Request request;
    request.WithBucket(bucketName);

    Aws::String continuationToken;
    Aws::Vector<Aws::S3::Model::Object> allObjects;

    do {
        if (!continuationToken.empty()) {
            request.SetContinuationToken(continuationToken);
        }
        Aws::S3::Model::ListObjectsV2Outcome outcome = client.ListObjectsV2(
            request);

        if (!outcome.IsSuccess()) {

```

```

        std::cerr << "Error: ListObjects: " <<
            outcome.GetError().GetMessage() << std::endl;
        break;
    } else {
        Aws::Vector<Aws::S3::Model::Object> objects =
            outcome.GetResult().GetContents();
        allObjects.insert(allObjects.end(), objects.begin(),
objects.end());
        continuationToken = outcome.GetResult().GetContinuationToken();
    }
} while (!continuationToken.empty());

std::cout << allObjects.size() << " objects in the bucket, '" <<
bucketName
    << "':" << std::endl;

for (Aws::S3::Model::Object &object: allObjects) {
    std::cout << "    '" << object.GetKey() << "'" << std::endl;
}
}

// 6. Delete all objects in the bucket.
// All objects in the bucket must be deleted before deleting the bucket.
AwsDoc::S3::deleteObjectFromBucket(bucketName, copiedToKey, client);
AwsDoc::S3::deleteObjectFromBucket(bucketName, key, client);

// 7. Delete the bucket.
return AwsDoc::S3::deleteBucket(bucketName, client);
}

bool AwsDoc::S3::deleteObjectFromBucket(const Aws::String &bucketName,
                                        const Aws::String &key,
                                        Aws::S3::S3Client &client) {
    Aws::S3::Model::DeleteObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(key);

    Aws::S3::Model::DeleteObjectOutcome outcome =
        client.DeleteObject(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: deleteObject: " <<
            outcome.GetError().GetMessage() << std::endl;
    } else {

```

```
        std::cout << "Deleted the object with the key, '" << key
                    << "', from the bucket, '"
                    << bucketName << "'.'" << std::endl;
    }

    return outcome.IsSuccess();
}

bool
AwsDoc::S3::deleteBucket(const Aws::String &bucketName, Aws::S3::S3Client
&client) {
    Aws::S3::Model::DeleteBucketRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::DeleteBucketOutcome outcome =
        client.DeleteBucket(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: deleteBucket: " <<
                    err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
    } else {
        std::cout << "Deleted the bucket, '" << bucketName << "'.'" << std::endl;
    }
    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的下列主題。

- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObjects](#)
- [GetObject](#)
- [ListObjectsV2](#)
- [PutObject](#)

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

定義包裝案例所使用之儲存貯體和物件動作的結構。

```
import (  
    "bytes"  
    "context"  
    "errors"  
    "fmt"  
    "io"  
    "log"  
    "os"  
    "time"  
  
    "github.com/aws/aws-sdk-go-v2/aws"  
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"  
    "github.com/aws/aws-sdk-go-v2/service/s3"  
    "github.com/aws/aws-sdk-go-v2/service/s3/types"  
    "github.com/aws/smithy-go"  
)  
  
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)  
// actions  
// used in the examples.  
// It contains S3Client, an Amazon S3 service client that is used to perform  
// bucket  
// and object actions.  
type BucketBasics struct {  
    S3Client *s3.Client  
}  
  
// ListBuckets lists the buckets in the current account.
```

```

func (basics BucketBasics) ListBuckets(ctx context.Context) ([]types.Bucket,
error) {
    var err error
    var output *s3.ListBucketsOutput
    var buckets []types.Bucket
    bucketPaginator := s3.NewListBucketsPaginator(basics.S3Client,
    &s3.ListBucketsInput{})
    for bucketPaginator.HasMorePages() {
        output, err = bucketPaginator.NextPage(ctx)
        if err != nil {
            var apiErr smithy.APIError
            if errors.As(err, &apiErr) && apiErr.ErrorCode() == "AccessDenied" {
                fmt.Println("You don't have permission to list buckets for this account.")
                err = apiErr
            } else {
                log.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
            }
            break
        } else {
            buckets = append(buckets, output.Buckets...)
        }
    }
    return buckets, err
}

// BucketExists checks whether a bucket exists in the current account.
func (basics BucketBasics) BucketExists(ctx context.Context, bucketName string)
(bool, error) {
    _, err := basics.S3Client.HeadBucket(ctx, &s3.HeadBucketInput{
        Bucket: aws.String(bucketName),
    })
    exists := true
    if err != nil {
        var apiError smithy.APIError
        if errors.As(err, &apiError) {
            switch apiError.(type) {
            case *types.NotFound:
                log.Printf("Bucket %v is available.\n", bucketName)
                exists = false
                err = nil
            default:

```

```
    log.Printf("Either you don't have access to bucket %v or another error
occurred. "+
    "Here's what happened: %v\n", bucketName, err)
}
}
} else {
    log.Printf("Bucket %v exists and you already own it.", bucketName)
}

return exists, err
}

// CreateBucket creates a bucket with the specified name in the specified Region.
func (basics BucketBasics) CreateBucket(ctx context.Context, name string, region
string) error {
_, err := basics.S3Client.CreateBucket(ctx, &s3.CreateBucketInput{
    Bucket: aws.String(name),
    CreateBucketConfiguration: &types.CreateBucketConfiguration{
        LocationConstraint: types.BucketLocationConstraint(region),
    },
})
if err != nil {
    var owned *types.BucketAlreadyOwnedByYou
    var exists *types.BucketAlreadyExists
    if errors.As(err, &owned) {
        log.Printf("You already own bucket %s.\n", name)
        err = owned
    } else if errors.As(err, &exists) {
        log.Printf("Bucket %s already exists.\n", name)
        err = exists
    }
} else {
    err = s3.NewBucketExistsWaiter(basics.S3Client).Wait(
        ctx, &s3.HeadBucketInput{Bucket: aws.String(name)}, time.Minute)
    if err != nil {
        log.Printf("Failed attempt to wait for bucket %s to exist.\n", name)
    }
}
return err
}
```

```
// UploadFile reads from a file and puts the data into an object in a bucket.
func (basics BucketBasics) UploadFile(ctx context.Context, bucketName string,
    objectKey string, fileName string) error {
    file, err := os.Open(fileName)
    if err != nil {
        log.Printf("Couldn't open file %v to upload. Here's why: %v\n", fileName, err)
    } else {
        defer file.Close()
        _, err = basics.S3Client.PutObject(ctx, &s3.PutObjectInput{
            Bucket: aws.String(bucketName),
            Key:    aws.String(objectKey),
            Body:   file,
        })
        if err != nil {
            var apiErr smithy.APIError
            if errors.As(err, &apiErr) && apiErr.ErrorCode() == "EntityTooLarge" {
                log.Printf("Error while uploading object to %s. The object is too large.\n"+
                    "To upload objects larger than 5GB, use the S3 console (160GB max)\n"+
                    "or the multipart upload API (5TB max).", bucketName)
            } else {
                log.Printf("Couldn't upload file %v to %v:%v. Here's why: %v\n",
                    fileName, bucketName, objectKey, err)
            }
        } else {
            err = s3.NewObjectExistsWaiter(basics.S3Client).Wait(
                ctx, &s3.HeadObjectInput{Bucket: aws.String(bucketName), Key:
                    aws.String(objectKey)}, time.Minute)
            if err != nil {
                log.Printf("Failed attempt to wait for object %s to exist.\n", objectKey)
            }
        }
    }
    return err
}

// UploadLargeObject uses an upload manager to upload data to an object in a
// bucket.
// The upload manager breaks large data into parts and uploads the parts
// concurrently.
func (basics BucketBasics) UploadLargeObject(ctx context.Context, bucketName
    string, objectKey string, largeObject []byte) error {
```



```

largeBuffer := bytes.NewReader(largeObject)
var partMiBs int64 = 10
uploader := manager.NewUploader(basics.S3Client, func(u *manager.Uploader) {
    u.PartSize = partMiBs * 1024 * 1024
})
_, err := uploader.Upload(ctx, &s3.PutObjectInput{
    Bucket: aws.String(bucketName),
    Key:    aws.String(objectKey),
    Body:   largeBuffer,
})
if err != nil {
    var apiErr smithy.APIError
    if errors.As(err, &apiErr) && apiErr.ErrorCode() == "EntityTooLarge" {
        log.Printf("Error while uploading object to %s. The object is too large.\n"+
            "The maximum size for a multipart upload is 5TB.", bucketName)
    } else {
        log.Printf("Couldn't upload large object to %v:%v. Here's why: %v\n",
            bucketName, objectKey, err)
    }
} else {
    err = s3.NewObjectExistsWaiter(basics.S3Client).Wait(
        ctx, &s3.HeadObjectInput{Bucket: aws.String(bucketName), Key:
aws.String(objectKey)}, time.Minute)
    if err != nil {
        log.Printf("Failed attempt to wait for object %s to exist.\n", objectKey)
    }
}

return err
}

// DownloadFile gets an object from a bucket and stores it in a local file.
func (basics BucketBasics) DownloadFile(ctx context.Context, bucketName string,
    objectKey string, fileName string) error {
    result, err := basics.S3Client.GetObject(ctx, &s3.GetObjectInput{
        Bucket: aws.String(bucketName),
        Key:    aws.String(objectKey),
    })
    if err != nil {
        var noKey *types.NoSuchKey
        if errors.As(err, &noKey) {

```

```

    log.Printf("Can't get object %s from bucket %s. No such key exists.\n",
objectKey, bucketName)
    err = noKey
} else {
    log.Printf("Couldn't get object %v:%v. Here's why: %v\n", bucketName,
objectKey, err)
}
return err
}
defer result.Body.Close()
file, err := os.Create(fileName)
if err != nil {
    log.Printf("Couldn't create file %v. Here's why: %v\n", fileName, err)
    return err
}
defer file.Close()
body, err := io.ReadAll(result.Body)
if err != nil {
    log.Printf("Couldn't read object body from %v. Here's why: %v\n", objectKey,
err)
}
_, err = file.Write(body)
return err
}

// DownloadLargeObject uses a download manager to download an object from a
// bucket.
// The download manager gets the data in parts and writes them to a buffer until
// all of
// the data has been downloaded.
func (basics BucketBasics) DownloadLargeObject(ctx context.Context, bucketName
string, objectKey string) ([]byte, error) {
    var partMiBs int64 = 10
    downloader := manager.NewDownloader(basics.S3Client, func(d *manager.Downloader)
    {
        d.PartSize = partMiBs * 1024 * 1024
    })
    buffer := manager.NewWriteAtBuffer([]byte{})
    _, err := downloader.Download(ctx, buffer, &s3.GetObjectInput{
        Bucket: aws.String(bucketName),
        Key:    aws.String(objectKey),
    })
}

```

```
if err != nil {
    log.Printf("Couldn't download large object from %v:%v. Here's why: %v\n",
        bucketName, objectKey, err)
}
return buffer.Bytes(), err
}

// CopyToFolder copies an object in a bucket to a subfolder in the same bucket.
func (basics BucketBasics) CopyToFolder(ctx context.Context, bucketName string,
    objectKey string, folderName string) error {
    objectDest := fmt.Sprintf("%v/%v", folderName, objectKey)
    _, err := basics.S3Client.CopyObject(ctx, &s3.CopyObjectInput{
        Bucket:      aws.String(bucketName),
        CopySource:   aws.String(fmt.Sprintf("%v/%v", bucketName, objectKey)),
        Key:          aws.String(objectDest),
    })
    if err != nil {
        var notActive *types.ObjectNotInActiveTierError
        if errors.As(err, &notActive) {
            log.Printf("Couldn't copy object %s from %s because the object isn't in the
                active tier.\n",
                objectKey, bucketName)
            err = notActive
        }
    } else {
        err = s3.NewObjectExistsWaiter(basics.S3Client).Wait(
            ctx, &s3.HeadObjectInput{Bucket: aws.String(bucketName), Key:
                aws.String(objectDest)}, time.Minute)
        if err != nil {
            log.Printf("Failed attempt to wait for object %s to exist.\n", objectDest)
        }
    }
    return err
}

// CopyToBucket copies an object in a bucket to another bucket.
func (basics BucketBasics) CopyToBucket(ctx context.Context, sourceBucket string,
    destinationBucket string, objectKey string) error {
    _, err := basics.S3Client.CopyObject(ctx, &s3.CopyObjectInput{
        Bucket:      aws.String(destinationBucket),
```

```
CopySource: aws.String(fmt.Sprintf("%v/%v", sourceBucket, objectKey)),
Key:        aws.String(objectKey),
})
if err != nil {
    var notActive *types.ObjectNotInActiveTierError
    if errors.As(err, &notActive) {
        log.Printf("Couldn't copy object %s from %s because the object isn't in the
active tier.\n",
            objectKey, sourceBucket)
        err = notActive
    }
} else {
    err = s3.NewObjectExistsWaiter(basics.S3Client).Wait(
        ctx, &s3.HeadObjectInput{Bucket: aws.String(destinationBucket), Key:
aws.String(objectKey)}, time.Minute)
    if err != nil {
        log.Printf("Failed attempt to wait for object %s to exist.\n", objectKey)
    }
}
return err
}

// ListObjects lists the objects in a bucket.
func (basics BucketBasics) ListObjects(ctx context.Context, bucketName string)
([]types.Object, error) {
    var err error
    var output *s3.ListObjectsV2Output
    input := &s3.ListObjectsV2Input{
        Bucket: aws.String(bucketName),
    }
    var objects []types.Object
    objectPaginator := s3.NewListObjectsV2Paginator(basics.S3Client, input)
    for objectPaginator.HasMorePages() {
        output, err = objectPaginator.NextPage(ctx)
        if err != nil {
            var noBucket *types.NoSuchBucket
            if errors.As(err, &noBucket) {
                log.Printf("Bucket %s does not exist.\n", bucketName)
                err = noBucket
            }
            break
        } else {
```

```
    objects = append(objects, output.Contents...)
}
}
return objects, err
}

// DeleteObjects deletes a list of objects from a bucket.
func (basics BucketBasics) DeleteObjects(ctx context.Context, bucketName string,
    objectKeys []string) error {
    var objectIds []types.ObjectIdentifier
    for _, key := range objectKeys {
        objectIds = append(objectIds, types.ObjectIdentifier{Key: aws.String(key)})
    }
    output, err := basics.S3Client.DeleteObjects(ctx, &s3.DeleteObjectsInput{
        Bucket: aws.String(bucketName),
        Delete: &types.Delete{Objects: objectIds, Quiet: aws.Bool(true)},
    })
    if err != nil || len(output.Errors) > 0 {
        log.Printf("Error deleting objects from bucket %s.\n", bucketName)
        if err != nil {
            var noBucket *types.NoSuchBucket
            if errors.As(err, &noBucket) {
                log.Printf("Bucket %s does not exist.\n", bucketName)
                err = noBucket
            }
        }
        } else if len(output.Errors) > 0 {
            for _, outErr := range output.Errors {
                log.Printf("%s: %s\n", *outErr.Key, *outErr.Message)
            }
            err = fmt.Errorf("%s", *output.Errors[0].Message)
        }
    } else {
        for _, delObjs := range output.Deleted {
            err = s3.NewObjectNotExistsWaiter(basics.S3Client).Wait(
                ctx, &s3.HeadObjectInput{Bucket: aws.String(bucketName), Key: delObjs.Key},
                time.Minute)
            if err != nil {
                log.Printf("Failed attempt to wait for object %s to be deleted.\n",
                    *delObjs.Key)
            } else {
                log.Printf("Deleted %s.\n", *delObjs.Key)
            }
        }
    }
}
```

```

    }
}
return err
}

// DeleteBucket deletes a bucket. The bucket must be empty or an error is
// returned.
func (basics BucketBasics) DeleteBucket(ctx context.Context, bucketName string)
error {
    _, err := basics.S3Client.DeleteBucket(ctx, &s3.DeleteBucketInput{
        Bucket: aws.String(bucketName)})
    if err != nil {
        var noBucket *types.NoSuchBucket
        if errors.As(err, &noBucket) {
            log.Printf("Bucket %s does not exist.\n", bucketName)
            err = noBucket
        } else {
            log.Printf("Couldn't delete bucket %v. Here's why: %v\n", bucketName, err)
        }
    } else {
        err = s3.NewBucketNotExistsWaiter(basics.S3Client).Wait(
            ctx, &s3.HeadBucketInput{Bucket: aws.String(bucketName)}, time.Minute)
        if err != nil {
            log.Printf("Failed attempt to wait for bucket %s to be deleted.\n",
                bucketName)
        } else {
            log.Printf("Deleted %s.\n", bucketName)
        }
    }
    return err
}

```

執行互動式情境，示範如何使用 S3 儲存貯體和物件。

```

import (
    "context"
    "fmt"
    "log"

```

```
"os"
"strings"

"github.com/aws/aws-sdk-go-v2/aws"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
"github.com/awsdocs/aws-doc-sdk-examples/gov2/s3/actions"
)

// RunGetStartedScenario is an interactive example that shows you how to use
// Amazon
// Simple Storage Service (Amazon S3) to create an S3 bucket and use it to store
// objects.
//
// 1. Create a bucket.
// 2. Upload a local file to the bucket.
// 3. Download an object to a local file.
// 4. Copy an object to a different folder in the bucket.
// 5. List objects in the bucket.
// 6. Delete all objects in the bucket.
// 7. Delete the bucket.
//
// This example creates an Amazon S3 service client from the specified sdkConfig
// so that
// you can replace it with a mocked or stubbed config for unit testing.
//
// It uses a questioner from the `demotools` package to get input during the
// example.
// This package can be found in the `..\..\demotools` folder of this repo.
func RunGetStartedScenario(ctx context.Context, sdkConfig aws.Config, questioner
demotools.IQuestioner) {
    defer func() {
        if r := recover(); r != nil {
            log.Println("Something went wrong with the demo.")
            _, isMock := questioner.(*demotools.MockQuestioner)
            if isMock || questioner.AskBool("Do you want to see the full error message (y/
n)?", "y") {
                log.Println(r)
            }
        }
    }()

    log.Println(strings.Repeat("-", 88))
    log.Println("Welcome to the Amazon S3 getting started demo.")
}
```



```
if err != nil {
    panic(err)
}
log.Printf("Uploaded %v as %v.\n", smallFile, smallKey)
log.Println(strings.Repeat("-", 88))

log.Printf("Let's download %v to a file.", smallKey)
downloadFileName := questioner.Ask("Enter a name for the downloaded file:",
demotools.NotEmpty{})
err = bucketBasics.DownloadFile(ctx, bucketName, smallKey, downloadFileName)
if err != nil {
    panic(err)
}
log.Printf("File %v downloaded.", downloadFileName)
log.Println(strings.Repeat("-", 88))

log.Printf("Let's copy %v to a folder in the same bucket.", smallKey)
folderName := questioner.Ask("Enter a folder name: ", demotools.NotEmpty{})
err = bucketBasics.CopyToFolder(ctx, bucketName, smallKey, folderName)
if err != nil {
    panic(err)
}
log.Printf("Copied %v to %v/%v.\n", smallKey, folderName, smallKey)
log.Println(strings.Repeat("-", 88))

log.Println("Let's list the objects in your bucket.")
questioner.Ask("Press Enter when you're ready.")
objects, err := bucketBasics.ListObjects(ctx, bucketName)
if err != nil {
    panic(err)
}
log.Printf("Found %v objects.\n", len(objects))
var objKeys []string
for _, object := range objects {
    objKeys = append(objKeys, *object.Key)
    log.Printf("\t\t%v\n", *object.Key)
}
log.Println(strings.Repeat("-", 88))

if questioner.AskBool("Do you want to delete your bucket and all of its "+
"contents? (y/n)", "y") {
    log.Println("Deleting objects.")
    err = bucketBasics.DeleteObjects(ctx, bucketName, objKeys)
    if err != nil {
```

```
    panic(err)
}
log.Println("Deleting bucket.")
err = bucketBasics.DeleteBucket(ctx, bucketName)
if err != nil {
    panic(err)
}
log.Printf("Deleting downloaded file %v.\n", downloadFileName)
err = os.Remove(downloadFileName)
if err != nil {
    panic(err)
}
} else {
    log.Println("Okay. Don't forget to delete objects from your bucket to avoid charges.")
}
log.Println(strings.Repeat("-", 88))

log.Println("Thanks for watching!")
log.Println(strings.Repeat("-", 88))
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的下列主題。

- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObjects](#)
- [GetObject](#)
- [ListObjectsV2](#)
- [PutObject](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

案例範例。

```
import java.io.IOException;
import java.util.Scanner;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 *
 * This Java code example performs the following tasks:
 *
 * 1. Creates an Amazon S3 bucket.
 * 2. Uploads an object to the bucket.
 * 3. Downloads the object to another local file.
 * 4. Uploads an object using multipart upload.
 * 5. List all objects located in the Amazon S3 bucket.
 * 6. Copies the object to another Amazon S3 bucket.
 * 7. Copy the object to another Amazon S3 bucket using multi copy.
 * 8. Deletes the object from the Amazon S3 bucket.
 * 9. Deletes the Amazon S3 bucket.
 */
```

```

public class S3Scenario {

    public static Scanner scanner = new Scanner(System.in);
    static S3Actions s3Actions = new S3Actions();
    public static final String DASHES = new String(new char[80]).replace("\0",
"-");
    private static final Logger logger =
LoggerFactory.getLogger(S3Scenario.class);
    public static void main(String[] args) throws IOException {
        final String usage = ""
            Usage:
                <bucketName> <key> <objectPath> <savePath> <toBucket>

            Where:
                bucketName - The name of the S3 bucket.
                key - The unique identifier for the object stored in the S3
bucket.
                objectPath - The full file path of the object within the S3
bucket (e.g., "documents/reports/annual_report.pdf").
                savePath - The local file path where the object will be
downloaded and saved (e.g., "C:/Users/username/Downloads/annual_report.pdf").
                toBucket - The name of the S3 bucket to which the object will be
copied.

            "";

        if (args.length != 5) {
            logger.info(usage);
            return;
        }

        String bucketName = args[0];
        String key = args[1];
        String objectPath = args[2];
        String savePath = args[3];
        String toBucket = args[4];

        logger.info(DASHES);
        logger.info("Welcome to the Amazon Simple Storage Service (S3) example
scenario.");
        logger.info("")
            Amazon S3 is a highly scalable and durable object storage
            service provided by Amazon Web Services (AWS). It is designed to
store and retrieve

```

any amount of data, from anywhere on the web, at any time.

The `S3AsyncClient` interface in the AWS SDK for Java 2.x provides a set of methods to programmatically interact with the Amazon S3 (Simple Storage Service) service. This allows developers to automate the management and manipulation of S3 buckets and objects as part of their application deployment pipelines. With S3, teams can focus on building and deploying their applications without having to worry about the underlying storage infrastructure required to host and manage large amounts of data.

This scenario walks you through how to perform key operations for this service.

Let's get started...

```
""");
waitForInputToContinue(scanner);
logger.info(DASHES);

try {
    // Run the methods that belong to this scenario.
    runScenario(bucketName, key, objectPath, savePath, toBucket);

} catch (Throwable rt) {
    Throwable cause = rt.getCause();
    if (cause instanceof S3Exception kmsEx) {
        logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
    } else {
        logger.info("An unexpected error occurred: " + rt.getMessage());
    }
}

private static void runScenario(String bucketName, String key, String
objectPath, String savePath, String toBucket) throws Throwable {
    logger.info(DASHES);
    logger.info("1. Create an Amazon S3 bucket.");
    try {
        CompletableFuture<Void> future =
s3Actions.createBucketAsync(bucketName);
        future.join();
    }
```

```
        waitForInputToContinue(scanner);

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof S3Exception s3Ex) {
            logger.info("S3 error occurred: Error message: {}, Error code  
{}, s3Ex.getMessage(), s3Ex.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        throw cause;
    }

    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("2. Upload a local file to the Amazon S3 bucket.");
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<PutObjectResponse> future =
s3Actions.uploadLocalFileAsync(bucketName, key, objectPath);
        future.join();
        logger.info("File uploaded successfully to {}/{}", bucketName, key);

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof S3Exception s3Ex) {
            logger.info("S3 error occurred: Error message: {}, Error code  
{}, s3Ex.getMessage(), s3Ex.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        throw cause;
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("3. Download the object to another local file.");
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<Void> future =
s3Actions.getObjectBytesAsync(bucketName, key, savePath);
```

```
        future.join();
        logger.info("Successfully obtained bytes from S3 object and wrote to
file {}", savePath);

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof S3Exception s3Ex) {
            logger.info("S3 error occurred: Error message: {}, Error code
{}", s3Ex.getMessage(), s3Ex.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        throw cause;
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("4. Perform a multipart upload.");
    waitForInputToContinue(scanner);
    String multipartKey = "multiPartKey";
    try {
        // Call the multipartUpload method
        CompletableFuture<Void> future =
s3Actions.multipartUpload(bucketName, multipartKey);
        future.join();
        logger.info("Multipart upload completed successfully for bucket '{} '
and key '{}'", bucketName, multipartKey);

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof S3Exception s3Ex) {
            logger.info("S3 error occurred: Error message: {}, Error code
{}", s3Ex.getMessage(), s3Ex.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        throw cause;
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("5. List all objects located in the Amazon S3 bucket.");
```

```
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Void> future =
s3Actions.listAllObjectsAsync(bucketName);
            future.join();
            logger.info("Object listing completed successfully.");

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof S3Exception s3Ex) {
                logger.info("S3 error occurred: Error message: {}, Error code
{}", s3Ex.getMessage(), s3Ex.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            throw cause;
        }
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("6. Copy the object to another Amazon S3 bucket.");
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<String> future =
s3Actions.copyBucketObjectAsync(bucketName, key, toBucket);
            String result = future.join();
            logger.info("Copy operation result: {}", result);

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof S3Exception s3Ex) {
                logger.info("S3 error occurred: Error message: {}, Error code
{}", s3Ex.getMessage(), s3Ex.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            throw cause;
        }
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        logger.info(DASHES);
```



```
        logger.info("7. Copy the object to another Amazon S3 bucket using multi
copy.");
        waitForInputToContinue(scanner);

        try {
            CompletableFuture<String> future =
s3Actions.performMultiCopy(toBucket, bucketName, key);
            String result = future.join();
            logger.info("Copy operation result: {}", result);

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof S3Exception s3Ex) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", s3Ex.getMessage(), s3Ex.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
        }
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("8. Delete objects from the Amazon S3 bucket.");
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Void> future =
s3Actions.deleteObjectFromBucketAsync(bucketName, key);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof S3Exception s3Ex) {
                logger.info("S3 error occurred: Error message: {}, Error code
{}", s3Ex.getMessage(), s3Ex.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            throw cause;
        }
        try {
            CompletableFuture<Void> future =
s3Actions.deleteObjectFromBucketAsync(bucketName, "multiPartKey");
```

```
        future.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof S3Exception s3Ex) {
            logger.info("S3 error occurred: Error message: {}, Error code
{}", s3Ex.getMessage(), s3Ex.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        throw cause;
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("9. Delete the Amazon S3 bucket.");
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<Void> future =
s3Actions.deleteBucketAsync(bucketName);
        future.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof S3Exception s3Ex) {
            logger.info("S3 error occurred: Error message: {}, Error code
{}", s3Ex.getMessage(), s3Ex.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        throw cause;
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("You successfully completed the Amazon S3 scenario.");
    logger.info(DASHES);
}

private static void waitForInputToContinue(Scanner scanner) {
    while (true) {
        logger.info("");
    }
}
```

```

        logger.info("Enter 'c' followed by <ENTER> to continue:");
        String input = scanner.nextLine();

        if (input.trim().equalsIgnoreCase("c")) {
            logger.info("Continuing with the program...");
            logger.info("");
            break;
        } else {
            // Handle invalid input.
            logger.info("Invalid input. Please try again.");
        }
    }
}
}
}

```

包含操作的包裝函式類別。

```

public class S3Actions {

    private static final Logger logger =
        LoggerFactory.getLogger(S3Actions.class);
    private static S3AsyncClient s3AsyncClient;

    public static S3AsyncClient getAsyncClient() {
        if (s3AsyncClient == null) {
            /*
             * The `NettyNioAsyncHttpClient` class is part of the AWS SDK for Java,
             * version 2,
             * and it is designed to provide a high-performance, asynchronous HTTP
             * client for interacting with AWS services.
             * It uses the Netty framework to handle the underlying network
             * communication and the Java NIO API to
             * provide a non-blocking, event-driven approach to HTTP requests and
             * responses.
             */

            SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
                .maxConcurrency(50) // Adjust as needed.
                .connectionTimeout(Duration.ofSeconds(60)) // Set the connection
                timeout.

                .readTimeout(Duration.ofSeconds(60)) // Set the read timeout.
                .writeTimeout(Duration.ofSeconds(60)) // Set the write timeout.

```

```

        .build();

        ClientOverrideConfiguration overrideConfig =
ClientOverrideConfiguration.builder()
        .apiCallTimeout(Duration.ofMinutes(2)) // Set the overall API
call timeout.
        .apiCallAttemptTimeout(Duration.ofSeconds(90)) // Set the
individual call attempt timeout.
        .retryStrategy(RetryMode.STANDARD)
        .build();

        s3AsyncClient = S3AsyncClient.builder()
        .region(Region.US_EAST_1)
        .httpClient(httpClient)
        .overrideConfiguration(overrideConfig)
        .build();
    }
    return s3AsyncClient;
}

/**
 * Creates an S3 bucket asynchronously.
 *
 * @param bucketName the name of the S3 bucket to create
 * @return a {@link CompletableFuture} that completes when the bucket is
created and ready
 * @throws RuntimeException if there is a failure while creating the bucket
 */
public CompletableFuture<Void> createBucketAsync(String bucketName) {
    CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
        .bucket(bucketName)
        .build();

    CompletableFuture<CreateBucketResponse> response =
getAsyncClient().createBucket(bucketRequest);
    return response.thenCompose(resp -> {
        S3AsyncWaiter s3Waiter = getAsyncClient().waiter();
        HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
            .bucket(bucketName)
            .build();

        CompletableFuture<WaiterResponse<HeadBucketResponse>>
waiterResponseFuture =

```

```

        s3Waiter.waitUntilBucketExists(bucketRequestWait);
        return waiterResponseFuture.thenAccept(waiterResponse -> {
            waiterResponse.matched().response().ifPresent(headBucketResponse
-> {
                logger.info(bucketName + " is ready");
            });
        });
    }).whenComplete((resp, ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to create bucket", ex);
        }
    });
}

/**
 * Uploads a local file to an AWS S3 bucket asynchronously.
 *
 * @param bucketName the name of the S3 bucket to upload the file to
 * @param key         the key (object name) to use for the uploaded file
 * @param objectPath  the local file path of the file to be uploaded
 * @return a {@link CompletableFuture} that completes with the {@link
PutObjectResponse} when the upload is successful, or throws a {@link
RuntimeException} if the upload fails
 */
public CompletableFuture<PutObjectResponse> uploadLocalFileAsync(String
bucketName, String key, String objectPath) {
    PutObjectRequest objectRequest = PutObjectRequest.builder()
        .bucket(bucketName)
        .key(key)
        .build();

    CompletableFuture<PutObjectResponse> response =
getAsyncClient().putObject(objectRequest,
AsyncRequestBody.fromFile(Paths.get(objectPath)));
    return response.whenComplete((resp, ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to upload file", ex);
        }
    });
}

/**

```

```

    * Asynchronously retrieves the bytes of an object from an Amazon S3 bucket
    and writes them to a local file.
    *
    * @param bucketName the name of the S3 bucket containing the object
    * @param keyName    the key (or name) of the S3 object to retrieve
    * @param path        the local file path where the object's bytes will be
    written
    * @return a {@link CompletableFuture} that completes when the object bytes
    have been written to the local file
    */
    public CompletableFuture<Void> getObjectBytesAsync(String bucketName, String
    keyName, String path) {
        GetObjectRequest objectRequest = GetObjectRequest.builder()
            .key(keyName)
            .bucket(bucketName)
            .build();

        CompletableFuture<ResponseBytes<GetObjectResponse>> response =
        getAsyncClient().getObject(objectRequest, AsyncResponseTransformer.toBytes());
        return response.thenAccept(objectBytes -> {
            try {
                byte[] data = objectBytes.asByteArray();
                Path filePath = Paths.get(path);
                Files.write(filePath, data);
                logger.info("Successfully obtained bytes from an S3 object");
            } catch (IOException ex) {
                throw new RuntimeException("Failed to write data to file", ex);
            }
        }).whenComplete((resp, ex) -> {
            if (ex != null) {
                throw new RuntimeException("Failed to get object bytes from S3",
ex);
            }
        });
    }

    /**
    * Asynchronously lists all objects in the specified S3 bucket.
    *
    * @param bucketName the name of the S3 bucket to list objects for
    * @return a {@link CompletableFuture} that completes when all objects have
    been listed
    */

```

```

    public CompletableFuture<Void> listAllObjectsAsync(String bucketName) {
        ListObjectsV2Request initialRequest = ListObjectsV2Request.builder()
            .bucket(bucketName)
            .maxKeys(1)
            .build();

        ListObjectsV2Publisher paginator =
getAsyncClient().listObjectsV2Paginator(initialRequest);
        return paginator.subscribe(response -> {
            response.contents().forEach(s3Object -> {
                logger.info("Object key: " + s3Object.key());
            });
        }).thenRun(() -> {
            logger.info("Successfully listed all objects in the bucket: " +
bucketName);
        }).exceptionally(ex -> {
            throw new RuntimeException("Failed to list objects", ex);
        });
    }

    /**
     * Asynchronously copies an object from one S3 bucket to another.
     *
     * @param fromBucket the name of the source S3 bucket
     * @param objectKey the key (name) of the object to be copied
     * @param toBucket the name of the destination S3 bucket
     * @return a {@link CompletableFuture} that completes with the copy result as
a {@link String}
     * @throws RuntimeException if the URL could not be encoded or an S3
exception occurred during the copy
     */
    public CompletableFuture<String> copyBucketObjectAsync(String fromBucket,
String objectKey, String toBucket) {
        CopyObjectRequest copyReq = CopyObjectRequest.builder()
            .sourceBucket(fromBucket)
            .sourceKey(objectKey)
            .destinationBucket(toBucket)
            .destinationKey(objectKey)
            .build();

        CompletableFuture<CopyObjectResponse> response =
getAsyncClient().copyObject(copyReq);
        response.whenComplete((copyRes, ex) -> {

```

```

        if (copyRes != null) {
            logger.info("The " + objectKey + " was copied to " + toBucket);
        } else {
            throw new RuntimeException("An S3 exception occurred during
copy", ex);
        }
    });

    return response.thenApply(CopyObjectResponse::copyObjectResult)
        .thenApply(Object::toString);
}

/**
 * Performs a multipart upload to an Amazon S3 bucket.
 *
 * @param bucketName the name of the S3 bucket to upload the file to
 * @param key         the key (name) of the file to be uploaded
 * @return a {@link CompletableFuture} that completes when the multipart
upload is successful
 */
public CompletableFuture<Void> multipartUpload(String bucketName, String key)
{
    int mB = 1024 * 1024;

    CreateMultipartUploadRequest createMultipartUploadRequest =
CreateMultipartUploadRequest.builder()
        .bucket(bucketName)
        .key(key)
        .build();

    return
getAsyncClient().createMultipartUpload(createMultipartUploadRequest)
        .thenCompose(createResponse -> {
            String uploadId = createResponse.uploadId();
            System.out.println("Upload ID: " + uploadId);

            // Upload part 1.
            UploadPartRequest uploadPartRequest1 =
UploadPartRequest.builder()
                .bucket(bucketName)
                .key(key)
                .uploadId(uploadId)
                .partNumber(1)
                .contentType((long) (5 * mB)) // Specify the content length

```



```
        .build());

        CompletableFuture<CompletedPart> part1Future =
getAsyncClient().uploadPart(uploadPartRequest1,
        AsyncRequestBody.fromByteBuffer(getRandomByteBuffer(5 *
mB)))

        .thenApply(uploadPartResponse -> CompletedPart.builder()
        .partNumber(1)
        .eTag(uploadPartResponse.eTag())
        .build());

        // Upload part 2.
        UploadPartRequest uploadPartRequest2 =
UploadPartRequest.builder()
        .bucket(bucketName)
        .key(key)
        .uploadId(uploadId)
        .partNumber(2)
        .contentLength((long) (3 * mB))
        .build();

        CompletableFuture<CompletedPart> part2Future =
getAsyncClient().uploadPart(uploadPartRequest2,
        AsyncRequestBody.fromByteBuffer(getRandomByteBuffer(3 *
mB)))

        .thenApply(uploadPartResponse -> CompletedPart.builder()
        .partNumber(2)
        .eTag(uploadPartResponse.eTag())
        .build());

        // Combine the results of both parts.
        return CompletableFuture.allOf(part1Future, part2Future)
        .thenCompose(v -> {
            CompletedPart part1 = part1Future.join();
            CompletedPart part2 = part2Future.join();

            CompletedMultipartUpload completedMultipartUpload =
CompletedMultipartUpload.builder()
                .parts(part1, part2)
                .build();

            CompleteMultipartUploadRequest
completeMultipartUploadRequest = CompleteMultipartUploadRequest.builder()
                .bucket(bucketName)
```

```

        .key(key)
        .uploadId(uploadId)
        .multipartUpload(completedMultipartUpload)
        .build();

        // Complete the multipart upload
        return
getAsyncClient().completeMultipartUpload(completeMultipartUploadRequest);
    });
})
    .thenAccept(response -> System.out.println("Multipart upload
completed successfully"))
    .exceptionally(ex -> {
        System.err.println("Failed to complete multipart upload: " +
ex.getMessage());
        throw new RuntimeException(ex);
    });
}

/**
 * Deletes an object from an S3 bucket asynchronously.
 *
 * @param bucketName the name of the S3 bucket
 * @param key         the key (file name) of the object to be deleted
 * @return a {@link CompletableFuture} that completes when the object has
been deleted
 */
public CompletableFuture<Void> deleteObjectFromBucketAsync(String bucketName,
String key) {
    DeleteObjectRequest deleteObjectRequest = DeleteObjectRequest.builder()
        .bucket(bucketName)
        .key(key)
        .build();

    CompletableFuture<DeleteObjectResponse> response =
getAsyncClient().deleteObject(deleteObjectRequest);
    response.whenComplete((deleteRes, ex) -> {
        if (deleteRes != null) {
            logger.info(key + " was deleted");
        } else {
            throw new RuntimeException("An S3 exception occurred during
delete", ex);
        }
    })
}

```

```
    });

    return response.thenApply(r -> null);
}

/**
 * Deletes an S3 bucket asynchronously.
 *
 * @param bucket the name of the bucket to be deleted
 * @return a {@link CompletableFuture} that completes when the bucket
deletion is successful, or throws a {@link RuntimeException}
 * if an error occurs during the deletion process
 */
public CompletableFuture<Void> deleteBucketAsync(String bucket) {
    DeleteBucketRequest deleteBucketRequest = DeleteBucketRequest.builder()
        .bucket(bucket)
        .build();

    CompletableFuture<DeleteBucketResponse> response =
getAsyncClient().deleteBucket(deleteBucketRequest);
    response.whenComplete((deleteRes, ex) -> {
        if (deleteRes != null) {
            logger.info(bucket + " was deleted.");
        } else {
            throw new RuntimeException("An S3 exception occurred during
bucket deletion", ex);
        }
    });
    return response.thenApply(r -> null);
}

public CompletableFuture<String> performMultiCopy(String toBucket, String
bucketName, String key) {
    CreateMultipartUploadRequest createMultipartUploadRequest =
CreateMultipartUploadRequest.builder()
        .bucket(toBucket)
        .key(key)
        .build();

    getAsyncClient().createMultipartUpload(createMultipartUploadRequest)
        .thenApply(createMultipartUploadResponse -> {
            String uploadId = createMultipartUploadResponse.uploadId();
            System.out.println("Upload ID: " + uploadId);
        });
}
```

```

        UploadPartCopyRequest uploadPartCopyRequest =
UploadPartCopyRequest.builder()
    .sourceBucket(bucketName)
    .destinationBucket(toBucket)
    .sourceKey(key)
    .destinationKey(key)
    .uploadId(uploadId) // Use the valid uploadId.
    .partNumber(1) // Ensure the part number is correct.
    .copySourceRange("bytes=0-1023") // Adjust range as needed
    .build();

        return getAsyncClient().uploadPartCopy(uploadPartCopyRequest);
    })
    .thenCompose(uploadPartCopyFuture -> uploadPartCopyFuture)
    .whenComplete((uploadPartCopyResponse, exception) -> {
        if (exception != null) {
            // Handle any exceptions.
            logger.error("Error during upload part copy: " +
exception.getMessage());
        } else {
            // Successfully completed the upload part copy.
            System.out.println("Upload Part Copy completed successfully.
ETag: " + uploadPartCopyResponse.copyPartResult().eTag());
        }
    });
    return null;
}

private static ByteBuffer getRandomByteBuffer(int size) {
    ByteBuffer buffer = ByteBuffer.allocate(size);
    for (int i = 0; i < size; i++) {
        buffer.put((byte) (Math.random() * 256));
    }
    buffer.flip();
    return buffer;
}
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
- [CopyObject](#)
- [CreateBucket](#)

- [DeleteBucket](#)
- [DeleteObjects](#)
- [GetObject](#)
- [ListObjectsV2](#)
- [PutObject](#)

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

首先，匯入所有必要模組。

```
// Used to check if currently running file is this file.
import { fileURLToPath } from "node:url";
import { readdirSync, readFileSync, writeFileSync } from "node:fs";

// Local helper utils.
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import { Prompter } from "@aws-doc-sdk-examples/lib/prompter.js";
import { wrapText } from "@aws-doc-sdk-examples/lib/utils/util-string.js";

import {
  S3Client,
  CreateBucketCommand,
  PutObjectCommand,
  ListObjectsCommand,
  CopyObjectCommand,
  GetObjectCommand,
  DeleteObjectsCommand,
  DeleteBucketCommand,
} from "@aws-sdk/client-s3";
```

前面的匯入動作參考了一些協助公用程式。這些公用程式是本節開頭連結的 GitHub 儲存庫的本機公用程式。如需相關內容，請參閱下列公用程式的實作方式。

```
export const dirnameFromMetaUrl = (metaUrl) =>
  fileURLToPath(new URL(".", metaUrl));

import { select, input, confirm, checkbox, password } from "@inquirer/prompts";

export class Prompter {
  /**
   * @param {{ message: string, choices: { name: string, value: string }[] }}
  options
  */
  select(options) {
    return select(options);
  }

  /**
   * @param {{ message: string }} options
  */
  input(options) {
    return input(options);
  }

  /**
   * @param {{ message: string }} options
  */
  password(options) {
    return password({ ...options, mask: true });
  }

  /**
   * @param {string} prompt
  */
  checkContinue = async (prompt = "") => {
    const prefix = prompt && `${prompt} `;
    const ok = await this.confirm({
      message: `${prefix}Continue?`,
    });
    if (!ok) throw new Error("Exiting...");
  };
}
```

```

    * @param {{ message: string }} options
    */
    confirm(options) {
        return confirm(options);
    }

    /**
     * @param {{ message: string, choices: { name: string, value: string }[] }}
    options
     */
    checkbox(options) {
        return checkbox(options);
    }
}

export const wrapText = (text, char = "=") => {
    const rule = char.repeat(80);
    return `${rule}\n    ${text}\n${rule}\n`;
};

```

物件存放在「儲存貯體」中。接下來，來定義一個建立新儲存貯體的函數。

```

export const createBucket = async () => {
    const bucketName = await prompter.input({
        message: "Enter a bucket name. Bucket names must be globally unique:",
    });
    const command = new CreateBucketCommand({ Bucket: bucketName });
    await s3Client.send(command);
    console.log("Bucket created successfully.\n");
    return bucketName;
};

```

儲存貯體包含「物件」。此功能會將儲存庫的內容做為物件上傳至您的儲存貯體。

```

export const uploadFilesToBucket = async ({ bucketName, folderPath }) => {
    console.log(`Uploading files from ${folderPath}\n`);
    const keys = readdirSync(folderPath);
    const files = keys.map((key) => {
        const filePath = `${folderPath}/${key}`;
        const fileContent = readFileSync(filePath);
        return {

```

```
        Key: key,
        Body: fileContent,
      });
    });

    for (const file of files) {
      await s3Client.send(
        new PutObjectCommand({
          Bucket: bucketName,
          Body: file.Body,
          Key: file.Key,
        }),
      );
      console.log(`${file.Key} uploaded successfully.`);
    }
  };
```

上傳物件之後，請檢查確認已正確上傳物件。您可以為此使用 `ListObjects`。您將使用「金鑰」屬性，但在回應中也有其他有用的屬性。

```
export const listFilesInBucket = async ({ bucketName }) => {
  const command = new ListObjectsCommand({ Bucket: bucketName });
  const { Contents } = await s3Client.send(command);
  const contentsList = Contents.map((c) => ` • ${c.Key}`).join("\n");
  console.log("\nHere's a list of files in the bucket:");
  console.log(`${contentsList}\n`);
};
```

有時您可能想從儲存貯體複製物件到另一個儲存貯體。為此使用 `CopyObject` 命令。

```
export const copyFileFromBucket = async ({ destinationBucket }) => {
  const proceed = await prompter.confirm({
    message: "Would you like to copy an object from another bucket?",
  });

  if (!proceed) {
    return;
  }

  const copy = async () => {
    try {
      const sourceBucket = await prompter.input({
```



```
        message: "Enter source bucket name:",
    });
    const sourceKey = await prompter.input({
        message: "Enter source key:",
    });
    const destinationKey = await prompter.input({
        message: "Enter destination key:",
    });

    const command = new CopyObjectCommand({
        Bucket: destinationBucket,
        CopySource: `${sourceBucket}/${sourceKey}`,
        Key: destinationKey,
    });
    await s3Client.send(command);
    await copyFileFromBucket({ destinationBucket });
} catch (err) {
    console.error("Copy error.");
    console.error(err);
    const retryAnswer = await prompter.confirm({ message: "Try again?" });
    if (retryAnswer) {
        await copy();
    }
}
};
await copy();
};
```

沒有從儲存貯體中取得多個物件的 SDK 方法。反之，您會建立一份待下載的物件清單，並重複執行這些物件。

```
export const downloadFilesFromBucket = async ({ bucketName }) => {
    const { Contents } = await s3Client.send(
        new ListObjectsCommand({ Bucket: bucketName }),
    );
    const path = await prompter.input({
        message: "Enter destination path for files:",
    });

    for (const content of Contents) {
        const obj = await s3Client.send(
            new GetObjectCommand({ Bucket: bucketName, Key: content.Key }),
        );
    }
};
```

```

    );
    writeFileSync(
      `${path}/${content.Key}`,
      await obj.Body.transformToByteArray(),
    );
  }
  console.log("Files downloaded successfully.\n");
};

```

該清除資源了。儲存貯體在刪除之前必須先清空。這兩個函數會清空並刪除儲存貯體。

```

export const emptyBucket = async ({ bucketName }) => {
  const listObjectsCommand = new ListObjectsCommand({ Bucket: bucketName });
  const { Contents } = await s3Client.send(listObjectsCommand);
  const keys = Contents.map((c) => c.Key);

  const deleteObjectsCommand = new DeleteObjectsCommand({
    Bucket: bucketName,
    Delete: { Objects: keys.map((key) => ({ Key: key })) },
  });
  await s3Client.send(deleteObjectsCommand);
  console.log(`${bucketName} emptied successfully.\n`);
};

export const deleteBucket = async ({ bucketName }) => {
  const command = new DeleteBucketCommand({ Bucket: bucketName });
  await s3Client.send(command);
  console.log(`${bucketName} deleted successfully.\n`);
};

```

「主要」函數會將所有內容放在一起。若你直接執行這個檔案，主要函數將受到叫用。

```

const main = async () => {
  const OBJECT_DIRECTORY = `${dirnameFromMetaUrl(
    import.meta.url,
  )}../../resources/sample_files/.sample_media`;

  try {
    console.log(wrapText("Welcome to the Amazon S3 getting started example."));
    console.log("Let's create a bucket.");
    const bucketName = await createBucket();

```

```
    await prompter.confirm({ message: continueMessage });

    console.log(wrapText("File upload."));
    console.log(
      "I have some default files ready to go. You can edit the source code to  
provide your own.",
    );
    await uploadFilesToBucket({
      bucketName,
      folderPath: OBJECT_DIRECTORY,
    });

    await listFilesInBucket({ bucketName });
    await prompter.confirm({ message: continueMessage });

    console.log(wrapText("Copy files."));
    await copyFileFromBucket({ destinationBucket: bucketName });
    await listFilesInBucket({ bucketName });
    await prompter.confirm({ message: continueMessage });

    console.log(wrapText("Download files."));
    await downloadFilesFromBucket({ bucketName });

    console.log(wrapText("Clean up."));
    await emptyBucket({ bucketName });
    await deleteBucket({ bucketName });
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的下列主題。
 - [CopyObject](#)
 - [CreateBucket](#)
 - [DeleteBucket](#)
 - [DeleteObjects](#)
 - [GetObject](#)
 - [ListObjectsV2](#)
 - [PutObject](#)

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <bucketName> <key> <objectPath> <savePath> <toBucket>

    Where:
        bucketName - The Amazon S3 bucket to create.
        key - The key to use.
        objectPath - The path where the file is located (for example, C:/AWS/
book2.pdf).
        savePath - The path where the file is saved after it's downloaded (for
example, C:/AWS/book2.pdf).
        toBucket - An Amazon S3 bucket to where an object is copied to (for
example, C:/AWS/book2.pdf).
    """

    if (args.size != 4) {
        println(usage)
        exitProcess(1)
    }

    val bucketName = args[0]
    val key = args[1]
    val objectPath = args[2]
    val savePath = args[3]
    val toBucket = args[4]

    // Create an Amazon S3 bucket.
    createBucket(bucketName)

    // Update a local file to the Amazon S3 bucket.
    putObject(bucketName, key, objectPath)
```

```
// Download the object to another local file.
getObjectFromMrap(bucketName, key, savePath)

// List all objects located in the Amazon S3 bucket.
listBucketObs(bucketName)

// Copy the object to another Amazon S3 bucket
copyBucketOb(bucketName, key, toBucket)

// Delete the object from the Amazon S3 bucket.
deleteBucketObs(bucketName, key)

// Delete the Amazon S3 bucket.
deleteBucket(bucketName)
println("All Amazon S3 operations were successfully performed")
}

suspend fun createBucket(bucketName: String) {
    val request =
        CreateBucketRequest {
            bucket = bucketName
        }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        s3.createBucket(request)
        println("$bucketName is ready")
    }
}

suspend fun putObject(
    bucketName: String,
    objectKey: String,
    objectPath: String,
) {
    val metadataVal = mutableMapOf<String, String>()
    metadataVal["myVal"] = "test"

    val request =
        PutObjectRequest {
            bucket = bucketName
            key = objectKey
            metadata = metadataVal
            this.body = Paths.get(objectPath).asByteStream()
        }
}
```

```
    }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        val response = s3.putObject(request)
        println("Tag information is ${response.eTag}")
    }
}

suspend fun getObjectFromMrap(
    bucketName: String,
    keyName: String,
    path: String,
) {
    val request =
        GetObjectRequest {
            key = keyName
            bucket = bucketName
        }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        s3.getObject(request) { resp ->
            val myFile = File(path)
            resp.body?.writeToFile(myFile)
            println("Successfully read $keyName from $bucketName")
        }
    }
}

suspend fun listBucketObs(bucketName: String) {
    val request =
        ListObjectsRequest {
            bucket = bucketName
        }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->

        val response = s3.listObjects(request)
        response.contents?.forEach { myObject ->
            println("The name of the key is ${myObject.key}")
            println("The owner is ${myObject.owner}")
        }
    }
}
```

```
suspend fun copyBucketOb(
    fromBucket: String,
    objectKey: String,
    toBucket: String,
) {
    var encodedUrl = ""
    try {
        encodedUrl = URLEncoder.encode("$fromBucket/$objectKey",
StandardCharsets.UTF_8.toString())
    } catch (e: UnsupportedEncodingException) {
        println("URL could not be encoded: " + e.message)
    }

    val request =
        CopyObjectRequest {
            copySource = encodedUrl
            bucket = toBucket
            key = objectKey
        }
    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        s3.copyObject(request)
    }
}

suspend fun deleteBucketObs(
    bucketName: String,
    objectName: String,
) {
    val objectId =
        ObjectIdentifier {
            key = objectName
        }

    val delOb =
        Delete {
            objects = listOf(objectId)
        }

    val request =
        DeleteObjectsRequest {
            bucket = bucketName
            delete = delOb
        }
}
```

```
S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
    s3.deleteObjects(request)
    println("$objectName was deleted from $bucketName")
}

suspend fun deleteBucket(bucketName: String?) {
    val request =
        DeleteBucketRequest {
            bucket = bucketName
        }
    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        s3.deleteBucket(request)
        println("The $bucketName was successfully deleted!")
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的下列主題。

- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObjects](#)
- [GetObject](#)
- [ListObjectsV2](#)
- [PutObject](#)

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
echo("\n");
```



```
echo("-----\n");
print("Welcome to the Amazon S3 getting started demo using PHP!\n");
echo("-----\n");

$region = 'us-west-2';

$this->s3client = new S3Client([
    'region' => $region,
]);
/* Inline declaration example
$s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);
*/

$this->bucketName = "amzn-s3-demo-bucket-" . uniqid();

try {
    $this->s3client->createBucket([
        'Bucket' => $this->bucketName,
        'CreateBucketConfiguration' => ['LocationConstraint' => $region],
    ]);
    echo "Created bucket named: $this->bucketName \n";
} catch (Exception $exception) {
    echo "Failed to create bucket $this->bucketName with error: " .
    $exception->getMessage();
    exit("Please fix error with bucket creation before continuing.");
}

$fileName = __DIR__ . "/local-file-" . uniqid();
try {
    $this->s3client->putObject([
        'Bucket' => $this->bucketName,
        'Key' => $fileName,
        'SourceFile' => __DIR__ . '/testfile.txt'
    ]);
    echo "Uploaded $fileName to $this->bucketName.\n";
} catch (Exception $exception) {
    echo "Failed to upload $fileName with error: " . $exception-
    >getMessage();
    exit("Please fix error with file upload before continuing.");
}

try {
    $file = $this->s3client->getObject([
        'Bucket' => $this->bucketName,
```

```

        'Key' => $fileName,
    ]);
    $body = $file->get('Body');
    $body->rewind();
    echo "Downloaded the file and it begins with: {$body->read(26)}.\n";
} catch (Exception $exception) {
    echo "Failed to download $fileName from $this->bucketName with error:
" . $exception->getMessage();
    exit("Please fix error with file downloading before continuing.");
}

try {
    $folder = "copied-folder";
    $this->s3client->copyObject([
        'Bucket' => $this->bucketName,
        'CopySource' => "$this->bucketName/$fileName",
        'Key' => "$folder/$fileName-copy",
    ]);
    echo "Copied $fileName to $folder/$fileName-copy.\n";
} catch (Exception $exception) {
    echo "Failed to copy $fileName with error: " . $exception-
>getMessage();
    exit("Please fix error with object copying before continuing.");
}

try {
    $contents = $this->s3client->listObjectsV2([
        'Bucket' => $this->bucketName,
    ]);
    echo "The contents of your bucket are: \n";
    foreach ($contents['Contents'] as $content) {
        echo $content['Key'] . "\n";
    }
} catch (Exception $exception) {
    echo "Failed to list objects in $this->bucketName with error: " .
$exception->getMessage();
    exit("Please fix error with listing objects before continuing.");
}

try {
    $objects = [];
    foreach ($contents['Contents'] as $content) {
        $objects[] = [
            'Key' => $content['Key'],

```

```

        ];
    }
    $this->s3client->deleteObjects([
        'Bucket' => $this->bucketName,
        'Delete' => [
            'Objects' => $objects,
        ],
    ]);
    $check = $this->s3client->listObjectsV2([
        'Bucket' => $this->bucketName,
    ]);
    if (isset($check['Contents']) && count($check['Contents']) > 0) {
        throw new Exception("Bucket wasn't empty.");
    }
    echo "Deleted all objects and folders from $this->bucketName.\n";
} catch (Exception $exception) {
    echo "Failed to delete $fileName from $this->bucketName with error:
" . $exception->getMessage();
    exit("Please fix error with object deletion before continuing.");
}

try {
    $this->s3client->deleteBucket([
        'Bucket' => $this->bucketName,
    ]);
    echo "Deleted bucket $this->bucketName.\n";
} catch (Exception $exception) {
    echo "Failed to delete $this->bucketName with error: " . $exception-
>getMessage();
    exit("Please fix error with bucket deletion before continuing.");
}

echo "Successfully ran the Amazon S3 with PHP demo.\n";

```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的下列主題。

- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObjects](#)

- [GetObject](#)
- [ListObjectsV2](#)
- [PutObject](#)

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import io
import os
import uuid

import boto3
from boto3.s3.transfer import S3UploadFailedError
from botocore.exceptions import ClientError

def do_scenario(s3_resource):
    print("-" * 88)
    print("Welcome to the Amazon S3 getting started demo!")
    print("-" * 88)

    bucket_name = f"amzn-s3-demo-bucket-{uuid.uuid4()}"
    bucket = s3_resource.Bucket(bucket_name)
    try:
        bucket.create(
            CreateBucketConfiguration={
                "LocationConstraint": s3_resource.meta.client.meta.region_name
            }
        )
        print(f"Created demo bucket named {bucket.name}.")
    except ClientError as err:
        print(f"Tried and failed to create demo bucket {bucket_name}.")
        print(f"\t{err.response['Error']['Code']}: {err.response['Error']\n['Message']}")
```

```
    print(f"\nCan't continue the demo without a bucket!")
    return

file_name = None
while file_name is None:
    file_name = input("\nEnter a file you want to upload to your bucket: ")
    if not os.path.exists(file_name):
        print(f"Couldn't find file {file_name}. Are you sure it exists?")
        file_name = None

obj = bucket.Object(os.path.basename(file_name))
try:
    obj.upload_file(file_name)
    print(
        f"Uploaded file {file_name} into bucket {bucket.name} with key
{obj.key}."
    )
except S3UploadFailedError as err:
    print(f"Couldn't upload file {file_name} to {bucket.name}.")
    print(f"\t{err}")

answer = input(f"\nDo you want to download {obj.key} into memory (y/n)? ")
if answer.lower() == "y":
    data = io.BytesIO()
    try:
        obj.download_fileobj(data)
        data.seek(0)
        print(f"Got your object. Here are the first 20 bytes:\n")
        print(f"\t{data.read(20)}")
    except ClientError as err:
        print(f"Couldn't download {obj.key}.")
        print(
            f"\t{err.response['Error']['Code']}: {err.response['Error']
['Message']}"
        )

answer = input(
    f"\nDo you want to copy {obj.key} to a subfolder in your bucket (y/n)? "
)
if answer.lower() == "y":
    dest_obj = bucket.Object(f"demo-folder/{obj.key}")
    try:
        dest_obj.copy({"Bucket": bucket.name, "Key": obj.key})
        print(f"Copied {obj.key} to {dest_obj.key}.")
```

```

        except ClientError as err:
            print(f"Couldn't copy {obj.key} to {dest_obj.key}.")
            print(
                f"\t{err.response['Error']['Code']}: {err.response['Error']
['Message']}"
            )

    print("\nYour bucket contains the following objects:")
    try:
        for o in bucket.objects.all():
            print(f"\t{o.key}")
    except ClientError as err:
        print(f"Couldn't list the objects in bucket {bucket.name}.")
        print(f"\t{err.response['Error']['Code']}: {err.response['Error']
['Message']}")

    answer = input(
        "\nDo you want to delete all of the objects as well as the bucket (y/n)?
"
    )
    if answer.lower() == "y":
        try:
            bucket.objects.delete()
            bucket.delete()
            print(f"Emptied and deleted bucket {bucket.name}.\n")
        except ClientError as err:
            print(f"Couldn't empty and delete bucket {bucket.name}.")
            print(
                f"\t{err.response['Error']['Code']}: {err.response['Error']
['Message']}"
            )

    print("Thanks for watching!")
    print("-" * 88)

if __name__ == "__main__":
    do_scenario(boto3.resource("s3"))

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的下列主題。
 - [CopyObject](#)
 - [CreateBucket](#)

- [DeleteBucket](#)
- [DeleteObjects](#)
- [GetObject](#)
- [ListObjectsV2](#)
- [PutObject](#)

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-s3'

# Wraps the getting started scenario actions.
class ScenarioGettingStarted
  attr_reader :s3_resource

  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def initialize(s3_resource)
    @s3_resource = s3_resource
  end

  # Creates a bucket with a random name in the currently configured account and
  # AWS Region.
  #
  # @return [Aws::S3::Bucket] The newly created bucket.
  def create_bucket
    bucket = @s3_resource.create_bucket(
      bucket: "amzn-s3-demo-bucket-#{Random.uuid}",
      create_bucket_configuration: {
        location_constraint: 'us-east-1' # NOTE: only certain regions permitted
      }
    )
    puts("Created demo bucket named #{bucket.name}.")
  rescue Aws::Errors::ServiceError => e
  end
end
```

```
puts('Tried and failed to create demo bucket.')
puts("\t#{e.code}: #{e.message}")
puts("\nCan't continue the demo without a bucket!")
raise
else
  bucket
end

# Requests a file name from the user.
#
# @return The name of the file.
def create_file
  File.open('demo.txt', 'w') { |f| f.write('This is a demo file.') }
end

# Uploads a file to an Amazon S3 bucket.
#
# @param bucket [Aws::S3::Bucket] The bucket object representing the upload
destination
# @return [Aws::S3::Object] The Amazon S3 object that contains the uploaded
file.
def upload_file(bucket)
  File.open('demo.txt', 'w+') { |f| f.write('This is a demo file.') }
  s3_object = bucket.object(File.basename('demo.txt'))
  s3_object.upload_file('demo.txt')
  puts("Uploaded file demo.txt into bucket #{bucket.name} with key
#{s3_object.key}.")
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't upload file demo.txt to #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  else
    s3_object
  end

# Downloads an Amazon S3 object to a file.
#
# @param s3_object [Aws::S3::Object] The object to download.
def download_file(s3_object)
  puts("\nDo you want to download #{s3_object.key} to a local file (y/n)? ")
  answer = gets.chomp.downcase
  if answer == 'y'
    puts('Enter a name for the downloaded file: ')
    file_name = gets.chomp
```



```

        s3_object.download_file(file_name)
        puts("Object #{s3_object.key} successfully downloaded to #{file_name}.")
    end
rescue Aws::Errors::ServiceError => e
    puts("Couldn't download #{s3_object.key}.")
    puts("\t#{e.code}: #{e.message}")
    raise
end

# Copies an Amazon S3 object to a subfolder within the same bucket.
#
# @param source_object [Aws::S3::Object] The source object to copy.
# @return [Aws::S3::Object, nil] The destination object.
def copy_object(source_object)
    dest_object = nil
    puts("\nDo you want to copy #{source_object.key} to a subfolder in your
bucket (y/n)? ")
    answer = gets.chomp.downcase
    if answer == 'y'
        dest_object = source_object.bucket.object("demo-folder/
#{source_object.key}")
        dest_object.copy_from(source_object)
        puts("Copied #{source_object.key} to #{dest_object.key}.")
    end
rescue Aws::Errors::ServiceError => e
    puts("Couldn't copy #{source_object.key}.")
    puts("\t#{e.code}: #{e.message}")
    raise
else
    dest_object
end

# Lists the objects in an Amazon S3 bucket.
#
# @param bucket [Aws::S3::Bucket] The bucket to query.
def list_objects(bucket)
    puts("\nYour bucket contains the following objects:")
    bucket.objects.each do |obj|
        puts("\t#{obj.key}")
    end
rescue Aws::Errors::ServiceError => e
    puts("Couldn't list the objects in bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
end

```

```
end

# Deletes the objects in an Amazon S3 bucket and deletes the bucket.
#
# @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
def delete_bucket(bucket)
  puts("\nDo you want to delete all of the objects as well as the bucket (y/n)?")
  answer = gets.chomp.downcase
  if answer == 'y'
    bucket.objects.batch_delete!
    bucket.delete
    puts("Emptied and deleted bucket #{bucket.name}.\n")
  end
rescue Aws::Errors::ServiceError => e
  puts("Couldn't empty and delete bucket #{bucket.name}.")
  puts("\t#{e.code}: #{e.message}")
  raise
end
end

# Runs the Amazon S3 getting started scenario.
def run_scenario(scenario)
  puts('-' * 88)
  puts('Welcome to the Amazon S3 getting started demo!')
  puts('-' * 88)

  bucket = scenario.create_bucket
  s3_object = scenario.upload_file(bucket)
  scenario.download_file(s3_object)
  scenario.copy_object(s3_object)
  scenario.list_objects(bucket)
  scenario.delete_bucket(bucket)

  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError
  puts('Something went wrong with the demo!')
end

run_scenario(ScenarioGettingStarted.new(Aws::S3::Resource.new)) if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的下列主題。
 - [CopyObject](#)
 - [CreateBucket](#)
 - [DeleteBucket](#)
 - [DeleteObjects](#)
 - [GetObject](#)
 - [ListObjectsV2](#)
 - [PutObject](#)

Rust

適用於 Rust 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

案例中執行的二進制文件的程式碼。

```
#![allow(clippy::result_large_err)]

//! Purpose
//! Shows how to use the AWS SDK for Rust to get started using
//! Amazon Simple Storage Service (Amazon S3). Create a bucket, move objects
    into and out of it,
//! and delete all resources at the end of the demo.
//!
//! This example follows the steps in "Getting started with Amazon S3" in the
    Amazon S3
    user guide.
    - https://docs.aws.amazon.com/AmazonS3/latest/userguide/GetStartedWithS3.html

use aws_config::meta::region::RegionProviderChain;
use aws_sdk_s3::{config::Region, Client};
use s3_code_examples::error::S3ExampleError;
```

```
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), S3ExampleError> {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));
    let region = region_provider.region().await.unwrap();
    let shared_config =
aws_config::from_env().region(region_provider).load().await;
    let client = Client::new(&shared_config);
    let bucket_name = format!("amzn-s3-demo-bucket-{}", Uuid::new_v4());
    let file_name = "s3/testfile.txt".to_string();
    let key = "test file key name".to_string();
    let target_key = "target_key".to_string();

    if let Err(e) = run_s3_operations(region, client, bucket_name, file_name,
key, target_key).await
    {
        eprintln!("{:?}", e);
    };

    Ok(())
}

async fn run_s3_operations(
    region: Region,
    client: Client,
    bucket_name: String,
    file_name: String,
    key: String,
    target_key: String,
) -> Result<(), S3ExampleError> {
    s3_code_examples::create_bucket(&client, &bucket_name, &region).await?;
    let run_example: Result<(), S3ExampleError> = (async {
        s3_code_examples::upload_object(&client, &bucket_name, &file_name,
&key).await?;
        let _object = s3_code_examples::download_object(&client, &bucket_name,
&key).await;
        s3_code_examples::copy_object(&client, &bucket_name, &bucket_name, &key,
&target_key)
            .await?;
        s3_code_examples::list_objects(&client, &bucket_name).await?;
        s3_code_examples::clear_bucket(&client, &bucket_name).await?;
    }).await;
    Ok(())
}
```

```

    })
    .await;
    if let Err(err) = run_example {
        eprintln!("Failed to complete getting-started example: {err:?}");
    }
    s3_code_examples::delete_bucket(&client, &bucket_name).await?;

    Ok(())
}

```

案例使用的常見動作。

```

pub async fn create_bucket(
    client: &aws_sdk_s3::Client,
    bucket_name: &str,
    region: &aws_config::Region,
) -> Result<Option<aws_sdk_s3::operation::create_bucket::CreateBucketOutput>,
S3ExampleError> {
    let constraint =
aws_sdk_s3::types::BucketLocationConstraint::from(region.to_string().as_str());
    let cfg = aws_sdk_s3::types::CreateBucketConfiguration::builder()
        .location_constraint(constraint)
        .build();
    let create = client
        .create_bucket()
        .create_bucket_configuration(cfg)
        .bucket(bucket_name)
        .send()
        .await;

    // BucketAlreadyExists and BucketAlreadyOwnedByYou are not problems for this
    task.
    create.map(Some).or_else(|err| {
        if err
            .as_service_error()
            .map(|se| se.is_bucket_already_exists()) ||
se.is_bucket_already_owned_by_you()
            == Some(true)
        {
            Ok(None)
        } else {
            Err(S3ExampleError::from(err))
        }
    })
}

```

```
    }
  })
}

pub async fn upload_object(
  client: &aws_sdk_s3::Client,
  bucket_name: &str,
  file_name: &str,
  key: &str,
) -> Result<aws_sdk_s3::operation::put_object::PutObjectOutput, S3ExampleError> {
  let body =
    aws_sdk_s3::primitives::ByteStream::from_path(std::path::Path::new(file_name)).await;
  client
    .put_object()
    .bucket(bucket_name)
    .key(key)
    .body(body.unwrap())
    .send()
    .await
    .map_err(S3ExampleError::from)
}

pub async fn download_object(
  client: &aws_sdk_s3::Client,
  bucket_name: &str,
  key: &str,
) -> Result<aws_sdk_s3::operation::get_object::GetObjectOutput, S3ExampleError> {
  client
    .get_object()
    .bucket(bucket_name)
    .key(key)
    .send()
    .await
    .map_err(S3ExampleError::from)
}

/// Copy an object from one bucket to another.
pub async fn copy_object(
  client: &aws_sdk_s3::Client,
  source_bucket: &str,
  destination_bucket: &str,
  source_object: &str,
  destination_object: &str,
) -> Result<(), S3ExampleError> {
```

```

    let source_key = format!("{source_bucket}/{source_object}");
    let response = client
        .copy_object()
        .copy_source(&source_key)
        .bucket(destination_bucket)
        .key(destination_object)
        .send()
        .await?;

    println!(
        "Copied from {source_key} to {destination_bucket}/{destination_object}
with etag {}",
        response
            .copy_object_result
            .unwrap_or_else(||
aws_sdk_s3::types::CopyObjectResult::builder().build())
            .e_tag()
            .unwrap_or("missing")
    );
    Ok(())
}

pub async fn list_objects(client: &aws_sdk_s3::Client, bucket: &str) ->
Result<(), S3ExampleError> {
    let mut response = client
        .list_objects_v2()
        .bucket(bucket.to_owned())
        .max_keys(10) // In this example, go 10 at a time.
        .into_paginator()
        .send();

    while let Some(result) = response.next().await {
        match result {
            Ok(output) => {
                for object in output.contents() {
                    println!(" - {}", object.key().unwrap_or("Unknown"));
                }
            }
            Err(err) => {
                eprintln!("{err:?}")
            }
        }
    }
}

```

```

    Ok(())
}

/// Given a bucket, remove all objects in the bucket, and then ensure no objects
/// remain in the bucket.
pub async fn clear_bucket(
    client: &aws_sdk_s3::Client,
    bucket_name: &str,
) -> Result<Vec<String>, S3ExampleError> {
    let objects = client.list_objects_v2().bucket(bucket_name).send().await?;

    // delete_objects no longer needs to be mutable.
    let objects_to_delete: Vec<String> = objects
        .contents()
        .iter()
        .filter_map(|obj| obj.key())
        .map(String::from)
        .collect();

    if objects_to_delete.is_empty() {
        return Ok(vec![]);
    }

    let return_keys = objects_to_delete.clone();

    delete_objects(client, bucket_name, objects_to_delete).await?;

    let objects = client.list_objects_v2().bucket(bucket_name).send().await?;

    eprintln!("{objects:?}");

    match objects.key_count {
        Some(0) => Ok(return_keys),
        _ => Err(S3ExampleError::new(
            "There were still objects left in the bucket.",
        )),
    }
}

pub async fn delete_bucket(
    client: &aws_sdk_s3::Client,
    bucket_name: &str,
) -> Result<(), S3ExampleError> {
    let resp = client.delete_bucket().bucket(bucket_name).send().await;

```



```
match resp {
  Ok(_) => Ok(()),
  Err(err) => {
    if err
      .as_service_error()
      .and_then(aws_sdk_s3::error::ProvideErrorMetadata::code)
      == Some("NoSuchBucket")
    {
      Ok(())
    } else {
      Err(S3ExampleError::from(err))
    }
  }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的下列主題。

- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObjects](#)
- [GetObject](#)
- [ListObjectsV2](#)
- [PutObject](#)

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
DATA(lo_session) = /aws1/cl_rt_session_aws=>create( cv_pfl ).
DATA(lo_s3) = /aws1/cl_s3_factory=>create( lo_session ).
```

```

" Create an Amazon Simple Storage Service (Amazon S3) bucket. "
TRY.
    " determine our region from our session
    DATA(lv_region) = CONV /aws1/s3_bucketlocationcnstrnt( lo_session-
>get_region( ) ).
    DATA lo_constraint TYPE REF TO /aws1/cl_s3_createbucketconf.
    " When in the us-east-1 region, you must not specify a constraint
    " In all other regions, specify the region as the constraint
    IF lv_region = 'us-east-1'.
        CLEAR lo_constraint.
    ELSE.
        lo_constraint = NEW /aws1/cl_s3_createbucketconf( lv_region ).
    ENDIF.

    lo_s3->createbucket(
        iv_bucket = iv_bucket_name
        io_createbucketconfiguration = lo_constraint ).
    MESSAGE 'S3 bucket created.' TYPE 'I'.
CATCH /aws1/cx_s3_bucketalrdyexists.
    MESSAGE 'Bucket name already exists.' TYPE 'E'.
CATCH /aws1/cx_s3_bktalrdyownedbyyou.
    MESSAGE 'Bucket already exists and is owned by you.' TYPE 'E'.
ENDTRY.

"Upload an object to an S3 bucket."
TRY.
    "Get contents of file from application server."
    DATA lv_file_content TYPE xstring.
    OPEN DATASET iv_key FOR INPUT IN BINARY MODE.
    READ DATASET iv_key INTO lv_file_content.
    CLOSE DATASET iv_key.

    lo_s3->putobject(
        iv_bucket = iv_bucket_name
        iv_key = iv_key
        iv_body = lv_file_content ).
    MESSAGE 'Object uploaded to S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.

" Get an object from a bucket. "

```

```

TRY.
    DATA(lo_result) = lo_s3->getobject(
        iv_bucket = iv_bucket_name
        iv_key = iv_key ).
    DATA(lv_object_data) = lo_result->get_body( ).
    MESSAGE 'Object retrieved from S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.

" Copy an object to a subfolder in a bucket. "
TRY.
    lo_s3->copyobject(
        iv_bucket = iv_bucket_name
        iv_key = |{ iv_copy_to_folder }/{ iv_key }|
        iv_copysource = |{ iv_bucket_name }/{ iv_key }| ).
    MESSAGE 'Object copied to a subfolder.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.

" List objects in the bucket. "
TRY.
    DATA(lo_list) = lo_s3->listobjects(
        iv_bucket = iv_bucket_name ).
    MESSAGE 'Retrieved list of objects in S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
DATA text TYPE string VALUE 'Object List - '.
DATA lv_object_key TYPE /aws1/s3_objectkey.
LOOP AT lo_list->get_contents( ) INTO DATA(lo_object).
    lv_object_key = lo_object->get_key( ).
    CONCATENATE lv_object_key ', ' INTO text.
ENDLOOP.
MESSAGE text TYPE 'I'.

" Delete the objects in a bucket. "
TRY.
    lo_s3->deleteobject(

```

```
        iv_bucket = iv_bucket_name
        iv_key = iv_key ).
    lo_s3->deleteobject(
        iv_bucket = iv_bucket_name
        iv_key = |{ iv_copy_to_folder }/{ iv_key }| ).
    MESSAGE 'Objects deleted from S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.

" Delete the bucket. "
TRY.
    lo_s3->deletebucket(
        iv_bucket = iv_bucket_name ).
    MESSAGE 'Deleted S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 SAP ABAP 的 AWS SDK API 參考》中的下列主題。

- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObjects](#)
- [GetObject](#)
- [ListObjectsV2](#)
- [PutObject](#)

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import AWSS3

import Foundation
import AWSS3
import Smithy
import ClientRuntime

/// A class containing all the code that interacts with the AWS SDK for Swift.
public class ServiceHandler {
    let configuration: S3Client.S3ClientConfiguration
    let client: S3Client

    enum HandlerError: Error {
        case getObjectBody(String)
        case readGetObjectBody(String)
        case missingContents(String)
    }

    /// Initialize and return a new ``ServiceHandler`` object, which is used to
    drive the AWS calls
    /// used for the example.
    ///
    /// - Returns: A new ``ServiceHandler`` object, ready to be called to
    ///           execute AWS operations.
    public init() async throws {
        do {
            configuration = try await S3Client.S3ClientConfiguration()
            // configuration.region = "us-east-2" // Uncomment this to set the
            region programmatically.
            client = S3Client(config: configuration)
        }
        catch {
            print("ERROR: ", dump(error, name: "Initializing S3 client"))
            throw error
        }
    }

    /// Create a new user given the specified name.
    ///
    /// - Parameters:
    ///   - name: Name of the bucket to create.
    ///   Throws an exception if an error occurs.
```

```
public func createBucket(name: String) async throws {
    var input = CreateBucketInput(
        bucket: name
    )

    // For regions other than "us-east-1", you must set the
    locationConstraint in the createBucketConfiguration.
    // For more information, see LocationConstraint in the S3 API guide.
    // https://docs.aws.amazon.com/AmazonS3/latest/API/
API_CreateBucket.html#API_CreateBucket_RequestBody
    if let region = configuration.region {
        if region != "us-east-1" {
            input.createBucketConfiguration =
S3ClientTypes.CreateBucketConfiguration(locationConstraint:
S3ClientTypes.BucketLocationConstraint(rawValue: region))
        }
    }

    do {
        _ = try await client.createBucket(input: input)
    }
    catch let error as BucketAlreadyOwnedByYou {
        print("The bucket '\(name)' already exists and is owned by you. You
may wish to ignore this exception.")
        throw error
    }
    catch {
        print("ERROR: ", dump(error, name: "Creating a bucket"))
        throw error
    }
}

/// Delete a bucket.
/// - Parameter name: Name of the bucket to delete.
public func deleteBucket(name: String) async throws {
    let input = DeleteBucketInput(
        bucket: name
    )
    do {
        _ = try await client.deleteBucket(input: input)
    }
    catch {
        print("ERROR: ", dump(error, name: "Deleting a bucket"))
        throw error
    }
}
```

```
    }  
  }  
  
  /// Upload a file from local storage to the bucket.  
  /// - Parameters:  
  ///   - bucket: Name of the bucket to upload the file to.  
  ///   - key: Name of the file to create.  
  ///   - file: Path name of the file to upload.  
  public func uploadFile(bucket: String, key: String, file: String) async  
throws {  
    let fileUrl = URL(fileURLWithPath: file)  
    do {  
      let fileData = try Data(contentsOf: fileUrl)  
      let dataStream = ByteStream.data(fileData)  
  
      let input = PutObjectInput(  
        body: dataStream,  
        bucket: bucket,  
        key: key  
      )  
  
      _ = try await client.putObject(input: input)  
    }  
    catch {  
      print("ERROR: ", dump(error, name: "Putting an object."))  
      throw error  
    }  
  }  
  
  /// Create a file in the specified bucket with the given name. The new  
  /// file's contents are uploaded from a `Data` object.  
  ///  
  /// - Parameters:  
  ///   - bucket: Name of the bucket to create a file in.  
  ///   - key: Name of the file to create.  
  ///   - data: A `Data` object to write into the new file.  
  public func createFile(bucket: String, key: String, withData data: Data)  
async throws {  
    let dataStream = ByteStream.data(data)  
  
    let input = PutObjectInput(  
      body: dataStream,  
      bucket: bucket,  
      key: key
```

```
    )

    do {
        _ = try await client.putObject(input: input)
    }
    catch {
        print("ERROR: ", dump(error, name: "Putting an object. "))
        throw error
    }
}

/// Download the named file to the given directory on the local device.
///
/// - Parameters:
///   - bucket: Name of the bucket that contains the file to be copied.
///   - key: The name of the file to copy from the bucket.
///   - to: The path of the directory on the local device where you want to
///         download the file.
public func downloadFile(bucket: String, key: String, to: String) async
throws {
    let fileUrl = URL(fileURLWithPath: to).appendingPathComponent(key)

    let input = GetObjectInput(
        bucket: bucket,
        key: key
    )
    do {
        let output = try await client.getObject(input: input)

        guard let body = output.body else {
            throw HandlerError.getObjectBody("GetObjectInput missing body.")
        }

        guard let data = try await body.readData() else {
            throw HandlerError.readGetObjectBody("GetObjectInput unable to
read data.")
        }

        try data.write(to: fileUrl)
    }
    catch {
        print("ERROR: ", dump(error, name: "Downloading a file. "))
        throw error
    }
}
```



```
}

/// Read the specified file from the given S3 bucket into a Swift
/// `Data` object.
///
/// - Parameters:
///   - bucket: Name of the bucket containing the file to read.
///   - key: Name of the file within the bucket to read.
///
/// - Returns: A `Data` object containing the complete file data.
public func readFile(bucket: String, key: String) async throws -> Data {
    let input = GetObjectInput(
        bucket: bucket,
        key: key
    )
    do {
        let output = try await client.getObject(input: input)

        guard let body = output.body else {
            throw HandlerError.getObjectBody("GetObjectInput missing body.")
        }

        guard let data = try await body.readData() else {
            throw HandlerError.readGetObjectBody("GetObjectInput unable to
read data.")
        }

        return data
    }
    catch {
        print("ERROR: ", dump(error, name: "Reading a file. "))
        throw error
    }
}

/// Copy a file from one bucket to another.
///
/// - Parameters:
///   - sourceBucket: Name of the bucket containing the source file.
///   - name: Name of the source file.
///   - destBucket: Name of the bucket to copy the file into.
public func copyFile(from sourceBucket: String, name: String, to destBucket:
String) async throws {
```

```
        let srcUrl = ("\"(sourceBucket)/
\\(name)").addingPercentEncoding(withAllowedCharacters: .urlPathAllowed)

        let input = CopyObjectInput(
            bucket: destBucket,
            copySource: srcUrl,
            key: name
        )
        do {
            _ = try await client.copyObject(input: input)
        }
        catch {
            print("ERROR: ", dump(error, name: "Copying an object. "))
            throw error
        }
    }

    /// Deletes the specified file from Amazon S3.
    ///
    /// - Parameters:
    ///   - bucket: Name of the bucket containing the file to delete.
    ///   - key: Name of the file to delete.
    ///
    public func deleteFile(bucket: String, key: String) async throws {
        let input = DeleteObjectInput(
            bucket: bucket,
            key: key
        )

        do {
            _ = try await client.deleteObject(input: input)
        }
        catch {
            print("ERROR: ", dump(error, name: "Deleting a file. "))
            throw error
        }
    }

    /// Returns an array of strings, each naming one file in the
    /// specified bucket.
    ///
    /// - Parameter bucket: Name of the bucket to get a file listing for.
    /// - Returns: An array of `String` objects, each giving the name of
    ///            one file contained in the bucket.
```

```
public func listBucketFiles(bucket: String) async throws -> [String] {
    do {
        let input = ListObjectsV2Input(
            bucket: bucket
        )

        // Use "Paginated" to get all the objects.
        // This lets the SDK handle the 'continuationToken' in
        "ListObjectsV2Output".
        let output = client.listObjectsV2Paginated(input: input)
        var names: [String] = []

        for try await page in output {
            guard let objList = page.contents else {
                print("ERROR: listObjectsV2Paginated returned nil contents.")
                continue
            }

            for obj in objList {
                if let objName = obj.key {
                    names.append(objName)
                }
            }
        }

        return names
    } catch {
        print("ERROR: ", dump(error, name: "Listing objects."))
        throw error
    }
}
```

```
import AWSS3

import Foundation
import ServiceHandler
import ArgumentParser

/// The command-line arguments and options available for this
```

```
/// example command.
struct ExampleCommand: ParsableCommand {
    @Argument(help: "Name of the S3 bucket to create")
    var bucketName: String

    @Argument(help: "Pathname of the file to upload to the S3 bucket")
    var uploadSource: String

    @Argument(help: "The name (key) to give the file in the S3 bucket")
    var objName: String

    @Argument(help: "S3 bucket to copy the object to")
    var destBucket: String

    @Argument(help: "Directory where you want to download the file from the S3
bucket")
    var downloadDir: String

    static var configuration = CommandConfiguration(
        commandName: "s3-basics",
        abstract: "Demonstrates a series of basic AWS S3 functions.",
        discussion: """
        Performs the following Amazon S3 commands:

        * `CreateBucket`
        * `PutObject`
        * `GetObject`
        * `CopyObject`
        * `ListObjects`
        * `DeleteObjects`
        * `DeleteBucket`
        """
    )

    /// Called by ``main()`` to do the actual running of the AWS
    /// example.
    func runAsync() async throws {
        let serviceHandler = try await ServiceHandler()

        // 1. Create the bucket.
        print("Creating the bucket \(bucketName)...")
        try await serviceHandler.createBucket(name: bucketName)

        // 2. Upload a file to the bucket.
```

```
        print("Uploading the file \(uploadSource)...")
        try await serviceHandler.uploadFile(bucket: bucketName, key: objName,
file: uploadSource)

        // 3. Download the file.
        print("Downloading the file \(objName) to \(downloadDir)...")
        try await serviceHandler.downloadFile(bucket: bucketName, key: objName,
to: downloadDir)

        // 4. Copy the file to another bucket.
        print("Copying the file to the bucket \(destBucket)...")
        try await serviceHandler.copyFile(from: bucketName, name: objName, to:
destBucket)

        // 5. List the contents of the bucket.

        print("Getting a list of the files in the bucket \(bucketName)")
        let fileList = try await serviceHandler.listBucketFiles(bucket:
bucketName)
        let numFiles = fileList.count
        if numFiles != 0 {
            print("\(numFiles) file\((numFiles > 1) ? "s" : "") in bucket
\(bucketName):")
            for name in fileList {
                print("  \(name)")
            }
        } else {
            print("No files found in bucket \(bucketName)")
        }

        // 6. Delete the objects from the bucket.

        print("Deleting the file \(objName) from the bucket \(bucketName)...")
        try await serviceHandler.deleteFile(bucket: bucketName, key: objName)
        print("Deleting the file \(objName) from the bucket \(destBucket)...")
        try await serviceHandler.deleteFile(bucket: destBucket, key: objName)

        // 7. Delete the bucket.
        print("Deleting the bucket \(bucketName)...")
        try await serviceHandler.deleteBucket(name: bucketName)

        print("Done.")
    }
}
```

```
//  
// Main program entry point.  
//  
@main  
struct Main {  
    static func main() async {  
        let args = Array(CommandLine.arguments.dropFirst())  
  
        do {  
            let command = try ExampleCommand.parse(args)  
            try await command.runAsync()  
        } catch {  
            ExampleCommand.exit(withError: error)  
        }  
    }  
}
```

- 如需 API 詳細資訊，請參閱《適用於 Swift 的 AWS SDK API 參考》中的下列主題。

- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObjects](#)
- [GetObject](#)
- [ListObjectsV2](#)
- [PutObject](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Amazon S3 AWS SDKs 的動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 Amazon S3 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

這些摘錄會呼叫 Amazon S3 API，是必須在內容中執行之大型程式的程式碼摘錄。您可以在 [Amazon S3 AWS SDKs 案例](#) 中查看內容中的動作。

下列範例僅包含最常使用的動作。如需完整的列表，請參閱 [Amazon Simple Storage Service API 參考資料](#)。

範例

- [AbortMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
- [CompleteMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
- [CopyObject 搭配 AWS SDK 或 CLI 使用](#)
- [CreateBucket 搭配 AWS SDK 或 CLI 使用](#)
- [CreateMultiRegionAccessPoint 搭配 AWS SDK 使用](#)
- [CreateMultipartUpload 搭配 AWS SDK 或 CLI 使用](#)
- [CreatePresignedPost 搭配 AWS SDK 使用](#)
- [DeleteBucket 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeleteBucketAnalyticsConfiguration 與 CLI](#)
- [DeleteBucketCors 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeleteBucketEncryption 與 CLI](#)
- [搭配使用 DeleteBucketInventoryConfiguration 與 CLI](#)
- [DeleteBucketLifecycle 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeleteBucketMetricsConfiguration 與 CLI](#)
- [DeleteBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeleteBucketReplication 與 CLI](#)
- [搭配使用 DeleteBucketTagging 與 CLI](#)
- [DeleteBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
- [DeleteObject 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeleteObjectTagging 與 CLI](#)
- [DeleteObjects 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 DeletePublicAccessBlock 與 CLI](#)
- [搭配使用 GetBucketAccelerateConfiguration 與 CLI](#)
- [GetBucketAcl 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketAnalyticsConfiguration 與 CLI](#)
- [GetBucketCors 搭配 AWS SDK 或 CLI 使用](#)
- [GetBucketEncryption 搭配 AWS SDK 或 CLI 使用](#)

- [搭配使用 GetBucketInventoryConfiguration 與 CLI](#)
- [GetBucketLifecycleConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [GetBucketLocation 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketLogging 與 CLI](#)
- [搭配使用 GetBucketMetricsConfiguration 與 CLI](#)
- [搭配使用 GetBucketNotification 與 CLI](#)
- [GetBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketPolicyStatus 與 CLI](#)
- [GetBucketReplication 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetBucketRequestPayment 與 CLI](#)
- [搭配使用 GetBucketTagging 與 CLI](#)
- [搭配使用 GetBucketVersioning 與 CLI](#)
- [GetBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
- [GetObject 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectAcl 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectAttributes 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectLegalHold 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectLockConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [GetObjectRetention 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetObjectTagging 與 CLI](#)
- [搭配使用 GetPublicAccessBlock 與 CLI](#)
- [HeadBucket 搭配 AWS SDK 或 CLI 使用](#)
- [HeadObject 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 ListBucketAnalyticsConfigurations 與 CLI](#)
- [搭配使用 ListBucketInventoryConfigurations 與 CLI](#)
- [ListBuckets 搭配 AWS SDK 或 CLI 使用](#)
- [ListMultipartUploads 搭配 AWS SDK 或 CLI 使用](#)
- [ListObjectVersions 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 ListObjects 與 CLI](#)
- [ListObjectsV2 搭配 AWS SDK 或 CLI 使用](#)

- [PutBucketAccelerateConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketAcl 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketCors 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketEncryption 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketLifecycleConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketLogging 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 PutBucketNotification 與 CLI](#)
- [PutBucketNotificationConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketReplication 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 PutBucketRequestPayment 與 CLI](#)
- [搭配使用 PutBucketTagging 與 CLI](#)
- [PutBucketVersioning 搭配 AWS SDK 或 CLI 使用](#)
- [PutBucketWebsite 搭配 AWS SDK 或 CLI 使用](#)
- [PutObject 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectAcl 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectLegalHold 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectLockConfiguration 搭配 AWS SDK 或 CLI 使用](#)
- [PutObjectRetention 搭配 AWS SDK 或 CLI 使用](#)
- [RestoreObject 搭配 AWS SDK 或 CLI 使用](#)
- [SelectObjectContent 搭配 AWS SDK 或 CLI 使用](#)
- [UploadPart 搭配 AWS SDK 或 CLI 使用](#)
- [UploadPartCopy 搭配 AWS SDK 或 CLI 使用](#)

AbortMultipartUpload 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 AbortMultipartUpload。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [刪除未完成的分段上傳](#)

- [使用 Amazon S3 物件完整性](#)

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
//! Abort a multipart upload to an S3 bucket.
/*!
    \param bucket: The name of the S3 bucket where the object will be uploaded.
    \param key: The unique identifier (key) for the object within the S3 bucket.
    \param uploadID: An upload ID string.
    \param client: The S3 client instance used to perform the upload operation.
    \return bool: Function succeeded.
*/

bool AwsDoc::S3::abortMultipartUpload(const Aws::String &bucket,
                                       const Aws::String &key,
                                       const Aws::String &uploadID,
                                       const Aws::S3::S3Client &client) {
    Aws::S3::Model::AbortMultipartUploadRequest request;
    request.SetBucket(bucket);
    request.SetKey(key);
    request.SetUploadId(uploadID);

    Aws::S3::Model::AbortMultipartUploadOutcome outcome =
        client.AbortMultipartUpload(request);

    if (outcome.IsSuccess()) {
        std::cout << "Multipart upload aborted." << std::endl;
    } else {
        std::cerr << "Error aborting multipart upload: " <<
            outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中 [AbortMultipartUpload](#)。

CLI

AWS CLI

中止指定的分段上傳

下列 `abort-multipart-upload` 命令會中止儲存貯體 `amzn-s3-demo-bucket` 中金鑰 `multipart/01` 的分段上傳。

```
aws s3api abort-multipart-upload \
  --bucket amzn-s3-demo-bucket \
  --key multipart/01 \
  --upload-
id dfRtDYU0WwCCcH43C3WfbkR0NycyCpTJJvxu2i5GYkZLjF.Yxwh6XG7WfS2vC4to6HiV6YjLx.cph0gtNBtJ8P
```

此命令所需的上傳 ID 是由 `create-multipart-upload` 輸出，也可以使用 `list-multipart-uploads` 擷取。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中 [AbortMultipartUpload](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.S3Client;
```

```
import software.amazon.awssdk.services.s3.model.AbortMultipartUploadRequest;
import software.amazon.awssdk.services.s3.model.AbortMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.CompletedMultipartUpload;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.LifecycleRule;
import software.amazon.awssdk.services.s3.model.ListMultipartUploadsRequest;
import software.amazon.awssdk.services.s3.model.ListMultipartUploadsResponse;
import software.amazon.awssdk.services.s3.model.MultipartUpload;
import
    software.amazon.awssdk.services.s3.model.PutBucketLifecycleConfigurationResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.UploadPartRequest;
import software.amazon.awssdk.services.s3.model.UploadPartResponse;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.utils.builder.SdkBuilder;

import java.io.IOException;
import java.io.RandomAccessFile;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.ByteBuffer;
import java.time.Duration;
import java.time.Instant;
import java.util.ArrayList;
import java.util.Collection;
import java.util.List;
import java.util.Objects;
import java.util.UUID;

import static software.amazon.awssdk.transfer.s3.SizeConstant.KB;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */

public class AbortMultipartUploadExamples {
```

```

    static final S3Client s3Client = S3Client.create();
    static final String classPathFilePath = "/multipartUploadFiles/s3-
userguide.pdf";
    static final String filePath = getFullFilePath(classPathFilePath);
    private static final Logger logger =
LoggerFactory.getLogger(AbortMultipartUploadExamples.class);
    private static final String accountId = getAccountId();

    public static void main(String[] args) {
        String bucketName = "amzn-s3-demo-bucket"; // Replace with your bucket
name.

        String key = UUID.randomUUID().toString();

        createBucket(bucketName);
        try {
            initiateAndInterruptMultiPartUpload(bucketName, key, "uploadThread");
            abortIncompleteMultipartUploadsFromList(bucketName);
            abortMultipartUploadUsingUploadId(bucketName, key);
            abortMultipartUploadsUsingLifecycleConfig(bucketName);
        } catch (S3Exception e) {
            logger.error(e.getMessage());
        } finally {
            deleteResources(bucketName, key);
        }
    }

    /**
     * Aborts all incomplete multipart uploads from the specified S3 bucket.
     *
     * @param bucketName the name of the S3 bucket
     */
    public static void abortIncompleteMultipartUploadsFromList(String bucketName)
    {
        ListMultipartUploadsRequest listMultipartUploadsRequest =
ListMultipartUploadsRequest.builder()
            .bucket(bucketName)
            .build();

        ListMultipartUploadsResponse response =
s3Client.listMultipartUploads(listMultipartUploadsRequest);
        List<MultipartUpload> uploads = response.uploads();

        AbortMultipartUploadRequest abortMultipartUploadRequest;
        for (MultipartUpload upload : uploads) {

```

```

        abortMultipartUploadRequest = AbortMultipartUploadRequest.builder()
            .bucket(bucketName)
            .key(upload.key())
            .expectedBucketOwner(accountId)
            .uploadId(upload.uploadId())
            .build();

        AbortMultipartUploadResponse abortMultipartUploadResponse =
s3Client.abortMultipartUpload(abortMultipartUploadRequest);
        if (abortMultipartUploadResponse.sdkHttpResponse().isSuccessful()) {
            logger.info("Upload ID [{}] to bucket [{}] successfully
aborted.", upload.uploadId(), bucketName);
        }
    }
}

/**
 * Aborts incomplete multipart uploads older than the specified point in
time.
 *
 * @param bucketName the name of the S3 bucket
 * @param pointInTime the cutoff time; uploads initiated before this are
aborted
 */
static void abortIncompleteMultipartUploadsOlderThan(String bucketName,
Instant pointInTime) {
    ListMultipartUploadsRequest listMultipartUploadsRequest =
ListMultipartUploadsRequest.builder()
        .bucket(bucketName)
        .build();

    ListMultipartUploadsResponse response =
s3Client.listMultipartUploads(listMultipartUploadsRequest);
    List<MultipartUpload> uploads = response.uploads();

    AbortMultipartUploadRequest abortMultipartUploadRequest;
    for (MultipartUpload upload : uploads) {
        logger.info("Found multipartUpload with upload ID [{}], initiated
[{}]", upload.uploadId(), upload.initiated());
        if (upload.initiated().isBefore(pointInTime)) {
            abortMultipartUploadRequest =
AbortMultipartUploadRequest.builder()
                .bucket(bucketName)
                .key(upload.key())

```

```

        .expectedBucketOwner(accountId)
        .uploadId(upload.uploadId())
        .build();

        AbortMultipartUploadResponse abortMultipartUploadResponse =
s3Client.abortMultipartUpload(abortMultipartUploadRequest);
        if
(abortMultipartUploadResponse.sdkHttpResponse().isSuccessful()) {
            logger.info("Upload ID [{}] to bucket [{}] successfully
aborted.", upload.uploadId(), bucketName);
        }
    }
}

/**
 * Aborts a multipart upload using the upload ID.
 *
 * @param bucketName the name of the S3 bucket
 * @param key         the object key
 */
static void abortMultipartUploadUsingUploadId(String bucketName, String key)
{
    String uploadId = startUploadReturningUploadId(bucketName, key);
    AbortMultipartUploadResponse response = s3Client.abortMultipartUpload(b -
> b
        .uploadId(uploadId)
        .bucket(bucketName)
        .key(key));

    if (response.sdkHttpResponse().isSuccessful()) {
        logger.info("Upload ID [{}] to bucket [{}] successfully aborted.",
uploadId, bucketName);
    }
}

/**
 * Configures a lifecycle rule to abort incomplete multipart uploads after 7
days.
 *
 * @param bucketName the name of the S3 bucket
 */
static void abortMultipartUploadsUsingLifecycleConfig(String bucketName) {

```

```

        Collection<LifecycleRule> lifeCycleRules =
List.of(LifecycleRule.builder()
        .abortIncompleteMultipartUpload(b -> b.
            daysAfterInitiation(7))
        .status("Enabled")
        .filter(SdkBuilder::build) // Filter element is required.
        .build());

        PutBucketLifecycleConfigurationResponse response =
s3Client.putBucketLifecycleConfiguration(b -> b
        .bucket(bucketName)
        .lifecycleConfiguration(b1 -> b1.rules(lifeCycleRules)));

        if (response.sdkHttpResponse().isSuccessful()) {
            logger.info("Rule to abort incomplete multipart uploads added to
bucket.");
        } else {
            logger.error("Unsuccessfully applied rule. HTTP status code is [{}]",
response.sdkHttpResponse().statusCode());
        }
    }

    /*****
    Multipart upload methods
    *****/

    static void initiateAndInterruptMultiPartUpload(String bucketName, String
key, String threadName) {
        Runnable upload = () -> {
            try {
                doMultipartUpload(bucketName, key);
            } catch (SdkException e) {
                logger.error(e.getMessage());
            }
        };
        Thread uploadThread = new Thread(upload, threadName);
        uploadThread.start();
        try {
            Thread.sleep(Duration.ofSeconds(1).toMillis());
        } catch (InterruptedException e) {
            logger.error(e.getMessage());
        }
        uploadThread.interrupt();
    }
}

```



```
static Instant initiateAndInterruptTwoUploads(String bucketName, String key)
{
    initiateAndInterruptMultiPartUpload(bucketName, key, "uploadThread1");
    try {
        Thread.sleep(Duration.ofSeconds(5).toMillis());
    } catch (InterruptedException e) {
        logger.error(e.getMessage());
    }
    Instant secondUploadInstant = Instant.now();
    initiateAndInterruptMultiPartUpload(bucketName, key, "uploadThread2");
    return secondUploadInstant;
}

static void doMultipartUpload(String bucketName, String key) {
    String uploadId = step1CreateMultipartUpload(bucketName, key);
    List<CompletedPart> completedParts = step2UploadParts(bucketName, key,
uploadId);
    step3CompleteMultipartUpload(bucketName, key, uploadId, completedParts);
}

static String step1CreateMultipartUpload(String bucketName, String key) {
    CreateMultipartUploadResponse createMultipartUploadResponse =
s3Client.createMultipartUpload(b -> b
        .bucket(bucketName)
        .key(key));
    return createMultipartUploadResponse.uploadId();
}

static List<CompletedPart> step2UploadParts(String bucketName, String key,
String uploadId) {
    int partNumber = 1;
    List<CompletedPart> completedParts = new ArrayList<>();
    ByteBuffer bb = ByteBuffer.allocate(Long.valueOf(1024 * KB).intValue());

    try (RandomAccessFile file = new RandomAccessFile(filePath, "r")) {
        long fileSize = file.length();
        long position = 0;
        while (position < fileSize) {
            file.seek(position);
            long read = file.getChannel().read(bb);

            bb.flip();
            UploadPartRequest uploadPartRequest = UploadPartRequest.builder()
```

```
        .bucket(bucketName)
        .key(key)
        .uploadId(uploadId)
        .partNumber(partNumber)
        .build();

UploadPartResponse partResponse = s3Client.uploadPart(
    uploadPartRequest,
    RequestBody.fromByteBuffer(bb));

CompletedPart part = CompletedPart.builder()
    .partNumber(partNumber)
    .eTag(partResponse.eTag())
    .build();
completedParts.add(part);
logger.info("Part {} upload", partNumber);

bb.clear();
position += read;
partNumber++;
    }
} catch (IOException | S3Exception e) {
    logger.error(e.getMessage());
    return null;
}
return completedParts;
}

static void step3CompleteMultipartUpload(String bucketName, String key,
String uploadId, List<CompletedPart> completedParts) {
    s3Client.completeMultipartUpload(b -> b
        .bucket(bucketName)
        .key(key)
        .uploadId(uploadId)

    .multipartUpload(CompletedMultipartUpload.builder().parts(completedParts).build()));
}

static String startUploadReturningUploadId(String bucketName, String key) {
    String uploadId = step1CreateMultipartUpload(bucketName, key);
    doMultipartUploadWithUploadId(bucketName, key, uploadId);
    return uploadId;
}
```

```

    static void doMultipartUploadWithUploadId(String bucketName, String key,
String uploadId) {
        new Thread(() -> {
            try {
                List<CompletedPart> completedParts = step2UploadParts(bucketName,
key, uploadId);
                step3CompleteMultipartUpload(bucketName, key, uploadId,
completedParts);
            } catch (SdkException e) {
                logger.error(e.getMessage());
            }
        }, "upload thread").start();
        try {
            Thread.sleep(Duration.ofSeconds(2L).toMillis());
        } catch (InterruptedException e) {
            logger.error(e.getMessage());
            System.exit(1);
        }
    }

    /*****
Resource handling methods
*****/

    static void createBucket(String bucketName) {
        logger.info("Creating bucket: [{}]", bucketName);
        s3Client.createBucket(b -> b.bucket(bucketName));
        try (S3Waiter s3Waiter = s3Client.waiter()) {
            s3Waiter.waitUntilBucketExists(b -> b.bucket(bucketName));
        }
        logger.info("Bucket created.");
    }

    static void deleteResources(String bucketName, String key) {
        logger.info("Deleting resources ...");
        s3Client.deleteObject(b -> b.bucket(bucketName).key(key));
        s3Client.deleteBucket(b -> b.bucket(bucketName));
        try (S3Waiter s3Waiter = s3Client.waiter()) {
            s3Waiter.waitUntilBucketNotExists(b -> b.bucket(bucketName));
        }
        logger.info("Resources deleted.");
    }

    private static String getAccountId() {

```

```
        try (StsClient stsClient = StsClient.create()) {
            return stsClient.getCallerIdentity().account();
        }
    }

    static String getFullFilePath(String filePath) {
        URL uploadDirectoryURL =
PerformMultiPartUpload.class.getResource(filePath);
        String fullFilePath;
        try {
            fullFilePath =
Objects.requireNonNull(uploadDirectoryURL).toURI().getPath();
        } catch (URISyntaxException e) {
            throw new RuntimeException(e);
        }
        return fullFilePath;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中 [AbortMultipartUpload](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令中止 5 天前建立的分段上傳。

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -DaysBefore 5
```

範例 2：此命令中止 2014 年 1 月 2 日之前建立的分段上傳。

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -InitiatedDate
"Thursday, January 02, 2014"
```

範例 3：此命令中止 2014 年 1 月 2 日 10:45:37 之前建立的分段上傳。

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -InitiatedDate
"2014/01/02 10:45:37"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中 [AbortMultipartUpload](#)。

Tools for PowerShell V5

範例 1：此命令中止 5 天前建立的分段上傳。

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -DaysBefore 5
```

範例 2：此命令中止 2014 年 1 月 2 日之前建立的分段上傳。

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -InitiatedDate  
"Thursday, January 02, 2014"
```

範例 3：此命令中止 2014 年 1 月 2 日 10:45:37 之前建立的分段上傳。

```
Remove-S3MultipartUpload -BucketName amzn-s3-demo-bucket -InitiatedDate  
"2014/01/02 10:45:37"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中 [AbortMultipartUpload](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CompleteMultipartUpload 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CompleteMultipartUpload。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [執行分段複製](#)
- [使用檢查總和](#)
- [使用 Amazon S3 物件完整性](#)

C++

適用於 C++ 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
//! Complete a multipart upload to an S3 bucket.
/*!
    \param bucket: The name of the S3 bucket where the object will be uploaded.
    \param key: The unique identifier (key) for the object within the S3 bucket.
    \param uploadID: An upload ID string.
    \param parts: A vector of CompleteParts.
    \param client: The S3 client instance used to perform the upload operation.
    \return CompleteMultipartUploadOutcome: The request outcome.
*/
Aws::S3::Model::CompleteMultipartUploadOutcome
AwsDoc::S3::completeMultipartUpload(const Aws::String &bucket,

    const Aws::String &key,

    const Aws::String &uploadID,

    const Aws::Vector<Aws::S3::Model::CompletedPart> &parts,

    const Aws::S3::S3Client &client) {
    Aws::S3::Model::CompletedMultipartUpload completedMultipartUpload;
    completedMultipartUpload.SetParts(parts);

    Aws::S3::Model::CompleteMultipartUploadRequest request;
    request.SetBucket(bucket);
    request.SetKey(key);
    request.SetUploadId(uploadID);
    request.SetMultipartUpload(completedMultipartUpload);

    Aws::S3::Model::CompleteMultipartUploadOutcome outcome =
        client.CompleteMultipartUpload(request);

    if (!outcome.IsSuccess()) {
```

```
std::cerr << "Error completing multipart upload: " <<
outcome.GetError().GetMessage() << std::endl;
}
return outcome;
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [CompleteMultipartUpload](#)。

CLI

AWS CLI

下列命令會完成儲存貯體 `amzn-s3-demo-bucket` 中金鑰 `multipart/01` 的分段上傳：

```
aws s3api complete-multipart-upload --multipart-upload file://
mpustruct --bucket amzn-s3-demo-bucket --key 'multipart/01' --upload-
id dfRtDYU0WMCCcH43C3WFbkR0NycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6YjlX.cph0gtNBtJ8P
```

此命令所需的上傳 ID 是由 `create-multipart-upload` 輸出，也可以使用 `list-multipart-uploads` 擷取。

上述命令中的分段上傳選項採用 JSON 結構，該結構描述了應重新組合成完整檔案的分段上傳的各個部分。在此範例中，`file://` 字首用於從本機資料夾中名為 `mpustruct` 的檔案載入 JSON 結構。

`mpustruct`：

```
{
  "Parts": [
    {
      "ETag": "e868e0f4719e394144ef36531ee6824c",
      "PartNumber": 1
    },
    {
      "ETag": "6bb2b12753d66fe86da4998aa33fffb0",
      "PartNumber": 2
    },
    {
      "ETag": "d0a0112e841abec9c9ec83406f0159c8",
```

```
    "PartNumber": 3
  }
]
}
```

每次使用 `upload-part` 命令上傳各部分時，都會輸出每個部分的 ETag 值，也可以透過呼叫 `list-parts` 或計算每個部分的 MD5 檢查總和來擷取。

輸出：

```
{
  "ETag": "\"3944a9f7a4faab7f78788ff6210f63f0-3\"",
  "Bucket": "amzn-s3-demo-bucket",
  "Location": "https://amzn-s3-demo-bucket.s3.amazonaws.com/multipart%2F01",
  "Key": "multipart/01"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CompleteMultipartUpload](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
// upload_parts: Vec<aws_sdk_s3::types::CompletedPart>
let completed_multipart_upload: CompletedMultipartUpload =
CompletedMultipartUpload::builder()
    .set_parts(Some(upload_parts))
    .build();

let _complete_multipart_upload_res = client
    .complete_multipart_upload()
    .bucket(&bucket_name)
    .key(&key)
    .multipart_upload(completed_multipart_upload)
```



```
.upload_id(upload_id)
.send()
.await?;
```

```
// Create a multipart upload. Use UploadPart and CompleteMultipartUpload to
// upload the file.
let multipart_upload_res: CreateMultipartUploadOutput = client
    .create_multipart_upload()
    .bucket(&bucket_name)
    .key(&key)
    .send()
    .await?;

let upload_id = multipart_upload_res.upload_id().ok_or(S3ExampleError::new(
    "Missing upload_id after CreateMultipartUpload",
))?;
```

```
let mut upload_parts: Vec<aws_sdk_s3::types::CompletedPart> = Vec::new();

for chunk_index in 0..chunk_count {
    let this_chunk = if chunk_count - 1 == chunk_index {
        size_of_last_chunk
    } else {
        CHUNK_SIZE
    };
    let stream = ByteStream::read_from()
        .path(path)
        .offset(chunk_index * CHUNK_SIZE)
        .length(Length::Exact(this_chunk))
        .build()
        .await
        .unwrap();

    // Chunk index needs to start at 0, but part numbers start at 1.
    let part_number = (chunk_index as i32) + 1;
    let upload_part_res = client
        .upload_part()
        .key(&key)
        .bucket(&bucket_name)
        .upload_id(upload_id)
        .body(stream)
```

```
        .part_number(part_number)
        .send()
        .await?;

    upload_parts.push(
        CompletedPart::builder()
            .e_tag(upload_part_res.e_tag.unwrap_or_default())
            .part_number(part_number)
            .build(),
    );
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [CompleteMultipartUpload](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CopyObject 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CopyObject。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)
- [開始使用加密](#)
- [物件儲存入門](#)
- [提出條件式請求](#)

.NET

適用於 .NET 的 SDK (v4)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Copies an object in an Amazon S3 bucket to a folder within the
/// same bucket.
/// </summary>
/// <param name="bucketName">The name of the Amazon S3 bucket where the
/// object to copy is located.</param>
/// <param name="objectName">The object to be copied.</param>
/// <param name="folderName">The folder to which the object will
/// be copied.</param>
/// <returns>A boolean value that indicates the success or failure of
/// the copy operation.</returns>
public async Task<bool> CopyObjectInBucketAsync(
    string bucketName,
    string objectName,
    string folderName)
{
    try
    {
        var request = new CopyObjectRequest
        {
            SourceBucket = bucketName,
            SourceKey = objectName,
            DestinationBucket = bucketName,
            DestinationKey = $"{folderName}\\{objectName}",
        };
        var response = await _amazonS3.CopyObjectAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error copying object: '{ex.Message}'");
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [CopyObject](#)。

適用於 .NET 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用條件式請求複製物件。

```
/// <summary>
/// Copies an object from one Amazon S3 bucket to another with a conditional
request.
/// </summary>
/// <param name="sourceKey">The key of the source object to copy.</param>
/// <param name="destKey">The key of the destination object.</param>
/// <param name="sourceBucket">The source bucket of the object.</param>
/// <param name="destBucket">The destination bucket of the object.</param>
/// <param name="conditionType">The type of condition to apply, e.g.
'CopySourceIfMatch', 'CopySourceIfNoneMatch', 'CopySourceIfModifiedSince',
'CopySourceIfUnmodifiedSince'.</param>
/// <param name="conditionDateValue">The value to use for the condition for
dates.</param>
/// <param name="etagConditionalValue">The value to use for the condition for
etags.</param>
/// <returns>True if the conditional copy is successful, False otherwise.</
returns>
public async Task<bool> CopyObjectConditional(string sourceKey, string
destKey, string sourceBucket, string destBucket,
    S3ConditionType conditionType, DateTime? conditionDateValue = null,
string? etagConditionalValue = null)
{
    try
    {
        var copyObjectRequest = new CopyObjectRequest
        {
            DestinationBucket = destBucket,
            DestinationKey = destKey,
            SourceBucket = sourceBucket,
            SourceKey = sourceKey
        };

        switch (conditionType)
```

```
        {
            case S3ConditionType.IfMatch:
                copyObjectRequest.ETagToMatch = etagConditionalValue;
                break;
            case S3ConditionType.IfNoneMatch:
                copyObjectRequest.ETagToNotMatch = etagConditionalValue;
                break;
            case S3ConditionType.IfModifiedSince:
                copyObjectRequest.ModifiedSinceDateUtc =
conditionDateValue.GetValueOrDefault();
                break;
            case S3ConditionType.IfUnmodifiedSince:
                copyObjectRequest.UnmodifiedSinceDateUtc =
conditionDateValue.GetValueOrDefault();
                break;
            default:
                throw new ArgumentOutOfRangeException(nameof(conditionType),
conditionType, null);
        }

        await _amazonS3.CopyObjectAsync(copyObjectRequest);
        _logger.LogInformation($"Conditional copy successful for key
{destKey} in bucket {destBucket}.");
        return true;
    }
    catch (AmazonS3Exception e)
    {
        if (e.ErrorCode == "PreconditionFailed")
        {
            _logger.LogError("Conditional copy failed: Precondition failed");
        }
        else if (e.ErrorCode == "304")
        {
            _logger.LogError("Conditional copy failed: Object not modified");
        }
        else
        {
            _logger.LogError($"Unexpected error: {e.ErrorCode}");
            throw;
        }
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [CopyObject](#)。

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function copy_item_in_bucket
#
# This function creates a copy of the specified file in the same bucket.
#
# Parameters:
#     $1 - The name of the bucket to copy the file from and to.
#     $2 - The key of the source file to copy.
#     $3 - The key of the destination file.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function copy_item_in_bucket() {
    local bucket_name=$1
    local source_key=$2
    local destination_key=$3
    local response
```

```
response=$(aws s3api copy-object \  
  --bucket "$bucket_name" \  
  --copy-source "$bucket_name/$source_key" \  
  --key "$destination_key")  
  
# shellcheck disable=SC2181  
if [[ $? -ne 0 ]]; then  
  errecho "ERROR: AWS reports s3api copy-object operation failed.\n$response"  
  return 1  
fi  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CopyObject](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::copyObject(const Aws::String &objectKey, const Aws::String  
  &fromBucket, const Aws::String &toBucket,  
                        const Aws::S3::S3ClientConfiguration &clientConfig) {  
  Aws::S3::S3Client client(clientConfig);  
  Aws::S3::Model::CopyObjectRequest request;  
  
  request.WithCopySource(fromBucket + "/" + objectKey)  
    .WithKey(objectKey)  
    .WithBucket(toBucket);  
  
  Aws::S3::Model::CopyObjectOutcome outcome = client.CopyObject(request);  
  if (!outcome.IsSuccess()) {  
    const Aws::S3::S3Error &err = outcome.GetError();  
    std::cerr << "Error: copyObject: " <<  
      err.GetExceptionName() << ": " << err.GetMessage() <<  
    std::endl;  
  }
```

```
    } else {  
        std::cout << "Successfully copied " << objectKey << " from " <<  
fromBucket <<  
        " to " << toBucket << "." << std::endl;  
    }  
  
    return outcome.IsSuccess();  
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [CopyObject](#)。

CLI

AWS CLI

下列命令會將物件從 bucket-1 複製到 bucket-2：

```
aws s3api copy-object --copy-source bucket-1/test.txt --key test.txt --  
bucket bucket-2
```

輸出：

```
{  
  "CopyObjectResult": {  
    "LastModified": "2015-11-10T01:07:25.000Z",  
    "ETag": "\"589c8b79c230a6ecd5a7e1d040a9a030\""  
  },  
  "VersionId": "YdnYvTCVDqRRFA.NFJjy36p0hxifM1kA"  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CopyObject](#)。

Go

SDK for Go V2

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (  
    "bytes"  
    "context"  
    "errors"  
    "fmt"  
    "io"  
    "log"  
    "os"  
    "time"  
  
    "github.com/aws/aws-sdk-go-v2/aws"  
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"  
    "github.com/aws/aws-sdk-go-v2/service/s3"  
    "github.com/aws/aws-sdk-go-v2/service/s3/types"  
    "github.com/aws/smithy-go"  
)  
  
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)  
// actions  
// used in the examples.  
// It contains S3Client, an Amazon S3 service client that is used to perform  
// bucket  
// and object actions.  
type BucketBasics struct {  
    S3Client *s3.Client  
}  
  
// CopyToBucket copies an object in a bucket to another bucket.
```

```
func (basics BucketBasics) CopyToBucket(ctx context.Context, sourceBucket string,
    destinationBucket string, objectKey string) error {
    _, err := basics.S3Client.CopyObject(ctx, &s3.CopyObjectInput{
        Bucket:      aws.String(destinationBucket),
        CopySource:  aws.String(fmt.Sprintf("%v/%v", sourceBucket, objectKey)),
        Key:         aws.String(objectKey),
    })
    if err != nil {
        var notActive *types.ObjectNotInActiveTierError
        if errors.As(err, &notActive) {
            log.Printf("Couldn't copy object %s from %s because the object isn't in the
                active tier.\n",
                objectKey, sourceBucket)
            err = notActive
        }
    } else {
        err = s3.NewObjectExistsWaiter(basics.S3Client).Wait(
            ctx, &s3.HeadObjectInput{Bucket: aws.String(destinationBucket), Key:
                aws.String(objectKey)}, time.Minute)
        if err != nil {
            log.Printf("Failed attempt to wait for object %s to exist.\n", objectKey)
        }
    }
    return err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [CopyObject](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用 [S3Client](#) 複製物件。

```

/**
 * Asynchronously copies an object from one S3 bucket to another.
 *
 * @param fromBucket the name of the source S3 bucket
 * @param objectKey the key (name) of the object to be copied
 * @param toBucket the name of the destination S3 bucket
 * @return a {@link CompletableFuture} that completes with the copy result as
a {@link String}
 * @throws RuntimeException if the URL could not be encoded or an S3
exception occurred during the copy
 */
public CompletableFuture<String> copyBucketObjectAsync(String fromBucket,
String objectKey, String toBucket) {
    CopyObjectRequest copyReq = CopyObjectRequest.builder()
        .sourceBucket(fromBucket)
        .sourceKey(objectKey)
        .destinationBucket(toBucket)
        .destinationKey(objectKey)
        .build();

    CompletableFuture<CopyObjectResponse> response =
getAsyncClient().copyObject(copyReq);
    response.whenComplete((copyRes, ex) -> {
        if (copyRes != null) {
            logger.info("The " + objectKey + " was copied to " + toBucket);
        } else {
            throw new RuntimeException("An S3 exception occurred during
copy", ex);
        }
    });

    return response.thenApply(CopyObjectResponse::copyObjectResult)
        .thenApply(Object::toString);
}

```

使用 [S3TransferManager](#) 將物件從一個儲存貯體複製到另一個儲存貯體。檢視[完整檔案](#)並[測試](#)。

```

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.sync.RequestBody;

```

```
import software.amazon.awssdk.services.s3.model.CopyObjectRequest;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedCopy;
import software.amazon.awssdk.transfer.s3.model.Copy;
import software.amazon.awssdk.transfer.s3.model.CopyRequest;

import java.util.UUID;

public String copyObject(S3TransferManager transferManager, String
bucketName,
    String key, String destinationBucket, String destinationKey) {
    CopyObjectRequest copyObjectRequest = CopyObjectRequest.builder()
        .sourceBucket(bucketName)
        .sourceKey(key)
        .destinationBucket(destinationBucket)
        .destinationKey(destinationKey)
        .build();

    CopyRequest copyRequest = CopyRequest.builder()
        .copyObjectRequest(copyObjectRequest)
        .build();

    Copy copy = transferManager.copy(copyRequest);

    CompletedCopy completedCopy = copy.completionFuture().join();
    return completedCopy.response().copyObjectResult().eTag();
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CopyObject](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

複製物件

```
import {
  S3Client,
  CopyObjectCommand,
  ObjectNotInActiveTierError,
  waitUntilObjectExists,
} from "@aws-sdk/client-s3";

/**
 * Copy an S3 object from one bucket to another.
 *
 * @param {{
 *   sourceBucket: string,
 *   sourceKey: string,
 *   destinationBucket: string,
 *   destinationKey: string }} config
 */
export const main = async ({
  sourceBucket,
  sourceKey,
  destinationBucket,
  destinationKey,
}) => {
  const client = new S3Client({});

  try {
    await client.send(
      new CopyObjectCommand({
        CopySource: `${sourceBucket}/${sourceKey}`,
        Bucket: destinationBucket,
        Key: destinationKey,
      }),
    );
    await waitUntilObjectExists(
      { client },
      { Bucket: destinationBucket, Key: destinationKey },
    );
    console.log(
      `Successfully copied ${sourceBucket}/${sourceKey} to ${destinationBucket}/${destinationKey}`,
    );
  } catch (caught) {
    if (caught instanceof ObjectNotInActiveTierError) {
      console.error(
```

```

        `Could not copy ${sourceKey} from ${sourceBucket}. Object is not in the
        active tier.` ,
    );
    } else {
        throw caught;
    }
}
};

```

如果物件的 ETag 與提供的 ETag 不相符，則複製該物件。

```

import {
    CopyObjectCommand,
    NoSuchKey,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

// Optionally edit the default key name of the copied object in
'object_name.json'
import data from "../scenarios/conditional-requests/object_name.json" assert {
    type: "json",
};

/**
 * Get a single object from a specified S3 bucket.
 * @param {{ sourceBucketName: string, sourceKeyName: string,
 * destinationBucketName: string, eTag: string }}
 */
export const main = async ({
    sourceBucketName,
    sourceKeyName,
    destinationBucketName,
    eTag,
}) => {
    const client = new S3Client({});
    const name = data.name;
    try {
        const response = await client.send(
            new CopyObjectCommand({
                CopySource: `${sourceBucketName}/${sourceKeyName}`,

```

```

        Bucket: destinationBucketName,
        Key: `${name}${sourceKeyName}`,
        CopySourceIfMatch: eTag,
    )),
    );
    console.log("Successfully copied object to bucket.");
} catch (caught) {
    if (caught instanceof NoSuchKey) {
        console.error(
            `Error from S3 while copying object "${sourceKeyName}" from
"${sourceBucketName}". No such key exists.`
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Unable to copy object "${sourceKeyName}" to bucket
"${sourceBucketName}": ${caught.name}: ${caught.message}`
        );
    } else {
        throw caught;
    }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
    isMain,
    validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
    const options = {
        sourceBucketName: {
            type: "string",
            required: true,
        },
        sourceKeyName: {
            type: "string",
            required: true,
        },
        destinationBucketName: {
            type: "string",
            required: true,
        },
    },

```

```

    eTag: {
      type: "string",
      required: true,
    },
  };
  const results = parseArgs({ options });
  const { errors } = validateArgs({ options }, results);
  return { errors, results };
};

if (isMain(import.meta.url)) {
  const { errors, results } = loadArgs();
  if (!errors) {
    main(results.values);
  } else {
    console.error(errors.join("\n"));
  }
}

```

如果物件的 ETag 與提供的 ETag 不相符，則複製該物件。

```

import {
  CopyObjectCommand,
  NoSuchKey,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

// Optionally edit the default key name of the copied object in
'object_name.json'
import data from "../scenarios/conditional-requests/object_name.json" assert {
  type: "json",
};

/**
 * Get a single object from a specified S3 bucket.
 * @param {{ sourceBucketName: string, sourceKeyName: string,
  destinationBucketName: string, eTag: string }}
 */
export const main = async ({
  sourceBucketName,

```



```
    sourceKeyName,
    destinationBucketName,
    eTag,
  }) => {
    const client = new S3Client({});
    const name = data.name;

    try {
      const response = await client.send(
        new CopyObjectCommand({
          CopySource: `${sourceBucketName}/${sourceKeyName}`,
          Bucket: destinationBucketName,
          Key: `${name}${sourceKeyName}`,
          CopySourceIfNoneMatch: eTag,
        }),
      );
      console.log("Successfully copied object to bucket.");
    } catch (caught) {
      if (caught instanceof NoSuchKey) {
        console.error(
          `Error from S3 while copying object "${sourceKeyName}" from
"${sourceBucketName}". No such key exists.`,
        );
      } else if (caught instanceof S3ServiceException) {
        console.error(
          `Unable to copy object "${sourceKeyName}" to bucket
"${sourceBucketName}": ${caught.name}: ${caught.message}`,
        );
      } else {
        throw caught;
      }
    }
  };

// Call function if run directly
import { parseArgs } from "node:util";
import {
  isMain,
  validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
  const options = {
    sourceBucketName: {
```

```

        type: "string",
        required: true,
    },
    sourceKeyName: {
        type: "string",
        required: true,
    },
    destinationBucketName: {
        type: "string",
        required: true,
    },
    eTag: {
        type: "string",
        required: true,
    },
    },
};
const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
return { errors, results };
};

if (isMain(import.meta.url)) {
    const { errors, results } = loadArgs();
    if (!errors) {
        main(results.values);
    } else {
        console.error(errors.join("\n"));
    }
}
}

```

在給定的時間範圍內建立或修改物件的情況下，複製該物件。

```

import {
    CopyObjectCommand,
    NoSuchKey,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

// Optionally edit the default key name of the copied object in
'object_name.json'

```

```
import data from "../scenarios/conditional-requests/object_name.json" assert {
  type: "json",
};

/**
 * Get a single object from a specified S3 bucket.
 * @param {{ sourceBucketName: string, sourceKeyName: string,
  destinationBucketName: string }}
 */
export const main = async ({
  sourceBucketName,
  sourceKeyName,
  destinationBucketName,
}) => {
  const date = new Date();
  date.setDate(date.getDate() - 1);

  const name = data.name;
  const client = new S3Client({});
  const copySource = `${sourceBucketName}/${sourceKeyName}`;
  const copiedKey = name + sourceKeyName;

  try {
    const response = await client.send(
      new CopyObjectCommand({
        CopySource: copySource,
        Bucket: destinationBucketName,
        Key: copiedKey,
        CopySourceIfModifiedSince: date,
      }),
    );
    console.log("Successfully copied object to bucket.");
  } catch (caught) {
    if (caught instanceof NoSuchKey) {
      console.error(
        `Error from S3 while copying object "${sourceKeyName}" from
        "${sourceBucketName}". No such key exists.`
      );
    } else if (caught instanceof S3ServiceException) {
      console.error(
        `Error from S3 while copying object from ${sourceBucketName}.
        ${caught.name}: ${caught.message}`
      );
    } else {

```

```
        throw caught;
    }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
    isMain,
    validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
    const options = {
        sourceBucketName: {
            type: "string",
            required: true,
        },
        sourceKeyName: {
            type: "string",
            required: true,
        },
        destinationBucketName: {
            type: "string",
            required: true,
        },
    };
};

const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
return { errors, results };
};

if (isMain(import.meta.url)) {
    const { errors, results } = loadArgs();
    if (!errors) {
        main(results.values);
    } else {
        console.error(errors.join("\n"));
    }
}
```

在給定的時間範圍內未建立或修改物件的情況下，複製該物件。

```
import {
  CopyObjectCommand,
  NoSuchKey,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

// Optionally edit the default key name of the copied object in
'object_name.json'
import data from "../scenarios/conditional-requests/object_name.json" assert {
  type: "json",
};

/**
 * Get a single object from a specified S3 bucket.
 * @param {{ sourceBucketName: string, sourceKeyName: string,
  destinationBucketName: string }}
 */
export const main = async ({
  sourceBucketName,
  sourceKeyName,
  destinationBucketName,
}) => {
  const date = new Date();
  date.setDate(date.getDate() - 1);
  const client = new S3Client({});
  const name = data.name;
  const copiedKey = name + sourceKeyName;
  const copySource = `${sourceBucketName}/${sourceKeyName}`;

  try {
    const response = await client.send(
      new CopyObjectCommand({
        CopySource: copySource,
        Bucket: destinationBucketName,
        Key: copiedKey,
        CopySourceIfUnmodifiedSince: date,
      }),
    );
    console.log("Successfully copied object to bucket.");
  } catch (caught) {
    if (caught instanceof NoSuchKey) {
```

```
        console.error(
            `Error from S3 while copying object "${sourceKeyName}" from
"${sourceBucketName}". No such key exists.`
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while copying object from ${sourceBucketName}.
${caught.name}: ${caught.message}`
        );
    } else {
        throw caught;
    }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
    isMain,
    validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
    const options = {
        sourceBucketName: {
            type: "string",
            required: true,
        },
        sourceKeyName: {
            type: "string",
            required: true,
        },
        destinationBucketName: {
            type: "string",
            required: true,
        },
    };
};

const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
return { errors, results };
};

if (isMain(import.meta.url)) {
    const { errors, results } = loadArgs();
```

```
if (!errors) {
    main(results.values);
} else {
    console.error(errors.join("\n"));
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [CopyObject](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun copyBucketObject(
    fromBucket: String,
    objectKey: String,
    toBucket: String,
) {
    var encodedUrl = ""
    try {
        encodedUrl = URLEncoder.encode("$fromBucket/$objectKey",
StandardCharsets.UTF_8.toString())
    } catch (e: UnsupportedEncodingException) {
        println("URL could not be encoded: " + e.message)
    }

    val request =
        CopyObjectRequest {
            copySource = encodedUrl
            bucket = toBucket
            key = objectKey
        }
    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        s3.copyObject(request)
    }
}
```

```
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [CopyObject](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

簡單複製物件。

```
$s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

try {
    $folder = "copied-folder";
    $this->s3client->copyObject([
        'Bucket' => $this->bucketName,
        'CopySource' => "$this->bucketName/$fileName",
        'Key' => "$folder/$fileName-copy",
    ]);
    echo "Copied $fileName to $folder/$fileName-copy.\n";
} catch (Exception $exception) {
    echo "Failed to copy $fileName with error: " . $exception-
    >getMessage();
    exit("Please fix error with object copying before continuing.");
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [CopyObject](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會將物件"sample.txt" 從儲存貯體 "test-files" 複製到相同的儲存貯體，但具有新索引鍵 "sample-copy.txt"。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -DestinationKey sample-copy.txt
```

範例 2：此命令會將物件"sample.txt" 從儲存貯體 "test-files" 複製到儲存貯體 "backup-files"，索引鍵為 "sample-copy.txt"。

```
Copy-S3Object -BucketName amzn-s3-demo-source-bucket -Key sample.txt -DestinationKey sample-copy.txt -DestinationBucket amzn-s3-demo-destination-bucket
```

範例 3：此命令會將物件"sample.txt" 從儲存貯體 "test-files" 下載至名為 "local-sample.txt" 的本機檔案。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -LocalFile local-sample.txt
```

範例 4：將單一物件下載至指定的檔案。下載的檔案位於 c:\downloads\data\archive.zip

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key data/archive.zip -LocalFolder c:\downloads
```

範例 5：將符合指定索引鍵字首的所有物件下載至本機資料夾。相對索引鍵階層會保留為整體下載位置中的子資料夾。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix data -LocalFolder c:\downloads
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [CopyObject](#)。

Tools for PowerShell V5

範例 1：此命令會將物件"sample.txt" 從儲存貯體 "test-files" 複製到相同的儲存貯體，但具有新索引鍵 "sample-copy.txt"。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -DestinationKey  
sample-copy.txt
```

範例 2：此命令會將物件 "sample.txt" 從儲存貯體 "test-files" 複製到儲存貯體 "backup-files"，索引鍵為 "sample-copy.txt"。

```
Copy-S3Object -BucketName amzn-s3-demo-source-bucket -Key sample.txt -  
DestinationKey sample-copy.txt -DestinationBucket amzn-s3-demo-destination-bucket
```

範例 3：此命令會將物件 "sample.txt" 從儲存貯體 "test-files" 下載至名為 "local-sample.txt" 的本機檔案。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -LocalFile local-  
sample.txt
```

範例 4：將單一物件下載至指定的檔案。下載的檔案位於 c:\downloads\data\archive.zip

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key data/archive.zip -LocalFolder  
c:\downloads
```

範例 5：將符合指定索引鍵字首的所有物件下載至本機資料夾。相對索引鍵階層會保留為整體下載位置中的子資料夾。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix data -LocalFolder c:  
\downloads
```

範例 6：使用分段平行下載將單一物件下載至本機檔案，以提升大型檔案的效能。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -LocalFile local-  
sample.txt -UseMultipartDownload
```

範例 7：使用具有自訂 64MB 組件大小的 RANGE 模式下載大型物件。RANGE 模式使用 HTTP 位元組範圍請求，並適用於所有物件，無論它們的上傳方式為何。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key large-file.zip -LocalFile  
local-large-file.zip -UseMultipartDownload -MultipartDownloadType RANGE -  
PartSize 64MB
```

範例 8：使用分段下載搭配並行檔案下載，將符合金鑰字首的所有物件下載至本機資料夾。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix data/ -LocalFolder C:\local-data -UseMultipartDownload -DownloadFilesConcurrently
```

範例 9：使用具有 20 個並行連線的分段下載來下載單一物件，以在高頻寬網路上達到最大輸送量。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -Key large-file.zip -LocalFile local-large-file.zip -UseMultipartDownload -ConcurrentServiceRequest 20
```

範例 10：使用分段下載將所有符合金鑰字首的物件下載至本機資料夾，繼續超過個別檔案失敗，而不是中止。

```
Copy-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix data/ -LocalFolder C:\local-data -UseMultipartDownload -FailurePolicy ContinueOnFailure
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [CopyObject](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
        in Boto3
```

```

        """
        that wraps object actions in a class-like structure.

        """
        self.object = s3_object
        self.key = self.object.key

    def copy(self, dest_object):
        """
        Copies the object to another bucket.

        :param dest_object: The destination object initialized with a bucket and
        key.

        This is a Boto3 Object resource.

        """
        try:
            dest_object.copy_from(
                CopySource={"Bucket": self.object.bucket_name, "Key":
self.object.key}
            )
            dest_object.wait_until_exists()
            logger.info(
                "Copied object from %s:%s to %s:%s.",
                self.object.bucket_name,
                self.object.key,
                dest_object.bucket_name,
                dest_object.key,
            )
        except ClientError:
            logger.exception(
                "Couldn't copy object from %s/%s to %s/%s.",
                self.object.bucket_name,
                self.object.key,
                dest_object.bucket_name,
                dest_object.key,
            )
            raise

```

使用條件式請求複製物件。

```

class S3ConditionalRequests:
    """Encapsulates S3 conditional request operations."""

```

```
def __init__(self, s3_client):
    self.s3 = s3_client

@classmethod
def from_client(cls):
    """
    Instantiates this class from a Boto3 client.
    """
    s3_client = boto3.client("s3")
    return cls(s3_client)

def copy_object_conditional(
    self,
    source_key: str,
    dest_key: str,
    source_bucket: str,
    dest_bucket: str,
    condition_type: str,
    condition_value: str,
):
    """
    Copies an object from one Amazon S3 bucket to another with a conditional
    request.

    :param source_key: The key of the source object to copy.
    :param dest_key: The key of the destination object.
    :param source_bucket: The source bucket of the object.
    :param dest_bucket: The destination bucket of the object.
    :param condition_type: The type of condition to apply, e.g.
        'CopySourceIfMatch', 'CopySourceIfNoneMatch',
        'CopySourceIfModifiedSince', 'CopySourceIfUnmodifiedSince'.
    :param condition_value: The value to use for the condition.
    """
    try:
        self.s3.copy_object(
            Bucket=dest_bucket,
            Key=dest_key,
            CopySource={"Bucket": source_bucket, "Key": source_key},
            **{condition_type: condition_value},
        )
        print(
```

```

        f"\tConditional copy successful for key {dest_key} in bucket
{dest_bucket}."
    )
except ClientError as e:
    error_code = e.response["Error"]["Code"]
    if error_code == "PreconditionFailed":
        print("\tConditional copy failed: Precondition failed")
    elif error_code == "304": # Not modified error code.
        print("\tConditional copy failed: Object not modified")
    else:
        logger.error(f"Unexpected error: {error_code}")
        raise

```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [CopyObject](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

複製物件

```

require 'aws-sdk-s3'

# Wraps Amazon S3 object actions.
class ObjectCopyWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
  # used as the source object for
  #                                     copy actions.
  def initialize(source_object)
    @source_object = source_object
  end
end

```

```

# Copy the source object to the specified target bucket and rename it with the
target key.
#
# @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
object is copied.
# @param target_object_key [String] The key to give the copy of the object.
# @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
nil.
def copy_object(target_bucket, target_object_key)
  @source_object.copy_to(bucket: target_bucket.name, key: target_object_key)
  target_bucket.object(target_object_key)
rescue Aws::Errors::ServiceError => e
  puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's
why: #{e.message}"
end
end

# Example usage:
def run_demo
  source_bucket_name = "amzn-s3-demo-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "amzn-s3-demo-bucket2"
  target_key = "my-target-file.txt"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
#{target_object.bucket_name}:#{target_object.key}."
end

run_demo if $PROGRAM_NAME == __FILE__

```

複製物件，然後將伺服器端加密新增至目標物件。

```

require 'aws-sdk-s3'

# Wraps Amazon S3 object actions.

```

```
class ObjectCopyEncryptWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
  # used as the source object for
  #                                     copy actions.
  def initialize(source_object)
    @source_object = source_object
  end

  # Copy the source object to the specified target bucket, rename it with the
  # target key, and encrypt it.
  #
  # @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
  # object is copied.
  # @param target_object_key [String] The key to give the copy of the object.
  # @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
  # nil.
  def copy_object(target_bucket, target_object_key, encryption)
    @source_object.copy_to(bucket: target_bucket.name, key: target_object_key,
      server_side_encryption: encryption)
    target_bucket.object(target_object_key)
    rescue Aws::Errors::ServiceError => e
      puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's
      why: #{e.message}"
    end
  end
end

# Example usage:
def run_demo
  source_bucket_name = "amzn-s3-demo-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "amzn-s3-demo-bucket2"
  target_key = "my-target-file.txt"
  target_encryption = "AES256"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyEncryptWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key,
  target_encryption)
  return unless target_object
end
```



```
puts "Copied #{source_key} from #{source_bucket_name} to
#{target_object.bucket_name}:#{target_object.key} and "\
    "encrypted the target with #{target_object.server_side_encryption}
encryption."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [CopyObject](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// Copy an object from one bucket to another.
pub async fn copy_object(
    client: &aws_sdk_s3::Client,
    source_bucket: &str,
    destination_bucket: &str,
    source_object: &str,
    destination_object: &str,
) -> Result<(), S3ExampleError> {
    let source_key = format!("{source_bucket}/{source_object}");
    let response = client
        .copy_object()
        .copy_source(&source_key)
        .bucket(destination_bucket)
        .key(destination_object)
        .send()
        .await?;

    println!(
        "Copied from {source_key} to {destination_bucket}/{destination_object}
with etag {}",
        response
```

```
        .copy_object_result
        .unwrap_or_else(||
aws_sdk_s3::types::CopyObjectResult::builder().build())
        .e_tag()
        .unwrap_or("missing")
    );
    Ok(())
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [CopyObject](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.
    lo_s3->copyobject(
        iv_bucket = iv_dest_bucket
        iv_key = iv_dest_object
        iv_copysource = |{ iv_src_bucket }/{ iv_src_object }| ).
    MESSAGE 'Object copied to another bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 SAP ABAP 的 AWS SDK API 參考》中的 [CopyObject](#)。

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import AWSS3

public func copyFile(from sourceBucket: String, name: String, to destBucket:
String) async throws {
    let srcUrl = ("\"(sourceBucket)/
\"(name)").addingPercentEncoding(withAllowedCharacters: .urlPathAllowed)

    let input = CopyObjectInput(
        bucket: destBucket,
        copySource: srcUrl,
        key: name
    )
    do {
        _ = try await client.copyObject(input: input)
    }
    catch {
        print("ERROR: ", dump(error, name: "Copying an object."))
        throw error
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Swift 的 AWS SDK API 參考》中的 [CopyObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateBucket 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateBucket。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)
- [組態管理入門](#)
- [機器學習功能存放區的入門](#)
- [物件儲存入門](#)
- [使用 S3 管理大型訊息](#)
- [使用版本化物件](#)

.NET

適用於 .NET 的 SDK (v4)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Shows how to create a new Amazon S3 bucket.
/// </summary>
/// <param name="bucketName">The name of the bucket to create.</param>
/// <returns>A boolean value representing the success or failure of
/// the bucket creation process.</returns>
public async Task<bool> CreateBucketAsync(string bucketName)
{
    try
    {
        var request = new PutBucketRequest
        {
            BucketName = bucketName,
            UseClientRegion = true,
        };

        var response = await _amazonS3.PutBucketAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
}
```

```
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error creating bucket: '{ex.Message}'");
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [CreateBucket](#)。

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立已啟用物件鎖定的儲存貯體。

```
/// <summary>
/// Create a new Amazon S3 bucket with object lock actions.
/// </summary>
/// <param name="bucketName">The name of the bucket to create.</param>
/// <param name="enableObjectLock">True to enable object lock on the
bucket.</param>
/// <returns>True if successful.</returns>
public async Task<bool> CreateBucketWithObjectLock(string bucketName, bool
enableObjectLock)
{
    Console.WriteLine($"\\tCreating bucket {bucketName} with object lock
{enableObjectLock}.");
    try
    {
        var request = new PutBucketRequest
        {
            BucketName = bucketName,
            UseClientRegion = true,
            ObjectLockEnabledForBucket = enableObjectLock,
        };
    }
```

```

        var response = await _amazonS3.PutBucketAsync(request);

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error creating bucket: '{ex.Message}'");
        return false;
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [CreateBucket](#)。

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {

```

```

    printf "%s\n" "$*" 1>&2
}

#####
# function create-bucket
#
# This function creates the specified bucket in the specified AWS Region, unless
# it already exists.
#
# Parameters:
#     -b bucket_name  -- The name of the bucket to create.
#     -r region_code  -- The code for an AWS Region in which to
#                        create the bucket.
#
# Returns:
#     The URL of the bucket that was created.
#     And:
#     0 - If successful.
#     1 - If it fails.
#####
function create_bucket() {
    local bucket_name region_code response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function create_bucket"
        echo "Creates an Amazon S3 bucket. You must supply a bucket name:"
        echo "  -b bucket_name    The name of the bucket. It must be globally
unique."
        echo "  [-r region_code]   The code for an AWS Region in which the bucket is
created."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "b:r:h" option; do
        case "${option}" in
            b) bucket_name="${OPTARG}" ;;
            r) region_code="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
        esac
    done
}

```

```
\?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
esac
done

if [[ -z "$bucket_name" ]]; then
    errecho "ERROR: You must provide a bucket name with the -b parameter."
    usage
    return 1
fi

local bucket_config_arg
# A location constraint for "us-east-1" returns an error.
if [[ -n "$region_code" ]] && [[ "$region_code" != "us-east-1" ]]; then
    bucket_config_arg="--create-bucket-configuration LocationConstraint=
$region_code"
fi

iecho "Parameters:\n"
iecho "    Bucket name:  $bucket_name"
iecho "    Region code:  $region_code"
iecho ""

# If the bucket already exists, we don't want to try to create it.
if (bucket_exists "$bucket_name"); then
    errecho "ERROR: A bucket with that name already exists. Try again."
    return 1
fi

# shellcheck disable=SC2086
response=$(aws s3api create-bucket \
    --bucket "$bucket_name" \
    $bucket_config_arg)

# shellcheck disable=SC2181
if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports create-bucket operation failed.\n$response"
    return 1
fi
}
```


- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CreateBucket](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::createBucket(const Aws::String &bucketName,
                              const Aws::S3::S3ClientConfiguration &clientConfig)
{
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::CreateBucketRequest request;
    request.SetBucket(bucketName);

    if (clientConfig.region != "us-east-1") {
        Aws::S3::Model::CreateBucketConfiguration createBucketConfig;
        createBucketConfig.SetLocationConstraint(

        Aws::S3::Model::BucketLocationConstraintMapper::GetBucketLocationConstraintForName(
            clientConfig.region));
        request.SetCreateBucketConfiguration(createBucketConfig);
    }

    Aws::S3::Model::CreateBucketOutcome outcome = client.CreateBucket(request);
    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error: createBucket: " <<
            err.GetExceptionName() << ": " << err.GetMessage() <<
        std::endl;
    } else {
        std::cout << "Created bucket " << bucketName <<
            " in the specified AWS Region." << std::endl;
    }

    return outcome.IsSuccess();
}
```

```
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [CreateBucket](#)。

CLI

AWS CLI

範例 1：建立儲存貯體

以下 create-bucket 範例會建立名為 amzn-s3-demo-bucket 的儲存貯體：

```
aws s3api create-bucket \  
  --bucket amzn-s3-demo-bucket \  
  --region us-east-1
```

輸出：

```
{  
  "Location": "/amzn-s3-demo-bucket"  
}
```

如需詳細資訊，請參閱《Amazon S3 使用者指南》中的 [建立儲存貯體](#)。

範例 2：在擁有者強制執行的情況下建立儲存貯體

下列 create-bucket 範例會建立名為 amzn-s3-demo-bucket 的儲存貯體，而該儲存貯體則會使用 S3 物件擁有權的儲存貯體擁有者強制執行的設定。

```
aws s3api create-bucket \  
  --bucket amzn-s3-demo-bucket \  
  --region us-east-1 \  
  --object-ownership BucketOwnerEnforced
```

輸出：

```
{  
  "Location": "/amzn-s3-demo-bucket"  
}
```

如需詳細資訊，請參閱《Amazon S3 使用者指南》中的[控制物件的所有權並停用 ACL](#)。

範例 3：在 ``us-east-1`` 區域外建立儲存貯體

下列 create-bucket 範例會在 eu-west-1 區域中建立名為 amzn-s3-demo-bucket 的儲存貯體。us-east-1 以外的區域需要指定適當的 LocationConstraint，才能在所需區域中建立儲存貯體。

```
aws s3api create-bucket \  
  --bucket amzn-s3-demo-bucket \  
  --region eu-west-1 \  
  --create-bucket-configuration LocationConstraint=eu-west-1
```

輸出：

```
{  
  "Location": "http://amzn-s3-demo-bucket.s3.amazonaws.com/"  
}
```

如需詳細資訊，請參閱《Amazon S3 使用者指南》中的[建立儲存貯體](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CreateBucket](#)。

Go

SDK for Go V2

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用預設組態建立儲存貯體。

```
import (  
  "bytes"  
  "context"  
  "errors"  
  "fmt"
```

```
"io"
"log"
"os"
"time"

"github.com/aws/aws-sdk-go-v2/aws"
"github.com/aws/aws-sdk-go-v2/feature/s3/manager"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/aws/aws-sdk-go-v2/service/s3/types"
"github.com/aws/smithy-go"
)

// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
// actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
// bucket
// and object actions.
type BucketBasics struct {
    S3Client *s3.Client
}

// CreateBucket creates a bucket with the specified name in the specified Region.
func (basics BucketBasics) CreateBucket(ctx context.Context, name string, region
string) error {
    _, err := basics.S3Client.CreateBucket(ctx, &s3.CreateBucketInput{
        Bucket: aws.String(name),
        CreateBucketConfiguration: &types.CreateBucketConfiguration{
            LocationConstraint: types.BucketLocationConstraint(region),
        },
    })
    if err != nil {
        var owned *types.BucketAlreadyOwnedByYou
        var exists *types.BucketAlreadyExists
        if errors.As(err, &owned) {
            log.Printf("You already own bucket %s.\n", name)
            err = owned
        } else if errors.As(err, &exists) {
            log.Printf("Bucket %s already exists.\n", name)
            err = exists
        }
    } else {
```

```
err = s3.NewBucketExistsWaiter(basics.S3Client).Wait(
    ctx, &s3.HeadBucketInput{Bucket: aws.String(name)}, time.Minute)
if err != nil {
    log.Printf("Failed attempt to wait for bucket %s to exist.\n", name)
}
}
return err
}
```

建立具有物件鎖定的儲存貯體，並等待其出現。

```
import (
    "bytes"
    "context"
    "errors"
    "fmt"
    "log"
    "time"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"
    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/aws-sdk-go-v2/service/s3/types"
    "github.com/aws/smithy-go"
)

// S3Actions wraps S3 service actions.
type S3Actions struct {
    S3Client    *s3.Client
    S3Manager   *manager.Uploader
}

// CreateBucketWithLock creates a new S3 bucket with optional object locking
// enabled
// and waits for the bucket to exist before returning.
func (actor S3Actions) CreateBucketWithLock(ctx context.Context, bucket string,
    region string, enableObjectLock bool) (string, error) {
    input := &s3.CreateBucketInput{
```

```
Bucket: aws.String(bucket),
CreateBucketConfiguration: &types.CreateBucketConfiguration{
    LocationConstraint: types.BucketLocationConstraint(region),
},
}

if enableObjectLock {
    input.ObjectLockEnabledForBucket = aws.Bool(true)
}

_, err := actor.S3Client.CreateBucket(ctx, input)
if err != nil {
    var owned *types.BucketAlreadyOwnedByYou
    var exists *types.BucketAlreadyExists
    if errors.As(err, &owned) {
        log.Printf("You already own bucket %s.\n", bucket)
        err = owned
    } else if errors.As(err, &exists) {
        log.Printf("Bucket %s already exists.\n", bucket)
        err = exists
    }
} else {
    err = s3.NewBucketExistsWaiter(actor.S3Client).Wait(
        ctx, &s3.HeadBucketInput{Bucket: aws.String(bucket)}, time.Minute)
    if err != nil {
        log.Printf("Failed attempt to wait for bucket %s to exist.\n", bucket)
    }
}

return bucket, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [CreateBucket](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立儲存貯體。

```
/**
 * Creates an S3 bucket asynchronously.
 *
 * @param bucketName the name of the S3 bucket to create
 * @return a {@link CompletableFuture} that completes when the bucket is
 *         created and ready
 * @throws RuntimeException if there is a failure while creating the bucket
 */
public CompletableFuture<Void> createBucketAsync(String bucketName) {
    CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
        .bucket(bucketName)
        .build();

    CompletableFuture<CreateBucketResponse> response =
        getAsyncClient().createBucket(bucketRequest);
    return response.thenCompose(resp -> {
        S3AsyncWaiter s3Waiter = getAsyncClient().waiter();
        HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
            .bucket(bucketName)
            .build();

        CompletableFuture<WaiterResponse<HeadBucketResponse>>
        waiterResponseFuture =
            s3Waiter.waitUntilBucketExists(bucketRequestWait);
        return waiterResponseFuture.thenAccept(waiterResponse -> {
            waiterResponse.matched().response().ifPresent(headBucketResponse
        -> {
            logger.info(bucketName + " is ready");
        });
    });
});
```

```
    }).whenComplete((resp, ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to create bucket", ex);
        }
    });
}
```

建立已啟用物件鎖定的儲存貯體。

```
// Create a new Amazon S3 bucket with object lock options.
public void createBucketWithLockOptions(boolean enableObjectLock, String
bucketName) {
    S3Waiter s3Waiter = getClient().waiter();
    CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
        .bucket(bucketName)
        .objectLockEnabledForBucket(enableObjectLock)
        .build();

    getClient().createBucket(bucketRequest);
    HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
        .bucket(bucketName)
        .build();

    // Wait until the bucket is created and print out the response.
    s3Waiter.waitUntilBucketExists(bucketRequestWait);
    System.out.println(bucketName + " is ready");
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CreateBucket](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立儲存貯體。

```
import {
  BucketAlreadyExists,
  BucketAlreadyOwnedByYou,
  CreateBucketCommand,
  S3Client,
  waitUntilBucketExists,
} from "@aws-sdk/client-s3";

/**
 * Create an Amazon S3 bucket.
 * @param {{ bucketName: string }} config
 */
export const main = async ({ bucketName }) => {
  const client = new S3Client({});

  try {
    const { Location } = await client.send(
      new CreateBucketCommand({
        // The name of the bucket. Bucket names are unique and have several other
        // constraints.
        // See https://docs.aws.amazon.com/AmazonS3/latest/userguide/bucketnamingrules.html
        Bucket: bucketName,
      }),
    );
    await waitUntilBucketExists({ client }, { Bucket: bucketName });
    console.log(`Bucket created with location ${Location}`);
  } catch (caught) {
    if (caught instanceof BucketAlreadyExists) {
      console.error(
        `The bucket "${bucketName}" already exists in another AWS account. Bucket names must be globally unique.`,
      );
    }
    // WARNING: If you try to create a bucket in the North Virginia region,
    // and you already own a bucket in that region with the same name, this
    // error will not be thrown. Instead, the call will return successfully
    // and the ACL on that bucket will be reset.
    else if (caught instanceof BucketAlreadyOwnedByYou) {
      console.error(
        `The bucket "${bucketName}" already exists in this AWS account.`,
      );
    }
  }
}
```

```
    } else {  
        throw caught;  
    }  
}  
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [CreateBucket](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun createNewBucket(bucketName: String) {  
    val request =  
        CreateBucketRequest {  
            bucket = bucketName  
        }  
  
    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->  
        s3.createBucket(request)  
        println("$bucketName is ready")  
    }  
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [CreateBucket](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立儲存貯體。

```
$s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

try {
    $this->s3client->createBucket([
        'Bucket' => $this->bucketName,
        'CreateBucketConfiguration' => ['LocationConstraint' => $region],
    ]);
    echo "Created bucket named: $this->bucketName \n";
} catch (Exception $exception) {
    echo "Failed to create bucket $this->bucketName with error: " .
    $exception->getMessage();
    exit("Please fix error with bucket creation before continuing.");
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [CreateBucket](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用預設設定建立儲存貯體。

```

class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def create(self, region_override=None):
        """
        Create an Amazon S3 bucket in the default Region for the account or in
        the
        specified Region.

        :param region_override: The Region in which to create the bucket. If this
        is
                                not specified, the Region configured in your
        shared
                                credentials is used.
        """
        if region_override is not None:
            region = region_override
        else:
            region = self.bucket.meta.client.meta.region_name
        try:
            self.bucket.create(CreateBucketConfiguration={"LocationConstraint":
            region})

            self.bucket.wait_until_exists()
            logger.info("Created bucket '%s' in region=%s", self.bucket.name,
            region)
        except ClientError as error:
            logger.exception(
                "Couldn't create bucket named '%s' in region=%s.",
                self.bucket.name,
                region,
            )
            raise error

```

透過生命週期組態建立版本控制儲存貯體。

```
def create_versioned_bucket(bucket_name, prefix):
    """
    Creates an Amazon S3 bucket, enables it for versioning, and configures a
    lifecycle
    that expires noncurrent object versions after 7 days.

    Adding a lifecycle configuration to a versioned bucket is a best practice.
    It helps prevent objects in the bucket from accumulating a large number of
    noncurrent versions, which can slow down request performance.

    Usage is shown in the usage_demo_single_object function at the end of this
    module.

    :param bucket_name: The name of the bucket to create.
    :param prefix: Identifies which objects are automatically expired under the
                    configured lifecycle rules.
    :return: The newly created bucket.
    """
    try:
        bucket = s3.create_bucket(
            Bucket=bucket_name,
            CreateBucketConfiguration={
                "LocationConstraint": s3.meta.client.meta.region_name
            },
        )
        logger.info("Created bucket %s.", bucket.name)
    except ClientError as error:
        if error.response["Error"]["Code"] == "BucketAlreadyOwnedByYou":
            logger.warning("Bucket %s already exists! Using it.", bucket_name)
            bucket = s3.Bucket(bucket_name)
        else:
            logger.exception("Couldn't create bucket %s.", bucket_name)
            raise

    try:
        bucket.Versioning().enable()
        logger.info("Enabled versioning on bucket %s.", bucket.name)
    except ClientError:
```

```
        logger.exception("Couldn't enable versioning on bucket %s.", bucket.name)
        raise

    try:
        expiration = 7
        bucket.LifecycleConfiguration().put(
            LifecycleConfiguration={
                "Rules": [
                    {
                        "Status": "Enabled",
                        "Prefix": prefix,
                        "NoncurrentVersionExpiration": {"NoncurrentDays":
expiration},
                    }
                ]
            }
        )
        logger.info(
            "Configured lifecycle to expire noncurrent versions after %s days "
            "on bucket %s.",
            expiration,
            bucket.name,
        )
    except ClientError as error:
        logger.warning(
            "Couldn't configure lifecycle on bucket %s because %s. "
            "Continuing anyway.",
            bucket.name,
            error,
        )

    return bucket
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [CreateBucket](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-s3'

# Wraps Amazon S3 bucket actions.
class BucketCreateWrapper
  attr_reader :bucket

  # @param bucket [Aws::S3::Bucket] An Amazon S3 bucket initialized with a name.
  # This is a client-side object until
  #                               create is called.
  def initialize(bucket)
    @bucket = bucket
  end

  # Creates an Amazon S3 bucket in the specified AWS Region.
  #
  # @param region [String] The Region where the bucket is created.
  # @return [Boolean] True when the bucket is created; otherwise, false.
  def create?(region)
    @bucket.create(create_bucket_configuration: { location_constraint: region })
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't create bucket. Here's why: #{e.message}"
    false
  end

  # Gets the Region where the bucket is located.
  #
  # @return [String] The location of the bucket.
  def location
    if @bucket.nil?
      'None. You must create a bucket before you can get its location!'
    else

```

```

        @bucket.client.get_bucket_location(bucket:
@bucket.name).location_constraint
      end
      rescue Aws::Errors::ServiceError => e
        "Couldn't get the location of #{@bucket.name}. Here's why: #{e.message}"
      end
    end
  end

# Example usage:
def run_demo
  region = "us-west-2"
  wrapper = BucketCreateWrapper.new(Aws::S3::Bucket.new("amzn-s3-demo-bucket-
#{@Random.uuid}"))
  return unless wrapper.create?(region)

  puts "Created bucket #{@wrapper.bucket.name}."
  puts "Your bucket's region is: #{@wrapper.location}"
end

run_demo if $PROGRAM_NAME == __FILE__

```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [CreateBucket](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

pub async fn create_bucket(
  client: &aws_sdk_s3::Client,
  bucket_name: &str,
  region: &aws_config::Region,
) -> Result<Option<aws_sdk_s3::operation::create_bucket::CreateBucketOutput>,
S3ExampleError> {
  let constraint =
aws_sdk_s3::types::BucketLocationConstraint::from(region.to_string().as_str());

```



```

let cfg = aws_sdk_s3::types::CreateBucketConfiguration::builder()
    .location_constraint(constraint)
    .build();
let create = client
    .create_bucket()
    .create_bucket_configuration(cfg)
    .bucket(bucket_name)
    .send()
    .await;

// BucketAlreadyExists and BucketAlreadyOwnedByYou are not problems for this
task.
create.map(Some).or_else(|err| {
    if err
        .as_service_error()
        .map(|se| se.is_bucket_already_exists() ||
se.is_bucket_already_owned_by_you())
        == Some(true)
    {
        Ok(None)
    } else {
        Err(S3ExampleError::from(err))
    }
}))
}

```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [CreateBucket](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " determine our region from our session

```

```

DATA(lv_region) = CONV /aws1/s3_bucketlocationcnstrnt( lo_session-
>get_region( ) ).
DATA lo_constraint TYPE REF TO /aws1/cl_s3_createbucketconf.
" When in the us-east-1 region, you must not specify a constraint
" In all other regions, specify the region as the constraint
IF lv_region = 'us-east-1'.
    CLEAR lo_constraint.
ELSE.
    lo_constraint = NEW /aws1/cl_s3_createbucketconf( lv_region ).
ENDIF.

lo_s3->createbucket(
    iv_bucket = iv_bucket_name
    io_createbucketconfiguration = lo_constraint ).
MESSAGE 'S3 bucket created.' TYPE 'I'.
CATCH /aws1/cx_s3_bucketalrdyexists.
    MESSAGE 'Bucket name already exists.' TYPE 'E'.
CATCH /aws1/cx_s3_bktalrdyownedbyyou.
    MESSAGE 'Bucket already exists and is owned by you.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 SAP ABAP 的 AWS SDK API 參考》中的 [CreateBucket](#)。

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

import AWSS3

public func createBucket(name: String) async throws {
    var input = CreateBucketInput(
        bucket: name
    )
}

```

```
// For regions other than "us-east-1", you must set the
locationConstraint in the createBucketConfiguration.
// For more information, see LocationConstraint in the S3 API guide.
// https://docs.aws.amazon.com/AmazonS3/latest/API/
API_CreateBucket.html#API_CreateBucket_RequestBody
if let region = configuration.region {
    if region != "us-east-1" {
        input.createBucketConfiguration =
S3ClientTypes.CreateBucketConfiguration(locationConstraint:
S3ClientTypes.BucketLocationConstraint(rawValue: region))
    }
}

do {
    _ = try await client.createBucket(input: input)
}
catch let error as BucketAlreadyOwnedByYou {
    print("The bucket '\(name)' already exists and is owned by you. You
may wish to ignore this exception.")
    throw error
}
catch {
    print("ERROR: ", dump(error, name: "Creating a bucket"))
    throw error
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 Swift 的 AWS SDK API 參考》中的 [CreateBucket](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateMultiRegionAccessPoint 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 CreateMultiRegionAccessPoint。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

設定 S3 控制用戶端，將請求傳送至 us-west-2 區域。

```
suspend fun createS3ControlClient(): S3ControlClient {
    // Configure your S3ControlClient to send requests to US West
    (Oregon).
    val s3Control = S3ControlClient.fromEnvironment {
        region = "us-west-2"
    }
    return s3Control
}
```

建立多區域存取點。

```
suspend fun createMrap(
    s3Control: S3ControlClient,
    accountIdParam: String,
    bucketName1: String,
    bucketName2: String,
    mrapName: String,
): String {
    println("Creating MRAP ...")
    val createMrapResponse: CreateMultiRegionAccessPointResponse =
        s3Control.createMultiRegionAccessPoint {
            accountId = accountIdParam
            clientToken = UUID.randomUUID().toString()
            details {
                name = mrapName
                regions = listOf(
                    Region {
                        bucket = bucketName1
                    },
                ),
            }
        }
    return createMrapResponse.accessPointArn
}
```

```

                Region {
                    bucket = bucketName2
                },
            )
        }
    }
    val requestToken: String? = createMrapResponse.requestTokenArn

    // Use the request token to check for the status of the
    CreateMultiRegionAccessPoint operation.
    if (requestToken != null) {
        waitForSucceededStatus(s3Control, requestToken, accountIdParam)
        println("MRAP created")
    }

    val getMrapResponse =
        s3Control.getMultiRegionAccessPoint(
            input = GetMultiRegionAccessPointRequest {
                accountId = accountIdParam
                name = mrapName
            },
        )
    val mrapAlias = getMrapResponse.accessPoint?.alias
    return "arn:aws:s3:::$accountIdParam:accesspoint/$mrpAlias"
}

```

等待多區域存取點變成可用。

```

suspend fun waitForSucceededStatus(
    s3Control: S3ControlClient,
    requestToken: String,
    accountIdParam: String,
    timeBetweenChecks: Duration = 1.minutes,
) {
    var describeResponse: DescribeMultiRegionAccessPointOperationResponse
    describeResponse = s3Control.describeMultiRegionAccessPointOperation(
        input = DescribeMultiRegionAccessPointOperationRequest {
            accountId = accountIdParam
            requestTokenArn = requestToken
        },
    )
}

```

```
var status: String? = describeResponse.asyncOperation?.requestStatus
while (status != "SUCCEEDED") {
    delay(timeBetweenChecks)
    describeResponse =
s3Control.describeMultiRegionAccessPointOperation(
    input = DescribeMultiRegionAccessPointOperationRequest {
        accountId = accountIdParam
        requestTokenArn = requestToken
    },
)
    status = describeResponse.asyncOperation?.requestStatus
    println(status)
}
}
```

- 如需詳細資訊，請參閱[適用於 Kotlin 的 AWS SDK 開發人員指南](#)。
- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的[CreateMultiRegionAccessPoint](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱[使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateMultipartUpload 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateMultipartUpload。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [執行分段複製](#)
- [使用檢查總和](#)
- [使用 Amazon S3 物件完整性](#)

C++

適用於 C++ 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
//! Create a multipart upload.
/*!
    \param bucket: The name of the S3 bucket where the object will be uploaded.
    \param key: The unique identifier (key) for the object within the S3 bucket.
    \param client: The S3 client instance used to perform the upload operation.
    \return Aws::String: Upload ID or empty string if failed.
*/
Aws::String
AwsDoc::S3::createMultipartUpload(const Aws::String &bucket, const Aws::String
    &key,
                                Aws::S3::Model::ChecksumAlgorithm
checksumAlgorithm,
                                const Aws::S3::S3Client &client) {
    Aws::S3::Model::CreateMultipartUploadRequest request;
    request.SetBucket(bucket);
    request.SetKey(key);

    if (checksumAlgorithm != Aws::S3::Model::ChecksumAlgorithm::NOT_SET) {
        request.SetChecksumAlgorithm(checksumAlgorithm);
    }

    Aws::S3::Model::CreateMultipartUploadOutcome outcome =
        client.CreateMultipartUpload(request);

    Aws::String uploadID;
    if (outcome.IsSuccess()) {
        uploadID = outcome.GetResult().GetUploadId();
    } else {
        std::cerr << "Error creating multipart upload: " <<
outcome.GetError().GetMessage() << std::endl;
    }
}
```

```
    return uploadID;
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [CreateMultipartUpload](#)。

CLI

AWS CLI

下列命令會使用金鑰 `multipart/01` 在儲存貯體 `amzn-s3-demo-bucket` 中建立分段上傳：

```
aws s3api create-multipart-upload --bucket amzn-s3-demo-bucket --key
'multipart/01'
```

輸出：

```
{
  "Bucket": "amzn-s3-demo-bucket",
  "UploadId":
  "dfRtDYU0WWCCcH43C3WFbkR0NycyCpTJJvxu2i5GYkZ1jF.Yxwh6XG7WfS2vC4to6HiV6Yj1x.cph0gtNBtJ8P3
  "Key": "multipart/01"
}
```

完成的檔案會在儲存貯體 `amzn-s3-demo-bucket` 中名為 `multipart` 的資料夾中命名為 `01`。儲存要與 `upload-part` 命令搭配使用的上傳 ID、金鑰和儲存貯體名稱。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CreateMultipartUpload](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。


```
// Create a multipart upload. Use UploadPart and CompleteMultipartUpload to
// upload the file.
let multipart_upload_res: CreateMultipartUploadOutput = client
    .create_multipart_upload()
    .bucket(&bucket_name)
    .key(&key)
    .send()
    .await?;

let upload_id = multipart_upload_res.upload_id().ok_or(S3ExampleError::new(
    "Missing upload_id after CreateMultipartUpload",
))?;
```

```
let mut upload_parts: Vec<aws_sdk_s3::types::CompletedPart> = Vec::new();

for chunk_index in 0..chunk_count {
    let this_chunk = if chunk_count - 1 == chunk_index {
        size_of_last_chunk
    } else {
        CHUNK_SIZE
    };
    let stream = ByteStream::read_from()
        .path(path)
        .offset(chunk_index * CHUNK_SIZE)
        .length(Length::Exact(this_chunk))
        .build()
        .await
        .unwrap();

    // Chunk index needs to start at 0, but part numbers start at 1.
    let part_number = (chunk_index as i32) + 1;
    let upload_part_res = client
        .upload_part()
        .key(&key)
        .bucket(&bucket_name)
        .upload_id(upload_id)
        .body(stream)
        .part_number(part_number)
        .send()
        .await?;

    upload_parts.push(
```

```
        CompletedPart::builder()
            .e_tag(upload_part_res.e_tag.unwrap_or_default())
            .part_number(part_number)
            .build(),
    );
}
```

```
// upload_parts: Vec<aws_sdk_s3::types::CompletedPart>
let completed_multipart_upload: CompletedMultipartUpload =
    CompletedMultipartUpload::builder()
        .set_parts(Some(upload_parts))
        .build();

let _complete_multipart_upload_res = client
    .complete_multipart_upload()
    .bucket(&bucket_name)
    .key(&key)
    .multipart_upload(completed_multipart_upload)
    .upload_id(upload_id)
    .send()
    .await?;
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [CreateMultipartUpload](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreatePresignedPost 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 CreatePresignedPost。

.NET

適用於 .NET 的 SDK (v4)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立預先簽章的 POST URL。

```
/// <summary>
/// Create a presigned POST URL with conditions.
/// </summary>
/// <param name="s3Client">The Amazon S3 client.</param>
/// <param name="bucketName">The name of the bucket.</param>
/// <param name="objectKey">The object key (path) where the uploaded file
will be stored.</param>
/// <param name="expires">When the presigned URL expires.</param>
/// <param name="fields">Dictionary of fields to add to the form.</param>
/// <param name="conditions">List of conditions to apply.</param>
/// <returns>A CreatePresignedPostResponse object with URL and form fields.</
returns>
public async Task<CreatePresignedPostResponse> CreatePresignedPostAsync(
    IAmazonS3 s3Client,
    string bucketName,
    string objectKey,
    DateTime expires,
    Dictionary<string, string>? fields = null,
    List<S3PostCondition>? conditions = null)
{
    var request = new CreatePresignedPostRequest
    {
        BucketName = bucketName,
        Key = objectKey,
        Expires = expires
    };

    // Add custom fields if provided
    if (fields != null)
    {
        foreach (var field in fields)
```

```
        {
            request.Fields.Add(field.Key, field.Value);
        }
    }

    // Add conditions if provided
    if (conditions != null)
    {
        foreach (var condition in conditions)
        {
            request.Conditions.Add(condition);
        }
    }

    return await s3Client.CreatePresignedPostAsync(request);
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [CreatePresignedPost](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteBucket 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteBucket。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)
- [組態管理入門](#)
- [機器學習功能存放區的入門](#)
- [物件儲存入門](#)

.NET

適用於 .NET 的 SDK (v4)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Shows how to delete an Amazon S3 bucket.
/// </summary>
/// <param name="bucketName">The name of the Amazon S3 bucket to delete.</
param>
/// <returns>A boolean value that represents the success or failure of
/// the delete operation.</returns>
public async Task<bool> DeleteBucketAsync(string bucketName)
{
    try
    {
        var request = new DeleteBucketRequest { BucketName = bucketName, };

        await _amazonS3.DeleteBucketAsync(request);
        return true;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error deleting bucket: {ex.Message}");
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [DeleteBucket](#)。

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function delete_bucket
#
# This function deletes the specified bucket.
#
# Parameters:
#     $1 - The name of the bucket.

# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function delete_bucket() {
    local bucket_name=$1
    local response

    response=$(aws s3api delete-bucket \
        --bucket "$bucket_name")

    # shellcheck disable=SC2181
    if [[ $? -ne 0 ]]; then
        errecho "ERROR: AWS reports s3api delete-bucket failed.\n$response"
        return 1
    fi
}
```

```
fi  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucket](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::deleteBucket(const Aws::String &bucketName,  
                               const Aws::S3::S3ClientConfiguration &clientConfig)  
{  
  
    Aws::S3::S3Client client(clientConfig);  
  
    Aws::S3::Model::DeleteBucketRequest request;  
    request.SetBucket(bucketName);  
  
    Aws::S3::Model::DeleteBucketOutcome outcome =  
        client.DeleteBucket(request);  
  
    if (!outcome.IsSuccess()) {  
        const Aws::S3::S3Error &err = outcome.GetError();  
        std::cerr << "Error: deleteBucket: " <<  
            err.GetExceptionName() << ": " << err.GetMessage() <<  
std::endl;  
    } else {  
        std::cout << "The bucket was deleted" << std::endl;  
    }  
  
    return outcome.IsSuccess();  
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [DeleteBucket](#)。

CLI

AWS CLI

以下命令會刪除一個名為 `amzn-s3-demo-bucket` 的儲存貯體：

```
aws s3api delete-bucket --bucket amzn-s3-demo-bucket --region us-east-1
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucket](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (  
    "bytes"  
    "context"  
    "errors"  
    "fmt"  
    "io"  
    "log"  
    "os"  
    "time"  
  
    "github.com/aws/aws-sdk-go-v2/aws"  
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"  
    "github.com/aws/aws-sdk-go-v2/service/s3"  
    "github.com/aws/aws-sdk-go-v2/service/s3/types"  
    "github.com/aws/smithy-go"  
)
```



```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
// actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
// bucket
// and object actions.
type BucketBasics struct {
    S3Client *s3.Client
}

// DeleteBucket deletes a bucket. The bucket must be empty or an error is
// returned.
func (basics BucketBasics) DeleteBucket(ctx context.Context, bucketName string)
error {
    _, err := basics.S3Client.DeleteBucket(ctx, &s3.DeleteBucketInput{
        Bucket: aws.String(bucketName)})
    if err != nil {
        var noBucket *types.NoSuchBucket
        if errors.As(err, &noBucket) {
            log.Printf("Bucket %s does not exist.\n", bucketName)
            err = noBucket
        } else {
            log.Printf("Couldn't delete bucket %v. Here's why: %v\n", bucketName, err)
        }
    } else {
        err = s3.NewBucketNotExistsWaiter(basics.S3Client).Wait(
            ctx, &s3.HeadBucketInput{Bucket: aws.String(bucketName)}, time.Minute)
        if err != nil {
            log.Printf("Failed attempt to wait for bucket %s to be deleted.\n",
                bucketName)
        } else {
            log.Printf("Deleted %s.\n", bucketName)
        }
    }
    return err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [DeleteBucket](#)。

Java

SDK for Java 2.x

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Deletes an S3 bucket asynchronously.
 *
 * @param bucket the name of the bucket to be deleted
 * @return a {@link CompletableFuture} that completes when the bucket
 * deletion is successful, or throws a {@link RuntimeException}
 * if an error occurs during the deletion process
 */
public CompletableFuture<Void> deleteBucketAsync(String bucket) {
    DeleteBucketRequest deleteBucketRequest = DeleteBucketRequest.builder()
        .bucket(bucket)
        .build();

    CompletableFuture<DeleteBucketResponse> response =
getAsyncClient().deleteBucket(deleteBucketRequest);
    response.whenComplete((deleteRes, ex) -> {
        if (deleteRes != null) {
            logger.info(bucket + " was deleted.");
        } else {
            throw new RuntimeException("An S3 exception occurred during
bucket deletion", ex);
        }
    });
    return response.thenApply(r -> null);
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteBucket](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除儲存貯體。

```
import {
  DeleteBucketCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Delete an Amazon S3 bucket.
 * @param {{ bucketName: string }}
 */
export const main = async ({ bucketName }) => {
  const client = new S3Client({});
  const command = new DeleteBucketCommand({
    Bucket: bucketName,
  });

  try {
    await client.send(command);
    console.log("Bucket was deleted.");
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
      console.error(
        `Error from S3 while deleting bucket. The bucket doesn't exist.`
      );
    } else if (caught instanceof S3ServiceException) {
      console.error(
        `Error from S3 while deleting the bucket. ${caught.name}: ${caught.message}`
      );
    }
  }
}
```

```
    );  
  } else {  
    throw caught;  
  }  
}  
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [DeleteBucket](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除空的儲存貯體。

```
$s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);  
  
try {  
    $this->s3client->deleteBucket([  
        'Bucket' => $this->bucketName,  
    ]);  
    echo "Deleted bucket $this->bucketName.\n";  
} catch (Exception $exception) {  
    echo "Failed to delete $this->bucketName with error: " . $exception-  
>getMessage();  
    exit("Please fix error with bucket deletion before continuing.");  
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [DeleteBucket](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會從儲存貯體 'test-files' 移除所有物件和物件版本，然後刪除儲存貯體。繼續之前，此命令會先提示確認。新增 - 強制切換以隱藏確認。請注意，不是空的儲存貯體無法刪除。

```
Remove-S3Bucket -BucketName amzn-s3-demo-bucket -DeleteBucketContent
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteBucket](#)。

Tools for PowerShell V5

範例 1：此命令會從儲存貯體 'test-files' 移除所有物件和物件版本，然後刪除儲存貯體。繼續之前，此命令會先提示確認。新增 - 強制切換以隱藏確認。請注意，不是空的儲存貯體無法刪除。

```
Remove-S3Bucket -BucketName amzn-s3-demo-bucket -DeleteBucketContent
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteBucket](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
```

```

        """
        self.bucket = bucket
        self.name = bucket.name

    def delete(self):
        """
        Delete the bucket. The bucket must be empty or an error is raised.
        """
        try:
            self.bucket.delete()
            self.bucket.wait_until_not_exists()
            logger.info("Bucket %s successfully deleted.", self.bucket.name)
        except ClientError:
            logger.exception("Couldn't delete bucket %s.", self.bucket.name)
            raise

```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [DeleteBucket](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

# Deletes the objects in an Amazon S3 bucket and deletes the bucket.
#
# @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
def delete_bucket(bucket)
  puts("\nDo you want to delete all of the objects as well as the bucket (y/n)?")
  answer = gets.chomp.downcase
  if answer == 'y'
    bucket.objects.batch_delete!
  end
end

```

```

    bucket.delete
    puts("Emptied and deleted bucket #{bucket.name}.\n")
  end
rescue Aws::Errors::ServiceError => e
  puts("Couldn't empty and delete bucket #{bucket.name}.")
  puts("\t#{e.code}: #{e.message}")
  raise
end

```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [DeleteBucket](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

pub async fn delete_bucket(
    client: &aws_sdk_s3::Client,
    bucket_name: &str,
) -> Result<(), S3ExampleError> {
    let resp = client.delete_bucket().bucket(bucket_name).send().await;
    match resp {
        Ok(_) => Ok(()),
        Err(err) => {
            if err
                .as_service_error()
                .and_then(aws_sdk_s3::error::ProvideErrorMetadata::code)
                == Some("NoSuchBucket")
            {
                Ok(())
            } else {
                Err(S3ExampleError::from(err))
            }
        }
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [DeleteBucket](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.
```

```
    lo_s3->deletebucket(  
        iv_bucket = iv_bucket_name ).  
    MESSAGE 'Deleted S3 bucket.' TYPE 'I'.  
    CATCH /aws1/cx_s3_nosuchbucket.  
        MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 SAP ABAP 的 AWS SDK API 參考》中的 [DeleteBucket](#)。

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import AWSS3
```



```
public func deleteBucket(name: String) async throws {
    let input = DeleteBucketInput(
        bucket: name
    )
    do {
        _ = try await client.deleteBucket(input: input)
    }
    catch {
        print("ERROR: ", dump(error, name: "Deleting a bucket"))
        throw error
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Swift 的 AWS SDK API 參考》中的 [DeleteBucket](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteBucketAnalyticsConfiguration** 與 CLI

下列程式碼範例示範如何使用 DeleteBucketAnalyticsConfiguration。

CLI

AWS CLI

刪除儲存貯體的分析組態

下列 delete-bucket-analytics-configuration 範例會移除指定儲存貯體和 ID 的分析組態。

```
aws s3api delete-bucket-analytics-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 1
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucketAnalyticsConfiguration](#)。

PowerShell

Tools for PowerShell V4

範例 1：命令會移除給定 S3 儲存貯體中名為 'testfilter' 的分析篩選條件。

```
Remove-S3BucketAnalyticsConfiguration -BucketName 'amzn-s3-demo-bucket' -  
AnalyticsId 'testfilter'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteBucketAnalyticsConfiguration](#)。

Tools for PowerShell V5

範例 1：命令會移除給定 S3 儲存貯體中名為 'testfilter' 的分析篩選條件。

```
Remove-S3BucketAnalyticsConfiguration -BucketName 'amzn-s3-demo-bucket' -  
AnalyticsId 'testfilter'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteBucketAnalyticsConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteBucketCors 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteBucketCors。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Deletes a CORS configuration from an Amazon S3 bucket.
/// </summary>
/// <param name="client">The initialized Amazon S3 client object used
/// to delete the CORS configuration from the bucket.</param>
private static async Task DeleteCORSConfigurationAsync(AmazonS3Client
client)
{
    DeleteCORSConfigurationRequest request = new
DeleteCORSConfigurationRequest()
    {
        BucketName = BucketName,
    };
    await client.DeleteCORSConfigurationAsync(request);
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [DeleteBucketCors](#)。

CLI

AWS CLI

下列命令會從名為 `amzn-s3-demo-bucket` 的儲存貯體中刪除跨來源資源共享組態：

```
aws s3api delete-bucket-cors --bucket amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucketCors](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def delete_cors(self):
        """
        Delete the CORS rules from the bucket.

        :param bucket_name: The name of the bucket to update.
        """
        try:
            self.bucket.Cors().delete()
            logger.info("Deleted CORS from bucket '%s'.", self.bucket.name)
        except ClientError:
            logger.exception("Couldn't delete CORS from bucket '%s'.",
self.bucket.name)
            raise
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [DeleteBucketCors](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-s3'

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors

  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with
  an existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Deletes the CORS configuration of a bucket.
  #
  # @return [Boolean] True if the CORS rules were deleted; otherwise, false.
  def delete_cors
    @bucket_cors.delete
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't delete CORS rules for #{@bucket_cors.bucket.name}. Here's why:
    #{e.message}"
    false
  end
end

end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [DeleteBucketCors](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

TRY.

```
lo_s3->deletebucketcors(  
    iv_bucket = iv_bucket_name ).  
    MESSAGE 'Bucket CORS configuration deleted.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [DeleteBucketCors](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteBucketEncryption** 與 CLI

下列程式碼範例示範如何使用 DeleteBucketEncryption。

CLI

AWS CLI

刪除儲存貯體的伺服器端加密組態

下列 delete-bucket-encryption 範例會刪除指定儲存貯體的伺服器端加密組態。

```
aws s3api delete-bucket-encryption \  
    --bucket amzn-s3-demo-bucket
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucketEncryption](#)。

PowerShell

Tools for PowerShell V4

範例 1：這會停用針對提供的 S3 儲存貯體啟用的加密。

```
Remove-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketEncryption (DeleteBucketEncryption)" on
target "s3casetestbucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteBucketEncryption](#)。

Tools for PowerShell V5

範例 1：這會停用針對提供的 S3 儲存貯體啟用的加密。

```
Remove-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketEncryption (DeleteBucketEncryption)" on
target "s3casetestbucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteBucketEncryption](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteBucketInventoryConfiguration** 與 CLI

下列程式碼範例示範如何使用 DeleteBucketInventoryConfiguration。

CLI

AWS CLI

刪除儲存貯體的庫存組態

下列 `delete-bucket-inventory-configuration` 範例會刪除具有指定儲存貯體 ID 1 的庫存組態。

```
aws s3api delete-bucket-inventory-configuration \  
  --bucket amzn-s3-demo-bucket \  
  --id 1
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucketInventoryConfiguration](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會移除與給定 S3 儲存貯體對應之名為 'testInventoryName' 的庫存。

```
Remove-S3BucketInventoryConfiguration -BucketName 'amzn-s3-demo-bucket' -  
InventoryId 'testInventoryName'
```

輸出：

```
Confirm  
Are you sure you want to perform this action?  
Performing the operation "Remove-S3BucketInventoryConfiguration  
(DeleteBucketInventoryConfiguration)" on target "amzn-s3-demo-bucket".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): Y
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteBucketInventoryConfiguration](#)。

Tools for PowerShell V5

範例 1：此命令會移除與給定 S3 儲存貯體對應之名為 'testInventoryName' 的庫存。

```
Remove-S3BucketInventoryConfiguration -BucketName 'amzn-s3-demo-bucket' -  
InventoryId 'testInventoryName'
```

輸出：

Confirm

Are you sure you want to perform this action?

Performing the operation "Remove-S3BucketInventoryConfiguration

(DeleteBucketInventoryConfiguration)" on target "amzn-s3-demo-bucket".

[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): Y

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteBucketInventoryConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteBucketLifecycle 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteBucketLifecycle。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// This method removes the Lifecycle configuration from the named
/// S3 bucket.
/// </summary>
/// <param name="client">The S3 client object used to call
/// the RemoveLifecycleConfigAsync method.</param>
/// <param name="bucketName">A string representing the name of the
/// S3 bucket from which the configuration will be removed.</param>
public static async Task RemoveLifecycleConfigAsync(IAmazonS3 client,
string bucketName)
{
    var request = new DeleteLifecycleConfigurationRequest()
    {
```

```
        BucketName = bucketName,
    };
    await client.DeleteLifecycleConfigurationAsync(request);
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [DeleteBucketLifecycle](#)。

CLI

AWS CLI

下列命令會從名為 `amzn-s3-demo-bucket` 的儲存貯體中刪除生命週期組態：

```
aws s3api delete-bucket-lifecycle --bucket amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucketLifecycle](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
```

```
self.name = bucket.name

def delete_lifecycle_configuration(self):
    """
    Remove the lifecycle configuration from the specified bucket.
    """
    try:
        self.bucket.LifecycleConfiguration().delete()
        logger.info(
            "Deleted lifecycle configuration for bucket '%s'.",
            self.bucket.name
        )
    except ClientError:
        logger.exception(
            "Couldn't delete lifecycle configuration for bucket '%s'.",
            self.bucket.name,
        )
        raise
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [DeleteBucketLifecycle](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

TRY.

```
lo_s3->deletebucketlifecycle(
    iv_bucket = iv_bucket_name ).
MESSAGE 'Bucket lifecycle configuration deleted.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
MESSAGE 'Bucket does not exist.' TYPE 'E'.
```

```
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [DeleteBucketLifecycle](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteBucketMetricsConfiguration** 與 CLI

下列程式碼範例示範如何使用 DeleteBucketMetricsConfiguration。

CLI

AWS CLI

刪除儲存貯體的指標組態

下列 delete-bucket-metrics-configuration 範例會移除指定儲存貯體和 ID 的指標組態。

```
aws s3api delete-bucket-metrics-configuration \  
  --bucket amzn-s3-demo-bucket \  
  --id 123
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucketMetricsConfiguration](#)。

PowerShell

Tools for PowerShell V4

範例 1：命令會移除給定 S3 儲存貯體中名為 'testmetrics' 的指標篩選條件。

```
Remove-S3BucketMetricsConfiguration -BucketName 'amzn-s3-demo-bucket' -MetricsId  
'testmetrics'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteBucketMetricsConfiguration](#)。

Tools for PowerShell V5

範例 1：命令會移除給定 S3 儲存貯體中名為 'testmetrics' 的指標篩選條件。

```
Remove-S3BucketMetricsConfiguration -BucketName 'amzn-s3-demo-bucket' -MetricsId  
'testmetrics'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteBucketMetricsConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteBucketPolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteBucketPolicy。

C++

SDK for C++

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::deleteBucketPolicy(const Aws::String &bucketName,  
                                     const Aws::S3::S3ClientConfiguration  
                                     &clientConfig) {  
    Aws::S3::S3Client client(clientConfig);  
  
    Aws::S3::Model::DeleteBucketPolicyRequest request;  
    request.SetBucket(bucketName);  
  
    Aws::S3::Model::DeleteBucketPolicyOutcome outcome =  
        client.DeleteBucketPolicy(request);  
  
    if (!outcome.IsSuccess()) {  
        const Aws::S3::S3Error &err = outcome.GetError();  
        std::cerr << "Error: deleteBucketPolicy: " <<
```

```
err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
} else {
    std::cout << "Policy was deleted from the bucket." << std::endl;
}

return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [DeleteBucketPolicy](#)。

CLI

AWS CLI

下列命令會從名為 `amzn-s3-demo-bucket` 的儲存貯體中刪除儲存貯體政策：

```
aws s3api delete-bucket-policy --bucket amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucketPolicy](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.DeleteBucketPolicyRequest;

/**
 * Before running this Java V2 code example, set up your development
```

```
* environment, including your credentials.
*
* For more information, see the following documentation topic:
*
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
*/

public class DeleteBucketPolicy {
    public static void main(String[] args) {

        final String usage = ""

            Usage:
                <bucketName>

            Where:
                bucketName - The Amazon S3 bucket to delete the policy from
(for example, bucket1).""";

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        System.out.format("Deleting policy from bucket: \"%s\"\n\n", bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        deleteS3BucketPolicy(s3, bucketName);
        s3.close();
    }

    /**
     * Deletes the S3 bucket policy for the specified bucket.
     *
     * @param s3 the {@link S3Client} instance to use for the operation
     * @param bucketName the name of the S3 bucket for which the policy should be
deleted
     *
     * @throws S3Exception if there is an error deleting the bucket policy
    */
}
```

```
    */
    public static void deleteS3BucketPolicy(S3Client s3, String bucketName) {
        DeleteBucketPolicyRequest delReq = DeleteBucketPolicyRequest.builder()
            .bucket(bucketName)
            .build();

        try {
            s3.deleteBucketPolicy(delReq);
            System.out.println("Done!");
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteBucketPolicy](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除儲存貯體政策。

```
import {
    DeleteBucketPolicyCommand,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Remove the policy from an Amazon S3 bucket.
 * @param {{ bucketName: string }}
 */
```



```
export const main = async ({ bucketName }) => {
  const client = new S3Client({});

  try {
    await client.send(
      new DeleteBucketPolicyCommand({
        Bucket: bucketName,
      }),
    );
    console.log(`Bucket policy deleted from "${bucketName}".`);
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
      console.error(
        `Error from S3 while deleting policy from ${bucketName}. The bucket
        doesn't exist.`,
      );
    } else if (caught instanceof S3ServiceException) {
      console.error(
        `Error from S3 while deleting policy from ${bucketName}.  ${caught.name}:
        ${caught.message}`,
      );
    } else {
      throw caught;
    }
  }
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [DeleteBucketPolicy](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun deleteS3BucketPolicy(bucketName: String?) {  
    val request =  
        DeleteBucketPolicyRequest {  
            bucket = bucketName  
        }  
  
    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->  
        s3.deleteBucketPolicy(request)  
        println("Done!")  
    }  
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [DeleteBucketPolicy](#)。

PowerShell

Tools for PowerShell V4

範例 1：命令會移除與給定 S3 儲存貯體相關聯的儲存貯體政策。

```
Remove-S3BucketPolicy -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteBucketPolicy](#)。

Tools for PowerShell V5

範例 1：命令會移除與給定 S3 儲存貯體相關聯的儲存貯體政策。

```
Remove-S3BucketPolicy -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteBucketPolicy](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def delete_policy(self):
        """
        Delete the security policy from the bucket.
        """
        try:
            self.bucket.Policy().delete()
            logger.info("Deleted policy for bucket '%s'.", self.bucket.name)
        except ClientError:
            logger.exception(
                "Couldn't delete policy for bucket '%s'.", self.bucket.name
            )
            raise
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [DeleteBucketPolicy](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object
  # configured with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  def delete_policy
    @bucket_policy.delete
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't delete the policy from #{@bucket_policy.bucket.name}. Here's
  why: #{e.message}"
    false
  end
end

end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [DeleteBucketPolicy](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    lo_s3->deletebucketpolicy(  
        iv_bucket = iv_bucket_name ).  
    MESSAGE 'Bucket policy deleted.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [DeleteBucketPolicy](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteBucketReplication** 與 CLI

下列程式碼範例示範如何使用 DeleteBucketReplication。

CLI

AWS CLI

下列命令會從名為 amzn-s3-demo-bucket 的儲存貯體中刪除複寫組態：

```
aws s3api delete-bucket-replication --bucket amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucketReplication](#)。

PowerShell

Tools for PowerShell V4

範例 1：刪除與名為 'amzn-s3-demo-bucket' 的儲存貯體相關聯的複寫組態。請注意，此操作需要 s3:DeleteReplicationConfiguration 動作的許可權。在繼續操作之前，系統會提示您確認 - 若要隱藏確認，請使用 -Force 切換變數。

```
Remove-S3BucketReplication -BucketName amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteBucketReplication](#)。

Tools for PowerShell V5

範例 1：刪除與名為 'amzn-s3-demo-bucket' 的儲存貯體相關聯的複寫組態。請注意，此操作需要 s3:DeleteReplicationConfiguration 動作的許可權。在繼續操作之前，系統會提示您確認 - 若要隱藏確認，請使用 -Force 切換變數。

```
Remove-S3BucketReplication -BucketName amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteBucketReplication](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteBucketTagging** 與 CLI

下列程式碼範例示範如何使用 DeleteBucketTagging。

CLI

AWS CLI

下列命令會從名為 amzn-s3-demo-bucket 的儲存貯體中刪除標記組態：

```
aws s3api delete-bucket-tagging --bucket amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucketTagging](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會移除與給定 S3 儲存貯體相關聯的所有標籤。

```
Remove-S3BucketTagging -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketTagging (DeleteBucketTagging)" on target
"amzn-s3-demo-bucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteBucketTagging](#)。

Tools for PowerShell V5

範例 1：此命令會移除與給定 S3 儲存貯體相關聯的所有標籤。

```
Remove-S3BucketTagging -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketTagging (DeleteBucketTagging)" on target
"amzn-s3-demo-bucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteBucketTagging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteBucketWebsite 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteBucketWebsite。

C++

SDK for C++

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::deleteBucketWebsite(const Aws::String &bucketName,
                                       const Aws::S3::S3ClientConfiguration
                                       &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::DeleteBucketWebsiteRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::DeleteBucketWebsiteOutcome outcome =
        client.DeleteBucketWebsite(request);

    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error: deleteBucketWebsite: " <<
            err.GetExceptionName() << ": " << err.GetMessage() <<
            std::endl;
    } else {
        std::cout << "Website configuration was removed." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [DeleteBucketWebsite](#)。

CLI

AWS CLI

下列命令會從名為 `amzn-s3-demo-bucket` 的儲存貯體中刪除網站組態：

```
aws s3api delete-bucket-website --bucket amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteBucketWebsite](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.DeleteBucketWebsiteRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */

public class DeleteWebsiteConfiguration {
    public static void main(String[] args) {
        final String usage = ""

        Usage:      <bucketName>
```

```

        Where:
            bucketName - The Amazon S3 bucket to delete the website
configuration from.
        """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        System.out.format("Deleting website configuration for Amazon S3 bucket:
%s\n", bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        deleteBucketWebsiteConfig(s3, bucketName);
        System.out.println("Done!");
        s3.close();
    }

    /**
     * Deletes the website configuration for an Amazon S3 bucket.
     *
     * @param s3 The {@link S3Client} instance used to interact with Amazon S3.
     * @param bucketName The name of the S3 bucket for which the website
configuration should be deleted.
     * @throws S3Exception If an error occurs while deleting the website
configuration.
     */
    public static void deleteBucketWebsiteConfig(S3Client s3, String bucketName)
    {
        DeleteBucketWebsiteRequest delReq = DeleteBucketWebsiteRequest.builder()
            .bucket(bucketName)
            .build();

        try {
            s3.deleteBucketWebsite(delReq);
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

```

```
        System.out.println("Failed to delete website configuration!");
        System.exit(1);
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteBucketWebsite](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

從儲存貯體刪除網站組態

```
import {
    DeleteBucketWebsiteCommand,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Remove the website configuration for a bucket.
 * @param {{ bucketName: string }}
 */
export const main = async ({ bucketName }) => {
    const client = new S3Client({});

    try {
        await client.send(
            new DeleteBucketWebsiteCommand({
                Bucket: bucketName,
            }),
        );
        // The response code will be successful for both removed configurations and
```

```
// configurations that did not exist in the first place.
console.log(
    `The bucket "${bucketName}" is not longer configured as a website, or it
never was.` ,
);
} catch (caught) {
    if (
        caught instanceof S3ServiceException &&
        caught.name === "NoSuchBucket"
    ) {
        console.error(
            `Error from S3 while removing website configuration from ${bucketName}.
The bucket doesn't exist.` ,
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while removing website configuration from ${bucketName}.
${caught.name}: ${caught.message}` ,
        );
    } else {
        throw caught;
    }
}
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [DeleteBucketWebsite](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會停用給定 S3 儲存貯體的靜態網站託管屬性。

```
Remove-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Confirm
```

```
Are you sure you want to perform this action?  
Performing the operation "Remove-S3BucketWebsite (DeleteBucketWebsite)" on target  
"amzn-s3-demo-bucket".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): Y
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteBucketWebsite](#)。

Tools for PowerShell V5

範例 1：此命令會停用給定 S3 儲存貯體的靜態網站託管屬性。

```
Remove-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Confirm  
Are you sure you want to perform this action?  
Performing the operation "Remove-S3BucketWebsite (DeleteBucketWebsite)" on target  
"amzn-s3-demo-bucket".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): Y
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteBucketWebsite](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteObject 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteObject。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [使用 Amazon S3 物件完整性](#)
- [使用版本化物件](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除未進行版本控制之 S3 儲存貯體中的物件。

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to delete an object from a non-versioned Amazon
/// Simple Storage Service (Amazon S3) bucket.
/// </summary>
public class DeleteObject
{
    /// <summary>
    /// The Main method initializes the necessary variables and then calls
    /// the DeleteObjectNonVersionedBucketAsync method to delete the object
    /// named by the keyName parameter.
    /// </summary>
    public static async Task Main()
    {
        const string bucketName = "amzn-s3-demo-bucket";
        const string keyName = "testfile.txt";

        // If the Amazon S3 bucket is located in an AWS Region other than the
        // Region of the default account, define the AWS Region for the
        // Amazon S3 bucket in your call to the AmazonS3Client constructor.
        // For example RegionEndpoint.USWest2.
        IAmazonS3 client = new AmazonS3Client();
        await DeleteObjectNonVersionedBucketAsync(client, bucketName,
keyName);
    }

    /// <summary>
```

```

    /// The DeleteObjectNonVersionedBucketAsync takes care of deleting the
    /// desired object from the named bucket.
    /// </summary>
    /// <param name="client">An initialized Amazon S3 client used to delete
    /// an object from an Amazon S3 bucket.</param>
    /// <param name="bucketName">The name of the bucket from which the
    /// object will be deleted.</param>
    /// <param name="keyName">The name of the object to delete.</param>
    public static async Task DeleteObjectNonVersionedBucketAsync(IAmazonS3
client, string bucketName, string keyName)
    {
        try
        {
            var deleteObjectRequest = new DeleteObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,
            };

            Console.WriteLine($"Deleting object: {keyName}");
            await client.DeleteObjectAsync(deleteObjectRequest);
            Console.WriteLine($"Object: {keyName} deleted from
{bucketName}.");
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Error encountered on server.
Message: '{ex.Message}' when deleting an object.");
        }
    }
}

```

刪除已進行版本控制之 S3 儲存貯體中的物件。

```

using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example creates an object in an Amazon Simple Storage Service

```

```

    /// (Amazon S3) bucket and then deletes the object version that was
    /// created.
    /// </summary>
    public class DeleteObjectVersion
    {
        public static async Task Main()
        {
            string bucketName = "amzn-s3-demo-bucket";
            string keyName = "versioned-object.txt";

            // If the AWS Region of the default user is different from the AWS
            // Region of the Amazon S3 bucket, pass the AWS Region of the
            // bucket region to the Amazon S3 client object's constructor.
            // Define it like this:
            //      RegionEndpoint bucketRegion = RegionEndpoint.USSouth1;
            IAmazonS3 client = new AmazonS3Client();

            await CreateAndDeleteObjectVersionAsync(client, bucketName, keyName);
        }

        /// <summary>
        /// This method creates and then deletes a versioned object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
        /// create and delete the object.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket where the
        /// object will be created and deleted.</param>
        /// <param name="keyName">The key name of the object to create.</param>
        public static async Task CreateAndDeleteObjectVersionAsync(IAmazonS3
client, string bucketName, string keyName)
        {
            try
            {
                // Add a sample object.
                string versionID = await PutAnObject(client, bucketName,
keyName);

                // Delete the object by specifying an object key and a version
ID.

                DeleteObjectRequest request = new DeleteObjectRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                    VersionId = versionID,

```



```

        };

        Console.WriteLine("Deleting an object");
        await client.DeleteObjectAsync(request);
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error: {ex.Message}");
    }
}

/// <summary>
/// This method is used to create the temporary Amazon S3 object.
/// </summary>
/// <param name="client">The initialized Amazon S3 object which will be
used
/// to create the temporary Amazon S3 object.</param>
/// <param name="bucketName">The name of the Amazon S3 bucket where the
object
/// will be created.</param>
/// <param name="objectKey">The name of the Amazon S3 object co create.</
param>
/// <returns>The Version ID of the created object.</returns>
public static async Task<string> PutAnObject(IAmazonS3 client, string
bucketName, string objectKey)
{
    PutObjectRequest request = new PutObjectRequest()
    {
        BucketName = bucketName,
        Key = objectKey,
        ContentBody = "This is the content body!",
    };

    PutObjectResponse response = await client.PutObjectAsync(request);
    return response.VersionId;
}
}

```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [DeleteObject](#)。

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function delete_item_in_bucket
#
# This function deletes the specified file from the specified bucket.
#
# Parameters:
#     $1 - The name of the bucket.
#     $2 - The key (file name) in the bucket to delete.

# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function delete_item_in_bucket() {
    local bucket_name=$1
    local key=$2
    local response

    response=$(aws s3api delete-object \
        --bucket "$bucket_name" \
        --key "$key")

    # shellcheck disable=SC2181
```

```
if [[ $? -ne 0 ]]; then
    errecho "ERROR: AWS reports s3api delete-object operation failed.\n
$response"
    return 1
fi
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteObject](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::deleteObject(const Aws::String &objectKey,
                               const Aws::String &fromBucket,
                               const Aws::S3::S3ClientConfiguration &clientConfig)
{
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::DeleteObjectRequest request;

    request.WithKey(objectKey)
        .WithBucket(fromBucket);

    Aws::S3::Model::DeleteObjectOutcome outcome =
        client.DeleteObject(request);

    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error: deleteObject: " <<
            err.GetExceptionName() << ": " << err.GetMessage() <<
        std::endl;
    } else {
        std::cout << "Successfully deleted the object." << std::endl;
    }
}
```

```
    return outcome.IsSuccess();  
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [DeleteObject](#)。

CLI

AWS CLI

下列命令會將名為 `test.txt` 的物件從名為 `amzn-s3-demo-bucket` 的儲存貯體中刪除。

```
aws s3api delete-object --bucket amzn-s3-demo-bucket --key test.txt
```

如果已啟用儲存貯體版本控制，輸出將包含刪除標記的版本 ID：

```
{  
  "VersionId": "9_gKg5vG56F.TTEUdwkxGpJ3tND1WlGq",  
  "DeleteMarker": true  
}
```

如需刪除物件的詳細資訊，請參閱《Amazon S3 開發人員指南》中的「刪除物件」。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteObject](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (  
    "bytes"  
    "context"  
    "errors"  
    "fmt"
```

```
"log"
"time"

"github.com/aws/aws-sdk-go-v2/aws"
"github.com/aws/aws-sdk-go-v2/feature/s3/manager"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/aws/aws-sdk-go-v2/service/s3/types"
"github.com/aws/smithy-go"
)

// S3Actions wraps S3 service actions.
type S3Actions struct {
    S3Client    *s3.Client
    S3Manager   *manager.Uploader
}

// DeleteObject deletes an object from a bucket.
func (actor S3Actions) DeleteObject(ctx context.Context, bucket string, key
    string, versionId string, bypassGovernance bool) (bool, error) {
    deleted := false
    input := &s3.DeleteObjectInput{
        Bucket: aws.String(bucket),
        Key:     aws.String(key),
    }
    if versionId != "" {
        input.VersionId = aws.String(versionId)
    }
    if bypassGovernance {
        input.BypassGovernanceRetention = aws.Bool(true)
    }
    _, err := actor.S3Client.DeleteObject(ctx, input)
    if err != nil {
        var noKey *types.NoSuchKey
        var apiErr *smithy.GenericAPIError
        if errors.As(err, &noKey) {
            log.Printf("Object %s does not exist in %s.\n", key, bucket)
            err = noKey
        } else if errors.As(err, &apiErr) {
            switch apiErr.ErrorCode() {
            case "AccessDenied":
                log.Printf("Access denied: cannot delete object %s from %s.\n", key, bucket)
            }
            err = nil
        }
    }
}
```

```

    case "InvalidArgument":
        if bypassGovernance {
            log.Printf("You cannot specify bypass governance on a bucket without lock
enabled.")
            err = nil
        }
    }
} else {
    err = s3.NewObjectNotExistsWaiter(actor.S3Client).Wait(
        ctx, &s3.HeadObjectInput{Bucket: aws.String(bucket), Key: aws.String(key)},
        time.Minute)
    if err != nil {
        log.Printf("Failed attempt to wait for object %s in bucket %s to be deleted.
\n", key, bucket)
    } else {
        deleted = true
    }
}
return deleted, err
}

```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [DeleteObject](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

/**
 * Deletes an object from an S3 bucket asynchronously.
 *
 * @param bucketName the name of the S3 bucket
 * @param key         the key (file name) of the object to be deleted

```

```
* @return a {@link CompletableFuture} that completes when the object has
been deleted
*/
public CompletableFuture<Void> deleteObjectFromBucketAsync(String bucketName,
String key) {
    DeleteObjectRequest deleteObjectRequest = DeleteObjectRequest.builder()
        .bucket(bucketName)
        .key(key)
        .build();

    CompletableFuture<DeleteObjectResponse> response =
getAsyncClient().deleteObject(deleteObjectRequest);
    response.whenComplete((deleteRes, ex) -> {
        if (deleteRes != null) {
            logger.info(key + " was deleted");
        } else {
            throw new RuntimeException("An S3 exception occurred during
delete", ex);
        }
    });

    return response.thenApply(r -> null);
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteObject](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除物件。

```
import {
    DeleteObjectCommand,
    S3Client,
```

```
S3ServiceException,
waitUntilObjectNotExists,
} from "@aws-sdk/client-s3";

/**
 * Delete one object from an Amazon S3 bucket.
 * @param {{ bucketName: string, key: string }}
 */
export const main = async ({ bucketName, key }) => {
  const client = new S3Client({});

  try {
    await client.send(
      new DeleteObjectCommand({
        Bucket: bucketName,
        Key: key,
      }),
    );
    await waitUntilObjectNotExists(
      { client },
      { Bucket: bucketName, Key: key },
    );
    // A successful delete, or a delete for a non-existent object, both return
    // a 204 response code.
    console.log(
      `The object "${key}" from bucket "${bucketName}" was deleted, or it didn't
      exist.`
    );
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
      console.error(
        `Error from S3 while deleting object from ${bucketName}. The bucket
        doesn't exist.`
      );
    } else if (caught instanceof S3ServiceException) {
      console.error(
        `Error from S3 while deleting object from ${bucketName}. ${caught.name}:
        ${caught.message}`
      );
    } else {
      throw caught;
    }
  }
}
```



```
    }  
  }  
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [DeleteObject](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
public function deleteObject(string $bucketName, string $fileName, array  
$args = [])  
{  
    $parameters = array_merge(['Bucket' => $bucketName, 'Key' => $fileName],  
$args);  
    try {  
        $this->client->deleteObject($parameters);  
        if ($this->verbose) {  
            echo "Deleted the object named: $fileName from $bucketName.\n";  
        }  
    } catch (AwsException $exception) {  
        if ($this->verbose) {  
            echo "Failed to delete $fileName from $bucketName with error:  
{ $exception->getMessage() }\n";  
            echo "Please fix error with object deletion before continuing.";  
        }  
        throw $exception;  
    }  
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [DeleteObject](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除物件。

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
        in Boto3
                               that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key

    def delete(self):
        """
        Deletes the object.
        """
        try:
            self.object.delete()
            self.object.wait_until_not_exists()
            logger.info(
                "Deleted object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't delete object '%s' from bucket '%s'.",
```

```
        self.object.key,  
        self.object.bucket_name,  
    )  
    raise
```

透過刪除較新的物件版本，將物件還原至先前的版本。

```
def rollback_object(bucket, object_key, version_id):  
    """  
    Rolls back an object to an earlier version by deleting all versions that  
    occurred after the specified rollback version.  
  
    Usage is shown in the usage_demo_single_object function at the end of this  
    module.  
  
    :param bucket: The bucket that holds the object to roll back.  
    :param object_key: The object to roll back.  
    :param version_id: The version ID to roll back to.  
    """  
    # Versions must be sorted by last_modified date because delete markers are  
    # at the end of the list even when they are interspersed in time.  
    versions = sorted(  
        bucket.object_versions.filter(Prefix=object_key),  
        key=attrgetter("last_modified"),  
        reverse=True,  
    )  
  
    logger.debug(  
        "Got versions:\n%s",  
        "\n".join(  
            [  
                f"\t{version.version_id}, last modified {version.last_modified}"  
                for version in versions  
            ]  
        ),  
    )  
  
    if version_id in [ver.version_id for ver in versions]:  
        print(f"Rolling back to version {version_id}")  
        for version in versions:  
            if version.version_id != version_id:
```

```

        version.delete()
        print(f"Deleted version {version.version_id}")
    else:
        break

    print(f"Active version is now {bucket.Object(object_key).version_id}")
else:
    raise KeyError(
        f"{version_id} was not found in the list of versions for "
        f"{object_key}."
    )

```

透過刪除物件啟用的刪除標記，恢復被刪除的物件。

```

def revive_object(bucket, object_key):
    """
    Revives a versioned object that was deleted by removing the object's active
    delete marker.
    A versioned object presents as deleted when its latest version is a delete
    marker.
    By removing the delete marker, we make the previous version the latest
    version
    and the object then presents as not deleted.

    Usage is shown in the usage_demo_single_object function at the end of this
    module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to revive.
    """
    # Get the latest version for the object.
    response = s3.meta.client.list_object_versions(
        Bucket=bucket.name, Prefix=object_key, MaxKeys=1
    )

    if "DeleteMarkers" in response:
        latest_version = response["DeleteMarkers"][0]
        if latest_version["IsLatest"]:
            logger.info(
                "Object %s was indeed deleted on %s. Let's revive it.",

```

```

        object_key,
        latest_version["LastModified"],
    )
    obj = bucket.Object(object_key)
    obj.Version(latest_version["VersionId"]).delete()
    logger.info(
        "Revived %s, active version is now %s with body '%s'",
        object_key,
        obj.version_id,
        obj.get()["Body"].read(),
    )
else:
    logger.warning(
        "Delete marker is not the latest version for %s!", object_key
    )
elif "Versions" in response:
    logger.warning("Got an active version for %s, nothing to do.",
object_key)
else:
    logger.error("Couldn't get any version info for %s.", object_key)

```

建立 Lambda 處理常式從 S3 物件中移除刪除標記。此處理常式可有效清除版本化的儲存貯體中無關的刪除標記。

```

import logging
from urllib import parse
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
logger.setLevel("INFO")

s3 = boto3.client("s3")

def lambda_handler(event, context):
    """
    Removes a delete marker from the specified versioned object.

    :param event: The S3 batch event that contains the ID of the delete marker

```

```

        to remove.
:param context: Context about the event.
:return: A result structure that Amazon S3 uses to interpret the result of
the
        operation. When the result code is TemporaryFailure, S3 retries the
        operation.
"""
# Parse job parameters from Amazon S3 batch operations
invocation_id = event["invocationId"]
invocation_schema_version = event["invocationSchemaVersion"]

results = []
result_code = None
result_string = None

task = event["tasks"][0]
task_id = task["taskId"]

try:
    obj_key = parse.unquote_plus(task["s3Key"], encoding="utf-8")
    obj_version_id = task["s3VersionId"]
    bucket_name = task["s3BucketArn"].split(":")[-1]

    logger.info(
        "Got task: remove delete marker %s from object %s.", obj_version_id,
obj_key
    )

    try:
        # If this call does not raise an error, the object version is not a
delete
        # marker and should not be deleted.
        response = s3.head_object(
            Bucket=bucket_name, Key=obj_key, VersionId=obj_version_id
        )
        result_code = "PermanentFailure"
        result_string = (
            f"Object {obj_key}, ID {obj_version_id} is not " f"a delete
marker."
        )

        logger.debug(response)
        logger.warning(result_string)
    except ClientError as error:

```

```

        delete_marker = error.response["ResponseMetadata"]
["HTTPHeaders"].get(
    "x-amz-delete-marker", "false"
)
    if delete_marker == "true":
        logger.info(
            "Object %s, version %s is a delete marker.", obj_key,
obj_version_id
        )
        try:
            s3.delete_object(
                Bucket=bucket_name, Key=obj_key, VersionId=obj_version_id
            )
            result_code = "Succeeded"
            result_string = (
                f"Successfully removed delete marker "
                f"{obj_version_id} from object {obj_key}."
            )
            logger.info(result_string)
        except ClientError as error:
            # Mark request timeout as a temporary failure so it will be
retried.

            if error.response["Error"]["Code"] == "RequestTimeout":
                result_code = "TemporaryFailure"
                result_string = (
                    f"Attempt to remove delete marker from "
                    f"object {obj_key} timed out."
                )
                logger.info(result_string)
            else:
                raise
        else:
            raise ValueError(
                f"The x-amz-delete-marker header is either not "
                f"present or is not 'true'."
            )
    except Exception as error:
        # Mark all other exceptions as permanent failures.
        result_code = "PermanentFailure"
        result_string = str(error)
        logger.exception(error)
    finally:
        results.append(
            {

```

```
        "taskId": task_id,  
        "resultCode": result_code,  
        "resultString": result_string,  
    }  
)  
return {  
    "invocationSchemaVersion": invocation_schema_version,  
    "treatMissingKeysAs": "PermanentFailure",  
    "invocationId": invocation_id,  
    "results": results,  
}
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [DeleteObject](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// Delete an object from a bucket.  
pub async fn remove_object(  
    client: &aws_sdk_s3::Client,  
    bucket: &str,  
    key: &str,  
) -> Result<(), S3ExampleError> {  
    client  
        .delete_object()  
        .bucket(bucket)  
        .key(key)  
        .send()  
        .await?;
```



```
// There are no modeled errors to handle when deleting an object.  
  
Ok()  
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [DeleteObject](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    lo_s3->deleteobject(  
        iv_bucket = iv_bucket_name  
        iv_key = iv_object_key ).  
    MESSAGE 'Object deleted from S3 bucket.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 SAP ABAP 的 AWS SDK API 參考》中的 [DeleteObject](#)。

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import AWSS3

public func deleteFile(bucket: String, key: String) async throws {
    let input = DeleteObjectInput(
        bucket: bucket,
        key: key
    )

    do {
        _ = try await client.deleteObject(input: input)
    }
    catch {
        print("ERROR: ", dump(error, name: "Deleting a file. "))
        throw error
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Swift 的 AWS SDK API 參考》中的 [DeleteObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteObjectTagging** 與 CLI

下列程式碼範例示範如何使用 DeleteObjectTagging。

CLI

AWS CLI

刪除物件的標籤集

下列 delete-object-tagging 範例會從物件 doc1.rtf 刪除具有指定索引鍵的標籤。

```
aws s3api delete-object-tagging \
    --bucket amzn-s3-demo-bucket \
    --key doc1.rtf
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteObjectTagging](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會移除與給定 S3 儲存貯體中具索引鍵 'testfile.txt' 之物件相關聯的所有標籤。

```
Remove-S3ObjectTagSet -Key 'testfile.txt' -BucketName 'amzn-s3-demo-bucket' -  
Select '^Key'
```

輸出：

```
Confirm  
Are you sure you want to perform this action?  
Performing the operation "Remove-S3ObjectTagSet (DeleteObjectTagging)" on target  
"testfile.txt".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): Y  
testfile.txt
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteObjectTagging](#)。

Tools for PowerShell V5

範例 1：此命令會移除與給定 S3 儲存貯體中具索引鍵 'testfile.txt' 之物件相關聯的所有標籤。

```
Remove-S3ObjectTagSet -Key 'testfile.txt' -BucketName 'amzn-s3-demo-bucket' -  
Select '^Key'
```

輸出：

```
Confirm  
Are you sure you want to perform this action?  
Performing the operation "Remove-S3ObjectTagSet (DeleteObjectTagging)" on target  
"testfile.txt".  
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is  
"Y"): Y  
testfile.txt
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteObjectTagging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteObjects 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteObjects。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)
- [刪除儲存貯體中的所有物件](#)
- [物件儲存入門](#)

.NET

適用於 .NET 的 SDK (v4)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Delete all of the objects stored in an existing Amazon S3 bucket.
/// </summary>
/// <param name="bucketName">The name of the bucket from which the
/// contents will be deleted.</param>
/// <returns>A boolean value that represents the success or failure of
/// deleting all of the objects in the bucket.</returns>
public async Task<bool> DeleteBucketContentsAsync(string bucketName)
{
    // Iterate over the contents of the bucket and delete all objects.
    try
    {
        // Delete all objects in the bucket.
        var deleteList = await ListBucketContentsAsync(bucketName, false);
        if (deleteList != null && deleteList.Any())
        {
```

```
        await _amazonS3.DeleteObjectsAsync(new DeleteObjectsRequest()
        {
            BucketName = bucketName,
            Objects = deleteList.Select(o => new KeyVersion { Key =
o.Key }).ToList(),
        });
    }

    return true;
}
catch (AmazonS3Exception ex)
{
    Console.WriteLine($"Error deleting objects: {ex.Message}");
    return false;
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [DeleteObjects](#)。
適用於 .NET 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除未進行版本控制之 S3 儲存貯體中的多個物件。

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to delete multiple objects from an Amazon Simple
/// Storage Service (Amazon S3) bucket.
/// </summary>
public class DeleteMultipleObjects
{

```

```

    /// <summary>
    /// The Main method initializes the Amazon S3 client and the name of
    /// the bucket and then passes those values to MultiObjectDeleteAsync.
    /// </summary>
    public static async Task Main()
    {
        const string bucketName = "amzn-s3-demo-bucket";

        // If the Amazon S3 bucket from which you wish to delete objects is
        // located in the same AWS Region as the default user, define the
        // AWS Region for the Amazon S3 bucket as a parameter to the client
        // constructor.
        IAmazonS3 s3Client = new AmazonS3Client();

        await MultiObjectDeleteAsync(s3Client, bucketName);
    }

    /// <summary>
    /// This method uses the passed Amazon S3 client to first create and then
    /// delete three files from the named bucket.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
    /// Amazon S3 methods.</param>
    /// <param name="bucketName">The name of the Amazon S3 bucket where
    /// objects
    /// will be created and then deleted.</param>
    public static async Task MultiObjectDeleteAsync(IAmazonS3 client, string
    bucketName)
    {
        // Create three sample objects which we will then delete.
        var keysAndVersions = await PutObjectsAsync(client, 3, bucketName);

        // Now perform the multi-object delete, passing the key names and
        // version IDs. Since we are working with a non-versioned bucket,
        // the object keys collection includes null version IDs.
        DeleteObjectsRequest multiObjectDeleteRequest = new
        DeleteObjectsRequest
        {
            BucketName = bucketName,
            Objects = keysAndVersions,
        };
    }

```

```

        // You can add a specific object key to the delete request using the
        // AddKey method of the multiObjectDeleteRequest.
        try
        {
            DeleteObjectsResponse response = await
client.DeleteObjectsAsync(multiObjectDeleteRequest);
            Console.WriteLine("Successfully deleted all the {0} items",
response.DeletedObjects.Count);
        }
        catch (DeleteObjectsException e)
        {
            PrintDeletionErrorStatus(e);
        }
    }

    /// <summary>
    /// Prints the list of errors raised by the call to DeleteObjectsAsync.
    /// </summary>
    /// <param name="ex">A collection of exceptions returned by the call to
    /// DeleteObjectsAsync.</param>
    public static void PrintDeletionErrorStatus(DeleteObjectsException ex)
    {
        DeleteObjectsResponse errorResponse = ex.Response;
        Console.WriteLine("x {0}", errorResponse.DeletedObjects.Count);

        Console.WriteLine($"Successfully deleted
{errorResponse.DeletedObjects.Count}.");
        Console.WriteLine($"No. of objects failed to delete =
{errorResponse.DeleteErrors.Count}");

        Console.WriteLine("Printing error data...");
        foreach (DeleteError deleteError in errorResponse.DeleteErrors)
        {
            Console.WriteLine($"Object Key:
{deleteError.Key}\t{deleteError.Code}\t{deleteError.Message}");
        }
    }

    /// <summary>
    /// This method creates simple text file objects that can be used in
    /// the delete method.
    /// </summary>
    /// <param name="client">The Amazon S3 client used to call
    PutObjectAsync.</param>

```

```
/// <param name="number">The number of objects to create.</param>
/// <param name="bucketName">The name of the bucket where the objects
/// will be created.</param>
/// <returns>A list of keys (object keys) and versions that the calling
/// method will use to delete the newly created files.</returns>
public static async Task<List<KeyVersion>> PutObjectsAsync(IAmazonS3
client, int number, string bucketName)
{
    List<KeyVersion> keys = new List<KeyVersion>();
    for (int i = 0; i < number; i++)
    {
        string key = "ExampleObject-" + new System.Random().Next();
        PutObjectRequest request = new PutObjectRequest
        {
            BucketName = bucketName,
            Key = key,
            ContentBody = "This is the content body!",
        };

        PutObjectResponse response = await
client.PutObjectAsync(request);

        // For non-versioned bucket operations, we only need the
        // object key.
        KeyVersion keyVersion = new KeyVersion
        {
            Key = key,
        };
        keys.Add(keyVersion);
    }

    return keys;
}
}
```

刪除已進行版本控制之 S3 儲存貯體中的多個物件。

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon.S3;
```



```
using Amazon.S3.Model;

/// <summary>
/// This example shows how to delete objects in a version-enabled Amazon
/// Simple StorageService (Amazon S3) bucket.
/// </summary>
public class DeleteMultipleObjects
{
    public static async Task Main()
    {
        string bucketName = "amzn-s3-demo-bucket";

        // If the AWS Region for your Amazon S3 bucket is different from
        // the AWS Region of the default user, define the AWS Region for
        // the Amazon S3 bucket and pass it to the client constructor
        // like this:
        // RegionEndpoint bucketRegion = RegionEndpoint.USSouth1;
        IAmazonS3 s3Client;

        s3Client = new AmazonS3Client();
        await DeleteMultipleObjectsFromVersionedBucketAsync(s3Client,
bucketName);
    }

    /// <summary>
    /// This method removes multiple versions and objects from a
    /// version-enabled Amazon S3 bucket.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
    /// RemoveDeleteMarkersAsync.</param>
    /// <param name="bucketName">The name of the bucket from which to delete
    /// objects.</param>
    public static async Task
DeleteMultipleObjectsFromVersionedBucketAsync(IAmazonS3 client, string
bucketName)
    {
        // Delete objects (specifying object version in the request).
        await DeleteObjectVersionsAsync(client, bucketName);

        // Delete objects (without specifying object version in the request).
        var deletedObjects = await DeleteObjectsAsync(client, bucketName);
    }
}
```

```

        // Additional exercise - remove the delete markers Amazon S3 returned
from
        // the preceding response. This results in the objects reappearing
        // in the bucket (you can verify the appearance/disappearance of
        // objects in the console).
        await RemoveDeleteMarkersAsync(client, bucketName, deletedObjects);
    }

    /// <summary>
    /// Creates and then deletes non-versioned Amazon S3 objects and then
deletes
    /// them again. The method returns a list of the Amazon S3 objects
deleted.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// PubObjectsAsync and NonVersionedDeleteAsync.</param>
    /// <param name="bucketName">The name of the bucket where the objects
    /// will be created and then deleted.</param>
    /// <returns>A list of DeletedObjects.</returns>
    public static async Task<List<DeletedObject>>
DeleteObjectsAsync(IAmazonS3 client, string bucketName)
    {
        // Upload the sample objects.
        var keysAndVersions2 = await PutObjectsAsync(client, bucketName, 3);

        // Delete objects using only keys. Amazon S3 creates a delete marker
and
        // returns its version ID in the response.
        List<DeletedObject> deletedObjects = await
NonVersionedDeleteAsync(client, bucketName, keysAndVersions2);
        return deletedObjects;
    }

    /// <summary>
    /// This method creates several temporary objects and then deletes them.
    /// </summary>
    /// <param name="client">The S3 client.</param>
    /// <param name="bucketName">Name of the bucket.</param>
    /// <returns>Async task.</returns>
    public static async Task DeleteObjectVersionsAsync(IAmazonS3 client,
string bucketName)
    {
        // Upload the sample objects.

```

```

        var keysAndVersions1 = await PutObjectsAsync(client, bucketName, 3);

        // Delete the specific object versions.
        await VersionedDeleteAsync(client, bucketName, keysAndVersions1);
    }

    /// <summary>
    /// Displays the list of information about deleted files to the console.
    /// </summary>
    /// <param name="e">Error information from the delete process.</param>
    private static void DisplayDeletionErrors>DeleteObjectsException e)
    {
        var errorResponse = e.Response;
        Console.WriteLine($"No. of objects successfully deleted =
{errorResponse.DeletedObjects.Count}");
        Console.WriteLine($"No. of objects failed to delete =
{errorResponse.DeleteErrors.Count}");
        Console.WriteLine("Printing error data...");
        foreach (var deleteError in errorResponse.DeleteErrors)
        {
            Console.WriteLine($"Object Key:
{deleteError.Key}\t{deleteError.Code}\t{deleteError.Message}");
        }
    }

    /// <summary>
    /// Delete multiple objects from a version-enabled bucket.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
    /// RemoveDeleteMarkersAsync.</param>
    /// <param name="bucketName">The name of the bucket from which to delete
    /// objects.</param>
    /// <param name="keys">A list of key names for the objects to delete.</
param>
    private static async Task VersionedDeleteAsync(IAmazonS3 client, string
bucketName, List<KeyVersion> keys)
    {
        var multiObjectDeleteRequest = new DeleteObjectsRequest
        {
            BucketName = bucketName,
            Objects = keys, // This includes the object keys and specific
version IDs.

```

```

        };

        try
        {
            Console.WriteLine("Executing VersionedDelete...");
            DeleteObjectsResponse response = await
client.DeleteObjectsAsync(multiObjectDeleteRequest);
            Console.WriteLine($"Successfully deleted all the
{response.DeletedObjects.Count} items");
        }
        catch (DeleteObjectsException ex)
        {
            DisplayDeletionErrors(ex);
        }
    }

    /// <summary>
    /// Deletes multiple objects from a non-versioned Amazon S3 bucket.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
    /// RemoveDeleteMarkersAsync.</param>
    /// <param name="bucketName">The name of the bucket from which to delete
    /// objects.</param>
    /// <param name="keys">A list of key names for the objects to delete.</
param>
    /// <returns>A list of the deleted objects.</returns>
    private static async Task<List<DeletedObject>>
NonVersionedDeleteAsync(IAmazonS3 client, string bucketName, List<KeyVersion>
keys)
    {
        // Create a request that includes only the object key names.
        DeleteObjectsRequest multiObjectDeleteRequest = new
DeleteObjectsRequest();
        multiObjectDeleteRequest.BucketName = bucketName;

        foreach (var key in keys)
        {
            multiObjectDeleteRequest.AddKey(key.Key);
        }

        // Execute DeleteObjectsAsync.
        // The DeleteObjectsAsync method adds a delete marker for each

```

```

        // object deleted. You can verify that the objects were removed
        // using the Amazon S3 console.
        DeleteObjectsResponse response;
        try
        {
            Console.WriteLine("Executing NonVersionedDelete...");
            response = await
client.DeleteObjectsAsync(multiObjectDeleteRequest);
            Console.WriteLine("Successfully deleted all the {0} items",
response.DeletedObjects.Count);
        }
        catch (DeleteObjectsException ex)
        {
            DisplayDeletionErrors(ex);
            throw; // Some deletions failed. Investigate before continuing.
        }

        // This response contains the DeletedObjects list which we use to
delete the delete markers.
        return response.DeletedObjects;
    }

    /// <summary>
    /// Deletes the markers left after deleting the temporary objects.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
    /// RemoveDeleteMarkersAsync.</param>
    /// <param name="bucketName">The name of the bucket from which to delete
    /// objects.</param>
    /// <param name="deletedObjects">A list of the objects that were
deleted.</param>
    private static async Task RemoveDeleteMarkersAsync(IAmazonS3 client,
string bucketName, List<DeletedObject> deletedObjects)
    {
        var keyVersionList = new List<KeyVersion>();

        foreach (var deletedObject in deletedObjects)
        {
            KeyVersion keyVersion = new KeyVersion
            {
                Key = deletedObject.Key,
                VersionId = deletedObject.DeleteMarkerVersionId,

```

```

        };
        keyVersionList.Add(keyVersion);
    }

    // Create another request to delete the delete markers.
    var multiObjectDeleteRequest = new DeleteObjectsRequest
    {
        BucketName = bucketName,
        Objects = keyVersionList,
    };

    // Now, delete the delete marker to bring your objects back to the
    bucket.
    try
    {
        Console.WriteLine("Removing the delete markers .....");
        var deleteObjectResponse = await
client.DeleteObjectsAsync(multiObjectDeleteRequest);
        Console.WriteLine($"Successfully deleted the
{deleteObjectResponse.DeletedObjects.Count} delete markers");
    }
    catch (DeleteObjectsException ex)
    {
        DisplayDeletionErrors(ex);
    }
}

/// <summary>
/// Create temporary Amazon S3 objects to show how object deletion works
in an
/// Amazon S3 bucket with versioning enabled.
/// </summary>
/// <param name="client">The initialized Amazon S3 client object used to
call
/// PutObjectAsync to create temporary objects for the example.</param>
/// <param name="bucketName">A string representing the name of the S3
/// bucket where we will create the temporary objects.</param>
/// <param name="number">The number of temporary objects to create.</
param>
/// <returns>A list of the KeyVersion objects.</returns>
private static async Task<List<KeyVersion>> PutObjectsAsync(IAmazonS3
client, string bucketName, int number)
{
    var keys = new List<KeyVersion>();

```

```
for (var i = 0; i < number; i++)
{
    string key = "ObjectToDelete-" + new System.Random().Next();
    PutObjectRequest request = new PutObjectRequest
    {
        BucketName = bucketName,
        Key = key,
        ContentBody = "This is the content body!",
    };

    var response = await client.PutObjectAsync(request);
    KeyVersion keyVersion = new KeyVersion
    {
        Key = key,
        VersionId = response.VersionId,
    };

    keys.Add(keyVersion);
}

return keys;
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [DeleteObjects](#)。

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
#####
# function errecho
#
```

```

# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function delete_items_in_bucket
#
# This function deletes the specified list of keys from the specified bucket.
#
# Parameters:
#     $1 - The name of the bucket.
#     $2 - A list of keys in the bucket to delete.

# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function delete_items_in_bucket() {
    local bucket_name=$1
    local keys=$2
    local response

    # Create the JSON for the items to delete.
    local delete_items
    delete_items="{\"Objects\":["
    for key in $keys; do
        delete_items="$delete_items{\"Key\": \"$key\"},"
    done
    delete_items=${delete_items%?} # Remove the final comma.
    delete_items="$delete_items]"

    response=$(aws s3api delete-objects \
        --bucket "$bucket_name" \
        --delete "$delete_items")

    # shellcheck disable=SC2181
    if [[ $? -ne 0 ]]; then
        errecho "ERROR: AWS reports s3api delete-object operation failed.\n
$response"
        return 1
    fi
}

```


- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteObjects](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::deleteObjects(const std::vector<Aws::String> &objectKeys,
                               const Aws::String &fromBucket,
                               const Aws::S3::S3ClientConfiguration
                               &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::DeleteObjectsRequest request;

    Aws::S3::Model::Delete deleteObject;
    for (const Aws::String &objectKey: objectKeys) {
        deleteObject.AddObjects(Aws::S3::Model::ObjectIdentifier().WithKey(objectKey));
    }

    request.SetDelete(deleteObject);
    request.SetBucket(fromBucket);

    Aws::S3::Model::DeleteObjectsOutcome outcome =
        client.DeleteObjects(request);

    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error deleting objects. " <<
            err.GetExceptionName() << ": " << err.GetMessage() <<
        std::endl;
    } else {
        std::cout << "Successfully deleted the objects.";
        for (size_t i = 0; i < objectKeys.size(); ++i) {
            std::cout << objectKeys[i];
        }
    }
}
```

```
        if (i < objectKeys.size() - 1) {
            std::cout << ", ";
        }

        std::cout << " from bucket " << fromBucket << "." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [DeleteObjects](#)。

CLI

AWS CLI

下列命令會將從名為 `amzn-s3-demo-bucket` 的儲存貯體中刪除物件。

```
aws s3api delete-objects --bucket amzn-s3-demo-bucket --delete file://delete.json
```

`delete.json` 是目前目錄中的 JSON 文件，其會指定要刪除的物件：

```
{
  "Objects": [
    {
      "Key": "test1.txt"
    }
  ],
  "Quiet": false
}
```

輸出：

```
{
  "Deleted": [
    {
      "DeleteMarkerVersionId": "mYAT5Mc6F7aeUL8SS7FAAqUP01koHwzU",
      "Key": "test1.txt",
      "DeleteMarker": true
    }
  ]
}
```

```
]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteObjects](#)。

Go

SDK for Go V2

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (
    "bytes"
    "context"
    "errors"
    "fmt"
    "log"
    "time"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"
    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/aws-sdk-go-v2/service/s3/types"
    "github.com/aws/smithy-go"
)

// S3Actions wraps S3 service actions.
type S3Actions struct {
    S3Client    *s3.Client
    S3Manager   *manager.Uploader
}

// DeleteObjects deletes a list of objects from a bucket.
func (actor S3Actions) DeleteObjects(ctx context.Context, bucket string, objects
    []types.ObjectIdentifier, bypassGovernance bool) error {
```

```
if len(objects) == 0 {
    return nil
}

input := s3.DeleteObjectsInput{
    Bucket: aws.String(bucket),
    Delete: &types.Delete{
        Objects: objects,
        Quiet:    aws.Bool(true),
    },
}

if bypassGovernance {
    input.BypassGovernanceRetention = aws.Bool(true)
}

delOut, err := actor.S3Client.DeleteObjects(ctx, &input)
if err != nil || len(delOut.Errors) > 0 {
    log.Printf("Error deleting objects from bucket %s.\n", bucket)
    if err != nil {
        var noBucket *types.NoSuchBucket
        if errors.As(err, &noBucket) {
            log.Printf("Bucket %s does not exist.\n", bucket)
            err = noBucket
        }
    } else if len(delOut.Errors) > 0 {
        for _, outErr := range delOut.Errors {
            log.Printf("%s: %s\n", *outErr.Key, *outErr.Message)
        }
        err = fmt.Errorf("%s", *delOut.Errors[0].Message)
    }
} else {
    for _, delObjs := range delOut.Deleted {
        err = s3.NewObjectNotExistsWaiter(actor.S3Client).Wait(
            ctx, &s3.HeadObjectInput{Bucket: aws.String(bucket), Key: delObjs.Key},
            time.Minute)
        if err != nil {
            log.Printf("Failed attempt to wait for object %s to be deleted.\n",
                *delObjs.Key)
        } else {
            log.Printf("Deleted %s.\n", *delObjs.Key)
        }
    }
}

return err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [DeleteObjects](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.services.s3.model.Delete;
import software.amazon.awssdk.services.s3.model.DeleteObjectsRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.util.ArrayList;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */

public class DeleteMultiObjects {
    public static void main(String[] args) {
        final String usage = ""

        Usage:    <bucketName>
```

```
Where:
    bucketName - the Amazon S3 bucket name.
    "";

if (args.length != 1) {
    System.out.println(usage);
    System.exit(1);
}

String bucketName = args[0];
Region region = Region.US_EAST_1;
S3Client s3 = S3Client.builder()
    .region(region)
    .build();

deleteBucketObjects(s3, bucketName);
s3.close();
}

/**
 * Deletes multiple objects from an Amazon S3 bucket.
 *
 * @param s3 An Amazon S3 client object.
 * @param bucketName The name of the Amazon S3 bucket to delete objects from.
 */
public static void deleteBucketObjects(S3Client s3, String bucketName) {
    // Upload three sample objects to the specified Amazon S3 bucket.
    ArrayList<ObjectIdentifier> keys = new ArrayList<>();
    PutObjectRequest putOb;
    ObjectIdentifier objectId;

    for (int i = 0; i < 3; i++) {
        String keyName = "delete object example " + i;
        objectId = ObjectIdentifier.builder()
            .key(keyName)
            .build();

        putOb = PutObjectRequest.builder()
            .bucket(bucketName)
            .key(keyName)
            .build();

        s3.putObject(putOb, RequestBody.fromString(keyName));
    }
}
```

```
        keys.add(objectId);
    }

    System.out.println(keys.size() + " objects successfully created.");

    // Delete multiple objects in one request.
    Delete del = Delete.builder()
        .objects(keys)
        .build();

    try {
        DeleteObjectsRequest multiObjectDeleteRequest =
        DeleteObjectsRequest.builder()
            .bucket(bucketName)
            .delete(del)
            .build();

        s3.deleteObjects(multiObjectDeleteRequest);
        System.out.println("Multiple objects are deleted!");

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteObjects](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除多個物件。

```
import {
  DeleteObjectsCommand,
  S3Client,
  S3ServiceException,
  waitUntilObjectNotExists,
} from "@aws-sdk/client-s3";

/**
 * Delete multiple objects from an S3 bucket.
 * @param {{ bucketName: string, keys: string[] }}
 */
export const main = async ({ bucketName, keys }) => {
  const client = new S3Client({});

  try {
    const { Deleted } = await client.send(
      new DeleteObjectsCommand({
        Bucket: bucketName,
        Delete: {
          Objects: keys.map((k) => ({ Key: k })),
        },
      }),
    );
    for (const key in keys) {
      await waitUntilObjectNotExists(
        { client },
        { Bucket: bucketName, Key: key },
      );
    }
    console.log(
      `Successfully deleted ${Deleted.length} objects from S3 bucket. Deleted objects:`,
    );
    console.log(Deleted.map((d) => ` • ${d.Key}`).join("\n"));
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
      console.error(
        `Error from S3 while deleting objects from ${bucketName}. The bucket doesn't exist.`,
      );
    }
  }
}
```



```
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while deleting objects from ${bucketName}.
            ${caught.name}: ${caught.message}`,
        );
    } else {
        throw caught;
    }
}
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [DeleteObjects](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun deleteBucketObjects(
    bucketName: String,
    objectName: String,
) {
    val objectId =
        ObjectIdentifier {
            key = objectName
        }

    val delOb =
        Delete {
            objects = listOf(objectId)
        }

    val request =
        DeleteObjectsRequest {
```

```
        bucket = bucketName
        delete = del0b
    }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        s3.deleteObjects(request)
        println("$objectName was deleted from $bucketName")
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [DeleteObjects](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除金鑰清單中一整組物件。

```
$s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

try {
    $objects = [];
    foreach ($contents['Contents'] as $content) {
        $objects[] = [
            'Key' => $content['Key'],
        ];
    }
    $this->s3client->deleteObjects([
        'Bucket' => $this->bucketName,
        'Delete' => [
            'Objects' => $objects,
        ],
    ]);
    $check = $this->s3client->listObjectsV2([
        'Bucket' => $this->bucketName,
```

```
]);  
if (isset($check['Contents']) && count($check['Contents']) > 0) {  
    throw new Exception("Bucket wasn't empty.");  
}  
echo "Deleted all objects and folders from $this->bucketName.\n";  
} catch (Exception $exception) {  
    echo "Failed to delete $fileName from $this->bucketName with error:  
" . $exception->getMessage();  
    exit("Please fix error with object deletion before continuing.");  
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [DeleteObjects](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會從儲存貯體 "test-files" 中移除物件 "sample.txt"。在執行命令之前，系統會提示您確認；若要隱藏提示，請使用 -Force 切換變數。

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt
```

範例 2：此命令會移除儲存貯體 "test-files" 中物件 "sample.txt" 的指定版本，假設儲存貯體已設定為啟用物件版本。

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -VersionId  
HLbxnx6V9omT6AQYVpks8mmFKQcejpqt
```

範例 3：此命令會將儲存貯體 "test-files" 中的物件 "sample1.txt"、"sample2.txt" 和 "sample3.txt" 移除，以成為單一批次操作。無論是刪除成功或錯誤狀態，服務回應都會列出處理的所有索引鍵。若要僅取得無法由服務處理之索引鍵的錯誤，請新增 -ReportErrorsOnly 參數 (也可以使用別名 -Quiet 指定此參數)。

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -KeyCollection @( "sample1.txt",  
"sample2.txt", "sample3.txt" )
```

範例 4：此範例搭配 -KeyCollection 參數使用內嵌表達式，以取得要刪除之物件的索引鍵。Get-S3Object 會傳回 Amazon.S3.Model.S3Object 執行個體的集合，每個執行個體都有一個字串類型的索引鍵成員來識別物件。

```
Remove-S3Object -bucketname "amzn-s3-demo-bucket" -KeyCollection (Get-S3Object
"test-files" -KeyPrefix "prefix/subprefix" | select -ExpandProperty Key)
```

範例 5：此範例會取得儲存貯體中具有索引鍵字首「字首/子字首」的所有物件，並將其刪除。請注意，一次只會處理一個傳入物件。對於大型集合，請考慮將集合傳遞至 Cmdlet 的 -InputObject (別名 -S3ObjectCollection) 參數，透過一次服務呼叫來批次執行刪除作業。

```
Get-S3Object -BucketName "amzn-s3-demo-bucket" -KeyPrefix "prefix/subprefix" |
Remove-S3Object -Force
```

範例 6：此範例會將代表刪除標記的 Amazon.S3.Model.S3ObjectVersion 執行個體集合傳輸至 Cmdlet，以進行刪除。請注意，一次只會處理一個傳入物件。對於大型集合，請考慮將集合傳遞至 Cmdlet 的 -InputObject (別名 -S3ObjectCollection) 參數，透過一次服務呼叫來批次執行刪除作業。

```
(Get-S3Version -BucketName "amzn-s3-demo-bucket").Versions | Where
{$_ .IsDeleteMarker -eq "True"} | Remove-S3Object -Force
```

範例 7：此指令碼示範如何透過建構要與 -KeyAndVersionCollection 參數搭配使用的物件陣列，執行一組物件的批次刪除 (在此案例中為刪除標記)。

```
$keyVersions = @()
$markers = (Get-S3Version -BucketName $BucketName).Versions | Where
{$_ .IsDeleteMarker -eq "True"}
foreach ($marker in $markers) { $keyVersions += @{ Key = $marker.Key; VersionId =
$marker.VersionId } }
Remove-S3Object -BucketName $BucketName -KeyAndVersionCollection $keyVersions -
Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeleteObjects](#)。

Tools for PowerShell V5

範例 1：此命令會從儲存貯體 "test-files" 中移除物件 "sample.txt"。在執行命令之前，系統會提示您確認；若要隱藏提示，請使用 -Force 切換變數。

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt
```

範例 2：此命令會移除儲存貯體 "test-files" 中物件 "sample.txt" 的指定版本，假設儲存貯體已設定為啟用物件版本。

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -VersionId
HLbxnx6V9omT6AQYVpks8mmFKQcejpqt
```

範例 3：此命令會將儲存貯體 "test-files" 中的物件 "sample1.txt"、"sample2.txt" 和 "sample3.txt" 移除，以成為單一批次操作。無論是刪除成功或錯誤狀態，服務回應都會列出處理的所有索引鍵。若要僅取得無法由服務處理之索引鍵的錯誤，請新增 -ReportErrorsOnly 參數 (也可以使用別名 -Quiet 指定此參數)。

```
Remove-S3Object -BucketName amzn-s3-demo-bucket -KeyCollection @( "sample1.txt",
"sample2.txt", "sample3.txt" )
```

範例 4：此範例搭配 -KeyCollection 參數使用內嵌表達式，以取得要刪除之物件的索引鍵。Get-S3Object 會傳回 Amazon.S3.Model.S3Object 執行個體的集合，每個執行個體都有一個字串類型的索引鍵成員來識別物件。

```
Remove-S3Object -bucketname "amzn-s3-demo-bucket" -KeyCollection (Get-S3Object
"test-files" -KeyPrefix "prefix/subprefix" | select -ExpandProperty Key)
```

範例 5：此範例會取得儲存貯體中具有索引鍵字首「字首/子字首」的所有物件，並將其刪除。請注意，一次只會處理一個傳入物件。對於大型集合，請考慮將集合傳遞至 Cmdlet 的 -InputObject (別名 -S3ObjectCollection) 參數，透過一次服務呼叫來批次執行刪除作業。

```
Get-S3Object -BucketName "amzn-s3-demo-bucket" -KeyPrefix "prefix/subprefix" |
Remove-S3Object -Force
```

範例 6：此範例會將代表刪除標記的 Amazon.S3.Model.S3ObjectVersion 執行個體集合傳輸至 Cmdlet，以進行刪除。請注意，一次只會處理一個傳入物件。對於大型集合，請考慮將集合傳遞至 Cmdlet 的 -InputObject (別名 -S3ObjectCollection) 參數，透過一次服務呼叫來批次執行刪除作業。

```
(Get-S3Version -BucketName "amzn-s3-demo-bucket").Versions | Where
{$_ .IsDeleteMarker -eq "True"} | Remove-S3Object -Force
```

範例 7：此指令碼示範如何透過建構要與 -KeyAndVersionCollection 參數搭配使用的物件陣列，執行一組物件的批次刪除 (在此案例中為刪除標記)。

```
$keyVersions = @()
$markers = (Get-S3Version -BucketName $BucketName).Versions | Where
{ $_.IsDeleteMarker -eq "True" }
foreach ($marker in $markers) { $keyVersions += @{ Key = $marker.Key; VersionId =
$marker.VersionId } }
Remove-S3Object -BucketName $BucketName -KeyAndVersionCollection $keyVersions -
Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeleteObjects](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用物件金鑰清單刪除一整組物件。

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
        in Boto3
                           that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key

    @staticmethod
    def delete_objects(bucket, object_keys):
        """
        Removes a list of objects from a bucket.
        This operation is done as a batch in a single request.
        """
```

```

        :param bucket: The bucket that contains the objects. This is a Boto3
        Bucket
                        resource.
        :param object_keys: The list of keys that identify the objects to remove.
        :return: The response that contains data about which objects were deleted
                and any that could not be deleted.
        """
        try:
            response = bucket.delete_objects(
                Delete={"Objects": [{"Key": key} for key in object_keys]}
            )
            if "Deleted" in response:
                logger.info(
                    "Deleted objects '%s' from bucket '%s'.",
                    [del_obj["Key"] for del_obj in response["Deleted"]],
                    bucket.name,
                )
            if "Errors" in response:
                logger.warning(
                    "Could not delete objects '%s' from bucket '%s'.",
                    [
                        f"{del_obj['Key']}: {del_obj['Code']}"
                        for del_obj in response["Errors"]
                    ],
                    bucket.name,
                )
        except ClientError:
            logger.exception("Couldn't delete any objects from bucket %s.",
                             bucket.name)
            raise
        else:
            return response

```

刪除儲存貯體中的所有物件。

```

class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """

```

```

        :param s3_object: A Boto3 Object resource. This is a high-level resource
in Boto3
                                that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key

    @staticmethod
    def empty_bucket(bucket):
        """
        Remove all objects from a bucket.

        :param bucket: The bucket to empty. This is a Boto3 Bucket resource.
        """
        try:
            bucket.objects.delete()
            logger.info("Emptied bucket '%s'.", bucket.name)
        except ClientError:
            logger.exception("Couldn't empty bucket '%s'.", bucket.name)
            raise

```

會透過刪除其所有版本，永久刪除已建立版本的物件。

```

def permanently_delete_object(bucket, object_key):
    """
    Permanently deletes a versioned object by deleting all of its versions.

    Usage is shown in the usage_demo_single_object function at the end of this
    module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to delete.
    """
    try:
        bucket.object_versions.filter(Prefix=object_key).delete()
        logger.info("Permanently deleted all versions of object %s.", object_key)
    except ClientError:
        logger.exception("Couldn't delete all versions of %s.", object_key)
        raise

```


- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [DeleteObjects](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
# Deletes the objects in an Amazon S3 bucket and deletes the bucket.
#
# @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
def delete_bucket(bucket)
  puts("\nDo you want to delete all of the objects as well as the bucket (y/n)?")
  answer = gets.chomp.downcase
  if answer == 'y'
    bucket.objects.batch_delete!
    bucket.delete
    puts("Emptied and deleted bucket #{bucket.name}.\n")
  end
rescue Aws::Errors::ServiceError => e
  puts("Couldn't empty and delete bucket #{bucket.name}.")
  puts("\t#{e.code}: #{e.message}")
  raise
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [DeleteObjects](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// Delete the objects in a bucket.
pub async fn delete_objects(
    client: &aws_sdk_s3::Client,
    bucket_name: &str,
    objects_to_delete: Vec<String>,
) -> Result<(), S3ExampleError> {
    // Push into a mut vector to use `?` early return errors while building
    object keys.
    let mut delete_object_ids: Vec<aws_sdk_s3::types::ObjectIdentifier> = vec![];
    for obj in objects_to_delete {
        let obj_id = aws_sdk_s3::types::ObjectIdentifier::builder()
            .key(obj)
            .build()
            .map_err(|err| {
                S3ExampleError::new(format!("Failed to build key for
delete_object: {err:?}"))
            })?;
        delete_object_ids.push(obj_id);
    }

    client
        .delete_objects()
        .bucket(bucket_name)
        .delete(
            aws_sdk_s3::types::Delete::builder()
                .set_objects(Some(delete_object_ids))
                .build()
                .map_err(|err| {
                    S3ExampleError::new(format!("Failed to build delete_object
input {err:?}"))
                })?,
        )
}
```

```
        .send()  
        .await?;  
    Ok(())  
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [DeleteObjects](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3->deleteobjects(          " oo_result is returned for  
testing purposes. "  
    iv_bucket = iv_bucket_name  
    io_delete = NEW /aws1/cl_s3_delete( it_objects = it_object_keys ) ).  
    MESSAGE 'Objects deleted from S3 bucket.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [DeleteObjects](#)。

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import AWSS3

public func deleteObjects(bucket: String, keys: [String]) async throws {
    let input = DeleteObjectsInput(
        bucket: bucket,
        delete: S3ClientTypes.Delete(
            objects: keys.map { S3ClientTypes.ObjectIdentifier(key: $0) },
            quiet: true
        )
    )

    do {
        _ = try await client.deleteObjects(input: input)
    } catch {
        print("ERROR: deleteObjects:", dump(error))
        throw error
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Swift 的 AWS SDK API 參考》中的 [DeleteObjects](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeletePublicAccessBlock** 與 CLI

下列程式碼範例示範如何使用 DeletePublicAccessBlock。

CLI

AWS CLI

刪除儲存貯體的封鎖公開存取組態

下列 `delete-public-access-block` 範例會移除指定儲存貯體上的封鎖公開存取組態。

```
aws s3api delete-public-access-block \  
  --bucket amzn-s3-demo-bucket
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeletePublicAccessBlock](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會關閉給定儲存貯體的封鎖公開存取設定。

```
Remove-S3PublicAccessBlock -BucketName 'amzn-s3-demo-bucket' -Force -Select  
'^BucketName'
```

輸出：

```
amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [DeletePublicAccessBlock](#)。

Tools for PowerShell V5

範例 1：此命令會關閉給定儲存貯體的封鎖公開存取設定。

```
Remove-S3PublicAccessBlock -BucketName 'amzn-s3-demo-bucket' -Force -Select  
'^BucketName'
```

輸出：

```
amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [DeletePublicAccessBlock](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetBucketAccelerateConfiguration** 與 CLI

下列程式碼範例示範如何使用 `GetBucketAccelerateConfiguration`。

CLI

AWS CLI

擷取儲存貯體的加速組態

下列 `get-bucket-accelerate-configuration` 範例會擷取指定儲存貯體的加速組態。

```
aws s3api get-bucket-accelerate-configuration \
  --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "Status": "Enabled"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketAccelerateConfiguration](#)。

PowerShell

Tools for PowerShell V4

範例 1：如果為指定的儲存貯體啟用傳輸加速設定，此命令會傳回已啟用的值。

```
Get-S3BucketAccelerateConfiguration -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Value
-----
Enabled
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketAccelerateConfiguration](#)。

Tools for PowerShell V5

範例 1：如果為指定的儲存貯體啟用傳輸加速設定，此命令會傳回已啟用的值。

```
Get-S3BucketAccelerateConfiguration -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Value
-----
Enabled
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketAccelerateConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetBucketAcl 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetBucketAcl。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [檢查儲存貯體是否存在](#)
- [管理存取控制清單 \(ACL\)](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
    /// <summary>
    /// Get the access control list (ACL) for the new bucket.
    /// </summary>
    /// <param name="client">The initialized client object used to get the
    /// access control list (ACL) of the bucket.</param>
    /// <param name="newBucketName">The name of the newly created bucket.</
param>
    /// <returns>An S3AccessControlList.</returns>
    public static async Task<S3AccessControlList>
    GetACLForBucketAsync(IAmazonS3 client, string newBucketName)
    {
        // Retrieve bucket ACL to show that the ACL was properly applied to
        // the new bucket.
        GetACLResponse getACLResponse = await client.GetACLAsync(new
    GetACLRequest
    {
        BucketName = newBucketName,
    });

        return getACLResponse.AccessControlList;
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetBucketAcl](#)。

C++

適用於 C++ 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::getBucketAcl(const Aws::String &bucketName,
                              const Aws::S3::S3ClientConfiguration &clientConfig)
{
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::GetBucketAclRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::GetBucketAclOutcome outcome =
        s3Client.GetBucketAcl(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: getBucketAcl: "
                    << err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
    } else {
        Aws::Vector<Aws::S3::Model::Grant> grants =
            outcome.GetResult().GetGrants();

        for (auto it = grants.begin(); it != grants.end(); it++) {
            Aws::S3::Model::Grant grant = *it;
            Aws::S3::Model::Grantee grantee = grant.GetGrantee();

            std::cout << "For bucket " << bucketName << ": "
                      << std::endl << std::endl;

            if (grantee.TypeHasBeenSet()) {
                std::cout << "Type:          "
                          << getGranteeTypeString(grantee.GetType()) <<
std::endl;
            }
        }
    }
}
```

```

        if (grantee.DisplayNameHasBeenSet()) {
            std::cout << "Display name: "
                      << grantee.GetDisplayName() << std::endl;
        }

        if (grantee.EmailAddressHasBeenSet()) {
            std::cout << "Email address: "
                      << grantee.GetEmailAddress() << std::endl;
        }

        if (grantee.IDHasBeenSet()) {
            std::cout << "ID: "
                      << grantee.GetID() << std::endl;
        }

        if (grantee.URIHasBeenSet()) {
            std::cout << "URI: "
                      << grantee.GetURI() << std::endl;
        }

        std::cout << "Permission: " <<
                  getPermissionString(grant.GetPermission()) <<
                  std::endl << std::endl;
    }
}

return outcome.IsSuccess();
}

//! Routine which converts a built-in type enumeration to a human-readable
    string.
    /*!
    \param type: Type enumeration.
    \return String: Human-readable string.
    */

    Aws::String getGranteeTypeString(const Aws::S3::Model::Type &type) {
        switch (type) {
            case Aws::S3::Model::Type::AmazonCustomerByEmail:
                return "Email address of an AWS account";
            case Aws::S3::Model::Type::CanonicalUser:
                return "Canonical user ID of an AWS account";
            case Aws::S3::Model::Type::Group:

```

```

        return "Predefined Amazon S3 group";
    case Aws::S3::Model::Type::NOT_SET:
        return "Not set";
    default:
        return "Type unknown";
    }
}

//! Routine which converts a built-in type enumeration to a human-readable
    string.
/*!
    \param permission: Permission enumeration.
    \return String: Human-readable string.
*/

Aws::String getPermissionString(const Aws::S3::Model::Permission &permission) {
    switch (permission) {
        case Aws::S3::Model::Permission::FULL_CONTROL:
            return "Can list objects in this bucket, create/overwrite/delete "
                "objects in this bucket, and read/write this "
                "bucket's permissions";
        case Aws::S3::Model::Permission::NOT_SET:
            return "Permission not set";
        case Aws::S3::Model::Permission::READ:
            return "Can list objects in this bucket";
        case Aws::S3::Model::Permission::READ_ACP:
            return "Can read this bucket's permissions";
        case Aws::S3::Model::Permission::WRITE:
            return "Can create, overwrite, and delete objects in this bucket";
        case Aws::S3::Model::Permission::WRITE_ACP:
            return "Can write this bucket's permissions";
        default:
            return "Permission unknown";
    }

    return "Permission unknown";
}

```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [GetBucketAcl](#)。

CLI

AWS CLI

下列命令會擷取名為 `amzn-s3-demo-bucket` 之儲存貯體的存取控制清單：

```
aws s3api get-bucket-acl --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "Owner": {
    "DisplayName": "my-username",
    "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
  },
  "Grants": [
    {
      "Grantee": {
        "DisplayName": "my-username",
        "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
      },
      "Permission": "FULL_CONTROL"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketAcl](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.services.s3.model.S3Exception;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectAclRequest;
import software.amazon.awssdk.services.s3.model.GetObjectAclResponse;
import software.amazon.awssdk.services.s3.model.Grant;

import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */

public class GetAcl {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <bucketName> <objectKey>

            Where:
                bucketName - The Amazon S3 bucket to get the access control list
(ACL) for.
                objectKey - The object to get the ACL for.\s
            "";

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String objectKey = args[1];
        System.out.println("Retrieving ACL for object: " + objectKey);
        System.out.println("in bucket: " + bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();
```

```
        getBucketACL(s3, objectKey, bucketName);
        s3.close();
        System.out.println("Done!");
    }

    /**
     * Retrieves the Access Control List (ACL) for an object in an Amazon S3
     * bucket.
     *
     * @param s3 The S3Client object used to interact with the Amazon S3 service.
     * @param objectKey The key of the object for which the ACL is to be
     * retrieved.
     * @param bucketName The name of the bucket containing the object.
     * @return The ID of the grantee who has permission on the object, or an
     * empty string if an error occurs.
     */
    public static String getBucketACL(S3Client s3, String objectKey, String
    bucketName) {
        try {
            GetObjectAclRequest aclReq = GetObjectAclRequest.builder()
                .bucket(bucketName)
                .key(objectKey)
                .build();

            GetObjectAclResponse aclRes = s3.getObjectAcl(aclReq);
            List<Grant> grants = aclRes.grants();
            String grantee = "";
            for (Grant grant : grants) {
                System.out.format("  %s: %s\n", grant.grantee().id(),
    grant.permission());
                grantee = grant.grantee().id();
            }

            return grantee;
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return "";
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetBucketAcl](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得 ACL 許可。

```
import {
  GetBucketAclCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Retrieves the Access Control List (ACL) for an S3 bucket.
 * @param {{ bucketName: string }}
 */
export const main = async ({ bucketName }) => {
  const client = new S3Client({});

  try {
    const response = await client.send(
      new GetBucketAclCommand({
        Bucket: bucketName,
      }),
    );
    console.log(`ACL for bucket "${bucketName}":`);
    console.log(JSON.stringify(response, null, 2));
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
```

```
        console.error(
            `Error from S3 while getting ACL for ${bucketName}. The bucket doesn't
            exist.` ,
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while getting ACL for ${bucketName}. ${caught.name}:
            ${caught.message}` ,
        );
    } else {
        throw caught;
    }
}
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的[GetBucketAcl](#)。

PowerShell

Tools for PowerShell V5

範例 1：命令會取得 S3 物件之物件擁有者的詳細資訊。

```
(Get-S3BucketACL -BucketName 'amzn-s3-demo-bucket' -Select *).Owner
```

輸出：

```
DisplayName Id
----- --
testusername
9988776a6554433d22f1100112e334acb45566778899009e9887bd7f66c5f544
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的[GetBucketAcl](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def get_acl(self):
        """
        Get the ACL of the bucket.

        :return: The ACL of the bucket.
        """
        try:
            acl = self.bucket.Acl()
            logger.info(
                "Got ACL for bucket %s. Owner is %s.", self.bucket.name,
                acl.owner
            )
        except ClientError:
            logger.exception("Couldn't get ACL for bucket %s.", self.bucket.name)
            raise
        else:
            return acl
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [GetBucketAcl](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3->getbucketacl(          " oo_result is returned for  
testing purposes. "  
    iv_bucket = iv_bucket_name ).  
    MESSAGE 'Retrieved bucket ACL.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetBucketAcl](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetBucketAnalyticsConfiguration** 與 CLI

下列程式碼範例示範如何使用 `GetBucketAnalyticsConfiguration`。

CLI

AWS CLI

擷取具有特定 ID 之儲存貯體的分析組態

下列 `get-bucket-analytics-configuration` 範例顯示指定儲存貯體和 ID 的分析組態。

```
aws s3api get-bucket-analytics-configuration \  
  --bucket amzn-s3-demo-bucket \  
  --id 1
```

輸出：

```
{  
  "AnalyticsConfiguration": {  
    "StorageClassAnalysis": {},  
    "Id": "1"  
  }  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketAnalyticsConfiguration](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回指定 S3 儲存貯體中名為 'testfilter' 的分析篩選條件詳細資訊。

```
Get-S3BucketAnalyticsConfiguration -BucketName 'amzn-s3-demo-bucket' -AnalyticsId  
'testfilter'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketAnalyticsConfiguration](#)。

Tools for PowerShell V5

範例 1：此命令會傳回指定 S3 儲存貯體中名為 'testfilter' 的分析篩選條件詳細資訊。

```
Get-S3BucketAnalyticsConfiguration -BucketName 'amzn-s3-demo-bucket' -AnalyticsId  
'testfilter'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketAnalyticsConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetBucketCors 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetBucketCors。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Retrieve the CORS configuration applied to the Amazon S3 bucket.
/// </summary>
/// <param name="client">The initialized Amazon S3 client object used
/// to retrieve the CORS configuration.</param>
/// <returns>The created CORS configuration object.</returns>
private static async Task<CORSConfiguration>
RetrieveCORSConfigurationAsync(AmazonS3Client client)
{
    GetCORSConfigurationRequest request = new
GetCORSConfigurationRequest()
    {
        BucketName = BucketName,
    };
    var response = await client.GetCORSConfigurationAsync(request);
    var configuration = response.Configuration;
    PrintCORSRules(configuration);
    return configuration;
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetBucketCors](#)。

CLI

AWS CLI

下列命令會擷取名為 `amzn-s3-demo-bucket` 之儲存貯體的跨來源資源共享組態：

```
aws s3api get-bucket-cors --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "CORSRules": [
    {
      "AllowedHeaders": [
        "*"
      ],
      "ExposeHeaders": [
        "x-amz-server-side-encryption"
      ],
      "AllowedMethods": [
        "PUT",
        "POST",
        "DELETE"
      ],
      "MaxAgeSeconds": 3000,
      "AllowedOrigins": [
        "http://www.example.com"
      ]
    },
    {
      "AllowedHeaders": [
        "Authorization"
      ],
      "MaxAgeSeconds": 3000,
      "AllowedMethods": [
        "GET"
      ],
      "AllowedOrigins": [
        "*"
      ]
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketCors](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得儲存貯體的 CORS 政策。

```
import {
  GetBucketCorsCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Log the Cross-Origin Resource Sharing (CORS) configuration information
 * set for the bucket.
 * @param {{ bucketName: string }}
 */
export const main = async ({ bucketName }) => {
  const client = new S3Client({});
  const command = new GetBucketCorsCommand({
    Bucket: bucketName,
  });

  try {
    const { CORSRules } = await client.send(command);
    console.log(JSON.stringify(CORSRules));
    CORSRules.forEach((cr, i) => {
      console.log(
        `\nCORSRule ${i + 1}`,
        `\n${"-"}.repeat(10)`,
        `\nAllowedHeaders: ${cr.AllowedHeaders}`,
        `\nAllowedMethods: ${cr.AllowedMethods}`,
        `\nAllowedOrigins: ${cr.AllowedOrigins}`,
        `\nExposeHeaders: ${cr.ExposeHeaders}`,
      );
    });
  } catch (err) {
    if (err instanceof S3ServiceException) {
      console.log(`S3ServiceException: ${err.message}`);
    } else {
      console.log(`Error: ${err.message}`);
    }
  }
};
```

```
        `\nMaxAgeSeconds: ${cr.MaxAgeSeconds}`,\n    );\n});\n} catch (caught) {\n    if (\n        caught instanceof S3ServiceException &&\n        caught.name === "NoSuchBucket"\n    ) {\n        console.error(\n            `Error from S3 while getting bucket CORS rules for ${bucketName}. The\n            bucket doesn't exist.`,\n        );\n    } else if (caught instanceof S3ServiceException) {\n        console.error(\n            `Error from S3 while getting bucket CORS rules for ${bucketName}.\n            ${caught.name}: ${caught.message}`,\n        );\n    } else {\n        throw caught;\n    }\n}\n};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [GetBucketCors](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:\n    """Encapsulates S3 bucket actions."""
```

```
def __init__(self, bucket):
    """
    :param bucket: A Boto3 Bucket resource. This is a high-level resource in
    Boto3
                    that wraps bucket actions in a class-like structure.
    """
    self.bucket = bucket
    self.name = bucket.name

def get_cors(self):
    """
    Get the CORS rules for the bucket.

    :return The CORS rules for the specified bucket.
    """
    try:
        cors = self.bucket.Cors()
        logger.info(
            "Got CORS rules %s for bucket '%s'.", cors.cors_rules,
            self.bucket.name
        )
    except ClientError:
        logger.exception(("Couldn't get CORS for bucket %s.",
            self.bucket.name))
        raise
    else:
        return cors
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考》中的 [GetBucketCors](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-s3'

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors

  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with
  # an existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Gets the CORS configuration of a bucket.
  #
  # @return [Aws::S3::Type::GetBucketCorsOutput, nil] The current CORS
  # configuration for the bucket.
  def cors
    @bucket_cors.data
    rescue Aws::Errors::ServiceError => e
      puts "Couldn't get CORS configuration for #{@bucket_cors.bucket.name}. Here's
      why: #{e.message}"
      nil
    end
  end
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [GetBucketCors](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3->getbucketcors(          " oo_result is returned for  
testing purposes. "  
    iv_bucket = iv_bucket_name ).  
    MESSAGE 'Retrieved bucket CORS configuration.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetBucketCors](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetBucketEncryption 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetBucketEncryption。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

/// <summary>
/// Get and print the encryption settings of a bucket.
/// </summary>
/// <param name="bucketName">Name of the bucket.</param>
/// <returns>Async task.</returns>
public static async Task GetEncryptionSettings(string bucketName)
{
    // Check and print the bucket encryption settings.
    Console.WriteLine($"Getting encryption settings for bucket
{bucketName}.");

    try
    {
        var settings =
            await _s3Client.GetBucketEncryptionAsync(
                new GetBucketEncryptionRequest() { BucketName =
bucketName });

        foreach (var encryptionSettings in
settings?.ServerSideEncryptionConfiguration?.ServerSideEncryptionRules!)
        {
            Console.WriteLine(
                $"Algorithm:
{encryptionSettings.ServerSideEncryptionByDefault.ServerSideEncryptionAlgorithm}");
            Console.WriteLine(
                $"Key:
{encryptionSettings.ServerSideEncryptionByDefault.ServerSideEncryptionKeyManagementServi
}
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine(ex.ErrorCode == "InvalidBucketName"
                ? $"Bucket {bucketName} was not found."
                : $"Unable to get bucket encryption for bucket {bucketName},
{ex.Message}");
        }
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetBucketEncryption](#)。

CLI

AWS CLI

擷取儲存貯體的伺服器端加密組態

下列 `get-bucket-encryption` 範例會擷取儲存貯體 `amzn-s3-demo-bucket` 的伺服器端加密組態。

```
aws s3api get-bucket-encryption \
  --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "ServerSideEncryptionConfiguration": {
    "Rules": [
      {
        "ApplyServerSideEncryptionByDefault": {
          "SSEAlgorithm": "AES256"
        }
      }
    ]
  }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketEncryption](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回與指定儲存貯體相關聯的所有伺服器端加密規則。

```
Get-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketEncryption](#)。

Tools for PowerShell V5

範例 1：此命令會傳回與指定儲存貯體相關聯的所有伺服器端加密規則。

```
Get-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketEncryption](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetBucketInventoryConfiguration** 與 CLI

下列程式碼範例示範如何使用 GetBucketInventoryConfiguration。

CLI

AWS CLI

擷取儲存貯體的庫存組態

下列 get-bucket-inventory-configuration 範例會擷取 ID 為 1 之指定儲存貯體的庫存組態。

```
aws s3api get-bucket-inventory-configuration \
  --bucket amzn-s3-demo-bucket \
  --id 1
```

輸出：

```
{
  "InventoryConfiguration": {
    "IsEnabled": true,
    "Destination": {
      "S3BucketDestination": {
        "Format": "ORC",
        "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket",
        "AccountId": "123456789012"
      }
    },
    "IncludedObjectVersions": "Current",
    "Id": "1",
    "Schedule": {
```

```
        "Frequency": "Weekly"
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketInventoryConfiguration](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回給定 S3 儲存貯體之名為 'testinventory' 的庫存詳細資訊。

```
Get-S3BucketInventoryConfiguration -BucketName 'amzn-s3-demo-bucket' -InventoryId
'testinventory'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketInventoryConfiguration](#)。

Tools for PowerShell V5

範例 1：此命令會傳回給定 S3 儲存貯體之名為 'testinventory' 的庫存詳細資訊。

```
Get-S3BucketInventoryConfiguration -BucketName 'amzn-s3-demo-bucket' -InventoryId
'testinventory'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketInventoryConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetBucketLifecycleConfiguration 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetBucketLifecycleConfiguration。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Returns a configuration object for the supplied bucket name.
/// </summary>
/// <param name="client">The S3 client object used to call
/// the GetLifecycleConfigurationAsync method.</param>
/// <param name="bucketName">The name of the S3 bucket for which a
/// configuration will be created.</param>
/// <returns>Returns a new LifecycleConfiguration object.</returns>
public static async Task<LifecycleConfiguration>
RetrieveLifecycleConfigAsync(IAmazonS3 client, string bucketName)
{
    var request = new GetLifecycleConfigurationRequest()
    {
        BucketName = bucketName,
    };
    var response = await client.GetLifecycleConfigurationAsync(request);
    var configuration = response.Configuration;
    return configuration;
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetBucketLifecycleConfiguration](#)。

CLI

AWS CLI

下列命令會擷取名為 `amzn-s3-demo-bucket` 之儲存貯體的生命週期組態：

```
aws s3api get-bucket-lifecycle-configuration --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "Rules": [
    {
      "ID": "Move rotated logs to Glacier",
      "Prefix": "rotated/",
      "Status": "Enabled",
      "Transitions": [
        {
          "Date": "2015-11-10T00:00:00.000Z",
          "StorageClass": "GLACIER"
        }
      ]
    },
    {
      "Status": "Enabled",
      "Prefix": "",
      "NoncurrentVersionTransitions": [
        {
          "NoncurrentDays": 0,
          "StorageClass": "GLACIER"
        }
      ],
      "ID": "Move old versions to Glacier"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI API 參考》中的 [GetBucketLifecycleConfiguration](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。


```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def get_lifecycle_configuration(self):
        """
        Get the lifecycle configuration of the bucket.

        :return: The lifecycle rules of the specified bucket.
        """
        try:
            config = self.bucket.LifecycleConfiguration()
            logger.info(
                "Got lifecycle rules %s for bucket '%s'.",
                config.rules,
                self.bucket.name,
            )
        except:
            logger.exception(
                "Couldn't get lifecycle rules for bucket '%s'.", self.bucket.name
            )
            raise
        else:
            return config.rules
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [GetBucketLifecycleConfiguration](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3->getbucketlifecycleconf(          " oo_result is  
returned for testing purposes. "  
    iv_bucket = iv_bucket_name ).  
    MESSAGE 'Retrieved bucket lifecycle configuration.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetBucketLifecycleConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetBucketLocation 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetBucketLocation。

CLI

AWS CLI

如果存在限制，以下命令會擷取名為 amzn-s3-demo-bucket 之儲存貯體的位置限制條件：

```
aws s3api get-bucket-location --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "LocationConstraint": "us-west-2"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketLocation](#)。

PowerShell

Tools for PowerShell V4

範例 1：如果存在限制，此命令會傳回儲存貯體 'amzn-s3-demo-bucket' 的位置限制。

```
Get-S3BucketLocation -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Value
-----
ap-south-1
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketLocation](#)。

Tools for PowerShell V5

範例 1：如果存在限制，此命令會傳回儲存貯體 'amzn-s3-demo-bucket' 的位置限制。

```
Get-S3BucketLocation -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Value
-----
ap-south-1
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketLocation](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
async fn show_buckets(
    strict: bool,
    client: &Client,
    region: BucketLocationConstraint,
) -> Result<(), S3ExampleError> {
    let mut buckets = client.list_buckets().into_paginator().send();

    let mut num_buckets = 0;
    let mut in_region = 0;

    while let Some(Ok(output)) = buckets.next().await {
        for bucket in output.buckets() {
            num_buckets += 1;
            if strict {
                let r = client
                    .get_bucket_location()
                    .bucket(bucket.name().unwrap_or_default())
                    .send()
                    .await?;

                if r.location_constraint() == Some(&region) {
                    println!("{}", bucket.name().unwrap_or_default());
                    in_region += 1;
                }
            } else {
                println!("{}", bucket.name().unwrap_or_default());
            }
        }
    }

    println!();
    if strict {
```

```
println!(
    "Found {} buckets in the {} region out of a total of {} buckets.",
    in_region, region, num_buckets
);
} else {
    println!("Found {} buckets in all regions.", num_buckets);
}

Ok(())
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [GetBucketLocation](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetBucketLogging** 與 CLI

下列程式碼範例示範如何使用 GetBucketLogging。

CLI

AWS CLI

擷取儲存貯體的記錄狀態

下列 get-bucket-logging 範例會擷取指定儲存貯體的記錄狀態。

```
aws s3api get-bucket-logging \
    --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "LoggingEnabled": {
    "TargetPrefix": "",
    "TargetBucket": "amzn-s3-demo-bucket-logs"
  }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketLogging](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回指定儲存貯體的記錄狀態。

```
Get-S3BucketLogging -BucketName 'amzn-s3-demo-bucket'
```

輸出：

TargetBucketName	Grants	TargetPrefix
testbucket1	{}	testprefix

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketLogging](#)。

Tools for PowerShell V5

範例 1：此命令會傳回指定儲存貯體的記錄狀態。

```
Get-S3BucketLogging -BucketName 'amzn-s3-demo-bucket'
```

輸出：

TargetBucketName	Grants	TargetPrefix
testbucket1	{}	testprefix

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketLogging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetBucketMetricsConfiguration** 與 CLI

下列程式碼範例示範如何使用 **GetBucketMetricsConfiguration**。

CLI

AWS CLI

擷取具有特定 ID 之儲存貯體的指標組態

下列 `get-bucket-metrics-configuration` 範例顯示指定儲存貯體和 ID 的指標組態。

```
aws s3api get-bucket-metrics-configuration \  
  --bucket amzn-s3-demo-bucket \  
  --id 123
```

輸出：

```
{  
  "MetricsConfiguration": {  
    "Filter": {  
      "Prefix": "logs"  
    },  
    "Id": "123"  
  }  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketMetricsConfiguration](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回給定 S3 儲存貯體之名為 'testfilter' 的指標篩選條件詳細資訊。

```
Get-S3BucketMetricsConfiguration -BucketName 'amzn-s3-demo-bucket' -MetricsId  
'testfilter'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketMetricsConfiguration](#)。

Tools for PowerShell V5

範例 1：此命令會傳回給定 S3 儲存貯體之名為 'testfilter' 的指標篩選條件詳細資訊。

```
Get-S3BucketMetricsConfiguration -BucketName 'amzn-s3-demo-bucket' -MetricsId  
'testfilter'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketMetricsConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetBucketNotification** 與 CLI

下列程式碼範例示範如何使用 GetBucketNotification。

CLI

AWS CLI

下列命令會擷取名為 amzn-s3-demo-bucket 之儲存貯體的通知組態：

```
aws s3api get-bucket-notification --bucket amzn-s3-demo-bucket
```

輸出：

```
{  
  "TopicConfiguration": {  
    "Topic": "arn:aws:sns:us-west-2:123456789012:my-notification-topic",  
    "Id": "YmQzMmEwM2EjZWVlI0NGItNzVtZjI1MC00ZjgyLWZDBiZWNI",  
    "Event": "s3:ObjectCreated:*",  
    "Events": [  
      "s3:ObjectCreated:*"  
    ]  
  }  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketNotification](#)。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取給定儲存貯體的通知組態


```
Get-S3BucketNotification -BucketName amzn-s3-demo-bucket | select -ExpandProperty  
TopicConfigurations
```

輸出：

```
Id    Topic  
--    -  
mimo  arn:aws:sns:eu-west-1:123456789012:topic-1
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketNotification](#)。

Tools for PowerShell V5

範例 1：此範例會擷取給定儲存貯體的通知組態

```
Get-S3BucketNotification -BucketName amzn-s3-demo-bucket | select -ExpandProperty  
TopicConfigurations
```

輸出：

```
Id    Topic  
--    -  
mimo  arn:aws:sns:eu-west-1:123456789012:topic-1
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketNotification](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetBucketPolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetBucketPolicy。

C++

SDK for C++

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::getBucketPolicy(const Aws::String &bucketName,
                                  const Aws::S3::S3ClientConfiguration
&clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::GetBucketPolicyRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::GetBucketPolicyOutcome outcome =
        s3Client.GetBucketPolicy(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: getBucketPolicy: "
                    << err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
    } else {
        Aws::StringStream policy_stream;
        Aws::String line;

        outcome.GetResult().GetPolicy() >> line;
        policy_stream << line;

        std::cout << "Retrieve the policy for bucket '" << bucketName << "':\n\n"
<<
                policy_stream.str() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [GetBucketPolicy](#)。

CLI

AWS CLI

下列命令會擷取名為 `amzn-s3-demo-bucket` 之儲存貯體的儲存貯體政策：

```
aws s3api get-bucket-policy --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "Policy": "{\"Version\":\"2008-10-17\",\"Statement\": [{\"Sid\":\"\",\"Effect\": \"Allow\", \"Principal\": \"*\", \"Action\": \"s3:GetObject\", \"Resource\": \"arn:aws:s3:::amzn-s3-demo-bucket/*\"}, {\"Sid\":\"\",\"Effect\": \"Deny\", \"Principal\": \"*\", \"Action\": \"s3:GetObject\", \"Resource\": \"arn:aws:s3:::amzn-s3-demo-bucket/secret/*\"}]}"
}
```

取得並放置儲存貯體政策。下列範例示範如何下載 Amazon S3 儲存貯體政策、修改檔案，然後使用 `put-bucket-policy` 套用修改過的儲存貯體政策。若要將儲存貯體政策下載至檔案，您可以執行：

```
aws s3api get-bucket-policy --bucket amzn-s3-demo-bucket --query Policy --output text > policy.json
```

然後視需要修改 `policy.json` 檔案。最後，您可以執行下列動作，將此修改後的政策套用回 S3 儲存貯體：

`policy.json` 檔案 (視需要)。最後，您可以執行下列動作，將此修改後的政策套用回 S3 儲存貯體：

檔案 (視需要)。最後，您可以執行下列動作，將此修改後的政策套用回 S3 儲存貯體：

```
aws s3api put-bucket-policy --bucket amzn-s3-demo-bucket --policy file://policy.json
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketPolicy](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetBucketPolicyRequest;
import software.amazon.awssdk.services.s3.model.GetBucketPolicyResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */

public class GetBucketPolicy {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <bucketName>

            Where:
                bucketName - The Amazon S3 bucket to get the policy from.
            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
    }
}
```

```
String bucketName = args[0];
System.out.format("Getting policy for bucket: \"%s\"\n\n", bucketName);
Region region = Region.US_EAST_1;
S3Client s3 = S3Client.builder()
    .region(region)
    .build();

String polText = getPolicy(s3, bucketName);
System.out.println("Policy Text: " + polText);
s3.close();
}

/**
 * Retrieves the policy for the specified Amazon S3 bucket.
 *
 * @param s3 the {@link S3Client} instance to use for making the request
 * @param bucketName the name of the S3 bucket for which to retrieve the
policy
 * @return the policy text for the specified bucket, or an empty string if an
error occurs
 */
public static String getPolicy(S3Client s3, String bucketName) {
    String policyText;
    System.out.format("Getting policy for bucket: \"%s\"\n\n", bucketName);
    GetBucketPolicyRequest policyReq = GetBucketPolicyRequest.builder()
        .bucket(bucketName)
        .build();

    try {
        GetBucketPolicyResponse policyRes = s3.getBucketPolicy(policyReq);
        policyText = policyRes.policy();
        return policyText;

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }

    return "";
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetBucketPolicy](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得儲存貯體政策。

```
import {
  GetBucketPolicyCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Logs the policy for a specified bucket.
 * @param {{ bucketName: string }}
 */
export const main = async ({ bucketName }) => {
  const client = new S3Client({});

  try {
    const { Policy } = await client.send(
      new GetBucketPolicyCommand({
        Bucket: bucketName,
      }),
    );
    console.log(`Policy for "${bucketName}":\n${Policy}`);
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
      console.error(
        `Error from S3 while getting policy from ${bucketName}. The bucket doesn't exist.`
      );
    }
  }
}
```

```
    );
  } else if (caught instanceof S3ServiceException) {
    console.error(
      `Error from S3 while getting policy from ${bucketName}.  ${caught.name}:
${caught.message}`,
    );
  } else {
    throw caught;
  }
}
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [GetBucketPolicy](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun getPolicy(bucketName: String): String? {
    println("Getting policy for bucket $bucketName")

    val request =
        GetBucketPolicyRequest {
            bucket = bucketName
        }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        val policyRes = s3.getBucketPolicy(request)
        return policyRes.policy
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [GetBucketPolicy](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會輸出與給定 S3 儲存貯體相關聯的儲存貯體政策。

```
Get-S3BucketPolicy -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketPolicy](#)。

Tools for PowerShell V5

範例 1：此命令會輸出與給定 S3 儲存貯體相關聯的儲存貯體政策。

```
Get-S3BucketPolicy -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketPolicy](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
```



```
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def get_policy(self):
        """
        Get the security policy of the bucket.

        :return: The security policy of the specified bucket, in JSON format.
        """
        try:
            policy = self.bucket.Policy()
            logger.info(
                "Got policy %s for bucket '%s'.", policy.policy, self.bucket.name
            )
        except ClientError:
            logger.exception("Couldn't get policy for bucket '%s'.",
self.bucket.name)
            raise
        else:
            return json.loads(policy.policy)
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [GetBucketPolicy](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object
  # configured with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  # Gets the policy of a bucket.
  #
  # @return [Aws::S3::GetBucketPolicyOutput, nil] The current bucket policy.
  def policy
    policy = @bucket_policy.data.policy
    policy.respond_to?(:read) ? policy.read : policy
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get the policy for #{@bucket_policy.bucket.name}. Here's why:
    #{e.message}"
    nil
  end
end

end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [GetBucketPolicy](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.
    oo_result = lo_s3->getbucketpolicy(          " oo_result is returned for
testing purposes. "
    iv_bucket = iv_bucket_name ).
```

```
DATA(lv_policy) = oo_result->get_policy( ).  
MESSAGE 'Retrieved bucket policy.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetBucketPolicy](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetBucketPolicyStatus** 與 CLI

下列程式碼範例示範如何使用 GetBucketPolicyStatus。

CLI

AWS CLI

擷取儲存貯體的政策狀態以指出儲存貯體是否為公有

下列 get-bucket-policy-status 範例會擷取儲存貯體 amzn-s3-demo-bucket 的政策狀態。

```
aws s3api get-bucket-policy-status \  
  --bucket amzn-s3-demo-bucket
```

輸出：

```
{  
  "PolicyStatus": {  
    "IsPublic": false  
  }  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketPolicyStatus](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回給定 S3 儲存貯體的政策狀態，其指出儲存貯體是否為公有。

```
Get-S3BucketPolicyStatus -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketPolicyStatus](#)。

Tools for PowerShell V5

範例 1：此命令會傳回給定 S3 儲存貯體的政策狀態，其指出儲存貯體是否為公有。

```
Get-S3BucketPolicyStatus -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketPolicyStatus](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetBucketReplication 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetBucketReplication。

CLI

AWS CLI

下列命令會擷取名為 amzn-s3-demo-bucket 之儲存貯體的複寫組態：

```
aws s3api get-bucket-replication --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "ReplicationConfiguration": {
    "Rules": [
```

```

        {
            "Status": "Enabled",
            "Prefix": "",
            "Destination": {
                "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket-backup",
                "StorageClass": "STANDARD"
            },
            "ID": "ZmUwNzE4ZmQ4tMjVhOS00MTlkLOGI4NDkzZTIWJjNTUtYTA1"
        },
        "Role": "arn:aws:iam::123456789012:role/s3-replication-role"
    }
}

```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketReplication](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

/**
 * Retrieves the replication details for the specified S3 bucket.
 *
 * @param s3Client          the S3 client used to interact with the S3
service
 * @param sourceBucketName the name of the S3 bucket to retrieve the
replication details for
 *
 * @throws S3Exception if there is an error retrieving the replication
details
 */
public static void getReplicationDetails(S3Client s3Client, String
sourceBucketName) {
    GetBucketReplicationRequest getRequest =
GetBucketReplicationRequest.builder()
        .bucket(sourceBucketName)

```

```
        .build();

        try {
            ReplicationConfiguration replicationConfig =
s3Client.getBucketReplication(getRequest).replicationConfiguration();
            ReplicationRule rule = replicationConfig.rules().get(0);
            System.out.println("Retrieved destination bucket: " +
rule.destination().bucket());
            System.out.println("Retrieved priority: " + rule.priority());
            System.out.println("Retrieved source-bucket replication rule status:
" + rule.status());

        } catch (S3Exception e) {
            System.err.println("Failed to retrieve replication details: " +
e.awsErrorDetails().errorMessage());
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetBucketReplication](#)。

PowerShell

Tools for PowerShell V4

範例 1：傳回在名為 'amzn-s3-demo-bucket' 的儲存貯體上設定的複寫組態資訊。

```
Get-S3BucketReplication -BucketName amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketReplication](#)。

Tools for PowerShell V5

範例 1：傳回在名為 'amzn-s3-demo-bucket' 的儲存貯體上設定的複寫組態資訊。

```
Get-S3BucketReplication -BucketName amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketReplication](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 `GetBucketRequestPayment` 與 CLI

下列程式碼範例示範如何使用 `GetBucketRequestPayment`。

CLI

AWS CLI

擷取儲存貯體的請求付款組態

下列 `get-bucket-request-payment` 範例會擷取請求者支付指定儲存貯體的組態。

```
aws s3api get-bucket-request-payment \
  --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "Payer": "BucketOwner"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketRequestPayment](#)。

PowerShell

Tools for PowerShell V4

範例 1：傳回名為 'amzn-s3-demo-bucket' 之儲存貯體的請求付款組態。根據預設，儲存貯體擁有者會支付從儲存貯體下載的費用。

```
Get-S3BucketRequestPayment -BucketName amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》的 [GetBucketRequestPayment](#)。

Tools for PowerShell V5

範例 1：傳回名為 'amzn-s3-demo-bucket' 之儲存貯體的請求付款組態。根據預設，儲存貯體擁有者會支付從儲存貯體下載的費用。

```
Get-S3BucketRequestPayment -BucketName amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》的 [GetBucketRequestPayment](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetBucketTagging** 與 CLI

下列程式碼範例示範如何使用 GetBucketTagging。

CLI

AWS CLI

下列命令會擷取名為 amzn-s3-demo-bucket 之儲存貯體的標記組態：

```
aws s3api get-bucket-tagging --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "TagSet": [
    {
      "Value": "marketing",
      "Key": "organization"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketTagging](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回與給定儲存貯體相關聯的所有標籤。

```
Get-S3BucketTagging -BucketName 'amzn-s3-demo-bucket'
```


- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketTagging](#)。

Tools for PowerShell V5

範例 1：此命令會傳回與給定儲存貯體相關聯的所有標籤。

```
Get-S3BucketTagging -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketTagging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetBucketVersioning** 與 CLI

下列程式碼範例示範如何使用 GetBucketVersioning。

CLI

AWS CLI

下列命令會擷取名為 amzn-s3-demo-bucket 之儲存貯體的版本控制組態：

```
aws s3api get-bucket-versioning --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "Status": "Enabled"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketVersioning](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回與給定儲存貯體相關的版本控制狀態。

```
Get-S3BucketVersioning -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketVersioning](#)。

Tools for PowerShell V5

範例 1：此命令會傳回與給定儲存貯體相關的版本控制狀態。

```
Get-S3BucketVersioning -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketVersioning](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetBucketWebsite 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetBucketWebsite。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
// Get the website configuration.
GetBucketWebsiteRequest getRequest = new
GetBucketWebsiteRequest()
{
    BucketName = bucketName,
};
GetBucketWebsiteResponse getResponse = await
client.GetBucketWebsiteAsync(getRequest);
Console.WriteLine($"Index document:
{getResponse.WebsiteConfiguration.IndexDocumentSuffix}");
```

```
Console.WriteLine($"Error document:
{getResponse.WebsiteConfiguration.ErrorDocument}");
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetBucketWebsite](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::getWebsiteConfig(const Aws::String &bucketName,
                                   const Aws::S3::S3ClientConfiguration
&clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::GetBucketWebsiteRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::GetBucketWebsiteOutcome outcome =
        s3Client.GetBucketWebsite(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();

        std::cerr << "Error: GetBucketWebsite: "
                    << err.GetMessage() << std::endl;
    } else {
        Aws::S3::Model::GetBucketWebsiteResult websiteResult =
            outcome.GetResult();

        std::cout << "Success: GetBucketWebsite: "
                    << std::endl << std::endl
                    << "For bucket '" << bucketName << "':"
```

```
        << std::endl
        << "Index page : "
        << websiteResult.GetIndexDocument().GetSuffix()
        << std::endl
        << "Error page: "
        << websiteResult.GetErrorDocument().GetKey()
        << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [GetBucketWebsite](#)。

CLI

AWS CLI

下列命令會擷取名為 `amzn-s3-demo-bucket` 之儲存貯體的靜態網站組態：

```
aws s3api get-bucket-website --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "IndexDocument": {
    "Suffix": "index.html"
  },
  "ErrorDocument": {
    "Key": "error.html"
  }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetBucketWebsite](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得網站組態。

```
import {
  GetBucketWebsiteCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Log the website configuration for a bucket.
 * @param {{ bucketName }}
 */
export const main = async ({ bucketName }) => {
  const client = new S3Client({});

  try {
    const response = await client.send(
      new GetBucketWebsiteCommand({
        Bucket: bucketName,
      }),
    );
    console.log(
      `Your bucket is set up to host a website with the following configuration:
\n${JSON.stringify(response, null, 2)}`,
    );
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchWebsiteConfiguration"
    ) {
      console.error(
        `Error from S3 while getting website configuration for ${bucketName}. The
bucket isn't configured as a website.`
      );
    }
  }
}
```

```
    );  
    } else if (caught instanceof S3ServiceException) {  
        console.error(  
            `Error from S3 while getting website configuration for ${bucketName}.  
${caught.name}: ${caught.message}`,  
        );  
    } else {  
        throw caught;  
    }  
    }  
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [GetBucketWebsite](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回給定 S3 儲存貯體之靜態網站組態的詳細資訊。

```
Get-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetBucketWebsite](#)。

Tools for PowerShell V5

範例 1：此命令會傳回給定 S3 儲存貯體之靜態網站組態的詳細資訊。

```
Get-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetBucketWebsite](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetObject 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetObject。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)
- [從儲存貯體中取得物件 \(如果其已修改的話\)](#)
- [從多區域存取點取得物件](#)
- [開始使用加密](#)
- [物件儲存入門](#)
- [提出條件式請求](#)
- [追蹤上傳和下載](#)

.NET

適用於 .NET 的 SDK (v4)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Shows how to download an object from an Amazon S3 bucket to the
/// local computer.
/// </summary>
/// <param name="bucketName">The name of the bucket where the object is
/// currently stored.</param>
/// <param name="objectName">The name of the object to download.</param>
/// <param name="filePath">The path, including filename, where the
/// downloaded object will be stored.</param>
/// <returns>A boolean value indicating the success or failure of the
/// download process.</returns>
public async Task<bool> DownloadObjectFromBucketAsync(
    string bucketName,
    string objectName,
    string filePath)
{
    var request = new GetObjectRequest
```

```
        {
            BucketName = bucketName,
            Key = objectName,
        };

        using GetObjectResponse response = await
            _amazonS3.GetObjectAsync(request);

        try
        {
            // Save object to local file
            await response.WriteResponseStreamToFileAsync($"{filePath}\
                \{objectName}", true, CancellationToken.None);
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Error saving {objectName}: {ex.Message}");
            return false;
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetObject](#)。
適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用條件式請求取得物件。

```
/// <summary>
/// Retrieves an object from Amazon S3 with a conditional request.
/// </summary>
/// <param name="objectKey">The key of the object to retrieve.</param>
/// <param name="sourceBucket">The source bucket of the object.</param>
/// <param name="conditionType">The type of condition: 'IfMatch',
'IfNoneMatch', 'IfModifiedSince', 'IfUnmodifiedSince'.</param>
```



```
/// <param name="conditionDateValue">The value to use for the condition for
dates.</param>
/// <param name="etagConditionalValue">The value to use for the condition for
etags.</param>
/// <returns>True if the conditional read is successful, False otherwise.</
returns>
public async Task<bool> GetObjectConditional(string objectKey, string
sourceBucket,
    S3ConditionType conditionType, DateTime? conditionDateValue = null,
string? etagConditionalValue = null)
{
    try
    {
        var getObjectRequest = new GetObjectRequest
        {
            BucketName = sourceBucket,
            Key = objectKey
        };

        switch (conditionType)
        {
            case S3ConditionType.IfMatch:
                getObjectRequest.EtagToMatch = etagConditionalValue;
                break;
            case S3ConditionType.IfNoneMatch:
                getObjectRequest.EtagToNotMatch = etagConditionalValue;
                break;
            case S3ConditionType.IfModifiedSince:
                getObjectRequest.ModifiedSinceDateUtc =
conditionDateValue.GetValueOrDefault();
                break;
            case S3ConditionType.IfUnmodifiedSince:
                getObjectRequest.UnmodifiedSinceDateUtc =
conditionDateValue.GetValueOrDefault();
                break;
            default:
                throw new ArgumentOutOfRangeException(nameof(conditionType),
conditionType, null);
        }

        var response = await _amazonS3.GetObjectAsync(getObjectRequest);
        var sampleBytes = new byte[20];
        await response.ResponseStream.ReadAsync(sampleBytes, 0, 20);
    }
}
```

```

        _logger.LogInformation($"Conditional read
        successful. Here are the first 20 bytes of the object:
        \n{System.Text.Encoding.UTF8.GetString(sampleBytes)}");
        return true;
    }
    catch (AmazonS3Exception e)
    {
        if (e.ErrorCode == "PreconditionFailed")
        {
            _logger.LogError("Conditional read failed: Precondition failed");
        }
        else if (e.ErrorCode == "NotModified")
        {
            _logger.LogError("Conditional read failed: Object not modified");
        }
        else
        {
            _logger.LogError($"Unexpected error: {e.ErrorCode}");
            throw;
        }
        return false;
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetObject](#)。

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####

```

```

function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function download_object_from_bucket
#
# This function downloads an object in a bucket to a file.
#
# Parameters:
#     $1 - The name of the bucket to download the object from.
#     $2 - The path and file name to store the downloaded bucket.
#     $3 - The key (name) of the object in the bucket.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function download_object_from_bucket() {
    local bucket_name=$1
    local destination_file_name=$2
    local object_name=$3
    local response

    response=$(aws s3api get-object \
        --bucket "$bucket_name" \
        --key "$object_name" \
        "$destination_file_name")

    # shellcheck disable=SC2181
    if [[ ${?} -ne 0 ]]; then
        errecho "ERROR: AWS reports put-object operation failed.\n$response"
        return 1
    fi
}

```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetObject](#)。

C++

適用於 C++ 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::getObject(const Aws::String &objectKey,
                           const Aws::String &fromBucket,
                           const Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client client(clientConfig);

    Aws::S3::Model::GetObjectRequest request;
    request.SetBucket(fromBucket);
    request.SetKey(objectKey);

    Aws::S3::Model::GetObjectOutcome outcome =
        client.GetObject(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: getObject: " <<
            err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
    } else {
        std::cout << "Successfully retrieved '" << objectKey << "' from '"
            << fromBucket << "'." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [GetObject](#)。

CLI

AWS CLI

以下範例使用 `get-object` 命令，從 Amazon S3 下載物件。

```
aws s3api get-object --bucket text-content --key dir/  
my_images.tar.bz2 my_images.tar.bz2
```

請注意，`outfile` 參數是在沒有選項名稱的情況下指定，例如 `--outfile`。輸出檔案名稱必須是命令中的最後一個參數。

以下範例展示如何使用 `--range` 從物件下載特定位元組範圍。請注意，位元組範圍必須以 `"bytes="` 開頭：

```
aws s3api get-object --bucket text-content --key dir/my_data --  
range bytes=8888-9999 my_data_range
```

如需擷取物件的詳細資訊，請參閱《Amazon S3 開發人員指南》中的「取得物件」。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetObject](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (  
    "bytes"  
    "context"  
    "errors"  
    "fmt"  
    "io"  
    "log"  
    "os"
```

```
"time"

"github.com/aws/aws-sdk-go-v2/aws"
"github.com/aws/aws-sdk-go-v2/feature/s3/manager"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/aws/aws-sdk-go-v2/service/s3/types"
"github.com/aws/smithy-go"
)

// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
// actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
// bucket
// and object actions.
type BucketBasics struct {
    S3Client *s3.Client
}

// DownloadFile gets an object from a bucket and stores it in a local file.
func (basics BucketBasics) DownloadFile(ctx context.Context, bucketName string,
    objectKey string, fileName string) error {
    result, err := basics.S3Client.GetObject(ctx, &s3.GetObjectInput{
        Bucket: aws.String(bucketName),
        Key:    aws.String(objectKey),
    })
    if err != nil {
        var noKey *types.NoSuchKey
        if errors.As(err, &noKey) {
            log.Printf("Can't get object %s from bucket %s. No such key exists.\n",
                objectKey, bucketName)
            err = noKey
        } else {
            log.Printf("Couldn't get object %v:%v. Here's why: %v\n", bucketName,
                objectKey, err)
        }
        return err
    }
    defer result.Body.Close()
    file, err := os.Create(fileName)
    if err != nil {
        log.Printf("Couldn't create file %v. Here's why: %v\n", fileName, err)
    }
}
```

```
    return err
}
defer file.Close()
body, err := io.ReadAll(result.Body)
if err != nil {
    log.Printf("Couldn't read object body from %v. Here's why: %v\n", objectKey,
err)
}
_, err = file.Write(body)
return err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [GetObject](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用 [S3Client](#) 將資料當作位元組陣列讀取。

```
/**
 * Asynchronously retrieves the bytes of an object from an Amazon S3 bucket
and writes them to a local file.
 *
 * @param bucketName the name of the S3 bucket containing the object
 * @param keyName     the key (or name) of the S3 object to retrieve
 * @param path        the local file path where the object's bytes will be
written
 * @return a {@link CompletableFuture} that completes when the object bytes
have been written to the local file
 */
public CompletableFuture<Void> getObjectBytesAsync(String bucketName, String
keyName, String path) {
```

```
GetObjectRequest objectRequest = GetObjectRequest.builder()
    .key(keyName)
    .bucket(bucketName)
    .build();

CompletableFuture<ResponseBytes<GetObjectResponse>> response =
getAsyncClient().getObject(objectRequest, AsyncResponseTransformer.toBytes());
return response.thenAccept(objectBytes -> {
    try {
        byte[] data = objectBytes.asByteArray();
        Path filePath = Paths.get(path);
        Files.write(filePath, data);
        logger.info("Successfully obtained bytes from an S3 object");
    } catch (IOException ex) {
        throw new RuntimeException("Failed to write data to file", ex);
    }
}).whenComplete((resp, ex) -> {
    if (ex != null) {
        throw new RuntimeException("Failed to get object bytes from S3",
ex);
    }
});
}
```

使用 [S3TransferManager](#) 將 S3 儲存貯體中的 [物件下載](#) 到本機檔案。檢視 [完整檔案](#) 並 [測試](#)。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedFileDownload;
import software.amazon.awssdk.transfer.s3.model.DownloadFileRequest;
import software.amazon.awssdk.transfer.s3.model.FileDownload;
import software.amazon.awssdk.transfer.s3.progress.LoggingTransferListener;

import java.io.IOException;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.file.Files;
import java.nio.file.NoSuchFileException;
```



```
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.UUID;

    public Long downloadFile(S3TransferManager transferManager, String
bucketName,

                                String key, String downloadedFilePath) {
        DownloadFileRequest downloadFileRequest = DownloadFileRequest.builder()
            .getObjectRequest(b -> b.bucket(bucketName).key(key))
            .destination(Paths.get(downloadedFilePath))
            .build();

        FileDownload downloadFile =
transferManager.downloadFile(downloadFileRequest);

        CompletedFileDownload downloadResult =
downloadFile.completionFuture().join();
        logger.info("Content length [{})",
downloadResult.response().contentLength());
        return downloadResult.response().contentLength();
    }
```

使用 [S3Client](#) 讀取屬於某個物件的索引標籤。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectTaggingRequest;
import software.amazon.awssdk.services.s3.model.GetObjectTaggingResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.Tag;

import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html

```

```
*/

public class GetObjectTags {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <bucketName> <keyName>\s

            Where:
                bucketName - The Amazon S3 bucket name.\s
                keyName - A key name that represents the object.\s
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        listTags(s3, bucketName, keyName);
        s3.close();
    }

    /**
     * Lists the tags associated with an Amazon S3 object.
     *
     * @param s3 the S3Client object used to interact with the Amazon S3 service
     * @param bucketName the name of the S3 bucket that contains the object
     * @param keyName the key (name) of the S3 object
     */
    public static void listTags(S3Client s3, String bucketName, String keyName) {
        try {
            GetObjectTaggingRequest getTaggingRequest = GetObjectTaggingRequest
                .builder()
                .key(keyName)
                .bucket(bucketName)
                .build();
        }
    }
}
```

```

        GetObjectTaggingResponse tags =
s3.getObjectTagging(getTaggingRequest);
        List<Tag> tagSet = tags.tagSet();
        for (Tag tag : tagSet) {
            System.out.println(tag.key());
            System.out.println(tag.value());
        }

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}

```

使用 [S3Client](#) 取得物件的 URL。

```

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetUrlRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.net.URL;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */

public class GetObjectUrl {
    public static void main(String[] args) {
        final String usage = ""

        Usage:
            <bucketName> <keyName>\s

```

```

        Where:
            bucketName - The Amazon S3 bucket name.
            keyName - A key name that represents the object.\s
        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String bucketName = args[0];
    String keyName = args[1];
    Region region = Region.US_EAST_1;
    S3Client s3 = S3Client.builder()
        .region(region)
        .build();

    getURL(s3, bucketName, keyName);
    s3.close();
}

/**
 * Retrieves the URL for a specific object in an Amazon S3 bucket.
 *
 * @param s3 the S3Client object used to interact with the Amazon S3 service
 * @param bucketName the name of the S3 bucket where the object is stored
 * @param keyName the name of the object for which the URL should be
retrieved
 * @throws S3Exception if there is an error retrieving the URL for the
specified object
 */
public static void getURL(S3Client s3, String bucketName, String keyName) {
    try {
        GetUrlRequest request = GetUrlRequest.builder()
            .bucket(bucketName)
            .key(keyName)
            .build();

        URL url = s3.utilities().getUrl(request);
        System.out.println("The URL for " + keyName + " is " + url);

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
    }
}

```

```
        System.exit(1);
    }
}
}
```

使用 [S3Client](#) 透過使用 S3Presigner 用戶端物件取得物件。

```
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.time.Duration;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import
    software.amazon.awssdk.services.s3.presigner.model.GetObjectPresignRequest;
import
    software.amazon.awssdk.services.s3.presigner.model.PresignedGetObjectRequest;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;
import software.amazon.awssdk.utils.IoUtils;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class GetObjectPresignedUrl {
    public static void main(String[] args) {
        final String USAGE = ""

            Usage:
                <bucketName> <keyName>\s

            Where:
                bucketName - The Amazon S3 bucket name.\s
    }
```

```
        keyName - A key name that represents a text file.\s
        """;

    if (args.length != 2) {
        System.out.println(USAGE);
        System.exit(1);
    }

    String bucketName = args[0];
    String keyName = args[1];
    Region region = Region.US_EAST_1;
    S3Presigner presigner = S3Presigner.builder()
        .region(region)
        .build();

    getPresignedUrl(presigner, bucketName, keyName);
    presigner.close();
}

/**
 * Generates a pre-signed URL for an Amazon S3 object.
 *
 * @param presigner The {@link S3Presigner} instance to use for generating
the pre-signed URL.
 * @param bucketName The name of the Amazon S3 bucket where the object is
stored.
 * @param keyName The key name (file name) of the object in the Amazon S3
bucket.
 *
 * @throws S3Exception If there is an error interacting with the Amazon S3
service.
 * @throws IOException If there is an error opening the HTTP connection or
reading/writing the request/response.
 */
public static void getPresignedUrl(S3Presigner presigner, String bucketName,
String keyName) {
    try {
        GetObjectRequest getObjectRequest = GetObjectRequest.builder()
            .bucket(bucketName)
            .key(keyName)
            .build();

        GetObjectPresignRequest getObjectPresignRequest =
GetObjectPresignRequest.builder()
```

```
        .signatureDuration(Duration.ofMinutes(60))
        .getObjectRequest(getObjectRequest)
        .build();

    PresignedGetObjectRequest presignedGetObjectRequest =
presigner.presignGetObject(getObjectPresignRequest);
    String theUrl = presignedGetObjectRequest.url().toString();
    System.out.println("Presigned URL: " + theUrl);
    HttpURLConnection connection = (HttpURLConnection)
presignedGetObjectRequest.url().openConnection();
    presignedGetObjectRequest.httpRequest().headers().forEach((header,
values) -> {
        values.forEach(value -> {
            connection.addRequestProperty(header, value);
        });
    });

    // Send any request payload that the service needs (not needed when
    // isBrowserExecutable is true).
    if (presignedGetObjectRequest.signedPayload().isPresent()) {
        connection.setDoOutput(true);

        try (InputStream signedPayload =
presignedGetObjectRequest.signedPayload().get().asInputStream();
            OutputStream httpOutputStream =
connection.getOutputStream()) {
            IoUtils.copy(signedPayload, httpOutputStream);
        }
    }

    // Download the result of executing the request.
    try (InputStream content = connection.getInputStream()) {
        System.out.println("Service returned response: ");
        IoUtils.copy(content, System.out);
    }

} catch (S3Exception | IOException e) {
    e.printStackTrace();
}

}
```

使用 ResponseTransformer 物件和 [S3Client](#) 取得物件。

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.core.sync.ResponseTransformer;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */

public class GetObjectData {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <bucketName> <keyName> <path>

            Where:
                bucketName - The Amazon S3 bucket name.\s
                keyName - The key name.\s
                path - The path where the file is written to.\s
            """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }
    }
}
```



```
String bucketName = args[0];
String keyName = args[1];
String path = args[2];
Region region = Region.US_EAST_1;
S3Client s3 = S3Client.builder()
    .region(region)
    .build();

getObjectBytes(s3, bucketName, keyName, path);
s3.close();
}

/**
 * Retrieves the bytes of an object stored in an Amazon S3 bucket and saves
 * them to a local file.
 *
 * @param s3 The S3Client instance used to interact with the Amazon S3
 * service.
 * @param bucketName The name of the S3 bucket where the object is stored.
 * @param keyName The key (or name) of the S3 object.
 * @param path The local file path where the object's bytes will be saved.
 * @throws IOException If an I/O error occurs while writing the bytes to the
 * local file.
 * @throws S3Exception If an error occurs while retrieving the object from
 * the S3 bucket.
 */
public static void getObjectBytes(S3Client s3, String bucketName, String
keyName, String path) {
    try {
        GetObjectRequest objectRequest = GetObjectRequest
            .builder()
            .key(keyName)
            .bucket(bucketName)
            .build();

        ResponseBytes<GetObjectResponse> objectBytes =
s3.getObject(objectRequest, ResponseTransformer.toBytes());
        byte[] data = objectBytes.asByteArray();

        // Write the data to a local file.
        File myFile = new File(path);
        OutputStream os = new FileOutputStream(myFile);
        os.write(data);
        System.out.println("Successfully obtained bytes from an S3 object");
    }
}
```

```
        os.close();

    } catch (IOException ex) {
        ex.printStackTrace();
    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetObject](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

下載物件。

```
import {
    GetObjectCommand,
    NoSuchKey,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Get a single object from a specified S3 bucket.
 * @param {{ bucketName: string, key: string }}
 */
export const main = async ({ bucketName, key }) => {
    const client = new S3Client({});

    try {
        const response = await client.send(
```

```

        new GetObjectCommand({
            Bucket: bucketName,
            Key: key,
        }),
    );
    // The Body object also has 'transformToArray' and 'transformToWebStream'
    methods.
    const str = await response.Body.transformToString();
    console.log(str);
} catch (caught) {
    if (caught instanceof NoSuchKey) {
        console.error(
            `Error from S3 while getting object "${key}" from "${bucketName}". No
            such key exists.`,
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while getting object from ${bucketName}. ${caught.name}:
            ${caught.message}`,
        );
    } else {
        throw caught;
    }
}
};

```

如果物件的 ETag 與提供的 ETag 相符，則下載該物件。

```

import {
    GetObjectCommand,
    NoSuchKey,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Get a single object from a specified S3 bucket.
 * @param {{ bucketName: string, key: string, eTag: string }}
 */
export const main = async ({ bucketName, key, eTag }) => {
    const client = new S3Client({});

```

```
try {
  const response = await client.send(
    new GetObjectCommand({
      Bucket: bucketName,
      Key: key,
      IfMatch: eTag,
    }),
  );
  // The Body object also has 'transformToByteArray' and 'transformToWebStream'
  methods.
  const str = await response.Body.transformToString();
  console.log("Success. Here is text of the file:", str);
} catch (caught) {
  if (caught instanceof NoSuchKey) {
    console.error(
      `Error from S3 while getting object "${key}" from "${bucketName}". No
such key exists.`,
    );
  } else if (caught instanceof S3ServiceException) {
    console.error(
      `Error from S3 while getting object from ${bucketName}. ${caught.name}:
${caught.message}`,
    );
  } else {
    throw caught;
  }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
  isMain,
  validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
  const options = {
    bucketName: {
      type: "string",
      required: true,
    },
    key: {
```

```

        type: "string",
        required: true,
    },
    eTag: {
        type: "string",
        required: true,
    },
};
const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
return { errors, results };
};

if (isMain(import.meta.url)) {
    const { errors, results } = loadArgs();
    if (!errors) {
        main(results.values);
    } else {
        console.error(errors.join("\n"));
    }
}

```

如果物件的 ETag 與提供的 ETag 不相符，則下載該物件。

```

import {
    GetObjectCommand,
    NoSuchKey,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Get a single object from a specified S3 bucket.
 * @param {{ bucketName: string, key: string, eTag: string }}
 */
export const main = async ({ bucketName, key, eTag }) => {
    const client = new S3Client({});

    try {
        const response = await client.send(
            new GetObjectCommand({

```

```
        Bucket: bucketName,
        Key: key,
        IfNoneMatch: eTag,
    )),
    );
    // The Body object also has 'transformToByteArray' and 'transformToWebStream'
    methods.
    const str = await response.Body.transformToString();
    console.log("Success. Here is text of the file:", str);
} catch (caught) {
    if (caught instanceof NoSuchKey) {
        console.error(
            `Error from S3 while getting object "${key}" from "${bucketName}". No
            such key exists.`
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while getting object from ${bucketName}. ${caught.name}:
            ${caught.message}`
        );
    } else {
        throw caught;
    }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
    isMain,
    validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
    const options = {
        bucketName: {
            type: "string",
            required: true,
        },
        key: {
            type: "string",
            required: true,
        },
        eTag: {
```

```

        type: "string",
        required: true,
    },
};
const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
return { errors, results };
};

if (isMain(import.meta.url)) {
    const { errors, results } = loadArgs();
    if (!errors) {
        main(results.values);
    } else {
        console.error(errors.join("\n"));
    }
}

```

在給定的時間範圍內建立或修改物件的情況下，下載該物件。

```

import {
    GetObjectCommand,
    NoSuchKey,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Get a single object from a specified S3 bucket.
 * @param {{ bucketName: string, key: string }}
 */
export const main = async ({ bucketName, key }) => {
    const client = new S3Client({});
    const date = new Date();
    date.setDate(date.getDate() - 1);
    try {
        const response = await client.send(
            new GetObjectCommand({
                Bucket: bucketName,
                Key: key,
                IfModifiedSince: date,
            })
        );
    } catch (error) {
        if (error instanceof S3ServiceException) {
            console.error(error.message);
        } else {
            console.error("An unexpected error occurred:", error);
        }
    }
};

```

```
    }},
  );
  // The Body object also has 'transformToByteArray' and 'transformToWebStream'
  methods.
  const str = await response.Body.transformToString();
  console.log("Success. Here is text of the file:", str);
} catch (caught) {
  if (caught instanceof NoSuchKey) {
    console.error(
      `Error from S3 while getting object "${key}" from "${bucketName}". No
such key exists.`
    );
  } else if (caught instanceof S3ServiceException) {
    console.error(
      `Error from S3 while getting object from ${bucketName}. ${caught.name}:
${caught.message}`
    );
  } else {
    throw caught;
  }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
  isMain,
  validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
  const options = {
    bucketName: {
      type: "string",
      required: true,
    },
    key: {
      type: "string",
      required: true,
    },
  };
};
const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
return { errors, results };
```



```
};

if (isMain(import.meta.url)) {
  const { errors, results } = loadArgs();
  if (!errors) {
    main(results.values);
  } else {
    console.error(errors.join("\n"));
  }
}
```

在給定的時間範圍內未建立或修改物件的情況下，下載該物件。

```
import {
  GetObjectCommand,
  NoSuchKey,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Get a single object from a specified S3 bucket.
 * @param {{ bucketName: string, key: string }}
 */
export const main = async ({ bucketName, key }) => {
  const client = new S3Client({});
  const date = new Date();
  date.setDate(date.getDate() - 1);
  try {
    const response = await client.send(
      new GetObjectCommand({
        Bucket: bucketName,
        Key: key,
        IfUnmodifiedSince: date,
      }),
    );
    // The Body object also has 'transformToByteArray' and 'transformToWebStream'
    methods.
    const str = await response.Body.transformToString();
    console.log("Success. Here is text of the file:", str);
  } catch (caught) {
```

```
    if (caught instanceof NoSuchKey) {
      console.error(
        `Error from S3 while getting object "${key}" from "${bucketName}". No
such key exists.`
      );
    } else if (caught instanceof S3ServiceException) {
      console.error(
        `Error from S3 while getting object from ${bucketName}. ${caught.name}:
${caught.message}`
      );
    } else {
      throw caught;
    }
  }
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
  isMain,
  validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
  const options = {
    bucketName: {
      type: "string",
      required: true,
    },
    key: {
      type: "string",
      required: true,
    },
  };
  const results = parseArgs({ options });
  const { errors } = validateArgs({ options }, results);
  return { errors, results };
};

if (isMain(import.meta.url)) {
  const { errors, results } = loadArgs();
  if (!errors) {
    main(results.values);
  } else {
```

```
        console.error(errors.join("\n"));
    }
}
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [GetObject](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun getObjectBytes(
    bucketName: String,
    keyName: String,
    path: String,
) {
    val request =
        GetObjectRequest {
            key = keyName
            bucket = bucketName
        }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        s3.getObject(request) { resp ->
            val myFile = File(path)
            resp.body?.writeToFile(myFile)
            println("Successfully read $keyName from $bucketName")
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [GetObject](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得物件。

```
$s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

try {
    $file = $this->s3client->getObject([
        'Bucket' => $this->bucketName,
        'Key' => $fileName,
    ]);
    $body = $file->get('Body');
    $body->rewind();
    echo "Downloaded the file and it begins with: {$body->read(26)}.\n";
} catch (Exception $exception) {
    echo "Failed to download $fileName from $this->bucketName with error:
" . $exception->getMessage();
    exit("Please fix error with file downloading before continuing.");
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [GetObject](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會從儲存貯體 "amzn-s3-demo-bucket" 擷取項目 "sample.txt"，並將其儲存至目前位置中名為 "local-sample.txt" 的檔案。在呼叫此命令之前，檔案 "local-sample.txt" 不需存在。

```
Read-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -File local-
sample.txt
```

範例 2：此命令會從儲存貯體 "amzn-s3-demo-bucket" 擷取虛擬目錄 "DIR"，並將其儲存至目前位置中名為 "Local-DIR" 的資料夾。呼叫此命令之前，資料夾 "Local-DIR" 不需存在。

```
Read-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix DIR -Folder Local-DIR
```

範例 3：將所有金鑰以 '.json' 結尾的物件，從儲存貯體名稱具有 'config' 的儲存貯體，下載到指定資料夾中的檔案。物件金鑰用於設定檔案名稱。

```
Get-S3Bucket | ? { $_.BucketName -like '*config*' } | Get-S3Object | ? { $_.Key -like '*.json' } | Read-S3Object -Folder C:\ConfigObjects
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetObject](#)。

Tools for PowerShell V5

範例 1：此命令會從儲存貯體 "amzn-s3-demo-bucket" 擷取項目 "sample.txt"，並將其儲存至目前位置中名為 "local-sample.txt" 的檔案。在呼叫此命令之前，檔案 "local-sample.txt" 不需存在。

```
Read-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -File local-sample.txt
```

範例 2：此命令會從儲存貯體 "amzn-s3-demo-bucket" 擷取虛擬目錄 "DIR"，並將其儲存至目前位置中名為 "Local-DIR" 的資料夾。呼叫此命令之前，資料夾 "Local-DIR" 不需存在。

```
Read-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix DIR -Folder Local-DIR
```

範例 3：將所有金鑰以 '.json' 結尾的物件，從儲存貯體名稱具有 'config' 的儲存貯體，下載到指定資料夾中的檔案。物件金鑰用於設定檔案名稱。

```
Get-S3Bucket | ? { $_.BucketName -like '*config*' } | Get-S3Object | ? { $_.Key -like '*.json' } | Read-S3Object -Folder C:\ConfigObjects
```

範例 4：使用分段平行下載來下載單一物件，以提升大型檔案的效能。

```
Read-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt -File local-sample.txt -UseMultipartDownload
```

範例 5：使用具有自訂 64MB 組件大小的 RANGE 模式下載大型物件。RANGE 模式使用 HTTP 位元組範圍請求，並適用於所有物件，無論它們的上傳方式為何。

```
Read-S3Object -BucketName amzn-s3-demo-bucket -Key large-file.zip -File local-large-file.zip -UseMultipartDownload -MultipartDownloadType RANGE -PartSize 64MB
```

範例 6：使用分段下載搭配並行檔案下載，將金鑰字首為 "data/" 的所有物件下載至本機資料夾。

```
Read-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix data/ -Folder C:\local-data -UseMultipartDownload -DownloadFilesConcurrently
```

範例 7：使用具有 20 個並行連線的分段下載來下載單一物件，以在高頻寬網路上達到最大輸送量。

```
Read-S3Object -BucketName amzn-s3-demo-bucket -Key large-file.zip -File local-large-file.zip -UseMultipartDownload -ConcurrentServiceRequest 20
```

範例 8：使用分段下載將所有金鑰字首為 "data/" 的物件下載至本機資料夾，繼續超過個別檔案失敗，而不是中止。

```
Read-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix data/ -Folder C:\local-data -UseMultipartDownload -FailurePolicy ContinueOnFailure
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetObject](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
        in Boto3
                           that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key

    def get(self):
        """
        Gets the object.

        :return: The object data in bytes.
        """
        try:
            body = self.object.get()["Body"].read()
            logger.info(
                "Got object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't get object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
            raise
        else:
            return body
```

使用條件式請求取得物件。

```
class S3ConditionalRequests:
    """Encapsulates S3 conditional request operations."""
```

```
def __init__(self, s3_client):
    self.s3 = s3_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        s3_client = boto3.client("s3")
        return cls(s3_client)

    def get_object_conditional(
        self,
        object_key: str,
        source_bucket: str,
        condition_type: str,
        condition_value: str,
    ):
        """
        Retrieves an object from Amazon S3 with a conditional request.

        :param object_key: The key of the object to retrieve.
        :param source_bucket: The source bucket of the object.
        :param condition_type: The type of condition: 'IfMatch', 'IfNoneMatch',
        'IfModifiedSince', 'IfUnmodifiedSince'.
        :param condition_value: The value to use for the condition.
        """
        try:
            response = self.s3.get_object(
                Bucket=source_bucket,
                Key=object_key,
                **{condition_type: condition_value},
            )
            sample_bytes = response["Body"].read(20)
            print(
                f"\tConditional read successful. Here are the first 20 bytes of"
                f"the object:\n"
            )
            print(f"\t{sample_bytes}")
        except ClientError as e:
            error_code = e.response["Error"]["Code"]
            if error_code == "PreconditionFailed":
```



```
        print("\tConditional read failed: Precondition failed")
    elif error_code == "304": # Not modified error code.
        print("\tConditional read failed: Object not modified")
    else:
        logger.error(f"Unexpected error: {error_code}")
        raise
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [GetObject](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得物件。

```
require 'aws-sdk-s3'

# Wraps Amazon S3 object actions.
class ObjectGetWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Gets the object directly to a file.
  #
  # @param target_path [String] The path to the file where the object is
  # downloaded.
  # @return [Aws::S3::Types::GetObjectOutput, nil] The retrieved object data if
  # successful; otherwise nil.
  def get_object(target_path)
```

```

    @object.get(response_target: target_path)
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get object #{@object.key}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  bucket_name = "amzn-s3-demo-bucket"
  object_key = "my-object.txt"
  target_path = "my-object-as-file.txt"

  wrapper = ObjectGetWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  obj_data = wrapper.get_object(target_path)
  return unless obj_data

  puts "Object #{object_key} (#{obj_data.content_length} bytes) downloaded to
  #{target_path}."
end

run_demo if $PROGRAM_NAME == __FILE__

```

取得物件並報告其伺服器端的加密狀態。

```

require 'aws-sdk-s3'

# Wraps Amazon S3 object actions.
class ObjectGetEncryptionWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Gets the object into memory.
  #
  # @return [Aws::S3::Types::GetObjectOutput, nil] The retrieved object data if
  successful; otherwise nil.
  def object
    @object.get
  rescue Aws::Errors::ServiceError => e

```

```

    puts "Couldn't get object #{@object.key}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  bucket_name = "amzn-s3-demo-bucket"
  object_key = "my-object.txt"

  wrapper = ObjectGetEncryptionWrapper.new(Aws::S3::Object.new(bucket_name,
    object_key))
  obj_data = wrapper.get_object
  return unless obj_data

  encryption = obj_data.server_side_encryption.nil? ? 'no' :
    obj_data.server_side_encryption
  puts "Object #{object_key} uses #{encryption} encryption."
end

run_demo if $PROGRAM_NAME == __FILE__

```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [GetObject](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

async fn get_object(client: Client, opt: Opt) -> Result<usize, S3ExampleError> {
  trace!("bucket:      {}", opt.bucket);
  trace!("object:      {}", opt.object);
  trace!("destination: {}", opt.destination.display());

  let mut file = File::create(opt.destination.clone()).map_err(|err| {
    S3ExampleError::new(format!(
      "Failed to initialize file for saving S3 download: {err:?}")

```

```
    ))
  })?;

  let mut object = client
    .get_object()
    .bucket(opt.bucket)
    .key(opt.object)
    .send()
    .await?;

  let mut byte_count = 0_usize;
  while let Some(bytes) = object.body.try_next().await.map_err(|err| {
    S3ExampleError::new(format!("Failed to read from S3 download stream:
{err:?}"))
  })? {
    let bytes_len = bytes.len();
    file.write_all(&bytes).map_err(|err| {
      S3ExampleError::new(format!(
        "Failed to write from S3 download stream to local file: {err:?}"
      ))
    })?;
    trace!("Intermediate write of {bytes_len}");
    byte_count += bytes_len;
  }

  Ok(byte_count)
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [GetObject](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

TRY.

```

        oo_result = lo_s3->getobject(
testing purposes. "
            iv_bucket = iv_bucket_name
            iv_key = iv_object_key ).
DATA(lv_object_data) = oo_result->get_body( ).
MESSAGE 'Object retrieved from S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
MESSAGE 'Bucket does not exist.' TYPE 'E'.
CATCH /aws1/cx_s3_nosuchkey.
MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 SAP ABAP 的 AWS SDK API 參考》中的 [GetObject](#)。

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

import AWSS3

public func downloadFile(bucket: String, key: String, to: String) async
throws {
    let fileUrl = URL(fileURLWithPath: to).appendingPathComponent(key)

    let input = GetObjectInput(
        bucket: bucket,
        key: key
    )
    do {
        let output = try await client.getObject(input: input)

        guard let body = output.body else {
            throw HandlerError.getObjectBody("GetObjectInput missing body.")
        }
    }
}

```

```
        guard let data = try await body.readData() else {
            throw HandlerError.readGetObjectBody("GetObjectInput unable to
read data.")
        }

        try data.write(to: fileUrl)
    }
    catch {
        print("ERROR: ", dump(error, name: "Downloading a file. "))
        throw error
    }
}
```

```
import AWSS3

public func readFile(bucket: String, key: String) async throws -> Data {
    let input = GetObjectInput(
        bucket: bucket,
        key: key
    )
    do {
        let output = try await client.getObject(input: input)

        guard let body = output.body else {
            throw HandlerError.getObjectBody("GetObjectInput missing body.")
        }

        guard let data = try await body.readData() else {
            throw HandlerError.readGetObjectBody("GetObjectInput unable to
read data.")
        }

        return data
    }
    catch {
        print("ERROR: ", dump(error, name: "Reading a file. "))
        throw error
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Swift 的 AWS SDK API 參考》中的 [GetObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetObjectAcl 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetObjectAcl。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [管理存取控制清單 \(ACL\)](#)

C++

SDK for C++

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::getObjectAcl(const Aws::String &bucketName,
                              const Aws::String &objectKey,
                              const Aws::S3::S3ClientConfiguration &clientConfig)
{
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::GetObjectAclRequest request;
    request.SetBucket(bucketName);
    request.SetKey(objectKey);

    Aws::S3::Model::GetObjectAclOutcome outcome =
        s3Client.GetObjectAcl(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: getObjectAcl: "
```

```

        << err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
    } else {
        Aws::Vector<Aws::S3::Model::Grant> grants =
            outcome.GetResult().GetGrants();

        for (auto it = grants.begin(); it != grants.end(); it++) {
            std::cout << "For object " << objectKey << ": "
                << std::endl << std::endl;

            Aws::S3::Model::Grant grant = *it;
            Aws::S3::Model::Grantee grantee = grant.GetGrantee();

            if (grantee.TypeHasBeenSet()) {
                std::cout << "Type:          "
                    << getGranteeTypeString(grantee.GetType()) <<
std::endl;
            }

            if (grantee.DisplayNameHasBeenSet()) {
                std::cout << "Display name:  "
                    << grantee.GetDisplayName() << std::endl;
            }

            if (grantee.EmailAddressHasBeenSet()) {
                std::cout << "Email address: "
                    << grantee.GetEmailAddress() << std::endl;
            }

            if (grantee.IDHasBeenSet()) {
                std::cout << "ID:           "
                    << grantee.GetID() << std::endl;
            }

            if (grantee.URIHasBeenSet()) {
                std::cout << "URI:          "
                    << grantee.GetURI() << std::endl;
            }

            std::cout << "Permission:    " <<
                getPermissionString(grant.GetPermission()) <<
                std::endl << std::endl;
        }
    }
}

```



```

        return outcome.IsSuccess();
    }

    //! Routine which converts a built-in type enumeration to a human-readable
    string.
    /*!
    \param type: Type enumeration.
    \return String: Human-readable string
    */
    Aws::String getGranteeTypeString(const Aws::S3::Model::Type &type) {
        switch (type) {
            case Aws::S3::Model::Type::AmazonCustomerByEmail:
                return "Email address of an AWS account";
            case Aws::S3::Model::Type::CanonicalUser:
                return "Canonical user ID of an AWS account";
            case Aws::S3::Model::Type::Group:
                return "Predefined Amazon S3 group";
            case Aws::S3::Model::Type::NOT_SET:
                return "Not set";
            default:
                return "Type unknown";
        }
    }

    //! Routine which converts a built-in type enumeration to a human-readable
    string.
    /*!
    \param permission: Permission enumeration.
    \return String: Human-readable string
    */
    Aws::String getPermissionString(const Aws::S3::Model::Permission &permission) {
        switch (permission) {
            case Aws::S3::Model::Permission::FULL_CONTROL:
                return "Can read this object's data and its metadata, "
                    "and read/write this object's permissions";
            case Aws::S3::Model::Permission::NOT_SET:
                return "Permission not set";
            case Aws::S3::Model::Permission::READ:
                return "Can read this object's data and its metadata";
            case Aws::S3::Model::Permission::READ_ACP:
                return "Can read this object's permissions";
            // case Aws::S3::Model::Permission::WRITE // Not applicable.
            case Aws::S3::Model::Permission::WRITE_ACP:

```

```
        return "Can write this object's permissions";
    default:
        return "Permission unknown";
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [GetObjectAcl](#)。

CLI

AWS CLI

下列命令會擷取名為 `amzn-s3-demo-bucket` 之儲存貯體中物件的存取控制清單：

```
aws s3api get-object-acl --bucket amzn-s3-demo-bucket --key index.html
```

輸出：

```
{
  "Owner": {
    "DisplayName": "my-username",
    "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
  },
  "Grants": [
    {
      "Grantee": {
        "DisplayName": "my-username",
        "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
      },
      "Permission": "FULL_CONTROL"
    },
    {
      "Grantee": {
        "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
      },
      "Permission": "READ"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetObjectAcl](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun getBucketACL(
    objectKey: String,
    bucketName: String,
) {
    val request =
        GetObjectAclRequest {
            bucket = bucketName
            key = objectKey
        }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        val response = s3.getObjectAcl(request)
        response.grants?.forEach { grant ->
            println("Grant permission is ${grant.permission}")
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [GetObjectAcl](#)。

PowerShell

Tools for PowerShell V5

範例 1：命令會取得 S3 物件之物件擁有者的詳細資訊。

```
(Get-S3ObjectACL -BucketName 'amzn-s3-demo-bucket' -key 'initialize.ps1' -Select
*).Owner
```

輸出：

```
DisplayName Id
----- --
testusername
9988776a6554433d22f1100112e334acb45566778899009e9887bd7f66c5f544
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetObjectAcl](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
        in Boto3
                           that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key

    def get_acl(self):
        """
        Gets the ACL of the object.
```

```

        :return: The ACL of the object.
        """
        try:
            acl = self.object.Acl()
            logger.info(
                "Got ACL for object %s owned by %s.",
                self.object.key,
                acl.owner["DisplayName"],
            )
        except ClientError:
            logger.exception("Couldn't get ACL for object %s.", self.object.key)
            raise
        else:
            return acl

```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [GetObjectAcl](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    oo_result = lo_s3->getobjectacl(           " oo_result is returned for
testing purposes. "
    iv_bucket = iv_bucket_name
    iv_key = iv_object_key ).
    MESSAGE 'Retrieved object ACL.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.

```

ENDTRY.

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetObjectAcl](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetObjectAttributes 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetObjectAttributes。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [使用 Amazon S3 物件完整性](#)

C++

SDK for C++

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
// ! Routine which retrieves the hash value of an object stored in an S3 bucket.
/*!
    \param bucket: The name of the S3 bucket where the object is stored.
    \param key: The unique identifier (key) of the object within the S3 bucket.
    \param hashMethod: The hashing algorithm used to calculate the hash value of
the object.
    \param[out] hashData: The retrieved hash.
    \param[out] partHashes: The part hashes if available.
    \param client: The S3 client instance used to retrieve the object.
    \return bool: Function succeeded.
*/
bool AwsDoc::S3::retrieveObjectHash(const Aws::String &bucket, const Aws::String
&key,
```

```

        AwsDoc::S3::HASH_METHOD hashMethod,
        Aws::String &hashData,
        std::vector<Aws::String> *partHashes,
        const Aws::S3::S3Client &client) {
    Aws::S3::Model::GetObjectAttributesRequest request;
    request.SetBucket(bucket);
    request.SetKey(key);

    if (hashMethod == MD5) {
        Aws::Vector<Aws::S3::Model::ObjectAttributes> attributes;
        attributes.push_back(Aws::S3::Model::ObjectAttributes::ETag);
        request.SetObjectAttributes(attributes);

        Aws::S3::Model::GetObjectAttributesOutcome outcome =
        client.GetObjectAttributes(
            request);
        if (outcome.IsSuccess()) {
            const Aws::S3::Model::GetObjectAttributesResult &result =
            outcome.GetResult();
            hashData = result.GetETag();
        } else {
            std::cerr << "Error retrieving object etag attributes." <<
                outcome.GetError().GetMessage() << std::endl;
            return false;
        }
    } else { // hashMethod != MD5
        Aws::Vector<Aws::S3::Model::ObjectAttributes> attributes;
        attributes.push_back(Aws::S3::Model::ObjectAttributes::Checksum);
        request.SetObjectAttributes(attributes);

        Aws::S3::Model::GetObjectAttributesOutcome outcome =
        client.GetObjectAttributes(
            request);
        if (outcome.IsSuccess()) {
            const Aws::S3::Model::GetObjectAttributesResult &result =
            outcome.GetResult();
            switch (hashMethod) {
                case AwsDoc::S3::DEFAULT: // NOLINT(*-branch-clone)
                    break; // Default is not supported.
#pragma clang diagnostic push
#pragma ide diagnostic ignored "UnreachableCode"
                case AwsDoc::S3::MD5:
                    break; // MD5 is not supported.
#pragma clang diagnostic pop

```

```

        case AwsDoc::S3::SHA1:
            hashData = result.GetChecksum().GetChecksumSHA1();
            break;
        case AwsDoc::S3::SHA256:
            hashData = result.GetChecksum().GetChecksumSHA256();
            break;
        case AwsDoc::S3::CRC32:
            hashData = result.GetChecksum().GetChecksumCRC32();
            break;
        case AwsDoc::S3::CRC32C:
            hashData = result.GetChecksum().GetChecksumCRC32C();
            break;
        default:
            std::cerr << "Unknown hash method." << std::endl;
            return false;
    }
} else {
    std::cerr << "Error retrieving object checksum attributes." <<
        outcome.GetError().GetMessage() << std::endl;
    return false;
}

if (nullptr != partHashes) {
    attributes.clear();
    attributes.push_back(Aws::S3::Model::ObjectAttributes::ObjectParts);
    request.SetObjectAttributes(attributes);
    outcome = client.GetObjectAttributes(request);
    if (outcome.IsSuccess()) {
        const Aws::S3::Model::GetObjectAttributesResult &result =
outcome.GetResult();
        const Aws::Vector<Aws::S3::Model::ObjectPart> parts =
result.GetObjectParts().GetParts();
        for (const Aws::S3::Model::ObjectPart &part: parts) {
            switch (hashMethod) {
                case AwsDoc::S3::DEFAULT: // Default is not supported.
NOLINT(*-branch-clone)
                    break;
                case AwsDoc::S3::MD5: // MD5 is not supported.
                    break;
                case AwsDoc::S3::SHA1:
                    partHashes->push_back(part.GetChecksumSHA1());
                    break;
                case AwsDoc::S3::SHA256:
                    partHashes->push_back(part.GetChecksumSHA256());

```



```

        break;
    case AwsDoc::S3::CRC32:
        partHashes->push_back(part.GetChecksumCRC32());
        break;
    case AwsDoc::S3::CRC32C:
        partHashes->push_back(part.GetChecksumCRC32C());
        break;
    default:
        std::cerr << "Unknown hash method." << std::endl;
        return false;
    }
}
} else {
    std::cerr << "Error retrieving object attributes for object
parts." <<
        outcome.GetError().GetMessage() << std::endl;
    return false;
}
}
}

return true;
}

```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [GetObjectAttributes](#)。

CLI

AWS CLI

從物件擷取中繼資料而不傳回物件本身

下列 get-object-attributes 範例會從物件 doc1.rtf 擷取中繼資料。

```
aws s3api get-object-attributes \
    --bucket amzn-s3-demo-bucket \
    --key doc1.rtf \
    --object-attributes "StorageClass" "ETag" "ObjectSize"
```

輸出：

```
{
  "LastModified": "2022-03-15T19:37:31+00:00",
  "VersionId": "IuCPjXTDzHNfIdAuitVBIKJpF2p1fg4P",
  "ETag": "b662d79adeb7c8d787ea7eafb9ef6207",
  "StorageClass": "STANDARD",
  "ObjectSize": 405
}
```

如需詳細資訊，請參閱《Amazon S3 API 參考》中的 [GetObjectAttributes](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetObjectAttributes](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetObjectLegalHold 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetObjectLegalHold。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [鎖定 Amazon S3 物件](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Get the legal hold details for an S3 object.
/// </summary>
/// <param name="bucketName">The bucket of the object.</param>
/// <param name="objectKey">The object key.</param>
/// <returns>The object legal hold details.</returns>
public async Task<ObjectLockLegalHold> GetObjectLegalHold(string bucketName,
```

```
string objectKey)
{
    try
    {
        var request = new GetObjectLegalHoldRequest()
        {
            BucketName = bucketName,
            Key = objectKey
        };

        var response = await _amazonS3.GetObjectLegalHoldAsync(request);
        Console.WriteLine($"{bucketName}: " +
            $"{response.LegalHold.Status}");
        return response.LegalHold;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Unable to fetch legal hold: '{ex.Message}'");
        return new ObjectLockLegalHold();
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetObjectLegalHold](#)。

CLI

AWS CLI

擷取物件的法務保存狀態

下列 `get-object-legal-hold` 範例會擷取指定物件的法務保存狀態。

```
aws s3api get-object-legal-hold \
  --bucket amzn-s3-demo-bucket-with-object-lock \
  --key doc1.rtf
```

輸出：

```
{
```

```
    "LegalHold": {  
        "Status": "ON"  
    }  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetObjectLegalHold](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (  
    "bytes"  
    "context"  
    "errors"  
    "fmt"  
    "log"  
    "time"  
  
    "github.com/aws/aws-sdk-go-v2/aws"  
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"  
    "github.com/aws/aws-sdk-go-v2/service/s3"  
    "github.com/aws/aws-sdk-go-v2/service/s3/types"  
    "github.com/aws/smithy-go"  
)  
  
// S3Actions wraps S3 service actions.  
type S3Actions struct {  
    S3Client    *s3.Client  
    S3Manager   *manager.Uploader  
}  
  
// GetObjectLegalHold retrieves the legal hold status for an S3 object.
```

```
func (actor S3Actions) GetObjectLegalHold(ctx context.Context, bucket string, key
string, versionId string) (*types.ObjectLockLegalHoldStatus, error) {
    var status *types.ObjectLockLegalHoldStatus
    input := &s3.GetObjectLegalHoldInput{
        Bucket:    aws.String(bucket),
        Key:       aws.String(key),
        VersionId: aws.String(versionId),
    }

    output, err := actor.S3Client.GetObjectLegalHold(ctx, input)
    if err != nil {
        var noSuchKeyErr *types.NoSuchKey
        var apiErr *smithy.GenericAPIError
        if errors.As(err, &noSuchKeyErr) {
            log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
            err = noSuchKeyErr
        } else if errors.As(err, &apiErr) {
            switch apiErr.ErrorCode() {
            case "NoSuchObjectLockConfiguration":
                log.Printf("Object %s does not have an object lock configuration.\n", key)
                err = nil
            case "InvalidRequest":
                log.Printf("Bucket %s does not have an object lock configuration.\n", bucket)
                err = nil
            }
        }
    } else {
        status = &output.LegalHold.Status
    }

    return status, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [GetObjectLegalHold](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
// Get the legal hold details for an S3 object.
public ObjectLockLegalHold getObjectLegalHold(String bucketName, String
objectKey) {
    try {
        GetObjectLegalHoldRequest legalHoldRequest =
GetObjectLegalHoldRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .build();

        GetObjectLegalHoldResponse response =
getClient().getObjectLegalHold(legalHoldRequest);
        System.out.println("Object legal hold for " + objectKey + " in " +
bucketName +
            ":\n\tStatus: " + response.legalHold().status());
        return response.legalHold();

    } catch (S3Exception ex) {
        System.out.println("\tUnable to fetch legal hold: '" +
ex.getMessage() + "'");
    }

    return null;
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetObjectLegalHold](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import {
  GetObjectLegalHoldCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Get an object's current legal hold status.
 * @param {{ bucketName: string, key: string }}
 */
export const main = async ({ bucketName, key }) => {
  const client = new S3Client({});

  try {
    const response = await client.send(
      new GetObjectLegalHoldCommand({
        Bucket: bucketName,
        Key: key,
        // Optionally, you can provide additional parameters
        // ExpectedBucketOwner: "<account ID that is expected to own the
        bucket>",
        // VersionId: "<the specific version id of the object to check>",
      }),
    );
    console.log(`Legal Hold Status: ${response.LegalHold.Status}`);
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
      console.error(
```

```
        `Error from S3 while getting legal hold status for ${key} in
        ${bucketName}. The bucket doesn't exist.`
    );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while getting legal hold status for ${key} in
            ${bucketName} from ${bucketName}. ${caught.name}: ${caught.message}`,
        );
    } else {
        throw caught;
    }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
    isMain,
    validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
    const options = {
        bucketName: {
            type: "string",
            required: true,
        },
        key: {
            type: "string",
            required: true,
        },
    };
};

const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
return { errors, results };
};

if (isMain(import.meta.url)) {
    const { errors, results } = loadArgs();
    if (!errors) {
        main(results.values);
    } else {
        console.error(errors.join("\n"));
    }
}
```



```
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [GetObjectLegalHold](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

放置物件法務保存。

```
def get_legal_hold(s3_client, bucket: str, key: str) -> None:
    """
    Get the legal hold status of a specific file in a bucket.

    Args:
        s3_client: Boto3 S3 client.
        bucket: The name of the bucket containing the file.
        key: The key of the file to get the legal hold status of.
    """
    print()
    logger.info("Getting legal hold status of file [%s] in bucket [%s]", key,
                bucket)
    try:
        response = s3_client.get_object_legal_hold(Bucket=bucket, Key=key)
        legal_hold_status = response["LegalHold"]["Status"]
        logger.debug(
            "Legal hold status of file [%s] in bucket [%s] is [%s]",
            key,
            bucket,
            legal_hold_status,
        )
    except Exception as e:
        logger.error(
```

```
        "Failed to get legal hold status of file [%s] in bucket [%s]: %s",  
        key,  
        bucket,  
        e,  
    )
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [GetObjectLegalHold](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3->getobjectlegalhold(          " oo_result is returned  
for testing purposes. "  
    iv_bucket = iv_bucket_name  
    iv_key = iv_object_key ).  
    MESSAGE 'Retrieved object legal hold status.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
CATCH /aws1/cx_s3_nosuchkey.  
    MESSAGE 'Object key does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetObjectLegalHold](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetObjectLockConfiguration 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetObjectLockConfiguration。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [鎖定 Amazon S3 物件](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Get the object lock configuration details for an S3 bucket.
/// </summary>
/// <param name="bucketName">The bucket to get details.</param>
/// <returns>The bucket's object lock configuration details.</returns>
public async Task<ObjectLockConfiguration>
GetBucketObjectLockConfiguration(string bucketName)
{
    try
    {
        var request = new GetObjectLockConfigurationRequest()
        {
            BucketName = bucketName
        };

        var response = await
_amazonS3.GetObjectLockConfigurationAsync(request);
        Console.WriteLine($"Bucket object lock config for {bucketName} in
{bucketName}: " +
            $"Enabled:
{response.ObjectLockConfiguration.ObjectLockEnabled}" +
            $"Rule:
{response.ObjectLockConfiguration.Rule?.DefaultRetention}");
    }
}
```

```
        return response.ObjectLockConfiguration;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"\\tUnable to fetch object lock config:
'{ex.Message}'");
        return new ObjectLockConfiguration();
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetObjectLockConfiguration](#)。

CLI

AWS CLI

擷取儲存貯體的物件鎖定組態

下列 `get-object-lock-configuration` 範例會擷取指定儲存貯體的物件鎖定組態。

```
aws s3api get-object-lock-configuration \
    --bucket amzn-s3-demo-bucket-with-object-lock
```

輸出：

```
{
  "ObjectLockConfiguration": {
    "ObjectLockEnabled": "Enabled",
    "Rule": {
      "DefaultRetention": {
        "Mode": "COMPLIANCE",
        "Days": 50
      }
    }
  }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetObjectLockConfiguration](#)。

Go

SDK for Go V2

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (  
    "bytes"  
    "context"  
    "errors"  
    "fmt"  
    "log"  
    "time"  
  
    "github.com/aws/aws-sdk-go-v2/aws"  
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"  
    "github.com/aws/aws-sdk-go-v2/service/s3"  
    "github.com/aws/aws-sdk-go-v2/service/s3/types"  
    "github.com/aws/smithy-go"  
)  
  
// S3Actions wraps S3 service actions.  
type S3Actions struct {  
    S3Client    *s3.Client  
    S3Manager   *manager.Uploader  
}  
  
// GetObjectLockConfiguration retrieves the object lock configuration for an S3  
// bucket.  
func (actor S3Actions) GetObjectLockConfiguration(ctx context.Context, bucket  
    string) (*types.ObjectLockConfiguration, error) {  
    var lockConfig *types.ObjectLockConfiguration  
    input := &s3.GetObjectLockConfigurationInput{  
        Bucket: aws.String(bucket),  
    }  
}
```

```
output, err := actor.S3Client.GetObjectLockConfiguration(ctx, input)
if err != nil {
    var noBucket *types.NoSuchBucket
    var apiErr *smithy.GenericAPIError
    if errors.As(err, &noBucket) {
        log.Printf("Bucket %s does not exist.\n", bucket)
        err = noBucket
    } else if errors.As(err, &apiErr) && apiErr.ErrorCode() ==
"ObjectLockConfigurationNotFoundError" {
        log.Printf("Bucket %s does not have an object lock configuration.\n", bucket)
        err = nil
    }
} else {
    lockConfig = output.ObjectLockConfiguration
}

return lockConfig, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [GetObjectLockConfiguration](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
// Get the object lock configuration details for an S3 bucket.
public void getBucketObjectLockConfiguration(String bucketName) {
    GetObjectLockConfigurationRequest objectLockConfigurationRequest =
GetObjectLockConfigurationRequest.builder()
        .bucket(bucketName)
        .build();
```

```
GetObjectLockConfigurationResponse response =
getClient().getObjectLockConfiguration(objectLockConfigurationRequest);
System.out.println("Bucket object lock config for "+bucketName+": ");
System.out.println("\tEnabled:
"+response.getObjectLockConfiguration().objectLockEnabled());
System.out.println("\tRule: "+
response.getObjectLockConfiguration().rule().defaultRetention());
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetObjectLockConfiguration](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import {
  GetObjectLockConfigurationCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Gets the Object Lock configuration for a bucket.
 * @param {{ bucketName: string }}
 */
export const main = async ({ bucketName }) => {
  const client = new S3Client({});

  try {
    const { ObjectLockConfiguration } = await client.send(
      new GetObjectLockConfigurationCommand({
        Bucket: bucketName,
```

```
// Optionally, you can provide additional parameters
// ExpectedBucketOwner: "<account ID that is expected to own the
bucket>",
  }),
);
console.log(
  `Object Lock Configuration:\n${JSON.stringify(ObjectLockConfiguration)}`,
);
} catch (caught) {
  if (
    caught instanceof S3ServiceException &&
    caught.name === "NoSuchBucket"
  ) {
    console.error(
      `Error from S3 while getting object lock configuration for ${bucketName}.
The bucket doesn't exist.`,
    );
  } else if (caught instanceof S3ServiceException) {
    console.error(
      `Error from S3 while getting object lock configuration for ${bucketName}.
${caught.name}: ${caught.message}`,
    );
  } else {
    throw caught;
  }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
  isMain,
  validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
  const options = {
    bucketName: {
      type: "string",
      required: true,
    },
  };
};
const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
```



```
    return { errors, results };
  };

  if (isMain(import.meta.url)) {
    const { errors, results } = loadArgs();
    if (!errors) {
      main(results.values);
    } else {
      console.error(errors.join("\n"));
    }
  }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [GetObjectLockConfiguration](#)。

PowerShell

Tools for PowerShell V4

範例 1：如果為給定的 S3 儲存貯體啟用物件鎖定組態，此命令會傳回值 'Enabled'。

```
Get-S3ObjectLockConfiguration -BucketName 'amzn-s3-demo-bucket' -Select
ObjectLockConfiguration.ObjectLockEnabled
```

輸出：

```
Value
-----
Enabled
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetObjectLockConfiguration](#)。

Tools for PowerShell V5

範例 1：如果為給定的 S3 儲存貯體啟用物件鎖定組態，此命令會傳回值 'Enabled'。

```
Get-S3ObjectLockConfiguration -BucketName 'amzn-s3-demo-bucket' -Select
ObjectLockConfiguration.ObjectLockEnabled
```

輸出：

```
Value
-----
Enabled
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetObjectLockConfiguration](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得物件鎖定組態。

```
def is_object_lock_enabled(s3_client, bucket: str) -> bool:
    """
    Check if object lock is enabled for a bucket.

    Args:
        s3_client: Boto3 S3 client.
        bucket: The name of the bucket to check.

    Returns:
        True if object lock is enabled, False otherwise.
    """
    try:
        response = s3_client.get_object_lock_configuration(Bucket=bucket)
        return (
            "ObjectLockConfiguration" in response
            and response["ObjectLockConfiguration"]["ObjectLockEnabled"] ==
            "Enabled"
        )
    except s3_client.exceptions.ClientError as e:
        if e.response["Error"]["Code"] == "ObjectLockConfigurationNotFoundError":
            return False
        else:
```

```
raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [GetObjectLockConfiguration](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3->getobjectlockconfiguration(          " oo_result is  
returned for testing purposes. "  
    iv_bucket = iv_bucket_name ).  
    MESSAGE 'Retrieved object lock configuration.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetObjectLockConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetObjectRetention 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetObjectRetention。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [鎖定 Amazon S3 物件](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Get the retention period for an S3 object.
/// </summary>
/// <param name="bucketName">The bucket of the object.</param>
/// <param name="objectKey">The object key.</param>
/// <returns>The object retention details.</returns>
public async Task<ObjectLockRetention> GetObjectRetention(string bucketName,
    string objectKey)
{
    try
    {
        var request = new GetObjectRetentionRequest()
        {
            BucketName = bucketName,
            Key = objectKey
        };

        var response = await _amazonS3.GetObjectRetentionAsync(request);
        Console.WriteLine($"{objectKey} in
{bucketName}: " +
            $"{response.Retention.Mode} until
{response.Retention.RetainUntilDate:d}");
        return response.Retention;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Unable to fetch object lock retention:
'{ex.Message}'");
        return new ObjectLockRetention();
    }
}
```

```
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetObjectRetention](#)。

CLI

AWS CLI

擷取物件的物件保留組態

下列 `get-object-retention` 範例會擷取指定物件的物件保留組態。

```
aws s3api get-object-retention \  
  --bucket amzn-s3-demo-bucket-with-object-lock \  
  --key doc1.rtf
```

輸出：

```
{  
  "Retention": {  
    "Mode": "GOVERNANCE",  
    "RetainUntilDate": "2025-01-01T00:00:00.000Z"  
  }  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetObjectRetention](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (
    "bytes"
    "context"
    "errors"
    "fmt"
    "log"
    "time"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"
    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/aws-sdk-go-v2/service/s3/types"
    "github.com/aws/smithy-go"
)

// S3Actions wraps S3 service actions.
type S3Actions struct {
    S3Client    *s3.Client
    S3Manager   *manager.Uploader
}

// GetObjectRetention retrieves the object retention configuration for an S3
// object.
func (actor S3Actions) GetObjectRetention(ctx context.Context, bucket string, key
string) (*types.ObjectLockRetention, error) {
    var retention *types.ObjectLockRetention
    input := &s3.GetObjectRetentionInput{
        Bucket: aws.String(bucket),
        Key:    aws.String(key),
    }

    output, err := actor.S3Client.GetObjectRetention(ctx, input)
    if err != nil {
        var noKey *types.NoSuchKey
        var apiErr *smithy.GenericAPIError
        if errors.As(err, &noKey) {
            log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
            err = noKey
        } else if errors.As(err, &apiErr) {
            switch apiErr.ErrorCode() {
            case "NoSuchObjectLockConfiguration":
                err = nil
            }
        }
    }
}
```

```
    case "InvalidRequest":
        log.Printf("Bucket %s does not have locking enabled.", bucket)
        err = nil
    }
}
} else {
    retention = output.Retention
}

return retention, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [GetObjectRetention](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
// Get the retention period for an S3 object.
public ObjectLockRetention getObjectRetention(String bucketName, String key){
    try {
        GetObjectRetentionRequest retentionRequest =
        GetObjectRetentionRequest.builder()
            .bucket(bucketName)
            .key(key)
            .build();

        GetObjectRetentionResponse response =
        getClient().getObjectRetention(retentionRequest);
        System.out.println("Object retention for "+key +"
in "+ bucketName +": " + response.retention().mode() +" until "+
response.retention().retainUntilDate() +".");
    }
```

```
        return response.retention();

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        return null;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetObjectRetention](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import {
    GetObjectRetentionCommand,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Log the "RetainUntilDate" for an object in an S3 bucket.
 * @param {{ bucketName: string, key: string }}
 */
export const main = async ({ bucketName, key }) => {
    const client = new S3Client({});

    try {
        const { Retention } = await client.send(
            new GetObjectRetentionCommand({
                Bucket: bucketName,
                Key: key,
            }),
        );
    }
};
```



```
    console.log(
      `${key} in ${bucketName} will be retained until
      ${Retention.RetainUntilDate}`,
    );
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchObjectLockConfiguration"
    ) {
      console.warn(
        `The object "${key}" in the bucket "${bucketName}" does not have an
        ObjectLock configuration.`,
      );
    } else if (caught instanceof S3ServiceException) {
      console.error(
        `Error from S3 while getting object retention settings for
        "${bucketName}". ${caught.name}: ${caught.message}`,
      );
    } else {
      throw caught;
    }
  }
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
  isMain,
  validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
  const options = {
    bucketName: {
      type: "string",
      required: true,
    },
    key: {
      type: "string",
      required: true,
    },
  };
};
const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
```

```
    return { errors, results };
};

if (isMain(import.meta.url)) {
    const { errors, results } = loadArgs();
    if (!errors) {
        main(results.values);
    } else {
        console.error(errors.join("\n"));
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [GetObjectRetention](#)。

PowerShell

Tools for PowerShell V4

範例 1：命令會傳回模式和日期，直到保留物件為止。

```
Get-S3ObjectRetention -BucketName 'amzn-s3-demo-bucket' -Key 'testfile.txt'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetObjectRetention](#)。

Tools for PowerShell V5

範例 1：命令會傳回模式和日期，直到保留物件為止。

```
Get-S3ObjectRetention -BucketName 'amzn-s3-demo-bucket' -Key 'testfile.txt'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetObjectRetention](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetObjectTagging** 與 CLI

下列程式碼範例示範如何使用 GetObjectTagging。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [開始使用索引標籤](#)

CLI

AWS CLI

擷取連接至物件的標籤

下列 `get-object-tagging` 範例會從指定的物件擷取指定的索引鍵值。

```
aws s3api get-object-tagging \  
  --bucket amzn-s3-demo-bucket \  
  --key doc1.rtf
```

輸出：

```
{  
  "TagSet": [  
    {  
      "Value": "confidential",  
      "Key": "designation"  
    }  
  ]  
}
```

下列 `get-object-tagging` 範例會嘗試擷取沒有標籤之物件 `doc2.rtf` 的標籤集。

```
aws s3api get-object-tagging \  
  --bucket amzn-s3-demo-bucket \  
  --key doc2.rtf
```

輸出：

```
{  
  "TagSet": []  
}
```

下列 `get-object-tagging` 範例會擷取具有多個標籤之物件 `doc3.rtf` 的標籤集。

```
aws s3api get-object-tagging \  
  --bucket amzn-s3-demo-bucket \  
  --key doc3.rtf
```

輸出：

```
{  
  "TagSet": [  
    {  
      "Value": "confidential",  
      "Key": "designation"  
    },  
    {  
      "Value": "finance",  
      "Key": "department"  
    },  
    {  
      "Value": "payroll",  
      "Key": "team"  
    }  
  ]  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetObjectTagging](#)。

PowerShell

Tools for PowerShell V4

範例 1：範例會傳回與給定 S3 儲存貯體上存在之物件相關聯的標籤。

```
Get-S3ObjectTagSet -Key 'testfile.txt' -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Key  Value  
---  -  
test value
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetObjectTagging](#)。

Tools for PowerShell V5

範例 1：範例會傳回與給定 S3 儲存貯體上存在之物件相關聯的標籤。

```
Get-S3ObjectTagSet -Key 'testfile.txt' -BucketName 'amzn-s3-demo-bucket'
```

輸出：

```
Key  Value
---  -
test value
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetObjectTagging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetPublicAccessBlock** 與 CLI

下列程式碼範例示範如何使用 GetPublicAccessBlock。

CLI

AWS CLI

設定或修改儲存貯體的封鎖公開存取組態

下列 get-public-access-block 範例顯示指定儲存貯體的封鎖公開存取組態。

```
aws s3api get-public-access-block \
  --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "PublicAccessBlockConfiguration": {
    "IgnorePublicAcls": true,
    "BlockPublicPolicy": true,
    "BlockPublicAcls": true,
    "RestrictPublicBuckets": true
  }
}
```

```
}  
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetPublicAccessBlock](#)。

PowerShell

Tools for PowerShell V4

範例 1：命令會傳回給定 S3 儲存貯體的公有存取區塊組態。

```
Get-S3PublicAccessBlock -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetPublicAccessBlock](#)。

Tools for PowerShell V5

範例 1：命令會傳回給定 S3 儲存貯體的公有存取區塊組態。

```
Get-S3PublicAccessBlock -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetPublicAccessBlock](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

HeadBucket 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 HeadBucket。

Bash

AWS CLI 搭配 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
#####
# function bucket_exists
#
# This function checks to see if the specified bucket already exists.
#
# Parameters:
#     $1 - The name of the bucket to check.
#
# Returns:
#     0 - If the bucket already exists.
#     1 - If the bucket doesn't exist.
#####
function bucket_exists() {
    local bucket_name
    bucket_name=$1

    # Check whether the bucket already exists.
    # We suppress all output - we're interested only in the return code.

    if aws s3api head-bucket \
        --bucket "$bucket_name" \
        >/dev/null 2>&1; then
        return 0 # 0 in Bash script means true.
    else
        return 1 # 1 in Bash script means false.
    fi
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [HeadBucket](#)。

CLI

AWS CLI

下列命令會驗證對名為 `amzn-s3-demo-bucket` 之儲存貯體的存取：

```
aws s3api head-bucket --bucket amzn-s3-demo-bucket
```

如果儲存貯體存在，且您有其存取權，則不會傳回任何輸出。否則，會顯示錯誤訊息。例如：

A client error (404) occurred when calling the HeadBucket operation: Not Found

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [HeadBucket](#)。

Go

SDK for Go V2

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (  
    "bytes"  
    "context"  
    "errors"  
    "fmt"  
    "io"  
    "log"  
    "os"  
    "time"  
  
    "github.com/aws/aws-sdk-go-v2/aws"  
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"  
    "github.com/aws/aws-sdk-go-v2/service/s3"  
    "github.com/aws/aws-sdk-go-v2/service/s3/types"  
    "github.com/aws/smithy-go"  
)  
  
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)  
// actions  
// used in the examples.  
// It contains S3Client, an Amazon S3 service client that is used to perform  
// bucket  
// and object actions.  
type BucketBasics struct {  
    S3Client *s3.Client  
}
```



```
// BucketExists checks whether a bucket exists in the current account.
func (basics BucketBasics) BucketExists(ctx context.Context, bucketName string)
    (bool, error) {
    _, err := basics.S3Client.HeadBucket(ctx, &s3.HeadBucketInput{
        Bucket: aws.String(bucketName),
    })
    exists := true
    if err != nil {
        var apiError smithy.APIError
        if errors.As(err, &apiError) {
            switch apiError.(type) {
            case *types.NotFound:
                log.Printf("Bucket %v is available.\n", bucketName)
                exists = false
                err = nil
            default:
                log.Printf("Either you don't have access to bucket %v or another error
occurred. "+
                    "Here's what happened: %v\n", bucketName, err)
            }
        }
    } else {
        log.Printf("Bucket %v exists and you already own it.", bucketName)
    }

    return exists, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [HeadBucket](#)。

PowerShell

Tools for PowerShell V5

範例 1：當使用者有權存取現有儲存貯體時，此命令會傳回 HTTP 狀態碼為 200 OK 的輸出。BucketArn 參數僅支援 S3 目錄儲存貯體。

```
Get-S3HeadBucket -BucketName amzn-s3-demo-bucket
```

輸出：

```
AccessPointAlias    : False
BucketArn           :
BucketLocationName  :
BucketLocationType  :
BucketRegion        : us-east-2
ResponseMetadata    : Amazon.Runtime.ResponseMetadata
ContentLength       : 0
HttpStatusCode       : OK
```

範例 2：此命令會針對不存在的儲存貯體擲回 HTTP 狀態碼 NotFound 的錯誤。

```
Get-S3HeadBucket -BucketName amzn-s3-non-existing-bucket
```

輸出：

```
Get-S3HeadBucket: Error making request with Error Code NotFound and Http Status
Code NotFound. No further error information was returned by the service.
```

範例 3：對於使用者無權存取的現有儲存貯體，此命令會擲回 HTTP 狀態碼為 Forbidden (禁止) 的錯誤。

```
Get-S3HeadBucket -BucketName amzn-s3-no-access-bucket
```

輸出：

```
Get-S3HeadBucket: Error making request with Error Code Forbidden and Http Status
Code Forbidden. No further error information was returned by the service.
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [HeadBucket](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def exists(self):
        """
        Determine whether the bucket exists and you have access to it.

        :return: True when the bucket exists; otherwise, False.
        """
        try:
            self.bucket.meta.client.head_bucket(Bucket=self.bucket.name)
            logger.info("Bucket %s exists.", self.bucket.name)
            exists = True
        except ClientError:
            logger.warning(
                "Bucket %s doesn't exist or you don't have access to it.",
                self.bucket.name,
            )
            exists = False
        return exists
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [HeadBucket](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3->headbucket(          " oo_result is returned for  
testing purposes. "  
    iv_bucket = iv_bucket_name ).  
    MESSAGE 'Bucket exists and you have access to it.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [HeadBucket](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

HeadObject 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 HeadObject。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [檢查物件是否存在](#)
- [物件儲存入門](#)

CLI

AWS CLI

下列命令會擷取名為 `amzn-s3-demo-bucket` 之儲存貯體中物件的中繼資料：

```
aws s3api head-object --bucket amzn-s3-demo-bucket --key index.html
```

輸出：

```
{
  "AcceptRanges": "bytes",
  "ContentType": "text/html",
  "LastModified": "Thu, 16 Apr 2015 18:19:14 GMT",
  "ContentLength": 77,
  "VersionId": "null",
  "ETag": "\"30a6ec7e1a9ad79c203d05a589c8b400\"",
  "Metadata": {}
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [HeadObject](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

判斷物件的內容類型。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class GetObjectContentType {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <bucketName> <keyName>

            Where:
                bucketName - The Amazon S3 bucket name.\s
                keyName - The key name.\s
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        getContentType(s3, bucketName, keyName);
        s3.close();
    }

    /**
     * Retrieves the content type of an object stored in an Amazon S3 bucket.
     *
     * @param s3 an instance of the {@link S3Client} class, which is used to
     * interact with the Amazon S3 service
     * @param bucketName the name of the S3 bucket where the object is stored
     * @param keyName the key (file name) of the object in the S3 bucket
     */
}
```

```

    */
    public static void getContentType(S3Client s3, String bucketName, String
keyName) {
        try {
            HeadObjectRequest objectRequest = HeadObjectRequest.builder()
                .key(keyName)
                .bucket(bucketName)
                .build();

            HeadObjectResponse objectHead = s3.headObject(objectRequest);
            String type = objectHead.contentType();
            System.out.println("The object content type is " + type);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}

```

取得物件的還原狀態。

```

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

public class GetObjectRestoreStatus {
    public static void main(String[] args) {
        final String usage = ""

        Usage:
            <bucketName> <keyName>\s

        Where:
            bucketName - The Amazon S3 bucket name.\s
            keyName - A key name that represents the object.\s
        """;

        if (args.length != 2) {

```

```
        System.out.println(usage);
        System.exit(1);
    }

    String bucketName = args[0];
    String keyName = args[1];
    Region region = Region.US_EAST_1;
    S3Client s3 = S3Client.builder()
        .region(region)
        .build();

    checkStatus(s3, bucketName, keyName);
    s3.close();
}

/**
 * Checks the restoration status of an Amazon S3 object.
 *
 * @param s3          an instance of the {@link S3Client} class used to
interact with the Amazon S3 service
 * @param bucketName the name of the Amazon S3 bucket where the object is
stored
 * @param keyName     the name of the Amazon S3 object to be checked
 * @throws S3Exception if an error occurs while interacting with the Amazon
S3 service
 */
public static void checkStatus(S3Client s3, String bucketName, String
keyName) {
    try {
        HeadObjectRequest headObjectRequest = HeadObjectRequest.builder()
            .bucket(bucketName)
            .key(keyName)
            .build();

        HeadObjectResponse response = s3.headObject(headObjectRequest);
        System.out.println("The Amazon S3 object restoration status is " +
response.restore());

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```


- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [HeadObject](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-s3'

# Wraps Amazon S3 object actions.
class ObjectExistsWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Checks whether the object exists.
  #
  # @return [Boolean] True if the object exists; otherwise false.
  def exists?
    @object.exists?
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't check existence of object
    #{@object.bucket.name}:#{@object.key}. Here's why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "amzn-s3-demo-bucket"
  object_key = "my-object.txt"
```

```
wrapper = ObjectExistsWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
exists = wrapper.exists?

puts "Object #{object_key} #{exists ? 'does' : 'does not'} exist."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [HeadObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListBucketAnalyticsConfigurations** 與 CLI

下列程式碼範例示範如何使用 ListBucketAnalyticsConfigurations。

CLI

AWS CLI

擷取儲存貯體的分析組態清單

下列 list-bucket-analytics-configurations 會擷取指定儲存貯體的分析組態清單。

```
aws s3api list-bucket-analytics-configurations \
  --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "AnalyticsConfigurationList": [
    {
      "StorageClassAnalysis": {},
      "Id": "1"
    }
  ],
  "IsTruncated": false
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListBucketAnalyticsConfigurations](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回給定 S3 儲存貯體的前 100 個分析組態。

```
Get-S3BucketAnalyticsConfigurationList -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [ListBucketAnalyticsConfigurations](#)。

Tools for PowerShell V5

範例 1：此命令會傳回給定 S3 儲存貯體的前 100 個分析組態。

```
Get-S3BucketAnalyticsConfigurationList -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [ListBucketAnalyticsConfigurations](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListBucketInventoryConfigurations** 與 CLI

下列程式碼範例示範如何使用 ListBucketInventoryConfigurations。

CLI

AWS CLI

擷取儲存貯體的庫存組態清單

下列 list-bucket-inventory-configurations 範例列出指定儲存貯體的庫存組態。

```
aws s3api list-bucket-inventory-configurations \
  --bucket amzn-s3-demo-bucket
```

輸出：

```
{
```

```
"InventoryConfigurationList": [
  {
    "IsEnabled": true,
    "Destination": {
      "S3BucketDestination": {
        "Format": "ORC",
        "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket",
        "AccountId": "123456789012"
      }
    },
    "IncludedObjectVersions": "Current",
    "Id": "1",
    "Schedule": {
      "Frequency": "Weekly"
    }
  },
  {
    "IsEnabled": true,
    "Destination": {
      "S3BucketDestination": {
        "Format": "CSV",
        "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket",
        "AccountId": "123456789012"
      }
    },
    "IncludedObjectVersions": "Current",
    "Id": "2",
    "Schedule": {
      "Frequency": "Daily"
    }
  }
],
"IsTruncated": false
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListBucketInventoryConfigurations](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回給定 S3 儲存貯體的前 100 個庫存組態。

```
Get-S3BucketInventoryConfigurationList -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [ListBucketInventoryConfigurations](#)。

Tools for PowerShell V5

範例 1：此命令會傳回給定 S3 儲存貯體的前 100 個庫存組態。

```
Get-S3BucketInventoryConfigurationList -BucketName 'amzn-s3-demo-bucket'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [ListBucketInventoryConfigurations](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListBuckets 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListBuckets。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
namespace ListBucketsExample
{
    using System;
    using System.Collections.Generic;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example uses the AWS SDK for .NET to list the Amazon Simple Storage
```

```
/// Service (Amazon S3) buckets belonging to the default account.
/// </summary>
public class ListBuckets
{
    private static IAmazonS3 _s3Client;

    /// <summary>
    /// Get a list of the buckets owned by the default user.
    /// </summary>
    /// <param name="client">An initialized Amazon S3 client object.</param>
    /// <returns>The response from the ListingBuckets call that contains a
    /// list of the buckets owned by the default user.</returns>
    public static async Task<ListBucketsResponse> GetBuckets(IAmazonS3
client)
    {
        return await client.ListBucketsAsync();
    }

    /// <summary>
    /// This method lists the name and creation date for the buckets in
    /// the passed List of S3 buckets.
    /// </summary>
    /// <param name="bucketList">A List of S3 bucket objects.</param>
    public static void DisplayBucketList(List<S3Bucket> bucketList)
    {
        bucketList
            .ForEach(b => Console.WriteLine($"Bucket name: {b.BucketName},
created on: {b.CreationDate}"));
    }

    public static async Task Main()
    {
        // The client uses the AWS Region of the default user.
        // If the Region where the buckets were created is different,
        // pass the Region to the client constructor. For example:
        // _s3Client = new AmazonS3Client(RegionEndpoint.USEast1);
        _s3Client = new AmazonS3Client();
        var response = await GetBuckets(_s3Client);
        DisplayBucketList(response.Buckets);
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [ListBuckets](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::listBuckets(const Aws::S3::S3ClientConfiguration &clientConfig)
{
    Aws::S3::S3Client client(clientConfig);

    auto outcome = client.ListBuckets();

    bool result = true;
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed with error: " << outcome.GetError() << std::endl;
        result = false;
    } else {
        std::cout << "Found " << outcome.GetResult().GetBuckets().size() << "
buckets\n";
        for (auto &b: outcome.GetResult().GetBuckets()) {
            std::cout << b.GetName() << std::endl;
        }
    }

    return result;
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [ListBuckets](#)。

CLI

AWS CLI

下列命令會使用 `list-buckets` 命令來顯示所有 Amazon S3 儲存貯體的名稱 (跨所有區域)：

```
aws s3api list-buckets --query "Buckets[].Name"
```

查詢選項會將 `list-buckets` 的輸出向下篩選出僅儲存貯體名稱。

如需儲存貯體的詳細資訊，請參閱《Amazon S3 開發人員指南》中的「使用 Amazon S3 儲存貯體」。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListBuckets](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (  
    "bytes"  
    "context"  
    "errors"  
    "fmt"  
    "io"  
    "log"  
    "os"  
    "time"  
  
    "github.com/aws/aws-sdk-go-v2/aws"  
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"  
    "github.com/aws/aws-sdk-go-v2/service/s3"  
    "github.com/aws/aws-sdk-go-v2/service/s3/types"  
    "github.com/aws/smithy-go"
```



```
)

// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
// actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
// bucket
// and object actions.
type BucketBasics struct {
    S3Client *s3.Client
}

// ListBuckets lists the buckets in the current account.
func (basics BucketBasics) ListBuckets(ctx context.Context) ([]types.Bucket,
    error) {
    var err error
    var output *s3.ListBucketsOutput
    var buckets []types.Bucket
    bucketPaginator := s3.NewListBucketsPaginator(basics.S3Client,
        &s3.ListBucketsInput{})
    for bucketPaginator.HasMorePages() {
        output, err = bucketPaginator.NextPage(ctx)
        if err != nil {
            var apiErr smithy.APIError
            if errors.As(err, &apiErr) && apiErr.ErrorCode() == "AccessDenied" {
                fmt.Println("You don't have permission to list buckets for this account.")
                err = apiErr
            } else {
                log.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
            }
            break
        } else {
            buckets = append(buckets, output.Buckets...)
        }
    }
    return buckets, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [ListBuckets](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.paginators.ListBucketsIterable;
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class ListBuckets {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        listAllBuckets(s3);
    }

    /**
     * Lists all the S3 buckets available in the current AWS account.
     *
     * @param s3 The {@link S3Client} instance to use for interacting with the
     * Amazon S3 service.
     */
    public static void listAllBuckets(S3Client s3) {
        ListBucketsIterable response = s3.listBucketsPaginator();
        response.buckets().forEach(bucket ->
```

```
        System.out.println("Bucket Name: " + bucket.name());  
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListBuckets](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

列出儲存貯體。

```
import {  
    paginateListBuckets,  
    S3Client,  
    S3ServiceException,  
} from "@aws-sdk/client-s3";  
  
/**  
 * List the Amazon S3 buckets in your account.  
 */  
export const main = async () => {  
    const client = new S3Client({});  
    /** @type {?import('@aws-sdk/client-s3').Owner} */  
    let Owner = null;  
  
    /** @type {import('@aws-sdk/client-s3').Bucket[]} */  
    const Buckets = [];  
  
    try {  
        const paginator = paginateListBuckets({ client }, {});  
  
        for await (const page of paginator) {  
            if (!Owner) {  
                Owner = page.Owner;  
            }  
        }  
    }  
}
```

```
Buckets.push(...page.Buckets);
}

console.log(
  `${Owner.DisplayName} owns ${Buckets.length} bucket${
    Buckets.length === 1 ? "" : "s"
  }:`,
);
console.log(`${Buckets.map((b) => ` • ${b.Name}`).join("\n")}`);
} catch (caught) {
  if (caught instanceof S3ServiceException) {
    console.error(
      `Error from S3 while listing buckets.  ${caught.name}:
${caught.message}`,
    );
  } else {
    throw caught;
  }
}
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [ListBuckets](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回所有 S3 儲存貯體。

```
Get-S3Bucket
```

範例 2：此命令會傳回名為 "test-files" 的儲存貯體

```
Get-S3Bucket -BucketName amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [ListBuckets](#)。

Tools for PowerShell V5

範例 1：此命令會傳回所有 S3 儲存貯體。

```
Get-S3Bucket
```

範例 2：此命令會傳回名為 "test-files" 的儲存貯體

```
Get-S3Bucket -BucketName amzn-s3-demo-bucket
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [ListBuckets](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    @staticmethod
    def list(s3_resource):
        """
        Get the buckets in all Regions for the current account.
```

```

        :param s3_resource: A Boto3 S3 resource. This is a high-level resource in
Boto3
                                that contains collections and factory methods to
create
                                other high-level S3 sub-resources.
        :return: The list of buckets.
        """
        try:
            buckets = list(s3_resource.buckets.all())
            logger.info("Got buckets: %s.", buckets)
        except ClientError:
            logger.exception("Couldn't get buckets.")
            raise
        else:
            return buckets

```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [ListBuckets](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

require 'aws-sdk-s3'

# Wraps Amazon S3 resource actions.
class BucketListWrapper
  attr_reader :s3_resource

  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def initialize(s3_resource)
    @s3_resource = s3_resource

```

```
end

# Lists buckets for the current account.
#
# @param count [Integer] The maximum number of buckets to list.
def list_buckets(count)
  puts 'Found these buckets:'
  @s3_resource.buckets.each do |bucket|
    puts "\t#{bucket.name}"
    count -= 1
    break if count.zero?
  end
  true
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list buckets. Here's why: #{e.message}"
  false
end
end

# Example usage:
def run_demo
  wrapper = BucketListWrapper.new(Aws::S3::Resource.new)
  wrapper.list_buckets(25)
end

run_demo if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [ListBuckets](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
async fn show_buckets(
  strict: bool,
```

```

    client: &Client,
    region: BucketLocationConstraint,
) -> Result<(), S3ExampleError> {
    let mut buckets = client.list_buckets().into_paginator().send();

    let mut num_buckets = 0;
    let mut in_region = 0;

    while let Some(Ok(output)) = buckets.next().await {
        for bucket in output.buckets() {
            num_buckets += 1;
            if strict {
                let r = client
                    .get_bucket_location()
                    .bucket(bucket.name().unwrap_or_default())
                    .send()
                    .await?;

                if r.location_constraint() == Some(&region) {
                    println!("{}", bucket.name().unwrap_or_default());
                    in_region += 1;
                }
            } else {
                println!("{}", bucket.name().unwrap_or_default());
            }
        }
    }

    println!();
    if strict {
        println!(
            "Found {} buckets in the {} region out of a total of {} buckets.",
            in_region, region, num_buckets
        );
    } else {
        println!("Found {} buckets in all regions.", num_buckets);
    }

    Ok(())
}

```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [ListBuckets](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3->listbuckets(           " oo_result is returned for  
testing purposes. "  
    ).  
    DATA(lv_bucket_count) = lines( oo_result->get_buckets( ) ).  
    MESSAGE |Retrieved { lv_bucket_count } buckets in all regions.| TYPE 'I'.  
CATCH /aws1/cx_rt_generic.  
    MESSAGE 'Unable to list buckets.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ListBuckets](#)。

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import AWSS3  
  
/// Return an array containing information about every available bucket.  
///  
/// - Returns: An array of ``S3ClientTypes.Bucket`` objects describing  
/// each bucket.
```

```
public func getAllBuckets() async throws -> [S3ClientTypes.Bucket] {
    return try await client.listBuckets(input: ListBucketsInput())
}
```

- 如需 API 詳細資訊，請參閱《適用於 Swift 的 AWS SDK API 參考》中的 [ListBuckets](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListMultipartUploads 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListMultipartUploads。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [刪除未完成的分段上傳](#)

CLI

AWS CLI

下列命令會列出名為 amzn-s3-demo-bucket 之儲存貯體的所有作用中分段上傳：

```
aws s3api list-multipart-uploads --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "Uploads": [
    {
      "Initiator": {
        "DisplayName": "username",
        "ID": "arn:aws:iam::0123456789012:user/username"
      },
      "Initiated": "2015-06-02T18:01:30.000Z",
      "UploadId":
"dfRtDYU0WWCCcH43C3WFbkR0NycyCpTJJvxu2i5GYkZ1jF.Yxwh6XG7WfS2vC4to6HiV6Yj1x.cph0gtNBtJ8P3",
      "StorageClass": "STANDARD",
      "Key": "multipart/01",
      "Owner": {
```

```
        "DisplayName": "aws-account-name",
        "ID":
        "100719349fc3b6dcd7c820a124bf7aec408092c3d7b51b38494939801fc248b"
    }
    ],
    "CommonPrefixes": []
}
```

進行中的分段上傳會在 Amazon S3 中產生儲存成本。完成或中止作用中的分段上傳，以從您的帳戶中移除該部分。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListMultipartUploads](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListMultipartUploadsRequest;
import software.amazon.awssdk.services.s3.model.ListMultipartUploadsResponse;
import software.amazon.awssdk.services.s3.model.MultipartUpload;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
```

```
public class ListMultipartUploads {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <bucketName>\s

            Where:
                bucketName - The name of the Amazon S3 bucket where an in-
progress multipart upload is occurring.
            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();
        listUploads(s3, bucketName);
        s3.close();
    }

    /**
     * Lists the multipart uploads currently in progress in the specified Amazon
S3 bucket.
     *
     * @param s3 the S3Client object used to interact with Amazon S3
     * @param bucketName the name of the Amazon S3 bucket to list the multipart
uploads for
     */
    public static void listUploads(S3Client s3, String bucketName) {
        try {
            ListMultipartUploadsRequest listMultipartUploadsRequest =
ListMultipartUploadsRequest.builder()
                .bucket(bucketName)
                .build();

            ListMultipartUploadsResponse response =
s3.listMultipartUploads(listMultipartUploadsRequest);
            List<MultipartUpload> uploads = response.uploads();
        }
    }
}
```

```
        for (MultipartUpload upload : uploads) {
            System.out.println("Upload in progress: Key = \"" + upload.key()
+ "\", id = " + upload.uploadId());
        }

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListMultipartUploads](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListObjectVersions 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListObjectVersions。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [物件儲存入門](#)
- [使用版本化物件](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
```

```

using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example lists the versions of the objects in a version enabled
/// Amazon Simple Storage Service (Amazon S3) bucket.
/// </summary>
public class ListObjectVersions
{
    public static async Task Main()
    {
        string bucketName = "amzn-s3-demo-bucket";

        // If the AWS Region where your bucket is defined is different from
        // the AWS Region where the Amazon S3 bucket is defined, pass the
constant
        // for the AWS Region to the client constructor like this:
        //     var client = new AmazonS3Client(RegionEndpoint.USWest2);
        IAmazonS3 client = new AmazonS3Client();
        await GetObjectListWithAllVersionsAsync(client, bucketName);
    }

    /// <summary>
    /// This method lists all versions of the objects within an Amazon S3
    /// version enabled bucket.
    /// </summary>
    /// <param name="client">The initialized client object used to call
    /// ListVersionsAsync.</param>
    /// <param name="bucketName">The name of the version enabled Amazon S3
bucket
    /// for which you want to list the versions of the contained objects.</
param>
    public static async Task GetObjectListWithAllVersionsAsync(IAmazonS3
client, string bucketName)
    {
        try
        {
            // When you instantiate the ListVersionRequest, you can
            // optionally specify a key name prefix in the request
            // if you want a list of object versions of a specific object.

            // For this example we set a small limit in MaxKeys to return
            // a small list of versions.
            ListVersionsRequest request = new ListVersionsRequest()

```

```
        {
            BucketName = bucketName,
            MaxKeys = 2,
        };

        do
        {
            ListVersionsResponse response = await
client.ListVersionsAsync(request);

            // Process response.
            foreach (S3ObjectVersion entry in response.Versions)
            {
                Console.WriteLine($"key: {entry.Key} size:
{entry.Size}");
            }

            // If response is truncated, set the marker to get the next
            // set of keys.
            if (response.IsTruncated)
            {
                request.KeyMarker = response.NextKeyMarker;
                request.VersionIdMarker = response.NextVersionIdMarker;
            }
            else
            {
                request = null;
            }
        }
        while (request != null);
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error: '{ex.Message}'");
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [ListObjectVersions](#)。

CLI

AWS CLI

下列命令會擷取名為 `amzn-s3-demo-bucket` 之儲存貯體中物件的版本資訊：

```
aws s3api list-object-versions --bucket amzn-s3-demo-bucket --prefix index.html
```

輸出：

```
{
  "DeleteMarkers": [
    {
      "Owner": {
        "DisplayName": "my-username",
        "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
      },
      "IsLatest": true,
      "VersionId": "B2VsEK5saUNNHKc0AJj7hIE86RozToyq",
      "Key": "index.html",
      "LastModified": "2015-11-10T00:57:03.000Z"
    },
    {
      "Owner": {
        "DisplayName": "my-username",
        "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
      },
      "IsLatest": false,
      "VersionId": ".FLQEZscLIcfxSq.jsFJ.szUkmng2Yw6",
      "Key": "index.html",
      "LastModified": "2015-11-09T23:32:20.000Z"
    }
  ],
  "Versions": [
    {
      "LastModified": "2015-11-10T00:20:11.000Z",
      "VersionId": "Rb_l2T8UHDkFEwCgJjhlgP0ZC0qJ.vpD",
      "ETag": "\"0622528de826c0df5db1258a23b80be5\"",
      "StorageClass": "STANDARD",
      "Key": "index.html",
      "Owner": {
```



```

        "DisplayName": "my-username",
        "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
    },
    "IsLatest": false,
    "Size": 38
},
{
    "LastModified": "2015-11-09T23:26:41.000Z",
    "VersionId": "rasWWGpgk9E4s0LyTJgusGeRQKLVIAff",
    "ETag": "\"06225825b8028de826c0df5db1a23be5\"",
    "StorageClass": "STANDARD",
    "Key": "index.html",
    "Owner": {
        "DisplayName": "my-username",
        "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
    },
    "IsLatest": false,
    "Size": 38
},
{
    "LastModified": "2015-11-09T22:50:50.000Z",
    "VersionId": "null",
    "ETag": "\"d1f45267a863c8392e07d24dd592f1b9\"",
    "StorageClass": "STANDARD",
    "Key": "index.html",
    "Owner": {
        "DisplayName": "my-username",
        "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
    },
    "IsLatest": false,
    "Size": 533823
}
]
}

```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListObjectVersions](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (
    "bytes"
    "context"
    "errors"
    "fmt"
    "log"
    "time"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"
    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/aws-sdk-go-v2/service/s3/types"
    "github.com/aws/smithy-go"
)

// S3Actions wraps S3 service actions.
type S3Actions struct {
    S3Client    *s3.Client
    S3Manager   *manager.Uploader
}

// ListObjectVersions lists all versions of all objects in a bucket.
func (actor S3Actions) ListObjectVersions(ctx context.Context, bucket string)
    ([]types.ObjectVersion, error) {
    var err error
    var output *s3.ListObjectVersionsOutput
    var versions []types.ObjectVersion
    input := &s3.ListObjectVersionsInput{Bucket: aws.String(bucket)}
    versionPaginator := s3.NewListObjectVersionsPaginator(actor.S3Client, input)
```

```
for versionPaginator.HasMorePages() {
    output, err = versionPaginator.NextPage(ctx)
    if err != nil {
        var noBucket *types.NoSuchBucket
        if errors.As(err, &noBucket) {
            log.Printf("Bucket %s does not exist.\n", bucket)
            err = noBucket
        }
        break
    } else {
        versions = append(versions, output.Versions...)
    }
}
return versions, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [ListObjectVersions](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
async fn show_versions(client: &Client, bucket: &str) -> Result<(), Error> {
    let resp = client.list_object_versions().bucket(bucket).send().await?;

    for version in resp.versions() {
        println!("{}", version.key().unwrap_or_default());
        println!(" version ID: {}", version.version_id().unwrap_or_default());
        println!();
    }

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [ListObjectVersions](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3->listobjectversions(           " oo_result is returned  
for testing purposes. "  
        iv_bucket = iv_bucket_name  
        iv_prefix = iv_prefix ).  
    MESSAGE 'Retrieved object versions.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ListObjectVersions](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListObjects** 與 CLI

下列程式碼範例示範如何使用 ListObjects。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [建立列出 Amazon S3 物件的網頁](#)

CLI

AWS CLI

下列範例使用 `list-objects` 命令來顯示指定儲存貯體中所有物件的名稱：

```
aws s3api list-objects --bucket text-content --query 'Contents[].{Key: Key, Size: Size}'
```

此範例使用 `--query` 引數，將 `list-objects` 的輸出篩選為每個物件的金鑰值和大小。

如需物件的詳細資訊，請參閱《Amazon S3 開發人員指南》中的使用 Amazon S3 物件。

- 如需 API 詳細資訊，請參閱《AWS CLI API 參考》中的 [ListObjects](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會擷取儲存貯體 "test-files" 中所有項目的相關資訊。

```
Get-S3Object -BucketName amzn-s3-demo-bucket
```

範例 2：此命令會從儲存貯體 "test-files" 擷取項目 "sample.txt" 的相關資訊。

```
Get-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt
```

範例 3：此命令會從儲存貯體 "test-files" 擷取字首為 "sample" 之所有項目的相關資訊。

```
Get-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix sample
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [ListObjects](#)。

Tools for PowerShell V5

範例 1：此命令會擷取儲存貯體 "test-files" 中所有項目的相關資訊。

```
Get-S3Object -BucketName amzn-s3-demo-bucket
```

範例 2：此命令會從儲存貯體 "test-files" 擷取項目 "sample.txt" 的相關資訊。

```
Get-S3Object -BucketName amzn-s3-demo-bucket -Key sample.txt
```

範例 3：此命令會從儲存貯體 "test-files" 擷取字首為 "sample" 之所有項目的相關資訊。

```
Get-S3Object -BucketName amzn-s3-demo-bucket -KeyPrefix sample
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [ListObjects](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListObjectsV2 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListObjectsV2。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)
- [刪除儲存貯體中的所有物件](#)
- [物件儲存入門](#)

.NET

適用於 .NET 的 SDK (v4)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Shows how to list the objects in an Amazon S3 bucket.
/// </summary>
/// <param name="bucketName">The name of the bucket for which to list.
/// <param name="printList">True to print out the list.
```

```
/// <returns>The collection of objects.</returns>
public async Task<List<S3Object>?> ListBucketContentsAsync(string bucketName,
bool printList = true)
{
    try
    {
        var request = new ListObjectsV2Request
        {
            BucketName = bucketName,
            MaxKeys = 5,
        };

        if (printList)
        {
            Console.WriteLine("-----");
            Console.WriteLine($"Listing the contents of {bucketName}:");
            Console.WriteLine("-----");
        }

        var listObjectsV2Paginator = _amazonS3.Paginators.ListObjectsV2(new
ListObjectsV2Request
        {
            BucketName = bucketName,
        });
        var s3objects = new List<S3Object>();
        await foreach (var response in listObjectsV2Paginator.Responses)
        {
            if (response.S3objects != null)
            {
                s3objects.AddRange(response.S3objects);
            }
        }

        if (printList)
        {
            Console.WriteLine($"Number of Objects: {s3objects.Count}");
            foreach (var entry in s3objects)
            {
                Console.WriteLine($"Key = {entry.Key} Size = {entry.Size}");
            }
        }

        return s3objects;
    }
}
```

```
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Error encountered on server.
Message: '{ex.Message}' getting list of objects.");
            return null;
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用分頁程式列出物件。

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// The following example lists objects in an Amazon Simple Storage
/// Service (Amazon S3) bucket.
/// </summary>
public class ListObjectsPaginator
{
    private const string BucketName = "amzn-s3-demo-bucket";

    public static async Task Main()
    {
        IAmazonS3 s3Client = new AmazonS3Client();

        Console.WriteLine($"Listing the objects contained in {BucketName}:
\n");

        await ListingObjectsAsync(s3Client, BucketName);
    }
}
```



```
/// <summary>
/// This method uses a paginator to retrieve the list of objects in an
/// an Amazon S3 bucket.
/// </summary>
/// <param name="client">An Amazon S3 client object.</param>
/// <param name="bucketName">The name of the S3 bucket whose objects
/// you want to list.</param>
public static async Task ListingObjectsAsync(IAmazonS3 client, string
bucketName)
{
    var listObjectsV2Paginator = client.Paginators.ListObjectsV2(new
ListObjectsV2Request
    {
        BucketName = bucketName,
    });

    await foreach (var response in listObjectsV2Paginator.Responses)
    {
        Console.WriteLine($"HttpStatusCode: {response.HttpStatusCode}");
        Console.WriteLine($"Number of Keys: {response.KeyCount}");
        foreach (var entry in response.S3Objects)
        {
            Console.WriteLine($"Key = {entry.Key} Size = {entry.Size}");
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

Bash

AWS CLI 搭配 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#     $1 - The name of the bucket.
#
# Returns:
#     The list of files in text format.
#     And:
#     0 - If successful.
#     1 - If it fails.
#####
function list_items_in_bucket() {
    local bucket_name=$1
    local response

    response=$(aws s3api list-objects \
        --bucket "$bucket_name" \
        --output text \
        --query 'Contents[].{Key: Key, Size: Size}')

    # shellcheck disable=SC2181
    if [[ ${?} -eq 0 ]]; then
        echo "$response"
    else
        errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
        return 1
    fi
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListObjectsV2](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::listObjects(const Aws::String &bucketName,
                             Aws::Vector<Aws::String> &keysResult,
                             const Aws::S3::S3ClientConfiguration &clientConfig)
{
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::ListObjectsV2Request request;
    request.WithBucket(bucketName);

    Aws::String continuationToken; // Used for pagination.
    Aws::Vector<Aws::S3::Model::Object> allObjects;

    do {
        if (!continuationToken.empty()) {
            request.SetContinuationToken(continuationToken);
        }

        auto outcome = s3Client.ListObjectsV2(request);

        if (!outcome.IsSuccess()) {
            std::cerr << "Error: listObjects: " <<
                outcome.GetError().GetMessage() << std::endl;
            return false;
        } else {
            Aws::Vector<Aws::S3::Model::Object> objects =
                outcome.GetResult().GetContents();

            allObjects.insert(allObjects.end(), objects.begin(), objects.end());
            continuationToken = outcome.GetResult().GetNextContinuationToken();
        }
    }
```

```
    } while (!continuationToken.empty());

    std::cout << allObjects.size() << " object(s) found:" << std::endl;

    for (const auto &object: allObjects) {
        std::cout << " " << object.GetKey() << std::endl;
        keysResult.push_back(object.GetKey());
    }

    return true;
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

CLI

AWS CLI

取得儲存貯體中的物件清單

下列 `list-objects-v2` 範例會列出指定儲存貯體中的物件。

```
aws s3api list-objects-v2 \
  --bucket amzn-s3-demo-bucket
```

輸出：

```
{
  "Contents": [
    {
      "LastModified": "2019-11-05T23:11:50.000Z",
      "ETag": "\"621503c373607d548b37cff8778d992c\"",
      "StorageClass": "STANDARD",
      "Key": "doc1.rtf",
      "Size": 391
    },
    {
      "LastModified": "2019-11-05T23:11:50.000Z",
      "ETag": "\"a2cecc36ab7c7fe3a71a273b9d45b1b5\"",
      "StorageClass": "STANDARD",
      "Key": "doc2.rtf",
      "Size": 373
    }
  ]
}
```

```
    },
    {
      "LastModified": "2019-11-05T23:11:50.000Z",
      "ETag": "\"08210852f65a2e9cb999972539a64d68\"",
      "StorageClass": "STANDARD",
      "Key": "doc3.rtf",
      "Size": 399
    },
    {
      "LastModified": "2019-11-05T23:11:50.000Z",
      "ETag": "\"d1852dd683f404306569471af106988e\"",
      "StorageClass": "STANDARD",
      "Key": "doc4.rtf",
      "Size": 6225
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListObjectsV2](#)。

Go

SDK for Go V2

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (
    "bytes"
    "context"
    "errors"
    "fmt"
    "io"
    "log"
    "os"
    "time"

    "github.com/aws/aws-sdk-go-v2/aws"
```

```
"github.com/aws/aws-sdk-go-v2/feature/s3/manager"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/aws/aws-sdk-go-v2/service/s3/types"
"github.com/aws/smithy-go"
)

// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
// actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
// bucket
// and object actions.
type BucketBasics struct {
    S3Client *s3.Client
}

// ListObjects lists the objects in a bucket.
func (basics BucketBasics) ListObjects(ctx context.Context, bucketName string)
([]types.Object, error) {
    var err error
    var output *s3.ListObjectsV2Output
    input := &s3.ListObjectsV2Input{
        Bucket: aws.String(bucketName),
    }
    var objects []types.Object
    objectPaginator := s3.NewListObjectsV2Paginator(basics.S3Client, input)
    for objectPaginator.HasMorePages() {
        output, err = objectPaginator.NextPage(ctx)
        if err != nil {
            var noBucket *types.NoSuchBucket
            if errors.As(err, &noBucket) {
                log.Printf("Bucket %s does not exist.\n", bucketName)
                err = noBucket
            }
            break
        } else {
            objects = append(objects, output.Contents...)
        }
    }
    return objects, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously lists all objects in the specified S3 bucket.
 *
 * @param bucketName the name of the S3 bucket to list objects for
 * @return a {@link CompletableFuture} that completes when all objects have
 * been listed
 */
public CompletableFuture<Void> listAllObjectsAsync(String bucketName) {
    ListObjectsV2Request initialRequest = ListObjectsV2Request.builder()
        .bucket(bucketName)
        .maxKeys(1)
        .build();

    ListObjectsV2Publisher paginator =
getAsyncClient().listObjectsV2Paginator(initialRequest);
    return paginator.subscribe(response -> {
        response.contents().forEach(s3object -> {
            logger.info("Object key: " + s3object.key());
        });
    }).thenRun(() -> {
        logger.info("Successfully listed all objects in the bucket: " +
bucketName);
    }).exceptionally(ex -> {
        throw new RuntimeException("Failed to list objects", ex);
    });
}
```

使用分頁列出物件。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.paginators.ListObjectsV2Iterable;

public class ListObjectsPaginated {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <bucketName>\s

            Where:
                bucketName - The Amazon S3 bucket from which objects are read.\s
            "";

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        listBucketObjects(s3, bucketName);
        s3.close();
    }

    /**
     * Lists the objects in the specified S3 bucket.
     *
     * @param s3 the S3Client instance used to interact with Amazon S3
     * @param bucketName the name of the S3 bucket to list the objects from
     */
}
```



```
public static void listBucketObjects(S3Client s3, String bucketName) {
    try {
        ListObjectsV2Request listReq = ListObjectsV2Request.builder()
            .bucket(bucketName)
            .maxKeys(1)
            .build();

        ListObjectsV2Iterable listRes = s3.listObjectsV2Paginator(listReq);
        listRes.stream()
            .flatMap(r -> r.contents().stream())
            .forEach(content -> System.out.println(" Key: " + content.key() +
" size = " + content.size()));

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListObjectsV2](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

條列儲存貯體中的所有物件。若有多個物件，IsTruncated 和 NextContinuationToken 將用於重複執行整份清單。

```
import {
    S3Client,
    S3ServiceException,
    // This command supersedes the ListObjectsCommand and is the recommended way to
    list objects.
    paginateListObjectsV2,
```

```

} from "@aws-sdk/client-s3";

/**
 * Log all of the object keys in a bucket.
 * @param {{ bucketName: string, pageSize: string }}
 */
export const main = async ({ bucketName, pageSize }) => {
  const client = new S3Client({});
  /** @type {string[][]} */
  const objects = [];
  try {
    const paginator = paginateListObjectsV2(
      { client, /* Max items per page */ pageSize: Number.parseInt(pageSize) },
      { Bucket: bucketName },
    );

    for await (const page of paginator) {
      objects.push(page.Contents.map((o) => o.Key));
    }
    objects.forEach((objectList, pageNum) => {
      console.log(
        `Page ${pageNum + 1}\n-----\n${objectList.map((o) => `•
${o}`)}.join("\n")}\n`,
      );
    });
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
      console.error(
        `Error from S3 while listing objects for "${bucketName}". The bucket
doesn't exist.`,
      );
    } else if (caught instanceof S3ServiceException) {
      console.error(
        `Error from S3 while listing objects for "${bucketName}".
${caught.name}: ${caught.message}`,
      );
    } else {
      throw caught;
    }
  }
};

```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun listBucketObjects(bucketName: String) {
    val request =
        ListObjectsRequest {
            bucket = bucketName
        }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        val response = s3.listObjects(request)
        response.contents?.forEach { myObject ->
            println("The name of the key is ${myObject.key}")
            println("The object is ${myObject.size?.let { calKb(it) }} KBs")
            println("The owner is ${myObject.owner}")
        }
    }
}

private fun calKb(intValue: Long): Long = intValue / 1024
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

列出儲存貯體中的物件。

```
$s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

try {
    $contents = $this->s3client->listObjectsV2([
        'Bucket' => $this->bucketName,
    ]);
    echo "The contents of your bucket are: \n";
    foreach ($contents['Contents'] as $content) {
        echo $content['Key'] . "\n";
    }
} catch (Exception $exception) {
    echo "Failed to list objects in $this->bucketName with error: " .
    $exception->getMessage();
    exit("Please fix error with listing objects before continuing.");
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
        in Boto3
                                that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key

    @staticmethod
    def list(bucket, prefix=None):
        """
        Lists the objects in a bucket, optionally filtered by a prefix.

        :param bucket: The bucket to query. This is a Boto3 Bucket resource.
        :param prefix: When specified, only objects that start with this prefix
        are listed.
        :return: The list of objects.
        """
        try:
            if not prefix:
                objects = list(bucket.objects.all())
            else:
                objects = list(bucket.objects.filter(Prefix=prefix))
            logger.info(
                "Got objects %s from bucket '%s'", [o.key for o in objects],
                bucket.name
            )
        except ClientError:
            logger.exception("Couldn't get objects for bucket '%s'.",
                bucket.name)
            raise
        else:
            return objects

```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-s3'

# Wraps Amazon S3 bucket actions.
class BucketListObjectsWrapper
  attr_reader :bucket

  # @param bucket [Aws::S3::Bucket] An existing Amazon S3 bucket.
  def initialize(bucket)
    @bucket = bucket
  end

  # Lists object in a bucket.
  #
  # @param max_objects [Integer] The maximum number of objects to list.
  # @return [Integer] The number of objects listed.
  def list_objects(max_objects)
    count = 0
    puts "The objects in #{@bucket.name} are:"
    @bucket.objects.each do |obj|
      puts "\t#{obj.key}"
      count += 1
      break if count == max_objects
    end
    count
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't list objects in bucket #{bucket.name}. Here's why:
#{e.message}"
    0
  end
end

# Example usage:
```

```
def run_demo
  bucket_name = "amzn-s3-demo-bucket"

  wrapper = BucketListObjectsWrapper.new(Aws::S3::Bucket.new(bucket_name))
  count = wrapper.list_objects(25)
  puts "Listed #{count} objects."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
pub async fn list_objects(client: &aws_sdk_s3::Client, bucket: &str) ->
  Result<(), S3ExampleError> {
  let mut response = client
    .list_objects_v2()
    .bucket(bucket.to_owned())
    .max_keys(10) // In this example, go 10 at a time.
    .into_paginator()
    .send();

  while let Some(result) = response.next().await {
    match result {
      Ok(output) => {
        for object in output.contents() {
          println!(" - {}", object.key().unwrap_or("Unknown"));
        }
      }
      Err(err) => {
        eprintln!("{err:?}")
      }
    }
  }
}
```

```
    }  
  }  
  
  Ok::<(),  
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3->listobjectsv2(          " oo_result is returned for  
testing purposes. "  
    iv_bucket = iv_bucket_name ).  
    MESSAGE 'Retrieved list of objects in S3 bucket.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 SAP ABAP 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import AWSS3

public func listBucketFiles(bucket: String) async throws -> [String] {
    do {
        let input = ListObjectsV2Input(
            bucket: bucket
        )

        // Use "Paginated" to get all the objects.
        // This lets the SDK handle the 'continuationToken' in
        "ListObjectsV2Output".
        let output = client.listObjectsV2Paginated(input: input)
        var names: [String] = []

        for try await page in output {
            guard let objList = page.contents else {
                print("ERROR: listObjectsV2Paginated returned nil contents.")
                continue
            }

            for obj in objList {
                if let objName = obj.key {
                    names.append(objName)
                }
            }
        }

        return names
    }
    catch {
        print("ERROR: ", dump(error, name: "Listing objects."))
    }
}
```

```
        throw error;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Swift 的 AWS SDK API 參考》中的 [ListObjectsV2](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketAccelerateConfiguration 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketAccelerateConfiguration。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// Amazon Simple Storage Service (Amazon S3) Transfer Acceleration is a
/// bucket-level feature that enables you to perform faster data transfers
/// to Amazon S3. This example shows how to configure Transfer
/// Acceleration.
/// </summary>
public class TransferAcceleration
{
    /// <summary>
    /// The main method initializes the client object and sets the
    /// Amazon Simple Storage Service (Amazon S3) bucket name before
    /// calling EnableAccelerationAsync.
    /// </summary>
```

```
public static async Task Main()
{
    var s3Client = new AmazonS3Client();
    const string bucketName = "amzn-s3-demo-bucket";

    await EnableAccelerationAsync(s3Client, bucketName);
}

/// <summary>
/// This method sets the configuration to enable transfer acceleration
/// for the bucket referred to in the bucketName parameter.
/// </summary>
/// <param name="client">An Amazon S3 client used to enable the
/// acceleration on an Amazon S3 bucket.</param>
/// <param name="bucketName">The name of the Amazon S3 bucket for which
the
/// method will be enabling acceleration.</param>
private static async Task EnableAccelerationAsync(AmazonS3Client client,
string bucketName)
{
    try
    {
        var putRequest = new PutBucketAccelerateConfigurationRequest
        {
            BucketName = bucketName,
            AccelerateConfiguration = new AccelerateConfiguration
            {
                Status = BucketAccelerateStatus.Enabled,
            },
        };
        await client.PutBucketAccelerateConfigurationAsync(putRequest);

        var getRequest = new GetBucketAccelerateConfigurationRequest
        {
            BucketName = bucketName,
        };
        var response = await
client.GetBucketAccelerateConfigurationAsync(getRequest);

        Console.WriteLine($"Acceleration state = '{response.Status}' ");
    }
    catch (AmazonS3Exception ex)
    {

```

```
        Console.WriteLine($"Error occurred. Message: '{ex.Message}' when  
        setting transfer acceleration");  
    }  
}  
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutBucketAccelerateConfiguration](#)。

CLI

AWS CLI

設定儲存貯體的加速組態

下列 put-bucket-accelerate-configuration 範例會啟用指定儲存貯體的加速組態。

```
aws s3api put-bucket-accelerate-configuration \  
    --bucket amzn-s3-demo-bucket \  
    --accelerate-configuration Status=Enabled
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketAccelerateConfiguration](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會使給定 S3 儲存貯體的傳輸加速。

```
$statusVal = New-Object Amazon.S3.BucketAccelerateStatus('Enabled')  
Write-S3BucketAccelerateConfiguration -BucketName 'amzn-s3-demo-bucket' -  
AccelerateConfiguration_Status $statusVal
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [PutBucketAccelerateConfiguration](#)。

Tools for PowerShell V5

範例 1：此命令會使給定 S3 儲存貯體的傳輸加速。

```
$statusVal = New-Object Amazon.S3.BucketAccelerateStatus('Enabled')
Write-S3BucketAccelerateConfiguration -BucketName 'amzn-s3-demo-bucket' -
AccelerateConfiguration_Status $statusVal
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [PutBucketAccelerateConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketAcl 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketAcl。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [管理存取控制清單 \(ACL\)](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Creates an Amazon S3 bucket with an ACL to control access to the
/// bucket and the objects stored in it.
/// </summary>
/// <param name="client">The initialized client object used to create
```

```
/// an Amazon S3 bucket, with an ACL applied to the bucket.
/// </param>
/// <param name="region">The AWS Region where the bucket will be
created.</param>
/// <param name="newBucketName">The name of the bucket to create.</param>
/// <returns>A boolean value indicating success or failure.</returns>
public static async Task<bool> CreateBucketUseCannedACLAsync(IAmazonS3
client, S3Region region, string newBucketName)
{
    try
    {
        // Create a new Amazon S3 bucket with Canned ACL.
        var putBucketRequest = new PutBucketRequest()
        {
            BucketName = newBucketName,
            BucketRegion = region,
            CannedACL = S3CannedACL.LogDeliveryWrite,
        };

        PutBucketResponse putBucketResponse = await
client.PutBucketAsync(putBucketRequest);

        return putBucketResponse.HttpStatusCode ==
System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Amazon S3 error: {ex.Message}");
    }

    return false;
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutBucketAcl](#)。

C++

適用於 C++ 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::putBucketAcl(const Aws::String &bucketName, const Aws::String
    &ownerID,
                                const Aws::String &granteePermission,
                                const Aws::String &granteeType, const Aws::String
    &granteeID,
                                const Aws::String &granteeEmailAddress,
                                const Aws::String &granteeURI, const
    Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::Owner owner;
    owner.SetID(ownerID);

    Aws::S3::Model::Grantee grantee;
    grantee.SetType(setGranteeType(granteeType));

    if (!granteeEmailAddress.empty()) {
        grantee.SetEmailAddress(granteeEmailAddress);
    }

    if (!granteeID.empty()) {
        grantee.SetID(granteeID);
    }

    if (!granteeURI.empty()) {
        grantee.SetURI(granteeURI);
    }

    Aws::S3::Model::Grant grant;
    grant.SetGrantee(grantee);
    grant.SetPermission(setGranteePermission(granteePermission));
}
```

```

    Aws::Vector<Aws::S3::Model::Grant> grants;
    grants.push_back(grant);

    Aws::S3::Model::AccessControlPolicy acp;
    acp.SetOwner(owner);
    acp.SetGrants(grants);

    Aws::S3::Model::PutBucketAclRequest request;
    request.SetAccessControlPolicy(acp);
    request.SetBucket(bucketName);

    Aws::S3::Model::PutBucketAclOutcome outcome =
        s3Client.PutBucketAcl(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &error = outcome.GetError();

        std::cerr << "Error: putBucketAcl: " << error.GetExceptionName()
            << " - " << error.GetMessage() << std::endl;
    } else {
        std::cout << "Successfully added an ACL to the bucket '" << bucketName
            << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which converts a human-readable string to a built-in type
    enumeration.
/*!
    \param access: Human readable string.
    \return Permission: A Permission enum.
    */

Aws::S3::Model::Permission setGranteePermission(const Aws::String &access) {
    if (access == "FULL_CONTROL")
        return Aws::S3::Model::Permission::FULL_CONTROL;
    if (access == "WRITE")
        return Aws::S3::Model::Permission::WRITE;
    if (access == "READ")
        return Aws::S3::Model::Permission::READ;
    if (access == "WRITE_ACP")
        return Aws::S3::Model::Permission::WRITE_ACP;
    if (access == "READ_ACP")

```



```

        return Aws::S3::Model::Permission::READ_ACP;
    return Aws::S3::Model::Permission::NOT_SET;
}

//! Routine which converts a human-readable string to a built-in type
enumeration.
/*!
 \param type: Human readable string.
 \return Type: Type enumeration
 */

Aws::S3::Model::Type setGranteeType(const Aws::String &type) {
    if (type == "Amazon customer by email")
        return Aws::S3::Model::Type::AmazonCustomerByEmail;
    if (type == "Canonical user")
        return Aws::S3::Model::Type::CanonicalUser;
    if (type == "Group")
        return Aws::S3::Model::Type::Group;
    return Aws::S3::Model::Type::NOT_SET;
}

```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [PutBucketAcl](#)。

CLI

AWS CLI

此範例full control授予兩個 AWS 使用者 (user1@example.com 和 user2@example.com) 和read許可給每個人：

```
aws s3api put-bucket-acl --bucket amzn-s3-demo-bucket --grant-full-
control emailaddress=user1@example.com,emailaddress=user2@example.com --grant-
read uri=http://acs.amazonaws.com/groups/global/AllUsers
```

如需自訂 ACL (s3api ACL 命令如 put-bucket-acl，使用相同的速記引數表示法)，請參閱 <http://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTacl.html>。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketAcl](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.AccessControlPolicy;
import software.amazon.awssdk.services.s3.model.Grant;
import software.amazon.awssdk.services.s3.model.Permission;
import software.amazon.awssdk.services.s3.model.PutBucketAclRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.Type;

import java.util.ArrayList;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SetAcl {
    public static void main(String[] args) {
        final String usage = ""

        Usage:
            <bucketName> <id>\s

        Where:
            bucketName - The Amazon S3 bucket to grant permissions on.\s
    }
```

```

        id - The ID of the owner of this bucket (you can get this value
from the AWS Management Console).
        """;

    if (args.length != 2) {
        System.out.println(usage);
        return;
    }

    String bucketName = args[0];
    String id = args[1];
    System.out.format("Setting access \n");
    System.out.println(" in bucket: " + bucketName);
    Region region = Region.US_EAST_1;
    S3Client s3 = S3Client.builder()
        .region(region)
        .build();

    setBucketAcl(s3, bucketName, id);
    System.out.println("Done!");
    s3.close();
}

/**
 * Sets the Access Control List (ACL) for an Amazon S3 bucket.
 *
 * @param s3 the S3Client instance to be used for the operation
 * @param bucketName the name of the S3 bucket to set the ACL for
 * @param id the ID of the AWS user or account that will be granted full
control of the bucket
 * @throws S3Exception if an error occurs while setting the bucket ACL
 */
public static void setBucketAcl(S3Client s3, String bucketName, String id) {
    try {
        Grant ownerGrant = Grant.builder()
            .grantee(builder -> builder.id(id)
                .type(Type.CANONICAL_USER))
            .permission(Permission.FULL_CONTROL)
            .build();

        List<Grant> grantList2 = new ArrayList<>();
        grantList2.add(ownerGrant);

        AccessControlPolicy acl = AccessControlPolicy.builder()

```

```
        .owner(builder -> builder.id(id))
        .grants(grantList2)
        .build();

    PutBucketAclRequest putAclReq = PutBucketAclRequest.builder()
        .bucket(bucketName)
        .accessControlPolicy(acl)
        .build();

    s3.putBucketAcl(putAclReq);

    } catch (S3Exception e) {
        e.printStackTrace();
        System.exit(1);
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutBucketAcl](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

放置儲存貯體的 ACL。

```
import {
    PutBucketAclCommand,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Grant read access to a user using their canonical AWS account ID.
 */
```

```

* Most Amazon S3 use cases don't require the use of access control lists (ACLs).
* We recommend that you disable ACLs, except in unusual circumstances where
* you need to control access for each object individually. Consider a policy
  instead.
* For more information see https://docs.aws.amazon.com/AmazonS3/latest/
userguide/bucket-policies.html.
* @param {{ bucketName: string, granteeCanonicalUserId: string,
  ownerCanonicalUserId }}
*/
export const main = async ({
  bucketName,
  granteeCanonicalUserId,
  ownerCanonicalUserId,
}) => {
  const client = new S3Client({});
  const command = new PutBucketAclCommand({
    Bucket: bucketName,
    AccessControlPolicy: {
      Grants: [
        {
          Grantee: {
            // The canonical ID of the user. This ID is an obfuscated form of
            your AWS account number.
            // It's unique to Amazon S3 and can't be found elsewhere.
            // For more information, see https://docs.aws.amazon.com/AmazonS3/
latest/userguide/finding-canonical-user-id.html.
            ID: granteeCanonicalUserId,
            Type: "CanonicalUser",
          },
          // One of FULL_CONTROL | READ | WRITE | READ_ACP | WRITE_ACP
          // https://docs.aws.amazon.com/AmazonS3/latest/API/
API\_Grant.html#AmazonS3-Type-Grant-Permission
          Permission: "READ",
        },
      ],
      Owner: {
        ID: ownerCanonicalUserId,
      },
    },
  });

  try {
    await client.send(command);
    console.log(`Granted READ access to ${bucketName}`);
  }

```

```
    } catch (caught) {
        if (
            caught instanceof S3ServiceException &&
            caught.name === "NoSuchBucket"
        ) {
            console.error(
                `Error from S3 while setting ACL for bucket ${bucketName}. The bucket
                doesn't exist.`
            );
        } else if (caught instanceof S3ServiceException) {
            console.error(
                `Error from S3 while setting ACL for bucket ${bucketName}.
                ${caught.name}: ${caught.message}`
            );
        } else {
            throw caught;
        }
    }
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [PutBucketAcl](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun setBucketAcl(
    bucketName: String,
    idVal: String,
) {
    val myGrant =
        Grantee {
```

```
        id = idVal
        type = Type.CanonicalUser
    }

    val ownerGrant =
        Grant {
            grantee = myGrant
            permission = Permission.FullControl
        }

    val grantList = mutableListOf<Grant>()
    grantList.add(ownerGrant)

    val ownerOb =
        Owner {
            id = idVal
        }

    val acl =
        AccessControlPolicy {
            owner = ownerOb
            grants = grantList
        }

    val request =
        PutBucketAclRequest {
            bucket = bucketName
            accessControlPolicy = acl
        }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        s3.putBucketAcl(request)
        println("An ACL was successfully set on $bucketName")
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [PutBucketAcl](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def grant_log_delivery_access(self):
        """
        Grant the AWS Log Delivery group write access to the bucket so that
        Amazon S3 can deliver access logs to the bucket. This is the only
        recommended
        use of an S3 bucket ACL.
        """
        try:
            acl = self.bucket.Acl()
            # Putting an ACL overwrites the existing ACL. If you want to preserve
            # existing grants, append new grants to the list of existing grants.
            grants = acl.grants if acl.grants else []
            grants.append(
                {
                    "Grantee": {
                        "Type": "Group",
                        "URI": "http://acs.amazonaws.com/groups/s3/LogDelivery",
                    },
                    "Permission": "WRITE",
                }
            )
```



```

        }
    )
    acl.put(AccessControlPolicy={"Grants": grants, "Owner": acl.owner})
    logger.info("Granted log delivery access to bucket '%s'",
self.bucket.name)
    except ClientError:
        logger.exception("Couldn't add ACL to bucket '%s'.",
self.bucket.name)
        raise

```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [PutBucketAcl](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " Example: Grant log delivery access to a bucket
    " iv_grantwrite = 'uri=http://acs.amazonaws.com/groups/s3/LogDelivery'
    lo_s3->putbucketacl(
        iv_bucket = iv_bucket_name
        iv_grantwrite = iv_grantwrite ).
    MESSAGE 'Bucket ACL updated.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutBucketAcl](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketCors 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketCors。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Add CORS configuration to the Amazon S3 bucket.
/// </summary>
/// <param name="client">The initialized Amazon S3 client object used
/// to apply the CORS configuration to an Amazon S3 bucket.</param>
/// <param name="configuration">The CORS configuration to apply.</param>
private static async Task PutCORSConfigurationAsync(AmazonS3Client
client, CORSConfiguration configuration)
{
    PutCORSConfigurationRequest request = new
PutCORSConfigurationRequest()
    {
        BucketName = BucketName,
        Configuration = configuration,
    };

    _ = await client.PutCORSConfigurationAsync(request);
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutBucketCors](#)。

CLI

AWS CLI

下列範例會啟用來自 `www.example.com` 的 PUT、POST 和 DELETE 請求，並啟用來自任何網域的 GET 請求：

```
aws s3api put-bucket-cors --bucket amzn-s3-demo-bucket --cors-configuration file://cors.json
```

cors.json:

```
{
  "CORSRules": [
    {
      "AllowedOrigins": ["http://www.example.com"],
      "AllowedHeaders": ["*"],
      "AllowedMethods": ["PUT", "POST", "DELETE"],
      "MaxAgeSeconds": 3000,
      "ExposeHeaders": ["x-amz-server-side-encryption"]
    },
    {
      "AllowedOrigins": ["*"],
      "AllowedHeaders": ["Authorization"],
      "AllowedMethods": ["GET"],
      "MaxAgeSeconds": 3000
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketCors](#)。

Java

SDK for Java 2.x

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;

import java.util.ArrayList;
import java.util.List;

import software.amazon.awssdk.services.s3.model.GetBucketCorsRequest;
import software.amazon.awssdk.services.s3.model.GetBucketCorsResponse;
import software.amazon.awssdk.services.s3.model.DeleteBucketCorsRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.CORSRule;
import software.amazon.awssdk.services.s3.model.CORSConfiguration;
import software.amazon.awssdk.services.s3.model.PutBucketCorsRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class S3Cors {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <bucketName> <accountId>\s

            Where:
                bucketName - The Amazon S3 bucket to upload an object into.
                accountId - The id of the account that owns the Amazon S3 bucket.
            "";

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String accountId = args[1];
```

```
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        setCorsInformation(s3, bucketName, accountId);
        getBucketCorsInformation(s3, bucketName, accountId);
        deleteBucketCorsInformation(s3, bucketName, accountId);
        s3.close();
    }

    /**
     * Deletes the CORS (Cross-Origin Resource Sharing) configuration for an
     * Amazon S3 bucket.
     *
     * @param s3          the {@link S3Client} instance used to interact with
     *                    the Amazon S3 service
     * @param bucketName  the name of the Amazon S3 bucket for which the CORS
     *                    configuration should be deleted
     * @param accountId   the expected AWS account ID of the bucket owner
     *
     * @throws S3Exception if an error occurs while deleting the CORS
     *                    configuration for the bucket
     */
    public static void deleteBucketCorsInformation(S3Client s3, String
        bucketName, String accountId) {
        try {
            DeleteBucketCorsRequest bucketCorsRequest =
                DeleteBucketCorsRequest.builder()
                    .bucket(bucketName)
                    .expectedBucketOwner(accountId)
                    .build();

            s3.deleteBucketCors(bucketCorsRequest);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    /**
     * Retrieves the CORS (Cross-Origin Resource Sharing) configuration for the
     * specified S3 bucket.
```

```

    *
    * @param s3 the S3Client instance to use for the operation
    * @param bucketName the name of the S3 bucket to retrieve the CORS
configuration for
    * @param accountId the expected bucket owner's account ID
    *
    * @throws S3Exception if there is an error retrieving the CORS configuration
    */
    public static void getBucketCorsInformation(S3Client s3, String bucketName,
String accountId) {
        try {
            GetBucketCorsRequest bucketCorsRequest =
GetBucketCorsRequest.builder()
                .bucket(bucketName)
                .expectedBucketOwner(accountId)
                .build();

            GetBucketCorsResponse corsResponse =
s3.getBucketCors(bucketCorsRequest);
            List<CORSRule> corsRules = corsResponse.corsRules();
            for (CORSRule rule : corsRules) {
                System.out.println("allowOrigins: " + rule.allowedOrigins());
                System.out.println("AllowedMethod: " + rule.allowedMethods());
            }

        } catch (S3Exception e) {

            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    /**
    * Sets the Cross-Origin Resource Sharing (CORS) rules for an Amazon S3
bucket.
    *
    * @param s3 The S3Client object used to interact with the Amazon S3 service.
    * @param bucketName The name of the S3 bucket to set the CORS rules for.
    * @param accountId The AWS account ID of the bucket owner.
    */
    public static void setCorsInformation(S3Client s3, String bucketName, String
accountId) {
        List<String> allowMethods = new ArrayList<>();
        allowMethods.add("PUT");
    }

```

```
allowMethods.add("POST");
allowMethods.add("DELETE");

List<String> allowOrigins = new ArrayList<>();
allowOrigins.add("http://example.com");
try {
    // Define CORS rules.
    CORSRule corsRule = CORSRule.builder()
        .allowedMethods(allowMethods)
        .allowedOrigins(allowOrigins)
        .build();

    List<CORSRule> corsRules = new ArrayList<>();
    corsRules.add(corsRule);
    CORSConfiguration configuration = CORSConfiguration.builder()
        .corsRules(corsRules)
        .build();

    PutBucketCorsRequest putBucketCorsRequest =
PutBucketCorsRequest.builder()
        .bucket(bucketName)
        .corsConfiguration(configuration)
        .expectedBucketOwner(accountId)
        .build();

    s3.putBucketCors(putBucketCorsRequest);

} catch (S3Exception e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutBucketCors](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

新增 CORS 規則。

```
import {
  PutBucketCorsCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Allows cross-origin requests to an S3 bucket by setting the CORS
 * configuration.
 * @param {{ bucketName: string }}
 */
export const main = async ({ bucketName }) => {
  const client = new S3Client({});

  try {
    await client.send(
      new PutBucketCorsCommand({
        Bucket: bucketName,
        CORSConfiguration: {
          CORSRules: [
            {
              // Allow all headers to be sent to this bucket.
              AllowedHeaders: ["*"],
              // Allow only GET and PUT methods to be sent to this bucket.
              AllowedMethods: ["GET", "PUT"],
              // Allow only requests from the specified origin.
              AllowedOrigins: ["https://www.example.com"],
              // Allow the entity tag (ETag) header to be returned in the
              // response. The ETag header
              // The entity tag represents a specific version of the object. The
              // ETag reflects
            }
          ]
        }
      })
    );
  } catch (err) {
    if (err instanceof S3ServiceException) {
      console.log("S3ServiceException: ", err.message);
    } else {
      console.log("Unknown error: ", err);
    }
  }
}
```



```
        // changes only to the contents of an object, not its metadata.
        ExposeHeaders: ["ETag"],
        // How long the requesting browser should cache the preflight
response. After
        // this time, the preflight request will have to be made again.
        MaxAgeSeconds: 3600,
    },
],
},
)),
);
console.log(`Successfully set CORS rules for bucket: ${bucketName}`);
} catch (caught) {
    if (
        caught instanceof S3ServiceException &&
        caught.name === "NoSuchBucket"
    ) {
        console.error(
            `Error from S3 while setting CORS rules for ${bucketName}. The bucket
doesn't exist.`
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while setting CORS rules for ${bucketName}.
${caught.name}: ${caught.message}`
        );
    } else {
        throw caught;
    }
}
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [PutBucketCors](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def put_cors(self, cors_rules):
        """
        Apply CORS rules to the bucket. CORS rules specify the HTTP actions that
        are
        allowed from other domains.

        :param cors_rules: The CORS rules to apply.
        """
        try:
            self.bucket.Cors().put(CORSConfiguration={"CORSRules": cors_rules})
            logger.info(
                "Put CORS rules %s for bucket '%s'.", cors_rules,
                self.bucket.name
            )
        except ClientError:
            logger.exception("Couldn't put CORS rules for bucket %s.",
                self.bucket.name)
            raise
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [PutBucketCors](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-s3'

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors

  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with
  # an existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Sets CORS rules on a bucket.
  #
  # @param allowed_methods [Array<String>] The types of HTTP requests to allow.
  # @param allowed_origins [Array<String>] The origins to allow.
  # @returns [Boolean] True if the CORS rules were set; otherwise, false.
  def set_cors(allowed_methods, allowed_origins)
    @bucket_cors.put(
      cors_configuration: {
        cors_rules: [
          {
            allowed_methods: allowed_methods,
            allowed_origins: allowed_origins,
            allowed_headers: %w[*],
            max_age_seconds: 3600
          }
        ]
      }
    )
  end
end
```

```

        }
      ]
    }
  )
  true
rescue Aws::Errors::ServiceError => e
  puts "Couldn't set CORS rules for #{@bucket_cors.bucket.name}. Here's why:
#{e.message}"
  false
end
end

```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [PutBucketCors](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
  " Example: Allow PUT, POST, DELETE methods from http://www.example.com
  lo_s3->putbucketcors(
    iv_bucket = iv_bucket_name
    io_corsconfiguration = NEW /aws1/cl_s3_corsconfiguration(
      it_corsrules = it_cors_rules ) ).
  MESSAGE 'Bucket CORS configuration set.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
  MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutBucketCors](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketEncryption 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketEncryption。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [物件儲存入門](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Set the bucket server side encryption to use AWSKMS with a customer-
managed key id.
/// </summary>
/// <param name="bucketName">Name of the bucket.</param>
/// <param name="kmsKeyId">The Id of the KMS Key.</param>
/// <returns>True if successful.</returns>
public static async Task<bool> SetBucketServerSideEncryption(string
bucketName, string kmsKeyId)
{
    var serverSideEncryptionByDefault = new ServerSideEncryptionConfiguration
    {
        ServerSideEncryptionRules = new List<ServerSideEncryptionRule>
        {
            new ServerSideEncryptionRule
            {
                ServerSideEncryptionByDefault = new
ServerSideEncryptionByDefault
                {
```

```
        ServerSideEncryptionAlgorithm =
ServerSideEncryptionMethod.AWSKMS,
        ServerSideEncryptionKeyManagementServiceKeyId = kmsKeyId
    }
}
};
try
{
    var encryptionResponse = await _s3Client.PutBucketEncryptionAsync(new
PutBucketEncryptionRequest
    {
        BucketName = bucketName,
        ServerSideEncryptionConfiguration =
serverSideEncryptionByDefault,
    });

    return encryptionResponse.HttpStatusCode == HttpStatusCode.OK;
}
catch (AmazonS3Exception ex)
{
    Console.WriteLine(ex.ErrorCode == "AccessDenied"
        ? $"This account does not have permission to set encryption on
{bucketName}, please try again."
        : $"Unable to set bucket encryption for bucket {bucketName},
{ex.Message}");
}
return false;
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutBucketEncryption](#)。

CLI

AWS CLI

針對儲存貯體設定伺服器端加密組態

下列 put-bucket-encryption 範例會將 AES256 加密設定為指定儲存貯體的預設值。

```
aws s3api put-bucket-encryption \
```

```
--bucket amzn-s3-demo-bucket \  
--server-side-encryption-configuration '{"Rules":  
[{"ApplyServerSideEncryptionByDefault": {"SSEAlgorithm": "AES256"}}]}'
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketEncryption](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會在給定的儲存貯體上使用 Amazon S3 受管金鑰 (SSE-S3)，以啟用預設 AES256 伺服器端加密。

```
$Encryptionconfig = @{ServerSideEncryptionByDefault =  
    @{ServerSideEncryptionAlgorithm = "AES256"}}  
Set-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket' -  
ServerSideEncryptionConfiguration_ServerSideEncryptionRule $Encryptionconfig
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [PutBucketEncryption](#)。

Tools for PowerShell V5

範例 1：此命令會在給定的儲存貯體上使用 Amazon S3 受管金鑰 (SSE-S3)，以啟用預設 AES256 伺服器端加密。

```
$Encryptionconfig = @{ServerSideEncryptionByDefault =  
    @{ServerSideEncryptionAlgorithm = "AES256"}}  
Set-S3BucketEncryption -BucketName 'amzn-s3-demo-bucket' -  
ServerSideEncryptionConfiguration_ServerSideEncryptionRule $Encryptionconfig
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [PutBucketEncryption](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketLifecycleConfiguration 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketLifecycleConfiguration。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [刪除未完成的分段上傳](#)
- [使用 S3 管理大型訊息](#)
- [使用版本化物件](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Adds lifecycle configuration information to the S3 bucket named in
/// the bucketName parameter.
/// </summary>
/// <param name="client">The S3 client used to call the
/// PutLifecycleConfigurationAsync method.</param>
/// <param name="bucketName">A string representing the S3 bucket to
/// which configuration information will be added.</param>
/// <param name="configuration">A LifecycleConfiguration object that
/// will be applied to the S3 bucket.</param>
public static async Task AddExampleLifecycleConfigAsync(IAmazonS3 client,
string bucketName, LifecycleConfiguration configuration)
{
    var request = new PutLifecycleConfigurationRequest()
    {
        BucketName = bucketName,
        Configuration = configuration,
    };
    var response = await client.PutLifecycleConfigurationAsync(request);
}
```


- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutBucketLifecycleConfiguration](#)。

CLI

AWS CLI

下列命令會將生命週期組態套用至名為 `amzn-s3-demo-bucket` 的儲存貯體：

```
aws s3api put-bucket-lifecycle-configuration --bucket amzn-s3-demo-bucket --  
lifecycle-configuration file://lifecycle.json
```

檔案 `lifecycle.json` 是目前資料夾中的 JSON 文件，其中指定兩個規則：

```
{  
  "Rules": [  
    {  
      "ID": "Move rotated logs to Glacier",  
      "Prefix": "rotated/",  
      "Status": "Enabled",  
      "Transitions": [  
        {  
          "Date": "2015-11-10T00:00:00.000Z",  
          "StorageClass": "GLACIER"  
        }  
      ]  
    },  
    {  
      "Status": "Enabled",  
      "Prefix": "",  
      "NoncurrentVersionTransitions": [  
        {  
          "NoncurrentDays": 2,  
          "StorageClass": "GLACIER"  
        }  
      ],  
      "ID": "Move old versions to Glacier"  
    }  
  ]  
}
```

第一個規則會在指定的日期，將字首為 `rotated` 的檔案移至 Glacier。當舊物件版本不再是目前版本時，第二個規則會將舊物件版本移至 Glacier。如需可接受時間戳記格式的資訊，請參閱《AWS CLI 使用者指南》中的「指定參數值」。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketLifecycleConfiguration](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.LifecycleRuleFilter;
import software.amazon.awssdk.services.s3.model.Transition;
import
    software.amazon.awssdk.services.s3.model.GetBucketLifecycleConfigurationRequest;
import
    software.amazon.awssdk.services.s3.model.GetBucketLifecycleConfigurationResponse;
import software.amazon.awssdk.services.s3.model.DeleteBucketLifecycleRequest;
import software.amazon.awssdk.services.s3.model.TransitionStorageClass;
import software.amazon.awssdk.services.s3.model.LifecycleRule;
import software.amazon.awssdk.services.s3.model.ExpirationStatus;
import software.amazon.awssdk.services.s3.model.BucketLifecycleConfiguration;
import
    software.amazon.awssdk.services.s3.model.PutBucketLifecycleConfigurationRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.ArrayList;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
```

```
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
*/

public class LifecycleConfiguration {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <bucketName> <accountId>\s

            Where:
                bucketName - The Amazon Simple Storage Service (Amazon S3) bucket
to upload an object into.
                accountId - The id of the account that owns the Amazon S3 bucket.
            "";

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String accountId = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        setLifecycleConfig(s3, bucketName, accountId);
        getLifecycleConfig(s3, bucketName, accountId);
        deleteLifecycleConfig(s3, bucketName, accountId);
        System.out.println("You have successfully created, updated, and deleted a
Lifecycle configuration");
        s3.close();
    }

    /**
     * Sets the lifecycle configuration for an Amazon S3 bucket.
     *
     * @param s3          The Amazon S3 client to use for the operation.
     * @param bucketName  The name of the Amazon S3 bucket.
     * @param accountId   The expected owner of the Amazon S3 bucket.
     */
}
```

```
    * @throws S3Exception if there is an error setting the lifecycle
configuration.
    */
    public static void setLifecycleConfig(S3Client s3, String bucketName, String
accountId) {
        try {
            // Create a rule to archive objects with the "glacierobjects/" prefix
to the
            // S3 Glacier Flexible Retrieval storage class immediately.
            LifecycleRuleFilter ruleFilter = LifecycleRuleFilter.builder()
                .prefix("glacierobjects/")
                .build();

            Transition transition = Transition.builder()
                .storageClass(TransitionStorageClass.GLACIER)
                .days(0)
                .build();

            LifecycleRule rule1 = LifecycleRule.builder()
                .id("Archive immediately rule")
                .filter(ruleFilter)
                .transitions(transition)
                .status(ExpirationStatus.ENABLED)
                .build();

            // Create a second rule.
            Transition transition2 = Transition.builder()
                .storageClass(TransitionStorageClass.GLACIER)
                .days(0)
                .build();

            List<Transition> transitionList = new ArrayList<>();
            transitionList.add(transition2);

            LifecycleRuleFilter ruleFilter2 = LifecycleRuleFilter.builder()
                .prefix("glacierobjects/")
                .build();

            LifecycleRule rule2 = LifecycleRule.builder()
                .id("Archive and then delete rule")
                .filter(ruleFilter2)
                .transitions(transitionList)
                .status(ExpirationStatus.ENABLED)
                .build();
```

```
// Add the LifecycleRule objects to an ArrayList.
ArrayList<LifecycleRule> ruleList = new ArrayList<>();
ruleList.add(rule1);
ruleList.add(rule2);

BucketLifecycleConfiguration lifecycleConfiguration =
BucketLifecycleConfiguration.builder()
    .rules(ruleList)
    .build();

PutBucketLifecycleConfigurationRequest
putBucketLifecycleConfigurationRequest = PutBucketLifecycleConfigurationRequest
    .builder()
    .bucket(bucketName)
    .lifecycleConfiguration(lifecycleConfiguration)
    .expectedBucketOwner(accountId)
    .build();

s3.putBucketLifecycleConfiguration(putBucketLifecycleConfigurationRequest);

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}

/**
 * Retrieves the lifecycle configuration for an Amazon S3 bucket and adds a
 * new lifecycle rule to it.
 *
 * @param s3 the S3Client instance used to interact with Amazon S3
 * @param bucketName the name of the Amazon S3 bucket
 * @param accountId the expected owner of the Amazon S3 bucket
 */
public static void getLifecycleConfig(S3Client s3, String bucketName, String
accountId) {
    try {
        GetBucketLifecycleConfigurationRequest
getBucketLifecycleConfigurationRequest = GetBucketLifecycleConfigurationRequest
            .builder()
            .bucket(bucketName)
            .expectedBucketOwner(accountId)
```

```
        .build());

    GetBucketLifecycleConfigurationResponse response = s3

    .getBucketLifecycleConfiguration(getBucketLifecycleConfigurationRequest);
    List<LifecycleRule> newList = new ArrayList<>();
    List<LifecycleRule> rules = response.rules();
    for (LifecycleRule rule : rules) {
        newList.add(rule);
    }

    // Add a new rule with both a prefix predicate and a tag predicate.
    LifecycleRuleFilter ruleFilter = LifecycleRuleFilter.builder()
        .prefix("YearlyDocuments/")
        .build();

    Transition transition = Transition.builder()
        .storageClass(TransitionStorageClass.GLACIER)
        .days(3650)
        .build();

    LifecycleRule rule1 = LifecycleRule.builder()
        .id("NewRule")
        .filter(ruleFilter)
        .transitions(transition)
        .status(ExpirationStatus.ENABLED)
        .build();

    // Add the new rule to the list.
    newList.add(rule1);
    BucketLifecycleConfiguration lifecycleConfiguration =
    BucketLifecycleConfiguration.builder()
        .rules(newList)
        .build();

    PutBucketLifecycleConfigurationRequest
    putBucketLifecycleConfigurationRequest = PutBucketLifecycleConfigurationRequest
        .builder()
        .bucket(bucketName)
        .lifecycleConfiguration(lifecycleConfiguration)
        .expectedBucketOwner(accountId)
        .build();
```

```
s3.putBucketLifecycleConfiguration(putBucketLifecycleConfigurationRequest);

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}

/**
 * Deletes the lifecycle configuration for an Amazon S3 bucket.
 *
 * @param s3 the {@link S3Client} to use for the operation
 * @param bucketName the name of the S3 bucket
 * @param accountId the expected account owner of the S3 bucket
 *
 * @throws S3Exception if an error occurs while deleting the lifecycle
 * configuration
 */
public static void deleteLifecycleConfig(S3Client s3, String bucketName,
String accountId) {
    try {
        DeleteBucketLifecycleRequest deleteBucketLifecycleRequest =
DeleteBucketLifecycleRequest
        .builder()
        .bucket(bucketName)
        .expectedBucketOwner(accountId)
        .build();

        s3.deleteBucketLifecycle(deleteBucketLifecycleRequest);

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutBucketLifecycleConfiguration](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def put_lifecycle_configuration(self, lifecycle_rules):
        """
        Apply a lifecycle configuration to the bucket. The lifecycle
        configuration can
        be used to archive or delete the objects in the bucket according to
        specified
        parameters, such as a number of days.

        :param lifecycle_rules: The lifecycle rules to apply.
        """
        try:
            self.bucket.LifecycleConfiguration().put(
                LifecycleConfiguration={"Rules": lifecycle_rules}
            )
            logger.info(
                "Put lifecycle rules %s for bucket '%s'.",
                lifecycle_rules,
                self.bucket.name,
            )
```



```
except ClientError:
    logger.exception(
        "Couldn't put lifecycle rules for bucket '%s'.", self.bucket.name
    )
    raise
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [PutBucketLifecycleConfiguration](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.
  " Example: Expire objects with prefix 'logs/' after 30 days
  lo_s3->putbucketlifecycleconf(
    iv_bucket = iv_bucket_name
    io_lifecycleconfiguration = NEW /aws1/cl_s3_bucketlcconf(
      it_rules = it_lifecycle_rule ) ).
  MESSAGE 'Bucket lifecycle configuration set.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
  MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutBucketLifecycleConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketLogging 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketLogging。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.IO;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Configuration;

/// <summary>
/// This example shows how to enable logging on an Amazon Simple Storage
/// Service (Amazon S3) bucket. You need to have two Amazon S3 buckets for
/// this example. The first is the bucket for which you wish to enable
/// logging, and the second is the location where you want to store the
/// logs.
/// </summary>
public class ServerAccessLogging
{
    private static IConfiguration _configuration = null!;

    public static async Task Main()
    {
        LoadConfig();

        string bucketName = _configuration["BucketName"];
        string logBucketName = _configuration["LogBucketName"];
        string logObjectKeyPrefix = _configuration["LogObjectKeyPrefix"];
        string accountId = _configuration["AccountId"];

        // If the AWS Region defined for your default user is different
        // from the Region where your Amazon S3 bucket is located,
```

```

        // pass the Region name to the Amazon S3 client object's constructor.
        // For example: RegionEndpoint.USWest2 or RegionEndpoint.USEast2.
        IAmazonS3 client = new AmazonS3Client();

        try
        {
            // Update bucket policy for target bucket to allow delivery of
logs to it.
            await SetBucketPolicyToAllowLogDelivery(
                client,
                bucketName,
                logBucketName,
                logObjectKeyPrefix,
                accountId);

            // Enable logging on the source bucket.
            await EnableLoggingAsync(
                client,
                bucketName,
                logBucketName,
                logObjectKeyPrefix);
        }
        catch (AmazonS3Exception e)
        {
            Console.WriteLine($"Error: {e.Message}");
        }
    }

    /// <summary>
    /// This method grants appropriate permissions for logging to the
    /// Amazon S3 bucket where the logs will be stored.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client which will be
used
    /// to apply the bucket policy.</param>
    /// <param name="sourceBucketName">The name of the source bucket.</param>
    /// <param name="logBucketName">The name of the bucket where logging
    /// information will be stored.</param>
    /// <param name="logPrefix">The logging prefix where the logs should be
delivered.</param>
    /// <param name="accountId">The account id of the account where the
source bucket exists.</param>
    /// <returns>Async task.</returns>
    public static async Task SetBucketPolicyToAllowLogDelivery(

```

```

        IAmazonS3 client,
        string sourceBucketName,
        string logBucketName,
        string logPrefix,
        string accountId)
    {
        var resourceArn = @""arn:aws:s3::" + logBucketName + "/" +
logPrefix + @"""";

        var newPolicy = @"{
            ""Statement"": [{
                ""Sid"": ""S3ServerAccessLogsPolicy"",
                ""Effect"": ""Allow"",
                ""Principal"": { ""Service"":
""logging.s3.amazonaws.com"" },
                ""Action"": [""s3:PutObject""],
                ""Resource"": ["" + resourceArn + @""],
                ""Condition"": {
                    ""ArnLike"": { ""aws:SourceArn"":
""arn:aws:s3::" + sourceBucketName + @"""" },
                    ""StringEquals"": { ""aws:SourceAccount"": """" +
accountId + @"""" }
                }
            }]
        }";

        Console.WriteLine($"The policy to apply to bucket {logBucketName} to
enable logging:");
        Console.WriteLine(newPolicy);

        PutBucketPolicyRequest putRequest = new PutBucketPolicyRequest
        {
            BucketName = logBucketName,
            Policy = newPolicy,
        };
        await client.PutBucketPolicyAsync(putRequest);
        Console.WriteLine("Policy applied.");
    }

    /// <summary>
    /// This method enables logging for an Amazon S3 bucket. Logs will be
stored
    /// in the bucket you selected for logging. Selected prefix
    /// will be prepended to each log object.
    /// </summary>

```

```
    /// <param name="client">The initialized Amazon S3 client which will be
used
    /// to configure and apply logging to the selected Amazon S3 bucket.</
param>
    /// <param name="bucketName">The name of the Amazon S3 bucket for which
you
    /// wish to enable logging.</param>
    /// <param name="logBucketName">The name of the Amazon S3 bucket where
logging
    /// information will be stored.</param>
    /// <param name="logObjectKeyPrefix">The prefix to prepend to each
    /// object key.</param>
    /// <returns>Async task.</returns>
    public static async Task EnableLoggingAsync(
        IAmazonS3 client,
        string bucketName,
        string logBucketName,
        string logObjectKeyPrefix)
    {
        Console.WriteLine($"Enabling logging for bucket {bucketName}.");
        var loggingConfig = new S3BucketLoggingConfig
        {
            TargetBucketName = logBucketName,
            TargetPrefix = logObjectKeyPrefix,
        };

        var putBucketLoggingRequest = new PutBucketLoggingRequest
        {
            BucketName = bucketName,
            LoggingConfig = loggingConfig,
        };
        await client.PutBucketLoggingAsync(putBucketLoggingRequest);
        Console.WriteLine($"Logging enabled.");
    }

    /// <summary>
    /// Loads configuration from settings files.
    /// </summary>
    public static void LoadConfig()
    {
        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
```

```
        .AddJsonFile("settings.local.json", true) // Optionally, load
local settings.
        .Build();
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutBucketLogging](#)。

CLI

AWS CLI

範例 1：設定儲存貯體政策記錄

以下 put-bucket-logging 範例會設定 amzn-s3-demo-bucket 的記錄政策。首先，使用 put-bucket-policy 命令在儲存貯體政策中，將許可權授予日誌記錄服務主體。

```
aws s3api put-bucket-policy \
  --bucket amzn-s3-demo-bucket \
  --policy file://policy.json
```

policy.json 的內容：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "S3ServerAccessLogsPolicy",
      "Effect": "Allow",
      "Principal": {"Service": "logging.s3.amazonaws.com"},
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/Logs/*",
      "Condition": {
        "ArnLike": {"aws:SourceARN": "arn:aws:s3:::SOURCE-BUCKET-NAME"},
        "StringEquals": {"aws:SourceAccount": "SOURCE-AWS-ACCOUNT-ID"}
      }
    }
  ]
}
```

使用 `put-bucket-logging` 套用記錄政策。

```
aws s3api put-bucket-logging \  
  --bucket amzn-s3-demo-bucket \  
  --bucket-logging-status file://logging.json
```

`logging.json` 的內容：

```
{  
  "LoggingEnabled": {  
    "TargetBucket": "amzn-s3-demo-bucket",  
    "TargetPrefix": "Logs/"  
  }  
}
```

需要 `put-bucket-policy` 命令才能將 `s3:PutObject` 許可權授予日誌記錄服務主體。

如需詳細資訊，請參閱《Amazon S3 使用者指南》中的 [Amazon S3 伺服器存取日誌記錄](#)。

範例 2：僅將日誌記錄的儲存貯體政策設定給單一使用者

以下 `put-bucket-logging` 範例會設定 `amzn-s3-demo-bucket` 的記錄政策。AWS 使用者 `bob@example.com` 將擁有對日誌檔案的完全控制權，且沒有人具有任何存取權。首先，使用 `put-bucket-acl` 授予 S3 許可權。

```
aws s3api put-bucket-acl \  
  --bucket amzn-s3-demo-bucket \  
  --grant-write URI=http://acs.amazonaws.com/groups/s3/LogDelivery \  
  --grant-read-acp URI=http://acs.amazonaws.com/groups/s3/LogDelivery
```

接著，使用 `put-bucket-logging` 套用日誌記錄政策。

```
aws s3api put-bucket-logging \  
  --bucket amzn-s3-demo-bucket \  
  --bucket-logging-status file://logging.json
```

`logging.json` 的內容：

```
{  
  "LoggingEnabled": {  
    "TargetBucket": "amzn-s3-demo-bucket",  
    "TargetPrefix": "amzn-s3-demo-bucket-logs/",  
  }  
}
```

```
    "TargetGrants": [
      {
        "Grantee": {
          "Type": "AmazonCustomerByEmail",
          "EmailAddress": "bob@example.com"
        },
        "Permission": "FULL_CONTROL"
      }
    ]
  }
}
```

必須使用 `put-bucket-acl` 命令，才能將必要許可權 (write 及 read-acp 許可權) 授予 S3 日誌交付系統。

如需詳細資訊，請參閱《Amazon S3 開發人員指南》中的 [Amazon S3 伺服器存取日誌記錄](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketLogging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **PutBucketNotification** 與 CLI

下列程式碼範例示範如何使用 `PutBucketNotification`。

CLI

AWS CLI

將通知組態套用至名為 `amzn-s3-demo-bucket` 的儲存貯體：

```
aws s3api put-bucket-notification --bucket amzn-s3-demo-bucket --notification-configuration file://notification.json
```

檔案 `notification.json` 是目前資料夾中的 JSON 文件，可用來指定 SNS 主題，以及要監控的事件類型：

```
{
  "TopicConfiguration": {
    "Event": "s3:ObjectCreated:*",
    "Topic": "arn:aws:sns:us-west-2:123456789012:s3-notification-topic"
  }
}
```



```
}
```

SNS 主題必須連接 IAM 政策，以允許 Amazon S3 發佈至該政策：

```
{
  "Version": "2012-10-17",
  "Id": "example-ID",
  "Statement": [
    {
      "Sid": "example-statement-ID",
      "Effect": "Allow",
      "Principal": {
        "Service": "s3.amazonaws.com"
      },
      "Action": [
        "SNS:Publish"
      ],
      "Resource": "arn:aws:sns:us-west-2:123456789012:amzn-s3-demo-bucket",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:s3:*:*:amzn-s3-demo-bucket"
        }
      }
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketNotification](#)。

PowerShell

Tools for PowerShell V4

範例 1：此範例會設定 S3 事件 ObjectRemovedDelete 的 SNS 主題組態，並啟用給定 s3 儲存貯體的通知

```
$topic = [Amazon.S3.Model.TopicConfiguration] @{
  Id = "delete-event"
  Topic = "arn:aws:sns:eu-west-1:123456789012:topic-1"
  Event = [Amazon.S3.EventType]::ObjectRemovedDelete
}
```

```
Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -TopicConfiguration
    $topic
```

範例 2：此範例會為傳送至 Lambda 函式的給定儲存貯體，啟用 ObjectCreatedAll 通知。

```
$lambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = "s3:ObjectCreated:*"
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:rdplock"
    Id = "ObjectCreated-Lambda"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".pem"}
            )
        }
    }
}

Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -
    LambdaFunctionConfiguration $lambdaConfig
```

範例 3：此範例會根據不同的索引鍵尾碼建立 2 個不同的 Lambda 組態，並在單一命令中同時設定兩者。

```
#Lambda Config 1

$firstLambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = "s3:ObjectCreated:*"
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:verifynet"
    Id = "ObjectCreated-dada-ps1"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".ps1"}
            )
        }
    }
}

#Lambda Config 2
```

```
$secondLambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = [Amazon.S3.EventType]::ObjectCreatedAll
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:verifyssm"
    Id = "ObjectCreated-dada-json"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".json"}
            )
        }
    }
}
```

```
Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -
LambdaFunctionConfiguration $firstLambdaConfig,$secondLambdaConfig
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [PutBucketNotification](#)。

Tools for PowerShell V5

範例 1：此範例會設定 S3 事件 ObjectRemovedDelete 的 SNS 主題組態，並啟用給定 s3 儲存貯體的通知

```
$topic = [Amazon.S3.Model.TopicConfiguration] @{
    Id = "delete-event"
    Topic = "arn:aws:sns:eu-west-1:123456789012:topic-1"
    Event = [Amazon.S3.EventType]::ObjectRemovedDelete
}

Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -TopicConfiguration
$topic
```

範例 2：此範例會為傳送至 Lambda 函式的給定儲存貯體，啟用 ObjectCreatedAll 通知。

```
$lambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = "s3:ObjectCreated:*"
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:rdplock"
    Id = "ObjectCreated-Lambda"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
```

```

        @{Name="Prefix";Value="dada"}
        @{Name="Suffix";Value=".pem"}
    )
}
}
}

```

```

Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -
LambdaFunctionConfiguration $lambdaConfig

```

範例 3：此範例會根據不同的索引鍵尾碼建立 2 個不同的 Lambda 組態，並在單一命令中同時設定兩者。

```
#Lambda Config 1
```

```

$firstLambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = "s3:ObjectCreated:*"
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:verifynet"
    Id = "ObjectCreated-dada-ps1"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".ps1"}
            )
        }
    }
}

```

```
#Lambda Config 2
```

```

$secondLambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
    Events = [Amazon.S3.EventType]::ObjectCreatedAll
    FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:verifyssm"
    Id = "ObjectCreated-dada-json"
    Filter = @{
        S3KeyFilter = @{
            FilterRules = @(
                @{Name="Prefix";Value="dada"}
                @{Name="Suffix";Value=".json"}
            )
        }
    }
}

```

```
}
```

```
Write-S3BucketNotification -BucketName amzn-s3-demo-bucket -  
LambdaFunctionConfiguration $firstLambdaConfig,$secondLambdaConfig
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [PutBucketNotification](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketNotificationConfiguration 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketNotificationConfiguration。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [處理 S3 事件通知](#)
- [將事件通知傳送至 EventBridge](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;  
using System.Collections.Generic;  
using System.Threading.Tasks;  
using Amazon.S3;  
using Amazon.S3.Model;  
  
/// <summary>  
/// This example shows how to enable notifications for an Amazon Simple  
/// Storage Service (Amazon S3) bucket.
```

```
/// </summary>
public class EnableNotifications
{
    public static async Task Main()
    {
        const string bucketName = "amzn-s3-demo-bucket1";
        const string snsTopic = "arn:aws:sns:us-east-2:0123456789ab:bucket-
notify";
        const string sqsQueue = "arn:aws:sqs:us-
east-2:0123456789ab:Example_Queue";

        IAmazonS3 client = new AmazonS3Client(Amazon.RegionEndpoint.USEast2);
        await EnableNotificationAsync(client, bucketName, snsTopic,
sqsQueue);
    }

    /// <summary>
    /// This method makes the call to the PutBucketNotificationAsync method.
    /// </summary>
    /// <param name="client">An initialized Amazon S3 client used to call
    /// the PutBucketNotificationAsync method.</param>
    /// <param name="bucketName">The name of the bucket for which
    /// notifications will be turned on.</param>
    /// <param name="snsTopic">The ARN for the Amazon Simple Notification
    /// Service (Amazon SNS) topic associated with the S3 bucket.</param>
    /// <param name="sqsQueue">The ARN of the Amazon Simple Queue Service
    /// (Amazon SQS) queue to which notifications will be pushed.</param>
    public static async Task EnableNotificationAsync(
        IAmazonS3 client,
        string bucketName,
        string snsTopic,
        string sqsQueue)
    {
        try
        {
            // The bucket for which we are setting up notifications.
            var request = new PutBucketNotificationRequest()
            {
                BucketName = bucketName,
            };

            // Defines the topic to use when sending a notification.
            var topicConfig = new TopicConfiguration()
            {
```

```
        Events = new List<EventType> { EventType.ObjectCreatedCopy },
        Topic = snsTopic,
    };
    request.TopicConfigurations = new List<TopicConfiguration>
    {
        topicConfig,
    };
    request.QueueConfigurations = new List<QueueConfiguration>
    {
        new QueueConfiguration()
        {
            Events = new List<EventType>
            { EventType.ObjectCreatedPut },
            Queue = sqsQueue,
        },
    };

    // Now apply the notification settings to the bucket.
    PutBucketNotificationResponse response = await
client.PutBucketNotificationAsync(request);
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error: {ex.Message}");
    }
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutBucketNotificationConfiguration](#)。

CLI

AWS CLI

啟用儲存貯體的指定通知

以下 `put-bucket-notification-configuration` 範例會套用通知組態至名為 `amzn-s3-demo-bucket` 的儲存貯體。檔案 `notification.json` 是目前資料夾中的 JSON 文件，可用來指定 SNS 主題，以及要監控的事件類型。

```
aws s3api put-bucket-notification-configuration \  
  --bucket amzn-s3-demo-bucket \  
  --notification-configuration file://notification.json
```

notification.json 的內容：

```
{  
  "TopicConfigurations": [  
    {  
      "TopicArn": "arn:aws:sns:us-west-2:123456789012:s3-notification-  
topic",  
      "Events": [  
        "s3:ObjectCreated:*"  
      ]  
    }  
  ]  
}
```

SNS 主題必須連接 IAM 政策，以允許 Amazon S3 發佈至該政策。

```
{  
  "Version": "2012-10-17",  
  "Id": "example-ID",  
  "Statement": [  
    {  
      "Sid": "example-statement-ID",  
      "Effect": "Allow",  
      "Principal": {  
        "Service": "s3.amazonaws.com"  
      },  
      "Action": [  
        "SNS:Publish"  
      ],  
      "Resource": "arn:aws:sns:us-west-2:123456789012::s3-notification-  
topic",  
      "Condition": {  
        "ArnLike": {  
          "aws:SourceArn": "arn:aws:s3:*:*:amzn-s3-demo-bucket"  
        }  
      }  
    }  
  ]  
}
```



```
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketNotificationConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketPolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketPolicy。

C++

SDK for C++

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::putBucketPolicy(const Aws::String &bucketName,
                                  const Aws::String &policyBody,
                                  const Aws::S3::S3ClientConfiguration
&clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    std::shared_ptr<Aws::StringStream> request_body =
        Aws::MakeShared<Aws::StringStream>("");
    *request_body << policyBody;

    Aws::S3::Model::PutBucketPolicyRequest request;
    request.SetBucket(bucketName);
    request.SetBody(request_body);

    Aws::S3::Model::PutBucketPolicyOutcome outcome =
        s3Client.PutBucketPolicy(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: putBucketPolicy: "
```

```

        << outcome.GetError().GetMessage() << std::endl;
    } else {
        std::cout << "Set the following policy body for the bucket '" <<
            bucketName << "':" << std::endl << std::endl;
        std::cout << policyBody << std::endl;
    }

    return outcome.IsSuccess();
}

//! Build a policy JSON string.
/*!
    \param userArn: Aws user Amazon Resource Name (ARN).
        For more information, see https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_identifiers.html#identifiers-arns.
    \param bucketName: Name of a bucket.
    \return String: Policy as JSON string.
*/

Aws::String getPolicyString(const Aws::String &userArn,
                           const Aws::String &bucketName) {
    return
        "{\n"
        "  \"Version\": \"2012-10-17\", \n"
        "  \"Statement\": [\n"
        "    {\n"
        "      \"Sid\": \"1\", \n"
        "      \"Effect\": \"Allow\", \n"
        "      \"Principal\": {\n"
        "        \"AWS\": \"\"
        + userArn +
        "\"\"\"      }, \n"
        "      \"Action\": [ \"s3:getObject\" ], \n"
        "      \"Resource\": [ \"arn:aws:s3::\"
        + bucketName +
        "\"/*\" ] \n"
        "    } \n"
        "  ] \n"
        "}";
}

```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [PutBucketPolicy](#)。

CLI

AWS CLI

此範例允許所有使用者擷取 `amzn-s3-demo-bucket` 中的任何物件，但 `MySecretFolder` 中的物件除外。它也會將 `put` 和 `delete` 許可授予 AWS 帳戶的根使用者 `1234-5678-9012`：

```
aws s3api put-bucket-policy --bucket amzn-s3-demo-bucket --policy file://policy.json
```

policy.json:

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    },
    {
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/MySecretFolder/*"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Action": [
        "s3:DeleteObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketPolicy](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutBucketPolicyRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;

import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.List;

import com.fasterxml.jackson.core.JsonParser;
import com.fasterxml.jackson.databind.ObjectMapper;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class SetBucketPolicy {
    public static void main(String[] args) {
        final String usage = ""

        Usage:
            <bucketName> <polFile>

        Where:
```

```

        bucketName - The Amazon S3 bucket to set the policy on.
        polFile - A JSON file containing the policy (see the Amazon S3
Readme for an example).\s
        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String bucketName = args[0];
    String polFile = args[1];
    String policyText = getBucketPolicyFromFile(polFile);
    Region region = Region.US_EAST_1;
    S3Client s3 = S3Client.builder()
        .region(region)
        .build();

    setPolicy(s3, bucketName, policyText);
    s3.close();
}

/**
 * Sets the policy for an Amazon S3 bucket.
 *
 * @param s3          the {@link S3Client} object used to interact with the
Amazon S3 service
 * @param bucketName the name of the Amazon S3 bucket
 * @param policyText the text of the policy to be set on the bucket
 * @throws S3Exception if there is an error setting the bucket policy
 */
public static void setPolicy(S3Client s3, String bucketName, String
policyText) {
    System.out.println("Setting policy:");
    System.out.println("----");
    System.out.println(policyText);
    System.out.println("----");
    System.out.format("On Amazon S3 bucket: \"%s\"\n", bucketName);

    try {
        PutBucketPolicyRequest policyReq = PutBucketPolicyRequest.builder()
            .bucket(bucketName)
            .policy(policyText)
            .build();
    }
}

```

```
s3.putBucketPolicy(policyReq);

} catch (S3Exception e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}

System.out.println("Done!");
}

/**
 * Retrieves the bucket policy from a specified file.
 *
 * @param policyFile the path to the file containing the bucket policy
 * @return the content of the bucket policy file as a string
 */
public static String getBucketPolicyFromFile(String policyFile) {
    StringBuilder fileText = new StringBuilder();
    try {
        List<String> lines = Files.readAllLines(Paths.get(policyFile),
StandardCharsets.UTF_8);
        for (String line : lines) {
            fileText.append(line);
        }

    } catch (IOException e) {
        System.out.format("Problem reading file: \"%s\"", policyFile);
        System.out.println(e.getMessage());
    }

    try {
        final JsonParser parser = new
ObjectMapper().getFactory().createParser(fileText.toString());
        while (parser.nextToken() != null) {
        }

    } catch (IOException jpe) {
        jpe.printStackTrace();
    }
    return fileText.toString();
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutBucketPolicy](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

新增政策。

```
import {
  PutBucketPolicyCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Grant an IAM role GetObject access to all of the objects
 * in the provided bucket.
 * @param {{ bucketName: string, iamRoleArn: string }}
 */
export const main = async ({ bucketName, iamRoleArn }) => {
  const client = new S3Client({});
  const command = new PutBucketPolicyCommand({
    // This is a resource-based policy. For more information on resource-based
    // policies,
    // see https://docs.aws.amazon.com/IAM/latest/UserGuide/
    // access_policies.html#policies_resource-based.
    Policy: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Principal: {
            AWS: iamRoleArn,
          },
        },
      ],
    }),
  });
```

```
        Action: "s3:GetObject",
        Resource: `arn:aws:s3:::${bucketName}/*`,
    },
],
}),
// Apply the preceding policy to this bucket.
Bucket: bucketName,
});

try {
    await client.send(command);
    console.log(
        `GetObject access to the bucket "${bucketName}" was granted to the provided
IAM role.`,
    );
} catch (caught) {
    if (
        caught instanceof S3ServiceException &&
        caught.name === "MalformedPolicy"
    ) {
        console.error(
            `Error from S3 while setting the bucket policy for the bucket
"${bucketName}". The policy was malformed.`,
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while setting the bucket policy for the bucket
"${bucketName}". ${caught.name}: ${caught.message}`,
        );
    } else {
        throw caught;
    }
}
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [PutBucketPolicy](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def put_policy(self, policy):
        """
        Apply a security policy to the bucket. Policies control users' ability
        to perform specific actions, such as listing the objects in the bucket.

        :param policy: The policy to apply to the bucket.
        """
        try:
            self.bucket.Policy().put(Policy=json.dumps(policy))
            logger.info("Put policy %s for bucket '%s'.", policy,
self.bucket.name)
        except ClientError:
            logger.exception("Couldn't apply policy to bucket '%s'.",
self.bucket.name)
            raise
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [PutBucketPolicy](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object
  # configured with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  # Sets a policy on a bucket.
  #
  def policy(policy)
    @bucket_policy.put(policy: policy)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't set the policy for #{@bucket_policy.bucket.name}. Here's why:
    #{e.message}"
    false
  end
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [PutBucketPolicy](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.
  " Example policy JSON string
  " iv_policy = '{"Version":"2012-10-17",          "Statement":
[{"Effect":"Allow","Principal":{"AWS":"arn:aws:iam::123456789012:user/
user"},"Action":["s3:GetObject"],"Resource":["arn:aws:s3:::bucketname/*"]}]}'
  lo_s3->putbucketpolicy(
    iv_bucket = iv_bucket_name
    iv_policy = iv_policy ).
  MESSAGE 'Bucket policy set.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
  MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutBucketPolicy](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketReplication 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketReplication。

CLI

AWS CLI

設定 S3 儲存貯體的複寫

下列 put-bucket-replication 範例會將複寫組態套用至指定的 S3 儲存貯體。

```
aws s3api put-bucket-replication \  
  --bucket amzn-s3-demo-bucket1 \  
  --replication-configuration file://replication.json
```

replication.json 的內容：

```
{  
  "Role": "arn:aws:iam::123456789012:role/s3-replication-role",  
  "Rules": [  
    {  
      "Status": "Enabled",  
      "Priority": 1,  
      "DeleteMarkerReplication": { "Status": "Disabled" },  
      "Filter" : { "Prefix": ""},  
      "Destination": {  
        "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket2"  
      }  
    }  
  ]  
}
```

目的地儲存貯體必須啟用版本控制。指定的角色必須具有寫入目的地儲存貯體的許可權，並具有允許 Amazon S3 擔任該角色的信任關係。

範例角色許可政策：

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "s3:GetReplicationConfiguration",  
        "s3:ListBucket"  
      ],  
      "Resource": [  
        "arn:aws:s3:::amzn-s3-demo-bucket1"  
      ]  
    },  
    {  
      "Effect": "Allow",  
      "Action": [  

```

```
        "s3:GetObjectVersion",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
    ],
    "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket1/*"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "s3:ReplicateObject",
        "s3:ReplicateDelete",
        "s3:ReplicateTags"
    ],
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket2/*"
}
]
```

範例信任關係政策：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "s3.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

此命令不會產生輸出。

如需詳細資訊，請參閱《Amazon Simple Storage Service 主控台使用者指南》中的[這是主題標題](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketReplication](#)。

Java

SDK for Java 2.x

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Sets the replication configuration for an Amazon S3 bucket.
 *
 * @param s3Client          the S3Client instance to use for the operation
 * @param sourceBucketName  the name of the source bucket
 * @param destBucketName    the name of the destination bucket
 * @param destinationBucketARN the Amazon Resource Name (ARN) of the
destination bucket
 * @param roleARN           the ARN of the IAM role to use for the
replication configuration
 */
public static void setReplication(S3Client s3Client, String sourceBucketName,
String destBucketName, String destinationBucketARN, String roleARN) {
    try {
        Destination destination = Destination.builder()
            .bucket(destinationBucketARN)
            .storageClass(StorageClass.STANDARD)
            .build();

        // Define a prefix filter for replication.
        ReplicationRuleFilter ruleFilter = ReplicationRuleFilter.builder()
            .prefix("documents/")
            .build();

        // Define delete marker replication setting.
        DeleteMarkerReplication deleteMarkerReplication =
DeleteMarkerReplication.builder()
            .status(DeleteMarkerReplicationStatus.DISABLED)
            .build();

        // Create the replication rule.
        ReplicationRule replicationRule = ReplicationRule.builder()
```

```
        .priority(1)
        .filter(ruleFilter)
        .status(ReplicationRuleStatus.ENABLED)
        .deleteMarkerReplication(deleteMarkerReplication)
        .destination(destination)
        .build();

List<ReplicationRule> replicationRuleList = new ArrayList<>();
replicationRuleList.add(replicationRule);

// Define the replication configuration with IAM role.
ReplicationConfiguration configuration =
ReplicationConfiguration.builder()
    .role(roleARN)
    .rules(replicationRuleList)
    .build();

// Apply the replication configuration to the source bucket.
PutBucketReplicationRequest replicationRequest =
PutBucketReplicationRequest.builder()
    .bucket(sourceBucketName)
    .replicationConfiguration(configuration)
    .build();

s3Client.putBucketReplication(replicationRequest);
System.out.println("Replication configuration set successfully.");

} catch (IllegalArgumentException e) {
    System.err.println("Configuration error: " + e.getMessage());
} catch (S3Exception e) {
    System.err.println("S3 Exception: " +
e.awsErrorDetails().errorMessage());
    System.err.println("Status Code: " + e.statusCode());
    System.err.println("Error Code: " + e.awsErrorDetails().errorCode());

} catch (SdkException e) {
    System.err.println("SDK Exception: " + e.getMessage());
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutBucketReplication](#)。

PowerShell

Tools for PowerShell V4

範例 1：此範例設定了一個複寫組態，其中的一條規則允許將儲存貯體 'amzn-s3-demo-bucket' 中鍵名字首為 "TaxDocs" 所建立的任何新物件，複製到 'amzn-s3-demo-bucket' 儲存貯體。

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Enabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-
bucket" }

$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1
}

Write-S3BucketReplication @params
```

範例 2：此範例設定具有多個規則的複寫組態，可將任何以鍵名字首 "TaxDocs" 或 "OtherDocs" 建立的新物件，複寫至 'amzn-s3-demo-bucket' 儲存貯體。索引鍵字首不得重疊。

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Enabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-
bucket" }

$rule2 = New-Object Amazon.S3.Model.ReplicationRule
$rule2.ID = "Rule-2"
$rule2.Status = "Enabled"
$rule2.Prefix = "OtherDocs"
$rule2.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-
bucket" }
```



```
$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1,$rule2
}

Write-S3BucketReplication @params
```

範例 3：此範例會更新指定之儲存貯體上的複寫組態，以停用控制將鍵名字首為 "TaxDocs" 之物件複寫至 'amzn-s3-demo-bucket' 儲存貯體的規則。

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Disabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-
bucket" }

$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1
}

Write-S3BucketReplication @params
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [PutBucketReplication](#)。

Tools for PowerShell V5

範例 1：此範例設定了一個複寫組態，其中的一條規則允許將儲存貯體 'amzn-s3-demo-bucket' 中鍵名字首為 "TaxDocs" 所建立的任何新物件，複製到 'amzn-s3-demo-bucket' 儲存貯體。

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Enabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-
bucket" }
```

```
$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1
}

Write-S3BucketReplication @params
```

範例 2：此範例設定具有多個規則的複寫組態，可將任何以鍵名字首 "TaxDocs" 或 "OtherDocs" 建立的新物件，複寫至 'amzn-s3-demo-bucket' 儲存貯體。索引鍵字首不得重疊。

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Enabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-
bucket" }

$rule2 = New-Object Amazon.S3.Model.ReplicationRule
$rule2.ID = "Rule-2"
$rule2.Status = "Enabled"
$rule2.Prefix = "OtherDocs"
$rule2.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-
bucket" }

$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1,$rule2
}

Write-S3BucketReplication @params
```

範例 3：此範例會更新指定之儲存貯體上的複寫組態，以停用控制將鍵名字首為 "TaxDocs" 之物件複寫至 'amzn-s3-demo-bucket' 儲存貯體的規則。

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Disabled"
```

```
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::amzn-s3-demo-destination-
bucket" }

$params = @{
    BucketName = "amzn-s3-demo-bucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1
}

Write-S3BucketReplication @params
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [PutBucketReplication](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **PutBucketRequestPayment** 與 CLI

下列程式碼範例示範如何使用 PutBucketRequestPayment。

CLI

AWS CLI

範例 1：啟用儲存貯體的 ``requester pays`` 組態

下列 put-bucket-request-payment 範例會啟用指定儲存貯體的 requester pays。

```
aws s3api put-bucket-request-payment \
    --bucket amzn-s3-demo-bucket \
    --request-payment-configuration '{"Payer":"Requester"}'
```

此命令不會產生輸出。

範例 2：停用儲存貯體的 ``requester pays`` 組態

下列 put-bucket-request-payment 範例會針對指定的儲存貯體停用 requester pays。

```
aws s3api put-bucket-request-payment \
```

```
--bucket amzn-s3-demo-bucket \  
--request-payment-configuration '{"Payer":"BucketOwner"}'
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketRequestPayment](#)。

PowerShell

Tools for PowerShell V4

範例 1：更新名為 'amzn-s3-demo-bucket' 之儲存貯體的請求付費組態，如此一來，請求從儲存貯體下載的人員將需支付下載費用。根據預設，儲存貯體擁有者需支付下載費用。若要將請求付費設回預設，請針對 RequestPaymentConfiguration_Payer 參數使用 'BucketOwner'。

```
Write-S3BucketRequestPayment -BucketName amzn-s3-demo-bucket -  
RequestPaymentConfiguration_Payer Requester
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [PutBucketRequestPayment](#)。

Tools for PowerShell V5

範例 1：更新名為 'amzn-s3-demo-bucket' 之儲存貯體的請求付費組態，如此一來，請求從儲存貯體下載的人員將需支付下載費用。根據預設，儲存貯體擁有者需支付下載費用。若要將請求付費設回預設，請針對 RequestPaymentConfiguration_Payer 參數使用 'BucketOwner'。

```
Write-S3BucketRequestPayment -BucketName amzn-s3-demo-bucket -  
RequestPaymentConfiguration_Payer Requester
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [PutBucketRequestPayment](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 PutBucketTagging 與 CLI

下列程式碼範例示範如何使用 PutBucketTagging。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [物件儲存入門](#)

CLI

AWS CLI

下列命令會將標記組態套用至名為 `amzn-s3-demo-bucket` 的儲存貯體：

```
aws s3api put-bucket-tagging --bucket amzn-s3-demo-bucket --tagging file://tagging.json
```

檔案 `tagging.json` 是目前資料夾中指定標籤的 JSON 文件：

```
{
  "TagSet": [
    {
      "Key": "organization",
      "Value": "marketing"
    }
  ]
}
```

或直接從命令列將標記組態套用至 `amzn-s3-demo-bucket`：

```
aws s3api put-bucket-tagging --bucket amzn-s3-demo-bucket --tagging 'TagSet=[{Key=organization,Value=marketing}]'
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketTagging](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會將兩個標籤套用至名為 `cloudtrail-test-2018` 的儲存貯體：一個是索引鍵為 `Stage` 且值為 `Test` 的標籤，另一個是索引鍵為 `Environment` 且值為 `Alpha` 的標籤。若要驗證標籤是否已新增至儲存貯體，請執行 `Get-S3BucketTagging -BucketName`

bucket_name。結果應該會顯示您在第一個命令中套用至儲存貯體的標籤。請注意，**Write-S3BucketTagging** 會覆寫儲存貯體上設定的整個現有標籤。若要新增或刪除個別標籤，請執行資源群組和標記 API Cmdlet，**Add-RGTResourceTag** 和 **Remove-RGTResourceTag**。或者，在 AWS 管理主控台中使用標籤編輯器來管理 S3 儲存貯體標籤。

```
Write-S3BucketTagging -BucketName amzn-s3-demo-bucket -TagSet @( @{ Key="Stage"; Value="Test" }, @{ Key="Environment"; Value="Alpha" } )
```

範例 2：此命令會將名為 **cloudtrail-test-2018** 的儲存貯體傳輸至 **Write-S3BucketTagging** cmdlet。範例會將標籤 Stage:Production 和 Department:Finance 套用至儲存貯體。請注意，**Write-S3BucketTagging** 會覆寫儲存貯體上設定的整個現有標籤。

```
Get-S3Bucket -BucketName amzn-s3-demo-bucket | Write-S3BucketTagging -TagSet @( @{ Key="Stage"; Value="Production" }, @{ Key="Department"; Value="Finance" } )
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [PutBucketTagging](#)。

Tools for PowerShell V5

範例 1：此命令會將兩個標籤套用至名為 **cloudtrail-test-2018** 的儲存貯體：一個是索引鍵為 Stage 且值為 Test 的標籤，另一個是索引鍵為 Environment 且值為 Alpha 的標籤。若要驗證標籤是否已新增至儲存貯體，請執行 **Get-S3BucketTagging -BucketName bucket_name**。結果應該會顯示您在第一個命令中套用至儲存貯體的標籤。請注意，**Write-S3BucketTagging** 會覆寫儲存貯體上設定的整個現有標籤。若要新增或刪除個別標籤，請執行資源群組和標記 API Cmdlet，**Add-RGTResourceTag** 和 **Remove-RGTResourceTag**。或者，在 AWS 管理主控台中使用標籤編輯器來管理 S3 儲存貯體標籤。

```
Write-S3BucketTagging -BucketName amzn-s3-demo-bucket -TagSet @( @{ Key="Stage"; Value="Test" }, @{ Key="Environment"; Value="Alpha" } )
```

範例 2：此命令會將名為 **cloudtrail-test-2018** 的儲存貯體傳輸至 **Write-S3BucketTagging** cmdlet。範例會將標籤 Stage:Production 和 Department:Finance 套用至儲存貯體。請注意，**Write-S3BucketTagging** 會覆寫儲存貯體上設定的整個現有標籤。

```
Get-S3Bucket -BucketName amzn-s3-demo-bucket | Write-S3BucketTagging -TagSet @( @{ Key="Stage"; Value="Production" }, @{ Key="Department"; Value="Finance" } )
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [PutBucketTagging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketVersioning 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketVersioning。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [物件儲存入門](#)

CLI

AWS CLI

下列命令會啟用名為 amzn-s3-demo-bucket 的儲存貯體上的版本控制：

```
aws s3api put-bucket-versioning --bucket amzn-s3-demo-bucket --versioning-configuration Status=Enabled
```

下列命令會啟用版本控制，並使用 mfa 程式碼

```
aws s3api put-bucket-versioning --bucket amzn-s3-demo-bucket --versioning-configuration Status=Enabled --mfa "SERIAL 123456"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketVersioning](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Enables bucket versioning for the specified S3 bucket.
 *
 * @param s3Client the S3 client to use for the operation
 * @param bucketName the name of the S3 bucket to enable versioning for
 */
public static void enableBucketVersioning(S3Client s3Client, String
bucketName){
    VersioningConfiguration versioningConfiguration =
VersioningConfiguration.builder()
        .status(BucketVersioningStatus.ENABLED)
        .build();

    PutBucketVersioningRequest versioningRequest =
PutBucketVersioningRequest.builder()
        .bucket(bucketName)
        .versioningConfiguration(versioningConfiguration)
        .build();

    s3Client.putBucketVersioning(versioningRequest);
    System.out.println("Bucket versioning has been enabled for "+bucketName);
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutBucketVersioning](#)。

PowerShell

Tools for PowerShell V4

範例 1：命令會啟用給定 S3 儲存貯體的版本控制。

```
Write-S3BucketVersioning -BucketName 'amzn-s3-demo-bucket' -
VersioningConfig_Status Enabled
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [PutBucketVersioning](#)。

Tools for PowerShell V5

範例 1：命令會啟用給定 S3 儲存貯體的版本控制。


```
Write-S3BucketVersioning -BucketName 'amzn-s3-demo-bucket' -  
VersioningConfig_Status Enabled
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [PutBucketVersioning](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    " Example: Enable versioning on a bucket  
    " iv_status = 'Enabled'  
    lo_s3->putbucketversioning(  
        iv_bucket = iv_bucket_name  
        io_versioningconfiguration = NEW /aws1/cl_s3_versioningconf(  
            iv_status = iv_status ) ).  
    MESSAGE 'Bucket versioning enabled.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutBucketVersioning](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketWebsite 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutBucketWebsite。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
// Put the website configuration.
PutBucketWebsiteRequest putRequest = new
PutBucketWebsiteRequest()
{
    BucketName = bucketName,
    WebsiteConfiguration = new WebsiteConfiguration()
    {
        IndexDocumentSuffix = indexDocumentSuffix,
        ErrorDocument = errorDocument,
    },
};
PutBucketWebsiteResponse response = await
client.PutBucketWebsiteAsync(putRequest);
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutBucketWebsite](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::putWebsiteConfig(const Aws::String &bucketName,
```

```

const Aws::String &indexPage, const Aws::String
&errorPage,
const Aws::S3::S3ClientConfiguration
&clientConfig) {
    Aws::S3::S3Client client(clientConfig);

    Aws::S3::Model::IndexDocument indexDocument;
    indexDocument.SetSuffix(indexPage);

    Aws::S3::Model::ErrorDocument errorDocument;
    errorDocument.SetKey(errorPage);

    Aws::S3::Model::WebsiteConfiguration websiteConfiguration;
    websiteConfiguration.SetIndexDocument(indexDocument);
    websiteConfiguration.SetErrorDocument(errorDocument);

    Aws::S3::Model::PutBucketWebsiteRequest request;
    request.SetBucket(bucketName);
    request.SetWebsiteConfiguration(websiteConfiguration);

    Aws::S3::Model::PutBucketWebsiteOutcome outcome =
        client.PutBucketWebsite(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: PutBucketWebsite: "
                  << outcome.GetError().GetMessage() << std::endl;
    } else {
        std::cout << "Success: Set website configuration for bucket '"
                  << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [PutBucketWebsite](#)。

CLI

AWS CLI

將靜態網站組態套用至名為 `amzn-s3-demo-bucket` 的儲存貯體：

```
aws s3api put-bucket-website --bucket amzn-s3-demo-bucket --website-configuration file://website.json
```

檔案 `website.json` 是目前資料夾中的 JSON 文件，可用於指定網站的索引和錯誤頁面：

```
{
  "IndexDocument": {
    "Suffix": "index.html"
  },
  "ErrorDocument": {
    "Key": "error.html"
  }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutBucketWebsite](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.IndexDocument;
import software.amazon.awssdk.services.s3.model.PutBucketWebsiteRequest;
import software.amazon.awssdk.services.s3.model.WebsiteConfiguration;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
```

```

    * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
    */

public class SetWebsiteConfiguration {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <bucketName> [indexdoc]\s

            Where:
                bucketName    - The Amazon S3 bucket to set the website
configuration on.\s
                indexdoc - The index document, ex. 'index.html'
                        If not specified, 'index.html' will be set.

            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String indexDoc = "index.html";
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        setWebsiteConfig(s3, bucketName, indexDoc);
        s3.close();
    }

    /**
     * Sets the website configuration for an Amazon S3 bucket.
     *
     * @param s3 The {@link S3Client} instance to use for the AWS SDK operations.
     * @param bucketName The name of the S3 bucket to configure.
     * @param indexDoc The name of the index document to use for the website
configuration.
     */
    public static void setWebsiteConfig(S3Client s3, String bucketName, String
indexDoc) {
        try {

```

```
        WebsiteConfiguration websiteConfig = WebsiteConfiguration.builder()
            .indexDocument(IndexDocument.builder().suffix(indexDoc).build())
            .build();

        PutBucketWebsiteRequest pubWebsiteReq =
        PutBucketWebsiteRequest.builder()
            .bucket(bucketName)
            .websiteConfiguration(websiteConfig)
            .build();

        s3.putBucketWebsite(pubWebsiteReq);
        System.out.println("The call was successful");

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutBucketWebsite](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

設定儲存貯體網站組態。

```
import {
    PutBucketWebsiteCommand,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
```

```
* Configure an Amazon S3 bucket to serve a static website.
* Website access must also be granted separately. For more information
* on setting the permissions for website access, see
* https://docs.aws.amazon.com/AmazonS3/latest/userguide/WebsiteAccessPermissionsReqd.html.
*
* @param {{ bucketName: string }}
*/
export const main = async ({ bucketName }) => {
  const client = new S3Client({});
  const command = new PutBucketWebsiteCommand({
    Bucket: bucketName,
    WebsiteConfiguration: {
      ErrorDocument: {
        // The object key name to use when a 4XX class error occurs.
        Key: "error.html",
      },
      IndexDocument: {
        // A suffix that is appended to a request when the request is
        // for a directory.
        Suffix: "index.html",
      },
    },
  });

  try {
    await client.send(command);
    console.log(
      `The bucket "${bucketName}" has been configured as a static website.`
    );
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
      console.error(
        `Error from S3 while configuring the bucket "${bucketName}" as a static website. The bucket doesn't exist.`
      );
    } else if (caught instanceof S3ServiceException) {
      console.error(
        `Error from S3 while configuring the bucket "${bucketName}" as a static website. ${caught.name}: ${caught.message}`
      );
    }
  }
}
```

```
    } else {  
        throw caught;  
    }  
}  
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [PutBucketWebsite](#)。

PowerShell

Tools for PowerShell V4

範例 1：命令會啟用給定儲存貯體的網站託管，其中索引文件為 'index.html'，錯誤文件為 'error.html'。

```
Write-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'  
-WebsiteConfiguration_IndexDocumentSuffix 'index.html' -  
WebsiteConfiguration_ErrorDocument 'error.html'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [PutBucketWebsite](#)。

Tools for PowerShell V5

範例 1：命令會啟用給定儲存貯體的網站託管，其中索引文件為 'index.html'，錯誤文件為 'error.html'。

```
Write-S3BucketWebsite -BucketName 'amzn-s3-demo-bucket'  
-WebsiteConfiguration_IndexDocumentSuffix 'index.html' -  
WebsiteConfiguration_ErrorDocument 'error.html'
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [PutBucketWebsite](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-s3'

# Wraps Amazon S3 bucket website actions.
class BucketWebsiteWrapper
  attr_reader :bucket_website

  # @param bucket_website [Aws::S3::BucketWebsite] A bucket website object
  # configured with an existing bucket.
  def initialize(bucket_website)
    @bucket_website = bucket_website
  end

  # Sets a bucket as a static website.
  #
  # @param index_document [String] The name of the index document for the
  # website.
  # @param error_document [String] The name of the error document to show for 4XX
  # errors.
  # @return [Boolean] True when the bucket is configured as a website; otherwise,
  # false.
  def set_website(index_document, error_document)
    @bucket_website.put(
      website_configuration: {
        index_document: { suffix: index_document },
        error_document: { key: error_document }
      }
    )
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't configure #{@bucket_website.bucket.name} as a website. Here's
    why: #{e.message}"
    false
  end
end
```

```
end
end

# Example usage:
def run_demo
  bucket_name = "amzn-s3-demo-bucket"
  index_document = "index.html"
  error_document = "404.html"

  wrapper = BucketWebsiteWrapper.new(Aws::S3::BucketWebsite.new(bucket_name))
  return unless wrapper.set_website(index_document, error_document)

  puts "Successfully configured bucket #{bucket_name} as a static website."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [PutBucketWebsite](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutObject 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutObject。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)
- [物件儲存入門](#)
- [提出條件式請求](#)
- [追蹤上傳和下載](#)
- [使用 Amazon S3 物件完整性](#)

.NET

適用於 .NET 的 SDK (v4)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Shows how to upload a file from the local computer to an Amazon S3
/// bucket.
/// </summary>
/// <param name="bucketName">The Amazon S3 bucket to which the object
/// will be uploaded.</param>
/// <param name="objectName">The object to upload.</param>
/// <param name="filePath">The path, including file name, of the object
/// on the local computer to upload.</param>
/// <returns>A boolean value indicating the success or failure of the
/// upload procedure.</returns>
public async Task<bool> UploadFileAsync(
    string bucketName,
    string objectName,
    string filePath)
{
    try
    {
        var request = new PutObjectRequest
        {
            BucketName = bucketName,
            Key = objectName,
            FilePath = filePath,
        };

        var response = await _amazonS3.PutObjectAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error uploading {objectName}: {ex.Message}");
    }
}
```

```
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutObject](#)。

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用伺服器端加密上傳物件。

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to upload an object to an Amazon Simple Storage
/// Service (Amazon S3) bucket with server-side encryption enabled.
/// </summary>
public class ServerSideEncryption
{
    public static async Task Main()
    {
        string bucketName = "amzn-s3-demo-bucket";
        string keyName = "samplefile.txt";

        // If the AWS Region defined for your default user is different
        // from the Region where your Amazon S3 bucket is located,
        // pass the Region name to the Amazon S3 client object's constructor.
        // For example: RegionEndpoint.USWest2.
        IAmazonS3 client = new AmazonS3Client();

        await WritingAnObjectAsync(client, bucketName, keyName);
    }
}
```

```
/// <summary>
/// Upload a sample object include a setting for encryption.
/// </summary>
/// <param name="client">The initialized Amazon S3 client object used to
/// to upload a file and apply server-side encryption.</param>
/// <param name="bucketName">The name of the Amazon S3 bucket where the
/// encrypted object will reside.</param>
/// <param name="keyName">The name for the object that you want to
/// create in the supplied bucket.</param>
public static async Task WritingAnObjectAsync(IAmazonS3 client, string
bucketName, string keyName)
{
    try
    {
        var putRequest = new PutObjectRequest
        {
            BucketName = bucketName,
            Key = keyName,
            ContentBody = "sample text",
            ServerSideEncryptionMethod =
ServerSideEncryptionMethod.AES256,
        };

        var putResponse = await client.PutObjectAsync(putRequest);

        // Determine the encryption state of an object.
        GetObjectMetadataRequest metadataRequest = new
GetObjectMetadataRequest
        {
            BucketName = bucketName,
            Key = keyName,
        };
        GetObjectMetadataResponse response = await
client.GetObjectMetadataAsync(metadataRequest);
        ServerSideEncryptionMethod objectEncryption =
response.ServerSideEncryptionMethod;

        Console.WriteLine($"Encryption method used: {0}",
objectEncryption.ToString());
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error: '{ex.Message}' when writing an
object");
    }
}
```

```

    }
  }
}

```

使用條件式請求放置物件。

```

/// <summary>
/// Uploads an object to Amazon S3 with a conditional request. Prevents
overwrite using an IfNoneMatch condition for the object key.
/// </summary>
/// <param name="objectKey">The key of the object to upload.</param>
/// <param name="bucket">The source bucket of the object.</param>
/// <param name="content">The content to upload as a string.</param>
/// <returns>The ETag if the conditional write is successful, empty
otherwise.</returns>
public async Task<string> PutObjectConditional(string objectKey, string
bucket, string content)
{
    try
    {
        var putObjectRequest = new PutObjectRequest
        {
            BucketName = bucket,
            Key = objectKey,
            ContentBody = content,
            IfNoneMatch = "*"
        };

        var putResult = await _amazonS3.PutObjectAsync(putObjectRequest);
        _logger.LogInformation($"Conditional write successful for key
{objectKey} in bucket {bucket}.");
        return putResult.ETag;
    }
    catch (AmazonS3Exception e)
    {
        if (e.ErrorCode == "PreconditionFailed")
        {
            _logger.LogError("Conditional write failed: Precondition
failed");
        }
        else
    }
}

```

```

        {
            _logger.LogError($"Unexpected error: {e.ErrorCode}");
            throw;
        }
        return string.Empty;
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutObject](#)。

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function copy_file_to_bucket
#
# This function creates a file in the specified bucket.
#
# Parameters:
#     $1 - The name of the bucket to copy the file to.
#     $2 - The path and file name of the local file to copy to the bucket.
#     $3 - The key (name) to call the copy of the file in the bucket.
#
# Returns:
#     0 - If successful.

```

```
#      1 - If it fails.
#####
function copy_file_to_bucket() {
    local response bucket_name source_file destination_file_name
    bucket_name=$1
    source_file=$2
    destination_file_name=$3

    response=$(aws s3api put-object \
        --bucket "$bucket_name" \
        --body "$source_file" \
        --key "$destination_file_name")

    # shellcheck disable=SC2181
    if [[ ${?} -ne 0 ]]; then
        errecho "ERROR: AWS reports put-object operation failed.\n$response"
        return 1
    fi
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutObject](#)。

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::putObject(const Aws::String &bucketName,
                           const Aws::String &fileName,
                           const Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::PutObjectRequest request;
    request.SetBucket(bucketName);
    //We are using the name of the file as the key for the object in the bucket.
```



```
//However, this is just a string and can be set according to your retrieval
needs.
request.SetKey(fileName);

std::shared_ptr<Aws::IOStream> inputData =
    Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
                                   fileName.c_str(),
                                   std::ios_base::in |
std::ios_base::binary);

if (!*inputData) {
    std::cerr << "Error unable to read file " << fileName << std::endl;
    return false;
}

request.SetBody(inputData);

Aws::S3::Model::PutObjectOutcome outcome =
    s3Client.PutObject(request);

if (!outcome.IsSuccess()) {
    std::cerr << "Error: putObject: " <<
        outcome.GetError().GetMessage() << std::endl;
} else {
    std::cout << "Added object '" << fileName << "' to bucket '"
        << bucketName << "'.";
}

return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [PutObject](#)。

CLI

AWS CLI

範例 1：將物件上傳至 Amazon S3

以下 put-object 命令會將物件上傳至 Amazon S3。

```
aws s3api put-object \
```

```
--bucket amzn-s3-demo-bucket \  
--key my-dir/MySampleImage.png \  
--body MySampleImage.png
```

如需有關上傳物件的詳細資訊，請參閱《Amazon S3 開發人員指南》中的「上傳物件」<<http://docs.aws.amazon.com/AmazonS3/latest/dev/UploadingObjects.html>>。

範例 2：將影片檔上傳至 Amazon S3

以下 `put-object` 範例會上傳影片檔。

```
aws s3api put-object \  
  --bucket amzn-s3-demo-bucket \  
  --key my-dir/big-video-file.mp4 \  
  --body /media/videos/f-sharp-3-data-services.mp4
```

如需有關上傳物件的詳細資訊，請參閱《Amazon S3 開發人員指南》中的「上傳物件」<<http://docs.aws.amazon.com/AmazonS3/latest/dev/UploadingObjects.html>>。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutObject](#)。

Go

SDK for Go V2

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用低階 API 將物件放入儲存貯體。

```
import (  
    "bytes"  
    "context"  
    "errors"  
    "fmt"  
    "io"  
    "log"  
    "os"
```

```
"time"

"github.com/aws/aws-sdk-go-v2/aws"
"github.com/aws/aws-sdk-go-v2/feature/s3/manager"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/aws/aws-sdk-go-v2/service/s3/types"
"github.com/aws/smithy-go"
)

// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
// actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
// bucket
// and object actions.
type BucketBasics struct {
    S3Client *s3.Client
}

// UploadFile reads from a file and puts the data into an object in a bucket.
func (basics BucketBasics) UploadFile(ctx context.Context, bucketName string,
    objectKey string, fileName string) error {
    file, err := os.Open(fileName)
    if err != nil {
        log.Printf("Couldn't open file %v to upload. Here's why: %v\n", fileName, err)
    } else {
        defer file.Close()
        _, err = basics.S3Client.PutObject(ctx, &s3.PutObjectInput{
            Bucket: aws.String(bucketName),
            Key:    aws.String(objectKey),
            Body:   file,
        })
        if err != nil {
            var apiErr smithy.APIError
            if errors.As(err, &apiErr) && apiErr.ErrorCode() == "EntityTooLarge" {
                log.Printf("Error while uploading object to %s. The object is too large.\n"+
                    "To upload objects larger than 5GB, use the S3 console (160GB max)\n"+
                    "or the multipart upload API (5TB max).", bucketName)
            } else {
                log.Printf("Couldn't upload file %v to %v:%v. Here's why: %v\n",
                    fileName, bucketName, objectKey, err)
            }
        }
    }
}
```

```
    } else {
        err = s3.NewObjectExistsWaiter(basics.S3Client).Wait(
            ctx, &s3.HeadObjectInput{Bucket: aws.String(bucketName), Key:
aws.String(objectKey)}, time.Minute)
        if err != nil {
            log.Printf("Failed attempt to wait for object %s to exist.\n", objectKey)
        }
    }
}
return err
}
```

使用傳輸管理員將物件上傳至儲存貯體。

```
import (
    "bytes"
    "context"
    "errors"
    "fmt"
    "log"
    "time"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"
    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/aws-sdk-go-v2/service/s3/types"
    "github.com/aws/smithy-go"
)

// S3Actions wraps S3 service actions.
type S3Actions struct {
    S3Client    *s3.Client
    S3Manager   *manager.Uploader
}

// UploadObject uses the S3 upload manager to upload an object to a bucket.
func (actor S3Actions) UploadObject(ctx context.Context, bucket string, key
string, contents string) (string, error) {
```

```
var outKey string
input := &s3.PutObjectInput{
    Bucket:      aws.String(bucket),
    Key:         aws.String(key),
    Body:        bytes.NewReader([]byte(contents)),
    ChecksumAlgorithm: types.ChecksumAlgorithmSha256,
}
output, err := actor.S3Manager.Upload(ctx, input)
if err != nil {
    var noBucket *types.NoSuchBucket
    if errors.As(err, &noBucket) {
        log.Printf("Bucket %s does not exist.\n", bucket)
        err = noBucket
    }
} else {
    err := s3.NewObjectExistsWaiter(actor.S3Client).Wait(ctx, &s3.HeadObjectInput{
        Bucket: aws.String(bucket),
        Key:    aws.String(key),
    }, time.Minute)
    if err != nil {
        log.Printf("Failed attempt to wait for object %s to exist in %s.\n", key,
            bucket)
    } else {
        outKey = *output.Key
    }
}
return outKey, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [PutObject](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用 [S3Client](#) 將文件上傳到儲存貯體。

```
/**
 * Uploads a local file to an AWS S3 bucket asynchronously.
 *
 * @param bucketName the name of the S3 bucket to upload the file to
 * @param key         the key (object name) to use for the uploaded file
 * @param objectPath  the local file path of the file to be uploaded
 * @return a {@link CompletableFuture} that completes with the {@link
 * PutObjectResponse} when the upload is successful, or throws a {@link
 * RuntimeException} if the upload fails
 */
public CompletableFuture<PutObjectResponse> uploadLocalFileAsync(String
bucketName, String key, String objectPath) {
    PutObjectRequest objectRequest = PutObjectRequest.builder()
        .bucket(bucketName)
        .key(key)
        .build();

    CompletableFuture<PutObjectResponse> response =
getAsyncClient().putObject(objectRequest,
AsyncRequestBody.fromFile(Paths.get(objectPath)));
    return response.whenComplete((resp, ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to upload file", ex);
        }
    });
}
```

使用 [S3TransferManager](#) 將[檔案上傳](#)至儲存貯體。檢視[完整檔案](#)並[測試](#)。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedFileUpload;
import software.amazon.awssdk.transfer.s3.model.FileUpload;
import software.amazon.awssdk.transfer.s3.model.UploadFileRequest;
import software.amazon.awssdk.transfer.s3.progress.LoggingTransferListener;
import java.net.URI;
import java.net.URISyntaxException;
import java.net.URL;
```

```
import java.nio.file.Paths;
import java.util.UUID;

    public String uploadFile(S3TransferManager transferManager, String
bucketName,

                            String key, URI filePathURI) {
        UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
            .putObjectRequest(b -> b.bucket(bucketName).key(key))
            .source(Paths.get(filePathURI))
            .build();

        FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);

        CompletedFileUpload uploadResult = fileUpload.completionFuture().join();
        return uploadResult.response().eTag();
    }
```

使用 [S3Client](#) 將物件上傳至儲存貯體並設定索引標籤。

```
/**
 * Puts tags on an Amazon S3 object.
 *
 * @param s3 An {@link S3Client} object that represents the Amazon S3 client.
 * @param bucketName The name of the Amazon S3 bucket.
 * @param objectKey The key of the Amazon S3 object.
 * @param objectPath The file path of the object to be uploaded.
 */
public static void putS3ObjectTags(S3Client s3, String bucketName, String
objectKey, String objectPath) {
    try {
        Tag tag1 = Tag.builder()
            .key("Tag 1")
            .value("This is tag 1")
            .build();

        Tag tag2 = Tag.builder()
            .key("Tag 2")
            .value("This is tag 2")
            .build();

        List<Tag> tags = new ArrayList<>();
        tags.add(tag1);
```

```
tags.add(tag2);

Tagging allTags = Tagging.builder()
    .tagSet(tags)
    .build();

PutObjectRequest putOb = PutObjectRequest.builder()
    .bucket(bucketName)
    .key(objectKey)
    .tagging(allTags)
    .build();

s3.putObject(putOb,
RequestBody.fromBytes(getObjectFile(objectPath)));

    } catch (S3Exception e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
}

/**
 * Updates the tags associated with an object in an Amazon S3 bucket.
 *
 * @param s3 an instance of the S3Client class, which is used to interact
with the Amazon S3 service
 * @param bucketName the name of the S3 bucket containing the object
 * @param objectKey the key (or name) of the object in the S3 bucket
 * @throws S3Exception if there is an error updating the object's tags
 */
public static void updateObjectTags(S3Client s3, String bucketName, String
objectKey) {
    try {
        GetObjectTaggingRequest taggingRequest =
GetObjectTaggingRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .build();

        GetObjectTaggingResponse getTaggingRes =
s3.getObjectTagging(taggingRequest);
        List<Tag> obTags = getTaggingRes.tagSet();
        for (Tag sinTag : obTags) {
            System.out.println("The tag key is: " + sinTag.key());
        }
    }
}
```



```
        System.out.println("The tag value is: " + sinTag.value());
    }

    // Replace the object's tags with two new tags.
    Tag tag3 = Tag.builder()
        .key("Tag 3")
        .value("This is tag 3")
        .build();

    Tag tag4 = Tag.builder()
        .key("Tag 4")
        .value("This is tag 4")
        .build();

    List<Tag> tags = new ArrayList<>();
    tags.add(tag3);
    tags.add(tag4);

    Tagging updatedTags = Tagging.builder()
        .tagSet(tags)
        .build();

    PutObjectTaggingRequest taggingRequest1 =
    PutObjectTaggingRequest.builder()
        .bucket(bucketName)
        .key(objectKey)
        .tagging(updatedTags)
        .build();

    s3.putObjectTagging(taggingRequest1);
    GetObjectTaggingResponse getTaggingRes2 =
    s3.getObjectTagging(taggingRequest);
    List<Tag> modTags = getTaggingRes2.tagSet();
    for (Tag sinTag : modTags) {
        System.out.println("The tag key is: " + sinTag.key());
        System.out.println("The tag value is: " + sinTag.value());
    }

    } catch (S3Exception e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
}
```

```
/**
 * Retrieves the contents of a file as a byte array.
 *
 * @param filePath the path of the file to be read
 * @return a byte array containing the contents of the file, or null if an
 * error occurs
 */
private static byte[] getObjectFile(String filePath) {
    FileInputStream fileInputStream = null;
    byte[] byteArray = null;

    try {
        File file = new File(filePath);
        byteArray = new byte[(int) file.length()];
        fileInputStream = new FileInputStream(file);
        fileInputStream.read(byteArray);

    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        if (fileInputStream != null) {
            try {
                fileInputStream.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    return byteArray;
}
}
```

使用 [S3Client](#) 將物件上傳至儲存貯體並設定中繼資料。

```
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
```

```
import java.io.File;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class PutObjectMetadata {
    public static void main(String[] args) {
        final String USAGE = ""

            Usage:
                <bucketName> <objectKey> <objectPath>\s

            Where:
                bucketName - The Amazon S3 bucket to upload an object into.
                objectKey - The object to upload (for example, book.pdf).
                objectPath - The path where the file is located (for example, C:/
AWS/book2.pdf).\s
            """;

        if (args.length != 3) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String bucketName = args[0];
        String objectKey = args[1];
        String objectPath = args[2];
        System.out.println("Putting object " + objectKey + " into bucket " +
bucketName);
        System.out.println("  in bucket: " + bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        putS3Object(s3, bucketName, objectKey, objectPath);
    }
}
```

```
s3.close();
}

/**
 * Uploads an object to an Amazon S3 bucket with metadata.
 *
 * @param s3 the S3Client object used to interact with the Amazon S3 service
 * @param bucketName the name of the S3 bucket to upload the object to
 * @param objectKey the name of the object to be uploaded
 * @param objectPath the local file path of the object to be uploaded
 */
public static void putS3Object(S3Client s3, String bucketName, String
objectKey, String objectPath) {
    try {
        Map<String, String> metadata = new HashMap<>();
        metadata.put("author", "Mary Doe");
        metadata.put("version", "1.0.0.0");

        PutObjectRequest putOb = PutObjectRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .metadata(metadata)
            .build();

        s3.putObject(putOb, RequestBody.fromFile(new File(objectPath)));
        System.out.println("Successfully placed " + objectKey + " into bucket
" + bucketName);

    } catch (S3Exception e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
}
}
```

使用 [S3Client](#) 將物件上傳至儲存貯體並設定物件保留值。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRetentionRequest;
import software.amazon.awssdk.services.s3.model.ObjectLockRetention;
```

```
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.time.Instant;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.ZoneOffset;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */

public class PutObjectRetention {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <key> <bucketName>\s

            Where:
                key - The name of the object (for example, book.pdf).\s
                bucketName - The Amazon S3 bucket name that contains the object
(for example, bucket1).\s
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String key = args[0];
        String bucketName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        setRetentionPeriod(s3, key, bucketName);
        s3.close();
    }
}
```

```
}

/**
 * Sets the retention period for an object in an Amazon S3 bucket.
 *
 * @param s3      the S3Client object used to interact with the Amazon S3
service
 * @param key     the key (name) of the object in the S3 bucket
 * @param bucket  the name of the S3 bucket where the object is stored
 *
 * @throws S3Exception if an error occurs while setting the object retention
period
 */
public static void setRentionPeriod(S3Client s3, String key, String bucket) {
    try {
        LocalDate localDate = LocalDate.parse("2020-07-17");
        LocalDateTime localDateTime = localDate.atStartOfDay();
        Instant instant = localDateTime.toInstant(ZoneOffset.UTC);

        ObjectLockRetention lockRetention = ObjectLockRetention.builder()
            .mode("COMPLIANCE")
            .retainUntilDate(instant)
            .build();

        PutObjectRetentionRequest retentionRequest =
PutObjectRetentionRequest.builder()
            .bucket(bucket)
            .key(key)
            .bypassGovernanceRetention(true)
            .retention(lockRetention)
            .build();

        // To set Retention on an object, the Amazon S3 bucket must support
object
        // locking, otherwise an exception is thrown.
        s3.putObjectRetention(retentionRequest);
        System.out.print("An object retention configuration was successfully
placed on the object");

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

```
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutObject](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

上傳物件。

```
import { readFile } from "node:fs/promises";

import {
  PutObjectCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Upload a file to an S3 bucket.
 * @param {{ bucketName: string, key: string, filePath: string }}
 */
export const main = async ({ bucketName, key, filePath }) => {
  const client = new S3Client({});
  const command = new PutObjectCommand({
    Bucket: bucketName,
    Key: key,
    Body: await readFile(filePath),
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (caught) {
    if (
```

```

        caught instanceof S3ServiceException &&
        caught.name === "EntityTooLarge"
    ) {
        console.error(
            `Error from S3 while uploading object to ${bucketName}. \
The object was too large. To upload objects larger than 5GB, use the S3 console \
(160GB max) \
or the multipart upload API (5TB max).`,
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while uploading object to ${bucketName}. ${caught.name}: \
${caught.message}`,
        );
    } else {
        throw caught;
    }
}
};

```

如果物件的 ETag 與提供的 ETag 相符，則上傳該物件。

```

import {
    GetObjectCommand,
    NoSuchKey,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Get a single object from a specified S3 bucket.
 * @param {{ bucketName: string, key: string, eTag: string }}
 */
export const main = async ({ bucketName, key, eTag }) => {
    const client = new S3Client({});

    try {
        const response = await client.send(
            new GetObjectCommand({
                Bucket: bucketName,
                Key: key,

```



```
        IfMatch: eTag,
      )),
    );
    // The Body object also has 'transformToByteArray' and 'transformToWebStream'
    methods.
    const str = await response.Body.transformToString();
    console.log("Success. Here is text of the file:", str);
  } catch (caught) {
    if (caught instanceof NoSuchKey) {
      console.error(
        `Error from S3 while getting object "${key}" from "${bucketName}". No
such key exists.`
      );
    } else if (caught instanceof S3ServiceException) {
      console.error(
        `Error from S3 while getting object from ${bucketName}. ${caught.name}:
${caught.message}`
      );
    } else {
      throw caught;
    }
  }
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
  isMain,
  validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
  const options = {
    bucketName: {
      type: "string",
      required: true,
    },
    key: {
      type: "string",
      required: true,
    },
    eTag: {
      type: "string",
      required: true,
    },
  };
};
```

```
    },
  };
  const results = parseArgs({ options });
  const { errors } = validateArgs({ options }, results);
  return { errors, results };
};

if (isMain(import.meta.url)) {
  const { errors, results } = loadArgs();
  if (!errors) {
    main(results.values);
  } else {
    console.error(errors.join("\n"));
  }
}
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [PutObject](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun putS3Object(
    bucketName: String,
    objectKey: String,
    objectPath: String,
) {
    val metadataVal = mutableMapOf<String, String>()
    metadataVal["myVal"] = "test"

    val request =
        PutObjectRequest {
            bucket = bucketName
```

```

        key = objectKey
        metadata = metadataVal
        body = File(objectPath).asByteStream()
    }

    S3Client.fromEnvironment { region = "us-east-1" }.use { s3 ->
        val response = s3.putObject(request)
        println("Tag information is ${response.eTag}")
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [PutObject](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

將物件上傳至儲存貯體。

```

$s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

$fileName = __DIR__ . "/local-file-" . uniqid();
try {
    $this->s3client->putObject([
        'Bucket' => $this->bucketName,
        'Key' => $fileName,
        'SourceFile' => __DIR__ . '/testfile.txt'
    ]);
    echo "Uploaded $fileName to $this->bucketName.\n";
} catch (Exception $exception) {
    echo "Failed to upload $fileName with error: " . $exception-
    >getMessage();
    exit("Please fix error with file upload before continuing.");
}

```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [PutObject](#)。

PowerShell

Tools for PowerShell V4

範例 1：此命令會將單一檔案 "local-sample.txt" 上傳至 Amazon S3，並在儲存貯體 "test-files" 中建立具索引鍵 "sample.txt" 的物件。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "sample.txt" -File .\local-sample.txt
```

範例 2：此命令會將單一檔案 "sample.txt" 上傳至 Amazon S3，並在儲存貯體 "test-files" 中建立具索引鍵 "sample.txt" 的物件。如果未提供 -Key 參數，則會將檔案名稱用作 S3 物件索引鍵。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -File .\sample.txt
```

範例 3：此命令會將單一檔案 "local-sample.txt" 上傳至 Amazon S3，並在儲存貯體 "test-files" 中建立具索引鍵 "prefix/to/sample.txt" 的物件。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "prefix/to/sample.txt" -File .\local-sample.txt
```

範例 4：此命令會將子目錄 "Scripts" 中的所有檔案上傳至儲存貯體 "test-files"，並將通用索引鍵字首 "SampleScripts" 套用至每個物件。每個上傳的檔案都會有一個索引鍵 "SampleScripts/filename"，其中 'filename' 會有所不同。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder .\Scripts -KeyPrefix SampleScripts\
```

範例 5：此命令會將本機目錄 "Scripts" 中的所有 *.ps1 檔案上傳至儲存貯體 "test-files"，並將通用索引鍵字首 "SampleScripts" 套用至每個物件。每個上傳的檔案都會有一個索引鍵 "SampleScripts/filename.ps1"，其中 'filename' 會有所不同。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder .\Scripts -KeyPrefix SampleScripts\ -SearchPattern *.ps1
```

範例 6：此命令會建立新的 S3 物件，其中包含索引鍵 'sample.txt' 的指定內容字串。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "sample.txt" -Content "object contents"
```

範例 7：此命令會上傳指定的檔案 (檔案名稱用作索引鍵)，並將指定的標籤套用至新物件。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -File "sample.txt" -TagSet @{Key="key1";Value="value1"},@{Key="key2";Value="value2"}
```

範例 8：此命令會以遞迴方式上傳指定的資料夾，並將指定的標籤套用至所有新物件。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder . -KeyPrefix "TaggedFiles" -Recurse -TagSet @{Key="key1";Value="value1"},@{Key="key2";Value="value2"}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [PutObject](#)。

Tools for PowerShell V5

範例 1：此命令會將單一檔案 "local-sample.txt" 上傳至 Amazon S3，並在儲存貯體 "test-files" 中建立具索引鍵 "sample.txt" 的物件。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "sample.txt" -File .\local-sample.txt
```

範例 2：此命令會將單一檔案 "sample.txt" 上傳至 Amazon S3，並在儲存貯體 "test-files" 中建立具索引鍵 "sample.txt" 的物件。如果未提供 -Key 參數，則會將檔案名稱用作 S3 物件索引鍵。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -File .\sample.txt
```

範例 3：此命令會將單一檔案 "local-sample.txt" 上傳至 Amazon S3，並在儲存貯體 "test-files" 中建立具索引鍵 "prefix/to/sample.txt" 的物件。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "prefix/to/sample.txt" -File .\local-sample.txt
```

範例 4：此命令會將子目錄 "Scripts" 中的所有檔案上傳至儲存貯體 "test-files"，並將通用索引鍵字首 "SampleScripts" 套用至每個物件。每個上傳的檔案都會有一個索引鍵 "SampleScripts/filename"，其中 'filename' 會有所不同。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder .\Scripts -KeyPrefix  
SampleScripts\
```

範例 5：此命令會將本機目錄 "Scripts" 中的所有 *.ps1 檔案上傳至儲存貯體 "test-files"，並將通用索引鍵字首 "SampleScripts" 套用至每個物件。每個上傳的檔案都會有一個索引鍵 "SampleScripts/filename.ps1"，其中 'filename' 會有所不同。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder .\Scripts -KeyPrefix  
SampleScripts\ -SearchPattern *.ps1
```

範例 6：此命令會建立新的 S3 物件，其中包含索引鍵 'sample.txt' 的指定內容字串。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Key "sample.txt" -Content "object  
contents"
```

範例 7：此命令會上傳指定的檔案 (檔案名稱用作索引鍵)，並將指定的標籤套用至新物件。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -File "sample.txt" -TagSet  
@{Key="key1";Value="value1"},@{Key="key2";Value="value2"}
```

範例 8：此命令會以遞迴方式上傳指定的資料夾，並將指定的標籤套用至所有新物件。

```
Write-S3Object -BucketName amzn-s3-demo-bucket -Folder . -KeyPrefix "TaggedFiles"  
-Recurse -TagSet @{Key="key1";Value="value1"},@{Key="key2";Value="value2"}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [PutObject](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
        in Boto3
                           that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key

    def put(self, data):
        """
        Upload data to the object.

        :param data: The data to upload. This can either be bytes or a string.
        When this
                           argument is a string, it is interpreted as a file name,
        which is
                           opened in read bytes mode.
        """
        put_data = data
        if isinstance(data, str):
            try:
                put_data = open(data, "rb")
            except IOError:
                logger.exception("Expected file name or binary data, got '%s'.",
data)
                raise
```

```
try:
    self.object.put(Body=put_data)
    self.object.wait_until_exists()
    logger.info(
        "Put object '%s' to bucket '%s'.",
        self.object.key,
        self.object.bucket_name,
    )
except ClientError:
    logger.exception(
        "Couldn't put object '%s' to bucket '%s'.",
        self.object.key,
        self.object.bucket_name,
    )
    raise
finally:
    if getattr(put_data, "close", None):
        put_data.close()
```

使用條件式請求上傳物件。

```
class S3ConditionalRequests:
    """Encapsulates S3 conditional request operations."""

    def __init__(self, s3_client):
        self.s3 = s3_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        s3_client = boto3.client("s3")
        return cls(s3_client)

    def put_object_conditional(self, object_key: str, source_bucket: str, data:
bytes):
        """
```



```
Uploads an object to Amazon S3 with a conditional request. Prevents
overwrite
using an IfNoneMatch condition for the object key.

:param object_key: The key of the object to upload.
:param source_bucket: The source bucket of the object.
:param data: The data to upload.
"""
try:
    self.s3.put_object(
        Bucket=source_bucket, Key=object_key, Body=data, IfNoneMatch="*"
    )
    print(
        f"\tConditional write successful for key {object_key} in bucket
{source_bucket}."
    )
except ClientError as e:
    error_code = e.response["Error"]["Code"]
    if error_code == "PreconditionFailed":
        print("\tConditional write failed: Precondition failed")
    else:
        logger.error(f"Unexpected error: {error_code}")
        raise
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [PutObject](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用受管的上傳工具上傳檔案 (Object.upload_file)。

```
require 'aws-sdk-s3'
```

```
# Wraps Amazon S3 object actions.
class ObjectUploadFileWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Uploads a file to an Amazon S3 object by using a managed uploader.
  #
  # @param file_path [String] The path to the file to upload.
  # @return [Boolean] True when the file is uploaded; otherwise false.
  def upload_file(file_path)
    @object.upload_file(file_path)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't upload file #{file_path} to #{@object.key}. Here's why:
#{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "amzn-s3-demo-bucket"
  object_key = "my-uploaded-file"
  file_path = "object_upload_file.rb"

  wrapper = ObjectUploadFileWrapper.new(Aws::S3::Object.new(bucket_name,
object_key))
  return unless wrapper.upload_file(file_path)

  puts "File #{file_path} successfully uploaded to #{bucket_name}:#{object_key}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

使用 `Object.put` 上傳檔案。

```
require 'aws-sdk-s3'
```

```

# Wraps Amazon S3 object actions.
class ObjectPutWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object(source_file_path)
    File.open(source_file_path, 'rb') do |file|
      @object.put(body: file)
    end
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put #{source_file_path} to #{object.key}. Here's why:
#{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "amzn-s3-demo-bucket"
  object_key = "my-object-key"
  file_path = "my-local-file.txt"

  wrapper = ObjectPutWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  success = wrapper.put_object(file_path)
  return unless success

  puts "Put file #{file_path} into #{object_key} in #{bucket_name}."
end

run_demo if $PROGRAM_NAME == __FILE__

```

使用 Object.Pet 上傳檔案並新增伺服器端加密。

```

require 'aws-sdk-s3'

# Wraps Amazon S3 object actions.

```

```
class ObjectPutSseWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object_encrypted(object_content, encryption)
    @object.put(body: object_content, server_side_encryption: encryption)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put your content to #{object.key}. Here's why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "amzn-s3-demo-bucket"
  object_key = "my-encrypted-content"
  object_content = "This is my super-secret content."
  encryption = "AES256"

  wrapper = ObjectPutSseWrapper.new(Aws::S3::Object.new(bucket_name,
    object_content))
  return unless wrapper.put_object_encrypted(object_content, encryption)

  puts "Put your content into #{bucket_name}:#{object_key} and encrypted it with
    #{encryption}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [PutObject](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
pub async fn upload_object(
    client: &aws_sdk_s3::Client,
    bucket_name: &str,
    file_name: &str,
    key: &str,
) -> Result<aws_sdk_s3::operation::put_object::PutObjectOutput, S3ExampleError> {
    let body =
        aws_sdk_s3::primitives::ByteStream::from_path(std::path::Path::new(file_name)).await;
    client
        .put_object()
        .bucket(bucket_name)
        .key(key)
        .body(body.unwrap())
        .send()
        .await
        .map_err(S3ExampleError::from)
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [PutObject](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

"Get contents of file from application server."
DATA lv_body TYPE xstring.
OPEN DATASET iv_file_name FOR INPUT IN BINARY MODE.
READ DATASET iv_file_name INTO lv_body.
CLOSE DATASET iv_file_name.

"Upload/put an object to an S3 bucket."
TRY.
    lo_s3->putobject(
        iv_bucket = iv_bucket_name
        iv_key = iv_file_name
        iv_body = lv_body ).
    MESSAGE 'Object uploaded to S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 SAP ABAP 的 AWS SDK API 參考》中的 [PutObject](#)。

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

import AWSS3
import Smithy

public func uploadFile(bucket: String, key: String, file: String) async
throws {
    let fileUrl = URL(fileURLWithPath: file)
    do {
        let fileData = try Data(contentsOf: fileUrl)
        let dataStream = ByteStream.data(fileData)

```

```
        let input = PutObjectInput(
            body: dataStream,
            bucket: bucket,
            key: key
        )

        _ = try await client.putObject(input: input)
    }
    catch {
        print("ERROR: ", dump(error, name: "Putting an object. "))
        throw error
    }
}
```

```
import AWSS3
import Smithy

public func createFile(bucket: String, key: String, withData data: Data)
async throws {
    let dataStream = ByteStream.data(data)

    let input = PutObjectInput(
        body: dataStream,
        bucket: bucket,
        key: key
    )

    do {
        _ = try await client.putObject(input: input)
    }
    catch {
        print("ERROR: ", dump(error, name: "Putting an object. "))
        throw error
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Swift 的 AWS SDK API 參考》中的 [PutObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutObjectAcl 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutObjectAcl。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [管理存取控制清單 \(ACL\)](#)

C++

SDK for C++

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
bool AwsDoc::S3::putObjectAcl(const Aws::String &bucketName, const Aws::String
    &objectKey, const Aws::String &ownerID,
                                const Aws::String &granteePermission, const
    Aws::String &granteeType,
                                const Aws::String &granteeID, const Aws::String
    &granteeEmailAddress,
                                const Aws::String &granteeURI, const
    Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::Owner owner;
    owner.SetID(ownerID);

    Aws::S3::Model::Grantee grantee;
    grantee.SetType(setGranteeType(granteeType));

    if (!granteeEmailAddress.empty()) {
        grantee.SetEmailAddress(granteeEmailAddress);
    }
}
```



```

    if (!granteeID.empty()) {
        grantee.SetID(granteeID);
    }

    if (!granteeURI.empty()) {
        grantee.SetURI(granteeURI);
    }

    Aws::S3::Model::Grant grant;
    grant.SetGrantee(grantee);
    grant.SetPermission(setGranteePermission(granteePermission));

    Aws::Vector<Aws::S3::Model::Grant> grants;
    grants.push_back(grant);

    Aws::S3::Model::AccessControlPolicy acp;
    acp.SetOwner(owner);
    acp.SetGrants(grants);

    Aws::S3::Model::PutObjectAclRequest request;
    request.SetAccessControlPolicy(acp);
    request.SetBucket(bucketName);
    request.SetKey(objectKey);

    Aws::S3::Model::PutObjectAclOutcome outcome =
        s3Client.PutObjectAcl(request);

    if (!outcome.IsSuccess()) {
        auto error = outcome.GetError();
        std::cerr << "Error: putObjectAcl: " << error.GetExceptionName()
            << " - " << error.GetMessage() << std::endl;
    } else {
        std::cout << "Successfully added an ACL to the object '" << objectKey
            << "' in the bucket '" << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which converts a human-readable string to a built-in type
enumeration.
/*!
\param access: Human readable string.
\return Permission: Permission enumeration.

```

```

*/
Aws::S3::Model::Permission setGranteePermission(const Aws::String &access) {
    if (access == "FULL_CONTROL")
        return Aws::S3::Model::Permission::FULL_CONTROL;
    if (access == "WRITE")
        return Aws::S3::Model::Permission::WRITE;
    if (access == "READ")
        return Aws::S3::Model::Permission::READ;
    if (access == "WRITE_ACP")
        return Aws::S3::Model::Permission::WRITE_ACP;
    if (access == "READ_ACP")
        return Aws::S3::Model::Permission::READ_ACP;
    return Aws::S3::Model::Permission::NOT_SET;
}

//! Routine which converts a human-readable string to a built-in type
enumeration.
/*!
\param type: Human readable string.
\return Type: Type enumeration.
*/
Aws::S3::Model::Type setGranteeType(const Aws::String &type) {
    if (type == "Amazon customer by email")
        return Aws::S3::Model::Type::AmazonCustomerByEmail;
    if (type == "Canonical user")
        return Aws::S3::Model::Type::CanonicalUser;
    if (type == "Group")
        return Aws::S3::Model::Type::Group;
    return Aws::S3::Model::Type::NOT_SET;
}

```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [PutObjectAcl](#)。

CLI

AWS CLI

下列命令 `full control` 會授予兩個 AWS 使用者 (`user1@example.com` 和 `user2@example.com`) 和 `read` 許可給每個人：

```
aws s3api put-object-acl --bucket amzn-s3-demo-bucket --key file.txt --grant-  
full-control emailaddress=user1@example.com,emailaddress=user2@example.com --  
grant-read uri=http://acs.amazonaws.com/groups/global/AllUsers
```

如需自訂 ACL (s3api ACL 命令如 `put-object-acl`，使用相同的速記引數表示法)，請參閱 <http://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTacl.html>。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutObjectAcl](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
        in Boto3
                           that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key

    def put_acl(self, email):
        """
        Applies an ACL to the object that grants read access to an AWS user
        identified
        by email address.

        :param email: The email address of the user to grant access.
        """
        try:
```

```

        acl = self.object.Acl()
        # Putting an ACL overwrites the existing ACL, so append new grants
        # if you want to preserve existing grants.
        grants = acl.grants if acl.grants else []
        grants.append(
            {
                "Grantee": {"Type": "AmazonCustomerByEmail", "EmailAddress":
email},
                "Permission": "READ",
            }
        )
        acl.put(AccessControlPolicy={"Grants": grants, "Owner": acl.owner})
        logger.info("Granted read access to %s.", email)
    except ClientError:
        logger.exception("Couldn't add ACL to object '%s'.", self.object.key)
        raise

```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [PutObjectAcl](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

TRY.

```

" Example: Grant read access to an AWS user
" iv_grantread = 'emailAddress=user@example.com'
lo_s3->putobjectacl(
    iv_bucket = iv_bucket_name
    iv_key = iv_object_key
    iv_grantread = iv_grantread ).
MESSAGE 'Object ACL updated.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.

```

```
MESSAGE 'Bucket does not exist.' TYPE 'E'.  
CATCH /aws1/cx_s3_nosuchkey.  
MESSAGE 'Object key does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutObjectAcl](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutObjectLegalHold 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutObjectLegalHold。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [鎖定 Amazon S3 物件](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>  
/// Set or modify a legal hold on an object in an S3 bucket.  
/// </summary>  
/// <param name="bucketName">The bucket of the object.</param>  
/// <param name="objectKey">The key of the object.</param>  
/// <param name="holdStatus">The On or Off status for the legal hold.</param>  
/// <returns>True if successful.</returns>  
public async Task<bool> ModifyObjectLegalHold(string bucketName,  
    string objectKey, ObjectLockLegalHoldStatus holdStatus)  
{
```

```
try
{
    var request = new PutObjectLegalHoldRequest()
    {
        BucketName = bucketName,
        Key = objectKey,
        LegalHold = new ObjectLockLegalHold()
        {
            Status = holdStatus
        }
    };

    var response = await _amazonS3.PutObjectLegalHoldAsync(request);
    Console.WriteLine($"\\tModified legal hold for {objectKey} in
{bucketName}.");
    return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}
catch (AmazonS3Exception ex)
{
    Console.WriteLine($"\\tError modifying legal hold: '{ex.Message}'");
    return false;
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutObjectLegalHold](#)。

CLI

AWS CLI

將法務保存套用至物件

下列 `put-object-legal-hold` 範例會在物件 `doc1.rtf` 上設定法務保存：

```
aws s3api put-object-legal-hold \
  --bucket amzn-s3-demo-bucket-with-object-lock \
  --key doc1.rtf \
  --legal-hold Status=ON
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutObjectLegalHold](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (
    "bytes"
    "context"
    "errors"
    "fmt"
    "log"
    "time"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"
    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/aws-sdk-go-v2/service/s3/types"
    "github.com/aws/smithy-go"
)

// S3Actions wraps S3 service actions.
type S3Actions struct {
    S3Client    *s3.Client
    S3Manager   *manager.Uploader
}

// PutObjectLegalHold sets the legal hold configuration for an S3 object.
func (actor S3Actions) PutObjectLegalHold(ctx context.Context, bucket string, key
    string, versionId string, legalHoldStatus types.ObjectLockLegalHoldStatus) error
{
    input := &s3.PutObjectLegalHoldInput{
        Bucket: aws.String(bucket),
```

```
    Key:      aws.String(key),
    LegalHold: &types.ObjectLockLegalHold{
        Status: legalHoldStatus,
    },
}
if versionId != "" {
    input.VersionId = aws.String(versionId)
}

_, err := actor.S3Client.PutObjectLegalHold(ctx, input)
if err != nil {
    var noKey *types.NoSuchKey
    if errors.As(err, &noKey) {
        log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
        err = noKey
    }
}

return err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [PutObjectLegalHold](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
// Set or modify a legal hold on an object in an S3 bucket.
public void modifyObjectLegalHold(String bucketName, String objectKey,
boolean legalHoldOn) {
    ObjectLockLegalHold legalHold ;
    if (legalHoldOn) {
```



```
        legalHold = ObjectLockLegalHold.builder()
            .status(ObjectLockLegalHoldStatus.ON)
            .build();
    } else {
        legalHold = ObjectLockLegalHold.builder()
            .status(ObjectLockLegalHoldStatus.OFF)
            .build();
    }

    PutObjectLegalHoldRequest legalHoldRequest =
    PutObjectLegalHoldRequest.builder()
        .bucket(bucketName)
        .key(objectKey)
        .legalHold(legalHold)
        .build();

    getClient().putObjectLegalHold(legalHoldRequest) ;
    System.out.println("Modified legal hold for " + objectKey + " in
    "+bucketName +".");
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutObjectLegalHold](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import {
    PutObjectLegalHoldCommand,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Apply a legal hold configuration to the specified object.
```

```
* @param {{ bucketName: string, objectKey: string, legalHoldStatus: "ON" |
"OFF" }}
*/
export const main = async ({ bucketName, objectKey, legalHoldStatus }) => {
  if (!["OFF", "ON"].includes(legalHoldStatus.toUpperCase())) {
    throw new Error(
      "Invalid parameter. legalHoldStatus must be 'ON' or 'OFF'.",
    );
  }

  const client = new S3Client({});
  const command = new PutObjectLegalHoldCommand({
    Bucket: bucketName,
    Key: objectKey,
    LegalHold: {
      // Set the status to 'ON' to place a legal hold on the object.
      // Set the status to 'OFF' to remove the legal hold.
      Status: legalHoldStatus,
    },
  });

  try {
    await client.send(command);
    console.log(
      `Legal hold status set to "${legalHoldStatus}" for "${objectKey}" in
"${bucketName}"`,
    );
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
      console.error(
        `Error from S3 while modifying legal hold status for "${objectKey}" in
"${bucketName}". The bucket doesn't exist.`,
      );
    } else if (caught instanceof S3ServiceException) {
      console.error(
        `Error from S3 while modifying legal hold status for "${objectKey}" in
"${bucketName}". ${caught.name}: ${caught.message}`,
      );
    } else {
      throw caught;
    }
  }
}
```

```
    }  
  };  
  
  // Call function if run directly  
  import { parseArgs } from "node:util";  
  import {  
    isMain,  
    validateArgs,  
  } from "@aws-doc-sdk-examples/lib/utills/util-node.js";  
  
  const loadArgs = () => {  
    const options = {  
      bucketName: {  
        type: "string",  
        required: true,  
      },  
      objectKey: {  
        type: "string",  
        required: true,  
      },  
      legalHoldStatus: {  
        type: "string",  
        default: "ON",  
      },  
    };  
  };  
  const results = parseArgs({ options });  
  const { errors } = validateArgs({ options }, results);  
  return { errors, results };  
};  
  
if (isMain(import.meta.url)) {  
  const { errors, results } = loadArgs();  
  if (!errors) {  
    main(results.values);  
  } else {  
    console.error(errors.join("\n"));  
  }  
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [PutObjectLegalHold](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

放置物件法務保存。

```
def set_legal_hold(s3_client, bucket: str, key: str) -> None:
    """
    Set a legal hold on a specific file in a bucket.

    Args:
        s3_client: Boto3 S3 client.
        bucket: The name of the bucket containing the file.
        key: The key of the file to set the legal hold on.
    """
    print()
    logger.info("Setting legal hold on file [%s] in bucket [%s]", key, bucket)
    try:
        before_status = "OFF"
        after_status = "ON"
        s3_client.put_object_legal_hold(
            Bucket=bucket, Key=key, LegalHold={"Status": after_status}
        )
        logger.debug(
            "Legal hold set successfully on file [%s] in bucket [%s]", key,
            bucket
        )
        _print_legal_hold_update(bucket, key, before_status, after_status)
    except Exception as e:
        logger.error(
            "Failed to set legal hold on file [%s] in bucket [%s]: %s", key,
            bucket, e
        )
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [PutObjectLegalHold](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
  " Example: Set legal hold status to ON  
  " iv_status = 'ON'  
  lo_s3->putobjectlegalhold(  
    iv_bucket = iv_bucket_name  
    iv_key = iv_object_key  
    io_legalhold = NEW /aws1/cl_s3_objlocklegalhold(  
      iv_status = iv_status ) ).  
  MESSAGE 'Object legal hold status set.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
  MESSAGE 'Bucket does not exist.' TYPE 'E'.  
CATCH /aws1/cx_s3_nosuchkey.  
  MESSAGE 'Object key does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutObjectLegalHold](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutObjectLockConfiguration 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutObjectLockConfiguration。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [鎖定 Amazon S3 物件](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

設定儲存貯體的物件鎖定組態。

```
/// <summary>
/// Enable object lock on an existing bucket.
/// </summary>
/// <param name="bucketName">The name of the bucket to modify.</param>
/// <returns>True if successful.</returns>
public async Task<bool> EnableObjectLockOnBucket(string bucketName)
{
    try
    {
        // First, enable Versioning on the bucket.
        await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
        {
            BucketName = bucketName,
            VersioningConfig = new S3BucketVersioningConfig()
            {
                EnableMfaDelete = false,
                Status = VersionStatus.Enabled
            }
        });

        var request = new PutObjectLockConfigurationRequest()
        {
            BucketName = bucketName,
            ObjectLockConfiguration = new ObjectLockConfiguration()
```

```

        {
            ObjectLockEnabled = new ObjectLockEnabled("Enabled"),
        },
    };

    var response = await
    _amazonS3.PutObjectLockConfigurationAsync(request);
    Console.WriteLine($"{"\tAdded an object lock policy to bucket
{bucketName}."});
    return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}
catch (AmazonS3Exception ex)
{
    Console.WriteLine($"Error modifying object lock: '{ex.Message}'");
    return false;
}
}

```

設定儲存貯體的預設保留期間。

```

/// <summary>
/// Set or modify a retention period on an S3 bucket.
/// </summary>
/// <param name="bucketName">The bucket to modify.</param>
/// <param name="retention">The retention mode.</param>
/// <param name="retainUntilDate">The date for retention until.</param>
/// <returns>True if successful.</returns>
public async Task<bool> ModifyBucketDefaultRetention(string bucketName, bool
enableObjectLock, ObjectLockRetentionMode retention, DateTime retainUntilDate)
{
    var enabledString = enableObjectLock ? "Enabled" : "Disabled";
    var timeDifference = retainUntilDate.Subtract(DateTime.Now);
    try
    {
        // First, enable Versioning on the bucket.
        await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
        {
            BucketName = bucketName,
            VersioningConfig = new S3BucketVersioningConfig()
            {
                EnableMfaDelete = false,
            }
        }
    }
}

```

```
        Status = VersionStatus.Enabled
    }
});

var request = new PutObjectLockConfigurationRequest()
{
    BucketName = bucketName,
    ObjectLockConfiguration = new ObjectLockConfiguration()
    {
        ObjectLockEnabled = new ObjectLockEnabled(enabledString),
        Rule = new ObjectLockRule()
        {
            DefaultRetention = new DefaultRetention()
            {
                Mode = retention,
                Days = timeDifference.Days // Can be specified in
days or years but not both.
            }
        }
    }
};

var response = await
_amazonS3.PutObjectLockConfigurationAsync(request);
Console.WriteLine($"{\tAdded a default retention to bucket
{bucketName}.");
return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}
catch (AmazonS3Exception ex)
{
    Console.WriteLine($"{\tError modifying object lock: '{ex.Message}'");
    return false;
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutObjectLockConfiguration](#)。

CLI

AWS CLI

在儲存貯體上設定物件鎖定組態

下列 `put-object-lock-configuration` 範例會在指定的儲存貯體上設定 50 天物件鎖定。

```
aws s3api put-object-lock-configuration \
  --bucket amzn-s3-demo-bucket-with-object-lock \
  --object-lock-configuration '{ "ObjectLockEnabled": "Enabled", "Rule":
  { "DefaultRetention": { "Mode": "COMPLIANCE", "Days": 50 }}}'
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutObjectLockConfiguration](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

設定儲存貯體的物件鎖定組態。

```
import (
    "bytes"
    "context"
    "errors"
    "fmt"
    "log"
    "time"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"
    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/aws-sdk-go-v2/service/s3/types"
)
```

```
"github.com/aws/smithy-go"
)

// S3Actions wraps S3 service actions.
type S3Actions struct {
    S3Client    *s3.Client
    S3Manager   *manager.Uploader
}

// EnableObjectLockOnBucket enables object locking on an existing bucket.
func (actor S3Actions) EnableObjectLockOnBucket(ctx context.Context, bucket
    string) error {
    // Versioning must be enabled on the bucket before object locking is enabled.
    verInput := &s3.PutBucketVersioningInput{
        Bucket: aws.String(bucket),
        VersioningConfiguration: &types.VersioningConfiguration{
            MFADelete: types.MFADeleteDisabled,
            Status:    types.BucketVersioningStatusEnabled,
        },
    }
    _, err := actor.S3Client.PutBucketVersioning(ctx, verInput)
    if err != nil {
        var noBucket *types.NoSuchBucket
        if errors.As(err, &noBucket) {
            log.Printf("Bucket %s does not exist.\n", bucket)
            err = noBucket
        }
        return err
    }

    input := &s3.PutObjectLockConfigurationInput{
        Bucket: aws.String(bucket),
        ObjectLockConfiguration: &types.ObjectLockConfiguration{
            ObjectLockEnabled: types.ObjectLockEnabledEnabled,
        },
    }
    _, err = actor.S3Client.PutObjectLockConfiguration(ctx, input)
    if err != nil {
        var noBucket *types.NoSuchBucket
        if errors.As(err, &noBucket) {
            log.Printf("Bucket %s does not exist.\n", bucket)
            err = noBucket
        }
    }
}
```

```
    }  
  }  
  
  return err  
}
```

設定儲存貯體的預設保留期間。

```
import (  
  "bytes"  
  "context"  
  "errors"  
  "fmt"  
  "log"  
  "time"  
  
  "github.com/aws/aws-sdk-go-v2/aws"  
  "github.com/aws/aws-sdk-go-v2/feature/s3/manager"  
  "github.com/aws/aws-sdk-go-v2/service/s3"  
  "github.com/aws/aws-sdk-go-v2/service/s3/types"  
  "github.com/aws/smithy-go"  
)  
  
// S3Actions wraps S3 service actions.  
type S3Actions struct {  
  S3Client    *s3.Client  
  S3Manager   *manager.Uploader  
}  
  
// ModifyDefaultBucketRetention modifies the default retention period of an  
// existing bucket.  
func (actor S3Actions) ModifyDefaultBucketRetention(  
  ctx context.Context, bucket string, lockMode types.ObjectLockEnabled,  
  retentionPeriod int32, retentionMode types.ObjectLockRetentionMode) error {  
  
  input := &s3.PutObjectLockConfigurationInput{  
    Bucket: aws.String(bucket),  
    ObjectLockConfiguration: &types.ObjectLockConfiguration{
```

```
ObjectLockEnabled: lockMode,
Rule: &types.ObjectLockRule{
    DefaultRetention: &types.DefaultRetention{
        Days: aws.Int32(retentionPeriod),
        Mode: retentionMode,
    },
},
},
}
_, err := actor.S3Client.PutObjectLockConfiguration(ctx, input)
if err != nil {
    var noBucket *types.NoSuchBucket
    if errors.As(err, &noBucket) {
        log.Printf("Bucket %s does not exist.\n", bucket)
        err = noBucket
    }
}

return err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [PutObjectLockConfiguration](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

設定儲存貯體的物件鎖定組態。

```
// Enable object lock on an existing bucket.
public void enableObjectLockOnBucket(String bucketName) {
    try {
```

```

        VersioningConfiguration versioningConfiguration =
VersioningConfiguration.builder()
            .status(BucketVersioningStatus.ENABLED)
            .build();

        PutBucketVersioningRequest putBucketVersioningRequest =
PutBucketVersioningRequest.builder()
            .bucket(bucketName)
            .versioningConfiguration(versioningConfiguration)
            .build();

        // Enable versioning on the bucket.
        getClient().putBucketVersioning(putBucketVersioningRequest);
        PutObjectLockConfigurationRequest request =
PutObjectLockConfigurationRequest.builder()
            .bucket(bucketName)
            .objectLockConfiguration(ObjectLockConfiguration.builder()
                .objectLockEnabled(ObjectLockEnabled.ENABLED)
                .build())
            .build();

        getClient().putObjectLockConfiguration(request);
        System.out.println("Successfully enabled object lock on
"+bucketName);

    } catch (S3Exception ex) {
        System.out.println("Error modifying object lock: '" + ex.getMessage()
+ "'");
    }
}

```

設定儲存貯體的預設保留期間。

```

// Set or modify a retention period on an S3 bucket.
public void modifyBucketDefaultRetention(String bucketName) {
    VersioningConfiguration versioningConfiguration =
VersioningConfiguration.builder()
        .mfaDelete(MFADelete.DISABLED)
        .status(BucketVersioningStatus.ENABLED)
        .build();

```

```
PutBucketVersioningRequest versioningRequest =
PutBucketVersioningRequest.builder()
    .bucket(bucketName)
    .versioningConfiguration(versioningConfiguration)
    .build();

getClient().putBucketVersioning(versioningRequest);
DefaultRetention rention = DefaultRetention.builder()
    .days(1)
    .mode(ObjectLockRetentionMode.GOVERNANCE)
    .build();

ObjectLockRule lockRule = ObjectLockRule.builder()
    .defaultRetention(rention)
    .build();

ObjectLockConfiguration objectLockConfiguration =
ObjectLockConfiguration.builder()
    .objectLockEnabled(ObjectLockEnabled.ENABLED)
    .rule(lockRule)
    .build();

PutObjectLockConfigurationRequest putObjectLockConfigurationRequest =
PutObjectLockConfigurationRequest.builder()
    .bucket(bucketName)
    .objectLockConfiguration(objectLockConfiguration)
    .build();

getClient().putObjectLockConfiguration(putObjectLockConfigurationRequest) ;
System.out.println("Added a default retention to bucket "+bucketName
+ ".");
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutObjectLockConfiguration](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

設定儲存貯體的物件鎖定組態。

```
import {
  PutObjectLockConfigurationCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Enable S3 Object Lock for an Amazon S3 bucket.
 * After you enable Object Lock on a bucket, you can't
 * disable Object Lock or suspend versioning for that bucket.
 * @param {{ bucketName: string, enabled: boolean }}
 */
export const main = async ({ bucketName }) => {
  const client = new S3Client({});
  const command = new PutObjectLockConfigurationCommand({
    Bucket: bucketName,
    // The Object Lock configuration that you want to apply to the specified
    bucket.
    ObjectLockConfiguration: {
      ObjectLockEnabled: "Enabled",
    },
  });

  try {
    await client.send(command);
    console.log(`Object Lock for "${bucketName}" enabled.`);
  } catch (caught) {
    if (
      caught instanceof S3ServiceException &&
      caught.name === "NoSuchBucket"
    ) {
```

```
        console.error(
            `Error from S3 while modifying the object lock configuration for the
            bucket "${bucketName}". The bucket doesn't exist.`,
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while modifying the object lock configuration for the
            bucket "${bucketName}". ${caught.name}: ${caught.message}`,
        );
    } else {
        throw caught;
    }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
    isMain,
    validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
    const options = {
        bucketName: {
            type: "string",
            required: true,
        },
    };
};

const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
return { errors, results };
};

if (isMain(import.meta.url)) {
    const { errors, results } = loadArgs();
    if (!errors) {
        main(results.values);
    } else {
        console.error(errors.join("\n"));
    }
}
```


設定儲存貯體的預設保留期間。

```
import {
  PutObjectLockConfigurationCommand,
  S3Client,
  S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Change the default retention settings for an object in an Amazon S3 bucket.
 * @param {{ bucketName: string, retentionDays: string }}
 */
export const main = async ({ bucketName, retentionDays }) => {
  const client = new S3Client({});

  try {
    await client.send(
      new PutObjectLockConfigurationCommand({
        Bucket: bucketName,
        // The Object Lock configuration that you want to apply to the specified
        bucket.
        ObjectLockConfiguration: {
          ObjectLockEnabled: "Enabled",
          Rule: {
            // The default Object Lock retention mode and period that you want to
            apply
            // to new objects placed in the specified bucket. Bucket settings
            require
            // both a mode and a period. The period can be either Days or Years
            but
            // you must select one.
            DefaultRetention: {
              // In governance mode, users can't overwrite or delete an object
              version
              // or alter its lock settings unless they have special permissions.
              With
              // governance mode, you protect objects against being deleted by
              most users,
              // but you can still grant some users permission to alter the
              retention settings
              // or delete the objects if necessary.
              Mode: "GOVERNANCE",
              Days: Number.parseInt(retentionDays),
```

```

        },
        },
    },
  )),
);
console.log(
  `Set default retention mode to "GOVERNANCE" with a retention period of
  ${retentionDays} day(s).`,
);
} catch (caught) {
  if (
    caught instanceof S3ServiceException &&
    caught.name === "NoSuchBucket"
  ) {
    console.error(
      `Error from S3 while setting the default object retention for a bucket.
      The bucket doesn't exist.`,
    );
  } else if (caught instanceof S3ServiceException) {
    console.error(
      `Error from S3 while setting the default object retention for a bucket.
      ${caught.name}: ${caught.message}`,
    );
  } else {
    throw caught;
  }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
  isMain,
  validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";

const loadArgs = () => {
  const options = {
    bucketName: {
      type: "string",
      required: true,
    },
    retentionDays: {
      type: "string",

```

```
        required: true,
    },
};
const results = parseArgs({ options });
const { errors } = validateArgs({ options }, results);
return { errors, results };
};

if (isMain(import.meta.url)) {
    const { errors, results } = loadArgs();
    if (!errors) {
        main(results.values);
    } else {
        console.error(errors.join("\n"));
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [PutObjectLockConfiguration](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

放置物件鎖定組態。

```
s3_client.put_object_lock_configuration(
    Bucket=bucket,
    ObjectLockConfiguration={"ObjectLockEnabled": "Disabled", "Rule":
{}}),
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [PutObjectLockConfiguration](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
  " Example: Enable object lock with default retention  
  " iv_enabled = 'Enabled'  
  lo_s3->putobjectlockconfiguration(  
    iv_bucket = iv_bucket_name  
    io_objectlockconfiguration = NEW /aws1/cl_s3_objectlockconf(  
      iv_objectlockenabled = iv_enabled ) ).  
  MESSAGE 'Object lock configuration set.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
  MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutObjectLockConfiguration](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutObjectRetention 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutObjectRetention。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [鎖定 Amazon S3 物件](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/// <summary>
/// Set or modify a retention period on an object in an S3 bucket.
/// </summary>
/// <param name="bucketName">The bucket of the object.</param>
/// <param name="objectKey">The key of the object.</param>
/// <param name="retention">The retention mode.</param>
/// <param name="retainUntilDate">The date retention expires.</param>
/// <returns>True if successful.</returns>
public async Task<bool> ModifyObjectRetentionPeriod(string bucketName,
    string objectKey, ObjectLockRetentionMode retention, DateTime
retainUntilDate)
{
    try
    {
        var request = new PutObjectRetentionRequest()
        {
            BucketName = bucketName,
            Key = objectKey,
            Retention = new ObjectLockRetention()
            {
                Mode = retention,
                RetainUntilDate = retainUntilDate
            }
        };

        var response = await _amazonS3.PutObjectRetentionAsync(request);
        Console.WriteLine($"{objectKey} in {bucketName}
until {retainUntilDate:d}.");
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {

```

```
        Console.WriteLine($"\\tError modifying retention period:
'{ex.Message}');
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutObjectRetention](#)。

CLI

AWS CLI

設定物件的物件保留組態

下列 put-object-retention 範例會設定指定物件的物件保留組態，直到 2025-01-01。

```
aws s3api put-object-retention \
  --bucket amzn-s3-demo-bucket-with-object-lock \
  --key doc1.rtf \
  --retention '{ "Mode": "GOVERNANCE", "RetainUntilDate":
"2025-01-01T00:00:00" }'
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutObjectRetention](#)。

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import (
    "bytes"
```

```
"context"
"errors"
"fmt"
"log"
"time"

"github.com/aws/aws-sdk-go-v2/aws"
"github.com/aws/aws-sdk-go-v2/feature/s3/manager"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/aws/aws-sdk-go-v2/service/s3/types"
"github.com/aws/smithy-go"
)

// S3Actions wraps S3 service actions.
type S3Actions struct {
    S3Client    *s3.Client
    S3Manager   *manager.Uploader
}

// PutObjectRetention sets the object retention configuration for an S3 object.
func (actor S3Actions) PutObjectRetention(ctx context.Context, bucket string, key
    string, retentionMode types.ObjectLockRetentionMode, retentionPeriodDays int32)
    error {
    input := &s3.PutObjectRetentionInput{
        Bucket: aws.String(bucket),
        Key:    aws.String(key),
        Retention: &types.ObjectLockRetention{
            Mode:          retentionMode,
            RetainUntilDate: aws.Time(time.Now().AddDate(0, 0, int(retentionPeriodDays))),
        },
        BypassGovernanceRetention: aws.Bool(true),
    }

    _, err := actor.S3Client.PutObjectRetention(ctx, input)
    if err != nil {
        var noKey *types.NoSuchKey
        if errors.As(err, &noKey) {
            log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
            err = noKey
        }
    }
}
```

```
    return err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 [PutObjectRetention](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
// Set or modify a retention period on an object in an S3 bucket.
public void modifyObjectRetentionPeriod(String bucketName, String objectKey)
{
    // Calculate the instant one day from now.
    Instant futureInstant = Instant.now().plus(1, ChronoUnit.DAYS);

    // Convert the Instant to a ZonedDateTime object with a specific time
    zone.
    ZonedDateTime zonedDateTime =
futureInstant.atZone(ZoneId.systemDefault());

    // Define a formatter for human-readable output.
    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd
HH:mm:ss");

    // Format the ZonedDateTime object to a human-readable date string.
    String humanReadableDate = formatter.format(zonedDateTime);

    // Print the formatted date string.
    System.out.println("Formatted Date: " + humanReadableDate);
    ObjectLockRetention retention = ObjectLockRetention.builder()
        .mode(ObjectLockRetentionMode.GOVERNANCE)
        .retainUntilDate(futureInstant)
```



```
        .build();

        PutObjectRetentionRequest retentionRequest =
        PutObjectRetentionRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .retention(retention)
            .build();

        getClient().putObjectRetention(retentionRequest);
        System.out.println("Set retention for "+objectKey +" in " +bucketName +"
until "+ humanReadableDate +".");
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutObjectRetention](#)。

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import {
    PutObjectRetentionCommand,
    S3Client,
    S3ServiceException,
} from "@aws-sdk/client-s3";

/**
 * Place a 24-hour retention period on an object in an Amazon S3 bucket.
 * @param {{ bucketName: string, key: string }}
 */
export const main = async ({ bucketName, key }) => {
    const client = new S3Client({});
    const command = new PutObjectRetentionCommand({
        Bucket: bucketName,
        Key: key,
```

```
BypassGovernanceRetention: false,
Retention: {
    // In governance mode, users can't overwrite or delete an object version
    // or alter its lock settings unless they have special permissions. With
    // governance mode, you protect objects against being deleted by most
    users,
    // but you can still grant some users permission to alter the retention
    settings
    // or delete the objects if necessary.
    Mode: "GOVERNANCE",
    RetainUntilDate: new Date(new Date().getTime() + 24 * 60 * 60 * 1000),
},
});

try {
    await client.send(command);
    console.log("Object Retention settings updated.");
} catch (caught) {
    if (
        caught instanceof S3ServiceException &&
        caught.name === "NoSuchBucket"
    ) {
        console.error(
            `Error from S3 while modifying the governance mode and retention period
on an object. The bucket doesn't exist.`
        );
    } else if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while modifying the governance mode and retention period
on an object. ${caught.name}: ${caught.message}`
        );
    } else {
        throw caught;
    }
}
};

// Call function if run directly
import { parseArgs } from "node:util";
import {
    isMain,
    validateArgs,
} from "@aws-doc-sdk-examples/lib/utils/util-node.js";
```

```
const loadArgs = () => {
  const options = {
    bucketName: {
      type: "string",
      required: true,
    },
    key: {
      type: "string",
      required: true,
    },
  };
  const results = parseArgs({ options });
  const { errors } = validateArgs({ options }, results);
  return { errors, results };
};

if (isMain(import.meta.url)) {
  const { errors, results } = loadArgs();
  if (!errors) {
    main(results.values);
  } else {
    console.error(errors.join("\n"));
  }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [PutObjectRetention](#)。

PowerShell

Tools for PowerShell V4

範例 1：命令會針對給定 S3 儲存貯體中的 'testfile.txt' 物件啟用控管保留模式，直到「2019 年 12 月 31 日 00:00:00」為止。

```
Write-S3ObjectRetention -BucketName 'amzn-s3-demo-bucket' -Key 'testfile.txt' -
Retention_Mode GOVERNANCE -Retention_RetainUntilDate "2019-12-31T00:00:00"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [PutObjectRetention](#)。

Tools for PowerShell V5

範例 1：命令會針對給定 S3 儲存貯體中的 'testfile.txt' 物件啟用控管保留模式，直到「2019 年 12 月 31 日 00:00:00」為止。

```
Write-S3ObjectRetention -BucketName 'amzn-s3-demo-bucket' -Key 'testfile.txt' -  
Retention_Mode GOVERNANCE -Retention_RetainUntilDate "2019-12-31T00:00:00"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [PutObjectRetention](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

放置物件保留。

```
s3_client.put_object_retention(  
    Bucket=bucket,  
    Key=key,  
    VersionId=version_id,  
    Retention={"Mode": "GOVERNANCE", "RetainUntilDate":  
far_future_date},  
    BypassGovernanceRetention=True,  
)
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [PutObjectRetention](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
  " Example: Set retention mode to GOVERNANCE for 30 days  
  " iv_mode = 'GOVERNANCE'  
  " iv_retain_date should be a timestamp in the future  
  lo_s3->putobjectretention(  
    iv_bucket = iv_bucket_name  
    iv_key = iv_object_key  
    io_retention = NEW /aws1/cl_s3_objectlockret(  
      iv_mode = iv_mode  
      iv_retainuntildate = iv_retain_date )  
    iv_bypassgovernanceretention = abap_true ).  
  MESSAGE 'Object retention set.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
  MESSAGE 'Bucket does not exist.' TYPE 'E'.  
CATCH /aws1/cx_s3_nosuchkey.  
  MESSAGE 'Object key does not exist.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutObjectRetention](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

RestoreObject 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `RestoreObject`。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to restore an archived object in an Amazon
/// Simple Storage Service (Amazon S3) bucket.
/// </summary>
public class RestoreArchivedObject
{
    public static void Main()
    {
        string bucketName = "amzn-s3-demo-bucket";
        string objectKey = "archived-object.txt";

        // Specify your bucket region (an example region is shown).
        RegionEndpoint bucketRegion = RegionEndpoint.USSouth1;

        IAmazonS3 client = new AmazonS3Client(bucketRegion);
        RestoreObjectAsync(client, bucketName, objectKey).Wait();
    }

    /// <summary>
    /// This method restores an archived object from an Amazon S3 bucket.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// RestoreObjectAsync.</param>
    /// <param name="bucketName">A string representing the name of the
    /// bucket where the object was located before it was archived.</param>
```

```

    /// <param name="objectKey">A string representing the name of the
    /// archived object to restore.</param>
    public static async Task RestoreObjectAsync(IAmazonS3 client, string
bucketName, string objectKey)
    {
        try
        {
            var restoreRequest = new RestoreObjectRequest
            {
                BucketName = bucketName,
                Key = objectKey,
                Days = 2,
            };
            RestoreObjectResponse response = await
client.RestoreObjectAsync(restoreRequest);

            // Check the status of the restoration.
            await CheckRestorationStatusAsync(client, bucketName, objectKey);
        }
        catch (AmazonS3Exception amazonS3Exception)
        {
            Console.WriteLine($"Error: {amazonS3Exception.Message}");
        }
    }

    /// <summary>
    /// This method retrieves the status of the object's restoration.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// GetObjectMetadataAsync.</param>
    /// <param name="bucketName">A string representing the name of the Amazon
    /// S3 bucket which contains the archived object.</param>
    /// <param name="objectKey">A string representing the name of the
    /// archived object you want to restore.</param>
    public static async Task CheckRestorationStatusAsync(IAmazonS3 client,
string bucketName, string objectKey)
    {
        GetObjectMetadataRequest metadataRequest = new
GetObjectMetadataRequest()
        {
            BucketName = bucketName,
            Key = objectKey,
        };
    }

```

```
        GetObjectMetadataResponse response = await
client.GetObjectMetadataAsync(metadataRequest);

        var restStatus = response.RestoreInProgress ? "in-progress" :
"finished or failed";
        Console.WriteLine($"Restoration status: {restStatus}");
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [RestoreObject](#)。

CLI

AWS CLI

建立物件的還原請求

下列 `restore-object` 範例會還原儲存貯體 `my-glacier-bucket` 的指定 Amazon S3 Glacier 物件 10 天。

```
aws s3api restore-object \
  --bucket my-glacier-bucket \
  --key doc1.rtf \
  --restore-request Days=10
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [RestoreObject](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。


```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.RestoreRequest;
import software.amazon.awssdk.services.s3.model.GlacierJobParameters;
import software.amazon.awssdk.services.s3.model.RestoreObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.Tier;

/*
 * For more information about restoring an object, see "Restoring an archived
 * object" at
 * https://docs.aws.amazon.com/AmazonS3/latest/userguide/restoring-objects.html
 *
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class RestoreObject {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <bucketName> <keyName> <expectedBucketOwner>

            Where:
                bucketName - The Amazon S3 bucket name.\s
                keyName - The key name of an object with a Storage class value of
                Glacier.\s
                expectedBucketOwner - The account that owns the bucket (you can
                obtain this value from the AWS Management Console).\s
            "";

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
```

```
String keyName = args[1];
String expectedBucketOwner = args[2];
Region region = Region.US_EAST_1;
S3Client s3 = S3Client.builder()
    .region(region)
    .build();

restoreS3Object(s3, bucketName, keyName, expectedBucketOwner);
s3.close();
}

/**
 * Restores an S3 object from the Glacier storage class.
 *
 * @param s3          an instance of the {@link S3Client} to be used
for interacting with Amazon S3
 * @param bucketName  the name of the S3 bucket where the object is
stored
 * @param keyName      the key (object name) of the S3 object to be
restored
 * @param expectedBucketOwner the AWS account ID of the expected bucket
owner
 */
public static void restoreS3Object(S3Client s3, String bucketName, String
keyName, String expectedBucketOwner) {
    try {
        RestoreRequest restoreRequest = RestoreRequest.builder()
            .days(10)

            .glacierJobParameters(GlacierJobParameters.builder().tier(Tier.STANDARD).build())
            .build();

        RestoreObjectRequest objectRequest = RestoreObjectRequest.builder()
            .expectedBucketOwner(expectedBucketOwner)
            .bucket(bucketName)
            .key(keyName)
            .restoreRequest(restoreRequest)
            .build();

        s3.restoreObject(objectRequest);
    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

```
    }  
  }  
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [RestoreObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

SelectObjectContent 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 SelectObjectContent。

CLI

AWS CLI

根據 SQL 陳述式篩選 Amazon S3 物件的內容

下列 select-object-content 範例會使用指定的 SQL 陳述式篩選物件 my-data-file.csv，並將輸出傳送至檔案。

```
aws s3api select-object-content \  
  --bucket amzn-s3-demo-bucket \  
  --key my-data-file.csv \  
  --expression "select * from s3object limit 100" \  
  --expression-type 'SQL' \  
  --input-serialization '{"CSV": {}, "CompressionType": "NONE"}' \  
  --output-serialization '{"CSV": {}}' "output.csv"
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [SelectObjectContent](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

下列範例示範使用 JSON 物件的查詢。[完整範例](#) 也會示範 CSV 物件的使用方式。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.async.AsyncRequestBody;
import software.amazon.awssdk.core.async.BlockingInputStreamAsyncRequestBody;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.services.s3.S3AsyncClient;
import software.amazon.awssdk.services.s3.model.CSVInput;
import software.amazon.awssdk.services.s3.model.CSVOutput;
import software.amazon.awssdk.services.s3.model.CompressionType;
import software.amazon.awssdk.services.s3.model.ExpressionType;
import software.amazon.awssdk.services.s3.model.FileHeaderInfo;
import software.amazon.awssdk.services.s3.model.InputSerialization;
import software.amazon.awssdk.services.s3.model.JSONInput;
import software.amazon.awssdk.services.s3.model.JSONOutput;
import software.amazon.awssdk.services.s3.model.JSONType;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.services.s3.model.OutputSerialization;
import software.amazon.awssdk.services.s3.model.Progress;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;
import software.amazon.awssdk.services.s3.model.SelectObjectContentRequest;
import
    software.amazon.awssdk.services.s3.model.SelectObjectContentResponseHandler;
import software.amazon.awssdk.services.s3.model.Stats;

import java.io.IOException;
import java.net.URL;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;
```

```
public class SelectObjectContentExample {
    static final Logger logger =
        LoggerFactory.getLogger(SelectObjectContentExample.class);
    static String BUCKET_NAME = "amzn-s3-demo-bucket"; // Replace with your
        bucket name.
    static final S3AsyncClient s3AsyncClient = S3AsyncClient.create();
    static String FILE_CSV = "csv";
    static String FILE_JSON = "json";
    static String URL_CSV = "https://raw.githubusercontent.com/mledoze/countries/
        master/dist/countries.csv";
    static String URL_JSON = "https://raw.githubusercontent.com/mledoze/
        countries/master/dist/countries.json";

    public static void main(String[] args) {
        SelectObjectContentExample selectObjectContentExample = new
        SelectObjectContentExample();
        try {
            SelectObjectContentExample.setUp();
            selectObjectContentExample.runSelectObjectContentMethodForJSON();
            selectObjectContentExample.runSelectObjectContentMethodForCSV();
        } catch (SdkException e) {
            logger.error(e.getMessage(), e);
            System.exit(1);
        } finally {
            SelectObjectContentExample.tearDown();
        }
    }

    EventStreamInfo runSelectObjectContentMethodForJSON() {
        // Set up request parameters.
        final String queryExpression = "select * from s3object[*][*] c where
        c.area < 350000";
        final String fileType = FILE_JSON;

        InputSerialization inputSerialization = InputSerialization.builder()
            .json(JSONInput.builder().type(JSONType.DOCUMENT).build())
            .compressionType(CompressionType.NONE)
            .build();

        OutputSerialization outputSerialization = OutputSerialization.builder()
            .json(JSONOutput.builder().recordDelimiter(null).build())
            .build();
    }
}
```

```

// Build the SelectObjectContentRequest.
SelectObjectContentRequest select = SelectObjectContentRequest.builder()
    .bucket(BUCKET_NAME)
    .key(FILE_JSON)
    .expression(queryExpression)
    .expressionType(ExpressionType.SQL)
    .inputSerialization(inputSerialization)
    .outputSerialization(outputSerialization)
    .build();

EventStreamInfo eventStreamInfo = new EventStreamInfo();
// Call the selectObjectContent method with the request and a response
handler.
// Supply an EventStreamInfo object to the response handler to gather
records and information from the response.
s3AsyncClient.selectObjectContent(select,
buildResponseHandler(eventStreamInfo)).join();

// Log out information gathered while processing the response stream.
long recordCount = eventStreamInfo.getRecords().stream().mapToInt(record
->
    record.split("\n").length
).sum();
logger.info("Total records {}: {}", fileType, recordCount);
logger.info("Visitor onRecords for fileType {} called {} times",
fileType, eventStreamInfo.getCountOnRecordsCalled());
logger.info("Visitor onStats for fileType {}, {}", fileType,
eventStreamInfo.getStats());
logger.info("Visitor onContinuations for fileType {}, {}", fileType,
eventStreamInfo.getCountContinuationEvents());
return eventStreamInfo;
}

static SelectObjectContentResponseHandler
buildResponseHandler(EventStreamInfo eventStreamInfo) {
    // Use a Visitor to process the response stream. This visitor logs
    information and gathers details while processing.
    final SelectObjectContentResponseHandler.Visitor visitor =
    SelectObjectContentResponseHandler.Visitor.builder()
        .onRecords(r -> {
            logger.info("Record event received.");
            eventStreamInfo.addRecord(r.payload().asUtf8String());
            eventStreamInfo.incrementOnRecordsCalled();
        })

```

```
        .onCont(ce -> {
            logger.info("Continuation event received.");
            eventStreamInfo.incrementContinuationEvents();
        })
        .onProgress(pe -> {
            Progress progress = pe.details();
            logger.info("Progress event received:\n bytesScanned:
{} \n bytesProcessed: {} \n bytesReturned: {}",
                progress.bytesScanned(),
                progress.bytesProcessed(),
                progress.bytesReturned());
        })
        .onEnd(ee -> logger.info("End event received."))
        .onStats(se -> {
            logger.info("Stats event received.");
            eventStreamInfo.addStats(se.details());
        })
        .build();

    // Build the SelectObjectContentResponseHandler with the visitor that
    // processes the stream.
    return SelectObjectContentResponseHandler.builder()
        .subscriber(visitor).build();
}

// The EventStreamInfo class is used to store information gathered while
// processing the response stream.
static class EventStreamInfo {
    private final List<String> records = new ArrayList<>();
    private Integer countOnRecordsCalled = 0;
    private Integer countContinuationEvents = 0;
    private Stats stats;

    void incrementOnRecordsCalled() {
        countOnRecordsCalled++;
    }

    void incrementContinuationEvents() {
        countContinuationEvents++;
    }

    void addRecord(String record) {
        records.add(record);
    }
}
```

```
void addStats(Stats stats) {
    this.stats = stats;
}

public List<String> getRecords() {
    return records;
}

public Integer getCountOnRecordsCalled() {
    return countOnRecordsCalled;
}

public Integer getCountContinuationEvents() {
    return countContinuationEvents;
}

public Stats getStats() {
    return stats;
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [SelectObjectContent](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UploadPart 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 UploadPart。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [使用檢查總和](#)
- [使用 Amazon S3 物件完整性](#)

C++

適用於 C++ 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

//! Upload a part to an S3 bucket.
/*!
    \param bucket: The name of the S3 bucket where the object will be uploaded.
    \param key: The unique identifier (key) for the object within the S3 bucket.
    \param uploadID: An upload ID string.
    \param partNumber:
    \param checksumAlgorithm: Checksum algorithm, ignored when NOT_SET.
    \param calculatedHash: A data integrity hash to set, depending on the
    checksum algorithm,
                                ignored when it is an empty string.
    \param body: An shared_ptr IOSTream of the data to be uploaded.
    \param client: The S3 client instance used to perform the upload operation.
    \return UploadPartOutcome: The outcome.
*/

Aws::S3::Model::UploadPartOutcome AwsDoc::S3::uploadPart(const Aws::String
    &bucket,
                                                         const Aws::String &key,
                                                         const Aws::String
    &uploadID,
                                                         int partNumber,
                                                         Aws::S3::Model::ChecksumAlgorithm checksumAlgorithm,
                                                         const Aws::String
    &calculatedHash,
                                                         const
    std::shared_ptr<Aws::IOStream> &body,
                                                         const Aws::S3::S3Client
    &client) {
    Aws::S3::Model::UploadPartRequest request;
    request.SetBucket(bucket);
    request.SetKey(key);

```

```

request.SetUploadId(uploadID);
request.SetPartNumber(partNumber);
if (checksumAlgorithm != Aws::S3::Model::ChecksumAlgorithm::NOT_SET) {
    request.SetChecksumAlgorithm(checksumAlgorithm);
}
request.SetBody(body);

if (!calculatedHash.empty()) {
    switch (checksumAlgorithm) {
        case Aws::S3::Model::ChecksumAlgorithm::NOT_SET:
            request.SetContentMD5(calculatedHash);
            break;
        case Aws::S3::Model::ChecksumAlgorithm::CRC32:
            request.SetChecksumCRC32(calculatedHash);
            break;
        case Aws::S3::Model::ChecksumAlgorithm::CRC32C:
            request.SetChecksumCRC32C(calculatedHash);
            break;
        case Aws::S3::Model::ChecksumAlgorithm::SHA1:
            request.SetChecksumSHA1(calculatedHash);
            break;
        case Aws::S3::Model::ChecksumAlgorithm::SHA256:
            request.SetChecksumSHA256(calculatedHash);
            break;
    }
}

return client.UploadPart(request);
}

```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 [UploadPart](#)。

CLI

AWS CLI

下列命令會上傳使用 create-multipart-upload 命令起始之分段上傳作業的第一個部分：

```

aws s3api upload-part --bucket amzn-s3-demo-bucket --key
'multipart/01' --part-number 1 --body part01 --upload-id
'dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6YjLx.cph0gtNBtJ8P

```

body 選項會取得本機檔案的名稱或路徑，以進行上傳作業 (請勿使用 file:// prefix)。最小部分的大小為 5 MB。上傳 ID 由 create-multipart-upload 傳回，也可以藉由 list-multipart-uploads 擷取。當您建立分段上傳時，會指定儲存貯體和索引鍵。

輸出：

```
{
  "ETag": "\"e868e0f4719e394144ef36531ee6824c\""
}
```

儲存每一個部分的 ETag 值，以供稍後使用。完成分段上傳時需要該值。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [UploadPart](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
let mut upload_parts: Vec<aws_sdk_s3::types::CompletedPart> = Vec::new();

for chunk_index in 0..chunk_count {
    let this_chunk = if chunk_count - 1 == chunk_index {
        size_of_last_chunk
    } else {
        CHUNK_SIZE
    };
    let stream = ByteStream::read_from()
        .path(path)
        .offset(chunk_index * CHUNK_SIZE)
        .length(Length::Exact(this_chunk))
        .build()
        .await
        .unwrap();

    // Chunk index needs to start at 0, but part numbers start at 1.
```

```

        let part_number = (chunk_index as i32) + 1;
        let upload_part_res = client
            .upload_part()
            .key(&key)
            .bucket(&bucket_name)
            .upload_id(upload_id)
            .body(stream)
            .part_number(part_number)
            .send()
            .await?;

        upload_parts.push(
            CompletedPart::builder()
                .e_tag(upload_part_res.e_tag.unwrap_or_default())
                .part_number(part_number)
                .build(),
        );
    }

```

```

// Create a multipart upload. Use UploadPart and CompleteMultipartUpload to
// upload the file.
let multipart_upload_res: CreateMultipartUploadOutput = client
    .create_multipart_upload()
    .bucket(&bucket_name)
    .key(&key)
    .send()
    .await?;

let upload_id = multipart_upload_res.upload_id().ok_or(S3ExampleError::new(
    "Missing upload_id after CreateMultipartUpload",
))?;

```

```

// upload_parts: Vec<aws_sdk_s3::types::CompletedPart>
let completed_multipart_upload: CompletedMultipartUpload =
    CompletedMultipartUpload::builder()
        .set_parts(Some(upload_parts))
        .build();

let _complete_multipart_upload_res = client
    .complete_multipart_upload()
    .bucket(&bucket_name)

```

```
.key(&key)
.multipart_upload(completed_multipart_upload)
.upload_id(upload_id)
.send()
.await?;
```

- 如需 API 的詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [UploadPart](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UploadPartCopy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 UploadPartCopy。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [執行分段複製](#)

CLI

AWS CLI

將現有物件中的資料複製為資料來源，以上傳部分的物件

下列 upload-part-copy 範例會將現有物件中的資料複製為資料來源，以上傳其中的一部分。

```
aws s3api upload-part-copy \
  --bucket amzn-s3-demo-bucket \
  --key "Map_Data_June.mp4" \
  --copy-source "amzn-s3-demo-bucket/copy_of_Map_Data_June.mp4" \
  --part-number 1 \
  --upload-
id "bq0tdE1CDpWQYRPLHuNG50xAT6pA5D.m_RiBy0gg0H6b13pVRY7QjvLlf75iFdJqp_2wztk5hvpUM2SesXgrz"
```

輸出：

```
{
  "CopyPartResult": {
```

```
        "LastModified": "2019-12-13T23:16:03.000Z",
        "ETag": "\"711470fc377698c393d94aed6305e245\""
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [UploadPartCopy](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
public CompletableFuture<String> performMultiCopy(String toBucket, String
bucketName, String key) {
    CreateMultipartUploadRequest createMultipartUploadRequest =
CreateMultipartUploadRequest.builder()
        .bucket(toBucket)
        .key(key)
        .build();

    getAsyncClient().createMultipartUpload(createMultipartUploadRequest)
        .thenApply(createMultipartUploadResponse -> {
            String uploadId = createMultipartUploadResponse.uploadId();
            System.out.println("Upload ID: " + uploadId);

            UploadPartCopyRequest uploadPartCopyRequest =
UploadPartCopyRequest.builder()
                .sourceBucket(bucketName)
                .destinationBucket(toBucket)
                .sourceKey(key)
                .destinationKey(key)
                .uploadId(uploadId) // Use the valid uploadId.
                .partNumber(1) // Ensure the part number is correct.
                .copySourceRange("bytes=0-1023") // Adjust range as needed
                .build();

            return getAsyncClient().uploadPartCopy(uploadPartCopyRequest);
```

```
    })
    .thenCompose(uploadPartCopyFuture -> uploadPartCopyFuture)
    .whenComplete((uploadPartCopyResponse, exception) -> {
        if (exception != null) {
            // Handle any exceptions.
            logger.error("Error during upload part copy: " +
exception.getMessage());
        } else {
            // Successfully completed the upload part copy.
            System.out.println("Upload Part Copy completed successfully.
ETag: " + uploadPartCopyResponse.copyPartResult().eTag());
        }
    });
    return null;
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [UploadPartCopy](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Amazon S3 AWS SDKs 案例

下列程式碼範例示範如何在 Amazon S3 AWS SDKs 中實作常見案例。這些案例示範如何呼叫 Amazon S3 中的多個函數，或與其他 AWS 服務結合，藉以完成特定任務。每個案例均包含完整原始碼的連結，您可在連結中找到如何設定和執程式碼的相關指示。

案例的目標是獲得中等水平的經驗，協助您了解內容中的服務動作。

範例

- [檢查儲存貯體是否存在](#)
- [檢查物件是否存在](#)
- [使用 AWS SDK 將文字轉換為語音並返回文字](#)
- [使用 AWS SDK 建立 Amazon S3 的預先簽章 URL](#)
- [建立相片資產管理應用程式，讓使用者以標籤管理相片](#)
- [使用 AWS SDK 列出 Amazon S3 物件的網頁](#)
- [建立 Amazon Textract Explorer 應用程式](#)
- [使用 AWS SDK 刪除指定 Amazon S3 儲存貯體中的所有物件](#)

- [使用 AWS SDK 刪除不完整的分段上傳至 Amazon S3](#)
- [使用 AWS SDK 透過 Amazon Rekognition 偵測映像中的個人防護裝備](#)
- [使用 AWS SDK 偵測從映像擷取的文字中的實體](#)
- [使用 AWS SDK 偵測映像中的臉部](#)
- [使用 AWS SDK 透過 Amazon Rekognition 偵測映像中的物件](#)
- [使用 AWS SDK 透過 Amazon Rekognition 偵測影片中的人物和物件](#)
- [從 Amazon Simple Storage Service \(Amazon S3\) 儲存貯體下載 S3 'directories'](#)
- [將 Amazon Simple Storage Service \(Amazon S3\) 儲存貯體中的所有物件下載至本機目錄](#)
- [使用 AWS SDK 從 Amazon S3 物件下載未知大小的串流](#)
- [使用 SDK 從多區域存取點取得 Amazon S3 物件 AWS](#)
- [使用 AWS SDK 從 Amazon S3 儲存貯體取得物件，指定 If-Modified-Since 標頭](#)
- [使用 AWS SDK 開始加密 Amazon S3 物件](#)
- [使用 AWS SDK 開始使用 Amazon S3 物件的標籤](#)
- [大數據處理叢集入門](#)
- [組態管理入門](#)
- [文件文字擷取入門](#)
- [機器學習功能存放區的入門](#)
- [物件儲存入門](#)
- [查詢分析入門](#)
- [使用 AWS SDK 使用 Amazon S3 物件鎖定功能](#)
- [使用 SDK 提出 Amazon S3 條件式請求 AWS](#)
- [使用 AWS SDK 管理 Amazon S3 儲存貯體的存取控制清單 \(ACLs\)](#)
- [透過 SDK 使用 Amazon S3 管理大型 Amazon SQS 訊息 Amazon S3 AWS](#)
- [使用 SDK 使用 Lambda 函數分批管理版本控制的 Amazon S3 物件 AWS](#)
- [使用 AWS SDK 剖析 Amazon S3 URIs](#)
- [使用 AWS SDK 執行 Amazon S3 物件的分段複製](#)
- [使用 SDK 接收和處理 Amazon S3 事件通知 AWS](#)
- [使用 AWS SDK 儲存 EXIF 和其他影像資訊](#)
- [使用 AWS SDK 將 S3 事件通知傳送至 Amazon EventBridge](#)
- [使用 SDK 追蹤 Amazon S3 物件上傳或下載 AWS](#)

- [使用 S3 Object Lambda 轉換應用程式的資料](#)
- [使用 AWS SDK 進行單元和整合測試的範例方法](#)
- [以遞迴的方式將本機目錄上傳至 Amazon Simple Storage Service \(Amazon S3\) 儲存貯體](#)
- [使用 AWS SDK 在 Amazon S3 之間上傳或下載大型檔案](#)
- [使用 AWS SDK 將大小不明的串流上傳至 Amazon S3 物件](#)
- [使用檢查總和來使用 AWS SDK 處理 Amazon S3 物件](#)
- [使用 AWS SDK 使用 Amazon S3 物件完整性功能](#)
- [使用 AWS SDK 處理 Amazon S3 版本控制的物件](#)

檢查儲存貯體是否存在

下列程式碼範例示範如何檢查儲存貯體是否存在。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

您可以使用下列 `doesBucketExists` 方法取代適用於 Java 的 SDK V1 的 [AmazonS3Client#doesBucketExistV2\(String\)](#) 方法。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.awscore.exception.AwsServiceException;
import software.amazon.awssdk.http.HttpStatusCode;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.utils.Validate;

public class DoesBucketExist {
    private static final Logger logger =
        LoggerFactory.getLogger(DoesBucketExist.class);

    public static void main(String[] args) {
```

```

        DoesBucketExist doesBucketExist = new DoesBucketExist();

        final S3Client s3SyncClient = S3Client.builder().build();
        final String bucketName = "amzn-s3-demo-bucket"; // Replace with the
        bucket name that you want to check.

        boolean exists = doesBucketExist.doesBucketExist(bucketName,
        s3SyncClient);
        logger.info("Bucket exists: {}", exists);
    }

    /**
     * Checks if the specified bucket exists. Amazon S3 buckets are named in a
     * global namespace; use this method to
     * determine if a specified bucket name already exists, and therefore can't
     * be used to create a new bucket.
     * <p>
     * Internally this method uses the <a
     * href="https://sdk.amazonaws.com/java/api/latest/software/amazon/awssdk/
     services/s3/
     S3Client.html#getBucketAcl(java.util.function.Consumer)">S3Client.getBucketAcl(String)</
     a>
     * operation to determine whether the bucket exists.
     * <p>
     * This method is equivalent to the AWS SDK for Java V1's <a
     * href="https://docs.aws.amazon.com/AWSJavaSDK/latest/javadoc/
     com/amazonaws/services/s3/AmazonS3Client.html#doesBucketExistV2-
     java.lang.String-">AmazonS3Client#doesBucketExistV2(String)</a>.
     *
     * @param bucketName The name of the bucket to check.
     * @param s3SyncClient An <code>S3Client</code> instance. The method checks
     for the bucket in the AWS Region
     *
     * configured on the instance.
     * @return The value true if the specified bucket exists in Amazon S3; the
     value false if there is no bucket in
     *
     Amazon S3 with that name.
     */
    public boolean doesBucketExist(String bucketName, S3Client s3SyncClient) {
        try {
            Validate.notEmpty(bucketName, "The bucket name must not be null or an
            empty string.", "");
            s3SyncClient.getBucketAcl(r -> r.bucket(bucketName));
            return true;
        } catch (AwsServiceException ase) {

```

```
        // A redirect error or an AccessDenied exception means the bucket
        exists but it's not in this region
        // or we don't have permissions to it.
        if ((ase.statusCode() == HttpStatusCode.MOVED_PERMANENTLY) ||
            "AccessDenied".equals(ase.awsErrorDetails().errorCode())) {
            return true;
        }
        if (ase.statusCode() == HttpStatusCode.NOT_FOUND) {
            return false;
        }
        throw ase;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetBucketAcl](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

檢查物件是否存在

下列程式碼範例示範如何檢查物件是否存在。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

您可以使用下列 `doesObjectExist` 方法取代適用於 Java V1 的 [SDK AmazonS3Client#doesObjectExist\(String, String\)](#) 方法。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.s3.S3Client;
```

```

import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.NoSuchKeyException;
import software.amazon.awssdk.utils.Validate;

public class DoesObjectExist {
    private static final Logger logger =
        LoggerFactory.getLogger(DoesObjectExist.class);

    public static void main(String[] args) {
        DoesObjectExist doesObjectExist = new DoesObjectExist();

        final S3Client s3SyncClient = S3Client.builder().build();
        final String bucketName = "amzn-s3-demo-bucket"; // Replace with your
        bucket name.
        final String key = "my-key"; // Replace with your object key.

        boolean exists = doesObjectExist.doesObjectExist(bucketName, key,
        s3SyncClient);
        logger.info("Object exists: {}", exists);
    }

    /**
     * Checks if the specified object exists in the specified bucket.
     * <p>
     * Internally this method uses the
     * <a href="https://sdk.amazonaws.com/java/api/latest/software.amazon/awssdk/
services/s3/
S3Client.html#headObject(java.util.function.Consumer)">S3Client.headObject</a>
     * operation to determine whether the object exists.
     * <p>
     * This method is equivalent to the AWS SDK for Java V1's
     * <a href="https://docs.aws.amazon.com/AWSJavaSDK/latest/javadoc/com/
amazonaws/services/s3/AmazonS3Client.html#doesObjectExist-java.lang.String-
java.lang.String-">AmazonS3Client#doesObjectExist(String, String)</a>.
     * <p>
     * <b>Note:</b> This method returns {@code false} only when S3 responds with
404 (NoSuchKey). If the caller
     * does not have {@code s3:ListBucket} permission on the bucket, S3 may
return 403 instead of 404 for
     * non-existent objects, which will be thrown as an exception.
     *
     * @param bucketName The name of the bucket containing the object.
     * @param key The key of the object to check.
     * @param s3SyncClient An {@code S3Client} instance.

```

```
    * @return {@code true} if the object exists; {@code false} if it does not
    exist.
    */
    public boolean doesObjectExist(String bucketName, String key, S3Client
s3SyncClient) {
        try {
            Validate.notEmpty(bucketName, "The bucket name must not be null or an
empty string.", "");
            Validate.notEmpty(key, "The object key must not be null or an empty
string.", "");
            s3SyncClient.headObject(HeadObjectRequest.builder()
                .bucket(bucketName)
                .key(key)
                .build());
            return true;
        } catch (NoSuchKeyException e) {
            return false;
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [HeadObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 將文字轉換為語音並返回文字

以下程式碼範例顯示做法：

- 使用 Amazon Polly 將純文字 (UTF-8) 輸入檔案合成至音訊檔案中。
- 將音訊檔案上傳至 Amazon S3 儲存貯體。
- 使用 Amazon Transcribe 將音訊檔案轉換為文字。
- 顯示文字。

Rust

適用於 Rust 的 SDK

使用 Amazon Polly 將純文字 (UTF-8) 輸入檔案合成至音訊檔案中，將音訊檔案上傳至 Amazon S3 儲存貯體，使用 Amazon Transcribe 將該音訊檔案轉換為文字，然後顯示文字。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Polly
- Amazon S3
- Amazon Transcribe

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 建立 Amazon S3 的預先簽章 URL

下列程式碼範例示範如何建立適用於 Amazon S3 預先簽署的 URL 並上傳物件。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

產生可以於限定時間內執行 Amazon S3 動作的預先簽署的 URL。

```
using System;
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;

public class GenPresignedUrl
{
    public static void Main()
```

```

{
    const string bucketName = "amzn-s3-demo-bucket";
    const string objectKey = "sample.txt";

    // Specify how long the presigned URL lasts, in hours
    const double timeoutDuration = 12;

    // Specify the AWS Region of your Amazon S3 bucket. If it is
    // different from the Region defined for the default user,
    // pass the Region to the constructor for the client. For
    // example: new AmazonS3Client(RegionEndpoint.USEast1);

    // If using the Region us-east-1, and server-side encryption with AWS
    KMS, you must specify Signature Version 4.
    // Region us-east-1 defaults to Signature Version 2 unless explicitly
    set to Version 4 as shown below.
    // For more details, see https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingAWSSDK.html#specify-signature-version
    // and https://docs.aws.amazon.com/sdkfornet/v3/apidocs/items/Amazon/TAWSConfigsS3.html
    AWSConfigsS3.UseSignatureVersion4 = true;
    IAmazonS3 s3Client = new AmazonS3Client(RegionEndpoint.USEast1);

    string urlString = GeneratePresignedURL(s3Client, bucketName,
objectKey, timeoutDuration);
    Console.WriteLine($"The generated URL is: {urlString}.");
}

/// <summary>
/// Generate a presigned URL that can be used to access the file named
/// in the objectKey parameter for the amount of time specified in the
/// duration parameter.
/// </summary>
/// <param name="client">An initialized S3 client object used to call
/// the GetPresignedUrl method.</param>
/// <param name="bucketName">The name of the S3 bucket containing the
/// object for which to create the presigned URL.</param>
/// <param name="objectKey">The name of the object to access with the
/// presigned URL.</param>
/// <param name="duration">The length of time for which the presigned
/// URL will be valid.</param>
/// <returns>A string representing the generated presigned URL.</returns>
public static string GeneratePresignedURL(IAmazonS3 client, string
bucketName, string objectKey, double duration)

```

```
{
    string urlString = string.Empty;
    try
    {
        var request = new GetPreSignedUrlRequest()
        {
            BucketName = bucketName,
            Key = objectKey,
            Expires = DateTime.UtcNow.AddHours(duration),
        };
        urlString = client.GetPreSignedURL(request);
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error: '{ex.Message}'");
    }

    return urlString;
}
}
```

產生預先簽章的 URL 並使用該 URL 執行上傳。

```
using System;
using System.IO;
using System.Net.Http;
using System.Threading.Tasks;
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to upload an object to an Amazon Simple Storage
/// Service (Amazon S3) bucket using a presigned URL. The code first
/// creates a presigned URL and then uses it to upload an object to an
/// Amazon S3 bucket using that URL.
/// </summary>
public class UploadUsingPresignedURL
{
    private static HttpClient httpClient = new HttpClient();
```



```
public static async Task Main()
{
    string bucketName = "amzn-s3-demo-bucket";
    string keyName = "samplefile.txt";
    string filePath = $"source\\{keyName}";

    // Specify how long the signed URL will be valid in hours.
    double timeoutDuration = 12;

    // Specify the AWS Region of your Amazon S3 bucket. If it is
    // different from the Region defined for the default user,
    // pass the Region to the constructor for the client. For
    // example: new AmazonS3Client(RegionEndpoint.USEast1);

    // If using the Region us-east-1, and server-side encryption with AWS
    KMS, you must specify Signature Version 4.
    // Region us-east-1 defaults to Signature Version 2 unless explicitly
    set to Version 4 as shown below.
    // For more details, see https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingAWSSDK.html#specify-signature-version
    // and https://docs.aws.amazon.com/sdkfornet/v3/apidocs/items/Amazon/TAWSConfigsS3.html
    AWSConfigsS3.UseSignatureVersion4 = true;
    IAmazonS3 client = new AmazonS3Client(RegionEndpoint.USEast1);

    var url = GeneratePreSignedURL(client, bucketName, keyName,
    timeoutDuration);
    var success = await UploadObject(filePath, url);

    if (success)
    {
        Console.WriteLine("Upload succeeded.");
    }
    else
    {
        Console.WriteLine("Upload failed.");
    }
}

/// <summary>
/// Uploads an object to an Amazon S3 bucket using the presigned URL
passed in
/// the url parameter.
/// </summary>
```

```

    /// <param name="filePath">The path (including file name) to the local
    /// file you want to upload.</param>
    /// <param name="url">The presigned URL that will be used to upload the
    /// file to the Amazon S3 bucket.</param>
    /// <returns>A Boolean value indicating the success or failure of the
    /// operation, based on the HttpResponseMessage.</returns>
    public static async Task<bool> UploadObject(string filePath, string url)
    {
        using var streamContent = new StreamContent(
            new FileStream(filePath, FileMode.Open, FileAccess.Read));

        var response = await httpClient.PutAsync(url, streamContent);
        return response.IsSuccessStatusCode;
    }

    /// <summary>
    /// Generates a presigned URL which will be used to upload an object to
    /// an Amazon S3 bucket.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// GetPreSignedURL.</param>
    /// <param name="bucketName">The name of the Amazon S3 bucket to which
the
    /// presigned URL will point.</param>
    /// <param name="objectKey">The name of the file that will be uploaded.</
param>
    /// <param name="duration">How long (in hours) the presigned URL will
    /// be valid.</param>
    /// <returns>The generated URL.</returns>
    public static string GeneratePreSignedURL(
        IAmazonS3 client,
        string bucketName,
        string objectKey,
        double duration)
    {
        var request = new GetPreSignedUrlRequest
        {
            BucketName = bucketName,
            Key = objectKey,
            Verb = HttpVerb.PUT,
            Expires = DateTime.UtcNow.AddHours(duration),
        };
    }

```

```
        string url = client.GetPreSignedURL(request);
        return url;
    }
}
```

適用於 .NET 的 SDK (v4)

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立並使用預先簽章的 POST URL 直接由瀏覽器上傳。

```
/// <summary>
/// Scenario demonstrating the complete workflow for presigned POST URLs:
/// 1. Create an S3 bucket
/// 2. Create a presigned POST URL
/// 3. Upload a file using the presigned POST URL
/// 4. Clean up resources
/// </summary>
public class CreatePresignedPostBasics
{
    public static ILogger<CreatePresignedPostBasics> _logger = null!;
    public static S3Wrapper _s3Wrapper = null!;
    public static UiMethods _uiMethods = null!;
    public static IHttpClientFactory _httpClientFactory = null!;
    public static bool _isInteractive = true;
    public static string? _bucketName;
    public static string? _objectKey;

    /// <summary>
    /// Set up the services and logging.
    /// </summary>
    /// <param name="host">The IHost instance.</param>
    public static void SetupServices(IHost host)
    {
        var loggerFactory = LoggerFactory.Create(builder =>
        {
            builder.AddConsole();
        });
    }
}
```

```

    });
    _logger = new Logger<CreatePresignedPostBasics>(loggerFactory);

    _s3Wrapper = host.Services.GetRequiredService<S3Wrapper>();
    _httpClientFactory =
host.Services.GetRequiredService<IHttpClientFactory>();
    _uiMethods = new UiMethods();
}

/// <summary>
/// Perform the actions defined for the Amazon S3 Presigned POST scenario.
/// </summary>
/// <param name="args">Command line arguments.</param>
/// <returns>A Task object.</returns>
public static async Task Main(string[] args)
{
    _isInteractive = !args.Contains("--non-interactive");

    // Set up dependency injection for Amazon S3
    using var host =
Microsoft.Extensions.Hosting.Host.CreateDefaultBuilder(args)
        .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonS3>()
                .AddTransient<S3Wrapper>()
                .AddHttpClient()
        )
        .Build();

    SetupServices(host);

    try
    {
        // Display overview
        _uiMethods.DisplayOverview();
        _uiMethods.PressEnter(_isInteractive);

        // Step 1: Create bucket
        await CreateBucketAsync();
        _uiMethods.PressEnter(_isInteractive);

        // Step 2: Create presigned URL
        _uiMethods.DisplayTitle("Step 2: Create presigned POST URL");
        var response = await CreatePresignedPostAsync();
        _uiMethods.PressEnter(_isInteractive);
    }
}

```

```

        // Step 3: Display URL and fields
        _uiMethods.DisplayTitle("Step 3: Presigned POST URL details");
        DisplayPresignedPostFields(response);
        _uiMethods.PressEnter(_isInteractive);

        // Step 4: Upload file
        _uiMethods.DisplayTitle("Step 4: Upload test file using presigned
POST URL");
        await UploadFileAsync(response);
        _uiMethods.PressEnter(_isInteractive);

        // Step 5: Verify file exists
        await VerifyFileExistsAsync();
        _uiMethods.PressEnter(_isInteractive);

        // Step 6: Cleanup
        _uiMethods.DisplayTitle("Step 6: Clean up resources");
        await CleanupAsync();

        _uiMethods.DisplayTitle("S3 Presigned POST Scenario completed
successfully!");
        _uiMethods.PressEnter(_isInteractive);
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, "Error in scenario");
        Console.WriteLine($"Error: {ex.Message}");

        // Attempt cleanup if there was an error
        if (!string.IsNullOrEmpty(_bucketName))
        {
            _uiMethods.DisplayTitle("Cleaning up resources after error");
            await _s3Wrapper.DeleteBucketAsync(_bucketName);
            Console.WriteLine($"Cleaned up bucket: {_bucketName}");
        }
    }
}

/// <summary>
/// Create an S3 bucket for the scenario.
/// </summary>
private static async Task CreateBucketAsync()
{

```

```
        _uiMethods.DisplayTitle("Step 1: Create an S3 bucket");

        // Generate a default bucket name for the scenario
        var defaultBucketName = $"presigned-post-demo-
{DateTime.Now:yyyyMMddHHmmss}".ToLower();

        // Prompt user for bucket name or use default in non-interactive mode
        _bucketName = _uiMethods.GetUserInput(
            $"Enter S3 bucket name (or press Enter for '{defaultBucketName}'): ",
            defaultBucketName,
            _isInteractive);

        // Basic validation to ensure bucket name is not empty
        if (string.IsNullOrEmpty(_bucketName))
        {
            _bucketName = defaultBucketName;
        }

        Console.WriteLine($"Creating bucket: {_bucketName}");

        await _s3Wrapper.CreateBucketAsync(_bucketName);

        Console.WriteLine($"Successfully created bucket: {_bucketName}");
    }

    /// <summary>
    /// Create a presigned POST URL.
    /// </summary>
    private static async Task<CreatePresignedPostResponse>
    CreatePresignedPostAsync()
    {
        _objectKey = "example-upload.txt";
        var expiration = DateTime.UtcNow.AddMinutes(10); // Short expiration for
        the demo

        Console.WriteLine($"Creating presigned POST URL for {_bucketName}/
{_objectKey}");
        Console.WriteLine($"Expiration: {expiration} UTC");

        var s3Client = _s3Wrapper.GetS3Client();

        var response = await _s3Wrapper.CreatePresignedPostAsync(
            s3Client, _bucketName!, _objectKey, expiration);
    }
}
```

```
        Console.WriteLine("Successfully created presigned POST URL");
        return response;
    }

    /// <summary>
    /// Upload a file using the presigned POST URL.
    /// </summary>
    private static async Task UploadFileAsync(CreatePresignedPostResponse
response)
    {

        // Create a temporary test file to upload
        string testFilePath = Path.GetTempFileName();
        string testContent = "This is a test file for the S3 presigned POST
scenario.";

        await File.WriteAllTextAsync(testFilePath, testContent);
        Console.WriteLine($"Created test file at: {testFilePath}");

        // Upload the file using the presigned POST URL
        Console.WriteLine("\nUploading file using the presigned POST URL...");
        var uploadResult = await UploadFileWithPresignedPostAsync(response,
testFilePath);

        // Display the upload result
        if (uploadResult.Success)
        {
            Console.WriteLine($"Upload successful! Status code:
{uploadResult.StatusCode}");
        }
        else
        {
            Console.WriteLine($"Upload failed with status code:
{uploadResult.StatusCode}");
            Console.WriteLine($"Error: {uploadResult.Response}");
            throw new Exception("File upload failed");
        }

        // Clean up the temporary file
        File.Delete(testFilePath);
        Console.WriteLine("Temporary file deleted");
    }
}
```

```
/// <summary>
/// Helper method to upload a file using a presigned POST URL.
/// </summary>
private static async Task<(bool Success, HttpStatusCode StatusCode, string
Response)> UploadFileWithPresignedPostAsync(
    CreatePresignedPostResponse response,
    string filePath)
{
    try
    {
        _logger.LogInformation("Uploading file {filePath} using presigned
POST URL", filePath);

        using var httpClient = _httpClientFactory.CreateClient();
        using var formContent = new MultipartFormDataContent();

        // Add all the fields from the presigned POST response
        foreach (var field in response.Fields)
        {
            formContent.Add(new StringContent(field.Value), field.Key);
        }

        // Add the file content
        var fileStream = File.OpenRead(filePath);
        var fileName = Path.GetFileName(filePath);
        var fileContent = new StreamContent(fileStream);
        fileContent.Headers.ContentType = new MediaTypeHeaderValue("text/
plain");
        formContent.Add(fileContent, "file", fileName);

        // Send the POST request
        var httpResponse = await httpClient.PostAsync(response.Url,
formContent);
        var responseContent = await httpResponse.Content.ReadAsStringAsync();

        // Log and return the result
        _logger.LogInformation("Upload completed with status code
{statusCode}", httpResponse.StatusCode);

        return (httpResponse.IsSuccessStatusCode, httpResponse.StatusCode,
responseContent);
    }
    catch (Exception ex)
    {
```



```

        _logger.LogError(ex, "Error uploading file");
        return (false, HttpStatusCode.InternalServerError, ex.Message);
    }
}

/// <summary>
/// Verify that the uploaded file exists in the S3 bucket.
/// </summary>
private static async Task VerifyFileExistsAsync()
{
    _uiMethods.DisplayTitle("Step 5: Verify uploaded file exists");

    Console.WriteLine($"Checking if file exists at {_bucketName}/
{_objectKey}...");

    try
    {
        var metadata = await _s3Wrapper.GetObjectMetadataAsync(_bucketName!,
            _objectKey!);

        Console.WriteLine($"File verification successful! File exists in the
bucket.");
        Console.WriteLine($"File size: {metadata.ContentLength} bytes");
        Console.WriteLine($"File type: {metadata.Headers.ContentType}");
        Console.WriteLine($"Last modified: {metadata.LastModified}");
    }
    catch (AmazonS3Exception ex) when (ex.StatusCode ==
System.Net.HttpStatusCode.NotFound)
    {
        Console.WriteLine($"Error: File was not found in the bucket.");
        throw;
    }
}

private static void DisplayPresignedPostFields(CreatePresignedPostResponse
response)
{
    Console.WriteLine($"Presigned POST URL: {response.Url}");
    Console.WriteLine("Form fields to include:");

    foreach (var field in response.Fields)
    {
        Console.WriteLine($"  {field.Key}: {field.Value}");
    }
}

```

```
}

/// <summary>
/// Clean up resources created by the scenario.
/// </summary>
private static async Task CleanupAsync()
{
    if (!string.IsNullOrEmpty(_bucketName))
    {
        Console.WriteLine($"Deleting bucket {_bucketName} and its
contents...");
        bool result = await _s3Wrapper.DeleteBucketAsync(_bucketName);

        if (result)
        {
            Console.WriteLine("Bucket deleted successfully");
        }
        else
        {
            Console.WriteLine("Failed to delete bucket - it may have been
already deleted");
        }
    }
}
}
```

C++

適用於 C++ 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

產生預先簽章的 URL 以下載物件。

```
//! Routine which demonstrates creating a pre-signed URL to download an object
from an
//! Amazon Simple Storage Service (Amazon S3) bucket.
```

```

/*!
 \param bucketName: Name of the bucket.
 \param key: Name of an object key.
 \param expirationSeconds: Expiration in seconds for pre-signed URL.
 \param clientConfig: Aws client configuration.
 \return Aws::String: A pre-signed URL.
*/
Aws::String AwsDoc::S3::generatePreSignedGetObjectUrl(const Aws::String
    &bucketName,
                                                    const Aws::String &key,
                                                    uint64_t expirationSeconds,
                                                    const
    Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    return client.GeneratePresignedUrl(bucketName, key,
    Aws::Http::HttpMethod::HTTP_GET,
                                                    expirationSeconds);
}

```

使用 libcurl 下載。

```

static size_t myCurlWriteBack(char *buffer, size_t size, size_t nitems, void
    *userdata) {
    Aws::StringStream *str = (Aws::StringStream *) userdata;

    if (nitems > 0) {
        str->write(buffer, size * nitems);
    }
    return size * nitems;
}

//! Utility routine to test getObject with a pre-signed URL.
/*!
 \param presignedURL: A pre-signed URL to get an object from a bucket.
 \param resultString: A string to hold the result.
 \return bool: Function succeeded.
*/
bool AwsDoc::S3::getObjectWithPresignedObjectUrl(const Aws::String &presignedURL,
                                                    Aws::String &resultString) {
    CURL *curl = curl_easy_init();
    CURLcode result;

```

```
std::stringstream outWriteString;

result = curl_easy_setopt(curl, CURLOPT_WRITEDATA, &outWriteString);

if (result != CURLE_OK) {
    std::cerr << "Failed to set CURLOPT_WRITEDATA " << std::endl;
    return false;
}

result = curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, myCurlWriteBack);

if (result != CURLE_OK) {
    std::cerr << "Failed to set CURLOPT_WRITEFUNCTION" << std::endl;
    return false;
}

result = curl_easy_setopt(curl, CURLOPT_URL, presignedURL.c_str());

if (result != CURLE_OK) {
    std::cerr << "Failed to set CURLOPT_URL" << std::endl;
    return false;
}

result = curl_easy_perform(curl);

if (result != CURLE_OK) {
    std::cerr << "Failed to perform CURL request" << std::endl;
    return false;
}

resultString = outWriteString.str();

if (resultString.find("<?xml") == 0) {
    std::cerr << "Failed to get object, response:\n" << resultString <<
std::endl;
    return false;
}

return true;
}
```

產生預先簽章的 URL 以上傳物件。

```

//! Routine which demonstrates creating a pre-signed URL to upload an object to
an
//! Amazon Simple Storage Service (Amazon S3) bucket.
/*!
    \param bucketName: Name of the bucket.
    \param key: Name of an object key.
    \param clientConfig: Aws client configuration.
    \return Aws::String: A pre-signed URL.
*/
Aws::String AwsDoc::S3::generatePreSignedPutObjectUrl(const Aws::String
    &bucketName,
                                                    const Aws::String &key,
                                                    uint64_t expirationSeconds,
                                                    const
    Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    return client.GeneratePresignedUrl(bucketName, key,
    Aws::Http::HttpMethod::HTTP_PUT,
                                expirationSeconds);
}

```

使用 libcurl 上傳。

```

static size_t myCurlReadBack(char *buffer, size_t size, size_t nitems, void
*userdata) {
    Aws::StringStream *str = (Aws::StringStream *) userdata;

    str->read(buffer, size * nitems);

    return str->gcount();
}

static size_t myCurlWriteBack(char *buffer, size_t size, size_t nitems, void
*userdata) {
    Aws::StringStream *str = (Aws::StringStream *) userdata;

    if (nitems > 0) {
        str->write(buffer, size * nitems);
    }
    return size * nitems;
}

```

```
//! Utility routine to test putObject with a pre-signed URL.
/*!
  \param presignedURL: A pre-signed URL to put an object in a bucket.
  \param data: Body of the putObject request.
  \return bool: Function succeeded.
*/
bool AwsDoc::S3::PutStringWithPresignedObjectURL(const Aws::String &presignedURL,
                                                  const Aws::String &data) {

    CURL *curl = curl_easy_init();
    CURLcode result;

    Aws::StringStream readStringStream;
    readStringStream << data;
    result = curl_easy_setopt(curl, CURLOPT_READFUNCTION, myCurlReadBack);

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_READFUNCTION" << std::endl;
        return false;
    }

    result = curl_easy_setopt(curl, CURLOPT_READDATA, &readStringStream);
    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_READDATA" << std::endl;
        return false;
    }

    result = curl_easy_setopt(curl, CURLOPT_INFILESIZE_LARGE,
                              (curl_off_t) data.size());

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_INFILESIZE_LARGE" << std::endl;
        return false;
    }

    result = curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, myCurlWriteBack);

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_WRITEFUNCTION" << std::endl;
        return false;
    }

    std::stringstream outWriteString;

    result = curl_easy_setopt(curl, CURLOPT_WRITEDATA, &outWriteString);
```

```
if (result != CURLE_OK) {
    std::cerr << "Failed to set CURLOPT_WRITEDATA " << std::endl;
    return false;
}

result = curl_easy_setopt(curl, CURLOPT_URL, presignedURL.c_str());

if (result != CURLE_OK) {
    std::cerr << "Failed to set CURLOPT_URL" << std::endl;
    return false;
}

result = curl_easy_setopt(curl, CURLOPT_UPLOAD, 1L);

if (result != CURLE_OK) {
    std::cerr << "Failed to set CURLOPT_PUT" << std::endl;
    return false;
}

result = curl_easy_perform(curl);

if (result != CURLE_OK) {
    std::cerr << "Failed to perform CURL request" << std::endl;
    return false;
}

std::string outString = outWriteString.str();
if (outString.empty()) {
    std::cout << "Successfully put object." << std::endl;
    return true;
} else {
    std::cout << "A server error was encountered, output:\n" << outString
                << std::endl;
    return false;
}
}
```

Go

SDK for Go V2

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立可包裝 S3 預先簽章動作的函數。

```
import (  
    "context"  
    "log"  
    "time"  
  
    "github.com/aws/aws-sdk-go-v2/aws"  
    v4 "github.com/aws/aws-sdk-go-v2/aws/signer/v4"  
    "github.com/aws/aws-sdk-go-v2/service/s3"  
)  
  
// Presigner encapsulates the Amazon Simple Storage Service (Amazon S3) presign  
// actions  
// used in the examples.  
// It contains PresignClient, a client that is used to presign requests to Amazon  
// S3.  
// Presigned requests contain temporary credentials and can be made from any HTTP  
// client.  
type Presigner struct {  
    PresignClient *s3.PresignClient  
}  
  
// GetObject makes a presigned request that can be used to get an object from a  
// bucket.  
// The presigned request is valid for the specified number of seconds.  
func (presigner Presigner) GetObject(  
    ctx context.Context, bucketName string, objectKey string, lifetimeSecs int64)  
    (*v4.PresignedHTTPRequest, error) {
```



```
request, err := presigner.PresignClient.PresignGetObject(ctx,
&s3.GetObjectInput{
    Bucket: aws.String(bucketName),
    Key:    aws.String(objectKey),
}, func(opts *s3.PresignOptions) {
    opts.Expires = time.Duration(lifetimeSecs * int64(time.Second))
})
if err != nil {
    log.Printf("Couldn't get a presigned request to get %v:%v. Here's why: %v\n",
        bucketName, objectKey, err)
}
return request, err
}

// PutObject makes a presigned request that can be used to put an object in a
// bucket.
// The presigned request is valid for the specified number of seconds.
func (presigner Presigner) PutObject(
    ctx context.Context, bucketName string, objectKey string, lifetimeSecs int64)
(*v4.PresignedHTTPRequest, error) {
    request, err := presigner.PresignClient.PresignPutObject(ctx,
&s3.PutObjectInput{
    Bucket: aws.String(bucketName),
    Key:    aws.String(objectKey),
}, func(opts *s3.PresignOptions) {
    opts.Expires = time.Duration(lifetimeSecs * int64(time.Second))
})
if err != nil {
    log.Printf("Couldn't get a presigned request to put %v:%v. Here's why: %v\n",
        bucketName, objectKey, err)
}
return request, err
}

// DeleteObject makes a presigned request that can be used to delete an object
// from a bucket.
func (presigner Presigner) DeleteObject(ctx context.Context, bucketName string,
    objectKey string) (*v4.PresignedHTTPRequest, error) {
    request, err := presigner.PresignClient.PresignDeleteObject(ctx,
&s3.DeleteObjectInput{
```

```

    Bucket: aws.String(bucketName),
    Key:     aws.String(objectKey),
  })
  if err != nil {
    log.Printf("Couldn't get a presigned request to delete object %v. Here's why: %v\n", objectKey, err)
  }
  return request, err
}

func (presigner Presigner) PresignPostObject(ctx context.Context, bucketName
string, objectKey string, lifetimeSecs int64) (*s3.PresignedPostRequest, error)
{
  request, err := presigner.PresignClient.PresignPostObject(ctx,
&s3.PutObjectInput{
    Bucket: aws.String(bucketName),
    Key:     aws.String(objectKey),
  }, func(options *s3.PresignPostOptions) {
    options.Expires = time.Duration(lifetimeSecs) * time.Second
  })
  if err != nil {
    log.Printf("Couldn't get a presigned post request to put %v:%v. Here's why: %v\n", bucketName, objectKey, err)
  }
  return request, nil
}

```

執行互動式範例，產生並使用預先簽章的 URL 來上傳、下載和刪除 S3 物件。

```

import (
    "bytes"
    "context"
    "io"
    "log"
    "mime/multipart"
    "net/http"
    "os"

```

```
"strings"

"github.com/aws/aws-sdk-go-v2/aws"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
"github.com/awsdocs/aws-doc-sdk-examples/gov2/s3/actions"
)

// RunPresigningScenario is an interactive example that shows you how to get
// presigned
// HTTP requests that you can use to move data into and out of Amazon Simple
// Storage
// Service (Amazon S3). The presigned requests contain temporary credentials and
// can
// be used by an HTTP client.
//
// 1. Get a presigned request to put an object in a bucket.
// 2. Use the net/http package to use the presigned request to upload a local
// file to the bucket.
// 3. Get a presigned request to get an object from a bucket.
// 4. Use the net/http package to use the presigned request to download the
// object to a local file.
// 5. Get a presigned request to delete an object from a bucket.
// 6. Use the net/http package to use the presigned request to delete the object.
//
// This example creates an Amazon S3 presign client from the specified sdkConfig
// so that
// you can replace it with a mocked or stubbed config for unit testing.
//
// It uses a questioner from the `demotools` package to get input during the
// example.
// This package can be found in the ../..demotools folder of this repo.
//
// It uses an IHttpRequester interface to abstract HTTP requests so they can be
// mocked
// during testing.
func RunPresigningScenario(ctx context.Context, sdkConfig aws.Config, questioner
demotools.IQuestioner, httpRequester IHttpRequester) {
defer func() {
if r := recover(); r != nil {
log.Println("Something went wrong with the demo.")
_, isMock := questioner.(*demotools.MockQuestioner)
```

```
    if isMock || questioner.AskBool("Do you want to see the full error message (y/n)?", "y") {
        log.Println(r)
    }
}
}()

log.Println(strings.Repeat("-", 88))
log.Println("Welcome to the Amazon S3 presigning demo.")
log.Println(strings.Repeat("-", 88))

s3Client := s3.NewFromConfig(sdkConfig)
bucketBasics := actions.BucketBasics{S3Client: s3Client}
presignClient := s3.NewPresignClient(s3Client)
presigner := actions.Presigner{PresignClient: presignClient}

bucketName := questioner.Ask("We'll need a bucket. Enter a name for a bucket "+
    "you own or one you want to create:", demotools.NotEmpty{})
bucketExists, err := bucketBasics.BucketExists(ctx, bucketName)
if err != nil {
    panic(err)
}
if !bucketExists {
    err = bucketBasics.CreateBucket(ctx, bucketName, sdkConfig.Region)
    if err != nil {
        panic(err)
    } else {
        log.Println("Bucket created.")
    }
}
log.Println(strings.Repeat("-", 88))

log.Printf("Let's presign a request to upload a file to your bucket.")
uploadFilename := questioner.Ask("Enter the path to a file you want to upload:",
    demotools.NotEmpty{})
uploadKey := questioner.Ask("What would you like to name the uploaded object?",
    demotools.NotEmpty{})
uploadFile, err := os.Open(uploadFilename)
if err != nil {
    panic(err)
}
defer uploadFile.Close()
presignedPutRequest, err := presigner.PutObject(ctx, bucketName, uploadKey, 60)
if err != nil {
```

```
    panic(err)
}
log.Printf("Got a presigned %v request to URL:\n\t%v\n",
presignedPutRequest.Method,
presignedPutRequest.URL)
log.Println("Using net/http to send the request...")
info, err := uploadFile.Stat()
if err != nil {
    panic(err)
}
putResponse, err := httpRequester.Put(presignedPutRequest.URL, info.Size(),
uploadFile)
if err != nil {
    panic(err)
}
log.Printf("%v object %v with presigned URL returned %v.",
presignedPutRequest.Method,
uploadKey, putResponse.StatusCode)
log.Println(strings.Repeat("-", 88))

log.Printf("Let's presign a request to download the object.")
questioner.Ask("Press Enter when you're ready.")
presignedGetRequest, err := presigner.GetObject(ctx, bucketName, uploadKey, 60)
if err != nil {
    panic(err)
}
log.Printf("Got a presigned %v request to URL:\n\t%v\n",
presignedGetRequest.Method,
presignedGetRequest.URL)
log.Println("Using net/http to send the request...")
getResponse, err := httpRequester.Get(presignedGetRequest.URL)
if err != nil {
    panic(err)
}
log.Printf("%v object %v with presigned URL returned %v.",
presignedGetRequest.Method,
uploadKey, getResponse.StatusCode)
defer getResponse.Body.Close()
downloadBody, err := io.ReadAll(getResponse.Body)
if err != nil {
    panic(err)
}
log.Printf("Downloaded %v bytes. Here are the first 100 of them:\n",
len(downloadBody))
```

```
log.Println(strings.Repeat("-", 88))
log.Println(string(downloadBody[:100]))
log.Println(strings.Repeat("-", 88))

log.Println("Now we'll create a new request to put the same object using a
presigned post request")
questioner.Ask("Press Enter when you're ready.")
presignPostRequest, err := presigner.PresignPostObject(ctx, bucketName,
uploadKey, 60)
if err != nil {
    panic(err)
}
log.Printf("Got a presigned post request to url %v with values %v\n",
presignPostRequest.URL, presignPostRequest.Values)
log.Println("Using net/http multipart to send the request...")
uploadFile, err = os.Open(uploadFilename)
if err != nil {
    panic(err)
}
defer uploadFile.Close()
multiPartResponse, err := sendMultipartRequest(presignPostRequest.URL,
presignPostRequest.Values, uploadFile, uploadKey, httpRequester)
if err != nil {
    panic(err)
}
log.Printf("Presign post object %v with presigned URL returned %v.", uploadKey,
multiPartResponse.StatusCode)

log.Println("Let's presign a request to delete the object.")
questioner.Ask("Press Enter when you're ready.")
presignedDelRequest, err := presigner.DeleteObject(ctx, bucketName, uploadKey)
if err != nil {
    panic(err)
}
log.Printf("Got a presigned %v request to URL:\n\t%v\n",
presignedDelRequest.Method,
presignedDelRequest.URL)
log.Println("Using net/http to send the request...")
delResponse, err := httpRequester.Delete(presignedDelRequest.URL)
if err != nil {
    panic(err)
}
log.Printf("%v object %v with presigned URL returned %v.\n",
presignedDelRequest.Method,
```

```
    uploadKey, delResponse.StatusCode)
log.Println(strings.Repeat("-", 88))

log.Println("Thanks for watching!")
log.Println(strings.Repeat("-", 88))
}
```

定義範例用來提出 HTTP 請求的 HTTP 請求包裝函式。

```
// IHttpRequester abstracts HTTP requests into an interface so it can be mocked
// during
// unit testing.
type IHttpRequester interface {
    Get(url string) (resp *http.Response, err error)
    Post(url, contentType string, body io.Reader) (resp *http.Response, err error)
    Put(url string, contentLength int64, body io.Reader) (resp *http.Response, err error)
    Delete(url string) (resp *http.Response, err error)
}

// HttpRequester uses the net/http package to make HTTP requests during the
// scenario.
type HttpRequester struct{}

func (httpReq HttpRequester) Get(url string) (resp *http.Response, err error) {
    return http.Get(url)
}

func (httpReq HttpRequester) Post(url, contentType string, body io.Reader) (resp
    *http.Response, err error) {
    postRequest, err := http.NewRequest("POST", url, body)
    if err != nil {
        return nil, err
    }
    postRequest.Header.Set("Content-Type", contentType)
    return http.DefaultClient.Do(postRequest)
}

func (httpReq HttpRequester) Put(url string, contentLength int64, body io.Reader)
    (resp *http.Response, err error) {
    putRequest, err := http.NewRequest("PUT", url, body)
```

```
    if err != nil {
        return nil, err
    }
    putRequest.ContentLength = contentLength
    return http.DefaultClient.Do(putRequest)
}
func (httpReq HttpRequester) Delete(url string) (resp *http.Response, err error)
{
    delRequest, err := http.NewRequest("DELETE", url, nil)
    if err != nil {
        return nil, err
    }
    return http.DefaultClient.Do(delRequest)
}
```

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

以下三個範例示範如何建立預先簽章 URL，並搭配 HTTP 用戶端程式庫使用 URL：

- 搭配三個 HTTP 用戶端程式庫使用 URL 的 HTTP GET 請求
- 標頭中包含中繼資料的 HTTP PUT 請求，搭配三個 HTTP 用戶端程式庫使用 URL
- 具有查詢參數的 HTTP PUT 請求，搭配一個 HTTP 用戶端程式庫使用 URL

產生物件的預先簽章 URL，然後予以下載 (GET 請求)。

匯入。

```
import com.example.s3.util.PresignUrlUtils;
import org.slf4j.Logger;
import software.amazon.awssdk.http.HttpExecuteRequest;
import software.amazon.awssdk.http.HttpExecuteResponse;
```



```
import software.amazon.awssdk.http.SdkHttpClient;
import software.amazon.awssdk.http.SdkHttpMethod;
import software.amazon.awssdk.http.SdkHttpRequest;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;
import
    software.amazon.awssdk.services.s3.presigner.model.GetObjectPresignRequest;
import
    software.amazon.awssdk.services.s3.presigner.model.PresignedGetObjectRequest;
import software.amazon.awssdk.utils.IoUtils;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.URISyntaxException;
import java.net.URL;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;
import java.nio.file.Paths;
import java.time.Duration;
import java.util.UUID;
```

產生 URL。

```
/* Create a pre-signed URL to download an object in a subsequent GET request.
*/
public String createPresignedGetUrl(String bucketName, String keyName) {
    try (S3Presigner presigner = S3Presigner.create()) {

        GetObjectRequest objectRequest = GetObjectRequest.builder()
            .bucket(bucketName)
            .key(keyName)
            .build();

        GetObjectPresignRequest presignRequest =
            GetObjectPresignRequest.builder()
```

```

        .signatureDuration(Duration.ofMinutes(10)) // The URL will
        expire in 10 minutes.
        .getObjectRequest(objectRequest)
        .build();

        PresignedGetObjectRequest presignedRequest =
        presigner.presignGetObject(presignRequest);
        logger.info("Presigned URL: [{}]",
        presignedRequest.url().toString());
        logger.info("HTTP method: [{}]",
        presignedRequest.httpRequest().method());

        return presignedRequest.url().toExternalForm();
    }
}

```

使用下列三種方法之一下載物件。

使用 JDK `URLConnection` (自 v1.1 起) 類別進行下載。

```

/* Use the JDK HttpURLConnection (since v1.1) class to do the download. */
public byte[] useHttpURLConnectionToGet(String presignedUrlString) {
    ByteArrayOutputStream byteArrayOutputStream = new
    ByteArrayOutputStream(); // Capture the response body to a byte array.

    try {
        URL presignedUrl = new URL(presignedUrlString);
        HttpURLConnection connection = (HttpURLConnection)
        presignedUrl.openConnection();
        connection.setRequestMethod("GET");
        // Download the result of executing the request.
        try (InputStream content = connection.getInputStream()) {
            IoUtils.copy(content, byteArrayOutputStream);
        }
        logger.info("HTTP response code is " + connection.getResponseCode());

    } catch (S3Exception | IOException e) {
        logger.error(e.getMessage(), e);
    }
    return byteArrayOutputStream.toByteArray();
}

```

使用 JDK HttpClient (自 v11 起) 類別進行下載。

```
/* Use the JDK HttpClient (since v11) class to do the download. */
public byte[] useHttpClientToGet(String presignedUrlString) {
    ByteArrayOutputStream byteArrayOutputStream = new
    ByteArrayOutputStream(); // Capture the response body to a byte array.

    HttpRequest.Builder requestBuilder = HttpRequest.newBuilder();
    HttpClient httpClient = HttpClient.newHttpClient();
    try {
        URL presignedUrl = new URL(presignedUrlString);
        HttpResponse<InputStream> response = httpClient.send(requestBuilder
            .uri(presignedUrl.toURI())
            .GET()
            .build(),
            HttpResponse.BodyHandlers.ofInputStream());

        IoUtils.copy(response.body(), byteArrayOutputStream);

        logger.info("HTTP response code is " + response.statusCode());
    } catch (URISyntaxException | InterruptedException | IOException e) {
        logger.error(e.getMessage(), e);
    }
    return byteArrayOutputStream.toByteArray();
}
```

使用適用於 Java 的 AWS SDK SdkHttpClient 類別執行下載。

```
/* Use the AWS SDK for Java SdkHttpClient class to do the download. */
public byte[] useSdkHttpClientToGet(String presignedUrlString) {

    ByteArrayOutputStream byteArrayOutputStream = new
    ByteArrayOutputStream(); // Capture the response body to a byte array.
    try {
        URL presignedUrl = new URL(presignedUrlString);
        SdkHttpRequest request = SdkHttpRequest.builder()
            .method(SdkHttpMethod.GET)
            .uri(presignedUrl.toURI())
            .build();

        HttpExecuteRequest executeRequest = HttpExecuteRequest.builder()
```

```

        .request(request)
        .build();

        try (SdkHttpClient sdkHttpClient = ApacheHttpClient.create()) {
            HttpExecuteResponse response =
sdkHttpClient.prepareRequest(executeRequest).call();
            response.responseBody().ifPresentOrElse(
                abortableInputStream -> {
                    try {
                        IoUtils.copy(abortableInputStream,
byteArrayOutputStream);
                    } catch (IOException e) {
                        throw new RuntimeException(e);
                    }
                },
                () -> logger.error("No response body."));

            logger.info("HTTP Response code is {}",
response.httpResponse().statusCode());
        }
        } catch (URISyntaxException | IOException e) {
            logger.error(e.getMessage(), e);
        }
        return byteArrayOutputStream.toByteArray();
    }
}

```

產生一個具有上傳標頭中繼資料的預先簽章 URL，然後上傳檔案 (PUT 請求)。

匯入。

```

import com.example.s3.util.PresignUrlUtils;
import org.slf4j.Logger;
import software.amazon.awssdk.core.internal.sync.FileContentStreamProvider;
import software.amazon.awssdk.http.HttpExecuteRequest;
import software.amazon.awssdk.http.HttpExecuteResponse;
import software.amazon.awssdk.http.SdkHttpClient;
import software.amazon.awssdk.http.SdkHttpMethod;
import software.amazon.awssdk.http.SdkHttpRequest;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;

```

```
import
    software.amazon.awssdk.services.s3.presigner.model.PresignedPutObjectRequest;
import
    software.amazon.awssdk.services.s3.presigner.model.PutObjectPresignRequest;

import java.io.File;
import java.io.IOException;
import java.io.OutputStream;
import java.io.RandomAccessFile;
import java.net.HttpURLConnection;
import java.net.URISyntaxException;
import java.net.URL;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;
import java.nio.ByteBuffer;
import java.nio.channels.FileChannel;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.time.Duration;
import java.util.Map;
import java.util.UUID;
```

產生 URL。

```
/* Create a presigned URL to use in a subsequent PUT request */
public String createPresignedUrl(String bucketName, String keyName,
    Map<String, String> metadata) {
    try (S3Presigner presigner = S3Presigner.create()) {

        PutObjectRequest objectRequest = PutObjectRequest.builder()
            .bucket(bucketName)
            .key(keyName)
            .metadata(metadata)
            .build();

        PutObjectPresignRequest presignRequest =
        PutObjectPresignRequest.builder()
            .signatureDuration(Duration.ofMinutes(10)) // The URL
            expires in 10 minutes.
            .putObjectRequest(objectRequest)
            .build();
```

```

        PresignedPutObjectRequest presignedRequest =
presigner.presignPutObject(presignRequest);
        String myURL = presignedRequest.url().toString();
        logger.info("Presigned URL to upload a file to: [{}]", myURL);
        logger.info("HTTP method: [{}]",
presignedRequest.httpRequest().method());

        return presignedRequest.url().toExternalForm();
    }
}

```

使用下列三種方法之一上傳檔案物件。

使用 JDK `URLConnection` (自 v1.1 起) 類別進行上傳。

```

/* Use the JDK HttpURLConnection (since v1.1) class to do the upload. */
public void useHttpURLConnectionToPut(String presignedUrlString, File
fileToPut, Map<String, String> metadata) {
    logger.info("Begin [{}] upload", fileToPut.toString());
    try {
        URL presignedUrl = new URL(presignedUrlString);
        HttpURLConnection connection = (HttpURLConnection)
presignedUrl.openConnection();
        connection.setDoOutput(true);
        metadata.forEach((k, v) -> connection.setRequestProperty("x-amz-
meta-" + k, v));
        connection.setRequestMethod("PUT");
        OutputStream out = connection.getOutputStream();

        try (RandomAccessFile file = new RandomAccessFile(fileToPut, "r");
            FileChannel inChannel = file.getChannel()) {
            ByteBuffer buffer = ByteBuffer.allocate(8192); //Buffer size is
8k

            while (inChannel.read(buffer) > 0) {
                buffer.flip();
                for (int i = 0; i < buffer.limit(); i++) {
                    out.write(buffer.get());
                }
                buffer.clear();
            }
        }
    }
}

```

```

        } catch (IOException e) {
            logger.error(e.getMessage(), e);
        }

        out.close();
        connection.getResponseCode();
        logger.info("HTTP response code is " + connection.getResponseCode());

    } catch (S3Exception | IOException e) {
        logger.error(e.getMessage(), e);
    }
}

```

使用 JDK `HttpClient` (自 v11 起) 類別進行上傳。

```

/* Use the JDK HttpClient (since v11) class to do the upload. */
public void useHttpClientToPut(String presignedUrlString, File fileToPut,
    Map<String, String> metadata) {
    logger.info("Begin [{}] upload", fileToPut.toString());

    HttpRequest.Builder requestBuilder = HttpRequest.newBuilder();
    metadata.forEach((k, v) -> requestBuilder.header("x-amz-meta-" + k, v));

    HttpClient httpClient = HttpClient.newHttpClient();
    try {
        final HttpResponse<Void> response = httpClient.send(requestBuilder
            .uri(new URL(presignedUrlString).toURI())

            .PUT(HttpRequest.BodyPublishers.ofFile(Path.of(fileToPut.toURI())))
            .build(),
            HttpResponse.BodyHandlers.discarding());

        logger.info("HTTP response code is " + response.statusCode());

    } catch (URISyntaxException | InterruptedException | IOException e) {
        logger.error(e.getMessage(), e);
    }
}

```

使用 AWS for Java V2 `SdkHttpClient` 類別執行上傳。

```

/* Use the AWS SDK for Java V2 SdkHttpClient class to do the upload. */
public void useSdkHttpClientToPut(String presignedUrlString, File fileToPut,
Map<String, String> metadata) {
    logger.info("Begin [{}] upload", fileToPut.toString());

    try {
        URL presignedUrl = new URL(presignedUrlString);

        SdkHttpRequest.Builder requestBuilder = SdkHttpRequest.builder()
            .method(SdkHttpMethod.PUT)
            .uri(presignedUrl.toURI());
        // Add headers
        metadata.forEach((k, v) -> requestBuilder.putHeader("x-amz-meta-" +
k, v));
        // Finish building the request.
        SdkHttpRequest request = requestBuilder.build();

        HttpExecuteRequest executeRequest = HttpExecuteRequest.builder()
            .request(request)
            .contentStreamProvider(new
FileContentStreamProvider(fileToPut.toPath()))
            .build();

        try (SdkHttpClient sdkHttpClient = ApacheHttpClient.create()) {
            HttpExecuteResponse response =
sdkHttpClient.prepareRequest(executeRequest).call();
            logger.info("Response code: {}",
response.httpResponse().statusCode());
        }
    } catch (URISyntaxException | IOException e) {
        logger.error(e.getMessage(), e);
    }
}

```

使用上傳的查詢參數產生預先簽章的 URL，然後上傳檔案 (PUT 請求)。

匯入。

```

import com.example.s3.util.PresignUrlUtils;
import org.slf4j.Logger;
import software.amazon.awssdk.awscore.AWSRequestOverrideConfiguration;
import software.amazon.awssdk.core.internal.sync.FileContentStreamProvider;

```



```

import software.amazon.awssdk.http.HttpExecuteRequest;
import software.amazon.awssdk.http.HttpExecuteResponse;
import software.amazon.awssdk.http.SdkHttpClient;
import software.amazon.awssdk.http.SdkHttpMethod;
import software.amazon.awssdk.http.SdkHttpRequest;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;
import
    software.amazon.awssdk.services.s3.presigner.model.PresignedPutObjectRequest;
import
    software.amazon.awssdk.services.s3.presigner.model.PutObjectPresignRequest;

import java.io.File;
import java.io.IOException;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.file.Paths;
import java.time.Duration;
import java.util.Map;
import java.util.UUID;

```

產生 URL。

```

/**
 * Creates a presigned URL to use in a subsequent HTTP PUT request. The code
 * adds query parameters
 * to the request instead of using headers. By using query parameters, you
 * do not need to add the
 * the parameters as headers when the PUT request is eventually sent.
 *
 * @param bucketName Bucket name where the object will be uploaded.
 * @param keyName Key name of the object that will be uploaded.
 * @param queryParams Query string parameters to be added to the presigned
 * URL.
 * @return
 */
public String createPresignedUrl(String bucketName, String keyName,
    Map<String, String> queryParams) {
    try (S3Presigner presigner = S3Presigner.create()) {
        // Create an override configuration to store the query parameters.

```

```

        AwsRequestOverrideConfiguration.Builder overrideConfigurationBuilder
= AwsRequestOverrideConfiguration.builder();

queryParams.forEach(overrideConfigurationBuilder::putRawQueryParameter);

        PutObjectRequest objectRequest = PutObjectRequest.builder()
                .bucket(bucketName)
                .key(keyName)

.overrideConfiguration(overrideConfigurationBuilder.build()) // Add the override
configuration.

                .build();

        PutObjectPresignRequest presignRequest =
PutObjectPresignRequest.builder()
                .signatureDuration(Duration.ofMinutes(10)) // The URL
expires in 10 minutes.
                .putObjectRequest(objectRequest)
                .build();

        PresignedPutObjectRequest presignedRequest =
presigner.presignPutObject(presignRequest);
        String myURL = presignedRequest.url().toString();
        logger.info("Presigned URL to upload a file to: [{}]", myURL);
        logger.info("HTTP method: [{}]",
presignedRequest.httpRequest().method());

        return presignedRequest.url().toExternalForm();
    }
}

```

使用 AWS for Java V2 SdkHttpClient 類別執行上傳。

```

/**
 * Use the AWS SDK for Java V2 SdkHttpClient class to execute the PUT
request. Since the
 * URL contains the query parameters, no headers are needed for metadata, SSE
settings, or ACL settings.
 *
 * @param presignedUrlString The URL for the PUT request.

```

```
* @param fileToPut File to upload
*/
public void useSdkHttpClientToPut(String presignedUrlString, File fileToPut)
{
    logger.info("Begin [{}] upload", fileToPut.toString());

    try {
        URL presignedUrl = new URL(presignedUrlString);

        SdkHttpRequest.Builder requestBuilder = SdkHttpRequest.builder()
            .method(SdkHttpMethod.PUT)
            .uri(presignedUrl.toURI());

        SdkHttpRequest request = requestBuilder.build();

        HttpExecuteRequest executeRequest = HttpExecuteRequest.builder()
            .request(request)
            .contentStreamProvider(new
FileContentStreamProvider(fileToPut.toPath()))
            .build();

        try (SdkHttpClient sdkHttpClient = ApacheHttpClient.create()) {
            HttpExecuteResponse response =
sdkHttpClient.prepareRequest(executeRequest).call();
            logger.info("Response code: {}",
response.httpResponse().statusCode());
        }
    } catch (URISyntaxException | IOException e) {
        logger.error(e.getMessage(), e);
    }
}
```

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立預先簽署的 URL 將物件上傳至儲存貯體。

```
import https from "node:https";

import { XMLParser } from "fast-xml-parser";
import { PutObjectCommand, S3Client } from "@aws-sdk/client-s3";
import { fromIni } from "@aws-sdk/credential-providers";
import { HttpRequest } from "@smithy/protocol-http";
import {
  getSignedUrl,
  S3RequestPresigner,
} from "@aws-sdk/s3-request-presigner";
import { parseUrl } from "@smithy/url-parser";
import { formatUrl } from "@aws-sdk/util-format-url";
import { Hash } from "@smithy/hash-node";

const createPresignedUrlWithoutClient = async ({ region, bucket, key }) => {
  const url = parseUrl(`https://${bucket}.s3.${region}.amazonaws.com/${key}`);
  const presigner = new S3RequestPresigner({
    credentials: fromIni(),
    region,
    sha256: Hash.bind(null, "sha256"),
  });

  const signedUrlObject = await presigner.presign(
    new HttpRequest({ ...url, method: "PUT" }),
  );
  return formatUrl(signedUrlObject);
};

const createPresignedUrlWithClient = ({ region, bucket, key }) => {
  const client = new S3Client({ region });
  const command = new PutObjectCommand({ Bucket: bucket, Key: key });
  return getSignedUrl(client, command, { expiresIn: 3600 });
};

/**
 * Make a PUT request to the provided URL.
 *
 * @param {string} url
 * @param {string} data
 */
const put = (url, data) => {
  return new Promise((resolve, reject) => {
```

```

const req = https.request(
  url,
  { method: "PUT", headers: { "Content-Length": new Blob([data]).size } },
  (res) => {
    let responseBody = "";
    res.on("data", (chunk) => {
      responseBody += chunk;
    });
    res.on("end", () => {
      const parser = new XMLParser();
      if (res.statusCode >= 200 && res.statusCode <= 299) {
        resolve(parser.parse(responseBody, true));
      } else {
        reject(parser.parse(responseBody, true));
      }
    });
  },
);
req.on("error", (err) => {
  reject(err);
});
req.write(data);
req.end();
});
};

/**
 * Create two presigned urls for uploading an object to an S3 bucket.
 * The first presigned URL is created with credentials from the shared INI file
 * in the current environment. The second presigned URL is created using an
 * existing S3Client instance that has already been provided with credentials.
 * @param {{ bucketName: string, key: string, region: string }}
 */
export const main = async ({ bucketName, key, region }) => {
  try {
    const noClientUrl = await createPresignedUrlWithoutClient({
      bucket: bucketName,
      key,
      region,
    });

    const clientUrl = await createPresignedUrlWithClient({
      bucket: bucketName,
      region,

```

```

        key,
    });

    // After you get the presigned URL, you can provide your own file
    // data. Refer to put() above.
    console.log("Calling PUT using presigned URL without client");
    await put(noClientUrl, "Hello World");

    console.log("Calling PUT using presigned URL with client");
    await put(clientUrl, "Hello World");

    console.log("\nDone. Check your S3 console.");
  } catch (caught) {
    if (caught instanceof Error && caught.name === "CredentialsProviderError") {
      console.error(
        `There was an error getting your credentials. Are your local credentials
        configured?\n${caught.name}: ${caught.message}`,
      );
    } else {
      throw caught;
    }
  }
};

```

建立預先簽署的 URL 從儲存貯體下載物件。

```

import { GetObjectCommand, S3Client } from "@aws-sdk/client-s3";
import { fromIni } from "@aws-sdk/credential-providers";
import { HttpRequest } from "@smithy/protocol-http";
import {
  getSignedUrl,
  S3RequestPresigner,
} from "@aws-sdk/s3-request-presigner";
import { parseUrl } from "@smithy/url-parser";
import { formatUrl } from "@aws-sdk/util-format-url";
import { Hash } from "@smithy/hash-node";

const createPresignedUrlWithoutClient = async ({ region, bucket, key }) => {
  const url = parseUrl(`https://${bucket}.s3.${region}.amazonaws.com/${key}`);
  const presigner = new S3RequestPresigner({
    credentials: fromIni(),
    region,
  });

```

```
    sha256: Hash.bind(null, "sha256"),
  });

  const signedUrlObject = await presigner.presign(new HttpRequest(url));
  return formatUrl(signedUrlObject);
};

const createPresignedUrlWithClient = ({ region, bucket, key }) => {
  const client = new S3Client({ region });
  const command = new GetObjectCommand({ Bucket: bucket, Key: key });
  return getSignedUrl(client, command, { expiresIn: 3600 });
};

/**
 * Create two presigned urls for downloading an object from an S3 bucket.
 * The first presigned URL is created with credentials from the shared INI file
 * in the current environment. The second presigned URL is created using an
 * existing S3Client instance that has already been provided with credentials.
 * @param {{ bucketName: string, key: string, region: string }}
 */
export const main = async ({ bucketName, key, region }) => {
  try {
    const noClientUrl = await createPresignedUrlWithoutClient({
      bucket: bucketName,
      region,
      key,
    });

    const clientUrl = await createPresignedUrlWithClient({
      bucket: bucketName,
      region,
      key,
    });

    console.log("Presigned URL without client");
    console.log(noClientUrl);
    console.log("\n");

    console.log("Presigned URL with client");
    console.log(clientUrl);
  } catch (caught) {
    if (caught instanceof Error && caught.name === "CredentialsProviderError") {
      console.error(
```

```
        `There was an error getting your credentials. Are your local credentials
        configured?\n${caught.name}: ${caught.message}`,
    );
    } else {
        throw caught;
    }
}
};
```

- 如需詳細資訊，請參閱《[適用於 JavaScript 的 AWS SDK 開發人員指南](#)》。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立 GetObject 預先簽章的請求，並使用 URL 下載物件。

```
suspend fun getObjectPresigned(
    s3: S3Client,
    bucketName: String,
    keyName: String,
): String {
    // Create a GetObjectRequest.
    val unsignedRequest =
        GetObjectRequest {
            bucket = bucketName
            key = keyName
        }

    // Presign the GetObject request.
    val presignedRequest = s3.presignGetObject(unsignedRequest, 24.hours)

    // Use the URL from the presigned HttpRequest in a subsequent HTTP GET
    request to retrieve the object.
    val objectContents = URL(presignedRequest.url.toString()).readText()
```



```
    return objectContents
}
```

使用進階選項建立 `GetObject` 預先簽章的請求。

```
suspend fun getObjectPresignedMoreOptions(
    s3: S3Client,
    bucketName: String,
    keyName: String,
): HttpRequest {
    // Create a GetObjectRequest.
    val unsignedRequest =
        GetObjectRequest {
            bucket = bucketName
            key = keyName
        }

    // Presign the GetObject request.
    val presignedRequest =
        s3.presignGetObject(unsignedRequest, signer = CrtAwsSigner) {
            signingDate = Instant.now() + 12.hours // Presigned request can be
            used 12 hours from now.
            algorithm = AwsSigningAlgorithm.SIGV4_ASYMMETRIC
            signatureType = AwsSignatureType.HTTP_REQUEST_VIA_QUERY_PARAMS
            expiresAfter = 8.hours // Presigned request expires 8 hours later.
        }
    return presignedRequest
}
```

建立 `PutObject` 預先簽章的請求，並使用該請求上傳物件。

```
suspend fun putObjectPresigned(
    s3: S3Client,
    bucketName: String,
    keyName: String,
    content: String,
) {
    // Create a PutObjectRequest.
    val unsignedRequest =
        PutObjectRequest {
```

```
        bucket = bucketName
        key = keyName
    }

    // Presign the request.
    val presignedRequest = s3.presignPutObject(unsignedRequest, 24.hours)

    // Use the URL and any headers from the presigned HttpRequest in a subsequent
    HTTP PUT request to retrieve the object.
    // Create a PUT request using the OkHttpClient API.
    val putRequest =
        Request
            .Builder()
            .url(presignedRequest.url.toString())
            .apply {
                presignedRequest.headers.forEach { key, values ->
                    header(key, values.joinToString(", "))
                }
            }
            .put(content.toRequestBody())
            .build()

    val response = OkHttpClient().newCall(putRequest).execute()
    assert(response.isSuccessful)
}
```

- 如需詳細資訊，請參閱[適用於 Kotlin 的 AWS SDK 開發人員指南](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
namespace S3;
use Aws\Exception\AwsException;
use Aws\Utilities\PrintableLineBreak;
use Aws\Utilities\TestableReadline;
```

```
use DateTime;

require 'vendor/autoload.php';

class PresignedURL
{
    use PrintableLineBreak;
    use TestableReadline;

    public function run()
    {
        $s3Service = new S3Service();

        $expiration = new DateTime("+20 minutes");
        $linebreak = $this->getLineBreak();

        echo $linebreak;
        echo ("Welcome to the Amazon S3 presigned URL demo.\n");
        echo $linebreak;

        $bucket = $this->testable_readline("First, please enter the name of the
S3 bucket to use: ");
        $key = $this->testable_readline("Next, provide the key of an object in
the given bucket: ");
        echo $linebreak;
        $command = $s3Service->getClient()->getCommand('GetObject', [
            'Bucket' => $bucket,
            'Key' => $key,
        ]);
        try {
            $preSignedUrl = $s3Service->preSignedUrl($command, $expiration);
            echo "Your preSignedUrl is \n$preSignedUrl\nand will be good for the
next 20 minutes.\n";
            echo $linebreak;
            echo "Thanks for trying the Amazon S3 presigned URL demo.\n";
        } catch (AwsException $exception) {
            echo $linebreak;
            echo "Something went wrong: $exception";
            die();
        }
    }
}

$runner = new PresignedURL();
```

```
$runner->run();

namespace S3;

use Aws\CommandInterface;
use Aws\Exception\AwsException;
use Aws\Result;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use AwsUtilities\AWSServiceClass;
use DateTimeInterface;

class S3Service extends AWSServiceClass
{
    protected S3Client $client;
    protected bool $verbose;

    public function __construct(S3Client $client = null, $verbose = false)
    {
        if ($client) {
            $this->client = $client;
        } else {
            $this->client = new S3Client([
                'version' => 'latest',
                'region' => 'us-west-2',
            ]);
        }
        $this->verbose = $verbose;
    }

    public function setVerbose($verbose)
    {
        $this->verbose = $verbose;
    }

    public function isVerbose(): bool
    {
        return $this->verbose;
    }

    public function getClient(): S3Client
    {

```

```
        return $this->client;
    }

    public function setClient(S3Client $client)
    {
        $this->client = $client;
    }

    public function emptyAndDeleteBucket($bucketName, array $args = [])
    {
        try {
            $objects = $this->listAllObjects($bucketName, $args);
            $this->deleteObjects($bucketName, $objects, $args);
            if ($this->verbose) {
                echo "Deleted all objects and folders from $bucketName.\n";
            }
            $this->deleteBucket($bucketName, $args);
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to delete $bucketName with error: {$exception-
>getMessage()}\n";
                echo "\nPlease fix error with bucket deletion before continuing.
\n";
            }
            throw $exception;
        }
    }

    public function createBucket(string $bucketName, array $args = [])
    {
        $parameters = array_merge(['Bucket' => $bucketName], $args);
        try {
            $this->client->createBucket($parameters);
            if ($this->verbose) {
                echo "Created the bucket named: $bucketName.\n";
            }
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to create $bucketName with error: {$exception-
>getMessage()}\n";
                echo "Please fix error with bucket creation before continuing.";
            }
        }
    }
}
```

```

        }
        throw $exception;
    }
}

public function putObject(string $bucketName, string $key, array $args = [])
{
    $parameters = array_merge(['Bucket' => $bucketName, 'Key' => $key],
    $args);
    try {
        $this->client->putObject($parameters);
        if ($this->verbose) {
            echo "Uploaded the object named: $key to the bucket named:
    $bucketName.\n";
        }
    } catch (AwsException $exception) {
        if ($this->verbose) {
            echo "Failed to create $key in $bucketName with error:
    {$exception->getMessage()}\n";
            echo "Please fix error with object uploading before continuing.";
        }
        throw $exception;
    }
}

public function getObject(string $bucketName, string $key, array $args = []):
Result
{
    $parameters = array_merge(['Bucket' => $bucketName, 'Key' => $key],
    $args);
    try {
        $object = $this->client->getObject($parameters);
        if ($this->verbose) {
            echo "Downloaded the object named: $key to the bucket named:
    $bucketName.\n";
        }
    } catch (AwsException $exception) {
        if ($this->verbose) {
            echo "Failed to download $key from $bucketName with error:
    {$exception->getMessage()}\n";

```

```

        echo "Please fix error with object downloading before
continuing.";
    }
    throw $exception;
}
return $object;
}

public function copyObject($bucketName, $key, $copySource, array $args = [])
{
    $parameters = array_merge(['Bucket' => $bucketName, 'Key' => $key,
"CopySource" => $copySource], $args);
    try {
        $this->client->copyObject($parameters);
        if ($this->verbose) {
            echo "Copied the object from: $copySource in $bucketName to:
$key.\n";
        }
    } catch (AwsException $exception) {
        if ($this->verbose) {
            echo "Failed to copy $copySource in $bucketName with error:
{$exception->getMessage()}\n";
            echo "Please fix error with object copying before continuing.";
        }
        throw $exception;
    }
}

public function listObjects(string $bucketName, $start = 0, $max = 1000,
array $args = [])
{
    $parameters = array_merge(['Bucket' => $bucketName, 'Marker' => $start,
"MaxKeys" => $max], $args);
    try {
        $objects = $this->client->listObjectsV2($parameters);
        if ($this->verbose) {
            echo "Retrieved the list of objects from: $bucketName.\n";
        }
    } catch (AwsException $exception) {
        if ($this->verbose) {

```

```
        echo "Failed to retrieve the objects from $bucketName with error:
{$exception->getMessage()}\n";
        echo "Please fix error with list objects before continuing.";
    }
    throw $exception;
}
return $objects;
}

public function listAllObjects($bucketName, array $args = [])
{
    $parameters = array_merge(['Bucket' => $bucketName], $args);

    $contents = [];
    $paginator = $this->client->getPaginator("ListObjectsV2", $parameters);

    foreach ($paginator as $result) {
        if($result['KeyCount'] == 0){
            break;
        }
        foreach ($result['Contents'] as $object) {
            $contents[] = $object;
        }
    }
    return $contents;
}

public function deleteObjects(string $bucketName, array $objects, array $args
= [])
{
    $listOfObjects = array_map(
        function ($object) {
            return ['Key' => $object];
        },
        array_column($objects, 'Key')
    );
    if(!$listOfObjects){
        return;
    }
}
```



```

        $parameters = array_merge(['Bucket' => $bucketName, 'Delete' =>
['Objects' => $listOfObjects]], $args);
        try {
            $this->client->deleteObjects($parameters);
            if ($this->verbose) {
                echo "Deleted the list of objects from: $bucketName.\n";
            }
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to delete the list of objects from $bucketName with
error: {$exception->getMessage()}\n";
                echo "Please fix error with object deletion before continuing.";
            }
            throw $exception;
        }
    }

    public function deleteBucket(string $bucketName, array $args = [])
    {
        $parameters = array_merge(['Bucket' => $bucketName], $args);
        try {
            $this->client->deleteBucket($parameters);
            if ($this->verbose) {
                echo "Deleted the bucket named: $bucketName.\n";
            }
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to delete $bucketName with error: {$exception-
>getMessage()}\n";
                echo "Please fix error with bucket deletion before continuing.";
            }
            throw $exception;
        }
    }

    public function deleteObject(string $bucketName, string $fileName, array
$args = [])
    {
        $parameters = array_merge(['Bucket' => $bucketName, 'Key' => $fileName],
$args);

```

```

        try {
            $this->client->deleteObject($parameters);
            if ($this->verbose) {
                echo "Deleted the object named: $fileName from $bucketName.\n";
            }
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to delete $fileName from $bucketName with error:
{$exception->getMessage()}\n";
                echo "Please fix error with object deletion before continuing.";
            }
            throw $exception;
        }
    }

    public function listBuckets(array $args = [])
    {
        try {
            $buckets = $this->client->listBuckets($args);
            if ($this->verbose) {
                echo "Retrieved all " . count($buckets) . "\n";
            }
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to retrieve bucket list with error: {$exception-
>getMessage()}\n";
                echo "Please fix error with bucket lists before continuing.";
            }
            throw $exception;
        }
        return $buckets;
    }

    public function preSignedUrl(CommandInterface $command, DateTimeInterface|
int|string $expires, array $options = [])
    {
        $request = $this->client->createPresignedRequest($command, $expires,
$options);
        try {
            $presignedUrl = (string)$request->getUri();

```

```
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to create a presigned url: {$exception-
>getMessage()}\n";
                echo "Please fix error with presigned urls before continuing.";
            }
            throw $exception;
        }
        return $presignedUrl;
    }

    public function createSession(string $bucketName)
    {
        try{
            $result = $this->client->createSession([
                'Bucket' => $bucketName,
            ]);
            return $result;
        }catch(S3Exception $caught){
            if($caught->getAwsErrorType() == "NoSuchBucket"){
                echo "The specified bucket does not exist.";
            }
            throw $caught;
        }
    }
}
```

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

產生可以於限定時間內執行 S3 動作的預先簽署的 URL。使用 Requests package 搭配 URL 發出請求。

```
import argparse
import logging
import boto3
from botocore.exceptions import ClientError
import requests

logger = logging.getLogger(__name__)

def generate_presigned_url(s3_client, client_method, method_parameters,
                           expires_in):
    """
    Generate a presigned Amazon S3 URL that can be used to perform an action.

    :param s3_client: A Boto3 Amazon S3 client.
    :param client_method: The name of the client method that the URL performs.
    :param method_parameters: The parameters of the specified client method.
    :param expires_in: The number of seconds the presigned URL is valid for.
    :return: The presigned URL.
    """
    try:
        url = s3_client.generate_presigned_url(
            ClientMethod=client_method, Params=method_parameters,
            ExpiresIn=expires_in
        )
        logger.info("Got presigned URL: %s", url)
    except ClientError:
        logger.exception(
            "Couldn't get a presigned URL for client method '%s'.", client_method
        )
        raise
    return url

def usage_demo():
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Welcome to the Amazon S3 presigned URL demo.")
    print("-" * 88)
```

```

parser = argparse.ArgumentParser()
parser.add_argument("bucket", help="The name of the bucket.")
parser.add_argument(
    "key",
    help="For a GET operation, the key of the object in Amazon S3. For a "
    "PUT operation, the name of a file to upload.",
)
parser.add_argument("action", choices=("get", "put"), help="The action to
perform.")
args = parser.parse_args()

s3_client = boto3.client("s3")
client_action = "get_object" if args.action == "get" else "put_object"
url = generate_presigned_url(
    s3_client, client_action, {"Bucket": args.bucket, "Key": args.key}, 1000
)

print("Using the Requests package to send a request to the URL.")
response = None
if args.action == "get":
    response = requests.get(url)
    if response.status_code == 200:
        with open(args.key.split("/")[-1], 'wb') as object_file:
            object_file.write(response.content)
elif args.action == "put":
    print("Putting data to the URL.")
    try:
        with open(args.key, "rb") as object_file:
            object_text = object_file.read()
            response = requests.put(url, data=object_text)
    except FileNotFoundError:
        print(
            f"Couldn't find {args.key}. For a PUT operation, the key must be
the "
            f"name of a file that exists on your computer."
        )

if response is not None:
    print(f"Status: {response.status_code}\nReason: {response.reason}")

print("-" * 88)

```

```
if __name__ == "__main__":
    usage_demo()
```

產生預先簽署的 POST 請求以上傳檔案。

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
        Boto3
                        that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name

    def generate_presigned_post(self, object_key, expires_in):
        """
        Generate a presigned Amazon S3 POST request to upload a file.
        A presigned POST can be used for a limited time to let someone without an
        AWS
        account upload a file to a bucket.

        :param object_key: The object key to identify the uploaded object.
        :param expires_in: The number of seconds the presigned POST is valid.
        :return: A dictionary that contains the URL and form fields that contain
                required access data.
        """
        try:
            response = self.bucket.meta.client.generate_presigned_post(
                Bucket=self.bucket.name, Key=object_key, ExpiresIn=expires_in
            )
            logger.info("Got presigned POST URL: %s", response["url"])
        except ClientError:
            logger.exception(
                "Couldn't get a presigned POST URL for bucket '%s' and object
                '%s'",
                self.bucket.name,
                object_key,
            )
```

```
        raise
    return response
```

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-s3'
require 'net/http'

# Creates a presigned URL that can be used to upload content to an object.
#
# @param bucket [Aws::S3::Bucket] An existing Amazon S3 bucket.
# @param object_key [String] The key to give the uploaded object.
# @return [URI, nil] The parsed URI if successful; otherwise nil.
def get_presigned_url(bucket, object_key)
  url = bucket.object(object_key).presigned_url(:put)
  puts "Created presigned URL: #{url}"
  URI(url)
rescue Aws::Errors::ServiceError => e
  puts "Couldn't create presigned URL for #{bucket.name}:#{object_key}. Here's
  why: #{e.message}"
end

# Example usage:
def run_demo
  bucket_name = "amzn-s3-demo-bucket"
  object_key = "my-file.txt"
  object_content = "This is the content of my-file.txt."

  bucket = Aws::S3::Bucket.new(bucket_name)
  presigned_url = get_presigned_url(bucket, object_key)
```

```
return unless presigned_url

response = Net::HTTP.start(presigned_url.host) do |http|
  http.send_request('PUT', presigned_url.request_uri, object_content,
'content_type' => '')
end

case response
when Net::HTTPSuccess
  puts 'Content uploaded!'
else
  puts response.value
end
end

run_demo if $PROGRAM_NAME == __FILE__
```

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立對 GET S3 物件的預先簽署請求。

```
/// Generate a URL for a presigned GET request.
async fn get_object(
  client: &Client,
  bucket: &str,
  object: &str,
  expires_in: u64,
) -> Result<(), Box<dyn Error>> {
  let expires_in = Duration::from_secs(expires_in);
  let presigned_request = client
    .get_object()
    .bucket(bucket)
    .key(object)
```



```

        .presigned(PresigningConfig::expires_in(expires_in?))
        .await?;

println!("Object URI: {}", presigned_request.uri());
let valid_until = chrono::offset::Local::now() + expires_in;
println!("Valid until: {valid_until}");

Ok(())
}

```

建立對 PUT S3 物件的預先簽署請求。

```

async fn put_object(
    client: &Client,
    bucket: &str,
    object: &str,
    expires_in: u64,
) -> Result<String, S3ExampleError> {
    let expires_in: std::time::Duration =
        std::time::Duration::from_secs(expires_in);
    let expires_in: aws_sdk_s3::presigning::PresigningConfig =
        PresigningConfig::expires_in(expires_in).map_err(|err| {
            S3ExampleError::new(format!(
                "Failed to convert expiration to PresigningConfig: {err:?}"
            ))
        })?;
    let presigned_request = client
        .put_object()
        .bucket(bucket)
        .key(object)
        .presigned(expires_in)
        .await?;

    Ok(presigned_request.uri().into())
}

```

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立對 GET S3 物件的預先簽署請求。

```
" iv_bucket_name is the bucket name
" iv_key is the object name like "myfile.txt"

DATA(lo_session) = /aws1/cl_rt_session_aws=>create( cv_pfl ).
DATA(lo_s3) = /aws1/cl_s3_factory=>create( lo_session ).

"Upload a nice Hello World file to an S3 bucket."
TRY.
    DATA(lv_contents) = cl_abap_codepage=>convert_to( 'Hello, World' ).
    lo_s3->putobject(
        iv_bucket = iv_bucket_name
        iv_key = iv_key
        iv_body = lv_contents
        iv_contenttype = 'text/plain' ).
    MESSAGE 'Object uploaded to S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.

" now generate a presigned URL with a 600-second expiration
DATA(lo_presigner) = lo_s3->get_presigner( iv_expires_sec = 600 ).
" the presigner getobject() method has the same signature as
" lo_s3->getobject(), but it doesn't actually make the call
" to the service. It just prepares a presigned URL for a future call
DATA(lo_presigned_req) = lo_presigner->getobject(
    iv_bucket = iv_bucket_name
    iv_key = iv_key ).

" You can provide this URL to a web page, user, email etc so they
" can retrieve the file. The URL will expire in 10 minutes.
ov_url = lo_presigned_req->get_url( ).
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

建立相片資產管理應用程式，讓使用者以標籤管理相片

下列程式碼範例示範如何建立無伺服器應用程式，讓使用者以標籤管理相片。

.NET

適用於 .NET 的 SDK

顯示如何開發照片資產管理應用程式，以便使用 Amazon Rekognition 偵測圖片中的標籤，並將其儲存以供日後擷取。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

如要深入探索此範例的來源，請參閱 [AWS 社群](#) 上的文章。

此範例中使用的服務

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

C++

適用於 C++ 的 SDK

顯示如何開發照片資產管理應用程式，以便使用 Amazon Rekognition 偵測圖片中的標籤，並將其儲存以供日後擷取。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

如要深入探索此範例的來源，請參閱 [AWS 社群](#) 上的文章。

此範例中使用的服務

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Java

適用於 Java 2.x 的 SDK

顯示如何開發照片資產管理應用程式，以便使用 Amazon Rekognition 偵測圖片中的標籤，並將其儲存以供日後擷取。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

如要深入探索此範例的來源，請參閱 [AWS 社群](#) 上的文章。

此範例中使用的服務

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

JavaScript

適用於 JavaScript (v3) 的 SDK

顯示如何開發照片資產管理應用程式，以便使用 Amazon Rekognition 偵測圖片中的標籤，並將其儲存以供日後擷取。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

如要深入探索此範例的來源，請參閱 [AWS 社群](#) 上的文章。

此範例中使用的服務

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Kotlin

SDK for Kotlin

顯示如何開發照片資產管理應用程式，以便使用 Amazon Rekognition 偵測圖片中的標籤，並將其儲存以供日後擷取。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

若要深入探索此範例的來源，請參閱 [AWS 社群](#) 上的文章。

此範例中使用的服務

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

PHP

適用於 PHP 的 SDK

顯示如何開發照片資產管理應用程式，以便使用 Amazon Rekognition 偵測圖片中的標籤，並將其儲存以供日後擷取。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

如要深入探索此範例的來源，請參閱 [AWS 社群](#) 上的文章。

此範例中使用的服務

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Rust

適用於 Rust 的 SDK

顯示如何開發照片資產管理應用程式，以便使用 Amazon Rekognition 偵測圖片中的標籤，並將其儲存以供日後擷取。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

如要深入探索此範例的來源，請參閱 [AWS 社群](#) 上的文章。

此範例中使用的服務

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 列出 Amazon S3 物件的網頁

下列程式碼範例顯示如何在網頁中列出 Amazon S3 物件。

JavaScript

SDK for JavaScript (v3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

下列程式碼是對 AWS SDK 進行呼叫的相關 React 元件。您可以在前面的 GitHub 連結中找到包含此元件之應用程式的可執行版本。

```
import { useEffect, useState } from "react";
import {
  ListObjectsCommand,
  type ListObjectsCommandOutput,
  S3Client,
} from "@aws-sdk/client-s3";
import { fromCognitoIdentityPool } from "@aws-sdk/credential-providers";
import "./App.css";

function App() {
  const [objects, setObjects] = useState<
    Required<ListObjectsCommandOutput>["Contents"]
  >([]);

  useEffect(() => {
    const client = new S3Client({
      region: "us-east-1",
      // Unless you have a public bucket, you'll need access to a private bucket.
      // One way to do this is to create an Amazon Cognito identity pool, attach
      // a role to the pool,
      // and grant the role access to the 's3:GetObject' action.
      //
      // You'll also need to configure the CORS settings on the bucket to allow
      // traffic from
      // this example site. Here's an example configuration that allows all
      // origins. Don't
      // do this in production.
      // [
      // {

```

```
//    "AllowedHeaders": ["*"],
//    "AllowedMethods": ["GET"],
//    "AllowedOrigins": ["*"],
//    "ExposeHeaders": [],
//  },
//]
//
credentials: fromCognitoIdentityPool({
  clientConfig: { region: "us-east-1" },
  identityPoolId: "<YOUR_IDENTITY_POOL_ID>",
}),
});
const command = new ListObjectsCommand({ Bucket: "bucket-name" });
client.send(command).then(({ Contents }) => setObjects(Contents || []));
}, []);

return (
  <div className="App">
    {objects.map((o) => (
      <div key={o.ETag}>{o.Key}</div>
    ))}
  </div>
);
}

export default App;
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 [ListObjects](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

建立 Amazon Textract Explorer 應用程式

下列程式碼範例示範如何透過互動式應用程式探索 Amazon Textract 輸出。

JavaScript

適用於 JavaScript (v3) 的 SDK

示範如何使用 適用於 JavaScript 的 AWS SDK 建置 React 應用程式，該應用程式使用 Amazon Textract 從文件映像擷取資料，並將其顯示在互動式網頁中。此範例會在 Web 瀏覽器中執

行，且登入資料需要經過驗證的 Amazon Cognito 身分。它使用 Amazon Simple Storage Service (Amazon S3 進行儲存，對於通知，它會輪詢訂閱 Amazon Simple Notification Service (Amazon SNS)) 主題的 Amazon Simple Queue Service (Amazon SQS) 佇列。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Cognito Identity
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Python

適用於 Python 的 SDK (Boto3)

顯示如何使用 適用於 Python (Boto3) 的 AWS SDK 搭配 Amazon Textract 來偵測文件映像中的文字、表單和資料表元素。輸入影像和 Amazon Textract 輸出會顯示在 Tkinter 應用程式中，可讓您探索偵測到的元素。

- 將文件影像提交到 Amazon Textract，並探索偵測到元素的輸出。
- 將影像直接傳送至 Amazon Textract 或透過 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
- 使用非同步 API 可以在任務完成時啟動將通知發布到 Amazon Simple Notification Service (Amazon SNS) 主題的任務。
- 輪詢 Amazon Simple Queue Service (Amazon SQS) 佇列以取得任務完成訊息並顯示結果。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Cognito Identity
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 刪除指定 Amazon S3 儲存貯體中的所有物件

下列程式碼範例示範如何刪除 Amazon S3 儲存貯體中的所有物件。

JavaScript

SDK for JavaScript (v3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除給定 Amazon S3 儲存貯體的所有物件。

```
import {
  DeleteObjectsCommand,
  paginateListObjectsV2,
  S3Client,
} from "@aws-sdk/client-s3";

/**
 *
 * @param {{ bucketName: string }} config
 */
export const main = async ({ bucketName }) => {
  const client = new S3Client({});
  try {
    console.log(`Deleting all objects in bucket: ${bucketName}`);

    const paginator = paginateListObjectsV2(
      { client },
      {
        Bucket: bucketName,
      },
    );

    const objectKeys = [];
```

```
for await (const { Contents } of paginator) {
  objectKeys.push(...Contents.map((obj) => ({ Key: obj.Key })));
}

const deleteCommand = new DeleteObjectsCommand({
  Bucket: bucketName,
  Delete: { Objects: objectKeys },
});

await client.send(deleteCommand);

console.log(`All objects deleted from bucket: ${bucketName}`);
} catch (caught) {
  if (caught instanceof Error) {
    console.error(
      `Failed to empty ${bucketName}. ${caught.name}: ${caught.message}`,
    );
  }
}
};

// Call function if run directly.
import { fileURLToPath } from "node:url";
import { parseArgs } from "node:util";
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  const options = {
    bucketName: {
      type: "string",
    },
  };

  const { values } = parseArgs({ options });
  main(values);
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的下列主題。
 - [DeleteObjects](#)
 - [ListObjectsV2](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 刪除不完整的分段上傳至 Amazon S3

下列程式碼範例示範如何刪除或停止未完成的 Amazon S3 分段上傳。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

若要停止進行中，或因任何原因未完成的分段上傳，您可以取得上傳清單，然後將其刪除，如下列範例所示。

```
/**
 * Aborts all incomplete multipart uploads from the specified S3 bucket.
 *
 * @param bucketName the name of the S3 bucket
 */
public static void abortIncompleteMultipartUploadsFromList(String bucketName)
{
    ListMultipartUploadsRequest listMultipartUploadsRequest =
ListMultipartUploadsRequest.builder()
        .bucket(bucketName)
        .build();

    ListMultipartUploadsResponse response =
s3Client.listMultipartUploads(listMultipartUploadsRequest);
    List<MultipartUpload> uploads = response.uploads();

    AbortMultipartUploadRequest abortMultipartUploadRequest;
    for (MultipartUpload upload : uploads) {
        abortMultipartUploadRequest = AbortMultipartUploadRequest.builder()
            .bucket(bucketName)
            .key(upload.key())
            .expectedBucketOwner(accountId)
            .uploadId(upload.uploadId())
            .build();
    }
}
```

```

        AbortMultipartUploadResponse abortMultipartUploadResponse =
s3Client.abortMultipartUpload(abortMultipartUploadRequest);
        if (abortMultipartUploadResponse.sdkHttpResponse().isSuccessful()) {
            logger.info("Upload ID [{}] to bucket [{}] successfully
aborted.", upload.uploadId(), bucketName);
        }
    }
}

```

若要刪除某日期之前或之後啟動的未完成分段上傳，您可以根據時間點選擇性地刪除分段上傳，如下列範例所示。

```

/**
 * Aborts incomplete multipart uploads older than the specified point in
time.
 *
 * @param bucketName the name of the S3 bucket
 * @param pointInTime the cutoff time; uploads initiated before this are
aborted
 */
static void abortIncompleteMultipartUploadsOlderThan(String bucketName,
Instant pointInTime) {
    ListMultipartUploadsRequest listMultipartUploadsRequest =
ListMultipartUploadsRequest.builder()
        .bucket(bucketName)
        .build();

    ListMultipartUploadsResponse response =
s3Client.listMultipartUploads(listMultipartUploadsRequest);
    List<MultipartUpload> uploads = response.uploads();

    AbortMultipartUploadRequest abortMultipartUploadRequest;
    for (MultipartUpload upload : uploads) {
        logger.info("Found multipartUpload with upload ID [{}], initiated
[{}]", upload.uploadId(), upload.initiated());
        if (upload.initiated().isBefore(pointInTime)) {
            abortMultipartUploadRequest =
AbortMultipartUploadRequest.builder()
                .bucket(bucketName)
                .key(upload.key())
                .expectedBucketOwner(accountId)
                .uploadId(upload.uploadId())

```

```

        .build();

        AbortMultipartUploadResponse abortMultipartUploadResponse =
s3Client.abortMultipartUpload(abortMultipartUploadRequest);
        if
(abortMultipartUploadResponse.sdkHttpResponse().isSuccessful()) {
            logger.info("Upload ID [{}] to bucket [{}] successfully
aborted.", upload.uploadId(), bucketName);
        }
    }
}
}
}

```

如果您可以在開始分段上傳後存取上傳 ID，便可以使用 ID 刪除進行中的上傳。

```

/**
 * Aborts a multipart upload using the upload ID.
 *
 * @param bucketName the name of the S3 bucket
 * @param key         the object key
 */
static void abortMultipartUploadUsingUploadId(String bucketName, String key)
{
    String uploadId = startUploadReturningUploadId(bucketName, key);
    AbortMultipartUploadResponse response = s3Client.abortMultipartUpload(b -
> b
        .uploadId(uploadId)
        .bucket(bucketName)
        .key(key));

    if (response.sdkHttpResponse().isSuccessful()) {
        logger.info("Upload ID [{}] to bucket [{}] successfully aborted.",
uploadId, bucketName);
    }
}
}

```

若要不續刪除某特定天數之前未完成的分段上傳，請設定儲存貯體的儲存貯體生命週期組態。下列範例示範如何建立規則，以刪除超過 7 天的未完成上傳。

```
/**
```

```
    * Configures a lifecycle rule to abort incomplete multipart uploads after 7
    days.
    *
    * @param bucketName the name of the S3 bucket
    */
    static void abortMultipartUploadsUsingLifecycleConfig(String bucketName) {
        Collection<LifecycleRule> lifeCycleRules =
        List.of(LifecycleRule.builder()
            .abortIncompleteMultipartUpload(b -> b.
                daysAfterInitiation(7))
            .status("Enabled")
            .filter(SdkBuilder::build) // Filter element is required.
            .build());

        PutBucketLifecycleConfigurationResponse response =
        s3Client.putBucketLifecycleConfiguration(b -> b
            .bucket(bucketName)
            .lifecycleConfiguration(b1 -> b1.rules(lifeCycleRules)));

        if (response.sdkHttpResponse().isSuccessful()) {
            logger.info("Rule to abort incomplete multipart uploads added to
            bucket.");
        } else {
            logger.error("Unsuccessfully applied rule. HTTP status code is [{}]",
            response.sdkHttpResponse().statusCode());
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
 - [AbortMultipartUpload](#)
 - [ListMultipartUploads](#)
 - [PutBucketLifecycleConfiguration](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 透過 Amazon Rekognition 偵測映像中的個人防護裝備

以下程式碼範例說明如何建置可使用 Amazon Rekognition 在映像中偵測個人防護裝備 (PPE) 的應用程式。

Java

適用於 Java 2.x 的 SDK

示範如何建立 AWS Lambda 函數，以偵測具有個人防護設備的映像。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- DynamoDB
- Amazon Rekognition
- Amazon S3
- Amazon SES

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 偵測從映像擷取的文字中的實體

下列程式碼範例示範如何使用 Amazon Comprehend 偵測 Amazon Textract 從存放在 Amazon S3 中的影像中提取的文字中的實體。

Python

適用於 Python 的 SDK (Boto3)

顯示如何使用 Jupyter 筆記本 適用於 Python (Boto3) 的 AWS SDK 中的 來偵測從影像擷取的文字中的實體。本範例使用 Amazon Textract 從儲存於 Amazon Simple Storage Service (Amazon S3) 和 Amazon Comprehend 中的影像提取文字，以偵測擷取文字中的實體。

此範例是 Jupyter 的筆記型電腦，必須在可以託管的筆記型電腦的環境中運行。如需使用 Amazon SageMaker AI 執行範例的指示，請參閱 [TextractAndComprehendNotebook.ipynb](#) 中的說明。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Comprehend
- Amazon S3
- Amazon Textract

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 偵測映像中的臉部

以下程式碼範例顯示做法：

- 在 Amazon S3 儲存貯體儲存映像。
- 使用 Amazon Rekognition 偵測面部細節，例如年齡範圍、性別和情感 (例如微笑)。
- 顯示這些詳細資訊。

Rust

適用於 Rust 的 SDK

將映像儲存在 Amazon S3 儲存貯體中，並包含上傳字首，使用 Amazon Rekognition 偵測面部細節，例如年齡範圍、性別和情感 (微笑等)，並顯示這些詳細資訊。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Rekognition
- Amazon S3

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 透過 Amazon Rekognition 偵測映像中的物件

下列程式碼範例說明如何建置可使用 Amazon Rekognition 按類別偵測映像中物件的應用程式。

.NET

適用於 .NET 的 SDK

說明如何使用 Amazon Rekognition .NET API 建立應用程式，該應用程式可使用 Amazon Rekognition 對 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的映像按類別識別物件。此應用程式可使用 Amazon Simple Email Service (Amazon SES) 向管理員傳送包含結果的電子郵件通知。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Rekognition
- Amazon S3
- Amazon SES

Java

適用於 Java 2.x 的 SDK

說明如何使用 Amazon Rekognition Java API 建立應用程式，該應用程式可使用 Amazon Rekognition 對 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的映像按類別識別物件。此應用程式可使用 Amazon Simple Email Service (Amazon SES) 向管理員傳送包含結果的電子郵件通知。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Rekognition
- Amazon S3
- Amazon SES

JavaScript

適用於 JavaScript (v3) 的 SDK

示範如何使用 Amazon Rekognition 搭配 適用於 JavaScript 的 AWS SDK 來建立使用 Amazon Rekognition 的應用程式，以在位於 Amazon Simple Storage Service (Amazon S3) 儲存貯體的影像中依類別識別物件。此應用程式可使用 Amazon Simple Email Service (Amazon SES) 向管理員傳送包含結果的電子郵件通知。

了解如何：

- 使用 Amazon Cognito 建立未經身分驗證的使用者。
- 使用 Amazon Rekognition 分析映像中的物件。
- 驗證 Amazon SES 的電子郵件地址。
- 使用 Amazon SES 傳送電子郵件通知。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Rekognition
- Amazon S3
- Amazon SES

Kotlin

適用於 Kotlin 的 SDK

展示如何使用 Amazon Rekognition Kotlin API 建立應用程式，該應用程式使用 Amazon Rekognition 對位於 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的映像按類別識別物件。此應用程式可使用 Amazon Simple Email Service (Amazon SES) 向管理員傳送包含結果的電子郵件通知。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Rekognition
- Amazon S3
- Amazon SES

Python

適用於 Python 的 SDK (Boto3)

說明如何使用 適用於 Python (Boto3) 的 AWS SDK 來建立可讓您執行下列動作的 Web 應用程式：

- 將相片上傳到 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
- 使用 Amazon Rekognition 分析和標籤照片。
- 使用 Amazon Simple Email Service (Amazon SES) 傳送映像分析的電子郵件報告。

此範例包含兩個主要組件：一個使用 React 內建 JavaScript 編寫的網頁，以及一個使用 Flask-RESTful 內建 Python 編寫的 REST 服務。

您可以使用 React 網頁執行以下操作：

- 顯示儲存於 S3 儲存貯體中的映像的清單。

- 將映像從您的電腦上傳至 S3 儲存貯體。
- 顯示識別映像中偵測到的專案的映像和標籤。
- 取得 S3 儲存貯體中所有映像的報告，並傳送報告的電子郵件。

該網頁呼叫 REST 服務。該服務將請求發送到 AWS 來執行下列動作：

- 取得並篩選 S3 儲存貯體中的映像的清單。
- 將相片上傳至 S3 儲存貯體。
- 使用 Amazon Rekognition 分析個別照片，並取得標識照片中偵測到的專案的標籤清單。
- 分析 S3 儲存貯體中的所有相片，然後使用 Amazon SES 傳送報告的電子郵件。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Rekognition
- Amazon S3
- Amazon SES

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 透過 Amazon Rekognition 偵測影片中的人物和物件

下列程式碼範例示範如何使用 Amazon Rekognition 偵測映像中的人物和物件。

Java

適用於 Java 2.x 的 SDK

示範如何使用 Amazon Rekognition Java API 來建立應用程式，以偵測位於 Amazon Simple Storage Service (Amazon S3) 儲存貯體的映像中的人臉和物件。此應用程式可使用 Amazon Simple Email Service (Amazon SES) 向管理員傳送包含結果的電子郵件通知。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Rekognition
- Amazon S3
- Amazon SES

- Amazon SNS
- Amazon SQS

Python

適用於 Python (Boto3) 的 SDK

使用 Amazon Rekognition 透過啟動非同步偵測任務來偵測映像中的人臉、物件和人物。此範例也會設定 Amazon Rekognition 以在任務完成時通知 Amazon Simple Notification Service (Amazon SNS) 主題，並訂閱 Amazon Simple Queue Service (Amazon SQS) 佇列到該主題。當佇列收到有關任務的訊息時，會擷取任務並輸出結果。

這個範例在 GitHub 上的檢視效果最佳。如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Rekognition
- Amazon S3
- Amazon SES
- Amazon SNS
- Amazon SQS

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

從 Amazon Simple Storage Service (Amazon S3) 儲存貯體下載 S3 'directories'

下列程式碼範例示範如何下載和篩選 Amazon S3 儲存貯體 'directories' 的內容。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

此範例示範如何使用 中的 [S3TransferManager](#) AWS SDK for Java 2.x ，從 Amazon S3 儲存貯體下載「目錄」。範例也會示範如何在請求中使用 [DownloadFilters](#)。

```
/**
 * For standard buckets, S3 provides the illusion of a directory structure
 * through the use of keys. When you upload
 *   * an object to an S3 bucket, you specify a key, which is essentially the
 * "path" to the object. The key can contain
 *   * forward slashes ("/") to make it appear as if the object is stored in a
 * directory structure, but this is just a
 *   * logical representation, not an actual directory.
 * <p><pre>
 * In this example, our S3 bucket contains the following objects:
 *
 * folder1/file1.txt
 * folder1/file2.txt
 * folder1/file3.txt
 * folder2/file1.txt
 * folder2/file2.txt
 * folder2/file3.txt
 * folder3/file1.txt
 * folder3/file2.txt
 * folder3/file3.txt
 *
 * When method `downloadS3Directories` is invoked with
 * `destinationPathURI` set to `/test`, the downloaded
 * directory looks like:
 *
 * |- test
 *   |- folder1
 *     |- file1.txt
 *     |- file2.txt
 *     |- file3.txt
 *   |- folder3
 *     |- file1.txt
 *     |- file2.txt
 *     |- file3.txt
 * </pre>
 *
 * @param transferManager An S3TransferManager instance.
 * @param destinationPathURI local directory to hold the downloaded S3
 * 'directories' and files.
```

```

    * @param bucketName      The S3 bucket that contains the 'directories' to
download.
    * @return The number of objects (files, in this case) that were downloaded.
    */
    public Integer downloadS3Directories(S3TransferManager transferManager,
                                         URI destinationPathURI, String
bucketName) {

        // Define the filters for which 'directories' we want to download.
        DownloadFilter folder1Filter = (S3Object s3Object) ->
s3Object.key().startsWith("folder1/");
        DownloadFilter folder3Filter = (S3Object s3Object) ->
s3Object.key().startsWith("folder3/");
        DownloadFilter folderFilter = s3Object ->
folder1Filter.or(folder3Filter).test(s3Object);

        DirectoryDownload directoryDownload =
transferManager.downloadDirectory(DownloadDirectoryRequest.builder()
                                .destination(Paths.get(destinationPathURI))
                                .bucket(bucketName)
                                .filter(folderFilter)
                                .build());
        CompletedDirectoryDownload completedDirectoryDownload =
directoryDownload.completionFuture().join();

        Integer numFilesInFolder1 =
Paths.get(destinationPathURI).resolve("folder1").toFile().list().length;
        Integer numFilesInFolder3 =
Paths.get(destinationPathURI).resolve("folder3").toFile().list().length;

        try {
            assert numFilesInFolder1 == 3;
            assert numFilesInFolder3 == 3;
            assert !
Paths.get(destinationPathURI).resolve("folder2").toFile().exists(); // `folder2`
was not downloaded.
        } catch (AssertionError e) {
            logger.error("An assertion failed.");
        }

        completedDirectoryDownload.failedTransfers()
            .forEach(fail -> logger.warn("Object failed to transfer [{})",
fail.exception().getMessage()));
        return numFilesInFolder1 + numFilesInFolder3;
    }

```

```
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

將 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的所有物件下載至本機目錄

下列程式碼範例示範如何將 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的所有物件下載至本機目錄。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用 [S3TransferManager](#) [下載](#) 相同 S3 儲存貯體中的所有 S3 物件。檢視 [完整檔案](#) 並 [測試](#)。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedDirectoryDownload;
import software.amazon.awssdk.transfer.s3.model.DirectoryDownload;
import software.amazon.awssdk.transfer.s3.model.DownloadDirectoryRequest;
import java.io.IOException;
import java.net.URI;
import java.net.URISyntaxException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.HashSet;
import java.util.Set;
import java.util.UUID;
```



```
import java.util.stream.Collectors;

public Integer downloadObjectsToDirectory(S3TransferManager transferManager,
    URI destinationPathURI, String bucketName) {
    DirectoryDownload directoryDownload =
transferManager.downloadDirectory(DownloadDirectoryRequest.builder()
    .destination(Paths.get(destinationPathURI))
    .bucket(bucketName)
    .build());
    CompletedDirectoryDownload completedDirectoryDownload =
directoryDownload.completionFuture().join();

    completedDirectoryDownload.failedTransfers()
        .forEach(fail -> logger.warn("Object [{}] failed to transfer",
fail.toString()));
    return completedDirectoryDownload.failedTransfers().size();
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DownloadDirectory](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 從 Amazon S3 物件下載未知大小的串流

下列程式碼範例示範如何將大小不明的串流從 Amazon S3 物件下載。

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import ArgumentParser
import AWSClientRuntime
import AWSS3
```

```
import Foundation
import Smithy
import SmithyHTTPAPI
import SmithyStreams

/// Download a file from the specified bucket.
///
/// - Parameters:
///   - bucket: The Amazon S3 bucket name to get the file from.
///   - key: The name (or path) of the file to download from the bucket.
///   - destPath: The pathname on the local filesystem at which to store
///     the downloaded file.
func downloadFile(bucket: String, key: String, destPath: String?) async
throws {
    let fileURL: URL

    // If no destination path was provided, use the key as the name to use
    // for the file in the downloads folder.

    if destPath == nil {
        do {
            try fileURL = FileManager.default.url(
                for: .downloadsDirectory,
                in: .userDomainMask,
                appropriateFor: URL(string: key),
                create: true
            ).appendingPathComponent(key)
        } catch {
            throw TransferError.directoryError
        }
    } else {
        fileURL = URL(fileURLWithPath: destPath!)
    }

    let config = try await S3Client.S3ClientConfiguration(region: region)
    let s3Client = S3Client(config: config)

    // Create a `FileHandle` referencing the local destination. Then
    // create a `ByteStream` from that.

    FileManager.default.createFile(atPath: fileURL.path, contents: nil,
    attributes: nil)
    let fileHandle = try FileHandle(forWritingTo: fileURL)
```

```
// Download the file using `GetObject`.

let getInput = GetObjectInput(
    bucket: bucket,
    key: key
)

do {
    let getOutput = try await s3Client.getObject(input: getInput)

    guard let body = getOutput.body else {
        throw TransferError.downloadError("Error: No data returned for
download")
    }

    // If the body is returned as a `Data` object, write that to the
    // file. If it's a stream, read the stream chunk by chunk,
    // appending each chunk to the destination file.

    switch body {
    case .data:
        guard let data = try await body.readData() else {
            throw TransferError.downloadError("Download error")
        }

        // Write the `Data` to the file.

        do {
            try data.write(to: fileURL)
        } catch {
            throw TransferError.writeError
        }
        break

    case .stream(let stream as ReadableStream):
        while (true) {
            let chunk = try await stream.readAsync(upToCount: 5 * 1024 *
1024)

            guard let chunk = chunk else {
                break
            }

            // Write the chunk to the destination file.
```

```
        do {
            try fileHandle.write(contentsOf: chunk)
        } catch {
            throw TransferError.writeError
        }
    }

    break
default:
    throw TransferError.downloadError("Received data is unknown
object type")
}
} catch {
    throw TransferError.downloadError("Error downloading the file:
\\(error)")
}

print("File downloaded to \\(fileURL.path).")
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 從多區域存取點取得 Amazon S3 物件 AWS

下列程式碼範例示範如何從多區域存取點取得物件。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

將 S3 用戶端設定為使用非對稱 Sigv4 (Sigv4a) 簽署演算法。

```
suspend fun createS3Client(): S3Client {
    // Configure your S3Client to use the Asymmetric SigV4 (SigV4a)
    signing algorithm.
    val sigV4aScheme = SigV4AsymmetricAuthScheme(DefaultAwsSigner)
    val s3 = S3Client.fromEnvironment {
        authSchemes = listOf(sigV4aScheme)
    }
    return s3
}
```

使用多區域存取點 ARN (而非儲存貯體名稱) 來擷取物件。

```
suspend fun getObjectFromMrap(
    s3: S3Client,
    mrapArn: String,
    keyName: String,
): String? {
    val request = GetObjectRequest {
        bucket = mrapArn // Use the ARN instead of the bucket name for object
        operations.
        key = keyName
    }

    var stringObj: String? = null
    s3.getObject(request) { resp ->
        stringObj = resp.body?.decodeToString()
        if (stringObj != null) {
            println("Successfully read $keyName from $mrapArn")
        }
    }
    return stringObj
}
```

- 如需詳細資訊，請參閱[適用於 Kotlin 的 AWS SDK 開發人員指南](#)。
- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [GetObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 從 Amazon S3 儲存貯體取得物件，指定 If-Modified-Since 標頭

下列程式碼範例示範如何從 S3 儲存貯體中物件讀取資料，但僅在自上次擷取時後該儲存貯體尚未修改時才能進行此讀取。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
use aws_sdk_s3::{
    error::SdkError,
    primitives::{ByteStream, DateTime, DateTimeFormat},
    Client,
};
use s3_code_examples::error::S3ExampleError;
use tracing::{error, warn};

const KEY: &str = "key";
const BODY: &str = "Hello, world!";

/// Demonstrate how `if-modified-since` reports that matching objects haven't
/// changed.
///
/// # Steps
/// - Create a bucket.
/// - Put an object in the bucket.
/// - Get the bucket headers.
/// - Get the bucket headers again but only if modified.
/// - Delete the bucket.
#[tokio::main]
async fn main() -> Result<(), S3ExampleError> {
    tracing_subscriber::fmt::init();

    // Get a new UUID to use when creating a unique bucket name.
    let uuid = uuid::Uuid::new_v4();
```

```
// Load the AWS configuration from the environment.
let client = Client::new(&aws_config::load_from_env().await);

// Generate a unique bucket name using the previously generated UUID.
// Then create a new bucket with that name.
let bucket_name = format!("if-modified-since-{{uuid}}");
client
    .create_bucket()
    .bucket(bucket_name.clone())
    .send()
    .await?;

// Create a new object in the bucket whose name is `KEY` and whose
// contents are `BODY`.
let put_object_output = client
    .put_object()
    .bucket(bucket_name.as_str())
    .key(KEY)
    .body(ByteStream::from_static(BODY.as_bytes()))
    .send()
    .await;

// If the `PutObject` succeeded, get the eTag string from it. Otherwise,
// report an error and return an empty string.
let e_tag_1 = match put_object_output {
    Ok(put_object) => put_object.e_tag.unwrap(),
    Err(err) => {
        error!("{err:?}");
        String::new()
    }
};

// Request the object's headers.
let head_object_output = client
    .head_object()
    .bucket(bucket_name.as_str())
    .key(KEY)
    .send()
    .await;

// If the `HeadObject` request succeeded, create a tuple containing the
// values of the headers `last-modified` and `etag`. If the request
// failed, return the error in a tuple instead.
let (last_modified, e_tag_2) = match head_object_output {
```

```

    Ok(head_object) => (
        Ok(head_object.last_modified().cloned().unwrap()),
        head_object.e_tag.unwrap(),
    ),
    Err(err) => (Err(err), String::new()),
};

warn!("last modified: {last_modified:?}");
assert_eq!(
    e_tag_1, e_tag_2,
    "PutObject and first GetObject had differing eTags"
);

println!("First value of last_modified: {last_modified:?}");
println!("First tag: {}\n", e_tag_1);

// Send a second `HeadObject` request. This time, the `if_modified_since`
// option is specified, giving the `last_modified` value returned by the
// first call to `HeadObject`.
//
// Since the object hasn't been changed, and there are no other objects in
// the bucket, there should be no matching objects.

let head_object_output = client
    .head_object()
    .bucket(bucket_name.as_str())
    .key(KEY)
    .if_modified_since(last_modified.unwrap())
    .send()
    .await;

// If the `HeadObject` request succeeded, the result is a tuple containing
// the `last_modified` and `e_tag_1` properties. This is not the expected
// result.
//
// The expected result of the second call to `HeadObject` is an
// `SdkError::ServiceError` containing the HTTP error response. If that's
// the case and the HTTP status is 304 (not modified), the output is a
// tuple containing the values of the HTTP `last-modified` and `etag`
// headers.
//
// If any other HTTP error occurred, the error is returned as an
// `SdkError::ServiceError`.

```



```

let (last_modified, e_tag_2) = match head_object_output {
  Ok(head_object) => (
    Ok(head_object.last_modified().cloned().unwrap()),
    head_object.e_tag.unwrap(),
  ),
  Err(err) => match err {
    SdkError::ServiceError(err) => {
      // Get the raw HTTP response. If its status is 304, the
      // object has not changed. This is the expected code path.
      let http = err.raw();
      match http.status().as_u16() {
        // If the HTTP status is 304: Not Modified, return a
        // tuple containing the values of the HTTP
        // `last-modified` and `etag` headers.
        304 => (
          Ok(DateTime::from_str(
            http.headers().get("last-modified").unwrap(),
            DateTimeFormat::HttpDate,
          )
            .unwrap()),
          http.headers().get("etag").map(|t| t.into()).unwrap(),
        ),
        // Any other HTTP status code is returned as an
        // `SdkError::ServiceError`.
        _ => (Err(SdkError::ServiceError(err)), String::new()),
      }
    }
    // Any other kind of error is returned in a tuple containing the
    // error and an empty string.
    _ => (Err(err), String::new()),
  },
};

warn!("last modified: {last_modified:?}");
assert_eq!(
  e_tag_1, e_tag_2,
  "PutObject and second HeadObject had different eTags"
);

println!("Second value of last modified: {last_modified:?}");
println!("Second tag: {}", e_tag_2);

// Clean up by deleting the object and the bucket.
client

```

```
        .delete_object()  
        .bucket(bucket_name.as_str())  
        .key(KEY)  
        .send()  
        .await?;  
  
    client  
        .delete_bucket()  
        .bucket(bucket_name.as_str())  
        .send()  
        .await?;  
  
    Ok(())  
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的 AWS SDK API 參考》中的 [GetObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 開始加密 Amazon S3 物件

下列程式碼範例示範如何開始使用 Amazon S3 物件的加密。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;  
using System.IO;  
using System.Security.Cryptography;  
using System.Threading.Tasks;  
using Amazon.S3;  
using Amazon.S3.Model;
```

```
/// <summary>
/// This example shows how to apply client encryption to an object in an
/// Amazon Simple Storage Service (Amazon S3) bucket.
/// </summary>
public class SSEClientEncryption
{
    public static async Task Main()
    {
        string bucketName = "amzn-s3-demo-bucket";
        string keyName = "exampleobject.txt";
        string copyTargetKeyName = "examplecopy.txt";

        // If the AWS Region defined for your default user is different
        // from the Region where your Amazon S3 bucket is located,
        // pass the Region name to the Amazon S3 client object's constructor.
        // For example: RegionEndpoint.USWest2.
        IAmazonS3 client = new AmazonS3Client();

        try
        {
            // Create an encryption key.
            Aes aesEncryption = Aes.Create();
            aesEncryption.KeySize = 256;
            aesEncryption.GenerateKey();
            string base64Key = Convert.ToBase64String(aesEncryption.Key);

            // Upload the object.
            PutObjectRequest putObjectRequest = await
UploadObjectAsync(client, bucketName, keyName, base64Key);

            // Download the object and verify that its contents match what
you uploaded.
            await DownloadObjectAsync(client, bucketName, keyName, base64Key,
putObjectRequest);

            // Get object metadata and verify that the object uses AES-256
encryption.
            await GetObjectMetadataAsync(client, bucketName, keyName,
base64Key);

            // Copy both the source and target objects using server-side
encryption with
            // an encryption key.
```

```

        await CopyObjectAsync(client, bucketName, keyName,
copyTargetKeyName, aesEncryption, base64Key);
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error: {ex.Message}");
    }
}

/// <summary>
/// Uploads an object to an Amazon S3 bucket.
/// </summary>
/// <param name="client">The initialized Amazon S3 client object used to
call
/// PutObjectAsync.</param>
/// <param name="bucketName">The name of the Amazon S3 bucket to which
the
/// object will be uploaded.</param>
/// <param name="keyName">The name of the object to upload to the Amazon
S3
/// bucket.</param>
/// <param name="base64Key">The encryption key.</param>
/// <returns>The PutObjectRequest object for use by
DownloadObjectAsync.</returns>
public static async Task<PutObjectRequest> UploadObjectAsync(
    IAmazonS3 client,
    string bucketName,
    string keyName,
    string base64Key)
{
    PutObjectRequest putObjectRequest = new PutObjectRequest
    {
        BucketName = bucketName,
        Key = keyName,
        ContentBody = "sample text",
        ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
        ServerSideEncryptionCustomerProvidedKey = base64Key,
    };
    PutObjectResponse putObjectResponse = await
client.PutObjectAsync(putObjectRequest);
    return putObjectRequest;
}

```

```

    /// <summary>
    /// Downloads an encrypted object from an Amazon S3 bucket.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// GetObjectAsync.</param>
    /// <param name="bucketName">The name of the Amazon S3 bucket where the
object
    /// is located.</param>
    /// <param name="keyName">The name of the Amazon S3 object to download.</
param>
    /// <param name="base64Key">The encryption key used to encrypt the
    /// object.</param>
    /// <param name="putObjectRequest">The PutObjectRequest used to upload
    /// the object.</param>
    public static async Task DownloadObjectAsync(
        IAmazonS3 client,
        string bucketName,
        string keyName,
        string base64Key,
        PutObjectRequest putObjectRequest)
    {
        GetObjectRequest getObjectRequest = new GetObjectRequest
        {
            BucketName = bucketName,
            Key = keyName,

            // Provide encryption information for the object stored in Amazon
S3.
            ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
            ServerSideEncryptionCustomerProvidedKey = base64Key,
        };

        using (GetObjectResponse getResponse = await
client.GetObjectAsync(getObjectRequest))
            using (StreamReader reader = new
StreamReader(getResponse.ResponseStream))
            {
                string content = reader.ReadToEnd();
                if (string.Compare(putObjectRequest.ContentBody, content) == 0)
                {
                    Console.WriteLine("Object content is same as we uploaded");
                }
            }
    }

```

```

        else
        {
            Console.WriteLine("Error...Object content is not same.");
        }

        if (getResponse.ServerSideEncryptionCustomerMethod ==
ServerSideEncryptionCustomerMethod.AES256)
        {
            Console.WriteLine("Object encryption method is AES256, same
as we set");
        }
        else
        {
            Console.WriteLine("Error...Object encryption method is not
the same as AES256 we set");
        }
    }
}

/// <summary>
/// Retrieves the metadata associated with an Amazon S3 object.
/// </summary>
/// <param name="client">The initialized Amazon S3 client object used
/// to call GetObjectMetadataAsync.</param>
/// <param name="bucketName">The name of the Amazon S3 bucket containing
the
/// object for which we want to retrieve metadata.</param>
/// <param name="keyName">The name of the object for which we wish to
/// retrieve the metadata.</param>
/// <param name="base64Key">The encryption key associated with the
/// object.</param>
public static async Task GetObjectMetadataAsync(
    IAmazonS3 client,
    string bucketName,
    string keyName,
    string base64Key)
{
    GetObjectMetadataRequest getObjectMetadataRequest = new
GetObjectMetadataRequest
    {
        BucketName = bucketName,
        Key = keyName,

```

```

        // The object stored in Amazon S3 is encrypted, so provide the
        necessary encryption information.
        ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
        ServerSideEncryptionCustomerProvidedKey = base64Key,
    };

    GetObjectMetadataResponse getObjectMetadataResponse = await
client.GetObjectMetadataAsync(getObjectMetadataRequest);
    Console.WriteLine("The object metadata show encryption method used
is: {0}", getObjectMetadataResponse.ServerSideEncryptionCustomerMethod);
}

/// <summary>
/// Copies an encrypted object from one Amazon S3 bucket to another.
/// </summary>
/// <param name="client">The initialized Amazon S3 client object used to
call
/// CopyObjectAsync.</param>
/// <param name="bucketName">The Amazon S3 bucket containing the object
/// to copy.</param>
/// <param name="keyName">The name of the object to copy.</param>
/// <param name="copyTargetKeyName">The Amazon S3 bucket to which the
object
/// will be copied.</param>
/// <param name="aesEncryption">The encryption type to use.</param>
/// <param name="base64Key">The encryption key to use.</param>
public static async Task CopyObjectAsync(
    IAmazonS3 client,
    string bucketName,
    string keyName,
    string copyTargetKeyName,
    Aes aesEncryption,
    string base64Key)
{
    aesEncryption.GenerateKey();
    string copyBase64Key = Convert.ToBase64String(aesEncryption.Key);

    CopyObjectRequest copyRequest = new CopyObjectRequest
    {
        SourceBucket = bucketName,
        SourceKey = keyName,
        DestinationBucket = bucketName,
        DestinationKey = copyTargetKeyName,
    }

```

```
        // Information about the source object's encryption.
        CopySourceServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
        CopySourceServerSideEncryptionCustomerProvidedKey = base64Key,

        // Information about the target object's encryption.
        ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
        ServerSideEncryptionCustomerProvidedKey = copyBase64Key,
    };
    await client.CopyObjectAsync(copyRequest);
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的下列主題。
- [CopyObject](#)
- [GetObject](#)
- [GetObjectMetadata](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 開始使用 Amazon S3 物件的標籤

下列程式碼範例示範如何開始使用 Amazon S3 物件的索引標籤。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
```



```
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to work with tags in Amazon Simple Storage
/// Service (Amazon S3) objects.
/// </summary>
public class ObjectTag
{
    public static async Task Main()
    {
        string bucketName = "amzn-s3-demo-bucket";
        string keyName = "newobject.txt";
        string filePath = @"*** file path ***";

        // Specify your bucket region (an example region is shown).
        RegionEndpoint bucketRegion = RegionEndpoint.USSouth1;

        var client = new AmazonS3Client(bucketRegion);
        await PutObjectsWithTagsAsync(client, bucketName, keyName, filePath);
    }

    /// <summary>
    /// This method uploads an object with tags. It then shows the tag
    /// values, changes the tags, and shows the new tags.
    /// </summary>
    /// <param name="client">The Initialized Amazon S3 client object used
    /// to call the methods to create and change an objects tags.</param>
    /// <param name="bucketName">A string representing the name of the
    /// bucket where the object will be stored.</param>
    /// <param name="keyName">A string representing the key name of the
    /// object to be tagged.</param>
    /// <param name="filePath">The directory location and file name of the
    /// object to be uploaded to the Amazon S3 bucket.</param>
    public static async Task PutObjectsWithTagsAsync(IAmazonS3 client, string
bucketName, string keyName, string filePath)
    {
        try
        {
            // Create an object with tags.
            var putRequest = new PutObjectRequest
```

```
        {
            BucketName = bucketName,
            Key = keyName,
            FilePath = filePath,
            TagSet = new List<Tag>
            {
                new Tag { Key = "Keyx1", Value = "Value1" },
                new Tag { Key = "Keyx2", Value = "Value2" },
            },
        };

        PutObjectResponse response = await
client.PutObjectAsync(putRequest);

        // Now retrieve the new object's tags.
        GetObjectTaggingRequest getTagsRequest = new
GetObjectTaggingRequest()
        {
            BucketName = bucketName,
            Key = keyName,
        };

        GetObjectTaggingResponse objectTags = await
client.GetObjectTaggingAsync(getTagsRequest);

        // Display the tag values.
        objectTags.Tagging
            .ForEach(t => Console.WriteLine($"Key: {t.Key}, Value:
{t.Value}"));

        Tagging newTagSet = new Tagging()
        {
            TagSet = new List<Tag>
            {
                new Tag { Key = "Key3", Value = "Value3" },
                new Tag { Key = "Key4", Value = "Value4" },
            },
        };

        PutObjectTaggingRequest putObjTagsRequest = new
PutObjectTaggingRequest()
        {
            BucketName = bucketName,
            Key = keyName,
```

```
        Tagging = newTagSet,
    };

    PutObjectTaggingResponse response2 = await
client.PutObjectTaggingAsync(putObjTagsRequest);

    // Retrieve the tags again and show the values.
    GetObjectTaggingRequest getTagsRequest2 = new
GetObjectTaggingRequest()
    {
        BucketName = bucketName,
        Key = keyName,
    };
    GetObjectTaggingResponse objectTags2 = await
client.GetObjectTaggingAsync(getTagsRequest2);

    objectTags2.Tagging
        .ForEach(t => Console.WriteLine($"Key: {t.Key}, Value:
{t.Value}"));
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine(
            $"Error: '{ex.Message}'");
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetObjectTagging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

大數據處理叢集入門

以下程式碼範例顯示做法：

- 建立 EC2 金鑰對
- 設定儲存並準備您的應用程式

- 清除資源

Bash

AWS CLI 搭配 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[範例開發人員教學課程](#)儲存庫中設定和執行。

```
#!/bin/bash

# EMR Getting Started Tutorial Script
# This script automates the steps in the Amazon EMR Getting Started tutorial

set -euo pipefail

# Security: Set strict mode and trap errors
trap 'handle_error "Script interrupted or command failed"' ERR

# Set up logging with secure permissions
LOG_FILE="emr-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon EMR Getting Started Tutorial Script"
echo "Logging to $LOG_FILE"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created so far:"
    if [ -n "${BUCKET_NAME:-}" ]; then echo "- S3 Bucket: $BUCKET_NAME"; fi
    if [ -n "${CLUSTER_ID:-}" ]; then echo "- EMR Cluster: $CLUSTER_ID"; fi

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}
```

```
# Function to clean up resources
cleanup() {
    echo ""
    echo "=====
    echo "CLEANUP IN PROGRESS"
    echo "=====
    echo "Starting cleanup process..."

    # Terminate EMR cluster if it exists
    if [ -n "${CLUSTER_ID:-}" ]; then
        echo "Terminating EMR cluster: $CLUSTER_ID"
        aws emr terminate-clusters --cluster-ids "$CLUSTER_ID" 2>/dev/null ||
true

        echo "Waiting for cluster to terminate..."
        aws emr wait cluster-terminated --cluster-id "$CLUSTER_ID" 2>/dev/null ||
true
        echo "Cluster terminated successfully."
    fi

    # Delete S3 bucket and contents if it exists and is not shared
    if [ -n "${BUCKET_NAME:-}" ] && [ "${BUCKET_IS_SHARED:-false}" != "true" ];
then
        echo "Deleting S3 bucket contents: $BUCKET_NAME"
        aws s3 rm "s3://$BUCKET_NAME" --recursive 2>/dev/null || true

        echo "Deleting S3 bucket: $BUCKET_NAME"
        aws s3 rb "s3://$BUCKET_NAME" 2>/dev/null || true
    fi

    # Remove temporary key pair file if created by this script
    if [ -f "${KEY_NAME_FILE:-}" ]; then
        rm -f "$KEY_NAME_FILE"
        echo "Removed temporary key pair file."
    fi

    echo "Cleanup completed."
}

# Validate AWS CLI is installed and configured
if ! command -v aws &> /dev/null; then
    handle_error "AWS CLI is not installed"
fi
```

```
# Test AWS credentials
if ! aws sts get-caller-identity > /dev/null 2>&1; then
    handle_error "AWS credentials are not configured or invalid"
fi

# Generate a random identifier for S3 bucket
RANDOM_ID=$(openssl rand -hex 6)

# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
text 2>/dev/null || true)

if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    BUCKET_NAME="emr-${RANDOM_ID}"
fi
echo "Using bucket name: $BUCKET_NAME"

# Create S3 bucket with security best practices
echo "Creating S3 bucket: $BUCKET_NAME"
aws s3 mb "s3://$BUCKET_NAME" --region "${AWS_REGION:-us-east-1}" || handle_error
"Failed to create S3 bucket"

# Tag the bucket
aws s3api put-bucket-tagging --bucket "$BUCKET_NAME" \
    --tagging 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=emr-
gs}]'

# Enable bucket versioning for safety
aws s3api put-bucket-versioning --bucket "$BUCKET_NAME" --versioning-
configuration Status=Enabled || true

# Block public access to bucket
aws s3api put-public-access-block --bucket "$BUCKET_NAME" \
    --public-access-block-configuration \
```

```

    "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
    || true

# Enable encryption on bucket
aws s3api put-bucket-encryption --bucket "$BUCKET_NAME" \
    --server-side-encryption-configuration '{
        "Rules": [{
            "ApplyServerSideEncryptionByDefault": {
                "SSEAlgorithm": "AES256"
            }
        }]
    }' || true

echo "S3 bucket created successfully with security best practices."

# Create PySpark script
echo "Creating PySpark script: health_violations.py"
cat > health_violations.py << 'EOL'
import argparse

from pyspark.sql import SparkSession

def calculate_red_violations(data_source, output_uri):
    """
    Processes sample food establishment inspection data and queries the data to
    find the top 10 establishments
    with the most Red violations from 2006 to 2020.

    :param data_source: The URI of your food establishment data CSV, such as
    's3://emr-tutorial-bucket/food-establishment-data.csv'.
    :param output_uri: The URI where output is written, such as 's3://emr-
    tutorial-bucket/restaurant_violation_results'.
    """
    with SparkSession.builder.appName("Calculate Red Health
    Violations").getOrCreate() as spark:
        # Load the restaurant violation CSV data
        if data_source is not None:
            restaurants_df = spark.read.option("header", "true").csv(data_source)

        # Create an in-memory DataFrame to query
        restaurants_df.createOrReplaceTempView("restaurant_violations")

```

```

        # Create a DataFrame of the top 10 restaurants with the most Red
        violations
        top_red_violation_restaurants = spark.sql("""SELECT name, count(*) AS
total_red_violations
        FROM restaurant_violations
        WHERE violation_type = 'RED'
        GROUP BY name
        ORDER BY total_red_violations DESC LIMIT 10""")

        # Write the results to the specified output URI
        top_red_violation_restaurants.write.option("header",
"true").mode("overwrite").csv(output_uri)

if __name__ == "__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument(
        '--data_source', help="The URI for you CSV restaurant data, like an S3
bucket location.")
    parser.add_argument(
        '--output_uri', help="The URI where output is saved, like an S3 bucket
location.")
    args = parser.parse_args()

    calculate_red_violations(args.data_source, args.output_uri)
EOL

# Secure the script file
chmod 600 health_violations.py

# Upload PySpark script to S3
echo "Uploading PySpark script to S3"
aws s3 cp health_violations.py "s3://$BUCKET_NAME/" --sse AES256 || handle_error
"Failed to upload PySpark script"
echo "PySpark script uploaded successfully."

# Download and prepare sample data
echo "Downloading sample data"
curl -sS -o food_establishment_data.zip "https://docs.aws.amazon.com/emr/latest/
ManagementGuide/samples/food_establishment_data.zip" || handle_error "Failed to
download sample data"

# Verify downloaded file
if [ ! -f food_establishment_data.zip ] || [ ! -s food_establishment_data.zip ];
then

```



```
    handle_error "Downloaded file is empty or missing"
fi

unzip -o food_establishment_data.zip || handle_error "Failed to unzip sample
data"
echo "Sample data downloaded and extracted successfully."

# Secure the sample data file
chmod 600 food_establishment_data.csv

# Upload sample data to S3
echo "Uploading sample data to S3"
aws s3 cp food_establishment_data.csv "s3://$BUCKET_NAME/" --sse AES256 ||
    handle_error "Failed to upload sample data"
echo "Sample data uploaded successfully."

# Clean up sensitive local files
rm -f food_establishment_data.zip health_violations.py

# Create IAM default roles for EMR
echo "Creating IAM default roles for EMR"
aws emr create-default-roles 2>/dev/null || true
echo "IAM default roles created successfully."

# Check if EC2 key pair exists
echo "Checking for EC2 key pair"
KEY_PAIRS=$(aws ec2 describe-key-pairs --query "KeyPairs[*].KeyName" --output
text 2>/dev/null || true)

if [ -z "$KEY_PAIRS" ]; then
    echo "No EC2 key pairs found. Creating a new key pair..."
    KEY_NAME="emr-tutorial-key-${RANDOM_ID}"
    KEY_NAME_FILE="${KEY_NAME}.pem"
    aws ec2 create-key-pair --key-name "$KEY_NAME" \
        --tag-specifications 'ResourceType=key-pair,Tags=[{Key=project,Value=doc-
smith},{Key=tutorial,Value=emr-gs}]' \
        --query "KeyMaterial" --output text > "$KEY_NAME_FILE"
    chmod 400 "$KEY_NAME_FILE"
    echo "Created new key pair: $KEY_NAME"
else
    # Use the first available key pair
    KEY_NAME=$(echo "$KEY_PAIRS" | awk '{print $1}')
    echo "Using existing key pair: $KEY_NAME"
fi
```

```
# Launch EMR cluster with security best practices
echo "Launching EMR cluster with Spark"
CLUSTER_RESPONSE=$(aws emr create-cluster \
  --name "EMR Tutorial Cluster" \
  --release-label emr-6.10.0 \
  --applications Name=Spark \
  --ec2-attributes KeyName="$KEY_NAME" \
  --instance-type m5.xlarge \
  --instance-count 3 \
  --use-default-roles \
  --log-uri "s3://$BUCKET_NAME/logs/" \
  --ebs-root-volume-size 100 \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=emr-gs \
  --security-configuration "EMR-Tutorial-SecurityConfig" 2>/dev/null || true)

# Check for errors in the response
if echo "$CLUSTER_RESPONSE" | grep -i "error" > /dev/null; then
  handle_error "Failed to create EMR cluster: $CLUSTER_RESPONSE"
fi

# Extract cluster ID using jq if available, otherwise use alternative parsing
if command -v jq &> /dev/null; then
  CLUSTER_ID=$(echo "$CLUSTER_RESPONSE" | jq -r '.ClusterId // empty')
else
  CLUSTER_ID=$(echo "$CLUSTER_RESPONSE" | grep -o '"ClusterId"[[:space:]]*:' \
  [[:space:]]*"^[^"]*" | grep -o 'j-[A-Z0-9]*' || true)
fi

if [ -z "$CLUSTER_ID" ] || [ "$CLUSTER_ID" == "null" ]; then
  handle_error "Failed to extract cluster ID from response: $CLUSTER_RESPONSE"
fi

echo "EMR cluster created with ID: $CLUSTER_ID"

# Wait for cluster to be ready
echo "Waiting for cluster to be ready (this may take several minutes)..."
aws emr wait cluster-running --cluster-id "$CLUSTER_ID" || handle_error "Cluster
failed to reach running state"

# Check if cluster is in WAITING state
CLUSTER_STATE=$(aws emr describe-cluster --cluster-id "$CLUSTER_ID" --query
  "Cluster.Status.State" --output text)
if [ "$CLUSTER_STATE" != "WAITING" ]; then
```

```

    echo "Waiting for cluster to reach WAITING state..."
    WAIT_COUNT=0
    MAX_WAIT=120
    while [ "$CLUSTER_STATE" != "WAITING" ]; do
        if [ $WAIT_COUNT -ge $MAX_WAIT ]; then
            handle_error "Cluster did not reach WAITING state within timeout
period"
        fi
        sleep 30
        CLUSTER_STATE=$(aws emr describe-cluster --cluster-id "$CLUSTER_ID" --
query "Cluster.Status.State" --output text)
        echo "Current cluster state: $CLUSTER_STATE"

        # Check for error states
        if [[ "$CLUSTER_STATE" == "TERMINATED_WITH_ERRORS" || "$CLUSTER_STATE" ==
"TERMINATED" ]]; then
            handle_error "Cluster entered error state: $CLUSTER_STATE"
        fi
        WAIT_COUNT=$((WAIT_COUNT + 1))
    done
fi

echo "Cluster is now in WAITING state and ready to accept work."

# Submit Spark application as a step
echo "Submitting Spark application as a step"
STEP_RESPONSE=$(aws emr add-steps \
    --cluster-id "$CLUSTER_ID" \
    --steps Type=Spark,Name="Health Violations
Analysis",ActionOnFailure=CONTINUE,Args=["s3://$BUCKET_NAME/
health_violations.py","--data_source","s3://$BUCKET_NAME/
food_establishment_data.csv","--output_uri","s3://$BUCKET_NAME/results/"])

# Check for errors in the response
if echo "$STEP_RESPONSE" | grep -i "error" > /dev/null; then
    handle_error "Failed to submit step: $STEP_RESPONSE"
fi

# Extract step ID using appropriate method
if command -v jq >> /dev/null; then
    STEP_ID=$(echo "$STEP_RESPONSE" | jq -r '.StepIds[0] // empty')
else
    STEP_ID=$(echo "$STEP_RESPONSE" | grep -o 's-[A-Z0-9]*' | head -1 || true)
fi

```

```
if [ -z "$STEP_ID" ] || [ "$STEP_ID" == "null" ]; then
    echo "Full step response: $STEP_RESPONSE"
    handle_error "Failed to extract valid step ID from response"
fi

echo "Step submitted with ID: $STEP_ID"

# Wait for step to complete with timeout
echo "Waiting for step to complete (this may take several minutes)..."
aws emr wait step-complete --cluster-id "$CLUSTER_ID" --step-id "$STEP_ID" ||
    handle_error "Step failed to complete"

# Check step status
STEP_STATE=$(aws emr describe-step --cluster-id "$CLUSTER_ID" --step-id
"$STEP_ID" --query "Step.Status.State" --output text)
if [ "$STEP_STATE" != "COMPLETED" ]; then
    handle_error "Step did not complete successfully. Final state: $STEP_STATE"
fi

echo "Step completed successfully."

# View results
echo "Listing output files in S3"
aws s3 ls "s3://$BUCKET_NAME/results/" || handle_error "Failed to list output
files"

# Download results
echo "Downloading results file"
RESULT_FILE=$(aws s3 ls "s3://$BUCKET_NAME/results/" | grep -o "part-[0-9]*\.csv"
| head -1 || true)
if [ -z "$RESULT_FILE" ]; then
    echo "No result file found with pattern 'part-[0-9]*.csv'. Trying to find any
CSV file..."
    RESULT_FILE=$(aws s3 ls "s3://$BUCKET_NAME/results/" | grep -o "part-.*\.csv"
| head -1 || true)
    if [ -z "$RESULT_FILE" ]; then
        echo "Listing all files in results directory:"
        aws s3 ls "s3://$BUCKET_NAME/results/"
        handle_error "No result file found in the output directory"
    fi
fi
fi
```

```
aws s3 cp "s3://$BUCKET_NAME/results/$RESULT_FILE" ./results.csv --sse AES256 ||
  handle_error "Failed to download results file"
chmod 600 ./results.csv

echo "Results downloaded to results.csv"
echo "Top 10 establishments with the most red violations:"
cat results.csv

# Display SSH connection information
echo ""
echo "To connect to the cluster via SSH, use the following command:"
echo "aws emr ssh --cluster-id $CLUSTER_ID --key-pair-file ${KEY_NAME_FILE:-./}
${KEY_NAME}.pem}"

# Display summary of created resources
echo ""
echo "=====
echo "RESOURCES CREATED"
echo "=====
echo "- S3 Bucket: $BUCKET_NAME"
echo "- EMR Cluster: $CLUSTER_ID"
echo "- Results file: results.csv"
if [ -f "${KEY_NAME_FILE:-}" ]; then
    echo "- EC2 Key Pair: $KEY_NAME (saved to ${KEY_NAME_FILE})"
fi

# Perform cleanup
cleanup

echo "Script completed successfully."
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
 - [AddSteps](#)
 - [Cp](#)
 - [CreateCluster](#)
 - [CreateDefaultRoles](#)
 - [CreateKeyPair](#)
 - [DescribeCluster](#)
 - [DescribeKeyPairs](#)
 - [DescribeStep](#)

- [Ls](#)
- [Mb](#)
- [Rb](#)
- [Rm](#)
- [Ssh](#)
- [TerminateClusters](#)
- [等候](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

組態管理入門

以下程式碼範例顯示做法：

- 建立 Amazon S3 儲存貯體
- 建立 Amazon SNS 主題
- 為 Config 建立 IAM 角色
- 設定 Config 組態記錄器
- 設定 Config 交付管道
- 啟動組態記錄器
- 驗證 Config 設定

Bash

AWS CLI 搭配 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[範例開發人員教學課程](#)儲存庫中設定和執行。

```
#!/bin/bash
```

```
# AWS Config Setup Script (v2)
# This script sets up AWS Config with the AWS CLI

# Error handling
set -e
LOGFILE="aws-config-setup-v2.log"
touch $LOGFILE
exec > >(tee -a $LOGFILE)
exec 2>&1

# Function to handle errors
handle_error() {
    echo "ERROR: An error occurred at line $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set trap for error handling
trap 'handle_error $LINENO' ERR

# Function to generate random identifier
generate_random_id() {
    echo $(openssl rand -hex 6)
}

# Function to check if command was successful
check_command() {
    if echo "$1" | grep -i "error" > /dev/null; then
        echo "ERROR: $1"
        return 1
    fi
    return 0
}

# Function to clean up resources
cleanup_resources() {
    if [ -n "$CONFIG_RECORDER_NAME" ]; then
        echo "Stopping configuration recorder..."
        aws configservice stop-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    # Check if we created a new delivery channel before trying to delete it
```

```
if [ -n "$DELIVERY_CHANNEL_NAME" ] && [ "$CREATED_NEW_DELIVERY_CHANNEL" =
"true" ]; then
    echo "Deleting delivery channel..."
    aws configservice delete-delivery-channel --delivery-channel-name
"$DELIVERY_CHANNEL_NAME" 2>/dev/null || true
fi

if [ -n "$CONFIG_RECORDER_NAME" ] && [ "$CREATED_NEW_CONFIG_RECORDER" =
"true" ]; then
    echo "Deleting configuration recorder..."
    aws configservice delete-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
fi

if [ -n "$ROLE_NAME" ]; then
    if [ -n "$POLICY_NAME" ]; then
        echo "Detaching custom policy from role..."
        aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
"$POLICY_NAME" 2>/dev/null || true
    fi

    if [ -n "$MANAGED_POLICY_ARN" ]; then
        echo "Detaching managed policy from role..."
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
"$MANAGED_POLICY_ARN" 2>/dev/null || true
    fi

    echo "Deleting IAM role..."
    aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
fi

if [ -n "$SNS_TOPIC_ARN" ]; then
    echo "Deleting SNS topic..."
    aws sns delete-topic --topic-arn "$SNS_TOPIC_ARN" 2>/dev/null || true
fi

if [ -n "$S3_BUCKET_NAME" ]; then
    echo "Emptying S3 bucket..."
    aws s3 rm "s3://$S3_BUCKET_NAME" --recursive 2>/dev/null || true

    echo "Deleting S3 bucket..."
    if [ "$BUCKET_IS_SHARED" = "false" ]; then
        aws s3api delete-bucket --bucket "$S3_BUCKET_NAME" 2>/dev/null ||
true
```



```

        fi
    fi
}

# Function to display created resources
display_resources() {
    echo ""
    echo "=====
    echo "CREATED RESOURCES"
    echo "=====
    echo "S3 Bucket: $S3_BUCKET_NAME"
    echo "SNS Topic ARN: $SNS_TOPIC_ARN"
    echo "IAM Role: $ROLE_NAME"
    if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (newly created)"
    else
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (existing)"
    fi
    if [ "$CREATED_NEW_DELIVERY_CHANNEL" = "true" ]; then
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (newly created)"
    else
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (existing)"
    fi
    echo "=====
}

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
if [ -z "$ACCOUNT_ID" ]; then
    echo "ERROR: Failed to get AWS account ID"
    exit 1
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Generate random identifier for resources
RANDOM_ID=$(generate_random_id)
echo "Generated random identifier: $RANDOM_ID"

# Step 1: Create an S3 bucket
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \

```

```

    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
    text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="configservice-${RANDOM_ID}"
    echo "Creating S3 bucket: $S3_BUCKET_NAME"
fi

# Get the current region
AWS_REGION=$(aws configure get region)
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1" # Default to us-east-1 if no region is configured
fi
echo "Using AWS Region: $AWS_REGION"

# Create bucket with appropriate command based on region
if [ "$BUCKET_IS_SHARED" = "false" ]; then
    if [ "$AWS_REGION" = "us-east-1" ]; then
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME")
    else
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" --
create-bucket-configuration LocationConstraint="$AWS_REGION")
    fi
    check_command "$BUCKET_RESULT"
    echo "S3 bucket created: $S3_BUCKET_NAME"

    aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-config-gs}]'
    echo "Tags applied to S3 bucket"
else
    echo "Using shared bucket: $S3_BUCKET_NAME (skipping creation)"
fi

# Block public access for the bucket
aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
"BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=true"
echo "Public access blocked for bucket"

```

```
# Step 2: Create an SNS topic
TOPIC_NAME="config-topic-`${RANDOM_ID}`"
echo "Creating SNS topic: $TOPIC_NAME"
SNS_RESULT=$(aws sns create-topic --name "$TOPIC_NAME" --tags
  Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs)
check_command "$SNS_RESULT"
SNS_TOPIC_ARN=$(echo "$SNS_RESULT" | grep -o 'arn:aws:sns:[^']*')
echo "SNS topic created: $SNS_TOPIC_ARN"

# Step 3: Create an IAM role for AWS Config
ROLE_NAME="config-role-`${RANDOM_ID}`"
POLICY_NAME="config-delivery-permissions"
MANAGED_POLICY_ARN="arn:aws:iam::aws:policy/service-role/AWS_ConfigRole"

echo "Creating trust policy document..."
cat > config-trust-policy.json << EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "config.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://config-trust-policy.json)
check_command "$ROLE_RESULT"
ROLE_ARN=$(echo "$ROLE_RESULT" | grep -o 'arn:aws:iam::[^']*' | head -1)
echo "IAM role created: $ROLE_ARN"

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
  Key=tutorial,Value=aws-config-gs
echo "Tags applied to IAM role"

echo "Attaching AWS managed policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy --role-name "$ROLE_NAME" --policy-arn
"$MANAGED_POLICY_ARN")
```

```
check_command "$ATTACH_RESULT"
echo "AWS managed policy attached"

echo "Creating custom policy document for S3 and SNS access..."
cat > config-delivery-permissions.json << EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}/AWSLogs/${ACCOUNT_ID}/*",
      "Condition": {
        "StringLike": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": "${SNS_TOPIC_ARN}"
    }
  ]
}
EOF

echo "Attaching custom policy to role..."
POLICY_RESULT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
"$POLICY_NAME" --policy-document file://config-delivery-permissions.json)
check_command "$POLICY_RESULT"
echo "Custom policy attached"
```

```
# Wait for IAM role to propagate
echo "Waiting for IAM role to propagate (15 seconds)..."
sleep 15

# Step 4: Check if configuration recorder already exists
CONFIG_RECORDER_NAME="default"
CREATED_NEW_CONFIG_RECORDER="false"

echo "Checking for existing configuration recorder..."
EXISTING_RECORDERS=$(aws configservice describe-configuration-records 2>/dev/
null || echo "")
if echo "$EXISTING_RECORDERS" | grep -q "name"; then
    echo "Configuration recorder already exists. Will update it."
    # Get the name of the existing recorder
    CONFIG_RECORDER_NAME=$(echo "$EXISTING_RECORDERS" | grep -o '"name": "[^"]*"'
| head -1 | cut -d'"' -f4)
    echo "Using existing configuration recorder: $CONFIG_RECORDER_NAME"
else
    echo "No existing configuration recorder found. Will create a new one."
    CREATED_NEW_CONFIG_RECORDER="true"
fi

echo "Creating configuration recorder configuration..."
cat > configurationRecorder.json << EOF
{
    "name": "${CONFIG_RECORDER_NAME}",
    "roleARN": "${ROLE_ARN}",
    "recordingMode": {
        "recordingFrequency": "CONTINUOUS"
    }
}
EOF

echo "Creating recording group configuration..."
cat > recordingGroup.json << EOF
{
    "allSupported": true,
    "includeGlobalResourceTypes": true
}
EOF

echo "Setting up configuration recorder..."
```

```

RECORDER_RESULT=$(aws configservice put-configuration-recorder --configuration-
recorder file://configurationRecorder.json --recording-group file://
recordingGroup.json)
check_command "$RECORDER_RESULT"
echo "Configuration recorder set up"

if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:config-recorder/${CONFIG_RECORDER_NAME}" --tags
    Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to configuration recorder"
fi

# Step 5: Check if delivery channel already exists
DELIVERY_CHANNEL_NAME="default"
CREATED_NEW_DELIVERY_CHANNEL="false"

echo "Checking for existing delivery channel..."
EXISTING_CHANNELS=$(aws configservice describe-delivery-channels 2>/dev/null ||
echo "")
if echo "$EXISTING_CHANNELS" | grep -q "name"; then
    echo "Delivery channel already exists."
    # Get the name of the existing channel
    DELIVERY_CHANNEL_NAME=$(echo "$EXISTING_CHANNELS" | grep -o '"name": "[^"]*"'
| head -1 | cut -d'"' -f4)
    echo "Using existing delivery channel: $DELIVERY_CHANNEL_NAME"

    # Update the existing delivery channel
    echo "Creating delivery channel configuration for update..."
    cat > deliveryChannel.json << EOF
{
    "name": "${DELIVERY_CHANNEL_NAME}",
    "s3BucketName": "${S3_BUCKET_NAME}",
    "snsTopicARN": "${SNS_TOPIC_ARN}",
    "configSnapshotDeliveryProperties": {
        "deliveryFrequency": "Six_Hours"
    }
}
EOF

    echo "Updating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"

```

```
    echo "Delivery channel updated"
else
    echo "No existing delivery channel found. Will create a new one."
    CREATED_NEW_DELIVERY_CHANNEL="true"

    echo "Creating delivery channel configuration..."
    cat > deliveryChannel.json << EOF
{
    "name": "${DELIVERY_CHANNEL_NAME}",
    "s3BucketName": "${S3_BUCKET_NAME}",
    "snsTopicARN": "${SNS_TOPIC_ARN}",
    "configSnapshotDeliveryProperties": {
        "deliveryFrequency": "Six_Hours"
    }
}
EOF

    echo "Creating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel created"

    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:delivery-channel/${DELIVERY_CHANNEL_NAME}" --tags
    Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to delivery channel"
fi

# Step 6: Start the configuration recorder
echo "Checking configuration recorder status..."
RECORDER_STATUS=$(aws configservice describe-configuration-recorder-status 2>/
dev/null || echo "")
if echo "$RECORDER_STATUS" | grep -q '"recording": true'; then
    echo "Configuration recorder is already running."
else
    echo "Starting configuration recorder..."
    START_RESULT=$(aws configservice start-configuration-recorder --
configuration-recorder-name "$CONFIG_RECORDER_NAME")
    check_command "$START_RESULT"
    echo "Configuration recorder started"
fi

# Step 7: Verify the AWS Config setup
```

```
echo "Verifying delivery channel..."
VERIFY_CHANNEL=$(aws configservice describe-delivery-channels)
check_command "$VERIFY_CHANNEL"
echo "$VERIFY_CHANNEL"

echo "Verifying configuration recorder..."
VERIFY_RECORDER=$(aws configservice describe-configuration-recorders)
check_command "$VERIFY_RECORDER"
echo "$VERIFY_RECORDER"

echo "Verifying configuration recorder status..."
VERIFY_STATUS=$(aws configservice describe-configuration-recorder-status)
check_command "$VERIFY_STATUS"
echo "$VERIFY_STATUS"

# Display created resources
display_resources

# Ask if user wants to clean up resources
echo ""
echo "======"
echo "CLEANUP CONFIRMATION"
echo "======"
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE='y'

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..."
    cleanup_resources
    echo "Cleanup completed."
else
    echo "Resources will not be cleaned up. You can manually clean them up later."
fi

echo "Script completed successfully!"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
 - [AttachRolePolicy](#)
 - [CreateBucket](#)
 - [CreateRole](#)
 - [CreateTopic](#)

- [DeleteBucket](#)
- [DeleteConfigurationRecorder](#)
- [DeleteDeliveryChannel](#)
- [DeleteRole](#)
- [DeleteRolePolicy](#)
- [DeleteTopic](#)
- [DescribeConfigurationRecorderStatus](#)
- [DescribeConfigurationRecorders](#)
- [DescribeDeliveryChannels](#)
- [DetachRolePolicy](#)
- [GetCallerIdentity](#)
- [PutConfigurationRecorder](#)
- [PutDeliveryChannel](#)
- [PutPublicAccessBlock](#)
- [PutRolePolicy](#)
- [Rm](#)
- [StartConfigurationRecorder](#)
- [StopConfigurationRecorder](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

文件文字擷取入門

以下程式碼範例顯示做法：

- 建立 S3 儲存貯體
- 將文件上傳至 S3
- 清除資源

Bash

AWS CLI 搭配 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[範例開發人員教學課程](#)儲存庫中設定和執行。

```
#!/bin/bash

# Amazon Textract Getting Started Tutorial Script
# This script demonstrates how to use Amazon Textract to analyze document text

set -euo pipefail

# Set up logging with restricted permissions
LOG_FILE="textract-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "====="
echo "Amazon Textract Getting Started Tutorial"
echo "====="
echo "This script will guide you through using Amazon Textract to analyze
document text."
echo ""

# Function to check for errors in command output and exit code
check_error() {
    local exit_code=$1
    local output=$2
    local cmd=$3

    if [ $exit_code -ne 0 ] || echo "$output" | grep -i "error" > /dev/null; then
        echo "ERROR: Command failed: $cmd"
        echo "$output" | sed 's/\(aws_secret_access_key\|Authorization\|X-Amz-
Security-Token\).*\/1=***REDACTED***\/g'
        cleanup_on_error
        exit 1
    fi
}
```

```
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Cleaning up resources..."

    # Clean up temporary JSON files
    if [ -f "document.json" ]; then
        rm -f document.json
    fi

    if [ -f "features.json" ]; then
        rm -f features.json
    fi

    if [ -n "${DOCUMENT_NAME:-}" ] && [ -n "${BUCKET_NAME:-}" ]; then
        echo "Deleting document from S3..."
        aws s3 rm "s3://${BUCKET_NAME}/${DOCUMENT_NAME}" || echo "Failed to
delete document"
    fi

    if [ -n "${BUCKET_NAME:-}" ] && [ "${BUCKET_IS_SHARED:-false}" = "false" ];
then
        echo "Deleting S3 bucket..."
        aws s3 rb "s3://${BUCKET_NAME}" --force || echo "Failed to delete bucket"
    fi
}

# Set up trap for cleanup on exit
trap cleanup_on_error EXIT

# Verify AWS CLI is installed and configured
echo "Verifying AWS CLI configuration..."
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed."
    exit 1
fi

AWS_CONFIG_OUTPUT=$(aws configure list 2>&1)
AWS_CONFIG_STATUS=$?
if [ $AWS_CONFIG_STATUS -ne 0 ]; then
    echo "ERROR: AWS CLI is not properly configured."
```

```
    echo "$AWS_CONFIG_OUTPUT" | sed 's/\(aws_secret_access_key\|Authorization
\).*\/1=***REDACTED***/'
    exit 1
fi

# Verify AWS region is configured and supports Textract
AWS_REGION=$(aws configure get region)
if [ -z "$AWS_REGION" ]; then
    echo "ERROR: No AWS region configured. Please run 'aws configure' to set a
default region."
    exit 1
fi

# Check if Textract is available in the configured region
echo "Checking if Amazon Textract is available in region $AWS_REGION..."
TEXTRACT_CHECK=$(aws textract help 2>&1)
TEXTRACT_CHECK_STATUS=$?
if [ $TEXTRACT_CHECK_STATUS -ne 0 ]; then
    echo "ERROR: Amazon Textract may not be available in region $AWS_REGION."
    exit 1
fi

# Generate a random identifier for S3 bucket
RANDOM_ID=$(openssl rand -hex 6)
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
text 2>/dev/null || echo "")
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    BUCKET_NAME="textract-${RANDOM_ID}"
fi
DOCUMENT_NAME="document.png"
RESOURCES_CREATED=()

# Step 1: Create S3 bucket
if [ "$BUCKET_IS_SHARED" = false ]; then
    echo "Creating S3 bucket: $BUCKET_NAME"
```

```

    CREATE_BUCKET_OUTPUT=$(aws s3 mb "s3://$BUCKET_NAME" --region "$AWS_REGION"
2>&1)
    CREATE_BUCKET_STATUS=$?
    echo "$CREATE_BUCKET_OUTPUT"
    check_error $CREATE_BUCKET_STATUS "$CREATE_BUCKET_OUTPUT" "aws s3 mb s3://
$BUCKET_NAME"

    aws s3api put-bucket-tagging \
        --bucket "$BUCKET_NAME" \
        --tagging 'TagSet=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-textract-gs}]'

    # Apply security settings to bucket
    aws s3api put-bucket-versioning --bucket "$BUCKET_NAME" --versioning-
configuration Status=Enabled 2>&1 || true
    aws s3api put-bucket-encryption --bucket "$BUCKET_NAME" --server-side-
encryption-configuration '{"Rules": [{"ApplyServerSideEncryptionByDefault":
{"SSEAlgorithm": "AES256"}]}]' 2>&1 || true
    aws s3api put-bucket-acl --bucket "$BUCKET_NAME" --acl private 2>&1 || true

    RESOURCES_CREATED+=("S3 Bucket: $BUCKET_NAME")
fi

# Step 2: Check if sample document exists, if not create a simple one
if [ ! -f "$DOCUMENT_NAME" ]; then
    echo "Sample document not found. Generating a sample document..."

    # Create a simple PNG document using ImageMagick or convert
    if command -v convert &> /dev/null; then
        convert -size 400x300 xc:white -pointsize 20 -fill black -draw "text
50,50 'Sample Document'" "$DOCUMENT_NAME"
        chmod 600 "$DOCUMENT_NAME"
        echo "Generated sample document: $DOCUMENT_NAME"
    else
        # Fallback: create a minimal valid PNG using base64
        echo "iVBORw0KGgoAAAANSUhEUgAAAAEAAAABCAyAAAAfFcSJAAAADU1EQVR42mNk
+M9QDwADhgGAWjR9awAAAABJRU5ErkJggg==" | base64 -d > "$DOCUMENT_NAME"
        chmod 600 "$DOCUMENT_NAME"
        echo "Created minimal sample document: $DOCUMENT_NAME"
    fi
fi

# Step 3: Upload document to S3
echo "Uploading document to S3..."

```

```
UPLOAD_OUTPUT=$(aws s3 cp ".$DOCUMENT_NAME" "s3://$BUCKET_NAME/" --sse AES256
2>&1)
UPLOAD_STATUS=$?
echo "$UPLOAD_OUTPUT"
check_error $UPLOAD_STATUS "$UPLOAD_OUTPUT" "aws s3 cp ".$DOCUMENT_NAME "s3://
$BUCKET_NAME/"
RESOURCES_CREATED+=("S3 Object: s3://$BUCKET_NAME/$DOCUMENT_NAME")

# Step 4: Analyze document with Amazon Textract
echo "Analyzing document with Amazon Textract..."
echo "This may take a few seconds..."

# Create a JSON file for the document parameter to avoid shell escaping issues
cat > document.json << 'EOF'
{
  "S3Object": {
    "Bucket": "BUCKET_PLACEHOLDER",
    "Name": "DOCUMENT_PLACEHOLDER"
  }
}
EOF

sed -i.bak "s|BUCKET_PLACEHOLDER|$BUCKET_NAME|g; s|DOCUMENT_PLACEHOLDER|
$DOCUMENT_NAME|g" document.json
rm -f document.json.bak
chmod 600 document.json

# Create a JSON file for the feature types parameter
cat > features.json << 'EOF'
["TABLES","FORMS","SIGNATURES"]
EOF
chmod 600 features.json

ANALYZE_OUTPUT=$(aws textract analyze-document --document file://document.json --
feature-types file://features.json 2>&1)
ANALYZE_STATUS=$?

echo "Analysis complete."
if [ $ANALYZE_STATUS -ne 0 ]; then
  echo "ERROR: Document analysis failed"
  echo "$ANALYZE_OUTPUT" | sed 's/\(aws_secret_access_key\|Authorization\|Token
\).*\1=***REDACTED***/'
  exit 1
fi
```

```
# Save the analysis results to a file with restricted permissions
echo "$ANALYZE_OUTPUT" > textract-analysis-results.json
chmod 600 textract-analysis-results.json
echo "Analysis results saved to textract-analysis-results.json"
RESOURCES_CREATED+=("Local file: textract-analysis-results.json")

# Display a summary of the analysis
echo ""
echo "====="
echo "Analysis Summary"
echo "====="
PAGES=$(echo "$ANALYZE_OUTPUT" | grep -o '"Pages": [0-9]*' | head -1 | awk
'{print $2}')
echo "Document pages: $PAGES"

BLOCKS_COUNT=$(echo "$ANALYZE_OUTPUT" | grep -o '"BlockType":' | wc -l)
echo "Total blocks detected: $BLOCKS_COUNT"

# Count different block types using jq if available, fallback to grep
PAGE_COUNT=$(echo "$ANALYZE_OUTPUT" | grep -o '"BlockType": "PAGE"' | wc -l ||
echo 0)
LINE_COUNT=$(echo "$ANALYZE_OUTPUT" | grep -o '"BlockType": "LINE"' | wc -l ||
echo 0)
WORD_COUNT=$(echo "$ANALYZE_OUTPUT" | grep -o '"BlockType": "WORD"' | wc -l ||
echo 0)
TABLE_COUNT=$(echo "$ANALYZE_OUTPUT" | grep -o '"BlockType": "TABLE"' | wc -l ||
echo 0)
CELL_COUNT=$(echo "$ANALYZE_OUTPUT" | grep -o '"BlockType": "CELL"' | wc -l ||
echo 0)
KEY_VALUE_COUNT=$(echo "$ANALYZE_OUTPUT" | grep -o '"BlockType": "KEY_VALUE_SET"'
| wc -l || echo 0)
SIGNATURE_COUNT=$(echo "$ANALYZE_OUTPUT" | grep -o '"BlockType": "SIGNATURE"' |
wc -l || echo 0)

echo "Pages: $PAGE_COUNT"
echo "Lines of text: $LINE_COUNT"
echo "Words: $WORD_COUNT"
echo "Tables: $TABLE_COUNT"
echo "Table cells: $CELL_COUNT"
echo "Key-value pairs: $KEY_VALUE_COUNT"
echo "Signatures: $SIGNATURE_COUNT"
echo ""
```

```

# Cleanup confirmation
echo ""
echo "=====
echo "RESOURCES CREATED"
echo "=====
for resource in "${RESOURCES_CREATED[@]}"; do
    echo "- $resource"
done
echo ""
echo "=====
echo "CLEANUP CONFIRMATION"
echo "=====
echo "Cleaning up resources..."

# Delete document from S3
echo "Deleting document from S3..."
DELETE_DOC_OUTPUT=$(aws s3 rm "s3://$BUCKET_NAME/$DOCUMENT_NAME" 2>&1)
DELETE_DOC_STATUS=$?
echo "$DELETE_DOC_OUTPUT"
check_error $DELETE_DOC_STATUS "$DELETE_DOC_OUTPUT" "aws s3 rm s3://$BUCKET_NAME/
$DOCUMENT_NAME"

# Delete S3 bucket (only if not shared)
if [ "$BUCKET_IS_SHARED" = false ]; then
    echo "Deleting S3 bucket..."
    DELETE_BUCKET_OUTPUT=$(aws s3 rb "s3://$BUCKET_NAME" --force 2>&1)
    DELETE_BUCKET_STATUS=$?
    echo "$DELETE_BUCKET_OUTPUT"
    check_error $DELETE_BUCKET_STATUS "$DELETE_BUCKET_OUTPUT" "aws s3 rb s3://
$BUCKET_NAME --force"
fi

# Delete local JSON files
rm -f document.json features.json

echo "Cleanup complete. The analysis results file (textract-analysis-
results.json) has been kept."

echo ""
echo "=====
echo "Tutorial complete!"
echo "=====
echo "You have successfully analyzed a document using Amazon Textract."
echo "The analysis results are available in textract-analysis-results.json"

```



```
echo ""  
  
trap - EXIT
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
 - [AnalyzeDocument](#)
 - [Cp](#)
 - [Help \(說明\)](#)
 - [Mb](#)
 - [Rb](#)
 - [Rm](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

機器學習功能存放區的入門

以下程式碼範例顯示做法：

- 設定 IAM 許可
- 建立一個 SageMaker 執行角色
- 建立特徵群組
- 清除資源

Bash

AWS CLI 搭配 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[範例開發人員教學課程](#)儲存庫中設定和執行。

```
#!/bin/bash  
  
# Amazon SageMaker Feature Store Tutorial Script - Version 3
```

```
# This script demonstrates how to use Amazon SageMaker Feature Store with AWS CLI

# Setup logging
LOG_FILE="sagemaker-featurestore-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting SageMaker Feature Store tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"
echo ""

# Track created resources for cleanup
CREATED_RESOURCES=()

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command status
check_status() {
    if echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to wait for feature group to be created
wait_for_feature_group() {
    local feature_group_name=$1
    local status="Creating"

    echo "Waiting for feature group ${feature_group_name} to be created..."

    while [ "$status" = "Creating" ]; do
        sleep 5
        status=$(aws sagemaker describe-feature-group \
            --feature-group-name "${feature_group_name}" \
            --query 'FeatureGroupStatus' \
            --output text)
        echo "Current status: ${status}"

        if [ "$status" = "Failed" ]; then
```

```

        handle_error "Feature group ${feature_group_name} creation failed"
    fi
done

    echo "Feature group ${feature_group_name} is now ${status}"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Clean up in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[$i]}"
        resource_type=$(echo "$resource" | cut -d: -f1)
        resource_name=$(echo "$resource" | cut -d: -f2)

        echo "Deleting $resource_type: $resource_name"

        case "$resource_type" in
            "FeatureGroup")
                aws sagemaker delete-feature-group --feature-group-name
"$resource_name"
                ;;
            "S3Bucket")
                echo "Emptying S3 bucket: $resource_name"
                aws s3 rm "s3://$resource_name" --recursive 2>/dev/null
                echo "Deleting S3 bucket: $resource_name"
                aws s3api delete-bucket --bucket "$resource_name" 2>/dev/null
                ;;
            "IAMRole")
                echo "Detaching policies from role: $resource_name"
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>/dev/null
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>/dev/null
                echo "Deleting IAM role: $resource_name"
                aws iam delete-role --role-name "$resource_name" 2>/dev/null
                ;;
            *)
                echo "Unknown resource type: $resource_type"
                ;;
        esac
    done
}

```

```
}

# Function to create SageMaker execution role
create_sagemaker_role() {
    local role_name="SageMakerFeatureStoreRole-$(openssl rand -hex 4)"

    echo "Creating SageMaker execution role: $role_name" >&2

    # Create trust policy document
    local trust_policy='{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "sagemaker.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }'

    # Create the role
    local role_result=$(aws iam create-role \
        --role-name "$role_name" \
        --assume-role-policy-document "$trust_policy" \
        --description "SageMaker execution role for Feature Store tutorial" 2>&1)

    if echo "$role_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to create IAM role: $role_result"
    fi

    echo "Role created successfully" >&2
    CREATED_RESOURCES+=("IAMRole:$role_name")

    # Tag the role
    echo "Tagging IAM role..." >&2
    aws iam tag-role --role-name "$role_name" --tags Key=project,Value=doc-smith
    Key=tutorial,Value=sagemaker-featurestore 2>&1

    # Attach necessary policies
    echo "Attaching policies to role..." >&2

    # SageMaker execution policy
```

```

local policy1_result=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>&1)

if echo "$policy1_result" | grep -i "error" > /dev/null; then
    handle_error "Failed to attach SageMaker policy: $policy1_result"
fi

# S3 access policy
local policy2_result=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>&1)

if echo "$policy2_result" | grep -i "error" > /dev/null; then
    handle_error "Failed to attach S3 policy: $policy2_result"
fi

# Get account ID for role ARN
local account_id=$(aws sts get-caller-identity --query Account --output text)
local role_arn="arn:aws:iam::${account_id}:role/${role_name}"

echo "Role ARN: $role_arn" >&2
echo "Waiting 10 seconds for role to propagate..." >&2
sleep 10

# Return only the role ARN to stdout
echo "$role_arn"
}

# Handle SageMaker execution role
ROLE_ARN=""

if [ -z "$1" ]; then
    echo "Creating SageMaker execution role automatically..."
    ROLE_ARN=$(create_sagemaker_role)
    if [ -z "$ROLE_ARN" ]; then
        handle_error "Failed to create SageMaker execution role"
    fi
else
    ROLE_ARN="$1"

    # Validate the role ARN
    ROLE_NAME=$(echo "$ROLE_ARN" | sed 's/.*role\\///')
    ROLE_CHECK=$(aws iam get-role --role-name "$ROLE_NAME" 2>&1)

```

```
    if echo "$ROLE_CHECK" | grep -i "error" > /dev/null; then
        echo "Creating a new role automatically..."
        ROLE_ARN=$(create_sagemaker_role)
        if [ -z "$ROLE_ARN" ]; then
            handle_error "Failed to create SageMaker execution role"
        fi
    fi
fi

# Handle cleanup option
AUTO_CLEANUP=""
if [ -n "$2" ]; then
    AUTO_CLEANUP="$2"
fi

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
echo "Using random identifier: $RANDOM_ID"

# Set variables
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_status "$ACCOUNT_ID"
echo "Account ID: $ACCOUNT_ID"

# Get current region
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No default region configured, using: $REGION"
else
    echo "Using region: $REGION"
fi

# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="sagemaker-featurestore-${RANDOM_ID}-${ACCOUNT_ID}"
```

```
fi
PREFIX="featurestore-tutorial"
CURRENT_TIME=$(date +%s)

echo "Creating S3 bucket: $S3_BUCKET_NAME"
# Create bucket in current region (skip if using shared bucket)
if [ "$BUCKET_IS_SHARED" = "false" ]; then
if [ "$REGION" = "us-east-1" ]; then
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" 2>&1)
else
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" \
        --create-bucket-configuration LocationConstraint="$REGION" 2>&1)
fi

if echo "$BUCKET_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to create S3 bucket: $BUCKET_RESULT"
    exit 1
fi

echo "$BUCKET_RESULT"
CREATED_RESOURCES+=("S3Bucket:$S3_BUCKET_NAME")

# Tag the S3 bucket
echo "Tagging S3 bucket: $S3_BUCKET_NAME"
aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
    'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=sagemaker-
featurestore}]' 2>&1

# Block public access to the bucket
BLOCK_RESULT=$(aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
    "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
2>&1)

if echo "$BLOCK_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to block public access to S3 bucket: $BLOCK_RESULT"
    cleanup_resources
    exit 1
fi
else
    echo "Using shared bucket (skipping creation)"
```

```

fi

# Create feature groups
echo "Creating feature groups..."

# Create customers feature group
CUSTOMERS_FEATURE_GROUP_NAME="customers-feature-group-`${RANDOM_ID}`"
echo "Creating customers feature group: $CUSTOMERS_FEATURE_GROUP_NAME"

CUSTOMERS_RESPONSE=$(aws sagemaker create-feature-group \
  --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
  --record-identifier-feature-name "customer_id" \
  --event-time-feature-name "EventTime" \
  --feature-definitions '[
    {"FeatureName": "customer_id", "FeatureType": "Integral"},
    {"FeatureName": "name", "FeatureType": "String"},
    {"FeatureName": "age", "FeatureType": "Integral"},
    {"FeatureName": "address", "FeatureType": "String"},
    {"FeatureName": "membership_type", "FeatureType": "String"},
    {"FeatureName": "EventTime", "FeatureType": "Fractional"}
  ]' \
  --online-store-config '{"EnableOnlineStore": true}' \
  --offline-store-config '{
    "S3StorageConfig": {
      "S3Uri": "s3://`${S3_BUCKET_NAME}`/`${PREFIX}`"
    },
    "DisableGlueTableCreation": false
  }' \
  --role-arn "$ROLE_ARN" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
2>&1)

if echo "$CUSTOMERS_RESPONSE" | grep -i "error" > /dev/null; then
  echo "Failed to create customers feature group: $CUSTOMERS_RESPONSE"
  cleanup_resources
  exit 1
fi

echo "$CUSTOMERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$CUSTOMERS_FEATURE_GROUP_NAME")

# Create orders feature group
ORDERS_FEATURE_GROUP_NAME="orders-feature-group-`${RANDOM_ID}`"
echo "Creating orders feature group: $ORDERS_FEATURE_GROUP_NAME"

```



```

ORDERS_RESPONSE=$(aws sagemaker create-feature-group \
  --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
  --record-identifier-feature-name "customer_id" \
  --event-time-feature-name "EventTime" \
  --feature-definitions '[
    {"FeatureName": "customer_id", "FeatureType": "Integral"},
    {"FeatureName": "order_id", "FeatureType": "String"},
    {"FeatureName": "order_date", "FeatureType": "String"},
    {"FeatureName": "product", "FeatureType": "String"},
    {"FeatureName": "quantity", "FeatureType": "Integral"},
    {"FeatureName": "amount", "FeatureType": "Fractional"},
    {"FeatureName": "EventTime", "FeatureType": "Fractional"}
  ]' \
  --online-store-config '{"EnableOnlineStore": true}' \
  --offline-store-config '{
    "S3StorageConfig": {
      "S3Uri": "s3://'$S3_BUCKET_NAME'/'$PREFIX'"
    },
    "DisableGlueTableCreation": false
  }' \
  --role-arn "$ROLE_ARN" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
2>&1)

if echo "$ORDERS_RESPONSE" | grep -i "error" > /dev/null; then
  echo "Failed to create orders feature group: $ORDERS_RESPONSE"
  cleanup_resources
  exit 1
fi

echo "$ORDERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$ORDERS_FEATURE_GROUP_NAME")

# Wait for feature groups to be created
wait_for_feature_group "$CUSTOMERS_FEATURE_GROUP_NAME"
wait_for_feature_group "$ORDERS_FEATURE_GROUP_NAME"

# Ingest data into feature groups
echo "Ingesting data into feature groups..."

# Ingest customer data
echo "Ingesting customer data..."
CUSTOMER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \

```

```

--feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
--record '[
    {"FeatureName": "customer_id", "ValueAsString": "573291"},
    {"FeatureName": "name", "ValueAsString": "John Doe"},
    {"FeatureName": "age", "ValueAsString": "35"},
    {"FeatureName": "address", "ValueAsString": "123 Main St"},
    {"FeatureName": "membership_type", "ValueAsString": "premium"},
    {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
]' 2>&1)

if echo "$CUSTOMER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 1 data: $CUSTOMER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER1_RESPONSE"

CUSTOMER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "name", "ValueAsString": "Jane Smith"},
        {"FeatureName": "age", "ValueAsString": "28"},
        {"FeatureName": "address", "ValueAsString": "456 Oak Ave"},
        {"FeatureName": "membership_type", "ValueAsString": "standard"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 2 data: $CUSTOMER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER2_RESPONSE"

# Ingest order data
echo "Ingesting order data..."
ORDER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-001"},

```

```

        {"FeatureName": "order_date", "ValueAsString": "2023-01-15"},
        {"FeatureName": "product", "ValueAsString": "Laptop"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "1299.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 1 data: $ORDER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER1_RESPONSE"

ORDER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-002"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-20"},
        {"FeatureName": "product", "ValueAsString": "Smartphone"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "899.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 2 data: $ORDER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER2_RESPONSE"

# Retrieve records from feature groups
echo "Retrieving records from feature groups..."

# Get a single customer record
echo "Getting customer record with ID 573291:"
CUSTOMER_RECORD=$(aws sagemaker-featurestore-runtime get-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-value-as-string "573291" 2>&1)

```

```
if echo "$CUSTOMER_RECORD" | grep -i "error" > /dev/null; then
    echo "Failed to get customer record: $CUSTOMER_RECORD"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER_RECORD"

# Get multiple records using batch-get-record
echo "Getting multiple records using batch-get-record:"
BATCH_RECORDS=$(aws sagemaker-featurestore-runtime batch-get-record \
    --identifiers '[
        {
            "FeatureGroupName": "'${CUSTOMERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        },
        {
            "FeatureGroupName": "'${ORDERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        }
    ]' 2>&1)

if echo "$BATCH_RECORDS" | grep -i "error" > /dev/null && ! echo "$BATCH_RECORDS"
| grep -i "Records" > /dev/null; then
    echo "Failed to get batch records: $BATCH_RECORDS"
    cleanup_resources
    exit 1
fi

echo "$BATCH_RECORDS"

# List feature groups
echo "Listing feature groups:"
FEATURE_GROUPS=$(aws sagemaker list-feature-groups 2>&1)

if echo "$FEATURE_GROUPS" | grep -i "error" > /dev/null; then
    echo "Failed to list feature groups: $FEATURE_GROUPS"
    cleanup_resources
    exit 1
fi

echo "$FEATURE_GROUPS"

# Display summary of created resources
```

```

echo ""
echo "=====
echo "TUTORIAL COMPLETED SUCCESSFULLY!"
echo "=====
echo "Resources created:"
echo "- S3 Bucket: $S3_BUCKET_NAME"
echo "- Customers Feature Group: $CUSTOMERS_FEATURE_GROUP_NAME"
echo "- Orders Feature Group: $ORDERS_FEATURE_GROUP_NAME"
if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
    echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o 'IAMRole:
[^[:space:]]*' | cut -d: -f2)"
fi
echo ""
echo "You can now:"
echo "1. View your feature groups in the SageMaker console"
echo "2. Query the offline store using Amazon Athena"
echo "3. Use the feature groups in your ML workflows"
echo "=====
echo ""

# Handle cleanup
if [ "$AUTO_CLEANUP" = "y" ]; then
    echo "Auto-cleanup enabled. Starting cleanup..."
    cleanup_resources
    echo "Cleanup completed."
elif [ "$AUTO_CLEANUP" = "n" ]; then
    echo "Auto-cleanup disabled. Resources will remain in your account."
    echo "To clean up later, run this script again with cleanup option 'y'"
else
    echo "=====
    echo "CLEANUP CONFIRMATION"
    echo "=====
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup..."
        cleanup_resources
        echo "Cleanup completed."
    else
        echo "Skipping cleanup. Resources will remain in your account."
        echo "To clean up later, delete the following resources:"
        echo "- Feature Groups: $CUSTOMERS_FEATURE_GROUP_NAME,
$ORDERS_FEATURE_GROUP_NAME"

```

```
    echo "- S3 Bucket: $S3_BUCKET_NAME"
    if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
        echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o
'IAMRole:[^[:space:]]*' | cut -d: -f2)"
    fi
    echo ""
    echo "Estimated ongoing cost: ~$0.01 per month for online store"
fi
fi

echo "Script completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。

- [AttachRolePolicy](#)
- [CreateBucket](#)
- [CreateFeatureGroup](#)
- [CreateRole](#)
- [DeleteBucket](#)
- [DeleteFeatureGroup](#)
- [DeleteRole](#)
- [DescribeFeatureGroup](#)
- [DetachRolePolicy](#)
- [GetCallerIdentity](#)
- [GetRole](#)
- [ListFeatureGroups](#)
- [PutPublicAccessBlock](#)
- [Rm](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

物件儲存入門

以下程式碼範例顯示做法：

建立您的第一個 S3 儲存貯體

案例

- 上傳物件
- 啟用版本控制
- 設定預設加密
- 將標籤新增至您的儲存貯體
- 列出物件和版本
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[範例開發人員教學課程](#)儲存庫中設定和執行。

```
#!/bin/bash
# S3 Getting Started - Create a bucket, upload and download objects, copy to a
# folder prefix, enable versioning, configure encryption and public access
# blocking, tag the bucket, list objects and versions, and clean up.

set -eE
set -o pipefail

# =====
# Prerequisites check
# =====

CONFIGURED_REGION=$(aws configure get region 2>/dev/null || true)
if [ -z "$CONFIGURED_REGION" ] && [ -z "$AWS_DEFAULT_REGION" ] && [ -z
"$AWS_REGION" ]; then
    echo "ERROR: No AWS region configured. Run 'aws configure' or set
AWS_DEFAULT_REGION."
    exit 1
fi

# Verify AWS credentials are configured
if ! aws sts get-caller-identity &>/dev/null; then
```

```

        echo "ERROR: AWS credentials not configured or invalid. Run 'aws configure'."
        exit 1
    fi

    # =====
    # Setup: logging, temp directory, resource tracking
    # =====

    UNIQUE_ID=$(head -c 6 /dev/urandom | od -An -tx1 | tr -d ' ')
    # Check for shared prereq bucket
    PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
        --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
text 2>/dev/null || true)
    if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
        BUCKET_NAME="$PREREQ_BUCKET"
        BUCKET_IS_SHARED=true
        echo "Using shared bucket: $BUCKET_NAME"
    else
        BUCKET_IS_SHARED=false
        BUCKET_NAME="s3api-${UNIQUE_ID}"
    fi

    TEMP_DIR=$(mktemp -d)
    trap 'rm -rf "$TEMP_DIR"' EXIT
    LOG_FILE="$TEMP_DIR/s3-gettingstarted.log"
    CREATED_RESOURCES=()

    exec > >(tee -a "$LOG_FILE") 2>&1

    echo "======"
    echo "S3 Getting Started"
    echo "======"
    echo "Bucket name: ${BUCKET_NAME}"
    echo "Temp directory: ${TEMP_DIR}"
    echo "Log file: ${LOG_FILE}"
    echo ""

    # =====
    # Helper functions
    # =====

    get_region() {
        echo "${AWS_REGION:-${AWS_DEFAULT_REGION:-${CONFIGURED_REGION}}}"
    }

```



```

}

delete_object_versions() {
    local bucket=$1
    local query=$2

    local versions
    versions=$(aws s3api list-object-versions \
        --bucket "$bucket" \
        --query "$query" \
        --output json 2>&1) || return 0

    if [ -z "$versions" ] || [ "$versions" = "null" ] || [ "$versions" = "[]" ];
    then
        return 0
    fi

    echo "$versions" | jq -r '.[ ] | "\(.Key)\t\(.VersionId)'" 2>/dev/null | while
    IFS=$'\t' read -r key version_id; do
        if [ -n "$key" ] && [ "$key" != "null" ]; then
            aws s3api delete-object --bucket "$bucket" --key "$key" --version-id
            "$version_id" >/dev/null 2>&1 || true
        fi
    done

    return 0
}

# =====
# Error handling and cleanup functions
# =====

cleanup() {
    echo ""
    echo "=====
    echo "CLEANUP"
    echo "=====

    if [ "$BUCKET_IS_SHARED" = "false" ]; then
        echo "Deleting all object versions in bucket..."

        delete_object_versions "$BUCKET_NAME" "Versions[ ].
        {Key:Key,VersionId:VersionId}" || true

```

```

        delete_object_versions "$BUCKET_NAME" "DeleteMarkers[]."
{Key:Key,VersionId:VersionId}" || true

    echo "Deleting bucket: ${BUCKET_NAME}"
    if ! aws s3api delete-bucket --bucket "$BUCKET_NAME" 2>/dev/null; then
        echo "WARNING: Failed to delete bucket ${BUCKET_NAME}"
    fi

    # Clean up logs bucket
    LOG_TARGET_BUCKET="${BUCKET_NAME}-logs"
    if aws s3api head-bucket --bucket "$LOG_TARGET_BUCKET" 2>/dev/null; then
        echo "Deleting log bucket: ${LOG_TARGET_BUCKET}"
        if ! aws s3api delete-bucket --bucket "$LOG_TARGET_BUCKET" 2>/dev/
null; then
            echo "WARNING: Failed to delete bucket ${LOG_TARGET_BUCKET}"
        fi
    fi
else
    echo "Keeping shared bucket: ${BUCKET_NAME}"
fi

echo ""
echo "Cleanup complete."
}

handle_error() {
    local line_number=$1
    echo ""
    echo "=====
echo "ERROR on line ${line_number}"
    echo "=====
    echo ""
    echo "Resources created before error:"
    if [ ${#CREATED_RESOURCES[@]} -gt 0 ]; then
        for RESOURCE in "${CREATED_RESOURCES[@]}"; do
            echo " - ${RESOURCE}"
        done
    else
        echo " (none)"
    fi
    echo ""
    echo "Attempting cleanup..."
    cleanup
    exit 1

```

```

}

trap 'handle_error "$LINENO"' ERR

# =====
# Step 1: Create a bucket
# =====

echo "Step 1: Creating bucket ${BUCKET_NAME}..."
if [ "$BUCKET_IS_SHARED" = "false" ]; then
    REGION=$(get_region)
    if [ "$REGION" = "us-east-1" ]; then
        if ! aws s3api create-bucket --bucket "$BUCKET_NAME" >/dev/null 2>&1;
        then
            echo "ERROR: Failed to create bucket $BUCKET_NAME"
            exit 1
        fi
    else
        if ! aws s3api create-bucket \
            --bucket "$BUCKET_NAME" \
            --region "$REGION" \
            --create-bucket-configuration LocationConstraint="$REGION" >/dev/null
        2>&1; then
            echo "ERROR: Failed to create bucket $BUCKET_NAME in region $REGION"
            exit 1
        fi
    fi
    CREATED_RESOURCES+=("s3:bucket:${BUCKET_NAME}")
    echo "Bucket created."

    if ! aws s3api put-bucket-tagging \
        --bucket "$BUCKET_NAME" \
        --tagging '{
            "TagSet": [
                {
                    "Key": "project",
                    "Value": "doc-smith"
                },
                {
                    "Key": "tutorial",
                    "Value": "s3-gettingstarted"
                }
            ]
        }' >/dev/null 2>&1; then

```

```

        echo "WARNING: Failed to tag bucket"
    fi
fi
echo ""

# =====
# Step 2: Upload a sample text file
# =====

echo "Step 2: Uploading a sample text file..."

SAMPLE_FILE="${TEMP_DIR}/sample.txt"
cat > "$SAMPLE_FILE" << 'EOF'
Hello, Amazon S3! This is a sample file for the getting started tutorial.
EOF

if ! aws s3api put-object \
    --bucket "$BUCKET_NAME" \
    --key "sample.txt" \
    --body "$SAMPLE_FILE" \
    --server-side-encryption AES256 \
    --metadata "tutorial=s3-gettingstarted" >/dev/null 2>&1; then
    echo "ERROR: Failed to upload sample.txt"
    exit 1
fi
echo "File uploaded."
echo ""

# =====
# Step 3: Download the object
# =====

echo "Step 3: Downloading the object..."

DOWNLOAD_FILE="${TEMP_DIR}/downloaded-sample.txt"
if ! aws s3api get-object \
    --bucket "$BUCKET_NAME" \
    --key "sample.txt" \
    "$DOWNLOAD_FILE" >/dev/null 2>&1; then
    echo "ERROR: Failed to download sample.txt"
    exit 1
fi
echo "Downloaded to: ${DOWNLOAD_FILE}"
echo "Contents:"

```

```
cat "$DOWNLOAD_FILE"
echo ""

# =====
# Step 4: Copy the object to a folder prefix
# =====

echo "Step 4: Copying object to a folder prefix..."

if ! aws s3api copy-object \
    --bucket "$BUCKET_NAME" \
    --copy-source "${BUCKET_NAME}/sample.txt" \
    --key "backup/sample.txt" \
    --server-side-encryption AES256 \
    --metadata-directive COPY >/dev/null 2>&1; then
    echo "ERROR: Failed to copy object to backup/sample.txt"
    exit 1
fi
echo "Object copied to backup/sample.txt."
echo ""

# =====
# Step 5: Enable versioning and upload a second version
# =====

echo "Step 5: Enabling versioning..."

if ! aws s3api put-bucket-versioning \
    --bucket "$BUCKET_NAME" \
    --versioning-configuration Status=Enabled >/dev/null 2>&1; then
    echo "ERROR: Failed to enable versioning"
    exit 1
fi
echo "Versioning enabled."

echo "Uploading a second version of sample.txt..."
cat > "$SAMPLE_FILE" << 'EOF'
Hello, Amazon S3! This is version 2 of the sample file.
EOF

if ! aws s3api put-object \
    --bucket "$BUCKET_NAME" \
    --key "sample.txt" \
    --body "$SAMPLE_FILE" \
```

```

--server-side-encryption AES256 \
--metadata "tutorial=s3-gettingstarted,version=2" >/dev/null 2>&1; then
echo "ERROR: Failed to upload second version of sample.txt"
exit 1
fi
echo "Second version uploaded."
echo ""

# =====
# Step 6: Configure SSE-S3 encryption
# =====

echo "Step 6: Configuring SSE-S3 default encryption..."

if ! aws s3api put-bucket-encryption \
--bucket "$BUCKET_NAME" \
--server-side-encryption-configuration '{
    "Rules": [
        {
            "ApplyServerSideEncryptionByDefault": {
                "SSEAlgorithm": "AES256"
            },
            "BucketKeyEnabled": true
        }
    ]
}' >/dev/null 2>&1; then
echo "ERROR: Failed to configure SSE-S3 encryption"
exit 1
fi
echo "SSE-S3 encryption configured."
echo ""

# =====
# Step 7: Block all public access
# =====

echo "Step 7: Blocking all public access..."

if ! aws s3api put-public-access-block \
--bucket "$BUCKET_NAME" \
--public-access-block-configuration '{
    "BlockPublicAcls": true,
    "IgnorePublicAcls": true,
    "BlockPublicPolicy": true,

```

```

        "RestrictPublicBuckets": true
    }' >/dev/null 2>&1; then
        echo "ERROR: Failed to block public access"
        exit 1
    fi
    echo "Public access blocked."
    echo ""

    # =====
    # Step 8: Configure bucket logging
    # =====

    echo "Step 8: Configuring bucket logging..."

    LOG_TARGET_BUCKET="${BUCKET_NAME}-logs"
    if [ "$BUCKET_IS_SHARED" = "false" ]; then
        REGION=$(get_region)
        if [ "$REGION" = "us-east-1" ]; then
            aws s3api create-bucket --bucket "$LOG_TARGET_BUCKET" >/dev/null 2>&1 ||
            true
        else
            aws s3api create-bucket \
                --bucket "$LOG_TARGET_BUCKET" \
                --region "$REGION" \
                --create-bucket-configuration LocationConstraint="$REGION" >/dev/null
            2>&1 || true
        fi

        if ! aws s3api put-bucket-tagging \
            --bucket "$LOG_TARGET_BUCKET" \
            --tagging '{
                "TagSet": [
                    {
                        "Key": "project",
                        "Value": "doc-smith"
                    },
                    {
                        "Key": "tutorial",
                        "Value": "s3-gettingstarted"
                    }
                ]
            }' >/dev/null 2>&1; then
            echo "WARNING: Failed to tag log bucket"
        fi
    fi

```

```
aws s3api put-bucket-acl --bucket "$LOG_TARGET_BUCKET" --acl log-delivery-
write 2>/dev/null || true

if ! aws s3api put-bucket-logging \
  --bucket "$BUCKET_NAME" \
  --bucket-logging-status '{
    "LoggingEnabled": {
      "TargetBucket": "'$LOG_TARGET_BUCKET'",
      "TargetPrefix": "logs/"
    }
  }' >/dev/null 2>&1; then
  echo "WARNING: Failed to configure bucket logging"
else
  echo "Bucket logging configured."
fi
else
  echo "Skipping logging configuration for shared bucket."
fi
echo ""

# =====
# Step 9: Tag the bucket
# =====

echo "Step 9: Tagging the bucket..."

if ! aws s3api put-bucket-tagging \
  --bucket "$BUCKET_NAME" \
  --tagging '{
    "TagSet": [
      {
        "Key": "project",
        "Value": "doc-smith"
      },
      {
        "Key": "tutorial",
        "Value": "s3-gettingstarted"
      },
      {
        "Key": "Environment",
        "Value": "Tutorial"
      },
      {
```



```

        "Key": "Project",
        "Value": "S3-GettingStarted"
    },
    {
        "Key": "ManagedBy",
        "Value": "Bash-Tutorial"
    }
]
}' >/dev/null 2>&1; then
echo "ERROR: Failed to tag bucket"
exit 1
fi
echo "Bucket tagged."

echo "Verifying tags..."
if ! aws s3api get-bucket-tagging --bucket "$BUCKET_NAME" 2>&1; then
    echo "WARNING: Failed to retrieve bucket tags"
fi
echo ""

# =====
# Step 10: List objects and versions
# =====

echo "Step 10: Listing objects..."

if ! aws s3api list-objects-v2 --bucket "$BUCKET_NAME" 2>&1; then
    echo "WARNING: Failed to list objects"
fi
echo ""

echo "Listing object versions..."

if ! aws s3api list-object-versions --bucket "$BUCKET_NAME" 2>&1; then
    echo "WARNING: Failed to list object versions"
fi
echo ""

# =====
# Step 11: Cleanup
# =====

echo ""
echo "=====
```

```
echo "TUTORIAL COMPLETE"
echo "======"
echo ""
echo "Resources created:"
if [ ${#CREATED_RESOURCES[@]} -gt 0 ]; then
    for RESOURCE in "${CREATED_RESOURCES[@}"; do
        echo "  - ${RESOURCE}"
    done
else
    echo "  (none)"
fi
echo ""
echo "======"
echo "CLEANUP"
echo "======"
echo "Cleaning up all created resources..."
cleanup

echo ""
echo "Done."
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。

- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObjects](#)
- [GetObject](#)
- [HeadObject](#)
- [ListObjectVersions](#)
- [ListObjectsV2](#)
- [PutBucketEncryption](#)
- [PutBucketTagging](#)
- [PutBucketVersioning](#)
- [PutObject](#)
- [PutPublicAccessBlock](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

查詢分析入門

以下程式碼範例顯示做法：

- 為查詢結果建立 S3 儲存貯體
- 建立 資料庫
- 建立資料表
- 執行查詢
- 建立和使用具名查詢
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[範例開發人員教學課程](#)儲存庫中設定和執行。

```
#!/bin/bash

# Amazon Athena Getting Started Script
# This script demonstrates how to use Amazon Athena with AWS CLI
# It creates a database, table, runs queries, and manages named queries

set -euo pipefail

# Security: Validate AWS credentials are configured
if ! aws sts get-caller-identity &>/dev/null; then
    echo "ERROR: AWS credentials not configured or invalid"
    exit 1
fi

# Security: Restrict umask to prevent world-readable files
```

```
umask 0077

# Set up logging with restricted permissions
LOG_FILE="athena-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Athena Getting Started Tutorial..."
echo "Logging to $LOG_FILE"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created:"
    if [ -n "${NAMED_QUERY_ID:-}" ]; then
        echo "- Named Query: $NAMED_QUERY_ID"
    fi
    if [ -n "${DATABASE_NAME:-}" ]; then
        echo "- Database: $DATABASE_NAME"
        if [ -n "${TABLE_NAME:-}" ]; then
            echo "- Table: $TABLE_NAME in $DATABASE_NAME"
        fi
    fi
    if [ -n "${S3_BUCKET:-}" ]; then
        echo "- S3 Bucket: $S3_BUCKET"
    fi

    echo "Exiting..."
    exit 1
}

# Security: Validate bucket name format
validate_bucket_name() {
    local bucket_name="$1"
    if [[ ! "$bucket_name" =~ ^[a-z0-9][a-z0-9-]*[a-z0-9]$ ]] ||
    [ ${#bucket_name} -lt 3 ] || [ ${#bucket_name} -gt 63 ]; then
        return 1
    fi
    return 0
}

# Security: Validate database and table names
validate_identifier() {
```

```

    local identifier="$1"
    if [[ ! "$identifier" =~ ^[a-zA-Z_][a-zA-Z0-9_]*$ ]]; then
        return 1
    fi
    return 0
}

# Security: Safely generate random identifier
if ! command -v openssl &>/dev/null; then
    RANDOM_ID=$(head -c 6 /dev/urandom | od -An -tx1 | tr -d ' ')
else
    RANDOM_ID=$(openssl rand -hex 6)
fi

# Security: Validate random ID format
if [[ ! "$RANDOM_ID" =~ ^[a-f0-9]{12}$ ]]; then
    handle_error "Failed to generate valid random ID"
fi

# Check for shared prereq bucket with proper error handling
PREREQ_BUCKET=""
if aws cloudformation describe-stacks --stack-name tutorial-prereqs-bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
    text 2>/dev/null | grep -qv "^$"; then
    PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-
prereqs-bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --
output text 2>/dev/null)
fi

if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET="athena-${RANDOM_ID}"
fi

if ! validate_bucket_name "$S3_BUCKET"; then
    handle_error "Invalid S3 bucket name: $S3_BUCKET"
fi

DATABASE_NAME="mydatabase"

```

```
TABLE_NAME="cloudfront_logs"

if ! validate_identifier "$DATABASE_NAME"; then
    handle_error "Invalid database name: $DATABASE_NAME"
fi

if ! validate_identifier "$TABLE_NAME"; then
    handle_error "Invalid table name: $TABLE_NAME"
fi

# Get the current AWS region with validation
AWS_REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"
    echo "No AWS region found in configuration, defaulting to $AWS_REGION"
fi

# Security: Validate region format - expanded regex for newer regions
if [[ ! "$AWS_REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]{1}$ ]] && [[ ! "$AWS_REGION" =~
    ^[a-z]+-[a-z]+-[0-9]{1}$ ]]; then
    echo "WARNING: Region format may be invalid: $AWS_REGION"
fi

echo "Using AWS Region: $AWS_REGION"

# Create S3 bucket for Athena query results
echo "Creating S3 bucket for Athena query results: $S3_BUCKET"
if [ "$BUCKET_IS_SHARED" = false ]; then
    CREATE_BUCKET_RESULT=$(aws s3 mb "s3://$S3_BUCKET" --region "$AWS_REGION"
2>&1)
    if echo "$CREATE_BUCKET_RESULT" | grep -qi "error\|failed"; then
        handle_error "Failed to create S3 bucket: $CREATE_BUCKET_RESULT"
    fi

    aws s3api put-bucket-tagging \
        --bucket "$S3_BUCKET" \
        --tagging 'TagSet=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-athena-gs}]'

# Security: Enable S3 bucket encryption with KMS validation
echo "Enabling default encryption on S3 bucket..."
if ! aws s3api put-bucket-encryption \
    --bucket "$S3_BUCKET" \
    --server-side-encryption-configuration '{
```

```

        "Rules": [{
            "ApplyServerSideEncryptionByDefault": {
                "SSEAlgorithm": "AES256"
            }
        }]
    }' 2>&1; then
    echo "Warning: Could not enable encryption on bucket"
fi

# Security: Block public access
echo "Blocking public access to S3 bucket..."
if ! aws s3api put-public-access-block \
    --bucket "$S3_BUCKET" \
    --public-access-block-configuration \

"BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
2>&1; then
    echo "Warning: Could not block public access on bucket"
fi

# Security: Enable versioning for data protection
echo "Enabling versioning on S3 bucket..."
if ! aws s3api put-bucket-versioning \
    --bucket "$S3_BUCKET" \
    --versioning-configuration Status=Enabled 2>&1; then
    echo "Warning: Could not enable versioning on bucket"
fi

    echo "S3 bucket created successfully: $S3_BUCKET"
fi

# Step 1: Create a database
echo "Step 1: Creating Athena database: $DATABASE_NAME"
CREATE_DB_RESULT=$(aws athena start-query-execution \
    --query-string "CREATE DATABASE IF NOT EXISTS $DATABASE_NAME" \
    --result-configuration "OutputLocation=s3://$S3_BUCKET/output/" \
    --region "$AWS_REGION" 2>&1)

if echo "$CREATE_DB_RESULT" | grep -qi "error\|failed"; then
    handle_error "Failed to create database: $CREATE_DB_RESULT"
fi

```

```
QUERY_ID=$(echo "$CREATE_DB_RESULT" | jq -r '.QueryExecutionId // empty' 2>/dev/
null || echo "$CREATE_DB_RESULT" | grep -o '"QueryExecutionId": "[^"]*' | cut -
d'"' -f4)
if [ -z "$QUERY_ID" ]; then
    handle_error "Failed to extract Query ID from database creation response"
fi
echo "Database creation query ID: $QUERY_ID"

# Wait for database creation to complete
echo "Waiting for database creation to complete..."
WAIT_TIMEOUT=60
ELAPSED=0
while [ $ELAPSED -lt $WAIT_TIMEOUT ]; do
    QUERY_STATUS=$(aws athena get-query-execution --query-execution-id
"$QUERY_ID" \
        --query "QueryExecution.Status.State" --output text --region
"$AWS_REGION" 2>&1)
    if [ "$QUERY_STATUS" = "SUCCEEDED" ]; then
        echo "Database creation completed successfully."
        break
    elif [ "$QUERY_STATUS" = "FAILED" ] || [ "$QUERY_STATUS" = "CANCELLED" ];
then
        handle_error "Database creation failed with status: $QUERY_STATUS"
    fi
    echo "Database creation in progress, status: $QUERY_STATUS"
    sleep 2
    ((ELAPSED+=2))
done

if [ $ELAPSED -ge $WAIT_TIMEOUT ]; then
    handle_error "Database creation timed out"
fi

# Verify the database was created
echo "Verifying database creation..."
LIST_DB_RESULT=$(aws athena list-databases --catalog-name AwsDataCatalog --region
"$AWS_REGION" 2>&1)
if echo "$LIST_DB_RESULT" | grep -qi "error\|failed"; then
    handle_error "Failed to list databases: $LIST_DB_RESULT"
fi
echo "$LIST_DB_RESULT"

# Step 2: Create a table
echo "Step 2: Creating Athena table: $TABLE_NAME"
```


[illegible]

```

while [ $ELAPSED -lt $WAIT_TIMEOUT ]; do
    QUERY_STATUS=$(aws athena get-query-execution --query-execution-id
"$QUERY_ID" \
        --query "QueryExecution.Status.State" --output text --region
"$AWS_REGION" 2>&1)
    if [ "$QUERY_STATUS" = "SUCCEEDED" ]; then
        echo "Table creation completed successfully."
        break
    elif [ "$QUERY_STATUS" = "FAILED" ] || [ "$QUERY_STATUS" = "CANCELLED" ];
then
        handle_error "Table creation failed with status: $QUERY_STATUS"
    fi
    echo "Table creation in progress, status: $QUERY_STATUS"
    sleep 2
    ((ELAPSED+=2))
done

if [ $ELAPSED -ge $WAIT_TIMEOUT ]; then
    handle_error "Table creation timed out"
fi

# Verify the table was created
echo "Verifying table creation..."
LIST_TABLE_RESULT=$(aws athena list-table-metadata \
    --catalog-name AwsDataCatalog \
    --database-name "$DATABASE_NAME" \
    --region "$AWS_REGION" 2>&1)
if echo "$LIST_TABLE_RESULT" | grep -qi "error\|failed"; then
    handle_error "Failed to list tables: $LIST_TABLE_RESULT"
fi
echo "$LIST_TABLE_RESULT"

# Step 3: Query data
echo "Step 3: Running a query on the table..."
QUERY="SELECT os, COUNT(*) count
FROM $DATABASE_NAME.$TABLE_NAME
WHERE date BETWEEN date '2014-07-05' AND date '2014-08-05'
GROUP BY os"

QUERY_RESULT=$(aws athena start-query-execution \
    --query-string "$QUERY" \
    --result-configuration "OutputLocation=s3://$S3_BUCKET/output/" \
    --region "$AWS_REGION" 2>&1)

```

```
if echo "$QUERY_RESULT" | grep -qi "error\|failed"; then
    handle_error "Failed to run query: $QUERY_RESULT"
fi

QUERY_ID=$(echo "$QUERY_RESULT" | jq -r '.QueryExecutionId // empty' 2>/dev/null
|| echo "$QUERY_RESULT" | grep -o '"QueryExecutionId": "[^"]*' | cut -d'"' -f4)
if [ -z "$QUERY_ID" ]; then
    handle_error "Failed to extract Query ID from query execution response"
fi
echo "Query execution ID: $QUERY_ID"

# Wait for query to complete
echo "Waiting for query to complete..."
ELAPSED=0
while [ $ELAPSED -lt $WAIT_TIMEOUT ]; do
    QUERY_STATUS=$(aws athena get-query-execution --query-execution-id
"$QUERY_ID" \
    --query "QueryExecution.Status.State" --output text --region
"$AWS_REGION" 2>&1)
    if [ "$QUERY_STATUS" = "SUCCEEDED" ]; then
        echo "Query completed successfully."
        break
    elif [ "$QUERY_STATUS" = "FAILED" ] || [ "$QUERY_STATUS" = "CANCELLED" ];
then
        handle_error "Query failed with status: $QUERY_STATUS"
    fi
    echo "Query in progress, status: $QUERY_STATUS"
    sleep 2
    ((ELAPSED+=2))
done

if [ $ELAPSED -ge $WAIT_TIMEOUT ]; then
    handle_error "Query execution timed out"
fi

# Get query results
echo "Getting query results..."
RESULTS=$(aws athena get-query-results --query-execution-id "$QUERY_ID" --region
"$AWS_REGION" 2>&1)
if echo "$RESULTS" | grep -qi "error\|failed"; then
    handle_error "Failed to get query results: $RESULTS"
fi
echo "$RESULTS"
```

```

# Download results from S3
echo "Downloading query results from S3..."
S3_PATH=$(aws athena get-query-execution --query-execution-id "$QUERY_ID" \
    --query "QueryExecution.ResultConfiguration.OutputLocation" --output text \
    --region "$AWS_REGION" 2>&1)
if echo "$S3_PATH" | grep -qi "error\|failed"; then
    handle_error "Failed to get S3 path for results: $S3_PATH"
fi

if [ -z "$S3_PATH" ] || [ "$S3_PATH" = "None" ]; then
    handle_error "S3 path for query results is empty"
fi

DOWNLOAD_RESULT=$(aws s3 cp "$S3_PATH" "./query-results.csv" 2>&1)
if echo "$DOWNLOAD_RESULT" | grep -qi "error\|failed"; then
    handle_error "Failed to download query results: $DOWNLOAD_RESULT"
fi

# Security: Secure the downloaded file
chmod 600 "./query-results.csv"
echo "Query results downloaded to query-results.csv (permissions: 600)"

# Step 4: Create a named query
echo "Step 4: Creating a named query..."
NAMED_QUERY_RESULT=$(aws athena create-named-query \
    --name "OS Count Query" \
    --description "Count of operating systems in CloudFront logs" \
    --database "$DATABASE_NAME" \
    --query-string "$QUERY" \
    --region "$AWS_REGION" 2>&1)

if echo "$NAMED_QUERY_RESULT" | grep -qi "error\|failed"; then
    handle_error "Failed to create named query: $NAMED_QUERY_RESULT"
fi

NAMED_QUERY_ID=$(echo "$NAMED_QUERY_RESULT" | jq -r '.NamedQueryId // empty' 2>/
dev/null || echo "$NAMED_QUERY_RESULT" | grep -o '"NamedQueryId": "[^"]*"' | cut -
d'"' -f4)
if [ -z "$NAMED_QUERY_ID" ]; then
    handle_error "Failed to extract Named Query ID from response"
fi
echo "Named query created with ID: $NAMED_QUERY_ID"

# List named queries

```

```

echo "Listing named queries..."
LIST_QUERIES_RESULT=$(aws athena list-named-queries --region "$AWS_REGION" 2>&1)
if echo "$LIST_QUERIES_RESULT" | grep -qi "error\|failed"; then
    handle_error "Failed to list named queries: $LIST_QUERIES_RESULT"
fi
echo "$LIST_QUERIES_RESULT"

# Get the named query details
echo "Getting named query details..."
GET_QUERY_RESULT=$(aws athena get-named-query --named-query-id "$NAMED_QUERY_ID" \
    \
    --region "$AWS_REGION" 2>&1)
if echo "$GET_QUERY_RESULT" | grep -qi "error\|failed"; then
    handle_error "Failed to get named query: $GET_QUERY_RESULT"
fi
echo "$GET_QUERY_RESULT"

# Execute the named query
echo "Executing the named query..."
QUERY_STRING=$(aws athena get-named-query --named-query-id "$NAMED_QUERY_ID" \
    --query "NamedQuery.QueryString" --output text --region "$AWS_REGION" 2>&1)
if echo "$QUERY_STRING" | grep -qi "error\|failed"; then
    handle_error "Failed to get query string: $QUERY_STRING"
fi

if [ -z "$QUERY_STRING" ] || [ "$QUERY_STRING" = "None" ]; then
    handle_error "Query string is empty"
fi

EXEC_RESULT=$(aws athena start-query-execution \
    --query-string "$QUERY_STRING" \
    --result-configuration "OutputLocation=s3://$S3_BUCKET/output/" \
    --region "$AWS_REGION" 2>&1)

if echo "$EXEC_RESULT" | grep -qi "error\|failed"; then
    handle_error "Failed to execute named query: $EXEC_RESULT"
fi

QUERY_ID=$(echo "$EXEC_RESULT" | jq -r '.QueryExecutionId // empty' 2>/dev/null
    || echo "$EXEC_RESULT" | grep -o '"QueryExecutionId": "[^"]*' | cut -d'"' -f4)
if [ -z "$QUERY_ID" ]; then
    handle_error "Failed to extract Query ID from named query execution response"
fi
echo "Named query execution ID: $QUERY_ID"

```

```

# Wait for named query to complete
echo "Waiting for named query execution to complete..."
ELAPSED=0
while [ $ELAPSED -lt $WAIT_TIMEOUT ]; do
    QUERY_STATUS=$(aws athena get-query-execution --query-execution-id
"$QUERY_ID" \
    --query "QueryExecution.Status.State" --output text --region
"$AWS_REGION" 2>&1)
    if [ "$QUERY_STATUS" = "SUCCEEDED" ]; then
        echo "Named query execution completed successfully."
        break
    elif [ "$QUERY_STATUS" = "FAILED" ] || [ "$QUERY_STATUS" = "CANCELLED" ];
then
        handle_error "Named query execution failed with status: $QUERY_STATUS"
    fi
    echo "Named query execution in progress, status: $QUERY_STATUS"
    sleep 2
    ((ELAPSED+=2))
done

if [ $ELAPSED -ge $WAIT_TIMEOUT ]; then
    handle_error "Named query execution timed out"
fi

# Summary of resources created
echo ""
echo "=====
echo "RESOURCES CREATED"
echo "=====
echo "- S3 Bucket: $S3_BUCKET"
echo "- Database: $DATABASE_NAME"
echo "- Table: $TABLE_NAME"
echo "- Named Query: $NAMED_QUERY_ID"
echo "- Query results saved to: query-results.csv"
echo "=====

# Auto-confirm cleanup
echo ""
echo "=====
echo "CLEANUP CONFIRMATION"
echo "=====
echo "Starting cleanup..."
CLEANUP_CHOICE="y"

```

```

if [ [ "$CLEANUP_CHOICE" =~ ^[Yy]$ ] ]; then
    echo "Starting cleanup..."

    # Delete named query
    echo "Deleting named query: $NAMED_QUERY_ID"
    DELETE_QUERY_RESULT=$(aws athena delete-named-query --named-query-id
"$NAMED_QUERY_ID" \
    --region "$AWS_REGION" 2>&1)
    if echo "$DELETE_QUERY_RESULT" | grep -qi "error\|failed"; then
        echo "Warning: Failed to delete named query: $DELETE_QUERY_RESULT"
    else
        echo "Named query deleted successfully."
    fi

    # Drop table
    echo "Dropping table: $TABLE_NAME"
    DROP_TABLE_RESULT=$(aws athena start-query-execution \
    --query-string "DROP TABLE IF EXISTS $DATABASE_NAME.$TABLE_NAME" \
    --result-configuration "OutputLocation=s3://$S3_BUCKET/output/" \
    --region "$AWS_REGION" 2>&1)

    if echo "$DROP_TABLE_RESULT" | grep -qi "error\|failed"; then
        echo "Warning: Failed to drop table: $DROP_TABLE_RESULT"
    else
        QUERY_ID=$(echo "$DROP_TABLE_RESULT" | jq -r '.QueryExecutionId // empty'
2>/dev/null || echo "$DROP_TABLE_RESULT" | grep -o '"QueryExecutionId": "[^"]*"'
| cut -d'"' -f4)
        if [ -n "$QUERY_ID" ]; then
            echo "Waiting for table deletion to complete..."

            ELAPSED=0
            while [ $ELAPSED -lt $WAIT_TIMEOUT ]; do
                QUERY_STATUS=$(aws athena get-query-execution --query-execution-
id "$QUERY_ID" \
                --query "QueryExecution.Status.State" --output text --region
"$AWS_REGION" 2>&1)
                if [ "$QUERY_STATUS" = "SUCCEEDED" ]; then
                    echo "Table dropped successfully."
                    break
                elif [ "$QUERY_STATUS" = "FAILED" ] || [ "$QUERY_STATUS" =
"CANCELLED" ]; then
                    echo "Warning: Table deletion failed with status:
$QUERY_STATUS"

```

```

        break
    fi
    echo "Table deletion in progress, status: $QUERY_STATUS"
    sleep 2
    ((ELAPSED+=2))
done
fi

# Drop database
echo "Dropping database: $DATABASE_NAME"
DROP_DB_RESULT=$(aws athena start-query-execution \
    --query-string "DROP DATABASE IF EXISTS $DATABASE_NAME" \
    --result-configuration "OutputLocation=s3://$S3_BUCKET/output/" \
    --region "$AWS_REGION" 2>&1)

if echo "$DROP_DB_RESULT" | grep -qi "error\|failed"; then
    echo "Warning: Failed to drop database: $DROP_DB_RESULT"
else
    QUERY_ID=$(echo "$DROP_DB_RESULT" | jq -r '.QueryExecutionId // empty'
2>/dev/null || echo "$DROP_DB_RESULT" | grep -o '"QueryExecutionId": "[^"]*"' |
cut -d'"' -f4)
    if [ -n "$QUERY_ID" ]; then
        echo "Waiting for database deletion to complete..."

        ELAPSED=0
        while [ $ELAPSED -lt $WAIT_TIMEOUT ]; do
            QUERY_STATUS=$(aws athena get-query-execution --query-execution-
id "$QUERY_ID" \
                --query "QueryExecution.Status.State" --output text --region
"$AWS_REGION" 2>&1)
            if [ "$QUERY_STATUS" = "SUCCEEDED" ]; then
                echo "Database dropped successfully."
                break
            elif [ "$QUERY_STATUS" = "FAILED" ] || [ "$QUERY_STATUS" =
"CANCELLED" ]; then
                echo "Warning: Database deletion failed with status:
$QUERY_STATUS"
                break
            fi
            echo "Database deletion in progress, status: $QUERY_STATUS"
            sleep 2
            ((ELAPSED+=2))
        done
    fi
fi

```



```

        fi
    fi

    # Empty and delete S3 bucket (only if not shared)
    if [ "$BUCKET_IS_SHARED" = false ]; then
        echo "Emptying S3 bucket: $S3_BUCKET"
        EMPTY_BUCKET_RESULT=$(aws s3 rm "s3://$S3_BUCKET" --recursive 2>&1)
        if echo "$EMPTY_BUCKET_RESULT" | grep -qi "error\|failed"; then
            echo "Warning: Failed to empty S3 bucket: $EMPTY_BUCKET_RESULT"
        else
            echo "S3 bucket emptied successfully."
        fi

        echo "Deleting S3 bucket: $S3_BUCKET"
        DELETE_BUCKET_RESULT=$(aws s3 rb "s3://$S3_BUCKET" 2>&1)
        if echo "$DELETE_BUCKET_RESULT" | grep -qi "error\|failed"; then
            echo "Warning: Failed to delete S3 bucket: $DELETE_BUCKET_RESULT"
        else
            echo "S3 bucket deleted successfully."
        fi
    else
        echo "Skipping S3 bucket deletion (shared resource)"
    fi

    # Security: Remove downloaded query results
    if [ -f "./query-results.csv" ]; then
        if command -v shred &>/dev/null; then
            shred -vfz -n 3 "./query-results.csv" 2>/dev/null || rm -f "./query-
results.csv"
        else
            rm -f "./query-results.csv"
        fi
        echo "Query results file securely removed."
    fi

    echo "Cleanup completed."
fi

echo "Tutorial completed successfully!"

```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
- [Cp](#)
- [CreateNamedQuery](#)

- [DeleteNamedQuery](#)
- [GetNamedQuery](#)
- [GetQueryExecution](#)
- [GetQueryResults](#)
- [ListDatabases](#)
- [ListNamedQueries](#)
- [ListTableMetadata](#)
- [Mb](#)
- [Rb](#)
- [Rm](#)
- [StartQueryExecution](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 使用 Amazon S3 物件鎖定功能

下列程式碼範例示範如何使用 S3 物件鎖定功能。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

執行展示 Amazon S3 物件鎖定功能的互動式情境。

```
using Amazon.S3;  
using Amazon.S3.Model;  
using Microsoft.Extensions.Configuration;  
using Microsoft.Extensions.DependencyInjection;  
using Microsoft.Extensions.Hosting;
```

```
using Microsoft.Extensions.Logging;
using Microsoft.Extensions.Logging.Console;
using Microsoft.Extensions.Logging.Debug;

namespace S3ObjectLockScenario;

public static class S3ObjectLockWorkflow
{
    /*
        Before running this .NET code example, set up your development environment,
        including your credentials.

        This .NET example performs the following tasks:
        1. Create test Amazon Simple Storage Service (S3) buckets with different
        lock policies.
        2. Upload sample objects to each bucket.
        3. Set some Legal Hold and Retention Periods on objects and buckets.
        4. Investigate lock policies by viewing settings or attempting to delete
        or overwrite objects.
        5. Clean up objects and buckets.
    */

    public static S3ActionsWrapper _s3ActionsWrapper = null!;
    public static IConfiguration _configuration = null!;
    private static string _resourcePrefix = null!;
    private static string noLockBucketName = null!;
    private static string lockEnabledBucketName = null!;
    private static string retentionAfterCreationBucketName = null!;
    private static List<string> bucketNames = new List<string>();
    private static List<string> fileNames = new List<string>();

    public static async Task Main(string[] args)
    {
        // Set up dependency injection for the Amazon service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
                        LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
                        LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonS3>()
                    .AddTransient<S3ActionsWrapper>())
```

```
        )
        .Build();

    _configuration = new ConfigurationBuilder()
        .SetBasePath(Directory.GetCurrentDirectory())
        .AddJsonFile("settings.json") // Load settings from .json file.
        .AddJsonFile("settings.local.json",
            true) // Optionally, load local settings.
        .Build();

    ConfigurationSetup();

    ServicesSetup(host);

    try
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Welcome to the Amazon Simple Storage Service (S3)
Object Locking Feature Scenario.");
        Console.WriteLine(new string('-', 80));
        await Setup(true);

        await DemoActionChoices();

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Cleaning up resources.");
        Console.WriteLine(new string('-', 80));
        await Cleanup(true);

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Amazon S3 Object Locking Scenario is complete.");
        Console.WriteLine(new string('-', 80));
    }
    catch (Exception ex)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine($"There was a problem: {ex.Message}");
        await Cleanup(true);
        Console.WriteLine(new string('-', 80));
    }
}

/// <summary>
/// Populate the services for use within the console application.
```

```
/// </summary>
/// <param name="host">The services host.</param>
private static void ServicesSetup(IHost host)
{
    _s3ActionsWrapper = host.Services.GetRequiredService<S3ActionsWrapper>();
}

/// <summary>
/// Any setup operations needed.
/// </summary>
public static void ConfigurationSetup()
{
    _resourcePrefix = _configuration["resourcePrefix"] ?? "dotnet-example";

    noLockBucketName = _resourcePrefix + "-no-lock";
    lockEnabledBucketName = _resourcePrefix + "-lock-enabled";
    retentionAfterCreationBucketName = _resourcePrefix + "-retention-after-creation";

    bucketNames.Add(noLockBucketName);
    bucketNames.Add(lockEnabledBucketName);
    bucketNames.Add(retentionAfterCreationBucketName);
}

// <summary>
/// Deploy necessary resources for the scenario.
/// </summary>
/// <param name="interactive">True to run as interactive.</param>
/// <returns>True if successful.</returns>
public static async Task<bool> Setup(bool interactive)
{
    Console.WriteLine(
        "\nFor this scenario, we will use the AWS SDK for .NET to create
several S3\n" +
        "buckets and files to demonstrate working with S3 locking features.
\n");

    Console.WriteLine(new string('-', 80));
    Console.WriteLine("Press Enter when you are ready to start.");
    if (interactive)
        Console.ReadLine();

    Console.WriteLine("\nS3 buckets can be created either with or without
object lock enabled.");
```

```
        await _s3ActionsWrapper.CreateBucketWithObjectLock(noLockBucketName,
false);
        await _s3ActionsWrapper.CreateBucketWithObjectLock(lockEnabledBucketName,
true);
        await
_s3ActionsWrapper.CreateBucketWithObjectLock(retentionAfterCreationBucketName,
false);

        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("\nA bucket can be configured to use object locking
with a default retention period.");
        await
_s3ActionsWrapper.ModifyBucketDefaultRetention(retentionAfterCreationBucketName,
true,
            ObjectLockRetentionMode.Governance, DateTime.UtcNow.AddDays(1));

        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("\nObject lock policies can also be added to existing
buckets.");
        await _s3ActionsWrapper.EnableObjectLockOnBucket(lockEnabledBucketName);

        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        // Upload some files to the buckets.
        Console.WriteLine("\nNow let's add some test files:");
        var fileName = _configuration["exampleFileName"] ?? "exampleFile.txt";
        int fileCount = 2;
        // Create the file if it does not already exist.
        if (!File.Exists(fileName))
        {
            await using StreamWriter sw = File.CreateText(fileName);
            await sw.WriteLineAsync(
                "This is a sample file for uploading to a bucket.");
        }

        foreach (var bucketName in bucketNames)
```

```

        {
            for (int i = 0; i < fileCount; i++)
            {
                var numberedFileName = Path.GetFileNameWithoutExtension(fileName)
+ i + Path.GetExtension(fileName);
                fileNames.Add(numberedFileName);
                await _s3ActionsWrapper.UploadFileAsync(bucketName,
numberedFileName, fileName);
            }
        }
        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        if (!interactive)
            return true;
        Console.WriteLine("\nNow we can set some object lock policies on
individual files:");
        foreach (var bucketName in bucketNames)
        {
            for (int i = 0; i < fileNames.Count; i++)
            {
                // No modifications to the objects in the first bucket.
                if (bucketName != bucketNames[0])
                {
                    var exampleFileName = fileNames[i];
                    switch (i)
                    {
                        case 0:
                            {
                                var question =
                                    $"Would you like to add a legal hold to
{exampleFileName} in {bucketName}? (y/n)";
                                if (GetYesNoResponse(question))
                                {
                                    // Set a legal hold.
                                    await
_s3ActionsWrapper.ModifyObjectLegalHold(bucketName, exampleFileName,
ObjectLockLegalHoldStatus.On);

                                }
                                break;
                            }
                        case 1:

```

```

        {
            var question =
                $"\\nWould you like to add a 1 day Governance
retention period to {exampleFileName} in {bucketName}? (y/n)" +
                "\\nReminder: Only a user with the
s3:BypassGovernanceRetention permission will be able to delete this file or its
bucket until the retention period has expired.";
            if (GetYesNoResponse(question))
            {
                // Set a Governance mode retention period for
1 day.

                await
                _s3ActionsWrapper.ModifyObjectRetentionPeriod(
                    bucketName, exampleFileName,
                    ObjectLockRetentionMode.Governance,
                    DateTime.UtcNow.AddDays(1));
            }
            break;
        }
    }
}

}
Console.WriteLine(new string('-', 80));
return true;
}

// <summary>
/// List all of the current buckets and objects.
/// </summary>
/// <param name="interactive">True to run as interactive.</param>
/// <returns>The list of buckets and objects.</returns>
public static async Task<List<S3ObjectVersion>> ListBucketsAndObjects(bool
interactive)
{
    var allObjects = new List<S3ObjectVersion>();
    foreach (var bucketName in bucketNames)
    {
        var objectsInBucket = await
        _s3ActionsWrapper.ListBucketObjectsAndVersions(bucketName);
        foreach (var objectKey in objectsInBucket.Versions)
        {
            allObjects.Add(objectKey);
        }
    }
}

```



```

    }

    if (interactive)
    {
        Console.WriteLine("\nCurrent buckets and objects:\n");
        int i = 0;
        foreach (var bucketObject in allObjects)
        {
            i++;
            Console.WriteLine(
                $"{i}: {bucketObject.Key} \n\tBucket:
{bucketObject.BucketName}\n\tVersion: {bucketObject.VersionId}");
        }
    }

    return allObjects;
}

/// <summary>
/// Present the user with the demo action choices.
/// </summary>
/// <returns>Async task.</returns>
public static async Task<bool> DemoActionChoices()
{
    var choices = new string[]{
        "List all files in buckets.",
        "Attempt to delete a file.",
        "Attempt to delete a file with retention period bypass.",
        "Attempt to overwrite a file.",
        "View the object and bucket retention settings for a file.",
        "View the legal hold settings for a file.",
        "Finish the scenario."};

    var choice = 0;
    // Keep asking the user until they choose to move on.
    while (choice != 6)
    {
        Console.WriteLine(new string('-', 80));
        choice = GetChoiceResponse(
            "\nExplore the S3 locking features by selecting one of the
following choices:"
            , choices);
        Console.WriteLine(new string('-', 80));
        switch (choice)

```

```
{
    case 0:
    {
        await ListBucketsAndObjects(true);
        break;
    }
    case 1:
    {
        Console.WriteLine("\nEnter the number of the object to
delete:");

        var allFiles = await ListBucketsAndObjects(true);
        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
        await
_s3ActionsWrapper.DeleteObjectFromBucket(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key, false, allFiles[fileChoice].VersionId);
        break;
    }
    case 2:
    {
        Console.WriteLine("\nEnter the number of the object to
delete:");

        var allFiles = await ListBucketsAndObjects(true);
        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
        await
_s3ActionsWrapper.DeleteObjectFromBucket(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key, true, allFiles[fileChoice].VersionId);
        break;
    }
    case 3:
    {
        var allFiles = await ListBucketsAndObjects(true);
        Console.WriteLine("\nEnter the number of the object to
overwrite:");

        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
        // Create the file if it does not already exist.
        if (!File.Exists(allFiles[fileChoice].Key))
        {
            await using StreamWriter sw =
File.CreateText(allFiles[fileChoice].Key);
            await sw.WriteLineAsync(
```

```

        "This is a sample file for uploading to a
bucket.");
    }
    await
_s3ActionsWrapper.UploadFileAsync(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key, allFiles[fileChoice].Key);
    break;
}
case 4:
{
    var allFiles = await ListBucketsAndObjects(true);
    Console.WriteLine("\nEnter the number of the object and
bucket to view:");
    var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
    await
_s3ActionsWrapper.GetObjectRetention(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key);
    await
_s3ActionsWrapper.GetBucketObjectLockConfiguration(allFiles[fileChoice].BucketName);
    break;
}
case 5:
{
    var allFiles = await ListBucketsAndObjects(true);
    Console.WriteLine("\nEnter the number of the object to
view:");
    var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
    await
_s3ActionsWrapper.GetObjectLegalHold(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key);
    break;
}
}
}
return true;
}

// <summary>
/// Clean up the resources from the scenario.
/// </summary>
/// <param name="interactive">True to run as interactive.</param>
/// <returns>True if successful.</returns>

```

```
public static async Task<bool> Cleanup(bool interactive)
{
    Console.WriteLine(new string('-', 80));

    if (!interactive || GetYesNoResponse("Do you want to clean up all files
and buckets? (y/n) "))
    {
        // Remove all locks and delete all buckets and objects.
        var allFiles = await ListBucketsAndObjects(false);
        foreach (var fileInfo in allFiles)
        {
            // Check for a legal hold.
            var legalHold = await
_s3ActionsWrapper.GetObjectLegalHold(fileInfo.BucketName, fileInfo.Key);
            if (legalHold?.Status?.Value == ObjectLockLegalHoldStatus.On)
            {
                await
_s3ActionsWrapper.ModifyObjectLegalHold(fileInfo.BucketName, fileInfo.Key,
ObjectLockLegalHoldStatus.Off);
            }

            // Check for a retention period.
            var retention = await
_s3ActionsWrapper.GetObjectRetention(fileInfo.BucketName, fileInfo.Key);
            var hasRetentionPeriod = retention?.Mode ==
ObjectLockRetentionMode.Governance && retention.RetainUntilDate >
DateTime.UtcNow.Date;
            await
_s3ActionsWrapper.DeleteObjectFromBucket(fileInfo.BucketName, fileInfo.Key,
hasRetentionPeriod, fileInfo.VersionId);
        }

        foreach (var bucketName in bucketNames)
        {
            await _s3ActionsWrapper.DeleteBucketByName(bucketName);
        }
    }
    else
    {
        Console.WriteLine(
            "Ok, we'll leave the resources intact.\n" +
            "Don't forget to delete them when you're done with them or you
might incur unexpected charges."
        );
    }
}
```

```

        );
    }

    Console.WriteLine(new string('-', 80));
    return true;
}

/// <summary>
/// Helper method to get a yes or no response from the user.
/// </summary>
/// <param name="question">The question string to print on the console.</
param>
/// <returns>True if the user responds with a yes.</returns>
private static bool GetYesNoResponse(string question)
{
    Console.WriteLine(question);
    var ynResponse = Console.ReadLine();
    var response = ynResponse != null && ynResponse.Equals("y",
StringComparison.InvariantCultureIgnoreCase);
    return response;
}

/// <summary>
/// Helper method to get a choice response from the user.
/// </summary>
/// <param name="question">The question string to print on the console.</
param>
/// <param name="choices">The choices to print on the console.</param>
/// <returns>The index of the selected choice</returns>
private static int GetChoiceResponse(string? question, string[] choices)
{
    if (question != null)
    {
        Console.WriteLine(question);

        for (int i = 0; i < choices.Length; i++)
        {
            Console.WriteLine($"{i + 1}. {choices[i]}");
        }
    }

    var choiceNumber = 0;
    while (choiceNumber < 1 || choiceNumber > choices.Length)
    {

```

```
        var choice = Console.ReadLine();
        Int32.TryParse(choice, out choiceNumber);
    }

    return choiceNumber - 1;
}
}
```

S3 函數的包裝函式類別。

```
using System.Net;
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Configuration;

namespace S3ObjectLockScenario;

/// <summary>
/// Encapsulate the Amazon S3 operations.
/// </summary>
public class S3ActionsWrapper
{
    private readonly IAmazonS3 _amazonS3;

    /// <summary>
    /// Constructor for the S3ActionsWrapper.
    /// </summary>
    /// <param name="amazonS3">The injected S3 client.</param>
    public S3ActionsWrapper(IAmazonS3 amazonS3, IConfiguration configuration)
    {
        _amazonS3 = amazonS3;
    }

    /// <summary>
    /// Create a new Amazon S3 bucket with object lock actions.
    /// </summary>
    /// <param name="bucketName">The name of the bucket to create.</param>
    /// <param name="enableObjectLock">True to enable object lock on the
    bucket.</param>
    /// <returns>True if successful.</returns>
```

```
public async Task<bool> CreateBucketWithObjectLock(string bucketName, bool
enableObjectLock)
{
    Console.WriteLine($"\\tCreating bucket {bucketName} with object lock
{enableObjectLock}.");
    try
    {
        var request = new PutBucketRequest
        {
            BucketName = bucketName,
            UseClientRegion = true,
            ObjectLockEnabledForBucket = enableObjectLock,
        };

        var response = await _amazonS3.PutBucketAsync(request);

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error creating bucket: '{ex.Message}'");
        return false;
    }
}

/// <summary>
/// Enable object lock on an existing bucket.
/// </summary>
/// <param name="bucketName">The name of the bucket to modify.</param>
/// <returns>True if successful.</returns>
public async Task<bool> EnableObjectLockOnBucket(string bucketName)
{
    try
    {
        // First, enable Versioning on the bucket.
        await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
        {
            BucketName = bucketName,
            VersioningConfig = new S3BucketVersioningConfig()
            {
                EnableMfaDelete = false,
                Status = VersionStatus.Enabled
            }
        });
    }
}
```

```

    });

    var request = new PutObjectLockConfigurationRequest()
    {
        BucketName = bucketName,
        ObjectLockConfiguration = new ObjectLockConfiguration()
        {
            ObjectLockEnabled = new ObjectLockEnabled("Enabled"),
        },
    };

    var response = await
        _amazonS3.PutObjectLockConfigurationAsync(request);
    Console.WriteLine($"\\tAdded an object lock policy to bucket
{bucketName}.");
    return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}
catch (AmazonS3Exception ex)
{
    Console.WriteLine($"Error modifying object lock: '{ex.Message}'");
    return false;
}
}

/// <summary>
/// Set or modify a retention period on an object in an S3 bucket.
/// </summary>
/// <param name="bucketName">The bucket of the object.</param>
/// <param name="objectKey">The key of the object.</param>
/// <param name="retention">The retention mode.</param>
/// <param name="retainUntilDate">The date retention expires.</param>
/// <returns>True if successful.</returns>
public async Task<bool> ModifyObjectRetentionPeriod(string bucketName,
    string objectKey, ObjectLockRetentionMode retention, DateTime
retainUntilDate)
{
    try
    {
        var request = new PutObjectRetentionRequest()
        {
            BucketName = bucketName,
            Key = objectKey,
            Retention = new ObjectLockRetention()
            {

```



```

        Mode = retention,
        RetainUntilDate = retainUntilDate
    }
};

var response = await _amazonS3.PutObjectRetentionAsync(request);
Console.WriteLine($"\\tSet retention for {objectKey} in {bucketName}
until {retainUntilDate:d}.");
return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}
catch (AmazonS3Exception ex)
{
    Console.WriteLine($"\\tError modifying retention period:
'{ex.Message}'");
    return false;
}
}

/// <summary>
/// Set or modify a retention period on an S3 bucket.
/// </summary>
/// <param name="bucketName">The bucket to modify.</param>
/// <param name="retention">The retention mode.</param>
/// <param name="retainUntilDate">The date for retention until.</param>
/// <returns>True if successful.</returns>
public async Task<bool> ModifyBucketDefaultRetention(string bucketName, bool
enableObjectLock, ObjectLockRetentionMode retention, DateTime retainUntilDate)
{
    var enabledString = enableObjectLock ? "Enabled" : "Disabled";
    var timeDifference = retainUntilDate.Subtract(DateTime.Now);
    try
    {
        // First, enable Versioning on the bucket.
        await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
        {
            BucketName = bucketName,
            VersioningConfig = new S3BucketVersioningConfig()
            {
                EnableMfaDelete = false,
                Status = VersionStatus.Enabled
            }
        });
    }
}

```

```

        var request = new PutObjectLockConfigurationRequest()
        {
            BucketName = bucketName,
            ObjectLockConfiguration = new ObjectLockConfiguration()
            {
                ObjectLockEnabled = new ObjectLockEnabled(enabledString),
                Rule = new ObjectLockRule()
                {
                    DefaultRetention = new DefaultRetention()
                    {
                        Mode = retention,
                        Days = timeDifference.Days // Can be specified in
days or years but not both.
                    }
                }
            }
        };

        var response = await
_amazonS3.PutObjectLockConfigurationAsync(request);
        Console.WriteLine($"\\tAdded a default retention to bucket
{bucketName}.");
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"\\tError modifying object lock: '{ex.Message}'");
        return false;
    }
}

/// <summary>
/// Get the retention period for an S3 object.
/// </summary>
/// <param name="bucketName">The bucket of the object.</param>
/// <param name="objectKey">The object key.</param>
/// <returns>The object retention details.</returns>
public async Task<ObjectLockRetention> GetObjectRetention(string bucketName,
    string objectKey)
{
    try
    {
        var request = new GetObjectRetentionRequest()
        {

```

```

        BucketName = bucketName,
        Key = objectKey
    };

    var response = await _amazonS3.GetObjectRetentionAsync(request);
    Console.WriteLine($"\\tObject retention for {objectKey} in
{bucketName}: " +
                    $"\\n\\t{response.Retention.Mode} until
{response.Retention.RetainUntilDate:d}.");
    return response.Retention;
}
catch (AmazonS3Exception ex)
{
    Console.WriteLine($"\\tUnable to fetch object lock retention:
'{ex.Message}'");
    return new ObjectLockRetention();
}
}

/// <summary>
/// Set or modify a legal hold on an object in an S3 bucket.
/// </summary>
/// <param name="bucketName">The bucket of the object.</param>
/// <param name="objectKey">The key of the object.</param>
/// <param name="holdStatus">The On or Off status for the legal hold.</param>
/// <returns>True if successful.</returns>
public async Task<bool> ModifyObjectLegalHold(string bucketName,
    string objectKey, ObjectLockLegalHoldStatus holdStatus)
{
    try
    {
        var request = new PutObjectLegalHoldRequest()
        {
            BucketName = bucketName,
            Key = objectKey,
            LegalHold = new ObjectLockLegalHold()
            {
                Status = holdStatus
            }
        };
    };

    var response = await _amazonS3.PutObjectLegalHoldAsync(request);
    Console.WriteLine($"\\tModified legal hold for {objectKey} in
{bucketName}.");
}

```

```

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"\\tError modifying legal hold: '{ex.Message}'");
        return false;
    }
}

/// <summary>
/// Get the legal hold details for an S3 object.
/// </summary>
/// <param name="bucketName">The bucket of the object.</param>
/// <param name="objectKey">The object key.</param>
/// <returns>The object legal hold details.</returns>
public async Task<ObjectLockLegalHold> GetObjectLegalHold(string bucketName,
    string objectKey)
{
    try
    {
        var request = new GetObjectLegalHoldRequest()
        {
            BucketName = bucketName,
            Key = objectKey
        };

        var response = await _amazonS3.GetObjectLegalHoldAsync(request);
        Console.WriteLine($"\\tObject legal hold for {objectKey} in
{bucketName}: " +
            $"\\n\\tStatus: {response.LegalHold.Status}");
        return response.LegalHold;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"\\tUnable to fetch legal hold: '{ex.Message}'");
        return new ObjectLockLegalHold();
    }
}

/// <summary>
/// Get the object lock configuration details for an S3 bucket.
/// </summary>
/// <param name="bucketName">The bucket to get details.</param>
/// <returns>The bucket's object lock configuration details.</returns>

```

```

    public async Task<ObjectLockConfiguration>
GetBucketObjectLockConfiguration(string bucketName)
    {
        try
        {
            var request = new GetObjectLockConfigurationRequest()
            {
                BucketName = bucketName
            };

            var response = await
_amazonS3.GetObjectLockConfigurationAsync(request);
            Console.WriteLine($"\\tBucket object lock config for {bucketName} in
{bucketName}: " +
                $"\\n\\tEnabled:
{response.ObjectLockConfiguration.ObjectLockEnabled}" +
                $"\\n\\tRule:
{response.ObjectLockConfiguration.Rule?.DefaultRetention}");

            return response.ObjectLockConfiguration;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\\tUnable to fetch object lock config:
'{ex.Message}'");
            return new ObjectLockConfiguration();
        }
    }

    /// <summary>
    /// Upload a file from the local computer to an Amazon S3 bucket.
    /// </summary>
    /// <param name="bucketName">The Amazon S3 bucket to use.</param>
    /// <param name="objectName">The object to upload.</param>
    /// <param name="filePath">The path, including file name, of the object to
upload.</param>
    /// <returns>True if success.</returns>
    public async Task<bool> UploadFileAsync(string bucketName, string objectName,
string filePath)
    {
        var request = new PutObjectRequest
        {
            BucketName = bucketName,
            Key = objectName,

```

```

        FilePath = filePath,
        ChecksumAlgorithm = ChecksumAlgorithm.SHA256
    };

    var response = await _amazonS3.PutObjectAsync(request);
    if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
    {
        Console.WriteLine($"{bucketName}\tSuccessfully uploaded {objectName} to
{bucketName}.");
        return true;
    }
    else
    {
        Console.WriteLine($"{bucketName}\tCould not upload {objectName} to
{bucketName}.");
        return false;
    }
}

/// <summary>
/// List bucket objects and versions.
/// </summary>
/// <param name="bucketName">The Amazon S3 bucket to use.</param>
/// <returns>The list of objects and versions.</returns>
public async Task<ListVersionsResponse> ListBucketObjectsAndVersions(string
bucketName)
{
    var request = new ListVersionsRequest()
    {
        BucketName = bucketName
    };

    var response = await _amazonS3.ListVersionsAsync(request);
    return response;
}

/// <summary>
/// Delete an object from a specific bucket.
/// </summary>
/// <param name="bucketName">The Amazon S3 bucket to use.</param>
/// <param name="objectKey">The key of the object to delete.</param>
/// <param name="hasRetention">True if the object has retention settings.</
param>
/// <param name="versionId">Optional versionId.</param>

```

```

    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteObjectFromBucket(string bucketName, string
objectKey, bool hasRetention, string? versionId = null)
    {
        try
        {
            var request = new DeleteObjectRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                VersionId = versionId,
            };
            if (hasRetention)
            {
                // Set the BypassGovernanceRetention header
                // if the file has retention settings.
                request.BypassGovernanceRetention = true;
            }
            await _amazonS3.DeleteObjectAsync(request);
            Console.WriteLine(
                $"Deleted {objectKey} in {bucketName}.");
            return true;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Unable to delete object {objectKey} in bucket
{bucketName}: " + ex.Message);
            return false;
        }
    }

    /// <summary>
    /// Delete a specific bucket.
    /// </summary>
    /// <param name="bucketName">The Amazon S3 bucket to use.</param>
    /// <param name="objectKey">The key of the object to delete.</param>
    /// <param name="versionId">Optional versionId.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteBucketByName(string bucketName)
    {
        try
        {
            var request = new DeleteBucketRequest() { BucketName = bucketName, };
            var response = await _amazonS3.DeleteBucketAsync(request);

```

```
        Console.WriteLine($"\\tDelete for {bucketName} complete.");
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"\\tUnable to delete bucket {bucketName}: " +
            ex.Message);
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的下列主題。
 - [GetObjectLegalHold](#)
 - [GetObjectLockConfiguration](#)
 - [GetObjectRetention](#)
 - [PutObjectLegalHold](#)
 - [PutObjectLockConfiguration](#)
 - [PutObjectRetention](#)

Go

SDK for Go V2

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

執行展示 Amazon S3 物件鎖定功能的互動式情境。

```
import (
    "context"
    "fmt"
```



```

"log"
"strings"

"s3_object_lock/actions"

"github.com/aws/aws-sdk-go-v2/aws"
"github.com/aws/aws-sdk-go-v2/feature/s3/manager"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/aws/aws-sdk-go-v2/service/s3/types"
"github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// ObjectLockScenario contains the steps to run the S3 Object Lock workflow.
type ObjectLockScenario struct {
    questioner demotools.IQuestioner
    resources  Resources
    s3Actions  *actions.S3Actions
    sdkConfig  aws.Config
}

// NewObjectLockScenario constructs a new ObjectLockScenario instance.
func NewObjectLockScenario(sdkConfig aws.Config, questioner
    demotools.IQuestioner) ObjectLockScenario {
    scenario := ObjectLockScenario{
        questioner: questioner,
        resources:  Resources{},
        s3Actions:  &actions.S3Actions{S3Client: s3.NewFromConfig(sdkConfig)},
        sdkConfig:  sdkConfig,
    }
    scenario.s3Actions.S3Manager = manager.NewUploader(scenario.s3Actions.S3Client)
    scenario.resources.init(scenario.s3Actions, questioner)
    return scenario
}

type nameLocked struct {
    name  string
    locked bool
}

var createInfo = []nameLocked{
    {"standard-bucket", false},
    {"lock-bucket", true},
    {"retention-bucket", false},
}

```

```
// CreateBuckets creates the S3 buckets required for the workflow.
func (scenario *ObjectLockScenario) CreateBuckets(ctx context.Context) {
    log.Println("Let's create some S3 buckets to use for this workflow.")
    success := false
    for !success {
        prefix := scenario.questioner.Ask(
            "This example creates three buckets. Enter a prefix to name your buckets
            (remember bucket names must be globally unique):")

        for _, info := range createInfo {
            log.Println(fmt.Sprintf("%s.%s", prefix, info.name))
            bucketName, err := scenario.s3Actions.CreateBucketWithLock(ctx,
                fmt.Sprintf("%s.%s", prefix, info.name), scenario.sdkConfig.Region, info.locked)
            if err != nil {
                switch err.(type) {
                    case *types.BucketAlreadyExists, *types.BucketAlreadyOwnedByYou:
                        log.Printf("Couldn't create bucket %s.\n", bucketName)
                    default:
                        panic(err)
                }
                break
            }
            scenario.resources.demoBuckets[info.name] = &DemoBucket{
                name:      bucketName,
                objectKeys: []string{},
            }
            log.Printf("Created bucket %s.\n", bucketName)
        }

        if len(scenario.resources.demoBuckets) < len(createInfo) {
            scenario.resources.deleteBuckets(ctx)
        } else {
            success = true
        }
    }

    log.Println("S3 buckets created.")
    log.Println(strings.Repeat("-", 88))
}

// EnableLockOnBucket enables object locking on an existing bucket.
func (scenario *ObjectLockScenario) EnableLockOnBucket(ctx context.Context) {
    log.Println("\nA bucket can be configured to use object locking.")
}
```

```
scenario.questioner.Ask("Press Enter to continue.")

var err error
bucket := scenario.resources.demoBuckets["retention-bucket"]
err = scenario.s3Actions.EnableObjectLockOnBucket(ctx, bucket.name)
if err != nil {
    switch err.(type) {
    case *types.NoSuchBucket:
        log.Printf("Couldn't enable object locking on bucket %s.\n", bucket.name)
    default:
        panic(err)
    }
} else {
    log.Printf("Object locking enabled on bucket %s.", bucket.name)
}

log.Println(strings.Repeat("-", 88))
}

// SetDefaultRetentionPolicy sets a default retention governance policy on a
// bucket.
func (scenario *ObjectLockScenario) SetDefaultRetentionPolicy(ctx
context.Context) {
    log.Println("\nA bucket can be configured to use object locking with a default
retention period.")

    bucket := scenario.resources.demoBuckets["retention-bucket"]
    retentionPeriod := scenario.questioner.AskInt("Enter the default retention
period in days: ")
    err := scenario.s3Actions.ModifyDefaultBucketRetention(ctx,
bucket.name, types.ObjectLockEnabledEnabled, int32(retentionPeriod),
types.ObjectLockRetentionModeGovernance)
    if err != nil {
        switch err.(type) {
        case *types.NoSuchBucket:
            log.Printf("Couldn't configure a default retention period on bucket %s.\n",
bucket.name)
        default:
            panic(err)
        }
    } else {
        log.Printf("Default retention policy set on bucket %s with %d day retention
period.", bucket.name, retentionPeriod)
        bucket.retentionEnabled = true
    }
}
```

```

    }

    log.Println(strings.Repeat("-", 88))
}

// UploadTestObjects uploads test objects to the S3 buckets.
func (scenario *ObjectLockScenario) UploadTestObjects(ctx context.Context) {
    log.Println("Uploading test objects to S3 buckets.")

    for _, info := range createInfo {
        bucket := scenario.resources.demoBuckets[info.name]
        for i := 0; i < 2; i++ {
            key, err := scenario.s3Actions.UploadObject(ctx, bucket.name,
                fmt.Sprintf("example-%d", i),
                fmt.Sprintf("Example object content #%d in bucket %s.", i, bucket.name))
            if err != nil {
                switch err.(type) {
                case *types.NoSuchBucket:
                    log.Printf("Couldn't upload %s to bucket %s.\n", key, bucket.name)
                default:
                    panic(err)
                }
            } else {
                log.Printf("Uploaded %s to bucket %s.\n", key, bucket.name)
                bucket.objectKeys = append(bucket.objectKeys, key)
            }
        }
    }
}

scenario.questioner.Ask("Test objects uploaded. Press Enter to continue.")
log.Println(strings.Repeat("-", 88))
}

// SetObjectLockConfigurations sets object lock configurations on the test
objects.
func (scenario *ObjectLockScenario) SetObjectLockConfigurations(ctx
    context.Context) {
    log.Println("Now let's set object lock configurations on individual objects.")

    buckets := []*DemoBucket{scenario.resources.demoBuckets["lock-bucket"],
        scenario.resources.demoBuckets["retention-bucket"]}
    for _, bucket := range buckets {
        for index, objKey := range bucket.objectKeys {
            switch index {

```

```

    case 0:
        if scenario.questioner.AskBool(fmt.Sprintf("\nDo you want to add a legal hold
to %s in %s (y/n)? ", objKey, bucket.name), "y") {
            err := scenario.s3Actions.PutObjectLegalHold(ctx, bucket.name, objKey, "",
types.ObjectLockLegalHoldStatusOn)
            if err != nil {
                switch err.(type) {
                    case *types.NoSuchKey:
                        log.Printf("Couldn't set legal hold on %s.\n", objKey)
                    default:
                        panic(err)
                }
            } else {
                log.Printf("Legal hold set on %s.\n", objKey)
            }
        }
    case 1:
        q := fmt.Sprintf("\nDo you want to add a 1 day Governance retention period to
%s in %s?\n"+
"Reminder: Only a user with the s3:BypassGovernanceRetention permission is
able to delete this object\n"+
"or its bucket until the retention period has expired. (y/n) ", objKey,
bucket.name)
        if scenario.questioner.AskBool(q, "y") {
            err := scenario.s3Actions.PutObjectRetention(ctx, bucket.name, objKey,
types.ObjectLockRetentionModeGovernance, 1)
            if err != nil {
                switch err.(type) {
                    case *types.NoSuchKey:
                        log.Printf("Couldn't set retention period on %s in %s.\n", objKey,
bucket.name)
                    default:
                        panic(err)
                }
            } else {
                log.Printf("Retention period set to 1 for %s.", objKey)
                bucket.retentionEnabled = true
            }
        }
    }
}
log.Println(strings.Repeat("-", 88))
}

```

```
const (
    ListAll = iota
    DeleteObject
    DeleteRetentionObject
    OverwriteObject
    ViewRetention
    ViewLegalHold
    Finish
)

// InteractWithObjects allows the user to interact with the objects and test the
// object lock configurations.
func (scenario *ObjectLockScenario) InteractWithObjects(ctx context.Context) {
    log.Println("Now you can interact with the objects to explore the object lock
    configurations.")
    interactiveChoices := []string{
        "List all objects and buckets.",
        "Attempt to delete an object.",
        "Attempt to delete an object with retention period bypass.",
        "Attempt to overwrite a file.",
        "View the retention settings for an object.",
        "View the legal hold settings for an object.",
        "Finish the workflow."}

    choice := ListAll
    for choice != Finish {
        objList := scenario.GetAllObjects(ctx)
        objChoices := scenario.makeObjectChoiceList(objList)
        choice = scenario.questioner.AskChoice("Choose an action from the menu:\n",
        interactiveChoices)
        switch choice {
        case ListAll:
            log.Println("The current objects in the example buckets are:")
            for _, objChoice := range objChoices {
                log.Println("\t", objChoice)
            }
        case DeleteObject, DeleteRetentionObject:
            objChoice := scenario.questioner.AskChoice("Enter the number of the object to
            delete:\n", objChoices)
            obj := objList[objChoice]
            deleted, err := scenario.s3Actions.DeleteObject(ctx, obj.bucket, obj.key,
            obj.versionId, choice == DeleteRetentionObject)
            if err != nil {
```

```
switch err.(type) {
case *types.NoSuchKey:
    log.Println("Nothing to delete.")
default:
    panic(err)
}
} else if deleted {
    log.Printf("Object %s deleted.\n", obj.key)
}
case OverwriteObject:
    objChoice := scenario.questioner.AskChoice("Enter the number of the object to
overwrite:\n", objChoices)
    obj := objList[objChoice]
    _, err := scenario.s3Actions.UploadObject(ctx, obj.bucket, obj.key,
fmt.Sprintf("New content in object %s.", obj.key))
    if err != nil {
        switch err.(type) {
        case *types.NoSuchBucket:
            log.Println("Couldn't upload to nonexistent bucket.")
        default:
            panic(err)
        }
    } else {
        log.Printf("Uploaded new content to object %s.\n", obj.key)
    }
case ViewRetention:
    objChoice := scenario.questioner.AskChoice("Enter the number of the object to
view:\n", objChoices)
    obj := objList[objChoice]
    retention, err := scenario.s3Actions.GetObjectRetention(ctx, obj.bucket,
obj.key)
    if err != nil {
        switch err.(type) {
        case *types.NoSuchKey:
            log.Printf("Can't get retention configuration for %s.\n", obj.key)
        default:
            panic(err)
        }
    } else if retention != nil {
        log.Printf("Object %s has retention mode %s until %v.\n", obj.key,
retention.Mode, retention.RetainUntilDate)
    } else {
        log.Printf("Object %s does not have object retention configured.\n", obj.key)
    }
}
```

```

    case ViewLegalHold:
        objChoice := scenario.questioner.AskChoice("Enter the number of the object to
view:\n", objChoices)
        obj := objList[objChoice]
        legalHold, err := scenario.s3Actions.GetObjectLegalHold(ctx, obj.bucket,
obj.key, obj.versionId)
        if err != nil {
            switch err.(type) {
            case *types.NoSuchKey:
                log.Printf("Can't get legal hold configuration for %s.\n", obj.key)
            default:
                panic(err)
            }
        } else if legalHold != nil {
            log.Printf("Object %s has legal hold %v.", obj.key, *legalHold)
        } else {
            log.Printf("Object %s does not have legal hold configured.", obj.key)
        }
    case Finish:
        log.Println("Let's clean up.")
    }
    log.Println(strings.Repeat("-", 88))
}
}

type BucketKeyVersionId struct {
    bucket    string
    key       string
    versionId string
}

// GetAllObjects gets the object versions in the example S3 buckets and returns
them in a flattened list.
func (scenario *ObjectLockScenario) GetAllObjects(ctx context.Context)
[]BucketKeyVersionId {
    var objectList []BucketKeyVersionId
    for _, info := range createInfo {
        bucket := scenario.resources.demoBuckets[info.name]
        versions, err := scenario.s3Actions.ListObjectVersions(ctx, bucket.name)
        if err != nil {
            switch err.(type) {
            case *types.NoSuchBucket:
                log.Printf("Couldn't get object versions for %s.\n", bucket.name)
            default:

```



```

    panic(err)
}
} else {
    for _, version := range versions {
        objectList = append(objectList,
            BucketKeyVersionId{bucket: bucket.name, key: *version.Key, versionId:
*version.VersionId})
    }
}
}
return objectList
}

// makeObjectChoiceList makes the object version list into a list of strings that
// are displayed
// as choices.
func (scenario *ObjectLockScenario) makeObjectChoiceList(bucketObjects
[]BucketKeyVersionId) []string {
    choices := make([]string, len(bucketObjects))
    for i := 0; i < len(bucketObjects); i++ {
        choices[i] = fmt.Sprintf("%s in %s with VersionId %s.",
            bucketObjects[i].key, bucketObjects[i].bucket, bucketObjects[i].versionId)
    }
    return choices
}

// Run runs the S3 Object Lock scenario.
func (scenario *ObjectLockScenario) Run(ctx context.Context) {
    defer func() {
        if r := recover(); r != nil {
            log.Println("Something went wrong with the demo.")
            _, isMock := scenario.questioner.(*demotools.MockQuestioner)
            if isMock || scenario.questioner.AskBool("Do you want to see the full error
message (y/n)?", "y") {
                log.Println(r)
            }
            scenario.resources.Cleanup(ctx)
        }
    }()

    log.Println(strings.Repeat("-", 88))
    log.Println("Welcome to the Amazon S3 Object Lock Feature Scenario.")
    log.Println(strings.Repeat("-", 88))

```

```
scenario.CreateBuckets(ctx)
scenario.EnableLockOnBucket(ctx)
scenario.SetDefaultRetentionPolicy(ctx)
scenario.UploadTestObjects(ctx)
scenario.SetObjectLockConfigurations(ctx)
scenario.InteractWithObjects(ctx)

scenario.resources.Cleanup(ctx)

log.Println(strings.Repeat("-", 88))
log.Println("Thanks for watching!")
log.Println(strings.Repeat("-", 88))
}
```

定義包裝此範例使用之 S3 動作的結構。

```
import (
    "bytes"
    "context"
    "errors"
    "fmt"
    "log"
    "time"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"
    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/aws-sdk-go-v2/service/s3/types"
    "github.com/aws/smithy-go"
)

// S3Actions wraps S3 service actions.
type S3Actions struct {
    S3Client    *s3.Client
    S3Manager   *manager.Uploader
}
```

```
// CreateBucketWithLock creates a new S3 bucket with optional object locking
// enabled
// and waits for the bucket to exist before returning.
func (actor S3Actions) CreateBucketWithLock(ctx context.Context, bucket string,
    region string, enableObjectLock bool) (string, error) {
    input := &s3.CreateBucketInput{
        Bucket: aws.String(bucket),
        CreateBucketConfiguration: &types.CreateBucketConfiguration{
            LocationConstraint: types.BucketLocationConstraint(region),
        },
    }

    if enableObjectLock {
        input.ObjectLockEnabledForBucket = aws.Bool(true)
    }

    _, err := actor.S3Client.CreateBucket(ctx, input)
    if err != nil {
        var owned *types.BucketAlreadyOwnedByYou
        var exists *types.BucketAlreadyExists
        if errors.As(err, &owned) {
            log.Printf("You already own bucket %s.\n", bucket)
            err = owned
        } else if errors.As(err, &exists) {
            log.Printf("Bucket %s already exists.\n", bucket)
            err = exists
        }
    } else {
        err = s3.NewBucketExistsWaiter(actor.S3Client).Wait(
            ctx, &s3.HeadBucketInput{Bucket: aws.String(bucket)}, time.Minute)
        if err != nil {
            log.Printf("Failed attempt to wait for bucket %s to exist.\n", bucket)
        }
    }

    return bucket, err
}

// GetObjectLegalHold retrieves the legal hold status for an S3 object.
func (actor S3Actions) GetObjectLegalHold(ctx context.Context, bucket string, key
    string, versionId string) (*types.ObjectLockLegalHoldStatus, error) {
    var status *types.ObjectLockLegalHoldStatus
```

```

input := &s3.GetObjectLegalHoldInput{
    Bucket:    aws.String(bucket),
    Key:       aws.String(key),
    VersionId: aws.String(versionId),
}

output, err := actor.S3Client.GetObjectLegalHold(ctx, input)
if err != nil {
    var noSuchKeyErr *types.NoSuchKey
    var apiErr *smithy.GenericAPIError
    if errors.As(err, &noSuchKeyErr) {
        log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
        err = noSuchKeyErr
    } else if errors.As(err, &apiErr) {
        switch apiErr.ErrorCode() {
        case "NoSuchObjectLockConfiguration":
            log.Printf("Object %s does not have an object lock configuration.\n", key)
            err = nil
        case "InvalidRequest":
            log.Printf("Bucket %s does not have an object lock configuration.\n", bucket)
            err = nil
        }
    }
} else {
    status = &output.LegalHold.Status
}

return status, err
}

// GetObjectLockConfiguration retrieves the object lock configuration for an S3
// bucket.
func (actor S3Actions) GetObjectLockConfiguration(ctx context.Context, bucket
string) (*types.ObjectLockConfiguration, error) {
    var lockConfig *types.ObjectLockConfiguration
    input := &s3.GetObjectLockConfigurationInput{
        Bucket: aws.String(bucket),
    }

    output, err := actor.S3Client.GetObjectLockConfiguration(ctx, input)
    if err != nil {
        var noBucket *types.NoSuchBucket

```

```

var apiErr *smithy.GenericAPIError
if errors.As(err, &noBucket) {
    log.Printf("Bucket %s does not exist.\n", bucket)
    err = noBucket
} else if errors.As(err, &apiErr) && apiErr.ErrorCode() ==
"ObjectLockConfigurationNotFoundError" {
    log.Printf("Bucket %s does not have an object lock configuration.\n", bucket)
    err = nil
}
} else {
    lockConfig = output.ObjectLockConfiguration
}

return lockConfig, err
}

// GetObjectRetention retrieves the object retention configuration for an S3
object.
func (actor S3Actions) GetObjectRetention(ctx context.Context, bucket string, key
string) (*types.ObjectLockRetention, error) {
    var retention *types.ObjectLockRetention
    input := &s3.GetObjectRetentionInput{
        Bucket: aws.String(bucket),
        Key:     aws.String(key),
    }

    output, err := actor.S3Client.GetObjectRetention(ctx, input)
    if err != nil {
        var noKey *types.NoSuchKey
        var apiErr *smithy.GenericAPIError
        if errors.As(err, &noKey) {
            log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
            err = noKey
        } else if errors.As(err, &apiErr) {
            switch apiErr.ErrorCode() {
            case "NoSuchObjectLockConfiguration":
                err = nil
            case "InvalidRequest":
                log.Printf("Bucket %s does not have locking enabled.", bucket)
                err = nil
            }
        }
    }
}

```

```
} else {
    retention = output.Retention
}

return retention, err
}

// PutObjectLegalHold sets the legal hold configuration for an S3 object.
func (actor S3Actions) PutObjectLegalHold(ctx context.Context, bucket string, key
    string, versionId string, legalHoldStatus types.ObjectLockLegalHoldStatus) error
{
    input := &s3.PutObjectLegalHoldInput{
        Bucket: aws.String(bucket),
        Key:     aws.String(key),
        LegalHold: &types.ObjectLockLegalHold{
            Status: legalHoldStatus,
        },
    }
    if versionId != "" {
        input.VersionId = aws.String(versionId)
    }

    _, err := actor.S3Client.PutObjectLegalHold(ctx, input)
    if err != nil {
        var noKey *types.NoSuchKey
        if errors.As(err, &noKey) {
            log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
            err = noKey
        }
    }

    return err
}

// ModifyDefaultBucketRetention modifies the default retention period of an
    existing bucket.
func (actor S3Actions) ModifyDefaultBucketRetention(
    ctx context.Context, bucket string, lockMode types.ObjectLockEnabled,
    retentionPeriod int32, retentionMode types.ObjectLockRetentionMode) error {
```

```

input := &s3.PutObjectLockConfigurationInput{
    Bucket: aws.String(bucket),
    ObjectLockConfiguration: &types.ObjectLockConfiguration{
        ObjectLockEnabled: lockMode,
        Rule: &types.ObjectLockRule{
            DefaultRetention: &types.DefaultRetention{
                Days: aws.Int32(retentionPeriod),
                Mode: retentionMode,
            },
        },
    },
}
_, err := actor.S3Client.PutObjectLockConfiguration(ctx, input)
if err != nil {
    var noBucket *types.NoSuchBucket
    if errors.As(err, &noBucket) {
        log.Printf("Bucket %s does not exist.\n", bucket)
        err = noBucket
    }
}

return err
}

// EnableObjectLockOnBucket enables object locking on an existing bucket.
func (actor S3Actions) EnableObjectLockOnBucket(ctx context.Context, bucket
string) error {
    // Versioning must be enabled on the bucket before object locking is enabled.
    verInput := &s3.PutBucketVersioningInput{
        Bucket: aws.String(bucket),
        VersioningConfiguration: &types.VersioningConfiguration{
            MFADelete: types.MFADeleteDisabled,
            Status:    types.BucketVersioningStatusEnabled,
        },
    }
    _, err := actor.S3Client.PutBucketVersioning(ctx, verInput)
    if err != nil {
        var noBucket *types.NoSuchBucket
        if errors.As(err, &noBucket) {
            log.Printf("Bucket %s does not exist.\n", bucket)
            err = noBucket
        }
    }
}

```

```

    return err
}

input := &s3.PutObjectLockConfigurationInput{
    Bucket: aws.String(bucket),
    ObjectLockConfiguration: &types.ObjectLockConfiguration{
        ObjectLockEnabled: types.ObjectLockEnabledEnabled,
    },
}
_, err = actor.S3Client.PutObjectLockConfiguration(ctx, input)
if err != nil {
    var noBucket *types.NoSuchBucket
    if errors.As(err, &noBucket) {
        log.Printf("Bucket %s does not exist.\n", bucket)
        err = noBucket
    }
}

return err
}

// PutObjectRetention sets the object retention configuration for an S3 object.
func (actor S3Actions) PutObjectRetention(ctx context.Context, bucket string, key
string, retentionMode types.ObjectLockRetentionMode, retentionPeriodDays int32)
error {
input := &s3.PutObjectRetentionInput{
    Bucket: aws.String(bucket),
    Key:    aws.String(key),
    Retention: &types.ObjectLockRetention{
        Mode:          retentionMode,
        RetainUntilDate: aws.Time(time.Now().AddDate(0, 0, int(retentionPeriodDays))),
    },
    BypassGovernanceRetention: aws.Bool(true),
}

_, err := actor.S3Client.PutObjectRetention(ctx, input)
if err != nil {
    var noKey *types.NoSuchKey
    if errors.As(err, &noKey) {
        log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
        err = noKey
    }
}

```



```
}

return err
}

// UploadObject uses the S3 upload manager to upload an object to a bucket.
func (actor S3Actions) UploadObject(ctx context.Context, bucket string, key
string, contents string) (string, error) {
    var outKey string
    input := &s3.PutObjectInput{
        Bucket:      aws.String(bucket),
        Key:         aws.String(key),
        Body:        bytes.NewReader([]byte(contents)),
        ChecksumAlgorithm: types.ChecksumAlgorithmSha256,
    }
    output, err := actor.S3Manager.Upload(ctx, input)
    if err != nil {
        var noBucket *types.NoSuchBucket
        if errors.As(err, &noBucket) {
            log.Printf("Bucket %s does not exist.\n", bucket)
            err = noBucket
        }
    } else {
        err := s3.NewObjectExistsWaiter(actor.S3Client).Wait(ctx, &s3.HeadObjectInput{
            Bucket: aws.String(bucket),
            Key:    aws.String(key),
        }, time.Minute)
        if err != nil {
            log.Printf("Failed attempt to wait for object %s to exist in %s.\n", key,
            bucket)
        } else {
            outKey = *output.Key
        }
    }
    return outKey, err
}

// ListObjectVersions lists all versions of all objects in a bucket.
func (actor S3Actions) ListObjectVersions(ctx context.Context, bucket string)
([]types.ObjectVersion, error) {
```

```

var err error
var output *s3.ListObjectVersionsOutput
var versions []types.ObjectVersion
input := &s3.ListObjectVersionsInput{Bucket: aws.String(bucket)}
versionPaginator := s3.NewListObjectVersionsPaginator(actor.S3Client, input)
for versionPaginator.HasMorePages() {
    output, err = versionPaginator.NextPage(ctx)
    if err != nil {
        var noBucket *types.NoSuchBucket
        if errors.As(err, &noBucket) {
            log.Printf("Bucket %s does not exist.\n", bucket)
            err = noBucket
        }
        break
    } else {
        versions = append(versions, output.Versions...)
    }
}
return versions, err
}

```

```

// DeleteObject deletes an object from a bucket.
func (actor S3Actions) DeleteObject(ctx context.Context, bucket string, key
string, versionId string, bypassGovernance bool) (bool, error) {
    deleted := false
    input := &s3.DeleteObjectInput{
        Bucket: aws.String(bucket),
        Key:     aws.String(key),
    }
    if versionId != "" {
        input.VersionId = aws.String(versionId)
    }
    if bypassGovernance {
        input.BypassGovernanceRetention = aws.Bool(true)
    }
    _, err := actor.S3Client.DeleteObject(ctx, input)
    if err != nil {
        var noKey *types.NoSuchKey
        var apiErr *smithy.GenericAPIError
        if errors.As(err, &noKey) {
            log.Printf("Object %s does not exist in %s.\n", key, bucket)
            err = noKey
        }
    }
}

```

```

    } else if errors.As(err, &apiErr) {
        switch apiErr.ErrorCode() {
        case "AccessDenied":
            log.Printf("Access denied: cannot delete object %s from %s.\n", key, bucket)
            err = nil
        case "InvalidArgument":
            if bypassGovernance {
                log.Printf("You cannot specify bypass governance on a bucket without lock
enabled.")
                err = nil
            }
        }
    }
} else {
    err = s3.NewObjectNotExistsWaiter(actor.S3Client).Wait(
        ctx, &s3.HeadObjectInput{Bucket: aws.String(bucket), Key: aws.String(key)},
        time.Minute)
    if err != nil {
        log.Printf("Failed attempt to wait for object %s in bucket %s to be deleted.
\n", key, bucket)
    } else {
        deleted = true
    }
}
return deleted, err
}

// DeleteObjects deletes a list of objects from a bucket.
func (actor S3Actions) DeleteObjects(ctx context.Context, bucket string, objects
[]types.ObjectIdentifier, bypassGovernance bool) error {
    if len(objects) == 0 {
        return nil
    }

    input := s3.DeleteObjectsInput{
        Bucket: aws.String(bucket),
        Delete: &types.Delete{
            Objects: objects,
            Quiet:   aws.Bool(true),
        },
    }
    if bypassGovernance {

```

```

    input.BypassGovernanceRetention = aws.Bool(true)
}
delOut, err := actor.S3Client.DeleteObjects(ctx, &input)
if err != nil || len(delOut.Errors) > 0 {
    log.Printf("Error deleting objects from bucket %s.\n", bucket)
    if err != nil {
        var noBucket *types.NoSuchBucket
        if errors.As(err, &noBucket) {
            log.Printf("Bucket %s does not exist.\n", bucket)
            err = noBucket
        }
    } else if len(delOut.Errors) > 0 {
        for _, outErr := range delOut.Errors {
            log.Printf("%s: %s\n", *outErr.Key, *outErr.Message)
        }
        err = fmt.Errorf("%s", *delOut.Errors[0].Message)
    }
} else {
    for _, delObjs := range delOut.Deleted {
        err = s3.NewObjectNotExistsWaiter(actor.S3Client).Wait(
            ctx, &s3.HeadObjectInput{Bucket: aws.String(bucket), Key: delObjs.Key},
            time.Minute)
        if err != nil {
            log.Printf("Failed attempt to wait for object %s to be deleted.\n",
                *delObjs.Key)
        } else {
            log.Printf("Deleted %s.\n", *delObjs.Key)
        }
    }
}
return err
}

```

清除資源。

```

import (
    "context"
    "log"
    "s3_object_lock/actions"

```

```

"time"

"github.com/aws/aws-sdk-go-v2/aws"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/aws/aws-sdk-go-v2/service/s3/types"
"github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// DemoBucket contains metadata for buckets used in this example.
type DemoBucket struct {
    name            string
    retentionEnabled bool
    objectKeys      []string
}

// Resources keeps track of AWS resources created during the ObjectLockScenario
// and handles
// cleanup when the scenario finishes.
type Resources struct {
    demoBuckets map[string]*DemoBucket

    s3Actions  *actions.S3Actions
    questioner demotools.IQuestioner
}

// init initializes objects in the Resources struct.
func (resources *Resources) init(s3Actions *actions.S3Actions, questioner
    demotools.IQuestioner) {
    resources.s3Actions = s3Actions
    resources.questioner = questioner
    resources.demoBuckets = map[string]*DemoBucket{}
}

// Cleanup deletes all AWS resources created during the ObjectLockScenario.
func (resources *Resources) Cleanup(ctx context.Context) {
    defer func() {
        if r := recover(); r != nil {
            log.Printf("Something went wrong during cleanup.\n%v\n", r)
            log.Println("Use the AWS Management Console to remove any remaining resources"
                +
                "that were created for this scenario.")
        }
    }()
}

```

```
wantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS
resources that were created "+
    "during this demo (y/n)?", "y")
if !wantDelete {
    log.Println("Be sure to remove resources when you're done with them to avoid
unexpected charges!")
    return
}

log.Println("Removing objects from S3 buckets and deleting buckets...")
resources.deleteBuckets(ctx)
//resources.deleteRetentionObjects(resources.retentionBucket,
resources.retentionObjects)

log.Println("Cleanup complete.")
}

// deleteBuckets empties and then deletes all buckets created during the
ObjectLockScenario.
func (resources *Resources) deleteBuckets(ctx context.Context) {
    for _, info := range createInfo {
        bucket := resources.demoBuckets[info.name]
        resources.deleteObjects(ctx, bucket)
        _, err := resources.s3Actions.S3Client.DeleteBucket(ctx, &s3.DeleteBucketInput{
            Bucket: aws.String(bucket.name),
        })
        if err != nil {
            panic(err)
        }
    }
    for _, info := range createInfo {
        bucket := resources.demoBuckets[info.name]
        err := s3.NewBucketNotExistsWaiter(resources.s3Actions.S3Client).Wait(
            ctx, &s3.HeadBucketInput{Bucket: aws.String(bucket.name)}, time.Minute)
        if err != nil {
            log.Printf("Failed attempt to wait for bucket %s to be deleted.\n",
bucket.name)
        } else {
            log.Printf("Deleted %s.\n", bucket.name)
        }
    }
    resources.demoBuckets = map[string]*DemoBucket{}
}
```

```
// deleteObjects deletes all objects in the specified bucket.
func (resources *Resources) deleteObjects(ctx context.Context, bucket
    *DemoBucket) {
    lockConfig, err := resources.s3Actions.GetObjectLockConfiguration(ctx,
        bucket.name)
    if err != nil {
        panic(err)
    }
    versions, err := resources.s3Actions.ListObjectVersions(ctx, bucket.name)
    if err != nil {
        switch err.(type) {
        case *types.NoSuchBucket:
            log.Printf("No objects to get from %s.\n", bucket.name)
        default:
            panic(err)
        }
    }
    delObjects := make([]types.ObjectIdentifier, len(versions))
    for i, version := range versions {
        if lockConfig != nil && lockConfig.ObjectLockEnabled ==
            types.ObjectLockEnabledEnabled {
            status, err := resources.s3Actions.GetObjectLegalHold(ctx, bucket.name,
                *version.Key, *version.VersionId)
            if err != nil {
                switch err.(type) {
                case *types.NoSuchKey:
                    log.Printf("Couldn't determine legal hold status for %s in %s.\n",
                        *version.Key, bucket.name)
                default:
                    panic(err)
                }
            } else if status != nil && *status == types.ObjectLockLegalHoldStatusOn {
                err = resources.s3Actions.PutObjectLegalHold(ctx, bucket.name, *version.Key,
                    *version.VersionId, types.ObjectLockLegalHoldStatusOff)
                if err != nil {
                    switch err.(type) {
                    case *types.NoSuchKey:
                        log.Printf("Couldn't turn off legal hold for %s in %s.\n", *version.Key,
                            bucket.name)
                    default:
                        panic(err)
                    }
                }
            }
        }
    }
}
```

```
}
    del0bjects[i] = types.ObjectIdentifier{Key: version.Key, VersionId:
version.VersionId}
}
err = resources.s3Actions.Delete0bjects(ctx, bucket.name, del0bjects,
bucket.retentionEnabled)
if err != nil {
    switch err.(type) {
    case *types.NoSuchBucket:
        log.Println("Nothing to delete.")
    default:
        panic(err)
    }
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的下列主題。

- [GetObjectLegalHold](#)
- [GetObjectLockConfiguration](#)
- [GetObjectRetention](#)
- [PutObjectLegalHold](#)
- [PutObjectLockConfiguration](#)
- [PutObjectRetention](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

執行展示 Amazon S3 物件鎖定功能的互動式情境。

```
import software.amazon.awssdk.services.s3.model.ObjectLockLegalHold;
```



```
import software.amazon.awssdk.services.s3.model.ObjectLockRetention;
import java.io.BufferedWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
import java.util.stream.Collectors;

/*
Before running this Java V2 code example, set up your development
environment, including your credentials.

For more information, see the following documentation topic:
https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/setup.html

This Java example performs the following tasks:
    1. Create test Amazon Simple Storage Service (S3) buckets with different lock
    policies.
    2. Upload sample objects to each bucket.
    3. Set some Legal Hold and Retention Periods on objects and buckets.
    4. Investigate lock policies by viewing settings or attempting to delete or
    overwrite objects.
    5. Clean up objects and buckets.
*/
public class S3ObjectLockWorkflow {

    public static final String DASHES = new String(new char[80]).replace("\0",
"-");
    static String bucketName;
    static S3LockActions s3LockActions;
    private static final List<String> bucketNames = new ArrayList<>();
    private static final List<String> fileNames = new ArrayList<>();

    public static void main(String[] args) {
        final String usage = ""
            Usage:
                <bucketName> \s

        Where:
            bucketName - The Amazon S3 bucket name.
        """;

        if (args.length != 1) {
            System.out.println(usage);
        }
    }
}
```

```
        System.exit(1);
    }
    s3LockActions = new S3LockActions();
    bucketName = args[0];
    Scanner scanner = new Scanner(System.in);

    System.out.println(DASHES);
    System.out.println("Welcome to the Amazon Simple Storage Service (S3)
Object Locking Feature Scenario.");
    System.out.println("Press Enter to continue...");
    scanner.nextLine();
    configurationSetup();
    System.out.println(DASHES);

    System.out.println(DASHES);
    setup();
    System.out.println("Setup is complete. Press Enter to continue...");
    scanner.nextLine();
    System.out.println(DASHES);

    System.out.println(DASHES);
    System.out.println("Lets present the user with choices.");
    System.out.println("Press Enter to continue...");
    scanner.nextLine();
    demoActionChoices() ;
    System.out.println(DASHES);

    System.out.println(DASHES);
    System.out.println("Would you like to clean up the resources? (y/n)");
    String delAns = scanner.nextLine().trim();
    if (delAns.equalsIgnoreCase("y")) {
        cleanup();
        System.out.println("Clean up is complete.");
    }

    System.out.println("Press Enter to continue...");
    scanner.nextLine();
    System.out.println(DASHES);

    System.out.println(DASHES);
    System.out.println("Amazon S3 Object Locking Workflow is complete.");
    System.out.println(DASHES);
}
```

```
// Present the user with the demo action choices.
public static void demoActionChoices() {
    String[] choices = {
        "List all files in buckets.",
        "Attempt to delete a file.",
        "Attempt to delete a file with retention period bypass.",
        "Attempt to overwrite a file.",
        "View the object and bucket retention settings for a file.",
        "View the legal hold settings for a file.",
        "Finish the workflow."
    };

    int choice = 0;
    while (true) {
        System.out.println(DASHES);
        choice = getChoiceResponse("Explore the S3 locking features by
selecting one of the following choices:", choices);
        System.out.println(DASHES);
        System.out.println("You selected "+choices[choice]);
        switch (choice) {
            case 0 -> {
                s3LockActions.listBucketsAndObjects(bucketNames, true);
            }

            case 1 -> {
                System.out.println("Enter the number of the object to
delete:");

                List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, true);
                List<String> fileKeys = allFiles.stream().map(f ->
f.getKeyName()).collect(Collectors.toList());
                String[] fileKeysArray = fileKeys.toArray(new String[0]);
                int fileChoice = getChoiceResponse(null, fileKeysArray);
                String objectKey = fileKeys.get(fileChoice);
                String bucketName = allFiles.get(fileChoice).getBucketName();
                String version = allFiles.get(fileChoice).getVersion();
                s3LockActions.deleteObjectFromBucket(bucketName, objectKey,
false, version);
            }

            case 2 -> {
                System.out.println("Enter the number of the object to
delete:");
            }
        }
    }
}
```

```
        List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, true);
        List<String> fileKeys = allFiles.stream().map(f ->
f.getKeyName()).collect(Collectors.toList());
        String[] fileKeysArray = fileKeys.toArray(new String[0]);
        int fileChoice = getChoiceResponse(null, fileKeysArray);
        String objectKey = fileKeys.get(fileChoice);
        String bucketName = allFiles.get(fileChoice).getBucketName();
        String version = allFiles.get(fileChoice).getVersion();
        s3LockActions.deleteObjectFromBucket(bucketName, objectKey,
true, version);
    }

    case 3 -> {
        System.out.println("Enter the number of the object to
overwrite:");

        List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, true);
        List<String> fileKeys = allFiles.stream().map(f ->
f.getKeyName()).collect(Collectors.toList());
        String[] fileKeysArray = fileKeys.toArray(new String[0]);
        int fileChoice = getChoiceResponse(null, fileKeysArray);
        String objectKey = fileKeys.get(fileChoice);
        String bucketName = allFiles.get(fileChoice).getBucketName();

        // Attempt to overwrite the file.
        try (BufferedWriter writer = new BufferedWriter(new
java.io.FileWriter(objectKey))) {
            writer.write("This is a modified text.");

        } catch (IOException e) {
            e.printStackTrace();
        }
        s3LockActions.uploadFile(bucketName, objectKey, objectKey);
    }

    case 4 -> {
        System.out.println("Enter the number of the object to
overwrite:");

        List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, true);
        List<String> fileKeys = allFiles.stream().map(f ->
f.getKeyName()).collect(Collectors.toList());
        String[] fileKeysArray = fileKeys.toArray(new String[0]);
```

```

        int fileChoice = getChoiceResponse(null, fileKeysArray);
        String objectKey = fileKeys.get(fileChoice);
        String bucketName = allFiles.get(fileChoice).getBucketName();
        s3LockActions.getObjectRetention(bucketName, objectKey);
    }

    case 5 -> {
        System.out.println("Enter the number of the object to
view:");

        List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, true);
        List<String> fileKeys = allFiles.stream().map(f ->
f.getKeyName()).collect(Collectors.toList());
        String[] fileKeysArray = fileKeys.toArray(new String[0]);
        int fileChoice = getChoiceResponse(null, fileKeysArray);
        String objectKey = fileKeys.get(fileChoice);
        String bucketName = allFiles.get(fileChoice).getBucketName();
        s3LockActions.getObjectLegalHold(bucketName, objectKey);
        s3LockActions.getBucketObjectLockConfiguration(bucketName);
    }

    case 6 -> {
        System.out.println("Exiting the workflow...");
        return;
    }

    default -> {
        System.out.println("Invalid choice. Please select again.");
    }
}
}

// Clean up the resources from the scenario.
private static void cleanup() {
    List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, false);
    for (S3InfoObject fileInfo : allFiles) {
        String bucketName = fileInfo.getBucketName();
        String key = fileInfo.getKeyName();
        String version = fileInfo.getVersion();
        if (bucketName.contains("lock-enabled") ||
(bucketName.contains("retention-after-creation"))) {

```

```

        ObjectLockLegalHold legalHold =
s3LockActions.getObjectLegalHold(bucketName, key);
        if (legalHold != null) {
            String holdStatus = legalHold.status().name();
            System.out.println(holdStatus);
            if (holdStatus.compareTo("ON") == 0) {
                s3LockActions.modifyObjectLegalHold(bucketName, key,
false);
            }
        }
        // Check for a retention period.
        ObjectLockRetention retention =
s3LockActions.getObjectRetention(bucketName, key);
        boolean hasRetentionPeriod ;
        hasRetentionPeriod = retention != null;
        s3LockActions.deleteObjectFromBucket(bucketName,
key,hasRetentionPeriod, version);

    } else {
        System.out.println(bucketName +" objects do not have a legal
lock");
        s3LockActions.deleteObjectFromBucket(bucketName, key,false,
version);
    }
}

// Delete the buckets.
System.out.println("Delete "+bucketName);
for (String bucket : bucketNames){
    s3LockActions.deleteBucketByName(bucket);
}
}

private static void setup() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("""
        For this workflow, we will use the AWS SDK for Java to create
several S3
        buckets and files to demonstrate working with S3 locking
features.
        """);

    System.out.println("S3 buckets can be created either with or without
object lock enabled.");

```

```
System.out.println("Press Enter to continue...");
scanner.nextLine();

// Create three S3 buckets.
s3LockActions.createBucketWithLockOptions(false, bucketNames.get(0));
s3LockActions.createBucketWithLockOptions(true, bucketNames.get(1));
s3LockActions.createBucketWithLockOptions(false, bucketNames.get(2));
System.out.println("Press Enter to continue.");
scanner.nextLine();

System.out.println("Bucket " + bucketNames.get(2) + " will be configured to
use object locking with a default retention period.");
s3LockActions.modifyBucketDefaultRetention(bucketNames.get(2));
System.out.println("Press Enter to continue.");
scanner.nextLine();

System.out.println("Object lock policies can also be added to existing
buckets. For this example, we will use " + bucketNames.get(1));
s3LockActions.enableObjectLockOnBucket(bucketNames.get(1));
System.out.println("Press Enter to continue.");
scanner.nextLine();

// Upload some files to the buckets.
System.out.println("Now let's add some test files:");
String fileName = "exampleFile.txt";
int fileCount = 2;
try (BufferedWriter writer = new BufferedWriter(new
java.io.FileWriter(fileName))) {
    writer.write("This is a sample file for uploading to a bucket.");

} catch (IOException e) {
    e.printStackTrace();
}

for (String bucketName : bucketNames){
    for (int i = 0; i < fileCount; i++) {
        // Get the file name without extension.
        String fileNameWithoutExtension =
java.nio.file.Paths.get(fileName).getFileName().toString();
        int extensionIndex = fileNameWithoutExtension.lastIndexOf('.');
        if (extensionIndex > 0) {
            fileNameWithoutExtension =
fileNameWithoutExtension.substring(0, extensionIndex);
        }
    }
}
```

```

        // Create the numbered file names.
        String numberedFileName = fileNameWithoutExtension + i +
getFileExtension(fileName);
        fileNames.add(numberedFileName);
        s3LockActions.uploadFile(bucketName, numberedFileName, fileName);
    }
}

String question = null;
System.out.print("Press Enter to continue...");
scanner.nextLine();
System.out.println("Now we can set some object lock policies on
individual files:");
for (String bucketName : bucketNames) {
    for (int i = 0; i < fileNames.size(); i++){

        // No modifications to the objects in the first bucket.
        if (!bucketName.equals(bucketNames.get(0))) {
            String exampleFileName = fileNames.get(i);
            switch (i) {
                case 0 -> {
                    question = "Would you like to add a legal hold to " +
exampleFileName + " in " + bucketName + " (y/n)?";
                    System.out.println(question);
                    String ans = scanner.nextLine().trim();
                    if (ans.equalsIgnoreCase("y")) {
                        System.out.println("**** You have selected to put
a legal hold " + exampleFileName);

                        // Set a legal hold.
                        s3LockActions.modifyObjectLegalHold(bucketName,
exampleFileName, true);
                    }
                }
                case 1 -> {
                    """"
                        Would you like to add a 1 day Governance
retention period to %s in %s (y/n)?
                        Reminder: Only a user with the
s3:BypassGovernanceRetention permission will be able to delete this file or its
bucket until the retention period has expired.
                        """".formatted(exampleFileName, bucketName);
                    System.out.println(question);

```



```

        String ans2 = scanner.nextLine().trim();
        if (ans2.equalsIgnoreCase("y")) {

s3LockActions.modifyObjectRetentionPeriod(bucketName, exampleFileName);
        }
    }
}

// Get file extension.
private static String getFileExtension(String fileName) {
    int dotIndex = fileName.lastIndexOf('.');
    if (dotIndex > 0) {
        return fileName.substring(dotIndex);
    }
    return "";
}

public static void configurationSetup() {
    String noLockBucketName = bucketName + "-no-lock";
    String lockEnabledBucketName = bucketName + "-lock-enabled";
    String retentionAfterCreationBucketName = bucketName + "-retention-after-
creation";
    bucketNames.add(noLockBucketName);
    bucketNames.add(lockEnabledBucketName);
    bucketNames.add(retentionAfterCreationBucketName);
}

public static int getChoiceResponse(String question, String[] choices) {
    Scanner scanner = new Scanner(System.in);
    if (question != null) {
        System.out.println(question);
        for (int i = 0; i < choices.length; i++) {
            System.out.println("\t" + (i + 1) + ". " + choices[i]);
        }
    }

    int choiceNumber = 0;
    while (choiceNumber < 1 || choiceNumber > choices.length) {
        String choice = scanner.nextLine();
        try {

```

```
        choiceNumber = Integer.parseInt(choice);
    } catch (NumberFormatException e) {
        System.out.println("Invalid choice. Please enter a valid
number.");
    }
}

return choiceNumber - 1;
}
}
```

S3 函數的包裝函式類別。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.BucketVersioningStatus;
import software.amazon.awssdk.services.s3.model.ChecksumAlgorithm;
import software.amazon.awssdk.services.s3.model.CreateBucketRequest;
import software.amazon.awssdk.services.s3.model.DefaultRetention;
import software.amazon.awssdk.services.s3.model.DeleteBucketRequest;
import software.amazon.awssdk.services.s3.model.DeleteObjectRequest;
import software.amazon.awssdk.services.s3.model.GetObjectLegalHoldRequest;
import software.amazon.awssdk.services.s3.model.GetObjectLegalHoldResponse;
import
    software.amazon.awssdk.services.s3.model.GetObjectLockConfigurationRequest;
import
    software.amazon.awssdk.services.s3.model.GetObjectLockConfigurationResponse;
import software.amazon.awssdk.services.s3.model.GetObjectRetentionRequest;
import software.amazon.awssdk.services.s3.model.GetObjectRetentionResponse;
import software.amazon.awssdk.services.s3.model.HeadBucketRequest;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsRequest;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsResponse;
import software.amazon.awssdk.services.s3.model.MFADelete;
import software.amazon.awssdk.services.s3.model.ObjectLockConfiguration;
import software.amazon.awssdk.services.s3.model.ObjectLockEnabled;
import software.amazon.awssdk.services.s3.model.ObjectLockLegalHold;
import software.amazon.awssdk.services.s3.model.ObjectLockLegalHoldStatus;
import software.amazon.awssdk.services.s3.model.ObjectLockRetention;
import software.amazon.awssdk.services.s3.model.ObjectLockRetentionMode;
import software.amazon.awssdk.services.s3.model.ObjectLockRule;
import software.amazon.awssdk.services.s3.model.PutBucketVersioningRequest;
import software.amazon.awssdk.services.s3.model.PutObjectLegalHoldRequest;
```

```
import
    software.amazon.awssdk.services.s3.model.PutObjectLockConfigurationRequest;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;
import software.amazon.awssdk.services.s3.model.PutObjectRetentionRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.VersioningConfiguration;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.time.Instant;
import java.time.ZoneId;
import java.time.ZonedDateTime;
import java.time.format.DateTimeFormatter;
import java.time.temporal.ChronoUnit;
import java.util.List;
import java.util.concurrent.atomic.AtomicInteger;
import java.util.stream.Collectors;

// Contains application logic for the Amazon S3 operations used in this workflow.
public class S3LockActions {

    private static S3Client getClient() {
        return S3Client.builder()
            .region(Region.US_EAST_1)
            .build();
    }

    // Set or modify a retention period on an object in an S3 bucket.
    public void modifyObjectRetentionPeriod(String bucketName, String objectKey)
    {
        // Calculate the instant one day from now.
        Instant futureInstant = Instant.now().plus(1, ChronoUnit.DAYS);

        // Convert the Instant to a ZonedDateTime object with a specific time
        zone.
        ZonedDateTime zonedDateTime =
        futureInstant.atZone(ZoneId.systemDefault());

        // Define a formatter for human-readable output.
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd
        HH:mm:ss");

        // Format the ZonedDateTime object to a human-readable date string.
```

```
String humanReadableDate = formatter.format(zonedDateTime);

// Print the formatted date string.
System.out.println("Formatted Date: " + humanReadableDate);
ObjectLockRetention retention = ObjectLockRetention.builder()
    .mode(ObjectLockRetentionMode.GOVERNANCE)
    .retainUntilDate(futureInstant)
    .build();

PutObjectRetentionRequest retentionRequest =
PutObjectRetentionRequest.builder()
    .bucket(bucketName)
    .key(objectKey)
    .retention(retention)
    .build();

getClient().putObjectRetention(retentionRequest);
System.out.println("Set retention for "+objectKey+" in "+bucketName+"
until "+ humanReadableDate +".");
}

// Get the legal hold details for an S3 object.
public ObjectLockLegalHold getObjectLegalHold(String bucketName, String
objectKey) {
    try {
        GetObjectLegalHoldRequest legalHoldRequest =
GetObjectLegalHoldRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .build();

        GetObjectLegalHoldResponse response =
getClient().getObjectLegalHold(legalHoldRequest);
        System.out.println("Object legal hold for " + objectKey + " in " +
bucketName +
            ":\n\tStatus: " + response.legalHold().status());
        return response.legalHold();

    } catch (S3Exception ex) {
        System.out.println("\tUnable to fetch legal hold: '" +
ex.getMessage() + "'");
    }

    return null;
}
```

```

    }

    // Create a new Amazon S3 bucket with object lock options.
    public void createBucketWithLockOptions(boolean enableObjectLock, String
bucketName) {
        S3Waiter s3Waiter = getClient().waiter();
        CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
            .bucket(bucketName)
            .objectLockEnabledForBucket(enableObjectLock)
            .build();

        getClient().createBucket(bucketRequest);
        HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
            .bucket(bucketName)
            .build();

        // Wait until the bucket is created and print out the response.
        s3Waiter.waitUntilBucketExists(bucketRequestWait);
        System.out.println(bucketName + " is ready");
    }

    public List<S3InfoObject> listBucketsAndObjects(List<String> bucketNames,
Boolean interactive) {
        AtomicInteger counter = new AtomicInteger(0); // Initialize counter.
        return bucketNames.stream()
            .flatMap(bucketName ->
listBucketObjectsAndVersions(bucketName).versions().stream()
                .map(version -> {
                    S3InfoObject s3InfoObject = new S3InfoObject();
                    s3InfoObject.setBucketName(bucketName);
                    s3InfoObject.setVersion(version.versionId());
                    s3InfoObject.setKeyName(version.key());
                    return s3InfoObject;
                })))
            .peek(s3InfoObject -> {
                int i = counter.incrementAndGet(); // Increment and get the
updated value.
                if (interactive) {
                    System.out.println(i + ": " + s3InfoObject.getKeyName());
                    System.out.printf("%5s Bucket name: %s\n", "",
s3InfoObject.getBucketName());
                    System.out.printf("%5s Version: %s\n", "",
s3InfoObject.getVersion());
                }
            })
    }

```

```
        })
        .collect(Collectors.toList());
    }

    public ListObjectVersionsResponse listBucketObjectsAndVersions(String
bucketName) {
        ListObjectVersionsRequest versionsRequest =
ListObjectVersionsRequest.builder()
            .bucket(bucketName)
            .build();

        return getClient().listObjectVersions(versionsRequest);
    }

    // Set or modify a retention period on an S3 bucket.
    public void modifyBucketDefaultRetention(String bucketName) {
        VersioningConfiguration versioningConfiguration =
VersioningConfiguration.builder()
            .mfaDelete(MFADelete.DISABLED)
            .status(BucketVersioningStatus.ENABLED)
            .build();

        PutBucketVersioningRequest versioningRequest =
PutBucketVersioningRequest.builder()
            .bucket(bucketName)
            .versioningConfiguration(versioningConfiguration)
            .build();

        getClient().putBucketVersioning(versioningRequest);
        DefaultRetention rention = DefaultRetention.builder()
            .days(1)
            .mode(ObjectLockRetentionMode.GOVERNANCE)
            .build();

        ObjectLockRule lockRule = ObjectLockRule.builder()
            .defaultRetention(rention)
            .build();

        ObjectLockConfiguration objectLockConfiguration =
ObjectLockConfiguration.builder()
            .objectLockEnabled(ObjectLockEnabled.ENABLED)
            .rule(lockRule)
            .build();
    }
}
```

```
PutObjectLockConfigurationRequest putObjectLockConfigurationRequest =
PutObjectLockConfigurationRequest.builder()
    .bucket(bucketName)
    .objectLockConfiguration(objectLockConfiguration)
    .build();

getClient().putObjectLockConfiguration(putObjectLockConfigurationRequest) ;
System.out.println("Added a default retention to bucket "+bucketName
+ ".");
}

// Enable object lock on an existing bucket.
public void enableObjectLockOnBucket(String bucketName) {
    try {
        VersioningConfiguration versioningConfiguration =
VersioningConfiguration.builder()
            .status(BucketVersioningStatus.ENABLED)
            .build();

        PutBucketVersioningRequest putBucketVersioningRequest =
PutBucketVersioningRequest.builder()
            .bucket(bucketName)
            .versioningConfiguration(versioningConfiguration)
            .build();

        // Enable versioning on the bucket.
        getClient().putBucketVersioning(putBucketVersioningRequest);
        PutObjectLockConfigurationRequest request =
PutObjectLockConfigurationRequest.builder()
            .bucket(bucketName)
            .objectLockConfiguration(ObjectLockConfiguration.builder()
                .objectLockEnabled(ObjectLockEnabled.ENABLED)
                .build())
            .build();

        getClient().putObjectLockConfiguration(request);
        System.out.println("Successfully enabled object lock on
"+bucketName);

    } catch (S3Exception ex) {
        System.out.println("Error modifying object lock: '" + ex.getMessage()
+ "'");
    }
}
```

```
}

public void uploadFile(String bucketName, String objectName, String filePath)
{
    Path file = Paths.get(filePath);
    PutObjectRequest request = PutObjectRequest.builder()
        .bucket(bucketName)
        .key(objectName)
        .checksumAlgorithm(ChecksumAlgorithm.SHA256)
        .build();

    PutObjectResponse response = getClient().putObject(request, file);
    if (response != null) {
        System.out.println("\tSuccessfully uploaded " + objectName + " to " +
            bucketName + ".");
    } else {
        System.out.println("\tCould not upload " + objectName + " to " +
            bucketName + ".");
    }
}

// Set or modify a legal hold on an object in an S3 bucket.
public void modifyObjectLegalHold(String bucketName, String objectKey,
    boolean legalHoldOn) {
    ObjectLockLegalHold legalHold ;
    if (legalHoldOn) {
        legalHold = ObjectLockLegalHold.builder()
            .status(ObjectLockLegalHoldStatus.ON)
            .build();
    } else {
        legalHold = ObjectLockLegalHold.builder()
            .status(ObjectLockLegalHoldStatus.OFF)
            .build();
    }

    PutObjectLegalHoldRequest legalHoldRequest =
        PutObjectLegalHoldRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .legalHold(legalHold)
            .build();

    getClient().putObjectLegalHold(legalHoldRequest) ;
}
```



```
        System.out.println("Modified legal hold for "+ objectKey +" in
        "+bucketName +".");
    }

    // Delete an object from a specific bucket.
    public void deleteObjectFromBucket(String bucketName, String objectKey,
    boolean hasRetention, String versionId) {
        try {
            DeleteObjectRequest objectRequest;
            if (hasRetention) {
                objectRequest = DeleteObjectRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .versionId(versionId)
                    .bypassGovernanceRetention(true)
                    .build();
            } else {
                objectRequest = DeleteObjectRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .versionId(versionId)
                    .build();
            }

            getClient().deleteObject(objectRequest) ;
            System.out.println("The object was successfully deleted");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

    // Get the retention period for an S3 object.
    public ObjectLockRetention getObjectRetention(String bucketName, String key){
        try {
            GetObjectRetentionRequest retentionRequest =
            GetObjectRetentionRequest.builder()
                .bucket(bucketName)
                .key(key)
                .build();

            GetObjectRetentionResponse response =
            getClient().getObjectRetention(retentionRequest);
```

```

        System.out.println("Object retention for "+key +"
in "+ bucketName +": " + response.retention().mode() +" until "+
response.retention().retainUntilDate() +".");
        return response.retention();

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        return null;
    }
}

public void deleteBucketByName(String bucketName) {
    try {
        DeleteBucketRequest request = DeleteBucketRequest.builder()
            .bucket(bucketName)
            .build();

        getClient().deleteBucket(request);
        System.out.println(bucketName +" was deleted.");

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
    }
}

// Get the object lock configuration details for an S3 bucket.
public void getBucketObjectLockConfiguration(String bucketName) {
    GetObjectLockConfigurationRequest objectLockConfigurationRequest =
GetObjectLockConfigurationRequest.builder()
        .bucket(bucketName)
        .build();

    GetObjectLockConfigurationResponse response =
getClient().getObjectLockConfiguration(objectLockConfigurationRequest);
    System.out.println("Bucket object lock config for "+bucketName +": ");
    System.out.println("\tEnabled:
"+response.objectLockConfiguration().objectLockEnabled());
    System.out.println("\tRule: "+
response.objectLockConfiguration().rule().defaultRetention());
}
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。

- [GetObjectLegalHold](#)
- [GetObjectLockConfiguration](#)
- [GetObjectRetention](#)
- [PutObjectLegalHold](#)
- [PutObjectLockConfiguration](#)
- [PutObjectRetention](#)

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

案例的進入點 (index.js)。這會協調所有步驟。請造訪 GitHub 以查看 Scenario、ScenarioInput、ScenarioOutput 和 ScenarioAction 的實作詳細資訊。

```
import * as Scenarios from "@aws-doc-sdk-examples/lib/scenario/index.js";
import {
  exitOnFalse,
  loadState,
  saveState,
} from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";

import { welcome, welcomeContinue } from "../welcome.steps.js";
import {
  confirmCreateBuckets,
  confirmPopulateBuckets,
  confirmSetLegalHoldFileEnabled,
  confirmSetLegalHoldFileRetention,
  confirmSetRetentionPeriodFileEnabled,
  confirmSetRetentionPeriodFileRetention,
  confirmUpdateLockPolicy,
  confirmUpdateRetention,
  createBuckets,
  createBucketsAction,
```

```
getBucketPrefix,
populateBuckets,
populateBucketsAction,
setLegalHoldFileEnabledAction,
setLegalHoldFileRetentionAction,
setRetentionPeriodFileEnabledAction,
setRetentionPeriodFileRetentionAction,
updateLockPolicy,
updateLockPolicyAction,
updateRetention,
updateRetentionAction,
} from "./setup.steps.js";

/**
 * @param {Scenarios} scenarios
 * @param {Record<string, any>} initialState
 */
export const getWorkflowStages = (scenarios, initialState = {}) => {
  const client = new S3Client({});

  return {
    deploy: new scenarios.Scenario(
      "S3 Object Locking - Deploy",
      [
        welcome(scenarios),
        welcomeContinue(scenarios),
        exitOnFalse(scenarios, "welcomeContinue"),
        getBucketPrefix(scenarios),
        createBuckets(scenarios),
        confirmCreateBuckets(scenarios),
        exitOnFalse(scenarios, "confirmCreateBuckets"),
        createBucketsAction(scenarios, client),
        updateRetention(scenarios),
        confirmUpdateRetention(scenarios),
        exitOnFalse(scenarios, "confirmUpdateRetention"),
        updateRetentionAction(scenarios, client),
        populateBuckets(scenarios),
        confirmPopulateBuckets(scenarios),
        exitOnFalse(scenarios, "confirmPopulateBuckets"),
        populateBucketsAction(scenarios, client),
        updateLockPolicy(scenarios),
        confirmUpdateLockPolicy(scenarios),
        exitOnFalse(scenarios, "confirmUpdateLockPolicy"),
        updateLockPolicyAction(scenarios, client),
      ]
    )
  };
}
```

```

        confirmSetLegalHoldFileEnabled(scenarios),
        setLegalHoldFileEnabledAction(scenarios, client),
        confirmSetRetentionPeriodFileEnabled(scenarios),
        setRetentionPeriodFileEnabledAction(scenarios, client),
        confirmSetLegalHoldFileRetention(scenarios),
        setLegalHoldFileRetentionAction(scenarios, client),
        confirmSetRetentionPeriodFileRetention(scenarios),
        setRetentionPeriodFileRetentionAction(scenarios, client),
        saveState,
    ],
    initialState,
),
demo: new scenarios.Scenario(
    "S3 Object Locking - Demo",
    [loadState, replAction(scenarios, client)],
    initialState,
),
clean: new scenarios.Scenario(
    "S3 Object Locking - Destroy",
    [
        loadState,
        confirmCleanup(scenarios),
        exitOnFalse(scenarios, "confirmCleanup"),
        cleanupAction(scenarios, client),
    ],
    initialState,
),
};
};

// Call function if run directly
import { fileURLToPath } from "node:url";
import { S3Client } from "@aws-sdk/client-s3";
import { cleanupAction, confirmCleanup } from "./clean.steps.js";
import { replAction } from "./repl.steps.js";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
    const objectLockingScenarios = getWorkflowStages(Scenarios);
    Scenarios.parseScenarioArgs(objectLockingScenarios, {
        name: "Amazon S3 object locking workflow",
        description:
            "Work with Amazon Simple Storage Service (Amazon S3) object locking features.",
        synopsis:

```

```
    "node index.js --scenario <deploy | demo | clean> [-h|--help] [-y|--yes] [-v|--verbose]",
    });
}
```

將歡迎訊息輸出至主控台 (welcome.steps.js)。

```
/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @param {Scenarios} scenarios
 */
const welcome = (scenarios) =>
  new scenarios.ScenarioOutput(
    "welcome",
    "Welcome to the Amazon Simple Storage Service (S3) Object Locking Feature Scenario. For this workflow, we will use the AWS SDK for JavaScript to create several S3 buckets and files to demonstrate working with S3 locking features.",
    { header: true },
  );

/**
 * @param {Scenarios} scenarios
 */
const welcomeContinue = (scenarios) =>
  new scenarios.ScenarioInput(
    "welcomeContinue",
    "Press Enter when you are ready to start.",
    { type: "confirm" },
  );

export { welcome, welcomeContinue };
```

部署儲存貯體、物件和檔案設定 (setup.steps.js)。

```
import {
  BucketVersioningStatus,
  ChecksumAlgorithm,
  CreateBucketCommand,
```

```

    MFADeleteStatus,
    PutBucketVersioningCommand,
    PutObjectCommand,
    PutObjectLockConfigurationCommand,
    PutObjectLegalHoldCommand,
    PutObjectRetentionCommand,
    ObjectLockLegalHoldStatus,
    ObjectLockRetentionMode,
    GetBucketVersioningCommand,
    BucketAlreadyExists,
    BucketAlreadyOwnedByYou,
    S3ServiceException,
    waitUntilBucketExists,
} from "@aws-sdk/client-s3";

import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @typedef {import("@aws-sdk/client-s3").S3Client} S3Client
 */

/**
 * @param {Scenarios} scenarios
 */
const getBucketPrefix = (scenarios) =>
  new scenarios.ScenarioInput(
    "bucketPrefix",
    "Provide a prefix that will be used for bucket creation.",
    { type: "input", default: "amzn-s3-demo-bucket" },
  );

/**
 * @param {Scenarios} scenarios
 */
const createBuckets = (scenarios) =>
  new scenarios.ScenarioOutput(
    "createBuckets",
    (state) => `The following buckets will be created:
      ${state.bucketPrefix}-no-lock with object lock False.
      ${state.bucketPrefix}-lock-enabled with object lock True.
    `
  );

```

```
        `${state.bucketPrefix}-retention-after-creation with object lock False.`,  
        { preformatted: true },  
    );  
  
    /**  
    * @param {Scenarios} scenarios  
    */  
    const confirmCreateBuckets = (scenarios) =>  
        new scenarios.ScenarioInput("confirmCreateBuckets", "Create the buckets?", {  
            type: "confirm",  
        });  
  
    /**  
    * @param {Scenarios} scenarios  
    * @param {S3Client} client  
    */  
    const createBucketsAction = (scenarios, client) =>  
        new scenarios.ScenarioAction("createBucketsAction", async (state) => {  
            const noLockBucketName = `${state.bucketPrefix}-no-lock`;  
            const lockEnabledBucketName = `${state.bucketPrefix}-lock-enabled`;  
            const retentionBucketName = `${state.bucketPrefix}-retention-after-creation`;  
  
            try {  
                await client.send(new CreateBucketCommand({ Bucket: noLockBucketName }));  
                await waitUntilBucketExists({ client }, { Bucket: noLockBucketName });  
                await client.send(  
                    new CreateBucketCommand({  
                        Bucket: lockEnabledBucketName,  
                        ObjectLockEnabledForBucket: true,  
                    }),  
                );  
                await waitUntilBucketExists(  
                    { client },  
                    { Bucket: lockEnabledBucketName },  
                );  
                await client.send(  
                    new CreateBucketCommand({ Bucket: retentionBucketName }),  
                );  
                await waitUntilBucketExists({ client }, { Bucket: retentionBucketName });  
  
                state.noLockBucketName = noLockBucketName;  
                state.lockEnabledBucketName = lockEnabledBucketName;  
                state.retentionBucketName = retentionBucketName;  
            } catch (caught) {
```



```

        if (
            caught instanceof BucketAlreadyExists ||
            caught instanceof BucketAlreadyOwnedByYou
        ) {
            console.error(`${caught.name}: ${caught.message}`);
            state.earlyExit = true;
        } else {
            throw caught;
        }
    }
});

/**
 * @param {Scenarios} scenarios
 */
const populateBuckets = (scenarios) =>
    new scenarios.ScenarioOutput(
        "populateBuckets",
        (state) => `The following test files will be created:
            file0.txt in ${state.bucketPrefix}-no-lock.
            file1.txt in ${state.bucketPrefix}-no-lock.
            file0.txt in ${state.bucketPrefix}-lock-enabled.
            file1.txt in ${state.bucketPrefix}-lock-enabled.
            file0.txt in ${state.bucketPrefix}-retention-after-creation.
            file1.txt in ${state.bucketPrefix}-retention-after-creation.`
        ,
        { preformatted: true },
    );

/**
 * @param {Scenarios} scenarios
 */
const confirmPopulateBuckets = (scenarios) =>
    new scenarios.ScenarioInput(
        "confirmPopulateBuckets",
        "Populate the buckets?",
        { type: "confirm" },
    );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const populateBucketsAction = (scenarios, client) =>
    new scenarios.ScenarioAction("populateBucketsAction", async (state) => {

```

```
try {
    await client.send(
        new PutObjectCommand({
            Bucket: state.noLockBucketName,
            Key: "file0.txt",
            Body: "Content",
            ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
        })),
    );
    await client.send(
        new PutObjectCommand({
            Bucket: state.noLockBucketName,
            Key: "file1.txt",
            Body: "Content",
            ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
        })),
    );
    await client.send(
        new PutObjectCommand({
            Bucket: state.lockEnabledBucketName,
            Key: "file0.txt",
            Body: "Content",
            ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
        })),
    );
    await client.send(
        new PutObjectCommand({
            Bucket: state.lockEnabledBucketName,
            Key: "file1.txt",
            Body: "Content",
            ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
        })),
    );
    await client.send(
        new PutObjectCommand({
            Bucket: state.retentionBucketName,
            Key: "file0.txt",
            Body: "Content",
            ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
        })),
    );
    await client.send(
        new PutObjectCommand({
            Bucket: state.retentionBucketName,
```

```

        Key: "file1.txt",
        Body: "Content",
        ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
    )),
    );
} catch (caught) {
    if (caught instanceof S3ServiceException) {
        console.error(
            `Error from S3 while uploading object.  ${caught.name}:
${caught.message}`,
        );
    } else {
        throw caught;
    }
}
});

/**
 * @param {Scenarios} scenarios
 */
const updateRetention = (scenarios) =>
    new scenarios.ScenarioOutput(
        "updateRetention",
        (state) => `A bucket can be configured to use object locking with a default
retention period.
A default retention period will be configured for ${state.bucketPrefix}-
retention-after-creation.` ,
        { preformatted: true },
    );

/**
 * @param {Scenarios} scenarios
 */
const confirmUpdateRetention = (scenarios) =>
    new scenarios.ScenarioInput(
        "confirmUpdateRetention",
        "Configure default retention period?",
        { type: "confirm" },
    );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */

```

```

const updateRetentionAction = (scenarios, client) =>
  new scenarios.ScenarioAction("updateRetentionAction", async (state) => {
    await client.send(
      new PutBucketVersioningCommand({
        Bucket: state.retentionBucketName,
        VersioningConfiguration: {
          MFADelete: MFADeleteStatus.Disabled,
          Status: BucketVersioningStatus.Enabled,
        },
      }),
    );

    const getBucketVersioning = new GetBucketVersioningCommand({
      Bucket: state.retentionBucketName,
    });

    await retry({ intervalInMs: 500, maxRetries: 10 }, async () => {
      const { Status } = await client.send(getBucketVersioning);
      if (Status !== "Enabled") {
        throw new Error("Bucket versioning is not enabled.");
      }
    });

    await client.send(
      new PutObjectLockConfigurationCommand({
        Bucket: state.retentionBucketName,
        ObjectLockConfiguration: {
          ObjectLockEnabled: "Enabled",
          Rule: {
            DefaultRetention: {
              Mode: "GOVERNANCE",
              Years: 1,
            },
          },
        },
      }),
    );
  });

/**
 * @param {Scenarios} scenarios
 */
const updateLockPolicy = (scenarios) =>
  new scenarios.ScenarioOutput(

```

```

    "updateLockPolicy",
    (state) => `Object lock policies can also be added to existing buckets.
An object lock policy will be added to ${state.bucketPrefix}-lock-enabled.` ,
    { preformatted: true },
  );

/**
 * @param {Scenarios} scenarios
 */
const confirmUpdateLockPolicy = (scenarios) =>
  new scenarios.ScenarioInput(
    "confirmUpdateLockPolicy",
    "Add object lock policy?",
    { type: "confirm" },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const updateLockPolicyAction = (scenarios, client) =>
  new scenarios.ScenarioAction("updateLockPolicyAction", async (state) => {
    await client.send(
      new PutObjectLockConfigurationCommand({
        Bucket: state.lockEnabledBucketName,
        ObjectLockConfiguration: {
          ObjectLockEnabled: "Enabled",
        },
      }),
    );
  });

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const confirmSetLegalHoldFileEnabled = (scenarios) =>
  new scenarios.ScenarioInput(
    "confirmSetLegalHoldFileEnabled",
    (state) =>
      `Would you like to add a legal hold to file0.txt in
${state.lockEnabledBucketName}?`,
    {
      type: "confirm",
    },
  );

```

```

    },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const setLegalHoldFileEnabledAction = (scenarios, client) =>
  new scenarios.ScenarioAction(
    "setLegalHoldFileEnabledAction",
    async (state) => {
      await client.send(
        new PutObjectLegalHoldCommand({
          Bucket: state.lockEnabledBucketName,
          Key: "file0.txt",
          LegalHold: {
            Status: ObjectLockLegalHoldStatus.ON,
          },
        }),
      );
      console.log(
        `Modified legal hold for file0.txt in ${state.lockEnabledBucketName}.`,
      );
    },
    { skipWhen: (state) => !state.confirmSetLegalHoldFileEnabled },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const confirmSetRetentionPeriodFileEnabled = (scenarios) =>
  new scenarios.ScenarioInput(
    "confirmSetRetentionPeriodFileEnabled",
    (state) =>
      `Would you like to add a 1 day Governance retention period to file1.txt in ${state.lockEnabledBucketName}?
Reminder: Only a user with the s3:BypassGovernanceRetention permission will be able to delete this file or its bucket until the retention period has expired.`,
    {
      type: "confirm",
    },
  );

```

```

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const setRetentionPeriodFileEnabledAction = (scenarios, client) =>
  new scenarios.ScenarioAction(
    "setRetentionPeriodFileEnabledAction",
    async (state) => {
      const retentionDate = new Date();
      retentionDate.setDate(retentionDate.getDate() + 1);
      await client.send(
        new PutObjectRetentionCommand({
          Bucket: state.lockEnabledBucketName,
          Key: "file1.txt",
          Retention: {
            Mode: ObjectLockRetentionMode.GOVERNANCE,
            RetainUntilDate: retentionDate,
          },
        }),
      );
      console.log(
        `Set retention for file1.txt in ${state.lockEnabledBucketName} until
        ${retentionDate.toISOString().split("T")[0]}.`,
      );
    },
    { skipWhen: (state) => !state.confirmSetRetentionPeriodFileEnabled },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const confirmSetLegalHoldFileRetention = (scenarios) =>
  new scenarios.ScenarioInput(
    "confirmSetLegalHoldFileRetention",
    (state) =>
      `Would you like to add a legal hold to file0.txt in
      ${state.retentionBucketName}?`,
    {
      type: "confirm",
    },
  );

/**

```

```

    * @param {Scenarios} scenarios
    * @param {S3Client} client
    */
const setLegalHoldFileRetentionAction = (scenarios, client) =>
  new scenarios.ScenarioAction(
    "setLegalHoldFileRetentionAction",
    async (state) => {
      await client.send(
        new PutObjectLegalHoldCommand({
          Bucket: state.retentionBucketName,
          Key: "file0.txt",
          LegalHold: {
            Status: ObjectLockLegalHoldStatus.ON,
          },
        }),
      );
      console.log(
        `Modified legal hold for file0.txt in ${state.retentionBucketName}.`,
      );
    },
    { skipWhen: (state) => !state.confirmSetLegalHoldFileRetention },
  );

/**
 * @param {Scenarios} scenarios
 */
const confirmSetRetentionPeriodFileRetention = (scenarios) =>
  new scenarios.ScenarioInput(
    "confirmSetRetentionPeriodFileRetention",
    (state) =>
      `Would you like to add a 1 day Governance retention period to file1.txt in
      ${state.retentionBucketName}?
      Reminder: Only a user with the s3:BypassGovernanceRetention permission will be
      able to delete this file or its bucket until the retention period has expired.`,
    {
      type: "confirm",
    },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const setRetentionPeriodFileRetentionAction = (scenarios, client) =>

```



```
new scenarios.ScenarioAction(
  "setRetentionPeriodFileRetentionAction",
  async (state) => {
    const retentionDate = new Date();
    retentionDate.setDate(retentionDate.getDate() + 1);
    await client.send(
      new PutObjectRetentionCommand({
        Bucket: state.retentionBucketName,
        Key: "file1.txt",
        Retention: {
          Mode: ObjectLockRetentionMode.GOVERNANCE,
          RetainUntilDate: retentionDate,
        },
        BypassGovernanceRetention: true,
      }),
    );
    console.log(
      `Set retention for file1.txt in ${state.retentionBucketName} until
      ${retentionDate.toISOString().split("T")[0]}.`,
    );
  },
  { skipWhen: (state) => !state.confirmSetRetentionPeriodFileRetention },
);

export {
  getBucketPrefix,
  createBuckets,
  confirmCreateBuckets,
  createBucketsAction,
  populateBuckets,
  confirmPopulateBuckets,
  populateBucketsAction,
  updateRetention,
  confirmUpdateRetention,
  updateRetentionAction,
  updateLockPolicy,
  confirmUpdateLockPolicy,
  updateLockPolicyAction,
  confirmSetLegalHoldFileEnabled,
  setLegalHoldFileEnabledAction,
  confirmSetRetentionPeriodFileEnabled,
  setRetentionPeriodFileEnabledAction,
  confirmSetLegalHoldFileRetention,
  setLegalHoldFileRetentionAction,
```

```
confirmSetRetentionPeriodFileRetention,  
setRetentionPeriodFileRetentionAction,  
};
```

檢視和刪除儲存貯體中的檔案 (repl.steps.js)。

```
import {  
  ChecksumAlgorithm,  
  DeleteObjectCommand,  
  GetObjectLegalHoldCommand,  
  GetObjectLockConfigurationCommand,  
  GetObjectRetentionCommand,  
  ListObjectVersionsCommand,  
  PutObjectCommand,  
} from "@aws-sdk/client-s3";  
  
/**  
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios  
 */  
  
/**  
 * @typedef {import("@aws-sdk/client-s3").S3Client} S3Client  
 */  
  
const choices = {  
  EXIT: 0,  
  LIST_ALL_FILES: 1,  
  DELETE_FILE: 2,  
  DELETE_FILE_WITH_RETENTION: 3,  
  OVERWRITE_FILE: 4,  
  VIEW_RETENTION_SETTINGS: 5,  
  VIEW_LEGAL_HOLD_SETTINGS: 6,  
};  
  
/**  
 * @param {Scenarios} scenarios  
 */  
const replInput = (scenarios) =>  
  new scenarios.ScenarioInput(  
    "replChoice",  
    "Explore the S3 locking features by selecting one of the following choices",  
    {
```

```

    type: "select",
    choices: [
      { name: "List all files in buckets", value: choices.LIST_ALL_FILES },
      { name: "Attempt to delete a file.", value: choices.DELETE_FILE },
      {
        name: "Attempt to delete a file with retention period bypass.",
        value: choices.DELETE_FILE_WITH_RETENTION,
      },
      { name: "Attempt to overwrite a file.", value: choices.OVERWRITE_FILE },
      {
        name: "View the object and bucket retention settings for a file.",
        value: choices.VIEW_RETENTION_SETTINGS,
      },
      {
        name: "View the legal hold settings for a file.",
        value: choices.VIEW_LEGAL_HOLD_SETTINGS,
      },
      { name: "Finish the workflow.", value: choices.EXIT },
    ],
  },
);

/**
 * @param {S3Client} client
 * @param {string[]} buckets
 */
const getAllFiles = async (client, buckets) => {
  /** @type {{bucket: string, key: string, version: string}[]} */
  const files = [];
  for (const bucket of buckets) {
    const objectsResponse = await client.send(
      new ListObjectVersionsCommand({ Bucket: bucket }),
    );
    for (const version of objectsResponse.Versions || []) {
      const { Key, VersionId } = version;
      files.push({ bucket, key: Key, version: VersionId });
    }
  }

  return files;
};

/**
 * @param {Scenarios} scenarios

```

```
* @param {S3Client} client
*/
const replAction = (scenarios, client) =>
  new scenarios.ScenarioAction(
    "replAction",
    async (state) => {
      const files = await getAllFiles(client, [
        state.noLockBucketName,
        state.lockEnabledBucketName,
        state.retentionBucketName,
      ]);

      const fileInput = new scenarios.ScenarioInput(
        "selectedFile",
        "Select a file:",
        {
          type: "select",
          choices: files.map((file, index) => ({
            name: `${index + 1}: ${file.bucket}: ${file.key} (version: ${
              file.version
            })`,
            value: index,
          })),
        },
      );

      const { replChoice } = state;

      switch (replChoice) {
        case choices.LIST_ALL_FILES: {
          const files = await getAllFiles(client, [
            state.noLockBucketName,
            state.lockEnabledBucketName,
            state.retentionBucketName,
          ]);
          state.replOutput = files
            .map(
              (file) =>
                `${file.bucket}: ${file.key} (version: ${file.version})`,
            )
            .join("\n");
          break;
        }
        case choices.DELETE_FILE: {
```

```

    /** @type {number} */
    const fileToDelete = await fileInput.handle(state);
    const selectedFile = files[fileToDelete];
    try {
      await client.send(
        new DeleteObjectCommand({
          Bucket: selectedFile.bucket,
          Key: selectedFile.key,
          VersionId: selectedFile.version,
        }),
      );
      state.replOutput = `Deleted ${selectedFile.key} in
${selectedFile.bucket}.`;
    } catch (err) {
      state.replOutput = `Unable to delete object ${selectedFile.key} in
bucket ${selectedFile.bucket}: ${err.message}`;
    }
    break;
  }
  case choices.DELETE_FILE_WITH_RETENTION: {
    /** @type {number} */
    const fileToDelete = await fileInput.handle(state);
    const selectedFile = files[fileToDelete];
    try {
      await client.send(
        new DeleteObjectCommand({
          Bucket: selectedFile.bucket,
          Key: selectedFile.key,
          VersionId: selectedFile.version,
          BypassGovernanceRetention: true,
        }),
      );
      state.replOutput = `Deleted ${selectedFile.key} in
${selectedFile.bucket}.`;
    } catch (err) {
      state.replOutput = `Unable to delete object ${selectedFile.key} in
bucket ${selectedFile.bucket}: ${err.message}`;
    }
    break;
  }
  case choices.OVERWRITE_FILE: {
    /** @type {number} */
    const fileToOverwrite = await fileInput.handle(state);
    const selectedFile = files[fileToOverwrite];

```

```

    try {
      await client.send(
        new PutObjectCommand({
          Bucket: selectedFile.bucket,
          Key: selectedFile.key,
          Body: "New content",
          ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
        })),
    );
    state.replOutput = `Overwrote ${selectedFile.key} in
${selectedFile.bucket}.`;
  } catch (err) {
    state.replOutput = `Unable to overwrite object ${selectedFile.key} in
bucket ${selectedFile.bucket}: ${err.message}`;
  }
  break;
}

case choices.VIEW_RETENTION_SETTINGS: {
  /** @type {number} */
  const fileToView = await fileInput.handle(state);
  const selectedFile = files[fileToView];
  try {
    const retention = await client.send(
      new GetObjectRetentionCommand({
        Bucket: selectedFile.bucket,
        Key: selectedFile.key,
        VersionId: selectedFile.version,
      })),
    );
    const bucketConfig = await client.send(
      new GetObjectLockConfigurationCommand({
        Bucket: selectedFile.bucket,
      })),
    );
    state.replOutput = `Object retention for ${selectedFile.key}
in ${selectedFile.bucket}: ${retention.Retention?.Mode} until
${retention.Retention?.RetainUntilDate?.toISOString()}.
Bucket object lock config for ${selectedFile.bucket} in ${selectedFile.bucket}:
Enabled: ${bucketConfig.ObjectLockConfiguration?.ObjectLockEnabled}
Rule:
${JSON.stringify(bucketConfig.ObjectLockConfiguration?.Rule?.DefaultRetention)}`;
  } catch (err) {
    state.replOutput = `Unable to fetch object lock retention:
'${err.message}'`;
  }
}

```

```

    }
    break;
  }
  case choices.VIEW_LEGAL_HOLD_SETTINGS: {
    /** @type {number} */
    const fileToView = await fileInput.handle(state);
    const selectedFile = files[fileToView];
    try {
      const legalHold = await client.send(
        new GetObjectLegalHoldCommand({
          Bucket: selectedFile.bucket,
          Key: selectedFile.key,
          VersionId: selectedFile.version,
        }),
      );
      state.replOutput = `Object legal hold for ${selectedFile.key} in
${selectedFile.bucket}: Status: ${legalHold.LegalHold?.Status}`;
    } catch (err) {
      state.replOutput = `Unable to fetch legal hold: '${err.message}'`;
    }
    break;
  }
  default:
    throw new Error(`Invalid replChoice: ${replChoice}`);
}
},
{
  whileConfig: {
    whileFn: ({ replChoice }) => replChoice !== choices.EXIT,
    input: replInput(scenarios),
    output: new scenarios.ScenarioOutput(
      "REPL output",
      (state) => state.replOutput,
      { preformatted: true },
    ),
  },
},
);

export { replInput, replAction, choices };

```

銷毀所有建立的資源 (clean.steps.js)。

```
import {
  DeleteObjectCommand,
  DeleteBucketCommand,
  ListObjectVersionsCommand,
  GetObjectLegalHoldCommand,
  GetObjectRetentionCommand,
  PutObjectLegalHoldCommand,
} from "@aws-sdk/client-s3";

/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @typedef {import("@aws-sdk/client-s3").S3Client} S3Client
 */

/**
 * @param {Scenarios} scenarios
 */
const confirmCleanup = (scenarios) =>
  new scenarios.ScenarioInput("confirmCleanup", "Clean up resources?", {
    type: "confirm",
  });

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const cleanupAction = (scenarios, client) =>
  new scenarios.ScenarioAction("cleanupAction", async (state) => {
    const { noLockBucketName, lockEnabledBucketName, retentionBucketName } =
      state;

    const buckets = [
      noLockBucketName,
      lockEnabledBucketName,
      retentionBucketName,
    ];

    for (const bucket of buckets) {
      /** @type {import("@aws-sdk/client-s3").ListObjectVersionsCommandOutput} */
      let objectsResponse;
```



```
try {
  objectsResponse = await client.send(
    new ListObjectVersionsCommand({
      Bucket: bucket,
    }),
  );
} catch (e) {
  if (e instanceof Error && e.name === "NoSuchBucket") {
    console.log("Object's bucket has already been deleted.");
    continue;
  }
  throw e;
}

for (const version of objectsResponse.Versions || []) {
  const { Key, VersionId } = version;

  try {
    const legalHold = await client.send(
      new GetObjectLegalHoldCommand({
        Bucket: bucket,
        Key,
        VersionId,
      }),
    );

    if (legalHold.LegalHold?.Status === "ON") {
      await client.send(
        new PutObjectLegalHoldCommand({
          Bucket: bucket,
          Key,
          VersionId,
          LegalHold: {
            Status: "OFF",
          },
        }),
      );
    }
  } catch (err) {
    console.log(
      `Unable to fetch legal hold for ${Key} in ${bucket}:` +
      `${err.message}`,
    );
  }
}
```

```
    }

    try {
      const retention = await client.send(
        new GetObjectRetentionCommand({
          Bucket: bucket,
          Key,
          VersionId,
        })),
    );

    if (retention.Retention?.Mode === "GOVERNANCE") {
      await client.send(
        new DeleteObjectCommand({
          Bucket: bucket,
          Key,
          VersionId,
          BypassGovernanceRetention: true,
        })),
    );
    }
  } catch (err) {
    console.log(
      `Unable to fetch object lock retention for ${Key} in ${bucket}:`
      + `${err.message}`,
    );
  }

  await client.send(
    new DeleteObjectCommand({
      Bucket: bucket,
      Key,
      VersionId,
    })),
  );
}

await client.send(new DeleteBucketCommand({ Bucket: bucket }));
console.log(`Delete for ${bucket} complete.`);
}
});

export { confirmCleanup, cleanupAction };
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的下列主題。
 - [GetObjectLegalHold](#)
 - [GetObjectLockConfiguration](#)
 - [GetObjectRetention](#)
 - [PutObjectLegalHold](#)
 - [PutObjectLockConfiguration](#)
 - [PutObjectRetention](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 提出 Amazon S3 條件式請求 AWS

下列程式碼範例示範如何將先決條件新增至 Amazon S3 請求。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

執行展示 Amazon S3 條件式請求功能的互動式情境。

```
using Amazon.S3;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using Microsoft.Extensions.Logging;
using Microsoft.Extensions.Logging.Console;
using Microsoft.Extensions.Logging.Debug;

namespace S3ConditionalRequestsScenario;
```

```
public static class S3ConditionalRequestsScenario
{
    /*
        Before running this .NET code example, set up your development environment,
        including your credentials.

        This example demonstrates the use of conditional requests for S3 operations.
        You can use conditional requests to add preconditions to S3 read requests to
        return or copy
        an object based on its Entity tag (ETag), or last modified date.
        You can use a conditional write requests to prevent overwrites by ensuring
        there is no existing object with the same key.
    */

    public static S3ActionsWrapper _s3ActionsWrapper = null!;
    public static IConfiguration _configuration = null!;
    public static string _resourcePrefix = null!;
    public static string _sourceBucketName = null!;
    public static string _destinationBucketName = null!;
    public static string _sampleObjectKey = null!;
    public static string _sampleObjectEtag = null!;
    public static bool _interactive = true;

    public static async Task Main(string[] args)
    {
        // Set up dependency injection for the Amazon service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
                        LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
                        LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonS3>()
                    .AddTransient<S3ActionsWrapper>()
            )
            .Build();

        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
    }
}
```

```

        .AddJsonFile("settings.local.json",
            true) // Optionally, load local settings.
        .Build();

    ServicesSetup(host);

    try
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Welcome to the Amazon Simple Storage Service (S3)
Conditional Requests Feature Scenario.");
        Console.WriteLine(new string('-', 80));
        ConfigurationSetup();
        _sampleObjectEtag = await Setup(_sourceBucketName,
            _destinationBucketName, _sampleObjectKey);

        await DisplayDemoChoices(_sourceBucketName, _destinationBucketName,
            _sampleObjectKey, _sampleObjectEtag, 0);

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Cleaning up resources.");
        Console.WriteLine(new string('-', 80));
        await Cleanup(true);

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Amazon S3 Conditional Requests Feature Scenario is
complete.");
        Console.WriteLine(new string('-', 80));
    }
    catch (Exception ex)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine($"There was a problem: {ex.Message}");
        await CleanupScenario(_sourceBucketName, _destinationBucketName);
        Console.WriteLine(new string('-', 80));
    }
}

/// <summary>
/// Populate the services for use within the console application.
/// </summary>
/// <param name="host">The services host.</param>
private static void ServicesSetup(IHost host)
{

```

```

        _s3ActionsWrapper = host.Services.GetRequiredService<S3ActionsWrapper>();
    }

    /// <summary>
    /// Any setup operations needed.
    /// </summary>
    public static void ConfigurationSetup()
    {
        _resourcePrefix = _configuration["resourcePrefix"] ?? "dotnet-example";

        _sourceBucketName = _resourcePrefix + "-source";
        _destinationBucketName = _resourcePrefix + "-dest";
        _sampleObjectKey = _resourcePrefix + "-sample-object.txt";
    }

    /// <summary>
    /// Sets up the scenario by creating a source and destination bucket, and
    uploading a test file to the source bucket.
    /// </summary>
    /// <param name="sourceBucket">The name of the source bucket.</param>
    /// <param name="destBucket">The name of the destination bucket.</param>
    /// <param name="objectKey">The name of the test file to add to the source
    bucket.</param>
    /// <returns>The ETag of the uploaded test file.</returns>
    public static async Task<string> Setup(string sourceBucket, string
    destBucket, string objectKey)
    {
        Console.WriteLine(
            "\nFor this scenario, we will use the AWS SDK for .NET to create
            several S3\n" +
            "buckets and files to demonstrate working with S3 conditional
            requests.\n" +
            "This example demonstrates the use of conditional requests for S3
            operations.\r\n" +
            "You can use conditional requests to add preconditions to S3 read
            requests to return or copy\r\n" +
            "an object based on its Entity tag (ETag), or last modified date. \r
            \n" +
            "You can use a conditional write requests to prevent overwrites by
            ensuring \r\n" +
            "there is no existing object with the same key. \r\n\r\n" +
            "This example will allow you to perform conditional reads\r\n" +
            "and writes that will succeed or fail based on your selected options.
            \r\n\r\n" +

```

```
        "Sample buckets and a sample object will be created as part of the
example.");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you are ready to start.");
        if (_interactive)
            Console.ReadLine();

        await _s3ActionsWrapper.CreateBucketWithName(sourceBucket);
        await _s3ActionsWrapper.CreateBucketWithName(destBucket);

        var eTag = await _s3ActionsWrapper.PutObjectConditional(objectKey,
sourceBucket,
        "Test file content.");

        return eTag;
    }

    /// <summary>
    /// Cleans up the scenario by deleting the source and destination buckets.
    /// </summary>
    /// <param name="sourceBucket">The name of the source bucket.</param>
    /// <param name="destBucket">The name of the destination bucket.</param>
    public static async Task CleanupScenario(string sourceBucket, string
destBucket)
    {
        await _s3ActionsWrapper.CleanupBucketByName(sourceBucket);
        await _s3ActionsWrapper.CleanupBucketByName(destBucket);
    }

    /// <summary>
    /// Displays a list of the objects in the test buckets.
    /// </summary>
    /// <param name="sourceBucket">The name of the source bucket.</param>
    /// <param name="destBucket">The name of the destination bucket.</param>
    public static async Task DisplayBuckets(string sourceBucket, string
destBucket)
    {
        await _s3ActionsWrapper.ListBucketContentsByName(sourceBucket);
        await _s3ActionsWrapper.ListBucketContentsByName(destBucket);
    }

    /// <summary>
    /// Displays the menu of conditional request options for the user.
```

```

    /// </summary>
    /// <param name="sourceBucket">The name of the source bucket.</param>
    /// <param name="destBucket">The name of the destination bucket.</param>
    /// <param name="objectKey">The key of the test object in the source
    bucket.</param>
    /// <param name="etag">The ETag of the test object in the source bucket.</
    param>
    public static async Task DisplayDemoChoices(string sourceBucket, string
    destBucket, string objectKey, string etag, int defaultChoice)
    {
        var actions = new[]
        {
            "Print a list of bucket items.",
            "Perform a conditional read.",
            "Perform a conditional copy.",
            "Perform a conditional write.",
            "Clean up and exit."
        };

        var conditions = new[]
        {
            "If-Match: using the object's ETag. This condition should succeed.",
            "If-None-Match: using the object's ETag. This condition should
fail.",
            "If-Modified-Since: using yesterday's date. This condition should
succeed.",
            "If-Unmodified-Since: using yesterday's date. This condition should
fail."
        };

        var conditionTypes = new[]
        {
            S3ConditionType.IfMatch,
            S3ConditionType.IfNoneMatch,
            S3ConditionType.IfModifiedSince,
            S3ConditionType.IfUnmodifiedSince,
        };

        var yesterdayDate = DateTime.UtcNow.AddDays(-1);

        int choice;
        while ((choice = GetChoiceResponse("\nExplore the S3 conditional request
features by selecting one of the following choices:", actions, defaultChoice)) != 4)

```



```
{
    switch (choice)
    {
        case 0:
            Console.WriteLine("Listing the objects and buckets.");
            await DisplayBuckets(sourceBucket, destBucket);
            break;
        case 1:
            int conditionTypeIndex = GetChoiceResponse("Perform a
conditional read:", conditions, 1);
            if (conditionTypeIndex == 0 || conditionTypeIndex == 1)
            {
                await _s3ActionsWrapper.GetObjectConditional(objectKey,
sourceBucket, conditionTypes[conditionTypeIndex], null, _sampleObjectEtag);
            }
            else if (conditionTypeIndex == 2 || conditionTypeIndex == 3)
            {
                await _s3ActionsWrapper.GetObjectConditional(objectKey,
sourceBucket, conditionTypes[conditionTypeIndex], yesterdayDate);
            }
            break;
        case 2:
            int copyConditionTypeIndex = GetChoiceResponse("Perform a
conditional copy:", conditions, 1);
            string destKey = GetStringResponse("Enter an object key:",
"sampleObjectKey");
            if (copyConditionTypeIndex == 0 || copyConditionTypeIndex ==
1)
            {
                await _s3ActionsWrapper.CopyObjectConditional(objectKey,
destKey, sourceBucket, destBucket, conditionTypes[copyConditionTypeIndex], null,
etag);
            }
            else if (copyConditionTypeIndex == 2 ||
copyConditionTypeIndex == 3)
            {
                await _s3ActionsWrapper.CopyObjectConditional(objectKey,
destKey, sourceBucket, destBucket, conditionTypes[copyConditionTypeIndex],
yesterdayDate);
            }
            break;
        case 3:
            Console.WriteLine("Perform a conditional write using
IfNoneMatch condition on the object key.");
```

```

        Console.WriteLine("If the key is a duplicate, the write will
fail.");

        string newObjectKey = GetStringResponse("Enter an object
key:", "newObjectKey");
        await _s3ActionsWrapper.PutObjectConditional(newObjectKey,
sourceBucket, "Conditional write example data.");
        break;
    }

    if (!_interactive)
    {
        break;
    }
}

Console.WriteLine("Proceeding to cleanup.");
}

// <summary>
/// Clean up the resources from the scenario.
/// </summary>
/// <param name="interactive">True to run as interactive.</param>
/// <returns>True if successful.</returns>
public static async Task<bool> Cleanup(bool interactive)
{
    Console.WriteLine(new string('-', 80));

    if (!interactive || GetYesNoResponse("Do you want to clean up all files
and buckets? (y/n) "))
    {
        await _s3ActionsWrapper.CleanUpBucketByName(_sourceBucketName);
        await _s3ActionsWrapper.CleanUpBucketByName(_destinationBucketName);

    }
    else
    {
        Console.WriteLine(
            "Ok, we'll leave the resources intact.\n" +
            "Don't forget to delete them when you're done with them or you
might incur unexpected charges."
        );
    }

    Console.WriteLine(new string('-', 80));
}

```

```
        return true;
    }

    /// <summary>
    /// Helper method to get a yes or no response from the user.
    /// </summary>
    /// <param name="question">The question string to print on the console.</
param>
    /// <returns>True if the user responds with a yes.</returns>
    private static bool GetYesNoResponse(string question)
    {
        Console.WriteLine(question);
        var ynResponse = Console.ReadLine();
        var response = ynResponse != null && ynResponse.Equals("y",
StringComparison.InvariantCultureIgnoreCase);
        return response;
    }

    /// <summary>
    /// Helper method to get a choice response from the user.
    /// </summary>
    /// <param name="question">The question string to print on the console.</
param>
    /// <param name="choices">The choices to print on the console.</param>
    /// <returns>The index of the selected choice</returns>
    private static int GetChoiceResponse(string? question, string[] choices, int
defaultChoice)
    {
        if (question != null)
        {
            Console.WriteLine(question);

            for (int i = 0; i < choices.Length; i++)
            {
                Console.WriteLine($"{i + 1}. {choices[i]}");
            }
        }

        if (!_interactive)
            return defaultChoice;

        var choiceNumber = 0;
        while (choiceNumber < 1 || choiceNumber > choices.Length)
        {
```

```
        var choice = Console.ReadLine();
        Int32.TryParse(choice, out choiceNumber);
    }

    return choiceNumber - 1;
}

/// <summary>
/// Get a string response from the user.
/// </summary>
/// <param name="question">The question to print.</param>
/// <param name="defaultAnswer">A default answer to use when not
interactive.</param>
/// <returns>The string response.</returns>
public static string GetStringResponse(string? question, string
defaultAnswer)
{
    string? answer = "";
    if (_interactive)
    {
        do
        {
            Console.WriteLine(question);
            answer = Console.ReadLine();
        } while (string.IsNullOrEmpty(answer));
    }
    else
    {
        answer = defaultAnswer;
    }

    return answer;
}
}
```

S3 函數的包裝函式類別。

```
using System.Net;
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Logging;
```

```

namespace S3ConditionalRequestsScenario;

/// <summary>
/// Encapsulate the Amazon S3 operations.
/// </summary>
public class S3ActionsWrapper
{
    private readonly IAmazonS3 _amazonS3;
    private readonly ILogger<S3ActionsWrapper> _logger;

    /// <summary>
    /// Constructor for the S3ActionsWrapper.
    /// </summary>
    /// <param name="amazonS3">The injected S3 client.</param>
    /// <param name="logger">The class logger.</param>
    public S3ActionsWrapper(IAmazonS3 amazonS3, ILogger<S3ActionsWrapper> logger)
    {
        _amazonS3 = amazonS3;
        _logger = logger;
    }

    /// <summary>
    /// Retrieves an object from Amazon S3 with a conditional request.
    /// </summary>
    /// <param name="objectKey">The key of the object to retrieve.</param>
    /// <param name="sourceBucket">The source bucket of the object.</param>
    /// <param name="conditionType">The type of condition: 'IfMatch',
    'IfNoneMatch', 'IfModifiedSince', 'IfUnmodifiedSince'.</param>
    /// <param name="conditionDateValue">The value to use for the condition for
    dates.</param>
    /// <param name="etagConditionalValue">The value to use for the condition for
    etags.</param>
    /// <returns>True if the conditional read is successful, False otherwise.</
returns>
    public async Task<bool> GetObjectConditional(string objectKey, string
sourceBucket,
        S3ConditionType conditionType, DateTime? conditionDateValue = null,
string? etagConditionalValue = null)
    {
        try
        {
            var getObjectRequest = new GetObjectRequest
            {

```

```
        BucketName = sourceBucket,
        Key = objectKey
    };

    switch (conditionType)
    {
        case S3ConditionType.IfMatch:
            getObjectRequest.EtagToMatch = etagConditionalValue;
            break;
        case S3ConditionType.IfNoneMatch:
            getObjectRequest.EtagToNotMatch = etagConditionalValue;
            break;
        case S3ConditionType.IfModifiedSince:
            getObjectRequest.ModifiedSinceDateUtc =
conditionDateValue.GetValueOrDefault();
            break;
        case S3ConditionType.IfUnmodifiedSince:
            getObjectRequest.UnmodifiedSinceDateUtc =
conditionDateValue.GetValueOrDefault();
            break;
        default:
            throw new ArgumentOutOfRangeException(nameof(conditionType),
conditionType, null);
    }

    var response = await _amazonS3.GetObjectAsync(getObjectRequest);
    var sampleBytes = new byte[20];
    await response.ResponseStream.ReadAsync(sampleBytes, 0, 20);
    _logger.LogInformation($"Conditional read
successful. Here are the first 20 bytes of the object:
\n{System.Text.Encoding.UTF8.GetString(sampleBytes)}");
    return true;
}

catch (AmazonS3Exception e)
{
    if (e.ErrorCode == "PreconditionFailed")
    {
        _logger.LogError("Conditional read failed: Precondition failed");
    }
    else if (e.ErrorCode == "NotModified")
    {
        _logger.LogError("Conditional read failed: Object not modified");
    }
    else
```

```

        {
            _logger.LogError($"Unexpected error: {e.ErrorCode}");
            throw;
        }
        return false;
    }
}

/// <summary>
/// Uploads an object to Amazon S3 with a conditional request. Prevents
overwrite using an IfNoneMatch condition for the object key.
/// </summary>
/// <param name="objectKey">The key of the object to upload.</param>
/// <param name="bucket">The source bucket of the object.</param>
/// <param name="content">The content to upload as a string.</param>
/// <returns>The ETag if the conditional write is successful, empty
otherwise.</returns>
public async Task<string> PutObjectConditional(string objectKey, string
bucket, string content)
{
    try
    {
        var putObjectRequest = new PutObjectRequest
        {
            BucketName = bucket,
            Key = objectKey,
            ContentBody = content,
            IfNoneMatch = "*"
        };

        var putResult = await _amazonS3.PutObjectAsync(putObjectRequest);
        _logger.LogInformation($"Conditional write successful for key
{objectKey} in bucket {bucket}.");
        return putResult.ETag;
    }
    catch (AmazonS3Exception e)
    {
        if (e.ErrorCode == "PreconditionFailed")
        {
            _logger.LogError("Conditional write failed: Precondition
failed");
        }
        else
        {

```

```

        _logger.LogError($"Unexpected error: {e.ErrorCode}");
        throw;
    }
    return string.Empty;
}
}

/// <summary>
/// Copies an object from one Amazon S3 bucket to another with a conditional
request.
/// </summary>
/// <param name="sourceKey">The key of the source object to copy.</param>
/// <param name="destKey">The key of the destination object.</param>
/// <param name="sourceBucket">The source bucket of the object.</param>
/// <param name="destBucket">The destination bucket of the object.</param>
/// <param name="conditionType">The type of condition to apply, e.g.
'CopySourceIfMatch', 'CopySourceIfNoneMatch', 'CopySourceIfModifiedSince',
'CopySourceIfUnmodifiedSince'.</param>
/// <param name="conditionDateValue">The value to use for the condition for
dates.</param>
/// <param name="etagConditionalValue">The value to use for the condition for
etags.</param>
/// <returns>True if the conditional copy is successful, False otherwise.</
returns>
public async Task<bool> CopyObjectConditional(string sourceKey, string
destKey, string sourceBucket, string destBucket,
    S3ConditionType conditionType, DateTime? conditionDateValue = null,
string? etagConditionalValue = null)
{
    try
    {
        var copyObjectRequest = new CopyObjectRequest
        {
            DestinationBucket = destBucket,
            DestinationKey = destKey,
            SourceBucket = sourceBucket,
            SourceKey = sourceKey
        };

        switch (conditionType)
        {
            case S3ConditionType.IfMatch:
                copyObjectRequest.ETagToMatch = etagConditionalValue;
                break;

```



```

        case S3ConditionType.IfNoneMatch:
            copyObjectRequest.ETagToNotMatch = etagConditionalValue;
            break;
        case S3ConditionType.IfModifiedSince:
            copyObjectRequest.ModifiedSinceDateUtc =
conditionDateValue.GetValueOrDefault();
            break;
        case S3ConditionType.IfUnmodifiedSince:
            copyObjectRequest.UnmodifiedSinceDateUtc =
conditionDateValue.GetValueOrDefault();
            break;
        default:
            throw new ArgumentOutOfRangeException(nameof(conditionType),
conditionType, null);
    }

    await _amazonS3.CopyObjectAsync(copyObjectRequest);
    _logger.LogInformation($"Conditional copy successful for key
{destKey} in bucket {destBucket}.");
    return true;
}
catch (AmazonS3Exception e)
{
    if (e.ErrorCode == "PreconditionFailed")
    {
        _logger.LogError("Conditional copy failed: Precondition failed");
    }
    else if (e.ErrorCode == "304")
    {
        _logger.LogError("Conditional copy failed: Object not modified");
    }
    else
    {
        _logger.LogError($"Unexpected error: {e.ErrorCode}");
        throw;
    }
    return false;
}
}

/// <summary>
/// Create a new Amazon S3 bucket with a specified name and check that the
bucket is ready.
/// </summary>

```

```
/// <param name="bucketName">The name of the bucket to create.</param>
/// <returns>True if successful.</returns>
public async Task<bool> CreateBucketWithName(string bucketName)
{
    Console.WriteLine($"\\tCreating bucket {bucketName}.");
    try
    {
        var request = new PutBucketRequest
        {
            BucketName = bucketName,
            UseClientRegion = true
        };

        await _amazonS3.PutBucketAsync(request);
        var bucketReady = false;
        var retries = 5;
        while (!bucketReady && retries > 0)
        {
            Thread.Sleep(5000);
            bucketReady = await
Amazon.S3.Util.AmazonS3Util.DoesS3BucketExistV2Async(_amazonS3, bucketName);
            retries--;
        }

        return bucketReady;
    }
    catch (BucketAlreadyExistsException ex)
    {
        Console.WriteLine($"Bucket already exists: '{ex.Message}'");
        return true;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error creating bucket: '{ex.Message}'");
        return false;
    }
}

/// <summary>
/// Cleans up objects and deletes the bucket by name.
/// </summary>
/// <param name="bucketName">The name of the bucket.</param>
/// <returns>Async task.</returns>
public async Task CleanupBucketByName(string bucketName)
```

```

    {
        try
        {
            var listObjectsResponse = await _amazonS3.ListObjectsV2Async(new
ListObjectsV2Request { BucketName = bucketName });
            foreach (var obj in listObjectsResponse.S3Objects)
            {
                await _amazonS3.DeleteObjectAsync(new DeleteObjectRequest
{ BucketName = bucketName, Key = obj.Key });
            }
            await _amazonS3.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
            Console.WriteLine($"Cleaned up bucket: {bucketName}.");
        }
        catch (AmazonS3Exception e)
        {
            if (e.ErrorCode == "NoSuchBucket")
            {
                Console.WriteLine($"Bucket {bucketName} does not exist, skipping
cleanup.");
            }
            else
            {
                Console.WriteLine($"Error deleting bucket: {e.ErrorCode}");
                throw;
            }
        }
    }
}

/// <summary>
/// List the contents of the bucket with their ETag.
/// </summary>
/// <param name="bucketName">The name of the bucket.</param>
/// <returns>Async task.</returns>
public async Task<List<S3Object>> ListBucketContentsByName(string bucketName)
{
    var results = new List<S3Object>();
    try
    {
        Console.WriteLine($"\\t Items in bucket {bucketName}");
        var listObjectsResponse = await _amazonS3.ListObjectsV2Async(new
ListObjectsV2Request { BucketName = bucketName });
        if (listObjectsResponse.S3Objects.Count == 0)
        {

```

```

        Console.WriteLine("\t\tNo objects found.");
    }
    else
    {
        foreach (var obj in listObjectsResponse.S3Objects)
        {
            Console.WriteLine($"{obj.Key} ETag {obj.ETag}");
        }
    }
    results = listObjectsResponse.S3Objects;

}
catch (AmazonS3Exception e)
{
    if (e.ErrorCode == "NoSuchBucket")
    {
        _logger.LogError($"Bucket {bucketName} does not exist.");
    }
    else
    {
        _logger.LogError($"Error listing bucket and objects:
{e.ErrorCode}");
        throw;
    }
}

return results;
}

/// <summary>
/// Delete an object from a specific bucket.
/// </summary>
/// <param name="bucketName">The Amazon S3 bucket to use.</param>
/// <param name="objectKey">The key of the object to delete.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DeleteObjectFromBucket(string bucketName, string
objectKey)
{
    try
    {
        var request = new DeleteObjectRequest()
        {
            BucketName = bucketName,
            Key = objectKey

```

```

        };
        await _amazonS3.DeleteObjectAsync(request);
        Console.WriteLine($"Deleted {objectKey} in {bucketName}.");
        return true;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Unable to delete object {objectKey} in bucket
{bucketName}: " + ex.Message);
        return false;
    }
}

/// <summary>
/// Delete a specific bucket by deleting the objects and then the bucket
itself.
/// </summary>
/// <param name="bucketName">The Amazon S3 bucket to use.</param>
/// <param name="objectKey">The key of the object to delete.</param>
/// <param name="versionId">Optional versionId.</param>
/// <returns>True if successful.</returns>
public async Task<bool> CleanUpBucketByName(string bucketName)
{
    try
    {
        var allFiles = await ListBucketContentsByName(bucketName);

        foreach (var fileInfo in allFiles)
        {
            await DeleteObjectFromBucket(fileInfo.BucketName, fileInfo.Key);
        }

        var request = new DeleteBucketRequest() { BucketName = bucketName, };
        var response = await _amazonS3.DeleteBucketAsync(request);
        Console.WriteLine($"Delete for {bucketName} complete.");
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Unable to delete bucket {bucketName}: " +
ex.Message);
        return false;
    }
}

```

```
}  
  
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的下列主題。
 - [CopyObject](#)
 - [GetObject](#)
 - [PutObject](#)

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

工作流程的進入點 (index.js)。這會協調所有步驟。請造訪 GitHub 以查看 Scenario、ScenarioInput、ScenarioOutput 和 ScenarioAction 的實作詳細資訊。

```
import * as Scenarios from "@aws-doc-sdk-examples/lib/scenario/index.js";  
import {  
  exitOnFalse,  
  loadState,  
  saveState,  
} from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";  
  
import { welcome, welcomeContinue } from "../welcome.steps.js";  
import {  
  confirmCreateBuckets,  
  confirmPopulateBuckets,  
  createBuckets,  
  createBucketsAction,  
  getBucketPrefix,  
  populateBuckets,  
  populateBucketsAction,  
} from "../setup.steps.js";
```

```
/**
 * @param {Scenarios} scenarios
 * @param {Record<string, any>} initialState
 */
export const getWorkflowStages = (scenarios, initialState = {}) => {
  const client = new S3Client({});

  return {
    deploy: new scenarios.Scenario(
      "S3 Conditional Requests - Deploy",
      [
        welcome(scenarios),
        welcomeContinue(scenarios),
        exitOnFalse(scenarios, "welcomeContinue"),
        getBucketPrefix(scenarios),
        createBuckets(scenarios),
        confirmCreateBuckets(scenarios),
        exitOnFalse(scenarios, "confirmCreateBuckets"),
        createBucketsAction(scenarios, client),
        populateBuckets(scenarios),
        confirmPopulateBuckets(scenarios),
        exitOnFalse(scenarios, "confirmPopulateBuckets"),
        populateBucketsAction(scenarios, client),
        saveState,
      ],
      initialState,
    ),
    demo: new scenarios.Scenario(
      "S3 Conditional Requests - Demo",
      [loadState, welcome(scenarios), replAction(scenarios, client)],
      initialState,
    ),
    clean: new scenarios.Scenario(
      "S3 Conditional Requests - Destroy",
      [
        loadState,
        confirmCleanup(scenarios),
        exitOnFalse(scenarios, "confirmCleanup"),
        cleanupAction(scenarios, client),
      ],
      initialState,
    ),
  };
};
```

```

};

// Call function if run directly
import { fileURLToPath } from "node:url";
import { S3Client } from "@aws-sdk/client-s3";
import { cleanupAction, confirmCleanup } from "./clean.steps.js";
import { replAction } from "./repl.steps.js";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  const objectLockingScenarios = getWorkflowStages(Scenarios);
  Scenarios.parseScenarioArgs(objectLockingScenarios, {
    name: "Amazon S3 object locking workflow",
    description:
      "Work with Amazon Simple Storage Service (Amazon S3) object locking features.",
    synopsis:
      "node index.js --scenario <deploy | demo | clean> [-h|--help] [-y|--yes] [-v|--verbose]",
  });
}

```

將歡迎訊息輸出至主控台 (welcome.steps.js)。

```

/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @param {Scenarios} scenarios
 */
const welcome = (scenarios) =>
  new scenarios.ScenarioOutput(
    "welcome",
    "This example demonstrates the use of conditional requests for S3 operations." +
    " You can use conditional requests to add preconditions to S3 read requests to return " +
    "or copy an object based on its Entity tag (ETag), or last modified date.You can use " +
    "a conditional write requests to prevent overwrites by ensuring there is no existing " +
    "object with the same key.\n" +

```



```

    "This example will enable you to perform conditional reads and writes that
    will succeed " +
    "or fail based on your selected options.\n" +
    "Sample buckets and a sample object will be created as part of the example.
\n" +
    "Some steps require a key name prefix to be defined by the user. Before you
    begin, you can " +
    "optionally edit this prefix in ./object_name.json. If you do so, please
    reload the scenario before you begin.",
    { header: true },
  );

/**
 * @param {Scenarios} scenarios
 */
const welcomeContinue = (scenarios) =>
  new scenarios.ScenarioInput(
    "welcomeContinue",
    "Press Enter when you are ready to start.",
    { type: "confirm" },
  );

export { welcome, welcomeContinue };

```

部署儲存貯體和物件(setup.steps.js)。

```

import {
  ChecksumAlgorithm,
  CreateBucketCommand,
  PutObjectCommand,
  BucketAlreadyExists,
  BucketAlreadyOwnedByYou,
  S3ServiceException,
  waitUntilBucketExists,
} from "@aws-sdk/client-s3";

/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @typedef {import("@aws-sdk/client-s3").S3Client} S3Client

```

```

    */

    /**
     * @param {Scenarios} scenarios
     */
    const getBucketPrefix = (scenarios) =>
      new scenarios.ScenarioInput(
        "bucketPrefix",
        "Provide a prefix that will be used for bucket creation.",
        { type: "input", default: "amzn-s3-demo-bucket" },
      );

    /**
     * @param {Scenarios} scenarios
     */
    const createBuckets = (scenarios) =>
      new scenarios.ScenarioOutput(
        "createBuckets",
        (state) => `The following buckets will be created:
          ${state.bucketPrefix}-source-bucket.
          ${state.bucketPrefix}-destination-bucket.` ,
        { preformatted: true },
      );

    /**
     * @param {Scenarios} scenarios
     */
    const confirmCreateBuckets = (scenarios) =>
      new scenarios.ScenarioInput("confirmCreateBuckets", "Create the buckets?", {
        type: "confirm",
      });

    /**
     * @param {Scenarios} scenarios
     * @param {S3Client} client
     */
    const createBucketsAction = (scenarios, client) =>
      new scenarios.ScenarioAction("createBucketsAction", async (state) => {
        const sourceBucketName = `${state.bucketPrefix}-source-bucket`;
        const destinationBucketName = `${state.bucketPrefix}-destination-bucket`;

        try {
          await client.send(
            new CreateBucketCommand({
              Bucket: sourceBucketName,

```

```

    })),
  );
  await waitUntilBucketExists({ client }, { Bucket: sourceBucketName });
  await client.send(
    new CreateBucketCommand({
      Bucket: destinationBucketName,
    }),
  );
  await waitUntilBucketExists(
    { client },
    { Bucket: destinationBucketName },
  );

  state.sourceBucketName = sourceBucketName;
  state.destinationBucketName = destinationBucketName;
} catch (caught) {
  if (
    caught instanceof BucketAlreadyExists ||
    caught instanceof BucketAlreadyOwnedByYou
  ) {
    console.error(`${caught.name}: ${caught.message}`);
    state.earlyExit = true;
  } else {
    throw caught;
  }
}
});

/**
 * @param {Scenarios} scenarios
 */
const populateBuckets = (scenarios) =>
  new scenarios.ScenarioOutput(
    "populateBuckets",
    (state) => `The following test files will be created:
      file01.txt in ${state.bucketPrefix}-source-bucket.`,
    { preformatted: true },
  );

/**
 * @param {Scenarios} scenarios
 */
const confirmPopulateBuckets = (scenarios) =>
  new scenarios.ScenarioInput(

```

```
        "confirmPopulateBuckets",
        "Populate the buckets?",
        { type: "confirm" },
    );
};

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const populateBucketsAction = (scenarios, client) =>
    new scenarios.ScenarioAction("populateBucketsAction", async (state) => {
        try {
            await client.send(
                new PutObjectCommand({
                    Bucket: state.sourceBucketName,
                    Key: "file01.txt",
                    Body: "Content",
                    ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
                }),
            );
        } catch (caught) {
            if (caught instanceof S3ServiceException) {
                console.error(
                    `Error from S3 while uploading object.  ${caught.name}:
                    ${caught.message}`,
                );
            } else {
                throw caught;
            }
        }
    });

export {
    confirmCreateBuckets,
    confirmPopulateBuckets,
    createBuckets,
    createBucketsAction,
    getBucketPrefix,
    populateBuckets,
    populateBucketsAction,
};
```

使用 S3 條件式請求 (repl.steps.js) 取得、複製和放置物件。

```
import path from "node:path";
import { fileURLToPath } from "node:url";
import { dirname } from "node:path";

import {
  ListObjectVersionsCommand,
  GetObjectCommand,
  CopyObjectCommand,
  PutObjectCommand,
} from "@aws-sdk/client-s3";
import data from "../object_name.json" assert { type: "json" };
import { readFile } from "node:fs/promises";
import {
  ScenarioInput,
  Scenario,
  ScenarioAction,
  ScenarioOutput,
} from "../../libs/scenario/index.js";

/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @typedef {import("@aws-sdk/client-s3").S3Client} S3Client
 */

const choices = {
  EXIT: 0,
  LIST_ALL_FILES: 1,
  CONDITIONAL_READ: 2,
  CONDITIONAL_COPY: 3,
  CONDITIONAL_WRITE: 4,
};

/**
 * @param {Scenarios} scenarios
 */
const replInput = (scenarios) =>
  new ScenarioInput(
    "replChoice",
```

```

    "Explore the S3 conditional request features by selecting one of the
    following choices",
    {
      type: "select",
      choices: [
        { name: "Print list of bucket items.", value: choices.LIST_ALL_FILES },
        {
          name: "Perform a conditional read.",
          value: choices.CONDITIONAL_READ,
        },
        {
          name: "Perform a conditional copy. These examples use the key name
          prefix defined in ./object_name.json.",
          value: choices.CONDITIONAL_COPY,
        },
        {
          name: "Perform a conditional write. This example use the sample file ./
          text02.txt.",
          value: choices.CONDITIONAL_WRITE,
        },
        { name: "Finish the workflow.", value: choices.EXIT },
      ],
    },
  );

/**
 * @param {S3Client} client
 * @param {string[]} buckets
 */
const getAllFiles = async (client, buckets) => {
  /** @type {{bucket: string, key: string, version: string}[]} */
  const files = [];
  for (const bucket of buckets) {
    const objectsResponse = await client.send(
      new ListObjectVersionsCommand({ Bucket: bucket }),
    );
    for (const version of objectsResponse.Versions || []) {
      const { Key } = version;
      files.push({ bucket, key: Key });
    }
  }
  return files;
};

```

```
/**
 * @param {S3Client} client
 * @param {string[]} buckets
 * @param {string} key
 */
const getEtag = async (client, bucket, key) => {
  const objectsResponse = await client.send(
    new GetObjectCommand({
      Bucket: bucket,
      Key: key,
    }),
  );
  return objectsResponse.ETag;
};

/**
 * @param {S3Client} client
 * @param {string[]} buckets
 */

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
export const replAction = (scenarios, client) =>
  new ScenarioAction(
    "replAction",
    async (state) => {
      const files = await getAllFiles(client, [
        state.sourceBucketName,
        state.destinationBucketName,
      ]);

      const fileInput = new scenarios.ScenarioInput(
        "selectedFile",
        "Select a file to use:",
        {
          type: "select",
          choices: files.map((file, index) => ({
            name: `${index + 1}: ${file.bucket}: ${file.key} (Etag: ${
              file.version
            })`,
            value: index,
          })),
        },
      );
    },
  );
```

```
    },
  );
  const condReadOptions = new scenarios.ScenarioInput(
    "selectOption",
    "Which conditional read action would you like to take?",
    {
      type: "select",
      choices: [
        "If-Match: using the object's ETag. This condition should succeed.",
        "If-None-Match: using the object's ETag. This condition should
fail.",
        "If-Modified-Since: using yesterday's date. This condition should
succeed.",
        "If-Unmodified-Since: using yesterday's date. This condition should
fail.",
      ],
    },
  );
  const condCopyOptions = new scenarios.ScenarioInput(
    "selectOption",
    "Which conditional copy action would you like to take?",
    {
      type: "select",
      choices: [
        "If-Match: using the object's ETag. This condition should succeed.",
        "If-None-Match: using the object's ETag. This condition should
fail.",
        "If-Modified-Since: using yesterday's date. This condition should
succeed.",
        "If-Unmodified-Since: using yesterday's date. This condition should
fail.",
      ],
    },
  );
  const condWriteOptions = new scenarios.ScenarioInput(
    "selectOption",
    "Which conditional write action would you like to take?",
    {
      type: "select",
      choices: [
        "IfNoneMatch condition on the object key: If the key is a duplicate,
the write will fail.",
      ],
    },
  );
```



```
);

const { replChoice } = state;

switch (replChoice) {
  case choices.LIST_ALL_FILES: {
    const files = await getAllFiles(client, [
      state.sourceBucketName,
      state.destinationBucketName,
    ]);
    state.replOutput = files
      .map(
        (file) => `Items in bucket ${file.bucket}: object: ${file.key} `,
      )
      .join("\n");
    break;
  }
  case choices.CONDITIONAL_READ:
    {
      const selectedCondRead = await condReadOptions.handle(state);
      if (
        selectedCondRead ===
        "If-Match: using the object's ETag. This condition should succeed."
      ) {
        const bucket = state.sourceBucketName;
        const key = "file01.txt";
        const ETag = await getEtag(client, bucket, key);

        try {
          await client.send(
            new GetObjectCommand({
              Bucket: bucket,
              Key: key,
              IfMatch: ETag,
            }),
          );
          state.replOutput = `${key} in bucket ${state.sourceBucketName}
read because ETag provided matches the object's ETag.`;
        } catch (err) {
          state.replOutput = `Unable to read object ${key} in bucket
${state.sourceBucketName}: ${err.message}`;
        }
        break;
      }
    }
}
```

```
        if (
            selectedCondRead ===
            "If-None-Match: using the object's ETag. This condition should
fail."
        ) {
            const bucket = state.sourceBucketName;
            const key = "file01.txt";
            const ETag = await getEtag(client, bucket, key);

            try {
                await client.send(
                    new GetObjectCommand({
                        Bucket: bucket,
                        Key: key,
                        IfNoneMatch: ETag,
                    }),
                );
                state.replOutput = `${key} in ${state.sourceBucketName} was
returned.`;
            } catch (err) {
                state.replOutput = `${key} in ${state.sourceBucketName} was not
read: ${err.message}`;
            }
            break;
        }
        if (
            selectedCondRead ===
            "If-Modified-Since: using yesterday's date. This condition should
succeed."
        ) {
            const date = new Date();
            date.setDate(date.getDate() - 1);

            const bucket = state.sourceBucketName;
            const key = "file01.txt";
            try {
                await client.send(
                    new GetObjectCommand({
                        Bucket: bucket,
                        Key: key,
                        IfModifiedSince: date,
                    }),
                );
            }
        }
    }
}
```

```

        state.replOutput = `${key} in bucket ${state.sourceBucketName}
read because it has been created or modified in the last 24 hours.`;
    } catch (err) {
        state.replOutput = `Unable to read object ${key} in bucket
${state.sourceBucketName}: ${err.message}`;
    }
    break;
}
if (
    selectedCondRead ===
    "If-Unmodified-Since: using yesterday's date. This condition should
fail."
) {
    const bucket = state.sourceBucketName;
    const key = "file01.txt";

    const date = new Date();
    date.setDate(date.getDate() - 1);
    try {
        await client.send(
            new GetObjectCommand({
                Bucket: bucket,
                Key: key,
                IfUnmodifiedSince: date,
            }),
        );
        state.replOutput = `${key} in ${state.sourceBucketName} was
read.`;
    } catch (err) {
        state.replOutput = `${key} in ${state.sourceBucketName} was not
read: ${err.message}`;
    }
    break;
}
break;
case choices.CONDITIONAL_COPY: {
    const selectedCondCopy = await condCopyOptions.handle(state);
    if (
        selectedCondCopy ===
        "If-Match: using the object's ETag. This condition should succeed."
    ) {
        const bucket = state.sourceBucketName;
        const key = "file01.txt";

```

```

    const ETag = await getEtag(client, bucket, key);

    const copySource = `${bucket}/${key}`;
    // Optionally edit the default key name prefix of the copied object
in ./object_name.json.
    const name = data.name;
    const copiedKey = `${name}${key}`;
    try {
      await client.send(
        new CopyObjectCommand({
          CopySource: copySource,
          Bucket: state.destinationBucketName,
          Key: copiedKey,
          CopySourceIfMatch: ETag,
        }),
      );
      state.replOutput = `${key} copied as ${copiedKey} to bucket
${state.destinationBucketName} because ETag provided matches the object's
ETag.`;
    } catch (err) {
      state.replOutput = `Unable to copy object ${key} as ${copiedKey} to
bucket ${state.destinationBucketName}: ${err.message}`;
    }
    break;
  }
  if (
    selectedCondCopy ===
    "If-None-Match: using the object's ETag. This condition should fail."
  ) {
    const bucket = state.sourceBucketName;
    const key = "file01.txt";
    const ETag = await getEtag(client, bucket, key);
    const copySource = `${bucket}/${key}`;
    // Optionally edit the default key name prefix of the copied object
in ./object_name.json.
    const name = data.name;
    const copiedKey = `${name}${key}`;

    try {
      await client.send(
        new CopyObjectCommand({
          CopySource: copySource,
          Bucket: state.destinationBucketName,
          Key: copiedKey,

```

```

        CopySourceIfNoneMatch: ETag,
    )),
    );
    state.replOutput = `${copiedKey} copied to bucket
${state.destinationBucketName}`;
    } catch (err) {
        state.replOutput = `Unable to copy object as ${key} as as
${copiedKey} to bucket ${state.destinationBucketName}: ${err.message}`;
    }
    break;
}
if (
    selectedCondCopy ===
    "If-Modified-Since: using yesterday's date. This condition should
succeed."
) {
    const bucket = state.sourceBucketName;
    const key = "file01.txt";
    const copySource = `${bucket}/${key}`;
    // Optionally edit the default key name prefix of the copied object
in ./object_name.json.
    const name = data.name;
    const copiedKey = `${name}${key}`;

    const date = new Date();
    date.setDate(date.getDate() - 1);

    try {
        await client.send(
            new CopyObjectCommand({
                CopySource: copySource,
                Bucket: state.destinationBucketName,
                Key: copiedKey,
                CopySourceIfModifiedSince: date,
            )),
        );
        state.replOutput = `${key} copied as ${copiedKey} to bucket
${state.destinationBucketName} because it has been created or modified in the
last 24 hours.`;
    } catch (err) {
        state.replOutput = `Unable to copy object ${key} as ${copiedKey} to
bucket ${state.destinationBucketName} : ${err.message}`;
    }
    break;
}

```

```

    }
    if (
      selectedCondCopy ===
        "If-Unmodified-Since: using yesterday's date. This condition should
fail."
    ) {
      const bucket = state.sourceBucketName;
      const key = "file01.txt";
      const copySource = `${bucket}/${key}`;
      // Optionally edit the default key name prefix of the copied object
in ./object_name.json.
      const name = data.name;
      const copiedKey = `${name}${key}`;

      const date = new Date();
      date.setDate(date.getDate() - 1);

      try {
        await client.send(
          new CopyObjectCommand({
            CopySource: copySource,
            Bucket: state.destinationBucketName,
            Key: copiedKey,
            CopySourceIfUnmodifiedSince: date,
          }),
        );
        state.replOutput = `${copiedKey} copied to bucket
${state.destinationBucketName} because it has not been created or modified in
the last 24 hours.`;
      } catch (err) {
        state.replOutput = `Unable to copy object ${key} to bucket
${state.destinationBucketName}: ${err.message}`;
      }
    }
    break;
  }
  case choices.CONDITIONAL_WRITE:
    {
      const selectedCondWrite = await condWriteOptions.handle(state);
      if (
        selectedCondWrite ===
          "IfNoneMatch condition on the object key: If the key is a
duplicate, the write will fail."
      ) {

```

```

        // Optionally edit the default key name prefix of the copied object
        in ./object_name.json.
        const key = "text02.txt";
        const __filename = fileURLToPath(import.meta.url);
        const __dirname = dirname(__filename);
        const filePath = path.join(__dirname, "text02.txt");
        try {
            await client.send(
                new PutObjectCommand({
                    Bucket: `${state.destinationBucketName}`,
                    Key: `${key}`,
                    Body: await readFile(filePath),
                    IfNoneMatch: "*",
                }),
            );
            state.replOutput = `${key} uploaded to bucket
${state.destinationBucketName} because the key is not a duplicate.`;
        } catch (err) {
            state.replOutput = `Unable to upload object to bucket
${state.destinationBucketName}:${err.message}`;
        }
        break;
    }
}
break;

default:
    throw new Error(`Invalid replChoice: ${replChoice}`);
}
},
{
    whileConfig: {
        whileFn: ({ replChoice }) => replChoice !== choices.EXIT,
        input: replInput(scenarios),
        output: new ScenarioOutput("REPL output", (state) => state.replOutput, {
            preformatted: true,
        }),
    },
},
);

export { replInput, choices };

```

銷毀所有建立的資源 (clean.steps.js)。

```
import {
  DeleteObjectCommand,
  DeleteBucketCommand,
  ListObjectVersionsCommand,
} from "@aws-sdk/client-s3";

/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @typedef {import("@aws-sdk/client-s3").S3Client} S3Client
 */

/**
 * @param {Scenarios} scenarios
 */
const confirmCleanup = (scenarios) =>
  new scenarios.ScenarioInput("confirmCleanup", "Clean up resources?", {
    type: "confirm",
  });

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const cleanupAction = (scenarios, client) =>
  new scenarios.ScenarioAction("cleanupAction", async (state) => {
    const { sourceBucketName, destinationBucketName } = state;
    const buckets = [sourceBucketName, destinationBucketName].filter((b) => b);

    for (const bucket of buckets) {
      try {
        let objectsResponse;
        objectsResponse = await client.send(
          new ListObjectVersionsCommand({
            Bucket: bucket,
          }),
        );
        for (const version of objectsResponse.Versions || []) {
          const { Key, VersionId } = version;
          try {
```



```
        await client.send(
            new DeleteObjectCommand({
                Bucket: bucket,
                Key,
                VersionId,
            }),
        );
    } catch (err) {
        console.log(`An error occurred: ${err.message} `);
    }
}
} catch (e) {
    if (e instanceof Error && e.name === "NoSuchBucket") {
        console.log("Objects and buckets have already been deleted.");
        continue;
    }
    throw e;
}

await client.send(new DeleteBucketCommand({ Bucket: bucket }));
console.log(`Delete for ${bucket} complete.`);
}
});

export { confirmCleanup, cleanupAction };
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的下列主題。
 - [CopyObject](#)
 - [GetObject](#)
 - [PutObject](#)

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

執行顯示 Amazon S3 條件式請求的互動式情境。

```
"""
Purpose

Shows how to use AWS SDK for Python (Boto3) to get started using conditional
requests for
Amazon Simple Storage Service (Amazon S3).

"""

import logging
import random
import sys
import datetime

import boto3
from botocore.exceptions import ClientError

from s3_conditional_requests import S3ConditionalRequests

# Add relative path to include demo_tools in this code example without need for
# setup.
sys.path.append(".././.././../")
import demo_tools.question as q # noqa

# Constants
FILE_CONTENT = "This is a test file for S3 conditional requests."
RANDOM_SUFFIX = str(random.randint(100, 999))

logger = logging.getLogger(__name__)

class ConditionalRequestsScenario:
    """Runs a scenario that shows how to use S3 Conditional Requests."""

    def __init__(self, conditional_requests, s3_client):
        """
        :param conditional_requests: An object that wraps S3 conditional request
        actions.
        :param s3_client: A Boto3 S3 client for setup and cleanup operations.
        """
        self.conditional_requests = conditional_requests
        self.s3_client = s3_client
```

```
def setup_scenario(self, source_bucket: str, dest_bucket: str, object_key:
str):
    """
    Sets up the scenario by creating a source and destination bucket.
    Prompts the user to provide a bucket name prefix.

    :param source_bucket: The name of the source bucket.
    :param dest_bucket: The name of the destination bucket.
    :param object_key: The name of a test file to add to the source bucket.
    """

    # Create the buckets.
    try:
        self.s3_client.create_bucket(Bucket=source_bucket)
        self.s3_client.create_bucket(Bucket=dest_bucket)
        print(
            f"Created source bucket: {source_bucket} and destination bucket:
{dest_bucket}"
        )
    except ClientError as e:
        error_code = e.response["Error"]["Code"]
        logger.error(f"Error creating buckets: {error_code}")
        raise

    # Upload test file into the source bucket.
    try:
        print(f"Uploading file {object_key} to bucket {source_bucket}")
        response = self.s3_client.put_object(
            Bucket=source_bucket, Key=object_key, Body=FILE_CONTENT
        )
        object_etag = response["ETag"]
        return object_etag

    except Exception as e:
        logger.error(
            f"Failed to upload file {object_key} to bucket {source_bucket}:
{e}"
        )

def cleanup_scenario(self, source_bucket: str, dest_bucket: str):
    """
    Cleans up the scenario by deleting the source and destination buckets.
```

```

        :param source_bucket: The name of the source bucket.
        :param dest_bucket: The name of the destination bucket.
        """
        self.cleanup_bucket(source_bucket)
        self.cleanup_bucket(dest_bucket)

def cleanup_bucket(self, bucket_name: str):
    """
    Cleans up the bucket by deleting all objects and then the bucket itself.

    :param bucket_name: The name of the bucket.
    """
    try:
        # Get list of all objects in the bucket.
        list_response = self.s3_client.list_objects_v2(Bucket=bucket_name)
        objs = list_response.get("Contents", [])
        for obj in objs:
            key = obj["Key"]
            self.s3_client.delete_object(Bucket=bucket_name, Key=key)
        self.s3_client.delete_bucket(Bucket=bucket_name)
        print(f"Cleaned up bucket: {bucket_name}.")
    except ClientError as e:
        error_code = e.response["Error"]["Code"]
        if error_code == "NoSuchBucket":
            logger.info(f"Bucket {bucket_name} does not exist, skipping
cleanup.")
        else:
            logger.error(f"Error deleting bucket: {error_code}")
            raise

def display_buckets(self, source_bucket: str, dest_bucket: str):
    """
    Display a list of the objects in the test buckets.

    :param source_bucket: The name of the source bucket.
    :param dest_bucket: The name of the destination bucket.
    """
    self.list_bucket_contents(source_bucket)
    self.list_bucket_contents(dest_bucket)

def list_bucket_contents(self, bucket_name):
    """

```

Display a list of the objects in the bucket.

```
:param bucket_name: The name of the bucket.
"""
try:
    # Get list of all objects in the bucket.
    print(f"\t Items in bucket {bucket_name}")
    list_response = self.s3_client.list_objects_v2(Bucket=bucket_name)
    objs = list_response.get("Contents", [])
    if not objs:
        print("\t\tNo objects found.")
    for obj in objs:
        key = obj["Key"]
        print(f"\t\t object: {key} ETag {obj['ETag']}")
    return objs
except ClientError as e:
    error_code = e.response["Error"]["Code"]
    if error_code == "NoSuchBucket":
        logger.info(f"Bucket {bucket_name} does not exist.")
    else:
        logger.error(f"Error listing bucket and objects: {error_code}")
        raise
```

```
def display_menu(
    self, source_bucket: str, dest_bucket: str, object_key: str, etag: str
):
    """
    Displays the menu of conditional request options for the user.

    :param source_bucket: The name of the source bucket.
    :param dest_bucket: The name of the destination bucket.
    :param object_key: The key of the test object in the source bucket.
    :param etag: The etag of the test object in the source bucket.
    """

    actions = [
        "Print list of bucket items.",
        "Perform a conditional read.",
        "Perform a conditional copy.",
        "Perform a conditional write.",
        "Clean up and exit.",
    ]
```

```

        conditions = [
            "If-Match: using the object's ETag. This condition should succeed.",
            "If-None-Match: using the object's ETag. This condition should
fail.",
            "If-Modified-Since: using yesterday's date. This condition should
succeed.",
            "If-Unmodified-Since: using yesterday's date. This condition should
fail.",
        ]

        condition_types = [
            "IfMatch",
            "IfNoneMatch",
            "IfModifiedSince",
            "IfUnmodifiedSince",
        ]
        copy_condition_types = [
            "CopySourceIfMatch",
            "CopySourceIfNoneMatch",
            "CopySourceIfModifiedSince",
            "CopySourceIfUnmodifiedSince",
        ]

        yesterday_date = datetime.datetime.utcnow() - datetime.timedelta(days=1)

        choice = 0
        while choice != 4:
            print("-" * 88)
            print("Choose an action to explore some example conditional
requests.")
            choice = q.choose("Which action would you like to take? ", actions)
            if choice == 0:
                print("Listing the objects and buckets.")
                self.display_buckets(source_bucket, dest_bucket)
            elif choice == 1:
                print("Perform a conditional read.")
                condition_type = q.choose("Enter the condition type : ",
conditions)
                if condition_type == 0 or condition_type == 1:
                    self.conditional_requests.get_object_conditional(
                        object_key, source_bucket,
condition_types[condition_type], etag
                    )
                elif condition_type == 2 or condition_type == 3:

```

```

        self.conditional_requests.get_object_conditional(
            object_key,
            source_bucket,
            condition_types[condition_type],
            yesterday_date,
        )
    elif choice == 2:
        print("Perform a conditional copy.")
        condition_type = q.choose("Enter the condition type : ",
conditions)
        dest_key = q.ask("Enter an object key: ", q.non_empty)
        if condition_type == 0 or condition_type == 1:
            self.conditional_requests.copy_object_conditional(
                object_key,
                dest_key,
                source_bucket,
                dest_bucket,
                copy_condition_types[condition_type],
                etag,
            )
        elif condition_type == 2 or condition_type == 3:
            self.conditional_requests.copy_object_conditional(
                object_key,
                dest_key,
                copy_condition_types[condition_type],
                yesterday_date,
            )
    elif choice == 3:
        print(
            "Perform a conditional write using IfNoneMatch condition on
the object key."
        )
        print("If the key is a duplicate, the write will fail.")
        object_key = q.ask("Enter an object key: ", q.non_empty)
        self.conditional_requests.put_object_conditional(
            object_key, source_bucket, b"Conditional write example data."
        )
    elif choice == 4:
        print("Proceeding to cleanup.")

def run_scenario(self):
    """
    Runs the interactive scenario.

```

```

"""
print("-" * 88)
print("Welcome to the Amazon S3 conditional requests example.")
print("-" * 88)

print(
    f"""\
    This example demonstrates the use of conditional requests for S3
operations.
    You can use conditional requests to add preconditions to S3 read requests
to return or copy
    an object based on its Entity tag (ETag), or last modified date.
    You can use a conditional write requests to prevent overwrites by
ensuring
    there is no existing object with the same key.

    This example will allow you to perform conditional reads
and writes that will succeed or fail based on your selected options.

    Sample buckets and a sample object will be created as part of the
example.
    """
)

bucket_prefix = q.ask("Enter a bucket name prefix: ", q.non_empty)
source_bucket_name = f"{bucket_prefix}-source-{RANDOM_SUFFIX}"
dest_bucket_name = f"{bucket_prefix}-dest-{RANDOM_SUFFIX}"
object_key = "test-upload-file.txt"

try:
    etag = self.setup_scenario(source_bucket_name, dest_bucket_name,
object_key)
    self.display_menu(source_bucket_name, dest_bucket_name, object_key,
etag)
finally:
    self.cleanup_scenario(source_bucket_name, dest_bucket_name)

print("-" * 88)
print("Thanks for watching.")
print("-" * 88)

if __name__ == "__main__":
    scenario = ConditionalRequestsScenario(

```



```
S3ConditionalRequests.from_client(), boto3.client("s3")
)
scenario.run_scenario()
```

定義條件式請求操作的包裝函式類別。

```
import boto3
import logging

from botocore.exceptions import ClientError

# Configure logging
logger = logging.getLogger(__name__)

class S3ConditionalRequests:
    """Encapsulates S3 conditional request operations."""

    def __init__(self, s3_client):
        self.s3 = s3_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        s3_client = boto3.client("s3")
        return cls(s3_client)

    def get_object_conditional(
        self,
        object_key: str,
        source_bucket: str,
        condition_type: str,
        condition_value: str,
    ):
        """
        Retrieves an object from Amazon S3 with a conditional request.
```

```

:param object_key: The key of the object to retrieve.
:param source_bucket: The source bucket of the object.
:param condition_type: The type of condition: 'IfMatch', 'IfNoneMatch',
'IfModifiedSince', 'IfUnmodifiedSince'.
:param condition_value: The value to use for the condition.
"""
try:
    response = self.s3.get_object(
        Bucket=source_bucket,
        Key=object_key,
        **{condition_type: condition_value},
    )
    sample_bytes = response["Body"].read(20)
    print(
        f"\tConditional read successful. Here are the first 20 bytes of
the object:\n"
    )
    print(f"\t{sample_bytes}")
except ClientError as e:
    error_code = e.response["Error"]["Code"]
    if error_code == "PreconditionFailed":
        print("\tConditional read failed: Precondition failed")
    elif error_code == "304": # Not modified error code.
        print("\tConditional read failed: Object not modified")
    else:
        logger.error(f"Unexpected error: {error_code}")
        raise

def put_object_conditional(self, object_key: str, source_bucket: str, data:
bytes):
    """
    Uploads an object to Amazon S3 with a conditional request. Prevents
    overwrite
    using an IfNoneMatch condition for the object key.

    :param object_key: The key of the object to upload.
    :param source_bucket: The source bucket of the object.
    :param data: The data to upload.
    """
    try:
        self.s3.put_object(
            Bucket=source_bucket, Key=object_key, Body=data, IfNoneMatch="*"

```

```

        )
        print(
            f"\tConditional write successful for key {object_key} in bucket
{source_bucket}."
        )
    except ClientError as e:
        error_code = e.response["Error"]["Code"]
        if error_code == "PreconditionFailed":
            print("\tConditional write failed: Precondition failed")
        else:
            logger.error(f"Unexpected error: {error_code}")
            raise

def copy_object_conditional(
    self,
    source_key: str,
    dest_key: str,
    source_bucket: str,
    dest_bucket: str,
    condition_type: str,
    condition_value: str,
):
    """
    Copies an object from one Amazon S3 bucket to another with a conditional
    request.

    :param source_key: The key of the source object to copy.
    :param dest_key: The key of the destination object.
    :param source_bucket: The source bucket of the object.
    :param dest_bucket: The destination bucket of the object.
    :param condition_type: The type of condition to apply, e.g.
        'CopySourceIfMatch', 'CopySourceIfNoneMatch',
        'CopySourceIfModifiedSince', 'CopySourceIfUnmodifiedSince'.
    :param condition_value: The value to use for the condition.
    """
    try:
        self.s3.copy_object(
            Bucket=dest_bucket,
            Key=dest_key,
            CopySource={"Bucket": source_bucket, "Key": source_key},
            **{condition_type: condition_value},
        )
        print(

```

```
        f"\tConditional copy successful for key {dest_key} in bucket\n{dest_bucket}."
    )
except ClientError as e:
    error_code = e.response["Error"]["Code"]
    if error_code == "PreconditionFailed":
        print("\tConditional copy failed: Precondition failed")
    elif error_code == "304": # Not modified error code.
        print("\tConditional copy failed: Object not modified")
    else:
        logger.error(f"Unexpected error: {error_code}")
        raise
```

- 如需 API 詳細資訊，請參閱下列《適用於 Python (Boto3) 的 AWS SDK API 參考》中的下列主題。
 - [CopyObject](#)
 - [GetObject](#)
 - [PutObject](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 管理 Amazon S3 儲存貯體的存取控制清單 (ACLs)

下列程式碼範例示範如何管理 Amazon S3 儲存貯體的存取控制清單 (ACL)。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Collections.Generic;
```

```

using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to manage Amazon Simple Storage Service
/// (Amazon S3) access control lists (ACLs) to control Amazon S3 bucket
/// access.
/// </summary>
public class ManageACLs
{
    public static async Task Main()
    {
        string bucketName = "amzn-s3-demo-bucket1";
        string newBucketName = "amzn-s3-demo-bucket2";
        string keyName = "sample-object.txt";
        string emailAddress = "someone@example.com";

        // If the AWS Region where your bucket is located is different from
        // the Region defined for the default user, pass the Amazon S3
bucket's
        // name to the client constructor. It should look like this:
        // RegionEndpoint bucketRegion = RegionEndpoint.USEast1;
        IAmazonS3 client = new AmazonS3Client();

        await TestBucketObjectACLsAsync(client, bucketName, newBucketName,
keyName, emailAddress);
    }

    /// <summary>
    /// Creates a new Amazon S3 bucket with a canned ACL, then retrieves the
ACL
    /// information and then adds a new ACL to one of the objects in the
    /// Amazon S3 bucket.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// methods to create a bucket, get an ACL, and add a different ACL to
    /// one of the objects.</param>
    /// <param name="bucketName">A string representing the original Amazon S3
    /// bucket name.</param>
    /// <param name="newBucketName">A string representing the name of the
    /// new bucket that will be created.</param>

```

```

S3    /// <param name="keyName">A string representing the key name of an Amazon
    /// object for which we will change the ACL.</param>
    /// <param name="emailAddress">A string representing the email address
    /// belonging to the person to whom access to the Amazon S3 bucket will
be    be
    /// granted.</param>
    public static async Task TestBucketObjectACLsAsync(
        IAmazonS3 client,
        string bucketName,
        string newBucketName,
        string keyName,
        string emailAddress)
    {
        try
        {
            // Create a new Amazon S3 bucket and specify canned ACL.
            var success = await CreateBucketWithCannedACLAsync(client,
newBucketName);

            // Get the ACL on a bucket.
            await GetBucketACLAsync(client, bucketName);

            // Add (replace) the ACL on an object in a bucket.
            await AddACLToExistingObjectAsync(client, bucketName, keyName,
emailAddress);
        }
        catch (AmazonS3Exception amazonS3Exception)
        {
            Console.WriteLine($"Exception: {amazonS3Exception.Message}");
        }
    }

    /// <summary>
    /// Creates a new Amazon S3 bucket with a canned ACL attached.
    /// </summary>
    /// <param name="client">The initialized client object used to call
    /// PutBucketAsync.</param>
    /// <param name="newBucketName">A string representing the name of the
    /// new Amazon S3 bucket.</param>
    /// <returns>Returns a boolean value indicating success or failure.</
returns>
    public static async Task<bool> CreateBucketWithCannedACLAsync(IAmazonS3
client, string newBucketName)

```

```

    {
        var request = new PutBucketRequest()
        {
            BucketName = newBucketName,
            BucketRegion = S3Region.EUWest1,

            // Add a canned ACL.
            CannedACL = S3CannedACL.LogDeliveryWrite,
        };

        var response = await client.PutBucketAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Retrieves the ACL associated with the Amazon S3 bucket name in the
    /// bucketName parameter.
    /// </summary>
    /// <param name="client">The initialized client object used to call
    /// PutBucketAsync.</param>
    /// <param name="bucketName">The Amazon S3 bucket for which we want to
get the
    /// ACL list.</param>
    /// <returns>Returns an S3AccessControlList returned from the call to
    /// GetACLAsync.</returns>
    public static async Task<S3AccessControlList> GetBucketACLAsync(IAmazonS3
client, string bucketName)
    {
        GetACLResponse response = await client.GetACLAsync(new GetACLRequest
        {
            BucketName = bucketName,
        });

        return response.AccessControlList;
    }

    /// <summary>
    /// Adds a new ACL to an existing object in the Amazon S3 bucket.
    /// </summary>
    /// <param name="client">The initialized client object used to call
    /// PutBucketAsync.</param>

```

```
S3    /// <param name="bucketName">A string representing the name of the Amazon
    /// bucket containing the object to which we want to apply a new ACL.</
param>    /// <param name="keyName">A string representing the name of the object
    /// to which we want to apply the new ACL.</param>
    /// <param name="emailAddress">The email address of the person to whom
    /// we will be applying to whom access will be granted.</param>
    public static async Task AddACLToExistingObjectAsync(IAmazonS3 client,
string bucketName, string keyName, string emailAddress)
    {
        // Retrieve the ACL for an object.
        GetACLResponse aclResponse = await client.GetACLAsync(new
GetACLRequest
    {
        BucketName = bucketName,
        Key = keyName,
    });

        S3AccessControlList acl = aclResponse.AccessControlList;

        // Retrieve the owner.
        Owner owner = acl.Owner;

        // Clear existing grants.
        acl.Grants.Clear();

        // Add a grant to reset the owner's full permission
        // (the previous clear statement removed all permissions).
        var fullControlGrant = new S3Grant
        {
            Grantee = new S3Grantee { CanonicalUser = acl.Owner.Id },
        };
        acl.AddGrant(fullControlGrant.Grantee, S3Permission.FULL_CONTROL);

        // Specify email to identify grantee for granting permissions.
        var grantUsingEmail = new S3Grant
        {
            Grantee = new S3Grantee { EmailAddress = emailAddress },
            Permission = S3Permission.WRITE_ACP,
        };

        // Specify log delivery group as grantee.
        var grantLogDeliveryGroup = new S3Grant
```



```
        {
            Grantee = new S3Grantee { URI = "http://acs.amazonaws.com/groups/
s3/LogDelivery" },
            Permission = S3Permission.WRITE,
        };

        // Create a new ACL.
        var newAcl = new S3AccessControlList
        {
            Grants = new List<S3Grant> { grantUsingEmail,
grantLogDeliveryGroup },
            Owner = owner,
        };

        // Set the new ACL. We're throwing away the response here.
        _ = await client.PutACLAsync(new PutACLRequest
        {
            BucketName = bucketName,
            Key = keyName,
            AccessControlList = newAcl,
        });
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的下列主題。
 - [GetBucketAcl](#)
 - [GetObjectAcl](#)
 - [PutBucketAcl](#)
 - [PutObjectAcl](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

透過 SDK 使用 Amazon S3 管理大型 Amazon SQS 訊息 Amazon S3 AWS

下列程式碼範例示範如何使用 Amazon SQS 擴充用戶端程式庫，處理大型 Amazon SQS 訊息。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import com.amazon.sqs.javamessaging.AmazonSQSExtendedClient;
import com.amazon.sqs.javamessaging.ExtendedClientConfiguration;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.joda.time.DateTime;
import org.joda.time.format.DateTimeFormat;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.BucketLifecycleConfiguration;
import software.amazon.awssdk.services.s3.model.CreateBucketRequest;
import software.amazon.awssdk.services.s3.model.DeleteBucketRequest;
import software.amazon.awssdk.services.s3.model.DeleteObjectRequest;
import software.amazon.awssdk.services.s3.model.ExpirationStatus;
import software.amazon.awssdk.services.s3.model.LifecycleExpiration;
import software.amazon.awssdk.services.s3.model.LifecycleRule;
import software.amazon.awssdk.services.s3.model.LifecycleRuleFilter;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsRequest;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsResponse;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import
    software.amazon.awssdk.services.s3.model.PutBucketLifecycleConfigurationRequest;
import software.amazon.awssdk.services.sqs.SqsClient;
import software.amazon.awssdk.services.sqs.model.CreateQueueRequest;
import software.amazon.awssdk.services.sqs.model.CreateQueueResponse;
import software.amazon.awssdk.services.sqs.model.DeleteMessageRequest;
import software.amazon.awssdk.services.sqs.model.DeleteQueueRequest;
import software.amazon.awssdk.services.sqs.model.Message;
import software.amazon.awssdk.services.sqs.model.ReceiveMessageRequest;
import software.amazon.awssdk.services.sqs.model.ReceiveMessageResponse;
import software.amazon.awssdk.services.sqs.model.SendMessageRequest;

import java.util.Arrays;
```

```
import java.util.List;
import java.util.UUID;

/**
 * Example of using Amazon SQS Extended Client Library for Java 2.x.
 */
public class SqsExtendedClientExample {
    private static final Logger logger =
        LoggerFactory.getLogger(SqsExtendedClientExample.class);

    private String s3BucketName;
    private String queueUrl;
    private final String queueName;
    private final S3Client s3Client;
    private final SqsClient sqsExtendedClient;
    private final int messageSize;

    /**
     * Constructor with default clients and message size.
     */
    public SqsExtendedClientExample() {
        this(S3Client.create(), 300000);
    }

    /**
     * Constructor with custom S3 client and message size.
     *
     * @param s3Client The S3 client to use
     * @param messageSize The size of the test message to create
     */
    public SqsExtendedClientExample(S3Client s3Client, int messageSize) {
        this.s3Client = s3Client;
        this.messageSize = messageSize;

        // Generate a unique bucket name.
        this.s3BucketName = UUID.randomUUID() + "-" +
            DateTimeFormat.forPattern("yyMMdd-hhmmss").print(new DateTime());

        // Generate a unique queue name.
        this.queueName = "MyQueue-" + UUID.randomUUID();

        // Configure the SQS extended client.
        final ExtendedClientConfiguration extendedClientConfig = new
            ExtendedClientConfiguration()
```

```
        .withPayloadSupportEnabled(s3Client, s3BucketName);

        this.sqsExtendedClient = new
AmazonSQSExtendedClient(SqsClient.builder().build(), extendedClientConfig);
    }

    public static void main(String[] args) {
        SqsExtendedClientExample example = new SqsExtendedClientExample();
        try {
            example.setup();
            example.sendAndReceiveMessage();
        } finally {
            example.cleanup();
        }
    }

    /**
     * Send a large message and receive it back.
     *
     * @return The received message
     */
    public Message sendAndReceiveMessage() {
        try {
            // Create a large message.
            char[] chars = new char[messageSize];
            Arrays.fill(chars, 'x');
            String largeMessage = new String(chars);

            // Send the message.
            final SendMessageRequest sendMessageRequest =
SendMessageRequest.builder()
                .queueUrl(queueUrl)
                .messageBody(largeMessage)
                .build();

            sqsExtendedClient.sendMessage(sendMessageRequest);
            logger.info("Sent message of size: {}", largeMessage.length());

            // Receive and return the message.
            final ReceiveMessageResponse receiveMessageResponse =
sqsExtendedClient.receiveMessage(
                ReceiveMessageRequest.builder().queueUrl(queueUrl).build());

            List<Message> messages = receiveMessageResponse.messages();
```

```
        if (messages.isEmpty()) {
            throw new RuntimeException("No messages received");
        }

        Message message = messages.getFirst();
        logger.info("\nMessage received.");
        logger.info("  ID: {}", message.messageId());
        logger.info("  Receipt handle: {}", message.receiptHandle());
        logger.info("  Message body size: {}", message.body().length());
        logger.info("  Message body (first 5 characters): {}",
            message.body().substring(0, 5));

        return message;
    } catch (RuntimeException e) {
        logger.error("Error during message processing: {}", e.getMessage(),
            e);
        throw e;
    }
}
```

- 如需詳細資訊，請參閱《[AWS SDK for Java 2.x 開發人員指南](#)》。
- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
 - [CreateBucket](#)
 - [PutBucketLifecycleConfiguration](#)
 - [ReceiveMessage](#)
 - [SendMessage](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 使用 Lambda 函數分批管理版本控制的 Amazon S3 物件 AWS

下列程式碼範例示範如何使用 Lambda 函數批次管理 S3 物件版本。

Python

適用於 Python 的 SDK (Boto3)

顯示如何透過建立呼叫 AWS Lambda 函數來執行處理的任務，分批操作 Amazon Simple Storage Service (Amazon S3) 版本控制的物件。此範例建立了一個啟用版本控制的儲存貯體，上傳一段 Lewis Carroll You Are Old, Father William 詩句，並使用 Amazon S3 Batch Job 以不同的方式扭曲詩句。

了解如何：

- 建立版本化物件上執行的 Lambda 函數。
- 建立更新物件的資訊清單。
- 建立叫用 Lambda 函數的批次任務以更新物件。
- 刪除 Lambda 函數。
- 清空並刪除版本化儲存貯體。

這個範例在 GitHub 上的檢視效果最佳。如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon S3

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 剖析 Amazon S3 URIs

下列程式碼範例示範如何剖析 Amazon S3 URI，以擷取如儲存貯體名稱和物件金鑰的重要元件。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

透過使用 [S3Uri](#) 類別剖析 Amazon S3 URI。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.S3Uri;
import software.amazon.awssdk.services.s3.S3Utilities;

import java.net.URI;
import java.util.List;
import java.util.Map;

/**
 *
 * @param s3Client      - An S3Client through which you acquire an S3Uri
instance.
 * @param s3objectUrl - A complex URL (String) that is used to demonstrate
S3Uri
 *
 * capabilities.
 */
public static void parseS3UriExample(S3Client s3Client, String s3objectUrl) {
    logger.info(s3objectUrl);
    // Console output:
    // 'https://s3.us-west-1.amazonaws.com/myBucket/resources/doc.txt?
versionId=abc123&partNumber=77&partNumber=88'.

    // Create an S3Utilities object using the configuration of the s3Client.
    S3Utilities s3Utilities = s3Client.utilities();

    // From a String URL create a URI object to pass to the parseUri()
method.
    URI uri = URI.create(s3objectUrl);
    S3Uri s3Uri = s3Utilities.parseUri(uri);

    // If the URI contains no value for the Region, bucket or key, the SDK
returns
    // an empty Optional.
    // The SDK returns decoded URI values.

    Region region = s3Uri.region().orElse(null);
    log("region", region);
    // Console output: 'region: us-west-1'.
```

```

String bucket = s3Uri.bucket().orElse(null);
log("bucket", bucket);
// Console output: 'bucket: myBucket'.

String key = s3Uri.key().orElse(null);
log("key", key);
// Console output: 'key: resources/doc.txt'.

Boolean isPathStyle = s3Uri.isPathStyle();
log("isPathStyle", isPathStyle);
// Console output: 'isPathStyle: true'.

// If the URI contains no query parameters, the SDK returns an empty map.
Map<String, List<String>> queryParams = s3Uri.rawQueryParameters();
log("rawQueryParameters", queryParams);
// Console output: 'rawQueryParameters: {versionId=[abc123],
partNumber=[77,
// 88]}'.

// Retrieve the first or all values for a query parameter as shown in the
// following code.
String versionId =
s3Uri.firstMatchingRawQueryParameter("versionId").orElse(null);
log("firstMatchingRawQueryParameter-versionId", versionId);
// Console output: 'firstMatchingRawQueryParameter-versionId: abc123'.

String partNumber =
s3Uri.firstMatchingRawQueryParameter("partNumber").orElse(null);
log("firstMatchingRawQueryParameter-partNumber", partNumber);
// Console output: 'firstMatchingRawQueryParameter-partNumber: 77'.

List<String> partNumbers =
s3Uri.firstMatchingRawQueryParameters("partNumber");
log("firstMatchingRawQueryParameter", partNumbers);
// Console output: 'firstMatchingRawQueryParameter: [77, 88]'.

/*
 * Object keys and query parameters with reserved or unsafe characters,
must be
 * URL-encoded.
 * For example replace whitespace " " with "%20".
 * Valid:
 * "https://s3.us-west-1.amazonaws.com/myBucket/object%20key?query=
%5Bbrackets%5D"

```



```
        * Invalid:
        * "https://s3.us-west-1.amazonaws.com/myBucket/object key?
query=[brackets]"
        *
        * Virtual-hosted-style URIs with bucket names that contain a dot, ".",
the dot
        * must not be URL-encoded.
        * Valid: "https://my.Bucket.s3.us-west-1.amazonaws.com/key"
        * Invalid: "https://my%2EBucket.s3.us-west-1.amazonaws.com/key"
        */
    }

    private static void log(String s3UriElement, Object element) {
        if (element == null) {
            logger.info("{}: {}", s3UriElement, "null");
        } else {
            logger.info("{}: {}", s3UriElement, element);
        }
    }
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 執行 Amazon S3 物件的分段複製

下列程式碼範例示範如何執行 Amazon S3 物件的分段複製。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
```

```
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to perform a multi-part copy from one Amazon
/// Simple Storage Service (Amazon S3) bucket to another.
/// </summary>
public class MPUApiCopyObj
{
    private const string SourceBucket = "amzn-s3-demo-bucket1";
    private const string TargetBucket = "amzn-s3-demo-bucket2";
    private const string SourceObjectKey = "example.mov";
    private const string TargetObjectKey = "copied_video_file.mov";

    /// <summary>
    /// This method starts the multi-part upload.
    /// </summary>
    public static async Task Main()
    {
        var s3Client = new AmazonS3Client();
        Console.WriteLine("Copying object...");
        await MPUCopyObjectAsync(s3Client);
    }

    /// <summary>
    /// This method uses the passed client object to perform a multipart
    /// copy operation.
    /// </summary>
    /// <param name="client">An Amazon S3 client object that will be used
    /// to perform the copy.</param>
    public static async Task MPUCopyObjectAsync(AmazonS3Client client)
    {
        // Create a list to store the copy part responses.
        var copyResponses = new List<CopyPartResponse>();

        // Setup information required to initiate the multipart upload.
        var initiateRequest = new InitiateMultipartUploadRequest
        {
            BucketName = TargetBucket,
            Key = TargetObjectKey,
        };

        // Initiate the upload.
        InitiateMultipartUploadResponse initResponse =
```

```
        await client.InitiateMultipartUploadAsync(initiateRequest);

// Save the upload ID.
string uploadId = initResponse.UploadId;

try
{
    // Get the size of the object.
    var metadataRequest = new GetObjectMetadataRequest
    {
        BucketName = SourceBucket,
        Key = SourceObjectKey,
    };

    GetObjectMetadataResponse metadataResponse =
        await client.GetObjectMetadataAsync(metadataRequest);
    var objectSize = metadataResponse.ContentLength; // Length in
bytes.

    // Copy the parts.
    var partSize = 5 * (long)Math.Pow(2, 20); // Part size is 5 MB.

    long bytePosition = 0;
    for (int i = 1; bytePosition < objectSize; i++)
    {
        var copyRequest = new CopyPartRequest
        {
            DestinationBucket = TargetBucket,
            DestinationKey = TargetObjectKey,
            SourceBucket = SourceBucket,
            SourceKey = SourceObjectKey,
            UploadId = uploadId,
            FirstByte = bytePosition,
            LastByte = bytePosition + partSize - 1 >= objectSize ?
objectSize - 1 : bytePosition + partSize - 1,
            PartNumber = i,
        };

        copyResponses.Add(await client.CopyPartAsync(copyRequest));

        bytePosition += partSize;
    }

    // Set up to complete the copy.
```

```
var completeRequest = new CompleteMultipartUploadRequest
{
    BucketName = TargetBucket,
    Key = TargetObjectKey,
    UploadId = initResponse.UploadId,
};
completeRequest.AddPartETags(copyResponses);

// Complete the copy.
CompleteMultipartUploadResponse completeUploadResponse =
    await client.CompleteMultipartUploadAsync(completeRequest);
}
catch (AmazonS3Exception e)
{
    Console.WriteLine($"Error encountered on server.
Message: '{e.Message}' when writing an object");
}
catch (Exception e)
{
    Console.WriteLine($"Unknown encountered on server.
Message: '{e.Message}' when writing an object");
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的下列主題。
 - [CompleteMultipartUpload](#)
 - [CreateMultipartUpload](#)
 - [GetObjectMetadata](#)
 - [UploadPartCopy](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 接收和處理 Amazon S3 事件通知 AWS

下列程式碼範例示範如何以物件導向的方式使用 S3 事件通知。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

此範例示範如何使用 Amazon SQS 處理 S3 通知事件。

```
/**
 * This method receives S3 event notifications by using an SqsAsyncClient.
 * After the client receives the messages it deserializes the JSON payload
and logs them. It uses
 * the S3EventNotification class (part of the S3 event notification API for
Java) to deserialize
 * the JSON payload and access the messages in an object-oriented way.
 *
 * @param queueUrl The URL of the AWS SQS queue that receives the S3 event
notifications.
 * @see <a href="https://sdk.amazonaws.com/java/api/latest/software/amazon/
awssdk/eventnotifications/s3/model/package-summary.html">S3EventNotification
API</a>.
 * <p>
 * To use S3 event notification serialization/deserialization to objects, add
the following
 * dependency to your Maven pom.xml file.
 * <dependency>
 * <groupId>software.amazon.awssdk</groupId>
 * <artifactId>s3-event-notifications</artifactId>
 * <version><LATEST></version>
 * </dependency>
 * <p>
 * The S3 event notification API became available with version 2.25.11 of the
Java SDK.
 * <p>
 * This example shows the use of the API with AWS SQS, but it can be used to
process S3 event notifications
 * in AWS SNS or AWS Lambda as well.
 * <p>
```

```

    * Note: The S3EventNotification class does not work with messages routed
    through AWS EventBridge.
    */
    static void processS3Events(String bucketName, String queueUrl, String
    queueArn) {
        try {
            // Configure the bucket to send Object Created and Object Tagging
            notifications to an existing SQS queue.
            s3Client.putBucketNotificationConfiguration(b -> b
                .notificationConfiguration(ncb -> ncb
                    .queueConfigurations(qcb -> qcb
                        .events(Event.S3_OBJECT_CREATED,
Event.S3_OBJECT_TAGGING)
                            .queueArn(queueArn)))
                    .bucket(bucketName)
                ).join();

            triggerS3EventNotifications(bucketName);
            // Wait for event notifications to propagate.
            Thread.sleep(Duration.ofSeconds(5).toMillis());

            boolean didReceiveMessages = true;
            while (didReceiveMessages) {
                // Display the number of messages that are available in the
                queue.
                sqsClient.getQueueAttributes(b -> b
                    .queueUrl(queueUrl)

                    .attributeNames(QueueAttributeName.APPROXIMATE_NUMBER_OF_MESSAGES)
                        .thenAccept(attributeResponse ->
                            logger.info("Approximate number of messages in
the queue: {}"),
attributeResponse.attributes().get(QueueAttributeName.APPROXIMATE_NUMBER_OF_MESSAGES)))
                    .join();

                // Receive the messages.
                ReceiveMessageResponse response = sqsClient.receiveMessage(b -> b
                    .queueUrl(queueUrl)
                ).get();
                logger.info("Count of received messages: {}",
response.messages().size());
                didReceiveMessages = !response.messages().isEmpty();
            }
        }
    }

```

```

        // Create a collection to hold the received message for deletion
        // after we log the messages.
        HashSet<DeleteMessageBatchRequestEntry> messagesToDelete = new
HashSet<>();
        // Process each message.
        response.messages().forEach(message -> {
            logger.info("Message id: {}", message.messageId());
            // Deserialize JSON message body to a S3EventNotification
object
            // to access messages in an object-oriented way.
            S3EventNotification event =
S3EventNotification.fromJson(message.body());

            // Log the S3 event notification record details.
            if (event.getRecords() != null) {
                event.getRecords().forEach(record -> {
                    String eventName = record.getEventName();
                    String key = record.getS3().getObject().getKey();
                    logger.info(record.toString());
                    logger.info("Event name is {} and key is {}",
eventName, key);
                });
            }
            // Add logged messages to collection for batch deletion.
            messagesToDelete.add(DeleteMessageBatchRequestEntry.builder()
                .id(message.messageId())
                .receiptHandle(message.receiptHandle())
                .build());
        });
        // Delete messages.
        if (!messagesToDelete.isEmpty()) {

sqsClient.deleteMessageBatch(DeleteMessageBatchRequest.builder()
            .queueUrl(queueUrl)
            .entries(messagesToDelete)
            .build()
        ).join();
        }
    } // End of while block.
} catch (InterruptedException | ExecutionException e) {
    throw new RuntimeException(e);
}
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
- [DeleteMessageBatch](#)
- [GetQueueAttributes](#)
- [PutBucketNotificationConfiguration](#)
- [ReceiveMessage](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 儲存 EXIF 和其他影像資訊

以下程式碼範例顯示做法：

- 從 JPG、JPEG 或 PNG 檔案中取得 EXIF 資訊。
- 將映像檔案上傳至 Amazon S3 儲存貯體。
- 使用 Amazon Rekognition 識別檔案中的三個主要屬性 (標籤)。
- 將 EXIF 和標籤資訊新增至區域中的 Amazon DynamoDB 資料表。

Rust

適用於 Rust 的 SDK

從 JPG、JPEG 或 PNG 檔案獲取 EXIF 資訊，將映像檔案上傳至 Amazon S3 儲存貯體，使用 Amazon Rekognition 識別三個主要屬性 (Amazon Rekognition 中的標籤)，然後將 EXIF 和標籤資訊新增至區域中的 Amazon DynamoDB 資料表中。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- DynamoDB
- Amazon Rekognition
- Amazon S3

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 將 S3 事件通知傳送至 Amazon EventBridge

下列程式碼範例示範如何啟用儲存貯體，將 S3 事件通知傳送至 EventBridge，並將通知路由至 Amazon SNS 主題和 Amazon SQS 佇列。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/** This method configures a bucket to send events to AWS EventBridge and
    creates a rule
    * to route the S3 object created events to a topic and a queue.
    *
    * @param bucketName Name of existing bucket
    * @param topicArn ARN of existing topic to receive S3 event notifications
    * @param queueArn ARN of existing queue to receive S3 event notifications
    *
    * An AWS CloudFormation stack sets up the bucket, queue, topic before the
    method runs.
    */
    public static String setBucketNotificationToEventBridge(String bucketName,
String topicArn, String queueArn) {
        try {
            // Enable bucket to emit S3 Event notifications to EventBridge.
            s3Client.putBucketNotificationConfiguration(b -> b
                .bucket(bucketName)
                .notificationConfiguration(b1 -> b1
                    .eventBridgeConfiguration(
                        SdkBuilder::build)
                ).build()).join();

            // Create an EventBridge rule to route Object Created notifications.
            PutRuleRequest putRuleRequest = PutRuleRequest.builder()
```

```

        .name(RULE_NAME)
        .eventPattern("""
            {
                "source": ["aws.s3"],
                "detail-type": ["Object Created"],
                "detail": {
                    "bucket": {
                        "name": ["%s"]
                    }
                }
            }
            """).formatted(bucketName))
        .build();

// Add the rule to the default event bus.
PutRuleResponse putRuleResponse =
eventBridgeClient.putRule(putRuleRequest)
    .whenComplete((r, t) -> {
        if (t != null) {
            logger.error("Error creating event bus rule: " +
t.getMessage(), t);
            throw new RuntimeException(t.getCause().getMessage(),
t);
        }
        logger.info("Event bus rule creation request sent
successfully. ARN is: {}", r.ruleArn());
    }).join();

// Add the existing SNS topic and SQS queue as targets to the rule.
eventBridgeClient.putTargets(b -> b
    .eventBusName("default")
    .rule(RULE_NAME)
    .targets(List.of (
        Target.builder()
            .arn(queueArn)
            .id("Queue")
            .build(),
        Target.builder()
            .arn(topicArn)
            .id("Topic")
            .build()
    )
    ).join();
return putRuleResponse.ruleArn();

```

```
    } catch (S3Exception e) {  
        System.err.println(e.awsErrorDetails().errorMessage());  
        System.exit(1);  
    }  
    return null;  
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
 - [PutBucketNotificationConfiguration](#)
 - [PutRule](#)
 - [PutTargets](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 追蹤 Amazon S3 物件上傳或下載 AWS

下列程式碼範例示範如何追蹤 Amazon S3 物件上傳或下載。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

追蹤檔案上傳的進度。

```
public void trackUploadFile(S3TransferManager transferManager, String  
bucketName,  
                           String key, URI filePathURI) {  
    UploadFileRequest uploadFileRequest = UploadFileRequest.builder()  
        .putObjectRequest(b -> b.bucket(bucketName).key(key))  
        .addTransferListener(LoggingTransferListener.create()) // Add  
listener.  
        .source(Paths.get(filePathURI))
```

```

        .build();

        FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);

        fileUpload.completionFuture().join();
    /*
        The SDK provides a LoggingTransferListener implementation of the
        TransferListener interface.
        You can also implement the interface to provide your own logic.

        Configure log4j2 with settings such as the following.
        <Configuration status="WARN">
            <Appenders>
                <Console name="AlignedConsoleAppender"
target="SYSTEM_OUT">
                    <PatternLayout pattern="%m%n"/>
                </Console>
            </Appenders>

            <Loggers>
                <logger
name="software.amazon.awssdk.transfer.s3.progress.LoggingTransferListener"
level="INFO" additivity="false">
                    <AppenderRef ref="AlignedConsoleAppender"/>
                </logger>
            </Loggers>
        </Configuration>

        Log4j2 logs the progress. The following is example output for a 21.3
        MB file upload.
        Transfer initiated...
        |                               | 0.0%
        |=====| 21.1%
        |=====| 60.5%
        |=====| 100.0%
        Transfer complete!
    */
}

```

追蹤檔案下載的進度。

```

    public void trackDownloadFile(S3TransferManager transferManager, String
    bucketName,
                                String key, String downloadedFilePath) {
        DownloadFileRequest downloadFileRequest = DownloadFileRequest.builder()
            .getObjectRequest(b -> b.bucket(bucketName).key(key))
            .addTransferListener(LoggingTransferListener.create()) // Add
listener.
                                .destination(Paths.get(downloadedFilePath))
                                .build();

```

```

        FileDownload downloadFile =
transferManager.downloadFile(downloadFileRequest);

```

```

        CompletedFileDownload downloadResult =
downloadFile.completionFuture().join();
        /*

```

The SDK provides a `LoggingTransferListener` implementation of the `TransferListener` interface.

You can also implement the interface to provide your own logic.

Configure log4j2 with settings such as the following.

```

<Configuration status="WARN">
    <Appenders>
        <Console name="AlignedConsoleAppender"
target="SYSTEM_OUT">
            <PatternLayout pattern="%m%n"/>
        </Console>
    </Appenders>

    <Loggers>
        <logger
name="software.amazon.awssdk.transfer.s3.progress.LoggingTransferListener"
level="INFO" additivity="false">
            <AppenderRef ref="AlignedConsoleAppender"/>
        </logger>
    </Loggers>
</Configuration>

```

Log4j2 logs the progress. The following is example output for a 21.3 MB file download.

```

Transfer initiated...
|=====          | 39.4%
|=====          | 78.8%

```

```
      |=====| 100.0%  
      Transfer complete!  
  
      */  
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
- [GetObject](#)
- [PutObject](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 S3 Object Lambda 轉換應用程式的資料

下列程式碼範例示範如何使用 S3 Object Lambda 轉換應用程式的資料。

.NET

適用於 .NET 的 SDK

展示如何將自訂程式碼新增至標準 S3 GET 請求，以修改從 S3 擷取的請求物件，如此一來，物件方能符合請求用戶端或應用程式的需求。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Lambda
- Amazon S3

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 進行單元和整合測試的範例方法

下列程式碼範例示範如何在使用 AWS SDK 撰寫單元和整合測試時，提供最佳實務技術的範例。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

Cargo.toml 用於測試範例。

```
[package]
name = "testing-examples"
version = "0.1.0"
authors = [
  "John Disanti <jdisanti@amazon.com>",
  "Doug Schwartz <dougsch@amazon.com>",
]
edition = "2021"

[dependencies]
async-trait = "0.1.51"
aws-config = { version = "1.0.1", features = ["behavior-version-latest"] }
aws-credential-types = { version = "1.0.1", features = [ "hardcoded-credentials", ] }
aws-sdk-s3 = { version = "1.4.0" }
aws-smithy-types = { version = "1.0.1" }
aws-smithy-runtime = { version = "1.0.1", features = ["test-util"] }
aws-smithy-runtime-api = { version = "1.0.1", features = ["test-util"] }
aws-types = { version = "1.0.1" }
clap = { version = "4.4", features = ["derive"] }
http = "0.2.9"
mockall = "0.11.4"
serde_json = "1"
tokio = { version = "1.20.1", features = ["full"] }
tracing-subscriber = { version = "0.3.15", features = ["env-filter"] }

[[bin]]
name = "main"
path = "src/main.rs"
```

使用自動模擬和服務包裝函式的單元測試範例。

```
use aws_sdk_s3 as s3;
#[allow(unused_imports)]
use mockall::automock;

use s3::operation::list_objects_v2::{ListObjectsV2Error, ListObjectsV2Output};

#[cfg(test)]
pub use MockS3Impl as S3;
#[cfg(not(test))]
pub use S3Impl as S3;

#[allow(dead_code)]
pub struct S3Impl {
    inner: s3::Client,
}

#[cfg_attr(test, automock)]
impl S3Impl {
    #[allow(dead_code)]
    pub fn new(inner: s3::Client) -> Self {
        Self { inner }
    }

    #[allow(dead_code)]
    pub async fn list_objects(
        &self,
        bucket: &str,
        prefix: &str,
        continuation_token: Option<String>,
    ) -> Result<ListObjectsV2Output, s3::error::SdkError<ListObjectsV2Error>> {
        self.inner
            .list_objects_v2()
            .bucket(bucket)
            .prefix(prefix)
            .set_continuation_token(continuation_token)
            .send()
            .await
    }
}

#[allow(dead_code)]
```



```

pub async fn determine_prefix_file_size(
    // Now we take a reference to our trait object instead of the S3 client
    // s3_list: ListObjectsService,
    s3_list: S3,
    bucket: &str,
    prefix: &str,
) -> Result<usize, s3::Error> {
    let mut next_token: Option<String> = None;
    let mut total_size_bytes = 0;
    loop {
        let result = s3_list
            .list_objects(bucket, prefix, next_token.take())
            .await?;

        // Add up the file sizes we got back
        for object in result.contents() {
            total_size_bytes += object.size().unwrap_or(0) as usize;
        }

        // Handle pagination, and break the loop if there are no more pages
        next_token = result.next_continuation_token.clone();
        if next_token.is_none() {
            break;
        }
    }
    Ok(total_size_bytes)
}

#[cfg(test)]
mod test {
    use super::*;
    use mockall::predicate::eq;

    #[tokio::test]
    async fn test_single_page() {
        let mut mock = MockS3Impl::default();
        mock.expect_list_objects()
            .with(eq("test-bucket"), eq("test-prefix"), eq(None))
            .return_once(|_, _, _| {
                Ok(ListObjectsV2Output::builder()
                    .set_contents(Some(vec![
                        // Mock content for ListObjectsV2 response
                        s3::types::Object::builder().size(5).build(),
                        s3::types::Object::builder().size(2).build(),
                    ]))
            })
    }
}

```

```

        ]))
        .build())
    });

    // Run the code we want to test with it
    let size = determine_prefix_file_size(mock, "test-bucket", "test-prefix")
        .await
        .unwrap();

    // Verify we got the correct total size back
    assert_eq!(7, size);
}

#[tokio::test]
async fn test_multiple_pages() {
    // Create the Mock instance with two pages of objects now
    let mut mock = MockS3Impl::default();
    mock.expect_list_objects()
        .with(eq("test-bucket"), eq("test-prefix"), eq(None))
        .return_once(|_, _, _| {
            Ok(ListObjectsV2Output::builder()
                .set_contents(Some(vec![
                    // Mock content for ListObjectsV2 response
                    s3::types::Object::builder().size(5).build(),
                    s3::types::Object::builder().size(2).build(),
                ]))
                .set_next_continuation_token(Some("next".to_string()))
                .build())
        });
    mock.expect_list_objects()
        .with(
            eq("test-bucket"),
            eq("test-prefix"),
            eq(Some("next".to_string())),
        )
        .return_once(|_, _, _| {
            Ok(ListObjectsV2Output::builder()
                .set_contents(Some(vec![
                    // Mock content for ListObjectsV2 response
                    s3::types::Object::builder().size(3).build(),
                    s3::types::Object::builder().size(9).build(),
                ]))
                .build())
        });
}

```

```
// Run the code we want to test with it
let size = determine_prefix_file_size(mock, "test-bucket", "test-prefix")
    .await
    .unwrap();

assert_eq!(19, size);
}
}
```

使用 `StaticReplayClient` 的整合測試範例。

```
use aws_sdk_s3 as s3;

#[allow(dead_code)]
pub async fn determine_prefix_file_size(
    // Now we take a reference to our trait object instead of the S3 client
    // s3_list: ListObjectsService,
    s3: s3::Client,
    bucket: &str,
    prefix: &str,
) -> Result<usize, s3::Error> {
    let mut next_token: Option<String> = None;
    let mut total_size_bytes = 0;
    loop {
        let result = s3
            .list_objects_v2()
            .prefix(prefix)
            .bucket(bucket)
            .set_continuation_token(next_token.take())
            .send()
            .await?;

        // Add up the file sizes we got back
        for object in result.contents() {
            total_size_bytes += object.size().unwrap_or(0) as usize;
        }

        // Handle pagination, and break the loop if there are no more pages
        next_token = result.next_continuation_token.clone();
        if next_token.is_none() {
```

```

        break;
    }
}
Ok(total_size_bytes)
}

#[allow(dead_code)]
fn make_s3_test_credentials() -> s3::config::Credentials {
    s3::config::Credentials::new(
        "ATESTCLIENT",
        "astestsecretkey",
        Some("atestsessiontoken".to_string()),
        None,
        "",
    )
}

#[cfg(test)]
mod test {
    use super::*;
    use aws_config::BehaviorVersion;
    use aws_sdk_s3 as s3;
    use aws_smithy_runtime::client::http::test_util::{ReplayEvent,
StaticReplayClient};
    use aws_smithy_types::body::SdkBody;

    #[tokio::test]
    async fn test_single_page() {
        let page_1 = ReplayEvent::new(
            http::Request::builder()
                .method("GET")
                .uri("https://test-bucket.s3.us-east-1.amazonaws.com/?list-
type=2&prefix=test-prefix")
                .body(SdkBody::empty())
                .unwrap(),
            http::Response::builder()
                .status(200)
                .body(SdkBody::from(include_str!("./testing/
response_1.xml")))
                .unwrap(),
        );
        let replay_client = StaticReplayClient::new(vec![page_1]);
        let client: s3::Client = s3::Client::from_conf(
            s3::Config::builder()

```

```

        .behavior_version(BehaviorVersion::latest())
        .credentials_provider(make_s3_test_credentials())
        .region(s3::config::Region::new("us-east-1"))
        .http_client(replay_client.clone())
        .build(),
    );

    // Run the code we want to test with it
    let size = determine_prefix_file_size(client, "test-bucket", "test-
prefix")
        .await
        .unwrap();

    // Verify we got the correct total size back
    assert_eq!(7, size);
    replay_client.assert_requests_match(&[]);
}

#[tokio::test]
async fn test_multiple_pages() {
    let page_1 = ReplayEvent::new(
        http::Request::builder()
            .method("GET")
            .uri("https://test-bucket.s3.us-east-1.amazonaws.com/?list-
type=2&prefix=test-prefix")
            .body(SdkBody::empty())
            .unwrap(),
        http::Response::builder()
            .status(200)
            .body(SdkBody::from(include_str!("./testing/
response_multi_1.xml")))
            .unwrap(),
    );
    let page_2 = ReplayEvent::new(
        http::Request::builder()
            .method("GET")
            .uri("https://test-bucket.s3.us-east-1.amazonaws.com/?list-
type=2&prefix=test-prefix&continuation-token=next")
            .body(SdkBody::empty())
            .unwrap(),
        http::Response::builder()
            .status(200)
            .body(SdkBody::from(include_str!("./testing/
response_multi_2.xml")))

```

```
        .unwrap(),
    );
    let replay_client = StaticReplayClient::new(vec![page_1, page_2]);
    let client: s3::Client = s3::Client::from_conf(
        s3::Config::builder()
            .behavior_version(BehaviorVersion::latest())
            .credentials_provider(make_s3_test_credentials())
            .region(s3::config::Region::new("us-east-1"))
            .http_client(replay_client.clone())
            .build(),
    );

    // Run the code we want to test with it
    let size = determine_prefix_file_size(client, "test-bucket", "test-
prefix")
        .await
        .unwrap();

    assert_eq!(19, size);

    replay_client.assert_requests_match(&[]);
}
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

以遞迴的方式將本機目錄上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體

下列程式碼範例示範如何以遞迴的方式將本機目錄上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用 [S3TransferManager](#) 上傳本機目錄。檢視[完整檔案](#)並[測試](#)。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedDirectoryUpload;
import software.amazon.awssdk.transfer.s3.model.DirectoryUpload;
import software.amazon.awssdk.transfer.s3.model.UploadDirectoryRequest;

import java.net.URI;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.file.Paths;

    public Integer uploadDirectory(S3TransferManager transferManager,
        URI sourceDirectory, String bucketName) {
        DirectoryUpload directoryUpload =
transferManager.uploadDirectory(UploadDirectoryRequest.builder()
        .source(Paths.get(sourceDirectory))
        .bucket(bucketName)
        .build());

        CompletedDirectoryUpload completedDirectoryUpload =
directoryUpload.completionFuture().join();
        completedDirectoryUpload.failedTransfers()
            .forEach(fail -> logger.warn("Object [{}] failed to transfer",
fail.toString()));
        return completedDirectoryUpload.failedTransfers().size();
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [UploadDirectory](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 在 Amazon S3 之間上傳或下載大型檔案

下列程式碼範例示範如何將大型檔案上傳至 Amazon S3，以及從中下載大型檔案。

如需詳細資訊，請參閱 [使用分段上傳以上傳物件](#)。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用 Amazon S3 TransferUtility 呼叫可在 S3 儲存貯體之間傳輸檔案的函數。

```
global using System.Text;
global using Amazon.S3;
global using Amazon.S3.Model;
global using Amazon.S3.Transfer;
global using TransferUtilityBasics;

// This Amazon S3 client uses the default user credentials
// defined for this computer.
using Microsoft.Extensions.Configuration;

IAmazonS3 client = new AmazonS3Client();
var transferUtil = new TransferUtility(client);
IConfiguration _configuration;

_configuration = new ConfigurationBuilder()
    .SetBasePath(Directory.GetCurrentDirectory())
    .AddJsonFile("settings.json") // Load test settings from JSON file.
```



```
.AddJsonFile("settings.local.json",
    true) // Optionally load local settings.
.Build();

// Edit the values in settings.json to use an S3 bucket and files that
// exist on your AWS account and on the local computer where you
// run this scenario.
var bucketName = _configuration["BucketName"];
var localPath =
    $"{Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData)}\
\TransferFolder";

DisplayInstructions();

PressEnter();

Console.WriteLine();

// Upload a single file to an S3 bucket.
DisplayTitle("Upload a single file");

var fileToUpload = _configuration["FileToUpload"];
Console.WriteLine($"Uploading {fileToUpload} to the S3 bucket, {bucketName}.");

var success = await TransferMethods.UploadSingleFileAsync(transferUtil,
    bucketName, fileToUpload, localPath);
if (success)
{
    Console.WriteLine($"Successfully uploaded the file, {fileToUpload} to
    {bucketName}.");
}

PressEnter();

// Upload a local directory to an S3 bucket.
DisplayTitle("Upload all files from a local directory");
Console.WriteLine("Upload all the files in a local folder to an S3 bucket.");
const string keyPrefix = "UploadFolder";
var uploadPath = $"{localPath}\\UploadFolder";

Console.WriteLine($"Uploading the files in {uploadPath} to {bucketName}");
DisplayTitle($"{uploadPath} files");
DisplayLocalFiles(uploadPath);
Console.WriteLine();
```

```
PressEnter();

success = await TransferMethods.UploadFullDirectoryAsync(transferUtil,
    bucketName, keyPrefix, uploadPath);
if (success)
{
    Console.WriteLine($"Successfully uploaded the files in {uploadPath} to
{bucketName}.");
    Console.WriteLine($"{bucketName} currently contains the following files:");
    await DisplayBucketFiles(client, bucketName, keyPrefix);
    Console.WriteLine();
}

PressEnter();

// Download a single file from an S3 bucket.
DisplayTitle("Download a single file");
Console.WriteLine("Now we will download a single file from an S3 bucket.");

var keyName = _configuration["FileToDownload"];

Console.WriteLine($"Downloading {keyName} from {bucketName}.");

success = await TransferMethods.DownloadSingleFileAsync(transferUtil, bucketName,
    keyName, localPath);
if (success)
{
    Console.WriteLine($"Successfully downloaded the file, {keyName} from
{bucketName}.");
}

PressEnter();

// Download the contents of a directory from an S3 bucket.
DisplayTitle("Download the contents of an S3 bucket");
var s3Path = _configuration["S3Path"];
var downloadPath = $"{localPath}\\{s3Path}";

Console.WriteLine($"Downloading the contents of {bucketName}\\{s3Path}");
Console.WriteLine($"{bucketName}\\{s3Path} contains the following files:");
await DisplayBucketFiles(client, bucketName, s3Path);
Console.WriteLine();
```

```
success = await TransferMethods.DownloadS3DirectoryAsync(transferUtil,
    bucketName, s3Path, downloadPath);
if (success)
{
    Console.WriteLine($"Downloaded the files in {bucketName} to
{downloadPath}.");
    Console.WriteLine($"{downloadPath} now contains the following files:");
    DisplayLocalFiles(downloadPath);
}

Console.WriteLine("\nThe TransferUtility Basics application has completed.");
PressEnter();

// Displays the title for a section of the scenario.
static void DisplayTitle(string titleText)
{
    var sepBar = new string('-', Console.WindowWidth);

    Console.WriteLine(sepBar);
    Console.WriteLine(CenterText(titleText));
    Console.WriteLine(sepBar);
}

// Displays a description of the actions to be performed by the scenario.
static void DisplayInstructions()
{
    var sepBar = new string('-', Console.WindowWidth);

    DisplayTitle("Amazon S3 Transfer Utility Basics");
    Console.WriteLine("This program shows how to use the Amazon S3 Transfer
Utility.");
    Console.WriteLine("It performs the following actions:");
    Console.WriteLine("\t1. Upload a single object to an S3 bucket.");
    Console.WriteLine("\t2. Upload an entire directory from the local computer to
an\n\t S3 bucket.");
    Console.WriteLine("\t3. Download a single object from an S3 bucket.");
    Console.WriteLine("\t4. Download the objects in an S3 bucket to a local
directory.");
    Console.WriteLine($"
{sepBar}");
}

// Pauses the scenario.
static void PressEnter()
{

```

```
        Console.WriteLine("Press <Enter> to continue.");
        _ = Console.ReadLine();
        Console.WriteLine("\n");
    }

    // Returns the string textToCenter, padded on the left with spaces
    // that center the text on the console display.
    static string CenterText(string textToCenter)
    {
        var centeredText = new StringBuilder();
        var screenWidth = Console.WindowWidth;
        centeredText.Append(new string(' ', (int)(screenWidth -
            textToCenter.Length) / 2));
        centeredText.Append(textToCenter);
        return centeredText.ToString();
    }

    // Displays a list of file names included in the specified path.
    static void DisplayLocalFiles(string localPath)
    {
        var fileList = Directory.GetFiles(localPath);
        if (fileList.Length > 0)
        {
            foreach (var fileName in fileList)
            {
                Console.WriteLine(fileName);
            }
        }
    }

    // Displays a list of the files in the specified S3 bucket and prefix.
    static async Task DisplayBucketFiles(IAmazonS3 client, string bucketName, string
        s3Path)
    {
        ListObjectsV2Request request = new()
        {
            BucketName = bucketName,
            Prefix = s3Path,
            MaxKeys = 5,
        };

        var response = new ListObjectsV2Response();

        do
```

```

{
    response = await client.ListObjectsV2Async(request);

    response.S3Objects
        .ForEach(obj => Console.WriteLine($"{obj.Key}"));

    // If the response is truncated, set the request ContinuationToken
    // from the NextContinuationToken property of the response.
    request.ContinuationToken = response.NextContinuationToken;
} while (response.IsTruncated);
}

```

上傳單一檔案。

```

/// <summary>
/// Uploads a single file from the local computer to an S3 bucket.
/// </summary>
/// <param name="transferUtil">The transfer initialized TransferUtility
/// object.</param>
/// <param name="bucketName">The name of the S3 bucket where the file
/// will be stored.</param>
/// <param name="fileName">The name of the file to upload.</param>
/// <param name="localPath">The local path where the file is stored.</
param>
/// <returns>A boolean value indicating the success of the action.</
returns>
public static async Task<bool> UploadSingleFileAsync(
    TransferUtility transferUtil,
    string bucketName,
    string fileName,
    string localPath)
{
    if (File.Exists($"{localPath}\\{fileName}"))
    {
        try
        {
            await transferUtil.UploadAsync(new
TransferUtilityUploadRequest
            {
                BucketName = bucketName,

```

```

        Key = fileName,
        FilePath = $"{localPath}\\{fileName}",
    });

    return true;
}
catch (AmazonS3Exception s3Ex)
{
    Console.WriteLine($"Could not upload {fileName} from
{localPath} because:");
    Console.WriteLine(s3Ex.Message);
    return false;
}
}
else
{
    Console.WriteLine($"{fileName} does not exist in {localPath}");
    return false;
}
}

```

上傳整個本機目錄。

```

/// <summary>
/// Uploads all the files in a local directory to a directory in an S3
/// bucket.
/// </summary>
/// <param name="transferUtil">The transfer initialized TransferUtility
/// object.</param>
/// <param name="bucketName">The name of the S3 bucket where the files
/// will be stored.</param>
/// <param name="keyPrefix">The key prefix is the S3 directory where
/// the files will be stored.</param>
/// <param name="localPath">The local directory that contains the files
/// to be uploaded.</param>
/// <returns>A Boolean value representing the success of the action.</
returns>
public static async Task<bool> UploadFullDirectoryAsync(
    TransferUtility transferUtil,
    string bucketName,
    string keyPrefix,

```

```
        string localPath)
    {
        if (Directory.Exists(localPath))
        {
            try
            {
                await transferUtil.UploadDirectoryAsync(new
TransferUtilityUploadDirectoryRequest
                {
                    BucketName = bucketName,
                    KeyPrefix = keyPrefix,
                    Directory = localPath,
                });

                return true;
            }
            catch (AmazonS3Exception s3Ex)
            {
                Console.WriteLine($"Can't upload the contents of {localPath}
because:");

                Console.WriteLine(s3Ex?.Message);
                return false;
            }
        }
        else
        {
            Console.WriteLine($"The directory {localPath} does not exist.");
            return false;
        }
    }
}
```

下載單一檔案。

```
/// <summary>
/// Download a single file from an S3 bucket to the local computer.
/// </summary>
/// <param name="transferUtil">The transfer initialized TransferUtility
/// object.</param>
/// <param name="bucketName">The name of the S3 bucket containing the
/// file to download.</param>
```

```

    /// <param name="keyName">The name of the file to download.</param>
    /// <param name="localPath">The path on the local computer where the
    /// downloaded file will be saved.</param>
    /// <returns>A Boolean value indicating the results of the action.</
returns>
    public static async Task<bool> DownloadSingleFileAsync(
        TransferUtility transferUtil,
        string bucketName,
        string keyName,
        string localPath)
    {
        await transferUtil.DownloadAsync(new TransferUtilityDownloadRequest
        {
            BucketName = bucketName,
            Key = keyName,
            FilePath = $"{localPath}\\{keyName}",
        });

        return (File.Exists($"{localPath}\\{keyName}"));
    }

```

下載 S3 儲存貯體的內容。

```

    /// <summary>
    /// Downloads the contents of a directory in an S3 bucket to a
    /// directory on the local computer.
    /// </summary>
    /// <param name="transferUtil">The transfer initialized TransferUtility
    /// object.</param>
    /// <param name="bucketName">The bucket containing the files to
download.</param>
    /// <param name="s3Path">The S3 directory where the files are located.</
param>
    /// <param name="localPath">The local path to which the files will be
    /// saved.</param>
    /// <returns>A Boolean value representing the success of the action.</
returns>
    public static async Task<bool> DownloadS3DirectoryAsync(
        TransferUtility transferUtil,
        string bucketName,

```



```
        string s3Path,
        string localPath)
    {
        int fileCount = 0;

        // If the directory doesn't exist, it will be created.
        if (Directory.Exists(s3Path))
        {
            var files = Directory.GetFiles(localPath);
            fileCount = files.Length;
        }

        await transferUtil.DownloadDirectoryAsync(new
TransferUtilityDownloadDirectoryRequest
        {
            BucketName = bucketName,
            LocalDirectory = localPath,
            S3Directory = s3Path,
        });

        if (Directory.Exists(localPath))
        {
            var files = Directory.GetFiles(localPath);
            if (files.Length > fileCount)
            {
                return true;
            }

            // No change in the number of files. Assume
            // the download failed.
            return false;
        }

        // The local directory doesn't exist. No files
        // were downloaded.
        return false;
    }
}
```

使用傳 TransferUtility 追蹤上傳進度。

```
using System;
```

```
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Transfer;

/// <summary>
/// This example shows how to track the progress of a multipart upload
/// using the Amazon Simple Storage Service (Amazon S3) TransferUtility to
/// upload to an Amazon S3 bucket.
/// </summary>
public class TrackMPUUsingHighLevelAPI
{
    public static async Task Main()
    {
        string bucketName = "amzn-s3-demo-bucket";
        string keyName = "sample_pic.png";
        string path = "filepath/directory/";
        string filePath = $"{path}{keyName}";

        // If the AWS Region defined for your default user is different
        // from the Region where your Amazon S3 bucket is located,
        // pass the Region name to the Amazon S3 client object's constructor.
        // For example: RegionEndpoint.USWest2 or RegionEndpoint.USEast2.
        IAmazonS3 client = new AmazonS3Client();

        await TrackMPUAsync(client, bucketName, filePath, keyName);
    }

    /// <summary>
    /// Starts an Amazon S3 multipart upload and assigns an event handler to
    /// track the progress of the upload.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
    /// perform the multipart upload.</param>
    /// <param name="bucketName">The name of the bucket to which to upload
    /// the file.</param>
    /// <param name="filePath">The path, including the file name of the
    /// file to be uploaded to the Amazon S3 bucket.</param>
    /// <param name="keyName">The file name to be used in the
    /// destination Amazon S3 bucket.</param>
    public static async Task TrackMPUAsync(
        IAmazonS3 client,
        string bucketName,
        string filePath,
        string keyName)
```

```
{
    try
    {
        var fileTransferUtility = new TransferUtility(client);

        // Use TransferUtilityUploadRequest to configure options.
        // In this example we subscribe to an event.
        var uploadRequest =
            new TransferUtilityUploadRequest
            {
                BucketName = bucketName,
                FilePath = filePath,
                Key = keyName,
            };

        uploadRequest.UploadProgressEvent +=
            new EventHandler<UploadProgressArgs>(
                UploadRequest_UploadPartProgressEvent);

        await fileTransferUtility.UploadAsync(uploadRequest);
        Console.WriteLine("Upload completed");
    }
    catch (AmazonS3Exception ex)
    {
        Console.WriteLine($"Error:: {ex.Message}");
    }
}

/// <summary>
/// Event handler to check the progress of the multipart upload.
/// </summary>
/// <param name="sender">The object that raised the event.</param>
/// <param name="e">The object that contains multipart upload
/// information.</param>
public static void UploadRequest_UploadPartProgressEvent(object sender,
UploadProgressArgs e)
{
    // Process event.
    Console.WriteLine($"{e.TransferredBytes}/{e.TotalBytes}");
}
}
```

使用加密上傳物件。

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Security.Cryptography;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// Uses the Amazon Simple Storage Service (Amazon S3) low level API to
/// perform a multipart upload to an Amazon S3 bucket.
/// </summary>
public class SSECLowLevelMPUCopyObject
{
    public static async Task Main()
    {
        string existingBucketName = "amzn-s3-demo-bucket";
        string sourceKeyName = "sample_file.txt";
        string targetKeyName = "sample_file_copy.txt";
        string filePath = $"sample\\{targetKeyName}";

        // If the AWS Region defined for your default user is different
        // from the Region where your Amazon S3 bucket is located,
        // pass the Region name to the Amazon S3 client object's constructor.
        // For example: RegionEndpoint.USEast1.
        IAmazonS3 client = new AmazonS3Client();

        // Create the encryption key.
        var base64Key = CreateEncryptionKey();

        await CreateSampleObjUsingClientEncryptionKeyAsync(
            client,
            existingBucketName,
            sourceKeyName,
            filePath,
            base64Key);
    }

    /// <summary>
    /// Creates the encryption key to use with the multipart upload.
    /// </summary>
    /// <returns>A string containing the base64-encoded key for encrypting
```

```

    /// the multipart upload.</returns>
    public static string CreateEncryptionKey()
    {
        Aes aesEncryption = Aes.Create();
        aesEncryption.KeySize = 256;
        aesEncryption.GenerateKey();
        string base64Key = Convert.ToBase64String(aesEncryption.Key);
        return base64Key;
    }

    /// <summary>
    /// Creates and uploads an object using a multipart upload.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 object used to
    /// initialize and perform the multipart upload.</param>
    /// <param name="existingBucketName">The name of the bucket to which
    /// the object will be uploaded.</param>
    /// <param name="sourceKeyName">The source object name.</param>
    /// <param name="filePath">The location of the source object.</param>
    /// <param name="base64Key">The encryption key to use with the upload.</
param>
    public static async Task CreateSampleObjUsingClientEncryptionKeyAsync(
        IAmazonS3 client,
        string existingBucketName,
        string sourceKeyName,
        string filePath,
        string base64Key)
    {
        List<UploadPartResponse> uploadResponses = new
List<UploadPartResponse>();

        InitiateMultipartUploadRequest initiateRequest = new
InitiateMultipartUploadRequest
        {
            BucketName = existingBucketName,
            Key = sourceKeyName,
            ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
            ServerSideEncryptionCustomerProvidedKey = base64Key,
        };

        InitiateMultipartUploadResponse initResponse =
            await client.InitiateMultipartUploadAsync(initiateRequest);
    }

```

```
long contentLength = new FileInfo(filePath).Length;
long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB

try
{
    long filePosition = 0;
    for (int i = 1; filePosition < contentLength; i++)
    {
        UploadPartRequest uploadRequest = new UploadPartRequest
        {
            BucketName = existingBucketName,
            Key = sourceKeyName,
            UploadId = initResponse.UploadId,
            PartNumber = i,
            PartSize = partSize,
            FilePosition = filePosition,
            FilePath = filePath,
            ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
            ServerSideEncryptionCustomerProvidedKey = base64Key,
        };

        // Upload part and add response to our list.
        uploadResponses.Add(await
client.UploadPartAsync(uploadRequest));

        filePosition += partSize;
    }

    CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
    {
        BucketName = existingBucketName,
        Key = sourceKeyName,
        UploadId = initResponse.UploadId,
    };
    completeRequest.AddPartETags(uploadResponses);

    CompleteMultipartUploadResponse completeUploadResponse =
        await client.CompleteMultipartUploadAsync(completeRequest);
}
catch (Exception exception)
{
    Console.WriteLine($"Exception occurred: {exception.Message}");
}
```

```
        // If there was an error, abort the multipart upload.
        AbortMultipartUploadRequest abortMPURequest = new
AbortMultipartUploadRequest
    {
        BucketName = existingBucketName,
        Key = sourceKeyName,
        UploadId = initResponse.UploadId,
    };

    await client.AbortMultipartUploadAsync(abortMPURequest);
    }
}
```

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立使用上傳和下載管理員的函數，將資料分成數個部分，並同時傳輸。

```
import (
    "bytes"
    "context"
    "errors"
    "fmt"
    "io"
    "log"
    "os"
    "time"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/feature/s3/manager"
```

```

    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/aws-sdk-go-v2/service/s3/types"
    "github.com/aws/smithy-go"
)

// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
// actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
// bucket
// and object actions.
type BucketBasics struct {
    S3Client *s3.Client
}

// UploadLargeObject uses an upload manager to upload data to an object in a
// bucket.
// The upload manager breaks large data into parts and uploads the parts
// concurrently.
func (basics BucketBasics) UploadLargeObject(ctx context.Context, bucketName
string, objectKey string, largeObject []byte) error {
    largeBuffer := bytes.NewReader(largeObject)
    var partMiBs int64 = 10
    uploader := manager.NewUploader(basics.S3Client, func(u *manager.Uploader) {
        u.PartSize = partMiBs * 1024 * 1024
    })
    _, err := uploader.Upload(ctx, &s3.PutObjectInput{
        Bucket: aws.String(bucketName),
        Key:     aws.String(objectKey),
        Body:    largeBuffer,
    })
    if err != nil {
        var apiErr smithy.APIError
        if errors.As(err, &apiErr) && apiErr.ErrorCode() == "EntityTooLarge" {
            log.Printf("Error while uploading object to %s. The object is too large.\n"+
                "The maximum size for a multipart upload is 5TB.", bucketName)
        } else {
            log.Printf("Couldn't upload large object to %v:%v. Here's why: %v\n",
                bucketName, objectKey, err)
        }
    } else {
        err = s3.NewObjectExistsWaiter(basics.S3Client).Wait(

```



```

    ctx, &s3.HeadObjectInput{Bucket: aws.String(bucketName), Key:
aws.String(objectKey)}, time.Minute)
    if err != nil {
        log.Printf("Failed attempt to wait for object %s to exist.\n", objectKey)
    }
}

return err
}

// DownloadLargeObject uses a download manager to download an object from a
// bucket.
// The download manager gets the data in parts and writes them to a buffer until
// all of
// the data has been downloaded.
func (basics BucketBasics) DownloadLargeObject(ctx context.Context, bucketName
string, objectKey string) ([]byte, error) {
    var partMiBs int64 = 10
    downloader := manager.NewDownloader(basics.S3Client, func(d *manager.Downloader)
    {
        d.PartSize = partMiBs * 1024 * 1024
    })
    buffer := manager.NewWriteAtBuffer([]byte{})
    _, err := downloader.Download(ctx, buffer, &s3.GetObjectInput{
        Bucket: aws.String(bucketName),
        Key:    aws.String(objectKey),
    })
    if err != nil {
        log.Printf("Couldn't download large object from %v:%v. Here's why: %v\n",
            bucketName, objectKey, err)
    }
    return buffer.Bytes(), err
}

```

執行互動式情境，示範如何在內容中使用上傳和下載管理員。

```

import (
    "context"

```

```
"crypto/rand"
"log"
"strings"

"github.com/aws/aws-sdk-go-v2/aws"
"github.com/aws/aws-sdk-go-v2/service/s3"
"github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
"github.com/awsdocs/aws-doc-sdk-examples/gov2/s3/actions"
)

// RunLargeObjectScenario is an interactive example that shows you how to use
// Amazon
// Simple Storage Service (Amazon S3) to upload and download large objects.
//
// 1. Create a bucket.
// 3. Upload a large object to the bucket by using an upload manager.
// 5. Download a large object by using a download manager.
// 8. Delete all objects in the bucket.
// 9. Delete the bucket.
//
// This example creates an Amazon S3 service client from the specified sdkConfig
// so that
// you can replace it with a mocked or stubbed config for unit testing.
//
// It uses a questioner from the `demotools` package to get input during the
// example.
// This package can be found in the ../../demotools folder of this repo.
func RunLargeObjectScenario(ctx context.Context, sdkConfig aws.Config, questioner
demotools.IQuestioner) {
defer func() {
if r := recover(); r != nil {
log.Println("Something went wrong with the demo.")
_, isMock := questioner.(*demotools.MockQuestioner)
if isMock || questioner.AskBool("Do you want to see the full error message (y/
n)?", "y") {
log.Println(r)
}
}
}()

log.Println(strings.Repeat("-", 88))
log.Println("Welcome to the Amazon S3 large object demo.")
log.Println(strings.Repeat("-", 88))
```

```
s3Client := s3.NewFromConfig(sdkConfig)
bucketBasics := actions.BucketBasics{S3Client: s3Client}

bucketName := questioner.Ask("Let's create a bucket. Enter a name for your
bucket:",
    demotools.NotEmpty{})
bucketExists, err := bucketBasics.BucketExists(ctx, bucketName)
if err != nil {
    panic(err)
}
if !bucketExists {
    err = bucketBasics.CreateBucket(ctx, bucketName, sdkConfig.Region)
    if err != nil {
        panic(err)
    } else {
        log.Println("Bucket created.")
    }
}
log.Println(strings.Repeat("-", 88))

mibs := 30
log.Printf("Let's create a slice of %v MiB of random bytes and upload it to your
bucket. ", mibs)
questioner.Ask("Press Enter when you're ready.")
largeBytes := make([]byte, 1024*1024*mibs)
_, _ = rand.Read(largeBytes)
largeKey := "doc-example-large"
log.Println("Uploading...")
err = bucketBasics.UploadLargeObject(ctx, bucketName, largeKey, largeBytes)
if err != nil {
    panic(err)
}
log.Printf("Uploaded %v MiB object as %v", mibs, largeKey)
log.Println(strings.Repeat("-", 88))

log.Printf("Let's download the %v MiB object.", mibs)
questioner.Ask("Press Enter when you're ready.")
log.Println("Downloading...")
largeDownload, err := bucketBasics.DownloadLargeObject(ctx, bucketName,
largeKey)
if err != nil {
    panic(err)
}
log.Printf("Downloaded %v bytes.", len(largeDownload))
```

```
log.Println(strings.Repeat("-", 88))

if questioner.AskBool("Do you want to delete your bucket and all of its "+
    "contents? (y/n)", "y") {
    log.Println("Deleting object.")
    err = bucketBasics.DeleteObjects(ctx, bucketName, []string{largeKey})
    if err != nil {
        panic(err)
    }
    log.Println("Deleting bucket.")
    err = bucketBasics.DeleteBucket(ctx, bucketName)
    if err != nil {
        panic(err)
    }
} else {
    log.Println("Okay. Don't forget to delete objects from your bucket to avoid
charges.")
}
log.Println(strings.Repeat("-", 88))

log.Println("Thanks for watching!")
log.Println(strings.Repeat("-", 88))
}
```

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用 S3TransferManager 呼叫可在 S3 儲存貯體之間傳輸檔案的函數。

```
public Integer downloadObjectsToDirectory(S3TransferManager transferManager,
    URI destinationPathURI, String bucketName) {
    DirectoryDownload directoryDownload =
transferManager.downloadDirectory(DownloadDirectoryRequest.builder()
```

```

        .destination(Paths.get(destinationPathURI))
        .bucket(bucketName)
        .build());
    CompletedDirectoryDownload completedDirectoryDownload =
directoryDownload.completionFuture().join();

    completedDirectoryDownload.failedTransfers()
        .forEach(fail -> logger.warn("Object [{}] failed to transfer",
fail.toString()));
    return completedDirectoryDownload.failedTransfers().size();
}

```

上傳整個本機目錄。

```

public Integer uploadDirectory(S3TransferManager transferManager,
    URI sourceDirectory, String bucketName) {
    DirectoryUpload directoryUpload =
transferManager.uploadDirectory(UploadDirectoryRequest.builder()
        .source(Paths.get(sourceDirectory))
        .bucket(bucketName)
        .build());

    CompletedDirectoryUpload completedDirectoryUpload =
directoryUpload.completionFuture().join();
    completedDirectoryUpload.failedTransfers()
        .forEach(fail -> logger.warn("Object [{}] failed to transfer",
fail.toString()));
    return completedDirectoryUpload.failedTransfers().size();
}

```

上傳單一檔案。

```

public String uploadFile(S3TransferManager transferManager, String
bucketName,
    String key, URI filePathURI) {
    UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
        .putObjectRequest(b -> b.bucket(bucketName).key(key))
        .source(Paths.get(filePathURI))
        .build();

    FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);
}

```

```
CompletedFileUpload uploadResult = fileUpload.completionFuture().join();
return uploadResult.response().eTag();
}
```

這些程式碼範例使用下列匯入。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.S3AsyncClient;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CompletedMultipartUpload;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;
import software.amazon.awssdk.services.s3.model.UploadPartRequest;
import software.amazon.awssdk.services.s3.model.UploadPartResponse;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.FileUpload;
import software.amazon.awssdk.transfer.s3.model.UploadFileRequest;

import java.io.IOException;
import java.io.RandomAccessFile;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.ByteBuffer;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.List;
import java.util.Objects;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;
```

除了 [AWS CRT 型 S3 用戶端](#) 之外，內容大小超過閾值時使用 [S3 Transfer Manager](#) 明確執行分段上傳。預設閾值大小為 8 MB。

```
/**
```

```

    * Uploads a file to an Amazon S3 bucket using the S3TransferManager.
    *
    * @param bucketName the name of the S3 bucket
    * @param key         the object key
    * @param filePath    the file path of the file to be uploaded
    */
    public void multipartUploadWithTransferManager(String bucketName, String key,
String filePath) {
        S3TransferManager transferManager = S3TransferManager.create();
        UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
            .putObjectRequest(b -> b
                .bucket(bucketName)
                .key(key))
            .source(Paths.get(filePath))
            .build();
        FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);
        fileUpload.completionFuture().join();
        transferManager.close();
    }

```

使用 [S3Client API](#) 執行分段上傳。

```

/**
 * Performs a multipart upload to Amazon S3 using the provided S3 client.
 *
 * @param bucketName the name of the S3 bucket
 * @param key         the object key
 * @param filePath    the path to the file to be uploaded
 */
    public void multipartUploadWithS3Client(String bucketName, String key, String
filePath) {

        // Initiate the multipart upload.
        CreateMultipartUploadResponse createMultipartUploadResponse =
s3Client.createMultipartUpload(b -> b
            .bucket(bucketName)
            .key(key));
        String uploadId = createMultipartUploadResponse.uploadId();

        // Upload the parts of the file.
        int partNumber = 1;
        List<CompletedPart> completedParts = new ArrayList<>();

```

```

        ByteBuffer bb = ByteBuffer.allocate(1024 * 1024 * 5); // 5 MB byte buffer

        try (RandomAccessFile file = new RandomAccessFile(filePath, "r")) {
            long fileSize = file.length();
            long position = 0;
            while (position < fileSize) {
                file.seek(position);
                long read = file.getChannel().read(bb);

                bb.flip(); // Swap position and limit before reading from the
buffer.

                UploadPartRequest uploadPartRequest = UploadPartRequest.builder()
                    .bucket(bucketName)
                    .key(key)
                    .uploadId(uploadId)
                    .partNumber(partNumber)
                    .build();

                UploadPartResponse partResponse = s3Client.uploadPart(
                    uploadPartRequest,
                    RequestBody.fromByteBuffer(bb));

                CompletedPart part = CompletedPart.builder()
                    .partNumber(partNumber)
                    .eTag(partResponse.eTag())
                    .build();
                completedParts.add(part);

                bb.clear();
                position += read;
                partNumber++;
            }
        } catch (IOException e) {
            logger.error(e.getMessage());
        }

        // Complete the multipart upload.
        s3Client.completeMultipartUpload(b -> b
            .bucket(bucketName)
            .key(key)
            .uploadId(uploadId)

            .multipartUpload(CompletedMultipartUpload.builder().parts(completedParts).build()));
    }

```


使用啟用了分段支援的 [S3AsyncClient API](#) 來執行分段上傳。

```
/**
 * Uploads a file to an S3 bucket using the S3AsyncClient and enabling
 * multipart support.
 *
 * @param bucketName the name of the S3 bucket
 * @param key         the object key
 * @param filePath    the local file path of the file to be uploaded
 */
public void multipartUploadWithS3AsyncClient(String bucketName, String key,
String filePath) {
    // Enable multipart support.
    S3AsyncClient s3AsyncClient = S3AsyncClient.builder()
        .multipartEnabled(true)
        .build();

    CompletableFuture<PutObjectResponse> response = s3AsyncClient.putObject(b
-> b
        .bucket(bucketName)
        .key(key),
        Paths.get(filePath));

    response.join();
    logger.info("File uploaded in multiple 8 MiB parts using
S3AsyncClient.");
}
```

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

上傳大型檔案。

```
import { S3Client } from "@aws-sdk/client-s3";
import { Upload } from "@aws-sdk/lib-storage";

import {
  ProgressBar,
  logger,
} from "@aws-doc-sdk-examples/lib/utils/util-log.js";

const twentyFiveMB = 25 * 1024 * 1024;

export const createString = (size = twentyFiveMB) => {
  return "x".repeat(size);
};

/**
 * Create a 25MB file and upload it in parts to the specified
 * Amazon S3 bucket.
 * @param {{ bucketName: string, key: string }}
 */
export const main = async ({ bucketName, key }) => {
  const str = createString();
  const buffer = Buffer.from(str, "utf8");
  const progressBar = new ProgressBar({
    description: `Uploading "${key}" to "${bucketName}"`,
    barLength: 30,
  });

  try {
    const upload = new Upload({
      client: new S3Client({}),
      params: {
        Bucket: bucketName,
        Key: key,
        Body: buffer,
      },
    });

    upload.on("httpUploadProgress", ({ loaded, total }) => {
      progressBar.update({ current: loaded, total });
    });

    await upload.done();
  }
}
```

```
    } catch (caught) {
      if (caught instanceof Error && caught.name === "AbortError") {
        logger.error(`Multipart upload was aborted. ${caught.message}`);
      } else {
        throw caught;
      }
    }
  }
};
```

下載大型檔案。

```
import { fileURLToPath } from "node:url";
import { GetObjectCommand, NoSuchKey, S3Client } from "@aws-sdk/client-s3";
import { createWriteStream, rmSync } from "node:fs";

const s3Client = new S3Client({});
const oneMB = 1024 * 1024;

export const getObjectRange = ({ bucket, key, start, end }) => {
  const command = new GetObjectCommand({
    Bucket: bucket,
    Key: key,
    Range: `bytes=${start}-${end}`,
  });

  return s3Client.send(command);
};

/**
 * @param {string | undefined} contentRange
 */
export const getRangeAndLength = (contentRange) => {
  const [range, length] = contentRange.split("/");
  const [start, end] = range.split("-");
  return {
    start: Number.parseInt(start),
    end: Number.parseInt(end),
    length: Number.parseInt(length),
  };
};
```

```

export const isComplete = ({ end, length }) => end === length - 1;

const downloadInChunks = async ({ bucket, key }) => {
  const writeStream = createWriteStream(
    fileURLToPath(new URL(`./${key}`, import.meta.url)),
  ).on("error", (err) => console.error(err));

  let rangeAndLength = { start: -1, end: -1, length: -1 };

  while (!isComplete(rangeAndLength)) {
    const { end } = rangeAndLength;
    const nextRange = { start: end + 1, end: end + oneMB };

    const { ContentRange, Body } = await getObjectRange({
      bucket,
      key,
      ...nextRange,
    });
    console.log(`Downloaded bytes ${nextRange.start} to ${nextRange.end}`);

    writeStream.write(await Body.transformToByteArray());
    rangeAndLength = getRangeAndLength(ContentRange);
  }
};

/**
 * Download a large object from and Amazon S3 bucket.
 *
 * When downloading a large file, you might want to break it down into
 * smaller pieces. Amazon S3 accepts a Range header to specify the start
 * and end of the byte range to be downloaded.
 *
 * @param {{ bucketName: string, key: string }}
 */
export const main = async ({ bucketName, key }) => {
  try {
    await downloadInChunks({
      bucket: bucketName,
      key: key,
    });
  } catch (caught) {
    if (caught instanceof NoSuchKey) {
      console.error(`Failed to download object. No such key "${key}".`);
      rmSync(key);
    }
  }
};

```

```
    }  
  }  
};
```

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立使用多個可用的傳輸管理設定函數傳輸檔案。使用回呼類別將回呼程序寫入檔案傳輸。

```
import sys  
import threading  
  
import boto3  
from boto3.s3.transfer import TransferConfig  
  
MB = 1024 * 1024  
s3 = boto3.resource("s3")  
  
class TransferCallback:  
    """  
    Handle callbacks from the transfer manager.  
  
    The transfer manager periodically calls the __call__ method throughout  
    the upload and download process so that it can take action, such as  
    displaying progress to the user and collecting data about the transfer.  
    """  
  
    def __init__(self, target_size):  
        self._target_size = target_size  
        self._total_transferred = 0  
        self._lock = threading.Lock()  
        self.thread_info = {}
```

```

def __call__(self, bytes_transferred):
    """
    The callback method that is called by the transfer manager.

    Display progress during file transfer and collect per-thread transfer
    data. This method can be called by multiple threads, so shared instance
    data is protected by a thread lock.
    """
    thread = threading.current_thread()
    with self._lock:
        self._total_transferred += bytes_transferred
        if thread.ident not in self.thread_info.keys():
            self.thread_info[thread.ident] = bytes_transferred
        else:
            self.thread_info[thread.ident] += bytes_transferred

        target = self._target_size * MB
        sys.stdout.write(
            f"\r{self._total_transferred} of {target} transferred "
            f"({(self._total_transferred / target) * 100:.2f}%)."
        )
        sys.stdout.flush()

def upload_with_default_configuration(
    local_file_path, bucket_name, object_key, file_size_mb
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, using the default
    configuration.
    """
    transfer_callback = TransferCallback(file_size_mb)
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, Callback=transfer_callback
    )
    return transfer_callback.thread_info

def upload_with_chunksize_and_meta(
    local_file_path, bucket_name, object_key, file_size_mb, metadata=None
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, setting a

```

multipart chunk size and adding metadata to the Amazon S3 object.

The multipart chunk size controls the size of the chunks of data that are sent in the request. A smaller chunk size typically results in the transfer manager using more threads for the upload.

The metadata is a set of key-value pairs that are stored with the object in Amazon S3.

```
"""
```

```
transfer_callback = TransferCallback(file_size_mb)
```

```
config = TransferConfig(multipart_chunksize=1 * MB)
```

```
extra_args = {"Metadata": metadata} if metadata else None
```

```
s3.Bucket(bucket_name).upload_file(
```

```
    local_file_path,
```

```
    object_key,
```

```
    Config=config,
```

```
    ExtraArgs=extra_args,
```

```
    Callback=transfer_callback,
```

```
)
```

```
return transfer_callback.thread_info
```

```
def upload_with_high_threshold(local_file_path, bucket_name, object_key,
    file_size_mb):
```

```
    """
```

Upload a file from a local folder to an Amazon S3 bucket, setting a multipart threshold larger than the size of the file.

Setting a multipart threshold larger than the size of the file results in the transfer manager sending the file as a standard upload instead of a multipart upload.

```
    """
```

```
transfer_callback = TransferCallback(file_size_mb)
```

```
config = TransferConfig(multipart_threshold=file_size_mb * 2 * MB)
```

```
s3.Bucket(bucket_name).upload_file(
```

```
    local_file_path, object_key, Config=config, Callback=transfer_callback
```

```
)
```

```
return transfer_callback.thread_info
```

```
def upload_with_sse(
```

```
    local_file_path, bucket_name, object_key, file_size_mb, sse_key=None
```

```
):
```

```
"""
```

Upload a file from a local folder to an Amazon S3 bucket, adding server-side encryption with customer-provided encryption keys to the object.

When this kind of encryption is specified, Amazon S3 encrypts the object at rest and allows downloads only when the expected encryption key is provided in the download request.

```
"""
```

```
transfer_callback = TransferCallback(file_size_mb)
if sse_key:
    extra_args = {"SSECustomerAlgorithm": "AES256", "SSECustomerKey":
sse_key}
else:
    extra_args = None
s3.Bucket(bucket_name).upload_file(
    local_file_path, object_key, ExtraArgs=extra_args,
    Callback=transfer_callback
)
return transfer_callback.thread_info
```

```
def download_with_default_configuration(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, using the
    default configuration.
    """
    transfer_callback = TransferCallback(file_size_mb)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Callback=transfer_callback
    )
    return transfer_callback.thread_info
```

```
def download_with_single_thread(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, using a
    single thread.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(use_threads=False)
```



```
s3.Bucket(bucket_name).Object(object_key).download_file(
    download_file_path, Config=config, Callback=transfer_callback
)
return transfer_callback.thread_info

def download_with_high_threshold(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, setting a
    multipart threshold larger than the size of the file.

    Setting a multipart threshold larger than the size of the file results
    in the transfer manager sending the file as a standard download instead
    of a multipart download.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(multipart_threshold=file_size_mb * 2 * MB)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info

def download_with_sse(
    bucket_name, object_key, download_file_path, file_size_mb, sse_key
):
    """
    Download a file from an Amazon S3 bucket to a local folder, adding a
    customer-provided encryption key to the request.

    When this kind of encryption is specified, Amazon S3 encrypts the object
    at rest and allows downloads only when the expected encryption key is
    provided in the download request.
    """
    transfer_callback = TransferCallback(file_size_mb)

    if sse_key:
        extra_args = {"SSECustomerAlgorithm": "AES256", "SSECustomerKey":
sse_key}
    else:
        extra_args = None
    s3.Bucket(bucket_name).Object(object_key).download_file(
```

```
        download_file_path, ExtraArgs=extra_args, Callback=transfer_callback
    )
    return transfer_callback.thread_info
```

展示傳輸管理函數並回報結果。

```
import hashlib
import os
import platform
import shutil
import time

import boto3
from boto3.s3.transfer import TransferConfig
from botocore.exceptions import ClientError
from botocore.exceptions import ParamValidationError
from botocore.exceptions import NoCredentialsError

import file_transfer

MB = 1024 * 1024
# These configuration attributes affect both uploads and downloads.
CONFIG_ATTRS = (
    "multipart_threshold",
    "multipart_chunksize",
    "max_concurrency",
    "use_threads",
)
# These configuration attributes affect only downloads.
DOWNLOAD_CONFIG_ATTRS = ("max_io_queue", "io_chunksize", "num_download_attempts")

class TransferDemoManager:
    """
    Manages the demonstration. Collects user input from a command line, reports
    transfer results, maintains a list of artifacts created during the
    demonstration, and cleans them up after the demonstration is completed.
    """

    def __init__(self):
```

```

        self._s3 = boto3.resource("s3")
        self._chore_list = []
        self._create_file_cmd = None
        self._size_multiplier = 0
        self.file_size_mb = 30
        self.demo_folder = None
        self.demo_bucket = None
        self._setup_platform_specific()
        self._terminal_width = shutil.get_terminal_size(fallback=(80, 80))[0]

    def collect_user_info(self):
        """
        Collect local folder and Amazon S3 bucket name from the user. These
        locations are used to store files during the demonstration.
        """
        while not self.demo_folder:
            self.demo_folder = input(
                "Which file folder do you want to use to store " "demonstration
files? "
            )
            if not os.path.isdir(self.demo_folder):
                print(f"{self.demo_folder} isn't a folder!")
                self.demo_folder = None

        while not self.demo_bucket:
            self.demo_bucket = input(
                "Which Amazon S3 bucket do you want to use to store "
"demonstration files? "
            )
            try:
                self._s3.meta.client.head_bucket(Bucket=self.demo_bucket)
            except ParamValidationError as err:
                print(err)
                self.demo_bucket = None
            except ClientError as err:
                print(err)
                print(
                    f"Either {self.demo_bucket} doesn't exist or you don't "
                    f"have access to it."
                )
                self.demo_bucket = None

    def demo(

```

```

        self, question, upload_func, download_func, upload_args=None,
download_args=None
    ):
        """Run a demonstration.

        Ask the user if they want to run this specific demonstration.
        If they say yes, create a file on the local path, upload it
        using the specified upload function, then download it using the
        specified download function.
        """
        if download_args is None:
            download_args = {}
        if upload_args is None:
            upload_args = {}
        question = question.format(self.file_size_mb)
        answer = input(f"{question} (y/n)")
        if answer.lower() == "y":
            local_file_path, object_key, download_file_path =
self._create_demo_file()

            file_transfer.TransferConfig = self._config_wrapper(
                TransferConfig, CONFIG_ATTRS
            )
            self._report_transfer_params(
                "Uploading", local_file_path, object_key, **upload_args
            )
            start_time = time.perf_counter()
            thread_info = upload_func(
                local_file_path,
                self.demo_bucket,
                object_key,
                self.file_size_mb,
                **upload_args,
            )
            end_time = time.perf_counter()
            self._report_transfer_result(thread_info, end_time - start_time)

            file_transfer.TransferConfig = self._config_wrapper(
                TransferConfig, CONFIG_ATTRS + DOWNLOAD_CONFIG_ATTRS
            )
            self._report_transfer_params(
                "Downloading", object_key, download_file_path, **download_args
            )
            start_time = time.perf_counter()

```

```

        thread_info = download_func(
            self.demo_bucket,
            object_key,
            download_file_path,
            self.file_size_mb,
            **download_args,
        )
        end_time = time.perf_counter()
        self._report_transfer_result(thread_info, end_time - start_time)

def last_name_set(self):
    """Get the name set used for the last demo."""
    return self._chore_list[-1]

def cleanup(self):
    """
    Remove files from the demo folder, and uploaded objects from the
    Amazon S3 bucket.
    """
    print("-" * self._terminal_width)
    for local_file_path, s3_object_key, downloaded_file_path in
self._chore_list:
        print(f"Removing {local_file_path}")
        try:
            os.remove(local_file_path)
        except FileNotFoundError as err:
            print(err)

        print(f"Removing {downloaded_file_path}")
        try:
            os.remove(downloaded_file_path)
        except FileNotFoundError as err:
            print(err)

        if self.demo_bucket:
            print(f"Removing {self.demo_bucket}:{s3_object_key}")
            try:
self._s3.Bucket(self.demo_bucket).Object(s3_object_key).delete()
                except ClientError as err:
                    print(err)

def _setup_platform_specific(self):
    """Set up platform-specific command used to create a large file."""

```

```

    if platform.system() == "Windows":
        self._create_file_cmd = "fsutil file createnew {} {}"
        self._size_multiplier = MB
    elif platform.system() == "Linux" or platform.system() == "Darwin":
        self._create_file_cmd = f"dd if=/dev/urandom of={{}} " f"bs={MB}
count={{}}"
        self._size_multiplier = 1
    else:
        raise EnvironmentError(
            f"Demo of platform {platform.system()} isn't supported."
        )

def _create_demo_file(self):
    """
    Create a file in the demo folder specified by the user. Store the local
    path, object name, and download path for later cleanup.

    Only the local file is created by this method. The Amazon S3 object and
    download file are created later during the demonstration.

    Returns:
    A tuple that contains the local file path, object name, and download
    file path.
    """
    file_name_template = "TestFile{}-{}.demo"
    local_suffix = "local"
    object_suffix = "s3object"
    download_suffix = "downloaded"
    file_tag = len(self._chore_list) + 1

    local_file_path = os.path.join(
        self.demo_folder, file_name_template.format(file_tag, local_suffix)
    )

    s3_object_key = file_name_template.format(file_tag, object_suffix)

    downloaded_file_path = os.path.join(
        self.demo_folder, file_name_template.format(file_tag,
download_suffix)
    )

    filled_cmd = self._create_file_cmd.format(
        local_file_path, self.file_size_mb * self._size_multiplier
    )

```

```

        print(
            f"Creating file of size {self.file_size_mb} MB "
            f"in {self.demo_folder} by running:"
        )
        print(f"{'':4}{filled_cmd}")
        os.system(filled_cmd)

        chore = (local_file_path, s3_object_key, downloaded_file_path)
        self._chore_list.append(chore)
        return chore

def _report_transfer_params(self, verb, source_name, dest_name, **kwargs):
    """Report configuration and extra arguments used for a file transfer."""
    print("-" * self._terminal_width)
    print(f"{verb} {source_name} ({self.file_size_mb} MB) to {dest_name}")
    if kwargs:
        print("With extra args:")
        for arg, value in kwargs.items():
            print(f'{"":4}{arg:<20}: {value}')

    @staticmethod
    def ask_user(question):
        """
        Ask the user a yes or no question.

        Returns:
        True when the user answers 'y' or 'Y'; otherwise, False.
        """
        answer = input(f"{question} (y/n) ")
        return answer.lower() == "y"

    @staticmethod
    def _config_wrapper(func, config_attrs):
        def wrapper(*args, **kwargs):
            config = func(*args, **kwargs)
            print("With configuration:")
            for attr in config_attrs:
                print(f'{"":4}{attr:<20}: {getattr(config, attr)}')
            return config

        return wrapper

    @staticmethod

```

```
def _report_transfer_result(thread_info, elapsed):
    """Report the result of a transfer, including per-thread data."""
    print(f"\nUsed {len(thread_info)} threads.")
    for ident, byte_count in thread_info.items():
        print(f"{'':4}Thread {ident} copied {byte_count} bytes.")
    print(f"Your transfer took {elapsed:.2f} seconds.")

def main():
    """
    Run the demonstration script for s3_file_transfer.
    """
    demo_manager = TransferDemoManager()
    demo_manager.collect_user_info()

    # Upload and download with default configuration. Because the file is 30 MB
    # and the default multipart_threshold is 8 MB, both upload and download are
    # multipart transfers.
    demo_manager.demo(
        "Do you want to upload and download a {} MB file "
        "using the default configuration?",
        file_transfer.upload_with_default_configuration,
        file_transfer.download_with_default_configuration,
    )

    # Upload and download with multipart_threshold set higher than the size of
    # the file. This causes the transfer manager to use standard transfers
    # instead of multipart transfers.
    demo_manager.demo(
        "Do you want to upload and download a {} MB file "
        "as a standard (not multipart) transfer?",
        file_transfer.upload_with_high_threshold,
        file_transfer.download_with_high_threshold,
    )

    # Upload with specific chunk size and additional metadata.
    # Download with a single thread.
    demo_manager.demo(
        "Do you want to upload a {} MB file with a smaller chunk size and "
        "then download the same file using a single thread?",
        file_transfer.upload_with_chunksize_and_meta,
        file_transfer.download_with_single_thread,
        upload_args={
            "metadata": {
```



```

        "upload_type": "chunky",
        "favorite_color": "aqua",
        "size": "medium",
    }
},
)

# Upload using server-side encryption with customer-provided
# encryption keys.
# Generate a 256-bit key from a passphrase.
sse_key = hashlib.sha256("demo_passphrase".encode("utf-8")).digest()
demo_manager.demo(
    "Do you want to upload and download a {} MB file using "
    "server-side encryption?",
    file_transfer.upload_with_sse,
    file_transfer.download_with_sse,
    upload_args={"sse_key": sse_key},
    download_args={"sse_key": sse_key},
)

# Download without specifying an encryption key to show that the
# encryption key must be included to download an encrypted object.
if demo_manager.ask_user(
    "Do you want to try to download the encrypted "
    "object without sending the required key?"
):
    try:
        _, object_key, download_file_path = demo_manager.last_name_set()
        file_transfer.download_with_default_configuration(
            demo_manager.demo_bucket,
            object_key,
            download_file_path,
            demo_manager.file_size_mb,
        )
    except ClientError as err:
        print(
            "Got expected error when trying to download an encrypted "
            "object without specifying encryption info:"
        )
        print(f"{'':4}{err}")

# Remove all created and downloaded files, remove all objects from
# S3 storage.
if demo_manager.ask_user(

```

```

        "Demonstration complete. Do you want to remove local files " "and S3
objects?"
    ):
        demo_manager.cleanup()

if __name__ == "__main__":
    try:
        main()
    except NoCredentialsError as error:
        print(error)
        print(
            "To run this example, you must have valid credentials in "
            "a shared credential file or set in environment variables."
        )

```

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

use std::fs::File;
use std::io::prelude::*;
use std::path::Path;

use aws_config::meta::region::RegionProviderChain;
use aws_sdk_s3::error::DisplayErrorContext;
use aws_sdk_s3::operation::{
    create_multipart_upload::CreateMultipartUploadOutput,
    get_object::GetObjectOutput,
};
use aws_sdk_s3::types::{CompletedMultipartUpload, CompletedPart};
use aws_sdk_s3::{config::Region, Client as S3Client};
use aws_smithy_types::byte_stream::{ByteStream, Length};

```

```
use rand::distributions::Alphanumeric;
use rand::{thread_rng, Rng};
use s3_code_examples::error::S3ExampleError;
use std::process;
use uuid::Uuid;

//In bytes, minimum chunk size of 5MB. Increase CHUNK_SIZE to send larger chunks.
const CHUNK_SIZE: u64 = 1024 * 1024 * 5;
const MAX_CHUNKS: u64 = 10000;

#[tokio::main]
pub async fn main() {
    if let Err(err) = run_example().await {
        eprintln!("Error: {}", DisplayErrorContext(err));
        process::exit(1);
    }
}

async fn run_example() -> Result<(), S3ExampleError> {
    let shared_config = aws_config::load_from_env().await;
    let client = S3Client::new(&shared_config);

    let bucket_name = format!("amzn-s3-demo-bucket-{}", Uuid::new_v4());
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));
    let region = region_provider.region().await.unwrap();
    s3_code_examples::create_bucket(&client, &bucket_name, &region).await?;

    let key = "sample.txt".to_string();
    // Create a multipart upload. Use UploadPart and CompleteMultipartUpload to
    // upload the file.
    let multipart_upload_res: CreateMultipartUploadOutput = client
        .create_multipart_upload()
        .bucket(&bucket_name)
        .key(&key)
        .send()
        .await?;

    let upload_id = multipart_upload_res.upload_id().ok_or(S3ExampleError::new(
        "Missing upload_id after CreateMultipartUpload",
    ))?;

    //Create a file of random characters for the upload.
    let mut file = File::create(&key).expect("Could not create sample file.");
```

```

// Loop until the file is 5 chunks.
while file.metadata().unwrap().len() <= CHUNK_SIZE * 4 {
    let rand_string: String = thread_rng()
        .sample_iter(&Alphanumeric)
        .take(256)
        .map(char::from)
        .collect();
    let return_string: String = "\n".to_string();
    file.write_all(rand_string.as_ref())
        .expect("Error writing to file.");
    file.write_all(return_string.as_ref())
        .expect("Error writing to file.");
}

let path = Path::new(&key);
let file_size = tokio::fs::metadata(path)
    .await
    .expect("it exists I swear")
    .len();

let mut chunk_count = (file_size / CHUNK_SIZE) + 1;
let mut size_of_last_chunk = file_size % CHUNK_SIZE;
if size_of_last_chunk == 0 {
    size_of_last_chunk = CHUNK_SIZE;
    chunk_count -= 1;
}

if file_size == 0 {
    return Err(S3ExampleError::new("Bad file size."));
}
if chunk_count > MAX_CHUNKS {
    return Err(S3ExampleError::new(
        "Too many chunks! Try increasing your chunk size.",
    ));
}

let mut upload_parts: Vec<aws_sdk_s3::types::CompletedPart> = Vec::new();

for chunk_index in 0..chunk_count {
    let this_chunk = if chunk_count - 1 == chunk_index {
        size_of_last_chunk
    } else {
        CHUNK_SIZE
    };
};

```

```

        let stream = ByteStream::read_from()
            .path(path)
            .offset(chunk_index * CHUNK_SIZE)
            .length(Length::Exact(this_chunk))
            .build()
            .await
            .unwrap();

// Chunk index needs to start at 0, but part numbers start at 1.
let part_number = (chunk_index as i32) + 1;
let upload_part_res = client
    .upload_part()
    .key(&key)
    .bucket(&bucket_name)
    .upload_id(upload_id)
    .body(stream)
    .part_number(part_number)
    .send()
    .await?;

upload_parts.push(
    CompletedPart::builder()
        .e_tag(upload_part_res.e_tag.unwrap_or_default())
        .part_number(part_number)
        .build(),
);
}

// upload_parts: Vec<aws_sdk_s3::types::CompletedPart>
let completed_multipart_upload: CompletedMultipartUpload =
CompletedMultipartUpload::builder()
    .set_parts(Some(upload_parts))
    .build();

let _complete_multipart_upload_res = client
    .complete_multipart_upload()
    .bucket(&bucket_name)
    .key(&key)
    .multipart_upload(completed_multipart_upload)
    .upload_id(upload_id)
    .send()
    .await?;

let data: GetObjectOutput =

```

```
s3_code_examples::download_object(&client, &bucket_name, &key).await?;  
let data_length: u64 = data  
    .content_length()  
    .unwrap_or_default()  
    .try_into()  
    .unwrap();  
if file.metadata().unwrap().len() == data_length {  
    println!("Data lengths match.");  
} else {  
    println!("The data was not the same size!");  
}  
  
s3_code_examples::clear_bucket(&client, &bucket_name)  
    .await  
    .expect("Error emptying bucket.");  
s3_code_examples::delete_bucket(&client, &bucket_name)  
    .await  
    .expect("Error deleting bucket.");  
  
Ok(())  
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 將大小不明的串流上傳至 Amazon S3 物件

下列程式碼範例顯示如何將大小不明的串流上傳至 Amazon S3 物件。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用 [AWS CRT 型 S3 用戶端](#)。

```
import com.example.s3.util.AsyncExampleUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.async.AsyncRequestBody;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.services.s3.S3AsyncClient;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;

import java.io.ByteArrayInputStream;
import java.io.InputStream;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

public class PutObjectFromStreamAsync {
    private static final Logger logger =
        LoggerFactory.getLogger(PutObjectFromStreamAsync.class);

    public static void main(String[] args) {
        String bucketName = "amzn-s3-demo-bucket"; // Replace with your bucket
        name.

        String key = UUID.randomUUID().toString();

        AsyncExampleUtils.createBucket(bucketName);
        try {
            PutObjectFromStreamAsync example = new PutObjectFromStreamAsync();
            S3AsyncClient s3AsyncClientCrt = S3AsyncClient.crtCreate();
            PutObjectResponse putObjectResponse =
example.putObjectFromStreamCrt(s3AsyncClientCrt, bucketName, key);
            logger.info("Object {} etag: {}", key, putObjectResponse.eTag());
            logger.info("Object {} uploaded to bucket {}. ", key, bucketName);
        } catch (SdkException e) {
            logger.error(e.getMessage(), e);
        } finally {
            AsyncExampleUtils.deleteObject(bucketName, key);
            AsyncExampleUtils.deleteBucket(bucketName);
        }
    }

    /**
```

```

    * @param s3CrtAsyncClient - To upload content from a stream of unknown
    size, use can the AWS CRT-based S3 client.
    * @param bucketName - The name of the bucket.
    * @param key - The name of the object.
    * @return software.amazon.awssdk.services.s3.model.PutObjectResponse -
    Returns metadata pertaining to the put object operation.
    */
    public PutObjectResponse putObjectFromStreamCrt(S3AsyncClient
    s3CrtAsyncClient, String bucketName, String key) {

        // AsyncExampleUtils.randomString() returns a random string up to 100
        characters.
        String randomString = AsyncExampleUtils.randomString();
        logger.info("random string to upload: {}: length={}", randomString,
        randomString.length());
        InputStream inputStream = new
        ByteArrayInputStream(randomString.getBytes());

        // Executor required to handle reading from the InputStream on a separate
        thread so the main upload is not blocked.
        ExecutorService executor = Executors.newSingleThreadExecutor();
        // Specify `null` for the content length when you don't know the content
        length.
        AsyncRequestBody body = AsyncRequestBody.fromInputStream(inputStream,
        null, executor);

        CompletableFuture<PutObjectResponse> responseFuture =
            s3CrtAsyncClient.putObject(r -> r.bucket(bucketName).key(key),
        body);

        PutObjectResponse response = responseFuture.join(); // Wait for the
        response.
        logger.info("Object {} uploaded to bucket {}. ", key, bucketName);
        executor.shutdown();
        return response;
    }
}

```

使用[啟用了分段上傳的標準非同步 S3 用戶端](#)。

```
import com.example.s3.util.AsyncExampleUtils;
```



```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.async.AsyncRequestBody;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.services.s3.S3AsyncClient;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;

import java.io.ByteArrayInputStream;
import java.io.InputStream;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

public class PutObjectFromStreamAsyncMp {
    private static final Logger logger =
        LoggerFactory.getLogger(PutObjectFromStreamAsyncMp.class);

    public static void main(String[] args) {
        String bucketName = "amzn-s3-demo-bucket"; // Replace with your bucket
        name.

        String key = UUID.randomUUID().toString();

        AsyncExampleUtils.createBucket(bucketName);
        try {
            PutObjectFromStreamAsyncMp example = new
            PutObjectFromStreamAsyncMp();
            S3AsyncClient s3AsyncClientMp =
            S3AsyncClient.builder().multipartEnabled(true).build();
            PutObjectResponse putObjectResponse =
            example.putObjectFromStreamMp(s3AsyncClientMp, bucketName, key);
            logger.info("Object {} etag: {}", key, putObjectResponse.eTag());
            logger.info("Object {} uploaded to bucket {}. ", key, bucketName);
        } catch (SdkException e) {
            logger.error(e.getMessage(), e);
        } finally {
            AsyncExampleUtils.deleteObject(bucketName, key);
            AsyncExampleUtils.deleteBucket(bucketName);
        }
    }

    /**
     * @param s3AsyncClientMp - To upload content from a stream of unknown size,
     use can the S3 asynchronous client with multipart enabled.
```

```

    * @param bucketName - The name of the bucket.
    * @param key - The name of the object.
    * @return software.amazon.awssdk.services.s3.model.PutObjectResponse -
Returns metadata pertaining to the put object operation.
    */
    public PutObjectResponse putObjectFromStreamMp(S3AsyncClient s3AsyncClientMp,
String bucketName, String key) {

        // AsyncExampleUtils.randomString() returns a random string up to 100
characters.
        String randomString = AsyncExampleUtils.randomString();
        logger.info("random string to upload: {}: length={}", randomString,
randomString.length());
        InputStream inputStream = new
ByteArrayInputStream(randomString.getBytes());

        // Executor required to handle reading from the InputStream on a separate
thread so the main upload is not blocked.
        ExecutorService executor = Executors.newSingleThreadExecutor();
        // Specify `null` for the content length when you don't know the content
length.
        AsyncRequestBody body = AsyncRequestBody.fromInputStream(inputStream,
null, executor);

        CompletableFuture<PutObjectResponse> responseFuture =
            s3AsyncClientMp.putObject(r -> r.bucket(bucketName).key(key),
body);

        PutObjectResponse response = responseFuture.join(); // Wait for the
response.
        logger.info("Object {} uploaded to bucket {}.", key, bucketName);
        executor.shutdown();
        return response;
    }
}

```

使用 [Amazon S3 Transfer Manager](#)。

```

import com.example.s3.util.AsyncExampleUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

```

```
import software.amazon.awssdk.core.async.AsyncRequestBody;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedUpload;
import software.amazon.awssdk.transfer.s3.model.Upload;

import java.io.ByteArrayInputStream;
import java.io.InputStream;
import java.util.UUID;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

public class UploadStream {
    private static final Logger logger =
        LoggerFactory.getLogger(UploadStream.class);

    public static void main(String[] args) {
        String bucketName = "amzn-s3-demo-bucket"; // Replace with your bucket
        name.

        String key = UUID.randomUUID().toString();

        AsyncExampleUtils.createBucket(bucketName);
        try {
            UploadStream example = new UploadStream();
            CompletedUpload completedUpload =
example.uploadStream(S3TransferManager.create(), bucketName, key);
            logger.info("Object {} etag: {}", key,
completedUpload.response().eTag());
            logger.info("Object {} uploaded to bucket {}. ", key, bucketName);
        } catch (SdkException e) {
            logger.error(e.getMessage(), e);
        } finally {
            AsyncExampleUtils.deleteObject(bucketName, key);
            AsyncExampleUtils.deleteBucket(bucketName);
        }
    }

    /**
     * @param transferManager - To upload content from a stream of unknown size,
you can use the S3TransferManager based on the AWS CRT-based S3 client.
     * @param bucketName - The name of the bucket.
     * @param key - The name of the object.
     * @return - software.amazon.awssdk.transfer.s3.model.CompletedUpload - The
result of the completed upload.
    */
}
```

```
    */
    public CompletedUpload uploadStream(S3TransferManager transferManager, String
bucketName, String key) {

        // AsyncExampleUtils.randomString() returns a random string up to 100
characters.
        String randomString = AsyncExampleUtils.randomString();
        logger.info("random string to upload: {}: length={}", randomString,
randomString.length());
        InputStream inputStream = new
ByteArrayInputStream(randomString.getBytes());

        // Executor required to handle reading from the InputStream on a separate
thread so the main upload is not blocked.
        ExecutorService executor = Executors.newSingleThreadExecutor();
        // Specify `null` for the content length when you don't know the content
length.
        AsyncRequestBody body = AsyncRequestBody.fromInputStream(inputStream,
null, executor);

        Upload upload = transferManager.upload(builder -> builder
            .requestBody(body)
            .putObjectRequest(req -> req.bucket(bucketName).key(key))
            .build());

        CompletedUpload completedUpload = upload.completionFuture().join();
        executor.shutdown();
        return completedUpload;
    }
}
```

Swift

適用於 Swift 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import ArgumentParser
import AWSClientRuntime
import AWSS3
import Foundation
import Smithy
import SmithyHTTPAPI
import SmithyStreams

/// Upload a file to the specified bucket.
///
/// - Parameters:
///   - bucket: The Amazon S3 bucket name to store the file into.
///   - key: The name (or path) of the file to upload to in the `bucket`.
///   - sourcePath: The pathname on the local filesystem of the file to
///     upload.
func uploadFile(sourcePath: String, bucket: String, key: String?) async
throws {
    let fileURL: URL = URL(fileURLWithPath: sourcePath)
    let fileName: String

    // If no key was provided, use the last component of the filename.

    if key == nil {
        fileName = fileURL.lastPathComponent
    } else {
        fileName = key!
    }

    let s3Client = try await S3Client()

    // Create a FileHandle for the source file.

    let fileHandle = FileHandle(forReadingAtPath: sourcePath)
    guard let fileHandle = fileHandle else {
        throw TransferError.readError
    }

    // Create a byte stream to retrieve the file's contents. This uses the
    // Smithy FileStream and ByteStream types.

    let stream = FileStream(fileHandle: fileHandle)
    let body = ByteStream.stream(stream)
```

```
// Create a `PutObjectInput` with the ByteStream as the body of the
// request's data. The AWS SDK for Swift will handle sending the
// entire file in chunks, regardless of its size.

let putInput = PutObjectInput(
    body: body,
    bucket: bucket,
    key: fileName
)

do {
    _ = try await s3Client.putObject(input: putInput)
} catch {
    throw TransferError.uploadError("Error uploading the file: \(error)")
}

print("File uploaded to \(fileURL.path).")
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用檢查總和來使用 AWS SDK 處理 Amazon S3 物件

下列程式碼範例示範如何使用總和檢查搭配 Amazon S3 物件。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

這些程式碼範例使用下列匯入的子集。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ChecksumAlgorithm;
import software.amazon.awssdk.services.s3.model.ChecksumMode;
import software.amazon.awssdk.services.s3.model.CompletedMultipartUpload;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.UploadPartRequest;
import software.amazon.awssdk.services.s3.model.UploadPartResponse;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.FileUpload;
import software.amazon.awssdk.transfer.s3.model.UploadFileRequest;

import java.io.FileInputStream;
import java.io.IOException;
import java.io.RandomAccessFile;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.ByteBuffer;
import java.nio.file.Paths;
import java.security.DigestInputStream;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.Base64;
import java.util.List;
import java.util.Objects;
import java.util.UUID;
```

當您[建置 PutObjectRequest](#)時，為 putObject 方法指定總和檢查演算法。

```
public void putObjectWithChecksum(String bucketName, String key) {
    s3Client.putObject(b -> b
        .bucket(bucketName)
        .key(key)
        .checksumAlgorithm(ChecksumAlgorithm.CRC32),
        RequestBody.fromString("This is a test"));
}
```

```
}
```

當您[建置 `GetObjectRequest`](#)時，請驗證 `getObject` 方法的總和檢查。

```
public GetObjectResponse getObjectWithChecksum(String bucketName, String key)
{
    return s3Client.getObject(b -> b
        .bucket(bucketName)
        .key(key)
        .checksumMode(ChecksumMode.ENABLED))
        .response();
}
```

當您[建置 `PutObjectRequest`](#)時，為 `putObject` 方法預先計算總和檢查。

```
public void putObjectWithPrecalculatedChecksum(String bucketName, String key,
String filePath) {
    String checksum = calculateChecksum(filePath, "SHA-256");

    s3Client.putObject((b -> b
        .bucket(bucketName)
        .key(key)
        .checksumSHA256(checksum)),
        RequestBody.fromFile(Paths.get(filePath)));
}
```

除了 [AWS CRT 型 S3 用戶端](#)之外，內容大小超過閾值時使用 [S3 Transfer Manager](#) 明確執行分段上傳。預設閾值大小為 8 MB。

您可以指定要讓 SDK 使用的總和檢查演算法。根據預設，SDK 會使用 CRC32 演算法。

```
public void multipartUploadWithChecksumTm(String bucketName, String key,
String filePath) {
    S3TransferManager transferManager = S3TransferManager.create();
    UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
        .putObjectRequest(b -> b
            .bucket(bucketName)
            .key(key)
```



```

        .checksumAlgorithm(ChecksumAlgorithm.SHA1))
        .source(Paths.get(filePath))
        .build();
FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);
fileUpload.completionFuture().join();
transferManager.close();
}

```

使用 [S3Client API](#) 或 ([S3AsyncClient API](#)) 執行分段上傳。如果您指定額外的總和檢查，則必須指定在上傳初始化時要使用的演算法。您還必須為每一個分段請求指定演算法，並在每一個分段上傳後提供為其計算的總和檢查。

```

public void multipartUploadWithChecksumS3Client(String bucketName, String
key, String filePath) {
    ChecksumAlgorithm algorithm = ChecksumAlgorithm.CRC32;

    // Initiate the multipart upload.
    CreateMultipartUploadResponse createMultipartUploadResponse =
s3Client.createMultipartUpload(b -> b
        .bucket(bucketName)
        .key(key)
        .checksumAlgorithm(algorithm)); // Checksum specified on initiation.
    String uploadId = createMultipartUploadResponse.uploadId();

    // Upload the parts of the file.
    int partNumber = 1;
    List<CompletedPart> completedParts = new ArrayList<>();
    ByteBuffer bb = ByteBuffer.allocate(1024 * 1024 * 5); // 5 MB byte buffer

    try (RandomAccessFile file = new RandomAccessFile(filePath, "r")) {
        long fileSize = file.length();
        long position = 0;
        while (position < fileSize) {
            file.seek(position);
            long read = file.getChannel().read(bb);

            bb.flip(); // Swap position and limit before reading from the
buffer.

            UploadPartRequest uploadPartRequest = UploadPartRequest.builder()
                .bucket(bucketName)
                .key(key)
                .uploadId(uploadId)

```

```
        .checksumAlgorithm(algorithm) // Checksum specified on each
part.
        .partNumber(partNumber)
        .build();

UploadPartResponse partResponse = s3Client.uploadPart(
    uploadPartRequest,
    RequestBody.fromByteBuffer(bb));

CompletedPart part = CompletedPart.builder()
    .partNumber(partNumber)
    .checksumCRC32(partResponse.checksumCRC32()) // Provide the
calculated checksum.
    .eTag(partResponse.eTag())
    .build();
completedParts.add(part);

    bb.clear();
    position += read;
    partNumber++;
}
} catch (IOException e) {
    System.err.println(e.getMessage());
}

// Complete the multipart upload.
s3Client.completeMultipartUpload(b -> b
    .bucket(bucketName)
    .key(key)
    .uploadId(uploadId)

.multipartUpload(CompletedMultipartUpload.builder().parts(completedParts).build()));
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
- [CompleteMultipartUpload](#)
- [CreateMultipartUpload](#)
- [UploadPart](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 使用 Amazon S3 物件完整性功能

下列程式碼範例示範如何使用 S3 物件完整性功能。

C++

SDK for C++

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

執行展示 Amazon S3 物件完整性功能的互動式情境。

```
//! Routine which runs the S3 object integrity workflow.
/*!
    \param clientConfig: Aws client configuration.
    \return bool: Function succeeded.
*/
bool AwsDoc::S3::s3ObjectIntegrityWorkflow(
    const Aws::S3::S3ClientConfiguration &clientConfiguration) {

    /*
     * Create a large file to be used for multipart uploads.
     */
    if (!createLargeFileIfNotExists()) {
        std::cerr << "Workflow exiting because large file creation failed." <<
std::endl;
        return false;
    }

    Aws::String bucketName = TEST_BUCKET_PREFIX;
    bucketName += Aws::Utils::UUID::RandomUUID();
    bucketName = Aws::Utils::StringUtils::ToLower(bucketName.c_str());

    bucketName.resize(std::min(bucketName.size(), MAX_BUCKET_NAME_LENGTH));

    introductoryExplanations(bucketName);
```

```

    if (!AwsDoc::S3::createBucket(bucketName, clientConfiguration)) {
        std::cerr << "Workflow exiting because bucket creation failed." <<
std::endl;
        return false;
    }

    Aws::S3::S3ClientConfiguration s3ClientConfiguration(clientConfiguration);
    std::shared_ptr<Aws::S3::S3Client> client =
    Aws::MakeShared<Aws::S3::S3Client>("S3Client", s3ClientConfiguration);

    printAsterisksLine();
    std::cout << "Choose from one of the following checksum algorithms."
        << std::endl;

    for (HASH_METHOD hashMethod = DEFAULT; hashMethod <= SHA256; ++hashMethod) {
        std::cout << "  " << hashMethod << " - " <<
stringForHashMethod(hashMethod)
        << std::endl;
    }

    HASH_METHOD chosenHashMethod = askQuestionForIntRange("Enter an index: ",
    DEFAULT,
                                                                    SHA256);

    gUseCalculatedChecksum = !askYesNoQuestion(
        "Let the SDK calculate the checksum for you? (y/n) ");

    printAsterisksLine();

    std::cout << "The workflow will now upload a file using PutObject."
        << std::endl;
    std::cout << "Object integrity will be verified using the "
        << stringForHashMethod(chosenHashMethod) << " algorithm."
        << std::endl;
    if (gUseCalculatedChecksum) {
        std::cout
            << "A checksum computed by this workflow will be used for object
integrity verification,"
            << std::endl;
        std::cout << "except for the TransferManager upload." << std::endl;
    } else {
        std::cout

```

```

        << "A checksum computed by the SDK will be used for object
integrity verification."
        << std::endl;
    }

    pressEnterToContinue();
    printAsterisksLine();

    std::shared_ptr<Aws::IOStream> inputData =
        Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
                                     TEST_FILE,
                                     std::ios_base::in |
                                     std::ios_base::binary);

    if (!*inputData) {
        std::cerr << "Error unable to read file " << TEST_FILE << std::endl;
        cleanUp(bucketName, clientConfiguration);
        return false;
    }

    Hasher hasher;
    HASH_METHOD putObjectHashMethod = chosenHashMethod;
    if (putObjectHashMethod == DEFAULT) {
        putObjectHashMethod = MD5; // MD5 is the default hash method for
PutObject.

        std::cout << "The default checksum algorithm for PutObject is "
                  << stringForHashMethod(putObjectHashMethod)
                  << std::endl;
    }

    // Demonstrate in code how the hash is computed.
    if (!hasher.calculateObjectHash(*inputData, putObjectHashMethod)) {
        std::cerr << "Error calculating hash for file " << TEST_FILE <<
std::endl;
        cleanUp(bucketName, clientConfiguration);
        return false;
    }
    Aws::String key = stringForHashMethod(putObjectHashMethod);
    key += "_";
    key += TEST_FILE_KEY;
    Aws::String localHash = hasher.getBase64HashString();

    // Upload the object with PutObject

```

```

    if (!putObjectWithHash(bucketName, key, localHash, putObjectHashMethod,
                          inputData, chosenHashMethod == DEFAULT,
                          *client)) {
        std::cerr << "Error putting file " << TEST_FILE << " to bucket "
                  << bucketName << " with key " << key << std::endl;
        cleanUp(bucketName, clientConfiguration);
        return false;
    }

    Aws::String retrievedHash;
    if (!retrieveObjectHash(bucketName, key,
                          putObjectHashMethod, retrievedHash,
                          nullptr, *client)) {
        std::cerr << "Error getting file " << TEST_FILE << " from bucket "
                  << bucketName << " with key " << key << std::endl;
        cleanUp(bucketName, clientConfiguration);
        return false;
    }

    explainPutObjectResults();
    verifyHashingResults(retrievedHash, hasher,
                        "PutObject upload", putObjectHashMethod);

    printAsterisksLine();
    pressEnterToContinue();

    key = "tr_";
    key += stringForHashMethod(chosenHashMethod) + "_" + MULTI_PART_TEST_FILE;

    introductoryTransferManagerUploadExplanations(key);

    HASH_METHOD transferManagerHashMethod = chosenHashMethod;
    if (transferManagerHashMethod == DEFAULT) {
        transferManagerHashMethod = CRC32; // The default hash method for the
        TransferManager is CRC32.

        std::cout << "The default checksum algorithm for TransferManager is "
                  << stringForHashMethod(transferManagerHashMethod)
                  << std::endl;
    }

    // Upload the large file using the transfer manager.

```

```

    if (!doTransferManagerUpload(bucketName, key, transferManagerHashMethod,
        chosenHashMethod == DEFAULT,
                                client)) {
        std::cerr << "Exiting because of an error in doTransferManagerUpload." <<
std::endl;
        cleanUp(bucketName, clientConfiguration);
        return false;
    }

    std::vector<Aws::String> retrievedTransferManagerPartHashes;
    Aws::String retrievedTransferManagerFinalHash;

    // Retrieve all the hashes for the TransferManager upload.
    if (!retrieveObjectHash(bucketName, key,
                            transferManagerHashMethod,
                            retrievedTransferManagerFinalHash,
                            &retrievedTransferManagerPartHashes, *client)) {
        std::cerr << "Exiting because of an error in retrieveObjectHash for
TransferManager." << std::endl;
        cleanUp(bucketName, clientConfiguration);
        return false;
    }

    AwsDoc::S3::Hasher locallyCalculatedFinalHash;
    std::vector<Aws::String> locallyCalculatedPartHashes;

    // Calculate the hashes locally to demonstrate how TransferManager hashes are
    computed.
    if (!calculatePartHashesForFile(transferManagerHashMethod,
        MULTI_PART_TEST_FILE,
                                UPLOAD_BUFFER_SIZE,
                                locallyCalculatedFinalHash,
                                locallyCalculatedPartHashes)) {
        std::cerr << "Exiting because of an error in calculatePartHashesForFile."
<< std::endl;
        cleanUp(bucketName, clientConfiguration);
        return false;
    }

    verifyHashingResults(retrievedTransferManagerFinalHash,
                        locallyCalculatedFinalHash, "TransferManager upload",
                        transferManagerHashMethod,
                        retrievedTransferManagerPartHashes,
                        locallyCalculatedPartHashes);

```

```

printAsterisksLine();

key = "mp_";
key += stringForHashMethod(chosenHashMethod) + "_" + MULTI_PART_TEST_FILE;

multiPartUploadExplanations(key, chosenHashMethod);

pressEnterToContinue();

std::shared_ptr<Aws::IOStream> largeFileInputData =
    Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
                                  MULTI_PART_TEST_FILE,
                                  std::ios_base::in |
                                  std::ios_base::binary);

if (!largeFileInputData->good()) {
    std::cerr << "Error unable to read file " << TEST_FILE << std::endl;
    cleanUp(bucketName, clientConfiguration);
    return false;
}

HASH_METHOD multipartUploadHashMethod = chosenHashMethod;
if (multipartUploadHashMethod == DEFAULT) {
    multipartUploadHashMethod = MD5; // The default hash method for
    multipart uploads is MD5.

    std::cout << "The default checksum algorithm for multipart upload is "
                << stringForHashMethod(putObjectHashMethod)
                << std::endl;
}

AwsDoc::S3::Hasher hashData;
std::vector<Aws::String> partHashes;

if (!doMultipartUpload(bucketName, key,
                        multipartUploadHashMethod,
                        largeFileInputData, chosenHashMethod == DEFAULT,
                        hashData,
                        partHashes,
                        *client)) {
    std::cerr << "Exiting because of an error in doMultipartUpload." <<
    std::endl;
    cleanUp(bucketName, clientConfiguration);
}

```



```

        return false;
    }

    std::cout << "Finished multipart upload of with hash method " <<
        stringForHashMethod(multipartUploadHashMethod) << std::endl;

    std::cout << "Now we will retrieve the checksums from the server." <<
std::endl;

    retrievedHash.clear();
    std::vector<Aws::String> retrievedPartHashes;
    if (!retrieveObjectHash(bucketName, key,
                            multipartUploadHashMethod,
                            retrievedHash, &retrievedPartHashes, *client)) {
        std::cerr << "Exiting because of an error in retrieveObjectHash for
multipart." << std::endl;
        cleanUp(bucketName, clientConfiguration);
        return false;
    }

    verifyHashingResults(retrievedHash, hashData, "MultiPart upload",
                        multipartUploadHashMethod,
                        retrievedPartHashes, partHashes);

    printAsterisksLine();

    if (askYesNoQuestion("Would you like to delete the resources created in this
workflow? (y/n)")) {
        return cleanUp(bucketName, clientConfiguration);
    } else {
        std::cout << "The bucket " << bucketName << " was not deleted." <<
std::endl;
        return true;
    }
}

//! Routine which uploads an object to an S3 bucket with different object
integrity hashing methods.
/*!
    \param bucket: The name of the S3 bucket where the object will be uploaded.
    \param key: The unique identifier (key) for the object within the S3 bucket.
    \param hashData: The hash value that will be associated with the uploaded
object.

```

```

    \param hashMethod: The hashing algorithm to use when calculating the hash
    value.
    \param body: The data content of the object being uploaded.
    \param useDefaultHashMethod: A flag indicating whether to use the default hash
    method or the one specified in the hashMethod parameter.
    \param client: The S3 client instance used to perform the upload operation.
    \return bool: Function succeeded.
*/
bool AwsDoc::S3::putObjectWithHash(const Aws::String &bucket, const Aws::String
    &key,
                                   const Aws::String &hashData,
                                   AwsDoc::S3::HASH_METHOD hashMethod,
                                   const std::shared_ptr<Aws::IOStream> &body,
                                   bool useDefaultHashMethod,
                                   const Aws::S3::S3Client &client) {
    Aws::S3::Model::PutObjectRequest request;
    request.SetBucket(bucket);
    request.SetKey(key);
    if (!useDefaultHashMethod) {
        if (hashMethod != MD5) {
            request.SetChecksumAlgorithm(getChecksumAlgorithmForHashMethod(hashMethod));
        }
    }

    if (gUseCalculatedChecksum) {
        switch (hashMethod) {
            case AwsDoc::S3::MD5:
                request.SetContentMD5(hashData);
                break;
            case AwsDoc::S3::SHA1:
                request.SetChecksumSHA1(hashData);
                break;
            case AwsDoc::S3::SHA256:
                request.SetChecksumSHA256(hashData);
                break;
            case AwsDoc::S3::CRC32:
                request.SetChecksumCRC32(hashData);
                break;
            case AwsDoc::S3::CRC32C:
                request.SetChecksumCRC32C(hashData);
                break;
            default:
                std::cerr << "Unknown hash method." << std::endl;
        }
    }
}

```

```

        return false;
    }
}
request.SetBody(body);
Aws::S3::Model::PutObjectOutcome outcome = client.PutObject(request);
body->seekg(0, body->beg);
if (outcome.IsSuccess()) {
    std::cout << "Object successfully uploaded." << std::endl;
} else {
    std::cerr << "Error uploading object." <<
        outcome.GetError().GetMessage() << std::endl;
}
return outcome.IsSuccess();
}

// ! Routine which retrieves the hash value of an object stored in an S3 bucket.
/*!
    \param bucket: The name of the S3 bucket where the object is stored.
    \param key: The unique identifier (key) of the object within the S3 bucket.
    \param hashMethod: The hashing algorithm used to calculate the hash value of
the object.
    \param[out] hashData: The retrieved hash.
    \param[out] partHashes: The part hashes if available.
    \param client: The S3 client instance used to retrieve the object.
    \return bool: Function succeeded.
*/
bool AwsDoc::S3::retrieveObjectHash(const Aws::String &bucket, const Aws::String
&key,

                                AwsDoc::S3::HASH_METHOD hashMethod,
                                Aws::String &hashData,
                                std::vector<Aws::String> *partHashes,
                                const Aws::S3::S3Client &client) {
    Aws::S3::Model::GetObjectAttributesRequest request;
    request.SetBucket(bucket);
    request.SetKey(key);

    if (hashMethod == MD5) {
        Aws::Vector<Aws::S3::Model::ObjectAttributes> attributes;
        attributes.push_back(Aws::S3::Model::ObjectAttributes::ETag);
        request.SetObjectAttributes(attributes);

        Aws::S3::Model::GetObjectAttributesOutcome outcome =
client.GetObjectAttributes(

```

```

        request);
    if (outcome.IsSuccess()) {
        const Aws::S3::Model::GetObjectAttributesResult &result =
outcome.GetResult();
        hashData = result.GetETag();
    } else {
        std::cerr << "Error retrieving object etag attributes." <<
            outcome.GetError().GetMessage() << std::endl;
        return false;
    }
} else { // hashMethod != MD5
    Aws::Vector<Aws::S3::Model::ObjectAttributes> attributes;
    attributes.push_back(Aws::S3::Model::ObjectAttributes::Checksum);
    request.SetObjectAttributes(attributes);

    Aws::S3::Model::GetObjectAttributesOutcome outcome =
client.GetObjectAttributes(
    request);
    if (outcome.IsSuccess()) {
        const Aws::S3::Model::GetObjectAttributesResult &result =
outcome.GetResult();
        switch (hashMethod) {
            case AwsDoc::S3::DEFAULT: // NOLINT(*-branch-clone)
                break; // Default is not supported.
#pragma clang diagnostic push
#pragma ide diagnostic ignored "UnreachableCode"
            case AwsDoc::S3::MD5:
                break; // MD5 is not supported.
#pragma clang diagnostic pop
            case AwsDoc::S3::SHA1:
                hashData = result.GetChecksum().GetChecksumSHA1();
                break;
            case AwsDoc::S3::SHA256:
                hashData = result.GetChecksum().GetChecksumSHA256();
                break;
            case AwsDoc::S3::CRC32:
                hashData = result.GetChecksum().GetChecksumCRC32();
                break;
            case AwsDoc::S3::CRC32C:
                hashData = result.GetChecksum().GetChecksumCRC32C();
                break;
            default:
                std::cerr << "Unknown hash method." << std::endl;
                return false;
        }
    }
}

```

```

    }
} else {
    std::cerr << "Error retrieving object checksum attributes." <<
        outcome.GetError().GetMessage() << std::endl;
    return false;
}

if (nullptr != partHashes) {
    attributes.clear();
    attributes.push_back(Aws::S3::Model::ObjectAttributes::ObjectParts);
    request.SetObjectAttributes(attributes);
    outcome = client.GetObjectAttributes(request);
    if (outcome.IsSuccess()) {
        const Aws::S3::Model::GetObjectAttributesResult &result =
outcome.GetResult();
        const Aws::Vector<Aws::S3::Model::ObjectPart> parts =
result.GetObjectParts().GetParts();
        for (const Aws::S3::Model::ObjectPart &part: parts) {
            switch (hashMethod) {
                case AwsDoc::S3::DEFAULT: // Default is not supported.
NOLINT(*-branch-clone)
                    break;
                case AwsDoc::S3::MD5: // MD5 is not supported.
                    break;
                case AwsDoc::S3::SHA1:
                    partHashes->push_back(part.GetChecksumSHA1());
                    break;
                case AwsDoc::S3::SHA256:
                    partHashes->push_back(part.GetChecksumSHA256());
                    break;
                case AwsDoc::S3::CRC32:
                    partHashes->push_back(part.GetChecksumCRC32());
                    break;
                case AwsDoc::S3::CRC32C:
                    partHashes->push_back(part.GetChecksumCRC32C());
                    break;
                default:
                    std::cerr << "Unknown hash method." << std::endl;
                    return false;
            }
        }
    } else {
        std::cerr << "Error retrieving object attributes for object
parts." <<

```

```

        outcome.GetError().GetMessage() << std::endl;
        return false;
    }
}

return true;
}

//! Verifies the hashing results between the retrieved and local hashes.
/*!
\param retrievedHash The hash value retrieved from the remote source.
\param localHash The hash value calculated locally.
\param uploadtype The type of upload (e.g., "multipart", "single-part").
\param hashMethod The hashing method used (e.g., MD5, SHA-256).
\param retrievedPartHashes (Optional) The list of hashes for the individual
parts retrieved from the remote source.
\param localPartHashes (Optional) The list of hashes for the individual parts
calculated locally.
*/
void AwsDoc::S3::verifyHashingResults(const Aws::String &retrievedHash,
                                      const Hasher &localHash,
                                      const Aws::String &uploadtype,
                                      HASH_METHOD hashMethod,
                                      const std::vector<Aws::String>
&retrievedPartHashes,
                                      const std::vector<Aws::String>
&localPartHashes) {
    std::cout << "For " << uploadtype << " retrieved hash is " << retrievedHash
<< std::endl;
    if (!retrievedPartHashes.empty()) {
        std::cout << retrievedPartHashes.size() << " part hash(es) were also
retrieved."
<< std::endl;
        for (auto &retrievedPartHash: retrievedPartHashes) {
            std::cout << " Part hash " << retrievedPartHash << std::endl;
        }
    }
    Aws::String hashString;
    if (hashMethod == MD5) {
        hashString = localHash.getHexHashString();
        if (!localPartHashes.empty()) {
            hashString += "-" + std::to_string(localPartHashes.size());
        }
    }
}

```

```

    } else {
        hashString = localHash.getBase64HashString();
    }

    bool allMatch = true;
    if (hashString != retrievedHash) {
        std::cerr << "For " << uploadtype << ", the main hashes do not match" <<
std::endl;
        std::cerr << "Local hash- '" << hashString << "'" << std::endl;
        std::cerr << "Remote hash - '" << retrievedHash << "'" << std::endl;
        allMatch = false;
    }

    if (hashMethod != MD5) {
        if (localPartHashes.size() != retrievedPartHashes.size()) {
            std::cerr << "For " << uploadtype << ", the number of part hashes do
not match" << std::endl;
            std::cerr << "Local number of hashes- '" << localPartHashes.size() <<
""
                << std::endl;
            std::cerr << "Remote number of hashes - '"
                << retrievedPartHashes.size()
                << "'" << std::endl;
        }

        for (int i = 0; i < localPartHashes.size(); ++i) {
            if (localPartHashes[i] != retrievedPartHashes[i]) {
                std::cerr << "For " << uploadtype << ", the part hashes do not
match for part " << i + 1
                    << "." << std::endl;
                std::cerr << "Local hash- '" << localPartHashes[i] << "'"
                    << std::endl;
                std::cerr << "Remote hash - '" << retrievedPartHashes[i] << "'"
                    << std::endl;
                allMatch = false;
            }
        }
    }

    if (allMatch) {
        std::cout << "For " << uploadtype << ", locally and remotely calculated
hashes all match!" << std::endl;
    }

```

```

}

static void transferManagerErrorCallback(const Aws::Transfer::TransferManager *,
                                         const std::shared_ptr<const
                                         Aws::Transfer::TransferHandle> &,
                                         const
                                         Aws::Client::AWSError<Aws::S3::S3Errors> &err) {
    std::cerr << "Error during transfer: '" << err.GetMessage() << "'" <<
    std::endl;
}

static void transferManagerStatusCallback(const Aws::Transfer::TransferManager *,
                                         const std::shared_ptr<const
                                         Aws::Transfer::TransferHandle> &handle) {
    if (handle->GetStatus() == Aws::Transfer::TransferStatus::IN_PROGRESS) {
        std::cout << "Bytes transferred: " << handle->GetBytesTransferred() <<
        std::endl;
    }
}

}

//! Routine which uploads an object to an S3 bucket using the AWS C++ SDK's
    Transfer Manager.
/*!
    \param bucket: The name of the S3 bucket where the object will be uploaded.
    \param key: The unique identifier (key) for the object within the S3 bucket.
    \param hashMethod: The hashing algorithm to use when calculating the hash
    value.
    \param useDefaultHashMethod: A flag indicating whether to use the default hash
    method or the one specified in the hashMethod parameter.
    \param client: The S3 client instance used to perform the upload operation.
    \return bool: Function succeeded.
*/
bool
AwsDoc::S3::doTransferManagerUpload(const Aws::String &bucket, const Aws::String
&key,
                                     AwsDoc::S3::HASH_METHOD hashMethod,
                                     bool useDefaultHashMethod,
                                     const std::shared_ptr<Aws::S3::S3Client>
&client) {
    std::shared_ptr<Aws::Utils::Threading::PooledThreadExecutor> executor =
    Aws::MakeShared<Aws::Utils::Threading::PooledThreadExecutor>(
        "executor", 25);
    Aws::Transfer::TransferManagerConfiguration transfer_config(executor.get());
    transfer_config.s3Client = client;

```



```

    transfer_config.bufferSize = UPLOAD_BUFFER_SIZE;
    if (!useDefaultHashMethod) {
        if (hashMethod == MD5) {
            transfer_config.computeContentMD5 = true;
        } else {
            transfer_config.checksumAlgorithm =
getChecksumAlgorithmForHashMethod(
                hashMethod);
        }
    }
    transfer_config.errorCallback = transferManagerErrorCallback;
    transfer_config.transferStatusUpdatedCallback =
transferManagerStatusCallback;

    std::shared_ptr<Aws::Transfer::TransferManager> transfer_manager =
Aws::Transfer::TransferManager::Create(
        transfer_config);

    std::cout << "Uploading the file..." << std::endl;
    std::shared_ptr<Aws::Transfer::TransferHandle> uploadHandle =
transfer_manager->UploadFile(MULTI_PART_TEST_FILE,

        bucket, key,

        "text/plain",

        Aws::Map<Aws::String, Aws::String>());
    uploadHandle->WaitUntilFinished();
    bool success =
        uploadHandle->GetStatus() ==
Aws::Transfer::TransferStatus::COMPLETED;
    if (!success) {
        Aws::Client::AWSError<Aws::S3::S3Errors> err = uploadHandle-
>GetLastError();
        std::cerr << "File upload failed:  " << err.GetMessage() << std::endl;
    }

    return success;
}

//! Routine which calculates the hash values for each part of a file being
    uploaded to an S3 bucket.
/*!

```

```

    \param hashMethod: The hashing algorithm to use when calculating the hash
    values.
    \param fileName: The path to the file for which the part hashes will be
    calculated.
    \param bufferSize: The size of the buffer to use when reading the file.
    \param[out] hashDataResult: The Hasher object that will store the concatenated
    hash value.
    \param[out] partHashes: The vector that will store the calculated hash values
    for each part of the file.
    \return bool: Function succeeded.
*/
bool AwsDoc::S3::calculatePartHashesForFile(AwsDoc::S3::HASH_METHOD hashMethod,
                                             const Aws::String &fileName,
                                             size_t bufferSize,
                                             AwsDoc::S3::Hasher &hashDataResult,
                                             std::vector<Aws::String> &partHashes)
{
    std::ifstream fileStream(fileName.c_str(), std::ifstream::binary);
    fileStream.seekg(0, std::ifstream::end);
    size_t objectSize = fileStream.tellg();
    fileStream.seekg(0, std::ifstream::beg);
    std::vector<unsigned char> totalHashBuffer;
    size_t uploadedBytes = 0;

    while (uploadedBytes < objectSize) {
        std::vector<unsigned char> buffer(bufferSize);
        std::streamsize bytesToRead =
static_cast<std::streamsize>(std::min(buffer.size(), objectSize -
uploadedBytes));
        fileStream.read((char *) buffer.data(), bytesToRead);
        Aws::Utils::Stream::PreallocatedStreamBuf
preallocatedStreamBuf(buffer.data(),

bytesToRead);
        std::shared_ptr<Aws::IOStream> body =
            Aws::MakeShared<Aws::IOStream>("SampleAllocationTag",
                                           &preallocatedStreamBuf);

        Hasher hasher;
        if (!hasher.calculateObjectHash(*body, hashMethod)) {
            std::cerr << "Error calculating hash." << std::endl;
            return false;
        }
        Aws::String base64HashString = hasher.getBase64HashString();
    }
}

```

```

        partHashes.push_back(base64HashString);

        Aws::Utils::ByteBuffer hashBuffer = hasher.getBytesBufferHash();

        totalHashBuffer.insert(totalHashBuffer.end(),
            hashBuffer.GetUnderlyingData(),
            hashBuffer.GetUnderlyingData() +
            hashBuffer.GetLength());

        uploadedBytes += bytesToRead;
    }

    return hashDataResult.calculateObjectHash(totalHashBuffer, hashMethod);
}

//! Create a multipart upload.
/*!
    \param bucket: The name of the S3 bucket where the object will be uploaded.
    \param key: The unique identifier (key) for the object within the S3 bucket.
    \param client: The S3 client instance used to perform the upload operation.
    \return Aws::String: Upload ID or empty string if failed.
*/
Aws::String
AwsDoc::S3::createMultipartUpload(const Aws::String &bucket, const Aws::String
    &key,
                                Aws::S3::Model::ChecksumAlgorithm
checksumAlgorithm,
                                const Aws::S3::S3Client &client) {
    Aws::S3::Model::CreateMultipartUploadRequest request;
    request.SetBucket(bucket);
    request.SetKey(key);

    if (checksumAlgorithm != Aws::S3::Model::ChecksumAlgorithm::NOT_SET) {
        request.SetChecksumAlgorithm(checksumAlgorithm);
    }

    Aws::S3::Model::CreateMultipartUploadOutcome outcome =
        client.CreateMultipartUpload(request);

    Aws::String uploadID;
    if (outcome.IsSuccess()) {
        uploadID = outcome.GetResult().GetUploadId();
    } else {

```

```

        std::cerr << "Error creating multipart upload: " <<
outcome.GetError().GetMessage() << std::endl;
    }

    return uploadID;
}

//! Upload a part to an S3 bucket.
/*!
    \param bucket: The name of the S3 bucket where the object will be uploaded.
    \param key: The unique identifier (key) for the object within the S3 bucket.
    \param uploadID: An upload ID string.
    \param partNumber:
    \param checksumAlgorithm: Checksum algorithm, ignored when NOT_SET.
    \param calculatedHash: A data integrity hash to set, depending on the
checksum algorithm,
                                ignored when it is an empty string.
    \param body: An shared_ptr IOSTream of the data to be uploaded.
    \param client: The S3 client instance used to perform the upload operation.
    \return UploadPartOutcome: The outcome.
*/

Aws::S3::Model::UploadPartOutcome AwsDoc::S3::uploadPart(const Aws::String
&bucket,
                                                    const Aws::String &key,
                                                    const Aws::String
&uploadID,
                                                    int partNumber,

Aws::S3::Model::ChecksumAlgorithm checksumAlgorithm,
                                                    const Aws::String
&calculatedHash,
                                                    const
std::shared_ptr<Aws::IOStream> &body,
                                                    const Aws::S3::S3Client
&client) {
    Aws::S3::Model::UploadPartRequest request;
    request.SetBucket(bucket);
    request.SetKey(key);
    request.SetUploadId(uploadID);
    request.SetPartNumber(partNumber);
    if (checksumAlgorithm != Aws::S3::Model::ChecksumAlgorithm::NOT_SET) {
        request.SetChecksumAlgorithm(checksumAlgorithm);
    }
}

```

```

    request.SetBody(body);

    if (!calculatedHash.empty()) {
        switch (checksumAlgorithm) {
            case Aws::S3::Model::ChecksumAlgorithm::NOT_SET:
                request.SetContentMD5(calculatedHash);
                break;
            case Aws::S3::Model::ChecksumAlgorithm::CRC32:
                request.SetChecksumCRC32(calculatedHash);
                break;
            case Aws::S3::Model::ChecksumAlgorithm::CRC32C:
                request.SetChecksumCRC32C(calculatedHash);
                break;
            case Aws::S3::Model::ChecksumAlgorithm::SHA1:
                request.SetChecksumSHA1(calculatedHash);
                break;
            case Aws::S3::Model::ChecksumAlgorithm::SHA256:
                request.SetChecksumSHA256(calculatedHash);
                break;
        }
    }

    return client.UploadPart(request);
}

///  

///! Abort a multipart upload to an S3 bucket.  

///!  

/// \param bucket: The name of the S3 bucket where the object will be uploaded.  

/// \param key: The unique identifier (key) for the object within the S3 bucket.  

/// \param uploadID: An upload ID string.  

/// \param client: The S3 client instance used to perform the upload operation.  

/// \return bool: Function succeeded.  

*//  

bool AwsDoc::S3::abortMultipartUpload(const Aws::String &bucket,  
                                       const Aws::String &key,  
                                       const Aws::String &uploadID,  
                                       const Aws::S3::S3Client &client) {  
    Aws::S3::Model::AbortMultipartUploadRequest request;  
    request.SetBucket(bucket);  
    request.SetKey(key);  
    request.SetUploadId(uploadID);  
  
    Aws::S3::Model::AbortMultipartUploadOutcome outcome =

```

```

        client.AbortMultipartUpload(request);

        if (outcome.IsSuccess()) {
            std::cout << "Multipart upload aborted." << std::endl;
        } else {
            std::cerr << "Error aborting multipart upload: " <<
outcome.GetError().GetMessage() << std::endl;
        }

        return outcome.IsSuccess();
    }

    //! Complete a multipart upload to an S3 bucket.
    /*!
        \param bucket: The name of the S3 bucket where the object will be uploaded.
        \param key: The unique identifier (key) for the object within the S3 bucket.
        \param uploadID: An upload ID string.
        \param parts: A vector of CompleteParts.
        \param client: The S3 client instance used to perform the upload operation.
        \return CompleteMultipartUploadOutcome: The request outcome.
    */
    Aws::S3::Model::CompleteMultipartUploadOutcome
    AwsDoc::S3::completeMultipartUpload(const Aws::String &bucket,

        const Aws::String &key,

        const Aws::String &uploadID,

        const Aws::Vector<Aws::S3::Model::CompletedPart> &parts,

        const Aws::S3::S3Client &client) {
        Aws::S3::Model::CompletedMultipartUpload completedMultipartUpload;
        completedMultipartUpload.SetParts(parts);

        Aws::S3::Model::CompleteMultipartUploadRequest request;
        request.SetBucket(bucket);
        request.SetKey(key);
        request.SetUploadId(uploadID);
        request.SetMultipartUpload(completedMultipartUpload);

        Aws::S3::Model::CompleteMultipartUploadOutcome outcome =
            client.CompleteMultipartUpload(request);

        if (!outcome.IsSuccess()) {

```

```

        std::cerr << "Error completing multipart upload: " <<
outcome.GetError().GetMessage() << std::endl;
    }
    return outcome;
}

//! Routine which performs a multi-part upload.
/*!
    \param bucket: The name of the S3 bucket where the object will be uploaded.
    \param key: The unique identifier (key) for the object within the S3 bucket.
    \param hashMethod: The hashing algorithm to use when calculating the hash
value.
    \param ioStream: An IOStream for the data to be uploaded.
    \param useDefaultHashMethod: A flag indicating whether to use the default
hash method or the one specified in the hashMethod parameter.
    \param[out] hashDataResult: The Hasher object that will store the
concatenated hash value.
    \param[out] partHashes: The vector that will store the calculated hash values
for each part of the file.
    \param client: The S3 client instance used to perform the upload operation.
    \return bool: Function succeeded.
*/
bool AwsDoc::S3::doMultipartUpload(const Aws::String &bucket,
                                   const Aws::String &key,
                                   AwsDoc::S3::HASH_METHOD hashMethod,
                                   const std::shared_ptr<Aws::IOStream>
&ioStream,
                                   bool useDefaultHashMethod,
                                   AwsDoc::S3::Hasher &hashDataResult,
                                   std::vector<Aws::String> &partHashes,
                                   const Aws::S3::S3Client &client) {

    // Get object size.
    ioStream->seekg(0, ioStream->end);
    size_t objectSize = ioStream->tellg();
    ioStream->seekg(0, ioStream->beg);

    Aws::S3::Model::ChecksumAlgorithm checksumAlgorithm =
Aws::S3::Model::ChecksumAlgorithm::NOT_SET;
    if (!useDefaultHashMethod) {
        if (hashMethod != MD5) {
            checksumAlgorithm = getChecksumAlgorithmForHashMethod(hashMethod);
        }
    }
}

```

```

    Aws::String uploadID = createMultipartUpload(bucket, key, checksumAlgorithm,
client);
    if (uploadID.empty()) {
        return false;
    }

    std::vector<unsigned char> totalHashBuffer;
    bool uploadSucceeded = true;
    std::streamsize uploadedBytes = 0;
    int partNumber = 1;
    Aws::Vector<Aws::S3::Model::CompletedPart> parts;
    while (uploadedBytes < objectSize) {
        std::cout << "Uploading part " << partNumber << "." << std::endl;

        std::vector<unsigned char> buffer(UPLOAD_BUFFER_SIZE);
        std::streamsize bytesToRead =
static_cast<std::streamsize>(std::min(buffer.size(),
objectSize - uploadedBytes));
        ioStream->read((char *) buffer.data(), bytesToRead);
        Aws::Utils::Stream::PreallocatedStreamBuf
preallocatedStreamBuf(buffer.data(),
bytesToRead);
        std::shared_ptr<Aws::IOStream> body =
            Aws::MakeShared<Aws::IOStream>("SampleAllocationTag",
                &preallocatedStreamBuf);

        Hasher hasher;
        if (!hasher.calculateObjectHash(*body, hashMethod)) {
            std::cerr << "Error calculating hash." << std::endl;
            uploadSucceeded = false;
            break;
        }

        Aws::String base64HashString = hasher.getBase64HashString();
        partHashes.push_back(base64HashString);

        Aws::Utils::ByteBuffer hashBuffer = hasher.getByteBufferHash();

        totalHashBuffer.insert(totalHashBuffer.end(),
hashBuffer.GetUnderlyingData(),
                                hashBuffer.GetUnderlyingData() +
hashBuffer.GetLength());

```



```

        Aws::String calculatedHash;
        if (gUseCalculatedChecksum) {
            calculatedHash = base64HashString;
        }
        Aws::S3::Model::UploadPartOutcome uploadPartOutcome = uploadPart(bucket,
key, uploadID, partNumber,

checksumAlgorithm, base64HashString, body,

                                                                    client);

        if (uploadPartOutcome.IsSuccess()) {
            const Aws::S3::Model::UploadPartResult &uploadPartResult =
uploadPartOutcome.GetResult();
            Aws::S3::Model::CompletedPart completedPart;
            completedPart.SetETag(uploadPartResult.GetETag());
            completedPart.SetPartNumber(partNumber);
            switch (hashMethod) {
                case AwsDoc::S3::MD5:
                    break; // Do nothing.
                case AwsDoc::S3::SHA1:

completedPart.SetChecksumSHA1(uploadPartResult.GetChecksumSHA1());
                    break;
                case AwsDoc::S3::SHA256:

completedPart.SetChecksumSHA256(uploadPartResult.GetChecksumSHA256());
                    break;
                case AwsDoc::S3::CRC32:

completedPart.SetChecksumCRC32(uploadPartResult.GetChecksumCRC32());
                    break;
                case AwsDoc::S3::CRC32C:

completedPart.SetChecksumCRC32C(uploadPartResult.GetChecksumCRC32C());
                    break;
                default:
                    std::cerr << "Unhandled hash method for completedPart." <<
std::endl;
                    break;
            }

            parts.push_back(completedPart);
        } else {
            std::cerr << "Error uploading part. " <<

```

```

        uploadPartOutcome.GetError().GetMessage() << std::endl;
        uploadSucceeded = false;
        break;
    }

    uploadedBytes += bytesToRead;
    partNumber++;
}

if (!uploadSucceeded) {
    abortMultipartUpload(bucket, key, uploadID, client);
    return false;
} else {

    Aws::S3::Model::CompleteMultipartUploadOutcome
completeMultipartUploadOutcome = completeMultipartUpload(bucket,

                                key,

                                uploadID,

                                parts,

                                client);

    if (completeMultipartUploadOutcome.IsSuccess()) {
        std::cout << "Multipart upload completed." << std::endl;
        if (!hashDataResult.calculateObjectHash(totalHashBuffer, hashMethod))
    {
        std::cerr << "Error calculating hash." << std::endl;
        return false;
    }
    } else {
        std::cerr << "Error completing multipart upload." <<
            completeMultipartUploadOutcome.GetError().GetMessage()
            << std::endl;
    }

    return completeMultipartUploadOutcome.IsSuccess();
}
}

//! Routine which retrieves the string for a HASH_METHOD constant.
/*!
```

```

    \param: hashMethod: A HASH_METHOD constant.
    \return: String: A string description of the hash method.
*/
Aws::String AwsDoc::S3::stringForHashMethod(AwsDoc::S3::HASH_METHOD hashMethod) {
    switch (hashMethod) {
        case AwsDoc::S3::DEFAULT:
            return "Default";
        case AwsDoc::S3::MD5:
            return "MD5";
        case AwsDoc::S3::SHA1:
            return "SHA1";
        case AwsDoc::S3::SHA256:
            return "SHA256";
        case AwsDoc::S3::CRC32:
            return "CRC32";
        case AwsDoc::S3::CRC32C:
            return "CRC32C";
        default:
            return "Unknown";
    }
}

//! Routine that returns the ChecksumAlgorithm for a HASH_METHOD constant.
/*!
    \param: hashMethod: A HASH_METHOD constant.
    \return: ChecksumAlgorithm: The ChecksumAlgorithm enum.
*/
Aws::S3::Model::ChecksumAlgorithm
AwsDoc::S3::getChecksumAlgorithmForHashMethod(AwsDoc::S3::HASH_METHOD hashMethod)
{
    Aws::S3::Model::ChecksumAlgorithm result =
    Aws::S3::Model::ChecksumAlgorithm::NOT_SET;
    switch (hashMethod) {
        case AwsDoc::S3::DEFAULT:
            std::cerr << "getChecksumAlgorithmForHashMethod- DEFAULT is not
valid." << std::endl;
            break; // Default is not supported.
        case AwsDoc::S3::MD5:
            break; // Ignore MD5.
        case AwsDoc::S3::SHA1:
            result = Aws::S3::Model::ChecksumAlgorithm::SHA1;
            break;
        case AwsDoc::S3::SHA256:
            result = Aws::S3::Model::ChecksumAlgorithm::SHA256;

```

```

        break;
    case AwsDoc::S3::CRC32:
        result = Aws::S3::Model::ChecksumAlgorithm::CRC32;
        break;
    case AwsDoc::S3::CRC32C:
        result = Aws::S3::Model::ChecksumAlgorithm::CRC32C;
        break;
    default:
        std::cerr << "Unknown hash method." << std::endl;
        break;
    }

    return result;
}

//! Routine which cleans up after the example is complete.
/*!
    \param bucket: The name of the S3 bucket where the object was uploaded.
    \param clientConfiguration: The client configuration for the S3 client.
    \return bool: Function succeeded.
*/
bool AwsDoc::S3::cleanUp(const Aws::String &bucketName,
                        const Aws::S3::S3ClientConfiguration
                        &clientConfiguration) {

    Aws::Vector<Aws::String> keysResult;
    bool result = true;
    if (AwsDoc::S3::listObjects(bucketName, keysResult, clientConfiguration)) {
        if (!keysResult.empty()) {
            result = AwsDoc::S3::deleteObjects(keysResult, bucketName,
                                                clientConfiguration);
        }
    } else {
        result = false;
    }

    return result && AwsDoc::S3::deleteBucket(bucketName, clientConfiguration);
}

//! Console interaction introducing the workflow.
/*!
    \param bucketName: The name of the S3 bucket to use.
*/

```

```
void AwsDoc::S3::introductoryExplanations(const Aws::String &bucketName) {

    std::cout
        << "Welcome to the Amazon Simple Storage Service (Amazon S3) object
integrity workflow."
        << std::endl;
    printAsterisksLine();
    std::cout
        << "This workflow demonstrates how Amazon S3 uses checksum values to
verify the integrity of data\n";
    std::cout << "uploaded to Amazon S3 buckets" << std::endl;
    std::cout
        << "The AWS SDK for C++ automatically handles checksums.\n";
    std::cout
        << "By default it calculates a checksum that is uploaded with an
object.\n"
        << "The default checksum algorithm for PutObject and MultiPart upload
is an MD5 hash.\n"
        << "The default checksum algorithm for TransferManager uploads is a
CRC32 checksum."
        << std::endl;
    std::cout
        << "You can override the default behavior, requiring one of the
following checksums,\n";
    std::cout << "MD5, CRC32, CRC32C, SHA-1 or SHA-256." << std::endl;
    std::cout << "You can also set the checksum hash value, instead of letting
the SDK calculate the value."
        << std::endl;
    std::cout
        << "For more information, see https://docs.aws.amazon.com/AmazonS3/
latest/userguide/checking-object-integrity.html."
        << std::endl;

    std::cout
        << "This workflow will locally compute checksums for files uploaded
to an Amazon S3 bucket,\n";
    std::cout << "even when the SDK also computes the checksum." << std::endl;
    std::cout
        << "This is done to provide demonstration code for how the checksums
are calculated."
        << std::endl;
    std::cout << "A bucket named '" << bucketName << "' will be created for the
object uploads."
        << std::endl;
```

```
}

//! Console interaction which explains the PutObject results.
/*!
*/
void AwsDoc::S3::explainPutObjectResults() {

    std::cout << "The upload was successful.\n";
    std::cout << "If the checksums had not matched, the upload would have failed."
               << std::endl;
    std::cout
               << "The checksums calculated by the server have been retrieved using the GetObjectAttributes."
               << std::endl;
    std::cout
               << "The locally calculated checksums have been verified against the retrieved checksums."
               << std::endl;
}

//! Console interaction explaining transfer manager uploads.
/*!
\param objectKey: The key for the object being uploaded.
*/
void AwsDoc::S3::introductoryTransferManagerUploadExplanations(
    const Aws::String &objectKey) {
    std::cout
               << "Now the workflow will demonstrate object integrity for TransferManager multi-part uploads."
               << std::endl;
    std::cout
               << "The AWS C++ SDK has a TransferManager class which simplifies multipart uploads."
               << std::endl;
    std::cout
               << "The following code lets the TransferManager handle much of the checksum configuration."
               << std::endl;

    std::cout << "An object with the key '" << objectKey
               << " will be uploaded by the TransferManager using a "
               << BUFFER_SIZE_IN_MEGABYTES << " MB buffer." << std::endl;
    if (gUseCalculatedChecksum) {
```

```

        std::cout << "For TransferManager uploads, this demo always lets the SDK
calculate the hash value."
                << std::endl;
    }

    pressEnterToContinue();
    printAsterisksLine();
}

//! Console interaction explaining multi-part uploads.
/*!
\param objectKey: The key for the object being uploaded.
\param chosenHashMethod: The hash method selected by the user.
*/
void AwsDoc::S3::multiPartUploadExplanations(const Aws::String &objectKey,
                                              HASH_METHOD chosenHashMethod) {
    std::cout
        << "Now we will provide an in-depth demonstration of multi-part
uploading by calling the multi-part upload APIs directly."
        << std::endl;
    std::cout << "These are the same APIs used by the TransferManager when
uploading large files."
        << std::endl;
    std::cout
        << "In the following code, the checksums are also calculated locally
and then compared."
        << std::endl;
    std::cout
        << "For multi-part uploads, a checksum is uploaded with each part.
The final checksum is a concatenation of"
        << std::endl;
    std::cout << "the checksums for each part." << std::endl;
    std::cout
        << "This is explained in the user guide, https://docs.aws.amazon.com/
AmazonS3/latest/userguide/checking-object-integrity.html,"
        << " in the section \"Using part-level checksums for multipart
uploads\"." << std::endl;

    std::cout << "Starting multipart upload of with hash method " <<
        stringForHashMethod(chosenHashMethod) << " uploading to with object
key\n"
        << "'" << objectKey << "'," << std::endl;
}

```

```
//! Create a large file for doing multi-part uploads.
/*!
*/
bool AwsDoc::S3::createLargeFileIfNotExists() {
    // Generate a large file by writing this source file multiple times to a new
    file.
    if (std::filesystem::exists(MULTI_PART_TEST_FILE)) {
        return true;
    }

    std::ofstream newFile(MULTI_PART_TEST_FILE, std::ios::out
                           | std::ios::binary);

    if (!newFile) {
        std::cerr << "createLargeFileIfNotExists- Error creating file " <<
MULTI_PART_TEST_FILE <<
        std::endl;
        return false;
    }

    std::ifstream input(TEST_FILE, std::ios::in
                        | std::ios::binary);

    if (!input) {
        std::cerr << "Error opening file " << TEST_FILE <<
        std::endl;
        return false;
    }
    std::stringstream buffer;
    buffer << input.rdbuf();

    input.close();

    while (newFile.tellp() < LARGE_FILE_SIZE && !newFile.bad()) {
        buffer.seekg(std::stringstream::beg);
        newFile << buffer.rdbuf();
    }

    newFile.close();

    return true;
}
```


- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的下列主題。
 - [AbortMultipartUpload](#)
 - [CompleteMultipartUpload](#)
 - [CreateMultipartUpload](#)
 - [DeleteObject](#)
 - [GetObjectAttributes](#)
 - [PutObject](#)
 - [UploadPart](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 處理 Amazon S3 版本控制的物件

以下程式碼範例顯示做法：

- 建立已使用版本控制的 S3 儲存貯體。
- 取得物件的所有版本。
- 將物件復原至先前的版本。
- 刪除並還原已使用版本控制的物件。
- 永久刪除物件的所有版本。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立可包裝 S3 動作的函數。

```
def create_versioned_bucket(bucket_name, prefix):
    """
    Creates an Amazon S3 bucket, enables it for versioning, and configures a
    lifecycle
    that expires noncurrent object versions after 7 days.

    Adding a lifecycle configuration to a versioned bucket is a best practice.
    It helps prevent objects in the bucket from accumulating a large number of
    noncurrent versions, which can slow down request performance.

    Usage is shown in the usage_demo_single_object function at the end of this
    module.

    :param bucket_name: The name of the bucket to create.
    :param prefix: Identifies which objects are automatically expired under the
                    configured lifecycle rules.
    :return: The newly created bucket.
    """
    try:
        bucket = s3.create_bucket(
            Bucket=bucket_name,
            CreateBucketConfiguration={
                "LocationConstraint": s3.meta.client.meta.region_name
            },
        )
        logger.info("Created bucket %s.", bucket.name)
    except ClientError as error:
        if error.response["Error"]["Code"] == "BucketAlreadyOwnedByYou":
            logger.warning("Bucket %s already exists! Using it.", bucket_name)
            bucket = s3.Bucket(bucket_name)
        else:
            logger.exception("Couldn't create bucket %s.", bucket_name)
            raise

    try:
        bucket.Versioning().enable()
        logger.info("Enabled versioning on bucket %s.", bucket.name)
    except ClientError:
        logger.exception("Couldn't enable versioning on bucket %s.", bucket.name)
        raise

    try:
        expiration = 7
```

```

        bucket.LifecycleConfiguration().put(
            LifecycleConfiguration={
                "Rules": [
                    {
                        "Status": "Enabled",
                        "Prefix": prefix,
                        "NoncurrentVersionExpiration": {"NoncurrentDays":
expiration},
                    }
                ]
            }
        )
        logger.info(
            "Configured lifecycle to expire noncurrent versions after %s days "
            "on bucket %s.",
            expiration,
            bucket.name,
        )
    except ClientError as error:
        logger.warning(
            "Couldn't configure lifecycle on bucket %s because %s. "
            "Continuing anyway.",
            bucket.name,
            error,
        )

    return bucket

def rollback_object(bucket, object_key, version_id):
    """
    Rolls back an object to an earlier version by deleting all versions that
    occurred after the specified rollback version.

    Usage is shown in the usage_demo_single_object function at the end of this
    module.

    :param bucket: The bucket that holds the object to roll back.
    :param object_key: The object to roll back.
    :param version_id: The version ID to roll back to.
    """
    # Versions must be sorted by last_modified date because delete markers are
    # at the end of the list even when they are interspersed in time.

```

```

versions = sorted(
    bucket.object_versions.filter(Prefix=object_key),
    key=attrgetter("last_modified"),
    reverse=True,
)

logger.debug(
    "Got versions:\n%s",
    "\n".join(
        [
            f"\t{version.version_id}, last modified {version.last_modified}"
            for version in versions
        ]
    ),
)

if version_id in [ver.version_id for ver in versions]:
    print(f"Rolling back to version {version_id}")
    for version in versions:
        if version.version_id != version_id:
            version.delete()
            print(f"Deleted version {version.version_id}")
        else:
            break

    print(f"Active version is now {bucket.Object(object_key).version_id}")
else:
    raise KeyError(
        f"{version_id} was not found in the list of versions for "
        f"{object_key}."
    )

def revive_object(bucket, object_key):
    """
    Revives a versioned object that was deleted by removing the object's active
    delete marker.
    A versioned object presents as deleted when its latest version is a delete
    marker.
    By removing the delete marker, we make the previous version the latest
    version
    and the object then presents as *not* deleted.
    """

```

Usage is shown in the `usage_demo_single_object` function at the end of this module.

```
:param bucket: The bucket that contains the object.
:param object_key: The object to revive.
"""
# Get the latest version for the object.
response = s3.meta.client.list_object_versions(
    Bucket=bucket.name, Prefix=object_key, MaxKeys=1
)

if "DeleteMarkers" in response:
    latest_version = response["DeleteMarkers"][0]
    if latest_version["IsLatest"]:
        logger.info(
            "Object %s was indeed deleted on %s. Let's revive it.",
            object_key,
            latest_version["LastModified"],
        )
        obj = bucket.Object(object_key)
        obj.Version(latest_version["VersionId"]).delete()
        logger.info(
            "Revived %s, active version is now %s with body '%s'",
            object_key,
            obj.version_id,
            obj.get()["Body"].read(),
        )
    else:
        logger.warning(
            "Delete marker is not the latest version for %s!", object_key
        )
elif "Versions" in response:
    logger.warning("Got an active version for %s, nothing to do.",
object_key)
else:
    logger.error("Couldn't get any version info for %s.", object_key)

def permanently_delete_object(bucket, object_key):
    """
    Permanently deletes a versioned object by deleting all of its versions.
```

Usage is shown in the `usage_demo_single_object` function at the end of this module.

```
:param bucket: The bucket that contains the object.
:param object_key: The object to delete.
"""
try:
    bucket.object_versions.filter(Prefix=object_key).delete()
    logger.info("Permanently deleted all versions of object %s.", object_key)
except ClientError:
    logger.exception("Couldn't delete all versions of %s.", object_key)
    raise
```

上傳詩句的片段至已使用版本控制的物件，並執行一系列動作。

```
def usage_demo_single_object(obj_prefix="demo-versioning/"):
    """
    Demonstrates usage of versioned object functions. This demo uploads a stanza
    of a poem and performs a series of revisions, deletions, and revivals on it.

    :param obj_prefix: The prefix to assign to objects created by this demo.
    """
    with open("father_william.txt") as file:
        stanzas = file.read().split("\n\n")

    width = get_terminal_size((80, 20))[0]
    print("-" * width)
    print("Welcome to the usage demonstration of Amazon S3 versioning.")
    print(
        "This demonstration uploads a single stanza of a poem to an Amazon "
        "S3 bucket and then applies various revisions to it."
    )
    print("-" * width)
    print("Creating a version-enabled bucket for the demo...")
    bucket = create_versioned_bucket("bucket-" + str(uuid.uuid1()), obj_prefix)

    print("\nThe initial version of our stanza:")
    print(stanzas[0])

    # Add the first stanza and revise it a few times.
```

```

print("\nApplying some revisions to the stanza...")
obj_stanza_1 = bucket.Object(f"{obj_prefix}stanza-1")
obj_stanza_1.put(Body=bytes(stanzas[0], "utf-8"))
obj_stanza_1.put(Body=bytes(stanzas[0].upper(), "utf-8"))
obj_stanza_1.put(Body=bytes(stanzas[0].lower(), "utf-8"))
obj_stanza_1.put(Body=bytes(stanzas[0][::-1], "utf-8"))
print(
    "The latest version of the stanza is now:",
    obj_stanza_1.get()["Body"].read().decode("utf-8"),
    sep="\n",
)

# Versions are returned in order, most recent first.
obj_stanza_1_versions =
bucket.object_versions.filter(Prefix=obj_stanza_1.key)
print(
    "The version data of the stanza revisions:",
    *[
        f"    {version.version_id}, last modified {version.last_modified}"
        for version in obj_stanza_1_versions
    ],
    sep="\n",
)

# Rollback two versions.
print("\nRolling back two versions...")
rollback_object(bucket, obj_stanza_1.key, list(obj_stanza_1_versions)
[2].version_id)
print(
    "The latest version of the stanza:",
    obj_stanza_1.get()["Body"].read().decode("utf-8"),
    sep="\n",
)

# Delete the stanza
print("\nDeleting the stanza...")
obj_stanza_1.delete()
try:
    obj_stanza_1.get()
except ClientError as error:
    if error.response["Error"]["Code"] == "NoSuchKey":
        print("The stanza is now deleted (as expected).")
    else:
        raise

```

```
# Revive the stanza
print("\nRestoring the stanza...")
revive_object(bucket, obj_stanza_1.key)
print(
    "The stanza is restored! The latest version is again:",
    obj_stanza_1.get()["Body"].read().decode("utf-8"),
    sep="\n",
)

# Permanently delete all versions of the object. This cannot be undone!
print("\nPermanently deleting all versions of the stanza...")
permanently_delete_object(bucket, obj_stanza_1.key)
obj_stanza_1_versions =
bucket.object_versions.filter(Prefix=obj_stanza_1.key)
if len(list(obj_stanza_1_versions)) == 0:
    print("The stanza has been permanently deleted and now has no versions.")
else:
    print("Something went wrong. The stanza still exists!")

print(f"\nRemoving {bucket.name}...")
bucket.delete()
print(f"{bucket.name} deleted.")
print("Demo done!")
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的下列主題。
 - [CreateBucket](#)
 - [DeleteObject](#)
 - [ListObjectVersions](#)
 - [PutBucketLifecycleConfiguration](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Amazon S3 的無伺服器範例

下列程式碼範例示範如何使用 Amazon S3 搭配 AWS SDKs。

範例

- [使用 Amazon S3 觸發條件調用 Lambda 函數](#)

使用 Amazon S3 觸發條件調用 Lambda 函數

下列程式碼範例顯示如何實作 Lambda 函數來接收上傳物件至 S3 儲存貯體時觸發的事件。此函數會從事件參數擷取 S3 儲存貯體名稱和物件金鑰，並呼叫 Amazon S3 API 以擷取和記錄物件的內容類型。

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[無伺服器範例](#)儲存庫中設定和執行。

使用 .NET 搭配 Lambda 來使用 S3 事件。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
using System.Threading.Tasks;
using Amazon.Lambda.Core;
using Amazon.S3;
using System;
using Amazon.Lambda.S3Events;
using System.Web;

// Assembly attribute to enable the Lambda function's JSON input to be converted
// into a .NET class.
[assembly:
    LambdaSerializer(typeof(Amazon.Lambda.Serialization.SystemTextJson.DefaultLambdaJsonSerializer))]

namespace S3Integration
{
    public class Function
    {
        private static AmazonS3Client _s3Client;
        public Function() : this(null)
        {
```

```
    }

    internal Function(AmazonS3Client s3Client)
    {
        _s3Client = s3Client ?? new AmazonS3Client();
    }

    public async Task<string> Handler(S3Event evt, ILambdaContext context)
    {
        try
        {
            if (evt.Records.Count <= 0)
            {
                context.Logger.LogLine("Empty S3 Event received");
                return string.Empty;
            }

            var bucket = evt.Records[0].S3.Bucket.Name;
            var key = HttpUtility.UrlDecode(evt.Records[0].S3.Object.Key);

            context.Logger.LogLine($"Request is for {bucket} and {key}");

            var objectResult = await _s3Client.GetObjectAsync(bucket, key);

            context.Logger.LogLine($"Returning {objectResult.Key}");

            return objectResult.Key;
        }
        catch (Exception e)
        {
            context.Logger.LogLine($"Error processing request - {e.Message}");

            return string.Empty;
        }
    }
}
```

Go

SDK for Go V2

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[無伺服器範例](#)儲存庫中設定和執行。

使用 Go 搭配 Lambda 來使用 S3 事件。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
package main

import (
    "context"
    "log"

    "github.com/aws/aws-lambda-go/events"
    "github.com/aws/aws-lambda-go/lambda"
    "github.com/aws/aws-sdk-go-v2/config"
    "github.com/aws/aws-sdk-go-v2/service/s3"
)

func handler(ctx context.Context, s3Event events.S3Event) error {
    sdkConfig, err := config.LoadDefaultConfig(ctx)
    if err != nil {
        log.Printf("failed to load default config: %s", err)
        return err
    }
    s3Client := s3.NewFromConfig(sdkConfig)

    for _, record := range s3Event.Records {
        bucket := record.S3.Bucket.Name
        key := record.S3.Object.URLDecodedKey
        headOutput, err := s3Client.HeadObject(ctx, &s3.HeadObjectInput{
            Bucket: &bucket,
            Key:    &key,
        })
        if err != nil {
            log.Printf("error getting head of object %s/%s: %s", bucket, key, err)
        }
    }
}
```

```
        return err
    }
    log.Printf("successfully retrieved %s/%s of type %s", bucket, key,
        *headOutput.ContentType)
    }

    return nil
}

func main() {
    lambda.Start(handler)
}
```

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[無伺服器範例](#)儲存庫中設定和執行。

使用 Java 搭配 Lambda 來使用 S3 事件。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
package example;

import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
import software.amazon.awssdk.services.s3.S3Client;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;
import com.amazonaws.services.lambda.runtime.events.S3Event;
import
    com.amazonaws.services.lambda.runtime.events.models.s3.S3EventNotification.S3EventNotifi

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
```

```
public class Handler implements RequestHandler<S3Event, String> {
    private static final Logger logger = LoggerFactory.getLogger(Handler.class);
    @Override
    public String handleRequest(S3Event s3event, Context context) {
        try {
            S3EventNotificationRecord record = s3event.getRecords().get(0);
            String srcBucket = record.getS3().getBucket().getName();
            String srcKey = record.getS3().getObject().getUrlDecodedKey();

            S3Client s3Client = S3Client.builder().build();
            HeadObjectResponse headObject = getHeadObject(s3Client, srcBucket,
srcKey);

            logger.info("Successfully retrieved " + srcBucket + "/" + srcKey + " of
type " + headObject.contentType());

            return "Ok";
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    private HeadObjectResponse getHeadObject(S3Client s3Client, String bucket,
String key) {
        HeadObjectRequest headObjectRequest = HeadObjectRequest.builder()
            .bucket(bucket)
            .key(key)
            .build();
        return s3Client.headObject(headObjectRequest);
    }
}
```

JavaScript

適用於 JavaScript (v3) 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[無伺服器範例](#)儲存庫中設定和執行。

使用 JavaScript 搭配 Lambda 來使用 S3 事件。

```
import { S3Client, HeadObjectCommand } from "@aws-sdk/client-s3";

const client = new S3Client();

export const handler = async (event, context) => {

    // Get the object from the event and show its content type
    const bucket = event.Records[0].s3.bucket.name;
    const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, ' '));

    try {
        const { ContentType } = await client.send(new HeadObjectCommand({
            Bucket: bucket,
            Key: key,
        }));

        console.log('CONTENT TYPE:', ContentType);
        return ContentType;

    } catch (err) {
        console.log(err);
        const message = `Error getting object ${key} from bucket ${bucket}. Make sure they exist and your bucket is in the same region as this function.`;
        console.log(message);
        throw new Error(message);
    }
};
```

使用 TypeScript 搭配 Lambda 來使用 S3 事件。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { S3Event } from 'aws-lambda';
import { S3Client, HeadObjectCommand } from '@aws-sdk/client-s3';

const s3 = new S3Client({ region: process.env.AWS_REGION });

export const handler = async (event: S3Event): Promise<string | undefined> => {
    // Get the object from the event and show its content type
```

```
const bucket = event.Records[0].s3.bucket.name;
const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, '
'));
const params = {
  Bucket: bucket,
  Key: key,
};
try {
  const { ContentType } = await s3.send(new HeadObjectCommand(params));
  console.log('CONTENT TYPE:', ContentType);
  return ContentType;
} catch (err) {
  console.log(err);
  const message = `Error getting object ${key} from bucket ${bucket}. Make sure
they exist and your bucket is in the same region as this function.`;
  console.log(message);
  throw new Error(message);
}
};
```

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[無伺服器範例](#)儲存庫中設定和執行。

使用 PHP 搭配 Lambda 來使用 S3 事件。

```
<?php

use Bref\Context\Context;
use Bref\Event\S3\S3Event;
use Bref\Event\S3\S3Handler;
use Bref\Logger\StderrLogger;

require __DIR__ . '/vendor/autoload.php';
```

```
class Handler extends S3Handler
{
    private StderrLogger $logger;
    public function __construct(StderrLogger $logger)
    {
        $this->logger = $logger;
    }

    public function handleS3(S3Event $event, Context $context) : void
    {
        $this->logger->info("Processing S3 records");

        // Get the object from the event and show its content type
        $records = $event->getRecords();

        foreach ($records as $record)
        {
            $bucket = $record->getBucket()->getName();
            $key = urldecode($record->getObject()->getKey());

            try {
                $fileSize = urldecode($record->getObject()->getSize());
                echo "File Size: " . $fileSize . "\n";
                // TODO: Implement your custom processing logic here
            } catch (Exception $e) {
                echo $e->getMessage() . "\n";
                echo 'Error getting object ' . $key . ' from bucket ' .
                $bucket . '. Make sure they exist and your bucket is in the same region as this
                function.' . "\n";
                throw $e;
            }
        }
    }
}

$logger = new StderrLogger();
return new Handler($logger);
```


Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[無伺服器範例](#)儲存庫中設定和執行。

使用 Python 搭配 Lambda 來使用 S3 事件。

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
import json
import urllib.parse
import boto3

print('Loading function')

s3 = boto3.client('s3')

def lambda_handler(event, context):
    #print("Received event: " + json.dumps(event, indent=2))

    # Get the object from the event and show its content type
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'],
    encoding='utf-8')
    try:
        response = s3.get_object(Bucket=bucket, Key=key)
        print("CONTENT TYPE: " + response['ContentType'])
        return response['ContentType']
    except Exception as e:
        print(e)
        print('Error getting object {} from bucket {}. Make sure they exist and
        your bucket is in the same region as this function.'.format(key, bucket))
        raise e
```

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[無伺服器範例](#)儲存庫中設定和執行。

使用 Ruby 搭配 Lambda 來使用 S3 事件。

```
require 'json'
require 'uri'
require 'aws-sdk'

puts 'Loading function'

def lambda_handler(event:, context:)
  s3 = Aws::S3::Client.new(region: 'region') # Your AWS region
  # puts "Received event: #{JSON.dump(event)}"

  # Get the object from the event and show its content type
  bucket = event['Records'][0]['s3']['bucket']['name']
  key = URI.decode_www_form_component(event['Records'][0]['s3']['object']['key'],
    Encoding::UTF_8)
  begin
    response = s3.get_object(bucket: bucket, key: key)
    puts "CONTENT TYPE: #{response.content_type}"
    return response.content_type
  rescue StandardError => e
    puts e.message
    puts "Error getting object #{key} from bucket #{bucket}. Make sure they exist
    and your bucket is in the same region as this function."
    raise e
  end
end
```

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[無伺服器範例](#)儲存庫中設定和執行。

使用 Rust 搭配 Lambda 來使用 S3 事件。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
use aws_lambda_events::event::s3::S3Event;
use aws_sdk_s3::{Client};
use lambda_runtime::{run, service_fn, Error, LambdaEvent};

/// Main function
#[tokio::main]
async fn main() -> Result<(), Error> {
    tracing_subscriber::fmt()
        .with_max_level(tracing::Level::INFO)
        .with_target(false)
        .without_time()
        .init();

    // Initialize the AWS SDK for Rust
    let config = aws_config::load_from_env().await;
    let s3_client = Client::new(&config);

    let res = run(service_fn(|request: LambdaEvent<S3Event>| {
        function_handler(&s3_client, request)
    })).await;

    res
}

async fn function_handler(
    s3_client: &Client,
    evt: LambdaEvent<S3Event>
) -> Result<(), Error> {
```

```
tracing::info!(records = ?evt.payload.records.len(), "Received request from
SQS");

if evt.payload.records.len() == 0 {
    tracing::info!("Empty S3 event received");
}

let bucket = evt.payload.records[0].s3.bucket.name.as_ref().expect("Bucket
name to exist");
let key = evt.payload.records[0].s3.object.key.as_ref().expect("Object key to
exist");

tracing::info!("Request is for {} and object {}", bucket, key);

let s3_get_object_result = s3_client
    .get_object()
    .bucket(bucket)
    .key(key)
    .send()
    .await;

match s3_get_object_result {
    Ok(_) => tracing::info!("S3 Get Object success, the s3GetObjectResult
contains a 'body' property of type ByteStream"),
    Err(_) => tracing::info!("Failure with S3 Get Object request")
}

Ok(())
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDKs 的 Amazon S3 控制程式碼範例

下列程式碼範例示範如何搭配 AWS 軟體開發套件 (SDK) 使用 Amazon S3 Control。

基本概念是程式碼範例，這些範例說明如何在服務內執行基本操作。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Amazon S3 Control

- [使用 AWS SDKs 的 Amazon S3 控制基本範例](#)
 - [Hello Amazon S3 Control](#)
 - [使用 AWS SDK 了解 Amazon S3 控制的基本概念](#)
 - [使用 AWS SDKs 的 Amazon S3 控制動作](#)
 - [CreateJob 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteJobTagging 搭配 AWS SDK 使用](#)
 - [DescribeJob 搭配 AWS SDK 或 CLI 使用](#)
 - [GetJobTagging 搭配 AWS SDK 使用](#)
 - [PutJobTagging 搭配 AWS SDK 使用](#)
 - [UpdateJobPriority 搭配 AWS SDK 或 CLI 使用](#)
 - [UpdateJobStatus 搭配 AWS SDK 或 CLI 使用](#)

使用 AWS SDKs 的 Amazon S3 控制基本範例

下列程式碼範例示範如何搭配 AWS SDK 使用 Amazon S3 Control 的基本功能。

範例

- [Hello Amazon S3 Control](#)
- [使用 AWS SDK 了解 Amazon S3 控制的基本概念](#)
- [使用 AWS SDKs 的 Amazon S3 控制動作](#)
 - [CreateJob 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteJobTagging 搭配 AWS SDK 使用](#)
 - [DescribeJob 搭配 AWS SDK 或 CLI 使用](#)
 - [GetJobTagging 搭配 AWS SDK 使用](#)
 - [PutJobTagging 搭配 AWS SDK 使用](#)
 - [UpdateJobPriority 搭配 AWS SDK 或 CLI 使用](#)
 - [UpdateJobStatus 搭配 AWS SDK 或 CLI 使用](#)

Hello Amazon S3 Control

下列程式碼範例示範如何開始使用 Amazon S3 Control。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import
    software.amazon.awssdk.auth.credentials.EnvironmentVariableCredentialsProvider;
import software.amazon.awssdk.core.client.config.ClientOverrideConfiguration;
import software.amazon.awssdk.core.retry.RetryMode;
import software.amazon.awssdk.core.retry.RetryPolicy;
import software.amazon.awssdk.http.async.SdkAsyncHttpClient;
import software.amazon.awssdk.http.nio.netty.NettyNioAsyncHttpClient;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlAsyncClient;
import software.amazon.awssdk.services.s3control.model.JobListDescriptor;
import software.amazon.awssdk.services.s3control.model.JobStatus;
import software.amazon.awssdk.services.s3control.model.ListJobsRequest;
import software.amazon.awssdk.services.s3control.paginators.ListJobsPublisher;
import java.time.Duration;
import java.util.List;
import java.util.concurrent.CompletableFuture;
import java.util.concurrent.CompletionException;

/**
 * Before running this example:
 * <p/>
 * The SDK must be able to authenticate AWS requests on your behalf. If you have
 * not configured
 * authentication for SDKs and tools, see https://docs.aws.amazon.com/sdkref/latest/guide/access.html in the AWS SDKs and Tools Reference Guide.
 * <p/>
 * You must have a runtime environment configured with the Java SDK.
```

```

* See https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/setup.html
in the Developer Guide if this is not set up.
*/
public class HelloS3Batch {
    private static S3ControlAsyncClient asyncClient;

    public static void main(String[] args) {
        S3BatchActions actions = new S3BatchActions();
        String accountId = actions.getAccountId();
        try {
            listBatchJobsAsync(accountId)
                .exceptionally(ex -> {
                    System.err.println("List batch jobs failed: " +
ex.getMessage());
                    return null;
                })
                .join();

        } catch (CompletionException ex) {
            System.err.println("Failed to list batch jobs: " + ex.getMessage());
        }
    }

    /**
     * Retrieves the asynchronous S3 Control client instance.
     * <p>
     * This method creates and returns a singleton instance of the {@link
S3ControlAsyncClient}. If the instance
     * has not been created yet, it will be initialized with the following
configuration:
     * <ul>
     * <li>Maximum concurrency: 100</li>
     * <li>Connection timeout: 60 seconds</li>
     * <li>Read timeout: 60 seconds</li>
     * <li>Write timeout: 60 seconds</li>
     * <li>API call timeout: 2 minutes</li>
     * <li>API call attempt timeout: 90 seconds</li>
     * <li>Retry policy: 3 retries</li>
     * <li>Region: US_EAST_1</li>
     * <li>Credentials provider: {@link
EnvironmentVariableCredentialsProvider}</li>
     * </ul>
     *
     * @return the asynchronous S3 Control client instance

```

```
    */
    private static S3ControlAsyncClient getAsyncClient() {
        if (asyncClient == null) {
            SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
                .maxConcurrency(100)
                .connectionTimeout(Duration.ofSeconds(60))
                .readTimeout(Duration.ofSeconds(60))
                .writeTimeout(Duration.ofSeconds(60))
                .build();

            ClientOverrideConfiguration overrideConfig =
                ClientOverrideConfiguration.builder()
                    .apiCallTimeout(Duration.ofMinutes(2))
                    .apiCallAttemptTimeout(Duration.ofSeconds(90))
                    .retryStrategy(RetryMode.STANDARD)
                    .build();

            asyncClient = S3ControlAsyncClient.builder()
                .region(Region.US_EAST_1)
                .httpClient(httpClient)
                .overrideConfiguration(overrideConfig)
                .build();
        }
        return asyncClient;
    }

    /**
     * Asynchronously lists batch jobs that have completed for the specified
     * account.
     *
     * @param accountId the ID of the account to list jobs for
     * @return a CompletableFuture that completes when the job listing operation
     * is finished
     */
    public static CompletableFuture<Void> listBatchJobsAsync(String accountId) {
        ListJobsRequest jobsRequest = ListJobsRequest.builder()
            .jobStatuses(JobStatus.COMPLETE)
            .accountId(accountId)
            .maxResults(10)
            .build();

        ListJobsPublisher publisher =
            getAsyncClient().listJobsPaginator(jobsRequest);
        return publisher.subscribe(response -> {
```



```

        List<JobListDescriptor> jobs = response.jobs();
        for (JobListDescriptor job : jobs) {
            System.out.println("The job id is " + job.jobId());
            System.out.println("The job priority is " + job.priority());
        }
    }).thenAccept(response -> {
        System.out.println("Listing batch jobs completed");
    }).exceptionally(ex -> {
        System.err.println("Failed to list batch jobs: " + ex.getMessage());
        throw new RuntimeException(ex);
    });
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListJobs](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

def list_jobs(self, account_id: str) -> None:
    """
    List all batch jobs for the account.

    Args:
        account_id (str): AWS account ID
    """
    try:
        response = self.s3control_client.list_jobs(
            AccountId=account_id,
            JobStatuses=['Active', 'Complete', 'Cancelled', 'Failed', 'New',
                        'Paused', 'Pausing', 'Preparing', 'Ready', 'Suspended']
        )
        jobs = response.get('Jobs', [])
        for job in jobs:
            print(f"The job id is {job['JobId']}")
    
```

```
        print(f"The job priority is {job['Priority']}")
    except ClientError as e:
        print(f"Error listing jobs: {e}")
        raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [ListJobs](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 了解 Amazon S3 控制的基本概念

下列程式碼範例示範如何學習 Amazon S3 Control 的核心操作。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

了解核心操作。

```
package com.example.s3.batch;

import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.IOException;
import java.util.Map;
import java.util.Scanner;
import java.util.UUID;
import java.util.concurrent.CompletionException;

public class S3BatchScenario {

    public static final String DASHES = new String(new char[80]).replace("\0",
"-");
    private static final String STACK_NAME = "MyS3Stack";
    public static void main(String[] args) throws IOException {
```

```
S3BatchActions actions = new S3BatchActions();
String accountId = actions.getAccountId();
String uuid = java.util.UUID.randomUUID().toString();
Scanner scanner = new Scanner(System.in);

System.out.println(DASHES);
System.out.println("Welcome to the Amazon S3 Batch basics scenario.");
System.out.println("""
    S3 Batch operations enables efficient and cost-effective processing
of large-scale
    data stored in Amazon S3. It automatically scales resources to handle
varying workloads
    without the need for manual intervention.

    One of the key features of S3 Batch is its ability to perform tagging
operations on objects stored in
    S3 buckets. Users can leverage S3 Batch to apply, update, or remove
tags on thousands or millions of
    objects in a single operation, streamlining the management and
organization of their data.

    This can be particularly useful for tasks such as cost allocation,
lifecycle management, or
    metadata-driven workflows, where consistent and accurate tagging is
essential.

    S3 Batch's scalability and serverless nature make it an ideal
solution for organizations with
    growing data volumes and complex data management requirements.

    This Java program walks you through Amazon S3 Batch operations.

    Let's get started...

    """);
waitForInputToContinue(scanner);
// Use CloudFormation to stand up the resource required for this
scenario.
System.out.println("Use CloudFormation to stand up the resource required
for this scenario.");
CloudFormationHelper.deployCloudFormationStack(STACK_NAME);

Map<String, String> stackOutputs =
CloudFormationHelper.getStackOutputs(STACK_NAME);
String iamRoleArn = stackOutputs.get("S3BatchRoleArn");
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Setup the required bucket for this scenario.");
        waitForInputToContinue(scanner);
        String bucketName = "amzn-s3-demo-bucket"; // Replace with your bucket
name.
        actions.createBucket(bucketName);
        String reportBucketName = "arn:aws:s3:::"+bucketName;
        String manifestLocation = "arn:aws:s3:::"+bucketName+"/job-manifest.csv";
        System.out.println("Populate the bucket with the required files.");
        String[] fileNames = {"job-manifest.csv", "object-key-1.txt", "object-
key-2.txt", "object-key-3.txt", "object-key-4.txt"};
        actions.uploadFilesToBucket(bucketName, fileNames, actions);
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("1. Create a S3 Batch Job");
        System.out.println("This job tags all objects listed in the manifest file
with tags");
        waitForInputToContinue(scanner);
        String jobId ;
        try {
            jobId = actions.createS3JobAsync(accountId, iamRoleArn,
manifestLocation, reportBucketName, uuid).join();
            System.out.println("The Job id is " + jobId);

        } catch (S3Exception e) {
            System.err.println("SSM error: " + e.getMessage());
            return;
        } catch (RuntimeException e) {
            System.err.println("Unexpected error: " + e.getMessage());
            return;
        }

        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Update an existing S3 Batch Operations job's
priority");
        System.out.println(""""
```

In this step, we modify the job priority value. The higher the number, the higher the priority.

So, a job with a priority of `30` would have a higher priority than a job with

a priority of `20`. This is a common way to represent the priority of a task

or job, with higher numbers indicating a higher priority.

Ensure that the job status allows for priority updates. Jobs in certain

states (e.g., Cancelled, Failed, or Completed) cannot have their priorities

updated. Only jobs in the Active or Suspended state typically allow priority

updates.

""");

```
try {
    actions.updateJobPriorityAsync(jobId, accountId)
        .exceptionally(ex -> {
            System.err.println("Update job priority failed: " +
ex.getMessage());
            return null;
        })
        .join();
} catch (CompletionException ex) {
    System.err.println("Failed to update job priority: " +
ex.getMessage());
}
waitForInputToContinue(scanner);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("3. Cancel the S3 Batch job");
System.out.print("Do you want to cancel the Batch job? (y/n): ");
String cancelAns = scanner.nextLine();
if (cancelAns != null && cancelAns.trim().equalsIgnoreCase("y")) {
    try {
        actions.cancelJobAsync(jobId, accountId)
            .exceptionally(ex -> {
                System.err.println("Cancel job failed: " +
ex.getMessage());
                return null;
            })
    }
```

```
        .join();
    } catch (CompletionException ex) {
        System.err.println("Failed to cancel job: " + ex.getMessage());
    }
} else {
    System.out.println("Job " + jobId + " was not canceled.");
}
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("4. Describe the job that was just created");
waitForInputToContinue(scanner);
try {
    actions.describeJobAsync(jobId, accountId)
        .exceptionally(ex -> {
            System.err.println("Describe job failed: " +
ex.getMessage());
            return null;
        })
        .join();
} catch (CompletionException ex) {
    System.err.println("Failed to describe job: " + ex.getMessage());
}
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("5. Describe the tags associated with the job");
waitForInputToContinue(scanner);
try {
    actions.getJobTagsAsync(jobId, accountId)
        .exceptionally(ex -> {
            System.err.println("Get job tags failed: " +
ex.getMessage());
            return null;
        })
        .join();
} catch (CompletionException ex) {
    System.err.println("Failed to get job tags: " + ex.getMessage());
}
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("6. Update Batch Job Tags");
waitForInputToContinue(scanner);
```

```
        try {
            actions.putJobTaggingAsync(jobId, accountId)
                .exceptionally(ex -> {
                    System.err.println("Put job tagging failed: " +
ex.getMessage());
                    return null;
                })
                .join();
        } catch (CompletionException ex) {
            System.err.println("Failed to put job tagging: " + ex.getMessage());
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Delete the Amazon S3 Batch job tagging.");
        System.out.print("Do you want to delete Batch job tagging? (y/n)");
        String delAns = scanner.nextLine();
        if (delAns != null && delAns.trim().equalsIgnoreCase("y")) {
            try {
                actions.deleteBatchJobTagsAsync(jobId, accountId)
                    .exceptionally(ex -> {
                        System.err.println("Delete batch job tags failed: " +
ex.getMessage());
                        return null;
                    })
                    .join();
            } catch (CompletionException ex) {
                System.err.println("Failed to delete batch job tags: " +
ex.getMessage());
            }
        } else {
            System.out.println("Tagging was not deleted.");
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.print("Do you want to delete the AWS resources used in this
scenario? (y/n)");
        String delResAns = scanner.nextLine();
        if (delResAns != null && delResAns.trim().equalsIgnoreCase("y")) {
            actions.deleteFilesFromBucket(bucketName, fileNames, actions);
            actions.deleteBucketFolderAsync(bucketName);
            actions.deleteBucket(bucketName)
                .thenRun(() -> System.out.println("Bucket deletion completed"))
```

```

        .exceptionally(ex -> {
            System.err.println("Error occurred: " + ex.getMessage());
            return null;
        });
        CloudFormationHelper.destroyCloudFormationStack(STACK_NAME);
    } else {
        System.out.println("The AWS resources were not deleted.");
    }
    System.out.println("The Amazon S3 Batch scenario has successfully
completed.");
    System.out.println(DASHES);
}

private static void waitForInputToContinue(Scanner scanner) {
    while (true) {
        System.out.println();
        System.out.println("Enter 'c' followed by <ENTER> to continue:");
        String input = scanner.nextLine();

        if (input.trim().equalsIgnoreCase("c")) {
            System.out.println("Continuing with the program...");
            System.out.println();
            break;
        } else {
            // Handle invalid input.
            System.out.println("Invalid input. Please try again.");
        }
    }
}
}
}

```

包裝操作的動作類別。

```

public class S3BatchActions {

    private static S3ControlAsyncClient asyncClient;

    private static S3AsyncClient s3AsyncClient ;
    /**
     * Retrieves the asynchronous S3 Control client instance.

```



```

    * <p>
    * This method creates and returns a singleton instance of the {@link
    S3ControlAsyncClient}. If the instance
    * has not been created yet, it will be initialized with the following
    configuration:
    * <ul>
    *   <li>Maximum concurrency: 100</li>
    *   <li>Connection timeout: 60 seconds</li>
    *   <li>Read timeout: 60 seconds</li>
    *   <li>Write timeout: 60 seconds</li>
    *   <li>API call timeout: 2 minutes</li>
    *   <li>API call attempt timeout: 90 seconds</li>
    *   <li>Retry policy: 3 retries</li>
    *   <li>Region: US_EAST_1</li>
    *   <li>Credentials provider: {@link
    EnvironmentVariableCredentialsProvider}</li>
    * </ul>
    *
    * @return the asynchronous S3 Control client instance
    */
    private static S3ControlAsyncClient getAsyncClient() {
        if (asyncClient == null) {
            SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
                .maxConcurrency(100)
                .connectionTimeout(Duration.ofSeconds(60))
                .readTimeout(Duration.ofSeconds(60))
                .writeTimeout(Duration.ofSeconds(60))
                .build();

            ClientOverrideConfiguration overrideConfig =
            ClientOverrideConfiguration.builder()
                .apiCallTimeout(Duration.ofMinutes(2))
                .apiCallAttemptTimeout(Duration.ofSeconds(90))
                .retryPolicy(RetryPolicy.builder()
                    .numRetries(3)
                    .build())
                .build();

            asyncClient = S3ControlAsyncClient.builder()
                .region(Region.US_EAST_1)
                .httpClient(httpClient)
                .overrideConfiguration(overrideConfig)
                .build();
        }
    }

```

```

        return asyncClient;
    }

    private static S3AsyncClient getS3AsyncClient() {
        if (asyncClient == null) {
            SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
                .maxConcurrency(100)
                .connectionTimeout(Duration.ofSeconds(60))
                .readTimeout(Duration.ofSeconds(60))
                .writeTimeout(Duration.ofSeconds(60))
                .build();

            ClientOverrideConfiguration overrideConfig =
                ClientOverrideConfiguration.builder()
                    .apiCallTimeout(Duration.ofMinutes(2))
                    .apiCallAttemptTimeout(Duration.ofSeconds(90))
                    .retryStrategy(RetryMode.STANDARD)
                    .build();

            s3AsyncClient = S3AsyncClient.builder()
                .region(Region.US_EAST_1)
                .httpClient(httpClient)
                .overrideConfiguration(overrideConfig)
                .build();
        }
        return s3AsyncClient;
    }

    /**
     * Cancels a job asynchronously.
     *
     * @param jobId The ID of the job to be canceled.
     * @param accountId The ID of the account associated with the job.
     * @return A {@link CompletableFuture} that completes when the job status has
     *         been updated to "CANCELLED".
     *         If an error occurs during the update, the returned future will
     *         complete exceptionally.
     */
    public CompletableFuture<Void> cancelJobAsync(String jobId, String accountId)
    {
        UpdateJobStatusRequest updateJobStatusRequest =
            UpdateJobStatusRequest.builder()
                .accountId(accountId)

```

```
        .jobId(jobId)
        .requestedJobStatus(String.valueOf(JobStatus.CANCELLED))
        .build();

    return asyncClient.updateJobStatus(updateJobStatusRequest)
        .thenAccept(updateJobStatusResponse -> {
            System.out.println("Job status updated to: " +
updateJobStatusResponse.status());
        })
        .exceptionally(ex -> {
            System.err.println("Failed to cancel job: " + ex.getMessage());
            throw new RuntimeException(ex); // Propagate the exception
        });
}

/**
 * Updates the priority of a job asynchronously.
 *
 * @param jobId      the ID of the job to update
 * @param accountId  the ID of the account associated with the job
 * @return a {@link CompletableFuture} that represents the asynchronous
operation, which completes when the job priority has been updated or an error
has occurred
 */
public CompletableFuture<Void> updateJobPriorityAsync(String jobId, String
accountId) {
    UpdateJobPriorityRequest priorityRequest =
UpdateJobPriorityRequest.builder()
        .accountId(accountId)
        .jobId(jobId)
        .priority(60)
        .build();

    CompletableFuture<Void> future = new CompletableFuture<>();
    getAsyncClient().updateJobPriority(priorityRequest)
        .thenAccept(response -> {
            System.out.println("The job priority was updated");
            future.complete(null); // Complete the CompletableFuture on
successful execution
        })
        .exceptionally(ex -> {
            System.err.println("Failed to update job priority: " +
ex.getMessage());

```

```

        future.completeExceptionally(ex); // Complete the
CompletableFuture exceptionally on error
        return null; // Return null to handle the exception
    });

    return future;
}

/**
 * Asynchronously retrieves the tags associated with a specific job in an AWS
account.
 *
 * @param jobId      the ID of the job for which to retrieve the tags
 * @param accountId the ID of the AWS account associated with the job
 * @return a {@link CompletableFuture} that completes when the job tags have
been retrieved, or with an exception if the operation fails
 * @throws RuntimeException if an error occurs while retrieving the job tags
 */
public CompletableFuture<Void> getJobTagsAsync(String jobId, String
accountId) {
    GetJobTaggingRequest request = GetJobTaggingRequest.builder()
        .jobId(jobId)
        .accountId(accountId)
        .build();

    return asyncClient.getJobTagging(request)
        .thenAccept(response -> {
            List<S3Tag> tags = response.tags();
            if (tags.isEmpty()) {
                System.out.println("No tags found for job ID: " + jobId);
            } else {
                for (S3Tag tag : tags) {
                    System.out.println("Tag key is: " + tag.key());
                    System.out.println("Tag value is: " + tag.value());
                }
            }
        })
        .exceptionally(ex -> {
            System.err.println("Failed to get job tags: " + ex.getMessage());
            throw new RuntimeException(ex); // Propagate the exception
        });
}

/**

```

```
* Asynchronously deletes the tags associated with a specific batch job.
*
* @param jobId      The ID of the batch job whose tags should be deleted.
* @param accountId  The ID of the account associated with the batch job.
* @return A CompletableFuture that completes when the job tags have been
successfully deleted, or an exception is thrown if the deletion fails.
*/
public CompletableFuture<Void> deleteBatchJobTagsAsync(String jobId, String
accountId) {
    DeleteJobTaggingRequest jobTaggingRequest =
DeleteJobTaggingRequest.builder()
        .accountId(accountId)
        .jobId(jobId)
        .build();

    return asyncClient.deleteJobTagging(jobTaggingRequest)
        .thenAccept(response -> {
            System.out.println("You have successfully deleted " + jobId + "
tagging.");
        })
        .exceptionally(ex -> {
            System.err.println("Failed to delete job tags: " +
ex.getMessage());
            throw new RuntimeException(ex);
        });
}

/**
* Asynchronously describes the specified job.
*
* @param jobId      the ID of the job to describe
* @param accountId  the ID of the AWS account associated with the job
* @return a {@link CompletableFuture} that completes when the job
description is available
* @throws RuntimeException if an error occurs while describing the job
*/
public CompletableFuture<Void> describeJobAsync(String jobId, String
accountId) {
    DescribeJobRequest jobRequest = DescribeJobRequest.builder()
        .jobId(jobId)
        .accountId(accountId)
        .build();

    return getAsyncClient().describeJob(jobRequest)
```

```

        .thenAccept(response -> {
            System.out.println("Job ID: " + response.job().jobId());
            System.out.println("Description: " +
response.job().description());
            System.out.println("Status: " + response.job().statusAsString());
            System.out.println("Role ARN: " + response.job().roleArn());
            System.out.println("Priority: " + response.job().priority());
            System.out.println("Progress Summary: " +
response.job().progressSummary());

            // Print out details about the job manifest.
            JobManifest manifest = response.job().manifest();
            System.out.println("Manifest Location: " +
manifest.location().objectArn());
            System.out.println("Manifest ETag: " +
manifest.location().eTag());

            // Print out details about the job operation.
            JobOperation operation = response.job().operation();
            if (operation.s3PutObjectTagging() != null) {
                System.out.println("Operation: S3 Put Object Tagging");
                System.out.println("Tag Set: " +
operation.s3PutObjectTagging().tagSet());
            }

            // Print out details about the job report.
            JobReport report = response.job().report();
            System.out.println("Report Bucket: " + report.bucket());
            System.out.println("Report Prefix: " + report.prefix());
            System.out.println("Report Format: " + report.format());
            System.out.println("Report Enabled: " + report.enabled());
            System.out.println("Report Scope: " +
report.reportScopeAsString());
        })
        .exceptionally(ex -> {
            System.err.println("Failed to describe job: " + ex.getMessage());
            throw new RuntimeException(ex);
        });
    }

    /**
     * Creates an asynchronous S3 job using the AWS Java SDK.
     *
     * @param accountId      the AWS account ID associated with the job

```

```

    * @param iamRoleArn      the ARN of the IAM role to be used for the job
    * @param manifestLocation the location of the job manifest file in S3
    * @param reportBucketName the name of the S3 bucket to store the job report
    * @param uuid            a unique identifier for the job
    * @return a CompletableFuture that represents the asynchronous creation of
the S3 job.
    *      The CompletableFuture will return the job ID if the job is created
successfully,
    *      or throw an exception if there is an error.
    */
    public CompletableFuture<String> createS3JobAsync(String accountId, String
                                                    iamRoleArn,
                                                    String manifestLocation,
String reportBucketName, String uuid) {

        String[] bucketName = new String[]{" "};
        String[] parts = reportBucketName.split("::");
        if (parts.length > 1) {
            bucketName[0] = parts[1];
        } else {
            System.out.println("The input string does not contain the expected
format.");
        }

        return CompletableFuture.supplyAsync(() -> getETag(bucketName[0], "job-
manifest.csv"))
            .thenCompose(eTag -> {
                ArrayList<S3Tag> tagSet = new ArrayList<>();
                S3Tag s3Tag = S3Tag.builder()
                    .key("keyOne")
                    .value("ValueOne")
                    .build();
                S3Tag s3Tag2 = S3Tag.builder()
                    .key("keyTwo")
                    .value("ValueTwo")
                    .build();
                tagSet.add(s3Tag);
                tagSet.add(s3Tag2);

                S3SetObjectTaggingOperation objectTaggingOperation =
S3SetObjectTaggingOperation.builder()
                    .tagSet(tagSet)
                    .build();
            });
    }

```

```
        JobOperation jobOperation = JobOperation.builder()
            .s3PutObjectTagging(objectTaggingOperation)
            .build();

        JobManifestLocation jobManifestLocation =
JobManifestLocation.builder()
            .objectArn(manifestLocation)
            .eTag(eTag)
            .build();

        JobManifestSpec manifestSpec = JobManifestSpec.builder()
            .fieldsWithStrings("Bucket", "Key")
            .format("S3BatchOperations_CSV_20180820")
            .build();

        JobManifest jobManifest = JobManifest.builder()
            .spec(manifestSpec)
            .location(jobManifestLocation)
            .build();

        JobReport jobReport = JobReport.builder()
            .bucket(reportBucketName)
            .prefix("reports")
            .format("Report_CSV_20180820")
            .enabled(true)
            .reportScope("AllTasks")
            .build();

        CreateJobRequest jobRequest = CreateJobRequest.builder()
            .accountId(accountId)
            .description("Job created using the AWS Java SDK")
            .manifest(jobManifest)
            .operation(jobOperation)
            .report(jobReport)
            .priority(42)
            .roleArn(iamRoleArn)
            .clientRequestToken(uuid)
            .confirmationRequired(false)
            .build();

        // Create the job asynchronously.
        return getAsyncClient().createJob(jobRequest)
            .thenApply(CreateJobResponse::jobId);
    })
```



```

        .handle((jobId, ex) -> {
            if (ex != null) {
                Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
                if (cause instanceof S3ControlException) {
                    throw new CompletionException(cause);
                } else {
                    throw new RuntimeException(cause);
                }
            }
            return jobId;
        });
    }

    /**
     * Retrieves the ETag (Entity Tag) for an object stored in an Amazon S3
     bucket.
     *
     * @param bucketName the name of the Amazon S3 bucket where the object is
     stored
     * @param key the key (file name) of the object in the Amazon S3 bucket
     * @return the ETag of the object
     */
    public String getETag(String bucketName, String key) {
        S3Client s3Client = S3Client.builder()
            .region(Region.US_EAST_1)
            .build();

        HeadObjectRequest headObjectRequest = HeadObjectRequest.builder()
            .bucket(bucketName)
            .key(key)
            .build();

        HeadObjectResponse headObjectResponse =
s3Client.headObject(headObjectRequest);
        return headObjectResponse.eTag();
    }

    /**
     * Asynchronously adds tags to a job in the system.
     *
     * @param jobId the ID of the job to add tags to
     * @param accountId the account ID associated with the job

```

```
    * @return a CompletableFuture that completes when the tagging operation is
    finished
    */
    public CompletableFuture<Void> putJobTaggingAsync(String jobId, String
accountId) {
        S3Tag departmentTag = S3Tag.builder()
            .key("department")
            .value("Marketing")
            .build();

        S3Tag fiscalYearTag = S3Tag.builder()
            .key("FiscalYear")
            .value("2020")
            .build();

        PutJobTaggingRequest putJobTaggingRequest =
PutJobTaggingRequest.builder()
            .jobId(jobId)
            .accountId(accountId)
            .tags(departmentTag, fiscalYearTag)
            .build();

        return asyncClient.putJobTagging(putJobTaggingRequest)
            .thenRun(() -> {
                System.out.println("Additional Tags were added to job " + jobId);
            })
            .exceptionally(ex -> {
                System.err.println("Failed to add tags to job: " +
ex.getMessage());
                throw new RuntimeException(ex); // Propagate the exception
            });
    }

    // Setup the S3 bucket required for this scenario.
    /**
     * Creates an Amazon S3 bucket with the specified name.
     *
     * @param bucketName the name of the S3 bucket to create
     * @throws S3Exception if there is an error creating the bucket
     */
    public void createBucket(String bucketName) {
        try {
            S3Client s3Client = S3Client.builder()
                .region(Region.US_EAST_1)
```

```
        .build();

        S3Waiter s3Waiter = s3Client.waiter();
        CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
            .bucket(bucketName)
            .build();

        s3Client.createBucket(bucketRequest);
        HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
            .bucket(bucketName)
            .build();

        // Wait until the bucket is created and print out the response.
        WaiterResponse<HeadBucketResponse> waiterResponse =
s3Waiter.waitUntilBucketExists(bucketRequestWait);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println(bucketName + " is ready");

    } catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}

/**
 * Uploads a file to an Amazon S3 bucket asynchronously.
 *
 * @param bucketName the name of the S3 bucket to upload the file to
 * @param fileName the name of the file to be uploaded
 * @throws RuntimeException if an error occurs during the file upload
 */
public void populateBucket(String bucketName, String fileName) {
    // Define the path to the directory.
    Path filePath = Paths.get("src/main/resources/batch/",
fileName).toAbsolutePath();
    PutObjectRequest putOb = PutObjectRequest.builder()
        .bucket(bucketName)
        .key(fileName)
        .build();

    CompletableFuture<PutObjectResponse> future =
getS3AsyncClient().putObject(putOb, AsyncRequestBody.fromFile(filePath));
    future.whenComplete((result, ex) -> {
```

```

        if (ex != null) {
            System.err.println("Error uploading file: " + ex.getMessage());
        } else {
            System.out.println("Successfully placed " + fileName + " into
bucket " + bucketName);
        }
    }).join();
}

// Update the bucketName in CSV.
public void updateCSV(String newValue) {
    Path csvFilePath = Paths.get("src/main/resources/batch/job-
manifest.csv").toAbsolutePath();
    try {
        // Read all lines from the CSV file.
        List<String> lines = Files.readAllLines(csvFilePath);

        // Update the first value in each line.
        List<String> updatedLines = lines.stream()
            .map(line -> {
                String[] parts = line.split(",");
                parts[0] = newValue;
                return String.join(",", parts);
            })
            .collect(Collectors.toList());

        // Write the updated lines back to the CSV file
        Files.write(csvFilePath, updatedLines);
        System.out.println("CSV file updated successfully.");
    } catch (Exception e) {
        e.printStackTrace();
    }
}

/**
 * Deletes an object from an Amazon S3 bucket asynchronously.
 *
 * @param bucketName The name of the S3 bucket where the object is stored.
 * @param objectName The name of the object to be deleted.
 * @return A {@link CompletableFuture} that completes when the object has
been deleted,
 *         or throws a {@link RuntimeException} if an error occurs during the
deletion.

```

```
    */
    public CompletableFuture<Void> deleteBucketObjects(String bucketName, String
objectName) {
        ArrayList<ObjectIdentifier> toDelete = new ArrayList<>();
        toDelete.add(ObjectIdentifier.builder()
            .key(objectName)
            .build());

        DeleteObjectsRequest dor = DeleteObjectsRequest.builder()
            .bucket(bucketName)
            .delete(Delete.builder()
                .objects(toDelete).build())
            .build();

        return getS3AsyncClient().deleteObjects(dor)
            .thenAccept(result -> {
                System.out.println("The object was deleted!");
            })
            .exceptionally(ex -> {
                throw new RuntimeException("Error deleting object: " +
ex.getMessage(), ex);
            });
    }

    /**
     * Deletes a folder and all its contents asynchronously from an Amazon S3
bucket.
     *
     * @param bucketName the name of the S3 bucket containing the folder to be
deleted
     * @return a {@link CompletableFuture} that completes when the folder and its
contents have been deleted
     * @throws RuntimeException if any error occurs during the deletion process
     */
    public void deleteBucketFolderAsync(String bucketName) {
        String folderName = "reports/";
        ListObjectsV2Request request = ListObjectsV2Request.builder()
            .bucket(bucketName)
            .prefix(folderName)
            .build();

        CompletableFuture<ListObjectsV2Response> listObjectsFuture =
getS3AsyncClient().listObjectsV2(request);
        listObjectsFuture.thenCompose(response -> {
```

```

        List<CompletableFuture<DeleteObjectResponse>> deleteFutures =
response.contents().stream()
    .map(obj -> {
        DeleteObjectRequest deleteRequest =
DeleteObjectRequest.builder()
            .bucket(bucketName)
            .key(obj.key())
            .build();
        return getS3AsyncClient().deleteObject(deleteRequest)
            .thenApply(deleteResponse -> {
                System.out.println("Deleted object: " + obj.key());
                return deleteResponse;
            });
    })
    .collect(Collectors.toList());

    return CompletableFuture.allOf(deleteFutures.toArray(new
CompletableFuture[0]))
        .thenCompose(v -> {
            // Delete the folder.
            DeleteObjectRequest deleteRequest =
DeleteObjectRequest.builder()
                .bucket(bucketName)
                .key(folderName)
                .build();
            return getS3AsyncClient().deleteObject(deleteRequest)
                .thenApply(deleteResponse -> {
                    System.out.println("Deleted folder: " + folderName);
                    return deleteResponse;
                });
        });
    }).join();
}

/**
 * Deletes an Amazon S3 bucket.
 *
 * @param bucketName the name of the bucket to delete
 * @return a {@link CompletableFuture} that completes when the bucket has
been deleted, or exceptionally if there is an error
 * @throws RuntimeException if there is an error deleting the bucket
 */
public CompletableFuture<Void> deleteBucket(String bucketName) {
    S3AsyncClient s3Client = getS3AsyncClient();

```

```

        return s3Client.deleteBucket(DeleteBucketRequest.builder()
            .bucket(bucketName)
            .build())
            .thenAccept(deleteBucketResponse -> {
                System.out.println(bucketName + " was deleted");
            })
            .exceptionally(ex -> {
                // Handle the exception or rethrow it.
                throw new RuntimeException("Failed to delete bucket: " +
bucketName, ex);
            });
    }

    /**
     * Uploads a set of files to an Amazon S3 bucket.
     *
     * @param bucketName the name of the S3 bucket to upload the files to
     * @param fileNames an array of file names to be uploaded
     * @param actions an instance of {@link S3BatchActions} that provides the
implementation for the necessary S3 operations
     * @throws IOException if there's an error creating the text files or
uploading the files to the S3 bucket
     */
    public static void uploadFilesToBucket(String bucketName, String[] fileNames,
S3BatchActions actions) throws IOException {
        actions.updateCSV(bucketName);
        createTextFiles(fileNames);
        for (String fileName : fileNames) {
            actions.populateBucket(bucketName, fileName);
        }
        System.out.println("All files are placed in the S3 bucket " +
bucketName);
    }

    /**
     * Deletes the specified files from the given S3 bucket.
     *
     * @param bucketName the name of the S3 bucket
     * @param fileNames an array of file names to be deleted from the bucket
     * @param actions the S3BatchActions instance to be used for the file
deletion
     * @throws IOException if an I/O error occurs during the file deletion
     */

```

```
public void deleteFilesFromBucket(String bucketName, String[] fileNames,
S3BatchActions actions) throws IOException {
    for (String fileName : fileNames) {
        actions.deleteBucketObjects(bucketName, fileName)
            .thenRun(() -> System.out.println("Object deletion completed"))
            .exceptionally(ex -> {
                System.err.println("Error occurred: " + ex.getMessage());
                return null;
            });
    }
    System.out.println("All files have been deleted from the bucket " +
bucketName);
}

public static void createTextFiles(String[] fileNames) {
    String currentDirectory = System.getProperty("user.dir");
    String directoryPath = currentDirectory + "\\src\\main\\resources\\
\\batch";
    Path path = Paths.get(directoryPath);

    try {
        // Create the directory if it doesn't exist.
        if (Files.notExists(path)) {
            Files.createDirectories(path);
            System.out.println("Created directory: " + path.toString());
        } else {
            System.out.println("Directory already exists: " +
path.toString());
        }

        for (String fileName : fileNames) {
            // Check if the file is a .txt file.
            if (fileName.endsWith(".txt")) {
                // Define the path for the new file.
                Path filePath = path.resolve(fileName);
                System.out.println("Attempting to create file: " +
filePath.toString());

                // Create and write content to the new file.
                Files.write(filePath, "This is a test".getBytes());

                // Verify the file was created.
                if (Files.exists(filePath)) {
```



```
        System.out.println("Successfully created file: " +
filePath.toString());
    } else {
        System.out.println("Failed to create file: " +
filePath.toString());
    }
}

} catch (IOException e) {
    System.err.println("An error occurred: " + e.getMessage());
    e.printStackTrace();
}

}

public String getAccountId() {
    StsClient stsClient = StsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    GetCallerIdentityResponse callerIdentityResponse =
stsClient.getCallerIdentity();
    return callerIdentityResponse.account();
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
 - [CreateJob](#)
 - [DeleteJobTagging](#)
 - [DescribeJob](#)
 - [GetJobTagging](#)
 - [ListJobs](#)
 - [PutJobTagging](#)
 - [UpdateJobPriority](#)
 - [UpdateJobStatus](#)

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

了解 S3 批次基本概念案例。

```
class S3BatchWrapper:
    """Wrapper class for managing S3 Batch Operations."""

    def __init__(self, s3_client: Any, s3control_client: Any, sts_client: Any) ->
None:
    """
    Initializes the S3BatchWrapper with AWS service clients.

    :param s3_client: A Boto3 Amazon S3 client. This client provides low-
level
                        access to AWS S3 services.
    :param s3control_client: A Boto3 Amazon S3 Control client. This client
provides
                        low-level access to AWS S3 Control services.
    :param sts_client: A Boto3 AWS STS client. This client provides low-level
                        access to AWS STS services.
    """
    self.s3_client = s3_client
    self.s3control_client = s3control_client
    self.sts_client = sts_client
    # Get region from the client for bucket creation logic
    self.region_name = self.s3_client.meta.region_name

    def get_account_id(self) -> str:
        """
        Get AWS account ID.

        Returns:
            str: AWS account ID
        """
        return self.sts_client.get_caller_identity()["Account"]
```

```
def create_bucket(self, bucket_name: str) -> None:
    """
    Create an S3 bucket.

    Args:
        bucket_name (str): Name of the bucket to create

    Raises:
        ClientError: If bucket creation fails
    """
    try:
        if self.region_name and self.region_name != 'us-east-1':
            self.s3_client.create_bucket(
                Bucket=bucket_name,
                CreateBucketConfiguration={
                    'LocationConstraint': self.region_name
                }
            )
        else:
            self.s3_client.create_bucket(Bucket=bucket_name)
        print(f"Created bucket: {bucket_name}")
    except ClientError as e:
        print(f"Error creating bucket: {e}")
        raise

def upload_files_to_bucket(self, bucket_name: str, file_names: List[str]) -> str:
    """
    Upload files to S3 bucket including manifest file.

    Args:
        bucket_name (str): Target bucket name
        file_names (list): List of file names to upload

    Returns:
        str: ETag of the manifest file

    Raises:
        ClientError: If file upload fails
    """
    try:
        for file_name in file_names:
            if file_name != "job-manifest.csv":
```

```

        content = f"Content for {file_name}"
        self.s3_client.put_object(
            Bucket=bucket_name,
            Key=file_name,
            Body=content.encode('utf-8')
        )
        print(f"Uploaded {file_name} to {bucket_name}")

    manifest_content = ""
    for file_name in file_names:
        if file_name != "job-manifest.csv":
            manifest_content += f"{bucket_name},{file_name}\n"

    manifest_response = self.s3_client.put_object(
        Bucket=bucket_name,
        Key="job-manifest.csv",
        Body=manifest_content.encode('utf-8')
    )
    print(f"Uploaded manifest file to {bucket_name}")
    print(f"Manifest content:\n{manifest_content}")
    return manifest_response['ETag'].strip('')

except ClientError as e:
    print(f"Error uploading files: {e}")
    raise

def create_s3_batch_job(self, account_id: str, role_arn: str,
    manifest_location: str,
    report_bucket_name: str) -> str:
    """
    Create an S3 batch operation job.

    Args:
        account_id (str): AWS account ID
        role_arn (str): IAM role ARN for batch operations
        manifest_location (str): Location of the manifest file
        report_bucket_name (str): Bucket for job reports

    Returns:
        str: Job ID

    Raises:
        ClientError: If job creation fails
    """

```

```
try:
    bucket_name = manifest_location.split('::')[1].split('/')[0]
    manifest_key = 'job-manifest.csv'
    manifest_obj = self.s3_client.head_object(
        Bucket=bucket_name,
        Key=manifest_key
    )
    etag = manifest_obj['ETag'].strip('')

    response = self.s3control_client.create_job(
        AccountId=account_id,
        Operation={
            'S3PutObjectTagging': {
                'TagSet': [
                    {
                        'Key': 'BatchTag',
                        'Value': 'BatchValue'
                    },
                ],
            }
        },
        Report={
            'Bucket': report_bucket_name,
            'Format': 'Report_CSV_20180820',
            'Enabled': True,
            'Prefix': 'batch-op-reports',
            'ReportScope': 'AllTasks'
        },
        Manifest={
            'Spec': {
                'Format': 'S3BatchOperations_CSV_20180820',
                'Fields': ['Bucket', 'Key']
            },
            'Location': {
                'ObjectArn': manifest_location,
                'ETag': etag
            }
        },
        Priority=10,
        RoleArn=role_arn,
        Description='Batch job for tagging objects',
        ConfirmationRequired=True
    )
    job_id = response['JobId']
```

```
        print(f"The Job id is {job_id}")
        return job_id
    except ClientError as e:
        print(f"Error creating batch job: {e}")
        if 'Message' in str(e):
            print(f"Detailed error message: {e.response['Message']}")
        raise

    def check_job_failure_reasons(self, job_id: str, account_id: str) ->
    List[Dict[str, Any]]:
        """
        Check for any failure reasons of a batch job.

        Args:
            job_id (str): ID of the batch job
            account_id (str): AWS account ID

        Returns:
            list: List of failure reasons

        Raises:
            ClientError: If checking job failure reasons fails
        """
        try:
            response = self.s3control_client.describe_job(
                AccountId=account_id,
                JobId=job_id
            )
            if 'FailureReasons' in response['Job']:
                for reason in response['Job']['FailureReasons']:
                    print(f"- {reason}")
            return response['Job'].get('FailureReasons', [])
        except ClientError as e:
            print(f"Error checking job failure reasons: {e}")
            raise

    def wait_for_job_ready(self, job_id: str, account_id: str, desired_status:
    str = 'Ready') -> bool:
        """
        Wait for a job to reach the desired status.

        Args:
            job_id (str): ID of the batch job
            account_id (str): AWS account ID
```

```

        desired_status (str): Target status to wait for

Returns:
    bool: True if desired status is reached, False otherwise

Raises:
    ClientError: If checking job status fails
    """
    print(f"Waiting for job to become {desired_status}...")
    max_attempts = 60
    attempt = 0
    while attempt < max_attempts:
        try:
            response = self.s3control_client.describe_job(
                AccountId=account_id,
                JobId=job_id
            )
            current_status = response['Job']['Status']
            print(f"Current job status: {current_status}")
            if current_status == desired_status:
                return True
            if current_status == 'Suspended':
                print("Job is in Suspended state, can proceed with
activation")
                return True
            if current_status in ['Active', 'Failed', 'Cancelled',
'Complete']:
                print(f"Job is in {current_status} state, cannot reach
{desired_status} status")
                if 'FailureReasons' in response['Job']:
                    print("Failure reasons:")
                    for reason in response['Job']['FailureReasons']:
                        print(f"- {reason}")
                    return False

            time.sleep(20)
            attempt += 1
        except ClientError as e:
            print(f"Error checking job status: {e}")
            raise
    print(f"Timeout waiting for job to become {desired_status}")
    return False

def update_job_priority(self, job_id: str, account_id: str) -> None:

```

```
"""
Update the priority of a batch job and start it.

Args:
    job_id (str): ID of the batch job
    account_id (str): AWS account ID
"""
try:
    response = self.s3control_client.describe_job(
        AccountId=account_id,
        JobId=job_id
    )
    current_status = response['Job']['Status']
    print(f"Current job status: {current_status}")

    if current_status in ['Ready', 'Suspended']:
        self.s3control_client.update_job_priority(
            AccountId=account_id,
            JobId=job_id,
            Priority=60
        )
        print("The job priority was updated")

    try:
        self.s3control_client.update_job_status(
            AccountId=account_id,
            JobId=job_id,
            RequestedJobStatus='Ready'
        )
        print("Job activated successfully")
    except ClientError as activation_error:
        print(f"Note: Could not activate job automatically:
{activation_error}")
        print("Job priority was updated successfully. Job may need
manual activation in the console.")
    elif current_status in ['Active', 'Completing', 'Complete']:
        print(f"Job is in '{current_status}' state - priority cannot be
updated")

    if current_status == 'Completing':
        print("Job is finishing up and will complete soon.")
    elif current_status == 'Complete':
        print("Job has already completed successfully.")
    else:
        print("Job is currently running.")
```



```

        else:
            print(f"Job is in '{current_status}' state - priority update not
allowed")

    except ClientError as e:
        print(f"Error updating job priority: {e}")
        print("Continuing with the scenario...")
        return

def cancel_job(self, job_id: str, account_id: str) -> None:
    """
    Cancel an S3 batch job.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """
    try:
        response = self.s3control_client.describe_job(
            AccountId=account_id,
            JobId=job_id
        )
        current_status = response['Job']['Status']
        print(f"Current job status: {current_status}")

        if current_status in ['Ready', 'Suspended', 'Active']:
            self.s3control_client.update_job_status(
                AccountId=account_id,
                JobId=job_id,
                RequestedJobStatus='Cancelled'
            )
            print(f"Job {job_id} was successfully canceled.")
        elif current_status in ['Completing', 'Complete']:
            print(f"Job is in '{current_status}' state - cannot be
cancelled")

            if current_status == 'Completing':
                print("Job is finishing up and will complete soon.")
            elif current_status == 'Complete':
                print("Job has already completed successfully.")
        else:
            print(f"Job is in '{current_status}' state - cancel not allowed")
    except ClientError as e:
        print(f"Error canceling job: {e}")
        raise

```

```

def describe_job_details(self, job_id: str, account_id: str) -> None:
    """
    Describe detailed information about a batch job.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """
    try:
        response = self.s3control_client.describe_job(
            AccountId=account_id,
            JobId=job_id
        )
        job = response['Job']
        print(f"Job ID: {job['JobId']}")
        print(f"Description: {job.get('Description', 'N/A')}")
        print(f"Status: {job['Status']}")
        print(f"Role ARN: {job['RoleArn']}")
        print(f"Priority: {job['Priority']}")
        if 'ProgressSummary' in job:
            progress = job['ProgressSummary']
            print(f"Progress Summary:
Total={progress.get('TotalNumberOfTasks', 0)}, "
                  f"Succeeded={progress.get('NumberOfTasksSucceeded', 0)}, "
                  f"Failed={progress.get('NumberOfTasksFailed', 0)}")
    except ClientError as e:
        print(f"Error describing job: {e}")
        raise

def get_job_tags(self, job_id: str, account_id: str) -> None:
    """
    Get tags associated with a batch job.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """
    try:
        response = self.s3control_client.get_job_tagging(
            AccountId=account_id,
            JobId=job_id
        )
        tags = response.get('Tags', [])
    
```

```
        if tags:
            print(f"Tags for job {job_id}:")
            for tag in tags:
                print(f"  {tag['Key']}: {tag['Value']}")
        else:
            print(f"No tags found for job ID: {job_id}")
    except ClientError as e:
        print(f"Error getting job tags: {e}")
        raise

def put_job_tags(self, job_id: str, account_id: str) -> None:
    """
    Add tags to a batch job.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """
    try:
        self.s3control_client.put_job_tagging(
            AccountId=account_id,
            JobId=job_id,
            Tags=[
                {'Key': 'Environment', 'Value': 'Development'},
                {'Key': 'Team', 'Value': 'DataProcessing'}
            ]
        )
        print(f"Additional tags were added to job {job_id}")
    except ClientError as e:
        print(f"Error adding job tags: {e}")
        raise

def list_jobs(self, account_id: str) -> None:
    """
    List all batch jobs for the account.

    Args:
        account_id (str): AWS account ID
    """
    try:
        response = self.s3control_client.list_jobs(
            AccountId=account_id,
            JobStatuses=['Active', 'Complete', 'Cancelled', 'Failed', 'New',
                        'Paused', 'Pausing', 'Preparing', 'Ready', 'Suspended']
        )
```

```

    )
    jobs = response.get('Jobs', [])
    for job in jobs:
        print(f"The job id is {job['JobId']}")
        print(f"The job priority is {job['Priority']}")
except ClientError as e:
    print(f"Error listing jobs: {e}")
    raise

def delete_job_tags(self, job_id: str, account_id: str) -> None:
    """
    Delete all tags from a batch job.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """
    try:
        self.s3control_client.delete_job_tagging(
            AccountId=account_id,
            JobId=job_id
        )
        print(f"You have successfully deleted {job_id} tagging.")
    except ClientError as e:
        print(f"Error deleting job tags: {e}")
        raise

def cleanup_resources(self, bucket_name: str, file_names: List[str]) -> None:
    """
    Clean up all resources created during the scenario.

    Args:
        bucket_name (str): Name of the bucket to clean up
        file_names (list): List of files to delete

    Raises:
        ClientError: If cleanup fails
    """
    try:
        for file_name in file_names:
            self.s3_client.delete_object(Bucket=bucket_name, Key=file_name)
            print(f"Deleted {file_name}")

        response = self.s3_client.list_objects_v2(

```

```
        Bucket=bucket_name,
        Prefix='batch-op-reports/'
    )
    if 'Contents' in response:
        for obj in response['Contents']:
            self.s3_client.delete_object(
                Bucket=bucket_name,
                Key=obj['Key']
            )
            print(f"Deleted {obj['Key']}")

    self.s3_client.delete_bucket(Bucket=bucket_name)
    print(f"Deleted bucket {bucket_name}")
except ClientError as e:
    print(f"Error in cleanup: {e}")
    raise
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的下列主題。
 - [CreateJob](#)
 - [DeleteJobTagging](#)
 - [DescribeJob](#)
 - [GetJobTagging](#)
 - [ListJobs](#)
 - [PutJobTagging](#)
 - [UpdateJobPriority](#)
 - [UpdateJobStatus](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDKs 的 Amazon S3 控制動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 Amazon S3 控制動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

下列範例僅包含最常使用的動作。如需完整清單，請參閱《[Amazon S3 Control API 參考](#)》。

範例

- [CreateJob 搭配 AWS SDK 或 CLI 使用](#)
- [DeleteJobTagging 搭配 AWS SDK 使用](#)
- [DescribeJob 搭配 AWS SDK 或 CLI 使用](#)
- [GetJobTagging 搭配 AWS SDK 使用](#)
- [PutJobTagging 搭配 AWS SDK 使用](#)
- [UpdateJobPriority 搭配 AWS SDK 或 CLI 使用](#)
- [UpdateJobStatus 搭配 AWS SDK 或 CLI 使用](#)

CreateJob 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateJob。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

建立 Amazon S3 Batch Operations 任務

下列 create-job 範例會建立 Amazon S3 Batch Operations 任務，將物件標記為 confidential` in the bucket ``employee-records。

```
aws s3control create-job \  
  --account-id 123456789012 \  
  --operation '{"S3PutObjectTagging": { "TagSet": [{"Key":"confidential",  
  "Value":"true"}] }}' \  
  --report '{"Bucket":"arn:aws:s3:::employee-records-logs","Prefix":"batch-op-  
create-job",  
  "Format":"Report_CSV_20180820","Enabled":true,"ReportScope":"AllTasks"}' \  
  --manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":  
["Bucket","Key"]},"Location":{"ObjectArn":"arn:aws:s3:::employee-records-logs/  
inv-report/7a6a9be4-072c-407e-85a2-  
ec3e982f773e.csv","ETag":"69f52a4e9f797e987155d9c8f5880897"}}' \  
  --priority 42 \  

```

```
--role-arn arn:aws:iam::123456789012:role/S3BatchJobRole
```

輸出：

```
{
  "JobId": "93735294-df46-44d5-8638-6356f335324e"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CreateJob](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立非同步 S3 任務。

```
/**
 * Creates an asynchronous S3 job using the AWS Java SDK.
 *
 * @param accountId      the AWS account ID associated with the job
 * @param iamRoleArn     the ARN of the IAM role to be used for the job
 * @param manifestLocation the location of the job manifest file in S3
 * @param reportBucketName the name of the S3 bucket to store the job report
 * @param uuid           a unique identifier for the job
 * @return a CompletableFuture that represents the asynchronous creation of
 * the S3 job.
 *      The CompletableFuture will return the job ID if the job is created
 * successfully,
 *      or throw an exception if there is an error.
 */
public CompletableFuture<String> createS3JobAsync(String accountId, String
                                                    iamRoleArn,
                                                    String manifestLocation,
                                                    String reportBucketName, String uuid) {
```

```
String[] bucketName = new String[]{"");
String[] parts = reportBucketName.split("::");
if (parts.length > 1) {
    bucketName[0] = parts[1];
} else {
    System.out.println("The input string does not contain the expected
format.");
}

return CompletableFuture.supplyAsync(() -> getETag(bucketName[0], "job-
manifest.csv"))
    .thenCompose(eTag -> {
        ArrayList<S3Tag> tagSet = new ArrayList<>();
        S3Tag s3Tag = S3Tag.builder()
            .key("keyOne")
            .value("ValueOne")
            .build();
        S3Tag s3Tag2 = S3Tag.builder()
            .key("keyTwo")
            .value("ValueTwo")
            .build();
        tagSet.add(s3Tag);
        tagSet.add(s3Tag2);

        S3SetObjectTaggingOperation objectTaggingOperation =
S3SetObjectTaggingOperation.builder()
            .tagSet(tagSet)
            .build();

        JobOperation jobOperation = JobOperation.builder()
            .s3PutObjectTagging(objectTaggingOperation)
            .build();

        JobManifestLocation jobManifestLocation =
JobManifestLocation.builder()
            .objectArn(manifestLocation)
            .eTag(eTag)
            .build();

        JobManifestSpec manifestSpec = JobManifestSpec.builder()
            .fieldsWithStrings("Bucket", "Key")
            .format("S3BatchOperations_CSV_20180820")
            .build();
```



```
JobManifest jobManifest = JobManifest.builder()
    .spec(manifestSpec)
    .location(jobManifestLocation)
    .build();

JobReport jobReport = JobReport.builder()
    .bucket(reportBucketName)
    .prefix("reports")
    .format("Report_CSV_20180820")
    .enabled(true)
    .reportScope("AllTasks")
    .build();

CreateJobRequest jobRequest = CreateJobRequest.builder()
    .accountId(accountId)
    .description("Job created using the AWS Java SDK")
    .manifest(jobManifest)
    .operation(jobOperation)
    .report(jobReport)
    .priority(42)
    .roleArn(iamRoleArn)
    .clientRequestToken(uuid)
    .confirmationRequired(false)
    .build();

// Create the job asynchronously.
return getAsyncClient().createJob(jobRequest)
    .thenApply(CreateJobResponse::jobId);
})
.handle((jobId, ex) -> {
    if (ex != null) {
        Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
        if (cause instanceof S3ControlException) {
            throw new CompletionException(cause);
        } else {
            throw new RuntimeException(cause);
        }
    }
    return jobId;
});
});
}
```

建立合規保留任務。

```
/**
 * Creates a compliance retention job in Amazon S3 Control.
 * <p>
 * A compliance retention job in Amazon S3 Control is a feature that allows
you to
 * set a retention period for objects stored in an S3 bucket.
 * This feature is particularly useful for organizations that need to comply
with
 * regulatory requirements or internal policies that mandate the retention of
data for
 * a specific duration.
 *
 * @param s3ControlClient The S3ControlClient instance to use for the API
call.
 * @return The job ID of the created compliance retention job.
 */
public static String createComplianceRetentionJob(final S3ControlClient
s3ControlClient, String roleArn, String bucketName, String accountId) {
    final String manifestObjectArn = "arn:aws:s3:::amzn-s3-demo-manifest-
bucket/compliance-objects-manifest.csv";
    final String manifestObjectVersionId = "your-object-version-Id";

    Instant jan2025 = Instant.parse("2025-01-01T00:00:00Z");
    JobOperation jobOperation = JobOperation.builder()
        .s3PutObjectRetention(S3SetObjectRetentionOperation.builder()
            .retention(S3Retention.builder()
                .mode(S3ObjectLockRetentionMode.COMPLIANCE)
                .retainUntilDate(jan2025)
                .build())
            .build())
        .build();

    JobManifestLocation manifestLocation = JobManifestLocation.builder()
        .objectArn(manifestObjectArn)
        .eTag(manifestObjectVersionId)
        .build();

    JobManifestSpec manifestSpec = JobManifestSpec.builder()
        .fieldsWithStrings("Bucket", "Key")
        .format("S3BatchOperations_CSV_20180820")
        .build();
```

```
JobManifest manifestToPublicApi = JobManifest.builder()
    .location(manifestLocation)
    .spec(manifestSpec)
    .build();

// Report details.
final String jobReportBucketArn = "arn:aws:s3:::" + bucketName;
final String jobReportPrefix = "reports/compliance-objects-bops";

JobReport jobReport = JobReport.builder()
    .enabled(true)
    .reportScope(JobReportScope.ALL_TASKS)
    .bucket(jobReportBucketArn)
    .prefix(jobReportPrefix)
    .format(JobReportFormat.REPORT_CSV_20180820)
    .build();

final Boolean requiresConfirmation = true;
final int priority = 10;
CreateJobRequest request = CreateJobRequest.builder()
    .accountId(accountId)
    .description("Set compliance retain-until to 1 Jan 2025")
    .manifest(manifestToPublicApi)
    .operation(jobOperation)
    .priority(priority)
    .roleArn(roleArn)
    .report(jobReport)
    .confirmationRequired(requiresConfirmation)
    .build();

// Create the job and get the result.
CreateJobResponse result = s3ControlClient.createJob(request);
return result.jobId();
}
```

建立法務保存任務。

```
/**
 * Creates a compliance retention job in Amazon S3 Control.
 * <p>
```

```

    * A compliance retention job in Amazon S3 Control is a feature that allows
    you to
    * set a retention period for objects stored in an S3 bucket.
    * This feature is particularly useful for organizations that need to comply
    with
    * regulatory requirements or internal policies that mandate the retention of
    data for
    * a specific duration.
    *
    * @param s3ControlClient The S3ControlClient instance to use for the API
    call.
    * @return The job ID of the created compliance retention job.
    */
    public static String createComplianceRetentionJob(final S3ControlClient
    s3ControlClient, String roleArn, String bucketName, String accountId) {
        final String manifestObjectArn = "arn:aws:s3:::amzn-s3-demo-manifest-
    bucket/compliance-objects-manifest.csv";
        final String manifestObjectVersionId = "your-object-version-Id";

        Instant jan2025 = Instant.parse("2025-01-01T00:00:00Z");
        JobOperation jobOperation = JobOperation.builder()
            .s3PutObjectRetention(S3SetObjectRetentionOperation.builder()
                .retention(S3Retention.builder()
                    .mode(S3ObjectLockRetentionMode.COMPLIANCE)
                    .retainUntilDate(jan2025)
                    .build())
                .build())
            .build();

        JobManifestLocation manifestLocation = JobManifestLocation.builder()
            .objectArn(manifestObjectArn)
            .eTag(manifestObjectVersionId)
            .build();

        JobManifestSpec manifestSpec = JobManifestSpec.builder()
            .fieldsWithStrings("Bucket", "Key")
            .format("S3BatchOperations_CSV_20180820")
            .build();

        JobManifest manifestToPublicApi = JobManifest.builder()
            .location(manifestLocation)
            .spec(manifestSpec)
            .build();

```

```

        // Report details.
        final String jobReportBucketArn = "arn:aws:s3:::" + bucketName;
        final String jobReportPrefix = "reports/compliance-objects-bops";

        JobReport jobReport = JobReport.builder()
            .enabled(true)
            .reportScope(JobReportScope.ALL_TASKS)
            .bucket(jobReportBucketArn)
            .prefix(jobReportPrefix)
            .format(JobReportFormat.REPORT_CSV_20180820)
            .build();

        final Boolean requiresConfirmation = true;
        final int priority = 10;
        CreateJobRequest request = CreateJobRequest.builder()
            .accountId(accountId)
            .description("Set compliance retain-until to 1 Jan 2025")
            .manifest(manifestToPublicApi)
            .operation(jobOperation)
            .priority(priority)
            .roleArn(roleArn)
            .report(jobReport)
            .confirmationRequired(requiresConfirmation)
            .build();

        // Create the job and get the result.
        CreateJobResponse result = s3ControlClient.createJob(request);
        return result.jobId();
    }

```

建立新的控管保留任務。

```

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CreateGovernanceRetentionJob {

```

```
public static void main(String[]args) throws ParseException {
    final String usage = ""

        Usage:
            <manifestObjectArn> <jobReportBucketArn> <roleArn> <accountId>
<manifestObjectVersionId>

        Where:
            manifestObjectArn - The Amazon Resource Name (ARN) of the S3
object that contains the manifest file for the governance objects.\s
            bucketName - The ARN of the S3 bucket where the job report will
be stored.
            roleArn - The ARN of the IAM role that will be used to perform
the governance retention operation.
            accountId - Your AWS account Id.
            manifestObjectVersionId = A unique value that is used as the
`eTag` property of the `JobManifestLocation` object.
        """;

    if (args.length != 4) {
        System.out.println(usage);
        return;
    }

    String manifestObjectArn = args[0];
    String jobReportBucketArn = args[1];
    String roleArn = args[2];
    String accountId = args[3];
    String manifestObjectVersionId = args[4];

    S3ControlClient s3ControlClient = S3ControlClient.create();
    createGovernanceRetentionJob(s3ControlClient, manifestObjectArn,
jobReportBucketArn, roleArn, accountId, manifestObjectVersionId);
}

public static String createGovernanceRetentionJob(final S3ControlClient
s3ControlClient, String manifestObjectArn, String jobReportBucketArn, String
roleArn, String accountId, String manifestObjectVersionId) throws ParseException
{
    final JobManifestLocation manifestLocation =
JobManifestLocation.builder()
        .objectArn(manifestObjectArn)
        .eTag(manifestObjectVersionId)
```

```
.build();

final JobManifestSpec manifestSpec = JobManifestSpec.builder()
    .format(JobManifestFormat.S3_BATCH_OPERATIONS_CSV_20180820)
    .fields(Arrays.asList(JobManifestFieldName.BUCKET,
JobManifestFieldName.KEY))
    .build();

final JobManifest manifestToPublicApi = JobManifest.builder()
    .location(manifestLocation)
    .spec(manifestSpec)
    .build();

final String jobReportPrefix = "reports/governance-objects";
final JobReport jobReport = JobReport.builder()
    .enabled(true)
    .reportScope(JobReportScope.ALL_TASKS)
    .bucket(jobReportBucketArn)
    .prefix(jobReportPrefix)
    .format(JobReportFormat.REPORT_CSV_20180820)
    .build();

final SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");
final Date jan30th = format.parse("30/01/2025");

final S3SetObjectRetentionOperation s3SetObjectRetentionOperation =
S3SetObjectRetentionOperation.builder()
    .retention(S3Retention.builder()
        .mode(S3ObjectLockRetentionMode.GOVERNANCE)
        .retainUntilDate(jan30th.toInstant())
        .build())
    .build();

final JobOperation jobOperation = JobOperation.builder()
    .s3PutObjectRetention(s3SetObjectRetentionOperation)
    .build();

final Boolean requiresConfirmation = true;
final int priority = 10;

final CreateJobRequest request = CreateJobRequest.builder()
    .accountId(accountId)
    .description("Put governance retention")
    .manifest(manifestToPublicApi)
```

```
        .operation(jobOperation)
        .priority(priority)
        .roleArn(roleArn)
        .report(jobReport)
        .confirmationRequired(requiresConfirmation)
        .build();

    final CreateJobResponse result = s3ControlClient.createJob(request);
    return result.jobId();
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CreateJob](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
def create_s3_batch_job(self, account_id: str, role_arn: str,
manifest_location: str,
                        report_bucket_name: str) -> str:
    """
    Create an S3 batch operation job.

    Args:
        account_id (str): AWS account ID
        role_arn (str): IAM role ARN for batch operations
        manifest_location (str): Location of the manifest file
        report_bucket_name (str): Bucket for job reports

    Returns:
        str: Job ID

    Raises:
        ClientError: If job creation fails
```



```
"""
try:
    bucket_name = manifest_location.split(':::')[1].split('/')[0]
    manifest_key = 'job-manifest.csv'
    manifest_obj = self.s3_client.head_object(
        Bucket=bucket_name,
        Key=manifest_key
    )
    etag = manifest_obj['ETag'].strip('""')

    response = self.s3control_client.create_job(
        AccountId=account_id,
        Operation={
            'S3PutObjectTagging': {
                'TagSet': [
                    {
                        'Key': 'BatchTag',
                        'Value': 'BatchValue'
                    },
                ]
            }
        },
        Report={
            'Bucket': report_bucket_name,
            'Format': 'Report_CSV_20180820',
            'Enabled': True,
            'Prefix': 'batch-op-reports',
            'ReportScope': 'AllTasks'
        },
        Manifest={
            'Spec': {
                'Format': 'S3BatchOperations_CSV_20180820',
                'Fields': ['Bucket', 'Key']
            },
            'Location': {
                'ObjectArn': manifest_location,
                'ETag': etag
            }
        },
        Priority=10,
        RoleArn=role_arn,
        Description='Batch job for tagging objects',
        ConfirmationRequired=True
    )
```

```

        job_id = response['JobId']
        print(f"The Job id is {job_id}")
        return job_id
    except ClientError as e:
        print(f"Error creating batch job: {e}")
        if 'Message' in str(e):
            print(f"Detailed error message: {e.response['Message']}")
        raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [CreateJob](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
  " iv_manifest_arn  = 'arn:aws:s3::my-bucket/job-manifest.csv'
  " iv_manifest_etag = 'abc123def456'
  " iv_report_bucket = 'arn:aws:s3::my-report-bucket'
  DATA(lo_result) = lo_s3c->createjob(
    iv_accountid      = iv_account_id
    iv_rolearn        = iv_role_arn
    iv_confirmationrequired = abap_true
    iv_priority        = 10
    iv_description     = 'Batch job for tagging objects'
    io_operation       = NEW /aws1/cl_s3cjoboperation(
      io_s3putobjecttagging = NEW /aws1/cl_s3cs3setobjecttagop(
        it_tagset = VALUE /aws1/cl_s3cs3tag=>tt_s3tagset(
          ( NEW /aws1/cl_s3cs3tag(
            iv_key   = 'BatchTag'
            iv_value = 'BatchValue' ) )
        )
      )
    )
    io_manifest      = NEW /aws1/cl_s3cjobmanifest(

```

```

        io_spec      = NEW /aws1/cl_s3cjobmanifestspec(
            iv_format = 'S3BatchOperations_CSV_20180820'
            it_fields = VALUE /aws1/
cl_s3cjobmanifestfield00=>tt_jobmanifestfieldlist(
            ( NEW /aws1/cl_s3cjobmanifestfield00( 'Bucket' ) )
            ( NEW /aws1/cl_s3cjobmanifestfield00( 'Key' ) )
        )
    )
    io_location = NEW /aws1/cl_s3cjobmanifestloc(
        iv_objectarn = iv_manifest_arn
        iv_etag      = iv_manifest_etag
    )
)
io_report      = NEW /aws1/cl_s3cjobreport(
    iv_bucket    = iv_report_bucket
    iv_format     = 'Report_CSV_20180820'
    iv_enabled    = abap_true
    iv_prefix     = 'batch-op-reports'
    iv_reportscope = 'AllTasks'
)
).
ov_job_id = lo_result->get_jobid( ).
MESSAGE |S3 Batch Operations job created: { ov_job_id }| TYPE 'I'.
CATCH /aws1/cx_s3cbadrequestex INTO DATA(lo_ex_bad).
MESSAGE lo_ex_bad->get_text( ) TYPE 'I'.
RAISE EXCEPTION TYPE /aws1/cx_rt_generic
    EXPORTING previous = lo_ex_bad.
CATCH /aws1/cx_s3cclientexc INTO DATA(lo_ex_cli).
MESSAGE lo_ex_cli->get_text( ) TYPE 'I'.
RAISE EXCEPTION TYPE /aws1/cx_rt_generic
    EXPORTING previous = lo_ex_cli.
CATCH /aws1/cx_s3cserverexc INTO DATA(lo_ex_srv).
MESSAGE lo_ex_srv->get_text( ) TYPE 'I'.
RAISE EXCEPTION TYPE /aws1/cx_rt_generic
    EXPORTING previous = lo_ex_srv.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [CreateJob](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteJobTagging 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 DeleteJobTagging。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously deletes the tags associated with a specific batch job.
 *
 * @param jobId      The ID of the batch job whose tags should be deleted.
 * @param accountId  The ID of the account associated with the batch job.
 * @return A CompletableFuture that completes when the job tags have been
 * successfully deleted, or an exception is thrown if the deletion fails.
 */
public CompletableFuture<Void> deleteBatchJobTagsAsync(String jobId, String
accountId) {
    DeleteJobTaggingRequest jobTaggingRequest =
DeleteJobTaggingRequest.builder()
        .accountId(accountId)
        .jobId(jobId)
        .build();

    return asyncClient.deleteJobTagging(jobTaggingRequest)
        .thenAccept(response -> {
            System.out.println("You have successfully deleted " + jobId + "
tagging.");
        })
        .exceptionally(ex -> {
```

```
        System.err.println("Failed to delete job tags: " +
ex.getMessage());
        throw new RuntimeException(ex);
    });
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteJobTagging](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
def delete_job_tags(self, job_id: str, account_id: str) -> None:
    """
    Delete all tags from a batch job.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """
    try:
        self.s3control_client.delete_job_tagging(
            AccountId=account_id,
            JobId=job_id
        )
        print(f"You have successfully deleted {job_id} tagging.")
    except ClientError as e:
        print(f"Error deleting job tags: {e}")
        raise
```

- 如需 API 詳細資訊，請參閱《適用於 AWS Python (Boto3) 的 SDK API 參考》中的 [DeleteJobTagging](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    lo_s3c->deletejobtagging(  
        iv_accountid = iv_account_id  
        iv_jobid      = iv_job_id  
    ).  
    MESSAGE |Tags deleted from job { iv_job_id }| TYPE 'I'.  
CATCH /aws1/cx_s3cnotfoundexception INTO DATA(lo_ex_nf).  
    MESSAGE lo_ex_nf->get_text( ) TYPE 'I'.  
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic  
        EXPORTING previous = lo_ex_nf.  
CATCH /aws1/cx_s3cclientexc INTO DATA(lo_ex_cli).  
    MESSAGE lo_ex_cli->get_text( ) TYPE 'I'.  
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic  
        EXPORTING previous = lo_ex_cli.  
CATCH /aws1/cx_s3cserverexc INTO DATA(lo_ex_srv).  
    MESSAGE lo_ex_srv->get_text( ) TYPE 'I'.  
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic  
        EXPORTING previous = lo_ex_srv.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [DeleteJobTagging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DescribeJob 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DescribeJob。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

描述 Amazon S3 Batch Operations 任務

下列 `describe-job` 提供指定批次操作任務的組態參數和狀態。

```
aws s3control describe-job \  
  --account-id 123456789012 \  
  --job-id 93735294-df46-44d5-8638-6356f335324e
```

輸出：

```
{  
  "Job": {  
    "TerminationDate": "2019-10-03T21:49:53.944Z",  
    "JobId": "93735294-df46-44d5-8638-6356f335324e",  
    "FailureReasons": [],  
    "Manifest": {  
      "Spec": {  
        "Fields": [  
          "Bucket",  
          "Key"  
        ],  
        "Format": "S3BatchOperations_CSV_20180820"  
      },  
      "Location": {  
        "ETag": "69f52a4e9f797e987155d9c8f5880897",  
        "ObjectArn": "arn:aws:s3:::employee-records-logs/inv-report/7a6a9be4-072c-407e-85a2-ec3e982f773e.csv"  
      }  
    },  
    "Operation": {  
      "S3PutObjectTagging": {  
        "TagSet": [  
          {  

```

```

        "Value": "true",
        "Key": "confidential"
    }
]
},
"RoleArn": "arn:aws:iam::123456789012:role/S3BatchJobRole",
"ProgressSummary": {
    "TotalNumberOfTasks": 8,
    "NumberOfTasksFailed": 0,
    "NumberOfTasksSucceeded": 8
},
"Priority": 42,
"Report": {
    "ReportScope": "AllTasks",
    "Format": "Report_CSV_20180820",
    "Enabled": true,
    "Prefix": "batch-op-create-job",
    "Bucket": "arn:aws:s3:::employee-records-logs"
},
"JobArn": "arn:aws:s3:us-west-2:123456789012:job/93735294-
df46-44d5-8638-6356f335324e",
"CreationTime": "2019-10-03T21:48:48.048Z",
"Status": "Complete"
}
}

```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DescribeJob](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

/**
 * Asynchronously describes the specified job.
 *

```



```
* @param jobId      the ID of the job to describe
* @param accountId  the ID of the AWS account associated with the job
* @return a {@link CompletableFuture} that completes when the job
description is available
* @throws RuntimeException if an error occurs while describing the job
*/
public CompletableFuture<Void> describeJobAsync(String jobId, String
accountId) {
    DescribeJobRequest jobRequest = DescribeJobRequest.builder()
        .jobId(jobId)
        .accountId(accountId)
        .build();

    return getAsyncClient().describeJob(jobRequest)
        .thenAccept(response -> {
            System.out.println("Job ID: " + response.job().jobId());
            System.out.println("Description: " +
response.job().description());
            System.out.println("Status: " + response.job().statusAsString());
            System.out.println("Role ARN: " + response.job().roleArn());
            System.out.println("Priority: " + response.job().priority());
            System.out.println("Progress Summary: " +
response.job().progressSummary());

            // Print out details about the job manifest.
            JobManifest manifest = response.job().manifest();
            System.out.println("Manifest Location: " +
manifest.location().objectArn());
            System.out.println("Manifest ETag: " +
manifest.location().eTag());

            // Print out details about the job operation.
            JobOperation operation = response.job().operation();
            if (operation.s3PutObjectTagging() != null) {
                System.out.println("Operation: S3 Put Object Tagging");
                System.out.println("Tag Set: " +
operation.s3PutObjectTagging().tagSet());
            }

            // Print out details about the job report.
            JobReport report = response.job().report();
            System.out.println("Report Bucket: " + report.bucket());
            System.out.println("Report Prefix: " + report.prefix());
            System.out.println("Report Format: " + report.format());
```

```
        System.out.println("Report Enabled: " + report.enabled());
        System.out.println("Report Scope: " +
report.reportScopeAsString());
    })
    .exceptionally(ex -> {
        System.err.println("Failed to describe job: " + ex.getMessage());
        throw new RuntimeException(ex);
    });
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DescribeJob](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
def describe_job_details(self, job_id: str, account_id: str) -> None:
    """
    Describe detailed information about a batch job.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """
    try:
        response = self.s3control_client.describe_job(
            AccountId=account_id,
            JobId=job_id
        )
        job = response['Job']
        print(f"Job ID: {job['JobId']}")
        print(f>Description: {job.get('Description', 'N/A')}")
        print(f>Status: {job['Status']}")
        print(f"Role ARN: {job['RoleArn']}")
        print(f>Priority: {job['Priority']}")
```

```

        if 'ProgressSummary' in job:
            progress = job['ProgressSummary']
            print(f"Progress Summary:
Total={progress.get('TotalNumberOfTasks', 0)}, "
                  f"Succeeded={progress.get('NumberOfTasksSucceeded', 0)}, "
                  f"Failed={progress.get('NumberOfTasksFailed', 0)}")
        except ClientError as e:
            print(f"Error describing job: {e}")
            raise

```

- 如需 API 的詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS 開發套件 API 參考》中的 [DescribeJob](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    oo_result = lo_s3c->describejob(          " oo_result is returned for
testing purposes.
    iv_accountid = iv_account_id
    iv_jobid      = iv_job_id
    ).
    DATA(lo_job) = oo_result->get_job( ).
    DATA(lv_status) = lo_job->get_status( ).
    DATA(lv_priority) = lo_job->get_priority( ).
    DATA(lo_progress) = lo_job->get_progresssummary( ).
    IF lo_progress IS NOT INITIAL.
        MESSAGE |Job { iv_job_id }: status={ lv_status },
priority={ lv_priority }, | &&
                |total={ lo_progress->get_totalnumberoftasks( ) }, | &&
                |succeeded={ lo_progress->get_numberoftaskssucceeded( ) }, | &&
                |failed={ lo_progress->get_numberoftasksfailed( ) }| TYPE 'I'.
    ELSE.

```

```
        MESSAGE |Job { iv_job_id }: status={ lv_status },
        priority={ lv_priority }| TYPE 'I'.
    ENDIF.
    CATCH /aws1/cx_s3cnotfoundexception INTO DATA(lo_ex_nf).
    MESSAGE lo_ex_nf->get_text( ) TYPE 'I'.
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic
        EXPORTING previous = lo_ex_nf.
    CATCH /aws1/cx_s3cclientexc INTO DATA(lo_ex_cli).
    MESSAGE lo_ex_cli->get_text( ) TYPE 'I'.
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic
        EXPORTING previous = lo_ex_cli.
    CATCH /aws1/cx_s3cserverexc INTO DATA(lo_ex_srv).
    MESSAGE lo_ex_srv->get_text( ) TYPE 'I'.
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic
        EXPORTING previous = lo_ex_srv.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [DescribeJob](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetJobTagging 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 GetJobTagging。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously retrieves the tags associated with a specific job in an AWS
 * account.
 *
 * @param jobId      the ID of the job for which to retrieve the tags
 * @param accountId  the ID of the AWS account associated with the job
 * @return a {@link CompletableFuture} that completes when the job tags have
 *         been retrieved, or with an exception if the operation fails
 * @throws RuntimeException if an error occurs while retrieving the job tags
 */
public CompletableFuture<Void> getJobTagsAsync(String jobId, String
accountId) {
    GetJobTaggingRequest request = GetJobTaggingRequest.builder()
        .jobId(jobId)
        .accountId(accountId)
        .build();

    return asyncClient.getJobTagging(request)
        .thenAccept(response -> {
            List<S3Tag> tags = response.tags();
            if (tags.isEmpty()) {
                System.out.println("No tags found for job ID: " + jobId);
            } else {
                for (S3Tag tag : tags) {
                    System.out.println("Tag key is: " + tag.key());
                    System.out.println("Tag value is: " + tag.value());
                }
            }
        })
        .exceptionally(ex -> {
            System.err.println("Failed to get job tags: " + ex.getMessage());
            throw new RuntimeException(ex); // Propagate the exception
        });
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetJobTagging](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
def get_job_tags(self, job_id: str, account_id: str) -> None:
    """
    Get tags associated with a batch job.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """
    try:
        response = self.s3control_client.get_job_tagging(
            AccountId=account_id,
            JobId=job_id
        )
        tags = response.get('Tags', [])
        if tags:
            print(f"Tags for job {job_id}:")
            for tag in tags:
                print(f"  {tag['Key']}: {tag['Value']}")
        else:
            print(f"No tags found for job ID: {job_id}")
    except ClientError as e:
        print(f"Error getting job tags: {e}")
        raise
```

- 如需 API 詳細資訊，請參閱《適用於 AWS Python (Boto3) 的 SDK API 參考》中的 [GetJobTagging](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.
    oo_result = lo_s3c->getjobtagging(          " oo_result is returned for
testing purposes.
    iv_accountid = iv_account_id
    iv_jobid      = iv_job_id
    ).
    DATA(lt_tags) = oo_result->get_tags( ).
    MESSAGE |Retrieved { lines( lt_tags ) } tag(s) for job { iv_job_id }|
TYPE 'I'.
CATCH /aws1/cx_s3cnotfoundexception INTO DATA(lo_ex_nf).
    MESSAGE lo_ex_nf->get_text( ) TYPE 'I'.
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic
        EXPORTING previous = lo_ex_nf.
CATCH /aws1/cx_s3cclientexc INTO DATA(lo_ex_cli).
    MESSAGE lo_ex_cli->get_text( ) TYPE 'I'.
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic
        EXPORTING previous = lo_ex_cli.
CATCH /aws1/cx_s3cserverexc INTO DATA(lo_ex_srv).
    MESSAGE lo_ex_srv->get_text( ) TYPE 'I'.
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic
        EXPORTING previous = lo_ex_srv.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetJobTagging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutJobTagging 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 PutJobTagging。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously adds tags to a job in the system.
 *
 * @param jobId      the ID of the job to add tags to
 * @param accountId the account ID associated with the job
 * @return a CompletableFuture that completes when the tagging operation is
 * finished
 */
public CompletableFuture<Void> putJobTaggingAsync(String jobId, String
accountId) {
    S3Tag departmentTag = S3Tag.builder()
        .key("department")
        .value("Marketing")
        .build();

    S3Tag fiscalYearTag = S3Tag.builder()
        .key("FiscalYear")
        .value("2020")
        .build();

    PutJobTaggingRequest putJobTaggingRequest =
    PutJobTaggingRequest.builder()
        .jobId(jobId)
```



```
        .accountId(accountId)
        .tags(departmentTag, fiscalYearTag)
        .build();

return asyncClient.putJobTagging(putJobTaggingRequest)
    .thenRun(() -> {
        System.out.println("Additional Tags were added to job " + jobId);
    })
    .exceptionally(ex -> {
        System.err.println("Failed to add tags to job: " +
ex.getMessage());
        throw new RuntimeException(ex); // Propagate the exception
    });
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutJobTagging](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
def put_job_tags(self, job_id: str, account_id: str) -> None:
    """
    Add tags to a batch job.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """
    try:
        self.s3control_client.put_job_tagging(
            AccountId=account_id,
            JobId=job_id,
            Tags=[
                {'Key': 'Environment', 'Value': 'Development'},
```

```

        {'Key': 'Team', 'Value': 'DataProcessing'}
    ]
)
print(f"Additional tags were added to job {job_id}")
except ClientError as e:
    print(f"Error adding job tags: {e}")
    raise

```

- 如需 API 詳細資訊，請參閱《適用於 AWS Python (Boto3) 的 SDK API 參考》中的 [PutJobTagging](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    lo_s3c->putjobtagging(
        iv_accountid = iv_account_id
        iv_jobid      = iv_job_id
        it_tags       = VALUE /aws1/cl_s3cs3tag=>tt_s3tagset(
            ( NEW /aws1/cl_s3cs3tag(
                iv_key   = 'Environment'
                iv_value = 'Development' ) )
            ( NEW /aws1/cl_s3cs3tag(
                iv_key   = 'Team'
                iv_value = 'DataProcessing' ) )
        )
    ).
    MESSAGE |Tags added to job { iv_job_id }| TYPE 'I'.
CATCH /aws1/cx_s3cnotfoundexception INTO DATA(lo_ex_nf).
    MESSAGE lo_ex_nf->get_text( ) TYPE 'I'.
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic
        EXPORTING previous = lo_ex_nf.
CATCH /aws1/cx_s3ctoomanytagsex INTO DATA(lo_ex_tags).

```

```
MESSAGE lo_ex_tags->get_text( ) TYPE 'I'.
RAISE EXCEPTION TYPE /aws1/cx_rt_generic
EXPORTING previous = lo_ex_tags.
CATCH /aws1/cx_s3cclientexc INTO DATA(lo_ex_cli).
MESSAGE lo_ex_cli->get_text( ) TYPE 'I'.
RAISE EXCEPTION TYPE /aws1/cx_rt_generic
EXPORTING previous = lo_ex_cli.
CATCH /aws1/cx_s3cserverexc INTO DATA(lo_ex_srv).
MESSAGE lo_ex_srv->get_text( ) TYPE 'I'.
RAISE EXCEPTION TYPE /aws1/cx_rt_generic
EXPORTING previous = lo_ex_srv.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutJobTagging](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UpdateJobPriority 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 UpdateJobPriority。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

更新 Amazon S3 Batch Operations 任務的任務優先順序

下列 update-job-priority 範例會將指定的任務更新為新的優先順序。

```
aws s3control update-job-priority \
  --account-id 123456789012 \
  --job-id 8d9a18fe-c303-4d39-8ccc-860d372da386 \
  --priority 52
```

輸出：

```
{
  "JobId": "8d9a18fe-c303-4d39-8ccc-860d372da386",
  "Priority": 52
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [UpdateJobPriority](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Updates the priority of a job asynchronously.
 *
 * @param jobId      the ID of the job to update
 * @param accountId  the ID of the account associated with the job
 * @return a {@link CompletableFuture} that represents the asynchronous
 * operation, which completes when the job priority has been updated or an error
 * has occurred
 */
public CompletableFuture<Void> updateJobPriorityAsync(String jobId, String
accountId) {
    UpdateJobPriorityRequest priorityRequest =
UpdateJobPriorityRequest.builder()
        .accountId(accountId)
        .jobId(jobId)
        .priority(60)
        .build();

    CompletableFuture<Void> future = new CompletableFuture<>();
    getAsyncClient().updateJobPriority(priorityRequest)
        .thenAccept(response -> {
            System.out.println("The job priority was updated");
        });
}
```

```
        future.complete(null); // Complete the CompletableFuture on
successful execution
    })
    .exceptionally(ex -> {
        System.err.println("Failed to update job priority: " +
ex.getMessage());
        future.completeExceptionally(ex); // Complete the
CompletableFuture exceptionally on error
        return null; // Return null to handle the exception
    });

    return future;
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [UpdateJobPriority](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
def update_job_priority(self, job_id: str, account_id: str) -> None:
    """
    Update the priority of a batch job and start it.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """
    try:
        response = self.s3control_client.describe_job(
            AccountId=account_id,
            JobId=job_id
        )
        current_status = response['Job']['Status']
        print(f"Current job status: {current_status}")
```

```
if current_status in ['Ready', 'Suspended']:
    self.s3control_client.update_job_priority(
        AccountId=account_id,
        JobId=job_id,
        Priority=60
    )
    print("The job priority was updated")

    try:
        self.s3control_client.update_job_status(
            AccountId=account_id,
            JobId=job_id,
            RequestedJobStatus='Ready'
        )
        print("Job activated successfully")
    except ClientError as activation_error:
        print(f"Note: Could not activate job automatically:
{activation_error}")
        print("Job priority was updated successfully. Job may need
manual activation in the console.")
    elif current_status in ['Active', 'Completing', 'Complete']:
        print(f"Job is in '{current_status}' state - priority cannot be
updated")

        if current_status == 'Completing':
            print("Job is finishing up and will complete soon.")
        elif current_status == 'Complete':
            print("Job has already completed successfully.")
        else:
            print("Job is currently running.")
    else:
        print(f"Job is in '{current_status}' state - priority update not
allowed")

except ClientError as e:
    print(f"Error updating job priority: {e}")
    print("Continuing with the scenario...")
    return
```

- 如需 API 詳細資訊，請參閱《適用於 AWS Python (Boto3) 的 SDK API 參考》中的 [UpdateJobPriority](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_s3c->updatejobpriority( " oo_result is returned for  
testing purposes.  
    iv_accountid = iv_account_id  
    iv_jobid      = iv_job_id  
    iv_priority   = 60  
    ).  
    MESSAGE |Job { oo_result->get_jobid( ) } priority updated to { oo_result-  
>get_priority( ) }| TYPE 'I'.  
    CATCH /aws1/cx_s3cnotfoundexception INTO DATA(lo_ex_nf).  
    MESSAGE lo_ex_nf->get_text( ) TYPE 'I'.  
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic  
    EXPORTING previous = lo_ex_nf.  
    CATCH /aws1/cx_s3cbadrequestex INTO DATA(lo_ex_bad).  
    MESSAGE lo_ex_bad->get_text( ) TYPE 'I'.  
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic  
    EXPORTING previous = lo_ex_bad.  
    CATCH /aws1/cx_s3cclientexc INTO DATA(lo_ex_cli).  
    MESSAGE lo_ex_cli->get_text( ) TYPE 'I'.  
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic  
    EXPORTING previous = lo_ex_cli.  
    CATCH /aws1/cx_s3cserverexc INTO DATA(lo_ex_srv).  
    MESSAGE lo_ex_srv->get_text( ) TYPE 'I'.  
    RAISE EXCEPTION TYPE /aws1/cx_rt_generic  
    EXPORTING previous = lo_ex_srv.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [UpdateJobPriority](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UpdateJobStatus 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 UpdateJobStatus。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

更新 Amazon S3 Batch Operations 任務的狀態

下列 update-job-status 範例會取消正在等待核准的指定任務。

```
aws s3control update-job-status \
  --account-id 123456789012 \
  --job-id 8d9a18fe-c303-4d39-8ccc-860d372da386 \
  --requested-job-status Cancelled
```

輸出：

```
{
  "Status": "Cancelled",
  "JobId": "8d9a18fe-c303-4d39-8ccc-860d372da386"
}
```

下列 update-job-status 範例會確認並執行正在等待核准的指定。

```
aws s3control update-job-status \
  --account-id 123456789012 \
  --job-id 5782949f-3301-4fb3-be34-8d5bab54dbca \
  --requested-job-status Ready
```

Output::


```
{
  "Status": "Ready",
  "JobId": "5782949f-3301-4fb3-be34-8d5bab54dbca"
}
```

下列 `update-job-status` 範例會取消正在執行的指定任務。

```
aws s3control update-job-status \
  --account-id 123456789012 \
  --job-id 5782949f-3301-4fb3-be34-8d5bab54dbca \
  --requested-job-status Cancelled

Output:
{
  "Status": "Cancelling",
  "JobId": "5782949f-3301-4fb3-be34-8d5bab54dbca"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [UpdateJobStatus](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Cancels a job asynchronously.
 *
 * @param jobId The ID of the job to be canceled.
 * @param accountId The ID of the account associated with the job.
 * @return A {@link CompletableFuture} that completes when the job status has
 *         been updated to "CANCELLED".
 *
 *         If an error occurs during the update, the returned future will
 *         complete exceptionally.
 */
```

```

    */
    public CompletableFuture<Void> cancelJobAsync(String jobId, String accountId)
    {
        UpdateJobStatusRequest updateJobStatusRequest =
        UpdateJobStatusRequest.builder()
            .accountId(accountId)
            .jobId(jobId)
            .requestedJobStatus(String.valueOf(JobStatus.CANCELLED))
            .build();

        return asyncClient.updateJobStatus(updateJobStatusRequest)
            .thenAccept(updateJobStatusResponse -> {
                System.out.println("Job status updated to: " +
                updateJobStatusResponse.status());
            })
            .exceptionally(ex -> {
                System.err.println("Failed to cancel job: " + ex.getMessage());
                throw new RuntimeException(ex); // Propagate the exception
            });
    }
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [UpdateJobStatus](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

def cancel_job(self, job_id: str, account_id: str) -> None:
    """
    Cancel an S3 batch job.

    Args:
        job_id (str): ID of the batch job
        account_id (str): AWS account ID
    """

```

```
try:
    response = self.s3control_client.describe_job(
        AccountId=account_id,
        JobId=job_id
    )
    current_status = response['Job']['Status']
    print(f"Current job status: {current_status}")

    if current_status in ['Ready', 'Suspended', 'Active']:
        self.s3control_client.update_job_status(
            AccountId=account_id,
            JobId=job_id,
            RequestedJobStatus='Cancelled'
        )
        print(f"Job {job_id} was successfully canceled.")
    elif current_status in ['Completing', 'Complete']:
        print(f"Job is in '{current_status}' state - cannot be
cancelled")
        if current_status == 'Completing':
            print("Job is finishing up and will complete soon.")
        elif current_status == 'Complete':
            print("Job has already completed successfully.")
    else:
        print(f"Job is in '{current_status}' state - cancel not allowed")
except ClientError as e:
    print(f"Error canceling job: {e}")
    raise
```

- 如需 API 詳細資訊，請參閱《適用於 AWS Python (Boto3) 的 SDK API 參考》中的 [UpdateJobStatus](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
  " iv_requested_status = 'Cancelled'
  oo_result = lo_s3c->updatejobstatus(      " oo_result is returned for
testing purposes.
    iv_accountid      = iv_account_id
    iv_jobid          = iv_job_id
    iv_requestedjobstatus = iv_requested_status
  ).
  MESSAGE |Job { oo_result->get_jobid( ) } status updated to { oo_result-
>get_status( ) }| TYPE 'I'.
  CATCH /aws1/cx_s3cjobstatusexception INTO DATA(lo_ex_js).
  MESSAGE lo_ex_js->get_text( ) TYPE 'I'.
  RAISE EXCEPTION TYPE /aws1/cx_rt_generic
  EXPORTING previous = lo_ex_js.
  CATCH /aws1/cx_s3cnotfoundexception INTO DATA(lo_ex_nf).
  MESSAGE lo_ex_nf->get_text( ) TYPE 'I'.
  RAISE EXCEPTION TYPE /aws1/cx_rt_generic
  EXPORTING previous = lo_ex_nf.
  CATCH /aws1/cx_s3cclientexc INTO DATA(lo_ex_cli).
  MESSAGE lo_ex_cli->get_text( ) TYPE 'I'.
  RAISE EXCEPTION TYPE /aws1/cx_rt_generic
  EXPORTING previous = lo_ex_cli.
  CATCH /aws1/cx_s3cserverexc INTO DATA(lo_ex_srv).
  MESSAGE lo_ex_srv->get_text( ) TYPE 'I'.
  RAISE EXCEPTION TYPE /aws1/cx_rt_generic
  EXPORTING previous = lo_ex_srv.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [UpdateJobStatus](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 的 S3 目錄儲存貯體程式碼範例 AWS SDKs

下列程式碼範例示範如何使用 S3 Directory Buckets 搭配 AWS 軟體開發套件 (SDK)。

基本概念是程式碼範例，這些範例說明如何在服務內執行基本操作。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

案例是向您展示如何呼叫服務中的多個函數或與其他 AWS 服務組合來完成特定任務的程式碼範例。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

S3 目錄儲存貯體

- [使用 AWS SDKs 的 S3 目錄儲存貯體基本範例](#)
 - [Hello Amazon S3 目錄儲存貯體](#)
 - [使用 AWS SDK 了解 S3 目錄儲存貯體的基本概念](#)
 - [使用 AWS SDKs 的 S3 目錄儲存貯體動作](#)
 - [AbortMultipartUpload 搭配 AWS SDK 使用](#)
 - [CompleteMultipartUpload 搭配 AWS SDK 使用](#)
 - [CopyObject 搭配 AWS SDK 使用](#)
 - [CreateBucket 搭配 AWS SDK 使用](#)
 - [CreateMultipartUpload 搭配 AWS SDK 使用](#)
 - [CreateSession 搭配 AWS SDK 使用](#)
 - [DeleteBucket 搭配 AWS SDK 使用](#)
 - [DeleteBucketEncryption 搭配 AWS SDK 使用](#)
 - [DeleteBucketPolicy 搭配 AWS SDK 使用](#)
 - [DeleteObject 搭配 AWS SDK 使用](#)
 - [DeleteObjects 搭配 AWS SDK 使用](#)
 - [GetBucketEncryption 搭配 AWS SDK 使用](#)
 - [GetBucketPolicy 搭配 AWS SDK 使用](#)
 - [GetObject 搭配 AWS SDK 使用](#)
 - [GetObjectAttributes 搭配 AWS SDK 使用](#)
 - [HeadBucket 搭配 AWS SDK 使用](#)
 - [HeadObject 搭配 AWS SDK 使用](#)
 - [ListDirectoryBuckets 搭配 AWS SDK 使用](#)
 - [ListMultipartUploads 搭配 AWS SDK 使用](#)
 - [ListObjectsV2 搭配 AWS SDK 使用](#)

- [ListParts 搭配 AWS SDK 使用](#)
- [PutBucketEncryption 搭配 AWS SDK 使用](#)
- [PutBucketPolicy 搭配 AWS SDK 使用](#)
- [PutObject 搭配 AWS SDK 使用](#)
- [UploadPart 搭配 AWS SDK 使用](#)
- [UploadPartCopy 搭配 AWS SDK 使用](#)
- [使用 AWS SDKs S3 目錄儲存貯體案例](#)
 - [為 Amazon S3 目錄儲存貯體建立預先簽章的 URL，以使用 AWS SDK 取得物件](#)

使用 AWS SDKs 的 S3 目錄儲存貯體基本範例

下列程式碼範例示範如何搭配 AWS SDK 使用 Amazon S3 目錄儲存貯體的基本功能。

範例

- [Hello Amazon S3 目錄儲存貯體](#)
- [使用 AWS SDK 了解 S3 目錄儲存貯體的基本概念](#)
- [使用 AWS SDKs 的 S3 目錄儲存貯體動作](#)
 - [AbortMultipartUpload 搭配 AWS SDK 使用](#)
 - [CompleteMultipartUpload 搭配 AWS SDK 使用](#)
 - [CopyObject 搭配 AWS SDK 使用](#)
 - [CreateBucket 搭配 AWS SDK 使用](#)
 - [CreateMultipartUpload 搭配 AWS SDK 使用](#)
 - [CreateSession 搭配 AWS SDK 使用](#)
 - [DeleteBucket 搭配 AWS SDK 使用](#)
 - [DeleteBucketEncryption 搭配 AWS SDK 使用](#)
 - [DeleteBucketPolicy 搭配 AWS SDK 使用](#)
 - [DeleteObject 搭配 AWS SDK 使用](#)
 - [DeleteObjects 搭配 AWS SDK 使用](#)
 - [GetBucketEncryption 搭配 AWS SDK 使用](#)
 - [GetBucketPolicy 搭配 AWS SDK 使用](#)
 - [GetObject 搭配 AWS SDK 使用](#)

- [GetObjectAttributes 搭配 AWS SDK 使用](#)
- [HeadBucket 搭配 AWS SDK 使用](#)
- [HeadObject 搭配 AWS SDK 使用](#)
- [ListDirectoryBuckets 搭配 AWS SDK 使用](#)
- [ListMultipartUploads 搭配 AWS SDK 使用](#)
- [ListObjectsV2 搭配 AWS SDK 使用](#)
- [ListParts 搭配 AWS SDK 使用](#)
- [PutBucketEncryption 搭配 AWS SDK 使用](#)
- [PutBucketPolicy 搭配 AWS SDK 使用](#)
- [PutObject 搭配 AWS SDK 使用](#)
- [UploadPart 搭配 AWS SDK 使用](#)
- [UploadPartCopy 搭配 AWS SDK 使用](#)

Hello Amazon S3 目錄儲存貯體

下列程式碼範例示範如何開始使用 Amazon S3 目錄儲存貯體。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
package com.example.s3.directorybucket;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.Bucket;
import software.amazon.awssdk.services.s3.model.BucketInfo;
```

```
import software.amazon.awssdk.services.s3.model.BucketType;
import software.amazon.awssdk.services.s3.model.CreateBucketConfiguration;
import software.amazon.awssdk.services.s3.model.CreateBucketRequest;
import software.amazon.awssdk.services.s3.model.CreateBucketResponse;
import software.amazon.awssdk.services.s3.model.DataRedundancy;
import software.amazon.awssdk.services.s3.model.DeleteBucketRequest;
import software.amazon.awssdk.services.s3.model.ListDirectoryBucketsRequest;
import software.amazon.awssdk.services.s3.model.ListDirectoryBucketsResponse;
import software.amazon.awssdk.services.s3.model.LocationInfo;
import software.amazon.awssdk.services.s3.model.LocationType;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.util.List;
import java.util.stream.Collectors;

import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;

/**
 * Before running this example:
 * <p>
 * The SDK must be able to authenticate AWS requests on your behalf. If you have
 * not configured
 * authentication for SDKs and tools, see
 * https://docs.aws.amazon.com/sdkref/latest/guide/access.html in the AWS SDKs
 * and Tools Reference Guide.
 * <p>
 * You must have a runtime environment configured with the Java SDK.
 * See
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/setup.html in
 * the Developer Guide if this is not set up.
 * <p>
 * To use S3 directory buckets, configure a gateway VPC endpoint. This is the
 * recommended method to enable directory bucket traffic without
 * requiring an internet gateway or NAT device. For more information on
 * configuring VPC gateway endpoints, visit
 * https://docs.aws.amazon.com/AmazonS3/latest/userguide/s3-express-networking.html#s3-express-networking-vpc-gateway.
 * <p>
 * Directory buckets are available in specific AWS Regions and Zones. For
 * details on Regions and Zones supporting directory buckets, see
 * https://docs.aws.amazon.com/AmazonS3/latest/userguide/s3-express-networking.html#s3-express-endpoints.
 */
```



```
public class HelloS3DirectoryBuckets {
    private static final Logger logger =
        LoggerFactory.getLogger(HelloS3DirectoryBuckets.class);

    public static void main(String[] args) {
        String bucketName = "amzn-s3-demo-bucket"; // Replace with your bucket
        name.

        Region region = Region.US_WEST_2;
        String zone = "usw2-az1";
        S3Client s3Client = createS3Client(region);

        try {
            // Create the directory bucket
            createDirectoryBucket(s3Client, bucketName, zone);
            logger.info("Created bucket: {}", bucketName);

            // List all directory buckets
            List<String> bucketNames = listDirectoryBuckets(s3Client);
            bucketNames.forEach(name -> logger.info("Bucket Name: {}", name));
        } catch (S3Exception e) {
            logger.error("An error occurred during S3 operations: {} - Error
            code: {}",
                e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        } finally {
            try {
                // Delete the created bucket
                deleteDirectoryBucket(s3Client, bucketName);
                logger.info("Deleted bucket: {}", bucketName);
            } catch (S3Exception e) {
                logger.error("Failed to delete the bucket due to S3 error: {} -
                Error code: {}",
                    e.awsErrorDetails().errorMessage(),
                e.awsErrorDetails().errorCode(), e);
            } catch (RuntimeException e) {
                logger.error("Failed to delete the bucket due to unexpected
                error: {}", e.getMessage(), e);
            } finally {
                s3Client.close();
            }
        }
    }

    /**
```

```

    * Creates a new S3 directory bucket in a specified Zone (For example, a
    * specified Availability Zone in this code example).
    *
    * @param s3Client The S3 client used to create the bucket
    * @param bucketName The name of the bucket to be created
    * @param zone The region where the bucket will be created
    * @throws S3Exception if there's an error creating the bucket
    */
    public static void createDirectoryBucket(S3Client s3Client, String
bucketName, String zone) throws S3Exception {
        logger.info("Creating bucket: {}", bucketName);

        CreateBucketConfiguration bucketConfiguration =
CreateBucketConfiguration.builder()
            .location(LocationInfo.builder()
                .type(LocationType.AVAILABILITY_ZONE)
                .name(zone).build())
            .bucket(BucketInfo.builder()
                .type(BucketType.DIRECTORY)
                .dataRedundancy(DataRedundancy.SINGLE_AVAILABILITY_ZONE)
                .build())
            .build();

        try {
            CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
                .bucket(bucketName)
                .createBucketConfiguration(bucketConfiguration).build();
            CreateBucketResponse response = s3Client.createBucket(bucketRequest);
            logger.info("Bucket created successfully with location: {}",
response.location());
        } catch (S3Exception e) {
            logger.error("Error creating bucket: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
                e.awsErrorDetails().errorCode(), e);
            throw e;
        }
    }

    /**
    * Lists all S3 directory buckets.
    *
    * @param s3Client The S3 client used to interact with S3
    * @return A list of bucket names
    */
    public static List<String> listDirectoryBuckets(S3Client s3Client) {

```

```
        logger.info("Listing all directory buckets");

        try {
            // Create a ListBucketsRequest
            ListDirectoryBucketsRequest listBucketsRequest =
ListDirectoryBucketsRequest.builder().build();

            // Retrieve the list of buckets
            ListDirectoryBucketsResponse response =
s3Client.listDirectoryBuckets(listBucketsRequest);

            // Extract bucket names
            List<String> bucketNames = response.buckets().stream()
                .map(Bucket::name)
                .collect(Collectors.toList());

            return bucketNames;
        } catch (S3Exception e) {
            logger.error("Failed to list buckets: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
                e.awsErrorDetails().errorCode(), e);
            throw e;
        }
    }

    /**
     * Deletes the specified S3 directory bucket.
     *
     * @param s3Client The S3 client used to interact with S3
     * @param bucketName The name of the bucket to delete
     */
    public static void deleteDirectoryBucket(S3Client s3Client, String
bucketName) {
        try {
            DeleteBucketRequest deleteBucketRequest =
DeleteBucketRequest.builder()
                .bucket(bucketName)
                .build();
            s3Client.deleteBucket(deleteBucketRequest);
        } catch (S3Exception e) {
            logger.error("Failed to delete bucket: " + bucketName + " - Error
code: " + e.awsErrorDetails().errorCode(),
                e);
            throw e;
        }
    }
}
```

```
    }  
  }  
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
 - [CreateBucket](#)
 - [ListDirectoryBuckets](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 了解 S3 目錄儲存貯體的基本概念

下列程式碼範例示範如何：

- 設定 VPC 和 VPC 端點。
- 設定政策、角色和使用者，以使用 S3 目錄儲存貯體和 S3 Express One Zone 儲存類別。
- 建立兩個 S3 用戶端。
- 建立兩個儲存貯體。
- 建立物件並進行複製。
- 展示效能差異。
- 填入儲存貯體以顯示字典序的差異。
- 提示使用者查看他們是否要清除資源。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

執行展示 Amazon S3 功能的互動式情境。

```
public class S3DirectoriesScenario {

    public static final String DASHES = new String(new char[80]).replace("\0",
    "-");

    private static final Logger logger =
    LoggerFactory.getLogger(S3DirectoriesScenario.class);
    static Scanner scanner = new Scanner(System.in);

    private static S3AsyncClient mS3RegularClient;
    private static S3AsyncClient mS3ExpressClient;

    private static String mdirectoryBucketName;
    private static String mregularBucketName;

    private static String stackName = "cfn-stack-s3-express-basics--" +
    UUID.randomUUID();

    private static String regularUser = "";
    private static String vpcId = "";
    private static String expressUser = "";

    private static String vpcEndpointId = "";

    private static final S3DirectoriesActions s3DirectoriesActions = new
    S3DirectoriesActions();

    public static void main(String[] args) {
        try {
            s3ExpressScenario();
        } catch (RuntimeException e) {
            logger.info(e.getMessage());
        }
    }

    // Runs the scenario.
    private static void s3ExpressScenario() {
        logger.info(DASHES);
        logger.info("Welcome to the Amazon S3 Express Basics demo using AWS SDK
    for Java V2.");
        logger.info("""
            Let's get started! First, please note that S3 Express One Zone works
    best when working within the AWS infrastructure,
```

```

        specifically when working in the same Availability Zone (AZ). To see
        the best results in this example and when you implement
        directory buckets into your infrastructure, it is best to put your
        compute resources in the same AZ as your directory
        bucket.
        "");
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    // Create an optional VPC and create 2 IAM users.
    UserNames userNames = createVpcUsers();
    String expressUserName = userNames.getExpressUserName();
    String regularUserName = userNames.getRegularUserName();

    // Set up two S3 clients, one regular and one express,
    // and two buckets, one regular and one directory.
    setupClientsAndBuckets(expressUserName, regularUserName);

    // Create an S3 session for the express S3 client and add objects to the
    buckets.
    logger.info("Now let's add some objects to our buckets and demonstrate
    how to work with S3 Sessions.");
    waitForInputToContinue(scanner);
    String bucketObject = createSessionAddObjects();

    // Demonstrate performance differences between regular and directory
    buckets.
    demonstratePerformance(bucketObject);

    // Populate the buckets to show the lexicographical difference between
    // regular and express buckets.
    showLexicographicalDifferences(bucketObject);

    logger.info(DASHES);
    logger.info("That's it for our tour of the basic operations for S3
    Express One Zone.");
    logger.info("Would you like to cleanUp the AWS resources? (y/n): ");
    String response = scanner.next().trim().toLowerCase();
    if (response.equals("y")) {
        cleanUp(stackName);
    }
}

/*

```

```

Delete resources created by this scenario.
*/
public static void cleanUp(String stackName) {
    try {
        if (mdirectoryBucketName != null) {

s3DirectoriesActions.deleteBucketAndObjectsAsync(mS3ExpressClient,
mdirectoryBucketName).join();
        }
        logger.info("Deleted directory bucket " + mdirectoryBucketName);
        mdirectoryBucketName = null;
        if (mregularBucketName != null) {

s3DirectoriesActions.deleteBucketAndObjectsAsync(mS3RegularClient,
mregularBucketName).join();
        }
    } catch (CompletionException ce) {
        Throwable cause = ce.getCause();
        if (cause instanceof S3Exception) {
            logger.error("S3Exception occurred: {}", cause.getMessage(), ce);
        } else {
            logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
        }
    }

    logger.info("Deleted regular bucket " + mregularBucketName);
    mregularBucketName = null;
    CloudFormationHelper.destroyCloudFormationStack(stackName);
}

private static void showLexicographicalDifferences(String bucketObject) {
    logger.info(DASHES);
    logger.info("")
        7. Populate the buckets to show the lexicographical (alphabetical)
difference
        when object names are listed. Now let's explore how directory buckets
store
        objects in a different manner to regular buckets. The key is in the
name
        "Directory". Where regular buckets store their key/value pairs in a
flat manner, directory buckets use actual directories/folders.
        This allows for more rapid indexing, traversing, and therefore
retrieval times!

```

The more segmented your bucket is, with lots of directories, sub-directories, and objects, the more efficient it becomes.

This structural difference also causes `ListObject` operations to behave

differently, which can cause unexpected results. Let's add a few more

objects in sub-directories to see how the output of `ListObjects` changes.

```
""");
```

```
waitForInputToContinue(scanner);
```

```
// Populate a few more files in each bucket so that we can use
```

```
// ListObjects and show the difference.
```

```
String otherObject = "other/" + bucketObject;
```

```
String altObject = "alt/" + bucketObject;
```

```
String otherAltObject = "other/alt/" + bucketObject;
```

```
try {
```

```
    s3DirectoriesActions.putObjectAsync(mS3RegularClient,
mregularBucketName, otherObject, "").join();
```

```
    s3DirectoriesActions.putObjectAsync(mS3ExpressClient,
mdirectoryBucketName, otherObject, "").join();
```

```
    s3DirectoriesActions.putObjectAsync(mS3RegularClient,
mregularBucketName, altObject, "").join();
```

```
    s3DirectoriesActions.putObjectAsync(mS3ExpressClient,
mdirectoryBucketName, altObject, "").join();
```

```
    s3DirectoriesActions.putObjectAsync(mS3RegularClient,
mregularBucketName, otherAltObject, "").join();
```

```
    s3DirectoriesActions.putObjectAsync(mS3ExpressClient,
mdirectoryBucketName, otherAltObject, "").join();
```

```
    } catch (CompletionException ce) {
```

```
        Throwable cause = ce.getCause();
```

```
        if (cause instanceof NoSuchBucketException) {
```

```
            logger.error("S3Exception occurred: {}", cause.getMessage(), ce);
```

```
        } else {
```

```
            logger.error("An unexpected error occurred: {}",
```

```
cause.getMessage(), ce);
```

```
        }
```

```
        return;
```

```
    }
```



```

        try {
            // List objects in both S3 buckets.
            List<String> dirBucketObjects =
s3DirectoriesActions.listObjectsAsync(mS3ExpressClient,
mdirectoryBucketName).join();
            List<String> regBucketObjects =
s3DirectoriesActions.listObjectsAsync(mS3RegularClient,
mregularBucketName).join();

            logger.info("Directory bucket content");
            for (String obj : dirBucketObjects) {
                logger.info(obj);
            }

            logger.info("Regular bucket content");
            for (String obj : regBucketObjects) {
                logger.info(obj);
            }
        } catch (CompletionException e) {
            logger.error("Async operation failed: {} ",
e.getCause().getMessage());
            return;
        }

        logger.info("""
            Notice how the regular bucket lists objects in lexicographical order,
while the directory bucket does not. This is
            because the regular bucket considers the whole "key" to be the object
identifier, while the directory bucket actually
            creates directories and uses the object "key" as a path to the
object.

            """);
        waitForInputToContinue(scanner);
    }

    /**
     * Demonstrates the performance difference between downloading an object from
a directory bucket and a regular bucket.
     *
     * <p>This method:
     * <ul>
     *     <li>Prompts the user to choose the number of downloads (default is
1,000).</li>

```

```

*      <li>Downloads the specified object from the directory bucket and
measures the total time.</li>
*      <li>Downloads the same object from the regular bucket and measures the
total time.</li>
*      <li>Compares the time differences and prints the results.</li>
* </ul>
*
* <p>Note: The performance difference will be more pronounced if this
example is run on an EC2 instance
* in the same Availability Zone as the buckets.
*
* @param bucketObject the name of the object to download
*/
private static void demonstratePerformance(String bucketObject) {
    logger.info(DASHES);
    logger.info("6. Demonstrate the performance difference.");
    logger.info("
        Now, let's do a performance test. We'll download the same object from
each
        bucket repeatedly and compare the total time needed.

        Note: the performance difference will be much more pronounced if this
example is run in an EC2 instance in the same Availability Zone as
the bucket.
        ");
    waitForInputToContinue(scanner);

    int downloads = 1000; // Default value.
    logger.info("The default number of downloads of the same object for this
example is set at " + downloads + ".");

    // Ask if the user wants to download a different number.
    logger.info("Would you like to download the file a different number of
times? (y/n): ");
    String response = scanner.next().trim().toLowerCase();
    if (response.equals("y")) {
        int maxDownloads = 1_000_000;

        // Ask for a valid number of downloads.
        while (true) {
            logger.info("Enter a number between 1 and " + maxDownloads + "
for the number of downloads: ");
            if (scanner.hasNextInt()) {
                downloads = scanner.nextInt();

```

```
        if (downloads >= 1 && downloads <= maxDownloads) {
            break;
        } else {
            logger.info("Please enter a number between 1 and " +
maxDownloads + ".");
        }
    } else {
        logger.info("Invalid input. Please enter a valid integer.");
        scanner.next();
    }
}

    logger.info("You have chosen to download {} items.", downloads);
} else {
    logger.info("No changes made. Using default downloads: {}",
downloads);
}
// Simulating the download process for the directory bucket.
logger.info("Downloading from the directory bucket.");
long directoryTimeStart = System.nanoTime();
for (int index = 0; index < downloads; index++) {
    if (index % 50 == 0) {
        logger.info("Download " + index + " of " + downloads);
    }

    try {
        // Get the object from the directory bucket.
        s3DirectoriesActions.getObjectAsync(mS3ExpressClient,
mdirectoryBucketName, bucketObject).join();
    } catch (CompletionException ce) {
        Throwable cause = ce.getCause();
        if (cause instanceof NoSuchKeyException) {
            logger.error("S3Exception occurred: {}", cause.getMessage(),
ce);
        } else {
            logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
        }
        return;
    }
}
long directoryTimeDifference = System.nanoTime() - directoryTimeStart;

// Download from the regular bucket.
```

```

        logger.info("Downloading from the regular bucket.");
        long normalTimeStart = System.nanoTime();
        for (int index = 0; index < downloads; index++) {
            if (index % 50 == 0) {
                logger.info("Download " + index + " of " + downloads);
            }

            try {
                s3DirectoriesActions.getObjectAsync(mS3RegularClient,
mregularBucketName, bucketObject).join();
            } catch (CompletionException ce) {
                Throwable cause = ce.getCause();
                if (cause instanceof NoSuchKeyException) {
                    logger.error("S3Exception occurred: {}", cause.getMessage(),
ce);
                } else {
                    logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
                }
                return;
            }
        }

        long normalTimeDifference = System.nanoTime() - normalTimeStart;
        logger.info("The directory bucket took " + directoryTimeDifference
+ " nanoseconds, while the regular bucket took " + normalTimeDifference + "
nanoseconds.");
        long difference = normalTimeDifference - directoryTimeDifference;
        logger.info("That's a difference of " + difference + " nanoseconds, or");
        logger.info(difference / 1_000_000_000.0 + " seconds.");

        if (difference < 0) {
            logger.info("The directory buckets were slower. This can happen if
you are not running on the cloud within a VPC.");
        }
        waitForInputToContinue(scanner);
    }

    private static String createSessionAddObjects() {
        logger.info(DASHES);
        logger.info(""""
        5. Create an object and copy it.
        We'll create an object consisting of some text and upload it to the
        regular bucket.

```

```

        """);
        waitForInputToContinue(scanner);

        String bucketObject = "basic-text-object.txt";
        try {
            s3DirectoriesActions.putObjectAsync(mS3RegularClient,
mregularBucketName, bucketObject, "Look Ma, I'm a bucket!").join();
            s3DirectoriesActions.createSessionAsync(mS3ExpressClient,
mdirectoryBucketName).join();

            // Copy the object to the destination S3 bucket.
            s3DirectoriesActions.copyObjectAsync(mS3ExpressClient,
mregularBucketName, bucketObject, mdirectoryBucketName, bucketObject).join();
        } catch (CompletionException ce) {
            Throwable cause = ce.getCause();
            if (cause instanceof S3Exception) {
                logger.error("S3Exception occurred: {}", cause.getMessage(), ce);
            } else {
                logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
            }
        }
        logger.info("""
            It worked! This is because the S3Client that performed the copy
operation
            is the expressClient using the credentials for the user with
permission to
            work with directory buckets.

            It's important to remember the user permissions when interacting
with
            directory buckets. Instead of validating permissions on every call
as
            regular buckets do, directory buckets utilize the user credentials
and session
            token to validate. This allows for much faster connection speeds on
every call.
            For single calls, this is low, but for many concurrent calls
            this adds up to a lot of time saved.
        """);
        waitForInputToContinue(scanner);
        return bucketObject;
    }

```

```

/**
 * Creates VPC users for the S3 Express One Zone scenario.
 * <p>
 * This method performs the following steps:
 * <ol>
 *     <li>Optionally creates a new VPC and VPC Endpoint if the application
is running in an EC2 instance in the same Availability Zone as the directory
buckets.</li>
 *     <li>Creates two IAM users: one with S3 Express One Zone permissions
and one without.</li>
 * </ol>
 *
 * @return a {@link UserNames} object containing the names of the created IAM
users
 */
public static UserNames createVpcUsers() {
    /**
     * Optionally create a VPC.
     * Create two IAM users, one with S3 Express One Zone permissions and one
without.
     */
    logger.info(DASHES);
    logger.info("""
        1. First, we'll set up a new VPC and VPC Endpoint if this program is
running in an EC2 instance in the same AZ as your\s
        directory buckets will be. Are you running this in an EC2 instance
located in the same AZ as your intended directory buckets?
        """);

    logger.info("Do you want to setup a VPC Endpoint? (y/n)");
    String endpointAns = scanner.nextLine().trim();
    if (endpointAns.equalsIgnoreCase("y")) {
        logger.info("""
            Great! Let's set up a VPC, retrieve the Route Table from it, and
create a VPC Endpoint to connect the S3 Client to.
            """);
        try {
            s3DirectoriesActions.setupVPCAsync().join();
        } catch (CompletionException ce) {
            Throwable cause = ce.getCause();
            if (cause instanceof Ec2Exception) {
                logger.error("IamException occurred: {}", cause.getMessage(),
ce);
            } else {

```

```

        logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
    }
}
waitForInputToContinue(scanner);
} else {
    logger.info("Skipping the VPC setup. Don't forget to use this in
production!");
}
logger.info(DASHES);
logger.info("""
    2. Create a RegularUser and ExpressUser by using the AWS CDK.
    One IAM User, named RegularUser, will have permissions to work only
    with regular buckets and one IAM user, named ExpressUser, will have
    permissions to work only with directory buckets.
    """);
waitForInputToContinue(scanner);

// Create two users required for this scenario.
Map<String, String> stackOutputs = createUsersUsingCDK(stackName);
regularUser = stackOutputs.get("RegularUser");
expressUser = stackOutputs.get("ExpressUser");

UserNames names = new UserNames();
names.setRegularUserName(regularUser);
names.setExpressUserName(expressUser);
return names;
}

/**
 * Creates users using AWS CloudFormation.
 *
 * @return a {@link Map} of String keys and String values representing the
stack outputs,
 * which may include user-related information such as user names and IDs.
 */
public static Map<String, String> createUsersUsingCDK(String stackName) {
    logger.info("We'll use an AWS CloudFormation template to create the IAM
users and policies.");
    CloudFormationHelper.deployCloudFormationStack(stackName);
    return CloudFormationHelper.getStackOutputsAsync(stackName).join();
}

/**

```

```

    * Sets up the necessary clients and buckets for the S3 Express service.
    *
    * @param expressUserName the username for the user with S3 Express
permissions
    * @param regularUserName the username for the user with regular S3
permissions
    */
    public static void setupClientsAndBuckets(String expressUserName, String
regularUserName) {
        Scanner locscanner = new Scanner(System.in);
        String accessKeyIdforRegUser;
        String secretAccessforRegUser;
        try {
            CreateAccessKeyResponse keyResponse =
s3DirectoriesActions.createAccessKeyAsync(regularUserName).join();
            accessKeyIdforRegUser = keyResponse.accessKey().accessKeyId();
            secretAccessforRegUser = keyResponse.accessKey().secretAccessKey();
        } catch (CompletionException ce) {
            Throwable cause = ce.getCause();
            if (cause instanceof IamException) {
                logger.error("IamException occurred: {}", cause.getMessage(),
ce);
            } else {
                logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
            }
            return;
        }

        String accessKeyIdforExpressUser;
        String secretAccessforExpressUser;
        try {
            CreateAccessKeyResponse keyResponseExpress =
s3DirectoriesActions.createAccessKeyAsync(expressUserName).join();
            accessKeyIdforExpressUser =
keyResponseExpress.accessKey().accessKeyId();
            secretAccessforExpressUser =
keyResponseExpress.accessKey().secretAccessKey();
        } catch (CompletionException ce) {
            Throwable cause = ce.getCause();
            if (cause instanceof IamException) {
                logger.error("IamException occurred: {}", cause.getMessage(),
ce);
            } else {

```



```

        logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
    }
    return;
}

logger.info(DASHES);
logger.info("""
    3. Create two S3Clients; one uses the ExpressUser's credentials and
one uses the RegularUser's credentials.
    The 2 S3Clients will use different credentials.
    """);
waitForInputToContinue(locscanner);
try {
    mS3RegularClient =
createS3ClientWithAccessKeyAsync(accessKeyIdforRegUser,
secretAccessforRegUser).join();
    mS3ExpressClient =
createS3ClientWithAccessKeyAsync(accessKeyIdforExpressUser,
secretAccessforExpressUser).join();
} catch (CompletionException ce) {
    Throwable cause = ce.getCause();
    if (cause instanceof IllegalArgumentException) {
        logger.error("An invalid argument exception occurred: {}",
cause.getMessage(), ce);
    } else {
        logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
    }
    return;
}

logger.info("""
    We can now use the ExpressUser client to make calls to S3 Express
operations.
    """);
waitForInputToContinue(locscanner);
logger.info(DASHES);
logger.info("""
    4. Create two buckets.
    Now we will create a directory bucket which is the linchpin of the S3
Express One Zone service. Directory buckets
    behave differently from regular S3 buckets which we will explore
here. We'll also create a regular bucket, put

```

```

        an object into the regular bucket, and copy it to the directory
bucket.
        """);

        logger.info("""
        Now, let's choose an availability zone (AZ) for the directory
bucket.

        We'll choose one that is supported.
        """);
        String zoneId;
        String regularBucketName;
        try {
            zoneId = s3DirectoriesActions.selectAvailabilityZoneIdAsync().join();
            regularBucketName = "amzn-s3-demo-bucket"; // Replace with your
bucket name.
        } catch (CompletionException ce) {
            Throwable cause = ce.getCause();
            if (cause instanceof Ec2Exception) {
                logger.error("EC2Exception occurred: {}", cause.getMessage(),
ce);
            } else {
                logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
            }
            return;
        }
        logger.info("""
        Now, let's create the actual directory bucket, as well as a regular
bucket."
        """);

        String directoryBucketName = "amzn-s3-demo-bucket--" + zoneId + "--x-
s3"; // Replace with your directory bucket name.
        try {
            s3DirectoriesActions.createDirectoryBucketAsync(mS3ExpressClient,
directoryBucketName, zoneId).join();
            logger.info("Created directory bucket {}", directoryBucketName);
        } catch (CompletionException ce) {
            Throwable cause = ce.getCause();
            if (cause instanceof BucketAlreadyExistsException) {
                logger.error("The bucket already exists. Moving on: {}",
cause.getMessage(), ce);
            } else {

```

```

        logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
        return;
    }
}

// Assign to the data member.
mdirectoryBucketName = directoryBucketName;
try {
    s3DirectoriesActions.createBucketAsync(mS3RegularClient,
regularBucketName).join();
    logger.info("Created regular bucket {} ", regularBucketName);
    mregularBucketName = regularBucketName;
} catch (CompletionException ce) {
    Throwable cause = ce.getCause();
    if (cause instanceof BucketAlreadyExistsException) {
        logger.error("The bucket already exists. Moving on: {}",
cause.getMessage(), ce);
    } else {
        logger.error("An unexpected error occurred: {}",
cause.getMessage(), ce);
        return;
    }
}
logger.info("Great! Both buckets were created.");
waitForInputToContinue(locscanner);
}

/**
 * Creates an asynchronous S3 client with the specified access key and secret
access key.
 *
 * @param accessKeyId    the AWS access key ID
 * @param secretAccessKey the AWS secret access key
 * @return a {@link CompletableFuture} that asynchronously creates the S3
client
 * @throws IllegalArgumentException if the access key ID or secret access key
is null
 */
public static CompletableFuture<S3AsyncClient>
createS3ClientWithAccessKeyAsync(String accessKeyId, String secretAccessKey) {
    return CompletableFuture.supplyAsync(() -> {
        // Validate input parameters

```

```

        if (accessKeyId == null || accessKeyId.isBlank() || secretAccessKey
== null || secretAccessKey.isBlank()) {
            throw new IllegalArgumentException("Access Key ID and Secret
Access Key must not be null or empty");
        }

        AwsBasicCredentials awsCredentials =
        AwsBasicCredentials.create(accessKeyId, secretAccessKey);
        return S3AsyncClient.builder()

        .credentialsProvider(StaticCredentialsProvider.create(awsCredentials))
            .region(Region.US_WEST_2)
            .build();
    });
}

private static void waitForInputToContinue(Scanner scanner) {
    while (true) {
        logger.info("");
        logger.info("Enter 'c' followed by <ENTER> to continue:");
        String input = scanner.nextLine();

        if (input.trim().equalsIgnoreCase("c")) {
            logger.info("Continuing with the program...");
            logger.info("");
            break;
        } else {
            logger.info("Invalid input. Please try again.");
        }
    }
}
}
}

```

Amazon S3 SDK 方法的包裝函式類別。

```

public class S3DirectoriesActions {

    private static IamAsyncClient iamAsyncClient;

    private static Ec2AsyncClient ec2AsyncClient;
    private static final Logger logger =
    LoggerFactory.getLogger(S3DirectoriesActions.class);

```

```
private static IamAsyncClient getIAMAsyncClient() {
    if (iamAsyncClient == null) {
        SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
            .maxConcurrency(100)
            .connectionTimeout(Duration.ofSeconds(60))
            .readTimeout(Duration.ofSeconds(60))
            .writeTimeout(Duration.ofSeconds(60))
            .build();

        ClientOverrideConfiguration overrideConfig =
ClientOverrideConfiguration.builder()
            .apiCallTimeout(Duration.ofMinutes(2))
            .apiCallAttemptTimeout(Duration.ofSeconds(90))
            .retryStrategy(RetryMode.STANDARD)
            .build();

        iamAsyncClient = IamAsyncClient.builder()
            .httpClient(httpClient)
            .overrideConfiguration(overrideConfig)
            .build();
    }
    return iamAsyncClient;
}

private static Ec2AsyncClient getEc2AsyncClient() {
    if (ec2AsyncClient == null) {
        SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
            .maxConcurrency(100)
            .connectionTimeout(Duration.ofSeconds(60))
            .readTimeout(Duration.ofSeconds(60))
            .writeTimeout(Duration.ofSeconds(60))
            .build();

        ClientOverrideConfiguration overrideConfig =
ClientOverrideConfiguration.builder()
            .apiCallTimeout(Duration.ofMinutes(2))
            .apiCallAttemptTimeout(Duration.ofSeconds(90))
            .retryStrategy(RetryMode.STANDARD)
            .build();

        ec2AsyncClient = Ec2AsyncClient.builder()
            .httpClient(httpClient)
            .region(Region.US_WEST_2)
```

```

        .overrideConfiguration(overrideConfig)
        .build();
    }
    return ec2AsyncClient;
}

/**
 * Deletes the specified S3 bucket and all the objects within it
 * asynchronously.
 *
 * @param s3AsyncClient the S3 asynchronous client to use for the operations
 * @param bucketName the name of the S3 bucket to be deleted
 * @return a {@link CompletableFuture} that completes with a {@link
 * WaiterResponse} containing the
 *
 *      {@link HeadBucketResponse} when the bucket has been successfully
 * deleted
 *
 * @throws CompletionException if there was an error deleting the bucket or
 * its objects
 */
public CompletableFuture<WaiterResponse<HeadBucketResponse>>
deleteBucketAndObjectsAsync(S3AsyncClient s3AsyncClient, String bucketName) {
    ListObjectsV2Request listRequest = ListObjectsV2Request.builder()
        .bucket(bucketName)
        .build();

    return s3AsyncClient.listObjectsV2(listRequest)
        .thenCompose(listResponse -> {
            if (!listResponse.contents().isEmpty()) {
                List<ObjectIdentifier> objectIdentifiers =
listResponse.contents().stream()
                    .map(s3Object ->
ObjectIdentifier.builder().key(s3Object.key()).build())
                    .collect(Collectors.toList());

                DeleteObjectsRequest deleteRequest =
DeleteObjectsRequest.builder()
                    .bucket(bucketName)

.delete(Delete.builder().objects(objectIdentifiers).build())
                    .build();

                return s3AsyncClient.deleteObjects(deleteRequest)
                    .thenAccept(deleteResponse -> {
                        if (!deleteResponse.errors().isEmpty()) {

```

```

                deleteResponse.errors().forEach(error ->
                    logger.error("Couldn't delete object " +
error.key() + ". Reason: " + error.message()));
                }
            });
        }
        return CompletableFuture.completedFuture(null);
    })
    .thenCompose(ignored -> {
        DeleteBucketRequest deleteBucketRequest =
DeleteBucketRequest.builder()
            .bucket(bucketName)
            .build();
        return s3AsyncClient.deleteBucket(deleteBucketRequest);
    })
    .thenCompose(ignored -> {
        S3AsyncWaiter waiter = s3AsyncClient.waiter();
        HeadBucketRequest headBucketRequest =
HeadBucketRequest.builder().bucket(bucketName).build();
        return waiter.waitUntilBucketNotExists(headBucketRequest);
    })
    .whenComplete((ignored, exception) -> {
        if (exception != null) {
            Throwable cause = exception.getCause();
            if (cause instanceof S3Exception) {
                throw new CompletionException("Error deleting bucket: " +
bucketName, cause);
            }
            throw new CompletionException("Failed to delete bucket and
objects: " + bucketName, exception);
        }
        logger.info("Bucket deleted successfully: " + bucketName);
    });
    }

/**
 * Lists the objects in an S3 bucket asynchronously.
 *
 * @param s3Client the S3 async client to use for the operation
 * @param bucketName the name of the S3 bucket containing the objects to list
 * @return a {@link CompletableFuture} that contains the list of object keys
 * in the specified bucket
 */

```

```

    public CompletableFuture<List<String>> listObjectsAsync(S3AsyncClient
s3Client, String bucketName) {
        ListObjectsV2Request request = ListObjectsV2Request.builder()
            .bucket(bucketName)
            .build();

        return s3Client.listObjectsV2(request)
            .thenApply(response -> response.contents().stream()
                .map(S3Object::key)
                .toList())
            .whenComplete((result, exception) -> {
                if (exception != null) {
                    throw new CompletionException("Couldn't list objects in
bucket: " + bucketName, exception);
                }
            });
    }

    /**
     * Retrieves an object from an Amazon S3 bucket asynchronously.
     *
     * @param s3Client    the S3 async client to use for the operation
     * @param bucketName the name of the S3 bucket containing the object
     * @param keyName     the unique identifier (key) of the object to retrieve
     * @return a {@link CompletableFuture} that, when completed, contains the
     object's content as a {@link ResponseBytes} of {@link GetObjectResponse}
     */
    public CompletableFuture<ResponseBytes<GetObjectResponse>>
getObjectAsync(S3AsyncClient s3Client, String bucketName, String keyName) {
        GetObjectRequest objectRequest = GetObjectRequest.builder()
            .key(keyName)
            .bucket(bucketName)
            .build();

        // Get the object asynchronously and transform it into a byte array
        return s3Client.getObject(objectRequest,
AsyncResponseTransformer.toBytes())
            .exceptionally(exception -> {
                Throwable cause = exception.getCause();
                if (cause instanceof NoSuchKeyException) {
                    throw new CompletionException("Failed to get the object.
Reason: " + ((S3Exception) cause).awsErrorDetails().errorMessage(), cause);
                }
            });
    }

```



```

        throw new CompletionException("Failed to get the object",
exception);
    });
}

/**
 * Asynchronously copies an object from one S3 bucket to another.
 *
 * @param s3Client          the S3 async client to use for the copy
operation
 * @param sourceBucket      the name of the source bucket
 * @param sourceKey         the key of the object to be copied in the source
bucket
 * @param destinationBucket the name of the destination bucket
 * @param destinationKey    the key of the copied object in the destination
bucket
 * @return a {@link CompletableFuture} that completes when the copy operation
is finished
 */
public CompletableFuture<Void> copyObjectAsync(S3AsyncClient s3Client, String
sourceBucket, String sourceKey, String destinationBucket, String destinationKey)
{
    CopyObjectRequest copyRequest = CopyObjectRequest.builder()
        .sourceBucket(sourceBucket)
        .sourceKey(sourceKey)
        .destinationBucket(destinationBucket)
        .destinationKey(destinationKey)
        .build();

    return s3Client.copyObject(copyRequest)
        .thenRun(() -> logger.info("Copied object '" + sourceKey + "' from
bucket '" + sourceBucket + "' to bucket '" + destinationBucket + "'"))
        .whenComplete((ignored, exception) -> {
            if (exception != null) {
                Throwable cause = exception.getCause();
                if (cause instanceof S3Exception) {
                    throw new CompletionException("Couldn't
copy object '" + sourceKey + "' from bucket '" + sourceBucket + "'
to bucket '" + destinationBucket + "'. Reason: " + ((S3Exception)
cause).awsErrorDetails().errorMessage(), cause);
                }
                throw new CompletionException("Failed to copy object",
exception);
            }
        })
}

```

```

        });
    }

    /**
     * Asynchronously creates a session for the specified S3 bucket.
     *
     * @param s3Client    the S3 asynchronous client to use for creating the
     session
     * @param bucketName the name of the S3 bucket for which to create the
     session
     * @return a {@link CompletableFuture} that completes when the session is
     created, or throws a {@link CompletionException} if an error occurs
     */
    public CompletableFuture<CreateSessionResponse>
    createSessionAsync(S3AsyncClient s3Client, String bucketName) {
        CreateSessionRequest request = CreateSessionRequest.builder()
            .bucket(bucketName)
            .build();

        return s3Client.createSession(request)
            .whenComplete((response, exception) -> {
                if (exception != null) {
                    Throwable cause = exception.getCause();
                    if (cause instanceof S3Exception) {
                        throw new CompletionException("Couldn't create the
     session. Reason: " + ((S3Exception) cause).awsErrorDetails().errorMessage(),
     cause);
                    }
                    throw new CompletionException("Unexpected error occurred
     while creating session", exception);
                }
                logger.info("Created session for bucket: " + bucketName);
            });
    }

    /**
     * Creates a new S3 directory bucket in a specified Zone (For example, a
     * specified Availability Zone in this code example).
     *
     * @param s3Client    The asynchronous S3 client used to create the bucket
     * @param bucketName The name of the bucket to be created
     * @param zone        The Availability Zone where the bucket will be created
     * @throws CompletionException if there's an error creating the bucket

```

```

    */
    public CompletableFuture<CreateBucketResponse>
createDirectoryBucketAsync(S3AsyncClient s3Client, String bucketName, String
zone) {
    logger.info("Creating bucket: " + bucketName);

    CreateBucketConfiguration bucketConfiguration =
CreateBucketConfiguration.builder()
        .location(LocationInfo.builder()
            .type(LocationType.AVAILABILITY_ZONE)
            .name(zone)
            .build())
        .bucket(BucketInfo.builder()
            .type(BucketType.DIRECTORY)
            .dataRedundancy(DataRedundancy.SINGLE_AVAILABILITY_ZONE)
            .build())
        .build();

    CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
        .bucket(bucketName)
        .createBucketConfiguration(bucketConfiguration)
        .build();

    return s3Client.createBucket(bucketRequest)
        .whenComplete((response, exception) -> {
            if (exception != null) {
                Throwable cause = exception.getCause();
                if (cause instanceof BucketAlreadyExistsException) {
                    throw new CompletionException("The bucket already exists:
" + ((S3Exception) cause).awsErrorDetails().errorMessage(), cause);
                }
                throw new CompletionException("Unexpected error occurred
while creating bucket", exception);
            }
            logger.info("Bucket created successfully with location: " +
response.location());
        });
    }

    /**
     * Creates an S3 bucket asynchronously.
     *
     * @param s3Client    the S3 async client to use for the bucket creation
     * @param bucketName  the name of the S3 bucket to create

```

```

    * @return a {@link CompletableFuture} that completes with the {@link
    WaiterResponse} containing the {@link HeadBucketResponse}
    *         when the bucket is successfully created
    * @throws CompletionException if there's an error creating the bucket
    */
    public CompletableFuture<WaiterResponse<HeadBucketResponse>>
    createBucketAsync(S3AsyncClient s3Client, String bucketName) {
        CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
            .bucket(bucketName)
            .build();

        return s3Client.createBucket(bucketRequest)
            .thenCompose(response -> {
                S3AsyncWaiter s3Waiter = s3Client.waiter();
                HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
                    .bucket(bucketName)
                    .build();
                return s3Waiter.waitUntilBucketExists(bucketRequestWait);
            })
            .whenComplete((response, exception) -> {
                if (exception != null) {
                    Throwable cause = exception.getCause();
                    if (cause instanceof BucketAlreadyExistsException) {
                        throw new CompletionException("The S3 bucket exists: " +
                            cause.getMessage(), cause);
                    } else {
                        throw new CompletionException("Failed to create access
                            key: " + exception.getMessage(), exception);
                    }
                }
                logger.info(bucketName + " is ready");
            });
    }

    /**
     * Uploads an object to an Amazon S3 bucket asynchronously.
     *
     * @param s3Client    the S3 async client to use for the upload
     * @param bucketName  the destination S3 bucket name
     * @param bucketObject the name of the object to be uploaded
     * @param text        the content to be uploaded as the object
     */
    public CompletableFuture<PutObjectResponse> putObjectAsync(S3AsyncClient
        s3Client, String bucketName, String bucketObject, String text) {

```

```

        PutObjectRequest objectRequest = PutObjectRequest.builder()
            .bucket(bucketName)
            .key(bucketObject)
            .build();

        return s3Client.putObject(objectRequest,
            AsyncRequestBody.fromString(text))
            .whenComplete((response, exception) -> {
                if (exception != null) {
                    Throwable cause = exception.getCause();
                    if (cause instanceof NoSuchBucketException) {
                        throw new CompletionException("The S3 bucket does not
exist: " + cause.getMessage(), cause);
                    } else {
                        throw new CompletionException("Failed to create access
key: " + exception.getMessage(), exception);
                    }
                }
            });
    }

    /**
     * Creates an AWS IAM access key asynchronously for the specified user name.
     *
     * @param userName the name of the IAM user for whom to create the access key
     * @return a {@link CompletableFuture} that completes with the {@link
CreateAccessKeyResponse} containing the created access key
     */
    public CompletableFuture<CreateAccessKeyResponse> createAccessKeyAsync(String
userName) {
        CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
            .userName(userName)
            .build();

        return getIAMAsyncClient().createAccessKey(request)
            .whenComplete((response, exception) -> {
                if (response != null) {
                    logger.info("Access Key Created.");
                } else {
                    if (exception == null) {
                        Throwable cause = exception.getCause();
                        if (cause instanceof IamException) {
                            throw new CompletionException("IAM error while
creating access key: " + cause.getMessage(), cause);

```

```

        } else {
            throw new CompletionException("Failed to create
access key: " + exception.getMessage(), exception);
        }
    }
}

});

}

/**
 * Asynchronously selects an Availability Zone ID from the available EC2
zones.
 *
 * @return A {@link CompletableFuture} that resolves to the selected
Availability Zone ID.
 * @throws CompletionException if an error occurs during the request or
processing.
 */
public CompletableFuture<String> selectAvailabilityZoneIdAsync() {
    DescribeAvailabilityZonesRequest zonesRequest =
DescribeAvailabilityZonesRequest.builder()
        .build();

    return getEc2AsyncClient().describeAvailabilityZones(zonesRequest)
        .thenCompose(response -> {
            List<AvailabilityZone> zonesList = response.availabilityZones();
            if (zonesList.isEmpty()) {
                logger.info("No availability zones found.");
                return CompletableFuture.completedFuture(null); // Return
null if no zones are found
            }

            List<String> zoneIds = zonesList.stream()
                .map(AvailabilityZone::zoneId) // Get the zoneId (e.g.,
"usw2-az1")
                .toList();

            return CompletableFuture.supplyAsync(() ->
promptUserForZoneSelection(zonesList, zoneIds))
                .thenApply(selectedZone -> {
                    // Return only the selected Zone ID (e.g., "usw2-az1").
                    return selectedZone.zoneId();
                });
        })
}

```

```

        .whenComplete((result, exception) -> {
            if (exception == null) {
                if (result != null) {
                    logger.info("Selected Availability Zone ID: " + result);
                } else {
                    logger.info("No availability zone selected.");
                }
            } else {
                Throwable cause = exception.getCause();
                if (cause instanceof Ec2Exception) {
                    throw new CompletionException("EC2 error while selecting
availability zone: " + cause.getMessage(), cause);
                }
                throw new CompletionException("Failed to select availability
zone: " + exception.getMessage(), exception);
            }
        });
    }

    /**
     * Prompts the user to select an Availability Zone from the given list.
     *
     * @param zonesList the list of Availability Zones
     * @param zoneIds the list of zone IDs
     * @return the selected Availability Zone
     */
    private static AvailabilityZone
promptUserForZoneSelection(List<AvailabilityZone> zonesList, List<String>
zoneIds) {
        Scanner scanner = new Scanner(System.in);
        int index = -1;

        while (index < 0 || index >= zoneIds.size()) {
            logger.info("Select an availability zone:");
            IntStream.range(0, zoneIds.size()).forEach(i ->
                logger.info(i + ": " + zoneIds.get(i))
            );

            logger.info("Enter the number corresponding to your choice: ");
            if (scanner.hasNextInt()) {
                index = scanner.nextInt();
            } else {
                scanner.next();
            }
        }
    }

```

```

    }

    AvailabilityZone selectedZone = zonesList.get(index);
    logger.info("You selected: " + selectedZone.zoneId());
    return selectedZone;
}

/**
 * Asynchronously sets up a new VPC, including creating the VPC, finding the
 * associated route table, and
 * creating a VPC endpoint for the S3 service.
 *
 * @return a {@link CompletableFuture} that, when completed, contains a
 * AbstractMap with the
 *      VPC ID and VPC endpoint ID.
 */
public CompletableFuture<AbstractMap.SimpleEntry<String, String>>
setupVPCAsync() {
    String cidr = "10.0.0.0/16";
    CreateVpcRequest vpcRequest = CreateVpcRequest.builder()
        .cidrBlock(cidr)
        .build();

    return getEc2AsyncClient().createVpc(vpcRequest)
        .thenCompose(vpcResponse -> {
            String vpcId = vpcResponse.vpc().vpcId();
            logger.info("VPC Created: {}", vpcId);

            Ec2AsyncWaiter waiter = getEc2AsyncClient().waiter();
            DescribeVpcsRequest request = DescribeVpcsRequest.builder()
                .vpcIds(vpcId)
                .build();

            return waiter.waitUntilVpcAvailable(request)
                .thenApply(waiterResponse -> vpcId);
        })
        .thenCompose(vpcId -> {
            Filter filter = Filter.builder()
                .name("vpc-id")
                .values(vpcId)
                .build();

            DescribeRouteTablesRequest describeRouteTablesRequest =
DescribeRouteTablesRequest.builder()

```



```

        .filters(filter)
        .build();

        return
getEc2AsyncClient().describeRouteTables(describeRouteTablesRequest)
        .thenApply(routeTablesResponse -> {
            if (routeTablesResponse.routeTables().isEmpty()) {
                throw new CompletionException("No route tables found
for VPC: " + vpcId, null);
            }
            String routeTableId =
routeTablesResponse.routeTables().get(0).routeTableId();
            logger.info("Route table found: {}", routeTableId);
            return new AbstractMap.SimpleEntry<>(vpcId,
routeTableId);
        });
    })
    .thenCompose(vpcAndRouteTable -> {
        String vpcId = vpcAndRouteTable.getKey();
        String routeTableId = vpcAndRouteTable.getValue();
        Region region =
getEc2AsyncClient().serviceClientConfiguration().region();
        String serviceName = String.format("com.amazonaws.%s.s3express",
region.id());

        CreateVpcEndpointRequest endpointRequest =
CreateVpcEndpointRequest.builder()
            .vpcId(vpcId)
            .routeTableIds(routeTableId)
            .serviceName(serviceName)
            .build();

        return getEc2AsyncClient().createVpcEndpoint(endpointRequest)
            .thenApply(vpcEndpointResponse -> {
                String vpcEndpointId =
vpcEndpointResponse.vpcEndpoint().vpcEndpointId();
                logger.info("VPC Endpoint created: {}", vpcEndpointId);
                return new AbstractMap.SimpleEntry<>(vpcId,
vpcEndpointId);
            });
    })
    .exceptionally(exception -> {
        Throwable cause = exception.getCause() != null ?
exception.getCause() : exception;

```

```
        if (cause instanceof Ec2Exception) {
            logger.error("EC2 error during VPC setup: {}",
                cause.getMessage(), cause);
            throw new CompletionException("EC2 error during VPC setup: "
                + cause.getMessage(), cause);
        }

        logger.error("VPC setup failed: {}", cause.getMessage(), cause);
        throw new CompletionException("VPC setup failed: " +
            cause.getMessage(), cause);
    });
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。

- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObject](#)
- [GetObject](#)
- [ListObjects](#)
- [PutObject](#)

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

執行展示 Amazon S3 目錄儲存貯體和 S3 Express One Zone 基本概念的案例。

```
echo "\n";
echo "-----\n";
```

```

    echo "Welcome to the Amazon S3 Express Basics demo using PHP!\n";
    echo "-----\n";

    // Change these both of these values to use a different region/
availability zone.
    $region = "us-west-2";
    $az = "usw2-az1";

    $this->s3Service = new S3Service(new S3Client(['region' => $region]));
    $this->iamService = new IAMService(new IamClient(['region' => $region]));

    $uuid = uniqid();

    echo <<<INTRO
Let's get started! First, please note that S3 Express One Zone works best when
working within the AWS infrastructure,
specifically when working in the same Availability Zone. To see the best results
in this example, and when you implement
Directory buckets into your infrastructure, it is best to put your Compute
resources in the same AZ as your Directory
bucket.\n
INTRO;

    pressEnter();
    // 1. Configure a gateway VPC endpoint. This is the recommended method to
allow S3 Express One Zone traffic without
    // the need to pass through an internet gateway or NAT device.
    echo "\n";
    echo "1. First, we'll set up a new VPC and VPC Endpoint if this program
is running in an EC2 instance in the same AZ as your Directory buckets will be.
\n";
    $ec2Choice = testable_readline("Are you running this in an EC2 instance
located in the same AZ as your intended Directory buckets? Enter Y/y to setup a
VPC Endpoint, or N/n/blank to skip this section.");
    if($ec2Choice == "Y" || $ec2Choice == "y") {
        echo "Great! Let's set up a VPC, retrieve the Route Table from it,
and create a VPC Endpoint to connect the S3 Client to.\n";
        pressEnter();
        $this->ec2Service = new EC2Service(new Ec2Client(['region' =>
$region]));
        $cidr = "10.0.0.0/16";
        $vpc = $this->ec2Service->createVpc($cidr);
        $this->resources['vpcId'] = $vpc['VpcId'];

        $this->ec2Service->waitForVpcAvailable($vpc['VpcId']);

```

```

        $routeTable = $this->ec2Service->describeRouteTables([], [
            [
                'Name' => "vpc-id",
                'Values' => [$vpc['VpcId']],
            ],
        ]);

        $serviceName = "com.amazonaws." . $this->ec2Service->getRegion() .
            ".s3express";
        $vpcEndpoint = $this->ec2Service->createVpcEndpoint($serviceName,
            $vpc['VpcId'], [$routeTable[0]]);
        $this->resources['vpcEndpointId'] = $vpcEndpoint['VpcEndpointId'];
    }else{
        echo "Skipping the VPC setup. Don't forget to use this in production!
\n";
    }

    // 2. Policies, user, and roles with CDK.
    echo "\n";
    echo "2. Policies, users, and roles with CDK.\n";
    echo "Now, we'll set up some policies, roles, and a user. This user will
only have permissions to do S3 Express One Zone actions.\n";
    pressEnter();

    $this->cloudFormationClient = new CloudFormationClient([]);
    $stackName = "cfn-stack-s3-express-basics-" . uniqid();
    $file = file_get_contents(__DIR__ . "/../..../resources/cfn/
s3_express_basics/s3_express_template.yml");
    $result = $this->cloudFormationClient->createStack([
        'StackName' => $stackName,
        'TemplateBody' => $file,
        'Capabilities' => ['CAPABILITY_IAM'],
    ]);
    $waiter = $this->cloudFormationClient->getWaiter("StackCreateComplete",
        ['StackName' => $stackName]);
    try {
        $waiter->promise()->wait();
    }catch(CloudFormationException $caught){
        echo "Error waiting for the CloudFormation stack to create: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
    $this->resources['stackName'] = $stackName;

```

```

$stackInfo = $this->cloudFormationClient->describeStacks([
    'StackName' => $result['StackId'],
]);

$expressUserName = "";
$regularUserName = "";
foreach($stackInfo['Stacks'][0]['Outputs'] as $output) {
    if ($output['OutputKey'] == "RegularUser") {
        $regularUserName = $output['OutputValue'];
    }
    if ($output['OutputKey'] == "ExpressUser") {
        $expressUserName = $output['OutputValue'];
    }
}
$regularKey = $this->iamService->createAccessKey($regularUserName);
$regularCredentials = new Credentials($regularKey['AccessKeyId'],
$regularKey['SecretAccessKey']);
$expressKey = $this->iamService->createAccessKey($expressUserName);
$expressCredentials = new Credentials($expressKey['AccessKeyId'],
$expressKey['SecretAccessKey']);

// 3. Create an additional client using the credentials with S3 Express
permissions.
echo "\n";
echo "3. Create an additional client using the credentials with S3
Express permissions.\n";
echo "This client is created with the credentials associated with the
user account with the S3 Express policy attached, so it can perform S3 Express
operations.\n";
pressEnter();
$s3RegularClient = new S3Client([
    'Region' => $region,
    'Credentials' => $regularCredentials,
]);
$s3RegularService = new S3Service($s3RegularClient);
$s3ExpressClient = new S3Client([
    'Region' => $region,
    'Credentials' => $expressCredentials,
]);
$s3ExpressService = new S3Service($s3ExpressClient);
echo "All the roles and policies were created an attached to the user.
Then, a new S3 Client and Service were created using that user's credentials.
\n";

```

```
    echo "We can now use this client to make calls to S3 Express operations.
Keeping permissions in mind (and adhering to least-privilege) is crucial to S3
Express.\n";
    pressEnter();

    // 4. Create two buckets.
    echo "\n";
    echo "3. Create two buckets.\n";
    echo "Now we will create a Directory bucket, which is the linchpin of the
S3 Express One Zone service.\n";
    echo "Directory buckets behave in different ways from regular S3 buckets,
which we will explore here.\n";
    echo "We'll also create a normal bucket, put an object into the normal
bucket, and copy it over to the Directory bucket.\n";
    pressEnter();

    // Create a directory bucket. These are different from normal S3 buckets
in subtle ways.
    $directoryBucketName = "s3-express-demo-directory-bucket-$uuid--$az--x-
s3";
    echo "Now, let's create the actual Directory bucket, as well as a regular
bucket.\n";
    pressEnter();
    $s3ExpressService->createBucket($directoryBucketName, [
        'CreateBucketConfiguration' => [
            'Bucket' => [
                'Type' => "Directory", // This is what causes S3 to create a
Directory bucket as opposed to a normal bucket.
                'DataRedundancy' => "SingleAvailabilityZone",
            ],
            'Location' => [
                'Name' => $az,
                'Type' => "AvailabilityZone",
            ],
        ],
    ]);
    $this->resources['directoryBucketName'] = $directoryBucketName;

    // Create a normal bucket.
    $normalBucketName = "normal-bucket-$uuid";
    $s3RegularService->createBucket($normalBucketName);
    $this->resources['normalBucketName'] = $normalBucketName;
    echo "Great! Both buckets were created.\n";
    pressEnter();
```

```
// 5. Create an object and copy it over.
echo "\n";
echo "5. Create an object and copy it over.\n";
echo "We'll create a basic object consisting of some text and upload it
to the normal bucket.\n";
echo "Next, we'll copy the object into the Directory bucket using the
regular client.\n";
echo "This works fine, because Copy operations are not restricted for
Directory buckets.\n";
pressEnter();

$objectKey = "basic-text-object";
$s3RegularService->putObject($normalBucketName, $objectKey, $args =
['Body' => "Look Ma, I'm a bucket!"]);
$this->resources['objectKey'] = $objectKey;

// Create a session to access the directory bucket. The SDK Client will
automatically refresh this as needed.
$s3ExpressService->createSession($directoryBucketName);
$s3ExpressService->copyObject($directoryBucketName, $objectKey,
"$normalBucketName/$objectKey");

echo "It worked! It's important to remember the user permissions when
interacting with Directory buckets.\n";
echo "Instead of validating permissions on every call as normal buckets
do, Directory buckets utilize the user credentials and session token to
validate.\n";
echo "This allows for much faster connection speeds on every call. For
single calls, this is low, but for many concurrent calls, this adds up to a lot
of time saved.\n";
pressEnter();

// 6. Demonstrate performance difference.
echo "\n";
echo "6. Demonstrate performance difference.\n";
$downloads = 1000;
echo "Now, let's do a performance test. We'll download the same object
from each bucket $downloads times and compare the total time needed. Note: the
performance difference will be much more pronounced if this example is run in an
EC2 instance in the same AZ as the bucket.\n";
$downloadChoice = testable_readline("If you would like to download each
object $downloads times, press enter. Otherwise, enter a custom amount and press
enter.");
```

```
if($downloadChoice && is_numeric($downloadChoice) && $downloadChoice <
1000000){ // A million is enough. I promise.
    $downloads = $downloadChoice;
}

// Download the object $downloads times from each bucket and time it to
demonstrate the speed difference.
$directoryStartTime = hrtime(true);
for($i = 0; $i < $downloads; ++$i){
    $s3ExpressService->getObject($directoryBucketName, $objectKey);
}
$directoryEndTime = hrtime(true);
$directoryTimeDiff = $directoryEndTime - $directoryStartTime;

$normalStartTime = hrtime(true);
for($i = 0; $i < $downloads; ++$i){
    $s3RegularService->getObject($normalBucketName, $objectKey);
}
$normalEndTime = hrtime(true);
$normalTimeDiff = $normalEndTime - $normalStartTime;

echo "The directory bucket took $directoryTimeDiff nanoseconds, while the
normal bucket took $normalTimeDiff.\n";
echo "That's a difference of " . ($normalTimeDiff - $directoryTimeDiff) .
" nanoseconds, or " . (($normalTimeDiff - $directoryTimeDiff)/1000000000) . "
seconds.\n";
pressEnter();

// 7. Populate the buckets to show the lexicographical difference.
echo "\n";
echo "7. Populate the buckets to show the lexicographical difference.\n";
echo "Now let's explore how Directory buckets store objects in a
different manner to regular buckets.\n";
echo "The key is in the name \"Directory!\"\n";
echo "Where regular buckets store their key/value pairs in a flat manner,
Directory buckets use actual directories/folders.\n";
echo "This allows for more rapid indexing, traversing, and therefore
retrieval times!\n";
echo "The more segmented your bucket is, with lots of directories, sub-
directories, and objects, the more efficient it becomes.\n";
echo "This structural difference also causes ListObjects to behave
differently, which can cause unexpected results.\n";
echo "Let's add a few more objects with layered directories as see how
the output of ListObjects changes.\n";
```



```
pressEnter();

// Populate a few more files in each bucket so that we can use
ListObjects and show the difference.
$otherObject = "other/$objectKey";
$altObject = "alt/$objectKey";
$otherAltObject = "other/alt/$objectKey";
$s3ExpressService->putObject($directoryBucketName, $otherObject);
$s3RegularService->putObject($normalBucketName, $otherObject);
$this->resources['otherObject'] = $otherObject;
$s3ExpressService->putObject($directoryBucketName, $altObject);
$s3RegularService->putObject($normalBucketName, $altObject);
$this->resources['altObject'] = $altObject;
$s3ExpressService->putObject($directoryBucketName, $otherAltObject);
$s3RegularService->putObject($normalBucketName, $otherAltObject);
$this->resources['otherAltObject'] = $otherAltObject;

$listDirectoryBucket = $s3ExpressService-
>listObjects($directoryBucketName);
$listNormalBucket = $s3RegularService->listObjects($normalBucketName);

// Directory bucket content
echo "Directory bucket content\n";
foreach($listDirectoryBucket['Contents'] as $result){
    echo $result['Key'] . "\n";
}

// Normal bucket content
echo "\nNormal bucket content\n";
foreach($listNormalBucket['Contents'] as $result){
    echo $result['Key'] . "\n";
}

echo "Notice how the normal bucket lists objects in lexicographical
order, while the directory bucket does not. This is because the normal bucket
considers the whole \"key\" to be the object identifies, while the directory
bucket actually creates directories and uses the object \"key\" as a path to the
object.\n";
pressEnter();

echo "\n";
echo "That's it for our tour of the basic operations for S3 Express One
Zone.\n";
```

```
        $cleanUp = testable_readline("Would you like to delete all the resources
created during this demo? Enter Y/y to delete all the resources.");
        if($cleanUp){
            $this->cleanUp();
        }

namespace S3;

use Aws\CommandInterface;
use Aws\Exception\AwsException;
use Aws\Result;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use AwsUtilities\AWSServiceClass;
use DateTimeInterface;

class S3Service extends AWSServiceClass
{
    protected S3Client $client;
    protected bool $verbose;

    public function __construct(S3Client $client = null, $verbose = false)
    {
        if ($client) {
            $this->client = $client;
        } else {
            $this->client = new S3Client([
                'version' => 'latest',
                'region' => 'us-west-2',
            ]);
        }
        $this->verbose = $verbose;
    }

    public function setVerbose($verbose)
    {
        $this->verbose = $verbose;
    }

    public function isVerbose(): bool
    {
        return $this->verbose;
    }
}
```

```
}

public function getClient(): S3Client
{
    return $this->client;
}

public function setClient(S3Client $client)
{
    $this->client = $client;
}

public function emptyAndDeleteBucket($bucketName, array $args = [])
{
    try {
        $objects = $this->listAllObjects($bucketName, $args);
        $this->deleteObjects($bucketName, $objects, $args);
        if ($this->verbose) {
            echo "Deleted all objects and folders from $bucketName.\n";
        }
        $this->deleteBucket($bucketName, $args);
    } catch (AwsException $exception) {
        if ($this->verbose) {
            echo "Failed to delete $bucketName with error: {$exception-
>getMessage()}\n";
            echo "\nPlease fix error with bucket deletion before continuing.
\n";
        }
        throw $exception;
    }
}

public function createBucket(string $bucketName, array $args = [])
{
    $parameters = array_merge(['Bucket' => $bucketName], $args);
    try {
        $this->client->createBucket($parameters);
        if ($this->verbose) {
            echo "Created the bucket named: $bucketName.\n";
        }
    } catch (AwsException $exception) {
```

```
        if ($this->verbose) {
            echo "Failed to create $bucketName with error: {$exception-
>getMessage()}\n";
            echo "Please fix error with bucket creation before continuing.";
        }
        throw $exception;
    }
}

public function putObject(string $bucketName, string $key, array $args = [])
{
    $parameters = array_merge(['Bucket' => $bucketName, 'Key' => $key],
    $args);
    try {
        $this->client->putObject($parameters);
        if ($this->verbose) {
            echo "Uploaded the object named: $key to the bucket named:
$bucketName.\n";
        }
    } catch (AwsException $exception) {
        if ($this->verbose) {
            echo "Failed to create $key in $bucketName with error:
{$exception->getMessage()}\n";
            echo "Please fix error with object uploading before continuing.";
        }
        throw $exception;
    }
}

public function getObject(string $bucketName, string $key, array $args = []):
Result
{
    $parameters = array_merge(['Bucket' => $bucketName, 'Key' => $key],
    $args);
    try {
        $object = $this->client->getObject($parameters);
        if ($this->verbose) {
            echo "Downloaded the object named: $key to the bucket named:
$bucketName.\n";
        }
    }
```

```
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to download $key from $bucketName with error:
{$exception->getMessage()}\n";
                echo "Please fix error with object downloading before
continuing.";
            }
            throw $exception;
        }
        return $object;
    }

    public function copyObject($bucketName, $key, $copySource, array $args = [])
    {
        $parameters = array_merge(['Bucket' => $bucketName, 'Key' => $key,
"CopySource" => $copySource], $args);
        try {
            $this->client->copyObject($parameters);
            if ($this->verbose) {
                echo "Copied the object from: $copySource in $bucketName to:
$key.\n";
            }
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to copy $copySource in $bucketName with error:
{$exception->getMessage()}\n";
                echo "Please fix error with object copying before continuing.";
            }
            throw $exception;
        }
    }

    public function listObjects(string $bucketName, $start = 0, $max = 1000,
array $args = [])
    {
        $parameters = array_merge(['Bucket' => $bucketName, 'Marker' => $start,
"MaxKeys" => $max], $args);
        try {
            $objects = $this->client->listObjectsV2($parameters);
            if ($this->verbose) {
```

```

        echo "Retrieved the list of objects from: $bucketName.\n";
    }
} catch (AwsException $exception) {
    if ($this->verbose) {
        echo "Failed to retrieve the objects from $bucketName with error:
{$exception->getMessage()}\n";
        echo "Please fix error with list objects before continuing.";
    }
    throw $exception;
}
return $objects;
}

public function listAllObjects($bucketName, array $args = [])
{
    $parameters = array_merge(['Bucket' => $bucketName], $args);

    $contents = [];
    $paginator = $this->client->getPaginator("ListObjectsV2", $parameters);

    foreach ($paginator as $result) {
        if($result['KeyCount'] == 0){
            break;
        }
        foreach ($result['Contents'] as $object) {
            $contents[] = $object;
        }
    }
    return $contents;
}

public function deleteObjects(string $bucketName, array $objects, array $args
= [])
{
    $listOfObjects = array_map(
        function ($object) {
            return ['Key' => $object];
        },
        array_column($objects, 'Key')
    );
};

```

```
        if(!$listOfObjects){
            return;
        }

        $parameters = array_merge(['Bucket' => $bucketName, 'Delete' =>
['Objects' => $listOfObjects]], $args);
        try {
            $this->client->deleteObjects($parameters);
            if ($this->verbose) {
                echo "Deleted the list of objects from: $bucketName.\n";
            }
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to delete the list of objects from $bucketName with
error: {$exception->getMessage()}\n";
                echo "Please fix error with object deletion before continuing.";
            }
            throw $exception;
        }
    }

    public function deleteBucket(string $bucketName, array $args = [])
    {
        $parameters = array_merge(['Bucket' => $bucketName], $args);
        try {
            $this->client->deleteBucket($parameters);
            if ($this->verbose) {
                echo "Deleted the bucket named: $bucketName.\n";
            }
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to delete $bucketName with error: {$exception-
>getMessage()}\n";
                echo "Please fix error with bucket deletion before continuing.";
            }
            throw $exception;
        }
    }
}
```

```

    public function deleteObject(string $bucketName, string $fileName, array
$args = [])
    {
        $parameters = array_merge(['Bucket' => $bucketName, 'Key' => $fileName],
$args);
        try {
            $this->client->deleteObject($parameters);
            if ($this->verbose) {
                echo "Deleted the object named: $fileName from $bucketName.\n";
            }
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to delete $fileName from $bucketName with error:
{$exception->getMessage()}\n";
                echo "Please fix error with object deletion before continuing.";
            }
            throw $exception;
        }
    }

    public function listBuckets(array $args = [])
    {
        try {
            $buckets = $this->client->listBuckets($args);
            if ($this->verbose) {
                echo "Retrieved all " . count($buckets) . "\n";
            }
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to retrieve bucket list with error: {$exception-
>getMessage()}\n";
                echo "Please fix error with bucket lists before continuing.";
            }
            throw $exception;
        }
        return $buckets;
    }

    public function preSignedUrl(CommandInterface $command, DateTimeInterface|
int|string $expires, array $options = [])

```



```

    {
        $request = $this->client->createPresignedRequest($command, $expires,
$options);
        try {
            $presignedUrl = (string)$request->getUri();
        } catch (AwsException $exception) {
            if ($this->verbose) {
                echo "Failed to create a presigned url: {$exception-
>getMessage()}\n";
                echo "Please fix error with presigned urls before continuing.";
            }
            throw $exception;
        }
        return $presignedUrl;
    }

    public function createSession(string $bucketName)
    {
        try{
            $result = $this->client->createSession([
                'Bucket' => $bucketName,
            ]);
            return $result;
        }catch(S3Exception $caught){
            if($caught->getAwsErrorType() == "NoSuchBucket"){
                echo "The specified bucket does not exist.";
            }
            throw $caught;
        }
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的下列主題。
- [CopyObject](#)
- [CreateBucket](#)
- [DeleteBucket](#)
- [DeleteObject](#)

- [GetObject](#)
- [ListObjects](#)
- [PutObject](#)

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

執行展示 Amazon S3 目錄儲存貯體和 S3 Express One Zone 基本概念的案例。

```
class S3ExpressScenario:
    """Runs an interactive scenario that shows how to get started with S3
    Express."""

    def __init__(
        self,
        cloud_formation_resource: ServiceResource,
        ec2_client: client,
        iam_client: client,
    ):
        self.cloud_formation_resource = cloud_formation_resource
        self.ec2_client = ec2_client
        self.iam_client = iam_client
        self.region = ec2_client.meta.region_name
        self.stack = None
        self.vpc_id = None
        self.vpc_endpoint_id = None
        self.regular_bucket_name = None
        self.directory_bucket_name = None
        self.s3_express_wrapper = None
        self.s3_regular_wrapper = None

    def s3_express_scenario(self):
        """
        Runs the scenario.
```

```

    """
    print("")
    print_dashes()
    print("Welcome to the Amazon S3 Express Basics demo using Python (Boto
3)!")

```

```

    print_dashes()

```

```

    print(
        """

```

Let's get started! First, please note that S3 Express One Zone works best when working within the AWS infrastructure, specifically when working in the same Availability Zone. To see the best results in this example and when you implement Directory buckets into your infrastructure, it is best to put your compute resources in the same AZ as your Directory bucket.

```

    """

```

```

    )

```

```

    press_enter_to_continue()

```

```

    # Create an optional VPC and create 2 IAM users.

```

```

    express_user_name, regular_user_name = self.create_vpc_and_users()

```

```

    # Set up two S3 clients, one regular and one express, and two buckets,
    one regular and one express.

```

```

    self.setup_clients_and_buckets(express_user_name, regular_user_name)

```

```

    # Create an S3 session for the express S3 client and add objects to the
    buckets.

```

```

    bucket_object = self.create_session_and_add_objects()

```

```

    # Demonstrate performance differences between regular and express
    buckets.

```

```

    self.demonstrate_performance(bucket_object)

```

```

    # Populate the buckets to show the lexicographical difference between
    regular and express buckets.

```

```

    self.show_lexicographical_differences(bucket_object)

```

```

    print("")

```

```

    print("That's it for our tour of the basic operations for S3 Express One
Zone.")

```

```

    if q.ask(

```

```

        "Would you like to delete all the resources created during this demo
(y/n)? ",
        q.is_yesno,
    ):
        self.cleanup()

    def create_vpc_and_users(self) -> None:
        """
        Optionally create a VPC.
        Create two IAM users, one with S3 Express One Zone permissions and one
        without.
        """
        # Configure a gateway VPC endpoint. This is the recommended method to
        allow S3 Express One Zone traffic without
        # the need to pass through an internet gateway or NAT device.
        print(
            """
1. First, we'll set up a new VPC and VPC Endpoint if this program is running in
an EC2 instance in the same AZ as your
Directory buckets will be. Are you running this in an EC2 instance located in the
same AZ as your intended Directory buckets?
"""
        )
        if q.ask("Do you want to setup a VPC Endpoint? (y/n) ", q.is_yesno):
            print(
                "Great! Let's set up a VPC, retrieve the Route Table from it, and
create a VPC Endpoint to connect the S3 Client to."
            )
            self.setup_vpc()
            press_enter_to_continue()
        else:
            print("Skipping the VPC setup. Don't forget to use this in
production!")
            print(
                """
2. Policies, users, and roles with CDK.
Now, we'll set up some policies, roles, and a user. This user will only have
permissions to do S3 Express One Zone actions.
"""
            )
            press_enter_to_continue()
            stack_name = f"cfn-stack-s3-express-basics--{uuid.uuid4()}"
            template_as_string = S3ExpressScenario.get_template_as_string()

```

```

        self.stack = self.deploy_cloudformation_stack(stack_name,
template_as_string)
        regular_user_name = None
        express_user_name = None
        outputs = self.stack.outputs
        for output in outputs:
            if output.get("OutputKey") == "RegularUser":
                regular_user_name = output.get("OutputValue")
            elif output.get("OutputKey") == "ExpressUser":
                express_user_name = output.get("OutputValue")
        if not regular_user_name or not express_user_name:
            error_string = f"""
Failed to retrieve required outputs from CloudFormation stack.
'regular_user_name'={regular_user_name},
'express_user_name'={express_user_name}
"""
            logger.error(error_string)
            raise ValueError(error_string)
        return express_user_name, regular_user_name

def setup_clients_and_buckets(
    self, express_user_name: str, regular_user_name: str
) -> None:
    """
    Set up two S3 clients, one regular and one express, and two buckets, one
    regular and one express.
    :param express_user_name: The name of the user with S3 Express
    permissions.
    :param regular_user_name: The name of the user with regular S3
    permissions.
    """
    regular_credentials = self.create_access_key(regular_user_name)
    express_credentials = self.create_access_key(express_user_name)
    # 3. Create an additional client using the credentials with S3 Express
    permissions.
    print(
        """
3. Create an additional client using the credentials with S3 Express permissions.
This client is created with the
credentials associated with the user account with the S3 Express policy attached,
so it can perform S3 Express operations.
"""
    )
    press_enter_to_continue()

```

```

s3_regular_client = self.create_s3__client_with_access_key_credentials(
    regular_credentials
)
self.s3_regular_wrapper = S3ExpressWrapper(s3_regular_client)
s3_express_client = self.create_s3__client_with_access_key_credentials(
    express_credentials
)
self.s3_express_wrapper = S3ExpressWrapper(s3_express_client)
print(
    """

```

All the roles and policies were created and attached to the user. Then a new S3 Client were created using that user's credentials. We can now use this client to make calls to S3 Express operations. Keeping permissions in mind (and adhering to least-privilege) is crucial to S3 Express.

```

"""
)
press_enter_to_continue()
# 4. Create two buckets.
print(
    """

```

3. Create two buckets.

Now we will create a Directory bucket which is the linchpin of the S3 Express One Zone service. Directory buckets behave in different ways from regular S3 buckets which we will explore here.

We'll also create a normal bucket, put an object into the normal bucket, and copy it over to the Directory bucket.

```

"""
)

# Create a directory bucket. These are different from normal S3 buckets
in subtle ways.
bucket_prefix = q.ask(
    "Enter a bucket name prefix that will be used for both buckets: ",
    q.re_match(r"[a-z0-9](?:[a-z0-9-\.]*)[a-z0-9]$",),
)

# Some availability zones are not supported for Directory buckets. We'll
choose one that is supported.
print(
    "Now, let's choose an availability zone for the Directory bucket.
We'll choose one that is supported."
)
while True:

```

```

        availability_zone = self.select_availability_zone_id(self.region)
        # Construct the parts of a directory bucket name that is made unique
        with a UUID string.
        directory_bucket_suffix = f"--{availability_zone['ZoneId']}--x-s3"
        max_uuid_length = 63 - len(bucket_prefix) -
len(directory_bucket_suffix) - 1
        bucket_uuid = str(uuid.uuid4()).replace("-", "")[:max_uuid_length]
        directory_bucket_name = (
            f"{bucket_prefix}-{bucket_uuid}{directory_bucket_suffix}"
        )
        regular_bucket_name = f"{bucket_prefix}-regular-{bucket_uuid}"
        configuration = {
            "Bucket": {
                "Type": "Directory",
                "DataRedundancy": "SingleAvailabilityZone",
            },
            "Location": {
                "Name": availability_zone["ZoneId"],
                "Type": "AvailabilityZone",
            },
        }
        press_enter_to_continue()
        print(
            "Now, let's create the actual Directory bucket, as well as a
regular bucket."
        )
        press_enter_to_continue()
        try:
            self.s3_express_wrapper.create_bucket(
                directory_bucket_name, configuration
            )
            break
        except ClientError as client_error:
            if client_error.response["Error"]["Code"] == "InvalidBucketName":
                print(
                    f"Bucket '{directory_bucket_name}' is invalid. This may
be because of selected availability zone."
                )
                if q.ask(
                    "Would you like to select a different availability zone?
",
                    q.is_yesno,
                ):
                    continue

```

```

        else:
            raise
    else:
        raise
    print(f"Created directory bucket, '{directory_bucket_name}'")
    self.directory_bucket_name = directory_bucket_name

    self.s3_regular_wrapper.create_bucket(regular_bucket_name)
    print(f"Created regular bucket, '{regular_bucket_name}'")
    self.regular_bucket_name = regular_bucket_name
    print("Great! Both buckets were created.")
    press_enter_to_continue()

def create_session_and_add_objects(self) -> None:
    """
    Create a session for the express S3 client and add objects to the
    buckets.
    """
    print(
        """
5. Create an object and copy it over.
We'll create a basic object consisting of some text and upload it to the normal
    bucket. Next we'll copy the object
into the Directory bucket using the regular client. This works fine because copy
    operations are not restricted for
Directory buckets.
        """
    )
    press_enter_to_continue()
    bucket_object = "basic-text-object"
    self.s3_regular_wrapper.put_object(
        self.regular_bucket_name, bucket_object, "Look Ma, I'm a bucket!"
    )
    self.s3_express_wrapper.create_session(self.directory_bucket_name)
    self.s3_express_wrapper.copy_object(
        self.regular_bucket_name,
        bucket_object,
        self.directory_bucket_name,
        bucket_object,
    )
    print(
        """
It worked! It's important to remember the user permissions when interacting with
Directory buckets. Instead of validating
    """
    )

```


permissions on every call as normal buckets do, Directory buckets utilize the user credentials and session token to validate. This allows for much faster connection speeds on every call. For single calls, this is low, but for many concurrent calls this adds up to a lot of time saved.

```
"""
    )
    press_enter_to_continue()
    return bucket_object

def demonstrate_performance(self, bucket_object: str) -> None:
    """
    Demonstrate performance differences between regular and Directory
    buckets.
    :param bucket_object: The name of the object to download from each
    bucket.
    """
    print("")
    print("6. Demonstrate performance difference.")
    print(
        """
```

Now, let's do a performance test. We'll download the same object from each bucket 'downloads' times and compare the total time needed. Note: the performance difference will be much more pronounced if this example is run in an EC2 instance in the same Availability Zone as the bucket.

```
"""
    )
    downloads = 1000
    print(
        f"The number of downloads of the same object for this example is set
    at {downloads}."
    )
    if q.ask("Would you like to download a different number? (y/n) ",
    q.is_yesno):
        max_downloads = 1000000
        downloads = q.ask(
            f"Enter a number between 1 and {max_downloads} for the number of
    downloads: ",
            q.is_int,
            q.in_range(1, max_downloads),
        )
        # Download the object 'downloads' times from each bucket and time it to
    demonstrate the speed difference.
```

```

print("Downloading from the Directory bucket.")
directory_time_start = time.time_ns()

for index in range(downloads):
    if index % 10 == 0:
        print(f"Download {index} of {downloads}")

    self.s3_express_wrapper.get_object(
        self.directory_bucket_name, bucket_object
    )

directory_time_difference = time.time_ns() - directory_time_start
print("Downloading from the normal bucket.")
normal_time_start = time.time_ns()

for index in range(downloads):
    if index % 10 == 0:
        print(f"Download {index} of {downloads}")
    self.s3_regular_wrapper.get_object(self.regular_bucket_name,
bucket_object)

normal_time_difference = time.time_ns() - normal_time_start
print(
    f"The directory bucket took {directory_time_difference} nanoseconds,
while the normal bucket took {normal_time_difference}."
)
difference = normal_time_difference - directory_time_difference
print(f"That's a difference of {difference} nanoseconds, or")
print(f"{{(difference) / 1000000000}} seconds.")
if difference < 0:
    print(
        "The directory buckets were slower. This can happen if you are
not running on the cloud within a vpc."
    )
    press_enter_to_continue()

def show_lexicographical_differences(self, bucket_object: str) -> None:
    """
    Show the lexicographical difference between Directory buckets and regular
    buckets.

    This is done by creating a few objects in each bucket and listing them to
    show the difference.

    :param bucket_object: The object to use for the listing operations.
    """

```

```

        print(
            """
7. Populate the buckets to show the lexicographical difference.
Now let's explore how Directory buckets store objects in a different manner to
regular buckets. The key is in the name
"Directory". Where regular buckets store their key/value pairs in a flat manner,
Directory buckets use actual
directories/folders. This allows for more rapid indexing, traversing, and
therefore retrieval times! The more segmented
your bucket is, with lots of directories, sub-directories, and objects, the more
efficient it becomes. This structural
difference also causes ListObjects to behave differently, which can cause
unexpected results. Let's add a few more
objects with layered directories to see how the output of ListObjects changes.
            """
        )
        press_enter_to_continue()
        # Populate a few more files in each bucket so that we can use ListObjects
        and show the difference.
        other_object = f"other/{bucket_object}"
        alt_object = f"alt/{bucket_object}"
        other_alt_object = f"other/alt/{bucket_object}"
        self.s3_regular_wrapper.put_object(self.regular_bucket_name,
other_object, "")
        self.s3_express_wrapper.put_object(self.directory_bucket_name,
other_object, "")
        self.s3_regular_wrapper.put_object(self.regular_bucket_name, alt_object,
        "")
        self.s3_express_wrapper.put_object(self.directory_bucket_name,
alt_object, "")
        self.s3_regular_wrapper.put_object(
            self.regular_bucket_name, other_alt_object, ""
        )
        self.s3_express_wrapper.put_object(
            self.directory_bucket_name, other_alt_object, ""
        )
        directory_bucket_objects = self.s3_express_wrapper.list_objects(
            self.directory_bucket_name
        )

        regular_bucket_objects = self.s3_regular_wrapper.list_objects(
            self.regular_bucket_name
        )

```

```

print("Directory bucket content")
for bucket_object in directory_bucket_objects:
    print(f"    {bucket_object['Key']}")
print("Normal bucket content")
for bucket_object in regular_bucket_objects:
    print(f"    {bucket_object['Key']}")
print(
    """

```

Notice how the normal bucket lists objects in lexicographical order, while the directory bucket does not. This is because the normal bucket considers the whole "key" to be the object identifier, while the directory bucket actually creates directories and uses the object "key" as a path to the object.

```

    """
)
press_enter_to_continue()

def cleanup(self) -> None:
    """
    Delete resources created by this scenario.
    """
    if self.directory_bucket_name is not None:
        self.s3_express_wrapper.delete_bucket_and_objects(
            self.directory_bucket_name
        )
        print(f"Deleted directory bucket, '{self.directory_bucket_name}'")
        self.directory_bucket_name = None

    if self.regular_bucket_name is not None:

self.s3_regular_wrapper.delete_bucket_and_objects(self.regular_bucket_name)
        print(f"Deleted regular bucket, '{self.regular_bucket_name}'")
        self.regular_bucket_name = None

    if self.stack is not None:
        self.destroy_cloudformation_stack(self.stack)
        self.stack = None

    self.tear_done_vpc()

def create_access_key(self, user_name: str) -> dict[str, any]:
    """
    Creates an access key for the user.
    :param user_name: The name of the user.

```

```

        :return: The access key for the user.
        """
    try:
        access_key = self.iam_client.create_access_key(UserName=user_name)
        return access_key["AccessKey"]
    except ClientError as client_error:
        logging.error(
            "Couldn't create the access key. Here's why: %s",
            client_error.response["Error"]["Message"],
        )
        raise

def create_s3_client_with_access_key_credentials(
    self, access_key: dict[str, any]
) -> client:
    """
    Creates an S3 client with access key credentials.
    :param access_key: The access key for the user.
    :return: The S3 Express One Zone client.
    """
    try:
        s3_express_client = boto3.client(
            "s3",
            aws_access_key_id=access_key["AccessKeyId"],
            aws_secret_access_key=access_key["SecretAccessKey"],
            region_name=self.region,
        )
        return s3_express_client
    except ClientError as client_error:
        logging.error(
            "Couldn't create the S3 Express One Zone client. Here's why: %s",
            client_error.response["Error"]["Message"],
        )
        raise

def select_availability_zone_id(self, region: str) -> dict[str, any]:
    """
    Selects an availability zone.
    :param region: The region to select the availability zone from.
    :return: The availability zone dictionary.
    """
    try:
        response = self.ec2_client.describe_availability_zones(
            Filters=[{"Name": "region-name", "Values": [region]}]

```

```

        )
        availability_zones = response["AvailabilityZones"]
        zone_names = [zone["ZoneName"] for zone in availability_zones]
        index = q.choose("Select an availability zone: ", zone_names)
        return availability_zones[index]
    except ClientError as client_error:
        logging.error(
            "Couldn't describe availability zones. Here's why: %s",
            client_error.response["Error"]["Message"],
        )
        raise

def deploy_cloudformation_stack(
    self, stack_name: str, cfn_template: str
) -> ServiceResource:
    """
    Deploys prerequisite resources used by the scenario. The resources are
    defined in the associated `cfn_template.yaml` AWS CloudFormation script
    and are deployed
    as a CloudFormation stack, so they can be easily managed and destroyed.

    :param stack_name: The name of the CloudFormation stack.
    :param cfn_template: The CloudFormation template as a string.
    :return: The CloudFormation stack resource.
    """
    print(f"Deploying CloudFormation stack: {stack_name}.")
    stack = self.cloud_formation_resource.create_stack(
        StackName=stack_name,
        TemplateBody=cfn_template,
        Capabilities=["CAPABILITY_NAMED_IAM"],
    )
    print(f"CloudFormation stack creation started: {stack_name}")
    print("Waiting for CloudFormation stack creation to complete...")
    waiter = self.cloud_formation_resource.meta.client.get_waiter(
        "stack_create_complete"
    )
    waiter.wait(StackName=stack.name)
    stack.load()
    print("CloudFormation stack creation complete.")

    return stack

def destroy_cloudformation_stack(self, stack: ServiceResource) -> None:
    """

```

```

        Destroys the resources managed by the CloudFormation stack, and the
        CloudFormation
        stack itself.

        :param stack: The CloudFormation stack that manages the example
        resources.
        """
        try:
            print(
                f"CloudFormation stack '{stack.name}' is being deleted. This may
take a few minutes."
            )
            stack.delete()
            waiter = self.cloud_formation_resource.meta.client.get_waiter(
                "stack_delete_complete"
            )
            waiter.wait(StackName=stack.name)
            print(f"CloudFormation stack '{stack.name}' has been deleted.")
        except ClientError as client_error:
            logging.error(
                "Couldn't delete the CloudFormation stack. Here's why: %s",
                client_error.response["Error"]["Message"],
            )

    @staticmethod
    def get_template_as_string() -> str:
        """
        Returns a string containing this scenario's CloudFormation template.
        """
        script_directory = os.path.dirname(os.path.abspath(__file__))
        template_file_path = os.path.join(script_directory,
"s3_express_template.yaml")
        file = open(template_file_path, "r")
        return file.read()

    def setup_vpc(self):
        cidr = "10.0.0.0/16"
        try:
            response = self.ec2_client.create_vpc(CidrBlock=cidr)
            self.vpc_id = response["Vpc"]["VpcId"]

            waiter = self.ec2_client.get_waiter("vpc_available")
            waiter.wait(VpcIds=[self.vpc_id])
            print(f"Created vpc {self.vpc_id}")

```

```

except ClientError as client_error:
    logging.error(
        "Couldn't create the vpc. Here's why: %s",
        client_error.response["Error"]["Message"],
    )
    raise
try:
    response = self.ec2_client.describe_route_tables(
        Filters=[{"Name": "vpc-id", "Values": [self.vpc_id]}]
    )
    route_table_id = response["RouteTables"][0]["RouteTableId"]
    service_name = f"com.amazonaws.
{self.ec2_client.meta.region_name}.s3express"

    response = self.ec2_client.create_vpc_endpoint(
        VpcId=self.vpc_id,
        RouteTableIds=[route_table_id],
        ServiceName=service_name,
    )
    self.vpc_endpoint_id = response["VpcEndpoint"]["VpcEndpointId"]
    print(f"Created vpc endpoint {self.vpc_endpoint_id}")

except ClientError as client_error:
    logging.error(
        "Couldn't create the vpc endpoint. Here's why: %s",
        client_error.response["Error"]["Message"],
    )
    raise

def tear_down_vpc(self) -> None:
    if self.vpc_endpoint_id is not None:
        try:
            self.ec2_client.delete_vpc_endpoints(
                VpcEndpointIds=[self.vpc_endpoint_id]
            )
            print(f"Deleted vpc endpoint {self.vpc_endpoint_id}.")
            self.vpc_endpoint_id = None
        except ClientError as client_error:
            logging.error(
                "Couldn't delete the vpc endpoint %s. Here's why: %s",
                self.vpc_endpoint_id,
                client_error.response["Error"]["Message"],
            )

```



```

    if self.vpc_id is not None:
        try:
            self.ec2_client.delete_vpc(VpcId=self.vpc_id)
            print(f"Deleted vpc {self.vpc_id}")
            self.vpc_id = None
        except ClientError as client_error:
            logging.error(
                "Couldn't delete the vpc %s. Here's why: %s",
                self.vpc_id,
                client_error.response["Error"]["Message"],
            )

```

Amazon S3 Express SDK 函式的包裝函式類別。

```

class S3ExpressWrapper:
    """Encapsulates Amazon S3 Express One Zone actions using the client
    interface."""

    def __init__(self, s3_client: Any) -> None:
        """
        Initializes the S3ExpressWrapper with an S3 client.

        :param s3_client: A Boto3 Amazon S3 client. This client provides low-
level
                           access to AWS S3 services.
        """
        self.s3_client = s3_client

    @classmethod
    def from_client(cls) -> "S3ExpressWrapper":
        """
        Creates an S3ExpressWrapper instance with a default s3 client.

        :return: An instance of S3ExpressWrapper initialized with the default S3
client.
        """
        s3_client = boto3.client("s3")
        return cls(s3_client)

```

```

def create_bucket(
    self, bucket_name: str, bucket_configuration: dict[str, any] = None
) -> None:
    """
    Creates a bucket.
    :param bucket_name: The name of the bucket.
    :param bucket_configuration: The optional configuration for the bucket.
    """
    try:
        params = {"Bucket": bucket_name}
        if bucket_configuration:
            params["CreateBucketConfiguration"] = bucket_configuration

        self.s3_client.create_bucket(**params)
    except ClientError as client_error:
        # Do not log InvalidBucketName error because it is logged elsewhere.
        if client_error.response["Error"]["Code"] != "InvalidBucketName":
            logging.error(
                "Couldn't create the bucket %s. Here's why: %s",
                bucket_name,
                client_error.response["Error"]["Message"],
            )
            raise

def delete_bucket_and_objects(self, bucket_name: str) -> None:
    """
    Deletes a bucket and its objects.
    :param bucket_name: The name of the bucket.
    """
    try:
        # Delete the objects in the bucket first. This is required for a
        bucket to be deleted.
        paginator = self.s3_client.get_paginator("list_objects_v2")
        page_iterator = paginator.paginate(Bucket=bucket_name)
        for page in page_iterator:
            if "Contents" in page:
                delete_keys = {
                    "Objects": [{"Key": obj["Key"]} for obj in
page["Contents"]]
                }
                response = self.s3_client.delete_objects(
                    Bucket=bucket_name, Delete=delete_keys
                )
                if "Errors" in response:

```

```

        for error in response["Errors"]:
            logging.error(
                "Couldn't delete object %s. Here's why: %s",
                error["Key"],
                error["Message"],
            )

        self.s3_client.delete_bucket(Bucket=bucket_name)
    except ClientError as client_error:
        logging.error(
            "Couldn't delete the bucket %s. Here's why: %s",
            bucket_name,
            client_error.response["Error"]["Message"],
        )

    def put_object(self, bucket_name: str, object_key: str, content: str) ->
None:
        """
        Puts an object into a bucket.
        :param bucket_name: The name of the bucket.
        :param object_key: The key of the object.
        :param content: The content of the object.
        """
        try:
            self.s3_client.put_object(Body=content, Bucket=bucket_name,
Key=object_key)
        except ClientError as client_error:
            logging.error(
                "Couldn't put the object %s into bucket %s. Here's why: %s",
                object_key,
                bucket_name,
                client_error.response["Error"]["Message"],
            )
            raise

    def list_objects(self, bucket: str) -> list[str]:
        """
        Lists objects in a bucket.
        :param bucket: The name of the bucket.
        :return: The list of objects in the bucket.
        """
        try:
            response = self.s3_client.list_objects_v2(Bucket=bucket)
            return response.get("Contents", [])

```

```

        except ClientError as client_error:
            logging.error(
                "Couldn't list objects in bucket %s. Here's why: %s",
                bucket,
                client_error.response["Error"]["Message"],
            )
            raise

    def copy_object(
        self,
        source_bucket: str,
        source_key: str,
        destination_bucket: str,
        destination_key: str,
    ) -> None:
        """
        Copies an object from one bucket to another.
        :param source_bucket: The source bucket.
        :param source_key: The source key.
        :param destination_bucket: The destination bucket.
        :param destination_key: The destination key.
        :return: None
        """
        try:
            self.s3_client.copy_object(
                CopySource={"Bucket": source_bucket, "Key": source_key},
                Bucket=destination_bucket,
                Key=destination_key,
            )
        except ClientError as client_error:
            logging.error(
                "Couldn't copy object %s from bucket %s to bucket %s. Here's why:
%s",
                source_key,
                source_bucket,
                destination_bucket,
                client_error.response["Error"]["Message"],
            )
            raise

    def create_session(self, bucket_name: str) -> None:
        """
        Creates an express session.
        :param bucket_name: The name of the bucket.

```

```
"""
try:
    self.s3_client.create_session(Bucket=bucket_name)
except ClientError as client_error:
    logging.error(
        "Couldn't create the express session for bucket %s. Here's why:
%s",
        bucket_name,
        client_error.response["Error"]["Message"],
    )
    raise

def get_object(self, bucket_name: str, object_key: str) -> None:
    """
    Gets an object from a bucket.
    :param bucket_name: The name of the bucket.
    :param object_key: The key of the object.
    """
    try:
        self.s3_client.get_object(Bucket=bucket_name, Key=object_key)
    except ClientError as client_error:
        logging.error(
            "Couldn't get the object %s from bucket %s. Here's why: %s",
            object_key,
            bucket_name,
            client_error.response["Error"]["Message"],
        )
        raise
```

- 如需 API 詳細資訊，請參閱下列《適用於 Python (Boto3) 的 AWS SDK API 參考》中的下列主題。
 - [CopyObject](#)
 - [CreateBucket](#)
 - [DeleteBucket](#)
 - [DeleteObject](#)
 - [GetObject](#)
 - [ListObjects](#)

- [PutObject](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDKs 的 S3 目錄儲存貯體動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 S3 Directory Buckets 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

這些摘錄會呼叫 S3 目錄儲存貯體 API，是必須在內容中執行之大型程式的程式碼摘錄。您可以在 [使用 AWS SDKs S3 目錄儲存貯體案例](#) 中查看內容中的動作。

下列範例僅包含最常使用的動作。如需完整清單，請參閱《[Amazon S3 目錄儲存貯體 API 參考](#)》。

範例

- [AbortMultipartUpload 搭配 AWS SDK 使用](#)
- [CompleteMultipartUpload 搭配 AWS SDK 使用](#)
- [CopyObject 搭配 AWS SDK 使用](#)
- [CreateBucket 搭配 AWS SDK 使用](#)
- [CreateMultipartUpload 搭配 AWS SDK 使用](#)
- [CreateSession 搭配 AWS SDK 使用](#)
- [DeleteBucket 搭配 AWS SDK 使用](#)
- [DeleteBucketEncryption 搭配 AWS SDK 使用](#)
- [DeleteBucketPolicy 搭配 AWS SDK 使用](#)
- [DeleteObject 搭配 AWS SDK 使用](#)
- [DeleteObjects 搭配 AWS SDK 使用](#)
- [GetBucketEncryption 搭配 AWS SDK 使用](#)
- [GetBucketPolicy 搭配 AWS SDK 使用](#)
- [GetObject 搭配 AWS SDK 使用](#)
- [GetObjectAttributes 搭配 AWS SDK 使用](#)
- [HeadBucket 搭配 AWS SDK 使用](#)
- [HeadObject 搭配 AWS SDK 使用](#)

- [ListDirectoryBuckets 搭配 AWS SDK 使用](#)
- [ListMultipartUploads 搭配 AWS SDK 使用](#)
- [ListObjectsV2 搭配 AWS SDK 使用](#)
- [ListParts 搭配 AWS SDK 使用](#)
- [PutBucketEncryption 搭配 AWS SDK 使用](#)
- [PutBucketPolicy 搭配 AWS SDK 使用](#)
- [PutObject 搭配 AWS SDK 使用](#)
- [UploadPart 搭配 AWS SDK 使用](#)
- [UploadPartCopy 搭配 AWS SDK 使用](#)

AbortMultipartUpload 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 AbortMultipartUpload。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

中止目錄儲存貯體中的分段上傳。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.AbortMultipartUploadRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static
    com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucketMultipartUpload;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
```

```
/**
 * Aborts a specific multipart upload for the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @param objectKey The key (name) of the object to be uploaded
 * @param uploadId The upload ID of the multipart upload to abort
 * @return True if the multipart upload is successfully aborted, false
 otherwise
 */
public static boolean abortDirectoryBucketMultipartUpload(S3Client s3Client,
String bucketName,
String objectKey, String uploadId) {
    logger.info("Aborting multipart upload: {} for bucket: {}", uploadId,
bucketName);
    try {
        // Abort the multipart upload
        AbortMultipartUploadRequest abortMultipartUploadRequest =
AbortMultipartUploadRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .uploadId(uploadId)
            .build();

        s3Client.abortMultipartUpload(abortMultipartUploadRequest);
        logger.info("Aborted multipart upload: {} for object: {}", uploadId,
objectKey);
        return true;
    } catch (S3Exception e) {
        logger.error("Failed to abort multipart upload: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中 [AbortMultipartUpload](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CompleteMultipartUpload 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 CompleteMultipartUpload。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

完成目錄儲存貯體中的分段上傳。

```
import com.example.s3.util.S3DirectoryBucketUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CompleteMultipartUploadRequest;
import software.amazon.awssdk.services.s3.model.CompleteMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.CompletedMultipartUpload;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.io.IOException;
import java.nio.file.Path;
import java.util.List;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static
    com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucketMultipartUpload;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static
    com.example.s3.util.S3DirectoryBucketUtils.deleteAllObjectsInDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.multipartUploadForDirectoryBucket;
```

```
/**
 * This method completes the multipart upload request by collating all the
 * upload parts.
 *
 * @param s3Client    The S3 client used to interact with S3
 * @param bucketName  The name of the directory bucket
 * @param objectKey   The key (name) of the object to be uploaded
 * @param uploadId    The upload ID used to track the multipart upload
 * @param uploadParts The list of completed parts
 * @return True if the multipart upload is successfully completed, false
 *         otherwise
 */
public static boolean completeDirectoryBucketMultipartUpload(S3Client
s3Client, String bucketName, String objectKey,
    String uploadId, List<CompletedPart> uploadParts) {
    try {
        CompletedMultipartUpload completedMultipartUpload =
CompletedMultipartUpload.builder()
            .parts(uploadParts)
            .build();

        CompleteMultipartUploadRequest completeMultipartUploadRequest =
CompleteMultipartUploadRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .uploadId(uploadId)
            .multipartUpload(completedMultipartUpload)
            .build();

        CompleteMultipartUploadResponse response =
s3Client.completeMultipartUpload(completeMultipartUploadRequest);
        logger.info("Multipart upload completed. ETag: {}", response.eTag());
        return true;
    } catch (S3Exception e) {
        logger.error("Failed to complete multipart upload: {} - Error code:
{}", e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CompleteMultipartUpload](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CopyObject 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 CopyObject。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

將物件從一個目錄儲存貯體複製到另一個目錄儲存貯體。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CopyObjectRequest;
import software.amazon.awssdk.services.s3.model.CopyObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.nio.file.Path;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
```

```
import static
    com.example.s3.util.S3DirectoryBucketUtils.deleteAllObjectsInDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.putDirectoryBucketObject;

/**
 * Copies an object from one S3 general purpose bucket to one S3 directory
 * bucket.
 *
 * @param s3Client      The S3 client used to interact with S3
 * @param sourceBucket The name of the source bucket
 * @param objectKey     The key (name) of the object to be copied
 * @param targetBucket The name of the target bucket
 */
public static void copyDirectoryBucketObject(S3Client s3Client, String
sourceBucket, String objectKey,
        String targetBucket) {
    logger.info("Copying object: {} from bucket: {} to bucket: {}",
objectKey, sourceBucket, targetBucket);

    try {
        // Create a CopyObjectRequest
        CopyObjectRequest copyReq = CopyObjectRequest.builder()
            .sourceBucket(sourceBucket)
            .sourceKey(objectKey)
            .destinationBucket(targetBucket)
            .destinationKey(objectKey)
            .build();

        // Copy the object
        CopyObjectResponse copyRes = s3Client.copyObject(copyReq);
        logger.info("Successfully copied {} from bucket {} into bucket {}.
CopyObjectResponse: {}",
            objectKey, sourceBucket, targetBucket,
copyRes.copyObjectResult().toString());

    } catch (S3Exception e) {
        logger.error("Failed to copy object: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        throw e;
    }
}
```

```
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CopyObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateBucket 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 CreateBucket。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立 S3 目錄儲存貯體。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.BucketInfo;
import software.amazon.awssdk.services.s3.model.BucketType;
import software.amazon.awssdk.services.s3.model.CreateBucketConfiguration;
import software.amazon.awssdk.services.s3.model.CreateBucketRequest;
import software.amazon.awssdk.services.s3.model.CreateBucketResponse;
import software.amazon.awssdk.services.s3.model.DataRedundancy;
import software.amazon.awssdk.services.s3.model.LocationInfo;
```

```
import software.amazon.awssdk.services.s3.model.LocationType;
import software.amazon.awssdk.services.s3.model.S3Exception;

import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;

/**
 * Creates a new S3 directory bucket in a specified Zone (For example, a
 * specified Availability Zone in this code example).
 *
 * @param s3Client The S3 client used to create the bucket
 * @param bucketName The name of the bucket to be created
 * @param zone The region where the bucket will be created
 * @throws S3Exception if there's an error creating the bucket
 */
public static void createDirectoryBucket(S3Client s3Client, String
bucketName, String zone) throws S3Exception {
    logger.info("Creating bucket: {}", bucketName);

    CreateBucketConfiguration bucketConfiguration =
CreateBucketConfiguration.builder()
        .location(LocationInfo.builder()
            .type(LocationType.AVAILABILITY_ZONE)
            .name(zone).build())
        .bucket(BucketInfo.builder()
            .type(BucketType.DIRECTORY)
            .dataRedundancy(DataRedundancy.SINGLE_AVAILABILITY_ZONE)
            .build())
        .build();

    try {
        CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
            .bucket(bucketName)
            .createBucketConfiguration(bucketConfiguration).build();
        CreateBucketResponse response = s3Client.createBucket(bucketRequest);
        logger.info("Bucket created successfully with location: {}",
response.location());
    } catch (S3Exception e) {
        logger.error("Error creating bucket: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        throw e;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CreateBucket](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateMultipartUpload 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 CreateMultipartUpload。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

建立目錄儲存貯體中的分段上傳。

```
import com.example.s3.util.S3DirectoryBucketUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadRequest;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;

/**
 * This method creates a multipart upload request that generates a unique
 * upload
 * ID used to track
 * all the upload parts.
 */
```

```
* @param s3Client    The S3 client used to interact with S3
* @param bucketName The name of the directory bucket
* @param objectKey   The key (name) of the object to be uploaded
* @return The upload ID used to track the multipart upload
*/
public static String createDirectoryBucketMultipartUpload(S3Client s3Client,
String bucketName, String objectKey) {
    logger.info("Creating multipart upload for object: {} in bucket: {}",
objectKey, bucketName);

    try {
        // Create a CreateMultipartUploadRequest
        CreateMultipartUploadRequest createMultipartUploadRequest =
CreateMultipartUploadRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .build();

        // Initiate the multipart upload
        CreateMultipartUploadResponse response =
s3Client.createMultipartUpload(createMultipartUploadRequest);
        String uploadId = response.uploadId();
        logger.info("Multipart upload initiated. Upload ID: {}", uploadId);
        return uploadId;

    } catch (S3Exception e) {
        logger.error("Failed to create multipart upload: {} - Error code:
{}", e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        throw e;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CreateMultipartUpload](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateSession 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 CreateSession。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class S3ExpressWrapper:
    """Encapsulates Amazon S3 Express One Zone actions using the client
    interface."""

    def __init__(self, s3_client: Any) -> None:
        """
        Initializes the S3ExpressWrapper with an S3 client.

        :param s3_client: A Boto3 Amazon S3 client. This client provides low-
level
                           access to AWS S3 services.
        """
        self.s3_client = s3_client

    @classmethod
    def from_client(cls) -> "S3ExpressWrapper":
        """
        Creates an S3ExpressWrapper instance with a default s3 client.

        :return: An instance of S3ExpressWrapper initialized with the default S3
client.
        """
        s3_client = boto3.client("s3")
        return cls(s3_client)

    def create_session(self, bucket_name: str) -> None:
        """
        Creates an express session.
        :param bucket_name: The name of the bucket.
        """
        try:
```

```
        self.s3_client.create_session(Bucket=bucket_name)
    except ClientError as client_error:
        logging.error(
            "Couldn't create the express session for bucket %s. Here's why:
%s",
            bucket_name,
            client_error.response["Error"]["Message"],
        )
        raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [CreateSession](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteBucket 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 DeleteBucket。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除 S3 目錄儲存貯體。

```
import org.slf4j.Logger;
```

```
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.DeleteBucketRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;

/**
 * Deletes the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket to delete
 */
public static void deleteDirectoryBucket(S3Client s3Client, String
bucketName) {
    logger.info("Deleting bucket: {}", bucketName);

    try {
        // Create a DeleteBucketRequest
        DeleteBucketRequest deleteBucketRequest =
DeleteBucketRequest.builder()
        .bucket(bucketName)
        .build();

        // Delete the bucket
        s3Client.deleteBucket(deleteBucketRequest);
        logger.info("Successfully deleted bucket: {}", bucketName);

    } catch (S3Exception e) {
        logger.error("Failed to delete bucket: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
        e.awsErrorDetails().errorCode(), e);
        throw e;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteBucket](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteBucketEncryption 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 DeleteBucketEncryption。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除目錄儲存貯體的加密組態。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.DeleteBucketEncryptionRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;

/**
 * Deletes the encryption configuration from an S3 bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 */
public static void deleteDirectoryBucketEncryption(S3Client s3Client, String
bucketName) {
    DeleteBucketEncryptionRequest deleteRequest =
DeleteBucketEncryptionRequest.builder()
        .bucket(bucketName)
        .build();

    try {
        s3Client.deleteBucketEncryption(deleteRequest);
        logger.info("Bucket encryption deleted for bucket: {}", bucketName);
    }
```

```
    } catch (S3Exception e) {  
        logger.error("Failed to delete bucket encryption: {} - Error code:  
        {}", e.awsErrorDetails().errorMessage(),  
            e.awsErrorDetails().errorCode(), e);  
        throw e;  
    }  
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteBucketEncryption](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteBucketPolicy 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 DeleteBucketPolicy。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除目錄儲存體的儲存體政策。

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.s3.S3Client;  
import software.amazon.awssdk.services.s3.model.DeleteBucketPolicyRequest;  
import software.amazon.awssdk.services.s3.model.S3Exception;  
  
import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;  
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;  
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;  
import static com.example.s3.util.S3DirectoryBucketUtils.getAwsAccountId;
```

```
import static
    com.example.s3.util.S3DirectoryBucketUtils.putDirectoryBucketPolicy;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

/**
 * Deletes the bucket policy for the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 */
public static void deleteDirectoryBucketPolicy(S3Client s3Client, String
bucketName) {
    logger.info("Deleting policy for bucket: {}", bucketName);

    try {
        // Create a DeleteBucketPolicyRequest
        DeleteBucketPolicyRequest deletePolicyReq =
DeleteBucketPolicyRequest.builder()
            .bucket(bucketName)
            .build();

        // Delete the bucket policy
        s3Client.deleteBucketPolicy(deletePolicyReq);
        logger.info("Successfully deleted bucket policy");

    } catch (S3Exception e) {
        logger.error("Failed to delete bucket policy: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        throw e;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteBucketPolicy](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteObject 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 DeleteObject。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除取得目錄儲存貯體中的物件。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.DeleteObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.nio.file.Path;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.putDirectoryBucketObject;
```

```
/**
 * Deletes an object from the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @param objectKey The key (name) of the object to be deleted
 */
public static void deleteDirectoryBucketObject(S3Client s3Client, String
bucketName, String objectKey) {
    logger.info("Deleting object: {} from bucket: {}", objectKey,
bucketName);

    try {
        // Create a DeleteObjectRequest
        DeleteObjectRequest deleteObjectRequest =
DeleteObjectRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .build();

        // Delete the object
        s3Client.deleteObject(deleteObjectRequest);
        logger.info("Object {} has been deleted", objectKey);

    } catch (S3Exception e) {
        logger.error("Failed to delete object: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        throw e;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteObjects 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 DeleteObjects。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

刪除目錄儲存貯體中的多個物件。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.Delete;
import software.amazon.awssdk.services.s3.model.DeleteObjectsRequest;
import software.amazon.awssdk.services.s3.model.DeleteObjectsResponse;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.net.URISyntaxException;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.List;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.putDirectoryBucketObject;

/**
 * Deletes multiple objects from the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @param objectKeys The list of keys (names) of the objects to be deleted
```

```
    */
    public static void deleteDirectoryBucketObjects(S3Client s3Client, String
bucketName, List<String> objectKeys) {
        logger.info("Deleting objects from bucket: {}", bucketName);

        try {
            // Create a list of ObjectIdentifier.
            List<ObjectIdentifier> identifiers = objectKeys.stream()
                .map(key -> ObjectIdentifier.builder().key(key).build())
                .toList();

            Delete delete = Delete.builder()
                .objects(identifiers)
                .build();

            DeleteObjectsRequest deleteObjectsRequest =
DeleteObjectsRequest.builder()
                .bucket(bucketName)
                .delete(delete)
                .build();

            DeleteObjectsResponse deleteObjectsResponse =
s3Client.deleteObjects(deleteObjectsRequest);
            deleteObjectsResponse.deleted().forEach(deleted ->
logger.info("Deleted object: {}", deleted.key()));

        } catch (S3Exception e) {
            logger.error("Failed to delete objects: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
                e.awsErrorDetails().errorCode(), e);
            throw e;
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteObjects](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetBucketEncryption 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 GetBucketEncryption。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得目錄儲存貯體的加密組態。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetBucketEncryptionRequest;
import software.amazon.awssdk.services.s3.model.GetBucketEncryptionResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.ServerSideEncryptionRule;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;

/**
 * Retrieves the encryption configuration for an S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @return The type of server-side encryption applied to the bucket (e.g.,
 *         AES256, aws:kms)
 */
public static String getDirectoryBucketEncryption(S3Client s3Client, String
bucketName) {
    try {
        // Create a GetBucketEncryptionRequest
        GetBucketEncryptionRequest getRequest =
GetBucketEncryptionRequest.builder()
            .bucket(bucketName)
            .build();
```

```
// Retrieve the bucket encryption configuration
GetBucketEncryptionResponse response =
s3Client.getBucketEncryption(getRequest);
ServerSideEncryptionRule rule =
response.getServerSideEncryptionConfiguration().rules().get(0);

String encryptionType =
rule.applyServerSideEncryptionByDefault().sseAlgorithmAsString();
logger.info("Bucket encryption algorithm: {}", encryptionType);
logger.info("KMS Customer Managed Key ID: {}",
rule.applyServerSideEncryptionByDefault().kmsMasterKeyID());
logger.info("Bucket Key Enabled: {}", rule.bucketKeyEnabled());

return encryptionType;
} catch (S3Exception e) {
    logger.error("Failed to get bucket encryption: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
        e.awsErrorDetails().errorCode(), e);
    throw e;
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetBucketEncryption](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetBucketPolicy 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 GetBucketPolicy。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得目錄儲存貯體的政策。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetBucketPolicyRequest;
import software.amazon.awssdk.services.s3.model.GetBucketPolicyResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getAwsAccountId;
import static
    com.example.s3.util.S3DirectoryBucketUtils.putDirectoryBucketPolicy;

/**
 * Retrieves the bucket policy for the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @return The bucket policy text
 */
public static String getDirectoryBucketPolicy(S3Client s3Client, String
bucketName) {
    logger.info("Getting policy for bucket: {}", bucketName);

    try {
        // Create a GetBucketPolicyRequest
        GetBucketPolicyRequest policyReq = GetBucketPolicyRequest.builder()
            .bucket(bucketName)
            .build();

        // Retrieve the bucket policy
        GetBucketPolicyResponse response =
s3Client.getBucketPolicy(policyReq);

        // Print and return the policy text
        String policyText = response.policy();
        logger.info("Bucket policy: {}", policyText);
        return policyText;
    }
```

```
    } catch (S3Exception e) {  
        logger.error("Failed to get bucket policy: {} - Error code: {}",  
            e.awsErrorDetails().errorMessage(),  
            e.awsErrorDetails().errorCode(), e);  
        throw e;  
    }  
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetBucketPolicy](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetObject 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 GetObject。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)
- [建立預先簽章的 URL 以取得物件](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得儲存貯體的物件。

```
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import software.amazon.awssdk.core.ResponseBytes;  
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.nio.charset.StandardCharsets;
import java.nio.file.Path;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static
    com.example.s3.util.S3DirectoryBucketUtils.deleteAllObjectsInDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.putDirectoryBucketObject;

/**
 * Retrieves an object from the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @param objectKey The key (name) of the object to be retrieved
 * @return The retrieved object as a ResponseInputStream
 */
public static boolean getDirectoryBucketObject(S3Client s3Client, String
bucketName, String objectKey) {
    logger.info("Retrieving object: {} from bucket: {}", objectKey,
bucketName);

    try {
        // Create a GetObjectRequest
        GetObjectRequest objectRequest = GetObjectRequest.builder()
            .key(objectKey)
            .bucket(bucketName)
            .build();

        // Retrieve the object as bytes
        ResponseBytes<GetObjectResponse> objectBytes =
s3Client.getObjectAsBytes(objectRequest);
        byte[] data = objectBytes.asByteArray();

        // Print object contents to console
        String objectContent = new String(data, StandardCharsets.UTF_8);
```

```
        logger.info("Object contents: \n{}", objectContent);

        return true;

    } catch (S3Exception e) {
        logger.error("Failed to retrieve object: {} - Error code: {}",
            e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetObjectAttributes 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 GetObjectAttributes。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得目錄儲存貯體的物件屬性。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectAttributesRequest;
import software.amazon.awssdk.services.s3.model.GetObjectAttributesResponse;
import software.amazon.awssdk.services.s3.model.ObjectAttributes;
```



```
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.nio.file.Path;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static
    com.example.s3.util.S3DirectoryBucketUtils.deleteAllObjectsInDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.putDirectoryBucketObject;

/**
 * Retrieves attributes for an object in the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @param objectKey The key (name) of the object to retrieve attributes for
 * @return True if the object attributes are successfully retrieved, false
 *         otherwise
 */
public static boolean getDirectoryBucketObjectAttributes(S3Client s3Client,
String bucketName, String objectKey) {
    logger.info("Retrieving attributes for object: {} from bucket: {}",
objectKey, bucketName);

    try {
        // Create a GetObjectAttributesRequest
        GetObjectAttributesRequest getObjectAttributesRequest =
GetObjectAttributesRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .objectAttributes(ObjectAttributes.E_TAG,
ObjectAttributes.STORAGE_CLASS,
                ObjectAttributes.OBJECT_SIZE)
            .build();

        // Retrieve the object attributes
        GetObjectAttributesResponse response =
s3Client.getObjectAttributes(getObjectAttributesRequest);
        logger.info("Attributes for object {}:", objectKey);
        logger.info("ETag: {}", response.eTag());
        logger.info("Storage Class: {}", response.storageClass());
    }
}
```

```
        logger.info("Object Size: {}", response.objectSize());
        return true;

    } catch (S3Exception e) {
        logger.error("Failed to retrieve object attributes: {} - Error code:
{}",
                    e.awsErrorDetails().errorMessage(),
e.awsErrorDetails().errorCode(), e);
        return false;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetObjectAttributes](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

HeadBucket 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 HeadBucket。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

檢查指定的 S3 目錄儲存貯體是否存在且可存取。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.HeadBucketRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
```

```
import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;

/**
 * Checks if the specified S3 directory bucket exists and is accessible.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket to check
 * @return True if the bucket exists and is accessible, false otherwise
 */
public static boolean headDirectoryBucket(S3Client s3Client, String
bucketName) {
    logger.info("Checking if bucket exists: {}", bucketName);

    try {
        // Create a HeadBucketRequest
        HeadBucketRequest headBucketRequest = HeadBucketRequest.builder()
            .bucket(bucketName)
            .build();
        // If the bucket doesn't exist, the following statement throws
        NoSuchBucketException,
        // which is a subclass of S3Exception.
        s3Client.headBucket(headBucketRequest);
        logger.info("Amazon S3 directory bucket: \"{}\" found.", bucketName);
        return true;

    } catch (S3Exception e) {
        logger.error("Failed to access bucket: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        throw e;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [HeadBucket](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

HeadObject 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 HeadObject。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

取得目錄儲存貯體中物件的中繼資料。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.nio.file.Path;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static
    com.example.s3.util.S3DirectoryBucketUtils.deleteAllObjectsInDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.putDirectoryBucketObject;

/**
 * Retrieves metadata for an object in the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @param objectKey The key (name) of the object to retrieve metadata for
 * @return True if the object exists, false otherwise
```

```
    */
    public static boolean headDirectoryBucketObject(S3Client s3Client, String
bucketName, String objectKey) {
        logger.info("Retrieving metadata for object: {} from bucket: {}",
objectKey, bucketName);

        try {
            // Create a HeadObjectRequest
            HeadObjectRequest headObjectRequest = HeadObjectRequest.builder()
                .bucket(bucketName)
                .key(objectKey)
                .build();

            // Retrieve the object metadata
            HeadObjectResponse response = s3Client.headObject(headObjectRequest);
            logger.info("Amazon S3 object: \"{}\" found in bucket: \"{}\" with
ETag: \"{}\"", objectKey, bucketName,
                response.eTag());
            logger.info("Content-Type: {}", response.contentType());
            logger.info("Content-Length: {}", response.contentLength());
            logger.info("Last Modified: {}", response.lastModified());
            return true;

        } catch (S3Exception e) {
            logger.error("Failed to retrieve object metadata: {} - Error code:
{}", e.awsErrorDetails().errorMessage(),
                e.awsErrorDetails().errorCode(), e);
            return false;
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [HeadObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListDirectoryBuckets 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 ListDirectoryBuckets。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

列出所有目錄儲存貯體。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.Bucket;
import software.amazon.awssdk.services.s3.model.ListDirectoryBucketsRequest;
import software.amazon.awssdk.services.s3.model.ListDirectoryBucketsResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.util.List;
import java.util.UUID;
import java.util.stream.Collectors;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;

/**
 * Lists all S3 directory buckets and no general purpose buckets.
 *
 * @param s3Client The S3 client used to interact with S3
 * @return A list of bucket names
 */
public static List<String> listDirectoryBuckets(S3Client s3Client) {
    logger.info("Listing all directory buckets");

    try {
        // Create a ListBucketsRequest
        ListDirectoryBucketsRequest listDirectoryBucketsRequest =
            ListDirectoryBucketsRequest.builder().build();
```

```
// Retrieve the list of buckets
ListDirectoryBucketsResponse response =
s3Client.listDirectoryBuckets(listDirectoryBucketsRequest);

// Extract bucket names
List<String> bucketNames = response.buckets().stream()
    .map(Bucket::name)
    .collect(Collectors.toList());

return bucketNames;
} catch (S3Exception e) {
    logger.error("Failed to list buckets: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
        e.awsErrorDetails().errorCode());
    throw e;
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListDirectoryBuckets](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListMultipartUploads 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 ListMultipartUploads。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

列出目錄儲存貯體中的分段上傳。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListMultipartUploadsRequest;
import software.amazon.awssdk.services.s3.model.ListMultipartUploadsResponse;
import software.amazon.awssdk.services.s3.model.MultipartUpload;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.io.IOException;
import java.nio.file.Path;
import java.util.List;

import static
    com.example.s3.util.S3DirectoryBucketUtils.abortDirectoryBucketMultipartUploads;
import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static
    com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucketMultipartUpload;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.multipartUploadForDirectoryBucket;

/**
 * Lists multipart uploads for the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @return A list of MultipartUpload objects representing the multipart
 uploads
 */
public static List<MultipartUpload>
listDirectoryBucketMultipartUploads(S3Client s3Client, String bucketName) {
    logger.info("Listing in-progress multipart uploads for bucket: {}",
bucketName);

    try {
        // Create a ListMultipartUploadsRequest
        ListMultipartUploadsRequest listMultipartUploadsRequest =
ListMultipartUploadsRequest.builder()
            .bucket(bucketName)
            .build();
```



```
// List the multipart uploads
ListMultipartUploadsResponse response =
s3Client.listMultipartUploads(listMultipartUploadsRequest);
List<MultipartUpload> uploads = response.uploads();
for (MultipartUpload upload : uploads) {
    logger.info("In-progress multipart upload: Upload ID: {}, Key:
{}, Initiated: {}", upload.uploadId(),
        upload.key(), upload.initiated());
}
return uploads;

} catch (S3Exception e) {
    logger.error("Failed to list multipart uploads: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
        e.awsErrorDetails().errorCode());
    return List.of(); // Return an empty list if an exception is thrown
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListMultipartUploads](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListObjectsV2 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 ListObjectsV2。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

列出目錄儲存貯體中的物件。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.S3Object;

import java.nio.file.Path;
import java.util.List;
import java.util.stream.Collectors;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static
    com.example.s3.util.S3DirectoryBucketUtils.deleteAllObjectsInDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.putDirectoryBucketObject;

/**
 * Lists objects in the specified S3 directory bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @return A list of object keys in the bucket
 */
public static List<String> listDirectoryBucketObjectsV2(S3Client s3Client,
String bucketName) {
    logger.info("Listing objects in bucket: {}", bucketName);

    try {
        // Create a ListObjectsV2Request
        ListObjectsV2Request listObjectsV2Request =
ListObjectsV2Request.builder()
            .bucket(bucketName)
            .build();

        // Retrieve the list of objects
```

```
ListObjectsV2Response response =
s3Client.listObjectsV2(listObjectsV2Request);

// Extract and return the object keys
return response.contents().stream()
    .map(S3Object::key)
    .collect(Collectors.toList());

} catch (S3Exception e) {
    logger.error("Failed to list objects: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
    e.awsErrorDetails().errorCode());
    throw e;
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListObjectsV2](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListParts 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 ListParts。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

列出目錄儲存貯體中分段上傳的部分。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListPartsRequest;
import software.amazon.awssdk.services.s3.model.ListPartsResponse;
import software.amazon.awssdk.services.s3.model.Part;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.io.IOException;
import java.nio.file.Path;
import java.util.List;

import static
    com.example.s3.util.S3DirectoryBucketUtils.abortDirectoryBucketMultipartUploads;
import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static
    com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucketMultipartUpload;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.multipartUploadForDirectoryBucket;

/**
 * Lists the parts of a multipart upload for the specified S3 directory
 * bucket.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @param objectKey The key (name) of the object being uploaded
 * @param uploadId The upload ID used to track the multipart upload
 * @return A list of Part representing the parts of the multipart upload
 */
public static List<Part> listDirectoryBucketMultipartUploadParts(S3Client
s3Client, String bucketName,
    String objectKey, String uploadId) {
    logger.info("Listing parts for object: {} in bucket: {}", objectKey,
bucketName);

    try {
        // Create a ListPartsRequest
        ListPartsRequest listPartsRequest = ListPartsRequest.builder()
            .bucket(bucketName)
            .uploadId(uploadId)
            .key(objectKey)
            .build();
```

```
// List the parts of the multipart upload
ListPartsResponse response = s3Client.listParts(listPartsRequest);
List<Part> parts = response.parts();
for (Part part : parts) {
    logger.info("Uploaded part: Part number = \"{}\", etag = {}",
part.partNumber(), part.eTag());
}
return parts;

} catch (S3Exception e) {
    logger.error("Failed to list parts: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
e.awsErrorDetails().errorCode());
    return List.of(); // Return an empty list if an exception is thrown
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListParts](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketEncryption 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 PutBucketEncryption。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

將儲存貯體加密設定為目錄儲存貯體。

```
import org.slf4j.Logger;
```

```
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.kms.KmsClient;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutBucketEncryptionRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.ServerSideEncryption;
import software.amazon.awssdk.services.s3.model.ServerSideEncryptionByDefault;
import
    software.amazon.awssdk.services.s3.model.ServerSideEncryptionConfiguration;
import software.amazon.awssdk.services.s3.model.ServerSideEncryptionRule;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createKmsClient;
import static com.example.s3.util.S3DirectoryBucketUtils.createKmsKey;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.scheduleKeyDeletion;

/**
 * Sets the default encryption configuration for an S3 bucket as SSE-KMS.
 *
 * @param s3Client The S3 client used to interact with S3
 * @param bucketName The name of the directory bucket
 * @param kmsKeyId The ID of the customer-managed KMS key
 */
public static void putDirectoryBucketEncryption(S3Client s3Client, String
bucketName, String kmsKeyId) {
    // Define the default encryption configuration to use SSE-KMS. For
directory
    // buckets, AWS managed KMS keys aren't supported. Only customer-managed
keys
    // are supported.
    ServerSideEncryptionByDefault encryptionByDefault =
ServerSideEncryptionByDefault.builder()
        .sseAlgorithm(ServerSideEncryption.AWS_KMS)
        .kmsMasterKeyID(kmsKeyId)
        .build();

    // Create a server-side encryption rule to apply the default encryption
    // configuration. For directory buckets, the bucketKeyEnabled field is
enforced
    // to be true.
    ServerSideEncryptionRule rule = ServerSideEncryptionRule.builder()
        .bucketKeyEnabled(true)
```

```
        .applyServerSideEncryptionByDefault(encryptionByDefault)
        .build();

// Create the server-side encryption configuration for the bucket
ServerSideEncryptionConfiguration encryptionConfiguration =
ServerSideEncryptionConfiguration.builder()
    .rules(rule)
    .build();

// Create the PutBucketEncryption request
PutBucketEncryptionRequest putRequest =
PutBucketEncryptionRequest.builder()
    .bucket(bucketName)
    .serverSideEncryptionConfiguration(encryptionConfiguration)
    .build();

// Set the bucket encryption
try {
    s3Client.putBucketEncryption(putRequest);
    logger.info("SSE-KMS Bucket encryption configuration set for the
directory bucket: {}", bucketName);
} catch (S3Exception e) {
    logger.error("Failed to set bucket encryption: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
        e.awsErrorDetails().errorCode());
    throw e;
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutBucketEncryption](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutBucketPolicy 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 PutBucketPolicy。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

將儲存貯體政策套用至目錄儲存貯體。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutBucketPolicyRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getAwsAccountId;

/**
 * Sets the following bucket policy for the specified S3 directory bucket.
 * <pre>
 * {
 *     "Version": "2012-10-17",
 *     "Statement": [
 *         {
 *             "Sid": "AdminPolicy",
 *             "Effect": "Allow",
 *             "Principal": {
 *                 "AWS": "arn:aws:iam::<ACCOUNT_ID>:root"
 *             },
 *             "Action": "s3express:*",
 *             "Resource": "arn:aws:s3express:us-west-2:<ACCOUNT_ID>:bucket/
 * <DIR_BUCKET_NAME>
 *         }
 *     ]
 * }
```



```

* </pre>
* This policy grants all S3 directory bucket actions to identities in the
same account as the bucket.
*
* @param s3Client The S3 client used to interact with S3
* @param bucketName The name of the directory bucket
* @param policyText The policy text to be applied
*/
public static void putDirectoryBucketPolicy(S3Client s3Client, String
bucketName, String policyText) {
    logger.info("Setting policy on bucket: {}", bucketName);
    logger.info("Policy: {}", policyText);

    try {
        PutBucketPolicyRequest policyReq = PutBucketPolicyRequest.builder()
            .bucket(bucketName)
            .policy(policyText)
            .build();

        s3Client.putBucketPolicy(policyReq);
        logger.info("Bucket policy set successfully!");

    } catch (S3Exception e) {
        logger.error("Failed to set bucket policy: {} - Error code: {}",
            e.awsErrorDetails().errorMessage(),
            e.awsErrorDetails().errorCode(), e);
        throw e;
    }
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutBucketPolicy](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutObject 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 PutObject。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

將物件置入目錄儲存貯體。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.awscore.exception.AwsErrorDetails;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;

import java.io.UncheckedIOException;
import java.nio.file.Path;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static
    com.example.s3.util.S3DirectoryBucketUtils.deleteAllObjectsInDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;

/**
 * Puts an object into the specified S3 directory bucket.
 *
 * @param s3Client    The S3 client used to interact with S3
 * @param bucketName  The name of the directory bucket
 * @param objectKey   The key (name) of the object to be placed in the bucket
 * @param filePath    The path of the file to be uploaded
 */
```

```
public static void putDirectoryBucketObject(S3Client s3Client, String
bucketName, String objectKey, Path filePath) {
    logger.info("Putting object: {} into bucket: {}", objectKey, bucketName);

    try {
        // Create a PutObjectRequest
        PutObjectRequest putObj = PutObjectRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .build();

        // Upload the object
        s3Client.putObject(putObj, filePath);
        logger.info("Successfully placed {} into bucket {}", objectKey,
bucketName);

    } catch (UncheckedIOException e) {
        throw S3Exception.builder().message("Failed to read the file: " +
e.getMessage()).cause(e)
            .awsErrorDetails(AwsErrorDetails.builder()
                .errorCode("ClientSideException:FailedToReadFile")
                .errorMessage(e.getMessage())
                .build())
            .build();
    } catch (S3Exception e) {
        logger.error("Failed to put object: {}", e.getMessage(), e);
        throw e;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UploadPart 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 UploadPart。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

上傳目錄儲存體的分段上傳的一部分。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.UploadPartRequest;
import software.amazon.awssdk.services.s3.model.UploadPartResponse;

import java.io.IOException;
import java.io.RandomAccessFile;
import java.nio.ByteBuffer;
import java.nio.file.Path;
import java.util.ArrayList;
import java.util.List;

import static
    com.example.s3.util.S3DirectoryBucketUtils.abortDirectoryBucketMultipartUploads;
import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static
    com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucketMultipartUpload;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static
    com.example.s3.util.S3DirectoryBucketUtils.deleteAllObjectsInDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;

/**
 * This method creates part requests and uploads individual parts to S3.
```

```

    * While it uses the UploadPart API to upload a single part, it does so
    * sequentially to handle multiple parts of a file, returning all the
    completed
    * parts.
    *
    * @param s3Client    The S3 client used to interact with S3
    * @param bucketName  The name of the directory bucket
    * @param objectKey   The key (name) of the object to be uploaded
    * @param uploadId    The upload ID used to track the multipart upload
    * @param filePath    The path to the file to be uploaded
    * @return A list of uploaded parts
    * @throws IOException if an I/O error occurs
    */
    public static List<CompletedPart> multipartUploadForDirectoryBucket(S3Client
s3Client, String bucketName,
        String objectKey, String uploadId, Path filePath) throws IOException
    {
        logger.info("Uploading parts for object: {} in bucket: {}", objectKey,
bucketName);

        int partNumber = 1;
        List<CompletedPart> uploadedParts = new ArrayList<>();
        ByteBuffer bb = ByteBuffer.allocate(1024 * 1024 * 5); // 5 MB byte buffer

        // Read the local file, break down into chunks and process
        try (RandomAccessFile file = new RandomAccessFile(filePath.toFile(),
"r")) {
            long fileSize = file.length();
            int position = 0;

            // Sequentially upload parts of the file
            while (position < fileSize) {
                file.seek(position);
                int read = file.getChannel().read(bb);

                bb.flip(); // Swap position and limit before reading from the
buffer

                UploadPartRequest uploadPartRequest = UploadPartRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .uploadId(uploadId)
                    .partNumber(partNumber)
                    .build();

```

```
UploadPartResponse partResponse = s3Client.uploadPart(
    uploadPartRequest,
    RequestBody.fromByteBuffer(bb));

// Build the uploaded part
CompletedPart uploadedPart = CompletedPart.builder()
    .partNumber(partNumber)
    .eTag(partResponse.eTag())
    .build();

// Add the uploaded part to the list
uploadedParts.add(uploadedPart);

// Log to indicate the part upload is done
logger.info("Uploaded part number: {} with ETag: {}", partNumber,
partResponse.eTag());

    bb.clear();
    position += read;
    partNumber++;
}
} catch (S3Exception e) {
    logger.error("Failed to list parts: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
        e.awsErrorDetails().errorCode());
    throw e;
}
return uploadedParts;
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [UploadPart](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UploadPartCopy 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 UploadPartCopy。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

根據來源物件大小建立複製零件，並透過個別零件複製到目錄儲存貯體。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.UploadPartCopyRequest;
import software.amazon.awssdk.services.s3.model.UploadPartCopyResponse;

import java.io.IOException;
import java.nio.file.Path;
import java.util.ArrayList;
import java.util.List;

import static
    com.example.s3.util.S3DirectoryBucketUtils.abortDirectoryBucketMultipartUploads;
import static
    com.example.s3.util.S3DirectoryBucketUtils.completeDirectoryBucketMultipartUpload;
import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static
    com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucketMultipartUpload;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static
    com.example.s3.util.S3DirectoryBucketUtils.deleteAllObjectsInDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.multipartUploadForDirectoryBucket;
```

```
/**
 * Creates copy parts based on source object size and copies over individual
 * parts.
 *
 * @param s3Client      The S3 client used to interact with S3
 * @param sourceBucket  The name of the source bucket
 * @param sourceKey      The key (name) of the source object
 * @param destinationBucket The name of the destination bucket
 * @param destinationKey The key (name) of the destination object
 * @param uploadId      The upload ID used to track the multipart upload
 * @return A list of completed parts
 */
public static List<CompletedPart>
multipartUploadCopyForDirectoryBucket(S3Client s3Client, String sourceBucket,
    String sourceKey, String destinationBucket, String destinationKey,
    String uploadId) {
    // Get the object size to track the end of the copy operation
    HeadObjectRequest headObjectRequest = HeadObjectRequest.builder()
        .bucket(sourceBucket)
        .key(sourceKey)
        .build();
    HeadObjectResponse headObjectResponse =
s3Client.headObject(headObjectRequest);
    long objectSize = headObjectResponse.contentLength();

    logger.info("Source Object size: {}", objectSize);

    // Copy the object using 20 MB parts
    long partSize = 20 * 1024 * 1024; // 20 MB
    long bytePosition = 0;
    int partNum = 1;
    List<CompletedPart> uploadedParts = new ArrayList<>();

    while (bytePosition < objectSize) {
        long lastByte = Math.min(bytePosition + partSize - 1, objectSize -
1);
        logger.info("Part Number: {}, Byte Position: {}, Last Byte: {}",
partNum, bytePosition, lastByte);

        try {
            UploadPartCopyRequest uploadPartCopyRequest =
UploadPartCopyRequest.builder()
                .sourceBucket(sourceBucket)
```



```

        .sourceKey(sourceKey)
        .destinationBucket(destinationBucket)
        .destinationKey(destinationKey)
        .uploadId(uploadId)
        .copySourceRange("bytes=" + bytePosition + "-" +
lastByte)

        .partNumber(partNum)
        .build();
        UploadPartCopyResponse uploadPartCopyResponse =
s3Client.uploadPartCopy(uploadPartCopyRequest);

        CompletedPart part = CompletedPart.builder()
            .partNumber(partNum)
            .eTag(uploadPartCopyResponse.copyPartResult().eTag())
            .build();
        uploadedParts.add(part);

        bytePosition += partSize;
        partNum++;
    } catch (S3Exception e) {
        logger.error("Failed to copy part number {}: {} - Error code:
{}", partNum,
            e.awsErrorDetails().errorMessage(),
e.awsErrorDetails().errorCode());
        throw e;
    }
}

return uploadedParts;
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [UploadPartCopy](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDKs S3 目錄儲存貯體案例

下列程式碼範例說明如何在 S3 Directory Buckets AWS SDKs 中實作常見案例。這些案例示範如何呼叫 S3 目錄儲存貯體中的多個函數，或與其他 AWS 服務結合，藉以完成特定任務。每個案例均包含完整原始碼的連結，您可在連結中找到如何設定和執行程式碼的相關指示。

案例的目標是獲得中等水平的經驗，協助您了解內容中的服務動作。

範例

- [為 Amazon S3 目錄儲存貯體建立預先簽章的 URL，以使用 AWS SDK 取得物件](#)

為 Amazon S3 目錄儲存貯體建立預先簽章的 URL，以使用 AWS SDK 取得物件

下列程式碼範例示範如何建立適用於 S3 目錄儲存貯體的 URL，並取得物件。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

產生預先簽章的 GET URL，以存取 S3 目錄儲存貯體中的物件。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;
import
    software.amazon.awssdk.services.s3.presigner.model.GetObjectPresignRequest;
import
    software.amazon.awssdk.services.s3.presigner.model.PresignedGetObjectRequest;

import java.nio.file.Path;
import java.time.Duration;

import static com.example.s3.util.S3DirectoryBucketUtils.createDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Client;
import static com.example.s3.util.S3DirectoryBucketUtils.createS3Presigner;
```

```
import static
    com.example.s3.util.S3DirectoryBucketUtils.deleteAllObjectsInDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.deleteDirectoryBucket;
import static com.example.s3.util.S3DirectoryBucketUtils.getFilePath;
import static
    com.example.s3.util.S3DirectoryBucketUtils.putDirectoryBucketObject;

/**
 * Generates a presigned URL for accessing an object in the specified S3
 * directory bucket.
 *
 * @param s3Presigner The S3 presigner client used to generate the presigned
URL
 * @param bucketName The name of the directory bucket
 * @param objectKey The key (name) of the object to access
 * @return A presigned URL for accessing the specified object
 */
public static String generatePresignedGetURLForDirectoryBucket(S3Presigner
s3Presigner, String bucketName,
    String objectKey) {
    logger.info("Generating presigned URL for object: {} in bucket: {}",
objectKey, bucketName);

    try {
        // Create a GetObjectRequest
        GetObjectRequest getObjectRequest = GetObjectRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .build();

        // Create a GetObjectPresignRequest
        GetObjectPresignRequest getObjectPresignRequest =
GetObjectPresignRequest.builder()
            .signatureDuration(Duration.ofMinutes(10)) // Presigned URL
valid for 10 minutes
            .getObjectRequest(getObjectRequest)
            .build();

        // Generate the presigned URL
        PresignedGetObjectRequest presignedGetObjectRequest =
s3Presigner.presignGetObject(getObjectPresignRequest);

        // Get the presigned URL
```

```
String presignedURL = presignedGetObjectRequest.url().toString();
logger.info("Presigned URL: {}", presignedURL);
return presignedURL;

} catch (S3Exception e) {
    logger.error("Failed to generate presigned URL: {} - Error code: {}",
e.awsErrorDetails().errorMessage(),
        e.awsErrorDetails().errorCode(), e);
    throw e;
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [使用 AWS SDKs 使用 Amazon S3 進行開發](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

驗證請求AWS (簽章第 4 版)

主題

- [身分驗證方法](#)
- [簽署請求簡介](#)
- [驗證請求：使用授權標頭AWS \(簽章第 4 版 \)](#)
- [驗證請求：使用查詢參數AWS \(簽章第 4 版 \)](#)
- [範例：簽章第 4 版中的 AWS 簽章計算](#)
- [驗證請求：使用 POST 以瀏覽器為基礎的上傳AWS \(簽章第 4 版 \)](#)
- [Amazon S3 Signature 第 4 版身分驗證特定政策金鑰](#)

與 Amazon S3 的每次互動，可以經過驗證身分或是匿名進行。本節說明使用 AWS Signature 第 4 版演算法請求身分驗證。

Note

如果您使用 AWS SDKs (請參閱[範例程式碼和程式庫](#)) 來傳送請求，則不需要閱讀本節，因為 SDK 用戶端會使用您提供的存取金鑰來驗證您的請求。除非您有充分理由不這麼做，否則您應該一律使用 AWS SDKs。在支援兩種簽章版本的區域中，您可以請求 AWS SDKs 使用特定的簽章版本。如需詳細資訊，請參閱[使用 AWS SDKs 傳送已驗證的請求](#)。只有在自訂用戶端中實作 AWS Signature 第 4 版演算法時，才需要閱讀本節。

使用 AWS Signature 第 4 版進行身分驗證可提供以下部分或全部內容，具體取決於您選擇簽署請求的方式：

- 驗證請求者的身分 – 驗證的請求需要您使用存取金鑰（存取金鑰 ID、私密存取金鑰）建立的簽章。如需有關取得存取金鑰的資訊，請參閱《[IAM 使用者指南](#)》中的[了解和取得您的安全登入資料](#)。如果您使用暫時安全憑證，則簽章計算還需要安全字符。如需詳細資訊，請參閱《IAM 使用者指南》中的[請求臨時安全憑證](#)。
- 傳輸中資料保護 – 為了防止在傳輸中竄改請求，您可以使用一些請求元素來計算請求簽章。收到請求時，Amazon S3 會使用相同的請求元素來計算簽章。如果 Amazon S3 收到的任何請求元件不符合用於計算簽章的元件，Amazon S3 將拒絕請求。

- 避免重複使用請求的已簽署部分 – 請求的已簽署部分（使用 AWS Signatures）在請求中的時間戳記的 15 分鐘內有效。有權存取已簽署請求的未經授權方可以修改請求的未簽署部分，而不會影響請求在 15 分鐘內的有效性。因此，我們建議您簽署請求標頭和內文、向 Amazon S3 提出 HTTPS 請求，以及使用 AWS 政策中的 `s3:x-amz-content-sha256` 條件金鑰（請參閱 [Amazon S3 Signature 第 4 版身分驗證特定政策金鑰](#)）要求使用者簽署 Amazon S3 請求內文，以最大化保護。

Note

Amazon S3 支援 Signature 第 4 版，這是一種通訊協定，用於驗證對 AWS 服務的傳入 API 請求 AWS 區域。目前，在 2014 年 1 月 30 日之前 AWS 區域建立的將繼續支援先前的通訊協定 Signature 第 2 版。2014 年 1 月 30 日之後的任何新區域將僅支援 Signature 第 4 版，因此所有對這些區域的請求都必須使用 Signature 第 4 版提出。如需 AWS Signature 第 2 版的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[簽署和驗證 REST 請求](#)。

身分驗證方法

您可以使用下列其中一種方法來表達身分驗證資訊：

- HTTP 授權標頭 – 使用 HTTP Authorization 標頭是驗證 Amazon S3 請求最常見的方法。所有 Amazon S3 REST 操作（使用 POST 請求的瀏覽器型上傳除外）都需要此標頭。如需 Authorization 標頭值以及如何計算簽章和相關選項的詳細資訊，請參閱 [驗證請求：使用授權標頭AWS（簽章第 4 版）](#)。
- 查詢字串參數 – 您可以使用查詢字串完全在 URL 中表達請求。在此情況下，您可以使用查詢參數來提供請求資訊，包括驗證資訊。由於請求簽章是 URL 的一部分，因此此類型 URL 通常稱為預先簽章的 URL。您可以使用預先簽章URLs，以 HTML 內嵌可點選連結，有效期間最長可達七天。如需詳細資訊，請參閱[驗證請求：使用查詢參數AWS（簽章第 4 版）](#)。

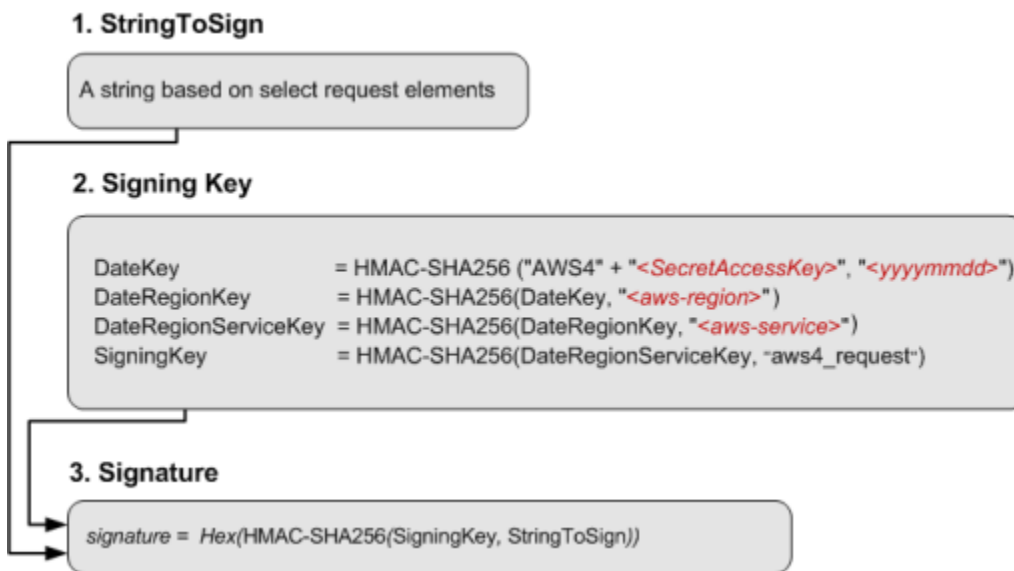
Amazon S3 也支援使用 HTTP POST 請求的瀏覽器型上傳。使用 HTTP POST 請求，您可以直接從瀏覽器將內容上傳至 Amazon S3。如需有關驗證 POST 請求的資訊，請參閱 [使用 POST 的瀏覽器型上傳AWS（簽章第 4 版）](#)。

簽署請求簡介

您在請求中傳送的驗證資訊必須包含簽章。若要計算簽章，請先串連選取請求元素以形成字串，稱為要簽署的字串。然後，您可以使用簽署金鑰來計算要簽署的字串的雜湊訊息驗證碼 (HMAC)。

在 AWS Signature 第 4 版中，您不使用您的私密存取金鑰來簽署請求。相反，首先使用私密存取金鑰來衍生簽署金鑰。衍生的簽署金鑰根據日期、服務和區域而特定。如需如何以不同程式設計語言衍生簽署金鑰的詳細資訊，請參閱[如何衍生 Signature 第 4 版簽署金鑰的範例](#)。

下圖說明計算簽章的一般程序。



要簽署的字串取決於請求類型。例如，當您使用 HTTP 授權標頭或查詢參數進行驗證時，請使用不同的請求元素組合來建立要簽署的字串。對於 HTTP POST 請求，請求中的 POST 政策是您簽署的字串。如需有關要簽署的運算字串的詳細資訊，請遵循本節結尾提供的連結。

對於簽署金鑰，圖表顯示了一系列的計算，然後將每一步驟的結果饋送到下一個步驟。最後的步驟是簽署金鑰。

收到已驗證的請求時，Amazon S3 伺服器會使用請求中包含的身分驗證資訊重新建立簽章。如果簽章相符，Amazon S3 會處理您的請求；否則，請求會遭到拒絕。

如需驗證請求的詳細資訊，請參閱下列主題：

- [驗證請求：使用授權標頭AWS \(簽章第 4 版\)](#)
- [驗證請求：使用查詢參數AWS \(簽章第 4 版\)](#)
- [使用 POST 的瀏覽器型上傳AWS \(簽章第 4 版\)](#)

驗證請求：使用授權標頭AWS（簽章第 4 版）

主題

- [概觀](#)
- [授權標頭的簽章計算：在單一區塊中傳輸承載AWS（簽章第 4 版）](#)
- [授權標頭的簽章計算：以多個區塊傳輸承載（區塊上傳）AWS（簽章版本 4）](#)
- [結尾標頭的簽章計算（區塊上傳）AWS（簽章第 4 版）](#)

概觀

使用 HTTP Authorization 標頭是提供身分驗證資訊的最常見方法。除了使用查詢參數簽署的 [POST 請求](#) 和請求之外，所有 Amazon S3 操作都會使用 Authorization 請求標頭來提供身分驗證資訊。

以下是 Authorization 標頭值的範例。為了便於閱讀，向此範例新增了分行符號：

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

下表說明上述範例中 Authorization 標頭值的各種元件：

元件	說明
AWS4-HMAC-SHA256	用於計算簽章的演算法。當您使用 AWS Signature 第 4 版進行身分驗證時，必須提供此值。 字串指定 AWS Signature 第 4 版 (AWS4) 和簽署演算法 (HMAC-SHA256)。
Credential	您的存取金鑰 ID 和範圍資訊，包括用於計算簽章的日期、區域和服務。 此字串的格式如下：

元件	說明
	<pre><your-access-key-id> /<date>/<aws-region> /<aws-service> /aws4_request</pre> 其中： • <date> 值是使用 YYYYMMDD 格式指定。 • <aws-service> 值是將請求傳送至 Amazon S3 s3 時。
SignedHeaders	用於計算 Signature 的請求標頭清單 (以分號分隔)。此清單僅包含標頭名稱，且標頭名稱必須為小寫。例如： <pre>host;range;x-amz-date</pre>
Signature	256 位元簽章以 64 個小寫十六進位字元表示。例如： <pre>fe5f80f77d5fa3beca038a248ff027d0445342fe2855d dc963176630326f1024</pre> 請注意，簽章計算因您選擇傳輸承載的選項而異。

簽章計算會根據您選擇傳輸請求承載的方法而有所不同。S3 支援下列選項：

- 在單一區塊中傳輸承載 – 在此情況下，您有下列簽章計算選項：
- 已簽章的承載選項 – 您可以選擇性地計算整個承載檢查總和，並將其包含在簽章計算中。這可提高安全性，但您需要讀取承載兩次或將其緩衝在記憶體中。

例如，若要上傳檔案，您需要先讀取檔案，以計算用於簽章計算的承載雜湊，並在建立請求時再次進行傳輸。對於較小的承載，此方法可能較適合。不過，對於大型檔案，讀取檔案兩次可能會效率不佳，因此您可能想要改為以區塊形式上傳資料。

我們建議您包含承載檢查總和，以提高安全性。

- 未簽章的承載選項 – 請勿在簽章計算中包含承載檢查總和。

如需計算簽章和建構授權標頭值的step-by-step說明，請參閱 [授權標頭的簽章計算：在單一區塊中傳輸承載AWS（簽章第 4 版）](#)。

- 在多個區塊中傳輸承載（區塊上傳）– 在此情況下，您會在區塊中傳輸承載。無論承載大小為何，您都可以以區塊傳輸承載。

您可以將承載分解為區塊。這些可以是固定或可變大小的區塊。透過在區塊中上傳資料，您可以避免讀取整個承載來計算簽章。相反地，對於第一個區塊，您可以計算僅使用請求標頭的種子簽章。第二個區塊包含第一個區塊的簽章，而每個後續區塊包含其前面區塊的簽章。在上傳結束時，您會傳送包含 0 個位元組資料的最終區塊，其中包含承載最後一個區塊的簽章。如需詳細資訊，請參閱[授權標頭的簽章計算：以多個區塊傳輸承載（區塊上傳）AWS（簽章版本 4）](#)。

簽署請求時，您可以使用 AWS Signature 第 4 版或 AWS Signature 第 4A 版。兩者之間的主要差異取決於簽章的計算方式。使用 AWS 第 4A 版簽署程序，簽章不包含區域特定的資訊，而是使用 AWS4-ECDSA-P256-SHA256 演算法計算。

除了這些選項之外，您還可以選擇在請求中包含預告片。若要在請求中包含尾隨，您需要透過將 `x-amz-content-sha256` 設定為適當的值，在標頭中指定。如果您使用的是結尾標頭，則必須在標頭 `x-amz-trailer` 中包含，並在逗號分隔清單中將結尾標頭名稱指定為字串。所有結尾標頭都會在最後一個區塊之後寫入。如果您要在多個區塊中上傳資料，您必須先傳送含有 0 個位元組資料的最終區塊，再傳送結尾標頭。

當您傳送請求時，您必須透過使用下列其中一個值新增 `x-amz-content-sha256` 標頭，來告知 Amazon S3 您在簽章計算中選擇的上述選項：

標頭值	說明
實際承載檢查總和值	此值是物件的實際檢查總和，只有在您以單一區塊上傳資料時，才能這麼做。
UNSIGNED-PAYLOAD	當您將物件上傳為單一未簽署區塊時，請使用此選項。
STREAMING-UNSIGNED-PAYLOAD-TRAILER	透過多個區塊傳送未簽章的承載時，請使用此選項。在這種情況下，在上傳區塊之後，您也會有一個結尾標頭。

標頭值	說明
STREAMING-AWS4-HMAC-SHA256-PAYLOAD	在透過多個區塊傳送承載時使用此值，並使用 簽署區塊AWS4-HMAC-SHA256 。這會產生 SigV4 簽章。
STREAMING-AWS4-HMAC-SHA256-PAYLOAD-TRAILER	在透過多個區塊傳送承載時使用此值，並使用 簽署區塊AWS4-HMAC-SHA256 。這會產生 SigV4 簽章。此外，區塊的摘要會納入為結尾標頭。
STREAMING-AWS4-ECDSA-P256-SHA256-PAYLOAD	在透過多個區塊傳送承載時使用此值，並使用 簽署區塊AWS4-ECDSA-P256-SHA256 。這會產生 SigV4A 簽章。
STREAMING-AWS4-ECDSA-P256-SHA256-PAYLOAD-TRAILER	在透過多個區塊傳送承載時使用此值，並使用 簽署區塊AWS4-ECDSA-P256-SHA256 。這會產生 SigV4A 簽章。此外，區塊的摘要會納入為結尾標頭。

收到請求時，Amazon S3 會使用 Authorization標頭和 date標頭中的資訊，重新建立要簽署的字串。然後，它會使用與簽章相符的身分驗證服務進行驗證。您可以使用 HTTP Date或 x-amz-date標頭來指定請求日期。如果兩個標頭都存在，x-amz-date則 優先。

如果簽章相符，Amazon S3 會處理您的請求；否則，您的請求將會失敗。

如需詳細資訊，請參閱下列主題：

[授權標頭的簽章計算：在單一區塊中傳輸承載AWS（簽章第 4 版）](#)

[授權標頭的簽章計算：以多個區塊傳輸承載（區塊上傳）AWS（簽章版本 4）](#)

[結尾標頭的簽章計算（區塊上傳）AWS（簽章第 4 版）](#)

授權標頭的簽章計算：在單一區塊中傳輸承載AWS（簽章第 4 版）

使用 Authorization 標頭來驗證請求時，標頭值包含簽章等等。簽章計算取決於您傳輸承載的選擇 ([概觀](#))。本節說明當您選擇在單一區塊中傳輸承載時的簽章計算。範例區段（請參閱 [範例：簽章計算](#)）顯示簽章計算和產生的Authorization標頭，您可以用作測試套件來驗證程式碼。

Important

在單一區塊中傳輸承載時，您可以選擇在簽章計算中包含承載雜湊，稱為已簽章的承載（如果您未包含，則承載會被視為未簽章）。以下章節討論的簽署程序適用於兩者，但請注意下列差異：

- 已簽章的承載選項 – 您在建構正式請求時包含承載雜湊（然後會成為 `StringToSign` 的一部分，如簽章計算一節所述）。傳送請求至 S3 時，您也可以指定與 `x-amz-content-sha256` 標頭值相同的值。
- 未簽署的承載選項 – 您在建構正式請求 `UNSIGNED-PAYLOAD` 時包含常值字串，並在將請求傳送至 Amazon S3 時設定與 `x-amz-content-sha256` 標頭值相同的值。

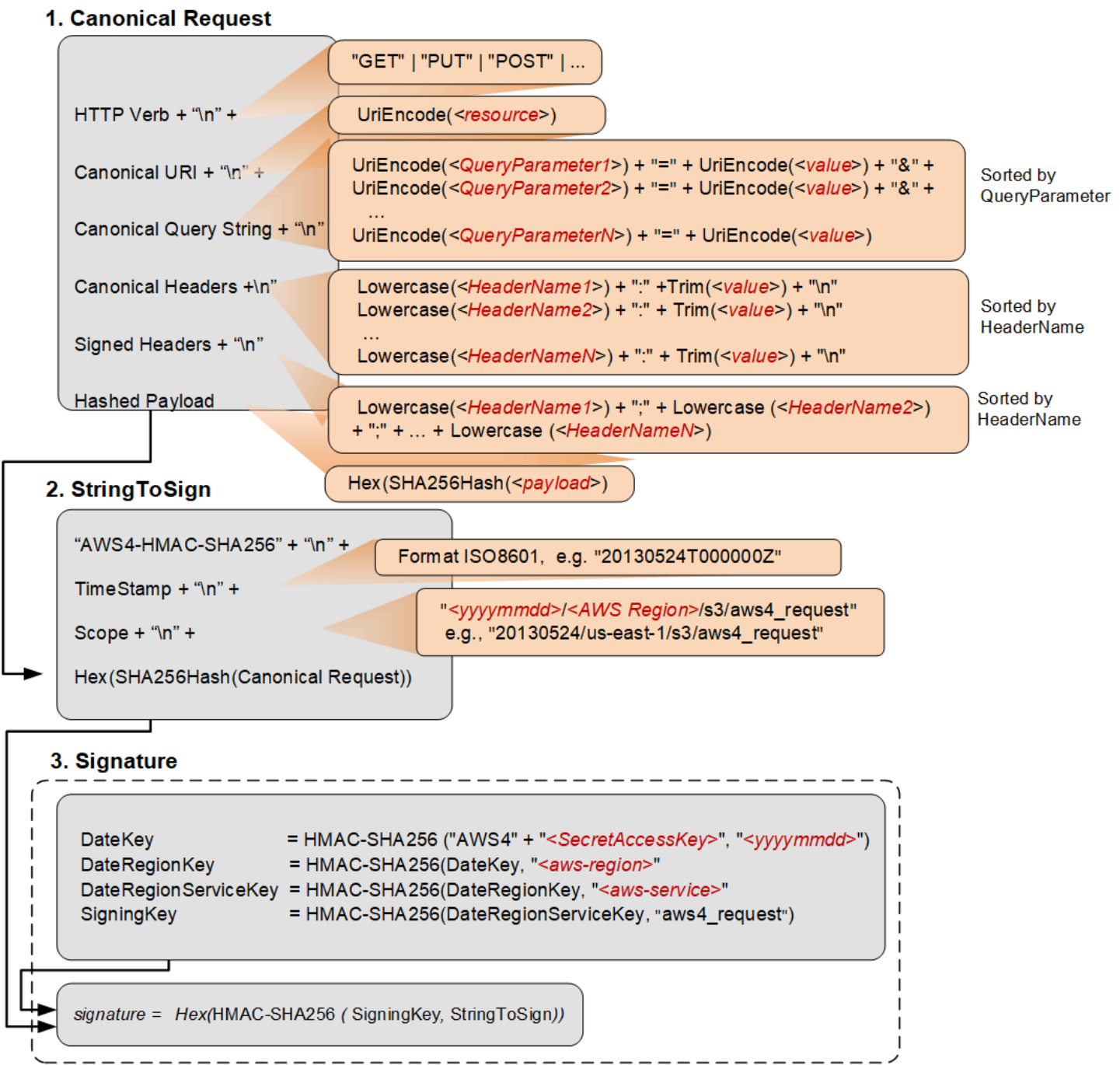
當您將請求傳送至 Amazon S3 時，`x-amz-content-sha256` 標頭值會通知 Amazon S3 承載是否已簽署。然後，Amazon S3 可以相應地建立簽章以進行驗證。

在這兩種情況下，由於 `x-amz-content-sha256` 標頭值已經是 `HashedPayload` 的一部分，因此您不需要將 `x-amz-content-sha256` 標頭包含為正式標頭。

計算簽章

若要計算簽章，首先需要一個要簽名的字串。然後，您可以使用簽署金鑰計算要簽署的字串雜 `HMAC-SHA256` 湊。下圖說明 程序，包括您為簽署所建立字串的各種元件

當 Amazon S3 收到已驗證的請求時，它會計算簽章，然後將其與您在請求中提供的簽章進行比較。因此，您必須使用與 Amazon S3 相同的方法來計算簽章。以商定格式提出簽署請求的程序稱為正式化。



下表描述圖表中顯示的函數。您需要為這些函數實作程式碼。

函式	說明
Lowercase()	將字串轉換成小寫。
Hex()	小寫基數 16 編碼。

函式	說明
SHA256Hash()	安全雜湊演算法 (SHA) 加密雜湊函數。
HMAC-SHA256()	使用 SHA256 演算法搭配提供的簽署金鑰來計算 HMAC。這是最後的簽章。
Trim()	移除任何前置和結尾空格。

函式	說明
UriEncode()	URI 編碼每個位元組。UriEncode() 必須強制執行下列規則： • URI 對除未預留字元之外的每個位元組進行編碼：'A'-'Z', 'a'-'z', '0'-'9', '-', '.', '_', and '~'。 • 空格字元是預留字元，必須編碼為 "%20" (而不是 "+")。 • 每個 URI 編碼的位元組都是由 '%' 和位元組的兩位數十六進位值組成。 • 十六進位值中的字母必須為大寫，例如 "%1A"。 • 對所有位置的正斜線字元 '/' 進行編碼 (物件索引鍵名稱除外)。例如，如果物件索引鍵名為 photos/Jan/sample.jpg，則不會對索引鍵名稱中的正斜線進行編碼。  Important 由於基礎 RFC 中的實作差異和相關模糊性，您的開發平台提供的標準 UriEncode 函數可能無法工作。我們建議您編寫自己的自訂 UriEncode 函數，以確保您的編碼能夠正常工作。 以下是 Java 中的 UriEncode() 函數範例。 <pre data-bbox="615 1312 1446 1858"> public static String UriEncode(CharSequence input, boolean encodeSlash) { StringBuilder result = new StringBui lder(); for (int i = 0; i < input.length(); i++) { char ch = input.charAt(i); if ((ch >= 'A' && ch <= 'Z') (ch >= 'a' && ch <= 'z') (ch >= '0' && ch <= '9') ch == '_' ch == '-' ch == '~' ch == '.') { result.append(ch); } else if (ch == '/') { result.append(encodeSlash ? "%2F" : ch); } else { </pre>

函式	說明
	<pre> result.append(toHexUTF8(ch)); } } return result.toString(); } </pre>

任務 1：建立正式請求

本節提供建立正式請求的概觀。

以下是 Amazon S3 用來計算簽章的正式請求格式。若要符合簽章，您必須以下列格式建立正式請求：

```

<HTTPMethod>\n
<CanonicalURI>\n
<CanonicalQueryString>\n
<CanonicalHeaders>\n
<SignedHeaders>\n
<HashedPayload>

```

其中：

- *HTTPMethod* 是其中一個 HTTP 方法，例如 GET、PUT、HEAD 和 DELETE。
- *CanonicalURI* 是 URI 絕對路徑元件的 URI 編碼版本，從網域名稱後面的「/」開始，直到字串結尾或問號字元（'?'）如果您有查詢字串參數。下列範例中的 URI /examplebucket/myphoto.jpg 是絕對路徑，您不會在絕對路徑中編碼 "/"：

```
http://s3.amazonaws.com/examplebucket/myphoto.jpg
```

Note

您不會標準化 Amazon S3 請求的 URI 路徑。例如，您可能有一個儲存貯體，其中包含名為 "my-object//example//photo.user" 的物件。標準化路徑會將請求中的物件名稱變更為「my-object/example/photo.user」。這是該物件的不正確路徑。

- **CanonicalQueryString** 會指定 URI 編碼的查詢字串參數。您可以個別對名稱和值進行 URI 編碼。您還必須依索引鍵名稱的字母順序來排序正式查詢字串中的參數。編碼後進行排序。下列 URI 範例中的查詢字串為 `prefix=somePrefix&marker=someMarker&max-keys=20`：

```
http://s3.amazonaws.com/examplebucket?prefix=somePrefix&marker=someMarker&max-keys=20
```

正式查詢字串如下所示 (為了便於閱讀，向此範例新增了分行符號)：

```
UriEncode("marker")+"="+UriEncode("someMarker")+"&"+
UriEncode("max-keys")+"="+UriEncode("20") + "&" +
UriEncode("prefix")+"="+UriEncode("somePrefix")
```

請求以子資源為目標時，對應的查詢參數值將是空字串 ("")。例如，下列 URI 會識別 `examplebucket` 儲存貯體上的 ACL 子資源：

```
http://s3.amazonaws.com/examplebucket?acl
```

在這種情況下，**CanonicalQueryString** 如下所示：

```
UriEncode("acl") + "=" + ""
```

如果 URI 不包含 '?'，則請求中沒有查詢字串，並且您將正式查詢字串設定為空字串 ("")。您仍然需要包含 `"\n"`。

- **CanonicalHeaders** 是包含其值的請求標頭清單。個別標頭名稱和值對以新行字元 ("\n") 分隔。標頭名稱必須為小寫。您必須依字母順序排序標頭名稱來建構字串，如下列範例所示：

```
Lowercase(<HeaderName1>)+":"+Trim(<value>)+"\n"
Lowercase(<HeaderName2>)+":"+Trim(<value>)+"\n"
...
Lowercase(<HeaderNameN>)+":"+Trim(<value>)+"\n"
```

本範例中使用的 `Lowercase()` 和 `Trim()` 函數在上一節中進行了描述。

CanonicalHeaders 清單必須包含下列項目：

- HTTP host 標頭。
- 如果 Content-MD5 標頭存在於請求中，您必須將其新增至 *CanonicalHeaders* 清單。
- 還必須新增您計劃包含在請求中的任何 x-amz-* 標頭。例如，如果您使用的是暫時安全憑證，需要將 x-amz-security-token 包含在您的請求中。您必須將此標頭新增在 *CanonicalHeader* 的清單中。

 Note

- 所有 AWS Signature 第 4 版請求都需要 x-amz-content-sha256 標頭。其提供請求承載的雜湊。不過，您不需要將包含 x-amz-content-sha256 為正式標頭，因為在計算請求中傳送的承載雜湊時，S3 會自動使用其值。
- 如果沒有承載，您必須提供空字串的雜湊。
- 如果您不希望 S3 檢查請求的雜湊，您可以 "UNSIGNED-PAYLOAD" 改為使用常值字串。

以下是範例 CanonicalHeaders 字串。標頭名稱為小寫且已排序。

Example 範例 1

```
host:s3.amazonaws.com
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130708T220855Z
```

Example 範例 2

```
host:s3.amazonaws.com
x-amz-content-sha256:UNSIGNED-PAYLOAD
x-amz-date:20130708T220855Z
```

Example 範例 3

```
host:s3.amazonaws.com
x-amz-date:20130708T220855Z
```

Note

為了計算授權簽章，`x-amz-content-sha256`需要主機標頭和 以外的所有`x-amz-*`標頭。不過，為了防止資料竄改，您應該考慮在簽章計算中包含所有標頭。

簽署 `x-amz-content-sha256`標頭是選用的，因為 S3 會在計算收到的請求承載雜湊時使用其值。

- *SignedHeaders* 是按字母順序排序、分號分隔的小寫請求標頭名稱清單。清單中的請求標頭與您在 CanonicalHeaders 字串中包含的標頭相同。在先前的範例中，*SignedHeaders* 的值如下：

Example範例 1 和 2

```
host;x-amz-content-sha256;x-amz-date
```

Example範例 3

```
host;x-amz-date
```

- *HashedPayload* 是請求承載之 SHA256 雜湊的十六進位值。

```
Hex(SHA256Hash(<payload>))
```

如果請求中沒有承載，則計算空字串的雜湊，如下所示：

```
Hex(SHA256Hash(""))
```

雜湊會傳回下列值：

```
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca95991b7852b855
```

例如，使用 PUT 請求上傳物件時，請在主體中提供物件資料。當您使用 GET 請求擷取物件時，會計算空字串雜湊。

任務 2：建立登入字串

本節概述如何建立要簽署的字串。如需step-by-step說明，請參閱 [AWS 一般參考](#) 中的 [任務 2：建立要簽署的字串](#)。

要簽署的字串是下列字串的串連：

```
"AWS4-HMAC-SHA256" + "\n" +  
timeStampISO8601Format + "\n" +  
<Scope> + "\n" +  
Hex(SHA256Hash(<CanonicalRequest>))
```

常數字串AWS4-HMAC-SHA256指定您正在使用的雜湊演算法 HMAC-SHA256。timeStamp 是 ISO 8601 格式的目前 UTC 時間（例如 20130524T000000Z）。

Scope 會將產生的簽章繫結至特定日期 AWS 區域、和服務。因此，產生的簽章僅適用於特定區域和特定服務。簽章在指定日期後七天內有效。

```
date.Format(<YYYYMMDD>) + "/" + <region> + "/" + <service> + "/aws4_request"
```

對於 Amazon S3，服務字串為 s3。如需##字串清單，請參閱《》中的 [區域和端點](#) AWS 一般參考。此表格中的區域欄提供有效的區域字串清單。

下列範圍會將產生的簽章限制為 us-east-1 區域和 Amazon S3。

```
20130606/us-east-1/s3/aws4_request
```

Note

Scope 必須使用您用來計算簽署金鑰的相同日期，如下節所述。

任務 3：計算簽章

在 AWS Signature 第 4 版中，您首先會建立範圍限定於特定區域和服務的簽署金鑰，而不是使用您的 AWS 存取金鑰來簽署請求。如需簽署金鑰的詳細資訊，請參閱 [簽署請求簡介](#)。

```

DateKey          = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")
DateRegionKey    = HMAC-SHA256(<DateKey>, "<aws-region>")
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")
SigningKey       = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")

```

Note

有些使用案例最多可以處理 7 天的簽章金鑰。如需詳細資訊，請參閱[與他人共用物件](#)。

如需區域字串清單，請參閱《》中的[區域和端點](#)AWS 一般參考。

使用簽署金鑰可讓您將 AWS 登入資料保存在安全的地方。例如，如果您有多個伺服器與 Amazon S3 通訊，您會與這些伺服器共用簽署金鑰；您不需要在每個伺服器上保留私密存取金鑰的副本。簽署金鑰的有效期限最長為七天。因此，每次計算簽署金鑰時，都需要與伺服器共用簽署金鑰。如需詳細資訊，請參閱[驗證請求AWS（簽章第 4 版）](#)。

最終簽章是要簽署之字串的 HMAC-SHA256 雜湊，使用簽署金鑰做為金鑰。

```
HMAC-SHA256(SigningKey, StringToSign)
```

如需建立簽章的step-by-step說明，請參閱《》中的[任務 3：建立簽章](#)AWS 一般參考。

範例：簽章計算

您可以使用本節中的範例做為參考，以檢查程式碼中的簽章計算。範例中顯示的計算會使用下列資料：

- 存取金鑰範例。

參數	Value
AWSSecretAccessKeyId	AKIAIOSFODNN7EXAMPLE
AWSSecretAccessKey	wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

- 請求時間戳記 20130524T000000Z (Fri, 24 May 2013 00:00:00 GMT)。
- 儲存貯體名稱 examplebucket。
- 儲存貯體假設位於美國東部（維吉尼亞北部）區域。登入資料Scope和Signing Key計算使用 us-east-1 做為區域指標。如需其他區域的資訊，請參閱《》中的[區域和端點](#) AWS 一般參考。
- 您可以使用路徑樣式或虛擬託管樣式的請求。下列範例示範如何簽署虛擬託管樣式請求，例如：

```
https://examplebucket.s3.amazonaws.com/photos/photo1.jpg
```

如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[儲存貯體虛擬託管](#)。

範例：GET 物件

下列範例會從取得物件的前 10 個位元組 (test.txt)。examplebucket 如需 API 動作的詳細資訊，請參閱 [GetObject](#)。

```
GET /test.txt HTTP/1.1
Host: examplebucket.s3.amazonaws.com
Authorization: SignatureToBeCalculated
Range: bytes=0-9
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date: 20130524T000000Z
```

由於此 GET 請求不提供任何內文內容，因此 x-amz-content-sha256 值可以是空白請求內文的雜湊或常值字串 "UNSIGNED-PAYLOAD"。下列步驟顯示使用空字串雜湊的 Authorization 標頭的簽章計算和建構。

1. StringToSign

a. CanonicalRequest

```
GET
/test.txt

host:examplebucket.s3.amazonaws.com
range:bytes=0-9
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

```
x-amz-date:20130524T000000Z
```

```
host;range;x-amz-content-sha256;x-amz-date
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

在正式請求字串中，最後一行是空白請求內文的雜湊。第三行是空的，因為請求中沒有查詢參數。

b. StringToSign

```
AWS4-HMAC-SHA256
20130524T000000Z
20130524/us-east-1/s3/aws4_request
7344ae5b7ee6c3e7e6b0fe0640412a37625d1fbfff95c48bbb2dc43964946972
```

2. SigningKey

```
signing key = HMAC-SHA256(HMAC-SHA256(HMAC-SHA256(HMAC-SHA256("AWS4" +
"<YourSecretAccessKey>", "20130524"), "us-east-1"), "s3"), "aws4_request")
```

3. 簽章

```
f0e8bdb87c964420e857bd35b5d6ed310bd44f0170aba48dd91039c6036bdb41
```

4. 授權標頭

產生的Authorization標頭如下：

```
AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/
s3/aws4_request,SignedHeaders=host;range;x-amz-content-sha256;x-amz-
date,Signature=f0e8bdb87c964420e857bd35b5d6ed310bd44f0170aba48dd91039c6036bdb41
```

範例：PUT 物件

此範例 PUT 請求會在 中建立物件 examplebucket (test\$file.text)。該範例假設如下：

- 您透過新增請求標頭來 `x-amz-storage-class` 請求 `REDUCED_REDUNDANCY` 做為儲存體方案。如需儲存類別的資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的儲存類別。
- 上傳檔案的內容是字串，請求 `x-amz-content-sha256` 中的 "Welcome to Amazon S3." 值是以此字串為基礎。

如需 API 動作的相關資訊，請參閱 [PutObject](#)。

```
PUT test$file.text HTTP/1.1
Host: examplebucket.s3.amazonaws.com
Date: Fri, 24 May 2013 00:00:00 GMT
Authorization: SignatureToBeCalculated
x-amz-date: 20130524T000000Z
x-amz-storage-class: REDUCED_REDUNDANCY
x-amz-content-sha256: 44ce7dd67c959e0d3524ffac1771dfbba87d2b6b4b4e99e42034a8b803f8b072

<Payload>
```

下列步驟顯示簽章計算。

1. StringToSign

a. CanonicalRequest

```
PUT
/test%24file.text

date:Fri, 24 May 2013 00:00:00 GMT
host:examplebucket.s3.amazonaws.com
x-amz-content-sha256:44ce7dd67c959e0d3524ffac1771dfbba87d2b6b4b4e99e42034a8b803f8b072
x-amz-date:20130524T000000Z
x-amz-storage-class:REDUCED_REDUNDANCY

date;host;x-amz-content-sha256;x-amz-date;x-amz-storage-class
44ce7dd67c959e0d3524ffac1771dfbba87d2b6b4b4e99e42034a8b803f8b072
```

在正式請求中，第三行是空的，因為請求中沒有查詢參數。 `x-amz-content-sha256` 正式標頭可以選擇性地簽署，因為其承載雜湊已在請求底部提供。最後一行是內文的雜湊，應與 HTTP 請求中傳送至 S3 `x-amz-content-sha256` header 的值相同。

b. StringToSign

```
AWS4-HMAC-SHA256
20130524T000000Z
20130524/us-east-1/s3/aws4_request
9e0e90d9c76de8fa5b200d8c849cd5b8dc7a3be3951ddb7f6a76b4158342019d
```

2. SigningKey

```
signing key = HMAC-SHA256(HMAC-SHA256(HMAC-SHA256(HMAC-SHA256("AWS4" +
"<YourSecretAccessKey>", "20130524"), "us-east-1"), "s3"), "aws4_request")
```

3. 簽章

```
98ad721746da40c64f1a55b78f14c238d841ea1380cd77a1b5971af0ece108bd
```

4. 授權標頭

產生的Authorization標頭如下：

```
AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/
aws4_request,SignedHeaders=date;host;x-amz-content-sha256;x-amz-date;x-amz-storage-
class,Signature=98ad721746da40c64f1a55b78f14c238d841ea1380cd77a1b5971af0ece108bd
```

範例：GET 儲存貯體生命週期

下列 GET 請求會擷取 的生命週期組態examplebucket。如需 API 動作的相關資訊，請參閱 [GetBucketLifecycleConfiguration](#)。

```
GET ?lifecycle HTTP/1.1
Host: examplebucket.s3.amazonaws.com
Authorization: SignatureToBeCalculated
x-amz-date: 20130524T000000Z
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

由於請求不提供任何內文內容，x-amz-content-sha256標頭值是空白請求內文的雜湊。下列步驟顯示簽章計算。

1. StringToSign

a. CanonicalRequest

```
GET
/
lifecycle=
host:examplebucket.s3.amazonaws.com
x-amz-content-
sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130524T000000Z

host;x-amz-content-sha256;x-amz-date
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

在正式請求中，最後一行是空白請求內文的雜湊。

b. StringToSign

```
AWS4-HMAC-SHA256
20130524T000000Z
20130524/us-east-1/s3/aws4_request
9766c798316ff2757b517bc739a67f6213b4ab36dd5da2f94eaebf79c77395ca
```

2. SigningKey

```
signing key = HMAC-SHA256(HMAC-SHA256(HMAC-SHA256(HMAC-SHA256("AWS4" +
"<YourSecretAccessKey>", "20130524"), "us-east-1"), "s3"), "aws4_request")
```

3. 簽章

```
fea454ca298b7da1c68078a5d1bdbfbbe0d65c699e0f91ac7a200a0136783543
```

4. 授權標頭

產生的Authorization標頭如下：

```
AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/
s3/aws4_request,SignedHeaders=host;x-amz-content-sha256;x-amz-
date,Signature=fea454ca298b7da1c68078a5d1bdbfbbe0d65c699e0f91ac7a200a0136783543
```

範例：取得儲存貯體（列出物件）

下列範例會從儲存examplebucket貯體擷取物件清單。如需 API 動作的相關資訊，請參閱 [ListObjects](#)。

```
GET ?max-keys=2&prefix=J HTTP/1.1
Host: examplebucket.s3.amazonaws.com
Authorization: SignatureToBeCalculated
x-amz-date: 20130524T000000Z
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

由於請求不提供內文，因此 的值x-amz-content-sha256是空白請求內文的雜湊。下列步驟顯示簽章計算。

1. StringToSign

a. CanonicalRequest

```
GET
/
max-keys=2&prefix=J
host:examplebucket.s3.amazonaws.com
x-amz-content-
sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130524T000000Z

host;x-amz-content-sha256;x-amz-date
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

在正式字串中，最後一行是空白請求內文的雜湊。

b. StringToSign

```
AWS4-HMAC-SHA256
20130524T000000Z
20130524/us-east-1/s3/aws4_request
df57d21db20da04d7fa30298dd4488ba3a2b47ca3a489c74750e0f1e7df1b9b7
```

2. SigningKey

```
signing key = HMAC-SHA256(HMAC-SHA256(HMAC-SHA256(HMAC-SHA256("AWS4" +
"<YourSecretAccessKey>", "20130524"), "us-east-1"), "s3"), "aws4_request")
```

3. 簽章

```
34b48302e7b5fa45bde8084f4b7868a86f0a534bc59db6670ed5711ef69dc6f7
```

4. 授權標頭

產生的Authorization標頭如下：

```
AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/
s3/aws4_request,SignedHeaders=host;x-amz-content-sha256;x-amz-
date,Signature=34b48302e7b5fa45bde8084f4b7868a86f0a534bc59db6670ed5711ef69dc6f7
```

授權標頭的簽章計算：以多個區塊傳輸承載（區塊上傳）AWS（簽章版本 4）

如中所述[概觀](#)，使用 Authorization 標頭驗證請求時，您可以選擇以區塊形式上傳承載。您可以使用固定大小或可變大小區塊來傳送資料。本節說明區塊上傳中的簽章計算程序、如何建立區塊內文，以及延遲簽署如何在您第一次上傳區塊的位置運作，並在後續區塊中傳送其簽章。範例區段（請參閱[範例：PUT 物件](#)）顯示簽章計算和產生的 Authorization 標頭，您可以將其做為測試套件來驗證程式碼。

Note

在一系列區塊中傳輸資料時，您必須執行下列其中一項操作：

- 使用 Content-Length HTTP 標頭明確指定總內容長度（物件長度以位元組為單位，加上每個區塊中的中繼資料）。若要這樣做，您必須在開始請求之前預先計算承載的總長度，包括您在每個區塊中傳送的中繼資料。
- 指定 Transfer-Encoding HTTP 標頭。如果您包含 Transfer-Encoding 標頭並指定以外的任何值 identity，則必須省略 Content-Length 標頭。

對於所有請求，您必須包含 x-amz-decoded-content-length 標頭，以位元組為單位指定物件的大小。

每個區塊簽章計算都包含上一個區塊的簽章。若要開始，請僅使用標頭建立種子簽章。您可以在第一個區塊的簽章計算中使用種子簽章。對於每個後續區塊，您可以建立包含上一個區塊簽章的區塊簽章。因此，區塊簽章會鏈結在一起；也就是說，區塊 n 的簽章是函數 $F(\text{區塊 } n, \text{簽章}(\text{區塊 } n-1))$ 。鏈結可確保您以正確的順序傳送區塊。

若要執行區塊上傳，請執行下列動作：

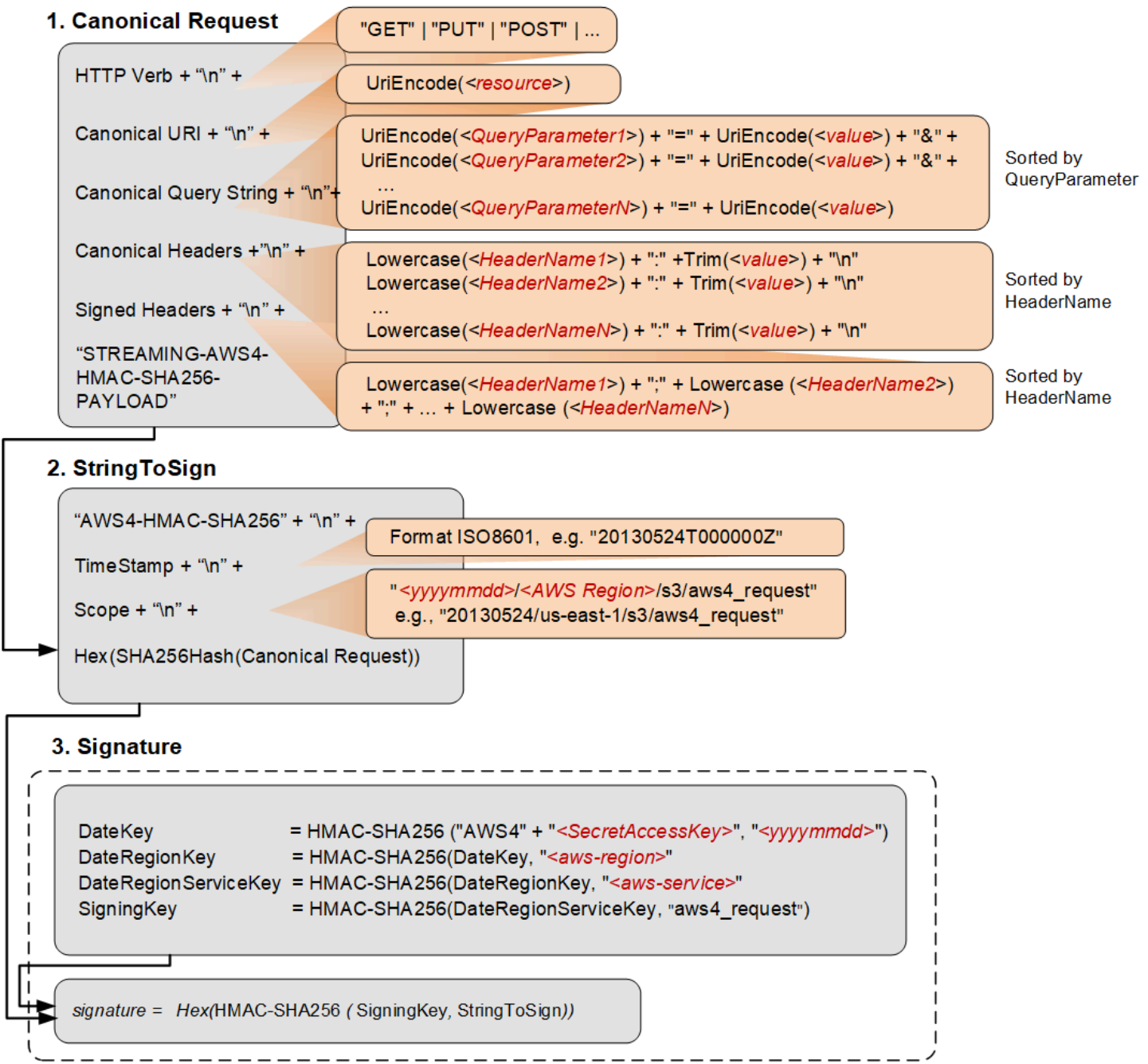
1. 決定承載區塊大小。撰寫程式碼時需要此項目。

區塊大小必須至少為 8 KB。我們建議區塊大小至少為 64 KB，以獲得更好的效能。此區塊大小適用於除最後一個區塊以外的所有區塊。您傳送的最後一個區塊可以小於 8 KB。如果您的承載很小，可以放入一個區塊中，則它可以小於 8 KB。

2. 建立要包含在第一個區塊中的種子簽章。如需詳細資訊，請參閱[計算種子簽章](#)。
3. 建立第一個區塊並將其串流。如需詳細資訊，請參閱[定義區塊內文](#)。
4. 對於每個後續區塊，計算包含您簽署字串中先前簽章的區塊簽章，建構區塊並傳送。如需詳細資訊，請參閱[定義區塊內文](#)。
5. 傳送最終的額外區塊，這與建構中的其他區塊相同，但資料位元組為零。如需詳細資訊，請參閱[定義區塊內文](#)。

計算種子簽章

下圖說明計算種子簽章的程序。



下表描述圖表中顯示的函數。您需要為這些函數實作程式碼。

函式	說明
Lowercase()	將字串轉換成小寫。
Hex()	小寫基數 16 編碼。

函式	說明
SHA256Hash()	安全雜湊演算法 (SHA) 加密雜湊函數。
HMAC-SHA256()	使用 SHA256 演算法搭配提供的簽署金鑰來計算 HMAC。這是最後的簽章。
Trim()	移除任何前置和結尾空格。

函式	說明
UriEncode()	URI 編碼每個位元組。UriEncode() 必須強制執行下列規則： - URI 對除未預留字元之外的每個位元組進行編碼：'A'-'Z', 'a'-'z', '0'-'9', '-', '.', '_', and '~'。 - 空格字元是預留字元，必須編碼為 "%20" (而不是 "+")。 - 每個 URI 編碼的位元組都是由 '%' 和位元組的兩位數十六進位值組成。 - 十六進位值中的字母必須為大寫，例如 "%1A"。 - 對所有位置的正斜線字元 '/' 進行編碼 (物件索引鍵名稱除外)。例如，如果物件索引鍵名為 photos/Jan/sample.jpg，則不會對索引鍵名稱中的正斜線進行編碼。  Important 由於基礎 RFC 中的實作差異和相關模糊性，您的開發平台提供的標準 UriEncode 函數可能無法工作。我們建議您編寫自己的自訂 UriEncode 函數，以確保您的編碼能夠正常工作。 以下是 Java 中的 UriEncode() 函數範例。 <pre> public static String UriEncode(CharSequence input, boolean encodeSlash) { StringBuilder result = new StringBui lder(); for (int i = 0; i < input.length(); i++) { char ch = input.charAt(i); if ((ch >= 'A' && ch <= 'Z') (ch >= 'a' && ch <= 'z') (ch >= '0' && ch <= '9') ch == '_' ch == '-' ch == '~' ch == '.') { result.append(ch); } else if (ch == '/') { result.append(encodeSlash ? "%2F" : ch); } else { </pre>

函式	說明
	<pre> result.append(toHexUTF8(ch)); } } return result.toString(); } </pre>

如需簽署程序的相關資訊，請參閱 [授權標頭的簽章計算：在單一區塊中傳輸承載AWS（簽章第 4 版）](#)。程序相同，但 的 建立CanonicalRequest不同，如下所示：

- 除了您計劃新增的請求標頭之外，您還必須包含下列標頭：

標頭	說明
x-amz-content-sha256	所有 AWS Signature 第 4 版請求都需要此標頭。將 值設定為 <code>STREAMING-AWS4-HMAC-SHA256-PAYLOAD</code> 表示簽章僅涵蓋標頭，而且沒有承載。
Content-Encoding	將值設定為 <code>aws-chunked</code> 。 Amazon S3 支援多個內容編碼值。您可以在使用 Signature 第 4 版串流 API 時指定自訂內容編碼。 例如： <pre>Content-Encoding : aws-chunked,gzip</pre> Amazon S3 會將沒有<code>aws-chunked</code> 值的產生物件存放在 <code>content-encoding</code> 標頭中。如果 <code>aws-chunked</code> 是您在 <code>content-encoding</code> 標頭中傳遞的唯一值，S3 會將<code>content-encoding</code> 標頭視為空白，而且不會在擷取物件時傳回此標頭。

標頭	說明
x-amz-decoded-content-length	將值設定為要區塊化之資料的長度，以位元組為單位，而不計算任何中繼資料。例如，如果您要上傳 4 GB 檔案，請將 值設定為 4294967296。這是要上傳之物件的原始大小（您要存放在 Amazon S3 中的資料）。
Content-Length	將值設定為已傳輸 HTTP 內文的實際大小，其中包括資料的長度（為設定的值x-amz-decoded-content-length ），以及區塊中繼資料。每個區塊都有中繼資料，例如上一個區塊的簽章。區塊計算會在下一節中討論。如果您包含 Transfer-Encoding 標頭並指定 以外的任何值identity，則不得包含 Content-Length 標頭。

您使用種子簽章傳送第一個區塊。您必須建構區塊，如下節所述。

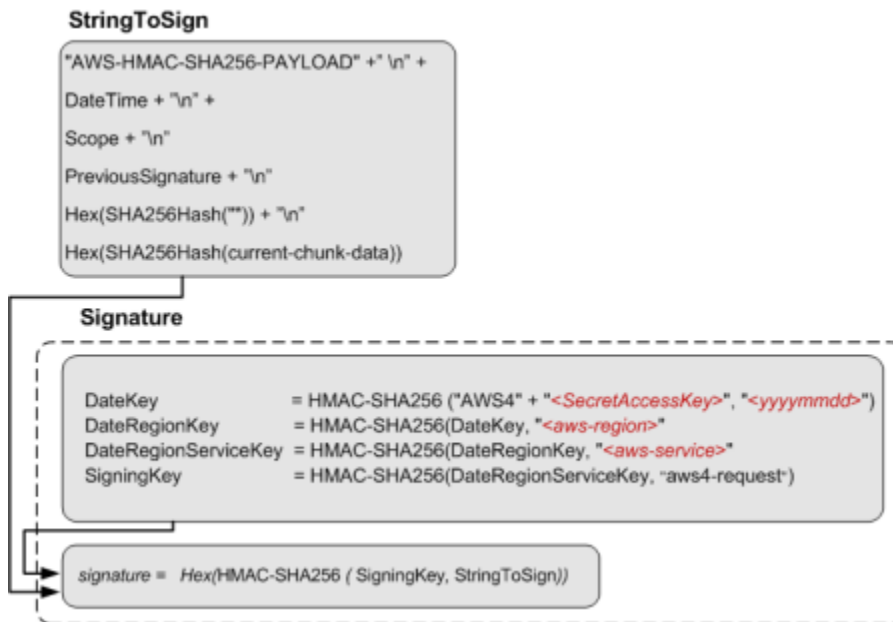
定義區塊內文

所有區塊都包含一些中繼資料。每個區塊必須符合下列結構：

```
string(IntHexBase(chunk-size)) + ";chunk-signature=" + signature + \r\n + chunk-data + \r\n
```

其中：

- IntHexBase() 是您寫入以將整數區塊大小轉換為十六進位的函數。例如，如果區塊大小為 65536，則十六進位字串為 "10000"。
- #### 是不含中繼資料的區塊資料大小，以位元組為單位。例如，如果您上傳 65 KB 物件並使用 64 KB 的區塊大小，則會以三個區塊上傳資料：第一個為 64 KB，第二個為 1 KB，最後一個區塊為 0 位元組。
- ## 對於每個區塊，您可以使用下列字串來計算簽章。對於第一個區塊，您可以使用 seed-signature 作為先前的簽章。



您傳送的最終區塊資料大小為 0，雖然區塊內文仍包含中繼資料，包括先前區塊的簽章。

範例：PUT 物件

您可以使用本節中的範例做為參考，以檢查程式碼中的簽章計算。檢閱範例之前，請注意下列事項：

- 這些範例中的簽章計算使用以下範例安全登入資料。

參數	Value
AWSAccessKeyId	AKIAIOSFODNN7EXAMPLE
AWSSecretAccessKey	wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

- 所有範例都使用請求時間戳記 20130524T000000Z (Fri, 24 May 2013 00:00:00 GMT)。
- 所有範例都會使用 examplebucket 做為儲存貯體名稱。
- 儲存貯體假設位於美國東部（維吉尼亞北部）區域，而登入資料 Scope 和 Signing Key 計算會使用 us-east-1 做為區域指標。如需詳細資訊，請參閱 [Amazon Web Services 一般參考](#) 中的區域與端點。
- 您可以使用路徑樣式或虛擬託管樣式請求。下列範例使用虛擬託管樣式請求，例如：

```
https://examplebucket.s3.amazonaws.com/photos/photo1.jpg
```

如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[儲存貯體虛擬託管](#)。

下列範例會傳送上傳物件的PUT請求。簽章計算假設如下：

- 您正在上傳 65 KB 文字檔案，而檔案內容是由字母 'a' 組成的單字元字串。
- 區塊大小為 64 KB。因此，承載會以三個區塊上傳：64 KB、1 KB，以及具有 0 個位元組區塊資料的最終區塊。
- 產生的物件具有金鑰名稱 chunkObject.txt。
- 您透過新增請求標頭來x-amz-storage-class請求 REDUCED_REDUNDANCY 做為儲存體方案。

如需 API 動作的相關資訊，請參閱 [PutObject](#)。一般請求語法如下：

```
PUT /examplebucket/chunkObject.txt HTTP/1.1
Host: s3.amazonaws.com
x-amz-date: 20130524T000000Z
x-amz-storage-class: REDUCED_REDUNDANCY
Authorization: SignatureToBeCalculated
x-amz-content-sha256: STREAMING-AWS4-HMAC-SHA256-PAYLOAD
Content-Encoding: aws-chunked
x-amz-decoded-content-length: 66560
Content-Length: 66824
<Payload>
```

下列步驟顯示簽章計算。

1. 種子簽章 — 建立要簽署的字串

a. CanonicalRequest

```
PUT
/examplebucket/chunkObject.txt

content-encoding:aws-chunked
content-length:66824
```

```
host:s3.amazonaws.com
x-amz-content-sha256:STREAMING-AWS4-HMAC-SHA256-PAYLOAD
x-amz-date:20130524T000000Z
x-amz-decoded-content-length:66560
x-amz-storage-class:REDUCED_REDUNDANCY

content-encoding;content-length;host;x-amz-content-sha256;x-amz-date;x-amz-
decoded-content-length;x-amz-storage-class
STREAMING-AWS4-HMAC-SHA256-PAYLOAD
```

在正式請求中，第三行是空的，因為請求中沒有查詢參數。最後一行是做為雜湊承載值提供的常數字串，應該與 `x-amz-content-sha256` header 的值相同。

b. StringToSign

```
AWS4-HMAC-SHA256
20130524T000000Z
20130524/us-east-1/s3/aws4_request
cee3fed04b70f867d036f722359b0b1f2f0e5dc0efadbc082b76c4c60e316455
```

Note

如需有關要簽署之字串中每一行的資訊，請參閱說明種子簽章計算的圖表。

2. SigningKey

```
signing key = HMAC-SHA256(HMAC-SHA256(HMAC-SHA256(HMAC-SHA256("AWS4" +
"<YourSecretAccessKey>", "20130524"), "us-east-1"), "s3"), "aws4_request")
```

3. 種子簽章

```
4f232c4386841ef735655705268965c44a0e4690baa4adea153f7db9fa80a0a9
```

4. 授權標頭

產生的授權標頭如下所示：

簽章計算：以多個區塊傳輸承載

```
AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/
aws4_request,SignedHeaders=content-encoding;content-length;host;x-amz-
content-sha256;x-amz-date;x-amz-decoded-content-length;x-amz-storage-
class,Signature=4f232c4386841ef735655705268965c44a0e4690baa4adea153f7db9fa80a0a9
```

5. 區塊 1：(65536 位元組，值 97 表示字母 'a')

a. 要簽署的區塊字串：

```
AWS4-HMAC-SHA256-PAYLOAD
20130524T000000Z
20130524/us-east-1/s3/aws4_request
4f232c4386841ef735655705268965c44a0e4690baa4adea153f7db9fa80a0a9
e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855
bf718b6f653bebc184e1479f1935b8da974d701b893afcf49e701f3e2f9f9c5a
```

Note

如需有關要簽署之字串中每一行的資訊，請參閱上圖，其中顯示要簽署之字串的各種元件（例如，最後三行為、previous-signaturehash("")和hash(current-chunk-data)）。

b. 區塊簽章：

```
ad80c730a21e5b8d04586a2213dd63b9a0e99e0e2307b0ade35a65485a288648
```

c. 已傳送的區塊資料：

```
10000;chunk-
signature=ad80c730a21e5b8d04586a2213dd63b9a0e99e0e2307b0ade35a65485a288648
<65536-bytes>
```

6. 區塊 2：(1024 個位元組，值 97 表示字母 'a')

a. 要簽署的區塊字串：

```
AWS4-HMAC-SHA256-PAYLOAD
20130524T000000Z
```

```
20130524/us-east-1/s3/aws4_request
ad80c730a21e5b8d04586a2213dd63b9a0e99e0e2307b0ade35a65485a288648
e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855
2edc986847e209b4016e141a6dc8716d3207350f416969382d431539bf292e4a
```

b. 區塊簽章：

```
0055627c9e194cb4542bae2aa5492e3c1575bbb81b612b7d234b86a503ef5497
```

c. 已傳送的區塊資料：

```
400;chunk-
signature=0055627c9e194cb4542bae2aa5492e3c1575bbb81b612b7d234b86a503ef5497
<1024 bytes>
```

7. 區塊 3：(0 位元組資料)

a. 要簽署的區塊字串：

```
AWS4-HMAC-SHA256-PAYLOAD
20130524T000000Z
20130524/us-east-1/s3/aws4_request
0055627c9e194cb4542bae2aa5492e3c1575bbb81b612b7d234b86a503ef5497
e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855
e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855
```

b. 區塊簽章：

```
b6c6ea8a5354eaf15b3cb7646744f4275b71ea724fed81ceb9323e279d449df9
```

c. 已傳送的區塊資料：

```
0;chunk-
signature=b6c6ea8a5354eaf15b3cb7646744f4275b71ea724fed81ceb9323e279d449df9
```


結尾標頭的簽章計算（區塊上傳）AWS（簽章第 4 版）

使用 Authorization 標頭驗證請求時，您也可以上傳區塊中的承載。當您以區塊傳送物件的資料時，您也可以選擇包含結尾標頭。(如需詳細資訊，請參閱 [授權標頭的簽章計算：以多個區塊傳輸承載（區塊上傳）AWS（簽章版本 4）](#)。) 本節說明當您要在多個區塊上傳結束時包含結尾標頭時，需要採取的步驟。

Important

當您包含結尾標頭時，您必須在初始標頭中傳送下列項目：

- 您必須 `x-amz-content-sha256` 將設定為適當的值，表示將包含追蹤器。若要查看可接受的值 `x-amz-content-sha256`，請參閱 [驗證請求：使用授權標頭AWS（簽章第 4 版）](#)。
- 您必須設定 `x-amz-trailer` 以指出您的 包含在結尾標頭中的內容。

追蹤標頭只會在區塊上傳之後傳送。先前的區塊會正常傳送並簽署，如先前章節所述，包括傳送承載為 0 位元組的最終區塊。尾端標頭會包含在自己的區塊中，並在承載為 0 位元組的最終區塊之後傳送。例如，如果您的資料以 100 KB 區塊結尾，您可以傳送下列項目：

- 先前的資料區塊
- 物件的最終區塊為 100 KB
- 0 個位元組區塊，表示物件的結尾
- 追蹤標頭區塊

範例：檢查簽章計算

您可以使用本節中的範例做為參考，以檢查程式碼中的簽章計算。檢閱範例之前，請注意下列事項：

- 這些範例中的簽章計算使用以下範例安全登入資料。

參數	Value
<code>AWSSecretAccessKey</code>	<code>AKIAIOSFODNN7EXAMPLE</code>

參數	Value
AWSecretAccessKey	wJa1rXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY

- 所有範例都使用請求時間戳記 20130524T000000Z (Fri, 24 May 2013 00:00:00 GMT)。
- 所有範例都會使用 examplebucket 做為儲存貯體名稱。
- 儲存貯體假設位於美國東部（維吉尼亞北部）區域，而登入資料 Scope 和 Signing Key 計算會使用 us-east-1 做為區域指標。如需詳細資訊，請參閱《Amazon Web Services 一般參考》中的 [區域和端點](#)。
- 您可以使用路徑樣式或虛擬託管樣式請求。下列範例使用虛擬託管樣式請求，例如：

```
https://examplebucket.s3.amazonaws.com/photos/photo1.jpg
```

如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [儲存貯體虛擬託管](#)。

下列範例會傳送上傳物件的 PUT 請求。簽章計算假設如下：

- 您正在上傳 65 KB 文字檔案，而檔案內容是由字母 'a' 組成的單字元字串。
- 區塊大小為 64 KB。因此，承載會以三個區塊上傳：64 KB、1 KB，以及具有 0 個位元組區塊資料的最終區塊。
- 產生的物件具有金鑰名稱 chunkObject.txt。
- 您透過新增請求標頭來 x-amz-storage-class 請求 REDUCED_REDUNDANCY 做為儲存體方案。
- 傳輸包含 CRC32C 檢查總和值作為結尾標頭。

如需 API 動作的相關資訊，請參閱 [PutObject](#)。一般請求語法如下：

```
PUT /examplebucket/chunkObject.txt HTTP/1.1
Host: s3.amazonaws.com
x-amz-date: 20130524T000000Z
x-amz-storage-class: REDUCED_REDUNDANCY
Authorization: SignatureToBeCalculated
x-amz-content-sha256: STREAMING-AWS4-HMAC-SHA256-PAYLOAD-TRAILER
Content-Encoding: aws-chunked
x-amz-decoded-content-length: 66560
x-amz-trailer: x-amz-checksum-crc32c
```

Content-Length: 66946

<Payload>

下列步驟顯示簽章計算。

1. 種子簽章 — 建立要簽署的字串

a. CanonicalRequest

```
PUT
/examplebucket/chunk0bject.txt

content-encoding:aws-chunked
host:s3.amazonaws.com
x-amz-content-sha256:STREAMING-AWS4-HMAC-SHA256-PAYLOAD-TRAILER
x-amz-date:20130524T000000Z
x-amz-decoded-content-length:66560
x-amz-storage-class:REDUCED_REDUNDANCY
x-amz-trailer:x-amz-checksum-crc32c

content-encoding;host;x-amz-content-sha256;x-amz-date;x-amz-decoded-content-
length;x-amz-storage-class;x-amz-trailer
STREAMING-AWS4-HMAC-SHA256-PAYLOAD-TRAILER
```

在正式請求中，第三行是空的，因為請求中沒有查詢參數。最後一行是做為雜湊承載值提供的常數字串，應該與 `x-amz-content-sha256` header 的值相同。

b. StringToSign

```
AWS4-HMAC-SHA256
20130524T000000Z
20130524/us-east-1/s3/aws4_request
44d48b8c2f70eae815a0198cc73d7a546a73a93359c070abbaa5e6c7de112559
```

Note

如需有關要簽署之字串中每一行的資訊，請參閱說明種子簽章計算的圖表。

2. SigningKey

```
signing key = HMAC-SHA256(HMAC-SHA256(HMAC-SHA256(HMAC-SHA256("AWS4" +
  "<YourSecretAccessKey>", "20130524"), "us-east-1"), "s3"), "aws4_request")
```

3. 種子簽章

```
106e2a8a18243abcf37539882f36619c00e2dfc72633413f02d3b74544bfeb8e
```

4. 授權標頭

產生的授權標頭如下所示：

```
AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/
aws4_request,SignedHeaders=content-encoding;content-length;host;x-amz-
content-sha256;x-amz-date;x-amz-decoded-content-length;x-amz-storage-
class,Signature=106e2a8a18243abcf37539882f36619c00e2dfc72633413f02d3b74544bfeb8e
```

5. 區塊 1：(65536 位元組，值 97 表示字母 'a')

a. 要簽署的區塊字串：

```
AWS4-HMAC-SHA256-PAYLOAD
20130524T000000Z
20130524/us-east-1/s3/aws4_request
106e2a8a18243abcf37539882f36619c00e2dfc72633413f02d3b74544bfeb8e
e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855
bf718b6f653bebc184e1479f1935b8da974d701b893afcf49e701f3e2f9f9c5a
```

Note

如需有關要簽署之字串中每一行的資訊，請參閱上一個主題 ([計算種子簽章](#)) 中的圖表，其中顯示要簽署之字串的各種元件。例如，最後三行包含下列項目：

- #####
- hash("")

- `hash(current-chunk-data)`

b. 區塊簽章：

```
b474d8862b1487a5145d686f57f013e54db672cee1c953b3010fb58501ef5aa2
```

c. 已傳送的區塊資料：

```
10000;chunk-  
signature=b474d8862b1487a5145d686f57f013e54db672cee1c953b3010fb58501ef5aa2  
<65536-bytes>
```

6. 區塊 2：(1024 個位元組，值 97 表示字母 'a')

a. 要簽署的區塊字串：

```
AWS4-HMAC-SHA256-PAYLOAD  
20130524T000000Z  
20130524/us-east-1/s3/aws4_request  
b474d8862b1487a5145d686f57f013e54db672cee1c953b3010fb58501ef5aa2  
e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855  
2edc986847e209b4016e141a6dc8716d3207350f416969382d431539bf292e4a
```

b. 區塊簽章：

```
1c1344b170168f8e65b41376b44b20fe354e373826ccbbe2c1d40a8cae51e5c7
```

c. 已傳送的區塊資料：

```
400;chunk-  
signature=1c1344b170168f8e65b41376b44b20fe354e373826ccbbe2c1d40a8cae51e5c7  
<1024-bytes>
```

7. 區塊 3：(0 位元組資料)

a. 要簽署的區塊字串：

```
AWS4-HMAC-SHA256-PAYLOAD  
20130524T000000Z  
20130524/us-east-1/s3/aws4_request  
1c1344b170168f8e65b41376b44b20fe354e373826ccbbe2c1d40a8cae51e5c7  
e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855
```

```
e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855
```

b. 區塊簽章：

```
2ca2aba2005185cf7159c6277faf83795951dd77a3a99e6e65d5c9f85863f992
```

c. 已傳送的區塊資料：

```
0;chunk-  
signature=2ca2aba2005185cf7159c6277faf83795951dd77a3a99e6e65d5c9f85863f992
```

8. 區塊 4：追蹤標頭

a. 要簽署的尾端區塊字串：

```
AWS4-HMAC-SHA256-TRAILER  
20130524T000000Z  
20130524/us-east-1/s3/aws4_request  
2ca2aba2005185cf7159c6277faf83795951dd77a3a99e6e65d5c9f85863f992  
1e376db7e1a34a8ef1c4bcee131a2d60a1cb62503747488624e10995f448d774
```

Note

最後兩行是 previous-signature(0 位元組的資料區塊簽章)、
hash(*trailing-checksum-header-name* : *base64-encoded-trailing-checksum-value*\n)。

雜湊的計算方式如下，不含空格：

- 結尾檢查總和標頭名稱
- 冒號 (:)
- base64 編碼的結尾檢查總和值
- 換行字元 (\n)。

在此範例中，如果您使用crc32c雜湊演算法，搭配 base64 編碼檢查總和值 s008/Q==，您可以代表運算，如下所示：hash('x-amz-checksum-crc32c:s008/Q==\n')。

b. 區塊簽章：

```
d81f82fc3505edab99d459891051a732e8730629a2e4a59689829ca17fe2e435
```

c. 已傳送的區塊資料：

```
x-amz-checksum-crc32c:s008/Q==  
x-amz-trailer-  
signature:d81f82fc3505edab99d459891051a732e8730629a2e4a59689829ca17fe2e435
```

驗證請求：使用查詢參數AWS（簽章第 4 版）

如身分驗證概觀中所述（請參閱[身分驗證方法](#)），您可以使用查詢字串參數提供身分驗證資訊。當您想要完全在 URL 中表達請求時，使用查詢參數來驗證請求非常有用。此方法也稱為預先簽署 URL。

預先簽章 URLs 的使用案例是，您可以授予暫時存取 Amazon S3 資源的權限。例如，您可以在網站上嵌入預先簽章的 URL，或將其用於命令列用戶端（例如 Curl）以下載物件。

Note

您也可以使用 AWS CLI 來建立預先簽章URLs。如需詳細資訊，請參閱《AWS CLI 命令參考》中的 [presign](#)。

以下是預先簽章的 URL 範例。

```
https://examplebucket.s3.amazonaws.com/test.txt  
?X-Amz-Algorithm=AWS4-HMAC-SHA256  
&X-Amz-Credential=<your-access-key-id>/20130721/us-east-1/s3/aws4_request  
&X-Amz-Date=20130721T201207Z  
&X-Amz-Expires=86400  
&X-Amz-SignedHeaders=host  
&X-Amz-Signature=<signature-value>
```

在範例 URL 中，請注意下列事項：

- 新增換行以獲得可讀性。

- URL 中的 X-Amz-Credential 值僅為了便於閱讀而顯示 "/" 字元。實際上，它應該編碼為 %2F。例如：

```
&X-Amz-Credential=<your-access-key-id>%2F20130721%2Fus-east-1%2Fs3%2Faws4_request
```

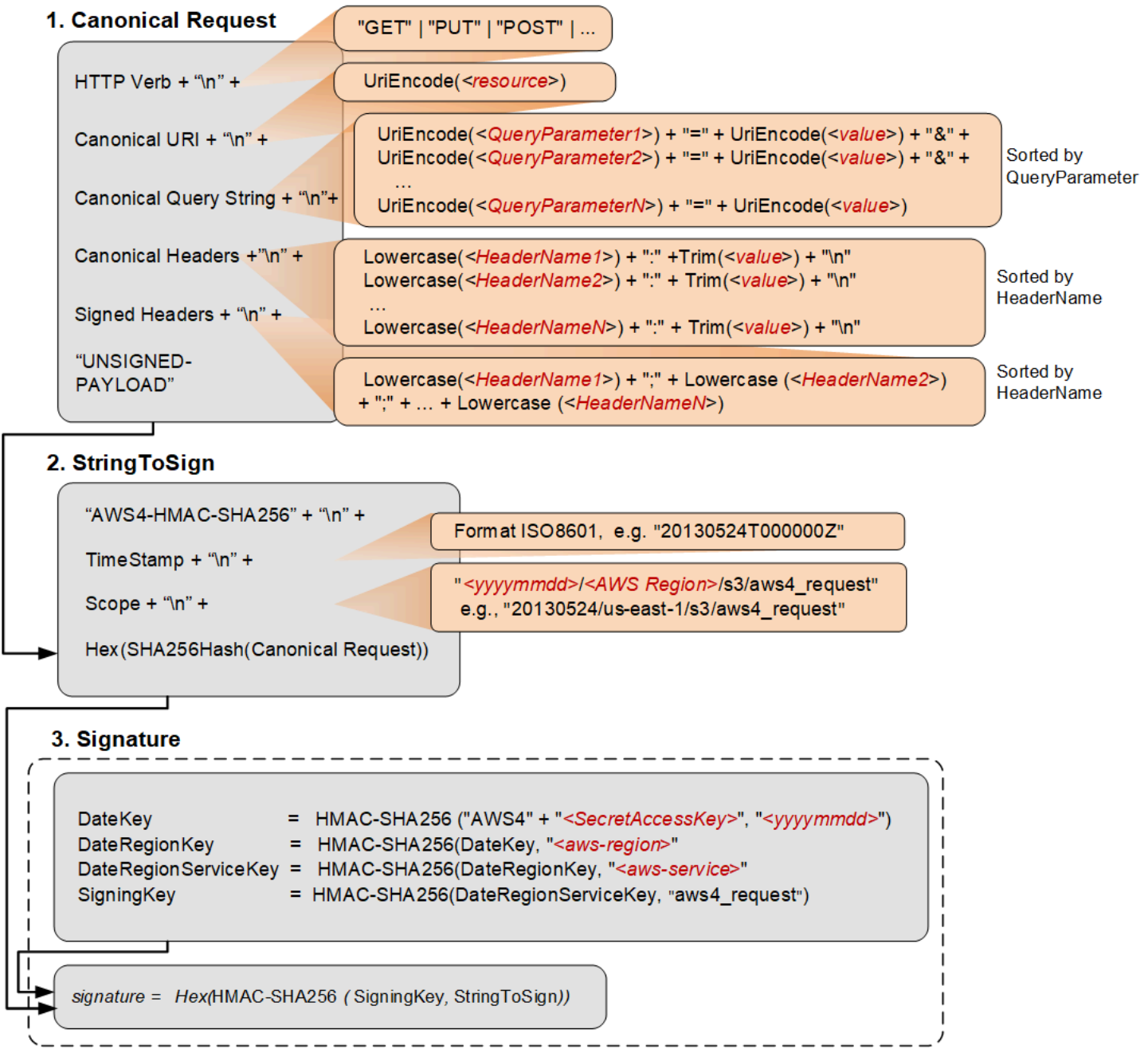
下表描述 URL 中提供驗證資訊的查詢參數。

查詢字串參數名稱	範例值
X-Amz-Algorithm	識別 AWS Signature 的版本，以及您用來計算簽章的演算法。 對於 AWS Signature 第 4 版，您可以將此參數值設定為 AWS4-HMAC-SHA256。此字串可識別 AWS 簽章版本 4 (AWS 4) 和 HMAC-SHA256 演算法 (HMAC-SHA256)。
X-Amz-Credential	除了您的存取金鑰 ID 之外，此參數也提供簽章有效的範圍 (AWS 區域 和服務)。此值必須與您在簽章計算中使用的範圍相符，這將在下一節中討論。此參數值的一般格式如下： <pre><your-access-key-id> /<date>/<AWS ##>/<AWS-service> /aws4_request</pre> 例如： <pre>AKIAIOSFODNN7EXAMPLE/20130721/us-east-1/s3/aw s4_request</pre> 對於 Amazon S3，AWS##字串為 s3。如需 S3 AWS-region 字串的清單，請參閱《》中的區域和端點AWS 一般參考。
X-Amz-Date	日期和時間格式必須遵循 ISO 8601 標準，且格式必須為「yyyyMMdd THHmss Z」格式。例如，如果日期和時間是 "08/01/2016 15:32:41.982-700"，則必須先將其轉換為 UTC (國際標準時間)，然後以 "20160801T223241Z" 形式提交。

查詢字串參數名稱	範例值
X-Amz-Expires	提供所產生預先簽章的 URL 有效的時段 (以秒為單位)。例如，86400 (24 小時)。此值為整數。您可以設定的最小值為 1，最大值為 604800 (七天)。 預先簽章的 URL 最多可有效七天，因為您在簽章計算中使用的簽署金鑰最多可有效七天。
X-Amz-SignedHeaders	列出您用來計算簽章的標頭。簽章計算中需要下列標頭： • HTTP host 標頭。 • 您計劃新增至請求的任何 x-amz-* 標頭。  Note 為了提高安全性，您應簽署計劃包含在請求中的所有請求標頭。
X-Amz-Signature	提供簽章以驗證您的請求。此簽章必須符合 Amazon S3 計算的簽章；否則，Amazon S3 會拒絕請求。例如 733255ef022bec3f2a8701cd61d4b371f3f28c9f193a1f02279211d48d5193d7 簽章計算將在下一節中描述。
X-Amz-Security-Token	選用的憑證參數 (如果使用來自 STS 服務的憑證)。

計算簽章

下圖說明簽章計算程序。



下表描述圖表中顯示的函數。您需要為這些函數實作程式碼。

函式	說明
Lowercase()	將字串轉換成小寫。
Hex()	小寫基數 16 編碼。

函式	說明
SHA256Hash()	安全雜湊演算法 (SHA) 加密雜湊函數。
HMAC-SHA256()	使用 SHA256 演算法搭配提供的簽署金鑰來計算 HMAC。這是最後的簽章。
Trim()	移除任何前置和結尾空格。

函式	說明
UriEncode()	URI 編碼每個位元組。UriEncode() 必須強制執行下列規則： • URI 對除未預留字元之外的每個位元組進行編碼：'A'-'Z', 'a'-'z', '0'-'9', '-', '.', '_', and '~'。 • 空格字元是預留字元，必須編碼為 "%20" (而不是 "+")。 • 每個 URI 編碼的位元組都是由 '%' 和位元組的兩位數十六進位值組成。 • 十六進位值中的字母必須為大寫，例如 "%1A"。 • 對所有位置的正斜線字元 '/' 進行編碼 (物件索引鍵名稱除外)。例如，如果物件索引鍵名為 photos/Jan/sample.jpg，則不會對索引鍵名稱中的正斜線進行編碼。  Important 由於基礎 RFC 中的實作差異和相關模糊性，您的開發平台提供的標準 UriEncode 函數可能無法工作。我們建議您編寫自己的自訂 UriEncode 函數，以確保您的編碼能夠正常工作。 以下是 Java 中的 UriEncode() 函數範例。 <pre data-bbox="613 1312 1448 1852"> public static String UriEncode(CharSequence input, boolean encodeSlash) { StringBuilder result = new StringBui lder(); for (int i = 0; i < input.length(); i++) { char ch = input.charAt(i); if ((ch >= 'A' && ch <= 'Z') (ch >= 'a' && ch <= 'z') (ch >= '0' && ch <= '9') ch == '_' ch == '-' ch == '~' ch == '.') { result.append(ch); } else if (ch == '/') { result.append(encodeSlash ? "%2F" : ch); } else { </pre>

函式	說明
	<pre> result.append(toHexUTF8(ch)); } } return result.toString(); } </pre>

如需簽署程序的詳細資訊（建立正式請求、要簽署的字串和簽章計算的詳細資訊），請參閱 [授權標頭的簽章計算：在單一區塊中傳輸承載AWS（簽章第 4 版）](#)。程序通常相同，但在預先簽章的 URL 中建立 CanonicalRequest 會有所不同，如下所示：

- 正式請求中不包含承載雜湊，因為當您建立預先簽章的 URL 時，您不知道承載內容，因為 URL 用於上傳任意承載。反之，您可以使用常數字串 UNSIGNED-PAYLOAD。
- 正式查詢字串必須包含上述資料表中的所有查詢參數，但除外 X-Amz-Signature。
- 對於 S3，如果使用來自 STS 服務的登入資料，則必須在 URL 中包含 X-Amz-Security-Token 查詢參數。
- 正式標頭必須包含 HTTP host 標頭。如果您計劃包含任何 x-amz-* 標頭，也必須新增這些標頭以進行簽章計算。您可以選擇性地新增計劃包含在請求中的所有其他標頭。為了提高安全性，您應該盡可能簽署多個標頭。如果您新增的已簽章標頭也是已簽章的查詢參數，且其值不同，則當輸入發生衝突時，您會收到 InvalidRequest 錯誤。

範例

假設您的儲存 examplebucket 貯體 test.txt 中有物件。您想要透過建立預先簽章的 URL，與其他人共用此物件 24 小時 (86400 秒)。

```

https://examplebucket.s3.amazonaws.com/test.txt
?X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=AKIAIOSFODNN7EXAMPLE%2F20130524%2Fus-east-1%2Fs3%2Faws4_request
&X-Amz-Date=20130524T000000Z&X-Amz-Expires=86400&X-Amz-SignedHeaders=host
&X-Amz-Signature=<signature-value>

```

下列步驟會先說明簽章計算，然後建構預先簽章的 URL。此範例會進行下列額外假設：

- 請求時間戳記為 Fri, 24 May 2013 00:00:00 GMT。

- 儲存貯體位於美國東部（維吉尼亞北部）區域，而登入資料Scope和Signing Key計算會使用 us-east-1 做為區域指標。如需詳細資訊，請參閱 https://docs.aws.amazon.com/general/latest/gr/rande.html#s3_region 中的 AWS 一般參考區域與端點。

您可以使用此範例做為測試案例來驗證程式碼計算的簽章；不過，您必須使用相同的儲存貯體名稱、物件金鑰、時間戳記和下列範例登入資料：

參數	Value
AWSAccessKeyId	AKIAIOSFODNN7EXAMPLE
AWSSecretAccessKey	wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY

1. StringToSign

a. CanonicalRequest

```
GET
/test.txt
X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIOSFODNN7EXAMPLE
%2F20130524%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20130524T000000Z&X-Amz-Expires=86400&X-Amz-SignedHeaders=host
host:examplebucket.s3.amazonaws.com

host
UNSIGNED-PAYLOAD
```

b. StringToSign

```
AWS4-HMAC-SHA256
20130524T000000Z
20130524/us-east-1/s3/aws4_request
3bfa292879f6447bbcd7001decf97f4a54dc650c8942174ae0a9121cf58ad04
```

2. SigningKey

```
signing key = HMAC-SHA256(HMAC-SHA256(HMAC-SHA256(HMAC-SHA256("AWS4" +
  "<YourSecretAccessKey>", "20130524"), "us-east-1"), "s3"), "aws4_request")
```

3. 簽章

```
aeed9bbccd4d02ee5c0109b86d86835f995330da4c265957d157751f604d404
```

現在您已擁有建構預先簽章 URL 的所有資訊。此範例產生的 URL 如下所示（您可以使用此 URL 來比較預先簽章的 URL）：

```
https://examplebucket.s3.amazonaws.com/test.txt?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIOSFODNN7EXAMPLE%2F20130524%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20130524T000000Z&X-Amz-Expires=86400&X-Amz-SignedHeaders=host&X-Amz-Signature=aeed9bbccd4d02ee5c0109b86d86835f995330da4c265957d157751f604d404
```

範例 2

以下是範例（與上一個範例無關），顯示具有 X-Amz-Security-Token 參數的預先簽章 URL。

```
https://examplebucket.s3.us-east-1.amazonaws.com/test.txt  
?X-Amz-Algorithm=AWS4-HMAC-SHA256  
&X-Amz-Credential=AKIAIOSFODNN7EXAMPLE%2F20130524%2Fus-east-1%2Fs3%2Faws4_request  
&X-Amz-Date=20200524T000000Z&X-Amz-Expires=86400&X-Amz-SignedHeaders=host  
&X-Amz-Security-Token=IQoJb3JpZ2luX2VjEMv%2F%2F%2F%2F%2F%2F%2F%2F%2F%2FwEaCXVzLWVhc3QtMSJGMEQCIBSubvdj9YGs2g0HkHs0HFdkw0ozjARSKHL987NhhOC8AiBPepRU1obMvIbGU0T%2BWphFPgK%2Fqpxaf5Snm5M57XFkCqlAgjz%2F%2F%2F%2F%2F%2F%2F%2F%2F2F8BEAAaDDQ3Mjm4NTU0NDY2MCIM83pULBe5%2F%2BNm1GZBKvkBVslSaJVgwSef7SsoZCJlfJ56weYL3QCWEGr2F4BmCZZyFpmWEYzWnhNK1AnHMj5nkfKlKBx30XAT5PZGVr%2F3HHMOkpdanMXn%2B4PY8lvM8RgnzSu90jOUUpGXEOAo%2F6G80qlMim3%2BZmaQmasn4VYRVESed7O72QGZ3%2BvDnDVnss0LSYjl v8PP7IujnvhzRNj0Woe0yMe1lL0wTG%2Fa9ush5he52w%2FYUJccOn00aZuyROuVsRV4Q70sbWQHuvYUt%2B0tUMKzm8vsFOp4BaNZFqobbjt b36Y92v%2Bx5ky6i0s8QE886jjtUWMP5ldMziClGxp0mN5dzsYlm3GyiJ%2F01mwkpQDwg3mtSpOA9oeeuAMPTA7qmQy9RNUtkBDsx9EW27wvpzBum3SJHEfxv48eudKg rIX3Z79ruQFSQ0c9LUrdJR%2B4SoWAJqK%2BGX8Q3vPSjsLxhqhemMwd6U4TXcm7ku3gxMbzbft8NDg%3D<br>&X-Amz-Signature=<i style='color:red'><signature-value>
```

範例：簽章第 4 版中的 AWS 簽章計算

主題

- [使用 Java AWS 的簽章計算範例（簽章第 4 版）](#)
- [使用 C# 的簽章計算範例AWS（簽章版本 4）](#)

對於已驗證的請求，除非您使用 AWS SDKs，否則您必須撰寫程式碼來計算在請求中提供驗證資訊的簽章。Signature 第 4 版中的 AWS 簽章計算（請參閱[驗證請求AWS（簽章第 4 版）](#)）可能是一項複雜的任務，我們建議您盡可能使用 AWS SDKs。

本節提供以 Java 和 C# 撰寫的簽章計算範例。程式碼範例會傳送下列請求，並使用 HTTP 授權標頭來提供身分驗證資訊：

- PUT 物件 – 個別範例說明一次上傳完整承載並以區塊形式上傳承載。如需使用授權標頭進行身分驗證的詳細資訊，請參閱 [驗證請求：使用授權標頭AWS（簽章第 4 版）](#)。
- GET 物件 – 此範例會產生預先簽章的 URL 以取得物件。查詢參數提供簽章和其他身分驗證資訊。使用者可以在瀏覽器中貼上預先簽章的 URL 來擷取物件，或者您可以使用 URL 來建立可點選的連結。如需使用查詢參數進行身分驗證的詳細資訊，請參閱[驗證請求：使用查詢參數AWS（簽章第 4 版）](#)。

本節其餘部分說明 Java 和 C# 中的範例。這些主題包含下載範例和執行範例的說明。

使用 Java AWS 的簽章計算範例（簽章第 4 版）

顯示簽章計算的 Java 範例可在 <https://docs.aws.amazon.com/AmazonS3/latest/API/samples/AWSS3SigV4JavaSamples.zip> 下載。請注意，這些範例使用適用於 Java 的 AWS SDK v1，我們建議您將[AWS 適用於 Java 的 SDK v2](#) 用於新應用程式。在 `RunAllSamples.java`，`main()` 函數會執行範例請求來建立物件、擷取物件，以及建立物件的預先簽章 URL。範例會從程式碼中提供的文字串建立物件：

```
PutS3ObjectSample.putS3Object(bucketName, regionName, awsAccessKey, awsSecretKey);
GetS3ObjectSample.getS3Object(bucketName, regionName, awsAccessKey, awsSecretKey);
PresignedUrlSample.getPresignedUrlToS3Object(bucketName, regionName, awsAccessKey,
awsSecretKey);
PutS3ObjectChunkedSample.putS3ObjectChunked(bucketName, regionName, awsAccessKey,
awsSecretKey);
```


在 Linux 型電腦上測試範例

下列指示適用於 Linux 作業系統。

1. 在終端機中，導覽至包含的目錄 `AWSS3SigV4JavaSamples.zip`。
2. 解壓縮 `.zip` 檔案。
3. 在文字編輯器中開啟檔案 `./com/amazonaws/services/s3/samples/RunAllSamples.java`。使用下列資訊更新程式碼：
 - 可以建立新物件的儲存貯體名稱。

Note

這些範例使用虛擬託管樣式請求來存取儲存貯體。為了避免潛在的錯誤，請確保您的儲存貯體名稱符合 Amazon Simple Storage Service 使用者指南中的 [儲存貯體限制和限制中所述](#) 的儲存貯體命名規則。

- AWS 區域 儲存貯體所在的位置。

如果儲存貯體位於美國東部（維吉尼亞北部）區域，請使用 `us-east-1` 指定區域。如需其他的清單 AWS 區域，請前往 [Amazon Simple Storage Service \(S3\)](#) AWS 一般參考。

4. 編譯原始程式碼，並將編譯的類別存放在 `bin/` 目錄中。

```
javac -d bin -source 6 -verbose com
```

5. 將目錄變更為 `bin/`，然後執行 `RunAllSamples`。

```
java com.amazonaws.services.s3.sample.RunAllSamples
```

此程式碼會執行中的所有方法 `main()`。對於每個請求，輸出將顯示正式請求、要簽署的字串和簽章。

使用 C# 的簽章計算範例AWS（簽章版本 4）

顯示簽章計算的 C# 範例可在 https://docs.aws.amazon.com/AmazonS3/latest/API/samples/AmazonS3SigV4_Samples_CSharp.zip 下載。請注意，這些範例使用適用於 .NET v1 的 AWS SDK，

我們建議您針對新應用程式使用[AWS 適用於 .NET v3 的 SDK](#)。在中 Program.cs , main() 函數會執行範例請求來建立物件、擷取物件，以及建立物件的預先簽章 URL。簽章計算的程式碼位於 \Signers 資料夾中。

```
PutS3ObjectSample.Run(awsRegion, bucketName, "MySampleFile.txt");

Console.WriteLine("\n\n*****");
PutS3ObjectChunkedSample.Run(awsRegion, bucketName, "MySampleFileChunked.txt");

Console.WriteLine("\n\n*****");
GetS3ObjectSample.Run(awsRegion, bucketName, "MySampleFile.txt");

Console.WriteLine("\n\n*****");
PresignedUrlSample.Run(awsRegion, bucketName, "MySampleFile.txt");
```

使用 Microsoft Visual Studio 2010 或更新版本測試範例

1. 解壓縮 .zip 檔案。
2. 啟動 Visual Studio，然後開啟 .sln 檔案。
3. 使用有效的安全登入資料更新 App.config 檔案。
4. 更新程式碼，如下所示：
 - 在中 Program.cs，提供儲存貯 AWS 區域 體名稱和儲存貯體所在的。此範例會在此儲存貯體中建立物件。
5. 執行程式碼。
6. 若要確認物件已建立，請複製程式建立的預先簽章 URL，然後將其貼入瀏覽器視窗。

驗證請求：使用 POST 以瀏覽器為基礎的上傳AWS（簽章第 4 版）

Amazon S3 支援 HTTP POST 請求，讓使用者可以直接將內容上傳至 Amazon S3。使用 HTTP POST 上傳內容可簡化上傳，並減少使用者上傳資料以存放在 Amazon S3 中的上傳延遲。本節說明如何驗證 HTTP POST 請求。如需 HTTP POST 請求、如何建立表單、建立 POST 政策以及範例的詳細資訊，請參閱 [使用 POST 的瀏覽器型上傳AWS（簽章第 4 版）](#)。

若要驗證 HTTP POST 請求，請執行下列動作：

1. 表單必須包含下列欄位，以提供 Amazon S3 在收到請求時可用來重新計算簽章的簽章和相關資訊：

元素名稱	說明
policy	描述請求中允許內容的 Base64-encoded 安全政策。對於簽章計算，此政策是您簽署的字串。Amazon S3 必須取得此政策，才能重新計算簽章。
x-amz-algorithm	使用的簽署演算法。對於 AWS Signature 第 4 版，值為 AWS4-HMAC-SHA256 。
x-amz-credential	除了您的存取金鑰 ID 之外，這還提供了您在計算簽章計算的簽署金鑰時使用的範圍資訊。 這是以下格式的字串： <pre><your-access-key-id> /<date>/<aws-region> /<aws-service> /aws4_request</pre> 例如： <pre>AKIAIOSFODNN7EXAMPLE/20130728/us-east-1/s3/aws4_request . .</pre> 對於 Amazon S3，aws-service 字串為 s3。如需 Amazon S3 aws-region 字串的清單，請參閱 中的區域和端點 AWS 一般參考。
x-amz-date	這是 ISO8601 格式的日期值。例如 20130728T000000Z 。 這是您在建立簽署金鑰時使用的相同日期。這也必須與您在簽署的政策 (x-amz-date) 中提供的值相同。

元素名稱	說明
x-amz-signature	(AWS 簽章版本 4) 安全政策的 HMAC-SHA256 雜湊。 如需簽章選項的詳細資訊，請參閱《》中的 將簽章新增至 HTTP 請求 AWS 一般參考。

2. POST 政策必須包含下列元素：

元素名稱	說明
x-amz-algorithm	您用來計算簽章的簽署演算法。對於 AWS Signature 第 4 版，值為 AWS4-HMAC-SHA256 。
x-amz-credential	除了您的存取金鑰 ID 之外，這還提供了您在計算簽章計算的簽署金鑰時使用的範圍資訊。 這是以下格式的字串： <i><your-access-key-id> /<date>/<aws-region> /<aws-service> /aws4_request</i> 例如 AKIAIOSFODNN7EXAMPLE/20130728/us-east-1/s3/aws4_request . .
x-amz-date	ISO8601 格式字串中指定的日期值。例如，「20130728T000000Z」。日期必須與您在建立簽章計算的簽署金鑰時所使用的日期相同。

3. 對於簽章計算，POST 政策是要簽署的字串。

計算簽章

下圖說明簽章計算程序。



計算簽章

1. 使用 UTF-8 編碼建立政策。
2. 將 UTF-8-encoded政策轉換為 Base64。結果是要簽署的字串。
3. 建立簽章做為要簽署之字串的 HMAC-SHA256 雜湊。您將提供簽署金鑰做為雜湊函數的金鑰。
4. 使用十六進位編碼來編碼簽章。

如需建立 HTML 表單、安全政策和範例的詳細資訊，請參閱下列子主題：

- [建立 HTML 表單（使用 AWS Signature 第 4 版）](#)
- [POST 政策](#)
- [範例：使用 HTTP POST 以瀏覽器為基礎的上傳（使用 AWS Signature 第 4 版）](#)

Amazon S3 Signature 第 4 版身分驗證特定政策金鑰

下表顯示可在 Amazon S3 政策中與 Amazon S3 Signature 第 4 版身分驗證相關的政策金鑰。在儲存貯體政策中，您可以新增這些條件，以便在使用第 4 版簽署程序來驗證請求時強制執行特定行為。如需範例政策，請參閱 [使用 Signature 第 4 版相關條件索引鍵的儲存貯體政策範例](#)。

適用的金鑰	說明
s3:signatureversion	識別您要支援已驗證請求的 AWS Signature 版本。對於已驗證的請求，Amazon S3 同時支援 Signature 第 4 版和 Signature 第 2 版。您可以在儲存貯體政策中新增此條件，以要求特定的簽章版本。 有效值： "AWS" 識別簽章版本 2 "AWS4-HMAC-SHA256" 識別 Signature 第 4 版
s3:authType	Amazon S3 支援各種身分驗證方法（請參閱 驗證請求AWS（簽章第 4 版））。您可以選擇性地使用此條件金鑰來限制傳入的請求，以使用特定的身分驗證方法。例如，您可以只允許在請求身分驗證中使用 HTTP Authorization 標頭。 有效值： REST-HEADER REST-QUERY-STRING POST
s3:signatureAge	簽章在已驗證請求中有效的時間長度 (以毫秒為單位)。 此條件適用於：

適用的金鑰	說明
	• 預先簽章URLs — 其中最嚴格的條件獲勝。如需詳細資訊，請參閱使用預先簽章URLs。 • 預先簽章的 POST — 使用預先簽章的 POST 將檔案直接上傳至 S3。如需詳細資訊，請參閱 Amazon S3 POST 政策。 在 Signature 第 2 版中，此值一律設定為 0。 在 第 4 版簽署程序中，簽署金鑰的有效期限最長七天。因此，簽章的有效期限也是最長七天。您可以使用此條件來進一步限制簽章存留期。如需詳細資訊，請參閱簽署請求簡介。 範例值：100
s3:x-amz-content-sha256	您可以使用此條件金鑰以不允許在儲存貯體中未簽署的內容。 當您使用第 4 版簽署程序時，針對使用 Authorization 標頭的請求，會在簽署計算中新增 x-amz-content-sha256 標頭，然後將其值設定為雜湊承載。請注意，此條件索引鍵不支援 x-amz-content-sha256 標頭做為查詢字串參數。 您可以在儲存貯體政策中使用此條件金鑰，以拒絕未簽署承載的任何上傳。例如，您可以拒絕使用 Authorization 標頭來驗證請求，但不簽署承載的上傳。如需詳細資訊，請參閱授權標頭的簽章計算：在單一區塊中傳輸承載AWS（簽章第 4 版）。 有效值：UNSIGNED-PAYLOAD

使用 Signature 第 4 版相關條件索引鍵的儲存貯體政策範例

examplebucket 如果簽章超過十分鐘，下列儲存貯體政策會拒絕 中物件的任何 Amazon S3 預先簽章 URL 請求。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Deny a presigned URL request if the signature is more than 10 min old",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::examplebucket3/*",
      "Condition": {
        "NumericGreaterThan": {
          "s3:signatureAge": 600000
        }
      }
    }
  ]
}
```

下列儲存貯體政策僅允許使用 Authorization 標頭以供請求驗證的任何請求。任何 POST 或預先簽章的 URL 請求都會遭到拒絕。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow only requests that use Authorization header for request authentication. Deny POST or presigned URL requests.",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:",
    }
  ]
}
```



```

        "Resource": "arn:aws:s3:::examplebucket3/*",
        "Condition": {
            "StringNotEquals": {
                "s3:authType": "REST-HEADER"
            }
        }
    }
]
}

```

下列儲存貯體政策拒絕使用預先簽章 URLS 進行請求身分驗證的請求：

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyUploadsUsingPresignedURL",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/*",
      "Condition": {
        "StringEquals": {
          "s3:authType": "REST-query-string"
        }
      }
    }
  ]
}

```

下列儲存貯體政策會拒絕任何具有未簽署承載的上傳：

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [

```

```
{
  "Sid": "Deny uploads with unsigned payloads that use the
Authorization header.",
  "Effect": "Deny",
  "Principal": "*",
  "Action": "s3:*",
  "Resource": "arn:aws:s3:::examplebucket3/*",
  "Condition": {
    "StringEquals": {
      "s3:x-amz-content-sha256": "UNSIGNED-PAYLOAD"
    }
  }
}
```

使用 POST 的瀏覽器型上傳AWS（簽章第 4 版）

本節討論如何使用 HTTP POST 請求，透過瀏覽器將檔案直接上傳至 Amazon S3。它還包含有關如何使用 AWS Amplify JavaScript 程式庫將瀏覽器型檔案上傳至 Amazon S3 的資訊。

主題

- [POST 物件](#)
- [POST 物件還原](#)
- [使用 HTTP POST 的瀏覽器型上傳](#)
- [計算簽章](#)
- [建立 HTML 表單（使用 AWS Signature 第 4 版）](#)
- [POST 政策](#)
- [範例：使用 HTTP POST 以瀏覽器為基礎的上傳（使用 AWS Signature 第 4 版）](#)
- [使用 AWS Amplify 程式庫將瀏覽器型上傳到 Amazon S3](#)

POST 物件

說明

POST 操作會使用 HTML 表單將物件新增至指定的儲存貯體。POST 是 的替代形式PUT，可啟用瀏覽器型上傳，做為將物件放入儲存貯體的方式。PUT 透過 HTTP 標頭傳遞至 的參數，會改為在分段/表單編碼訊息內文POST中以表單欄位形式傳遞至 。若要將物件新增至儲存貯體，您必須在儲存貯體上具有WRITE存取權。Amazon S3 絕不會存放部分物件。如果您收到成功的回應，您可以確信儲存了整個物件。

Amazon S3 是分散式系統。除非您已啟用儲存貯體的版本控制，否則如果 Amazon S3 同時收到相同物件的多個寫入請求，則只會儲存最後一個寫入的物件版本。

若要確保資料在周遊網路時不會損毀，請使用 Content-MD5 表單欄位。當您使用此表單欄位時，Amazon S3 會根據提供的 MD5 值檢查物件。如果不相符，Amazon S3 會傳回錯誤。此外，您可以在將物件發佈至 Amazon S3 時計算 MD5 值，並將傳回的 ETag 與計算的 MD5 值進行比較。ETag 僅反映物件內容的變更，而不是其中繼資料。

Note

若要設定應用程式在傳送請求內文之前傳送請求標頭，請使用 HTTP 狀態碼 100（繼續）。對於 POST 操作，如果根據標頭拒絕訊息（例如，因為身分驗證失敗或重新導向），使用此狀態碼可協助您避免傳送訊息內文。如需 HTTP 狀態碼 100（繼續）的詳細資訊，請前往 <http://www.ietf.org/rfc/rfc2616.txt>：// 的第 8.2.3 節。

Amazon S3 會自動加密上傳到 S3 儲存貯體的所有新物件。上傳物件的加密設定取決於目的地儲存貯體的預設加密組態。根據預設，所有儲存貯體的預設加密組態都是使用 Amazon S3 受管金鑰的伺服器端加密 (SSE-S3)。

如果目的地儲存貯體的加密組態使用伺服器端加密搭配 AWS Key Management Service (AWS KMS) 金鑰 (SSE-KMS)、雙層伺服器端加密搭配 AWS KMS 金鑰 (DSSE-KMS) 或客戶提供的加密金鑰 (SSE-C)，Amazon S3 會使用對應的 KMS 金鑰或客戶提供的金鑰來加密上傳的物件。上傳物件時，如果您想要變更上傳物件的加密設定，您可以指定伺服器端加密的類型。您可以設定 SSE-S3、SSE-KMS、DSSE-KMS 或 SSE-C。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[使用伺服器端加密保護資料](#)。

Important

建構請求時，請確定 file 欄位是表單中的最後一個欄位。

版本控制

如果您為儲存貯體啟用版本控制，POST會自動為要新增的物件產生唯一的版本 ID。Amazon S3 會使用回應標頭在x-amz-version-id回應中傳回此 ID。

如果您暫停儲存貯體的版本控制，Amazon S3 一律使用 null做為儲存貯體中所存放物件的版本 ID。

如需傳回儲存貯體版本控制狀態的詳細資訊，請參閱 [GetBucketVersioning](#)。

Amazon S3 是分散式系統。如果您為儲存貯體啟用版本控制，且 Amazon S3 同時收到相同物件的多個寫入請求，則會儲存物件的所有版本。

若要查看使用版本控制的範例請求，請參閱[範例請求](#)。

要求

語法

```
POST / HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
User-Agent: browser_data
Accept: file_types
Accept-Language: Regions
Accept-Encoding: encoding
Accept-Charset: character_set
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=9431149156168
Content-Length: length

--9431149156168
Content-Disposition: form-data; name="key"

acl
--9431149156168
Content-Disposition: form-data; name="tagging"
```

```
<Tagging><TagSet><Tag><Key>Tag Name</Key><Value>Tag Value</Value></Tag></TagSet></Tagging>
--9431149156168
Content-Disposition: form-data; name="success_action_redirect"

success_redirect
--9431149156168
Content-Disposition: form-data; name="Content-Type"

content_type
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-uuid"

uuid
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-tag"

metadata
--9431149156168
Content-Disposition: form-data; name="AWSAccessKeyId"

access-key-id
--9431149156168
Content-Disposition: form-data; name="Policy"

encoded_policy
--9431149156168
Content-Disposition: form-data; name="Signature"

signature=
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename.jpg"
Content-Type: image/jpeg

file_content
--9431149156168
Content-Disposition: form-data; name="submit"

Upload to Amazon S3
--9431149156168--
```

請求參數

此實作操作不使用請求參數。

表單欄位

此操作可以使用下列表單欄位。

名稱	描述	必要
AWSAccessKeyId	儲存貯體擁有者的 AWS 存取金鑰 ID，其會針對滿足政策中一組限制的請求授予匿名使用者存取。 類型：字串 預設：無 限制條件：如果請求中包含政策文件，則為必要。	有條件
acl	指定的 Amazon S3 存取控制清單 (ACL)。如果指定的 ACL 無效，則會產生錯誤。如需 ACLs 的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 存取控制清單 (ACL) 概觀。 類型：字串 預設值：private 有效值: private public-read public-read-write aws-exec-read authenticated-read bucket-owner-read bucket-owner-full-control	否
Cache-Control , Content-Type , Content-Disposition , Content-Encoding , Expires	REST 特定的標頭。如需詳細資訊，請參閱 PutObject。 類型：字串 預設：無	否
file	檔案或文字內容。	是

名稱	描述	必要
	檔案或文字內容必須是表單中的最後一個欄位。 您無法一次上傳多個檔案。 類型：檔案或文字內容 預設：無	
key	上傳的金鑰名稱。 若要使用使用者提供的檔案名稱，請使用 <code>\${filename}</code> 變數。例如，如果名為 Mary 的使用者上傳檔案，<code>example.jpg</code> 而您指定 <code>/user/mary/\${filename}</code>，則金鑰名為 <code>/user/mary/example.jpg</code>。 如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 物件金鑰和中繼資料。 類型：字串 預設：無	是
policy	描述請求中允許內容的安全政策。沒有安全政策的請求會被視為匿名，並且僅適用於可公開寫入的儲存貯體。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 HTML 表單和上傳範例。 類型：字串 預設：無 限制條件：如果儲存貯體無法公開寫入，則需要安全政策。	有條件

名稱	描述	必要
<code>success_action_redirect</code> , <code>redirect</code>	用戶端在成功上傳時重新導向的 URL。 若未指定 <code>success_action_redirect</code> , Amazon S3 會傳回 <code>success_action_status</code> 欄位中所指定的空白文件類型。 如果 Amazon S3 無法解譯 URL , 就好像 欄位不存在一樣。 如果上傳失敗 , Amazon S3 會顯示錯誤 , 而不會將使用者重新導向至 URL。 類型 : 字串 預設 : 無  Note <code>redirect</code> 欄位名稱已棄用 , 未來將移除對 <code>redirect</code> 欄位名稱的支援。	否
<code>success_action_status</code>	如果您未指定 <code>success_action_redirect</code> , 狀態碼會在上傳成功時傳回給用戶端。 此欄位接受值 200、201 或 204 (預設值) 。 如果值設定為 200 或 204 , Amazon S3 會傳回狀態碼為 200 或 204 的空白文件。 如果值設定為 201 , Amazon S3 會傳回狀態碼為 201 的 XML 文件。 如果未設定 值 , 或將其設定為無效的值 , Amazon S3 會傳回狀態碼為 204 的空白文件。 類型 : 字串 預設 : 無	否

名稱	描述	必要
tagging	要新增至物件的指定標籤集。若要新增標籤，請使用下列編碼方案。 <pre><Tagging> <TagSet> <Tag> <Key>TagName</Key> <Value>TagValue</Value> </Tag> ... </TagSet> </Tagging></pre> 如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的物件標記。 類型：字串 預設：無	否
x-amz-storage-class	用於存放物件的儲存類別。如果您未指定類別，Amazon S3 會使用預設儲存類別 STANDARD。Amazon S3 支援其他儲存類別。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的儲存類別。 類型：字串 預設：STANDARD 有效值：REDUCED_REDUNDANCY EXPRESS_ONEZONE DEEP_ARCHIVE GLACIER GLACIER_IR INTELLIGENT_TIERING ONEZONE_IA STANDARD STANDARD_IA	否

名稱	描述	必要
x-amz-meta-*	以此字首開頭的標頭是使用者定義的中繼資料。每個項目都會儲存並傳回為一組索引鍵/值組。Amazon S3 不會驗證或解譯使用者定義的中繼資料。如需詳細資訊，請參閱 PutObject。 類型：字串 預設：無	否
x-amz-security-token	Amazon DevPay 安全性字符。 每個使用 Amazon DevPay 的請求都需要兩個 x-amz-security-token 表單欄位：一個表示產品字符，另一個表示使用者字符。 類型：字串 預設：無	否
x-amz-signature	(AWS 簽章版本 4) 安全政策的 HMAC-SHA256 雜湊。 類型：字串 預設：無	有條件

名稱	描述	必要
x-amz-website-redirect-location	如果儲存貯體設定為網站，此欄位會將此物件的請求重新導向至相同儲存貯體中的另一個物件或外部 URL。Amazon S3 會將此標頭的值儲存在物件中繼資料中。如需有關物件中繼資料的資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的物件金鑰和中繼資料。 在下列範例中，請求標頭會將重新導向設定為相同儲存貯體中的物件 (anotherPage.html)： <pre>x-amz-website-redirect-location: /anotherPage.html</pre> 在下列範例中，請求標頭會將物件重新導向設定為另一個網站： <pre>x-amz-website-redirect-location: http://www.example.com/</pre> 如需在 Amazon S3 中託管網站的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的在Amazon S3 上託管網站以及如何設定網站頁面重新導向。 類型：字串 預設：無 限制條件：值必須以 /、http://或 為字首https://。值的長度限制為 2 KB。	否

其他檢查總和申請表欄位

上傳物件時，您可以指定要用來驗證資料完整性的各種檢查總和。您可以指定一個額外的檢查總和演算法供 Amazon S3 使用。如需其他檢查總和值的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[檢查物件完整性](#)。

名稱	描述	必要
x-amz-checksum-algorithm	指出用來建立物件檢查總和的演算法。如果指定了值，您必須包含相符的檢查總和標頭。否則，您的請求會產生 400 錯誤。 可能的值包括 CRC32、CRC32C、SHA1 與 SHA256。	否
x-amz-checksum-crc32	指定物件的 base64 編碼、32 位元 CRC32 檢查總和。 如果的值 <code>x-amz-checksum-algorithm</code> 為 <code>crc32</code> ，則需要此參數 CRC32。	有條件
x-amz-checksum-crc32c	指定物件的 base64 編碼、32 位元 CRC32C 檢查總和。 如果的值 <code>x-amz-checksum-algorithm</code> 為 <code>crc32c</code> ，則需要此參數 CRC32C。	有條件
x-amz-checksum-sha1	指定物件的 base64 編碼、160 位元 SHA-1 摘要。 如果的值 <code>x-amz-checksum-algorithm</code> 為 <code>sha1</code> ，則需要此參數 SHA1。	有條件
x-amz-checksum-sha256	指定物件的 base64 編碼、256 位元 SHA-256 摘要。 如果的值 <code>x-amz-checksum-algorithm</code> 為 <code>sha256</code> ，則需要此參數 SHA256。	有條件

伺服器端加密特定請求表單欄位

伺服器端加密是靜態資料加密。Amazon S3 會在將資料寫入 AWS 資料中心的磁碟時加密資料，並在您存取資料時解密資料。上傳物件時，您可以指定您希望 Amazon S3 用來加密物件的伺服器端加密類型。

伺服器端加密有四種類型：

- 使用 Amazon S3 受管金鑰 (SSE-S3) 的伺服器端加密 – 從 2022 年 5 月開始，所有 Amazon S3 儲存貯體預設都會設定加密。伺服器端加密的預設選項是使用 SSE-S3。每個物件都使用不重複的金鑰加密。SSE-S3 使用定期輪換的根金鑰自行加密金鑰，提供額外的防護。SSE-S3 使用目前最強大的其中一種區塊加密法 (256 位元進階加密標準 (AES-256))，加密您的資料。

- 伺服器端加密與 AWS KMS 金鑰 (SSE-KMS) – SSE-KMS 透過整合 AWS KMS 服務與 Amazon S3 來提供。使用 AWS KMS，您可以更好地控制您的金鑰。例如，您可以檢視個別金鑰、編輯控制政策，並遵循 AWS CloudTrail 中的金鑰。此外，您可以建立和管理客戶受管金鑰，或是使用您、您服務和您區域唯一的 AWS 受管金鑰。
- 使用 AWS KMS 金鑰的雙層伺服器端加密 (DSSE-KMS) – 使用 AWS KMS keys (DSSE-KMS) 的雙層伺服器端加密類似於 SSE-KMS，但會套用兩個個別的物件層級加密，而不是一層。
- 伺服器端加密搭配客戶提供的金鑰 (SSE-C) – 透過 SSE-C，您可以管理加密金鑰，Amazon S3 會在寫入磁碟時管理加密，並在您存取物件時管理解密。

 Note

如果您的一般用途儲存貯體封鎖了客戶提供的金鑰 (SSE-C) 的伺服器端加密，則在將新資料寫入儲存貯體時，當您指定 SSE-C 請求標頭時，將會收到 HTTP 403 存取遭拒錯誤。如需詳細資訊，請參閱 [一般用途儲存貯體的封鎖或取消封鎖 SSE-C](#)。

如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [使用伺服器端加密保護資料](#)。

根據您要使用的伺服器端加密類型，指定下列表單欄位。

- 使用 SSE-S3、SSE-KMS 或 DSSE-KMS – 如果您想要使用這些類型的伺服器端加密，請在請求中指定下列表單欄位。

名稱	描述	必要
x-amz-server-side-encryption	指定 Amazon S3 建立物件時要使用的伺服器端加密演算法。若要使用 SSE-S3，請指定 AES256。若要使用 SSE-KMS，請指定 <code>aws:kms</code> 。若要使用 DSSE-KMS，請指定 <code>aws:kms:dsse</code> 。 類型：字串 有效值： <code>aws:kms</code> 、 <code>AES256</code> 、 <code>aws:kms:dsse</code>	是
x-amz-server-side-encryption	如果 <code>x-amz-server-side-encryption</code> 標頭的有效值為 <code>aws:kms</code> 或 <code>aws:kms:dsse</code> ，此標頭會指定 AWS KMS key 用來加密物件的 ID。	是，如果的值 <code>x-amz-ser</code>

名稱	描述	必要
n-aws-kms-key-id	類型：字串	version 是 aws:kms 或 aws:kms:dsse
x-amz-server-side-encryption-context	如果 x-amz-server-side-encryption 的有效值為 aws:kms 或 aws:kms:dsse，則此標頭會指定物件的加密內容。此標頭的值是 base64 編碼的 UTF-8 字串，其中包含加密內容的 JSON 格式金鑰值對。 類型：字串	否
x-amz-server-side-encryption-bucket-key-enabled	如果 x-amz-server-side-encryption 的有效值為 aws:kms 或 aws:kms:dsse，此標頭會指定 Amazon S3 是否應使用 S3 儲存貯體金鑰搭配 SSE-KMS 或 DSSE-KMS。將此標頭設定為 true 會導致 Amazon S3 使用 S3 儲存貯體金鑰進行 SSE-KMS 或 DSSE-KMS 的物件加密。 類型：布林值	否

Note

如果您指定 x-amz-server-side-encryption:aws:kms 或 x-amz-server-side-encryption:aws:kms:dsse，但未提供 x-amz-server-side-encryption-aws-kms-key-id，Amazon S3 會使用 AWS 受管金鑰 (aws/S3) 來保護資料。

- 使用 SSE-C – 如果您想要管理自己的加密金鑰，您必須在請求中提供下列所有表單欄位。

Note

如果您使用 SSE-C，Amazon S3 在回應中傳回 ETag 的值不是物件的 MD5。

名稱	描述	必要
<code>x-amz-server-side-encryption-customer-algorithm</code>	指定加密物件時要使用的演算法。 類型：字串 預設：無 有效值：AES256 限制條件：必須隨附有效的 <code>x-amz-server-side-encryption-customer-key</code> 和 <code>x-amz-server-side-encryption-customer-key-MD5</code> 欄位。	是
<code>x-amz-server-side-encryption-customer-key</code>	指定客戶提供的 base64 編碼加密金鑰，讓 Amazon S3 用於加密資料。此值用於存放物件，然後將其捨棄。Amazon 不會存放加密金鑰。金鑰必須適合與 <code>x-amz-server-side-encryption-customer-algorithm</code> 標頭中指定的演算法搭配使用。 類型：字串 預設：無 限制條件：必須隨附有效的 <code>x-amz-server-side-encryption-customer-algorithm</code> 和 <code>x-amz-server-side-encryption-customer-key-MD5</code> 欄位。	是

名稱	描述	必要
x-amz-server-side-encryption-customer-key-MD5	根據 RFC 1321 指定加密金鑰的 base64 編碼 128 位元 MD5 摘要。Amazon S3 使用此標頭進行訊息完整性檢查，以確保加密金鑰的傳輸沒有錯誤。 類型：字串 預設：無 限制條件：必須隨附有效的 x-amz-server-side-encryption-customer-algorithm 和 x-amz-server-side-encryption-customer-key 欄位。	是

回應

回應標頭

除了所有回應通用的回應標頭之外，此操作的實作還可以包含下列回應標頭。如需詳細資訊，請參閱[常見回應標頭](#)。

名稱	描述
x-amz-checksum-crc32	物件的 base64 編碼、32 位元 CRC32 檢查總和。 類型：字串
x-amz-checksum-crc32c	物件的 base64 編碼、32 位元 CRC32C 檢查總和。 類型：字串
x-amz-checksum-sha1	物件的 base64 編碼、160 位元 SHA-1 摘要。 類型：字串
x-amz-checksum-sha256	物件的 base64 編碼、256 位元 SHA-256 摘要。 類型：字串

名稱	描述
x-amz-expiration	如果已為物件設定Expiration 動作做為儲存貯體生命週期組態的一部分，Amazon S3 會傳回此標頭。標頭值包含 expiry-date 元件和 URL 編碼rule-id元件。對於已啟用版本的儲存貯體，此標頭僅適用於目前的版本。Amazon S3 不提供標頭來指出非目前版本何時符合永久刪除的資格。如需詳細資訊，請參閱 PutBucketLifecycleConfiguration。 類型：字串
success_action_redirect, redirect	用戶端在成功上傳時重新導向到的 URL。 類型：字串 Ancestor：PostResponse
x-amz-server-side-encryption	在 Amazon S3 中存放此物件時所使用的伺服器端加密演算法（例如，、AES256aws:kms、aws:kms:dssse ）。 類型：字串
x-amz-server-side-encryption-aws-kms-key-id	如果x-amz-server-side-encryption 標頭的有效值為 aws:kms，此標頭會指定用來加密物件的 KMS 金鑰 ID。 類型：字串
x-amz-server-side-encryption-bucket-key-enabled	如果 x-amz-server-side-encryption 的有效值為 aws:kms，此標頭會指出物件是否使用 S3 儲存貯體金鑰以 SSE-KMS 加密。如果此標頭設定為 true，物件會使用 S3 儲存貯體金鑰搭配 SSE-KMS。 類型：布林值

名稱	描述
x-amz-server-side-encryption-customer-algorithm	如果請求 SSE-C，回應會包含此標頭，確認使用的加密演算法。 類型：字串 有效值: AES256
x-amz-server-side-encryption-customer-key-MD5	如果請求 SSE-C，回應會包含此標頭，以驗證客戶提供加密金鑰的往返訊息完整性。 類型：字串
x-amz-version-id	物件的版本。 類型：字串

回應元素

名稱	描述
Bucket	物件存放所在的儲存貯體名稱。 類型：字串 Ancestor：PostResponse
ETag	實體標籤 (ETag) 是物件的 MD5 雜湊，您可以使用 If-Modified 請求標籤搭配 GET 請求 GET 操作來執行條件式操作。只會 ETag 反映物件內容的變更，而不會反映至其中繼資料。 類型：字串 Ancestor：PostResponse
Key	物件金鑰名稱。 類型：字串

名稱	描述
	Ancestor : PostResponse
Location	物件的 URI。 類型：字串 Ancestor : PostResponse

特殊錯誤

此操作的實作不會傳回特殊錯誤。如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱[錯誤回應](#)。

範例

請求範例

```
POST /Neo HTTP/1.1
Content-Length: 4
Host: quotes.s3.amazonaws.com
Date: Wed, 01 Mar 2006 12:00:00 GMT
Authorization: authorization string
Content-Type: text/plain
Expect: the 100-continue HTTP status code
```

ObjectContent

暫停版本控制的回應範例

以下是儲存貯體版本控制暫停時的回應範例：

```
HTTP/1.1 100 Continue
HTTP/1.1 200 OK
x-amz-id-2: LriYPLdm0dAiIfgSm/F1YsViT1LW94/xUQxMsF7xiEb1a0wiI0Ix1+zbwZ163pt7
x-amz-request-id: 0A49CE4060975EAC
x-amz-version-id: default
Date: Wed, 12 Oct 2009 17:50:00 GMT
ETag: "1b2cf535f27731c974343645a3985328"
Content-Length: 0
```

```
Connection: close
Server: AmazonS3
```

在此回應中，版本 ID 為 null。

啟用版本控制的範例回應

以下是啟用儲存貯體版本控制時的範例回應。

```
HTTP/1.1 100 Continue
HTTP/1.1 200 OK
x-amz-id-2: LriYPLdmOdAiIfgSm/F1YsViT1LW94/xUQxMsF7xiEb1a0wiIOIx1+zbwZ163pt7
x-amz-request-id: 0A49CE4060975EAC
x-amz-version-id: 43jfkodU8493jnFJD9fjj3HHNVfdsQUIFDNsidf038jfdsjGFDSIRp
Date: Wed, 01 Mar 2006 12:00:00 GMT
ETag: "828ef3fdfa96f00ad9f27c383fc9ac7f"
Content-Length: 0
Connection: close
Server: AmazonS3
```

相關資源

- [CopyObject](#)
- [POST 物件](#)
- [GetObject](#)

POST 物件還原

說明

此操作會執行下列類型的請求：

- `select` – 在封存的物件上執行選取查詢
- `restore an archive` – 還原封存的物件

若要使用此操作，您必須具有執行 `s3:RestoreObject` 和 `s3:GetObject` 動作的許可。根據預設，儲存貯體擁有者具有此許可，並且可以將此許可授予其他人。如需許可的詳細資訊，請參閱 [《Amazon S3與儲存貯體子資源操作和管理 Amazon S3 資源存取許可相關的許可》](#)。

使用選取請求查詢封存

您可以使用選取類型的請求，對封存的物件執行 SQL 查詢。選取請求查詢的封存物件必須格式化為未壓縮的逗號分隔值 (CSV) 檔案。您可以對封存的資料執行查詢和自訂分析，而不必將資料還原至較熱的 Amazon S3 層。如需選取請求的概觀，請參閱《Amazon Simple Storage Service 使用者指南》中的 [查詢封存物件](#)。

提出選取請求時，請執行下列動作：

- 定義選取查詢輸出的輸出位置。這必須與包含正在查詢之封存物件的儲存貯體 AWS 區域 位於相同的 Amazon S3 儲存貯體。AWS 帳戶 啟動任務的 必須具有寫入 S3 儲存貯體的許可。您可以為存放在儲存貯體中的輸出物件指定儲存類別和加密。如需輸出的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [查詢封存物件](#)。

如需請求內文中 S3 結構的詳細資訊，請參閱下列內容：

- [PutObject](#)
- 《Amazon Simple Storage Service 使用者指南》中的 [使用 ACLs 管理存取權](#)
- 《Amazon Simple Storage Service 使用者指南》中的 [使用伺服器端加密保護資料](#)
- 在請求內文 `SelectParameters` 的結構中為您的查詢定義還原 `SELECT` 類型的 SQL 表達式。您可以使用如下範例的表達式。
 - 下列表達式會傳回指定物件的所有記錄。

```
SELECT * FROM Object
```

- 假設您沒有針對存放在 物件中的資料使用任何標頭，您可以指定具有位置標頭的資料欄。

```
SELECT s._1, s._2 FROM Object s WHERE s._3 > 100
```

- 如果您有 標頭，而且您在請求內文的 CSV 結構 `fileHeaderInfo` 中將 設定為 `USE`，您可以在查詢中指定 標頭。(如果您將 `fileHeaderInfo` 欄位設定為 `IGNORE`，則跳過查詢的第一行。) 您不能將序號位置與標題列名稱混合。

```
SELECT s.Id, s.FirstName, s.SSN FROM S3Object s
```

如需搭配 Amazon Glacier Select 還原使用 SQL 的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [Amazon S3 Select 和 Amazon Glacier Select 的 SQL 參考](#)。

提出選取請求時，您也可以執行下列動作：

- 若要加速查詢，請指定 `Expedited` 層。如需層的詳細資訊，請參閱本主題稍後的「還原封存」。
- 指定正在查詢之輸入物件的資料序列化格式的詳細資訊，以及 CSV 編碼查詢結果的序列化。

以下是有關選取功能的其他重要事實：

- 輸出結果是新的 Amazon S3 物件。與封存擷取不同，它們會儲存到明確刪除為止，無論是手動或透過生命週期政策。
- 您可以在同一個 Amazon S3 物件上發出多個選取請求。Amazon S3 不會刪除重複的請求，因此請避免發出重複的請求。
- 即使物件已還原，Amazon S3 也會接受選取請求。選取請求不會傳回錯誤回應 409。

還原封存

GLACIER 和 DEEP_ARCHIVE 儲存類別中的物件會封存。若要存取封存的物件，您必須先啟動還原請求。這會還原封存物件的暫時副本。在還原請求中，您可以指定要還原複本存在的天數。在指定的期間之後，Amazon S3 會刪除暫時複本，但物件會保留在還原物件的 GLACIER 或 DEEP_ARCHIVE 儲存類別中封存。

若要還原特定物件版本，您可以提供版本 ID。如果您未提供版本 ID，Amazon S3 會還原目前的版本。

還原任務完成所需的時間取決於物件正在還原的儲存體方案，以及您指定的資料存取層。

還原封存的物件（或使用選取請求）時，您可以在請求內文的 `Tier` 元素中指定下列其中一個資料存取層選項：

- **Expedited** - 快速擷取可讓您在偶爾需要緊急請求封存子集時，快速存取存放在 GLACIER 儲存類別中的資料。對於除了最大封存物件 (250 MB+) 之外的所有封存物件，使用快速擷取存取的資料通常會在 1-5 分鐘內提供。佈建容量可確保快速擷取在需要時有可用的擷取容量。DEEP_ARCHIVE 儲存類別無法使用快速擷取和佈建容量。
- **Standard** - 標準擷取可讓您能在幾小時內存取任何已存檔的物件。這是未指定擷取選項的 GLACIER 和 DEEP_ARCHIVE 擷取請求的預設選項。標準擷取通常在 GLACIER 儲存類別的 3-5 小時內完成，通常在 DEEP_ARCHIVE 儲存類別的 12 小時內完成。
- **Bulk** - 大量擷取是 Amazon Glacier 成本最低的擷取選項，可讓您在一天中以廉價的方式擷取大量資料，甚至是 PB。大量擷取通常在 GLACIER 儲存類別的 5-12 小時內完成，通常在 DEEP_ARCHIVE 儲存類別的 48 小時內完成。

如需封存擷取選項和Expedited資料存取佈建容量的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[還原封存物件](#)。

您可以使用 Amazon S3 還原速度升級，在執行時將還原速度變更為更快的速度。您可以透過向相同物件發出另一個還原請求，設定新的Tier請求元素，來升級進行中還原的速度。發出升級還原方案的請求時，您必須選擇比進行中還原方案更快的方案。不可變更任何其他參數 (例如 Days 請求元素)。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[升級進行中還原的速度](#)。

若要取得物件還原的狀態，您可以傳送HEAD請求。操作會在回應中傳回 x-amz-restore標頭，提供還原狀態的相關資訊。您可以使用 Amazon S3 事件通知，在啟動或完成還原時通知您。如需詳細資訊，請參閱 [《Amazon Simple Storage Service 使用者指南》中的設定 Amazon S3 事件通知](#)。

還原封存的物件之後，您可以透過使用新期間重新發出請求來更新還原期間。Amazon S3 只會針對請求更新相對於目前時間和費用的還原期間，不會收取資料傳輸費用。當 Amazon S3 主動處理物件目前的還原請求時，您無法更新還原期間。

如果您的儲存貯體具有包含過期動作的規則的生命週期組態，物件過期會覆寫您在還原請求中指定的生命週期。例如，如果您還原物件複本 10 天，但物件排定在 3 天後過期，Amazon S3 會在 3 天內刪除物件。如需生命週期組態的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[PutBucketLifecycleConfiguration](#) 和 [物件生命週期管理](#)。

要求

語法

```
POST /ObjectName?restore&versionId=VersionID HTTP/1.1
Host: BucketName.s3.amazonaws.com
Date: date
```


Authorization: *authorization string* (see [####AWS#### 4 ##](#))

Content-MD5: *MD5*

request body

Note

語法會顯示一些請求標頭。如需完整清單，請參閱本主題稍後的「請求標頭」。

請求參數

此實作操作不使用請求參數。

請求標頭

名稱	描述	必要
Content-MD5	資料的 base64 編碼 128 位元 MD5 摘要。您必須使用此標頭做為訊息完整性檢查，以確認請求內文未在傳輸中損毀。如需詳細資訊，請參閱 RFC 1864。 類型：字串 預設：無	是

要求元素

以下是用於還原封存之請求內文的 XML 範例。

```
<RestoreRequest>
  <Days>2</Days>
  <GlacierJobParameters>
    <Tier>Bulk</Tier>
  </GlacierJobParameters>
</RestoreRequest>
```

下表說明在請求內文中封存還原的 XML。

名稱	描述	必要
RestoreRequest	還原資訊的容器。 類型：容器	是
Days	還原（作用中）複本的生命週期。您可以從 Amazon Glacier 還原物件的天數下限為 1。在物件複製達到指定的生命週期後，Amazon S3 會從儲存貯體中移除它。如果您要還原封存，則需要此元素。 請勿將此元素與 請求SELECT類型搭配使用。 類型：正整數 Ancestors：RestoreRequest	是，如果還原封存
GlacierJobParameters	S3 Glacier 儲存類別任務參數的容器。 請勿將此元素與 請求SELECT類型搭配使用。 類型：容器 Ancestors：RestoreRequest	否
Tier	還原封存時要使用的資料存取層。Standard是預設值。 類型：列舉 有效值：Expedited Standard Bulk Ancestors：GlacierJobParameters	否

下列 XML 是封存物件上選取查詢的請求內文：

```
<RestoreRequest>
  <Type>SELECT</Type>
  <Tier>Expedited</Tier>
```

```

<Description>Job description</Description>
<SelectParameters>
  <Expression>Select * from Object</Expression>
  <ExpressionType>SQL</ExpressionType>
  <InputSerialization>
    <CSV>
      <FileHeaderInfo>IGNORE</FileHeaderInfo>
      <RecordDelimiter>\n</RecordDelimiter>
      <FieldDelimiter>,</FieldDelimiter>
      <QuoteCharacter>"</QuoteCharacter>
      <QuoteEscapeCharacter>"</QuoteEscapeCharacter>
      <Comments>#</Comments>
    </CSV>
  </InputSerialization>
  <OutputSerialization>
    <CSV>
      <QuoteFields>ASNEEDED</QuoteFields>
      <RecordDelimiter>\n</RecordDelimiter>
      <FieldDelimiter>,</FieldDelimiter>
      <QuoteCharacter>"</QuoteCharacter>
      <QuoteEscapeCharacter>"</QuoteEscapeCharacter>
    </CSV>
  </OutputSerialization>
</SelectParameters>
<OutputLocation>
  <S3>
    <BucketName>Name of bucket</BucketName>
    <Prefix>Key prefix</Prefix>
    <CannedACL>Canned ACL string</CannedACL>
    <AccessControlList>
      <Grantee>
        <Type>Grantee Type</Type>
      </Grantee>
    </AccessControlList>
    <ID>Grantee identifier</ID>
    <URI>Grantee URI</URI>
    <Permission>Granted permission</Permission>
    <DisplayNmae>Display Name</DisplayName>
    <EmailAddress>email</EmailAddress>
  </S3>
</OutputLocation>
<Encryption>
  <EncryptionType>Encryption type</EncryptionType>
  <KMSKeyId>KMS Key ID</KMSKeyId>
  <KMSContext>Base64-encoded JSON<KMSContext>
</Encryption>

```

```
<UserMetadata>
  <MetadataEntry>
    <Name>Key</Name>
    <Value>Value</Value>
  </MetadataEntry>
</UserMetadata>
<Tagging>
  <TagSet>
    <Tag>
      <Key>Tag name</Key>
      <Value>Tag value</Value>
    </Tag>
  </TagSet>
</Tagging>
<StorageClass>Storage class</StorageClass>
</S3>
</OutputLocation>
</RestoreRequest>
```

下表說明請求內文中還原SELECT類型的 XML。

名稱	描述	必要
RestoreRequest	還原資訊的容器。 類型：容器	是
Tier	還原封存時要使用的資料存取層。 Standard是預設值。 類型：列舉 有效值：Expedited Standard Bulk Ancestors：RestoreRequest	否
Description	請求的選用描述。 類型：字串 Ancestors：RestoreRequest	否

名稱	描述	必要
SelectParameters	描述選取任務請求的參數。 類型：容器 Ancestors：RestoreRequest	是，如果請求類型為 SELECT
OutputLocation	描述接收選取還原請求結果的位置。 類型：Amazon S3 的容器 Ancestors：RestoreRequest	是，如果請求類型為 SELECT

SelectParameters 容器元素包含下列元素。

名稱	描述	必要
Expression	SQL 表達式。例如： - 下列 SQL 表達式會從以 CSV 格式存放的物件擷取資料的第一欄： <pre>SELECT s._1 FROM Object s</pre> - 下列 SQL 表達式會從 物件傳回所有內容： <pre>SELECT * FROM Object</pre> 類型：字串 Ancestors：SelectParameters	是
ExpressionType	識別表達式類型。 類型：字串	是

名稱	描述	必要
InputSerialization	有效值：SQL	是
	Ancestors：SelectParameters	
	描述 物件的序列化格式。	
	類型：CSV 的容器	
OutputSerialization	Ancestors：SelectParameters	是
	描述如何序列化選擇任務的結果。	
	類型：CSV 的容器	
	Ancestors：SelectParameters	

元素中的 CSV 容器**InputSerialization**元素包含下列元素。

名稱	描述	必要
RecordDelimiter	用於分隔輸入中個別記錄的單一字元。您可以指定任意分隔符號，而不是預設值。 類型：字串 預設：\n Ancestors：CSV	否
FieldDelimiter	用於分隔記錄中個別欄位的單一字元。您可以指定任意分隔符號。 類型：字串 預設：,	否

名稱	描述	必要
	Ancestors : CSV	
QuoteCharacter	當欄位分隔符號是 值的一部分時，用於逸出的單一字元。 在 CSV 檔案中考慮此範例： "a, b" 以引號包裝值會使此值成為單一欄位。如果您不使用引號，則逗號是欄位分隔符號（讓它有兩個不同的欄位值，a 和 b）。 類型：字串 預設：" Ancestors : CSV	否
QuoteEscapeCharacter	用於在已逸出值內逸出引號字元的單一字元。例如，值會""a , b ""剖析為 " a , b "。 類型：字串 預設：" Ancestors : CSV	否

名稱	描述	必要
FileHeaderInfo	描述輸入資料的第一行。這是其中一個 ENUM 值。 - NONE：第一行不是標頭。 - IGNORE：第一行是標頭，但您無法使用標頭值來表示表達式中的資料欄。您可以使用資料欄位置（例如 _1、_2、...）來表示資料欄 ()SELECT s._1 FROM OBJECT s。 - Use：第一行是標頭，您可以使用標頭值來識別表達式 () 中的資料欄SELECT "name" FROM OBJECT 。 類型：列舉 有效值：NONE USE IGNORE Ancestors：CSV	否
Comments	單一字元用於表示當該字元出現在該資料列的開頭時應應該忽略該資料列。您可以指定任何字元來表示註解行。 類型：字串 Ancestors：CSV	否

CSV 容器元素（在 **OutputSerialization** 元素中）包含下列元素。

名稱	描述	必要
QuoteFields	指出是否在輸出欄位周圍使用引號。 - ALWAYS：輸出欄位一律使用引號。 - ASNEEDED：視需要使用輸出欄位的引號。	否

名稱	描述	必要
	類型：列舉 有效值：ALWAYS ASNEEDED 預設：AsNeeded Ancestors：CSV	
RecordDelimiter	用於分隔輸出中個別記錄的單一字元。您可以指定任意分隔符號，而不是預設值。 類型：字串 預設：\n Ancestors：CSV	否
FieldDelimiter	用於分隔記錄中個別欄位的單一字元。您可以指定任意分隔符號。 類型：字串 預設：, Ancestors：CSV	否
QuoteCharacter	當欄位分隔符號是值的一部分時，用於逸出的單一字元。例如，如果值為 a, b, Amazon S3 會以引號包裝此欄位值，如下所示：" a , b "。 類型：字串 預設：" Ancestors：CSV	否

名稱	描述	必要
QuoteEscapeCharacter	用於在已逸出值內逸出引號字元的單一字元。例如，如果值為 " a , b "，Amazon S3 會以引號包裝值，如下所示：" " a , b " "。 類型：字串 Ancestors：CSV	否

S3 容器元素（在 **OutputLocation** 元素中）包含下列元素。

名稱	描述	必要
AccessControlList	控制暫存結果存取的授予清單。 類型：授予的容器 Ancestors：S3	否
BucketName	儲存選取還原結果的 S3 儲存貯體名稱。儲存貯體必須與包含輸入封存物件的儲存貯體位於相同的 AWS 區域。 類型：字串 Ancestors：S3	是
CannedACL	要套用至選取還原結果的固定存取控制清單 (ACL)。 類型：字串 有效值： private public-read public-read-write aws-exec-read authenticated-read bucket-owner-read bucket-owner-full-control	否

名稱	描述	必要
	Ancestors : S3	
Encryption	包含儲存結果的加密資訊。 類型 : Container for Encryption Ancestors : S3	否
Prefix	選取還原結果前綴的字首。字首的最大長度為 512 位元組。 類型 : 字串 Ancestors : S3	是
StorageClass	用於存放選取請求結果的儲存體類別。 類型 : 字串 有效值 : STANDARD REDUCED_REDUNDANCY STANDARD_IA ONEZONE_IA Ancestors : S3	否
Tagging	標籤資訊的容器。 類型 : 標籤結構 Ancestors : S3	否
UserMetadata	包含要與選取還原結果一起存放的中繼資料清單。 類型 : MetadataEntry 結構 Ancestors : S3	否

承授者容器元素（在 **AccessControlList** 元素中）包含下列元素。

名稱	描述	必要
DisplayName	承授者的螢幕名稱。 類型：字串 Ancestors：Grantee	否
EmailAddress	承授者的電子郵件地址。 類型：字串 Ancestors：Grantee	否
ID	承授者的正式使用者 ID。 類型：字串 Ancestors：Grantee	否
Type	承授者的類型。 類型：字串 Ancestors：Grantee	否
URI	承授者群組的 URI。 類型：字串 Ancestors：Grantee	否
Permission	授予的許可。 類型：字串	否

名稱	描述	必要
	Ancestors : Grantee	

Encryption 容器元素（在 S3 中）包含下列元素。

名稱	描述	必要
EncryptionType	儲存任務結果時使用的伺服器端加密演算法。預設值是不加密。 類型：字串 有效值 <code>aws:kms</code> <code>AES256</code> Ancestors : Encryption	否
KMSContext	選用。如果加密類型為 <code>aws:kms</code>，您可以使用此值來指定選取還原結果的加密內容。 類型：字串 Ancestors : 加密	否
KMSKeyId	用於物件加密的 AWS Key Management Service (AWS KMS) 金鑰 ID。 類型：字串 Ancestors : Encryption	否

TagSet 容器元素（在 **Tagging** 元素中）包含下列元素。

名稱	描述	必要
Tag	包含標籤。	否

名稱	描述	必要
	類型：容器 Ancestors：TagSet	

標籤容器元素（在 **TagSet** 元素中）包含下列元素。

名稱	描述	必要
Key	標籤的名稱。 類型：字串 Ancestors：Tag	否
Value	標籤的值。 類型：字串 Ancestors：Tag	否

MetadataEntry 容器元素（在 **UserMetadata** 元素中）包含要與物件一起存放的下列鍵值對元素。

名稱	描述	必要
MetadataKey	中繼資料索引鍵。 類型：字串 Ancestors：	否
MetadataEntry	中繼資料值。 類型：字串	否

名稱	描述	必要
	Ancestors :	

回應

成功的操作會傳回 200 OK或 202 Accepted 狀態碼。

- 如果之前未還原物件複本，則 Amazon S3 會在回應202 Accepted中傳回。
- 如果物件複本先前已還原，Amazon S3 會在回應200 OK中傳回。

回應標頭

此操作的實作只會使用大多數回應通用的回應標頭。如需詳細資訊，請參閱[常見回應標頭](#)。

回應元素

此操作不會傳回回應元素。

特殊錯誤

錯誤程式碼	說明	HTTP 狀態碼	SOAP 故障碼字首
RestoreAlreadyInProgress	物件還原已在進行中。(此錯誤不適用於 SELECT 類型請求。)	409 衝突	用戶端
GlacierExpeditedRetrievalNotAvailable	S3 Glacier 儲存類別快速擷取目前不可用。請稍後再試。(如果沒有足夠的容量來處理Expedited 請求，則傳回。此錯誤僅適用於Expedited 擷取，不適用於Standard或 Bulk擷取。)	503	N/A

範例

使用快速擷取選項還原物件兩天

下列還原請求會使用快速擷取選項，從 Amazon Glacier 還原photo1.jpg物件的副本兩天。

```
POST /photo1.jpg?restore HTTP/1.1
Host: examplebucket.s3.amazonaws.com
Date: Mon, 22 Oct 2012 01:49:52 GMT
Authorization: authorization string
Content-Length: content length
```

```
<RestoreRequest>
  <Days>2</Days>
  <GlacierJobParameters>
    <Tier>Expedited</Tier>
  </GlacierJobParameters>
</RestoreRequest>
```

如果 examplebucket 沒有物件的還原複本，Amazon S3 會傳回下列202 Accepted回應。

```
HTTP/1.1 202 Accepted
x-amz-id-2: GFihv3y6+kE7KG11GEkQhU7/2/cHR3Yb2fCb2S04nxI423Dqwg2XiQ0B/
UZ1zYQvPiBlZNRcovw=
x-amz-request-id: 9F341CD3C4BA79E0
Date: Sat, 20 Oct 2012 23:54:05 GMT
Content-Length: 0
Server: AmazonS3
```

如果物件的複本已還原，Amazon S3 會傳回200 OK回應，並僅更新還原複本的到期時間。

使用 SELECT 請求查詢封存

以下是範例選取還原請求。

```
POST /object-one.csv?restore HTTP/1.1
Host: examplebucket.s3.amazonaws.com
Date: Sat, 20 Oct 2012 23:54:05 GMT
Authorization: authorization string
Content-Length: content length
```

```
<RestoreRequest xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
```



```

<Type>SELECT</Type>
<Tier>Expedited</Tier>
<Description>this is a description</Description>
<SelectParameters>
  <InputSerialization>
    <CSV>
      <FileHeaderInfo>IGNORE</FileHeaderInfo>
      <Comments>#</Comments>
      <QuoteEscapeCharacter>"</QuoteEscapeCharacter>
      <RecordDelimiter>\n</RecordDelimiter>
      <FieldDelimiter>,</FieldDelimiter>
      <QuoteCharacter>"</QuoteCharacter>
    </CSV>
  </InputSerialization>
  <ExpressionType>SQL</ExpressionType>
  <Expression>select * from object</Expression>
  <OutputSerialization>
    <CSV>
      <QuoteFields>ALWAYS</QuoteFields>
      <QuoteEscapeCharacter>"</QuoteEscapeCharacter>
      <RecordDelimiter>\n</RecordDelimiter>
      <FieldDelimiter>\t</FieldDelimiter>
      <QuoteCharacter>\'</QuoteCharacter>
    </CSV>
  </OutputSerialization>
</SelectParameters>
<OutputLocation>
  <S3>
    <BucketName>example-output-bucket</BucketName>
    <Prefix>test-s3</Prefix>
    <AccessControlList>
      <Grant>
        <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="AmazonCustomerByEmail">
          <EmailAddress>jane-doe@example.com</EmailAddress>
        </Grantee>
        <Permission>FULL_CONTROL</Permission>
      </Grant>
    </AccessControlList>
    <UserMetadata>
      <MetadataEntry>
        <Name>test</Name>
        <Value>test-value</Value>
      </MetadataEntry>

```

```
<MetadataEntry>
  <Name>other</Name>
  <Value>something else</Value>
</MetadataEntry>
</UserMetadata>
<StorageClass>STANDARD</StorageClass>
</S3>
</OutputLocation>
</RestoreRequest>
```

Amazon S3 傳回下列 202 Accepted 回應。

```
HTTP/1.1 202 Accepted
x-amz-id-2: GFihv3y6+kE7KG11GEkQhU7/2/cHR3Yb2fCb2S04nxI423Dqwg2XiQ0B/
UZ1zYQvPiBlZNRcovw=
x-amz-request-id: 9F341CD3C4BA79E0
x-amz-restore-output-path: js-test-s3/qE8nk5M0XIj-LuZE2HXNw6empQm3znLkHlMWInRYPS-
0rl2W0uj6LyYm-neTvm1-btz3wbBxfMhPykd3jk1-1vZE7w42/
Date: Sat, 20 Oct 2012 23:54:05 GMT
Content-Length: 0
Server: AmazonS3
```

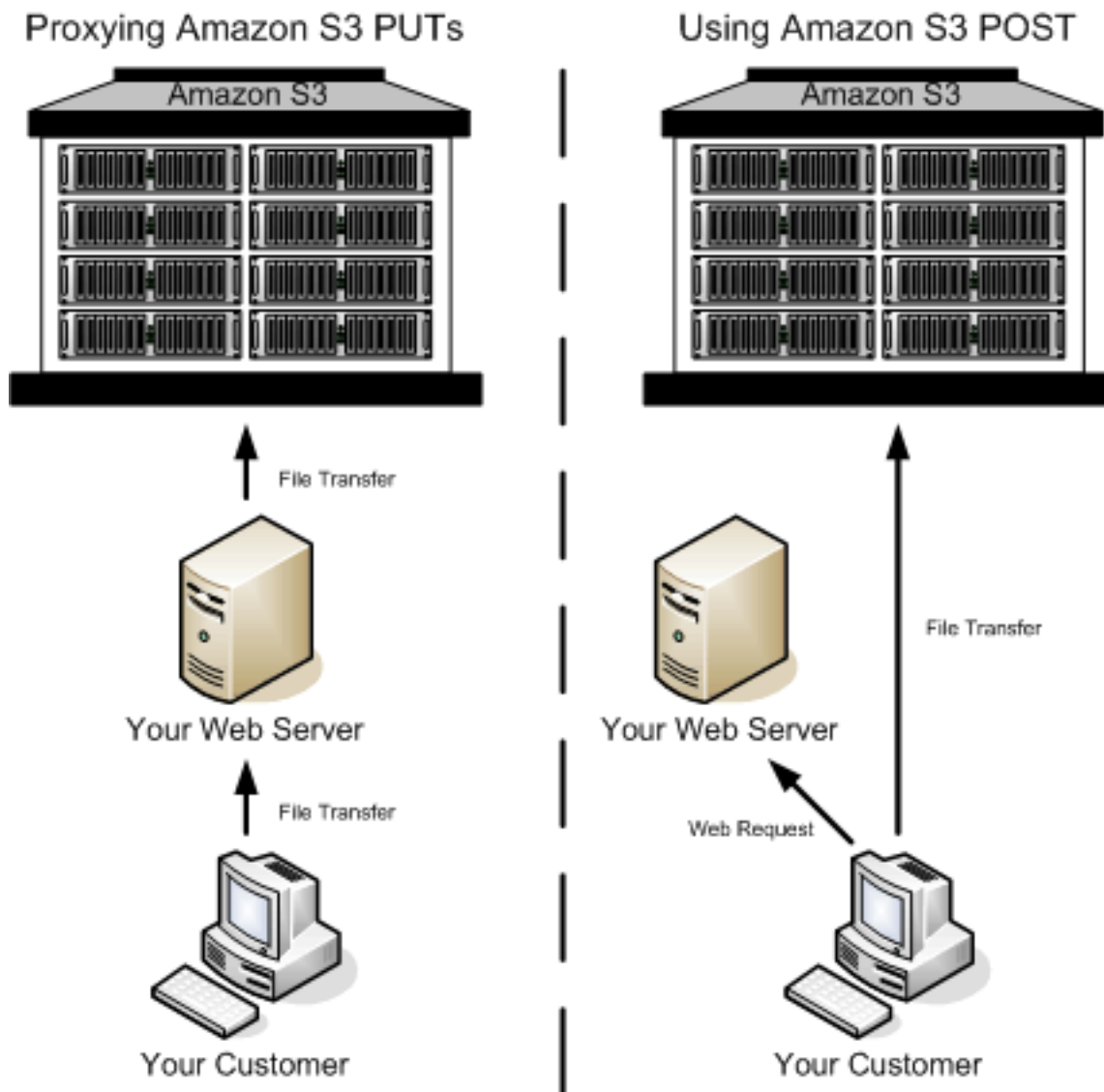
詳細資訊

- [GetBucketLifecycleConfiguration](#)
- [PutBucketLifecycleConfiguration](#)
- [《Amazon Simple Storage Service 使用者指南》中的 Amazon S3 Select 和 Amazon Glacier Select 的 SQL 參考](#)

使用 HTTP POST 的瀏覽器型上傳

Amazon S3 支援 HTTP POST 請求，讓使用者可以直接將內容上傳至 Amazon S3。透過使用 POST，最終使用者可以驗證請求，而不必透過保護憑證的安全中介節點來傳遞資料。因此，HTTP POST 有可能減少延遲。

下圖顯示使用 POST 請求的 Amazon S3 上傳。



1. 使用者會從 Web 瀏覽器存取您的頁面。
2. 您的網頁包含 HTML 表單，其中包含使用者將內容上傳至 Amazon S3 所需的所有必要資訊。
3. 使用者透過 Web 瀏覽器將內容上傳至 Amazon S3。

傳送瀏覽器型 POST 請求的程序如下：

1. 建立安全政策，指定限制您要在請求中允許的條件，例如可上傳物件的儲存貯體名稱，以及您要允許建立物件的金鑰名稱字首。
2. 根據政策建立簽章。對於已驗證的請求，表單必須包含有效的簽章和政策。
3. 建立使用者可存取的 HTML 表單，以便將物件上傳至 Amazon S3 儲存貯體。

下一節說明如何建立簽章來驗證請求。如需建立表單和安全性政策的相關資訊，請參閱 [建立 HTML 表單（使用 AWS Signature 第 4 版）](#)。

計算簽章

對於已驗證的請求，HTML 表單必須包含安全政策和簽章的欄位。

- 安全政策（請參閱 [POST 政策](#)）控制請求中允許的項目。
- 安全政策是簽章計算中的 StringToSign（請參閱 [簽署請求簡介](#)）。



計算簽章

1. 使用 UTF-8 編碼建立政策。
2. 將 UTF-8-encoded 的政策位元組轉換為 base64。結果是 StringToSign。
3. 建立簽署金鑰。
4. 使用簽署金鑰來 StringToSign 使用 HMAC-SHA256 簽署演算法簽署。

如需建立 HTML 表單、安全政策和範例的詳細資訊，請參閱下列內容：

- [建立 HTML 表單（使用 AWS Signature 第 4 版）](#)
- [POST 政策](#)
- [範例：使用 HTTP POST 以瀏覽器為基礎的上傳（使用 AWS Signature 第 4 版）](#)

建立 HTML 表單（使用 AWS Signature 第 4 版）

主題

- [HTML 表單宣告](#)
- [HTML 表單欄位](#)

若要允許使用者使用瀏覽器 (HTTP POST 請求) 將內容上傳至 Amazon S3，您可以使用 HTML 表單。HTML 表單是由一個表單宣告與多個表單欄位所組成。表單宣告包含要求的高階資訊。表單欄位包含詳細的請求資訊。

本節說明如何建立 HTML 表單。如需使用 HTTP POST 的瀏覽器型上傳工作範例，以及請求身分驗證的相關簽章計算，請參閱 [範例：使用 HTTP POST 以瀏覽器為基礎的上傳（使用 AWS Signature 第 4 版）](#)。

表單與政策必須經過 UTF-8 編碼。您可以在 content 屬性 charset=UTF-8 中指定，將 UTF-8 編碼套用至表單。以下是 HTML 標題中 UTF-8 編碼的範例。

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
```

以下是請求標頭中 UTF-8 編碼的範例。

```
Content-Type: text/html; charset=UTF-8
```

Note

表單資料和邊界（檔案內容除外）不得超過 20KB。

HTML 表單宣告

HTML 表單宣告具有下列三個屬性：

- **action** – 處理請求的 URL，必須設定為儲存貯體的 URL。例如，如果您的儲存貯體名稱為 `examplebucket`，則 URL 為 `http://examplebucket.s3.amazonaws.com/`。

Note

金鑰名稱是在表單欄位中指定。

- **method** – 方法必須為 POST。
- **enctype** – 檔案上傳和文字區域上傳的附件類型 (enctype) 必須設定為分段/表單資料。如需的詳細資訊 enctype，請參閱 [RFC 1867](#)。

這是儲存貯體的表單宣告 `examplebucket`。

```
<form action="http://examplebucket.s3.amazonaws.com/" method="post"
enctype="multipart/form-data">
```

HTML 表單欄位

下表說明您可以在表單中使用的欄位清單。在其他欄位中，有一個簽章欄位，您可以用來驗證請求。有些欄位可讓您指定簽章計算演算法 (`x-amz-algorithm`)、用於產生簽署金鑰的登入資料範圍 (`x-amz-credential`)，以及用於計算簽章的日期 (`x-amz-date`)。Amazon S3 會使用此資訊來重新建立簽章。如果簽章相符，Amazon S3 會處理請求。

Note

系統會自動以使用者提供的檔案名稱來取代變數 `${filename}`，所有表單欄位都可識別此變數。如果瀏覽器或用戶端提供檔案的完整或部分路徑，則只會使用最後一個斜線 (/) 或

反斜線 (\) 後面的文字 (例如 , C:\Program Files\directory1\file.txt 解譯為 file.txt)。若未提供檔案或檔案名稱, 則會以空字串取代變數。

如果您未提供驗證請求所需的元素, 例如 policy 元素, 則請求會假設為匿名, 而且只有在您已設定儲存貯體進行公有讀取和寫入時, 才會成功。

元素名稱	說明	必要
acl	Amazon S3 存取控制清單 (ACL)。如果指定無效的 ACL, Amazon S3 會拒絕請求。如需 ACLs 的詳細資訊, 請參閱使用 Amazon S3 ACLs。 類型: 字串 預設值: private 有效值: private public-read public-read-write aws-exec-read authenticated-read bucket-owner-read bucket-owner-full-control	否
Cache-Control	REST 特定標頭。如需詳細資訊, 請參閱 PutObject 。	否
Content-Type		
Content-Disposition		
Content-Encoding		
Expires		
key	上傳物件的金鑰名稱。 若要使用使用者提供的檔案名稱, 請使用 <code>\${filename}</code> 變數。例如, 如果您上傳檔案, photo1.jpg 並將 指定 /user/user1/	是

元素名稱	說明	必要
	<code>\${filename}</code> 為金鑰名稱，則檔案會儲存為 <code>/user/user1/photo1.jpg</code> 。 如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的物件金鑰和中繼資料。	
<code>policy</code>	描述請求中允許內容的 base64 編碼安全政策。對於已驗證的請求，需要政策。 沒有安全政策的請求會被視為匿名，而且只會在可公開寫入的儲存貯體上成功。	已驗證請求的必要項目
<code>success_action_redirect</code>	用戶端成功上傳時被重新導向的目標 URL。 如果 <code>success_action_redirect</code> 未指定，或 Amazon S3 無法解譯 URL，Amazon S3 會傳回 <code>success_action_status</code> 欄位中指定的空白文件類型。 如果上傳失敗，Amazon S3 會傳回錯誤，而不會將使用者重新導向至另一個 URL。	否

元素名稱	說明	必要
success_action_status	如果success_action_redirect 未指定，則在成功上傳時傳回用戶端的狀態碼。 有效值為 200、201或 204 (預設值)。 如果值設定為 200 或 204，Amazon S3 會傳回具有指定狀態碼的空白文件。 如果值設定為 201，Amazon S3 會傳回狀態碼為 201 的 XML 文件。如需 XML 文件內容的詳細資訊，請參閱 POST 物件。 如果值未設定或無效，Amazon S3 會傳回狀態碼為 204 的空白文件。  Note Adobe Flash Player 的某些版本無法正確處理具有空白主體的 HTTP 回應。若要支援經由 Adobe Flash 上傳，建議將 success_action_status 設定為 201。	否
x-amz-algorithm	用來驗證請求的簽署演算法。對於 AWS Signature 第 4 版，值為 AWS4-HMAC-SHA256。 如果要求包含政策文件，則需要此欄位。	已驗證請求的必要項目

元素名稱	說明	必要
x-amz-credential	除了您的存取金鑰 ID 之外，此欄位還提供範圍資訊，識別簽章有效的區域和服務。這應該與您用於計算簽署金鑰以進行簽章計算的範圍相同。 這是以下格式的字串： <pre><your-access-key-id> /<date>/<aws-region> /<aws-service> /aws4_request</pre> 例如： <pre>AKIAIOSFODNN7EXAMPLE/20130728/us-east-1/s3/aws4_request</pre> 對於 Amazon S3，aws-service 字串為 s3。如需 Amazon S3 aws-region 字串的清單，請參閱《》中的區域和端點AWS 一般參考。如果請求中包含政策文件，則此為必要項目。	已驗證請求的必要項目
x-amz-date	這是 ISO8601 格式的日期值。例如 20130728T000000Z 。 這是您在建立簽署金鑰時所使用的相同日期（例如 20130728）。這也必須與您簽署的政策（x-amz-date）中提供的值相同。 如果請求中包含政策文件，則此為必要項目。	已驗證請求的必要項目

元素名稱	說明	必要
x-amz-security-token	Amazon DevPay 和工作階段登入資料使用的安全字符 如果請求使用 Amazon DevPay，則需要兩個 x-amz-security-token 表單欄位：一個用於產品字符，另一個用於使用者字符。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的使用 DevPay。 如果請求使用工作階段登入資料，則需要一個 x-amz-security-token 表單。如需詳細資訊，請參閱《IAM 使用者指南》中的請求臨時安全憑證。	否
x-amz-signature	(AWS 簽章版本 4) 安全政策的 HMAC-SHA256 雜湊。 如果要求包含政策文件，則需要此欄位。	已驗證請求的必要項目
x-amz-meta-*	以此字首開頭的欄位名稱是使用者定義的中繼資料。每個項目都會儲存並傳回為一組鍵/值對。Amazon S3 不會驗證或解譯使用者定義的中繼資料。如需詳細資訊，請參閱 PutObject。	否
x-amz-*	如需其他 x-amz-* 標頭，請參閱 POST 物件 (POST 物件)。	否
file	檔案或文字內容。 檔案或內容必須是表單中的最後一個欄位。 您無法一次上傳多個檔案。	是

驗證請求需要條件式項目，匿名請求則為選用項目。

現在您已了解如何建立表單，接下來您可以建立您可以簽署的安全政策。如需詳細資訊，請參閱[POST 政策](#)。

POST 政策

主題

- [Expiration](#)
- [條件比對](#)
- [條件](#)
- [字元逸出](#)

使用 HTTP POST 提出已驗證請求所需的政策是以 JavaScript 物件標記法 (JSON) 撰寫的 UTF-8 和 base64 編碼文件，指定請求必須符合的條件。根據您設計政策文件的方式，您可以控制每個上傳、每個使用者、所有上傳的存取精細程度，或根據符合您需求的其他設計。

本節說明 POST 政策。如需使用 POST 政策的簽章計算範例，請參閱 [範例：使用 HTTP POST 以瀏覽器為基礎的上傳（使用 AWS Signature 第 4 版）](#)。

Note

雖然政策文件是選用的，但強烈建議您使用政策文件來控制請求中允許的項目。如果您讓儲存貯體可公開寫入，則完全無法控制哪些使用者可以寫入您的儲存貯體。

以下是 POST 政策文件的範例。

```
{ "expiration": "2007-12-01T12:00:00.000Z",
  "conditions": [
    {"acl": "public-read" },
    {"bucket": "johnsmith" },
    ["starts-with", "$key", "user/eric/"],
  ]
}
```

POST 政策一律包含 `expiration` 和 `conditions` 元素。範例政策使用兩種條件比對類型（完全比對和開頭比對）。下列各節說明這些元素。

Expiration

`expiration` 元素會以 ISO8601 GMT 日期格式指定 POST 政策的過期日期和時間。例如，`2013-08-01T12:00:00.000Z` 指定 POST 政策在 2013 年 8 月 1 日 GMT 午夜之後無效。

條件比對

以下是描述條件比對類型的表格，您可以用來指定 POST 政策條件（下一節說明）。雖然您必須為表單中指定的每個表單欄位指定至少一個條件，但您可以透過為表單欄位指定多個條件來建立更複雜的相符條件。

條件比對類型	說明
完全符合	表單欄位值必須符合指定的值。下列範例表示 ACL 必須設定為 <code>public-read</code>： <pre>{"acl": "public-read" }</pre> 您也可以使用下列範例來表示 ACL 必須設定為 <code>public-read</code>： <pre>["eq", "\$acl", "public-read"]</pre>
開頭為	值必須以指定的值開頭。此範例指出物件金鑰必須以 <code>user/user1</code> 開頭： <pre>["starts-with", "\$key", "user/user1/"]</pre>
符合逗號分隔清單中的內容類型	包含逗號之 <code>starts-with</code> 條件的 <code>Content-Types</code> 值會解譯為清單。清單中的每個值必須符合要傳遞整個條件的條件。例如，提供下列條件： <pre>["starts-with", "\$Content-Type", "image/"]</pre> 下列值會傳遞 條件： <pre>"image/jpg,image/png,image/gif"</pre> 下列值不會傳遞 條件：

條件比對類型	說明
	<pre>["image/jpg,text/plain"]</pre>  Note 以外的資料元素Content-Type 會視為字串，無論逗號是否存在。
符合任何內容	若要設定 POST 政策以允許表單欄位中的任何內容，請使用 starts-with 搭配空值 ("")。此範例允許 的任何值success_action_redirect： <pre>["starts-with", "\$success_action_redirect", ""]</pre>
指定範圍	對於接受範圍的表單欄位，請以逗號分隔上限和下限。此範例允許 1 到 10 MiB 的檔案大小： <pre>["content-length-range", 1048576, 10485760]</pre>

POST 政策支援的特定條件如 中所述[條件](#)。

條件

POST 政策conditions中的 是 物件陣列，每個都用來驗證請求。您可以使用這些條件來限制請求中允許的內容。例如，上述政策條件需要下列項目：

- 請求必須指定儲存johnsmith貯體名稱。
- 物件金鑰名稱必須有 user/eric字首。
- 物件 ACL 必須設定為 public-read。

您以形式指定的每個表單欄位（除了 x-amz-signature、policy、file和具有字x-ignore-首的欄位名稱）都必須出現在條件清單中。

Note

驗證 POST 政策之前，會展開表單中的所有變數。因此，所有條件比對都應該與展開的表單欄位相符。假設您想要將物件金鑰名稱限制為特定字首 (user/user1)。在此情況下，您將金鑰表單欄位設定為 user/user1/\${filename}。您的 POST 政策應該是 ["starts-with", "\$key", "user/user1/"] (請勿輸入 ["starts-with", "\$key", "user/user1/\${filename}"])。如需詳細資訊，請參閱[條件比對](#)。

下表說明政策文件條件。

元素名稱	說明
acl	指定必須在表單提交中使用的 ACL 值。 此條件支援精確比對，以及下節中討論starts-with 的條件比對類型。
bucket	指定可接受的儲存貯體名稱。 此條件支援完全相符的條件比對類型。
content-length-range	上傳內容的允許大小下限和上限。 此條件支援content-length-range 條件比對類型。
Cache-Control	REST 特定標頭。如需詳細資訊，請參閱POST 物件。 此條件支援完全相符和starts-with 條件相符類型。
Content-Type	
Content-Disposition	
Content-Encoding	
Expires	
key	可接受的金鑰名稱或已上傳物件的字首。

元素名稱	說明
	此條件支援完全相符和starts-with 條件相符類型。
success_action_redirect redirect	用戶端成功上傳時被重新導向的目標 URL。 此條件支援完全相符和starts-with 條件相符類型。
success_action_status	如果success_action_redirect 未指定，則在成功上傳時傳回用戶端的狀態碼。 此條件支援完全相符。
x-amz-algorithm	在簽章計算期間必須使用的簽署演算法。對於 AWS Signature 第 4 版，值為 AWS4-HMAC-SHA256 。 此條件支援完全相符。
x-amz-credential	您用來計算簽章的登入資料。它提供存取金鑰 ID 和範圍資訊，識別簽章有效的區域和服務。這應該與您用於計算簽署金鑰以進行簽章計算的範圍相同。 這是以下格式的字串： <i><your-access-key-id> /<date>/<aws-region> /<aws-service> /aws4_request</i> 例如： AKIAIOSFODNN7EXAMPLE/20130728/us-east-1/s3/aws4_request 對於 Amazon S3，aws-service 字串為 s3。如需 Amazon S3 aws-region 字串的清單，請參閱 中的 區域和端點 AWS 一般參考。如果請求中包含 POST 政策文件，則此為必要項目。 此條件支援完全相符。

元素名稱	說明
x-amz-date	ISO8601 格式字串中指定的日期值。例如 20130728T000000Z 。日期必須與您在建立簽章計算的簽署金鑰時所使用的日期相同。 如果請求中包含 POST 政策文件，則此為必要項目。 此條件支援完全相符。
x-amz-security-token	Amazon DevPay 安全字符。 每個使用 Amazon DevPay 的請求都需要兩個 x-amz-security-token 表單欄位：一個表示產品字符，另一個表示使用者字符。因此，必須以逗號分隔其值。例如，如果使用者字符是 eW91dHViZQ== 而產品字符是 b0hnNVNKWVJIQTA= ，您可以將 POST 政策項目設定為：{ "x-amz-security-token": "eW91dHViZQ==,b0hnNVNKWVJIQTA=" } 。 如需 Amazon DevPay 的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的使用 DevPay。
x-amz-meta-*	使用者指定的中繼資料。 此條件支援完全相符和starts-with 條件相符類型。
x-amz-*	如需其他x-amz-*標頭，請參閱 POST 物件 (POST 物件)。 此條件支援完全相符。

 Note

如果您的工具組新增更多表單欄位（例如 Flash 新增 filename），您必須將它們新增至 POST 政策文件。如果您可以控制此功能，請在欄位前面加上 x-ignore-，讓 Amazon S3 忽略此功能，這不會影響此功能的未來版本。

字元逸出

下表說明必須在 POST 政策文件中逸出的字元。

逸出序列	描述
\\	反斜線
\\\$	美元符號
\\b	退格鍵
\\f	Form Feed
\\n	新行
\\r	歸位字元
\\t	水平標籤
\\v	垂直標籤
\\uXXXX	所有 Unicode 字元

現在您已熟悉表單和政策，並了解簽署的運作方式，您可以嘗試 POST 上傳範例。您需要撰寫程式碼來計算簽章。此範例提供範列表單，以及可用來測試簽章計算的 POST 政策。如需詳細資訊，請參閱[範例：使用 HTTP POST 以瀏覽器為基礎的上傳（使用 AWS Signature 第 4 版）](#)。

範例：使用 HTTP POST 以瀏覽器為基礎的上傳（使用 AWS Signature 第 4 版）

本節顯示使用 HTTP POST 請求將內容直接上傳至 Amazon S3 的範例。

如需 Signature 第 4 版的詳細資訊，請參閱 [Signature 第 4 版簽署程序](#)。

使用 HTTP POST 將檔案上傳至 Amazon S3

此範例提供範例 POST 政策，以及可用來上傳檔案的表單。主題使用範例政策和虛構登入資料，向您顯示工作流程和產生的簽章和政策雜湊。您可以使用此資料做為測試套件來驗證您的簽章計算程式碼。

此範例使用下列範例登入簽章計算。您可以使用這些登入資料來驗證簽章計算程式碼。不過，在傳送請求到時，您必須以自己的登入資料取代這些登入資料 AWS。

參數	Value
AWSAccessKeyId	AKIAIOSFODNN7EXAMPLE
AWSSecretAccessKey	wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY

範例政策和表單

下列 POST 政策支援在特定條件下上傳至 Amazon S3。

```
{ "expiration": "2015-12-30T12:00:00.000Z",
  "conditions": [
    {"bucket": "sigv4examplebucket"},
    ["starts-with", "$key", "user/user1/"],
    {"acl": "public-read"},
    {"success_action_redirect": "http://sigv4examplebucket.s3.amazonaws.com/successful_upload.html"},
    ["starts-with", "$Content-Type", "image/"],
    {"x-amz-meta-uuid": "14365123651274"},
    {"x-amz-server-side-encryption": "AES256"},
    ["starts-with", "$x-amz-meta-tag", ""],

    {"x-amz-credential": "AKIAIOSFODNN7EXAMPLE/20151229/us-east-1/s3/aws4_request"},
    {"x-amz-algorithm": "AWS4-HMAC-SHA256"},
```

```
{ "x-amz-date": "20151229T000000Z" }  
]  
}
```

此 POST 政策會在請求上設定下列條件：

- 上傳必須在 2015 年 12 月 30 日中午 UTC 之前進行。
- 內容只能上傳至 sigv4examplebucket。儲存貯體必須位於您在登入資料範圍 (x-amz-credential 表單參數) 中指定的區域，因為您提供的簽章僅在此範圍內有效。
- 您可以提供開頭為的任何金鑰名稱user/user1。例如 user/user1/MyPhoto.jpg。
- ACL 必須設定為 public-read。
- 如果上傳成功，使用者的瀏覽器會重新導向至 `http://sigv4examplebucket.s3.amazonaws.com/successful_upload.html`。
- 物件必須是映像檔案。
- x-amz-meta-uuid 標籤必須設定為 14365123651274。
- x-amz-meta-tag 可以包含任何值。

以下是此 POST 政策的 Base64-encoded 版本。您可以在簽章計算中使用此值做為 StringToSign。

```
eyJhZiZxhwaXJhdGlvbiI6IClyMDE1LTUyLTUwVDEyOjAwOjAwLjAwMFoiLA0KICAiY29uZGl0aW9ucyI6IFsNCiAgICB7ImJ
```

當您複製/貼上上述政策時，它應該具有歸位符號，以及運算雜湊的新行，以符合此值（即 ASCII 文字，具有 CRLF 行終端器）。

使用範例登入資料來建立簽章，簽章值如下所示（在簽章計算中，日期與政策 (20151229 x-amz-date 中的 相同)：

```
8afdbf4008c03f22c2cd3cdb72e4afbb1f6a588f3255ac628749a66d7f09699e
```

下列範例表單會指定上述 POST 政策，並支援對提出 POST 請求 sigv4examplebucket。在文字編輯器中複製/貼上內容，並將其儲存為 exampleform.html。然後，您可以使用 exampleform.html 將映像檔案上傳至特定儲存貯體。如果您提供的簽章符合 Amazon S3 計算的簽章，您的請求將會成功。

Note

您必須使用有效值更新儲存貯體名稱、日期、登入資料、政策和簽章，才能成功上傳至 S3。

```

<html>
  <head>

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

  </head>
  <body>

    <form action="http://sigv4examplebucket.s3.amazonaws.com/" method="post"
    enctype="multipart/form-data">
      Key to upload:
      <input type="input" name="key" value="user/user1/${filename}" /><br />
      <input type="hidden" name="acl" value="public-read" />
      <input type="hidden" name="success_action_redirect" value="http://
sigv4examplebucket.s3.amazonaws.com/successful_upload.html" />
      Content-Type:
      <input type="input" name="Content-Type" value="image/jpeg" /><br />
      <input type="hidden" name="x-amz-meta-uuid" value="14365123651274" />
      <input type="hidden" name="x-amz-server-side-encryption" value="AES256" />
      <input type="text" name="X-Amz-Credential" value="AKIAIOSFODNN7EXAMPLE/20151229/
us-east-1/s3/aws4_request" />
      <input type="text" name="X-Amz-Algorithm" value="AWS4-HMAC-SHA256" />
      <input type="text" name="X-Amz-Date" value="20151229T000000Z" />

      Tags for File:
      <input type="input" name="x-amz-meta-tag" value="" /><br />
      <input type="hidden" name="Policy" value='<Base64-encoded policy string>' />
      <input type="hidden" name="X-Amz-Signature" value="<signature-value>" />
      File:
      <input type="file" name="file" /> <br />
      <!-- The elements after this will be ignored -->
      <input type="submit" name="submit" value="Upload to Amazon S3" />
    </form>

  </body>
</html>

```

後置參數不區分大小寫。例如，您可以指定 `x-amz-signature` 或 `X-Amz-Signature`。

使用 AWS Amplify 程式庫將瀏覽器型上傳到 Amazon S3

本節說明如何使用 AWS Amplify JavaScript 程式庫將檔案上傳至 Amazon S3。

如需設定 AWS Amplify 程式庫的詳細資訊，請參閱 [AWS Amplify 安裝和組態](#)。

使用 AWS Amplify JavaScript 程式庫將檔案上傳至 Amazon S3

AWS Amplify 程式庫 Storage 模組提供簡單的瀏覽器型上傳機制，用於管理公有或私有 Amazon S3 儲存體中的使用者內容。

Example：AWS Amplify 手動設定

下列範例顯示使用 AWS Amplify Storage 模組的手動設定。Storage 模組的預設實作使用 Amazon S3。

```
import Amplify from 'aws-amplify';
Amplify.configure({
  Auth: {
    identityPoolId: 'XX-XXXX-X:XXXXXXXXX-XXXX-1234-abcd-1234567890ab', //REQUIRED -
    Amazon Cognito Identity Pool ID
    region: 'XX-XXXX-X', // REQUIRED - Amazon Cognito Region
    userPoolId: 'XX-XXXX-X_abcd1234', //OPTIONAL - Amazon Cognito User Pool ID
    userPoolWebClientId: 'XX-XXXX-X_abcd1234', //OPTIONAL - Amazon Cognito Web
    Client ID
  },
  Storage: {
    bucket: '', //REQUIRED - Amazon S3 bucket
    region: 'XX-XXXX-X', //OPTIONAL - Amazon service region
  }
});
```

Example：將資料放入 Amazon S3

下列範例示範如何將資料put公開至 Amazon S3。

```
Storage.put('test.txt', 'Hello')
  .then(result => console.log(result))
  .catch(err => console.log(err));
```

下列範例示範如何將資料put私有至 Amazon S3。

```
Storage.put('test.txt', 'Private Content', {
  level: 'private',
  contentType: 'text/plain'
})
```

```
.then (result => console.log(result))  
.catch(err => console.log(err));
```

如需使用 AWS Amplify Storage 模組的詳細資訊，請參閱 [AWS Amplify Storage](#)。

詳細資訊

[AWS Amplify 快速入門](#)

常見請求標頭

下表說明各種 Amazon S3 REST 請求類型可以使用的標頭。

標頭名稱	說明
Authorization	請求驗證所需的資訊。如需詳細資訊，請參閱《Amazon Simple Storage Service 開發人員指南》中的 身分驗證標頭 。對於匿名請求，不需要此標頭。
Access-Control-Request-Method	作為預檢 CORS 請求傳送的 HTTP 方法清單。如果傳輸前 CORS 評估成功，則允許在下列 CORS 請求中使用指定的方法。
Content-Length	根據 RFC 2616 的訊息長度 (無標題)。載入 XML PUTs 和操作需要此標頭，例如記錄和 ACLs。
Content-Type	資源的內容類型，以防請求在內文中有內容。範例：text/plain
Content-MD5	根據 RFC 1864，訊息的 base64 編碼 128 位元 MD5 摘要 (不含標頭)。此標頭可以用作訊息完整性檢查，以驗證資料是否與最初傳送的資料相同。雖然這是選用的，但我們建議您使用 Content-MD5 機制做為 end-to-end 完整性檢查。如需 REST 請求身分驗證的詳細資訊，請參閱《Amazon Simple Storage Service 開發人員指南》中的 REST 身分驗證 。
Date	用來建立 Authorization 標頭中包含的簽章的日期。如果要使用 Date 標頭簽署時，必須指定要使用 ISO 8601 基本格式。在這種情況下，不需要 x-amz-date 標頭。請注意，當 x-amz-date 存在時，一律覆寫 Date 標頭的值。 如果日期標頭不會用來簽署，則它可以是RFC 2616 (3.3 節) 指定的完整日期格式之一。例如，日期/時間 Wed, 01 Mar 2006 12:00:00 GMT 是與 Amazon S3 搭配使用的有效日期/時間標頭。

標頭名稱	說明
	如果您使用 Date 標頭進行簽署，則必須採用 ISO 8601 基本的 YYYYMMDD'T'HHMMSS'Z' 格式。 如果 Date 已指定，但不是 ISO 8601 基本格式，則您也必須包含 x-amz-date 標頭。如果 Date 是以 ISO 8601 基本格式指定的，那麼這對於簽署請求已足夠，並且您不需要 x-amz-date 標頭。如需詳細資訊，請參閱 Amazon Web Services 詞彙表 中的 Signature 第 4 版中的處理日期。
Expect	當您的應用程式使用 100 個繼續時，它在收到確認之前不會傳送請求內文。如果根據標頭拒絕了訊息，則不會傳送訊息本文。只有在您傳送內文時，才能使用此標頭。 有效值：100-continue
Host	對於路徑樣式請求，值為 s3.amazonaws.com 。對於虛擬樣式請求，值為 BucketName.s3.amazonaws.com 。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 虛擬託管。 HTTP 1.1 需要此標頭（大多數工具組會自動新增此標頭）；HTTP/1.0 請求為選用。
Origin	指定初始請求者的伺服器名稱的端點。
x-amz-content-sha256	使用簽章第 4 版來驗證請求時，此標頭會提供請求承載的雜湊。如需更多資訊，請參閱授權標頭的簽章計算：在單一區塊中傳輸承載AWS（簽章第 4 版）。在區塊中上傳物件時，您可以將值設定為，STREAMING-AWS4-HMAC-SHA256-PAYLOAD 表示簽章僅涵蓋標頭，而且沒有承載。如需詳細資訊，請參閱授權標頭的簽章計算：以多個區塊傳輸承載（區塊上傳）AWS（簽章版本 4）。

標頭名稱	說明
x-amz-date	用來在授權標頭中建立簽章的日期。格式必須是 ISO 8601 基本的 YYYYMMDD'T'HHMMSS'Z' 格式。例如，日期/時間 20170210T120000Z 可搭配 x-amz-date Amazon S3 使用。 x-amz-date 對於所有請求都是選用的；可用來覆寫用於簽署請求的日期。如果 Date 標頭是以 ISO 8601 基本格式指定的，則不需要 x-amz-date。當 x-amz-date 存在時，一律覆寫 Date 標頭的值。如需詳細資訊，請參閱 Amazon Web Services 詞彙表 中的 Signature 第 4 版中的處理日期。
x-amz-security-token	此標頭可用於下列案例： - 若要提供 Amazon DevPay 操作的安全字符 - 使用 Amazon DevPay 的每個請求都需要兩個 x-amz-security-token 標頭：一個用於產品字符，另一個用於使用者字符。當 Amazon S3 收到已驗證的請求時，它會比較計算的簽章與提供的簽章。用於計算簽章的格式不正確的多值標頭可能會導致身分驗證問題。 - 若要在使用臨時安全登入資料時提供安全字符 - 使用您從 IAM 取得的臨時安全登入資料提出請求時，您必須使用此標頭提供安全字符。若要進一步了解暫時安全登入資料，請參閱 提出請求。 使用 Amazon DevPay 的請求和使用臨時安全登入資料簽署的請求需要此標頭。

常見回應標頭

下表說明大多數 Amazon S3 回應常見的回應標頭。

名稱	描述
Access-Control-Allow-Credentials	布林值，可判斷伺服器是否允許 CORS 請求包含登入資料。如果Access-Control-Allow-Origin 請求標頭設定為 '*'，則會省略Access-Control-Allow-Credentials 回應標頭，否則會在 CORS 評估成功時將其設定為 true。 類型：布林值 預設：無
Access-Control-Allow-Headers	CORS 請求允許的 HTTP 標頭清單。Access-Control-Allow-Headers 回應標頭會針對成功的 CORS 評估傳回，並明確指定所有允許的Access-Control-Request-Headers。 類型：字串 預設：無
Access-Control-Allow-Methods	指定允許哪些 HTTP 方法的清單。只有在 CORS 評估成功時，Amazon S3 才會允許來自允許 CORS 方法的 CORS 請求。 類型：字串 預設：無
Access-Control-Allow-Origin	允許原始伺服器的位置。只有在 CORS 評估成功時，Amazon S3 才會傳送Access-Control-Allow-Origin 回應標頭。如果請求原始伺服器 '*' 符合 CORS 組態允許的原始伺服器，則會在此回應標頭中 '*' 傳回，而非原始原始伺服器。 類型：字串 預設：無

名稱	描述
Access-Control-Expose-Headers	允許伺服器識別回應標頭的清單，在 CORS 評估成功時公開應用程式的存取權。 類型：字串 預設：無
Access-Control-Max-Age	瀏覽器可以快取資源、HTTP 方法和原始伺服器所識別之 CORS 預檢請求回應的秒數。Access-Control-Max-Age 回應標頭只會在 CORS 評估成功時傳回。 類型：整數 預設：無
Vary	一份清單，指出 CORS 評估結果在哪些請求標頭上有所不同。Vary 回應標頭只會在 CORS 評估成功時傳回。 類型：字串 預設：無
Content-Length	回應內文以位元組為單位的長度。 類型：字串 預設：無
Content-Type	內容的 MIME 類型。例如 Content-Type: text/html; charset=utf-8。 類型：字串 預設：無

名稱	描述
Connection	指定與伺服器的連線是開啟還是關閉的值。 類型：列舉 有效值: open close 預設：無
Date	Amazon S3 回應的日期和時間；例如，2006 年 3 月 1 日星期三 12:00:00 GMT。 類型：字串 預設：無
ETag	實體標籤 (ETag) 代表物件的特定版本。ETag 只會反映物件內容的變更，而非其中繼資料的變更。ETag 可能是 (也可能不是) 物件資料的 MD5 摘要。它是否取決於物件的建立方式及其加密方式，如下所示： • 透過 AWS 管理主控台 或 物件POST Object、或 Copy操作建立的PUT物件： • 使用 Amazon S3 受管金鑰 (SSE-S3) 進行伺服器端加密的純文字物件，其 ETags是其資料的 MD5 摘要。 • 使用客戶提供的金鑰 (SSE-C) 或 AWS Key Management Service (AWS KMS) 金鑰 (SSE-KMS) 進行伺服器端加密的物件，其 ETags不是其物件資料的 MD5 摘要。 • 由分段上傳或上傳組件複製操作建立的物件具有非 MD5 摘要的 ETags，無論加密方法為何。 類型：字串
Server	建立回應的伺服器名稱。 類型：字串 預設：AmazonS3

名稱	描述
x-amz-delete-marker	指定傳回的物件是 (true) 還是不是 (false) 刪除標記的值。 類型：布林值 有效值: true false 預設：false
x-amz-id-2	特殊字符，可與 x-amz-request-id 標頭搭配使用，以協助 AWS 故障診斷問題。如需 支援 使用這些請求 IDs 的詳細資訊，請參閱疑難排解 Amazon S3。 類型：字串 預設：無
x-amz-request-id	Amazon S3 建立的值，可唯一識別請求。此值會與 x-amz-id-2 標頭搭配使用，以協助 AWS 故障診斷問題。如需 支援 使用這些請求 IDs 的詳細資訊，請參閱疑難排解 Amazon S3。 類型：字串 預設：無
x-amz-server-side-encryption	在 Amazon S3 中存放此物件時使用的伺服器端加密演算法（例如，AES256、aws:kms）。 有效值: AES256 aws:kms
x-amz-version-id	物件的版本。當您啟用版本控制時，Amazon S3 會為新增至儲存貯體的物件產生隨機數字。值為 UTF-8 編碼且 URL 就緒。當您 PUT 在暫停版本控制的儲存貯體中有物件時，版本 ID 一律為 null。 類型：字串 有效值：null 任何 URL-ready、UTF-8 編碼字串 預設：null

錯誤回應

本節提供有關 Amazon S3 錯誤的參考資訊。

Note

- 一般而言，AWS 會針對 HTTP 200 OK 成功回應和 HTTP 4XX 用戶端錯誤回應的請求對 S3 儲存貯體擁有者收費。儲存貯體擁有者不會收到 HTTP 5XX 伺服器錯誤回應的帳單，例如 HTTP 503 Slow Down 錯誤。如需 HTTP 下的 S3 錯誤代碼 3XX 和未計費 4XX 狀態碼的詳細資訊，請參閱《[Amazon S3 使用者指南](#)》中的 [Amazon S3 錯誤回應的帳單](#)。Amazon S3 如需有關儲存貯體設定為申請者付款儲存貯體之帳單費用的詳細資訊，請參閱《Amazon S3 使用者指南》中的 [申請者付款費用的運作方式](#)。
- HTTP 上的 SOAP 支援已淘汰，但仍可透過 HTTPS 取得 SOAP。SOAP 不支援新的 Amazon S3 功能。我們建議您使用 REST API 或 AWS SDKs，而不是使用 SOAP。

主題

- [REST 錯誤回應](#)
- [錯誤代碼清單](#)
- [SELECT 物件內容錯誤代碼的清單](#)
- [複寫相關錯誤代碼的清單](#)
- [標記相關錯誤代碼的清單](#)
- [Amazon S3 註釋相關錯誤代碼的清單](#)
- [Amazon S3 on Outposts 錯誤代碼清單](#)
- [Amazon S3 Storage Lens 錯誤代碼清單](#)
- [Amazon S3 Object Lambda 錯誤代碼清單](#)
- [Amazon S3 非同步錯誤代碼清單](#)
- [Amazon S3 Access Grants 錯誤代碼清單](#)
- [Amazon FSx 相關錯誤代碼的清單](#)
- [Amazon S3 Tables 錯誤代碼清單](#)
- [Amazon S3 錯誤的最佳實務](#)

REST 錯誤回應

發生錯誤時，標頭資訊會包含下列項目：

- Content-Type: application/xml
- 適當的 3xx、4xx 或 5xx HTTP 狀態碼

回應的內文也包含錯誤的相關資訊。下列範例錯誤回應顯示所有 REST 錯誤回應通用的回應元素結構。

```
<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>NoSuchKey</Code>
  <Message>The resource you requested does not exist</Message>
  <Resource>/mybucket/myfoto.jpg</Resource>
  <RequestId>4442587FB7D0A2F9</RequestId>
</Error>
```

下表說明 REST 錯誤回應元素。

名稱	描述
Code	錯誤代碼是一個字串，專門用於識別錯誤狀況。偵測及依類型處理錯誤的程式必須讀取及了解此代碼。如需詳細資訊，請參閱錯誤代碼清單。 類型：字串 Ancestor：Error
Error	所有錯誤元素的容器。 類型：容器 Ancestor：無
Message	錯誤訊息包含錯誤狀況的一般英文描述，目標對象是人。一般簡單的程式在發生其無法處理的錯誤狀況時，會直接對最終使用者顯示訊息。複雜一些的程序因為錯誤處理更加周密，而且經過適當的國際化，所以很可能會忽略錯誤訊息。 類型：字串

名稱	描述
	Ancestor : Error
RequestId	與錯誤相關聯的請求 ID。 類型：字串 Ancestor : Error
Resource	涉及錯誤的儲存貯體或物件。 類型：字串 Ancestor : Error

許多錯誤回應包含額外的結構化資料，負責診斷程式設計錯誤的開發人員應詳加閱讀及了解。例如，如果您傳送的 Content-MD5 標頭具有與伺服器上計算的摘要不相符的 REST PUT 請求，您會收到BadDigest錯誤。錯誤回應也包含伺服器計算摘要的詳細元素，以及您告知伺服器預期的摘要。您可以在開發時使用此資訊診斷錯誤。在生產環境中，一支運作良好的程式可能將此資訊包含在其錯誤日誌中。

如需一般回應元素的相關資訊，請前往[錯誤回應](#)。

錯誤代碼清單

下表列出 Amazon S3 錯誤代碼。

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
AccessControlListNotSupported	儲存貯ACLs。	400 Bad Request	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
AccessDenied	存取遭拒	403 Forbidden	用戶端
AccessPointAlreadyOwnedByYou	帳戶中已存在具有相同名稱的存取點。	409 Conflict	用戶端
AccountProblem	您的 發生問題 AWS 帳戶 ，導致操作無法成功完成。如需進一步協助，請參閱 聯絡我們 。	403 Forbidden	用戶端
AllAccessDisabled	此 Amazon S3 資源的所有存取權都已停用。如需進一步協助，請參閱 聯絡我們 。	403 Forbidden	用戶端
AmbiguousGrantByEmailAddress	您提供的電子郵件地址與多個帳戶相關聯。	400 Bad Request	用戶端
AuthorizationHeaderMalformed	您提供的授權標頭無效。	400 Bad Request	N/A
AuthorizationQueryParametersError	您提供的授權查詢參數無效。	400 Bad Request	N/A
BadDigest	您指定的 Content-MD5 或檢查總和值不符合伺服器收到的值。	400 Bad Request	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
BucketAlreadyExists	無法使用請求的儲存貯體名稱。儲存貯體命名空間由系統的所有使用者共用。請指定不同的名稱，然後再試一次。	409 Conflict	用戶端
BucketAlreadyOwnedByYou	您嘗試建立的儲存貯體已存在，且您擁有該儲存貯體。Amazon S3 會在除美國東部（維吉尼亞北部）區域 (us-east-1) AWS 區域 以外的所有中傳回此錯誤。為了舊版相容性，如果您重新建立已在 us-east-1 中擁有的現有儲存貯體，Amazon S3 會傳回 200 OK 並重設儲存貯體存取控制清單 (ACLs)。 對於 Amazon S3 on Outposts，您嘗試建立的儲存貯體已存在於 Outpost 中，且您擁有該儲存貯體。	409 Conflict (在 us-east-1 以外的所有區域中)	用戶端
BucketHasAccessPointsAttached	您嘗試刪除的儲存貯體已連接存取點。在刪除儲存貯體之前刪除存取點。	400 Bad Request	用戶端
BucketNotEmpty	您嘗試刪除的儲存貯體不是空的。	409 Conflict	用戶端
ClientTokenConflict	您的多區域存取點羣等性字符已用於不同的請求。	409 Conflict	用戶端
ConnectionClosedByRequester	在讀取 WriteGetObjectResponse 本文而發生錯誤時，傳回至原始發起人。	400 Bad Request	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
ConditionalRequestConflict	發生衝突的操作。如果使用 PutObject，您可以重試請求。如果使用分段上傳，您應該啟動另一個CreateMultipartUpload 請求並重新上傳每個部分。	409 Conflict	用戶端
CredentialsNotSupported	此請求不支援 登入資料。	400 Bad Request	用戶端
CrossLocationLoggingProhibited	不允許跨區域記錄。一個儲存貯體中的儲存貯體 AWS 區域 無法將資訊記錄到另一個區域中的儲存貯體。	403 Forbidden	用戶端
DeviceNotActiveError	裝置目前未處於作用中。	400 Bad Request	用戶端
EndpointNotFound	將請求導向正確的端點。	400 Bad Request	用戶端
EntityTooSmall	您建議的上傳小於允許的物件大小下限。	400 Bad Request	用戶端
EntityTooLarge	您提議的上傳超過允許的物件大小上限。如需詳細資訊，請參閱《》中的 Amazon Simple Storage Service 端點和配額 AWS 一般參考。	400 Bad Request	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
	您建議的下載超過允許的大小上限。	405 Method Not Allowed	用戶端
ExpiredToken	提供的權杖已過期。	400 Bad Request	用戶端
IllegalLocationConstraintException	此錯誤可能發生的原因如下： - 您嘗試從與存在儲存貯體的區域所不同的區域存取儲存貯體。 - 您嘗試建立儲存貯體，其位置限制對應至與傳送請求的區域端點不同的區域。	400 Bad Request	用戶端
IllegalVersioningConfigurationException	請求中指定的版本控制組態無效。	400 Bad Request	用戶端
IncompleteBody	您未提供 Content-Length HTTP 標頭指定的位元組數。	400 Bad Request	用戶端
IncorrectEndpoint	指定的儲存貯體存在於另一個區域中。將請求導向正確的端點。	400 Bad Request	用戶端
IncorrectNumberOfFilesInPostRequest	POST 只需要每個請求一個檔案上傳。	400 Bad Request	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InlineDataTooLarge	內嵌資料超過允許的大小上限。	400 Bad Request	用戶端
InternalError	發生內部錯誤。請再試一次。	500 Internal Server Error	Server
InvalidAccessKeyId	您提供的 AWS 存取金鑰 ID 不存在於我們的記錄中。	403 Forbidden	用戶端
InvalidAccessPoint	指定的存取點名稱或帳戶無效。	400 Bad Request	用戶端
InvalidAccessPointAliasError	指定的存取點別名名稱無效。	400 Bad Request	用戶端
InvalidAddressingHeader	您必須指定匿名角色。	N/A	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidArgument	此錯誤可能發生的原因如下： - ListBuckets 請求的地區端點與 bucket-region 參數中指定的區域不同。 - 指定的引數無效。 - 請求缺少必要的標頭。 - 指定的引數不完整或格式錯誤。 - 指定的引數長度必須大於或等於 3。	400 Bad Request	用戶端
InvalidBucketAclWithObjectOwnership	儲存貯體無法以 ObjectOwnership 的 BucketOwnerEnforced 設定設定 ACLs。	400 Bad Request	用戶端
InvalidBucketName	指定的儲存貯體無效。	400 Bad Request	用戶端
InvalidBucketOwnerAWSAccountID	預期的儲存貯體擁有者參數的值必須是 AWS 帳戶 ID。	400 Bad Request	用戶端
InvalidBucketState	請求對儲存貯體的目前狀態無效。	409 Conflict	用戶端
InvalidDigest	您指定的 Content-MD5 或檢查總和值無效。	400 Bad Request	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidEncryptionAlgorithmError	您指定的加密請求無效。有效值為 AES256。	400 Bad Request	用戶端
InvalidHostHeader	請求中提供的主機標頭使用了不正確的樣式定址。	400 Bad Request	用戶端
InvalidHttpMethod	使用非預期的 HTTP 方法提出請求。	400 Bad Request	用戶端
InvalidLocationConstraint	指定的位置（區域）限制無效。如需為儲存貯體選取區域的詳細資訊，請參閱 儲存貯體概觀 。	400 Bad Request	用戶端
InvalidObjectState	操作對物件的目前狀態無效。	403 Forbidden	用戶端
InvalidPart	找不到一或多個指定的組件。組件可能尚未上傳，或指定的實體標籤可能不符合組件的實體標籤。	400 Bad Request	用戶端
InvalidPartOrder	組件清單未按遞增順序排列。組件清單必須依組件編號的順序指定。	400 Bad Request	用戶端
InvalidPayer	已停用針對此物件的所有存取權。如需進一步協助，請參閱 聯絡我們 。	403 Forbidden	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidPolicyDocument	表單的內容不符合政策文件中指定的條件。	400 Bad Request	用戶端
InvalidRange	無法滿足請求的範圍。	416 請求的範圍無法滿足	用戶端
InvalidRegion	您已嘗試在尚未選擇加入的區域中建立多區域存取點。	403 禁止	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidRequest	此錯誤可能發生的原因如下： • 未分頁的 ListBuckets 請求是從核准的一般用途儲存貯體配額高於 10,000 的帳戶提出。您必須提出分頁請求，以列出帳戶中超過 10,000 個儲存貯體的儲存貯體。 • 請求使用錯誤的簽章版本。使用 AWS4-HMAC-SHA256 (Signature 第 4 版)。 • 只能為現有儲存貯體建立存取點。 • 存取點未處於可以刪除的狀態。 • 只能針對現有儲存貯體列出存取點。 • 下一個權杖無效。 • 必須至少在生命週期規則中指定一個動作。 • 必須指定至少一個生命週期規則。 • 生命週期規則的數量不得超過允許的 1000 個規則限制。 •	400 Bad Request	用戶 端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
	MaxResults 參數值的範圍無效。 • 必須透過 HTTPS 連線提出 SOAP 請求。 • 具有非 DNS 相容名稱的儲存貯體不支援 Amazon S3 Transfer Acceleration。 • 名稱中具有句點 (.) 的儲存貯體不支援 Amazon S3 Transfer Acceleration。 • Amazon S3 Transfer Acceleration 端點僅支援虛擬樣式請求。 • 此儲存貯體上未設定 Amazon S3 Transfer Acceleration。 • 此儲存貯體已停用 Amazon S3 Transfer Acceleration。 • 此儲存貯體不支援 Amazon S3 Transfer Acceleration。如需協助，請聯絡 支援。 • 無法在此儲存貯體上啟用 Amazon S3 Transfer Acceleration。如需協助，請聯絡 支援。 •		

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
	HTTP 標頭和查詢參數中提供的值衝突。 - HTTP 標頭和 POST 表單欄位中提供的衝突值。 - 對大小超過 5GB 的物件提出 CopyObject 請求。		
InvalidSessionException	如果工作階段由於逾時或過期而不再存在，則傳回。	400 Bad Request	用戶端
InvalidSignature	伺服器計算的請求簽章與您提供的簽章不相符。檢查您的 AWS 私密存取金鑰和簽署方法。如需詳細資訊，請參閱 簽署和驗證 REST 請求 。	400 Bad Request	用戶端
InvalidSecurity	提供的安全憑證無效。	403 Forbidden	用戶端
InvalidSOAPRequest	SOAP 請求本文無效。	400 Bad Request	用戶端
InvalidStorageClass	您指定的儲存類別無效。	400 Bad Request	用戶端
InvalidTargetBucketForLogging	用於記錄的目標儲存貯體不存在、不是您擁有，或沒有日誌傳遞群組的適當授權。	400 Bad Request	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidToken	提供的權杖格式錯誤或以其他方式無效。	400 Bad Request	用戶端
InvalidURI	無法剖析指定的 URI。	400 Bad Request	用戶端
KeyTooLongError	您的金鑰過長。	400 Bad Request	用戶端
KMS.DisabledException	因為指定的 KMS 未啟用，所以請求遭到拒絕。	400 Bad Request	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
KMS.InvalidKeyUsageException	由於下列其中一個原因，請求遭拒： • KMS 金鑰的 KeyUsage 值與 API 操作不相容。 • 為操作指定的加密演算法或簽署演算法與 KMS 金鑰 (KeySpec) 中的金鑰材料類型不相容。 對於加密、解密、重新加密和產生資料金鑰，KeyUsage 必須是 ENCRYPT_DECRYPT。對於簽署和驗證訊息，KeyUsage 必須是 SIGN_VERIFY。對於產生和驗證訊息身分驗證代碼 (MAC)，KeyUsage 必須是 GENERATE_VERIFY_MAC。對於衍生金鑰協議秘密，KeyUsage 必須是 KEY_AGREEMENT。若要尋找 KMS 金鑰的 KeyUsage，請使用 DescribeKey 操作。若要尋找特定 KMS 金鑰支援的加密或簽署演算法，請使用 DescribeKey 操作。	400 Bad Request	用戶 端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
KMS.KMSInvalidStateException	因為所指定資源的狀態對於此請求無效，所以請求遭到拒絕。此例外狀況代表發生下列其中一種情況： - KMS 金鑰的金鑰狀態與操作不相容。 若要尋找金鑰狀態，請使用 DescribeKey 操作。如需哪些金鑰狀態與每個 KMS 操作相容的詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的 AWS KMS keys 金鑰狀態。 - 對於自訂金鑰存放區中 KMS 金鑰的密碼編譯操作，此例外狀況代表具有許多可能原因的一般失敗情形。若要識別原因，請參閱例外狀況隨附的錯誤訊息。	400 Bad Request	用戶 端
KMS.NotFoundException	因為找不到指定的實體或資源，所以請求遭到拒絕。	400 Bad Request	用戶 端
MalformedACLError	您提供的 ACL 格式不正確，或未經過我們發佈的結構描述驗證。	400 Bad Request	用戶 端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
MalformedPOSTRequest	POST 請求的內文格式不正確 multipart/form-data。	400 Bad Request	用戶端
MalformedXML	您提供的 XML 格式不正確或未經過我們發佈的結構描述驗證。	400 Bad Request	用戶端
MaxMessageLengthExceeded	您的請求太大。	400 Bad Request	用戶端
MaxPostPreDataLengthExceededError	上傳檔案之前的 POST 請求欄位過大。	400 Bad Request	用戶端
MetadataTooLarge	您的中繼資料標頭超過允許的中繼資料大小上限。	400 Bad Request	用戶端
MethodNotAllowed	指定的方法不得使用此資源。	405 Method Not Allowed	用戶端
MissingAttachment	預期 SOAP 連接，但找不到。	400 錯誤的請求	用戶端
MissingAuthenticationToken	請求未簽署。	403 禁止	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
MissingContentLength	您必須提供 Content-Length HTTP 標頭。	411 Length Required	用戶端
MissingRequestBodyError	您傳送了空白 XML 文件作為請求。	400 Bad Request	用戶端
MissingSecurityElement	SOAP 1.1 請求缺少安全元素。	400 Bad Request	用戶端
MissingSecurityHeader	您的請求缺少必要的標頭。	400 Bad Request	用戶端
NoLoggingStatusForKey	不存在金鑰的日誌狀態子資源。	400 Bad Request	用戶端
NoSuchAsyncRequest	找不到指定的請求。	404 Not Found	用戶端
NoSuchBucket	指定的儲存貯體不存在。	404 Not Found	用戶端
NoSuchBucketPolicy	指定的儲存貯體沒有儲存貯體政策。	404 Not Found	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
NoSuchCORSConfiguration	指定的儲存貯體沒有 CORS 組態。	404 Not Found	用戶端
NoSuchKey	指定的金鑰不存在。	404 Not Found	用戶端
NoSuchLifecycleConfiguration	指定的生命週期組態不存在。	404 Not Found	用戶端
NoSuchMultiRegionAccessPoint	指定的多區域存取點不存在。	404 Not Found	用戶端
NoSuchObjectLockConfiguration	指定的物件沒有 ObjectLock 組態。	404 Not Found	用戶端
NoSuchWebsiteConfiguration	指定的儲存貯體沒有網站組態。	404 Not Found	用戶端
NoSuchTagSet	指定的標籤不存在。	404 Not Found	用戶端
NoSuchUpload	指定的分段上傳不存在。上傳 ID 可能無效，或分段上傳可能已中止或完成。	404 Not Found	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
NoSuchVersion	請求中指定的版本 ID 與現有版本不符。	404 Not Found	用戶端
NotDeviceOwnerError	產生權杖的裝置不是由經過驗證的使用者所擁有。	400 Bad Request	用戶端
NotImplemented	您提供的標頭表示未實作的功能。	501 Not Implemented	Server
NotModified	資源未變更。	304 Not Modified	Server
NoTransformationDefined	找不到此 Object Lambda 存取點的轉換。	404 Not Found	用戶端
NotSignedUp	您的帳戶未註冊 Amazon S3 服務。您必須進行註冊才可使用 Amazon S3。您可以在下列 URL 註冊： https://aws.amazon.com/s3	403 Forbidden	用戶端
ObjectLockConfigurationNotFoundError	此儲存貯體不存在物件鎖定組態。	404 Not Found	用戶端
OwnershipControlsNotFoundError	找不到儲存貯體擁有權控制項。	404 Not Found	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
OperationAborted	目前正在針對此資源進行衝突的條件式操作。請再試一次。	409 Conflict	用戶端
PermanentRedirect	必須使用指定的端點來定址您嘗試存取的儲存貯體。將所有未來的請求傳送至此端點。	301 Moved Permanently	用戶端
PermanentRedirectControlError	必須使用指定的端點來定址您嘗試存取的 API 操作。將所有未來的請求傳送至此端點。	301 Moved Permanently	用戶端
PreconditionFailed	您指定的至少一個先決條件未保留。	412 Precondition Failed	用戶端
Redirect	暫時重新導向。當網域名稱系統 (DNS) 伺服器更新時，系統會將您重新導向至儲存貯體。	307 Temporary Redirect	用戶端
RequestHeaderSectionTooLarge	用於提出請求的請求標頭和查詢參數超過允許的大小上限。	400 Bad Request	用戶端
RequestIsNotMultipartContent	儲存貯體 POST 請求必須是 enclosure-type multipart/form-data。	412 Precondition Failed	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
RequestTimeout	您在逾時期間內未讀取或寫入伺服器的通訊端連線。	400 Bad Request	用戶端
RequestTimeTooSkewed	請求時間和伺服器時間之間的差異過大。	403 Forbidden	用戶端
RequestTorrentOfBucketError	不允許請求儲存貯體的 torrent 檔案。	400 Bad Request	用戶端
ResponseInterrupted	在讀取 WriteGetObjectResponse 本文而發生錯誤時，傳回至原始發起人。	400 Bad Request	用戶端
RestoreAlreadyInProgress	物件還原已在進行中。	409 Conflict	用戶端
ServerSideEncryptionConfigurationNotFound	找不到伺服器端加密組態。	400 Bad Request	用戶端
ServiceUnavailable	服務無法處理請求。	503 Service Unavailable	Server
SignatureDoesNotMatch	伺服器計算的請求簽章與您提供的簽章不相符。檢查您的 AWS 私密存取金鑰和簽署方法。如需詳細資訊，請參閱 REST 身分驗證 和 SOAP 身分驗證 。	403 Forbidden	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
SlowDown	請降低您的請求率。	503 Slow Down	Server
503 SlowDown	減速	503 Slow Down	Server
TagPolicyException	標籤政策不允許下列標籤索引鍵的指定值。	400 Bad Request	用戶端
TemporaryRedirect	當網域名稱系統 (DNS) 伺服器更新時，系統會將您重新導向至儲存貯體。	307 Temporary Redirect	用戶端
TokenCodeInvalidError	您提供的序號和/或權杖代碼無效。	400 Bad Request	用戶端
TokenRefreshRequired	提供的權杖必須重新整理。	400 Bad Request	用戶端
TooManyAccessPoints	您已嘗試建立比 帳戶允許的更多存取點。如需詳細資訊，請參閱《》中的 Amazon Simple Storage Service 端點和配額 AWS 一般參考。	400 Bad Request	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
TooManyBuckets	您嘗試建立的儲存貯體數量超過帳戶的允許數量。如需詳細資訊，請參閱《》中的 Amazon Simple Storage Service 端點和配額 AWS 一般參考。	400 Bad Request	用戶端
TooManyMultiRegionAccessPointregionsError	您嘗試建立多區域存取點的區域超過帳戶允許的區域。如需詳細資訊，請參閱《》中的 Amazon Simple Storage Service 端點和配額 AWS 一般參考。	400 Bad Request	用戶端
TooManyMultiRegionAccessPoints	您已嘗試建立比帳戶允許的多區域存取點更多。如需詳細資訊，請參閱《》中的 Amazon Simple Storage Service 端點和配額 AWS 一般參考。	400 Bad Request	用戶端
UnauthorizedAccessError	僅適用於中國區域。向沒有 ICP 授權的儲存貯體提出請求時傳回。如需詳細資訊，請參閱 ICP 備案 。	403 Forbidden	用戶端
UnexpectedContent	此請求包含不支援的內容。	400 Bad Request	用戶端
UnexpectedIPError	僅適用於中國區域。因為具有未預期 IP，所以此請求遭到拒絕。	403 Forbidden	用戶端
UnsupportedArgument	請求包含不支援的引數。	400 Bad Request	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
UnsupportedSignature	提供的請求使用不支援的 STS 權杖版本簽署，或者不支援簽章版本。	400 Bad Request	用戶端
UnresolvableGrantByEmailAddress	您提供的電子郵件地址不符合任何記錄的帳戶。	400 Bad Request	用戶端
UserKeyMustBeSpecified	儲存貯體 POST 請求必須包含指定的欄位名稱。如果已指定，請檢查欄位的順序。	400 Bad Request	用戶端
NoSuchAccessPoint	指定的存取點不存在。	404 Not Found	用戶端
InvalidTag	您的請求包含無效的標籤輸入。例如，您的請求可能包含重複的金鑰、太長的金鑰或值，或系統標籤。	400 Bad Request	用戶端
MalformedPolicy	您的政策包含無效的委託人。	400 Bad Request	用戶端

SELECT 物件內容錯誤代碼的清單

Important

Amazon S3 Select 不再提供給新客戶。Amazon S3 Select 的現有客戶可以繼續照常使用此功能。[進一步了解](#)

下表包含SELECT Object Content可能會傳回的特殊錯誤。如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
AmbiguousFieldName	欄位名稱符合 檔案中的多個欄位。請檢查 SQL 表達式和 檔案，然後再試一次。	400	用戶端
Busy	服務無法使用。請稍後再試。	503	用戶端
CastFailed	嘗試使用 SQL 表達式中的 CAST 失敗，從一種資料類型轉換為另一種資料類型。	400	用戶端
ColumnTooLong	結果中資料欄的長度大於 maxCharsPerColumn 1 MB。	400	用戶端
CSVEscapingRecordDelimiter	在 檔案中找到引號的記錄分隔符號。若要允許引用的記錄分隔符號，請將 AllowQuotedRecordDelimiter 設定為 'TRUE'。	400	用戶端
CSVParsingError	剖析 CSV 檔案時發生錯誤。請檢查檔案，然後再試一次。	400	用戶端
CSVUnescapedQuote	在剖析 CSV 檔案時，發現未逸出的引號。若要允許引用的記錄分隔符號，請將 AllowQuotedRecordDelimiter 設定為 'TRUE'。	400	用戶端
EmptyRequestBody	請求內文不能為空白。	400	用戶端
EvaluatorBindingDoesNotExist	SQL 表達式中不存在提供的資料欄名稱或路徑。	400	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
EvaluatorInvalidArguments	SQL 表達式中函數呼叫中的引數數目不正確。	400	用戶端
EvaluatorInvalidTimestampFormatPattern	SQL 表達式中的時間戳記格式字串無效。	400	用戶端
EvaluatorInvalidTimestampFormatPatternSymbol	時間戳記格式模式在 SQL 表達式中包含無效的符號。	400	用戶端
EvaluatorInvalidTimestampFormatPatternSymbolForParsing	時間戳記格式模式包含無法套用至 SQL 表達式中時間戳記剖析的有效格式符號。	400	用戶端
EvaluatorInvalidTimestampFormatPatternToken	時間戳記格式模式在 SQL 表達式中包含無效的字符。	400	用戶端
EvaluatorLikePatternInvalidEscapeSequence	提供給LIKE表達式的引數無效。	400	用戶端
EvaluatorNegativeLimit	LIMIT 不得為負數。	400	用戶端
EvaluatorTimestampFormatPatternDuplicateFields	時間戳記格式模式包含代表 SQL 表達式中時間戳記欄位的多個格式指標。	400	用戶端
EvaluatorTimestampFormatPatternHourClockAmPmMismatch	時間戳記格式模式包含一天 12 小時的格式符號，但也不包含 AM/PM 欄位，或包含一天 24 小時的格式指標，並在 SQL 表達式中包含 AM/PM 欄位。	400	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
EvaluatorUnterminatedTimestampFormatPatternToken	時間戳記格式模式在 SQL 表達式中包含未終止的權杖。	400	用戶端
ExpressionTooLong	SQL 表達式太長。SQL 表達式的最大位元組長度為 256 KB。	400	用戶端
ExternalEvalException	無法評估查詢。請檢查檔案，然後再試一次。	400	用戶端
IllegalSqlFunctionArgument	SQL 函數中使用非法引數。	400	用戶端
IncorrectSqlFunctionArgumentType	SQL 表達式中的函數呼叫中指定的引數類型不正確。	400	用戶端
IntegerOverflow	SQL 表達式中發生整數溢出或下溢。	400	用戶端
InternalError	發生內部錯誤。	500	用戶端
InvalidCast	嘗試使用 SQL 表達式中的 CAST 失敗，從一種資料類型轉換為另一種資料類型。	400	用戶端
InvalidColumnIndex	SQL 表達式中的資料欄索引無效。	400	用戶端
InvalidCompressionFormat	檔案不是支援的壓縮格式。僅支援 GZIP 和 BZIP2。	400	用戶端
InvalidDataSource	資料來源類型無效。僅支援 CSV、JSON 和 Parquet。	400	用戶端
InvalidDataType	SQL 表達式包含無效的資料類型。	400	用戶端
InvalidExpressionType	此 ExpressionType 值無效。僅支援 SQL 表達式。	400	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidFileHeaderInfo	此FileHeaderInfo 值無效。僅IGNORE支援 USE、NONE和 。	400	用戶端
InvalidJsonType	此JsonType值無效。僅支援 LINES DOCUMENT和 。	400	用戶端
InvalidKeyPath	SQL 表達式中的金鑰路徑無效。	400	用戶端
InvalidQuoteFields	此QuoteFields 值無效。僅支援 ASNEEDED ALWAYS和 。	400	用戶端
InvalidRequestParameter	SelectRequest 元素中的參數值無效。請檢查服務 API 文件，然後再試一次。	400	用戶端
InvalidScanRange	提供的掃描範圍無效。	400	用戶端
InvalidTableAlias	SQL 表達式包含無效的資料表別名。	400	用戶端
InvalidTextEncoding	編碼類型無效。僅支援 UTF-8 編碼。	400	用戶端
JSONParsingError	剖析 JSON 檔案時發生錯誤。請檢查檔案，然後再試一次。	400	用戶端
LexerInvalidChar	SQL 表達式包含無效的字元。	400	用戶端
LexerInvalidIONLiteral	SQL 表達式包含無效的運算子。	400	用戶端
LexerInvalidLiteral	SQL 表達式包含無效的運算子。	400	用戶端
LexerInvalidOperator	SQL 表達式包含無效的常值。	400	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
LikeInvalidInputs	SQL 表達式中提供給 LIKE 子句的引數無效。	400	用戶端
MalformedXML	提供的 XML 格式不正確，或未針對我們發佈的結構描述進行驗證。請檢查服務文件，然後再試一次。	400	用戶端
MaxOperatorsExceeded	無法剖析 SQL 表達式，請嘗試降低複雜性。例如，減少使用的運算子數量。	400	用戶端
MethodNotAllowed	指定的方法不得使用此資源。	405 Method Not Allowed	用戶端
MissingRequiredParameter	SelectRequest 實體缺少必要的參數。請檢查服務文件，然後再試一次。	400	用戶端
MultipleDataSourcesUnsupported	不支援多個資料來源。	400	用戶端
NumberFormatError	剖析數字時發生錯誤。此錯誤可能是由於整數的不足或溢位所造成。	400	用戶端
ObjectSerializationConflict	InputSerialization 指定多個格式 (CSV、JSON 或 Parquet)，或 OutputSerialization 指定多個格式 (CSV 或 JSON)。對於 InputSerialization 和 OutputSerialization，您只能為每個格式指定一個格式。	400	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
OverMaxColumn	結果中的欄數大於允許的欄數上限。	400	用戶端
OverMaxParquetBlockSize	Parquet 檔案高於資料列群組大小上限。	400	用戶端
OverMaxRecordSize	輸入或結果中的記錄長度大於 1 MB maxCharsPerRecord 的限制。	400	用戶端
ParquetParsingError	剖析 Parquet 檔案時發生錯誤。請檢查檔案，然後再試一次。	400	用戶端
ParquetUnsupportedCompressionCodec	不支援指定的 Parquet 壓縮轉碼器。	400	用戶端
ParseAsteriskIsNotAloneInSelectList	在 SQL 表達式中沒有點符號的情況下*使用時，SELECT清單中不允許其他表達式。	400	用戶端
ParseCannotMixSqbandWildcardInSelectList	無法在 SQL 表達式的SELECT清單中混合相同表達式*中的 [] 和。	400	用戶端
ParseCastArity	SQL 表達式CAST的可用性不正確。	400	用戶端
ParseEmptySelect	SQL 表達式包含空的 SELECT 子句。	400	用戶端
ParseExpected2TokenTypes	找不到 SQL 表達式中預期的字符。	400	用戶端
ParseExpectedArgumentDelimiter	找不到 SQL 表達式中預期的引數分隔符號。	400	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
ParseExpectedDatePart	找不到 SQL 表達式中的預期日期部分。	400	用戶端
ParseExpectedExpression	找不到預期的 SQL 表達式。	400	用戶端
ParseExpectedIdentForAlias	找不到 SQL 表達式中別名的預期識別符。	400	用戶端
ParseExpectedIdentForAt	找不到 SQL 表達式中 AT 名稱的預期識別符。	400	用戶端
ParseExpectedIdentForGroupName	GROUP SQL 表達式中不支援。	400	用戶端
ParseExpectedKeyword	找不到 SQL 表達式中的預期關鍵字。	400	用戶端
ParseExpectedLeftParenAfterCast	在 SQL 表達式 CAST 中找不到預期的左括號。	400	用戶端
ParseExpectedLeftParenBuiltinFunctionCall	找不到 SQL 表達式中預期的左括號。	400	用戶端
ParseExpectedLeftParenValueConstructor	找不到 SQL 表達式中預期的左括號。	400	用戶端
ParseExpectedMember	SQL 表達式包含不支援使用 MEMBER。	400	用戶端
ParseExpectedNumber	找不到 SQL 表達式中的預期數字。	400	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
ParseExpectedRightParenBuiltinFunctionCall	找不到 SQL 表達式中預期的右括號字元。	400	用戶端
ParseExpectedTokenType	找不到 SQL 表達式中預期的字符。	400	用戶端
ParseExpectedTypeName	找不到 SQL 表達式中的預期類型名稱。	400	用戶端
ParseExpectedWhenClause	找不到 SQL 表達式中的預期 WHEN 子句。CASE 不支援。	400	用戶端
ParseInvalidContextForWildcardInSelectList	在 SQL 表達 * 式的 SELECT 清單中使用 是無效的。	400	用戶端
ParseInvalidPathComponent	SQL 表達式包含無效的路徑元件。	400	用戶端
ParseInvalidTypeParameter	SQL 表達式包含無效的參數值。	400	用戶端
ParseMalformedJoin	JOIN SQL 表達式中不支援。	400	用戶端
ParseMissingIdentifierAfterAt	找不到 SQL 表達式中 @ 符號後面的預期識別符。	400	用戶端
ParseNonUnaryAggregateFunctionCall	SQL 表達式中的彙總函數僅支援一個引數。	400	用戶端
ParseSelectMissingFrom	SQL 表達式在 SELECT 清單 FROM 之後包含遺失的。	400	用戶端
ParseUnexpectedKeyword	SQL 表達式包含非預期的關鍵字。	400	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
ParseUnexpectedOperator	SQL 表達式包含非預期的運算子。	400	用戶端
ParseUnexpectedTerm	SQL 表達式包含非預期的詞彙。	400	用戶端
ParseUnexpectedToken	SQL 表達式包含非預期的字符。	400	用戶端
ParseUnknownOperator	SQL 表達式包含無效的運算子。	400	用戶端
ParseUnsupportedAlias	SQL 表達式包含不支援使用 ALIAS。	400	用戶端
ParseUnsupportedCallWithStar	SQL 表達式僅支援 COUNT 使用 (*) 做為參數。	400	用戶端
ParseUnsupportedCase	SQL 表達式包含不支援使用 CASE。	400	用戶端
ParseUnsupportedCaseClause	SQL 表達式包含不支援使用 CASE。	400	用戶端
ParseUnsupportedLiteralsGroupBy	SQL 表達式包含不支援使用 GROUP BY 。	400	用戶端
ParseUnsupportedSelect	SQL 表達式包含不支援的使用方式 SELECT。	400	用戶端
ParseUnsupportedSyntax	SQL 表達式包含不支援的語法。	400	用戶端
ParseUnsupportedToken	SQL 表達式包含不支援的字符。	400	用戶端
TruncatedInput	物件解壓縮失敗。檢查物件是否使用請求中指定的格式正確壓縮。	400	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
UnauthorizedAccess	未授權您執行此操作。	401	用戶端
UnrecognizedFormatException	記錄類型無效。	400	用戶端
UnsupportedFunction	S3 遇到不支援的 SQL 函數。	400	用戶端
UnsupportedParquetType	不支援指定的 Parquet 類型。	400	用戶端
UnsupportedRangeHeader	此操作不支援範圍標頭。	400	用戶端
UnsupportedScanRangeInput	此類型的物件不支援掃描範圍查詢。	400	用戶端
UnsupportedSqlOperation	S3 遇到不支援的 SQL 操作。	400	用戶端
UnsupportedSqlStructure	S3 遇到不支援的 SQL 結構。檢查 SQL 參考。	400	用戶端
UnsupportedStorageClass	不支援儲存體方案。僅支援 STANDARD、STANDARD_IA 和 ONEZONE_IA 儲存體方案。	400	用戶端
UnsupportedSyntax	語法無效。	400	用戶端
UnsupportedTypeForQuerying	您的查詢包含不支援的比較類型 (例如, 驗證 Parquet INT96 資料欄類型大於 0)。	400	用戶端
ValueParseFailure	SQL 表達式中發生時間戳記剖析失敗。	400	用戶端

複寫相關錯誤代碼的清單

下表包含 Replication 操作可能傳回的特殊錯誤。如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidArgument	此錯誤可能發生的原因如下： • <code><Account></code> 元素為空。它必須包含有效的帳戶 ID。 • <code><Account></code> 元素中 AWS 帳戶指定的 必須符合目的地儲存貯體擁有者。 • <code>ReplicationTime-Status</code> 必須包含值。 • <code>ReplicationTime-ReplicationTimeValue</code> 必須包含值。 • <code>Replication-ReplicationTimeValue-Minutes</code> 值必須為 15。 • <code>ReplicationMetrics</code> 必須包含 <code>Status</code>。 • <code>ReplicationMetrics</code> 必須包含 <code>EventThreshold</code> 。 • <code>EventThreshold-ReplicationTimeValue-Minutes</code> 值必須為 15。	400	用戶端

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
	Rule ID 不得包含非 ASCII 字元。		
InvalidRequest	此錯誤可能發生的原因如下： <Owner> 中的 <AccessControlTranslation> 具有值，因此必須指定 <Account> 元素。 <Account> 元素為空。它必須包含有效的帳戶 ID。 - 複寫destination 必須同時包含 ReplicationTime 和 Metrics或兩者。 - ReplicationTime 和 ReplicationMetrics 必須具有相同的狀態。 - 此 不支援 S3 複寫時間控制 (S3 RTC) AWS 區域。	400	用戶端
ReplicationConfigurationNotFoundError	此儲存貯體沒有複寫組態。	404 找不到	用戶端

標記相關錯誤代碼的清單

下表包含 TagResource、UntagResource和 ListTagsForResource操作可能針對 Storage Lens 群組傳回的特殊錯誤。如需一般 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

錯誤程式碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidRequest	資源 ARN AWS 區域 中的 不符合此請求中指定的區域。資源 ARN 中的 AWS 帳戶與此請求中指定的帳戶 ID 不相符。resourceArn 中的 AWS 分割區無效。	400 Bad Request	不支援
InvalidTag	此請求包含無效的標籤索引鍵或值。有效字元包括下列項目：[a-zA-Z+-. _:/]。標籤索引鍵最多可包含 128 個字元。標籤值最多可包含 256 個字元。您的請求中有重複的標籤索引鍵。使用者定義的標籤索引鍵不能以 開頭aws:。	400 Bad Request	不支援
NoSuchResource	指定的資源不存在。	404 Not Found	不支援
TagPolicyException	標籤政策不允許下列標籤索引鍵的指定值。	400 Bad Request	不支援
TooManyTags	標籤數量超過 50 個標籤的限制。	400 Bad Request	不支援

Amazon S3 註釋相關錯誤代碼的清單

下表包含註釋操作

(PutObjectAnnotation、GetObjectAnnotation、ListObjectAnnotations、UpdateBucketMetadata、DeleteObjectAnnotation和註釋相關CopyObject行為可能傳回的特殊錯誤。如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

錯誤程式碼	說明	HTTP 狀態碼	SOAP 故障碼字首
AccessDenied	存取遭拒。發起人沒有執行請求註釋操作的許可，或物件受到 COMPLIANCE 模式的物件鎖定保護。	403 Forbidden	不支援
AnnotationLimitExceeded	物件註釋計數超過限制 (1000)。	400 Bad Request	不支援
AnnotationNameTooLong	您的註釋名稱太長。註釋名稱的長度最多可達 512 個位元組。	400 Bad Request	不支援
AnnotationRequestConflict	目前正在針對此註釋進行衝突操作。當跨區域複製期間刪除來源註釋時，可能會發生這種情況。	409 Conflict	不支援
EntityTooSmall	您提議的上傳小於允許的最小大小。註釋承載必須至少為 1 個位元組。	400 Bad Request	不支援
InvalidAnnotationName	您的註釋名稱包含無效的字元，或名稱開頭為預留字首 (aws 或 s3，不區分大小寫)。註釋名稱只能包含 Unicode 字母、數字、底線 (_)、句點 (.) 和連字號 (-)。	400 Bad Request	不支援
InvalidArgument	註釋指令無效。有效值為 COPY 和 EXCLUDE。對於 ListObjectAnnotations，max-annotation-results 必須介於 1 到 1000 之間。	400 Bad Request	不支援
InvalidPrefix	您提供的註釋字首無效。	400 Bad Request	不支援

錯誤程式碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidRequest	此錯誤可能發生的原因如下：物件使用不支援註釋的 SSE-C 加密。來源物件具有無法複製到目的地的註釋，因為目的地位於不支援的區域。註釋資料表組態請求無效（例如，指定加密時停用資料表、啟用已啟用的資料表、更新處於 CREATING 狀態的資料表，或提交空白請求）。	400 Bad Request	不支援
MethodNotAllowed	指定的方法不得使用此資源。例如，當請求以刪除標記為目標時，會發生此錯誤。	405 Method Not Allowed	不支援
NoSuchAnnotation	指定的註釋不存在。	404 Not Found	不支援
NoSuchKey	指定的金鑰不存在。	404 Not Found	不支援
NoSuchVersion	請求中指定的版本 ID 與現有版本不符。	404 Not Found	不支援
NotImplemented	Amazon S3 目錄儲存貯體不支援註釋。	501 Not Implemented	不支援
OperationAborted	目前正在針對此資源進行衝突的條件式操作。這可能發生在並行修改期間，或在跨區域複製期間來源註釋計數或 ETag 變更時。請再試一次。	409 Conflict	不支援

錯誤程式碼	說明	HTTP 狀態碼	SOAP 故障碼字首
PreconditionFailed	您指定的至少一個先決條件未保留。x-amz-object-if-match ETag 與目前的物件 ETag 不相符。	412 Precondition Failed	不支援
UnsupportedMediaType	請求承載包含不支援的媒體類型。註釋承載必須是有效的 UTF-8 編碼文字。	415 Unsupported Media Type	不支援
UserAnnotationNameMustBeSpecified	您必須指定使用者註釋名稱。	400 Bad Request	不支援

Amazon S3 on Outposts 錯誤代碼清單

下表包含 Amazon S3 on Outposts 操作可能傳回的特殊錯誤。如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
BadRequest	由於先前的刪除嘗試，儲存貯體處於轉換狀態。請稍後再試。	400 錯誤的請求	不支援
InvalidRequest	此錯誤可能發生的原因如下： - 需要 Amazon VPC 組態。 - S3 on Outposts 存取點上不允許公開存取。	400 錯誤的請求	用戶端
InvalidOutpostState	請求對 Outpost 的目前狀態無效。	409 衝突	不支援

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidRequest	存取點未處於可以刪除的狀態。	400 錯誤的請求	不支援
NoSuchOutpost	指定的 Outpost 不存在。	404 找不到	不支援
UnsupportedOperation	不支援指定的動作。	404 找不到	不支援
InsufficientCapacity	容量不足。	507 儲存不足	不支援

Amazon S3 Storage Lens 錯誤代碼清單

下表包含 Amazon S3 Storage Lens 操作可能會傳回的特殊錯誤。如需一般 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
AccessDenied	此區域不支援做為 S3 Storage Lens 的主區域。	403 Forbidden	不支援
AccountNotAuthorized	此帳戶未獲授權使用 AWS Organizations。使用您的管理帳戶或委派管理員帳戶。	403 Forbidden	不支援
ActivityMetricsMustEnabled	必須啟用活動指標。	400 錯誤的請求	不支援
AWSOrganizationsNotInUseException	此帳戶不屬於您的組織。	403 Forbidden	不支援

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
DefaultConfigurationDeleteForbidden	無法刪除預設組態。	403 Forbidden	不支援
DuplicateStorageLensGroupARN	此組態中有兩個或多個相同 Storage Lens 群組 ARN 的項目。	400 Bad Request	不支援
EmptyExcludeContainer	發生此錯誤的原因如下： - 排除容器不能為空。 - 排除容器不能有零個儲存貯體。 - 排除容器不能有零個區域。	400 錯誤的請求	不支援
EmptyExcludeElement	您必須使用排除元素指定 Storage Lens 群組。	400 Bad Request	不支援
EmptyIncludeContainer	發生此錯誤的原因如下： - 包含的容器不能為空。 - 包含容器不能有零個儲存貯體。 - 包含的容器不能有零個區域。	400 錯誤的請求	不支援
InvalidAWSOrgArn	組態中有格式不正確的 AWS Organizations ARN。	400 錯誤的請求	不支援
EmptyIncludeElement	您必須使用包含元素指定 Storage Lens 群組。	400 Bad Request	不支援
InvalidBucketFilter	組織層級組態不支援儲存貯體篩選條件。	400 錯誤的請求	不支援

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidConfigId	組態 ID 無效。	400 錯誤的請求	不支援
InvalidDestination	S3 儲存貯體 ARN 格式錯誤。	400 錯誤的請求	不支援
InvalidEncryptionMethod	只能指定一種加密方法。	400 錯誤的請求	不支援
InvalidFilterForDefaultConfiguration	預設組態不得包含任何篩選條件。	400 錯誤的請求	不支援
InvalidIncludeExcludeContainers	您可以在組態中指定包含容器或排除容器。您無法在組態中同時指定兩者。	400 錯誤的請求	不支援
InvalidIncludeExcludeElements	僅允許一個包含或排除元素。至少必須存在一個包含或排除元素。	400 Bad Request	不支援
InvalidKMSEncryptionKeyId	KMS 金鑰 ID ARN 無效。	400 錯誤的請求	不支援
InvalidMaximumPrefixDepth	MaxDepth 必須在 【1 , 10】 範圍內。	400 錯誤的請求	不支援
InvalidMinimumStorageBytesPercentage	MinStorageBytesPercentage 必須在 【1.00 , 100.00】 範圍內。	400 錯誤的請求	不支援
InvalidOrganizationARN	組態中的 AWS Organizations ARN 無效。	400 錯誤的請求	不支援
InvalidOrganizationForDefaultConfiguration	預設組態不支援組織層級指標。	400 錯誤的請求	不支援

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidRegionForDefaultConfiguration	預設組態不支援指定的區域。	400 錯誤的請求	不支援
InvalidRegionName	區域名稱無效。	400 錯誤的請求	不支援
InvalidStorageLensArn	輸入中不需要 S3 Storage Lens ARN。	400 錯誤的請求	不支援
InvalidStorageLensGroupARN	此 Storage Lens 群組 ARN 無效，或僅允許您帳戶中的 Storage Lens 群組。此外，您必須遵循 Storage Lens 群組 ARN 結構：arn::s3::storage-lens-group/ 並遵循 64 個字元的限制。Storage Lens 群組名稱只能包含下列字元：a-z、A-Z、0-9、連字號 (-) 和底線 (_)。	400 Bad Request	不支援
MissingAccountLevelActivityMetrics	在儲存貯體層級啟用活動指標時，必須在帳戶層級啟用活動指標。	400 錯誤的請求	不支援
MissingBucketLevelActivityMetrics	在帳戶層級啟用活動指標時，必須在儲存貯體層級啟用活動指標。	400 錯誤的請求	不支援
MissingEncryptionMethod	加密方法不能為空白。指定 SSE-KMS 或 SSE-S3。	400 錯誤的請求	不支援
MissingPrefixLevelStorageMetrics	啟用字首層級時，字首層級的儲存指標是強制性的。	400 錯誤的請求	不支援
OrganizationAccessDenied	此帳戶未獲授權新增 AWS Organizations。	403 禁止	不支援
OrgConfigurationNotSupported	指定的區域不支援 AWS Organizations 組態。	403 禁止	不支援

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
ServiceNotEnabledForOrg	組織未啟用 S3 Storage Lens 服務連結角色。	403 禁止	不支援
StorageMetricsMustEnabled	必須啟用字首層級儲存指標。	400 錯誤的請求	不支援
TooManyBuckets	儲存貯體容器不能有超過 50 個儲存貯體。	400 錯誤的請求	不支援
TooManyRegions	區域容器不能有超過 50 個區域。	400 錯誤的請求	不支援
TooManyStorageLensGroups	您無法將超過 50 個 Storage Lens 群組連接至 Storage Lens 儀表板。	400 Bad Request	不支援

下表包含 S3 Storage Lens 群組操作可能傳回的特殊錯誤。如需一般 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
AccessDenied	您沒有執行 Storage Lens 群組動作的許可。不支援此區域做為 S3 Storage Lens 群組的主區域。	403 Forbidden	不支援
ConfigurationAlreadyExists	指定的組態已存在。	409 Conflict	不支援
DuplicateElement	標籤必須是唯一的。And 邏輯運算子包含重複的標籤索引鍵。Or 邏輯運算子包含重複的標籤。邏輯運算子包含重複的字首或尾碼。	400 Bad Request	不支援

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidAge	DaysLessThan 和 DaysGreaterThan 必須是正數。	400 Bad Request	不支援
InvalidFilter	篩選條件必須包含下列其中一個元素：And、Or、MatchAnyTag、MatchAnyPrefix、MatchAnySuffix、MatchObjectAge、MatchObjectSize。	400 Bad Request	不支援
InvalidLogicalOperator	邏輯運算子And或Or中必須至少有兩個子元素Or。	400 Bad Request	不支援
InvalidMatchAnyPrefix	MatchAnyPrefix 參數不可為空白。	400 Bad Request	不支援
InvalidMatchAnySuffix	MatchAnySuffix 參數不可為空白。	400 Bad Request	不支援
InvalidMatchAnyTag	MatchAnyTag 參數不可為空白。	400 Bad Request	不支援
InvalidMatchObjectAge	MatchObjectAge 參數不可為空白。	400 Bad Request	不支援
InvalidMatchObjectSize	MatchObjectSize 參數不可為空白。	400 Bad Request	不支援
InvalidName	Storage Lens 群組名稱參數必須介於 1 到 64 個字元之間。Storage Lens 群組Name參數必須使用^[a-zA-Z0-9\-_]+\$ 模式。	400 Bad Request	不支援
InvalidNumericCombination	此物件存留期或物件大小組合無效。	400 Bad Request	不支援

錯誤碼	說明	HTTP 狀態碼	SOAP 故障碼字首
InvalidPrefix	字首的長度上限為 1,024 個字元。字首字串不可空白。	400 Bad Request	不支援
InvalidSize	BytesLessThan 和 BytesGreaterThan 必須是正數。物件大小上限不能超過 50 TB。物件大小下限不能大於或等於 50 TB。	400 Bad Request	不支援
InvalidSuffix	尾碼的長度上限為 1,024 個字元。尾碼字串不可空白。	400 Bad Request	不支援
InvalidTag	物件標籤索引鍵不可超過 128 個字元。物件標籤金鑰字串不能為 null 或空白。標籤值的長度上限為 256 個字元。物件標籤索引鍵包含無效的字元。物件標籤金鑰只能包含 a-z、A-Z、0-9、空格和下列字元：^(_.: / = + \ - @] *)\$ 。	400 Bad Request	不支援
MismatchedName	請求中指定的名稱與 Storage Lens 群組名稱不相符。	400 Bad Request	不支援
TooManyConfigurations	您已嘗試建立比允許的 50 更多的 Storage Lens 群組組態。	400 Bad Request	不支援
TooManyElements	Element 超過邏輯運算子中允許的元素數目上限。僅允許 10 個字首、尾碼或標籤。	400 Bad Request	不支援

Amazon S3 Object Lambda 錯誤代碼清單

下表包含 S3 Object Lambda 可能傳回的特殊錯誤。如需有關一般 Amazon S3 錯誤和錯誤代碼清單的資訊，請參閱 [錯誤回應](#)。

在非GetObject 請求期間從支援存取點收到的錯誤回應，會無變更地傳送給發起人。

錯誤碼	說明	HTTP 狀態碼
LambdaInvalidResponse	當 WriteGetObjectResponse 回應時，傳回原始呼叫者 ValidationError AWS Lambda。 如需詳細資訊，請參閱 ValidationError 訊息。並非所有案例都 ValidationError 會導致 LambdaInvalidResponse 錯誤。	400 Bad Request
LambdaInvocationFailed	Lambda 函數調用失敗。 當 S3 Object Lambda 無法成功調用設定的 Lambda 函數時，發起者可能會收到下列錯誤。 錯誤訊息可能包含呼叫函數時，AWS Lambda 服務所傳回之最終錯誤的詳細資訊（例如，狀態碼、錯誤碼、錯誤訊息和請求 ID）。	400 Bad Request
LambdaNotFound	找不到 AWS Lambda 函數。 嘗試調用 Lambda 函數時，找不到已設定的 Lambda 函數、版本或別名。確保 S3 Object Lambda 存取點組態指向正確的 Lambda 函數 ARN。 錯誤訊息可能包含呼叫函數時，AWS Lambda 服務所傳回之最終錯誤的詳細資訊（例如，狀態碼、錯誤碼、錯誤訊息和請求 ID）。	404 Not Found
LambdaPermissionError	發起者無權調用 Lambda 函數。	403 Forbidden

錯誤碼	說明	HTTP 狀態碼
	發起者必須具有調用 Lambda 函數的許可。檢查連接到發起人的政策，並確保其已獲准將 <code>lambda:Invoke</code> 用於已設定的函數。 錯誤訊息可能包含呼叫函數時，AWS Lambda 服務所傳回之最終錯誤的詳細資訊（例如，狀態碼、錯誤碼、錯誤訊息和請求 ID）。	
LambdaResponseNotReceived	Lambda 函數已結束，但未成功呼叫 <code>WriteGetObjectResponse</code>。 <code>GetObject</code> 回應資料是由 Lambda 函數透過呼叫 <code>WriteGetObjectResponse</code> API 操作提供。函數的 Amazon CloudWatch logs 可能會更深入地了解在正常退出時，函數為何無法成功呼叫此 API 操作。	500 Internal Service Error
LambdaRuntimeError	Lambda 函數在執行期間失敗。 從 Lambda 函數收到明確錯誤。如需失敗的詳細資訊，請檢查 AWS CloudFormation 日誌。	500 Internal Service Error
LambdaTimeout	Lambda 函數未在允許時間內回應。 Lambda 函數無法在 60 秒 <code>WriteGetObjectResponse</code> 內完成對的呼叫。	500 Internal Service Error

錯誤碼	說明	HTTP 狀態碼
SlowDown	降低涉及 之操作的請求率 AWS Lambda。 函數呼叫是由 調節 AWS Lambda，可能是因為它已達到其設定的並行限制。如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的管理 Lambda 函數的並行。 錯誤訊息可能包含呼叫函數時，AWS Lambda 服務所傳回之最終錯誤的詳細資訊（例如，狀態碼、錯誤碼、錯誤訊息和請求 ID）。	503 Slow Down
ValidationError	驗證錯誤可能會從 WriteGetObjectResponse API 操作傳回，並可能因多種原因而發生。如需詳細資訊，請參閱錯誤訊息。	400 Bad Request

Amazon S3 非同步錯誤代碼清單

下表包含非同步請求可能會傳回的特殊錯誤。如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

當您查詢非同步請求的狀態時，例如使用 `DescribeMultiRegionAccessPointOperation`，會傳回這些錯誤。由於這些請求是非同步的，因此所有這些錯誤的狀態碼皆為 200 OK。

錯誤碼	說明	HTTP 狀態碼
AccessDenied	存取遭拒。	200 OK
InternalErrors	發生內部伺服器錯誤。	200 OK
MalformedPolicy	指定的政策語法無效。	200 OK

錯誤碼	說明	HTTP 狀態碼
MultiRegionAccessPointAlreadyOwnedByYou	您已擁有同名的多區域存取點。	200 OK
MultiRegionAccessPointModifiedByAnotherRequest	動作失敗，因為另一個請求正在修改指定的資源。嘗試在先前的請求完成後重新提交您的請求。	200 OK
MultiRegionAccessPointNotReady	指定的多區域存取點尚未準備好更新。	200 OK
MultiRegionAccessPointSameBucketRegion	用於建立多區域存取點的儲存貯體不能位於相同區域。	200 OK
MultiRegionAccessPointUnsupportedRegion	提供用來建立多區域存取點的其中一個儲存貯體位於不支援的區域。	200 OK
NoSuchBucket	指定的儲存貯體不存在。	200 OK
NoSuchMultiRegionAccessPoint	指定的多區域存取點不存在。	200 OK

Amazon S3 Access Grants 錯誤代碼清單

下表包含 S3 Access Grants 請求可能傳回的特殊錯誤。如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

錯誤程式碼	說明	HTTP 狀態碼
AccessGrantAlreadyExists	指定的存取授權已存在	409
AccessGrantsInstanceAlreadyExists	Access Grants 執行個體已存在	409

錯誤程式碼	說明	HTTP 狀態碼
AccessGrantsInstanceNotEmptyError	刪除存取授權執行個體之前，請先清除位置	400
AccessGrantsInstanceNotExistsError	Access Grants 執行個體不存在	404
AccessGrantsInstanceResourcePolicyNotExists	Access Grants 執行個體資源政策不存在	404
AccessGrantsLocationAlreadyExistsError	指定的存取授權位置已存在	409
AccessGrantsLocationNotEmptyError	刪除存取權位置之前，請先清除存取權	400
AccessGrantsLocationsQuotaExceededError	已超過存取授權位置配額。Access Grants Locations Quota : <i><value></i> 。如果需要增加，請聯絡 S3。	409
AccessGrantsQuotaExceededError	已超過存取授與配額。Access Grants Quota : <i><value></i> 。如果需要增加，請聯絡 S3。	409
InvalidTag	您的請求中有重複的標籤索引鍵。移除重複的標籤索引鍵，然後再試一次。	400
InvalidAccessGrant	指定的存取授權無效	400
InvalidAccessGrantLocation	指定的存取授權位置無效	400
InvalidIamRole	指定的 IAM 角色無效	400

錯誤程式碼	說明	HTTP 狀態碼
InvalidIdentityCenterInstance	指定的身分中心執行個體無效	400
InvalidResourcePolicy	指定的資源政策無效	400
InvalidResourcePolicy	指定的資源政策無效	400
InvalidTag	此請求包含無效的標籤索引鍵或值。有效字元包括下列項目：【a-zA-Z+-._: /】。標籤索引鍵最多可包含 128 個字元。標籤值最多可包含 256 個字元。	400
NoSuchAccessGrantError	指定的存取授權不存在	404
NoSuchAccessGrantsLocationError	指定的存取授權位置不存在	404
AccessDenied	您沒有請求 S3 字首的 <i><requested permission></i> 許可： <i><requested target></i>	403 Forbidden
StsNotAuthorizedError	呼叫 GetDataAccess 操作時發生錯誤 (StsNotAuthorizedError)：使用者：access-grants.s3.amazonaws.com 無權在資源sts:AssumeRole 上執行： <i><IAM Role ARN></i>	403
StsPackedPolicyTooLargeError	呼叫 GetDataAccess 操作時發生錯誤 (StsPackedPolicyTooLargeError)：序列化字符對於工作階段太大	400
StsValidationError	錯誤訊息會根據驗證錯誤而有所不同。	400

錯誤程式碼	說明	HTTP 狀態碼
InvalidTags	標籤索引鍵不能以系統標籤的 AWS 預留字首開頭。」	400
TooManyTags	標籤數量超過 50 個標籤的限制。請移除一些標籤，然後再試一次。	400

Amazon FSx 相關錯誤代碼的清單

下表包含透過 Amazon S3 存取點向 Amazon FSx 發出的請求可能會傳回的特殊錯誤。如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

錯誤程式碼	說明	HTTP 狀態碼
AccessDenied	此錯誤可能發生的原因如下： • 呼叫者無權執行指定的 S3 操作。 • 呼叫者無權存取指定金鑰的 Amazon FSx 資源。	403 Forbidden
InsufficientCapacity	已達到檔案系統的儲存容量上限。	507 Insufficient Capacity
InvalidKey	指定的金鑰無效。	400 Bad Request
SlowDown	請降低您的請求率。	503 Slow Down

Amazon S3 Tables 錯誤代碼清單

下表包含 Amazon S3 Tables 操作可能會傳回的特殊錯誤。如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

錯誤程式碼	說明	HTTP 狀態碼
AccessDeniedException	無法執行動作，因為您沒有必要的許可。	403 Forbidden
BadRequestException	請求無效或格式不正確。	400 Bad Request
ConflictException	請求失敗，因為與先前的寫入發生衝突。 您可重試請求。	409 Conflict
ForbiddenException	發起人無權提出請求。	403 Forbidden
InternalServerErrorException	由於內部伺服器錯誤，請求失敗。	500 Internal Server Error
NotFoundException	請求被拒絕，因為找不到指定的資源。	404 Not Found
TooManyRequestsException	已超過每秒請求數的限制。	429 Too Many Requests

Amazon S3 錯誤的最佳實務

許多錯誤回應包含額外的結構化資料，負責診斷程式設計錯誤的開發人員應詳加閱讀及了解。例如，若您傳送之 REST PUT 要求中所包含的 Content-MD5 標頭，不符合伺服器計算所得的摘要，您就會收到 BadDigest 錯誤。錯誤回應也包含 作為計算摘要的詳細元素，以及您提供的預期摘要。您可以在開發時使用此資訊診斷錯誤。在生產環境中，一支運作良好的程式可能將此資訊包含在其錯誤日誌中。

當您設計可與 Amazon S3 搭配使用的應用程式時，正確處理 Amazon S3 錯誤十分重要。本節說明設計應用程式時須考量的問題。

重試 InternalError

內部錯誤是指 Amazon S3 環境內發生的錯誤。

收到 InternalError 回應的要求可能尚未處理。例如，若 PUT 要求傳回 InternalError，後續的 GET 可能會擷取舊值，也可能會擷取更新後的值。

若 Amazon S3 傳回 InternalError 回應，請重試該請求。

針對重複出現的 SlowDown 錯誤微調應用程式

一如所有的分散式系統，S3 也設有保護機制，可以偵測有意或無意的資源過度使用，同時採取相應的回應。當要求率觸發其中一項機制時，即會產生 SlowDown 錯誤。降低要求率即可減少或消除此類型的錯誤。一般而言，大多數使用者不會定期遇到這些錯誤；不過，如果您想要更多資訊，或遇到高或非預期的 SlowDown 錯誤，請張貼到我們的 [Amazon S3 開發人員論壇](https://aws.amazon.com/premiumsupport/) 或註冊 支援 <https://aws.amazon.com/premiumsupport/>。

隔離錯誤

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

Amazon S3 提供一組錯誤代碼供 SOAP 與 REST API 使用。SOAP API 會傳回標準的 Amazon S3 錯誤代碼。REST API 的設計類似標準的 HTTP 伺服器，會與現有的 HTTP 用戶端 (例如瀏覽器、HTTP 用戶端程式庫、代理伺服器、快取等等) 互動。為了確保 HTTP 用戶端正確處理錯誤，Amazon S3 會將每個 Amazon S3 錯誤對應至 HTTP 狀態碼。

HTTP 狀態碼不如 Amazon S3 錯誤碼易懂，而且包含的錯誤資訊也較少。例如 NoSuchKey 及 NoSuchBucket 兩個 Amazon S3 錯誤都可對應到 HTTP 404 Not Found 狀態碼。

雖然 HTTP 狀態碼包含的錯誤資訊較少，但了解 HTTP 卻不了解 Amazon S3 API 的用戶端，通常都能正確處理錯誤。

因此在處理錯誤或將 Amazon S3 錯誤回報給使用者時，會使用 Amazon S3 錯誤代碼，而不會使用 HTTP 狀態碼，因為前者包含了大部分的錯誤資訊。此外，在對應用程式執行除錯時，您也應該參考 XML 錯誤回應之 <Details> 元素中可供人閱讀的內容。

AWS 詞彙表

如需最新的 AWS 術語，請參閱 AWS 詞彙表 參考中的[AWS 詞彙表](#)。

Amazon S3 資源

下表列出使用此服務時的相關實用資源。

資源	說明
Amazon Simple Storage Service 使用者指南	入門指南根據簡單的使用案例提供服務的快速教學。
Amazon Simple Storage Service 使用者指南	開發人員指南說明如何使用 Amazon S3 操作完成任務。
Amazon S3 技術常見問答集	常見問答集涵蓋開發人員針對此產品所詢問的前 20 個問題。
Amazon S3 版本備註	版本備註提供目前版本的高階概觀。它們會特別注意任何新的功能、更正與已知問題。
適用於 Amazon Web Services 的工具	尋找文件、程式碼範例、版本備註和其他資訊的中心起點，可協助您使用 AWS SDKs和工具建置創新應用程式。
AWS 管理主控台	主控台可讓您執行 Amazon S3 的大部分功能，無需程式設計。
開發論壇	以社群為基礎的論壇，供開發人員討論與 Amazon Web Services 相關的技術問題。
支援 中心	AWS 技術支援的首頁，包括存取我們的開發人員論壇、技術FAQs、服務狀態頁面和進階支援。
支援	有關one-on-one快速回應支援管道的資訊的主要網頁 支援，可協助您在 AWS Infrastructure Services 上建置和執行應用程式。
Amazon S3 產品資訊	Amazon S3 相關資訊的主要網頁。
聯絡我們	詢問有關 AWS 帳單、帳戶、事件、濫用等問題的中央聯絡窗口。

資源	說明
使用條件	Amazon.com 與其他主題的著作權和商標使用詳細資訊。

附錄

主題

- [附錄：SelectObjectContent 回應](#)
- [附錄：OPONS 物件](#)
- [附錄：SOAP API](#)
- [附錄：驗證請求 \(AWS 簽章第 2 版\)](#)
- [附錄：生命週期組態 APIs \(已棄用\)](#)

附錄：SelectObjectContent 回應

說明

Amazon S3 Select 操作會根據簡單的結構化查詢語言 (SQL) 陳述式來篩選 Amazon S3 物件的內容。由於此操作的回應大小未知，Amazon S3 Select 會將回應串流為一系列訊息，並在回應中包含具有 **chunked** 作為其值的 Transfer-Encoding 標頭。

如需 Amazon S3 Select 的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[從物件選取內容](#)。

如需搭配 Amazon S3 Select 使用 SQL 的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [Amazon S3 Select 和 Amazon Glacier Select 的 SQL 參考](#)。

回應

成功的 Amazon S3 Select 操作會傳回 200 OK 狀態碼。

回應標頭

此操作的實作只會使用大多數回應通用的回應標頭。如需詳細資訊，請參閱[常見回應標頭](#)。

回應內文

由於 Amazon S3 Select 回應大小未知，Amazon S3 會將回應串流為一系列訊息，並在回應中包含具有 **chunked** 作為其值的 Transfer-Encoding 標頭。下列範例顯示最上層的回應格式：

```
<Message 1>
<Message 2>
<Message 3>
.....
<Message n>
```

每則訊息包含兩個部分：序言和資料。前導區段包含 1) 訊息的總位元組長度，以及 2) 所有標頭的合併位元組長度。資料區段包含 1) 標頭，以及 2) 承載。

每個區段以 4 位元組大端整數檢查總和 (CRC) 結尾。Amazon S3 Select 使用 CRC32 (通常稱為 GZIP CRC32) 來計算這兩個 CRCs。如需 CRC32 的詳細資訊，請參閱 [GZIP 檔案格式規格 4.3 版](#)。

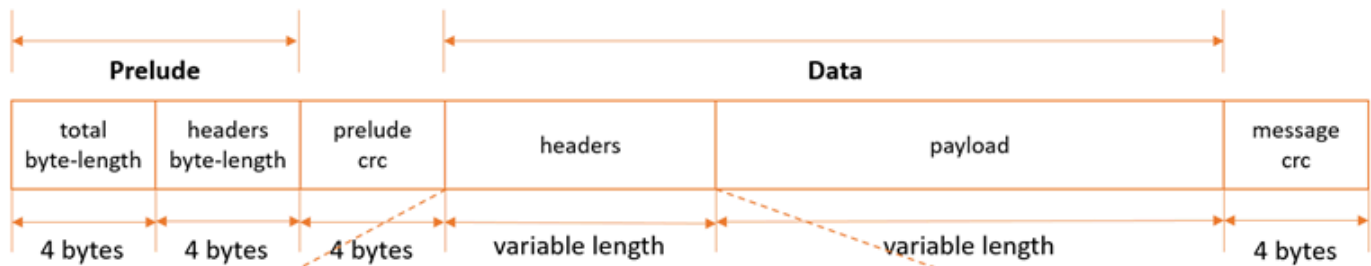
包括前導和兩個檢查總和在內的總訊息負荷為 16 個位元組。

Note

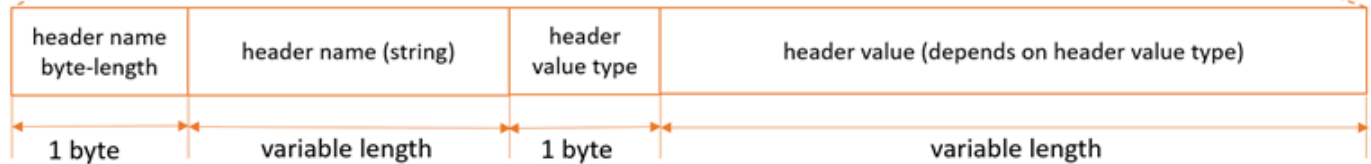
訊息內的所有integer值都採用網路位元組順序或大端順序。

下圖顯示構成訊息和標題的元件。請注意，每則訊息有多個標頭。

Message:



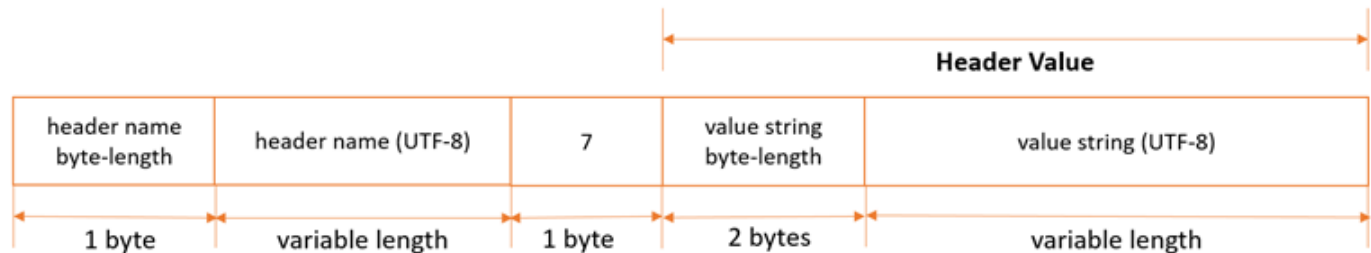
Headers (multiple headers per message):



Note

對於 Amazon S3 Select，標頭值類型一律為 7 (type=String)。對於此類型，標頭值包含兩個元件：2 位元組大端整數長度，以及該位元組長度的 UTF-8 字串。下圖顯示組成 Amazon S3 Select 標頭的元件。

Amazon S3 Select Headers (type=String):



承載位元組長度計算（這兩個計算相等）：

- $\text{payload\_length} = \text{total\_length} - \text{header\_length} - \text{sizeof}(\text{total\_length}) - \text{sizeof}(\text{header\_length}) - \text{sizeof}(\text{prelude\_crc}) - \text{sizeof}(\text{message\_crc})$
- $\text{payload\_length} = \text{total\_length} - \text{header\_length} - 16$

每則訊息都包含下列元件：

- Prelude：一律固定大小為 8 個位元組（兩個欄位各 4 個位元組）：
 - 前四個位元組：總位元組長度：整個訊息的大端整數位元組長度（包括 4 位元組總長度欄位本身）。
 - 第二四個位元組：標頭位元組長度：訊息標頭部分的大端整數位元組長度（標頭長度欄位本身除外）。
- Prelude CRC：訊息的 prelude 部分 (CRC 本身除外) 的 4 位元組大端整數檢查總和 (CRC)。前置詞具有與訊息 CRC 不同的 CRC（請參閱下文），以確保可立即偵測到損毀的位元組長度資訊，而不會導致路徑緩衝行為。
- 標頭：一組註釋訊息的中繼資料，例如訊息類型、承載格式等。訊息可以有多個標頭，因此訊息的這個部分可以有不同的位元組長度，具體取決於訊息類型。標頭是索引鍵/值對，其中索引鍵和值都是 UTF-8 字串。標頭可以在訊息的標頭部分中以任何順序顯示，而且任何指定的標頭類型只能顯示一次。

對於 Amazon S3 Select，根據訊息類型，以下是標頭名稱和有效值的清單。

- MessageType 標頭：
 - HeaderName => " : message-type"
 - 有效 HeaderValues => "error"、"event"
- EventType 標頭：
 - HeaderName => " : event-type"
 - 有效 HeaderValues => "Records"、"Cont"、"Progress"、"Stats"、"End"
- ErrorCode 標頭：
 - HeaderName => " : error-code"
 - 有效的 HeaderValues => [SELECT 物件內容錯誤代碼的清單](#)區段中資料表的錯誤碼。
- ErrorMessage 標頭：

回應 • HeaderName => " : error-message"

- 有效的 HeaderValues => 服務傳回的錯誤訊息，以協助診斷請求層級錯誤。
- 承載：可以是任何項目。
- 訊息 CRC：從訊息開始到檢查總和開始的 4 位元組大端整數檢查總和 (CRC) (即訊息中的所有項目，但不包括訊息 CRC 本身)。

每個標頭都包含下列元件。每則訊息可以有多个標頭。

- 標頭名稱位元組長度：標頭名稱的位元組長度。
- 標頭名稱：標頭的名稱，指出標頭類型。有效值：" : message-type" " : event-type" " : error-code" " : error-message"
- 標頭值類型：列舉，指出標頭值類型。對於 Amazon S3 Select，一律為 7。
- 值字串位元組長度：(適用於 Amazon S3 Select) 標頭值字串的位元組長度。
- 標頭值字串：(適用於 Amazon S3 Select) 標頭字串的值。此欄位的有效值會根據標頭的類型而有所不同。如需每個標頭類型和訊息類型的有效值，請參閱以下各節。

對於 Amazon S3 Select，回應可以是下列類型的訊息：

- 記錄訊息：可包含單一記錄、部分記錄或多個記錄。根據結果的大小，回應可以包含一或多個這些訊息。
- 持續訊息：Amazon S3 會定期傳送此訊息，以保持 TCP 連線開啟。這些訊息會隨機出現在回應中。用戶端必須偵測訊息類型並相應地進行處理。
- 進度訊息：如果請求，Amazon S3 會定期傳送此訊息。它包含已開始但尚未完成之查詢進度的相關資訊。
- 統計資料訊息：Amazon S3 會在請求結束時傳送此訊息。它包含有關查詢的統計資料。
- 結束訊息：表示請求已完成，而且不會再傳送任何訊息。在用戶端收到End訊息之前，您不應該假設請求已完成。
- RequestLevelError 訊息：如果請求因任何原因失敗，Amazon S3 會傳送此訊息。它包含失敗的錯誤代碼和錯誤訊息。如果 Amazon S3 傳送訊息RequestLevelError，則不會傳送訊息End。

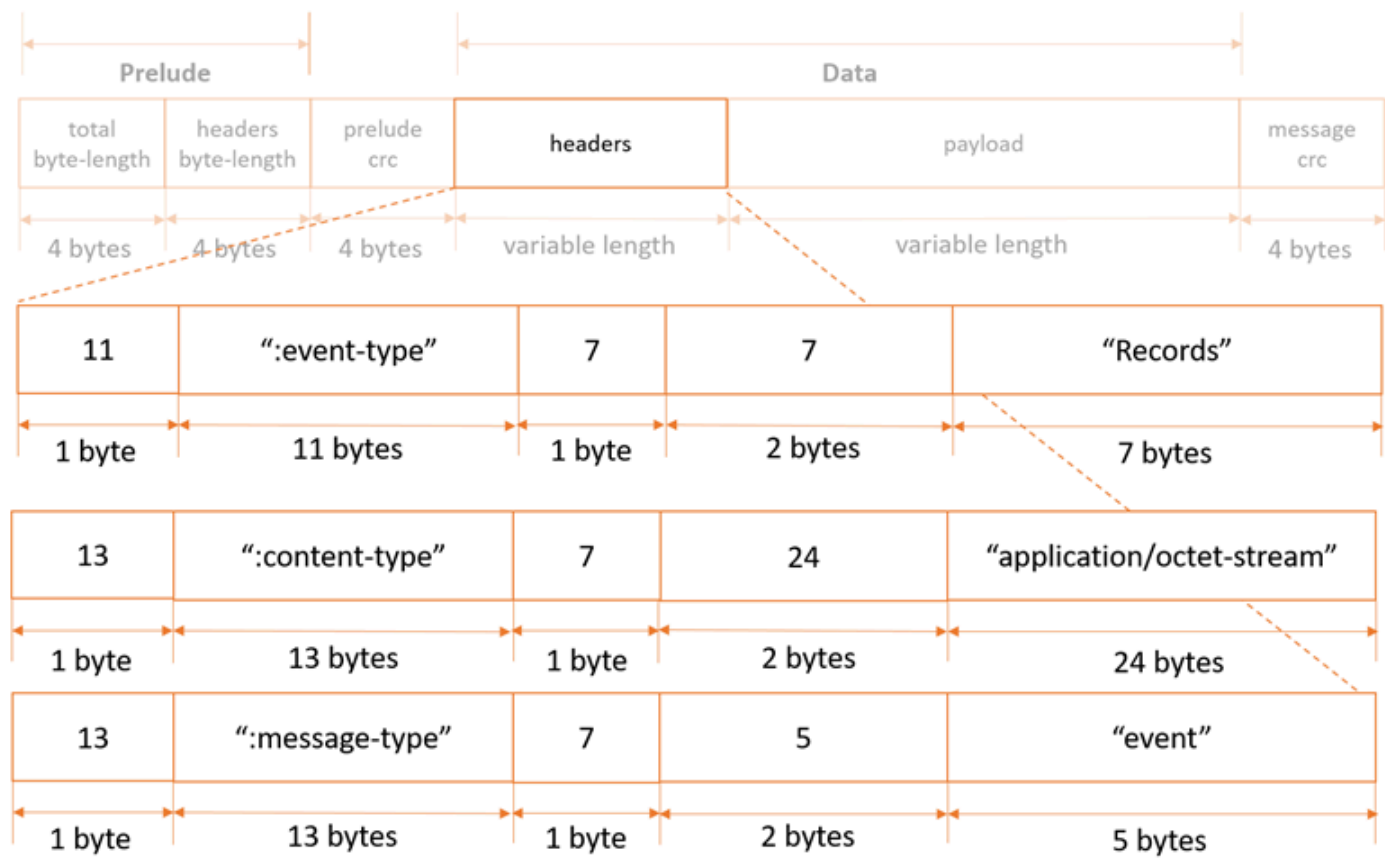
以下各節會更詳細地說明每個訊息類型的結構。

如需使用此通訊協定的範例程式碼和單元測試，請參閱 GitHub 網站上的 [AWS C 事件串流](#)。

記錄訊息

標頭規格

記錄訊息包含三個標頭，如下所示：



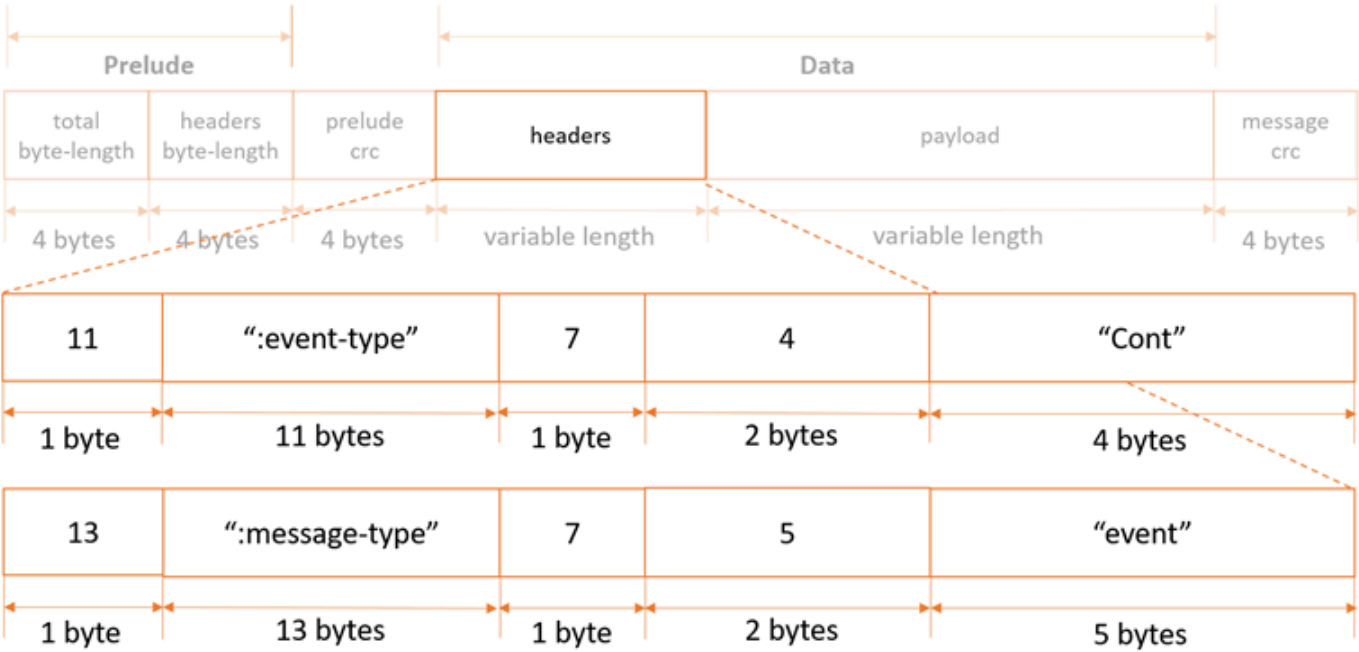
承載規格

記錄訊息承載可以包含單一記錄、部分記錄或多個記錄。

持續訊息

標頭規格

持續訊息包含兩個標頭，如下所示：



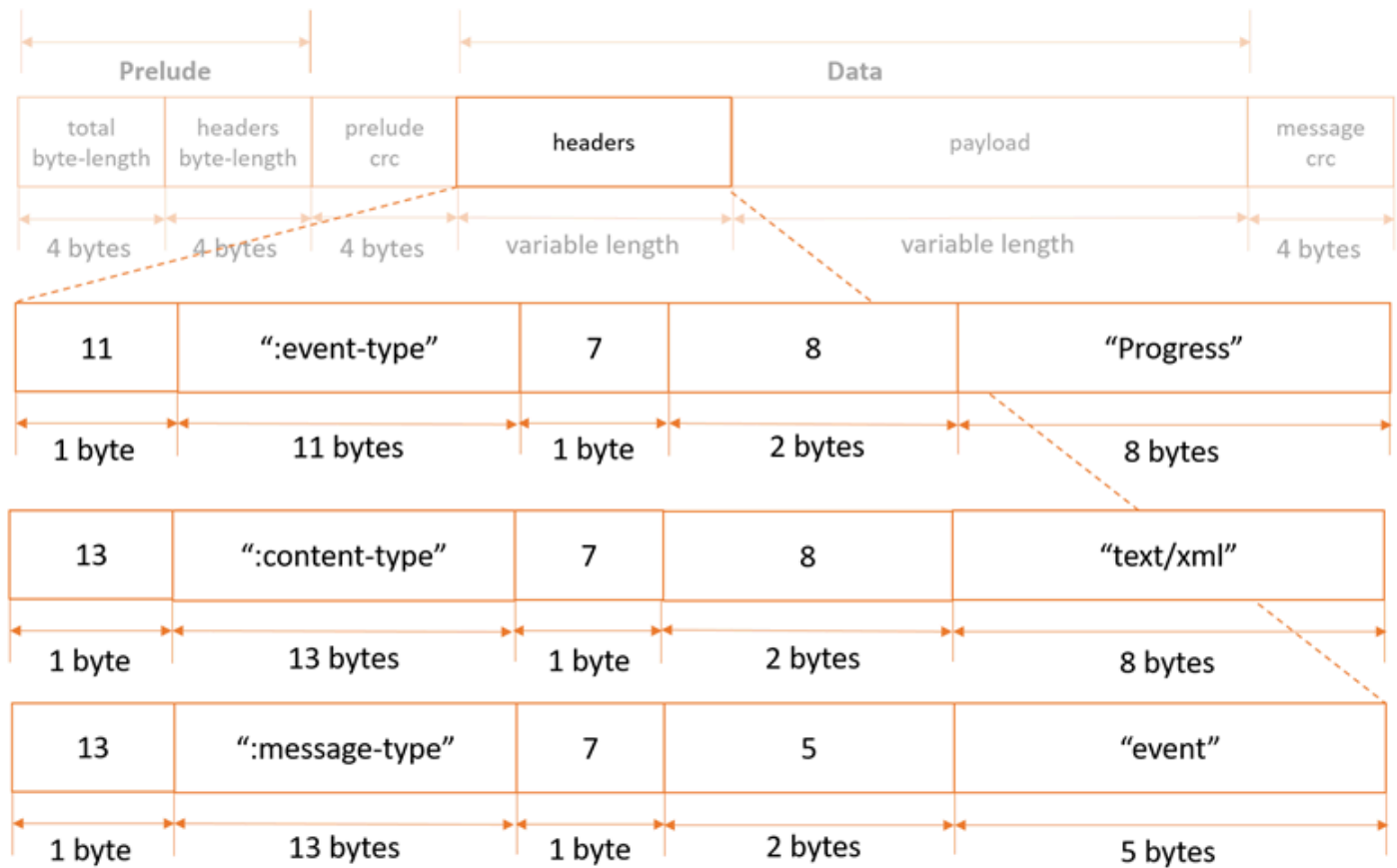
承載規格

持續訊息沒有承載。

進度訊息

標頭規格

進度訊息包含三個標頭，如下所示：



承載規格

進度訊息承載是 XML 文件，其中包含請求進度的相關資訊。

- BytesScanned => 在解壓縮之前已處理的位元組數（如果檔案已壓縮）。
- BytesProcessed => 解壓縮後已處理的位元組數（如果檔案已壓縮）。
- BytesReturned => Amazon S3 傳回的記錄承載資料目前位元組數。

對於未壓縮的檔案，BytesScanned和 BytesProcessed 相等。

範例：

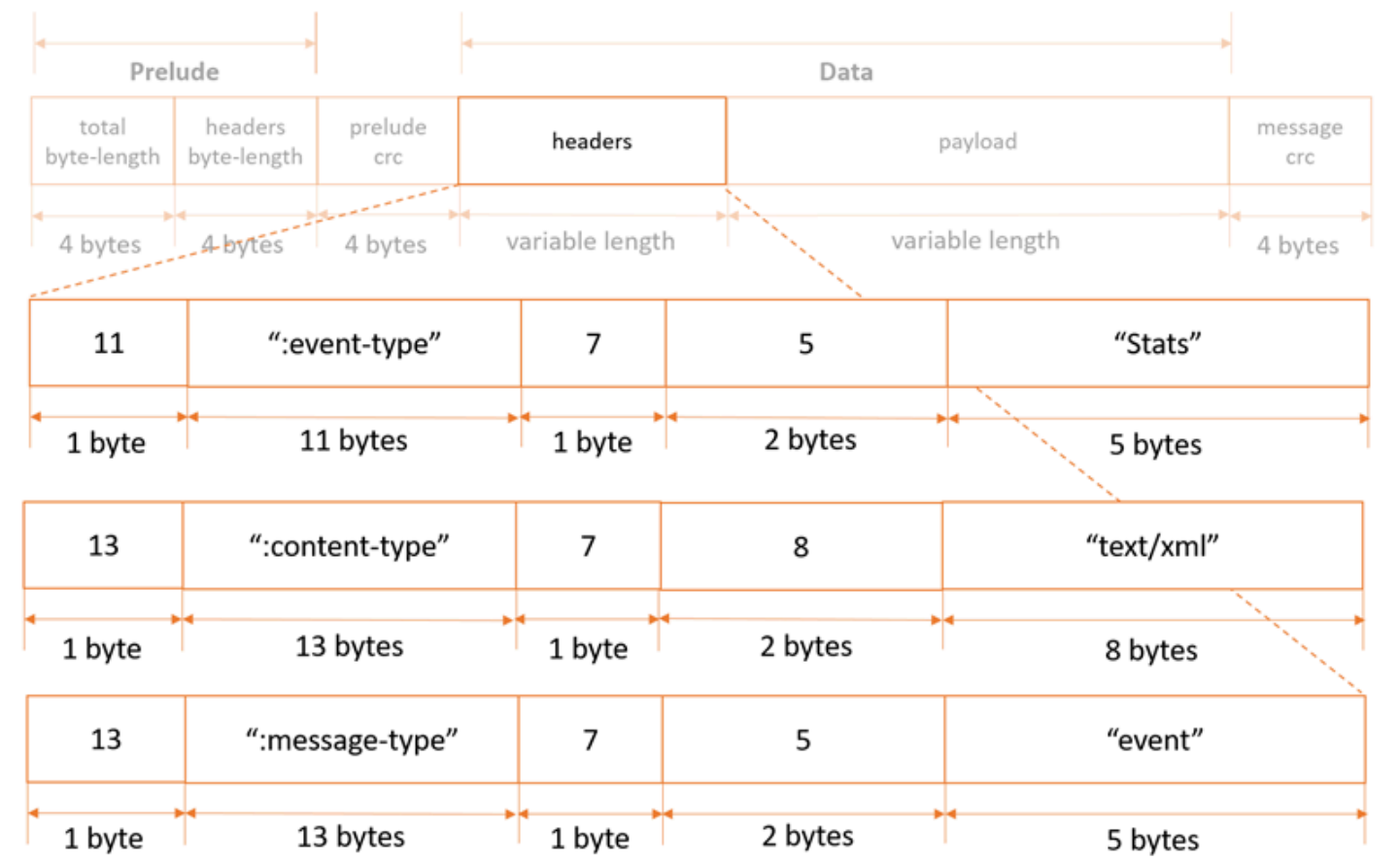
```
<?xml version="1.0" encoding="UTF-8"?>
<Progress>
  <BytesScanned>512</BytesScanned>
  <BytesProcessed>1024</BytesProcessed>
  <BytesReturned>1024</BytesReturned>
```

</Progress>

統計資料訊息

標頭規格

統計資料訊息包含三個標頭，如下所示：



承載規格

Stats 訊息承載是 XML 文件，其中包含處理完成時請求統計資料的相關資訊。

- BytesScanned => 在解壓縮之前已處理的位元組數（如果檔案已壓縮）。
- BytesProcessed => 解壓縮後已處理的位元組數（如果檔案已壓縮）。
- BytesReturned => Amazon S3 傳回的記錄承載資料總位元組數。

對於未壓縮的檔案，BytesScanned和 BytesProcessed 相等。

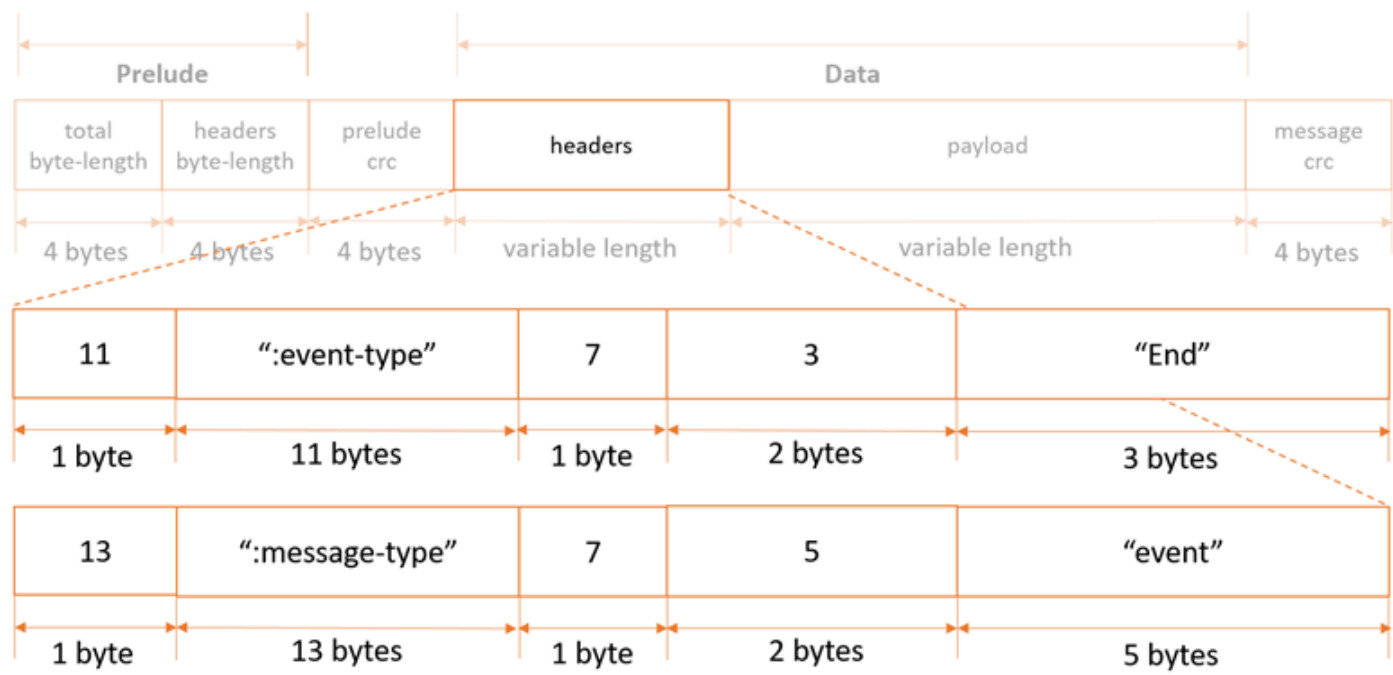
範例：

```
<?xml version="1.0" encoding="UTF-8"?>
<Stats>
  <BytesScanned>512</BytesScanned>
  <BytesProcessed>1024</BytesProcessed>
  <BytesReturned>1024</BytesReturned>
</Stats>
```

結束訊息

標頭規格

結束訊息包含兩個標頭，如下所示：



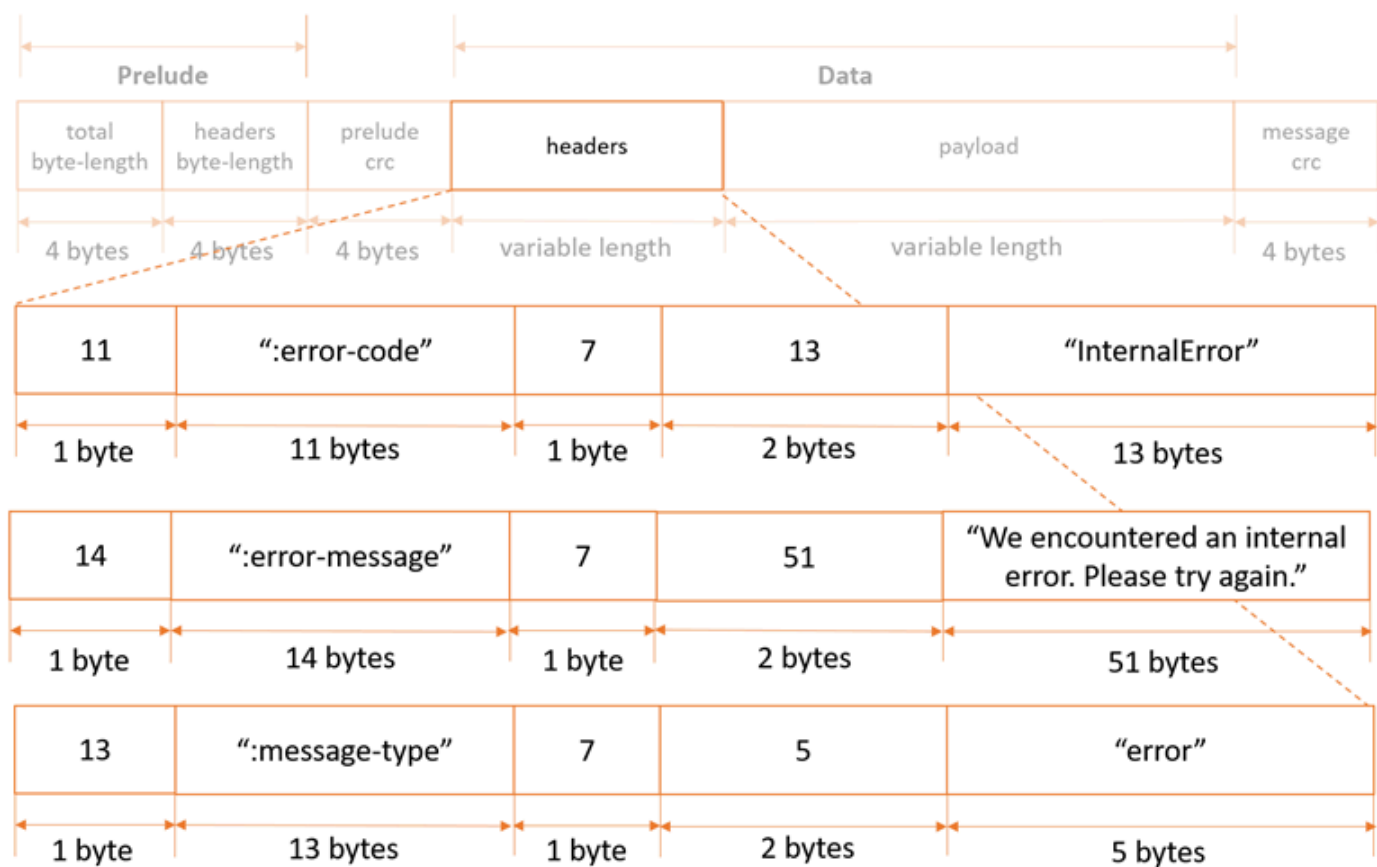
承載規格

結束訊息沒有承載。

請求層級錯誤訊息

標頭規格

請求層級錯誤訊息包含三個標頭，如下所示：



如需可能的錯誤代碼和錯誤訊息清單，請參閱 [SELECT 物件內容錯誤代碼的清單](#)。

承載規格

請求層級錯誤訊息沒有承載。

相關資源

- [SelectObjectContent](#)
- [GetObject](#)
- [GetBucketLifecycleConfiguration](#)
- [PutBucketLifecycleConfiguration](#)

附錄：OPONS 物件

說明

瀏覽器可以將此預檢請求傳送至 Amazon S3，以判斷是否可以傳送具有特定原始伺服器、HTTP 方法和標頭的實際請求。

Amazon S3 可讓您在儲存貯體上新增 `cors` 子資源，以支援跨來源資源共用 (CORS)。當瀏覽器傳送此預檢請求時，Amazon S3 會透過評估 `cors` 組態中定義的規則來回應。

如果在儲存貯體上未啟用 `cors`，則 Amazon S3 會傳回 403 Forbidden 回應。

如需 CORS 的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[啟用跨來源資源共用](#)。

要求

語法

```
OPTIONS /ObjectName HTTP/1.1
Host: BucketName.s3.amazonaws.com
Origin: Origin
Access-Control-Request-Method: HTTPMethod
Access-Control-Request-Headers: RequestHeader
```

請求參數

此操作不會引入任何特定請求參數，但可能包含實際請求所需的任何請求參數。

請求標頭

名稱	描述	必要
Origin	識別對 Amazon S3 的跨來源請求的來源。例如，https://http://www.example.com。 類型：字串	是

名稱	描述	必要
	預設：無	
Access-Control-Request-Method	識別要在實際請求中使用的 HTTP 方法。 類型：字串 預設：無	是
Access-Control-Request-Headers	將在實際請求中傳送的以逗號分隔的 HTTP 標頭清單。 例如，若要放置具有伺服器端加密的物件，此預檢請求會判斷是否可以在請求中包含 <code>x-amz-server-side-encryption</code> 標頭。 類型：字串 預設：無	否

要求元素

此 操作的實作不會使用請求元素。

回應

回應標頭

標頭	說明
Access-Control-Allow-Origin	您在請求中傳送的原始伺服器。如果不允許請求中的原始伺服器，Amazon S3 將不會在回應中包含此標頭。 類型：字串
Access-Control-Max-Age	預檢請求的結果可以快取多久，以秒為單位。

標頭	說明
	類型：字串
Access-Control-Allow-Methods	在原始請求中傳送的 HTTP 方法。如果不允許請求中的方法，Amazon S3 將不會在回應中包含此標頭。 類型：字串
Access-Control-Allow-Headers	瀏覽器可在實際請求中傳送的以逗號分隔的 HTTP 標頭清單。如果不允許任何請求的標頭，Amazon S3 將不會在回應中包含該標頭，也不會包含任何具有 Access-Control 字首的標頭。 類型：字串
Access-Control-Expose-Headers	HTTP 標頭的逗號分隔清單。此標頭可讓 JavaScript 用戶端在對實際請求的回應中存取這些標頭。 類型：字串

回應元素

此操作的實作不會傳回回應元素。

範例

範例：將預檢 OPTIONS 請求傳送至 **cors** 已啟用的儲存貯體

瀏覽器可以將此預檢請求傳送至 Amazon S3，以判斷是否可以将實際 PUT 請求從 `http://www.example.com` 原始伺服器傳送至名為 `examplebucket` 的 Amazon S3 儲存貯體。

請求範例

```
OPTIONS /exampleobject HTTP/1.1
Host: examplebucket.s3.amazonaws.com
Origin: http://www.example.com
Access-Control-Request-Method: PUT
```

回應範例

```
HTTP/1.1 200 OK
x-amz-id-2: 6SvaESv3VULYPLik5LL171SPPtSnBvDdGmnk1X1HfU17uS2m1DF6td6KWKNjYMXZ
x-amz-request-id: BDC4B83DF5096BBE
Date: Wed, 21 Aug 2012 23:09:55 GMT
Etag: "1f1a1af1f111111111111111c11aed1da1"
Access-Control-Allow-Origin: http://www.example.com
Access-Control-Allow-Methods: PUT
Access-Control-Expose-Headers: x-amz-request-id
Content-Length: 0
Server: AmazonS3
```

相關資源

- [GetBucketCors](#)
- [DeleteBucketCors](#)
- [PutBucketCors](#)

附錄：SOAP API

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

本節說明與服務、儲存貯體和物件操作相關的 SOAP API。請注意，必須使用 SSL 將經過身分驗證和匿名的 SOAP 請求傳送至 Amazon S3。當您透過 HTTP 傳送 SOAP 請求時，Amazon S3 會傳回錯誤。

最新的 Amazon S3 WSDL 可在 <https://docs.aws.amazon.com/2006-03-01/AmazonS3.wSDL> 取得。

主題

- [服務上的操作 \(SOAP API\)](#)
- [儲存貯體上的操作 \(SOAP API\)](#)
- [物件上的操作 \(SOAP API\)](#)

- [對 SOAP 要求進行身分驗證](#)
- [使用 SOAP 設定存取原則](#)
- [常見元素](#)
- [SOAP 錯誤回應](#)

服務上的操作 (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

本節說明您可以在 Amazon S3 服務上執行的操作。

主題

- [ListAllMyBuckets \(SOAP API\)](#)

ListAllMyBuckets (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

ListAllMyBuckets 操作會傳回請求寄件者擁有的所有儲存貯體清單。

Example

Sample Request

```
<ListAllMyBuckets xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
```

```
</ListAllMyBuckets>
```

Sample Response

```
<ListAllMyBucketsResult xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <Owner>
    <ID>bcaf1fffd86f41161ca5fb16fd081034f</ID>
    <DisplayName>webfile</DisplayName>
  </Owner>
  <Buckets>
    <Bucket>
      <Name>quotes</Name>
      <CreationDate>2006-02-03T16:45:09.000Z</CreationDate>
    </Bucket>
    <Bucket>
      <Name>samples</Name>
      <CreationDate>2006-02-03T16:41:58.000Z</CreationDate>
    </Bucket>
  </Buckets>
</ListAllMyBucketsResult>
```

回應內文

- Owner:

這提供 Amazon S3 用來代表您的身分的資訊，以用於身分驗證和存取控制。ID 是提出請求的開發人員唯一且永久的識別符。DisplayName 是人類可讀的名稱，代表提出請求的開發人員。它不是唯一的，而且可能會隨著時間而變更。我們建議您將 DisplayName 與論壇名稱配對。

- Name:

儲存貯體的名稱。請注意，如果您的其中一個儲存貯體最近遭到刪除，刪除的儲存貯體名稱可能仍存在此清單中一段時間。

- CreationDate:

建立儲存貯體的時間。

存取控制

您必須使用有效的 AWS 存取金鑰 ID 進行身分驗證。匿名請求絕不允許列出儲存貯體，而且您只能列出您為擁有者的儲存貯體。

儲存貯體上的操作 (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

本節說明您可以在 Amazon S3 儲存貯體上執行的操作。

主題

- [CreateBucket \(SOAP API\)](#)
- [DeleteBucket \(SOAP API\)](#)
- [ListBucket \(SOAP API\)](#)
- [GetBucketAccessControlPolicy \(SOAP API\)](#)
- [SetBucketAccessControlPolicy \(SOAP API\)](#)
- [GetBucketLoggingStatus \(SOAP API\)](#)
- [SetBucketLoggingStatus \(SOAP API\)](#)

CreateBucket (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

CreateBucket 操作會建立儲存貯體。不是每個字串都是可接受的儲存貯體名稱。如需儲存貯體命名限制的資訊，請參閱[使用 Amazon S3 儲存貯體](#)。

Note

若要判斷儲存貯體名稱是否存在，請使用 ListBucket 並 MaxKeys 設為 0。NoSuchBucket 回應表示儲存貯體可用，AccessDenied 回應表示其他人擁有儲存貯體，而 Success 回應表示您擁有儲存貯體或具有存取儲存貯體的許可。

Example 建立名為「引號」的儲存貯體

Sample Request

```
<CreateBucket xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</CreateBucket>
```

Sample Response

```
<CreateBucketResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <CreateBucketResponse>
    <Bucket>quotes</Bucket>
  </CreateBucketResponse>
</CreateBucketResponse>
```

元素

- **Bucket**：您嘗試建立的儲存貯體名稱。
- **AccessControlList**：新儲存貯體的存取控制清單。此元素是選用的。如果未提供，則會使用授予請求者 FULL_CONTROL 存取權的存取政策來建立儲存貯體。

存取控制

您必須使用有效的 AWS 存取金鑰 ID 進行身分驗證。不允許匿名請求建立儲存貯體。

相關資源

- [ListBucket \(SOAP API\)](#)

DeleteBucket (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

DeleteBucket 操作會刪除儲存貯體。必須先刪除儲存貯體中的所有物件，才能刪除儲存貯體本身。

Example

此範例會刪除「引號」儲存貯體。

Sample Request

```
<DeleteBucket xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <AWSAccessKeyId> AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE= </Signature>
</DeleteBucket>
```

Sample Response

```
<DeleteBucketResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <DeleteBucketResponse>
    <Code>204</Code>
    <Description>No Content</Description>
  </DeleteBucketResponse>
</DeleteBucketResponse>
```

元素

- **Bucket**：您要刪除的儲存貯體名稱。

存取控制

無論儲存貯體上的存取控制政策為何，都只允許儲存貯體擁有者將其刪除。

ListBucket (SOAP API)

 Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

ListBucket 操作會傳回儲存貯體中某些項目的相關資訊。

如需清單操作的一般簡介，請參閱[列出物件金鑰](#)。

要求

此範例列出「引號」儲存貯體中字首為「備註」的最多 1000 個金鑰。

語法

```
<ListBucket xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Prefix>notes</Prefix>
  <Delimiter>/</Delimiter>
  <MaxKeys>1000</MaxKeys>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</ListBucket>
```

參數

名稱	描述	必要
prefix	將回應限制為以指定字首開頭的金鑰。您可以使用字首，以類似於檔案系統使用資料夾的方式，將儲存貯體分成不同的金鑰集。 類型：字串 預設：無	否

名稱	描述	必要
marker	指出儲存貯體中要開始列出的位置。清單只會包含標記後以詞典方式出現的索引鍵。這對於分頁很方便：若要取得下一頁的結果，請使用目前頁面的最後一個索引鍵做為標記。 類型：字串 預設：無	否
max-keys	您想要在回應內文中看到的金鑰數目上限。伺服器可能會傳回少於此數量的金鑰，但不會傳回更多金鑰。 類型：字串 預設：無	否
delimiter	導致在字首與第一個出現的分隔符號之間包含相同字串的索引鍵彙總為 CommonPrefixes 集合中的單一結果元素。這些彙總金鑰不會在回應的其他地方傳回。 類型：字串 預設：無	否

成功回應

此回應假設儲存貯體包含下列金鑰：

```
notes/todos.txt
notes/2005-05-23/customer_mtg_notes.txt
notes/2005-05-23/phone_notes.txt
notes/2005-05-28/sales_notes.txt
```

語法

```
<?xml version="1.0" encoding="UTF-8"?>
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>backups</Name>
  <Prefix>notes</Prefix>
  <MaxKeys>1000</MaxKeys>
```

```
<Delimiter>/</Delimiter>
<IsTruncated>>false</IsTruncated>
<Contents>
  <Key>notes/todos.txt</Key>
  <LastModified>2006-01-01T12:00:00.000Z</LastModified>
  <ETag>"828ef3fdfa96f00ad9f27c383fc9ac7f"</ETag>
  <Size>5126</Size>
  <StorageClass>STANDARD</StorageClass>
  <Owner>
    <ID>75aa57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
    <DisplayName>webfile</DisplayName>
  </Owner>
  <StorageClass>STANDARD</StorageClass>
</Contents>
<CommonPrefixes>
  <Prefix>notes/2005-05-23/</Prefix>
</CommonPrefixes>
<CommonPrefixes>
  <Prefix>notes/2005-05-28/</Prefix>
</CommonPrefixes>
</ListBucketResult>
```

如您所見，回應中的許多欄位都會回應請求參數。IsTruncated、Contents和CommonPrefixes是唯一可包含新資訊的回應元素。

回應元素

名稱	描述
Contents	傳回之每個物件的中繼資料。 類型：XML 中繼資料 Ancestor：ListBucketResult
CommonPrefixes	CommonPrefixes 只有在您指定時，回應才能包含 delimiter。當您這樣做時，會CommonPrefixes 包含介於 Prefix和 之間的所有（如果有的話）金鑰，以及 所指定字串的下一次出現delimiter。實際上，CommonPrefixes 會列出 指定目錄中作用類似子目錄的索引鍵Prefix。例如，如果 prefix 是 notes/且 delimiter 是斜線 (/)notes/summer/july，則在 中，通用字首為 notes/summer/。

名稱	描述
Delimiter	類型：字串
	Ancestor：ListBucketResult
	導致在字首與第一個出現的分隔符號之間包含相同字串的索引鍵彙總為 CommonPrefixes 集合中的單一結果元素。這些彙總金鑰不會在回應的其他地方傳回。
IsTruncated	類型：字串
	Ancestor：ListBucketResult
	指定是否傳回 (true) 所有結果 (false)。如果結果數量超過 指定的數量，則可能不會傳回所有結果MaxKeys。
Marker	類型：字串
	Ancestor：布林值
	指出儲存貯體中要開始列出的位置。
MaxKeys	類型：字串
	Ancestor：ListBucketResult
	回應內文中傳回的金鑰數目上限。
Name	類型：字串
	Ancestor：ListBucketResult
	儲存貯體的名稱。

名稱	描述
Prefix	以指定字首開頭的金鑰。 類型：字串 Ancestor：ListBucketResult

回應內文

如需清單回應的資訊，請參閱[列出金鑰回應](#)。

存取控制

若要列出儲存貯體的金鑰，您需要獲得儲存貯體的READ存取權。

GetBucketAccessControlPolicy (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

GetBucketAccessControlPolicy 操作會擷取儲存貯體的存取控制政策。

Example

此範例會擷取「引號」儲存貯體的存取控制政策。

Sample Request

```
<GetBucketAccessControlPolicy xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</GetBucketAccessControlPolicy>
```

Sample Response

```
<AccessControlPolicy>
  <Owner>
    <ID>a9a7b886d6fd2441bf9b1c61be666e9</ID>
    <DisplayName>chriscustomer</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="CanonicalUser">
        <ID>a9a7b886d6fd41bf9b1c61be666e9</ID>
        <DisplayName>chriscustomer</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

回應內文

回應包含儲存貯體的存取控制政策。如需此回應的說明，請參閱 [SOAP 存取政策](#)。

存取控制

您必須擁有儲存貯體READ_ACP體的權限，才能擷取儲存貯體的存取控制政策。

SetBucketAccessControlPolicy (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

SetBucketAccessControlPolicy 操作會設定現有儲存貯體的存取控制政策。如果成功，儲存貯體的先前存取控制政策會完全取代為指定的存取控制政策。

Example

讓指定的使用者（通常是擁有者）FULL_CONTROL存取「引號」儲存貯體。

Sample Request

```
<SetBucketAccessControlPolicy xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="CanonicalUser">
        <ID>a9a7b8863000e241bf9b1c61be666e9</ID>
        <DisplayName>chriscustomer</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</SetBucketAccessControlPolicy >
```

Sample Response

```
<GetBucketAccessControlPolicyResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <GetBucketAccessControlPolicyResponse>
    <Code>200</Code>
    <Description>OK</Description>
  </GetBucketAccessControlPolicyResponse>
</GetBucketAccessControlPolicyResponse>
```

存取控制

您必須擁有儲存貯WRITE_ACP體的權限，才能設定儲存貯體的存取控制政策。

GetBucketLoggingStatus (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

會GetBucketLoggingStatus擷取現有儲存貯體的記錄狀態。

如需此功能的一般簡介，請參閱[伺服器日誌](#)。

Example

Sample Request

```
<?xml version="1.0" encoding="utf-8"?>
  <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://
www.w3.org/2001/XMLSchema">
    <soap:Body>
      <GetBucketLoggingStatus xmlns="https://doc.s3.amazonaws.com/2006-03-01">
        <Bucket>mybucket</Bucket>
        <AWSAccessKeyId>YOUR_AWS_ACCESS_KEY_ID</AWSAccessKeyId>
        <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
        <Signature>YOUR_SIGNATURE_HERE</Signature>
      </GetBucketLoggingStatus>
    </soap:Body>
  </soap:Envelope>
```

Sample Response

```
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" >
    <soapenv:Header>
    </soapenv:Header>
    <soapenv:Body>
      <GetBucketLoggingStatusResponse xmlns="https://s3.amazonaws.com/
doc/2006-03-01">
        <GetBucketLoggingStatusResponse>
          <LoggingEnabled>
            <TargetBucket>mylogs</TargetBucket>
            <TargetPrefix>mybucket-access_log-</TargetPrefix>
          </LoggingEnabled>
        </GetBucketLoggingStatusResponse>
      </GetBucketLoggingStatusResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

存取控制

只有儲存貯體的擁有者才能叫用此操作。

SetBucketLoggingStatus (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

SetBucketLoggingStatus 操作會更新現有儲存貯體的記錄狀態。

如需此功能的一般簡介，請參閱[伺服器日誌](#)。

Example

此範例請求會啟用 'mybucket' 儲存貯體的伺服器存取記錄，並將日誌設定為在字首 'access_log-' 下傳送到 'mylogs'

Sample Request

```
<?xml version="1.0" encoding="utf-8"?>
  <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://
www.w3.org/2001/XMLSchema">
    <soap:Body>
      <SetBucketLoggingStatus xmlns="https://doc.s3.amazonaws.com/2006-03-01">
        <Bucket>myBucket</Bucket>
        <AWSAccessKeyId>YOUR_AWS_ACCESS_KEY_ID</AWSAccessKeyId>
        <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
        <Signature>YOUR_SIGNATURE_HERE</Signature>
        <BucketLoggingStatus>
          <LoggingEnabled>
            <TargetBucket>mylogs</TargetBucket>
            <TargetPrefix>mybucket-access_log-</TargetPrefix>
          </LoggingEnabled>
        </BucketLoggingStatus>
      </SetBucketLoggingStatus>
    </soap:Body>
```

```
:</soap:Envelope>
```

Sample Response

```
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" >
    <soapenv:Header>
    </soapenv:Header>
    <soapenv:Body>
      <SetBucketLoggingStatusResponse xmlns="https://s3.amazonaws.com/
doc/2006-03-01"/>
    </soapenv:Body>
  </soapenv:Envelope>
```

存取控制

只有儲存貯體的擁有者才能叫用此操作。

物件上的操作 (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

本節說明您可以在 Amazon S3 物件上執行的操作。

主題

- [PutObjectInline \(SOAP API\)](#)
- [PutObject \(SOAP API\)](#)
- [CopyObject \(SOAP API\)](#)
- [GetObject \(SOAP API\)](#)
- [GetObjectExtended \(SOAP API\)](#)

- [DeleteObject \(SOAP API\)](#)
- [GetObjectAccessControlPolicy \(SOAP API\)](#)
- [SetObjectAccessControlPolicy \(SOAP API\)](#)

PutObjectInline (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

PutObjectInline 操作會將物件新增至儲存貯體。物件的資料會在 SOAP 訊息的內文中提供。

如果儲存貯體中已存在物件，則新物件會覆寫它，因為 Amazon S3 會儲存最後一個寫入請求。不過，Amazon S3 是分散式系統。如果 Amazon S3 幾乎同時收到相同物件的多個寫入請求，所有物件都可能儲存，即使最後只有一個獲勝。Amazon S3 不提供物件鎖定；如果您需要，請務必將其建置到您的應用程式層。

為了確保物件不會在網路上損毀，您可以計算物件的 MD5、將物件 PUT 轉換為 Amazon S3，並將傳回的 Etag 與計算的 MD5 值進行比較。

PutObjectInline 不適用於大型物件。系統會限制此操作使用 1MB 或更小的物件。如果資料參數編碼大於 1MB 的物件，PutObjectInline 將使用 `InlineDataTooLargeError` 狀態碼失敗。若要上傳大型物件，請考慮改用非內嵌 PutObject API 或 REST API。

Example

此範例會將一些文字和中繼資料寫入「引號」儲存貯體中的「Nelson」物件，讓使用者（通常是擁有者）FULL_CONTROL 存取物件，並讓匿名方讀取物件。

Sample Request

```
<PutObjectInline xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Key>Nelson</Key>
  <Metadata>
    <Name>Content-Type</Name>
    <Value>text/plain</Value>
  </Metadata>
```

```

<Metadata>
  <Name>family</Name>
  <Value>Muntz</Value>
</Metadata>
<Data>aGEtaGE=</Data>
<ContentLength>5</ContentLength>
<AccessControlList>
  <Grant>
    <Grantee xsi:type="CanonicalUser">
      <ID>a9a7b886d6fde241bf9b1c61be666e9</ID>
      <DisplayName>chriscustomer</DisplayName>
    </Grantee>
    <Permission>FULL_CONTROL</Permission>
  </Grant>
  <Grant>
    <Grantee xsi:type="Group">
      <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
    </Grantee>
    <Permission>READ</Permission>
  </Grant>
</AccessControlList>
<AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
<Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
<Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</PutObjectInline>

```

Sample Response

```

<PutObjectInlineResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <PutObjectInlineResponse>
    <ETag>"828ef3fdfa96f00ad9f27c383fc9ac7f"</ETag>
    <LastModified>2006-01-01T12:00:00.000Z</lastModified>
  </PutObjectInlineResponse>
</PutObjectInlineResponse>

```

元素

- **Bucket**：要新增物件的儲存貯體。
- **Key**：要指派給物件的金鑰。
- **Metadata**：您可以在中繼資料元素中提供名稱值中繼資料對。這些會與物件一起存放。

- **Data**: 資料的基礎 64 編碼形式。
- **ContentLength**: 資料的長度，以位元組為單位。
- **AccessControlList**: 資源的存取控制清單。此元素是選用的。如果省略，請求者會獲得物件的 FULL_CONTROL 存取權。如果物件已存在，則會取代預先存在的存取控制政策。

回應

- **ETag**: 實體標籤是物件的 MD5 雜湊，您可以使用對物件執行條件式擷取 GetObjectExtended。ETag 只會反映物件內容的變更，而不是其中繼資料。
- **LastModified**: 已儲存物件的 Amazon S3 時間戳記。

存取控制

您必須擁有儲存貯體的 WRITE 存取權，才能將物件放入儲存貯體。

相關資源

- [PutObject \(SOAP API\)](#)
- [CopyObject \(SOAP API\)](#)

PutObject (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

PutObject 操作會將物件新增至儲存貯體。物件的資料會附加為 DIME 連接。

為了確保物件不會在網路上損毀，您可以計算物件的 MD5、將物件 PUT 轉換為 Amazon S3，並將傳回的 Etag 與計算的 MD5 值進行比較。

如果儲存貯體中已存在物件，則新物件會覆寫它，因為 Amazon S3 會儲存最後一個寫入請求。不過，Amazon S3 是分散式系統。如果 Amazon S3 幾乎同時收到相同物件的多個寫入請求，所有物件

都可能儲存，即使最後只有一個獲勝。Amazon S3 不提供物件鎖定；如果您需要，請務必將其建置到您的應用程式層。

Example

此範例會將一些資料和中繼資料放入「引號」儲存貯體的「Nelson」物件中，讓使用者（通常是擁有者）FULL_CONTROL存取物件，並讓匿名方讀取物件。在此範例中，不會顯示實際附件。

Sample Request

```
<PutObject xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Key>Nelson</Key>
  <Metadata>
    <Name>Content-Type</Name>
    <Value>text/plain</Value>
  </Metadata>
  <Metadata>
    <Name>family</Name>
    <Value>Muntz</Value>
  </Metadata>
  <ContentLength>5</ContentLength>
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="CanonicalUser">
        <ID>a9a7b886d6241bf9b1c61be666e9</ID>
        <DisplayName>chriscustomer</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </AccessControlList>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2007-05-11T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</PutObject>
```

Sample Response

```
<PutObjectResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <PutObjectResponse>
    <ETag>"828ef3fd96f00ad9f27c383fc9ac7f"</ETag>
    <LastModified>2006-03-01T12:00:00.183Z</LastModified>
  </PutObjectResponse>
</PutObjectResponse>
```

元素

- **Bucket**: 要新增物件的儲存貯體。
- **Key**: 要指派給物件的金鑰。
- **Metadata**: 您可以在中繼資料元素中提供名稱值中繼資料對。這些會與物件一起存放。
- **ContentLength**: 資料的長度，以位元組為單位。
- **AccessControlList**: 資源的存取控制清單。此元素是選用的。如果省略，請求者會獲得物件的FULL_CONTROL存取權。如果物件已存在，則會取代預先存在的存取控制政策。

回應

- **ETag**: 實體標籤是物件的 MD5 雜湊，您可以使用對物件執行條件式擷取GetObjectExtended。ETag 只會反映物件內容的變更，而不是其中繼資料。
- **LastModified**: 已儲存物件的 Amazon S3 時間戳記。

存取控制

若要將物件放入儲存貯體，您必須擁有儲存貯體的WRITE存取權。

相關資源

- [CopyObject \(SOAP API\)](#)

CopyObject (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

說明

當您指定來源物件的金鑰和儲存貯體，以及目標目的地的金鑰和儲存貯體時，CopyObject操作會建立物件的複本。

複製物件時，您可以保留所有中繼資料（預設）或指定新的中繼資料。不過，ACL 不會保留，並且private會針對提出請求的使用者設定為。若要覆寫預設 ACL 設定，請在產生複製請求時指定新的 ACL。如需詳細資訊，請參閱[使用 ACL](#)。

必須驗證所有複製請求。此外，您必須擁有來源物件的讀取存取權，以及目的地儲存貯體的寫入存取權。如需詳細資訊，請參閱[使用身分驗證存取](#)。

若要僅在特定條件下複製物件，例如 Etag 是否與指定日期之前或之後修改物件，請使用請求參數 CopySourceIfUnmodifiedSince、CopyIfUnmodifiedSince、CopySourceIfMatch或 CopySourceIfNoneMatch。

Note

您可能需要設定 SOAP 堆疊通訊端逾時，以複製大型物件。

請求語法

```
<CopyObject xmlns="http://bucket_name.s3.amazonaws.com/2006-03-01">
  <SourceBucket>source_bucket</SourceBucket>
  <SourceObject>source_object</SourceObject>
  <DestinationBucket>destination_bucket</DestinationBucket>
  <DestinationObject>destination_object</DestinationObject>
  <MetadataDirective>{REPLACE | COPY}</MetadataDirective>
  <Metadata>
    <Name>metadata_name</Name>
    <Value>metadata_value</Value>
  </Metadata>
  ...
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="user_type">
        <ID>user_id</ID>
        <DisplayName>display_name</DisplayName>
      </Grantee>
      <Permission>permission</Permission>
    </Grant>
  </AccessControlList>
</CopyObject>
```

```

</Grant>
...
</AccessControlList>
<CopySourceIfMatch>etag</CopySourceIfMatch>
<CopySourceIfNoneMatch>etag</CopySourceIfNoneMatch>
<CopySourceIfModifiedSince>date_time</CopySourceIfModifiedSince>
<CopySourceIfUnmodifiedSince>date_time</CopySourceIfUnmodifiedSince>
<AWSAccessKeyId>AWSAccessKeyId</AWSAccessKeyId>
<Timestamp>TimeStamp</Timestamp>
<Signature>Signature</Signature>
</CopyObject>

```

請求參數

名稱	描述	必要
SourceBucket	來源儲存貯體的名稱。 類型：字串 預設：無 限制條件：有效的來源儲存貯體。	是
SourceKey	來源物件的金鑰名稱。 類型：字串 預設：無 限制：您具有 READ 存取權之有效來源物件的金鑰。	是
DestinationBucket	目的地 儲存貯體的名稱。 類型：字串 預設：無 限制條件：您必須擁有目的地儲存貯體的 WRITE 存取權。	是

名稱	描述	必要
DestinationKey	目的地物件的金鑰。 類型：字串 預設：無 限制條件：您必須擁有目的地儲存貯體的 WRITE 存取權。	是
MetadataDirective	指定中繼資料是從來源物件複製，還是取代為請求中提供的中繼資料。 類型：字串 預設：COPY 有效值：COPY REPLACE 限制條件：COPY 或 以外的值 REPLACE 將導致立即錯誤。除非已指定 MetadataDirective 標頭，且其值設為 ，否則您無法將物件複製到其本身 REPLACE。	否
Metadata	指定要為物件設定的中繼資料名稱/值對。如果 MetadataDirective 設定為 COPY，則會忽略所有中繼資料。 類型：字串 預設：無 限制條件：無。	否

名稱	描述	必要
AccessControlList	授予使用者電子郵件地址或正式使用者 ID 的存取權。 類型：字串 預設：無 限制條件：無	否
CopySourceIfMatch	如果物件的實體標籤 (ETag) 符合指定的標籤，則複製物件；否則傳回 PreconditionFailed。 類型：字串 預設：無 限制條件：無。如果 Etag 不相符，則不會複製物件。	否
CopySourceIfNoneMatch	如果物件的實體標籤 (ETag) 與指定的 Etag 不同，則複製物件；否則 會傳回錯誤。 類型：字串 預設：無 限制條件：無。	否
CopySourceIfUnmodifiedSince	如果物件自指定時間以來尚未修改，請複製物件；否則 會傳回 PreconditionFailed。 類型：dateTime 預設：無	否

名稱	描述	必要
CopySourceIfModifiedSince	如果物件自指定時間以來已經過修改，則複製物件；否則 會傳回錯誤。 類型：dateTime 預設：無	否

回應語法

```
<CopyObjectResponse xmlns="http://bucket_name.s3.amazonaws.com/2006-03-01">  
  <CopyObjectResponse>  
    <ETag>"etag"</ETag>  
    <LastModified>timestamp</LastModified>  
  </CopyObjectResponse>  
</CopyObjectResponse>
```

回應元素

以下是回應元素的清單。

Note

SOAP API 不會傳回額外的空格。只有 REST API 會傳回額外的空格。

名稱	描述
Etag	傳回新物件的 etag。ETag 只會反映物件內容的變更，而不是其中繼資料。 類型：字串 Ancestor：CopyObjectResult
LastModified	傳回物件上次修改的日期。 類型：字串

名稱	描述
	Ancestor : CopyObjectResult

如需一般回應元素的相關資訊，請參閱[使用 REST 錯誤回應標頭](#)。

特殊錯誤

此操作沒有特殊錯誤。如需一般 Amazon S3 錯誤的資訊，請參閱 [錯誤代碼清單](#)。

範例

此範例會將 flotsam 物件從 pacific 儲存貯體複製到 jetsam 儲存貯體的 atlantic 物件，並保留其中繼資料。

請求範例

```
<CopyObject xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <SourceBucket>pacific</SourceBucket>
  <SourceObject>flotsam</SourceObject>
  <DestinationBucket>atlantic</DestinationBucket>
  <DestinationObject>jetsam</DestinationObject>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2008-02-18T13:54:10.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbq7RrtSFmw=</Signature>
</CopyObject>
```

回應範例

```
<CopyObjectResponse xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <CopyObjectResponse>
    <ETag>"828ef3fdfa96f00ad9f27c383fc9ac7f"</ETag>
    <LastModified>2008-02-18T13:54:10.183Z</LastModified>
  </CopyObjectResponse>
</CopyObjectResponse>
```

此範例會將「tweedledee」物件從奇觀樂園儲存貯體複製到奇觀樂園儲存貯體的「tweedledum」物件，以取代其中繼資料。

請求範例

```
<CopyObject xmlns="http://doc.s3.amazonaws.com/2006-03-01">
```

```
<SourceBucket>wonderland</SourceBucket>
<SourceObject>tweedledee</SourceObject>
<DestinationBucket>wonderland</DestinationBucket>
<DestinationObject>tweedledum</DestinationObject>
<MetadataDirective>REPLACE</MetadataDirective>
<Metadata>
  <Name>Content-Type</Name>
  <Value>text/plain</Value>
</Metadata>
<Metadata>
  <Name>relationship</Name>
  <Value>twins</Value>
</Metadata>
<AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
<Timestamp>2008-02-18T13:54:10.183Z</Timestamp>
<Signature>Iuyz3d3P0aTou39dzbq7RrtSFmw=</Signature>
</CopyObject>
```

回應範例

```
<CopyObjectResponse xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <CopyObjectResponse>
    <ETag>"828ef3fdfa96f00ad9f27c383fc9ac7f"</ETag>
    <LastModified>2008-02-18T13:54:10.183Z</LastModified>
  </CopyObjectResponse>
</CopyObjectResponse>
```

相關資源

- [PutObject \(SOAP API\)](#)
- [PutObjectInline \(SOAP API\)](#)

GetObject (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

GetObject 操作會傳回物件的目前版本。如果您嘗試GetObject使用具有刪除標記作為其目前版本的物件，S3 會傳回 404 錯誤。您無法使用 SOAP API 擷取物件的指定版本。若要這樣做，請使用 REST API。如需詳細資訊，請參閱[版本控制](#)。如需更多選項，請使用 [GetObjectExtended \(SOAP API\)](#)操作。

Note

[虛擬託管樣式要求](#)不支援值為「soap」的物件金鑰名稱。對於使用「soap」的物件金鑰名稱值，必須改用[路徑樣式 URL](#)。

Example

此範例會從「引號」儲存貯體取得「Nelson」物件。

Sample Request

```
<GetObject xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Key>Nelson</Key>
  <GetMetadata>true</GetMetadata>
  <GetData>true</GetData>
  <InlineData>true</InlineData>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</GetObject>
```

Sample Response

```
<GetObjectResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <GetObjectResponse>
    <Status>
      <Code>200</Code>
      <Description>OK</Description>
    </Status>
    <Metadata>
      <Name>Content-Type</Name>
      <Value>text/plain</Value>
    </Metadata>
    <Metadata>
```



```
<Name>family</Name>
<Value>Muntz</Value>
</Metadata>
<Data>aGEtaGE=</Data>
<LastModified>2006-01-01T12:00:00.000Z</LastModified>
<ETag>&quot;828ef3fdfa96f00ad9f27c383fc9ac7f&quot;</ETag>
</GetObjectResponse>
</GetObjectResponse>
```

元素

- **Bucket**：要從中擷取物件的儲存貯體。
- **Key**：識別物件的金鑰。
- **GetMetadata**：如果這是 true，則會隨物件傳回中繼資料。
- **GetData**：如果這是 true，則會傳回物件資料。
- **InlineData**：如果這是 true，則會傳回以基底 64 編碼的資料，做為回應 SOAP 內文的一部分。如果為 false，則會將資料傳回為 SOAP 附件。InlineData 選項不適用於大型物件。系統會將此操作限制為使用 1MB 或更少的資料。如果產生的資料參數編碼超過 1MB，則具有 InlineData 旗標集的 GetObject 請求將使用 `InlineDataTooLargeError` 狀態碼失敗。若要下載大型物件，請考慮在未設定 InlineData 旗標的情況下呼叫 GetObject，或改用 REST API。

傳回的元素

- **Metadata**：與物件一起存放的名稱值配對中繼資料。
- **Data**：如果 InlineData 在請求中為 true，則這包含基本 64 編碼的物件資料。
- **LastModified**：物件存放在 Amazon S3 中的時間。
- **ETag**：物件的實體標籤。這是物件的雜湊，可用於執行條件式取得。ETag 只會反映物件內容的變更，而不是其中繼資料。

存取控制

只有在您獲得物件的 READ 存取權時，才能讀取物件。

SOAP 區塊和可繼續下載

為了提供 GET 彈性，Amazon S3 支援區塊和可繼續的下載。

從下列選取：

- 對於大型物件下載，您可能想要將它們分成較小的區塊。如需詳細資訊，請參閱[範圍 GETs](#)
- 對於失敗GET的操作，您可以設計應用程式來下載剩餘部分，而不是整個檔案。如需詳細資訊，請參閱[REST GET 錯誤復原](#)

範圍 GETs

對於某些用戶端，您可能想要將大型下載分成較小的下載。若要將 GET 分成較小的單位，請使用範圍。

您必須先判斷 GET 的大小，才能將 GET 分成較小的單位。例如，以下請求會取得 bigfile 物件的大小。

```
<ListBucket xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>bigbucket</Bucket>
  <Prefix>bigfile</Prefix>
  <MaxKeys>1</MaxKeys>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</ListBucket>
```

Amazon S3 傳回下列回應。

```
<ListBucketResult xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <Name>quotes</Name>
  <Prefix>N</Prefix>
  <MaxKeys>1</MaxKeys>
  <IsTruncated>>false</IsTruncated>
  <Contents>
    <Key>bigfile</Key>
    <LastModified>2006-01-01T12:00:00.000Z</LastModified>
    <ETag>"828ef3fdfa96f00ad9f27c383fc9ac7f"</ETag>
    <Size>2023276</Size>
    <StorageClass>STANDARD</StorageClass>
    <Owner>
      <ID>bcaf1ffd86f41161ca5fb16fd081034f</ID>
      <DisplayName>bigfile</DisplayName>
    </Owner>
  </Contents>
</ListBucketResult>
```

以下是從 Bigfile 物件下載第一個 MB 的請求。

```
<GetObject xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>bigbucket</Bucket>
  <Key>bigfile</Key>
  <GetMetadata>true</GetMetadata>
  <GetData>true</GetData>
  <InlineData>true</InlineData>
  <ByteRangeStart>0</ByteRangeStart>
  <ByteRangeEnd>1048576</ByteRangeEnd>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</GetObject>
```

Amazon S3 會傳回檔案的第一個 MB 和檔案的 Etag。

```
<GetObjectResponse xmlns="http://s3.amazonaws.com/doc/2006-03-01">
  <GetObjectResponse>
    <Status>
      <Code>200</Code>
      <Description>OK</Description>
    </Status>
    <Metadata>
      <Name>Content-Type</Name>
      <Value>text/plain</Value>
    </Metadata>
    <Metadata>
      <Name>family</Name>
      <Value>Muntz</Value>
    </Metadata>
    <Data>--first megabyte of bigfile--</Data>
    <LastModified>2006-01-01T12:00:00.000Z</LastModified>
    <ETag>"828ef3fdfa96f00ad9f27c383fc9ac7f"</ETag>
  </GetObjectResponse>
</GetObjectResponse>
```

若要確保檔案自上一個部分下載後沒有變更，請指定 IfMatch 元素。雖然不需要 IfMatch 元素，但建議用於可能變更的內容。

以下是使用 IfMatch 請求標頭取得檔案其餘部分的請求。

```
<GetObject xmlns="https://doc.s3.amazonaws.com/2006-03-01">
```

```

<Bucket>bigbucket</Bucket>
<Key>bigfile</Key>
<GetMetadata>true</GetMetadata>
<GetData>true</GetData>
<InlineData>true</InlineData>
<ByteRangeStart>10485761</ByteRangeStart>
<ByteRangeEnd>2023276</ByteRangeEnd>
<IfMatch>"828ef3fdfa96f00ad9f27c383fc9ac7f"</IfMatch>
<AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
<Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
<Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</GetObject>

```

Amazon S3 會傳回下列回應和檔案的其餘部分。

```

<GetObjectResponse xmlns="http://s3.amazonaws.com/doc/2006-03-01">
  <GetObjectResponse>
    <Status>
      <Code>200</Code>
      <Description>OK</Description>
    </Status>
    <Metadata>
      <Name>Content-Type</Name>
      <Value>text/plain</Value>
    </Metadata>
    <Metadata>
      <Name>family</Name>
      <Value>>Muntz</Value>
    </Metadata>
    <Data>--remainder of bigfile--</Data>
    <LastModified>2006-01-01T12:00:00.000Z</LastModified>
    <ETag>"828ef3fdfa96f00ad9f27c383fc9ac7f"</ETag>
  </GetObjectResponse>
</GetObjectResponse>

```

版本化 GetObject

下列請求會傳回儲存貯體中指定版本的物件。

```

<GetObject xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Key>Nelson</Key>
  <GetMetadata>true</GetMetadata>

```

```
<GetData>true</GetData>
<InlineData>true</InlineData>
<AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
<Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
<Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</GetObject>
```

回應範例

```
<GetObjectResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <GetObjectResponse>
    <Status>
      <Code>200</Code>
      <Description>OK</Description>
    </Status>
    <Metadata>
      <Name>Content-Type</Name>
      <Value>text/plain</Value>
    </Metadata>
    <Metadata>
      <Name>family</Name>
      <Value>Muntz</Value>
    </Metadata>
    <Data>aGEtaGE=</Data>
    <LastModified>2006-01-01T12:00:00.000Z</LastModified>
    <ETag>"828ef3fdfa96f00ad9f27c383fc9ac7f"</ETag>
  </GetObjectResponse>
</GetObjectResponse>
```

REST GET 錯誤復原

如果物件 GET 失敗，您可以透過指定要下載的範圍來取得其餘的檔案。若要這樣做，您必須使用取得物件的大小，ListBucket並在其餘檔案GET上執行範圍。如需詳細資訊，請參閱[GetObjectExtended \(SOAP API\)](#)。

相關資源

[物件上的操作 \(SOAP API\)](#)

GetObjectExtended (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

GetObjectExtended 與完全相同 [GetObject \(SOAP API\)](#)，但它支援下列其他元素，可用於完成 HTTP GET 標頭提供的許多相同功能（請前往 <http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html> : //）。

GetObjectExtended 還支援下列元素：GetObject

- ByteRangeStart, ByteRangeEnd: 這些元素指定僅應擷取一部分的物件資料。它們遵循 HTTP 位元組範圍的行為（前往 <https://http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.35>）。
- IfModifiedSince: 只有在物件的時間戳記晚於指定的時間戳記時，才傳回物件。（前往 <http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.25> : //）
- IfUnmodifiedSince: 只有在物件的時間戳記早於或等於指定的時間戳記時，才傳回物件。（前往 <http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.28> : //）
- IfMatch: 只有在物件的 ETag 符合提供的標籤時，才傳回物件（前往 <http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.24> : //）。
- IfNoneMatch: 只有在物件的 ETag 不符合提供的標籤時，才傳回物件（前往 <http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.26> : //）。
- ReturnCompleteObjectOnConditionFailure: ReturnCompleteObjectOnConditionFailure : 如果為 true，則如果請求包含範圍元素和一個或兩個 IfUnmodifiedSince/IfMatch 元素，且條件失敗，則傳回整個物件，而不是故障。這會啟用 If-Range 功能（前往 <https://http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.27>）。

DeleteObject (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

DeleteObject 操作會從 Amazon S3 移除指定的物件。一旦刪除，就無法還原或取消刪除物件。

Note

如果您刪除不存在的物件，Amazon S3 會傳回成功（而非錯誤訊息）。

Example

此範例會從「引號」儲存貯體中刪除「Nelson」物件。

Sample Request

```
<DeleteObject xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Key>Nelson</Key>
  <AWSAccessKeyId> AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</DeleteObject>
```

Sample Response

```
<DeleteObjectResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <DeleteObjectResponse>
    <Code>200</Code>
    <Description>OK</Description>
  </DeleteObjectResponse>
</DeleteObjectResponse>
```

元素

- Bucket：存放物件的儲存貯體。
- Key：識別物件的金鑰。

存取控制

只有在您可以WRITE存取儲存貯體時，才能刪除物件，無論誰擁有物件或授予物件哪些權利。

GetObjectAccessControlPolicy (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

GetObjectAccessControlPolicy 操作會擷取物件的存取控制政策。

Example

此範例會從「引號」儲存貯體擷取「Nelson」物件的存取控制政策。

Sample Request

```
<GetObjectAccessControlPolicy xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Key>Nelson</Key>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</GetObjectAccessControlPolicy>
```

Sample Response

```
<AccessControlPolicy>
  <Owner>
    <ID>a9a7b886d6fd24a541bf9b1c61be666e9</ID>
    <DisplayName>chriscustomer</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="CanonicalUser">
        <ID>a9a7b841bf9b1c61be666e9</ID>
        <DisplayName>chriscustomer</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
```



```
</Grantee>
  <Permission>READ</Permission>
</Grant>
</AccessControlList>
</AccessControlPolicy>
```

回應內文

回應包含儲存貯體的存取控制政策。如需此回應的說明，請參閱 [SOAP 存取政策](#)。

存取控制

您必須擁有物件READ_ACP的權限，才能擷取物件的存取控制政策。

SetObjectAccessControlPolicy (SOAP API)

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

SetObjectAccessControlPolicy 操作會設定現有物件的存取控制政策。如果成功，物件的先前存取控制政策會完全取代為指定的存取控制政策。

Example

此範例可讓指定的使用者（通常是擁有者）從「引號」儲存貯體FULL_CONTROL存取「Nelson」物件。

Sample Request

```
<SetObjectAccessControlPolicy xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Key>Nelson</Key>
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="CanonicalUser">
        <ID>a9a7b886d6fd24a52fe8ca5bef65f89a64e0193f23000e241bf9b1c61be666e9</ID>
        <DisplayName>chriscustomer</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
```

```
</AccessControlList>
<AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
<Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
<Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</SetObjectAccessControlPolicy>
```

Sample Response

```
<SetObjectAccessControlPolicyResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <SetObjectAccessControlPolicyResponse>
    <Code>200</Code>
    <Description>OK</Description>
  </SetObjectAccessControlPolicyResponse>
</SetObjectAccessControlPolicyResponse>
```

金鑰

存取控制

您必須擁有物件WRITE_ACP的權限，才能設定儲存貯體的存取控制政策。

對 SOAP 要求進行身分驗證

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

每個非匿名要求都必須包含身分驗證資訊，才能建立提出要求之委託人的身分。在 SOAP 中，身分驗證資訊會放入 SOAP 要求的下列元素中：

- 您的 AWS 存取金鑰 ID

Note

提出經過身分驗證的 SOAP 要求時，不支援暫時性安全登入資料。如需登入資料類型的詳細資訊，請參閱「[提出要求](#)」。

- **Timestamp:** 這必須是國際標準時間 (格林威治標準時間) 時區 (例如 2009-01-01T12:00:00.000Z) 的 `dateTime` (請前往 <http://www.w3.org/TR/xmlschema-2/#dateTime>)。如果此時間戳記與 Amazon S3 伺服器上的時鐘相差超過 15 分鐘，則授權會失敗。
- **Signature:** 使用 AWS 私密存取金鑰做為金鑰，串連 "AmazonS3" + OPERATION + Timestamp 的 RFC 2104 HMAC-SHA1 摘要 (前往 <http://www.ietf.org/rfc/rfc2104.txt> : //)。例如，在下列 CreateBucket 要求範例中，簽章元素將會包含值為 "AmazonS3CreateBucket2009-01-01T12:00:00.000Z" 的 HMAC-SHA1 摘要：

例如，在下列 CreateBucket 要求範例中，簽章元素將會包含值為 "AmazonS3CreateBucket2009-01-01T12:00:00.000Z" 的 HMAC-SHA1 摘要：

Example

```
<CreateBucket xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Acl>private</Acl>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2009-01-01T12:00:00.000Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</CreateBucket>
```

Note

SOAP 請求 (已驗證與匿名) 必須透過 SSL 傳送至 Amazon S3。當您透過 HTTP 傳送 SOAP 請求時，Amazon S3 會傳回錯誤。

Important

因為對於如何去除多餘的時間精確度有不同的解釋，.NET 使用者應該注意不要將過於精確的時間戳記傳送給 Amazon S3。這可以透過手動建構只有毫秒精準度的 `DateTime` 物件來完成。

使用 SOAP 設定存取原則

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

存取控制可以在儲存貯體或物件寫入時，藉由將要求的「AccessControlList」元素加入 CreateBucket、PutObjectInline 或 PutObject 中設定。AccessControlList 元素說明於 [Amazon S3 的 Identity and Access Management](#)。若這些操作未指定任何存取控制清單，將會以預設存取原則建立資源，以授予申請者 FULL_CONTROL 存取權 (即便要求是既有物 PutObjectInline 或 PutObject 要求亦是如此)。

下列要求會將資料寫入物件、允許匿名委託人讀取物件，以及將儲存貯體的 FULL_CONTROL 權限授予指定的使用者 (大部分的開發人員會授予自己其擁有之儲存貯體的 FULL_CONTROL 存取權)。

Example

下列要求會將資料寫入物件，並允許匿名委託人讀取物件。

Sample Request

```
<PutObjectInline xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Key>Nelson</Key>
  <Metadata>
    <Name>Content-Type</Name>
    <Value>text/plain</Value>
  </Metadata>
  <Data>aGEtaGE=</Data>
  <ContentLength>5</ContentLength>
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="CanonicalUser">
        <ID>75cc57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
        <DisplayName>chriscustomer</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
```

```
<Grantee xsi:type="Group">
  <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
</Grantee>
<Permission>READ</Permission>
</Grant>
</AccessControlList>
<AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
<Timestamp>2009-03-01T12:00:00.183Z</Timestamp>
<Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</PutObjectInline>
```

Sample Response

```
<PutObjectInlineResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <PutObjectInlineResponse>
    <ETag>"828ef3fdfa96f00ad9f27c383fc9ac7f"</ETag>
    <LastModified>2009-01-01T12:00:00.000Z</LastModified>
  </PutObjectInlineResponse>
</PutObjectInlineResponse>
```

使用

GetBucketAccessControlPolicy、GetObjectAccessControlPolicy、SetBucketAccessControlPolicy 及 SetObjectAccessControlPolicy 方法可以讀取或設定現有儲存貯體或物件的存取控制政策。如需詳細資訊，請參閱這些方法的詳細說明。

常見元素

您可以在任何 SOAP 要求中納入下列與授權相關的元素：

- AWSAccessKeyId：請求者的 AWS 存取金鑰 ID
- Timestamp：系統上目前的時間
- Signature：要求的簽章

SOAP 錯誤回應

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

在 SOAP 中，傳回給用戶端的錯誤結果是 SOAP 錯誤，其 HTTP 回應碼為 500。若未收到 SOAP 錯誤，表示要求成功。Amazon S3 的 SOAP 錯誤碼是由標準的 SOAP 1.1 錯誤碼加上 Amazon S3 專用錯誤代碼（「Server」或「Client」）組成。例如："Server.InternalError" 或 "Client.NoSuchBucket"。SOAP 錯誤字串元素包含可供人閱讀的一般英文錯誤訊息。最後，SOAP 錯誤詳細資訊元素也包含錯誤相關的其他資訊。

例如，當您嘗試刪除不存在的物件 "Fred" 時，SOAP 回應的內文中將會出現 "NoSuchKey" SOAP 錯誤。

下列範例顯示 SOAP 錯誤回應範例。

```
<soapenv:Body>
  <soapenv:Fault>
    <Faultcode>soapenv:Client.NoSuchKey</Faultcode>
    <Faultstring>The specified key does not exist.</Faultstring>
    <Detail>
      <Key>Fred</Key>
    </Detail>
  </soapenv:Fault>
</soapenv:Body>
```

下表說明 SOAP 錯誤回應元素

名稱	描述
Detail	錯誤中涉及之金鑰的容器 類型：容器 Ancestor：Body.Fault
Fault	錯誤資訊的容器。 類型：容器 Ancestor：內文
Faultcode	故障碼是可唯一識別錯誤條件的字串。偵測及依類型處理錯誤的程式必須讀取及了解此代碼。如需詳細資訊，請參閱 錯誤代碼清單 。 類型：字串

名稱	描述
	Ancestor : Body.Fault
Faultstring	錯誤字串包含英文錯誤條件的一般描述。目標對象是人。一般簡單的程式在發生其無法處理的錯誤狀況時，會直接對最終使用者顯示訊息。具有更詳盡錯誤處理和適當國際化的複雜程式更有可能忽略錯誤字串。 類型：字串 Ancestor : Body.Fault
Key	識別錯誤中涉及的金鑰 類型：字串 Ancestor : Body.Fault

附錄：驗證請求 (AWS 簽章第 2 版)

Important

本節說明如何使用 AWS Signature 第 2 版來驗證請求。Signature 第 2 版開始停止支援 (已淘汰)，Amazon S3 再來只接受使用 Signature 第 4 版簽署的 API 請求。如需詳細資訊，請參閱[AWS Amazon S3 的 Signature 第 2 版已關閉 \(已棄用\)](#)。

所有 都支援 Signature 第 4 版 AWS 區域，而且它是新區域唯一支援的版本。如需詳細資訊，請參閱《Amazon Simple Storage Service API 參考》中的[驗證請求AWS \(簽章版本 4\)](#)。

Amazon S3 可讓您識別簽署請求是使用哪個 API 簽章版本。您必須識別是否有任何工作流程使用 Signature 第 2 版進行簽署，並將這些工作流程升級為使用 Signature 第 4 版，以避免對您的業務造成影響。

- 如果您使用的是 CloudTrail 事件日誌 (建議選項)，請參閱[使用 CloudTrail 識別 Amazon S3 Signature 第 2 版請求](#)，了解如何查詢和識別此類請求。
- 如果您使用 Amazon S3 伺服器存取日誌，請參閱[使用 Amazon S3 伺服器存取日誌來識別請求](#)。

主題

- [使用 REST API 驗證請求 \(AWS 簽章版本 2\)](#)
- [簽署和驗證 REST 請求 \(AWS 簽章版本 2\)](#)
- [使用 POST 的瀏覽器型上傳 \(AWS 簽章版本 2\)](#)

使用 REST API 驗證請求 (AWS 簽章版本 2)

使用 REST 存取 Amazon S3 時，必須在您的請求中提供下列項目，才可驗證請求：

請求元素

- AWS 存取金鑰 ID – 每個請求必須包含您用來傳送請求之身分的存取金鑰 ID。
- 簽章 – 每個請求都必須包含有效的請求簽章，否則會拒絕請求。

使用您的私密存取金鑰即可計算出請求簽章，而該私密存取金鑰只有您和 AWS 知道。

- 時間戳記 – 每個請求都必須包含建立請求的日期和時間，以 UTC 字串表示。
- 日期 – 每個請求都必須包含請求的時間戳記。

視目前使用的 API 動作之不同，您可以改為提供要求過期日期與時間，或是與時間戳記一起提供。請參閱特定動作的驗證主題，以了解所需要的資訊為何。

以下是對傳給 Amazon S3 的請求進行身分驗證的一般步驟。假設您已有必要的安全性登入資料、存取金鑰 ID，以及私密存取金鑰。

You**1 Create a request:****Request**

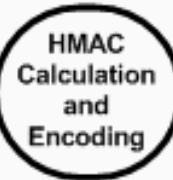
AccessKeyId = ...
Action = ...
Timestamp = ...
ParameterA = ...

2 Create an HMAC-SHA1 signature:

String based on
request contents

+

Your Secret Access Key



Your Signature

3 Send the request and signature to AWS:**Request**

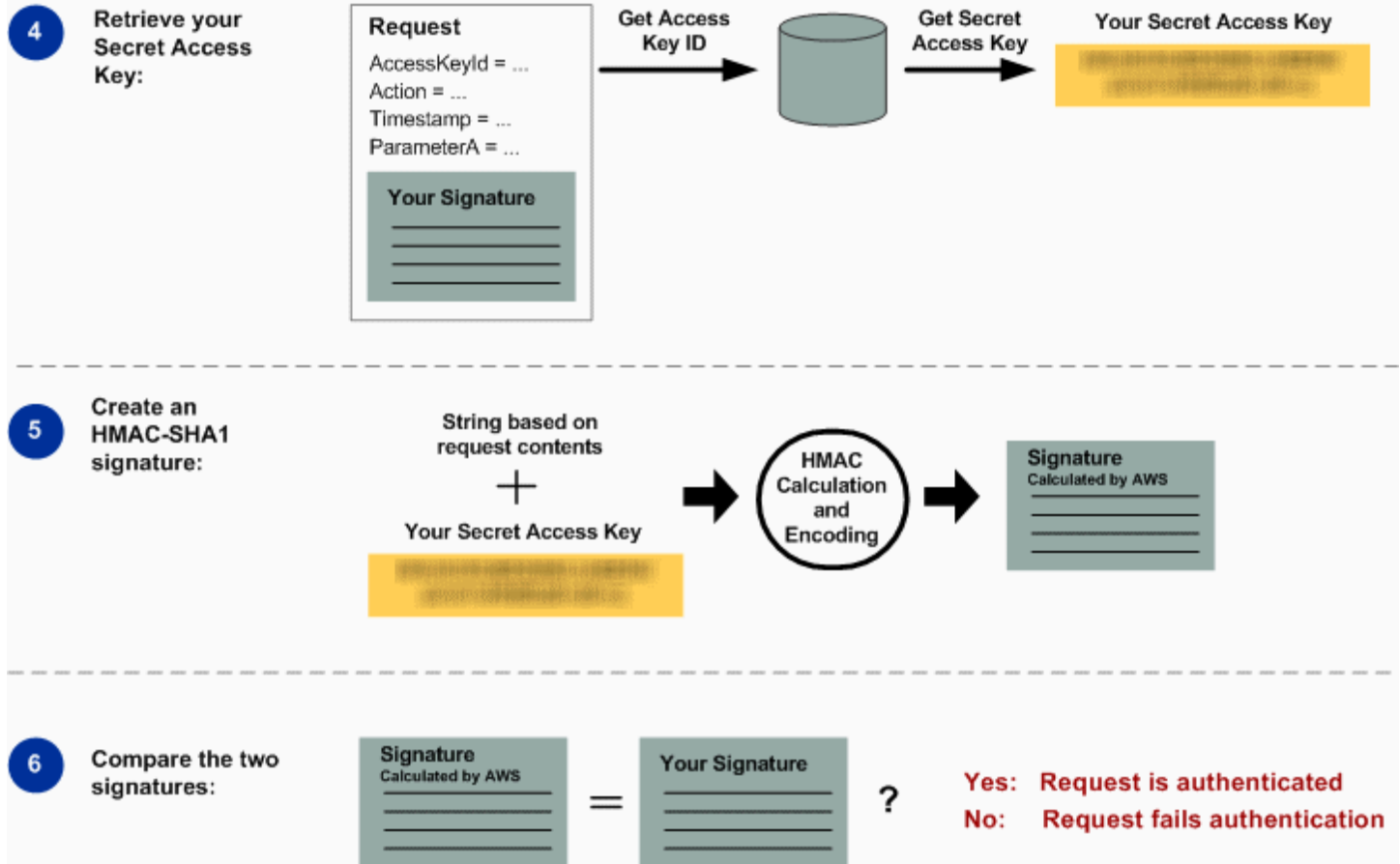
AccessKeyId = ...
Action = ...
Timestamp = ...
ParameterA = ...

Your Signature

**AWS**

- 1 建構 請求 AWS。
- 2 使用私密存取金鑰計算簽章。
- 3 將請求傳送至 Amazon S3。在要求中放入您的存取金鑰 ID 及簽章。Amazon S3 會執行接下來的三個步驟。

AWS



- 4 Amazon S3 使用存取金鑰 ID 來查詢您的私密存取金鑰。
- 5 Amazon S3 按照您對請求中傳送的簽章所使用的計算演算法，採用同樣的演算法來計算請求資料及私密存取金鑰中的簽章。
- 6 如果 Amazon S3 產生的簽章與請求中傳送的簽章相符，該請求即視為真實。若比對失敗，則會捨棄請求，Amazon S3 會傳回錯誤回應。

詳細的驗證資訊

如需 REST 驗證的詳細資訊，請參閱「[簽署和驗證 REST 請求 \(AWS 簽章版本 2\)](#)」。

簽署和驗證 REST 請求 (AWS 簽章版本 2)

主題

- [使用臨時安全登入資料](#)
- [Authentication 標頭](#)
- [將要求標準化以供簽署](#)
- [建構 CanonicalizedResource 元素](#)
- [建構 CanonicalizedAmzHeaders 元素](#)
- [位置性與具名的 HTTP 標頭 StringToSign 元素之比較](#)
- [時間戳記需求](#)
- [身分驗證範例](#)
- [REST 要求簽署問題](#)
- [查詢字串要求身分驗證替代項](#)

Note

本主題說明如何使用 Signature 第 2 版驗證要求。Amazon S3 現在支援最新的 Signature 第 4 版。所有區域都支援此最新的 Signature 版本，而且 2014 年 1 月 30 日以後的任何新區域只會支援 Signature 第 4 版。如需詳細資訊，請前往《Amazon Simple Storage Service API 參考》中的[驗證請求 \(AWS 簽章第 4 版\)](#)。

身分驗證是向系統表明身分的程序。身分是 Amazon S3 存取控制決策中很重要的因素。要求的允許或拒絕部分取決於要求者的身分。例如，建立儲存貯體的權利會保留給已註冊的開發人員，而 (根據預設) 建立儲存貯體中物件的權利則保留給討論中之儲存貯體的擁有者。身為開發人員，您會提出要求來叫用這些權限，因此您需要透過驗證要求，向系統表明身分。本節將告訴您如何做到。

Note

本節中的內容並不適用於 HTTP POST。如需詳細資訊，請參閱「[使用 POST 的瀏覽器型上傳 \(AWS 簽章版本 2\)](#)」。

Amazon S3 REST API 採用基於金鑰式 HMAC (雜湊訊息驗證碼) 的自訂 HTTP 機制進行身分驗證。若要驗證要求，請先將要求的選取元素串連成一個字串。然後，您可以使用 AWS 私密存取金鑰來計算該

字串的 HMAC。非正式地，此程序稱為「簽署請求」，而 HMAC 演算法的輸出是簽章，因為它模擬真實簽章的安全屬性。最後，請使用本節中所述的語法，將此簽章新增為要求的參數。

當系統收到已驗證的請求時，它會擷取您宣告擁有的 AWS 私密存取金鑰，並以相同方式使用該金鑰來計算收到訊息的簽章。然後，它會將所計算的簽章與要求者提供的簽章進行比較。如果兩個簽章相符，系統會得出請求者必須能夠存取 AWS 私密存取金鑰的結論，因此，會與發出金鑰的委託人授權一起行動。如果兩個簽章不符，則會捨棄要求，且系統會回應錯誤訊息。

Example 已驗證的 Amazon S3 REST 請求

```
GET /photos/puppy.jpg HTTP/1.1
Host: awsexamplebucket1.us-west-1.s3.amazonaws.com
Date: Tue, 27 Mar 2007 19:36:42 +0000

Authorization: AWS AKIAIOSFODNN7EXAMPLE:
qgk2+6Sv9/oM7G3qLEjTH1a1l1g=
```

使用臨時安全登入資料

如果您使用暫時性安全登入資料簽署要求 (請參閱「[提出要求](#)」)，您必須新增 x-amz-security-token 標頭，以在您的要求中包含對應的安全字符。

當您使用 AWS Security Token Service API 取得暫時性安全登入資料時，回應會包含暫時性安全登入資料與工作階段字符。當您將請求傳送至 Amazon S3 時，您會在 x-amz-security-token 標頭中提供此工作階段字符值。如需 IAM 提供之 AWS Security Token Service API 的相關資訊，請參閱 AWS Security Token Service 《API 參考指南》中的[動作](#)。

Authentication 標頭

Amazon S3 REST API 使用標準 HTTP Authorization 標頭來傳遞身分驗證資訊。(標準標頭的名稱並不適當，因為它傳遞身分驗證資訊，而不是身分驗證)。根據 Amazon S3 身分驗證機制，Authorization 標頭的格式如下：

```
Authorization: AWS AWSAccessKeyId:Signature
```

開發人員在註冊時會獲得 AWS 存取金鑰 ID 和 AWS 私密存取金鑰。進行要求身分驗證時，AWSAccessKeyId 元素會識別用來計算簽章的存取金鑰 ID；間接來說，就是提出要求的開發人員。

Signature 元素是要求中所選元素的 RFC 2104 HMAC-SHA1，因此 Authorization 標頭的 Signature 部分會因要求而有所不同。如果系統計算的請求簽章符合請求中 Signature 包含的，則

請求者將已示範擁有 AWS 秘密存取金鑰。此要求接著會以金鑰核發對象的開發人員身分 (具授權) 進行處理。

下列虛擬語法說明 Authorization 要求標頭的建構 (在此範例中，\n 表示 Unicode 字碼指標 U+000A，通常稱為新行字元)。

```
Authorization = "AWS" + " " + AWSAccessKeyId + ":" + Signature;

Signature = Base64( HMAC-SHA1( UTF-8-Encoding-Of(YourSecretAccessKey), UTF-8-Encoding-Of( StringToSign ) ) );

StringToSign = HTTP-Verb + "\n" +
  Content-MD5 + "\n" +
  Content-Type + "\n" +
  Date + "\n" +
  CanonicalizedAmzHeaders +
  CanonicalizedResource;

CanonicalizedResource = [ "/" + Bucket ] +
  <HTTP-Request-URI, from the protocol name up to the query string> +
  [ subresource, if present. For example "?acl", "?location", or "?logging"];

CanonicalizedAmzHeaders = <described below>
```

HMAC-SHA1 是 [RFC 2104 - Keyed-Hashing for Message Authentication](#) 所定義的演算法。該演算法接受兩個位元組字串、一個金鑰與一則訊息的輸入。對於 Amazon S3 請求身分驗證，請使用您的 AWS 私密存取金鑰 (YourSecretAccessKey) 做為金鑰，並將的 UTF-8 編碼StringToSign做為訊息。HMAC-SHA1 的輸出也是位元組字串，稱為 Digest。Signature 要求參數是由編碼此 Digest 的 Base64 所建構。

將要求標準化以供簽署

之前提到，當系統收到經過驗證的要求時，它會將計算的要求簽章與要求中 StringToSign 所提供的簽章進行比較。因此，您必須使用 Amazon S3 所用的相同方法來計算簽章。以商定格式提出請求以簽署正式化的程序。

建構 CanonicalizedResource 元素

CanonicalizedResource 代表所要請求的 Amazon S3 資源。您可以透過下列方式針對 REST 要求建構此元素：

1. 從空字串 ("") 開始。
2. 如果要求使用 HTTP Host 標頭 (虛擬託管型) 指定儲存貯體，請在儲存貯體名稱前面加上 "/" (例如 "/bucketname")。針對路徑型要求及未定址儲存貯體的要求，則不用執行任何操作。如需虛擬託管樣式請求的詳細資訊，請參閱 [儲存貯體的虛擬託管](#)。

針對虛擬託管型請求 "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/photos/puppy.jpg"，CanonicalizedResource 會是 "/awsexamplebucket1"。

針對路徑型請求 "https://s3.us-west-1.amazonaws.com/awsexamplebucket1/photos/puppy.jpg"，CanonicalizedResource 會是 ""。

3. 附加未解碼之 HTTP 要求 URI 的路徑部分，一直到 (但不包括) 查詢字串。

針對虛擬託管型請求 "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/photos/puppy.jpg"，CanonicalizedResource 會是 "/awsexamplebucket1/photos/puppy.jpg"。

針對路徑型的請求 "https://s3.us-west-1.amazonaws.com/awsexamplebucket1/photos/puppy.jpg"，CanonicalizedResource 會是 "/awsexamplebucket1/photos/puppy.jpg"。此時，虛擬託管型要求與路徑型要求的 CanonicalizedResource 會相同。

針對未定址儲存貯體的請求 (例如 [GET Service](#))，請附加 "/"。

4. 如果請求定址一項子資源 (例如 ?versioning、?location、?acl、?lifecycle 或 ?versionid)，請附加該子資源、其值 (若有) 與問號。請注意，如有多項子資源，則必須按詞典編纂方式依子資源名稱排序並以 '&' 分隔，例如 ?acl&versionId=*value*。

建構 CanonicalizedResource 元素時必須包含的子資源包含

acl、lifecycle、location、logging、notification、partNumber、policy、requestPayment、uploadId、uploadPartNumber 與 website。

如果請求指定查詢字串參數來覆寫回應標頭值 (請參閱 [Get 物件](#))，請附加查詢字串參數及其值。簽署時，您不會編碼這些值；不過，您必須在提出要求時編碼這些參數值。GET 要求中的查詢字串參數包含 response-content-type、response-content-language、response-expires、response-cache-control、response-content-disposition 與 response-content-encoding。

當您為多個物件的 Delete 要求建立 CanonicalizedResource 時，必須包含 delete 查詢字串參數。

來自 HTTP 要求 URI 的 CanonicalizedResource 元素必須依其出現在 HTTP 要求的原狀簽署，包括 URL 編碼的中繼字元。

CanonicalizedResource 可能與 HTTP 要求 URI 不同。特別是如果您的要求使用 HTTP Host 標頭指定儲存貯體，該儲存貯體不會出現在 HTTP 要求 URI 中。不過，CanonicalizedResource 會繼續包含該儲存貯體。查詢字串參數也可能出現在要求 URI 中，但未包含在 CanonicalizedResource 中。如需詳細資訊，請參閱[儲存貯體的虛擬託管](#)。

建構 CanonicalizedAmzHeaders 元素

若要建構屬於 StringToSign 一部分的 CanonicalizedAmzHeaders，請選取開頭為 'x-amz-' 的所有 HTTP 要求標頭 (使用不區分大小寫比較)，然後使用下列程序。

1. 將每個 HTTP 標頭名稱轉換成小寫。例如，'X-Amz-Date' 會變成 'x-amz-date'。
2. 按詞典編纂方式依標頭名稱排序標頭集合。
3. 按照 RFC 2616 第 4.2 節的規定，將具有相同名稱的標頭欄位合併成一個「標頭名稱:逗號分隔值清單」配對，值與值之間沒有任何空格。例如，兩個中繼資料標頭 'x-amz-meta-username: fred' 與 'x-amz-meta-username: barney' 會合併成單一標頭 'x-amz-meta-username: fred,barney'。
4. 「展開」跨多行的長標頭 (如 RFC 2616 第 4.2 節所允許)，方法是以單一空格取代摺疊的空格 (包括換行符號)。
5. 修剪標頭中冒號前後的任何空格。例如，標頭 'x-amz-meta-username: fred,barney' 會變成 'x-amz-meta-username:fred,barney'。
6. 最後，將新行字元 (U+000A) 附加至所產生之清單中的每個標準化標頭。將此清單中的所有標頭串連成一個字串，以建構 CanonicalizedResource 元素。

位置性與具名的 HTTP 標頭 StringToSign 元素之比較

StringToSign 的前幾個標頭元素 (Content-Type、日期和 Content-MD5) 屬於位置性標頭。StringToSign 不包含這些標頭的名稱，僅包含其在請求中的值。相對的，'x-amz-' 元素是具名的。其標頭名稱與標頭值都會出現在 StringToSign 中。

如果 StringToSign 定義中所需的位置性標頭不會出現在您的要求中 (例如 Content-Type 或 Content-MD5 對 PUT 要求為選用但對 GET 要求則沒有意義)，請以空字串 ("") 取代該位置。

時間戳記需求

經過驗證的要求必須具有有效的時間戳記 (使用 HTTP Date 標頭或 x-amz-date 替代項)。此外，經過驗證的請求所包含的用戶端時間戳記，必須在收到要求後不超過 Amazon S3 系統時間的 15 分鐘。否則，要求會失敗並出現 RequestTimeTooSkewed 錯誤碼。這些限制的用意是為了降低要求可能遭對手攔截而被重播的可能性。若要增強防範竊聽的保護，請使用 HTTPS 傳輸經過驗證的要求。

Note

要求日期的驗證限制僅適用於未使用查詢字串身分驗證之經過驗證的要求。如需詳細資訊，請參閱「[查詢字串要求身分驗證替代項](#)」。

某些 HTTP 用戶端程式庫無法設定要求的 Date 標頭。如果您無法在標準化標頭中包含 'Date' 標頭的值，您可以改用 'x-amz-date' 標頭來設定要求的時間戳記。x-amz-date 標頭的值必須是其中一個 RFC 2616 格式 (<http://www.ietf.org/rfc/rfc2616.txt>)。如果要求中有 x-amz-date 標頭，系統會在計算要求簽章時略過任何 Date 標頭。因此，如果您包含 x-amz-date 標頭，請在建構 Date 時，使用空字串來表示 StringToSign。如需範例，請參閱下一節。

身分驗證範例

本節中的範例使用下表中的 (非工作) 登入資料。

參數	Value
AWSAccessKeyId	AKIAIOSFODNN7EXAMPLE
AWSSecretAccessKey	wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY

在範例 StringToSign 中，格式並不重要；而 \n 表示 Unicode 字碼指標 U+000A，通常稱為新行字元。此外，範例使用 "+0000" 來指定時區。您可以改用 "GMT" 來指定時區，但範例中將會顯示不同的簽章。

Object GET

此範例會從 awsexamplebucket1 儲存貯體取得物件。

要求	StringToSign
<pre>GET /photos/puppy.jpg HTTP/1.1 Host: awsexamplebucket1.us-west-1.s3.amazonaws.com</pre>	<pre>GET\n \n \n Tue, 27 Mar 2007 19:36:42 +0000\n</pre>

要求	StringToSign
Date: Tue, 27 Mar 2007 19:36:42 +0000 <i>Authorization: AWS AKIAIOSFODNN7EXAMPLE: qgk2+6Sv9/oM7G3qLEjTH1a1l1g=</i>	/awsexamplebucket1/photos/puppy.jpg

請注意，CanonicalizedResource 包含儲存貯體名稱，但 HTTP 要求 URI 則否。(儲存貯體是由 Host 標頭指定)。

Note

下列 Python 指令碼會使用提供的參數計算上述簽章。您可以使用此腳本來建構您自己的簽名，視需要取代金鑰和 StringToSign。

```
import base64
import hmac
from hashlib import sha1

access_key = 'AKIAIOSFODNN7EXAMPLE'.encode("UTF-8")
secret_key = 'wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY'.encode("UTF-8")

string_to_sign = 'GET\n\n\nTue, 27 Mar 2007 19:36:42 +0000\n/awsexamplebucket1/
photos/puppy.jpg'.encode("UTF-8")
signature = base64.b64encode(
    hmac.new(
        secret_key, string_to_sign, sha1
    ).digest()
).strip()

print(f"AWS {access_key.decode()}:{signature.decode()}")
```

Object PUT

這個例子將物件放入 awsexamplebucket1 儲存貯體中。

要求	StringToSign
<pre>PUT /photos/puppy.jpg HTTP/1.1 Content-Type: image/jpeg Content-Length: 94328 Host: awsexamplebucket1.s3.us-west-1.amazonaws.com Date: Tue, 27 Mar 2007 21:15:45 +0000 Authorization: AWS AKIAIOSFODNN7EXAMPLE: iqRzw+ileNPu1fhspnRs8n0jjIA=</pre>	<pre>PUT\n \n image/jpeg\n Tue, 27 Mar 2007 21:15:45 +0000\n /awsexamplebucket1/photos/puppy.jpg</pre>

請注意要求與 StringToSign 中的 Content-Type 標頭。請注意，由於 Content-MD5 不在要求中，因此在 StringToSign 中會保留空白。

清單

此範例列出 awsexamplebucket1 儲存貯體的內容。

要求	StringToSign
<pre>GET /?prefix=photos&max-keys=50&marker=puppy HTTP/1.1 User-Agent: Mozilla/5.0 Host: awsexamplebucket1.s3.us-west-1.amazonaws.com Date: Tue, 27 Mar 2007 19:42:41 +0000 Authorization: AWS AKIAIOSFODNN7EXAMPLE: m0WP8eCtspQl5Ahe6L1SozdX9YA=</pre>	<pre>GET\n \n \n Tue, 27 Mar 2007 19:42:41 +0000\n /awsexamplebucket1/</pre>

請注意，CanonicalizedResource 後面有斜線，而且沒有查詢字串參數。

擷取

此範例會擷取 'awsexamplebucket1' 儲存貯體的存取控制政策子資源。

要求	StringToSign
<pre>GET /?acl HTTP/1.1 Host: awsexamplebucket1.s3.us-west-1.amazonaws.com Date: Tue, 27 Mar 2007 19:44:46 +0000 Authorization: AWS AKIAIOSFODNN7EXAMPLE: 82ZHiFIjc+WbcwFKGUEQspPn+0=</pre>	<pre>GET\n \n \n Tue, 27 Mar 2007 19:44:46 +0000\n /awsexamplebucket1/?acl</pre>

請注意子資源查詢字串參數如何包含在 CanonicalizedResource 中。

刪除

此範例會使用路徑型與 Date 替代項，從 'awsexamplebucket1' 儲存貯體中刪除物件。

要求	StringToSign
<pre>DELETE /awsexamplebucket1/photos/puppy.jpg HTTP/1.1 User-Agent: dotnet Host: s3.us-west-1.amazonaws.com Date: Tue, 27 Mar 2007 21:20:27 +0000 x-amz-date: Tue, 27 Mar 2007 21:20:26 +0000 Authorization: AWS AKIAIOSFODNN7EXAMPLE:XbyTlbQdu9Xw5o8P4iMwPktxQd8=</pre>	<pre>DELETE\n \n \n Tue, 27 Mar 2007 21:20:26 +0000\n /awsexamplebucket1/photos/puppy.jpg</pre>

請注意，替代的 'x-amz-date' 方法如何指定日期（因為我們的用戶端程式庫阻止我們設定日期，例如）。在此情況下，x-amz-date 會優先於 Date 標頭。因此，簽章中的日期項目必須包含 x-amz-date 標頭的值。

上傳

此範例會將物件連同中繼資料一起上傳至 CNAME 型虛擬託管儲存貯體。

要求	StringToSign
<pre> PUT /db-backup.dat.gz HTTP/1.1 User-Agent: curl/7.15.5 Host: static.example.com:8080 Date: Tue, 27 Mar 2007 21:06:08 +0000 x-amz-acl: public-read content-type: application/x-download Content-MD5: 4gJE4saaMU4BqNR0kLY+lw== X-Amz-Meta-ReviewedBy: joe@example.com X-Amz-Meta-ReviewedBy: jane@example.com X-Amz-Meta-FileChecksum: 0x02661779 X-Amz-Meta-ChecksumAlgorithm: crc32 Content-Disposition: attachment; filename=database.dat Content-Encoding: gzip Content-Length: 5913339 Authorization: AWS AKIAIOSFODNN7EXAMPLE: jtBQa0Aq+DkULFI8qrpwIjGEx0E= </pre>	<pre> PUT\n 4gJE4saaMU4BqNR0kLY+lw==\n application/x-download\n Tue, 27 Mar 2007 21:06:08 +0000\n x-amz-acl:public-read\n x-amz-meta-checksumalgorithm:crc32\n x-amz-meta-filechecksum:0x02661779\n x-amz-meta-reviewedby:joe@example.com,jane@example.com\n /static.example.com/db-backup.dat.gz </pre>

請注意 'x-amz-' 標頭如何排序、修剪多餘空格及轉換成小寫。另請注意，已聯結多個具有相同名稱的標頭，並使用逗號分隔其值。

請注意，只有 Content-Type 與 Content-MD5 HTTP 實體標頭會出現在 StringToSign 中。其他 Content-* 實體標頭則否。

同樣地，請注意 CanonicalizedResource 包含儲存貯體名稱，但 HTTP 要求 URI 則否。(儲存貯體是由 Host 標頭指定)。

列出我的所有儲存貯體

要求	StringToSign
<pre> GET / HTTP/1.1 Host: s3.us-west-1.amazonaws.com </pre>	<pre> GET\n \n </pre>

要求	StringToSign
Date: Wed, 28 Mar 2007 01:29:59 +0000 <i>Authorization: AWS AKIAIOSFODNN7EXAMPLE:qGdzdE RIC03wnaRNKh60qZehG9s=</i>	\n Wed, 28 Mar 2007 01:29:59 +0000\n /

Unicode 金鑰

要求	StringToSign
GET /dictionary/fran%C3%A7ais/pr %c3%a9f%c3%a8re HTTP/1.1 Host: s3.us-west-1.amazonaws.com Date: Wed, 28 Mar 2007 01:49:49 +0000 <i>Authorization: AWS AKIAIOSFODNN7EXAMP LE:DNEZGsoieTZ92F3bUfSPQcbGmLM=</i>	GET\n \n \n Wed, 28 Mar 2007 01:49:49 +0000\n /dictionary/fran%C3%A7ais/pr %c3%a9f%c3%a8re

Note

衍生自要求 URI 的 StringToSign 元素必須依原狀採用，包括 URL 編碼與大寫。

REST 要求簽署問題

當 REST 要求身分驗證失敗時，系統會以 XML 錯誤文件回應要求。此錯誤文件中所包含的資訊是為了協助開發人員診斷問題。特別是 StringToSign 錯誤文件中的 SignatureDoesNotMatch 元素會告訴您系統正在使用的要求標準化。

某些工具組會在您事先不知情的情況下，以無訊息方式插入標頭，例如在 PUT 期間加入標頭 Content-Type。在大部分情況下，插入的標頭值仍然保持不變，因此您可以使用 Ethereal 或 tcpmon 等工具來探索遺漏的標頭。

查詢字串要求身分驗證替代項

您可以將必要資訊當作查詢字串參數來傳遞，以驗證特定類型的請求，而不是使用 Authorization HTTP 標頭。這有助於讓第三方瀏覽器直接存取您的私有 Amazon S3 資料，而不需要代理請求。此概

念是建構「預先簽章」的請求，然後將其編碼為最終使用者可以擷取的 URL。此外，您可以指定過期時間，限制預先簽章的請求。

如需使用查詢參數驗證請求的詳細資訊，請參閱《Amazon Simple Storage Service API 參考》中的[驗證請求：使用查詢參數AWS（簽章第 4 版）](#)。如需使用 AWS SDKs 產生預先簽章 URLs 的範例，請參閱[使用預先簽章 URLs 共用物件](#)。

建立簽章

以下為經過查詢字串驗證的 Amazon S3 REST 請求範例。

```
GET /photos/puppy.jpg
?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Expires=1141889120&Signature=vjbyPxybdZaNmGa%2ByT272YEAiv4%3D HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
Date: Mon, 26 Mar 2007 19:37:58 +0000
```

查詢字串要求身分驗證方法不需要任何特殊 HTTP 標頭。相反地，必要的身分驗證元素會指定為查詢字串參數。

查詢字串參數名稱	範例值	說明
AWSAccessKeyId	AKIAIOSFODNN7EXAMPLE	您的 AWS 存取金鑰 ID。指定用於簽署請求的 AWS 私密存取金鑰，以及間接指定提出請求之開發人員的身分。
Expires	1141889120	簽章的過期時間，指定為自 epoch (1970 年 1 月 1 日 00:00:00 UTC) 起經過的秒數。在此時間 (根據伺服器) 以後收到的要求會遭到拒絕。
Signature	vjbyPxybdZaNmGa%2ByT272YEAiv4%3D	StringToSign 之 Base64 編碼 HMAC-SHA1 的 URL 編碼。

查詢字串要求身分驗證方法與一般方法稍有不同，但只是 Signature 要求參數與 StringToSign 元素之間的格式不同。下列虛擬語法說明查詢字串要求身分驗證方法。

```
Signature = URL-Encode( Base64( HMAC-SHA1( YourSecretAccessKey, UTF-8-Encoding-Of( StringToSign ) ) ) );
```

```
StringToSign = HTTP-VERB + "\n" +
  Content-MD5 + "\n" +
  Content-Type + "\n" +
  Expires + "\n" +
  CanonicalizedAmzHeaders +
  CanonicalizedResource;
```

YourSecretAccessKey 是當您註冊為 Amazon Web Service 開發人員時，Amazon 指派給您的 AWS 私密存取金鑰 ID。請注意 Signature 如何經過 URL 編碼，以便適當放在查詢字串中。另請注意，在 StringToSign 中，HTTP Date 位置性元素已取代為 Expires。CanonicalizedAmzHeaders 與 CanonicalizedResource 相同。

Note

在查詢字串身分驗證方法中，您不會使用 Date 或 x-amz-date request 標頭來計算 StringToSign。

查詢字串要求身分驗證

要求	StringToSign
<pre>GET /photos/puppy.jpg?AWSAccess KeyId=AKIAIOSFODNN7EXAMPLE& Signature=NpgCjnDzrM%2BWFzo ENXmpNDUsSn8%3D& Expires=1175139620 HTTP/1.1 Host: awsexamplebucket1.s3.us-wes t-1.amazonaws.com</pre>	<pre>GET\n \n \n 1175139620\n /awsexamplebucket1/photos/puppy.jpg</pre>

此範例假設當瀏覽器提出 GET 請求時，不會提供 Content-MD5 或 Content-Type 標頭，也不會設定任何 x-amz- 標頭，因此的這些部分StringToSign會保留空白。

使用 Base64 編碼

HMAC 要求簽章必須經過 Base64 編碼。Base64 編碼會將簽章轉換成可附加至要求的簡單 ASCII 字串。簽章字串中可能出現的字元 (例如加號 (+)、正斜線 (/) 與等號 (=)) 若用於 URI，則必須經過編碼。例如，如果驗證碼包含加號 (+)，請在要求中將它編碼為 %2B。正斜線會編碼為 %2F，而等號則會編碼為 %3D。

如需 Base64 編碼的範例，請參閱 Amazon S3 [身分驗證範例](#)。

使用 POST 的瀏覽器型上傳 (AWS 簽章版本 2)

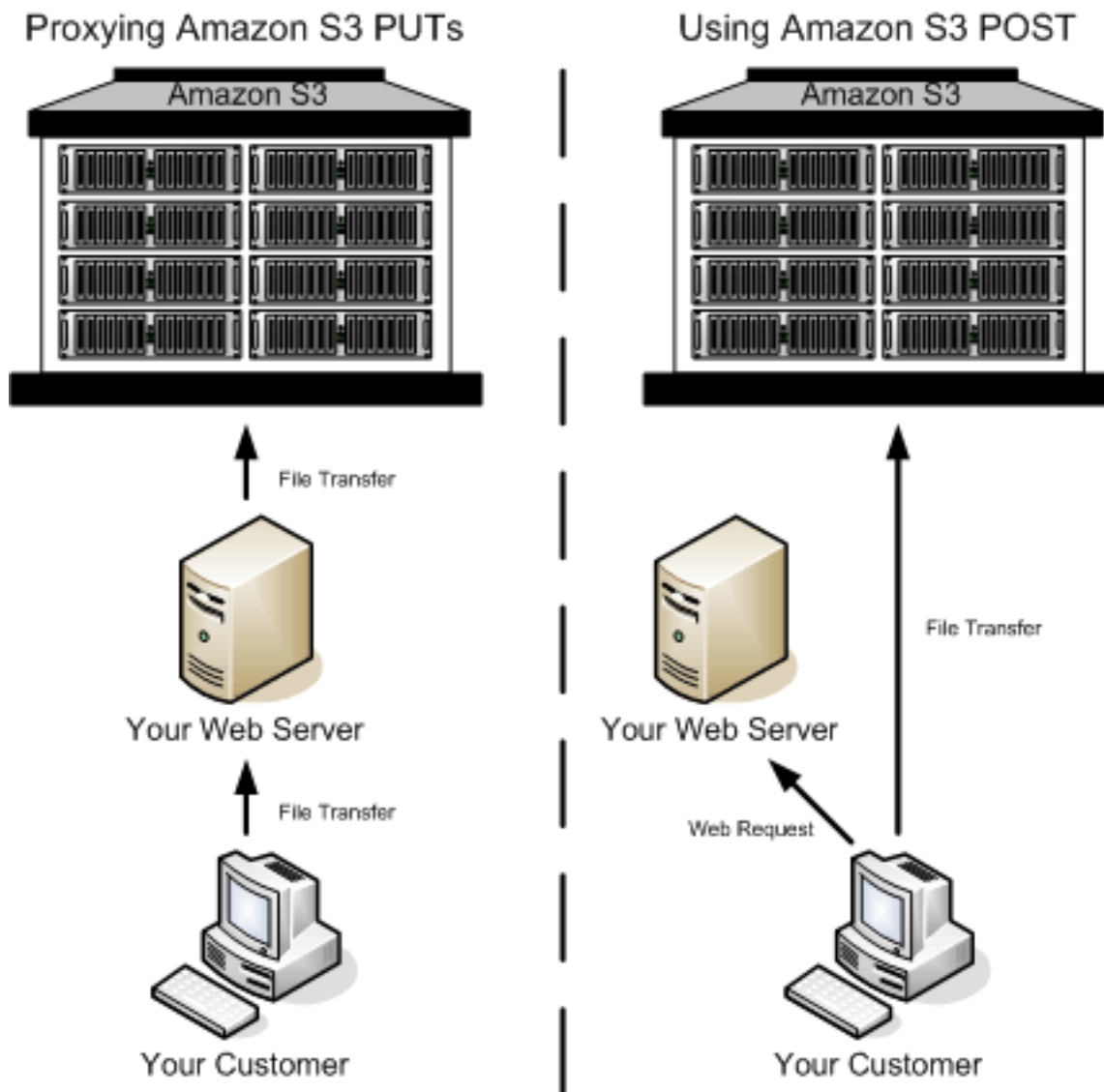
Amazon S3 支援 POST，可讓您的使用者直接將內容上傳至 Amazon S3。POST 旨在簡化上傳、降低上傳延遲，並省下您花在應用程式上讓使用者上傳資料以存放於 Amazon S3 的成本。

Note

本節討論的請求身分驗證是以 AWS Signature 第 2 版為基礎，這是一種通訊協定，用於驗證對 AWS 服務的傳入 API 請求。

Amazon S3 現在支援 Signature 第 4 版，這是一種通訊協定，用於驗證對 AWS 服務的傳入 API 請求 AWS 區域。目前，在 2014 年 1 月 30 日之前 AWS 區域建立的將繼續支援先前的通訊協定 Signature 第 2 版。2014 年 1 月 30 日以後的任何新區域只會支援 Signature 第 4 版，因此傳送至這些區域的所有要求都必須使用 Signature 第 4 版提出。如需詳細資訊，請參閱《Amazon Simple Storage Service API 參考》中的[使用 POST \(AWS 簽章第 4 版\) 驗證瀏覽器型上傳中的請求](#)。

下圖示範如何使用 Amazon S3 POST 進行上傳。



使用 POST 進行上傳

- 1 使用者開啟 Web 瀏覽器並存取您的網頁。
- 2 您的網頁包含一個 HTTP 表單，其中含有讓使用者將內容上傳至 Amazon S3 的所有必要資訊。
- 3 使用者直接將內容上傳至 Amazon S3。

Note

POST 不支援查詢字串身分驗證。

HTML 表單 (AWS 簽章版本 2)

主題

- [HTML 表單編碼](#)
- [HTML 表單宣告](#)
- [HTML 表單欄位](#)
- [政策建構](#)
- [建構簽章](#)
- [重新導向](#)

當您與 Amazon S3 通訊時，您通常會使用 REST 或 SOAP API 來執行放置、取得、刪除及其他作業。使用 POST 時，使用者經由瀏覽器直接將資料上傳至 Amazon S3，因此無法處理 SOAP API 或建立 REST PUT 請求。

Note

Amazon S3 的 SOAP API 不適用於新客戶，並且將於 2025 年 8 月 31 日接近生命週期結束 (EOL)。我們建議您使用 REST API 或 AWS SDKs。

若要允許使用者使用瀏覽器將內容上傳至 Amazon S3，您可以使用 HTML 表單。HTML 表單是由一個表單宣告與多個表單欄位所組成。表單宣告包含要求的高階資訊。表單欄位包含要求的詳細資訊，以及用來對要求進行身分驗證及確保其符合所指定條件的政策。

Note

表單資料與邊界 (檔案內容除外) 不得超過 20 KB。

本節說明如何使用 HTML 表單。

HTML 表單編碼

表單與政策必須經過 UTF-8 編碼。您可以在 HTML 標題中指定 UTF-8 編碼，或將它指定作為要求標頭來套用至表單。

 Note

HTML 表單宣告不接受查詢字串身分驗證參數。

以下為 HTML 標題中的 UTF-8 編碼範例：

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
```

以下為要求標頭中的 UTF-8 編碼範例：

```
Content-Type: text/html; charset=UTF-8
```

HTML 表單宣告

表單宣告有三個元件：動作、方法與封閉類型。若有任何值未正確設定，要求會失敗。

動作指定處理要求的 URL，必須將它設定為儲存貯體的 URL。例如，如果儲存貯體的名稱是 `awsexamplebucket1`，而區域是美國西部 (加利佛尼亞北部)，則 URL 為 `https://awsexamplebucket1.s3.us-west-1.amazonaws.com/`。

 Note

金鑰名稱是在表單欄位中指定。

方法必須是 POST。

封閉類型 (enctype) 必須加以指定，而且針對檔案上傳與文字區域上傳都必須設定為 `multipart/form-data`。如需詳細資訊，請參閱 [RFC 1867](#)。

Example

下列範例是儲存貯體 "awsexamplebucket1" 的表單宣告。

```
<form action="https://awsexamplebucket1.s3.us-west-1.amazonaws.com/" method="post"
  enctype="multipart/form-data">
```

HTML 表單欄位

下表說明可在 HTML 表單中使用的欄位。

 Note

系統會自動以使用者提供的檔案名稱來取代變數 `${filename}`，所有表單欄位都可識別此變數。如果瀏覽器或用戶端提供檔案的完整或部分路徑，則只會使用最後一個正斜線 (/) 或反斜線 (\) 後面的文字。例如，"C:\Program Files\directory1\file.txt" 會解譯為 "file.txt"。若未提供檔案或檔案名稱，則會以空字串取代變數。

欄位名稱	描述	必要
AWSAccessKeyId	儲存貯體擁有者的 AWS 存取金鑰 ID，其會針對滿足政策中一組限制的請求授予匿名使用者存取。如果要求包含政策文件，則需要此欄位。	有條件
acl	Amazon S3 存取控制清單 (ACL)。如果指定了無效的存取控制清單，則會產生錯誤。 類型：字串 預設值：private 有效值: private public-read public-read-write aws-exec-read authenticated-read	否

欄位名稱	描述	必要
	bucket-owner-read bucket-owner-full-control	
Cache-Control, Content-Type, Content-Disposition, Content-Encoding, Expires	REST 特定標頭。如需詳細資訊，請參閱 PUT 物件 。	否
key	上傳的金鑰名稱。 若要使用使用者提供的檔名，請使用 <code>\${filename}</code> 變數。例如，如果使用者 Betty 上傳檔案 lolcatz.jpg 且您指定 <code>/user/betty/\${filename}</code>，則檔案會另存為 <code>/user/betty/lolcatz.jpg</code>。 如需詳細資訊，請參閱 使用物件中繼資料。	是
policy	說明要求中所允許之內容的安全政策。未使用安全政策的要求會視為匿名，只有在可公開寫入的儲存貯體上才會成功。	否

欄位名稱	描述	必要
success_action_redirect, redirect	用戶端成功上傳時被重新導向的目標 URL。Amazon S3 會將 bucket、key 和 etag 值當作查詢字串參數附加至 URL。 若未指定 success_action_redirect，Amazon S3 會傳回 success_action_status 欄位中所指定的空白文件類型。 如果 Amazon S3 無法解譯 URL，則會忽略 欄位。 如果上傳失敗，Amazon S3 會顯示錯誤，而不會將使用者重新導向至 URL。 如需詳細資訊，請參閱重新導向。  Note redirect 欄位名稱已被取代，未來將會移除對 redirect 欄位名稱的支援。	否

欄位名稱	描述	必要
success_action_status	成功上傳時，傳回給用戶端的狀態碼 (若未指定 success_action_redirect)。 有效值為 200、201 或 204 (預設值)。 如果值設定為 200 或 204，Amazon S3 會傳回狀態碼為 200 或 204 的空白文件。 如果值設定為 201，Amazon S3 會傳回狀態碼為 201 的 XML 文件。如需 XML 文件內容的相關資訊，請參閱 POST 物件。 如果值未設定或設定為無效的值，Amazon S3 會傳回狀態碼為 204 的空白文件。  Note Adobe Flash Player 的某些版本無法正確處理具有空白主體的 HTTP 回應。若要支援經由 Adobe Flash 上傳，建議將 success_action_status 設定為 201。	否
signature	HMAC 簽章是使用對應至提供之 AWSSecretAccessKey 的私密存取金鑰所建構。如果要求包含政策文件，則需要此欄位。 如需詳細資訊，請參閱 Amazon S3 的 Identity and Access Management。	有條件

欄位名稱	描述	必要
x-amz-security-token	工作階段登入資料所使用的安全字符 如果請求使用 Amazon DevPay，則需要兩個 x-amz-security-token 表單欄位：一個表示產品字符，另一個表示使用者字符。 如果要求使用工作階段登入資料，則需要一個 x-amz-security-token 表單。如需詳細資訊，請參閱《IAM 使用者指南》中的臨時安全登入資料。	否
前面加上 x-amz-meta- 的其他欄位名稱	使用者指定的中繼資料。 Amazon S3 不會驗證或使用此資料。 如需詳細資訊，請參閱 PUT 物件。	否
file	檔案或文字內容。 檔案或內容必須是表單中的最後一個欄位。其下方的任何欄位都會被忽略。 您無法一次上傳多個檔案。	是

政策建構

主題

- [Expiration](#)
- [條件](#)
- [條件比對](#)
- [字元逸出](#)

政策是 UTF-8 與 Base64 編碼的 JSON 文件，其指定要求必須符合的條件，並用來對內容進行身分驗證。根據您設計政策文件的方式，您可以針對每次上傳、每位使用者、所有上傳，或根據符合您需求的其他設計來使用這些文件。

Note

雖然您不一定要使用政策文件，但強烈建議在將儲存貯體設為可公開寫入時使用它。

以下為政策文件的範例：

```
{ "expiration": "2007-12-01T12:00:00.000Z",  
  "conditions": [  
    {"acl": "public-read" },  
    {"bucket": "awsexamplebucket1" },  
    ["starts-with", "$key", "user/eric/"],  
  ]  
}
```

政策文件包含過期與條件。

Expiration

過期元素是以 ISO 8601 UTC 日期格式來指定政策的過期日期。例如，"2007-12-01T12:00:00.000Z" 指定政策在過了 2007-12-01 午夜 UTC 後無效。政策中需要有過期資訊。

條件

政策文件中的條件會驗證上傳的物件內容。您在表單中指定的每個表單欄位 (AWSAccessKeyId、簽章、檔案、政策與具有 x-ignore- 字首的欄位名稱除外) 必須包含在條件清單中。

Note

如果您有多個同名的欄位，則必須以逗號分隔其值。例如，如果您有名為 "x-amz-meta-tag" 的兩個欄位，且第一個欄位的值為 "Ninja"，第二個欄位的值為 "Stallman"，您會將政策文件設定為 `Ninja,Stallman`。

系統會展開表單中的所有變數，再驗證政策。因此，必須對展開的欄位執行所有條件比對。

例如，如果將金鑰欄位設定為 `user/betty/${filename}`，您的政策可能是 ["starts-with", "\$key", "user/betty/"]。請勿輸入 ["starts-with", "\$key", "user/betty/\${filename}"]。如需詳細資訊，請參閱「[條件比對](#)」。

下表說明政策文件條件。

元素名稱	描述
acl	指定 ACL 必須符合的條件。 支援完全符合與 starts-with 。
content-length-range	指定上傳內容允許的大小上下限。 支援範圍比對。
Cache-Control、Content-Type、Content-Disposition、Content-Encoding、Expires	REST 特定標頭。 支援完全符合與 starts-with 。
金鑰	上傳的金鑰名稱。 支援完全符合與 starts-with 。
success_action_redirect、redirect	用戶端成功上傳時被重新導向的目標 URL。 支援完全符合與 starts-with 。
success_action_status	

元素名稱	描述
	成功上傳時，傳回給用戶端的狀態碼 (若未指定 <code>success_action_redirect</code>)。 支援完全符合。
<code>x-amz-security-token</code>	Amazon DevPay 安全字符。 每個使用 Amazon DevPay 的請求都需要兩個 <code>x-amz-security-token</code> 表單欄位：一個表示產品字符，另一個表示使用者字符。因此，必須以逗號分隔其值。例如，如果使用使用者字符為 <code>ew91dHViZQ==</code> ，而產品字符為 <code>b0hnNVNKNWJlQTA=</code> ，您會將政策項目設定為： <code>{ "x-amz-security-token": "ew91dHViZQ==,b0hnNVNKNWJlQTA=" }</code> 。
前面加上 <code>x-amz-meta-</code> 的其他欄位名稱	使用者指定的中繼資料。 支援完全符合與 <code>starts-with</code> 。

Note

如果您的工具組新增其他欄位 (例如 Flash 新增檔案名稱)，則必須將它新增至政策文件。如果您可以控制此功能，請在欄位前面加上 `x-ignore-`，讓 Amazon S3 忽略此功能，這不會影響此功能的未來版本。

條件比對

下表說明條件比對類型。雖然您必須針對表單中所指定的每個表單欄位指定一個條件，但您可以藉由針對表單欄位指定多個條件，來建立更複雜的符合條件。

Condition	描述
完全符合	完全符合會確認欄位符合特定值。下列範例表示 ACL 必須設定為 <code>public-read</code> ：

Condition	描述
	<pre>{"acl": "public-read" }</pre> 您也可以使用下列範例來表示 ACL 必須設定為 public-read： <pre>["eq", "\$acl", "public-read"]</pre>
開頭為	如果值必須以特定值開頭，請使用 starts-with。下列範例表示金鑰必須以 user/betty 開頭： <pre>["starts-with", "\$key", "user/betty/"]</pre>
符合任何內容	若要設定政策以允許欄位中的任何內容，請使用 starts-with 並提供空白值。下列範例允許任何 success_action_redirect： <pre>["starts-with", "\$success_action_redirect", ""]</pre>
指定範圍	針對接受範圍的欄位，請以逗號分隔範圍的上下限。下列範例允許 1 到 10 MB 的檔案大小： <pre>["content-length-range", 1048579, 10485760]</pre>

字元逸出

下表說明政策文件中必須逸出的字元。

逸出序列	描述
\\	反斜線

逸出序列	描述
\\$	貨幣符號
\b	退格鍵
\f	Form Feed
\n	新行
\r	歸位字元
\t	水平標籤
\v	垂直標籤
\uXXXX	所有 Unicode 字元

建構簽章

步驟	描述
1	使用 UTF-8 進行政策的編碼。
2	使用 Base64 進行這些 UTF-8 位元組的編碼。
3	使用 HMAC SHA-1 透過私密存取金鑰簽署政策。
4	使用 Base64 進行 SHA-1 簽章的編碼。

如需身分驗證的一般資訊，請參閱 [Amazon S3 的 Identity and Access Management](#)。

重新導向

本節說明如何處理重新導向。

一般重新導向

完成 POST 要求後，使用者會被重新導向至您在 `success_action_redirect` 欄位中指定的位置。如果 Amazon S3 無法解譯 URL，則會忽略 `success_action_redirect` 欄位。

若未指定 `success_action_redirect`，Amazon S3 會傳回 `success_action_status` 欄位中所指定的空白文件類型。

如果 POST 請求失敗，Amazon S3 會顯示錯誤，而不會提供重新導向。

上傳前重新導向

如果您的儲存貯體是使用 `<CreateBucketConfiguration>` 所建立，您的最終使用者可能需要重新導向。如果發生此情況，某些瀏覽器可能無法正確處理重新導向。這是相當少見的情況，最可能在建立儲存貯體之後立即發生。

上傳範例 (AWS 簽章第 2 版)

主題

- [檔案上傳](#)
- [文字區域上傳](#)

Note

本節討論的請求身分驗證是以 AWS Signature 第 2 版為基礎，這是一種通訊協定，用於驗證對 AWS 服務的傳入 API 請求。

Amazon S3 現在支援 Signature 第 4 版，這是一種通訊協定，用於驗證對 AWS 服務的傳入 API 請求 AWS 區域。目前，在 2014 年 1 月 30 日之前 AWS 區域 建立的 將繼續支援先前的通訊協定 Signature 第 2 版。2014 年 1 月 30 日以後的任何新區域只會支援 Signature 第 4 版，因此傳送至這些區域的所有要求都必須使用 Signature 第 4 版提出。如需詳細資訊，請參閱《Amazon Simple Storage Service API 參考》中的[範例：使用 HTTP POST（使用 AWS Signature 第 4 版）進行瀏覽器型上傳](#)。

使用您的登入資料建立簽章；例如，0RavWzkygo6QX9caELEqKi9kDbU= 是上述政策文件的簽章。

下列表單支援對使用此政策的儲存 amzn-s3-demo-bucket 貯體提出 POST 請求。

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
    ...
    <form action="https://amzn-s3-demo-bucket.s3.us-west-1.amazonaws.com/" method="post"
    enctype="multipart/form-data">
      Key to upload: <input type="input" name="key" value="user/eric/" /><br />
      <input type="hidden" name="acl" value="public-read" />
      <input type="hidden" name="success_action_redirect" value="https://
awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html" />
      Content-Type: <input type="input" name="Content-Type" value="image/jpeg" /><br />
      <input type="hidden" name="x-amz-meta-uuid" value="14365123651274" />
      Tags for File: <input type="input" name="x-amz-meta-tag" value="" /><br />
      <input type="hidden" name="AWSAccessKeyId" value="AKIAIOSFODNN7EXAMPLE" />
      <input type="hidden" name="Policy" value="POLICY" />
      <input type="hidden" name="Signature" value="SIGNATURE" />
      File: <input type="file" name="file" /> <br />
      <!-- The elements after this will be ignored -->
      <input type="submit" name="submit" value="Upload to Amazon S3" />
    </form>
    ...
  </html>
```

請求範例

此要求假設上傳的映像為 117,108 個位元組 (不包含映像資料)。

```
POST / HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.10) Gecko/20071115
Firefox/2.0.0.10
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
```

```
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=9431149156168
Content-Length: 118698

--9431149156168
Content-Disposition: form-data; name="key"

user/eric/MyPicture.jpg
--9431149156168
Content-Disposition: form-data; name="acl"

public-read
--9431149156168
Content-Disposition: form-data; name="success_action_redirect"

https://awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html
--9431149156168
Content-Disposition: form-data; name="Content-Type"

image/jpeg
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-uuid"

14365123651274
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-tag"

Some,Tag,For,Picture
--9431149156168
Content-Disposition: form-data; name="AWSAccessKeyId"

AKIAIOSFODNN7EXAMPLE
--9431149156168
Content-Disposition: form-data; name="Policy"

eyJAiZXhwaXJhdGlvbiI6IClYMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWN
--9431149156168
Content-Disposition: form-data; name="Signature"

0RavWzkygo6QX9caELEqKi9kDbU=
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename.jpg"
```

```
Content-Type: image/jpeg

...file content...
--9431149156168
Content-Disposition: form-data; name="submit"

Upload to Amazon S3
--9431149156168--
```

回應範例

```
HTTP/1.1 303 Redirect
x-amz-request-id: 1AEE782442F35865
x-amz-id-2: cxzFLJRatFHy+NGtaDFRR8YvI9BHmgLxjvJzNiGGICARZ/mVXHj7T+qQKhdpzHFh
Content-Type: application/xml
Date: Wed, 14 Nov 2007 21:21:33 GMT
Connection: close
Location: https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
successful_upload.html?bucket=awsexamplebucket1&key=user/eric/
MyPicture.jpg&etag="39d459dfbc0faabbb5e179358dfb94c3";
Server: AmazonS3
```

文字區域上傳

主題

- [政策與表單建構](#)
- [請求範例](#)
- [回應範例](#)

下列範例顯示建構政策與表單以上傳文字區域的完整程序。上傳文字區域有助於提交使用者建立的內容，例如部落格張貼內容。

政策與表單建構

下列政策支援將文字區域上傳至 Amazon S3 的 awsexamplebucket1 儲存貯體。

```
{ "expiration": "2007-12-01T12:00:00.000Z",
  "conditions": [
    {"bucket": "awsexamplebucket1"},
    ["starts-with", "$key", "user/eric/"],
    {"acl": "public-read"},
```

```

    {"success_action_redirect": "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
new_post.html"},
    ["eq", "$Content-Type", "text/html"],
    {"x-amz-meta-uuid": "14365123651274"},
    ["starts-with", "$x-amz-meta-tag", ""]
  ]
}

```

此政策需要進行下列操作：

- 上傳必須發生在 2007-12-01 12:00 GMT 以前。
- 內容必須上傳到 `awsexamplebucket1` 儲存貯體。
- 金鑰必須以 `"user/eric/"` 開頭。
- ACL 已設定為 `public-read`。
- `success_action_redirect` 已設定為 `https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html`。
- 物件是 HTML 文字。
- `x-amz-meta-uuid` 標籤必須設定為 `14365123651274`。
- `x-amz-meta-tag` 可包含任意值。

以下為此政策的 Base64 編碼版本。

[illegible]

使用您的登入資料建立簽章。例如，qA7FWXKq6VvU68lI9KdveT1cWqF= 是上述政策文件的簽章。

下列表單支援對使用此政策的儲存 `amzn-s3-demo-bucket` 貯體提出 POST 請求。

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
```

```

<body>
...
<form action="https://amzn-s3-demo-bucket.s3.us-west-1.amazonaws.com/" method="post"
enctype="multipart/form-data">
  Key to upload: <input type="input" name="key" value="user/eric/" /><br />
  <input type="hidden" name="acl" value="public-read" />
  <input type="hidden" name="success_action_redirect" value="https://
awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html" />
  <input type="hidden" name="Content-Type" value="text/html" />
  <input type="hidden" name="x-amz-meta-uuid" value="14365123651274" />
  Tags for File: <input type="input" name="x-amz-meta-tag" value="" /><br />
  <input type="hidden" name="AWSAccessKeyId" value="AKIAIOSFODNN7EXAMPLE" />
  <input type="hidden" name="Policy" value="POLICY" />
  <input type="hidden" name="Signature" value="SIGNATURE" />
  Entry: <textarea name="file" cols="60" rows="10">

Your blog post goes here.

  </textarea><br />
  <!-- The elements after this will be ignored -->
  <input type="submit" name="submit" value="Upload to Amazon S3" />
</form>
...
</html>

```

請求範例

此要求假設上傳的映像為 117,108 個位元組 (不包含映像資料)。

```

POST / HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.10) Gecko/20071115
Firefox/2.0.0.10
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=178521717625888
Content-Length: 118635

-178521717625888

```

```
Content-Disposition: form-data; name="key"

ser/eric/NewEntry.html
--178521717625888
Content-Disposition: form-data; name="acl"

public-read
--178521717625888
Content-Disposition: form-data; name="success_action_redirect"

https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html
--178521717625888
Content-Disposition: form-data; name="Content-Type"

text/html
--178521717625888
Content-Disposition: form-data; name="x-amz-meta-uuid"

14365123651274
--178521717625888
Content-Disposition: form-data; name="x-amz-meta-tag"

Interesting Post
--178521717625888
Content-Disposition: form-data; name="AWSAccessKeyId"

AKIAIOSFODNN7EXAMPLE
--178521717625888
Content-Disposition: form-data; name="Policy"

eyJhZXBwaXJhdGlvbiI6IClYMDA3LTExVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWN
--178521717625888
Content-Disposition: form-data; name="Signature"

qA7FWXKq6VvU68lI9KdveT1cWgF=
--178521717625888
Content-Disposition: form-data; name="file"

...content goes here...
--178521717625888
Content-Disposition: form-data; name="submit"

Upload to Amazon S3
```

```
--178521717625888--
```

回應範例

```
HTTP/1.1 303 Redirect
x-amz-request-id: 1AEE782442F35865
x-amz-id-2: cxzFLJRatFHy+NGtaDFRR8YvI9BHmgLxjvJzNiGGICARZ/mVXHj7T+qQKhdpzHFh
Content-Type: application/xml
Date: Wed, 14 Nov 2007 21:21:33 GMT
Connection: close
Location: https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html?
bucket=awsexamplebucket1&key=user/eric/
NewEntry.html&etag=40c3271af26b7f1672e41b8a274d28d4
Server: AmazonS3
```

POST 搭配 adobe flash (AWS 簽章版本 2)

本節說明如何搭配 Adobe Flash 使用 POST。

Adobe Flash Player 安全

根據預設，Adobe Flash Player 安全模型禁止 Adobe Flash Player 與提供 SWF 檔案之網域外部的伺服器建立網路連線。

若要覆寫預設值，您必須將可公開讀取的 crossdomain.xml 檔案上傳至接受 POST 上傳的儲存貯體。以下為 crossdomain.xml 檔案範例。

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.macromedia.com/xml/dtds/cross-domain-policy.dtd">
<cross-domain-policy>
<allow-access-from domain="*" secure="false" />
</cross-domain-policy>
```

Note

如需 Adobe Flash 安全模型的詳細資訊，請前往 Adobe 網站。
將 crossdomain.xml 檔案新增至您的儲存貯體，可讓任何 Adobe Flash Player 連接到儲存貯體中的 crossdomain.xml 檔案；但這不會授予對實際 Amazon S3 儲存貯體的存取權。

Adobe Flash 考量

Adobe Flash 中的 FileReference API 會將 Filename 表單欄位新增至 POST 要求。當您建立 Adobe Flash 應用程式來支援使用 FileReference API 動作上傳至 Amazon S3 時，請在您的政策中包含下列條件：

```
['starts-with', '$Filename', '']
```

Adobe Flash Player 的某些版本無法正確處理具有空白主體的 HTTP 回應。若要設定 POST 以傳回沒有空白主體的回應，請將 success_action_status 設定為 201。Amazon S3 將會傳回狀態碼為 201 的 XML 文件。如需 XML 文件內容的相關資訊，請參閱 [POST 物件](#)。如需表單欄位的資訊，請參閱「[HTML 表單欄位](#)」。

附錄：生命週期組態 APIs (已棄用)

儲存貯體生命週期組態已更新，以支援以物件標籤為基礎的篩選條件。也就是說，您現在可以指定指定金鑰名稱字首、一或多個物件標籤或兩者的規則，以選取規則套用的物件子集。APIs 已相應更新。下列主題說明舊版 PUT 和 GET 儲存貯體生命週期操作的回溯相容性。

主題

- [PUT 儲存貯體生命週期 \(已棄用 \)](#)
- [GET 儲存貯體生命週期 \(已棄用 \)](#)

PUT 儲存貯體生命週期（已棄用）

說明

Important

如需此 API 的更新版本，請參閱 [PutBucketLifecycleConfiguration](#)。此版本已棄用。現有的生命週期組態將會運作。對於新的生命週期組態，請使用更新的 API。

為儲存貯體建立新的生命週期組態，或取代現有的生命週期組態。如需生命週期組態的相關資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [物件生命週期管理](#)。

許可

根據預設，所有 Amazon S3 資源，包括儲存貯體、物件和相關子資源（例如生命週期組態和網站組態）都是私有的。只有 AWS 帳戶 建立資源的 資源擁有者 可以存取它。資源擁有者可以透過撰寫存取政策，選擇性地將存取許可授予其他人。對於此操作，使用者必須取得 `s3:PutLifecycleConfiguration` 許可。

您也可以明確拒絕許可。明確拒絕也會取代任何其他許可。如果您想要防止使用者或帳戶從儲存貯體中移除或刪除物件，您必須拒絕他們執行下列動作的許可：

- `s3:DeleteObject`
- `s3:DeleteObjectVersion`
- `s3:PutLifecycleConfiguration`

如需許可的詳細資訊，請參閱《[Amazon Simple Storage Service 使用者指南](#)》中的 [管理 Amazon S3 資源的存取許可](#)。

要求

語法

```
PUT /?lifecycle HTTP/1.1
Host: bucketname.s3.amazonaws.com
Content-Length: length
Date: date
Authorization: authorization string
```

Content-MD5: *MD5*

Lifecycle configuration in the request body

如需授權字串的詳細資訊，請參閱 [驗證請求AWS \(簽章第 4 版 \)](#)。

請求參數

此實作操作不使用請求參數。

請求標頭

名稱	描述	必要
Content-MD5	資料的 base64 編碼 128 位元 MD5 摘要。您必須使用此標頭做為訊息完整性檢查，以確認請求內文未在傳輸中損毀。如需詳細資訊，請參閱 RFC 1864。 類型：字串 預設：無	是

請求主體

在請求中，您可以在請求內文中指定生命週期組態。生命週期組態指定為 XML。以下是基本生命週期組態的範例。它會指定一個規則。規則Prefix中的 會識別套用規則的物件。此規則也會指定兩個動作 (Transition和 Expiration)。每個動作都會指定 Amazon S3 應執行動作的時間軸。Status 指出規則是啟用或停用。

```
<LifecycleConfiguration>
  <Rule>
    <ID>sample-rule</ID>
    <Prefix>key-prefix</Prefix>
    <Status>rule-status</Status>
    <Transition>
      <Date>value</Date>
      <StorageClass>storage class</StorageClass>
    </Transition>
    <Expiration>
```

```

    <Days>value</Days>
  </Expiration>
</Rule>
</LifecycleConfiguration>

```

如果儲存貯體的狀態已啟用版本控制或暫停版本控制，您可以擁有相同物件的多個版本：一個目前版本和零個或多個非目前版本。下列生命週期組態會指定非目前物件版本特有的動作 (NoncurrentVersionTransition、NoncurrentVersionExpiration)。

```

<LifecycleConfiguration>
  <Rule>
    <ID>sample-rule</ID>
    <Prefix>key-prefix</Prefix>
    <Status>rule-status</Status>
    <NoncurrentVersionTransition>
      <NoncurrentDays>value</NoncurrentDays>
      <StorageClass>storage class</StorageClass>
    </NoncurrentVersionTransition>
    <NoncurrentVersionExpiration>
      <NoncurrentDays>value</NoncurrentDays>
    </NoncurrentVersionExpiration>
  </Rule>
</LifecycleConfiguration>

```

您可以使用分段上傳 API，分段上傳大型物件。如需分段上傳的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[分段上傳概觀](#)。透過生命週期組態，您可以指示 Amazon S3 取消未完成的分段上傳，如果未在指定的天數內完成，則會透過規則中指定的金鑰名稱字首來識別。當 Amazon S3 取消分段上傳時，它會刪除與上傳相關聯的所有組件。這可確保您沒有未完成的分段上傳，這些分段上傳有存放於 Amazon S3 的剩餘部分，因此您不需要為它們支付儲存成本。下列生命週期組態範例指定了一項規則，其會採取 AbortIncompleteMultipartUpload 動作。此動作會通知 Amazon S3 在啟動七天後取消未完成的分段上傳。

```

<LifecycleConfiguration>
  <Rule>
    <ID>sample-rule</ID>
    <Prefix>SomeKeyPrefix</Prefix>
    <Status>rule-status</Status>
    <AbortIncompleteMultipartUpload>
      <DaysAfterInitiation>7</DaysAfterInitiation>
    </AbortIncompleteMultipartUpload>
  </Rule>

```

</LifecycleConfiguration>

下表說明生命週期組態中的 XML 元素。

名稱	描述	必要
AbortIncompleteMultipartUpload	用於指定未完成分段上傳何時符合中止操作資格的容器。 子系：DaysAfterInitiation 類型：容器 Ancestor：Rule	是，如果沒有為規則指定其他動作
Date	您希望 Amazon S3 採取動作的日期。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的生命週期規則：根據特定日期。 日期值必須符合 ISO 8601 格式。時間一律是午夜 UTC。 類型：字串 Ancestor：Expiration 或 Transition	是，如果 Days 和 ExpiredObjectDeleteMarker 不存在
Days	指定特定規則動作生效時，物件建立後的天數。 類型：與 搭配使用時的非負整數 Transition，與 搭配使用時為正整數 Expiration Ancestor：Expiration、Transition	是，如果 Date 和 ExpiredObjectDeleteMarker 不存在
DaysAfterInitiation	指定必須完成分段上傳時，啟動分段上傳後的天數。如果未在指定的天數之前完成，則會符	是，如果指定父標籤

名稱	描述	必要
	合中止操作的資格，且 Amazon S3 會取消未完成的分段上傳。 類型：正整數 Ancestor：AbortIncompleteMultipartUpload	

名稱	描述	必要
Expiration	此動作會指定物件生命週期內 Amazon S3 應採取適當過期動作的期間。Amazon S3 採取的動作取決於儲存貯體是否已啟用版本控制。 • 如果從未在儲存貯體上啟用版本控制，Amazon S3 會永久刪除物件的唯一副本。 • 如果儲存貯體已啟用版本控制（或版本控制已暫停），則動作僅適用於物件的目前版本。已啟用版本控制的儲存貯體可以有許多相同物件的版本：一個目前版本和零個或多個非目前版本。 Amazon S3 不會刪除目前版本，而是新增刪除標記做為新的目前版本，使其成為非目前版本。	是，如果中不存在其他動作Rule。

 **Important**

如果儲存貯體的狀態已暫停版本控制，Amazon S3 會建立版本 ID 為的刪除標記null。如果您有版本 ID 為的版本null，Amazon S3 會覆寫該版本。

 **Note**

若要設定非目前物件的過期時間，請使用 NoncurrentVersionExpiration 動作。

名稱	描述	必要
	類型：容器 子系：天或日期 Ancestor：規則	
ID	規則的唯一識別碼。該值不可超過 255 個字元。 類型：字串 Ancestor：規則	否
LifecycleConfiguration	生命週期規則的容器。最多可以新增 1000 個規則。 類型：容器 子系：規則 Ancestor：無	是

名稱	描述	必要
ExpiredObjectDeleteMarker	在版本控制儲存貯體（已啟用版本控制或暫停版本控制儲存貯體）上，您可以在生命週期組態中新增此元素，以指示 Amazon S3 刪除過期的物件刪除標記。如需範例，請參閱《Amazon Simple Storage Service 使用者指南》中的範例 8：移除過期的物件刪除標記。請勿將其新增至非版本控制的儲存貯體，因為該類型的儲存貯體不能包含刪除標記。 類型：字串 有效值：true false (false 允許該值，但非 op，這表示 Amazon S3 不會採取動作) Ancestor：Expiration	是，如果 Date 和 Days 不存在
NoncurrentDays	指定物件為非目前狀態的天數，在此之後，Amazon S3 才可以執行相關聯的動作。如需非目前天數計算的資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 Amazon S3 如何計算物件何時成為非目前物件。 類型：與 搭配使用時的非負整數 NoncurrentVersionTransition，與 搭配使用時為正整數 NoncurrentVersionExpiration Ancestor：NoncurrentVersionExpiration 或 NoncurrentVersionTransition	是

名稱	描述	必要
NoncurrentVersionExpiration	指定非目前物件版本的到期時間。過期時，Amazon S3 會永久刪除非目前的物件版本。 在已啟用（或暫停）版本控制的儲存貯體上設定此生命週期組態動作，以指示 Amazon S3 在物件生命週期內的特定期間內刪除非目前物件版本。 類型：容器 子系：NoncurrentDays Ancestor：規則	是，如果中不存在其他動作 Rule
NoncurrentVersionTransition	轉換規則的容器，描述非目前物件何時轉換至 STANDARD_IA、ONEZONE_IA 或 儲存類別。 如果您的儲存貯體已啟用版本控制（或暫停版本控制），您可以設定此動作，指示 Amazon S3 在物件生命週期的特定期間轉換非目前物件版本。 類型：容器 子系：NoncurrentDays 和 StorageClass Ancestor：規則	是，如果中不存在其他動作 Rule
Prefix	物件金鑰字首，可識別套用規則的一或多個物件。 類型：字串 Ancestor：規則	是

名稱	描述	必要
Rule	生命週期規則的容器。生命週期組態最多可包含 1000 個規則。 類型：容器 Ancestor：LifecycleConfiguration	是
Status	如果啟用，Amazon S3 會依排程執行規則。如果停用，Amazon S3 會忽略規則。 類型：字串 Ancestor：規則 有效值：已啟用、已停用	是
StorageClass	指定您希望物件轉換的 Amazon S3 儲存體方案。 類型：字串 Ancestor：Transition NoncurrentVersionTransition 有效值：STANDARD_IA ONEZONE_IA GLACIER	是 只有在您指定一個或兩個前代時，才需要此元素。

名稱	描述	必要
Transition	此動作會指定物件生命週期內 Amazon S3 應將物件轉換為 STANDARD_IA 、 ONEZONE_IA 或 儲存體方案的期間。當此動作生效時，Amazon S3 執行的操作取決於儲存貯體是否已啟用版本控制。 • 如果從未在儲存貯體上啟用版本控制，Amazon S3 會將物件的唯一複本轉換為指定的儲存類別。 • 如果您的儲存貯體已啟用版本控制（或版本控制已暫停），Amazon S3 只會轉換規則中識別的物件目前版本。  Note 已啟用版本控制的儲存貯體可以有多個物件版本。此動作不會影響非目前物件版本。若要轉換非目前物件，您必須使用 NoncurrentVersionTransition 動作。 類型：容器 子系：天或日期，以及 StorageClass Ancestor：規則	是，如果中不存在其他動作 Rule

回應

回應標頭

此操作的實作只會使用大多數回應通用的回應標頭。如需詳細資訊，請參閱[常見回應標頭](#)。

回應元素

此操作的實作不會傳回回應元素。

特殊錯誤

此操作的實作不會傳回特殊錯誤。如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱[錯誤回應](#)。

範例

範例 1：將生命週期組態新增至未啟用版本控制的儲存貯體

下列生命週期組態會指定兩個規則，每個規則都有一個動作。

- 轉換動作會指示 Amazon S3 在建立物件的 30 天後，將具有「文件/」字首的物件轉換為儲存類別。
- 過期動作會指示 Amazon S3 在建立後 365 天刪除字首為「logs/」的物件。

```
<LifecycleConfiguration>
  <Rule>
    <ID>id1</ID>
    <Prefix>documents</Prefix>
    <Status>Enabled</Status>
    <Transition>
      <Days>30</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
  </Rule>
  <Rule>
    <ID>id2</ID>
    <Prefix>logs</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <Days>365</Days>
```

```
</Expiration>
</Rule>
</LifecycleConfiguration>
```

以下是將上述生命週期組態新增至儲存examplebucket貯體的範例PUT `/?lifecycle`請求。

```
PUT /?lifecycle HTTP/1.1
Host: examplebucket.s3.amazonaws.com
x-amz-date: Wed, 14 May 2014 02:11:21 GMT
Content-MD5: q6yJDlIkBaGGfb3QLY69A==
Authorization: authorization string
Content-Length: 415

<LifecycleConfiguration>
  <Rule>
    <ID>id1</ID>
    <Prefix>documents/</Prefix>
    <Status>Enabled</Status>
    <Transition>
      <Days>30</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
  </Rule>
  <Rule>
    <ID>id2</ID>
    <Prefix>logs/</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <Days>365</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

以下是範例回應。

```
HTTP/1.1 200 OK
x-amz-id-2: r+qR7+nhXtJDDIJ0JJYcd+1j5nM/rUFiiiZ/fNbD0sd3JUE8NWMLNHXmvPfwMpdC
x-amz-request-id: 9E26D08072A8EF9E
Date: Wed, 14 May 2014 02:11:22 GMT
Content-Length: 0
Server: AmazonS3
```

範例 2：將生命週期組態新增至已啟用版本控制的儲存貯體

下列生命週期組態會指定兩個規則，每個規則都有一個動作供 Amazon S3 執行。當您的儲存貯體已啟用版本控制或暫停版本控制時，您可以指定這些動作：

- `NoncurrentVersionExpiration` 動作會指示 Amazon S3 在物件變成非最新版本的 100 天後，將具有「logs/」字首的物件版本過期。
- `NoncurrentVersionTransition` 動作會通知 Amazon S3 在物件變成非最新版本的 30 天後，將具有「文件/」字首的物件轉換為儲存類別。

```
<LifecycleConfiguration>
  <Rule>
    <ID>DeleteAfterBecomingNonCurrent</ID>
    <Prefix>logs/</Prefix>
    <Status>Enabled</Status>
    <NoncurrentVersionExpiration>
      <NoncurrentDays>100</NoncurrentDays>
    </NoncurrentVersionExpiration>
  </Rule>
  <Rule>
    <ID>TransitionAfterBecomingNonCurrent</ID>
    <Prefix>documents/</Prefix>
    <Status>Enabled</Status>
    <NoncurrentVersionTransition>
      <NoncurrentDays>30</NoncurrentDays>
      <StorageClass>GLACIER</StorageClass>
    </NoncurrentVersionTransition>
  </Rule>
</LifecycleConfiguration>
```

以下是將上述生命週期組態新增至儲存examplebucket貯體的範例PUT `/?lifecycle`請求。

```
PUT /?lifecycle HTTP/1.1
Host: examplebucket.s3.amazonaws.com
x-amz-date: Wed, 14 May 2014 02:21:48 GMT
Content-MD5: 96rxH9mDqVnKkaZDddgnw==
Authorization: authorization string
Content-Length: 598

<LifecycleConfiguration>
  <Rule>
```

```

<ID>DeleteAfterBecomingNonCurrent</ID>
<Prefix>logs/</Prefix>
<Status>Enabled</Status>
<NoncurrentVersionExpiration>
  <NoncurrentDays>1</NoncurrentDays>
</NoncurrentVersionExpiration>
</Rule>
<Rule>
  <ID>TransitionSoonAfterBecomingNonCurrent</ID>
  <Prefix>documents/</Prefix>
  <Status>Enabled</Status>
  <NoncurrentVersionTransition>
    <NoncurrentDays>0</NoncurrentDays>
    <StorageClass>GLACIER</StorageClass>
  </NoncurrentVersionTransition>
</Rule>
</LifecycleConfiguration>

```

以下是範例回應。

```

HTTP/1.1 200 OK
x-amz-id-2: aXQ+KbIrmMmo0//3bMdDTw/CnjArwje+J49Hf+j44yRb/VmbIkgI05A+PT98Cp/6k07hf
+LD2mY=
x-amz-request-id: 02D7EC4C10381EB1
Date: Wed, 14 May 2014 02:21:50 GMT
Content-Length: 0
Server: AmazonS3

```

其他範例

如需將物件轉換為儲存類別的更多範例，例如 STANDARD_IA 或 ONEZONE_IA，請參閱[生命週期組態範例](#)。

相關資源

- [GetBucketLifecycleConfiguration](#)
- [POST 物件還原](#)
- 根據預設，資源擁有者 - 在這種情況下，建立儲存貯體 AWS 帳戶 的 儲存貯體擁有者 - 可以執行任何操作。資源擁有者也可以授予其他人執行操作的許可。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的下列主題：
 - [在政策中指定許可](#)

- [管理對您 Amazon S3 資源的存取許可](#)

GET 儲存貯體生命週期 (已棄用)

說明

Important

如需此 API 的更新版本，請參閱 [GetBucketLifecycleConfiguration](#)。如果您使用 <filter> 元素設定儲存貯體生命週期，您應該會看到本主題的更新版本。本主題是為了回溯相容性而提供。

傳回儲存貯體上設定的lifecycle組態資訊。如需生命週期組態的相關資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[物件生命週期管理](#)。

若要使用此操作，您必須具有執行 s3:GetLifecycleConfiguration動作的許可。根據預設，儲存貯體擁有者具有此許可。儲存貯體擁有者可以將此許可授予其他人。如需許可的詳細資訊，請參閱《[Amazon Simple Storage Service 使用者指南](#)》中的管理 Amazon S3 資源的存取許可。

要求

語法

```
GET /?lifecycle HTTP/1.1
Host: bucketname.s3.amazonaws.com
Date: date
Authorization: authorization string (see ####AWS#### 4 ##)
```

請求參數

此實作操作不使用請求參數。

請求標頭

此操作的實作只會使用所有操作通用的請求標頭。如需詳細資訊，請參閱[常見請求標頭](#)。

要求元素

此 操作的實作不會使用請求元素。

回應

回應標頭

此操作的實作只會使用大多數回應通用的回應標頭。如需詳細資訊，請參閱[常見回應標頭](#)。

回應元素

此 實作GET會傳回下列回應元素。

名稱	描述	必要
AbortIncompleteMultipartUpload	用於指定未完成分段上傳何時符合中止操作資格的容器。 子系：DaysAfterInitiation 類型：容器 Ancestor：Rule	是，如果沒有為規則指定其他動作
Date	您希望 Amazon S3 採取動作的日期。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 生命週期規則：根據特定日期 。 日期值必須符合 ISO 8601 格式。時間一律是午夜 UTC。 類型：字串 Ancestor：Expiration 或 Transition	是，如果 Days 和 ExpiredObjectDeleteMarker 不存在
Days	指定特定規則動作生效時，物件建立後的天數。物件的資格時間計算方式為建立時間 + 產生的時間四捨五入至隔天午夜 UTC 的天數。 類型：與 搭配使用時的非負整數Transition，與 搭配使用時為正整數Expiration。	是，如果 Date 和 ExpiredObjectDeleteMarker 不存在

名稱	描述	必要
Ancestor : Transition 或 Expiration		
DaysAfterInitiation	指定必須完成分段上傳時，啟動分段上傳後的天數。如果未在指定的天數之前完成，則會符合中止操作的資格，且 Amazon S3 會取消未完成的分段上傳。 類型：正整數 Ancestor : AbortIncompleteMultipartUpload	是，如果 Date 不存在

名稱	描述	必要
Expiration	此動作會指定物件生命週期內 Amazon S3 應採取適當過期動作的期間。過期動作只會發生在根據子項Date或Days元素中指定的期間符合資格的物件上。Amazon S3 採取的動作取決於儲存貯體是否已啟用版本控制。 • 如果從未在儲存貯體上啟用版本控制，Amazon S3 會永久刪除物件的唯一複本。 • 否則，如果您的儲存貯體已啟用版本控制（或版本控制已暫停），則動作僅適用於物件的目前版本。已啟用版本控制或暫停版本控制的儲存貯體可以有許多相同物件的版本：一個目前版本，以及零或多個非目前版本。 Amazon S3 不會刪除目前版本，而是新增刪除標記做為新的目前版本，使其成為非目前版本。	是，如果指定父標籤

 **Important**

如果儲存貯體的狀態已暫停版本控制，Amazon S3 會建立版本 ID 為的刪除標記null。如果您有版本 ID 為的版本null，則 Amazon S3 會覆寫該版本。

名稱	描述	必要
	Note 若要設定非目前物件的過期，您必須使用 NoncurrentVersionExpiration 動作。 類型：容器 子系：天或日期 Ancestor：規則	
ID	規則的唯一識別碼。該值不可超過 255 個字元。 類型：字串 Ancestor：規則	否
LifecycleConfiguration	生命週期規則的容器。最多可以新增 1000 個規則。 類型：容器 子系：規則 Ancestor：無	是

名稱	描述	必要
ExpiredObjectDeleteMarker	在版本控制儲存貯體（啟用版本控制或暫停版本控制儲存貯體）上，此元素會指出 Amazon S3 是否會刪除儲存貯體中任何過期的物件刪除標記。如需範例，請參閱《Amazon Simple Storage Service 使用者指南》中的範例 8：指定過期動作以移除過期的物件刪除標記。 類型：字串 有效值：true false（允許 false 值，但為無操作，如果值為 false，Amazon S3 不會採取動作） Ancestor：Expiration	是，如果 Date 和 Days 不存在
NoncurrentDays	指定物件在 Amazon S3 可以執行關聯動作之前為非最新版本的天數。如需計算非目前天數的資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的以天數為基礎的生命週期規則。 類型：與 搭配使用時的非負整數 NoncurrentVersionTransition，與 搭配使用時為正整數 NoncurrentVersionExpiration Ancestor：NoncurrentVersionExpiration 或 NoncurrentVersionTransition	是，只有在上階存在時

名稱	描述	必要
NoncurrentVersionExpiration	指定非目前物件版本的到期時間。過期時，Amazon S3 會永久刪除非目前的物件版本。 在已啟用（或暫停）版本控制的儲存貯體上設定此生命週期組態動作，以請求 Amazon S3 在物件生命週期內的特定期間內刪除非目前物件版本。 類型：容器 子系：NoncurrentDays Ancestor：規則	是，如果中不存在其他動作 Rule
NoncurrentVersionTransition	轉換規則的容器，描述非目前物件何時轉換至 STANDARD_IA、ONEZONE_IA 或 儲存體方案。 如果您的儲存貯體已啟用版本控制（或版本控制已暫停），您可以設定此動作，請求 Amazon S3 在物件生命週期的特定期間，將非目前物件版本轉換為儲存類別。 類型：容器 子系：NoncurrentDays 和 StorageClass Ancestor：規則	是，如果中不存在其他動作 Rule
Prefix	物件金鑰字首，識別套用規則的一或多個物件。 類型：字串 Ancestor：規則	是

名稱	描述	必要
Rule	生命週期規則的容器。 類型：容器 Ancestor：LifecycleConfiguration	是
Status	如果已啟用，Amazon S3 會依排程執行規則。 如果停用，Amazon S3 會忽略規則。 類型：字串 Ancestor：規則 有效值：啟用或停用	是
StorageClass	指定您要轉換物件的 Amazon S3 儲存體方案。 類型：字串 Ancestor：Transition NoncurrentVersionTransition 有效值：STANDARD_IA ONEZONE_IA	是

名稱	描述	必要
Transition	此動作會指定物件生命週期內 Amazon S3 應將物件轉換為 STANDARD_IA 、 ONEZONE_IA 或 儲存類別的期間。當此動作生效時，Amazon S3 執行的操作取決於儲存貯體是否已啟用版本控制。 • 如果從未在儲存貯體上啟用版本控制，Amazon S3 會將物件的唯一複本轉換為指定的儲存類別。 • 當您的儲存貯體已啟用版本控制（或暫停版本控制）時，Amazon S3 只會轉換規則中識別之物件的目前版本。  Note 啟用版本控制或暫停版本控制的儲存貯體可以包含物件的許多版本。此動作不會影響非目前物件版本。若要轉換非目前物件，您必須使用 NoncurrentVersionTransition 動作。 類型：容器 子系：天或日期，以及 StorageClass Ancestor：規則	是，如果中不存在其他動作 Rule

特殊錯誤

錯誤程式碼	說明	HTTP 狀態碼	SOAP 故障碼字首
NoSuchLifecycleConfiguration	生命週期組態不存在。	404 找不到	用戶端

如需 Amazon S3 錯誤和錯誤代碼清單的一般資訊，請參閱 [錯誤回應](#)。

範例

範例 1：擷取生命週期子資源

此範例是從指定儲存貯體擷取 lifecycle 子資源的 GET 請求，以及傳回生命週期組態的範例回應。

請求範例

```
GET /?lifecycle HTTP/1.1
Host: examplebucket.s3.amazonaws.com
x-amz-date: Thu, 15 Nov 2012 00:17:21 GMT
Authorization: signatureValue
```

回應範例

```
HTTP/1.1 200 OK
x-amz-id-2: ITnGT1y4RyTmXa3rPi4hklTXouTf0hccUjo0iCPjz6FnfIutBj3M7fPGlW02SEWp
x-amz-request-id: 51991C342C575321
Date: Thu, 15 Nov 2012 00:17:23 GMT
Server: AmazonS3
Content-Length: 358

<?xml version="1.0" encoding="UTF-8"?>
<LifecycleConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Rule>
    <ID>Archive and then delete rule</ID>
    <Prefix>projectdocs</Prefix>
    <Status>Enabled</Status>
    <Transition>
      <Days>30</Days>
```

```
<StorageClass>STANDARD_IA</StorageClass>
</Transition>
<Transition>
  <Days>365</Days>
  <StorageClass>GLACIER</StorageClass>
</Transition>
<Expiration>
  <Days>3650</Days>
</Expiration>
</Rule>
</LifecycleConfiguration>
```

相關資源

- [PutBucketLifecycleConfiguration](#)
- [DeleteBucketLifecycle](#)

文件歷史紀錄

下表說明 Amazon S3 開發人員指南的重要變更。如需有關此文件更新的通知，您可以訂閱 RSS 訂閱源。

變更	描述	日期
新開發人員指南	將所有非 API 內容，例如使用 S3 進行開發、程式碼範例、錯誤回應等，從 Amazon S3 API 參考指南移至此新的 Amazon S3 開發人員指南。如需詳細資訊，請參閱 Amazon S3 API 參考 。	2026 年 6 月 15 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。