aws

開發人員指南

Amazon Elastic Compute Cloud

Amazon Elastic Compute Cloud: 開發人員指南

Table of Contents

以程式設計方式存取 Amazon EC2

您可以使用 AWS Management Console 或程式設計界面建立和管理 Amazon EC2 資源。如需使用 Amazon EC2 主控台的相關資訊，請參閱 Amazon EC2 使用者指南。

運作方式

- Amazon EC2 端點
- 最終一致性
- 冪等
- 請求節流

程式設計界面

- AWS Command Line Interface (AWS CLI)
- AWS CloudFormation
- AWS 開發套件
- 低階 API

開始使用

- 程式碼範例
- Console-to-Code

監控

- AWS CloudTrail
- 監控請求

Amazon EC2 服務端點

端點是作為 AWS Web 服務進入點的 URL。Amazon EC2 支援下列端點類型：

- IPv4 端點

- 同時支援 IPv4 和 IPv6 的雙堆疊端點
- FIPS 端點

當您提出請求時，您可以指定要使用的端點和區域。如果您沒有指定端點，則預設使用 IPv4 端點。若要使用不同的端點類型，您必須在請求中將其指定。如需如何執行此作業的範例，請參閱 指定端點。

區域名稱	區域	端點	通訊協定	
美國東部 (俄亥俄)	us-east-2	ec2.us-east-2.amazonaws.com ec2-fips.us-east-2.amazonaws.com ec2.us-east-2.api.aws	HTTP 和 HTTPS HTTPS HTTPS	
美國東部 (維吉尼亞北部)	us-east-1	ec2.us-east-1.amazonaws.com ec2-fips.us-east-1.amazonaws.com ec2.us-east-1.api.aws	HTTP 和 HTTPS HTTPS HTTPS	
美國西部 (加利佛尼亞北部)	us-west-1	ec2.us-west-1.amazonaws.com ec2-fips.us-west-1.amazonaws.com ec2.us-west-1.api.aws	HTTP 和 HTTPS HTTPS HTTPS	
美國西部 (奧勒岡)	us-west-2	ec2.us-west-2.amazonaws.com ec2-fips.us-west-2.amazonaws.com ec2.us-west-2.api.aws	HTTP 和 HTTPS HTTPS HTTPS	
非洲 (開普敦)	af-south-1	ec2.af-south-1.amazonaws.com ec2.af-south-1.api.aws	HTTP 和 HTTPS HTTPS	

區域名稱	區域	端點	通訊協定	
亞太區域 (香港)	ap-east-1	ec2.ap-east-1.amazonaws.com ec2.ap-east-1.api.aws	HTTP 和 HTTPS HTTPS	
亞太區域 (海德拉巴)	ap-south-2	ec2.ap-south-2.amazonaws.com	HTTPS	
亞太區域 (雅加達)	ap-southeast-3	ec2.ap-southeast-3.amazonaws.com	HTTPS	
亞太區域 (墨爾本)	ap-southeast-4	ec2.ap-southeast-4.amazonaws.com	HTTPS	
亞太區域 (孟買)	ap-south-1	ec2.ap-south-1.amazonaws.com ec2.ap-south-1.api.aws	HTTP 和 HTTPS HTTPS	
亞太區域 (大阪)	ap-northeast-3	ec2.ap-northeast-3.amazonaws.com	HTTP 和 HTTPS	
亞太區域 (首爾)	ap-northeast-2	ec2.ap-northeast-2.amazonaws.com ec2.ap-northeast-2.api.aws	HTTP 和 HTTPS HTTPS	
亞太區域 (新加坡)	ap-southeast-1	ec2.ap-southeast-1.amazonaws.com ec2.ap-southeast-1.api.aws	HTTP 和 HTTPS HTTPS	

區域名稱	區域	端點	通訊協定
亞太區域 (雪梨)	ap-southeast-2	ec2.ap-southeast-2.amazonaws.com ec2.ap-southeast-2.api.aws	HTTP 和 HTTPS HTTPS
亞太區域 (東京)	ap-northeast-1	ec2.ap-northeast-1.amazonaws.com ec2.ap-northeast-1.api.aws	HTTP 和 HTTPS HTTPS
加拿大 (中部)	ca-central-1	ec2.ca-central-1.amazonaws.com ec2-fips.ca-central-1.amazonaws.com ec2.ca-central-1.api.aws	HTTP 和 HTTPS HTTPS HTTPS
加拿大西部 (卡加利)	ca-west-1	ec2.ca-west-1.amazonaws.com ec2-fips.ca-west-1.amazonaws.com	HTTPS HTTPS
歐洲 (法蘭克福)	eu-central-1	ec2.eu-central-1.amazonaws.com ec2.eu-central-1.api.aws	HTTP 和 HTTPS HTTPS
歐洲 (愛爾蘭)	eu-west-1	ec2.eu-west-1.amazonaws.com ec2.eu-west-1.api.aws	HTTP 和 HTTPS HTTPS
歐洲 (倫敦)	eu-west-2	ec2.eu-west-2.amazonaws.com ec2.eu-west-2.api.aws	HTTP 和 HTTPS HTTPS

區域名稱	區域	端點	通訊協定
歐洲 (米蘭)	eu-south-1	ec2.eu-south-1.amazonaws.com ec2.eu-south-1.api.aws	HTTP 和 HTTPS HTTPS
歐洲 (巴黎)	eu-west-3	ec2.eu-west-3.amazonaws.com ec2.eu-west-3.api.aws	HTTP 和 HTTPS HTTPS
歐洲 (西班牙)	eu-south-2	ec2.eu-south-2.amazonaws.com	HTTPS
歐洲 (斯德哥爾摩)	eu-north-1	ec2.eu-north-1.amazonaws.com ec2.eu-north-1.api.aws	HTTP 和 HTTPS HTTPS
歐洲 (蘇黎世)	eu-central-2	ec2.eu-central-2.amazonaws.com	HTTPS
以色列 (特拉維夫)	il-central-1	ec2.il-central-1.amazonaws.com	HTTPS
中東 (巴林)	me-south-1	ec2.me-south-1.amazonaws.com ec2.me-south-1.api.aws	HTTP 和 HTTPS HTTPS
中東 (阿拉伯聯合大公國)	me-central-1	ec2.me-central-1.amazonaws.com	HTTPS
南美洲 (聖保羅)	sa-east-1	ec2.sa-east-1.amazonaws.com ec2.sa-east-1.api.aws	HTTP 和 HTTPS HTTPS

區域名稱	區域	端點	通訊協定
AWS GovCloud (美國東部)	us-gov-east-1	ec2.us-gov-east-1.amazonaws.com ec2.us-gov-east-1.api.aws	HTTPS HTTPS
AWS GovCloud (美國西部)	us-gov-west-1	ec2.us-gov-west-1.amazonaws.com ec2.us-gov-west-1.api.aws	HTTPS HTTPS

如需區域的詳細資訊，請參閱 Amazon EC2 使用者指南中的區域和可用區域。如需 Amazon EC2 的端點清單，請參閱 Amazon Web Services 一般參考中的區域和端點。

主題

- IPv4 端點
- 雙堆疊 (IPv4 和 IPv6) 端點
- 指定端點

如需 FIPS 端點的詳細資訊，請參閱 Amazon Web Services 一般參考中的 FIPS 端點。

IPv4 端點

IPv4 端點僅支援 IPv4 流量。IPv4 端點適用於所有區域。

如果您指定一般端點、`ec2.amazonaws.com`，我們會使用 `us-east-1` 的端點。若要使用不同的區域，請指定其相關聯的端點。例如，如果您指定 `ec2.us-east-2.amazonaws.com` 做為端點，我們會將您的請求導向 `us-east-2` 端點。

IPv4 端點名稱使用以下命名慣例：

- *`service`*`.`*`region`*`.amazonaws.com`

例如，`eu-west-1` 區域的 IPv4 端點名稱是 `ec2.eu-west-1.amazonaws.com`。如需 Amazon EC2 的端點清單，請參閱 Amazon Web Services 一般參考中的區域和端點。

雙堆疊 (IPv4 和 IPv6) 端點

雙堆疊端點同時支援 IPv4 和 IPv6 流量。雙堆疊端點僅適用於下列區域：

- us-east-1— 美國東部 (維吉尼亞北部)
- us-east-2— 美國東部 (俄亥俄州)
- us-west-2— 美國西部 (奧勒岡)
- eu-west-1--歐洲 (愛爾蘭)
- ap-south-1— 亞太區域 (孟買)
- sa-east-1--南美洲 (聖保羅)
- us-gov-east-1—AWS GovCloud (美國東部)
- us-gov-west-1—AWS GovCloud (美國西部)

當您請求雙堆疊端點時，端點 URL 會解析為 IPv6 或 IPv4 地址，具體視您的網路和用戶端使用的通訊協定而異。

Amazon EC2 僅支援區域雙堆疊端點，這表示您必須將區域指定為端點名稱的一部分。雙堆疊端點名稱使用以下命名慣例：

- ec2.*region*.api.aws

例如，eu-west-1 區域的雙堆疊端點名稱是 ec2.eu-west-1.api.aws。如需 Amazon EC2 的端點清單，請參閱 Amazon Web Services 一般參考中的區域和端點。

指定端點

本節提供一些在提出請求時如何指定端點的範例。

AWS CLI

下列範例顯示如何使用指定「us-east-2區域」的端點 AWS CLI。

- 雙堆疊

```
aws ec2 describe-regions --region us-east-2 --endpoint-url https://ec2.us-east-2.api.aws
```

- IPv4

```
aws ec2 describe-regions --region us-east-2 --endpoint-url https://ec2.us-
east-2.amazonaws.com
```

AWS SDK for Java 2.x

下列範例顯示如何使用指定「us-east-2區域」的端點 AWS SDK for Java 2.x。

- **雙堆疊**

```
Ec2Client client = Ec2Client.builder()
    .region(Region.US_EAST_2)
    .endpointOverride(URI.create("https://ec2.us-east-2.api.aws"))
    .build();
```

- IPv4

```
Ec2Client client = Ec2Client.builder()
    .region(Region.US_EAST_2)
    .endpointOverride(URI.create("https://ec2.us-east-2.amazonaws.com"))
    .build();
```

AWS SDK for Java 1.x

下列範例顯示如何使用 AWS SDK for Java 1.x 指定「eu-west-1區域」的端點。

- **雙堆疊**

```
AmazonEC2 s3 = AmazonEC2ClientBuilder.standard()
     .withEndpointConfiguration(new EndpointConfiguration(
          "https://ec2.eu-west-1.api.aws",
          "eu-west-1"))
     .build();
```

- IPv4

```
AmazonEC2 s3 = AmazonEC2ClientBuilder.standard()
     .withEndpointConfiguration(new EndpointConfiguration(
          "https://ec2.eu-west-1.amazonaws.com",
          "eu-west-1"))
```

```
        .build();
```

AWS SDK for Go

下列範例顯示如何使用指定「us-east-1區域」的端點 AWS SDK for Go。

- **雙堆疊**

```
sess := session.Must(session.NewSession())
svc := ec2.New(sess, &aws.Config{
    Region: aws.String(endpoints.UsEast1RegionID),
    Endpoint: aws.String("https://ec2.us-east-1.api.aws")
})
```

- IPv4

```
sess := session.Must(session.NewSession())
svc := ec2.New(sess, &aws.Config{
    Region: aws.String(endpoints.UsEast1RegionID),
    Endpoint: aws.String("https://ec2.us-east-1.amazonaws.com")
})
```

Amazon EC2 API 中的最終一致性

由於系統支援 API 的分散式特性，Amazon EC2 API 遵循最終一致性模型。這表示您執行的 API 命令會影響 Amazon EC2 資源的結果，可能無法立即顯示您執行的所有後續命令。當您執行緊接在先前 API 命令之後的 API 命令時，應牢記這一點。

最終一致性會影響您管理資源的方式。例如，如果您執行命令來建立資源，其他命令最終會看到該資源。這表示，如果您執行命令來修改或描述您剛建立的資源，其 ID 可能不會在整個系統中傳播，而且您會收到回應資源不存在的錯誤訊息。

要管理最終一致性，您可以執行以下操作：

- 在執行命令修改資源之前，請先確認資源的狀態。使用指數輪詢演算法執行適當的Describe命令，以確保您有足夠的時間讓先前的指令在系統中傳播。若要這麼做，請重複執行Describe命令，從幾秒鐘的等待時間開始，並逐漸增加至五分鐘的等待時間。
- 在後續指令之間增加等待時間，即使指Describe令傳回準確的回應也是如此。從幾秒鐘的等待時間開始套用指數輪詢演算法，並逐漸增加至大約五分鐘的等待時間。

最終一致性錯誤示例

以下是由於最終一致性而可能遇到的錯誤代碼示例。

- `InvalidInstanceID.NotFound`

  如果您成功執行`RunInstances`命令，然後使用回應中提供的執行個體 ID 立即執行另一個命令`RunInstances`，則可能會傳回錯`InvalidInstanceID.NotFound`誤。這並不意味著該實例不存在。

  一些可能受到影響的特定命令包括：

  - `DescribeInstances`：若要確認執行個體的實際狀態，請使用指數輪詢演算法執行此命令。
  - `TerminateInstances`：若要確認執行個體的狀態，請先使用指數輪詢演算法執行`DescribeInstances`命令。

  > **Important**
  >
  > 如果執行後出`InvalidInstanceID.NotFound`現錯誤`TerminateInstances`，這並不表示執行個體已終止或將會終止。您的執行個體可能仍在執行中。這就是為什麼首先使用`DescribeInstances`.

- `InvalidGroup.NotFound`

  如果您成功執行`CreateSecurityGroup`命令，然後使用回應中提供的安全性群組識別碼立即執行另一個命令`CreateSecurityGroup`，則可能會傳回錯`InvalidGroup.NotFound`誤。若要確認安全性群組的狀態，請使用指數輪詢演算法執行`DescribeSecurityGroups`命令。

- `InstanceLimitExceeded`

  您所請求的執行個體數量超過目前執行個體限制允許的指定執行個體類型。如果您快速啟動和終止執行個體，可能會意外達到此限制，因為終止的執行個體在終止執行個體後會計入您的執行個體限制一段時間。

確保 Amazon EC2 API 請求中的冪等性

當您提出變動的 API 請求時，該請求一般會在操作的非同步工作流程完成之前傳回結果。即使請求已傳回結果，操作還是可能會在完成前就逾時或發生其他伺服器問題。這可能會讓您難以判斷請求是否成功，而且可能導致系統多次重試以確保操作能成功完成。但是，如果原始請求和後續的重試有成功，則操作會完成多次。這表示您可能會建立比預期更多的資源。

等冪性可確保 API 請求不會完成超過一次。使用等冪請求時，如果原始請求成功完成，則任何後續的重試都會成功完成，而不必執行任何進一步的動作。但是，結果可能包含更新的資訊，例如目前的建立狀態。

目錄

Amazon EC2 中的冪等性

下列 API 動作依預設為冪等，不需要額外的設定。默認情況下，相應的 AWS CLI 命令也支持冪等。

默認為冪等

- AssociateAddress
- CreateVpnConnection
- DisassociateAddress
- ReplaceNetworkAclAssociation
- TerminateInstances

以下 API 操作可選擇使用客戶端令牌支持冪等性。相應的 AWS CLI 命令還支持使用客戶端令牌的冪等性。客戶端令牌是一個唯一的，區分大小寫的字符串，最多 64 個 ASCII 字符。若要使用這些動作之一發出冪等 API 要求，請在要求中指定用戶端 Token。您不應該為其他 API 請求重複使用相同的客戶端令牌。如果您重試使用相同用戶端 Token 和相同參數成功完成的要求，則重試會成功，而不會執行任何進一步的動作。如果您使用相同的用戶端 Token 重試成功的要求，但有一或多個參數不同 (區域或可用區域除外)，則重試會失敗並顯示錯`IdempotentParameterMismatch`誤。

使用客戶端令牌的冪等

- AllocateHosts
- AllocateIpamPoolCidr
- AssociateClientVpnTargetNetwork
- AssociateIpamResourceDiscovery

- AttachVerifiedAccessTrustProvider
- AuthorizeClientVpnIngress
- CopyFpgaImage
- CopyImage
- CreateCapacityReservation
- CreateCapacityReservationFleet
- CreateClientVpnEndpoint
- CreateClientVpnRoute
- CreateEgressOnlyInternetGateway
- CreateFleet
- CreateFlowLogs
- CreateFpgaImage
- CreateInstanceConnectEndpoint
- CreateIpam
- CreateIpamPool
- CreateIpamResourceDiscovery
- CreateIpamScope
- CreateLaunchTemplate
- CreateLaunchTemplateVersion
- CreateManagedPrefixList
- CreateNatGateway
- CreateNetworkAcl
- CreateNetworkInsightsAccessScope
- CreateNetworkInsightsPath
- CreateNetworkInterface
- CreateReplaceRootVolumeTask
- CreateReservedInstancesListing
- CreateRouteTable
- CreateTrafficMirrorFilter

- CreateTrafficMirrorFilterRule
- CreateTrafficMirrorSession
- CreateTrafficMirrorTarget
- CreateVerifiedAccessEndpoint
- CreateVerifiedAccessGroup
- CreateVerifiedAccessInstance
- CreateVerifiedAccessTrustProvider
- CreateVolume
- CreateVpcEndpoint
- CreateVpcEndpointConnectionNotification
- CreateVpcEndpointServiceConfiguration
- DeleteVerifiedAccessEndpoint
- DeleteVerifiedAccessGroup
- DeleteVerifiedAccessInstance
- DeleteVerifiedAccessTrustProvider
- DetachVerifiedAccessTrustProvider
- ExportImage
- ImportImage
- ImportSnapshot
- ModifyInstanceCreditSpecification
- ModifyLaunchTemplate
- ModifyReservedInstances
- ModifyVerifiedAccessEndpoint
- ModifyVerifiedAccessEndpointPolicy
- ModifyVerifiedAccessGroup
- ModifyVerifiedAccessGroupPolicy
- ModifyVerifiedAccessInstance
- ModifyVerifiedAccessInstanceLoggingConfiguration
- ModifyVerifiedAccessTrustProvider
- ProvisionIpamPoolCidr

- PurchaseHostReservation
- RequestSpotFleet
- RequestSpotInstances
- RunInstances
- StartNetworkInsightsAccessScopeAnalysis
- StartNetworkInsightsAnalysis

冪等性的類型

- 區域 — 請求在每個區域中都是冪等的。但是，您可以在不同的區域中使用相同的請求，包括相同的客戶端令牌。
- 區域 — 請求在一個區域中的每個可用區域中都是冪等的。例如，如果您在相同區域中的兩個呼叫中指定相同的用戶端 Token，則呼叫會AllocateHosts在為AvailabilityZone參數指定不同的值時成功。

RunInstances 冪等

RunInstancesAPI 動作同時使用區域和區域等級。

使用的冪等性類型取決於您在 RunInstances API 請求中指定可用區域的方式。在下列情況下，請求會使用區域等級：

- 如果您使用位置資料類型中的AvailabilityZone參數明確指定可用區域
- 如果您使用參SubnetId數隱含地指定可用區域

如果您沒有明確或隱含地指定可用區域，請求會使用區域冪等性。

區域冪等性

區域冪等性可確保 RunInstances API 請求在區域中的每個可用區域中都是冪等的。這可確保具有相同用戶端權杖的要求在區域中的每個可用區域內只能完成一次。不過，相同的用戶端 Token 可用於在該區域的其他可用區域中啟動執行個體。

例如，如果您傳送冪等要求以在可用區域中啟動執行個體，然後在`us-east-1a`可用區域的要求中使用相同的用戶端 Token，我們會在每個`us-east-1b`可用區域中啟動執行個體。如果一或多個參數不同，在這些可用區域中使用相同用戶端 Token 的後續重試會順利傳回，而不執行任何進一步的動作，或者失敗並顯示錯`IdempotentParameterMismatch`誤。

区域冪等

區域冪等性確保 RunInstances API 請求在區域中是冪等的。這確保了具有相同客戶端令牌的請求只能在一個區域內完成一次。但是，具有相同客戶端令牌的完全相同請求可用於在不同區域中啟動實例。

例如，如果您傳送冪等要求以在 Region 中啟動執行個體，然後在 us-east-1 Region 的要求中使用相同的用戶端 Token，我們會在每個 eu-west-1 Region 中啟動執行個體。如果一個或多個參數不同，則在這些區域中使用相同客戶端令牌的後續重試會成功返回，而不執行任何進一步的操作，或者失敗並顯示錯IdempotentParameterMismatch誤。

> **Tip**
>
> 如果請求的區域中的其中一個可用區域不可用，則使用區域冪等的RunInstances 請求可能會失敗。若要利用 AWS 基礎結構提供的可用區域功能，建議您在啟動執行個體時使用區域等級。RunInstances 即使所要求區域中的另一個可用區域無法使用，使用區域等級並以可用區域為目標的要求仍會成功。

範例

AWS CLI 指令範例

若要使 AWS CLI 指令變為冪等，請新增選項--client-token。

示例 1：冪等

以下分配主機命令使用冪等性，因為它包含客戶端令牌。

```
aws ec2 allocate-hosts  --instance-type m5.large  --availability-zone eu-west-1a  --
auto-placement on  --quantity 1 --client-token 550e8400-e29b-41d4-a716-446655440000
```

示例 2：運行實例區域冪等

以下運行實例命令使用區域冪等性，因為它包含客戶端令牌，但不明確或隱式地指定可用區域。

```
aws ec2 run-instances --image-id ami-b232d0db --count 1 --key-name my-key-pair --
client-token 550e8400-e29b-41d4-a716-446655440000
```

示例 3：運行實例區域冪等

以下運行實例命令使用區域冪等性，因為它包含客戶端令牌和明確指定的可用區域。

```
aws ec2 run-instances  --placement "AvailabilityZone=us-east-1a" --image-id ami-
b232d0db --count 1 --key-name my-key-pair --client-token 550e8400-e29b-41d4-
a716-446655440000
```

API 要求範例

要使 API 請求冪等，請添加參數ClientToken。

示例 1：冪等

以下 AllocateHostsAPI 請求使用冪等性，因為它包含客戶端令牌。

```
https://ec2.amazonaws.com/?Action=AllocateHosts
&AvailabilityZone=us-east-1b
&InstanceType=m5.large
&Quantity=1
&AutoPlacement=off
&ClientToken=550e8400-e29b-41d4-a716-446655440000
&AUTHPARAMS
```

示例 2： RunInstances 區域冪等

以下 RunInstancesAPI 請求使用區域冪等性，因為它包含客戶端令牌，但未明確或隱含地指定可用區域。

```
https://ec2.amazonaws.com/?Action=RunInstances
&ImageId=ami-3ac33653
&MaxCount=1
&MinCount=1
&KeyName=my-key-pair
&ClientToken=550e8400-e29b-41d4-a716-446655440000
&AUTHPARAMS
```

示例 3： RunInstances 區域等級

以下 RunInstancesAPI 請求使用區域等級，因為它包含客戶端令牌和明確指定的可用區域。

```
https://ec2.amazonaws.com/?Action=RunInstances
&Placement.AvailabilityZone=us-east-1d
&ImageId=ami-3ac33653
&MaxCount=1
&MinCount=1
```

```
&KeyName=my-key-pair
&ClientToken=550e8400-e29b-41d4-a716-446655440000
&AUTHPARAMS
```

重試冪等請求的建議

下表顯示您可能會從等冪 API 請求得到的一些常見回應，並提供重試建議。

回應	建議	說明
200 (OK)	請勿重試	原始請求已成功完成。任何後續的重試都會成功傳回。
400 系列回應碼 (用戶端錯誤)	請勿重試	請求有下列方面的問題： • 其包含無效的參數或參數組合。 • 其使用您沒有許可的動作或資源。 • 其使用處於變更狀態過程的資源。 如果請求涉及處於變更狀態過程的資源，則重試請求有可能會成功。
500 系列回應碼 (伺服器錯誤)	重試	該錯誤是由 AWS 服務器端問題引起的，通常是短暫的。請使用適當的退避策略來重複請求。

Amazon EC2 API 的請求節流

Amazon EC2 會針對每個區域針對每個 AWS 帳戶進行調節 EC2 API 請求。我們這樣做是為了協助服務的效能，並確保所有 Amazon EC2 客戶的公平使用。節流可確保對 Amazon EC2 API 的呼叫不會超過允許的最大 API 請求限制。API 調用受到請求限制的限制，無論它們來自：

- 第三方應用程式

- 命令行工具
- Amazon EC2 主控台

如果超過 API 節流限制，則會收到RequestLimitExceeded錯誤代碼。

目錄

如何套用節流

Amazon EC2 使用令牌存儲桶算法來實現 API 節流。使用此算法，您的帳戶擁有一個存儲區，其中包含特定數量的令牌。存儲桶中的令牌數量代表您在任何給定秒鐘的節流限制。

Amazon EC2 實現了兩種類型的 API 節流：

API 節流類型

- 請求率限制
- 資源速率限制

請求率限制

使用請求速率限制，您可以限制發出的 API 請求數量。您每提出一個請求，就會從儲存貯體中刪除一個字符。例如，非變化 (Describe*) API 操作的存儲桶大小為 100 個令牌，因此您可以在一秒鐘內發出多達 100 個Describe*請求。如果您在一秒鐘內超過 100 個請求，則會受到限制，而第二個內的其餘要求會失敗。

時段會以設定的比率自動補充。如果存儲桶低於其最大容量，則每秒會向其添加一組數量的令牌，直到達到其最大容量為止。如果補充令牌到達時存儲桶已滿，則將其丟棄。值區的容量不能超過其最大數量的代幣。例如，非變異 (Describe*) API 操作的存儲桶大小為 100 個令牌，重新填充率為每秒 20 個令牌。如果您在一秒內發出 100 個 Describe* API 請求，則存儲桶會立即減少為零（0）令牌。然後每秒會重新填充 20 個代幣，直到達到 100 個代幣的最大容量為止。這意味著之前的空桶在 5 秒後達到其最大容量。

您不需要等待值區完全滿，就可以發出 API 請求。您可以在添加到存儲桶中時使用令牌。如果您立即使用補充令牌，則存儲桶未達到其最大容量。例如，控制台非變異操作的存儲桶大小為 100 個令牌，並且補充率為每秒 10 個令牌。如果您在一秒內發出 100 個 API 請求來耗盡值區，則可以繼續每秒發出 10 個 API 請求。只有當您每秒發出少於 10 個 API 請求時，值區才能重新填充到最大容量。

資源速率限制

部分 API 動作 (例如`RunInstances`和) 如下表所述`TerminateInstances`，除了要求速率限制外，還會使用資源速率限制。這些 API 動作具有單獨的資源令牌存儲桶，該值區會根據受請求影響的資源數量耗盡。與請求令牌存儲桶一樣，資源令牌存儲桶具有允許您突發的存儲桶最大值，以及可讓您在需要時保持穩定的請求速率的重新填充率。如果您超過 API 的特定值區限制，包括值區尚未重新填入以支援下一個 API 呼叫時，即使您尚未達到總 API 節流限制，API 的動作也會受到限制。

例如，的資源令牌存儲桶大小`RunInstances`為 1000 個令牌，重新填充率為每秒兩個令牌。因此，您可以使用任意數量的 API 請求立即啟動 1000 個執行個體，例如 1000 個執行個體的一個請求或 250 個執行個體的四個請求。資源令牌存儲桶為空後，您可以每秒啟動最多兩個實例，使用兩個實例的一個請求或一個實例的兩個請求。

如需詳細資訊，請參閱 資源令牌存儲桶大小和重新填充率。

節流限制

以下各節說明請求 Token 值區以及資源 Token 值區大小和重新填充率。

限制

- 請求令牌存儲桶大小和補充率
- 資源令牌存儲桶大小和重新填充率

請求令牌存儲桶大小和補充率

出於請求率限制目的，API 動作分為以下類別：

- 非變異動作 — 擷取資源相關資料的 API 動作。此類別通常包括所有`Describe*`動作`DescribeRouteTables`，例如`DescribeImages`、和`DescribeHosts`。這些 API 動作通常具有最高的 API 節流限制。
- 未過濾和未分頁的非變異動作 — 非變異 API 動作的特定子集，在未指定分頁或過濾器的情況下調用時，使用較小令牌存儲桶中的令牌。建議您使用分頁和篩選，以便從標準（較大的）令牌存儲桶中扣除令牌。

- 變更動作 — 建立、修改或刪除資源的 API 動作。此類別通常包括未歸類為非變異動作的所有 API 動作，例如`CreateVolume`、`ModifyHosts`和。`DeleteSnapshot`與非變異 API 呼叫相比，這些動作的節流限制較低。
- 資源密集型動作 — 更改需要花費最多時間並消耗最多資源來完成的 API 動作。與變異動作相比，這些動作的節流限制甚至更低。它們與其他變異動作分開進行節流。
- 主控台非變異動作 — 從 Amazon EC2 主控台呼叫的非變異 API 動作。這些 API 動作會與其他非變異 API 動作分開進行限制。
- 未分類操作 — 即使根據定義，這些 API 操作適合其他類別之一，也會收到自己的令牌存儲桶大小和補充率。

下表顯示了所有 AWS 區域的請求令牌存儲桶大小和重新填充率。

API 動作類別	動作	儲存貯體容量上限	鏟斗補充率
非變異動作	• `Describe*` • `Get*`	100	20
未過濾和未分頁的非變異動作	• `DescribeInstances` • `DescribeNetworkInterfaces` • `DescribeVolumes` • `DescribeInstanceStatus` • `DescribeSnapshots`	50	10

API 動作類別	動作	儲存貯體容量上限	鏟斗補充率
	• DescribeSecurityGroups • DescribeSpotInstanceRequests		
變異動作	未歸類為非變異動作的 API 動作。	200	5

API 動作類別	動作	儲存貯體容量上限	鏟斗補充率
資源密集型動作	• AuthorizeSecurityGroupIngress • CancelSpotInstanceRequests • CreateKeyPair • RequestSpotInstances • RevokeSecurityGroupIngress • CreateVpcPeeringConnection • AcceptVpcPeeringConnection • RejectVpcPeeringConnection • DeleteVpcPeeringConnection	50	5

API 動作類別	動作	儲存貯體容量上限	鏟斗補充率
控制台非變異動作	• Describe* • Get*	100	10
未分類動作	RunInstances	5	2
	StartInstances	5	2
	CreateVpcEndpoint	4	0.3
	ModifyVpcEndpoint	4	0.3
	DeleteVpcEndpoints	4	0.3
	AcceptVpcEndpointConnections	10	1
	RejectVpcEndpointConnections	10	1
	CreateVpcEndpointServiceConfiguration	10	1
	ModifyVpcEndpointServiceConfiguration	10	1

API 動作類別	動作	儲存貯體容量上限	鏟斗補充率
	DeleteVpcEndpointServiceConfigurations	10	1
	CreateDefaultVpc	1	1
	CreateDefaultSubnet	1	1
	MoveAddressToVpc	1	1
	RestoreAddressToClassic	1	1
	DescribeMovingAddresses	1	1
	AdvertiseByoipCidr	1	0.1
	ProvisionByoipCidr	1	0.1
	DescribeByoipCidrs	1	0.5
	DeprovisionByoipCidr	1	0.1
	WithdrawByoipCidr	1	0.1

API 動作類別	動作	儲存貯體容量上限	鏟斗補充率
	DescribeReservedInstancesOfferings	10	10
	PurchaseReservedInstancesOffering	5	5
	DescribeSpotFleetRequests	50	3
	DescribeSpotFleetInstances	100	5
	DescribeSpotFleetRequestHistory	100	5
	AssociateEnclaveCertificateIamRole	10	1
	DisassociateEnclaveCertificateIamRole	10	1
	GetAssociatedEnclaveCertificateIamRoles	10	1

API 動作類別	動作	儲存貯體容量上限	鏟斗補充率
	GetConsoleScreenshot	每個 帳戶 5 個 每個執行個體 2	每個 帳戶 5 個 每個執行個體 1

資源令牌存儲桶大小和重新填充率

下表列出使用資源速率限制之 API 動作的資源 Token 值區大小和重新填充率。

API 動作	儲存貯體容量上限	鏟斗補充率
RunInstances	1000	2
TerminateInstances	1000	20
StartInstances	1000	2
StopInstances	1000	20

監控 API 節流

您可以使用亞馬遜 CloudWatch 監控 Amazon EC2 API 呼叫，以及收集和追蹤 API 節流相關的指標。您也可以建立警示，以便在接近 API 節流限制時發出警告。如需詳細資訊，請參閱 使用 Amazon 監控亞馬遜 EC2 API 請求 CloudWatch。

重試和指數輪詢

您的應用程式可能需要重試 API 要求。例如：

- 若要檢查資源狀態是否有更新
- 列舉大量資源 (例如，您的所有磁碟區)
- 若要在要求失敗且發生伺服器錯誤 (5xx) 或節流錯誤後重試

不過，對於用戶端錯誤 (4xx)，您必須先修改要求以更正問題，然後再次嘗試要求。

資源狀態變更

在您開始輪詢以檢查狀態更新之前，請先提供可能完成要求的時間。例如，請等待幾分鐘，然後再檢查執行個體是否處於作用中狀態。當您開始輪詢時，請在連續要求之間使用適當的睡眠間隔，以降低 API 要求的速率。為了獲得最佳結果，請使用較長或可變的休眠間隔。

或者，您可以使 EventBridge 用 Amazon 通知您某些資源的狀態。例如，您可以使用 EC2 執行個體狀態變更通知事件通知您執行個體的狀態變更。如需詳細資訊，請參閱使用自動化 Amazon EC2 EventBridge。

重試

當您需要輪詢或重試 API 要求時，建議您使用指數輪詢演算法來計算 API 呼叫之間的睡眠間隔。指數退避的背後概念是，對於連續錯誤回應，讓重試之間的等待時間漸進拉長。您應該實作延遲間隔上限，以及重試次數上限。您也可以使用抖動 (隨機延遲) 來防止連續衝突。如需詳細資訊，請參閱逾時、重試和含抖動的輪詢。

每個 AWS SDK 都會實作自動重試邏輯。如需詳細資訊，請參閱 AWS SDK 和工具參考指南中的重試行為。

請求 提高限制

您可以要求提高. AWS 帳戶

要求存取此功能

1. 打開AWS Support 中心。
2. 選擇建立案例。
3. 選擇 帳戶和帳單。
4. 在「服務」中，選擇「一般信息」和「入門」。
5. 在「類別」中，選擇「使用 AWS 與服務」。
6. 選擇 Next step: Additional information (下一步：其他資訊)。
7. 對於 Subject (主旨)，請輸入 **Request an increase in my Amazon EC2 API throttling limits**。
8. 對於 Description (說明)，輸入 **Please increase the API throttling limits for my account. Related page: https://docs.aws.amazon.com/AWSEC2/latest/APIReference/throttling.html**。也包含下列資訊：
   - 您的使用案例的描述。
   - 您需要增加的區域。

- 發生峰值節流或使用量時的一小時窗口（以 UTC 為單位）（用於計算新的節流限制）。

9. 選擇下一步驟：立即解決或聯絡我們。
10. 在 [聯絡我們] 索引標籤上，選擇您偏好的聯絡語言和聯絡方式。
11. 選擇提交。

使用建立 Amazon EC2 資源 AWS CLI

您可以使用命令列殼層中的 AWS Command Line Interface (AWS CLI) 建立和管理 Amazon EC2 資源。提 AWS CLI 供直接存取的 API AWS 服務，例如 Amazon EC2。

如需 Amazon EC2 命令的語法和範例，請參閱AWS CLI 命令參考中的 ec2。 您也可以在 github 上的 aws-cli/awscli/示例/ec2 中找到這些示例。

進一步了解 AWS CLI

若要進一步了解 AWS CLI，請參閱下列資源：

- AWS Command Line Interface
- AWS Command Line Interface 第 2 版的使用者指南
- AWS Command Line Interface 第 1 版的使用者指南

使用建立 Amazon EC2 資源 AWS CloudFormation

Amazon EC2 整合了這項服務 AWS CloudFormation，可協助您建立資源模型和設定資 AWS 源，以減少建立和管理資源和基礎設施的時間。您可以建立描述所需資 AWS 源 (例如執行個體和子網路) 的範本，並為您 AWS CloudFormation 佈建和設定這些資源。

使用時 AWS CloudFormation，您可以重複使用範本，以一致且重複地設定 Amazon EC2 資源。描述您的資源一次，然後在多個區域中一遍又一遍地佈建相同 AWS 帳戶 的資源。

Amazon EC2 和 AWS CloudFormation 模板

若要佈建和設定 Amazon EC2 和相關服務的資源，您必須了解AWS CloudFormation 範本。範本是以 JSON 或 YAML 格式化的文本檔案。這些範本說明您將在 AWS CloudFormation 堆疊中佈建的資源。如果您不熟悉 JSON 或 YAML，可以使用 AWS CloudFormation 設計師來協助您開始 AWS CloudFormation 使用範本。如需詳細資訊，請參閱什麼是 AWS CloudFormation 設計師？ 在《AWS CloudFormation 使用者指南》中。

Amazon EC2 的資源

運算資源

- AWS::EC2::CapacityReservation
- AWS::EC2::CapacityReservation艦隊
- AWS::EC2::EC2Fleet
- AWS::EC2::EC2Fleet
- AWS::EC2::Host
- AWS::EC2::Instance
- AWS::EC2::InstanceConnect端點
- AWS::EC2::LaunchTemplate
- AWS::EC2::PlacementGroup
- AWS::EC2::SpotFleet

聯網資源

- AWS::EC2::CarrierGateway

- AWS::EC2::ClientVpnAuthorizationRule
- AWS::EC2::ClientVpn端點
- AWS::EC2::ClientVpn路線
- AWS::EC2::ClientVpnTargetNetworkAssociation
- AWS::EC2::CustomerGateway
- AWS::EC2::DHCPOptions
- AWS::EC2::EgressOnlyInternetGateway
- AWS::EC2::EIP
- AWS::EC2::EIPAssociation
- AWS::EC2::FlowLog
- AWS::EC2::GatewayRouteTableAssociation
- AWS::EC2::InternetGateway
- AWS::EC2::IPAM
- AWS::EC2::IPAMAllocation
- AWS::EC2::IPAMPool
- AWS::EC2::IPAMPoolCidr
- AWS::EC2::IPAMResourceDiscovery
- AWS::EC2::IPAMResourceDiscoveryAssociation
- AWS::EC2::IPAMScope
- AWS::EC2::LocalGateway路線
- AWS::EC2::LocalGatewayRouteTable
- AWS::EC2::LocalGatewayRouteTableVirtualInterfaceGroupAssociation
- AWS::EC2::LocalGatewayRouteTable虛擬私人網絡化
- AWS::EC2::NatGateway
- AWS::EC2::NetworkInterface
- AWS::EC2::NetworkInsightsAccessScope
- AWS::EC2::NetworkInsightsAccessScopeAnalysis
- AWS::EC2::NetworkInsights分析
- AWS::EC2::NetworkInsights路徑
- AWS::EC2::NetworkInterface附件

- AWS::EC2::NetworkInterface權限
- AWS::EC2::NetworkPerformanceMetricSubscription
- AWS::EC2::PrefixList
- AWS::EC2::Route
- AWS::EC2::RouteTable
- AWS::EC2::Subnet
- AWS::EC2::SubnetCidr阻止
- AWS::EC2::SubnetNetworkAclAssociation
- AWS::EC2::SubnetRouteTableAssociation
- AWS::EC2::TrafficMirror過濾器
- AWS::EC2::TrafficMirrorFilterRule
- AWS::EC2::TrafficMirror階段
- AWS::EC2::TrafficMirror目標
- AWS::EC2::TransitGateway
- AWS::EC2::TransitGateway附件
- AWS::EC2::TransitGatewayConnect
- AWS::EC2::TransitGatewayMulticastDomain
- AWS::EC2::TransitGatewayMulticastDomainAssociation
- AWS::EC2::TransitGatewayMulticastGroupMember
- AWS::EC2::TransitGatewayMulticastGroupSource
- AWS::EC2::TransitGatewayPeeringAttachment
- AWS::EC2::TransitGateway路線
- AWS::EC2::TransitGatewayRouteTable
- AWS::EC2::TransitGatewayRouteTableAssociation
- AWS::EC2::TransitGatewayRouteTablePropagation
- AWS::EC2::TransitGatewayVpcAttachment
- AWS::EC2::VPC
- AWS::EC2::VPCCidrBlock
- AWS:: EC2:: VPCDHCP OptionsAssociation
- AWS::EC2::VPCEndpoint

- AWS::EC2::VPCEndpointConnectionNotification
- AWS::EC2::VPCEndpointService
- AWS::EC2::VPCEndpointServicePermissions
- AWS::EC2::VPCGatewayAttachment
- AWS::EC2::VPCPeeringConnection
- AWS::EC2::VPNConnection
- AWS:: EC2:: 虛擬私人網絡 ConnectionRoute
- AWS::EC2::VPNGateway
- AWS:: EC2:: 虛擬私人網絡 GatewayRoutePropagation

安全性資源

- AWS::EC2::KeyPair
- AWS::EC2::NetworkAcl
- AWS::EC2::NetworkAcl入境
- AWS::EC2::SecurityGroup
- AWS::EC2::SecurityGroup出口
- AWS::EC2::SecurityGroup入口
- AWS::EC2::VerifiedAccess端點
- AWS::EC2::VerifiedAccess集團
- AWS::EC2::VerifiedAccess實例
- AWS::EC2::VerifiedAccessTrustProvider

儲存資源

- AWS::EC2::SnapshotBlockPublicAccess
- AWS::EC2::Volume
- AWS::EC2::VolumeAttachment

進一步了解 AWS CloudFormation

若要進一步了解 AWS CloudFormation，請參閱下列資源：

- AWS CloudFormation
- AWS CloudFormation 使用者指南

使用 AWS 開發套件建立 Amazon EC2 資源

AWS 為許多流行的編程語言提供軟件開發工具包（SDK）。SDK 通過提供以下內容使開發更有效率：

- 可以合併到應用程式中的預先建置元件和程式庫
- 特定於語言的工具，例如編譯器和調試器
- 服務要求的密碼編譯簽署
- 請求重試
- 錯誤回應處理

Amazon EC2 API 的程式碼範例

提供的程式碼範例會 AWS 示範如何使用 API 以及完成特定工作。如需 Amazon EC2 API 的範例，請參閱亞 Amazon EC2 的程式碼範例。如需其他範例，請參閱尋找 AWS SDK 或 github aws-doc-sdk-examples上的程式碼範例。

進一步了解 AWS 軟體開發套件

若要深入了解 AWS SDK，請參閱下列資源：

- AWS SDK 和工具參考指南
- 建置基礎的工具 AWS
- 什麼是軟體開發套件？

適用於 Amazon EC2 的低級 API

Amazon EC2 的低級別 API 是亞馬 Amazon EC2 的協議級界面。使用低級 API 時，您必須正確格式化每個 HTTPS 請求，並在每個請求中添加有效的數字簽名。如需詳細資訊，請參閱 Amazon EC2 API 參考中的向 Amazon EC2 API 發出請求。或者，您可以使用 AWS SDK，它代表您構建和簽署請求。如需詳細資訊，請參閱 使用 AWS 開發套件。

Amazon EC2 API 包含多個服務的動作和資料類型。若要檢視每個服務的動作，請參閱 Amazon EC2 API 參考中的以下頁面。

- AWS Client VPN 動作
- Amazon EBS 動作
- Amazon EC2 動作
- AWS Network Manager 動作
- AWS 硝基飛地行動
- AWS Outposts 動作
- AWS PrivateLink 動作
- 資源回收筒動作
- AWS Site-to-Site VPN動作
- AWS Transit Gateway 動作
- AWS Verified Access 動作
- 虛擬機器匯入/
- Amazon VPC 動作
- Amazon VPC IPAM 動作
- AWS Wavelength 動作

使用 Console-to-Code 為主控台動作產生程式碼

Console-to-Code 處於 Amazon EC2 的預覽版本中，且可能會有所變動。僅在美國東部 (維吉尼亞北部) 區域提供。

該主控台提供建立資源和測試原型的指導路徑。如果您想要大規模建立相同的資源，則需要自動化程式碼。Console-to-Code 是 Amazon EC2 主控台的一項功能，可協助您開始使用自動化程式碼。Console-to-Code 會記錄您的主控台動作，包括預設值和相容參數。然後，它會使用生成式 AI，為您想要的動作建議您偏好的 infrastructure-as-code (IaC) 格式的程式碼。您可以使用該程式碼作為起點，對其進行自訂以使其針對特定使用案例準備好投入生產。

使用 Console-to-Code 無需額外付費。

運作方式

Console-to-Code 可協助您開始使用自動化程式碼，如下所示：

1. 您可以在主控台中執行動作，例如啟動執行個體或啟用詳細監控。
2. Console-to-Code 可記錄您的所有操作，包括主控台提供的所有預設設定和相容參數。
3. 您可以選擇要在自動化指令碼中使用的動作。這些動作可以是變動或唯讀 (非變動) 動作，或這兩種類型的動作。
4. 控制台到代碼生成您想要的 infrastructure-as-code （ IAC ）格式的代碼，例如，. TypeScript
5. 您可複製程式碼以在程式碼開發工具中使用，或下載程式碼以進行共用。
6. 然後，您可以使用該程式碼作為自動化指令碼的起點。您需要驗證程式碼是否符合您的意圖，並且參數是否會如預期設定您的資源。您需要自訂程式碼，以使其針對您的使用案例準備好投入生產。一旦您對程式碼感到滿意，即可在自動化指令碼中使用它。

如需有關如何在 Amazon EC2 主控台中使用 Console-to-Code 的說明，請參閱使用 Console-to-Code。

限制

使用 Console-to-Code 時存在以下限制。

支援地區

目前僅在美國東部 (維吉尼亞北部) 區域提供。

支援的程式碼格式

控制台到代碼當前可以生成以下代碼格式 infrastructure-as-code （ IAC ）：

- CDK Java
- CDK Python
- CDK TypeScript
- CloudFormation JSON
- CloudFormation YAML

保留的動作

- 目前階段作業：只有在目前階段作業期間執行的動作才會顯示在「已錄製的動作」 表 不會保留先前工作階段期間採取的動作。
- 瀏覽器重新整理：重新整理瀏覽器標籤時，記錄的動作會遺失。
- 標籤隔離：「已記錄的動作」表格特定於執行動作的瀏覽器標籤。在某個標籤中執行的動作不會顯示在另一個標籤的「已記錄動作」表格中。

已記錄的動作表格

以下表格列出並說明了 Console-to-Code 主控台中已記錄的動作表格中的資料欄。

資料欄標題	描述
主控台頁面	執行動作的主控台頁面。
操作	API 操作。
類型	動作的類型。 • 變動：建立、修改或刪除資源的 API 動作。 • 唯讀：擷取資源相關資料的 API 動作 (通常是所有 `Describe*` 動作)。

資料欄標題	描述
CLI 命令	有關所採取動作的詳細資訊，包括參數和值。
建立時間	採取動作的時間。

使用 Console-to-Code

請依照下列說明，在 Amazon EC2 主控台中使用 Console-to-Code 產生程式碼。

若要檢視這些步驟的動畫，請參閱 檢視動畫：在 Amazon EC2 主控台中使用 Console-to-Code 產生程式碼。

使用 Console-to-Code 產生程式碼

1. 在美國東部 (維吉尼亞北部) 區域開啟 Amazon EC2 主控台，網址為 https://console.aws.amazon.com/ec2/home?region=us-east-1。

   > **Note**
   >
   > 主控台對程式碼提供預覽版本，目前僅在美國東部 (維吉尼亞北部) 區域提供。只會記錄在此區域執行的動作。

2. 使用主控台建立資源並測試原型。例如，使用主控台設定和啟動執行個體以及啟用詳細監控。

   Console-to-Code 會記錄您執行的每個動作。

3. 在左側導覽窗格中，選擇 Console-to-Code。
4. 在已記錄的動作表格中，檢閱已記錄的動作，並決定要包含哪些動作來產生程式碼。

   - 使用搜尋欄位，依特定主控台頁面或動作篩選表格。當您開始輸入時，即會篩選表格。
   - 使用類型下拉式清單，依所有動作、變動動作或唯讀動作進行篩選。

   > **Note**
   >
   > 僅列出在目前工作階段期間採取的動作。如需詳細資訊，請參閱 保留的動作。

5. 選取您需要產生程式碼的每個動作旁的核取方塊。

> **Note**
>
> 一次最多可以選取 5 個動作。

6. 選擇產生 {code} 程式碼按鈕。

   按鈕標籤預設為上次選取的程式碼格式。若要選取不同的程式碼格式，請選擇按鈕旁的箭頭。

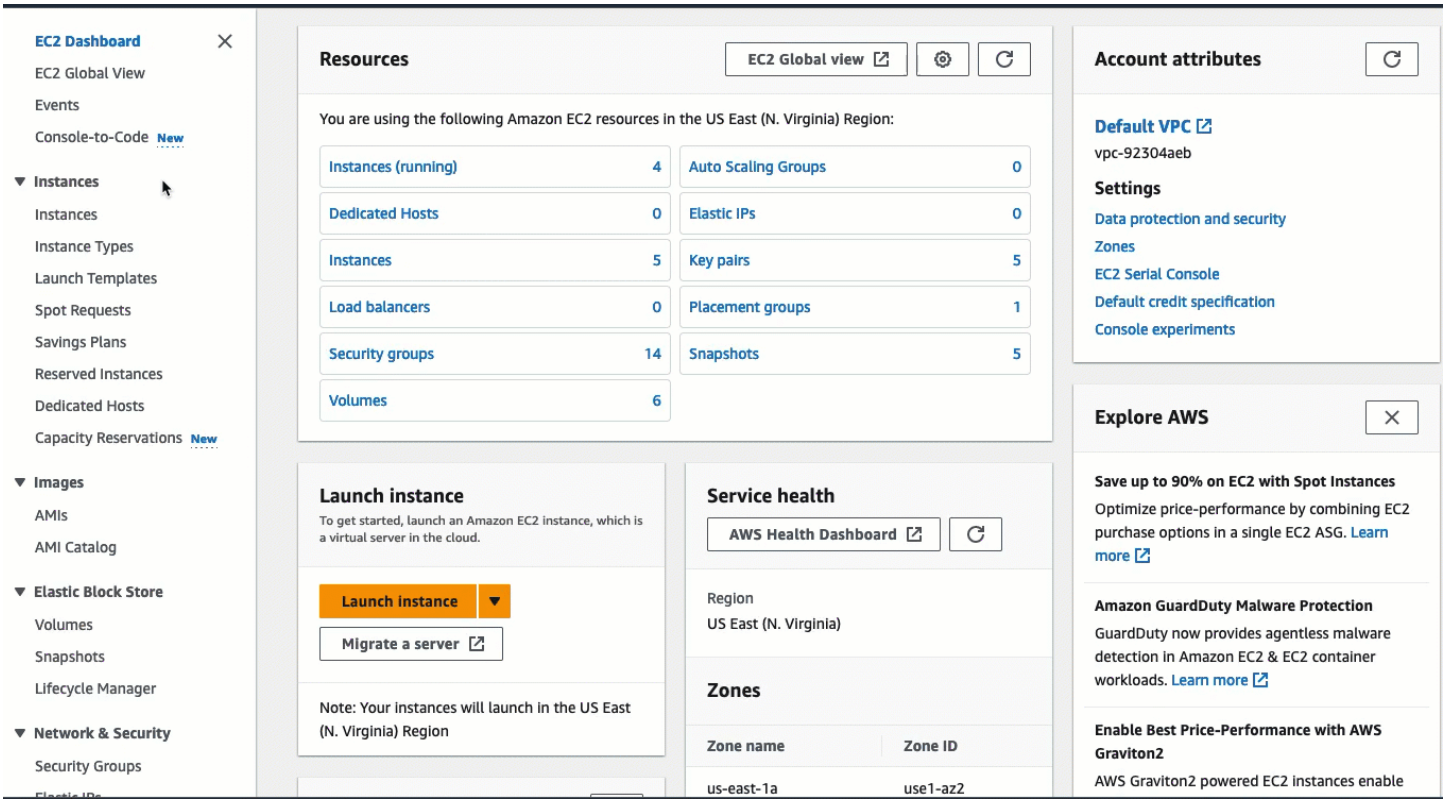
7. 在檢閱程式碼下，選擇複製以複製要在開發工具中使用的程式碼，或選擇下載以下載要共用的檔案。

8. 使用程式碼作為您的 infrastructure-as-code. 您需要自訂程式碼，以使其針對您的特定使用案例準備好投入生產。

> **Note**
>
> 如果您發現代碼尚未準備好生產，請向我們提供有關如何改進的反饋（請參閱以下步驟 9）。 AWS Support 無法協助您完成產生的程式碼或自訂程式碼開發。

9. (選用) 選擇「拇指向上」或「拇指向下」，讓我們知道 Console-to-Code 是否有幫助。如果您選擇「拇指向下」，則可選擇提供意見回饋，告訴我們如何改進程式碼以便為您提供更完善的協助。

檢視動畫：在 Amazon EC2 主控台中使用 Console-to-Code 產生程式碼



使用 AWS 開發套件的 Amazon EC2 程式碼範例

下列程式碼範例說明如何搭配 AWS 軟體開發套件 (SDK) 使用 Amazon EC2。

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和跨服務範例中查看內容中的動作。

Scenarios (案例) 是向您展示如何呼叫相同服務中的多個函數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 開發套件建立 Amazon EC2 資源。此主題也包含入門相關資訊和舊版 SDK 的詳細資訊。

開始使用

您好 Amazon EC2

下列程式碼範例示範如何開始使用 Amazon EC2。

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
namespace EC2Actions;

public class HelloEc2
{
    /// <summary>
    /// HelloEc2 lists the existing security groups for the default users.
    /// </summary>
    /// <param name="args">Command line arguments</param>
    /// <returns>A Task object.</returns>
    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon Elastic Compute Cloud (Amazon
 EC2).
```

```
        using var host =
 Microsoft.Extensions.Hosting.Host.CreateDefaultBuilder(args)
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonEC2>()
                .AddTransient<EC2Wrapper>()
            )
            .Build();

        // Now the client is available for injection.
        var ec2Client = host.Services.GetRequiredService<IAmazonEC2>();

        var request = new DescribeSecurityGroupsRequest
        {
            MaxResults = 10,
        };


        // Retrieve information about up to 10 Amazon EC2 security groups.
        var response = await ec2Client.DescribeSecurityGroupsAsync(request);

        // Now print the security groups returned by the call to
        // DescribeSecurityGroupsAsync.
        Console.WriteLine("Security Groups:");
        response.SecurityGroups.ForEach(group =>
        {
            Console.WriteLine($"Security group: {group.GroupName} ID:
 {group.GroupId}");
        });
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DescribeSecurityGroups中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

C MakeLists .txt 的 CMake 文件的代碼。

```
# 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 ec2)

# Set this project's name.
project("hello_ec2")

# 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_ec2.cpp)
```

```
target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

hello_ec2.cpp 來源檔案的程式碼。

```
#include <aws/core/Aws.h>
#include <aws/ec2/EC2Client.h>
#include <aws/ec2/model/DescribeInstancesRequest.h>
#include <iomanip>
#include <iostream>

/*
 *  A "Hello EC2" starter application which initializes an Amazon Elastic Compute
 Cloud (Amazon EC2) client and describes
 *  the Amazon EC2 instances.
 *
 *  main function
 *
 *  Usage: 'hello_ec2'
 *
 */

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";

        Aws::EC2::EC2Client ec2Client(clientConfig);
        Aws::EC2::Model::DescribeInstancesRequest request;
        bool header = false;
        bool done = false;
        while (!done) {
            auto outcome = ec2Client.DescribeInstances(request);
            if (outcome.IsSuccess()) {
                if (!header) {
                    std::cout << std::left <<
```

```
                            std::setw(48) << "Name" <<
                            std::setw(20) << "ID" <<
                            std::setw(25) << "Ami" <<
                            std::setw(15) << "Type" <<
                            std::setw(15) << "State" <<
                            std::setw(15) << "Monitoring" << std::endl;
                    header = true;
                }

                const std::vector<Aws::EC2::Model::Reservation> &reservations =
                        outcome.GetResult().GetReservations();

                for (const auto &reservation: reservations) {
                    const std::vector<Aws::EC2::Model::Instance> &instances =
                            reservation.GetInstances();
                    for (const auto &instance: instances) {
                        Aws::String instanceStateString =

Aws::EC2::Model::InstanceStateNameMapper::GetNameForInstanceStateName(
                                        instance.GetState().GetName());

                        Aws::String typeString =

Aws::EC2::Model::InstanceTypeMapper::GetNameForInstanceType(
                                        instance.GetInstanceType());

                        Aws::String monitorString =

Aws::EC2::Model::MonitoringStateMapper::GetNameForMonitoringState(
                                        instance.GetMonitoring().GetState());
                        Aws::String name = "Unknown";

                        const std::vector<Aws::EC2::Model::Tag> &tags =
instance.GetTags();
                        auto nameIter = std::find_if(tags.cbegin(), tags.cend(),
                                                     [](const
Aws::EC2::Model::Tag &tag) {
                                                         return tag.GetKey() ==
"Name";
                                                     });
                        if (nameIter != tags.cend()) {
                            name = nameIter->GetValue();
                        }
                        std::cout <<
```

```
                                std::setw(48) << name <<
                                std::setw(20) << instance.GetInstanceId() <<
                                std::setw(25) << instance.GetImageId() <<
                                std::setw(15) << typeString <<
                                std::setw(15) << instanceStateString <<
                                std::setw(15) << monitorString << std::endl;
                        }
                    }

                    if (!outcome.GetResult().GetNextToken().empty()) {
                        request.SetNextToken(outcome.GetResult().GetNextToken());
                    } else {
                        done = true;
                    }
                } else {
                    std::cerr << "Failed to describe EC2 instances:" <<
                              outcome.GetError().GetMessage() << std::endl;
                    result = 1;
                    break;
                }
            }
        }


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

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考DescribeSecurityGroups中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    public static void describeSecurityGroups(Ec2Client ec2, String groupId) {
```

```
        try {
            DescribeSecurityGroupsRequest request =
 DescribeSecurityGroupsRequest.builder()
                .groupIds(groupId)
                .build();

            // Use a paginator.
            DescribeSecurityGroupsIterable listGroups =
 ec2.describeSecurityGroupsPaginator(request);
            listGroups.stream()
                .flatMap(r -> r.securityGroups().stream())
                .forEach(group -> System.out
                    .println(" Group id: " +group.groupId() + " group name = " +
 group.groupName()));

        } catch (Ec2Exception 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DescribeSecurityGroups中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { DescribeSecurityGroupsCommand } from "@aws-sdk/client-ec2";

import { client } from "./libs/client.js";

// Call DescribeSecurityGroups and display the result.
export const main = async () => {
  try {
    const { SecurityGroups } = await client.send(
```

```
      new DescribeSecurityGroupsCommand({}),
    );

    const securityGroupList = SecurityGroups.slice(0, 9)
      .map((sg) => ` • ${sg.GroupId}: ${sg.GroupName}`)
      .join("\n");

    console.log(
      "Hello, Amazon EC2! Let's list up to 10 of your security groups:",
    );
    console.log(securityGroupList);
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考[DescribeSecurityGroups](#)中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在[AWS 設定和執行程式碼範例儲存庫](#)。

```
suspend fun describeEC2SecurityGroups(groupId: String) {
    val request =
        DescribeSecurityGroupsRequest {
            groupIds = listOf(groupId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->

        val response = ec2.describeSecurityGroups(request)
        response.securityGroups?.forEach { group ->
```

```
            println("Found Security Group with id ${group.groupId}, vpc id
 ${group.vpcId} and description ${group.description}")
        }
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK DescribeSecurityGroups中的 Kotlin API 參考。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import boto3


def hello_ec2(ec2_resource):
    """
    Use the AWS SDK for Python (Boto3) to create an Amazon Elastic Compute Cloud
    (Amazon EC2) resource and list the security groups in your account.
    This example uses the default settings specified in your shared credentials
    and config files.

    :param ec2_resource: A Boto3 EC2 ServiceResource object. This object is a
 high-level
                         resource that wraps the low-level EC2 service API.
    """
    print("Hello, Amazon EC2! Let's list up to 10 of your security groups:")
    for sg in ec2_resource.security_groups.limit(10):
        print(f"\t{sg.id}: {sg.group_name}")


if __name__ == "__main__":
    hello_ec2(boto3.resource("ec2"))
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeSecurityGroups中的 Python (博托 3) API 參考。

程式碼範例

- 使用 AWS 開發套件執行的 Amazon EC2 動作
  - 搭AcceptVpcPeeringConnection配 AWS 開發套件或 CLI 使用
  - 搭AllocateAddress配 AWS 開發套件或 CLI 使用
  - 搭AllocateHosts配 AWS 開發套件或 CLI 使用
  - 搭AssignPrivateIpAddresses配 AWS 開發套件或 CLI 使用
  - 搭AssociateAddress配 AWS 開發套件或 CLI 使用
  - 搭AssociateDhcpOptions配 AWS 開發套件或 CLI 使用
  - 搭AssociateRouteTable配 AWS 開發套件或 CLI 使用
  - 搭AttachInternetGateway配 AWS 開發套件或 CLI 使用
  - 搭AttachNetworkInterface配 AWS 開發套件或 CLI 使用
  - 搭AttachVolume配 AWS 開發套件或 CLI 使用
  - 搭AttachVpnGateway配 AWS 開發套件或 CLI 使用
  - 搭AuthorizeSecurityGroupEgress配 AWS 開發套件或 CLI 使用
  - 搭AuthorizeSecurityGroupIngress配 AWS 開發套件或 CLI 使用
  - 搭CancelCapacityReservation配 AWS 開發套件或 CLI 使用
  - 搭CancelImportTask配 AWS 開發套件或 CLI 使用
  - 搭CancelSpotFleetRequests配 AWS 開發套件或 CLI 使用
  - 搭CancelSpotInstanceRequests配 AWS 開發套件或 CLI 使用
  - 搭ConfirmProductInstance配 AWS 開發套件或 CLI 使用
  - 搭CopyImage配 AWS 開發套件或 CLI 使用
  - 搭CopySnapshot配 AWS 開發套件或 CLI 使用
  - 搭CreateCapacityReservation配 AWS 開發套件或 CLI 使用
  - 搭CreateCustomerGateway配 AWS 開發套件或 CLI 使用
  - 搭CreateDhcpOptions配 AWS 開發套件或 CLI 使用
  - 搭CreateFlowLogs配 AWS 開發套件或 CLI 使用
  - 搭CreateImage配 AWS 開發套件或 CLI 使用
  - 搭CreateInstanceExportTask配 AWS 開發套件或 CLI 使用

- 搭CreateInternetGateway配 AWS 開發套件或 CLI 使用
- 搭CreateKeyPair配 AWS 開發套件或 CLI 使用
- 搭CreateLaunchTemplate配 AWS 開發套件或 CLI 使用
- 搭CreateNetworkAcl配 AWS 開發套件或 CLI 使用
- 搭CreateNetworkAclEntry配 AWS 開發套件或 CLI 使用
- 搭CreateNetworkInterface配 AWS 開發套件或 CLI 使用
- 搭CreatePlacementGroup配 AWS 開發套件或 CLI 使用
- 搭CreateRoute配 AWS 開發套件或 CLI 使用
- 搭CreateRouteTable配 AWS 開發套件或 CLI 使用
- 搭CreateSecurityGroup配 AWS 開發套件或 CLI 使用
- 搭CreateSnapshot配 AWS 開發套件或 CLI 使用
- 搭CreateSpotDatafeedSubscription配 AWS 開發套件或 CLI 使用
- 搭CreateSubnet配 AWS 開發套件或 CLI 使用
- 搭CreateTags配 AWS 開發套件或 CLI 使用
- 搭CreateVolume配 AWS 開發套件或 CLI 使用
- 搭CreateVpc配 AWS 開發套件或 CLI 使用
- 搭CreateVpcEndpoint配 AWS 開發套件或 CLI 使用
- 搭CreateVpnConnection配 AWS 開發套件或 CLI 使用
- 搭CreateVpnConnectionRoute配 AWS 開發套件或 CLI 使用
- 搭CreateVpnGateway配 AWS 開發套件或 CLI 使用
- 搭DeleteCustomerGateway配 AWS 開發套件或 CLI 使用
- 搭DeleteDhcpOptions配 AWS 開發套件或 CLI 使用
- 搭DeleteFlowLogs配 AWS 開發套件或 CLI 使用
- 搭DeleteInternetGateway配 AWS 開發套件或 CLI 使用
- 搭DeleteKeyPair配 AWS 開發套件或 CLI 使用
- 搭DeleteLaunchTemplate配 AWS 開發套件或 CLI 使用
- 搭DeleteNetworkAcl配 AWS 開發套件或 CLI 使用
- 搭DeleteNetworkAclEntry配 AWS 開發套件或 CLI 使用
- 搭DeleteNetworkInterface配 AWS 開發套件或 CLI 使用
- 搭DeletePlacementGroup配 AWS 開發套件或 CLI 使用

- 搭DeleteRoute配 AWS 開發套件或 CLI 使用
- 搭DeleteRouteTable配 AWS 開發套件或 CLI 使用
- 搭DeleteSecurityGroup配 AWS 開發套件或 CLI 使用
- 搭DeleteSnapshot配 AWS 開發套件或 CLI 使用
- 搭DeleteSpotDatafeedSubscription配 AWS 開發套件或 CLI 使用
- 搭DeleteSubnet配 AWS 開發套件或 CLI 使用
- 搭DeleteTags配 AWS 開發套件或 CLI 使用
- 搭DeleteVolume配 AWS 開發套件或 CLI 使用
- 搭DeleteVpc配 AWS 開發套件或 CLI 使用
- 搭DeleteVpnConnection配 AWS 開發套件或 CLI 使用
- 搭DeleteVpnConnectionRoute配 AWS 開發套件或 CLI 使用
- 搭DeleteVpnGateway配 AWS 開發套件或 CLI 使用
- 搭DeregisterImage配 AWS 開發套件或 CLI 使用
- 搭DescribeAccountAttributes配 AWS 開發套件或 CLI 使用
- 搭DescribeAddresses配 AWS 開發套件或 CLI 使用
- 搭DescribeAvailabilityZones配 AWS 開發套件或 CLI 使用
- 搭DescribeBundleTasks配 AWS 開發套件或 CLI 使用
- 搭DescribeCapacityReservations配 AWS 開發套件或 CLI 使用
- 搭DescribeCustomerGateways配 AWS 開發套件或 CLI 使用
- 搭DescribeDhcpOptions配 AWS 開發套件或 CLI 使用
- 搭DescribeFlowLogs配 AWS 開發套件或 CLI 使用
- 搭DescribeHostReservationOfferings配 AWS 開發套件或 CLI 使用
- 搭DescribeHosts配 AWS 開發套件或 CLI 使用
- 搭DescribeIamInstanceProfileAssociations配 AWS 開發套件或 CLI 使用
- 搭DescribeIdFormat配 AWS 開發套件或 CLI 使用
- 搭DescribeIdentityIdFormat配 AWS 開發套件或 CLI 使用
- 搭DescribeImageAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeImages配 AWS 開發套件或 CLI 使用
- 搭DescribeImportImageTasks配 AWS 開發套件或 CLI 使用
- 搭DescribeImportSnapshotTasks配 AWS 開發套件或 CLI 使用

- 搭DescribeInstanceAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeInstanceStatus配 AWS 開發套件或 CLI 使用
- 搭DescribeInstanceTypes配 AWS 開發套件或 CLI 使用
- 搭DescribeInstances配 AWS 開發套件或 CLI 使用
- 搭DescribeInternetGateways配 AWS 開發套件或 CLI 使用
- 搭DescribeKeyPairs配 AWS 開發套件或 CLI 使用
- 搭DescribeNetworkAcls配 AWS 開發套件或 CLI 使用
- 搭DescribeNetworkInterfaceAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeNetworkInterfaces配 AWS 開發套件或 CLI 使用
- 搭DescribePlacementGroups配 AWS 開發套件或 CLI 使用
- 搭DescribePrefixLists配 AWS 開發套件或 CLI 使用
- 搭DescribeRegions配 AWS 開發套件或 CLI 使用
- 搭DescribeRouteTables配 AWS 開發套件或 CLI 使用
- 搭DescribeScheduledInstanceAvailability配 AWS 開發套件或 CLI 使用
- 搭DescribeScheduledInstances配 AWS 開發套件或 CLI 使用
- 搭DescribeSecurityGroups配 AWS 開發套件或 CLI 使用
- 搭DescribeSnapshotAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeSnapshots配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotDatafeedSubscription配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotFleetInstances配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotFleetRequestHistory配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotFleetRequests配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotInstanceRequests配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotPriceHistory配 AWS 開發套件或 CLI 使用
- 搭DescribeSubnets配 AWS 開發套件或 CLI 使用
- 搭DescribeTags配 AWS 開發套件或 CLI 使用
- 搭DescribeVolumeAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeVolumeStatus配 AWS 開發套件或 CLI 使用
- 搭DescribeVolumes配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcAttribute配 AWS 開發套件或 CLI 使用

- 搭DescribeVpcClassicLink配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcClassicLinkDnsSupport配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcEndpointServices配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcEndpoints配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcs配 AWS 開發套件或 CLI 使用
- 搭DescribeVpnConnections配 AWS 開發套件或 CLI 使用
- 搭DescribeVpnGateways配 AWS 開發套件或 CLI 使用
- 搭DetachInternetGateway配 AWS 開發套件或 CLI 使用
- 搭DetachNetworkInterface配 AWS 開發套件或 CLI 使用
- 搭DetachVolume配 AWS 開發套件或 CLI 使用
- 搭DetachVpnGateway配 AWS 開發套件或 CLI 使用
- 搭DisableVgwRoutePropagation配 AWS 開發套件或 CLI 使用
- 搭DisableVpcClassicLink配 AWS 開發套件或 CLI 使用
- 搭DisableVpcClassicLinkDnsSupport配 AWS 開發套件或 CLI 使用
- 搭DisassociateAddress配 AWS 開發套件或 CLI 使用
- 搭DisassociateRouteTable配 AWS 開發套件或 CLI 使用
- 搭EnableVgwRoutePropagation配 AWS 開發套件或 CLI 使用
- 搭EnableVolumeIo配 AWS 開發套件或 CLI 使用
- 搭EnableVpcClassicLink配 AWS 開發套件或 CLI 使用
- 搭EnableVpcClassicLinkDnsSupport配 AWS 開發套件或 CLI 使用
- 搭GetConsoleOutput配 AWS 開發套件或 CLI 使用
- 搭GetHostReservationPurchasePreview配 AWS 開發套件或 CLI 使用
- 搭GetPasswordData配 AWS 開發套件或 CLI 使用
- 搭ImportImage配 AWS 開發套件或 CLI 使用
- 搭ImportKeyPair配 AWS 開發套件或 CLI 使用
- 搭ImportSnapshot配 AWS 開發套件或 CLI 使用
- 搭ModifyCapacityReservation配 AWS 開發套件或 CLI 使用
- 搭ModifyHosts配 AWS 開發套件或 CLI 使用
- 搭ModifyIdFormat配 AWS 開發套件或 CLI 使用
- 搭ModifyImageAttribute配 AWS 開發套件或 CLI 使用

- 搭ModifyInstanceAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifyInstanceCreditSpecification配 AWS 開發套件或 CLI 使用
- 搭ModifyNetworkInterfaceAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifyReservedInstances配 AWS 開發套件或 CLI 使用
- 搭ModifySnapshotAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifySpotFleetRequest配 AWS 開發套件或 CLI 使用
- 搭ModifySubnetAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifyVolumeAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifyVpcAttribute配 AWS 開發套件或 CLI 使用
- 搭MonitorInstances配 AWS 開發套件或 CLI 使用
- 搭MoveAddressToVpc配 AWS 開發套件或 CLI 使用
- 搭PurchaseHostReservation配 AWS 開發套件或 CLI 使用
- 搭PurchaseScheduledInstances配 AWS 開發套件或 CLI 使用
- 搭RebootInstances配 AWS 開發套件或 CLI 使用
- 搭RegisterImage配 AWS 開發套件或 CLI 使用
- 搭RejectVpcPeeringConnection配 AWS 開發套件或 CLI 使用
- 搭ReleaseAddress配 AWS 開發套件或 CLI 使用
- 搭ReleaseHosts配 AWS 開發套件或 CLI 使用
- 搭ReplaceIamInstanceProfileAssociation配 AWS 開發套件或 CLI 使用
- 搭ReplaceNetworkAclAssociation配 AWS 開發套件或 CLI 使用
- 搭ReplaceNetworkAclEntry配 AWS 開發套件或 CLI 使用
- 搭ReplaceRoute配 AWS 開發套件或 CLI 使用
- 搭ReplaceRouteTableAssociation配 AWS 開發套件或 CLI 使用
- 搭ReportInstanceStatus配 AWS 開發套件或 CLI 使用
- 搭RequestSpotFleet配 AWS 開發套件或 CLI 使用
- 搭RequestSpotInstances配 AWS 開發套件或 CLI 使用
- 搭ResetImageAttribute配 AWS 開發套件或 CLI 使用
- 搭ResetInstanceAttribute配 AWS 開發套件或 CLI 使用
- 搭ResetNetworkInterfaceAttribute配 AWS 開發套件或 CLI 使用
- 搭ResetSnapshotAttribute配 AWS 開發套件或 CLI 使用

- 搭RevokeSecurityGroupEgress配 AWS 開發套件或 CLI 使用
- 搭RevokeSecurityGroupIngress配 AWS 開發套件或 CLI 使用
- 搭RunInstances配 AWS 開發套件或 CLI 使用
- 搭RunScheduledInstances配 AWS 開發套件或 CLI 使用
- 搭StartInstances配 AWS 開發套件或 CLI 使用
- 搭StopInstances配 AWS 開發套件或 CLI 使用
- 搭TerminateInstances配 AWS 開發套件或 CLI 使用
- 搭UnassignPrivateIpAddresses配 AWS 開發套件或 CLI 使用
- 搭UnmonitorInstances配 AWS 開發套件或 CLI 使用

- 使用 AWS 開發套件的 Amazon EC2 案例
  - 使用 AWS SDK 建置及管理彈性服務
  - 使用開 AWS 發套件開始使用 Amazon EC2 執行個體

使用 AWS 開發套件執行的 Amazon EC2 動作

下列程式碼範例示範如何使用 AWS 開發套件執行個別 Amazon EC2 動作。這些摘錄會呼叫 Amazon EC2 API，是必須在內容中執行之大型程式的程式碼摘錄。每個範例都包含一個連結 GitHub，您可以在其中找到設定和執行程式碼的指示。

下列範例僅包含最常使用的動作。如需完整清單，請參閱 Amazon Elastic Compute Cloud (Amazon EC2) 參考。

範例

- 搭AcceptVpcPeeringConnection配 AWS 開發套件或 CLI 使用
- 搭AllocateAddress配 AWS 開發套件或 CLI 使用
- 搭AllocateHosts配 AWS 開發套件或 CLI 使用
- 搭AssignPrivateIpAddresses配 AWS 開發套件或 CLI 使用
- 搭AssociateAddress配 AWS 開發套件或 CLI 使用
- 搭AssociateDhcpOptions配 AWS 開發套件或 CLI 使用
- 搭AssociateRouteTable配 AWS 開發套件或 CLI 使用
- 搭AttachInternetGateway配 AWS 開發套件或 CLI 使用
- 搭AttachNetworkInterface配 AWS 開發套件或 CLI 使用

- 搭AttachVolume配 AWS 開發套件或 CLI 使用
- 搭AttachVpnGateway配 AWS 開發套件或 CLI 使用
- 搭AuthorizeSecurityGroupEgress配 AWS 開發套件或 CLI 使用
- 搭AuthorizeSecurityGroupIngress配 AWS 開發套件或 CLI 使用
- 搭CancelCapacityReservation配 AWS 開發套件或 CLI 使用
- 搭CancelImportTask配 AWS 開發套件或 CLI 使用
- 搭CancelSpotFleetRequests配 AWS 開發套件或 CLI 使用
- 搭CancelSpotInstanceRequests配 AWS 開發套件或 CLI 使用
- 搭ConfirmProductInstance配 AWS 開發套件或 CLI 使用
- 搭CopyImage配 AWS 開發套件或 CLI 使用
- 搭CopySnapshot配 AWS 開發套件或 CLI 使用
- 搭CreateCapacityReservation配 AWS 開發套件或 CLI 使用
- 搭CreateCustomerGateway配 AWS 開發套件或 CLI 使用
- 搭CreateDhcpOptions配 AWS 開發套件或 CLI 使用
- 搭CreateFlowLogs配 AWS 開發套件或 CLI 使用
- 搭CreateImage配 AWS 開發套件或 CLI 使用
- 搭CreateInstanceExportTask配 AWS 開發套件或 CLI 使用
- 搭CreateInternetGateway配 AWS 開發套件或 CLI 使用
- 搭CreateKeyPair配 AWS 開發套件或 CLI 使用
- 搭CreateLaunchTemplate配 AWS 開發套件或 CLI 使用
- 搭CreateNetworkAcl配 AWS 開發套件或 CLI 使用
- 搭CreateNetworkAclEntry配 AWS 開發套件或 CLI 使用
- 搭CreateNetworkInterface配 AWS 開發套件或 CLI 使用
- 搭CreatePlacementGroup配 AWS 開發套件或 CLI 使用
- 搭CreateRoute配 AWS 開發套件或 CLI 使用
- 搭CreateRouteTable配 AWS 開發套件或 CLI 使用
- 搭CreateSecurityGroup配 AWS 開發套件或 CLI 使用
- 搭CreateSnapshot配 AWS 開發套件或 CLI 使用
- 搭CreateSpotDatafeedSubscription配 AWS 開發套件或 CLI 使用
- 搭CreateSubnet配 AWS 開發套件或 CLI 使用

- 搭CreateTags配 AWS 開發套件或 CLI 使用
- 搭CreateVolume配 AWS 開發套件或 CLI 使用
- 搭CreateVpc配 AWS 開發套件或 CLI 使用
- 搭CreateVpcEndpoint配 AWS 開發套件或 CLI 使用
- 搭CreateVpnConnection配 AWS 開發套件或 CLI 使用
- 搭CreateVpnConnectionRoute配 AWS 開發套件或 CLI 使用
- 搭CreateVpnGateway配 AWS 開發套件或 CLI 使用
- 搭DeleteCustomerGateway配 AWS 開發套件或 CLI 使用
- 搭DeleteDhcpOptions配 AWS 開發套件或 CLI 使用
- 搭DeleteFlowLogs配 AWS 開發套件或 CLI 使用
- 搭DeleteInternetGateway配 AWS 開發套件或 CLI 使用
- 搭DeleteKeyPair配 AWS 開發套件或 CLI 使用
- 搭DeleteLaunchTemplate配 AWS 開發套件或 CLI 使用
- 搭DeleteNetworkAcl配 AWS 開發套件或 CLI 使用
- 搭DeleteNetworkAclEntry配 AWS 開發套件或 CLI 使用
- 搭DeleteNetworkInterface配 AWS 開發套件或 CLI 使用
- 搭DeletePlacementGroup配 AWS 開發套件或 CLI 使用
- 搭DeleteRoute配 AWS 開發套件或 CLI 使用
- 搭DeleteRouteTable配 AWS 開發套件或 CLI 使用
- 搭DeleteSecurityGroup配 AWS 開發套件或 CLI 使用
- 搭DeleteSnapshot配 AWS 開發套件或 CLI 使用
- 搭DeleteSpotDatafeedSubscription配 AWS 開發套件或 CLI 使用
- 搭DeleteSubnet配 AWS 開發套件或 CLI 使用
- 搭DeleteTags配 AWS 開發套件或 CLI 使用
- 搭DeleteVolume配 AWS 開發套件或 CLI 使用
- 搭DeleteVpc配 AWS 開發套件或 CLI 使用
- 搭DeleteVpnConnection配 AWS 開發套件或 CLI 使用
- 搭DeleteVpnConnectionRoute配 AWS 開發套件或 CLI 使用
- 搭DeleteVpnGateway配 AWS 開發套件或 CLI 使用
- 搭DeregisterImage配 AWS 開發套件或 CLI 使用

- 搭DescribeAccountAttributes配 AWS 開發套件或 CLI 使用
- 搭DescribeAddresses配 AWS 開發套件或 CLI 使用
- 搭DescribeAvailabilityZones配 AWS 開發套件或 CLI 使用
- 搭DescribeBundleTasks配 AWS 開發套件或 CLI 使用
- 搭DescribeCapacityReservations配 AWS 開發套件或 CLI 使用
- 搭DescribeCustomerGateways配 AWS 開發套件或 CLI 使用
- 搭DescribeDhcpOptions配 AWS 開發套件或 CLI 使用
- 搭DescribeFlowLogs配 AWS 開發套件或 CLI 使用
- 搭DescribeHostReservationOfferings配 AWS 開發套件或 CLI 使用
- 搭DescribeHosts配 AWS 開發套件或 CLI 使用
- 搭DescribeIamInstanceProfileAssociations配 AWS 開發套件或 CLI 使用
- 搭DescribeIdFormat配 AWS 開發套件或 CLI 使用
- 搭DescribeIdentityIdFormat配 AWS 開發套件或 CLI 使用
- 搭DescribeImageAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeImages配 AWS 開發套件或 CLI 使用
- 搭DescribeImportImageTasks配 AWS 開發套件或 CLI 使用
- 搭DescribeImportSnapshotTasks配 AWS 開發套件或 CLI 使用
- 搭DescribeInstanceAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeInstanceStatus配 AWS 開發套件或 CLI 使用
- 搭DescribeInstanceTypes配 AWS 開發套件或 CLI 使用
- 搭DescribeInstances配 AWS 開發套件或 CLI 使用
- 搭DescribeInternetGateways配 AWS 開發套件或 CLI 使用
- 搭DescribeKeyPairs配 AWS 開發套件或 CLI 使用
- 搭DescribeNetworkAcls配 AWS 開發套件或 CLI 使用
- 搭DescribeNetworkInterfaceAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeNetworkInterfaces配 AWS 開發套件或 CLI 使用
- 搭DescribePlacementGroups配 AWS 開發套件或 CLI 使用
- 搭DescribePrefixLists配 AWS 開發套件或 CLI 使用
- 搭DescribeRegions配 AWS 開發套件或 CLI 使用
- 搭DescribeRouteTables配 AWS 開發套件或 CLI 使用

- 搭DescribeScheduledInstanceAvailability配 AWS 開發套件或 CLI 使用
- 搭DescribeScheduledInstances配 AWS 開發套件或 CLI 使用
- 搭DescribeSecurityGroups配 AWS 開發套件或 CLI 使用
- 搭DescribeSnapshotAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeSnapshots配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotDatafeedSubscription配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotFleetInstances配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotFleetRequestHistory配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotFleetRequests配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotInstanceRequests配 AWS 開發套件或 CLI 使用
- 搭DescribeSpotPriceHistory配 AWS 開發套件或 CLI 使用
- 搭DescribeSubnets配 AWS 開發套件或 CLI 使用
- 搭DescribeTags配 AWS 開發套件或 CLI 使用
- 搭DescribeVolumeAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeVolumeStatus配 AWS 開發套件或 CLI 使用
- 搭DescribeVolumes配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcAttribute配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcClassicLink配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcClassicLinkDnsSupport配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcEndpointServices配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcEndpoints配 AWS 開發套件或 CLI 使用
- 搭DescribeVpcs配 AWS 開發套件或 CLI 使用
- 搭DescribeVpnConnections配 AWS 開發套件或 CLI 使用
- 搭DescribeVpnGateways配 AWS 開發套件或 CLI 使用
- 搭DetachInternetGateway配 AWS 開發套件或 CLI 使用
- 搭DetachNetworkInterface配 AWS 開發套件或 CLI 使用
- 搭DetachVolume配 AWS 開發套件或 CLI 使用
- 搭DetachVpnGateway配 AWS 開發套件或 CLI 使用
- 搭DisableVgwRoutePropagation配 AWS 開發套件或 CLI 使用
- 搭DisableVpcClassicLink配 AWS 開發套件或 CLI 使用

- 搭DisableVpcClassicLinkDnsSupport配 AWS 開發套件或 CLI 使用
- 搭DisassociateAddress配 AWS 開發套件或 CLI 使用
- 搭DisassociateRouteTable配 AWS 開發套件或 CLI 使用
- 搭EnableVgwRoutePropagation配 AWS 開發套件或 CLI 使用
- 搭EnableVolumeIo配 AWS 開發套件或 CLI 使用
- 搭EnableVpcClassicLink配 AWS 開發套件或 CLI 使用
- 搭EnableVpcClassicLinkDnsSupport配 AWS 開發套件或 CLI 使用
- 搭GetConsoleOutput配 AWS 開發套件或 CLI 使用
- 搭GetHostReservationPurchasePreview配 AWS 開發套件或 CLI 使用
- 搭GetPasswordData配 AWS 開發套件或 CLI 使用
- 搭ImportImage配 AWS 開發套件或 CLI 使用
- 搭ImportKeyPair配 AWS 開發套件或 CLI 使用
- 搭ImportSnapshot配 AWS 開發套件或 CLI 使用
- 搭ModifyCapacityReservation配 AWS 開發套件或 CLI 使用
- 搭ModifyHosts配 AWS 開發套件或 CLI 使用
- 搭ModifyIdFormat配 AWS 開發套件或 CLI 使用
- 搭ModifyImageAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifyInstanceAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifyInstanceCreditSpecification配 AWS 開發套件或 CLI 使用
- 搭ModifyNetworkInterfaceAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifyReservedInstances配 AWS 開發套件或 CLI 使用
- 搭ModifySnapshotAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifySpotFleetRequest配 AWS 開發套件或 CLI 使用
- 搭ModifySubnetAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifyVolumeAttribute配 AWS 開發套件或 CLI 使用
- 搭ModifyVpcAttribute配 AWS 開發套件或 CLI 使用
- 搭MonitorInstances配 AWS 開發套件或 CLI 使用
- 搭MoveAddressToVpc配 AWS 開發套件或 CLI 使用
- 搭PurchaseHostReservation配 AWS 開發套件或 CLI 使用
- 搭PurchaseScheduledInstances配 AWS 開發套件或 CLI 使用

- 搭RebootInstances配 AWS 開發套件或 CLI 使用
- 搭RegisterImage配 AWS 開發套件或 CLI 使用
- 搭RejectVpcPeeringConnection配 AWS 開發套件或 CLI 使用
- 搭ReleaseAddress配 AWS 開發套件或 CLI 使用
- 搭ReleaseHosts配 AWS 開發套件或 CLI 使用
- 搭ReplaceIamInstanceProfileAssociation配 AWS 開發套件或 CLI 使用
- 搭ReplaceNetworkAclAssociation配 AWS 開發套件或 CLI 使用
- 搭ReplaceNetworkAclEntry配 AWS 開發套件或 CLI 使用
- 搭ReplaceRoute配 AWS 開發套件或 CLI 使用
- 搭ReplaceRouteTableAssociation配 AWS 開發套件或 CLI 使用
- 搭ReportInstanceStatus配 AWS 開發套件或 CLI 使用
- 搭RequestSpotFleet配 AWS 開發套件或 CLI 使用
- 搭RequestSpotInstances配 AWS 開發套件或 CLI 使用
- 搭ResetImageAttribute配 AWS 開發套件或 CLI 使用
- 搭ResetInstanceAttribute配 AWS 開發套件或 CLI 使用
- 搭ResetNetworkInterfaceAttribute配 AWS 開發套件或 CLI 使用
- 搭ResetSnapshotAttribute配 AWS 開發套件或 CLI 使用
- 搭RevokeSecurityGroupEgress配 AWS 開發套件或 CLI 使用
- 搭RevokeSecurityGroupIngress配 AWS 開發套件或 CLI 使用
- 搭RunInstances配 AWS 開發套件或 CLI 使用
- 搭RunScheduledInstances配 AWS 開發套件或 CLI 使用
- 搭StartInstances配 AWS 開發套件或 CLI 使用
- 搭StopInstances配 AWS 開發套件或 CLI 使用
- 搭TerminateInstances配 AWS 開發套件或 CLI 使用
- 搭UnassignPrivateIpAddresses配 AWS 開發套件或 CLI 使用
- 搭UnmonitorInstances配 AWS 開發套件或 CLI 使用

搭**AcceptVpcPeeringConnection**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AcceptVpcPeeringConnection。

CLI

AWS CLI

接受 VPC 對等連線

此範例接受指定的 VPC 對等連線要求。

命令：

```
aws ec2 accept-vpc-peering-connection --vpc-peering-connection-id pcx-1a2b3c4d
```

輸出：

```
{
  "VpcPeeringConnection": {
    "Status": {
      "Message": "Provisioning",
      "Code": "provisioning"
    },
    "Tags": [],
    "AccepterVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-44455566",
      "CidrBlock": "10.0.1.0/28"
    },
    "VpcPeeringConnectionId": "pcx-1a2b3c4d",
    "RequesterVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-111abc45",
      "CidrBlock": "10.0.0.0/28"
    }
  }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AcceptVpcPeeringConnection中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會核准請求 VpcPeeringConnectionId 的

```
Approve-EC2VpcPeeringConnection -VpcPeeringConnectionId pcx-1dfad234b56ff78be
```

輸出：

```
AccepterVpcInfo        : Amazon.EC2.Model.VpcPeeringConnectionVpcInfo
ExpirationTime         : 1/1/0001 12:00:00 AM
RequesterVpcInfo       : Amazon.EC2.Model.VpcPeeringConnectionVpcInfo
Status                 : Amazon.EC2.Model.VpcPeeringConnectionStateReason
Tags                   : {}
VpcPeeringConnectionId : pcx-1dfad234b56ff78be
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AcceptVpcPeeringConnection式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭AllocateAddress配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AllocateAddress。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Allocate an Elastic IP address.
    /// </summary>
```

```
    /// <returns>The allocation Id of the allocated address.</returns>
    public async Task<string> AllocateAddress()
    {
        var request = new AllocateAddressRequest();

        var response = await _amazonEC2.AllocateAddressAsync(request);
        return response.AllocationId;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考AllocateAddress中的。

Bash

AWS CLI 與 bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
#############################################################################
# function ec2_allocate_address
#
# This function allocates an Elastic IP address for use with Amazon Elastic
 Compute Cloud (Amazon EC2) instances in a specific AWS Region.
#
# Parameters:
#       -d domain - The domain for the Elastic IP address (either 'vpc' or
 'standard').
#
# Returns:
#       The allocated Elastic IP address, or an error message if the operation
 fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
#############################################################################
function ec2_allocate_address() {
  local domain response
```

```
 # Function to display usage information
 function usage() {
   echo "function ec2_allocate_address"
   echo "Allocates an Elastic IP address for use with Amazon Elastic Compute
Cloud (Amazon EC2) instances in a specific AWS Region."
   echo "  -d domain - The domain for the Elastic IP address (either 'vpc' or
'standard')."
   echo ""
 }

 # Parse the command-line arguments
 while getopts "d:h" option; do
   case "${option}" in
     d) domain="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 # Validate the input parameters
 if [[ -z "$domain" ]]; then
   errecho "ERROR: You must provide a domain with the -d parameter (either 'vpc'
or 'standard')."
   return 1
 fi

 if [[ "$domain" != "vpc" && "$domain" != "standard" ]]; then
   errecho "ERROR: Invalid domain value. Must be either 'vpc' or 'standard'."
   return 1
 fi

 # Allocate the Elastic IP address
 response=$(aws ec2 allocate-address \
   --domain "$domain" \
   --query "[PublicIp,AllocationId]" \
```

```
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports allocate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}
```

此範例中使用的公用程式函數。

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
########################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
```

```
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AllocateAddress中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::AllocateAddressRequest request;
    request.SetDomain(Aws::EC2::Model::DomainType::vpc);

    const Aws::EC2::Model::AllocateAddressOutcome outcome =
            ec2Client.AllocateAddress(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to allocate Elastic IP address:" <<
                  outcome.GetError().GetMessage() << std::endl;
        return false;
    }

    allocationId = outcome.GetResult().GetAllocationId();
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考AllocateAddress中的。

CLI

AWS CLI

範例 1：從 Amazon 的地址集區配置彈性 IP 地址

以下 allocate-address 範例會配置彈性 IP 地址。Amazon EC2 會從 Amazon 的地址集區中選取地址。

```
aws ec2 allocate-address
```

輸出：

```
{
    "PublicIp": "70.224.234.241",
    "AllocationId": "eipalloc-01435ba59eEXAMPLE",
    "PublicIpv4Pool": "amazon",
    "NetworkBorderGroup": "us-west-2",
    "Domain": "vpc"
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的彈性 IP 地址。

範例 2：配置彈性 IP 地址並將其與網路邊界群組建立關聯

下列 allocate-address 範例會配置彈性 IP 地址，並將其與指定的網路邊界群組建立關聯。

```
aws ec2 allocate-address \
    --network-border-group us-west-2-lax-1
```

輸出：

```
{
    "PublicIp": "70.224.234.241",
    "AllocationId": "eipalloc-e03dd489ceEXAMPLE",
    "PublicIpv4Pool": "amazon",
    "NetworkBorderGroup": "us-west-2-lax-1",
    "Domain": "vpc"
```

```
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的彈性 IP 地址。

範例 3：從您擁有的地址集區配置彈性 IP 地址

以下 `allocate-address` 範例會從您已用於 Amazon Web Services 帳戶的地址集區配置彈性 IP 地址。Amazon EC2 會從此地址集區中選取地址。

```
aws ec2 allocate-address \
    --public-ipv4-pool ipv4pool-ec2-1234567890abcdef0
```

輸出：

```
{
    "AllocationId": "eipalloc-02463d08ceEXAMPLE",
    "NetworkBorderGroup": "us-west-2",
    "CustomerOwnedIp": "18.218.95.81",
    "CustomerOwnedIpv4Pool": "ipv4pool-ec2-1234567890abcdef0",
    "Domain": "vpc"
    "NetworkBorderGroup": "us-west-2",
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的彈性 IP 地址。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AllocateAddress中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    public static String allocateAddress(Ec2Client ec2) {
        try {
            AllocateAddressRequest allocateRequest =
AllocateAddressRequest.builder()
```

```
                    .domain(DomainType.VPC)
                    .build();

            AllocateAddressResponse allocateResponse =
 ec2.allocateAddress(allocateRequest);
            return allocateResponse.allocationId();

        } catch (Ec2Exception 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AllocateAddress中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { AllocateAddressCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new AllocateAddressCommand({});

  try {
    const { AllocationId, PublicIp } = await client.send(command);
    console.log("A new IP address has been allocated to your account:");
    console.log(`ID: ${AllocationId} Public IP: ${PublicIp}`);
    console.log(
      "You can view your IP addresses in the AWS Management Console for Amazon
 EC2. Look under Network & Security > Elastic IPs",
    );
```

```
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考AllocateAddress中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun getAllocateAddress(instanceIdVal: String?): String? {
    val allocateRequest =
        AllocateAddressRequest {
            domain = DomainType.Vpc
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val allocateResponse = ec2.allocateAddress(allocateRequest)
        val allocationIdVal = allocateResponse.allocationId

        val request =
            AssociateAddressRequest {
                instanceId = instanceIdVal
                allocationId = allocationIdVal
            }

        val associateResponse = ec2.associateAddress(request)
        return associateResponse.associationId
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK AllocateAddress中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會配置彈性 IP 位址以搭配 VPC 中的執行個體使用。

```
New-EC2Address -Domain Vpc
```

輸出：

```
AllocationId         Domain      PublicIp
------------         ------      --------
eipalloc-12345678    vpc         198.51.100.2
```

範例 2：此範例會配置彈性 IP 位址，以便與 EC2-Classic 中的執行個體搭配使用。

```
New-EC2Address
```

輸出：

```
AllocationId         Domain      PublicIp
------------         ------      --------
                     standard    203.0.113.17
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AllocateAddress式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
 actions."""
```

```
    def __init__(self, ec2_resource, elastic_ip=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param elastic_ip: A Boto3 VpcAddress object. This is a high-level object
that
                           wraps Elastic IP actions.
        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def allocate(self):
        """
        Allocates an Elastic IP address that can be associated with an Amazon EC2
        instance. By using an Elastic IP address, you can keep the public IP
address
        constant even when you restart the associated instance.

        :return: The newly created Elastic IP object. By default, the address is
not
                 associated with any instance.
        """
        try:
            response =
self.ec2_resource.meta.client.allocate_address(Domain="vpc")
            self.elastic_ip =
self.ec2_resource.VpcAddress(response["AllocationId"])
        except ClientError as err:
            logger.error(
                "Couldn't allocate Elastic IP. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.elastic_ip
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件AllocateAddress中的 Python (博托 3) API 參考。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
# Creates an Elastic IP address in Amazon Virtual Private Cloud (Amazon VPC).
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @return [String] The allocation ID corresponding to the Elastic IP address.
# @example
#   puts allocate_elastic_ip_address(Aws::EC2::Client.new(region: 'us-west-2'))
def allocate_elastic_ip_address(ec2_client)
  response = ec2_client.allocate_address(domain: "vpc")
  return response.allocation_id
rescue StandardError => e
  puts "Error allocating Elastic IP address: #{e.message}"
  return "Error"
end
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考AllocateAddress中的。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        oo_result = lo_ec2->allocateaddress( iv_domain = 'vpc' ).   " oo_result
 is returned for testing purposes. "
        MESSAGE 'Allocated an Elastic IP address.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK AllocateAddress中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭`AllocateHosts`配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用`AllocateHosts`。

CLI

AWS CLI

範例 1：配置專用主機

下列`allocate-hosts`範例會在可`eu-west-1a`用區域中配置單一專用主機，您可以在其中啟動`m5.large`執行個體。依預設，專用主機僅接受目標執行個體啟動，且不支援主機復原。

```
aws ec2 allocate-hosts \
```

```
    --instance-type m5.large \
    --availability-zone eu-west-1a \
    --quantity 1
```

輸出：

```
{
    "HostIds": [
        "h-07879acf49EXAMPLE"
    ]
}
```

範例 2：配置已啟用自動放置和主機復原的專用主機

下列allocate-hosts範例會在啟用自動放置和主機復原的eu-west-1a可用區域中配置單一專用主機。

```
aws ec2 allocate-hosts \
    --instance-type m5.large \
    --availability-zone eu-west-1a \
    --auto-placement on \
    --host-recovery on \
    --quantity 1
```

輸出：

```
{
     "HostIds": [
         "h-07879acf49EXAMPLE"
     ]
}
```

範例 3：配置具有標籤的專用主機

下列allocate-hosts範例會配置單一專用主機，並套用具有名為的金鑰purpose和值的production標籤。

```
aws ec2 allocate-hosts \
    --instance-type m5.large \
    --availability-zone eu-west-1a \
```

```
    --quantity 1 \
    --tag-specifications 'ResourceType=dedicated-
host,Tags={Key=purpose,Value=production}'
```

輸出：

```
{
    "HostIds": [
        "h-07879acf49EXAMPLE"
    ]
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端使用者指南中的 Linux 執行個體配置專用主機。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AllocateHosts中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會針對指定執行個體類型和可用區域，將專用主機配置給您的帳戶

```
New-EC2Host -AutoPlacement on -AvailabilityZone eu-west-1b -InstanceType
 m4.xlarge -Quantity 1
```

輸出：

```
h-01e23f4cd567890f3
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AllocateHosts式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭AssignPrivateIpAddresses配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AssignPrivateIpAddresses。

CLI

AWS CLI

為網路介面指派特定的次要私有 IP 位址

此範例會將指定的次要私有 IP 位址指派給指定的網路介面。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 assign-private-ip-addresses --network-interface-id eni-e5aa89a3 --
private-ip-addresses 10.0.0.82
```

將 Amazon EC2 選取的次要私有 IP 地址指派給網路界面

此範例會將兩個次要私有 IP 位址指派給指定的網路介面。Amazon EC2 會從與網路界面相關聯之子網路的 CIDR 區塊範圍內的可用 IP 地址自動指派這些 IP 地址。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 assign-private-ip-addresses --network-interface-id eni-e5aa89a3 --
secondary-private-ip-address-count 2
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AssignPrivateIpAddresses中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的次要私有 IP 位址指派給指定的網路介面。

```
Register-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -PrivateIpAddress
 10.0.0.82
```

範例 2：此範例會建立兩個次要私有 IP 位址，並將它們指派給指定的網路介面。

```
Register-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -
SecondaryPrivateIpAddressCount 2
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AssignPrivateIpAddresses式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭AssociateAddress配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AssociateAddress。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Associate an Elastic IP address to an EC2 instance.
    /// </summary>
    /// <param name="allocationId">The allocation Id of an Elastic IP address.</
param>
    /// <param name="instanceId">The instance Id of the EC2 instance to
    /// associate the address with.</param>
    /// <returns>The association Id that represents
    /// the association of the Elastic IP address with an instance.</returns>
    public async Task<string> AssociateAddress(string allocationId, string
 instanceId)
    {
        var request = new AssociateAddressRequest
        {
            AllocationId = allocationId,
```

```
            InstanceId = instanceId
        };

        var response = await _amazonEC2.AssociateAddressAsync(request);
        return response.AssociationId;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考AssociateAddress中的。

Bash

AWS CLI 與 bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
###############################################################################
# function ec2_associate_address
#
# This function associates an Elastic IP address with an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to
 associate.
#       -i instance_id - The ID of the EC2 instance to associate the Elastic IP
 address with.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_associate_address() {
  local allocation_id instance_id response

  # Function to display usage information
  function usage() {
```

```
   echo "function ec2_associate_address"
   echo "Associates an Elastic IP address with an Amazon Elastic Compute Cloud
(Amazon EC2) instance."
   echo "  -a allocation_id - The allocation ID of the Elastic IP address to
associate."
   echo "  -i instance_id - The ID of the EC2 instance to associate the Elastic
IP address with."
   echo ""
 }

 # Parse the command-line arguments
 while getopts "a:i:h" option; do
   case "${option}" in
     a) allocation_id="${OPTARG}" ;;
     i) instance_id="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 # Validate the input parameters
 if [[ -z "$allocation_id" ]]; then
   errecho "ERROR: You must provide an allocation ID with the -a parameter."
   return 1
 fi

 if [[ -z "$instance_id" ]]; then
   errecho "ERROR: You must provide an instance ID with the -i parameter."
   return 1
 fi

 # Associate the Elastic IP address
 response=$(aws ec2 associate-address \
   --allocation-id "$allocation_id" \
   --instance-id "$instance_id" \
   --query "AssociationId" \
```

```
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports associate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}
```

此範例中使用的公用程式函數。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
```

```
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AssociateAddress中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::AssociateAddressRequest associate_request;
    associate_request.SetInstanceId(instanceId);
    associate_request.SetAllocationId(allocationId);

    const Aws::EC2::Model::AssociateAddressOutcome associate_outcome =
            ec2Client.AssociateAddress(associate_request);
    if (!associate_outcome.IsSuccess()) {
        std::cerr << "Failed to associate Elastic IP address " << allocationId
                  << " with instance " << instanceId << ":" <<
                  associate_outcome.GetError().GetMessage() << std::endl;
        return false;
    }
```

```
    std::cout << "Successfully associated Elastic IP address " << allocationId
              << " with instance " << instanceId << std::endl;
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考[AssociateAddress](#)中的。

CLI

AWS CLI

在 EC2-Classic 中建立彈性 IP 地址的關聯

此範例會在 EC2-Classic 中將彈性 IP 地址與執行個體建立關聯。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 associate-address --instance-id i-07ffe74c7330ebf53 --public-ip
 198.51.100.0
```

在 EC2-VPC 中建立彈性 IP 地址的關聯

此範例會在 VPC 中將彈性 IP 地址與執行個體建立關聯。

命令：

```
aws ec2 associate-address --instance-id i-0b263919b6498b123 --allocation-id
 eipalloc-64d5890a
```

輸出：

```
{
    "AssociationId": "eipassoc-2bebb745"
}
```

此範例會將彈性 IP 地址與網路介面建立關聯。

命令：

```
aws ec2 associate-address --allocation-id eipalloc-64d5890a --network-interface-
id eni-1a2b3c4d
```

此範例會將彈性 IP 地址與已和網路介面相關聯的私有 IP 地址建立關聯。

命令：

```
aws ec2 associate-address --allocation-id eipalloc-64d5890a --network-interface-
id eni-1a2b3c4d --private-ip-address 10.0.0.85
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AssociateAddress中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   public static String associateAddress(Ec2Client ec2, String instanceId,
String allocationId) {
       try {
           AssociateAddressRequest associateRequest =
AssociateAddressRequest.builder()
                   .instanceId(instanceId)
                   .allocationId(allocationId)
                   .build();

           AssociateAddressResponse associateResponse =
ec2.associateAddress(associateRequest);
           return associateResponse.associationId();

       } catch (Ec2Exception 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AssociateAddress中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { AssociateAddressCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  // You need to allocate an Elastic IP address before associating it with an
 instance.
  // You can do that with the AllocateAddressCommand.
  const allocationId = "ALLOCATION_ID";
  // You need to create an EC2 instance before an IP address can be associated
 with it.
  // You can do that with the RunInstancesCommand.
  const instanceId = "INSTANCE_ID";
  const command = new AssociateAddressCommand({
    AllocationId: allocationId,
    InstanceId: instanceId,
  });

  try {
    const { AssociationId } = await client.send(command);
    console.log(
      `Address with allocation ID ${allocationId} is now associated with instance
 ${instanceId}.`,
      `The association ID is ${AssociationId}.`,
    );
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考AssociateAddress中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun associateAddressSc(
    instanceIdVal: String?,
    allocationIdVal: String?,
): String? {
    val associateRequest =
        AssociateAddressRequest {
            instanceId = instanceIdVal
            allocationId = allocationIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val associateResponse = ec2.associateAddress(associateRequest)
        return associateResponse.associationId
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK AssociateAddress中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的彈性 IP 位址與 VPC 中的指定執行個體產生關聯。

```
C:\> Register-EC2Address -InstanceId i-12345678 -AllocationId eipalloc-12345678
```

輸出：

```
eipassoc-12345678
```

範例 2：此範例會將指定的彈性 IP 位址與 EC2-Classic 中的指定執行個體產生關聯。

```
C:\> Register-EC2Address -InstanceId i-12345678 -PublicIp 203.0.113.17
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AssociateAddress式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
 actions."""

    def __init__(self, ec2_resource, elastic_ip=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param elastic_ip: A Boto3 VpcAddress object. This is a high-level object
 that
                           wraps Elastic IP actions.
        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def associate(self, instance):
```

```
        """
        Associates an Elastic IP address with an instance. When this association
is
        created, the Elastic IP's public IP address is immediately used as the
public
        IP address of the associated instance.

        :param instance: A Boto3 Instance object. This is a high-level object
that wraps
                         Amazon EC2 instance actions.
        :return: A response that contains the ID of the association.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to associate.")
            return

        try:
            response = self.elastic_ip.associate(InstanceId=instance.id)
        except ClientError as err:
            logger.error(
                "Couldn't associate Elastic IP %s with instance %s. Here's why:
%s: %s",
                self.elastic_ip.allocation_id,
                instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        return response
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件AssociateAddress中的 Python (博托 3) API 參考。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
# Associates an Elastic IP address with an Amazon Elastic Compute Cloud
# (Amazon EC2) instance.
#
# Prerequisites:
#
# - The allocation ID corresponding to the Elastic IP address.
# - The Amazon EC2 instance.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param allocation_id [String] The ID of the allocation corresponding to
#   the Elastic IP address.
# @param instance_id [String] The ID of the instance.
# @return [String] The assocation ID corresponding to the association of the
#   Elastic IP address to the instance.
# @example
#   puts allocate_elastic_ip_address(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'eipalloc-04452e528a66279EX',
#     'i-033c48ef067af3dEX')
def associate_elastic_ip_address_with_instance(
  ec2_client,
  allocation_id,
  instance_id
)
  response = ec2_client.associate_address(
    allocation_id: allocation_id,
    instance_id: instance_id,
  )
  return response.association_id
rescue StandardError => e
  puts "Error associating Elastic IP address with instance: #{e.message}"
  return "Error"
```

```
end
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考AssociateAddress中的。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        oo_result = lo_ec2->associateaddress(                          " oo_result
 is returned for testing purposes. "
            iv_allocationid = iv_allocation_id
            iv_instanceid = iv_instance_id
        ).
        MESSAGE 'Associated an Elastic IP address with an EC2 instance.' TYPE
 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK AssociateAddress中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭`AssociateDhcpOptions`配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用`AssociateDhcpOptions`。

CLI

AWS CLI

將 DHCP 選項集與您的 VPC 產生關聯

此範例會將指定的 DHCP 選項集與指定的 VPC 產生關聯。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 associate-dhcp-options --dhcp-options-id dopt-d9070ebb --vpc-id vpc-a01106c2
```

將預設 DHCP 選項集與您的 VPC 產生關聯

此範例會將預設 DHCP 選項設定與指定的 VPC 產生關聯。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 associate-dhcp-options --dhcp-options-id default --vpc-id vpc-a01106c2
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AssociateDhcpOptions中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例將指定的 DHCP 選項集與指定的 VPC 產生關聯。

```
Register-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d -VpcId vpc-12345678
```

範例 2：此範例將預設 DHCP 選項設定與指定的 VPC 產生關聯。

```
Register-EC2DhcpOption -DhcpOptionsId default -VpcId vpc-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AssociateDhcpOptions式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭AssociateRouteTable配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AssociateRouteTable。

CLI

AWS CLI

將路由表與子網路產生關聯

此範例會將指定的路由表與指定的子網路產生關聯。

命令：

```
aws ec2 associate-route-table --route-table-id rtb-22574640 --subnet-id
 subnet-9d4a7b6c
```

輸出：

```
{
    "AssociationId": "rtbassoc-781d0d1a"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AssociateRouteTable中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的路由表與指定的子網路產生關聯。

```
Register-EC2RouteTable -RouteTableId rtb-1a2b3c4d -SubnetId subnet-1a2b3c4d
```

輸出：

```
rtbassoc-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AssociateRouteTable式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭AttachInternetGateway配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AttachInternetGateway。

CLI

AWS CLI

將網際網路閘道連接至您的 VPC

下列attach-internet-gateway範例會將指定的網際網路閘道附加至特定 VPC。

```
aws ec2 attach-internet-gateway \
    --internet-gateway-id igw-0d0fb496b3EXAMPLE \
    --vpc-id vpc-0a60eb65b4EXAMPLE
```

此命令不會產生輸出。

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的網際網路閘道。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AttachInternetGateway中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的網際網路閘道附加至指定的 VPC。

```
Add-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d -VpcId vpc-12345678
```

範例 2：此範例會建立 VPC 和網際網路閘道，然後將網際網路閘道連接至 VPC。

```
$vpc = New-EC2Vpc -CidrBlock 10.0.0.0/16
New-EC2InternetGateway | Add-EC2InternetGateway -VpcId $vpc.VpcId
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AttachInternetGateway式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭AttachNetworkInterface配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AttachNetworkInterface。

CLI

AWS CLI

範例 1：將網路介面連接至執行個體

下列attach-network-interface範例會將指定的網路介面附加至指定的執行個體。

```
aws ec2 attach-network-interface \
    --network-interface-id eni-0dc56a8d4640ad10a \
    --instance-id i-1234567890abcdef0 \
    --device-index 1
```

輸出：

```
{
    "AttachmentId": "eni-attach-01a8fc87363f07cf9"
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的彈性網路界面。

範例 2：將網路介面連接至具有多張網路卡的執行個體

下列attach-network-interface範例會將指定的網路介面附加至指定的執行個體和網路卡。

```
aws ec2 attach-network-interface \
    --network-interface-id eni-07483b1897541ad83 \
    --instance-id i-01234567890abcdef \
    --network-card-index 1 \
    --device-index 1
```

輸出：

```
{
    "AttachmentId": "eni-attach-0fbd7ee87a88cd06c"
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的彈性網路界面。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AttachNetworkInterface中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的網路介面附加至指定的執行個體。

```
Add-EC2NetworkInterface -NetworkInterfaceId eni-12345678 -InstanceId i-1a2b3c4d -
DeviceIndex 1
```

輸出：

```
eni-attach-1a2b3c4d
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AttachNetworkInterface式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**AttachVolume**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AttachVolume。

CLI

AWS CLI

將磁碟區附加至執行個體

此範例指令將體積 (vol-1234567890abcdef0) 貼附至例證 (i-01474ef662b89480) 做為/dev/sdf。

命令：

```
aws ec2 attach-volume --volume-id vol-1234567890abcdef0 --instance-id
 i-01474ef662b89480 --device /dev/sdf
```

輸出：

```
{
    "AttachTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "InstanceId": "i-01474ef662b89480",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "attaching",
    "Device": "/dev/sdf"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AttachVolume中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的磁碟區附加至指定的執行個體，並以指定的裝置名稱公開該磁碟區。

```
Add-EC2Volume -VolumeId vol-12345678 -InstanceId i-1a2b3c4d -Device /dev/sdh
```

輸出：

```
AttachTime          : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device              : /dev/sdh
InstanceId          : i-1a2b3c4d
State               : attaching
VolumeId            : vol-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AttachVolume式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭AttachVpnGateway配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AttachVpnGateway。

CLI

AWS CLI

將虛擬私有閘道連接至您的 VPC

下列attach-vpn-gateway範例會將指定的虛擬私有閘道附加至指定的 VPC。

```
aws ec2 attach-vpn-gateway \
    --vpn-gateway-id vgw-9a4cacf3 \
    --vpc-id vpc-a01106c2
```

輸出：

```
{
    "VpcAttachment": {
        "State": "attaching",
        "VpcId": "vpc-a01106c2"
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AttachVpnGateway中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的虛擬私有閘道附加至指定的 VPC。

```
Add-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d -VpcId vpc-12345678
```

輸出：

```
State        VpcId
-----        -----
```

```
attaching    vpc-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AttachVpnGateway式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭AuthorizeSecurityGroupEgress配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AuthorizeSecurityGroupEgress。

CLI

AWS CLI

新增允許輸出流量至特定位址範圍的規則

此範例命令會新增規則，以授與 TCP 連接埠 80 上指定位址範圍的存取權。

命令 (Linux):

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
 IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges='[{CidrIp=10.0.0.0/16}]'
```

指令 (視窗) :

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
 IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges=[{CidrIp=10.0.0.0/16}]
```

新增允許輸出流量至特定安全性群組的規則

此範例命令會新增一個規則，以授與 TCP 連接埠 80 上指定安全性群組的存取權。

命令：

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
 IpProtocol=tcp,FromPort=80,ToPort=80,UserIdGroupPairs='[{GroupId=sg-4b51a32f}]'
```

指令 (視窗) :

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
 IpProtocol=tcp,FromPort=80,ToPort=80,UserIdGroupPairs=[{GroupId=sg-4b51a32f}]
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AuthorizeSecurityGroupEgress中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會針對 EC2-VPC 的指定安全性群組定義輸出規則。此規則會授與 TCP 連接埠 80 上指定 IP 位址範圍的存取權。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
$ip = @{ IpProtocol="tcp"; FromPort="80"; ToPort="80";
 IpRanges="203.0.113.0/24" }
Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

範例 2：在 PowerShell 版本 2 中，您必須使用新物件來建立 IpPermission 物件。

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 80
$ip.ToPort = 80
$ip.IpRanges.Add("203.0.113.0/24")

Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

範例 3：此範例會授與 TCP 連接埠 80 上指定來源安全性群組的存取權。

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"

Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission
 @( @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; UserIdGroupPairs=$ug } )
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AuthorizeSecurityGroupEgress式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭AuthorizeSecurityGroupIngress配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AuthorizeSecurityGroupIngress。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Authorize the local computer ingress to EC2 instances associated
    /// with the virtual private cloud (VPC) security group.
    /// </summary>
    /// <param name="groupName">The name of the security group.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AuthorizeSecurityGroupIngress(string groupName)
    {
        // Get the IP address for the local computer.
        var ipAddress = await GetIpAddress();
        Console.WriteLine($"Your IP address is: {ipAddress}");
        var ipRanges = new List<IpRange> { new IpRange { CidrIp =
$"{ipAddress}/32" } };
        var permission = new IpPermission
        {
            Ipv4Ranges = ipRanges,
            IpProtocol = "tcp",
            FromPort = 22,
            ToPort = 22
        };
        var permissions = new List<IpPermission> { permission };
        var response = await _amazonEC2.AuthorizeSecurityGroupIngressAsync(
```

```
            new AuthorizeSecurityGroupIngressRequest(groupName, permissions));
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Authorize the local computer for ingress to
    /// the Amazon EC2 SecurityGroup.
    /// </summary>
    /// <returns>The IPv4 address of the computer running the scenario.</returns>
    private static async Task<string> GetIpAddress()
    {
        var httpClient = new HttpClient();
        var ipString = await httpClient.GetStringAsync("https://
checkip.amazonaws.com");

        // The IP address is returned with a new line
        // character on the end. Trim off the whitespace and
        // return the value to the caller.
        return ipString.Trim();
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考AuthorizeSecurityGroupIngress中的。

Bash

AWS CLI 與 bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
###############################################################################
# function ec2_authorize_security_group_ingress
#
# This function authorizes an ingress rule for an Amazon Elastic Compute Cloud
 (Amazon EC2) security group.
#
```

```
# Parameters:
#       -g security_group_id - The ID of the security group.
#       -i ip_address - The IP address or CIDR block to authorize.
#       -p protocol - The protocol to authorize (e.g., tcp, udp, icmp).
#       -f from_port - The start of the port range to authorize.
#       -t to_port - The end of the port range to authorize.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_authorize_security_group_ingress() {
  local security_group_id ip_address protocol from_port to_port response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_authorize_security_group_ingress"
    echo "Authorizes an ingress rule for an Amazon Elastic Compute Cloud (Amazon
 EC2) security group."
    echo "  -g security_group_id - The ID of the security group."
    echo "  -i ip_address - The IP address or CIDR block to authorize."
    echo "  -p protocol - The protocol to authorize (e.g., tcp, udp, icmp)."
    echo "  -f from_port - The start of the port range to authorize."
    echo "  -t to_port - The end of the port range to authorize."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "g:i:p:f:t:h" option; do
    case "${option}" in
      g) security_group_id="${OPTARG}" ;;
      i) ip_address="${OPTARG}" ;;
      p) protocol="${OPTARG}" ;;
      f) from_port="${OPTARG}" ;;
      t) to_port="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
```

```
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -g parameter."
    usage
    return 1
  fi

  if [[ -z "$ip_address" ]]; then
    errecho "ERROR: You must provide an IP address or CIDR block with the -i
parameter."
    usage
    return 1
  fi

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

  if [[ -z "$from_port" ]]; then
    errecho "ERROR: You must provide a start port with the -f parameter."
    usage
    return 1
  fi

  if [[ -z "$to_port" ]]; then
    errecho "ERROR: You must provide an end port with the -t parameter."
    usage
    return 1
  fi

  response=$(aws ec2 authorize-security-group-ingress \
    --group-id "$security_group_id" \
    --cidr "${ip_address}/32" \
    --protocol "$protocol" \
    --port "$from_port-$to_port" \
    --output text) || {
    aws_cli_error_log ${?}
```

```
    errecho "ERROR: AWS reports authorize-security-group-ingress operation
 failed.$response"
    return 1
  }

  return 0
}
```

此範例中使用的公用程式函數。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
```

```
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[AuthorizeSecurityGroupIngress](#)中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在[AWS 設定和執行程式碼範例儲存庫](#)。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::IpRange ip_range;
    ip_range.SetCidrIp("0.0.0.0/0");

    Aws::EC2::Model::IpPermission permission1;
    permission1.SetIpProtocol("tcp");
    permission1.SetToPort(80);
    permission1.SetFromPort(80);
    permission1.AddIpRanges(ip_range);

    authorize_request.AddIpPermissions(permission1);

    Aws::EC2::Model::IpPermission permission2;
    permission2.SetIpProtocol("tcp");
    permission2.SetToPort(22);
    permission2.SetFromPort(22);
    permission2.AddIpRanges(ip_range);
```

```
    authorize_request.AddIpPermissions(permission2);

    const Aws::EC2::Model::AuthorizeSecurityGroupIngressOutcome authorizeOutcome
 =
            ec2Client.AuthorizeSecurityGroupIngress(authorizeRequest);

    if (!authorizeOutcome.IsSuccess()) {
        std::cerr << "Failed to set ingress policy for security group " <<
                  groupName << ":" << authorizeOutcome.GetError().GetMessage() <<
                  std::endl;
        return false;
    }

    std::cout << "Successfully added ingress policy to security group " <<
              groupName << std::endl;
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考AuthorizeSecurityGroupIngress中的。

CLI

AWS CLI

範例 1：新增規則，以允許傳入 SSH 流量

以下 authorize-security-group-ingress 範例會新增規則，以允許 TCP 連接埠 22 (SSH) 上的傳入流量。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --protocol tcp \
    --port 22 \
    --cidr 203.0.113.0/24
```

輸出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
```

```
            "SecurityGroupRuleId": "sgr-01afa97ef3e1bedfc",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "203.0.113.0/24"
        }
    ]
}
```

範例 2：新增規則，以允許來自其他安全群組的傳入 HTTP 流量

下列 authorize-security-group-ingress 範例會新增規則，以允許 TCP 連接埠 80 上來自來源安全群組 sg-1a2b3c4d 的傳入存取。來源群組必須在相同 VPC 或對等 VPC 中 (需要 VPC 對等互連)。傳入流量會根據與來源安全群組相關聯之執行個體的私有 IP 地址允許 (而非公有 IP 地址或彈性 IP 地址)。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --protocol tcp \
    --port 80 \
    --source-group sg-1a2b3c4d
```

輸出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-01f4be99110f638a7",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "ReferencedGroupInfo": {
                "GroupId": "sg-1a2b3c4d",
                "UserId": "123456789012"
            }
```

```
        }
    ]
}
```

範例 3：在相同的呼叫中新增多個規則

下列 `authorize-security-group-ingress` 範例會使用 `ip-permissions` 參數來新增兩個傳入規則；一個規則可在 TCP 連接埠 3389 (RDP) 上啟用傳入存取，另一個規則可啟用 Ping/ICMP。

AWS EC2- authorize-security-group-ingress 組識別碼 SG-1234567890-IP 權限 IpProtocol = TCP，= 3 FromPort 389，=「[{= 172.31.0.0/16}]」= ICMP，= -1，=「[{= 172.31.0.0/16}]」= ICMP，= -1，=「[{ToPort= 172.31.0.0/16}]」IpRanges CidrIp IpProtocol FromPort ToPort IpRanges CidrIp

輸出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-00e06e5d3690f29f3",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 3389,
            "ToPort": 3389,
            "CidrIpv4": "172.31.0.0/16"
        },
        {
            "SecurityGroupRuleId": "sgr-0a133dd4493944b87",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": -1,
            "ToPort": -1,
            "CidrIpv4": "172.31.0.0/16"
        }
    ]
}
```

範例 4：為 ICMP 流量新增規則

下列 `authorize-security-group-ingress` 範例會使用 `ip-permissions` 參數來新增傳入規則，以允許來自任何地方的 ICMP 訊息 `Destination Unreachable: Fragmentation Needed and Don't Fragment was Set` (類型 3，代碼 4)。

AWS EC2- authorize-security-group-ingress 組標識符號 SG-1234567890-IP-IP 權限 = ICMP，= 3，= 4，=「[{= 0.0.0/0}]」 IpProtocol FromPort ToPort IpRanges CidrIp

輸出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0de3811019069b787",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "icmp",
            "FromPort": 3,
            "ToPort": 4,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

範例 5：為 IPv6 流量新增規則

下列 `authorize-security-group-ingress` 範例會使用 `ip-permissions` 參數新增傳入規則，以允許來自 IPv6 範圍 `2001:db8:1234:1a00::/64` 的 SSH 存取 (連接埠 22)。

AWS EC2- authorize-security-group-ingress 組識別碼 SG-1234567890 固定-IP 權限 = TCP, = 22, = 22, 知識產品 6 範圍 = "[{6 = 2001: 分貝 8:1234:1 A00::/ IpProtocol64}]」FromPort ToPort CidrIpv

輸出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
```

```
            "SecurityGroupRuleId": "sgr-0455bc68b60805563",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv6": "2001:db8:1234:1a00::/64"
        }
    ]
}
```

範例 6：為 ICMPv6 流量新增規則

下列 `authorize-security-group-ingress` 範例會使用 `ip-permissions` 參數來新增傳入規則，以允許來自任何地方的 ICMPv6 流量。

AWS EC2- authorize-security-group-ingress 組識別碼 SG-1234567890ABCDEF-IP 權限 = ICMP6 的範圍 = "[{6 =:: /0}]」 IpProtocol CidrIpv

輸出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-04b612d9363ab6327",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "icmpv6",
            "FromPort": -1,
            "ToPort": -1,
            "CidrIpv6": "::/0"
        }
    ]
}
```

範例 7：新增具有描述的規則

下列 `authorize-security-group-ingress` 範例會使用 `ip-permissions` 參數來新增傳入規則，以允許來自指定 IPv4 地址範圍的 RDP 流量。此規則提供描述，可於稍後協助識別。

AWS EC2- authorize-security-group-ingress 組識別碼 SG-1234567890 安裝-IP 權限 IpProtocol = TCP，= 3389，= 3389，= "[{= 203.0.113.0/24，描述 = 'RDP 訪問來自紐約辦公室'}]」FromPort ToPort IpRanges CidrIp

輸出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0397bbcc01e974db3",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 3389,
            "ToPort": 3389,
            "CidrIpv4": "203.0.113.0/24",
            "Description": "RDP access from NY office"
        }
    ]
}
```

範例 8：新增使用字首清單的傳入規則

下列 `authorize-security-group-ingress` 範例會使用 `ip-permissions` 參數來新增傳入規則，以允許指定字首清單中 CIDR 範圍適用的所有流量。

AWS EC2 authorize-security-group-ingress -組識別碼標識符號 IpProtocol PrefixListIds PrefixListId

輸出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-09c74b32f677c6c7c",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "-1",
            "FromPort": -1,
```

```
            "ToPort": -1,
            "PrefixListId": "pl-0721453c7ac4ec009"
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的安全群組。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AuthorizeSecurityGroupIngress中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   public static String createSecurityGroup(Ec2Client ec2, String groupName,
String groupDesc, String vpcId,
           String myIpAddress) {
       try {
           CreateSecurityGroupRequest createRequest =
CreateSecurityGroupRequest.builder()
                   .groupName(groupName)
                   .description(groupDesc)
                   .vpcId(vpcId)
                   .build();

           CreateSecurityGroupResponse resp =
ec2.createSecurityGroup(createRequest);
           IpRange ipRange = IpRange.builder()
                   .cidrIp(myIpAddress + "/0")
                   .build();

           IpPermission ipPerm = IpPermission.builder()
                   .ipProtocol("tcp")
                   .toPort(80)
                   .fromPort(80)
                   .ipRanges(ipRange)
```

```
                    .build();

            IpPermission ipPerm2 = IpPermission.builder()
                    .ipProtocol("tcp")
                    .toPort(22)
                    .fromPort(22)
                    .ipRanges(ipRange)
                    .build();

            AuthorizeSecurityGroupIngressRequest authRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                    .groupName(groupName)
                    .ipPermissions(ipPerm, ipPerm2)
                    .build();

            ec2.authorizeSecurityGroupIngress(authRequest);
            System.out.println("Successfully added ingress policy to security
group " + groupName);
            return resp.groupId();

        } catch (Ec2Exception 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AuthorizeSecurityGroupIngress中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { AuthorizeSecurityGroupIngressCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// Grant permissions for a single IP address to ssh into instances
// within the provided security group.
export const main = async () => {
  const command = new AuthorizeSecurityGroupIngressCommand({
    // Replace with a security group ID from the AWS console or
    // the DescribeSecurityGroupsCommand.
    GroupId: "SECURITY_GROUP_ID",
    IpPermissions: [
      {
        IpProtocol: "tcp",
        FromPort: 22,
        ToPort: 22,
        // Replace 0.0.0.0 with the IP address to authorize.
        // For more information on this notation, see
        // https://en.wikipedia.org/wiki/Classless_Inter-
Domain_Routing#CIDR_notation
        IpRanges: [{ CidrIp: "0.0.0.0/32" }],
      },
    ],
  });

  try {
    const { SecurityGroupRules } = await client.send(command);
    console.log(JSON.stringify(SecurityGroupRules, null, 2));
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考AuthorizeSecurityGroupIngress中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun createEC2SecurityGroupSc(
    groupNameVal: String?,
    groupDescVal: String?,
    vpcIdVal: String?,
    myIpAddress: String?,
): String? {
    val request =
        CreateSecurityGroupRequest {
            groupName = groupNameVal
            description = groupDescVal
            vpcId = vpcIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val resp = ec2.createSecurityGroup(request)
        val ipRange =
            IpRange {
                cidrIp = "$myIpAddress/0"
            }

        val ipPerm =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 80
                fromPort = 80
                ipRanges = listOf(ipRange)
            }

        val ipPerm2 =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 22
```

```
                fromPort = 22
                ipRanges = listOf(ipRange)
            }

        val authRequest =
            AuthorizeSecurityGroupIngressRequest {
                groupName = groupNameVal
                ipPermissions = listOf(ipPerm, ipPerm2)
            }
        ec2.authorizeSecurityGroupIngress(authRequest)
        println("Successfully added ingress policy to Security Group
 $groupNameVal")
        return resp.groupId
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK AuthorizeSecurityGroupIngress中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例定義 EC2-VPC 安全性群組的輸入規則。這些規則會授與 SSH (連接埠 22) 和 RDC (連接埠 3389) 的特定 IP 位址的存取權。請注意，您必須使用安全性群組識別碼而非安全群組名稱來識別 EC2-VPC 的安全性群組。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
$ip1 = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";
 IpRanges="203.0.113.25/32" }
$ip2 = @{ IpProtocol="tcp"; FromPort="3389"; ToPort="3389";
 IpRanges="203.0.113.25/32" }

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission @( $ip1, $ip2 )
```

例 2：對於 PowerShell 版本 2，您必須使用新對象創建對 IpPermission 象。

```
$ip1 = New-Object Amazon.EC2.Model.IpPermission
$ip1.IpProtocol = "tcp"
$ip1.FromPort = 22
```

```
$ip1.ToPort = 22
$ip1.IpRanges.Add("203.0.113.25/32")

$ip2 = new-object Amazon.EC2.Model.IpPermission
$ip2.IpProtocol = "tcp"
$ip2.FromPort = 3389
$ip2.ToPort = 3389
$ip2.IpRanges.Add("203.0.113.25/32")

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission @( $ip1, $ip2 )
```

範例 3：此範例定義 EC2-Classic 之安全性群組的輸入規則。這些規則會授與 SSH (連接埠 22) 和 RDC (連接埠 3389) 的特定 IP 位址的存取權。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
$ip1 = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";
 IpRanges="203.0.113.25/32" }
$ip2 = @{ IpProtocol="tcp"; FromPort="3389"; ToPort="3389";
 IpRanges="203.0.113.25/32" }

Grant-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission
 @( $ip1, $ip2 )
```

實施例 4：在 PowerShell 版本 2 中，您必須使用新對象來創建對 IpPermission 象。

```
$ip1 = New-Object Amazon.EC2.Model.IpPermission
$ip1.IpProtocol = "tcp"
$ip1.FromPort = 22
$ip1.ToPort = 22
$ip1.IpRanges.Add("203.0.113.25/32")

$ip2 = new-object Amazon.EC2.Model.IpPermission
$ip2.IpProtocol = "tcp"
$ip2.FromPort = 3389
$ip2.ToPort = 3389
$ip2.IpRanges.Add("203.0.113.25/32")

Grant-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission
 @( $ip1, $ip2 )
```

範例 5：此範例會將指定的來源安全性群組 (sg-1a2b3c4d) 存取權授與指定的安全性群組 (sg-12345678)。

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission
 @( @{ IpProtocol="tcp"; FromPort="8081"; ToPort="8081"; UserIdGroupPairs=$ug } )
```

範例 6：此範例會將 CIDR 5.5.5/32 新增至安全性群組 sg-1234abcd 的輸入規則 (適用於 TCP 連接埠 22 流量)，並附上說明。

```
$IpRange = New-Object -TypeName Amazon.EC2.Model.IpRange
$IpRange.CidrIp = "5.5.5.5/32"
$IpRange.Description = "SSH from Office"
$IpPermission = New-Object Amazon.EC2.Model.IpPermission
$IpPermission.IpProtocol = "tcp"
$IpPermission.ToPort = 22
$IpPermission.FromPort = 22
$IpPermission.Ipv4Ranges = $IpRange
Grant-EC2SecurityGroupIngress -GroupId sg-1234abcd -IpPermission $IpPermission
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程AuthorizeSecurityGroupIngress式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class SecurityGroupWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) security group
 actions."""

    def __init__(self, ec2_resource, security_group=None):
        """
```

```
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param security_group: A Boto3 SecurityGroup object. This is a high-level
object
                               that wraps security group actions.
        """
        self.ec2_resource = ec2_resource
        self.security_group = security_group

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def authorize_ingress(self, ssh_ingress_ip):
        """
        Adds a rule to the security group to allow access to SSH.

        :param ssh_ingress_ip: The IP address that is granted inbound access to
connect
                               to port 22 over TCP, used for SSH.
        :return: The response to the authorization request. The 'Return' field of
the
                 response indicates whether the request succeeded or failed.
        """
        if self.security_group is None:
            logger.info("No security group to update.")
            return

        try:
            ip_permissions = [
                {
                    # SSH ingress open to only the specified IP address.
                    "IpProtocol": "tcp",
                    "FromPort": 22,
                    "ToPort": 22,
                    "IpRanges": [{"CidrIp": f"{ssh_ingress_ip}/32"}],
                }
            ]
            response = self.security_group.authorize_ingress(
                IpPermissions=ip_permissions
```

```
            )
        except ClientError as err:
            logger.error(
                "Couldn't authorize inbound rules for %s. Here's why: %s: %s",
                self.security_group.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件AuthorizeSecurityGroupIngress中的 Python (博托 3) API 參考。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CancelCapacityReservation配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CancelCapacityReservation。

CLI

AWS CLI

若要取消容量保留

下列cancel-capacity-reservation範例會取消指定的容量保留。

```
aws ec2 cancel-capacity-reservation \
    --capacity-reservation-id cr-1234abcd56EXAMPLE
```

輸出：

```
{
    "Return": true
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的取消容量保留。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CancelCapacityReservation中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例取消容量保留

```
Remove-EC2CapacityReservation -CapacityReservationId cr-0c1f2345db6f7cdba
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2CapacityReservation
 (CancelCapacityReservation)" on target "cr-0c1f2345db6f7cdba".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): y
True
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CancelCapacityReservation式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CancelImportTask配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CancelImportTask。

CLI

AWS CLI

若要取消匯入工作

下列cancel-import-task範例會取消指定的匯入影像工作。

```
aws ec2 cancel-import-task \
    --import-task-id import-ami-1234567890abcdef0
```

輸出：

```
{
    "ImportTaskId": "import-ami-1234567890abcdef0",
    "PreviousState": "active",
    "State": "deleting"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CancelImportTask中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會取消指定的匯入工作 (快照或影像匯入)。如果需要，可以使用**-CancelReason**參數提供原因。

```
Stop-EC2ImportTask -ImportTaskId import-ami-abcdefgh
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CancelImportTask式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**CancelSpotFleetRequests**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CancelSpotFleetRequests。

CLI

AWS CLI

範例 1：取消 Spot 叢集請求並終止關聯的執行個體

下列cancel-spot-fleet-requests範例會取消 Spot 叢集請求，並終止關聯的隨需執行個體和 Spot 執行個體。

```
aws ec2 cancel-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
```

```
    --terminate-instances
```

輸出：

```
{
    "SuccessfulFleetRequests": [
        {
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "CurrentSpotFleetRequestState": "cancelled_terminating",
            "PreviousSpotFleetRequestState": "active"
        }
    ],
    "UnsuccessfulFleetRequests": []
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的取消 Spot 叢集請求。

範例 2：在不終止關聯執行個體的情況下取消 Spot 叢集請求

下列cancel-spot-fleet-requests範例會取消 Spot 叢集請求，而不終止關聯的隨需執行個體和 Spot 執行個體。

```
aws ec2 cancel-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --no-terminate-instances
```

輸出：

```
{
    "SuccessfulFleetRequests": [
        {
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "CurrentSpotFleetRequestState": "cancelled_running",
            "PreviousSpotFleetRequestState": "active"
        }
    ],
    "UnsuccessfulFleetRequests": []
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的取消 Spot 叢集請求。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CancelSpotFleetRequests中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會取消指定的 Spot 叢集請求並終止相關聯的 Spot 執行個體。

```
Stop-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -TerminateInstance $true
```

範例 2：此範例會取消指定的 Spot 叢集請求，而不終止關聯的 Spot 執行個體。

```
Stop-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -TerminateInstance $false
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CancelSpotFleetRequests式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CancelSpotInstanceRequests配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CancelSpotInstanceRequests。

CLI

AWS CLI

取消競價型執行個體請求

此範例指令會取消 Spot 執行個體請求。

命令：

```
aws ec2 cancel-spot-instance-requests --spot-instance-request-ids sir-08b93456
```

輸出：

```
{
    "CancelledSpotInstanceRequests": [
        {
            "State": "cancelled",
            "SpotInstanceRequestId": "sir-08b93456"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CancelSpotInstanceRequests中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會取消指定的 Spot 執行個體請求。

```
Stop-EC2SpotInstanceRequest -SpotInstanceRequestId sir-12345678
```

輸出：

```
SpotInstanceRequestId    State
---------------------    -----
sir-12345678             cancelled
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CancelSpotInstanceRequests式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ConfirmProductInstance配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ConfirmProductInstance。

CLI

AWS CLI

若要確認產品實例

此範例會判斷指定的產品代碼是否與指定的執行個體相關聯。

命令：

```
aws ec2 confirm-product-instance --product-code 774F4FF8 --instance-id
 i-1234567890abcdef0
```

輸出：

```
{
  "OwnerId": "123456789012"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ConfirmProductInstance中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會判斷指定的產品代碼是否與指定的執行個體相關聯。

```
Confirm-EC2ProductInstance -ProductCode 774F4FF8 -InstanceId i-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ConfirmProductInstance式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CopyImage配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CopyImage。

CLI

AWS CLI

範例 1：將 AMI 複製到另一個區域

下列copy-image範例指令會將指定的 AMI 從「us-west-2區域」複製到「us-east-1區域」，並新增簡短描述。

```
aws ec2 copy-image \
    --region us-east-1 \
    --name ami-name \
    --source-region us-west-2 \
    --source-image-id ami-066877671789bd71b \
    --description "This is my copied image."
```

輸出：

```
{
    "ImageId": "ami-0123456789abcdefg"
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的複製 AMI。

範例 2：將 AMI 複製到另一個區域並加密備份快照

下列copy-image命令會將指定的 AMI 從us-west-2區域複製到目前的區域，並使用指定的 KMS 金鑰加密備份快照。

```
aws ec2 copy-image \
    --source-region us-west-2 \
    --name ami-name \
    --source-image-id ami-066877671789bd71b \
    --encrypted \
    --kms-key-id alias/my-kms-key
```

輸出：

```
{
    "ImageId": "ami-0123456789abcdefg"
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的複製 AMI。

範例 3：複製 AMI 時包含使用者定義的 AMI 標籤

複製 AMI 時，下列copy-image指令會使用--copy-image-tags參數來複製使用者定義的 AMI 標籤。

```
aws ec2 copy-image \
```

```
    --region us-east-1 \
    --name ami-name \
    --source-region us-west-2 \
    --source-image-id ami-066877671789bd71b \
    --description "This is my copied image."
    --copy-image-tags
```

輸出：

```
{
    "ImageId": "ami-0123456789abcdefg"
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的複製 AMI。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CopyImage中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將「歐洲 (愛爾蘭)」區域中指定的 AMI 複製到「美國西部 (奧勒岡)」區域。如果未指定-Region，則會使用目前的預設區域作為目的地區域。

```
Copy-EC2Image -SourceRegion eu-west-1 -SourceImageId ami-12345678 -Region us-
west-2 -Name "Copy of ami-12345678"
```

輸出：

```
ami-87654321
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CopyImage式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CopySnapshot配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CopySnapshot。

CLI

AWS CLI

範例 1：將快照複製到另一個區域

下列copy-snapshot範例指令會將指定的快照從「us-west-2區域」複製到「us-east-1區域」，並新增簡短描述。

```
aws ec2 copy-snapshot \
    --region us-east-1 \
    --source-region us-west-2 \
    --source-snapshot-id snap-066877671789bd71b \
    --description "This is my copied snapshot."
```

輸出：

```
{
    "SnapshotId": "snap-066877671789bd71b"
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的複製 Amazon EBS 快照。

範例 2：複製未加密的快照並加密新快照

下列copy-snapshot命令會將指定的未加密快照從us-west-2區域複製到目前區域，並使用指定的 KMS 金鑰加密新快照。

```
aws ec2 copy-snapshot \
    --source-region us-west-2 \
    --source-snapshot-id snap-066877671789bd71b \
    --encrypted \
    --kms-key-id alias/my-kms-key
```

輸出：

```
{
    "SnapshotId": "snap-066877671789bd71b"
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的複製 Amazon EBS 快照。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CopySnapshot中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的快照從歐洲 (愛爾蘭) 區域複製到美國西部 (奧勒岡) 區域。

```
Copy-EC2Snapshot -SourceRegion eu-west-1 -SourceSnapshotId snap-12345678 -Region
 us-west-2
```

範例 2：如果您設定了預設區域並省略「區域」參數，則預設目的地區域為預設區域。

```
Set-DefaultAWSRegion us-west-2
Copy-EC2Snapshot -SourceRegion eu-west-1 -SourceSnapshotId snap-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CopySnapshot式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateCapacityReservation配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateCapacityReservation。

CLI

AWS CLI

範例 1：建立容量保留

下列create-capacity-reservation範例會在可eu-west-1a用區域中建立容量保留，您可以在其中啟動三個t2.medium執行 Linux/Unix 作業系統的執行個體。根據預設，容量保留是以開放式執行個體比對準則建立，且不支援暫時儲存，且在您手動取消之前，它會保持作用中狀態。

```
aws ec2 create-capacity-reservation \
    --availability-zone eu-west-1a \
    --instance-type t2.medium \
    --instance-platform Linux/UNIX \
    --instance-count 3
```

輸出：

```
{
    "CapacityReservation": {
        "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
        "EndDateType": "unlimited",
        "AvailabilityZone": "eu-west-1a",
        "InstanceMatchCriteria": "open",
        "EphemeralStorage": false,
        "CreateDate": "2019-08-16T09:27:35.000Z",
        "AvailableInstanceCount": 3,
        "InstancePlatform": "Linux/UNIX",
        "TotalInstanceCount": 3,
        "State": "active",
        "Tenancy": "default",
        "EbsOptimized": false,
        "InstanceType": "t2.medium"
    }
}
```

範例 2：建立在指定日期/時間自動結束的容量保留

下列create-capacity-reservation範例會在可eu-west-1a用區域中建立容量保留，您可以在其中啟動三個m5.large執行 Linux/Unix 作業系統的執行個體。此容量保留將於 2019 年 8 月 31 日晚上 9 時 59 分自動結束。

```
aws ec2 create-capacity-reservation \
    --availability-zone eu-west-1a \
    --instance-type m5.large \
    --instance-platform Linux/UNIX \
    --instance-count 3 \
    --end-date-type limited \
    --end-date 2019-08-31T23:59:59Z
```

輸出：

```
{
    "CapacityReservation": {
        "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
        "EndDateType": "limited",
        "AvailabilityZone": "eu-west-1a",
        "EndDate": "2019-08-31T23:59:59.000Z",
```

```
        "InstanceMatchCriteria": "open",
        "EphemeralStorage": false,
        "CreateDate": "2019-08-16T10:15:53.000Z",
        "AvailableInstanceCount": 3,
        "InstancePlatform": "Linux/UNIX",
        "TotalInstanceCount": 3,
        "State": "active",
        "Tenancy": "default",
        "EbsOptimized": false,
        "InstanceType": "m5.large"
    }
}
```

範例 3：建立僅接受目標執行個體啟動的容量保留

下列create-capacity-reservation範例會建立只接受目標執行個體啟動的容量保留。

```
aws ec2 create-capacity-reservation \
    --availability-zone eu-west-1a \
    --instance-type m5.large \
    --instance-platform Linux/UNIX \
    --instance-count 3 \
    --instance-match-criteria targeted
```

輸出：

```
{
    "CapacityReservation": {
        "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
        "EndDateType": "unlimited",
        "AvailabilityZone": "eu-west-1a",
        "InstanceMatchCriteria": "targeted",
        "EphemeralStorage": false,
        "CreateDate": "2019-08-16T10:21:57.000Z",
        "AvailableInstanceCount": 3,
        "InstancePlatform": "Linux/UNIX",
        "TotalInstanceCount": 3,
        "State": "active",
        "Tenancy": "default",
        "EbsOptimized": false,
        "InstanceType": "m5.large"
    }
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的建立容量保留。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateCapacityReservation中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例建立具有指定屬性的新容量保留

```
Add-EC2CapacityReservation -InstanceType m4.xlarge -InstanceCount 2 -
AvailabilityZone eu-west-1b -EbsOptimized True -InstancePlatform Windows
```

輸出：

```
AvailabilityZone       : eu-west-1b
AvailableInstanceCount : 2
CapacityReservationId  : cr-0c1f2345db6f7cdba
CreateDate             : 3/28/2019 9:29:41 AM
EbsOptimized           : True
EndDate                : 1/1/0001 12:00:00 AM
EndDateType            : unlimited
EphemeralStorage       : False
InstanceMatchCriteria  : open
InstancePlatform       : Windows
InstanceType           : m4.xlarge
State                  : active
Tags                   : {}
Tenancy                : default
TotalInstanceCount     : 2
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateCapacityReservation式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateCustomerGateway配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateCustomerGateway。

CLI

AWS CLI

若要建立客戶閘道

此範例會使用其外部介面的指定 IP 位址建立客戶閘道。

命令：

```
aws ec2 create-customer-gateway --type ipsec.1 --public-ip 12.1.2.3 --bgp-asn
 65534
```

輸出：

```
{
    "CustomerGateway": {
        "CustomerGatewayId": "cgw-0e11f167",
        "IpAddress": "12.1.2.3",
        "State": "available",
        "Type": "ipsec.1",
        "BgpAsn": "65534"
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[CreateCustomerGateway](#)中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會建立指定的客戶閘道。

```
New-EC2CustomerGateway -Type ipsec.1 -PublicIp 203.0.113.12 -BgpAsn 65534
```

輸出：

```
BgpAsn           : 65534
CustomerGatewayId : cgw-1a2b3c4d
IpAddress        : 203.0.113.12
State            : available
Tags             : {}
```

```
Type                : ipsec.1
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateCustomerGateway式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateDhcpOptions配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateDhcpOptions。

CLI

AWS CLI

建立一組 DHCP 選項的步驟

下列create-dhcp-options範例會建立一組指定網域名稱、網域名稱伺服器和 NetBIOS 節點類型的 DHCP 選項。

```
aws ec2 create-dhcp-options \
    --dhcp-configuration \
        "Key=domain-name-servers,Values=10.2.5.1,10.2.5.2" \
        "Key=domain-name,Values=example.com" \
        "Key=netbios-node-type,Values=2"
```

輸出：

```
{
    "DhcpOptions": {
        "DhcpConfigurations": [
            {
                "Key": "domain-name",
                "Values": [
                    {
                        "Value": "example.com"
                    }
                ]
            },
            {
```

```
                "Key": "domain-name-servers",
                "Values": [
                    {
                        "Value": "10.2.5.1"
                    },
                    {
                        "Value": "10.2.5.2"
                    }
                ]
            },
            {
                "Key": "netbios-node-type",
                "Values": [
                    {
                        "Value": "2"
                    }
                ]
            }
        ],
        "DhcpOptionsId": "dopt-06d52773eff4c55f3"
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[CreateDhcpOptions](#)中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會建立指定的一組 DHCP 選項。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$options = @( @{Key="domain-name";Values=@("abc.local")}, @{Key="domain-name-
servers";Values=@("10.0.0.101","10.0.0.102")})
New-EC2DhcpOption -DhcpConfiguration $options
```

輸出：

```
DhcpConfigurations                    DhcpOptionsId    Tags
------------------                    -------------    ----
{domain-name, domain-name-servers}    dopt-1a2b3c4d    {}
```

範例 2：在 PowerShell 版本 2 中，您必須使用新物件來建立每個 DHCP 選項。

```
$option1 = New-Object Amazon.EC2.Model.DhcpConfiguration
$option1.Key = "domain-name"
$option1.Values = "abc.local"

$option2 = New-Object Amazon.EC2.Model.DhcpConfiguration
$option2.Key = "domain-name-servers"
$option2.Values = @("10.0.0.101","10.0.0.102")

New-EC2DhcpOption -DhcpConfiguration @($option1, $option2)
```

輸出：

```
DhcpConfigurations                   DhcpOptionsId    Tags
------------------                   -------------    ----
{domain-name, domain-name-servers}   dopt-2a3b4c5d    {}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateDhcpOptions式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateFlowLogs配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateFlowLogs。

CLI

AWS CLI

範例 1：建立流程記錄

下列create-flow-logs範例會建立流程記錄，以擷取指定網路介面的所有拒絕流量。流程記錄會使用指定 IAM 角色中的許可傳送至 CloudWatch 記錄檔中的日誌群組。

```
aws ec2 create-flow-logs \
    --resource-type NetworkInterface \
    --resource-ids eni-11223344556677889 \
    --traffic-type REJECT \
```

```
    --log-group-name my-flow-logs \
    --deliver-logs-permission-arn arn:aws:iam::123456789101:role/publishFlowLogs
```

輸出：

```
{
    "ClientToken": "so0eNA2uSHUNlHI0S2cJ305GuIX1CezaRdGtexample",
    "FlowLogIds": [
        "fl-12345678901234567"
    ],
    "Unsuccessful": []
}
```

如需詳細資訊，請參閱 Amazon VPC 使用者指南中的 VPC 流量日誌。

範例 2：使用自訂格式建立流程記錄

下列create-flow-logs範例會建立一個流程日誌，以擷取指定 VPC 的所有流量，並將流程日誌傳送到 Amazon S3 儲存貯體。--log-format 參數會指定流量日誌記錄的自訂格式。若要在 Windows 上執行此命令，請將單引號 (') 變更為雙引號 (「)。

```
aws ec2 create-flow-logs \
    --resource-type VPC \
    --resource-ids vpc-00112233344556677 \
    --traffic-type ALL \
    --log-destination-type s3 \
    --log-destination arn:aws:s3:::flow-log-bucket/my-custom-flow-logs/ \
    --log-format '${version} ${vpc-id} ${subnet-id} ${instance-id} ${srcaddr}
 ${dstaddr} ${srcport} ${dstport} ${protocol} ${tcp-flags} ${type} ${pkt-srcaddr}
 ${pkt-dstaddr}'
```

如需詳細資訊，請參閱 Amazon VPC 使用者指南中的 VPC 流量日誌。

範例 3：建立具有一分鐘最大彙總間隔的流程記錄

下列create-flow-logs範例會建立一個流程日誌，以擷取指定 VPC 的所有流量，並將流程日誌傳送到 Amazon S3 儲存貯體。此--max-aggregation-interval參數指定 60 秒 (1 分鐘) 的最大聚總間隔。

```
aws ec2 create-flow-logs \
    --resource-type VPC \
```

```
    --resource-ids vpc-00112233344556677 \
    --traffic-type ALL \
    --log-destination-type s3 \
    --log-destination arn:aws:s3:::flow-log-bucket/my-custom-flow-logs/ \
    --max-aggregation-interval 60
```

如需詳細資訊，請參閱 Amazon VPC 使用者指南中的 VPC 流量日誌。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateFlowLogs中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例使用「管理員」角色的屬性，為子網路子網 1d234567 建立 EC2 流程日誌，並為所有「拒絕」流量建立 cloud-watch-log 指定的「子網路 1 日誌」

```
New-EC2FlowLog -ResourceId "subnet-1d234567" -LogDestinationType cloud-watch-
logs -LogGroupName subnet1-log -TrafficType "REJECT" -ResourceType Subnet -
DeliverLogsPermissionArn "arn:aws:iam::98765432109:role/Admin"
```

輸出：

```
ClientToken                                  FlowLogIds             Unsuccessful
-----------                                  ----------             ------------
m1VN2cxP3iB4qo//VUKl5EU6cF7gQLOxcqNefvjeTGw= {fl-012fc34eed5678c9d} {}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateFlowLogs式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateImage配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateImage。

CLI

AWS CLI

範例 1：從 Amazon EBS 支援的執行個體建立 AMI

下列create-image範例會從指定的執行個體建立 AMI。

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "My server" \
    --description "An AMI for my server"
```

輸出：

```
{
    "ImageId": "ami-abcdef01234567890"
}
```

如需有關為 AMI 指定區塊裝置對應的詳細資訊，請參閱 Amazon EC2 使用者指南中的指定 AMI 的區塊裝置對應。

範例 2：在不重新開機的情況下從 Amazon EBS 支援的執行個體建立 AMI

下列create-image範例會建立 AMI 並設定-no-reboot 參數，以便在建立映像檔之前不會重新開機執行個體。

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "My server" \
    --no-reboot
```

輸出：

```
{
    "ImageId": "ami-abcdef01234567890"
}
```

如需有關為 AMI 指定區塊裝置對應的詳細資訊，請參閱 Amazon EC2 使用者指南中的指定 AMI 的區塊裝置對應。

範例 3：在建立時標記 AMI 和快照

下列create-image範例會建立 AMI，並使用相同的標籤標記 AMI 和快照 cost-center=cc123

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "My server" \
    --tag-specifications "ResourceType=image,Tags=[{Key=cost-
center,Value=cc123}]" "ResourceType=snapshot,Tags=[{Key=cost-
center,Value=cc123}]"
```

輸出：

```
{
    "ImageId": "ami-abcdef01234567890"
}
```

如需有關在建立時標記資源的詳細資訊，請參閱 Amazon EC2 使用者指南中的在資源建立時新增標籤。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateImage中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會從指定的執行個體建立具有指定名稱和說明的 AMI。Amazon EC2 會嘗試在建立映像之前徹底關閉執行個體，並在完成時重新啟動執行個體。

```
New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description "My web
 server AMI"
```

範例 2：此範例會從指定的執行個體建立具有指定名稱和說明的 AMI。Amazon EC2 在不關閉和重新啟動執行個體的情況下建立映像檔；因此，無法保證所建立映像檔上的檔案系統完整性。

```
New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description "My web
 server AMI" -NoReboot $true
```

範例 3：此範例會建立具有三個磁碟區的 AMI。第一個磁碟區是以 Amazon EBS 快照為基礎。第二個卷是一個空的 100 GiB Amazon EBS 卷。第三個磁碟區是執行個體儲存磁碟區。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
$ebsBlock1 = @{SnapshotId="snap-1a2b3c4d"}
```

```
$ebsBlock2 = @{VolumeSize=100}

New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description
 "My web server AMI" -BlockDeviceMapping @( @{DeviceName="/dev/sdf";Ebs=
$ebsBlock1}, @{DeviceName="/dev/sdg";Ebs=$ebsBlock2}, @{DeviceName="/dev/
sdc";VirtualName="ephemeral0"})
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateImage式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateInstanceExportTask配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateInstanceExportTask。

CLI

AWS CLI

若要匯出例證

此範例命令會建立一個任務，將執行個體 i-1234567890abcdef0 匯出至 Amazon S3 儲存貯體我的匯出儲存貯體。

命令：

```
aws ec2 create-instance-export-task --description "RHEL5 instance" --instance-
id i-1234567890abcdef0 --target-environment vmware --export-to-s3-task
 DiskImageFormat=vmdk,ContainerFormat=ova,S3Bucket=myexportbucket,S3Prefix=RHEL5
```

輸出：

```
{
    "ExportTask": {
        "State": "active",
        "InstanceExportDetails": {
            "InstanceId": "i-1234567890abcdef0",
            "TargetEnvironment": "vmware"
        },
        "ExportToS3Task": {
```

```
            "S3Bucket": "myexportbucket",
            "S3Key": "RHEL5export-i-fh8sjjsq.ova",
            "DiskImageFormat": "vmdk",
            "ContainerFormat": "ova"
        },
        "Description": "RHEL5 instance",
        "ExportTaskId": "export-i-fh8sjjsq"
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateInstanceExportTask中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例將停止的執行個體匯出為虛擬硬碟 (VHD) 至 S3 儲存貯**testbucket-export-instances-2019**體。**i-0800b00a00EXAMPLE**目標環境為**Microsoft**，且會新增區域參數，因為執行個體位於該**us-east-1**區域中，而使用者的預設「 AWS 區域」不是 us-east-1。若要取得匯出工作的狀態，請複製此命令結果中的**ExportTaskId**值，然後執行 **Get-EC2ExportTask -ExportTaskId export_task_ID_from_results.**

```
New-EC2InstanceExportTask -InstanceId i-0800b00a00EXAMPLE -
ExportToS3Task_DiskImageFormat VHD -ExportToS3Task_S3Bucket "testbucket-export-
instances-2019" -TargetEnvironment Microsoft -Region us-east-1
```

輸出：

```
Description           :
ExportTaskId          : export-i-077c73108aEXAMPLE
ExportToS3Task        : Amazon.EC2.Model.ExportToS3Task
InstanceExportDetails : Amazon.EC2.Model.InstanceExportDetails
State                 : active
StatusMessage         :
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateInstanceExportTask式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateInternetGateway配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateInternetGateway。

CLI

AWS CLI

建立網際網路閘道

下列create-internet-gateway範例會使用標籤建立網際網路閘道Name=my-igw。

```
aws ec2 create-internet-gateway \
    --tag-specifications ResourceType=internet-gateway,Tags=[{Key=Name,Value=my-
igw}]
```

輸出：

```
{
    "InternetGateway": {
        "Attachments": [],
        "InternetGatewayId": "igw-0d0fb496b3994d755",
        "OwnerId": "123456789012",
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-igw"
            }
        ]
    }
}
```

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的網際網路閘道。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateInternetGateway中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會建立網際網路閘道。

```
New-EC2InternetGateway
```

輸出：

```
Attachments    InternetGatewayId    Tags
-----------    -----------------    ----
{}             igw-1a2b3c4d         {}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateInternetGateway式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateKeyPair配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateKeyPair。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Create an Amazon EC2 key pair.
    /// </summary>
    /// <param name="keyPairName">The name for the new key pair.</param>
    /// <returns>The Amazon EC2 key pair created.</returns>
```

```
    public async Task<KeyPair?> CreateKeyPair(string keyPairName)
    {
        var request = new CreateKeyPairRequest
        {
            KeyName = keyPairName,
        };

        var response = await _amazonEC2.CreateKeyPairAsync(request);

        if (response.HttpStatusCode == HttpStatusCode.OK)
        {
            var kp = response.KeyPair;
            return kp;
        }
        else
        {
            Console.WriteLine("Could not create key pair.");
            return null;
        }
    }

    /// <summary>
    /// Save KeyPair information to a temporary file.
    /// </summary>
    /// <param name="keyPair">The name of the key pair.</param>
    /// <returns>The full path to the temporary file.</returns>
    public string SaveKeyPair(KeyPair keyPair)
    {
        var tempPath = Path.GetTempPath();
        var tempFileName = $"{tempPath}\\{Path.GetRandomFileName()}";
        var pemFileName = Path.ChangeExtension(tempFileName, "pem");

        // Save the key pair to a file in a temporary folder.
        using var stream = new FileStream(pemFileName, FileMode.Create);
        using var writer = new StreamWriter(stream);
        writer.WriteLine(keyPair.KeyMaterial);

        return pemFileName;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考CreateKeyPair中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
#############################################################################
# function ec2_create_keypair
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or
 2048-bit RSA key pair
# and writes it to a file.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#       -f file_path - File to store the key pair.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function ec2_create_keypair() {
  local key_pair_name file_path response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_create_keypair"
    echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or 2048-
bit RSA key pair"
    echo " and writes it to a file."
    echo "  -n key_pair_name - A key pair name."
    echo "  -f file_path - File to store the key pair."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:f:h" option; do
```

```
    case "${option}" in
      n) key_pair_name="${OPTARG}" ;;
      f) file_path="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

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

  if [[ -z "$file_path" ]]; then
    errecho "ERROR: You must provide a file path with the -f parameter."
    usage
    return 1
  fi

  response=$(aws ec2 create-key-pair \
    --key-name "$key_pair_name" \
    --query 'KeyMaterial' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  }

  if [[ -n "$file_path" ]]; then
    echo "$response" >"$file_path"
  fi

  return 0
}
```

此範例中使用的公用程式函數。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi
```

```
  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateKeyPair中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   Aws::EC2::EC2Client ec2Client(clientConfiguration);
   Aws::EC2::Model::CreateKeyPairRequest request;
   request.SetKeyName(keyPairName);

   Aws::EC2::Model::CreateKeyPairOutcome outcome =
ec2Client.CreateKeyPair(request);
   if (!outcome.IsSuccess()) {
       std::cerr << "Failed to create key pair:" <<
                 outcome.GetError().GetMessage() << std::endl;
   }
   else {
       std::cout << "Successfully created key pair named " <<
                 keyPairName << std::endl;
   }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考CreateKeyPair中的。

CLI

AWS CLI

建立一組金鑰對

此範例會建立名稱為 MyKeyPair 的金鑰對。

命令：

```
aws ec2 create-key-pair --key-name MyKeyPair
```

輸出是 ASCII 版本的私有金鑰和金鑰指紋。您需要將金鑰儲存到檔案。

如需詳細資訊，請參閱《AWS 命令行介面使用者指南》中的「使用金鑰對」。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateKeyPair中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   public static void createKeyPair(Ec2Client ec2, String keyName, String
fileName) {
       try {
           CreateKeyPairRequest request = CreateKeyPairRequest.builder()
                   .keyName(keyName)
                   .build();

           CreateKeyPairResponse response = ec2.createKeyPair(request);
           String content = response.keyMaterial();
           BufferedWriter writer = new BufferedWriter(new FileWriter(fileName));
           writer.write(content);
           writer.close();
           System.out.println("Successfully created key pair named " + keyName);

       } catch (Ec2Exception | IOException 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CreateKeyPair中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { CreateKeyPairCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Create a key pair in Amazon EC2.
    const { KeyMaterial, KeyName } = await client.send(
      // A unique name for the key pair. Up to 255 ASCII characters.
      new CreateKeyPairCommand({ KeyName: "KEY_PAIR_NAME" }),
    );
    // This logs your private key. Be sure to save it.
    console.log(KeyName);
    console.log(KeyMaterial);
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考CreateKeyPair中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun createEC2KeyPair(keyNameVal: String) {
    val request =
        CreateKeyPairRequest {
            keyName = keyNameVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.createKeyPair(request)
        println("The key ID is ${response.keyPairId}")
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK CreateKeyPair中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會建立 key pair，並擷取具有指定名稱之檔案中的 PEM 編碼 RSA 私密金鑰。當您使用時 PowerShell，編碼必須設置為 ascii 才能生成有效的密鑰。如需詳細資訊，請參閱 AWS 命令列介面使用者指南中的建立、顯示和刪除 Amazon EC2 金鑰配對 (https://docs.aws.amazon.com/cli/latest/userguide/ cli-services-ec 2-keypairs.html)。

```
(New-EC2KeyPair -KeyName "my-key-pair").KeyMaterial | Out-File -Encoding ascii -
FilePath C:\path\my-key-pair.pem
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateKeyPair式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class KeyPairWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair
 actions."""

    def __init__(self, ec2_resource, key_file_dir, key_pair=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param key_file_dir: The folder where the private key information is
 stored.
                             This should be a secure folder.
        :param key_pair: A Boto3 KeyPair object. This is a high-level object that
                         wraps key pair actions.
        """
        self.ec2_resource = ec2_resource
        self.key_pair = key_pair
        self.key_file_path = None
        self.key_file_dir = key_file_dir

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource, tempfile.TemporaryDirectory())


    def create(self, key_name):
        """
        Creates a key pair that can be used to securely connect to an EC2
 instance.
```

```
        The returned key pair contains private key information that cannot be
retrieved
        again. The private key data is stored as a .pem file.

        :param key_name: The name of the key pair to create.
        :return: A Boto3 KeyPair object that represents the newly created key
pair.
        """
        try:
            self.key_pair = self.ec2_resource.create_key_pair(KeyName=key_name)
            self.key_file_path = os.path.join(
                self.key_file_dir.name, f"{self.key_pair.name}.pem"
            )
            with open(self.key_file_path, "w") as key_file:
                key_file.write(self.key_pair.key_material)
        except ClientError as err:
            logger.error(
                "Couldn't create key %s. Here's why: %s: %s",
                key_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.key_pair
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件CreateKeyPair中的 Python (博托 3) API 參考。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
# This code example does the following:
```

```
# 1. Creates a key pair in Amazon Elastic Compute Cloud (Amazon EC2).
# 2. Displays information about available key pairs.
# 3. Deletes the key pair.

require "aws-sdk-ec2"

# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param key_pair_name [String] The name for the key pair and private
#   key file.
# @return [Boolean] true if the key pair and private key file were
#   created; otherwise, false.
# @example
#   exit 1 unless key_pair_created?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'my-key-pair'
#   )
def key_pair_created?(ec2_client, key_pair_name)
  key_pair = ec2_client.create_key_pair(key_name: key_pair_name)
  puts "Created key pair '#{key_pair.key_name}' with fingerprint " \
    "'#{key_pair.key_fingerprint}' and ID '#{key_pair.key_pair_id}'."
  filename = File.join(Dir.home, key_pair_name + ".pem")
  File.open(filename, "w") { |file| file.write(key_pair.key_material) }
  puts "Private key file saved locally as '#{filename}'."
  return true
rescue Aws::EC2::Errors::InvalidKeyPairDuplicate
  puts "Error creating key pair: a key pair named '#{key_pair_name}' " \
    "already exists."
  return false
rescue StandardError => e
  puts "Error creating key pair or saving private key file: #{e.message}"
  return false
end

# Displays information about available key pairs in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @example
#   describe_key_pairs(Aws::EC2::Client.new(region: 'us-west-2'))
def describe_key_pairs(ec2_client)
  result = ec2_client.describe_key_pairs
  if result.key_pairs.count.zero?
    puts "No key pairs found."
  else
```

```
    puts "Key pair names:"
    result.key_pairs.each do |key_pair|
      puts key_pair.key_name
    end
  end
rescue StandardError => e
  puts "Error getting information about key pairs: #{e.message}"
end

# Deletes a key pair in Amazon Elastic Compute Cloud (Amazon EC2).
#
# Prerequisites:
#
# - The key pair to delete.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param key_pair_name [String] The name of the key pair to delete.
# @return [Boolean] true if the key pair was deleted; otherwise, false.
# @example
#   exit 1 unless key_pair_deleted?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'my-key-pair'
#   )
def key_pair_deleted?(ec2_client, key_pair_name)
  ec2_client.delete_key_pair(key_name: key_pair_name)
  return true
rescue StandardError => e
  puts "Error deleting key pair: #{e.message}"
  return false
end

# Example usage:
def run_me
  key_pair_name = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-key-pairs.rb KEY_PAIR_NAME REGION"
    puts "Example: ruby ec2-ruby-example-key-pairs.rb my-key-pair us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    key_pair_name = "my-key-pair"
```

```
  region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  key_pair_name = ARGV[0]
  region = ARGV[1]
end

ec2_client = Aws::EC2::Client.new(region: region)

puts "Displaying existing key pair names before creating this key pair..."
describe_key_pairs(ec2_client)

puts "-" * 10
puts "Creating key pair..."
unless key_pair_created?(ec2_client, key_pair_name)
  puts "Stopping program."
  exit 1
end

puts "-" * 10
puts "Displaying existing key pair names after creating this key pair..."
describe_key_pairs(ec2_client)

puts "-" * 10
puts "Deleting key pair..."
unless key_pair_deleted?(ec2_client, key_pair_name)
  puts "Stopping program. You must delete the key pair yourself."
  exit 1
end
puts "Key pair deleted."

puts "-" * 10
puts "Now that the key pair is deleted, " \
  "also deleting the related private key pair file..."
filename = File.join(Dir.home, key_pair_name + ".pem")
File.delete(filename)
if File.exist?(filename)
  puts "Could not delete file at '#{filename}'. You must delete it yourself."
else
  puts "File deleted."
end

puts "-" * 10
puts "Displaying existing key pair names after deleting this key pair..."
```

```
  describe_key_pairs(ec2_client)
end

run_me if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考CreateKeyPair中的。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        oo_result = lo_ec2->createkeypair( iv_keyname = iv_key_name ).
                 " oo_result is returned for testing purposes. "
        MESSAGE 'Amazon EC2 key pair created.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK CreateKeyPair中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateLaunchTemplate配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateLaunchTemplate。

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：

- 建置及管理彈性服務

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   /// <summary>
   /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
   /// The launch template specifies a Bash script in its user data field that
runs after
   /// the instance is started. This script installs the Python packages and
starts a Python
   /// web server on the instance.
   /// </summary>
   /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
   /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
   /// <returns>The template object.</returns>
   public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
   {
       await CreateKeyPair(_keyPairName);
       await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName, _instanceProfileName, instancePolicyPath);

       var startServerText = await File.ReadAllTextAsync(startupScriptPath);
       var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(startServerText);

       var amiLatest = await _amazonSsm.GetParameterAsync(
           new GetParameterRequest() { Name = _amiParam });
       var amiId = amiLatest.Parameter.Value;
       var launchTemplateResponse = await _amazonEc2.CreateLaunchTemplateAsync(
           new CreateLaunchTemplateRequest()
           {
```

```
                LaunchTemplateName = _launchTemplateName,
                LaunchTemplateData = new RequestLaunchTemplateData()
                {
                    InstanceType = _instanceType,
                    ImageId = amiId,
                    IamInstanceProfile =
                        new

 LaunchTemplateIamInstanceProfileSpecificationRequest()
                        {
                            Name = _instanceProfileName
                        },
                    KeyName = _keyPairName,
                    UserData = System.Convert.ToBase64String(plainTextBytes)
                }
            });
        return launchTemplateResponse.LaunchTemplate;

    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考CreateLaunchTemplate中的。

CLI

AWS CLI

範例 1：建立啟動範本

以下 `create-launch-template` 範例會建立啟動範本，而此範本可指定執行個體啟動所在的子網路、將公有 IP 地址和 IPv6 地址指派給執行個體，並為執行個體建立標籤。

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForWebServer \
    --version-description WebVersion1 \
    --launch-template-data '{"NetworkInterfaces":
[{"AssociatePublicIpAddress":true,"DeviceIndex":0,"Ipv6AddressCount":1,"SubnetId":"subnet
[{"ResourceType":"instance","Tags":[{"Key":"purpose","Value":"webserver"}]}]}'
```

輸出：

```
{
```

```
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-01238c059e3466abc",
        "LaunchTemplateName": "TemplateForWebServer",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2019-01-27T09:13:24.000Z"
    }
}
```

如需詳細資訊，請參閱《Amazon Elastic Compute Cloud 使用者指南》中的「從啟動範本啟動執行個體」。如需有關引用 JSON 格式參數的詳細資訊，請參閱《AWS 命令行介面使用者指南》中的「引用字串」。

範例 2：為 Amazon EC2 Auto Scaling 建立啟動範本

以下 `create-launch-template` 範例會建立具備多個標籤和區塊型裝置映射的啟動範本，以指定執行個體啟動時的額外 EBS 磁碟區。請為 `Groups`，即對應至 Auto Scaling 群組在其中啟動執行個體的 VPC 安全群組指定數值。將 VPC 和子網路指定為 Auto Scaling 群組的屬性。

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForAutoScaling \
    --version-description AutoScalingVersion1 \
    --launch-template-data '{"NetworkInterfaces":
[{"DeviceIndex":0,"AssociatePublicIpAddress":true,"Groups":
["sg-7c227019,sg-903004f8"],"DeleteOnTermination":true}],"ImageId":"ami-
b42209de","InstanceType":"m4.large","TagSpecifications":
[{"ResourceType":"instance","Tags":[{"Key":"environment","Value":"production"},
{"Key":"purpose","Value":"webserver"}]},{"ResourceType":"volume","Tags":
[{"Key":"environment","Value":"production"},{"Key":"cost-
center","Value":"cc123"}]}],"BlockDeviceMappings":[{"DeviceName":"/dev/
sda1","Ebs":{"VolumeSize":100}}]}' --region us-east-1
```

輸出：

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-0123c79c33a54e0abc",
        "LaunchTemplateName": "TemplateForAutoScaling",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
```

```
        "CreateTime": "2019-04-30T18:16:06.000Z"
    }
}
```

如需詳細資訊，請參閱《Amazon EC2 Auto Scaling 使用者指南》中的「建立 Auto Scaling 群組的啟動範本」。如需有關引用 JSON 格式參數的詳細資訊，請參閱《AWS 命令行介面使用者指南》中的「引用字串」。

範例 3：建立指定 EBS 磁碟區加密的啟動範本

下列 create-launch-template 範例會建立啟動範本，其中包含從未加密快照中建立的已加密 EBS 磁碟區。此範本也會在建立期間標記磁碟區。如果預設為停用加密，則您必須指定 "Encrypted" 選項，如下列範例所示。如果您使用 "KmsKeyId" 選項來指定客戶受管的 CMK，即使預設為啟用加密，您也必須指定 "Encrypted" 選項。

```
aws ec2 create-launch-template \
  --launch-template-name TemplateForEncryption \
  --launch-template-data file://config.json
```

config.json 的內容：

```
{
    "BlockDeviceMappings":[
        {
            "DeviceName":"/dev/sda1",
            "Ebs":{
                "VolumeType":"gp2",
                "DeleteOnTermination":true,
                "SnapshotId":"snap-066877671789bd71b",
                "Encrypted":true,
                "KmsKeyId":"arn:aws:kms:us-east-1:012345678910:key/abcd1234-
a123-456a-a12b-a123b4cd56ef"
            }
        }
    ],
    "ImageId":"ami-00068cd7555f543d5",
    "InstanceType":"c5.large",
    "TagSpecifications":[
        {
            "ResourceType":"volume",
            "Tags":[
                {
```

```
                    "Key":"encrypted",
                    "Value":"yes"
                }
            ]
        }
    ]
}
```

輸出：

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-0d5bd51bcf8530abc",
        "LaunchTemplateName": "TemplateForEncryption",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2020-01-07T19:08:36.000Z"
    }
}
```

如需詳細資訊，請參閱《Amazon Elastic Compute Cloud 使用者指南》中的「從快照還原 Amazon EBS 磁碟區」和「預設加密」。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateLaunchTemplate中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
```

```
    );
    const ec2Client = new EC2Client({});
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考CreateLaunchTemplate中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

此範例會建立一個啟動範本，其中包含可授予執行個體特定許可的執行個體設定檔，以及在執行個體啟動後在其上執行的使用者資料 Bash 指令碼。

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
```

```
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def create_template(self, server_startup_script_file, instance_policy_file):
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
```

```
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )
            with open(server_startup_script_file) as file:
                start_server_script = file.read()
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
            template = lt_response["LaunchTemplate"]
            log.info(
                "Created launch template %s for AMI %s on %s.",
                self.launch_template_name,
                ami_id,
                self.inst_type,
            )
        except ClientError as err:
```

```
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.AlreadyExistsException"
            ):
                log.info(
                    "Launch template %s already exists, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create launch template
{self.launch_template_name}: {err}."
                )
        return template
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件CreateLaunchTemplate中的 Python (博托 3) API 參考。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateNetworkAcl配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateNetworkAcl。

CLI

AWS CLI

建立網路 ACL 的步驟

此範例會為指定的 VPC 建立網路 ACL。

命令：

```
aws ec2 create-network-acl --vpc-id vpc-a01106c2
```

輸出：

```
{
```

```
    "NetworkAcl": {
        "Associations": [],
        "NetworkAclId": "acl-5fb85d36",
        "VpcId": "vpc-a01106c2",
        "Tags": [],
        "Entries": [
            {
                "CidrBlock": "0.0.0.0/0",
                "RuleNumber": 32767,
                "Protocol": "-1",
                "Egress": true,
                "RuleAction": "deny"
            },
            {
                "CidrBlock": "0.0.0.0/0",
                "RuleNumber": 32767,
                "Protocol": "-1",
                "Egress": false,
                "RuleAction": "deny"
            }
        ],
        "IsDefault": false
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateNetworkAcl中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會為指定的 VPC 建立網路 ACL。

```
New-EC2NetworkAcl -VpcId vpc-12345678
```

輸出：

```
Associations : {}
Entries      : {Amazon.EC2.Model.NetworkAclEntry,
 Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
```

```
Tags         : {}
VpcId        : vpc-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateNetworkAcl式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateNetworkAclEntry配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateNetworkAclEntry。

CLI

AWS CLI

建立網路 ACL 項目的步驟

此範例會為指定的網路 ACL 建立項目。此規則允許從 UDP 連接埠 53 (DNS) 上的任何 IPv4 位址 (0.0.0.0/0) 輸入流量至任何關聯的子網路。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 create-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 100 --protocol udp --port-range From=53,To=53 --cidr-block 0.0.0.0/0 --rule-action allow
```

此範例會針對指定的網路 ACL 建立規則，以允許來自 TCP 連接埠 80 (HTTP) 上任何 IPv6 位址 (:: /0) 的輸入流量。

命令：

```
aws ec2 create-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 120 --protocol tcp --port-range From=80,To=80 --ipv6-cidr-block ::/0 --rule-action allow
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateNetworkAclEntry中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會為指定的網路 ACL 建立項目。此規則允許從 UDP 連接埠 53 (DNS) 上的任何位置 (0.0.0.0/0) 傳入流量進入任何關聯的子網路。

```
New-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber
 100 -Protocol 17 -PortRange_From 53 -PortRange_To 53 -CidrBlock 0.0.0.0/0 -
RuleAction allow
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateNetworkAclEntry式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateNetworkInterface配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateNetworkInterface。

CLI

AWS CLI

範例 1：指定網路介面的 IPv4 位址

下列create-network-interface範例會使用指定的主要 IPv4 位址，為指定的子網路建立網路介面。

```
aws ec2 create-network-interface \
    --subnet-id subnet-00a24d0d67acf6333 \
    --description "my network interface" \
    --groups sg-09dfba7ed20cda78b \
    --private-ip-address 10.0.8.17
```

輸出：

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
```

```
        "Description": "my network interface",
        "Groups": [
            {
                "GroupName": "my-security-group",
                "GroupId": "sg-09dfba7ed20cda78b"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [],
        "MacAddress": "06:6a:0f:9a:49:37",
        "NetworkInterfaceId": "eni-0492b355f0cf3b3f8",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.17",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-17.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.17"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b"
    }
}
```

範例 2：使用 IPv4 位址和 IPv6 位址建立網路介面

下列create-network-interface範例會使用 Amazon EC2 選取的 IPv4 位址和 IPv6 位址，為指定的子網路建立網路界面。

```
aws ec2 create-network-interface \
    --subnet-id subnet-00a24d0d67acf6333 \
    --description "my dual stack network interface" \
    --ipv6-address-count 1 \
    --groups sg-09dfba7ed20cda78b
```

輸出：

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "Description": "my dual stack network interface",
        "Groups": [
            {
                "GroupName": "my-security-group",
                "GroupId": "sg-09dfba7ed20cda78b"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [
            {
                "Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7",
                "IsPrimaryIpv6": false
            }
        ],
        "MacAddress": "06:b8:68:d2:b2:2d",
        "NetworkInterfaceId": "eni-05da417453f9a84bf",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.18",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.18"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b",
        "Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7"
    }
}
```

範例 3：建立具有連線追蹤組態選項的網路介面

下列create-network-interface範例會建立網路介面，並設定閒置連線追蹤逾時。

```
aws ec2 create-network-interface \
    --subnet-id subnet-00a24d0d67acf6333 \
    --groups sg-02e57dbcfe0331c1b \
    --connection-tracking-specification TcpEstablishedTimeout=86400,UdpTimeout=60
```

輸出：

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "ConnectionTrackingConfiguration": {
            "TcpEstablishedTimeout": 86400,
            "UdpTimeout": 60
        },
        "Description": "",
        "Groups": [
            {
                "GroupName": "my-security-group",
                "GroupId": "sg-02e57dbcfe0331c1b"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [],
        "MacAddress": "06:4c:53:de:6d:91",
        "NetworkInterfaceId": "eni-0c133586e08903d0b",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-94.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.94",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-94.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.94"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b"
    }
```

```
}
```

範例 4：建立彈性織物適配器

下列create-network-interface範例會建立 EFA。

```
aws ec2 create-network-interface \
    --interface-type efa \
    --subnet-id subnet-00a24d0d67acf6333 \
    --description "my efa" \
    --groups sg-02e57dbcfe0331c1b
```

輸出：

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "Description": "my efa",
        "Groups": [
            {
                "GroupName": "my-efa-sg",
                "GroupId": "sg-02e57dbcfe0331c1b"
            }
        ],
        "InterfaceType": "efa",
        "Ipv6Addresses": [],
        "MacAddress": "06:d7:a4:f7:4d:57",
        "NetworkInterfaceId": "eni-034acc2885e862b65",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-180.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.180",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-180.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.180"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
```

```
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b"
    }
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的彈性網路界面。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateNetworkInterface中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會建立指定的網路介面。

```
New-EC2NetworkInterface -SubnetId subnet-1a2b3c4d -Description "my network
 interface" -Group sg-12345678 -PrivateIpAddress 10.0.0.17
```

輸出：

```
Association        :
Attachment         :
AvailabilityZone   : us-west-2c
Description        : my network interface
Groups             : {my-security-group}
MacAddress         : 0a:72:bc:1a:cd:7f
NetworkInterfaceId : eni-12345678
OwnerId            : 123456789012
PrivateDnsName     : ip-10-0-0-17.us-west-2.compute.internal
PrivateIpAddress   : 10.0.0.17
PrivateIpAddresses : {}
RequesterId        :
RequesterManaged   : False
SourceDestCheck    : True
Status             : pending
SubnetId           : subnet-1a2b3c4d
TagSet             : {}
VpcId              : vpc-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateNetworkInterface式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreatePlacementGroup配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreatePlacementGroup。

CLI

AWS CLI

建立放置群組

此範例指令會建立具有指定名稱的放置群組。

命令：

```
aws ec2 create-placement-group --group-name my-cluster --strategy cluster
```

建立分割區放置群組

此範例指令會建立一個HDFS-Group-A以五個分割區命名的分割區放置群組。

命令：

```
aws ec2 create-placement-group --group-name HDFS-Group-A --strategy partition --
partition-count 5
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreatePlacementGroup中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會建立具有指定名稱的置放群組。

```
New-EC2PlacementGroup -GroupName my-placement-group -Strategy cluster
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreatePlacementGroup式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateRoute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateRoute。

CLI

AWS CLI

建立路線的步驟

此範例會為指定的路由表建立路由。路由符合所有 IPv4 流量 (0.0.0.0/0)，並將其路由至指定的網際網路閘道。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 create-route --route-table-id rtb-22574640 --destination-cidr-block
 0.0.0.0/0 --gateway-id igw-c0a643a9
```

這個例子命令在路由表 rtb-g8ff4ea2 中創建一個路由。此路由會比對 IPv4 CIDR 區塊 10.0.0.0/16 的流量，並將其路由至 VPC 私人雲端對等連線。此路由可將流量導向至 VPC 對等連線中的對等 VPC。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 create-route --route-table-id rtb-g8ff4ea2 --destination-cidr-block
 10.0.0.0/16 --vpc-peering-connection-id pcx-1a2b3c4d
```

此範例會在指定的路由表中建立符合所有 IPv6 流量 (::/0) 的路由，並將其路由至指定的僅限輸出網際網路閘道。

命令：

```
aws ec2 create-route --route-table-id rtb-dce620b8 --destination-ipv6-cidr-
block ::/0 --egress-only-internet-gateway-id eigw-01eadbd45ecd7943f
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateRoute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會為指定的路由表建立指定的路由。路由會比對所有流量，並將其傳送至指定的網際網路閘道。

```
New-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0 -
GatewayId igw-1a2b3c4d
```

輸出：

```
True
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateRoute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateRouteTable配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateRouteTable。

CLI

AWS CLI

建立路由表

此範例會建立指定 VPC 的路由表。

命令：

```
aws ec2 create-route-table --vpc-id vpc-a01106c2
```

輸出：

```
{
    "RouteTable": {
```

```
        "Associations": [],
        "RouteTableId": "rtb-22574640",
        "VpcId": "vpc-a01106c2",
        "PropagatingVgws": [],
        "Tags": [],
        "Routes": [
            {
                "GatewayId": "local",
                "DestinationCidrBlock": "10.0.0.0/16",
                "State": "active"
            }
        ]
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateRouteTable中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會為指定的 VPC 建立路由表。

```
New-EC2RouteTable -VpcId vpc-12345678
```

輸出：

```
Associations    : {}
PropagatingVgws : {}
Routes          : {}
RouteTableId    : rtb-1a2b3c4d
Tags            : {}
VpcId           : vpc-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateRouteTable式參考中的。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
require "aws-sdk-ec2"

# Prerequisites:
#
# - A VPC in Amazon VPC.
# - A subnet in that VPC.
# - A gateway attached to that subnet.
#
# @param ec2_resource [Aws::EC2::Resource] An initialized
#   Amazon Elastic Compute Cloud (Amazon EC2) resource object.
# @param vpc_id [String] The ID of the VPC for the route table.
# @param subnet_id [String] The ID of the subnet for the route table.
# @param gateway_id [String] The ID of the gateway for the route.
# @param destination_cidr_block [String] The destination CIDR block
#   for the route.
# @param tag_key [String] The key portion of the tag for the route table.
# @param tag_value [String] The value portion of the tag for the route table.
# @return [Boolean] true if the route table was created and associated;
#   otherwise, false.
# @example
#   exit 1 unless route_table_created_and_associated?(
#     Aws::EC2::Resource.new(region: 'us-west-2'),
#     'vpc-0b6f769731EXAMPLE',
#     'subnet-03d9303b57EXAMPLE',
#     'igw-06ca90c011EXAMPLE',
#     '0.0.0.0/0',
#     'my-key',
#     'my-value'
#   )
def route_table_created_and_associated?(
  ec2_resource,
  vpc_id,
```

```
  subnet_id,
  gateway_id,
  destination_cidr_block,
  tag_key,
  tag_value
)
  route_table = ec2_resource.create_route_table(vpc_id: vpc_id)
  puts "Created route table with ID '#{route_table.id}'."
  route_table.create_tags(
    tags: [
      {
        key: tag_key,
        value: tag_value
      }
    ]
  )
  puts "Added tags to route table."
  route_table.create_route(
    destination_cidr_block: destination_cidr_block,
    gateway_id: gateway_id
  )
  puts "Created route with destination CIDR block " \
    "'#{destination_cidr_block}' and associated with gateway " \
    "with ID '#{gateway_id}'."
  route_table.associate_with_subnet(subnet_id: subnet_id)
  puts "Associated route table with subnet with ID '#{subnet_id}'."
  return true
rescue StandardError => e
  puts "Error creating or associating route table: #{e.message}"
  puts "If the route table was created but not associated, you should " \
    "clean up by deleting the route table."
  return false
end

# Example usage:
def run_me
  vpc_id = ""
  subnet_id = ""
  gateway_id = ""
  destination_cidr_block = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
```

```
if ARGV[0] == "--help" || ARGV[0] == "-h"
  puts "Usage: ruby ec2-ruby-example-create-route-table.rb " \
    "VPC_ID SUBNET_ID GATEWAY_ID DESTINATION_CIDR_BLOCK " \
    "TAG_KEY TAG_VALUE REGION"
# Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  puts "Example: ruby ec2-ruby-example-create-route-table.rb " \
    "vpc-0b6f769731EXAMPLE subnet-03d9303b57EXAMPLE igw-06ca90c011EXAMPLE " \
    "'0.0.0.0/0' my-key my-value us-west-2"
  exit 1
# If no values are specified at the command prompt, use these default values.
elsif ARGV.count.zero?
  vpc_id = "vpc-0b6f769731EXAMPLE"
  subnet_id = "subnet-03d9303b57EXAMPLE"
  gateway_id = "igw-06ca90c011EXAMPLE"
  destination_cidr_block = "0.0.0.0/0"
  tag_key = "my-key"
  tag_value = "my-value"
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  vpc_id = ARGV[0]
  subnet_id = ARGV[1]
  gateway_id = ARGV[2]
  destination_cidr_block = ARGV[3]
  tag_key = ARGV[4]
  tag_value = ARGV[5]
  region = ARGV[6]
end

ec2_resource = Aws::EC2::Resource.new(region: region)

if route_table_created_and_associated?(
  ec2_resource,
  vpc_id,
  subnet_id,
  gateway_id,
  destination_cidr_block,
  tag_key,
  tag_value
)
  puts "Route table created and associated."
else
  puts "Route table not created or not associated."
```

```
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考CreateRouteTable中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateSecurityGroup配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateSecurityGroup。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Create an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupName">The name for the new security group.</param>
    /// <param name="groupDescription">A description of the new security group.</
param>
    /// <returns>The group Id of the new security group.</returns>
    public async Task<string> CreateSecurityGroup(string groupName, string
 groupDescription)
```

```
    {
        var response = await _amazonEC2.CreateSecurityGroupAsync(
            new CreateSecurityGroupRequest(groupName, groupDescription));

        return response.GroupId;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考CreateSecurityGroup中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
###############################################################################
# function ec2_create_security_group
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) security
 group.
#
# Parameters:
#       -n security_group_name - The name of the security group.
#       -d security_group_description - The description of the security group.
#
# Returns:
#       The ID of the created security group, or an error message if the
 operation fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_create_security_group() {
  local security_group_name security_group_description response
```

```
 # Function to display usage information
 function usage() {
   echo "function ec2_create_security_group"
   echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group."
   echo "  -n security_group_name - The name of the security group."
   echo "  -d security_group_description - The description of the security
group."
   echo ""
 }

 # Parse the command-line arguments
 while getopts "n:d:h" option; do
   case "${option}" in
     n) security_group_name="${OPTARG}" ;;
     d) security_group_description="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 # Validate the input parameters
 if [[ -z "$security_group_name" ]]; then
   errecho "ERROR: You must provide a security group name with the -n
parameter."
   return 1
 fi

 if [[ -z "$security_group_description" ]]; then
   errecho "ERROR: You must provide a security group description with the -d
parameter."
   return 1
 fi

 # Create the security group
 response=$(aws ec2 create-security-group \
   --group-name "$security_group_name" \
```

```
    --description "$security_group_description" \
    --query "GroupId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-security-group operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}
```

此範例中使用的公用程式函數。

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
########################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
```

```
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateSecurityGroup中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::CreateSecurityGroupRequest request;

    request.SetGroupName(groupName);
    request.SetDescription(description);
    request.SetVpcId(vpcID);

    const Aws::EC2::Model::CreateSecurityGroupOutcome outcome =
            ec2Client.CreateSecurityGroup(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to create security group:" <<
```

```
                  outcome.GetError().GetMessage() << std::endl;
        return false;
    }

    std::cout << "Successfully created security group named " << groupName <<
              std::endl;
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考CreateSecurityGroup中的。

CLI

AWS CLI

為 EC2-Classic 建立安全群組

此範例會建立名為 MySecurityGroup 的安全群組。

命令：

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My
 security group"
```

輸出：

```
{
    "GroupId": "sg-903004f8"
}
```

為 EC2-VPC 建立安全群組

此範例會為指定 VPC 建立名為 MySecurityGroup 的安全群組。

命令：

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My
 security group" --vpc-id vpc-1a2b3c4d
```

輸出：

```
{
```

```
    "GroupId": "sg-903004f8"
}
```

如需詳細資訊，請參閱《AWS 命令行介面使用者指南》中的「使用安全群組」。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateSecurityGroup中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   public static String createSecurityGroup(Ec2Client ec2, String groupName,
String groupDesc, String vpcId,
           String myIpAddress) {
       try {
           CreateSecurityGroupRequest createRequest =
CreateSecurityGroupRequest.builder()
                   .groupName(groupName)
                   .description(groupDesc)
                   .vpcId(vpcId)
                   .build();

           CreateSecurityGroupResponse resp =
ec2.createSecurityGroup(createRequest);
           IpRange ipRange = IpRange.builder()
                   .cidrIp(myIpAddress + "/0")
                   .build();

           IpPermission ipPerm = IpPermission.builder()
                   .ipProtocol("tcp")
                   .toPort(80)
                   .fromPort(80)
                   .ipRanges(ipRange)
                   .build();

           IpPermission ipPerm2 = IpPermission.builder()
```

```
                    .ipProtocol("tcp")
                    .toPort(22)
                    .fromPort(22)
                    .ipRanges(ipRange)
                    .build();

            AuthorizeSecurityGroupIngressRequest authRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                    .groupName(groupName)
                    .ipPermissions(ipPerm, ipPerm2)
                    .build();

            ec2.authorizeSecurityGroupIngress(authRequest);
            System.out.println("Successfully added ingress policy to security
group " + groupName);
            return resp.groupId();

        } catch (Ec2Exception 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CreateSecurityGroup中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { CreateSecurityGroupCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
```

```
  const command = new CreateSecurityGroupCommand({
    // Up to 255 characters in length. Cannot start with sg-.
    GroupName: "SECURITY_GROUP_NAME",
    // Up to 255 characters in length.
    Description: "DESCRIPTION",
  });

  try {
    const { GroupId } = await client.send(command);
    console.log(GroupId);
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考[CreateSecurityGroup](#)中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在[AWS 設定和執行程式碼範例儲存庫](#)。

```
suspend fun createEC2SecurityGroup(
    groupNameVal: String?,
    groupDescVal: String?,
    vpcIdVal: String?,
): String? {
    val request =
        CreateSecurityGroupRequest {
            groupName = groupNameVal
            description = groupDescVal
            vpcId = vpcIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val resp = ec2.createSecurityGroup(request)
```

```
        val ipRange =
            IpRange {
                cidrIp = "0.0.0.0/0"
            }

        val ipPerm =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 80
                fromPort = 80
                ipRanges = listOf(ipRange)
            }

        val ipPerm2 =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 22
                fromPort = 22
                ipRanges = listOf(ipRange)
            }

        val authRequest =
            AuthorizeSecurityGroupIngressRequest {
                groupName = groupNameVal
                ipPermissions = listOf(ipPerm, ipPerm2)
            }
        ec2.authorizeSecurityGroupIngress(authRequest)
        println("Successfully added ingress policy to Security Group
 $groupNameVal")
        return resp.groupId
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK CreateSecurityGroup中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會為指定的 VPC 建立安全性群組。

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security
 group" -VpcId vpc-12345678
```

輸出：

```
sg-12345678
```

範例 2：此範例會為 EC2-Classic 建立安全性群組。

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security
 group"
```

輸出：

```
sg-45678901
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateSecurityGroup式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class SecurityGroupWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) security group
 actions."""

    def __init__(self, ec2_resource, security_group=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
```

```
        :param security_group: A Boto3 SecurityGroup object. This is a high-level
object
                               that wraps security group actions.
        """
        self.ec2_resource = ec2_resource
        self.security_group = security_group

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def create(self, group_name, group_description):
        """
        Creates a security group in the default virtual private cloud (VPC) of
the
        current account.

        :param group_name: The name of the security group to create.
        :param group_description: The description of the security group to
create.
        :return: A Boto3 SecurityGroup object that represents the newly created
security group.
        """
        try:
            self.security_group = self.ec2_resource.create_security_group(
                GroupName=group_name, Description=group_description
            )
        except ClientError as err:
            logger.error(
                "Couldn't create security group %s. Here's why: %s: %s",
                group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.security_group
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件CreateSecurityGroup中的 Python (博托 3) API 參考。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
# This code example does the following:
# 1. Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group.
# 2. Adds inbound rules to the security group.
# 3. Displays information about available security groups.
# 4. Deletes the security group.

require "aws-sdk-ec2"

# Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Prerequisites:
#
# - A VPC in Amazon Virtual Private Cloud (Amazon VPC).
#
# @param ec2_client [Aws::EC2::Client] An initialized
#   Amazon EC2 client.
# @param group_name [String] A name for the security group.
# @param description [String] A description for the security group.
# @param vpc_id [String] The ID of the VPC for the security group.
# @return [String] The ID of security group that was created.
# @example
#   puts create_security_group(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'my-security-group',
#     'This is my security group.',
#     'vpc-6713dfEX'
#   )
```

```
def create_security_group(
  ec2_client,
  group_name,
  description,
  vpc_id
)
  security_group = ec2_client.create_security_group(
    group_name: group_name,
    description: description,
    vpc_id: vpc_id
  )
  puts "Created security group '#{group_name}' with ID " \
    "'#{security_group.group_id}' in VPC with ID '#{vpc_id}'."
  return security_group.group_id
rescue StandardError => e
  puts "Error creating security group: #{e.message}"
  return "Error"
end

# Adds an inbound rule to an Amazon Elastic Compute Cloud (Amazon EC2)
# security group.
#
# Prerequisites:
#
# - The security group.
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @param security_group_id [String] The ID of the security group.
# @param ip_protocol [String] The network protocol for the inbound rule.
# @param from_port [String] The originating port for the inbound rule.
# @param to_port [String] The destination port for the inbound rule.
# @param cidr_ip_range [String] The CIDR IP range for the inbound rule.
# @return
# @example
#   exit 1 unless security_group_ingress_authorized?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'sg-030a858e078f1b9EX',
#     'tcp',
#     '80',
#     '80',
#     '0.0.0.0/0'
#   )
def security_group_ingress_authorized?(
  ec2_client,
```

```
  security_group_id,
  ip_protocol,
  from_port,
  to_port,
  cidr_ip_range
)
  ec2_client.authorize_security_group_ingress(
    group_id: security_group_id,
    ip_permissions: [
      {
        ip_protocol: ip_protocol,
        from_port: from_port,
        to_port: to_port,
        ip_ranges: [
          {
            cidr_ip: cidr_ip_range
          }
        ]
      }
    ]
  )
  puts "Added inbound rule to security group '#{security_group_id}' for protocol
 " \
    "'#{ip_protocol}' from port '#{from_port}' to port '#{to_port}' " \
    "with CIDR IP range '#{cidr_ip_range}'."
  return true
rescue StandardError => e
  puts "Error adding inbound rule to security group: #{e.message}"
  return false
end

# Displays information about a security group's IP permissions set in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# Prerequisites:
#
# - A security group with inbound rules, outbound rules, or both.
#
# @param p [Aws::EC2::Types::IpPermission] The IP permissions set.
# @example
#   ec2_client = Aws::EC2::Client.new(region: 'us-west-2')
#   response = ec2_client.describe_security_groups
#   unless sg.ip_permissions.empty?
#     describe_security_group_permissions(
```

```
#       response.security_groups[0].ip_permissions[0]
#     )
#   end
def describe_security_group_permissions(perm)
  print "  Protocol: #{perm.ip_protocol == '-1' ? 'All' : perm.ip_protocol}"

  unless perm.from_port.nil?
    if perm.from_port == "-1" || perm.from_port == -1
      print ", From: All"
    else
      print ", From: #{perm.from_port}"
    end
  end

  unless perm.to_port.nil?
    if perm.to_port == "-1" || perm.to_port == -1
      print ", To: All"
    else
      print ", To: #{perm.to_port}"
    end
  end

  if perm.key?(:ipv_6_ranges) && perm.ipv_6_ranges.count.positive?
    print ", CIDR IPv6: #{perm.ipv_6_ranges[0].cidr_ipv_6}"
  end

  if perm.key?(:ip_ranges) && perm.ip_ranges.count.positive?
    print ", CIDR IPv4: #{perm.ip_ranges[0].cidr_ip}"
  end

  print "\n"
end

# Displays information about available security groups in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @example
#   describe_security_groups(Aws::EC2::Client.new(region: 'us-west-2'))
def describe_security_groups(ec2_client)
  response = ec2_client.describe_security_groups

  if response.security_groups.count.positive?
    response.security_groups.each do |sg|
```

```
      puts "-" * (sg.group_name.length + 13)
      puts "Name:        #{sg.group_name}"
      puts "Description: #{sg.description}"
      puts "Group ID:    #{sg.group_id}"
      puts "Owner ID:    #{sg.owner_id}"
      puts "VPC ID:      #{sg.vpc_id}"

      if sg.tags.count.positive?
        puts "Tags:"
        sg.tags.each do |tag|
          puts "  Key: #{tag.key}, Value: #{tag.value}"
        end
      end

      unless sg.ip_permissions.empty?
        puts "Inbound rules:" if sg.ip_permissions.count.positive?
        sg.ip_permissions.each do |p|
          describe_security_group_permissions(p)
        end
      end

      unless sg.ip_permissions_egress.empty?
        puts "Outbound rules:" if sg.ip_permissions.count.positive?
        sg.ip_permissions_egress.each do |p|
          describe_security_group_permissions(p)
        end
      end
    end
  else
    puts "No security groups found."
  end
rescue StandardError => e
  puts "Error getting information about security groups: #{e.message}"
end

# Deletes an Amazon Elastic Compute Cloud (Amazon EC2)
# security group.
#
# Prerequisites:
#
# - The security group.
#
# @param ec2_client [Aws::EC2::Client] An initialized
#   Amazon EC2 client.
```

```
# @param security_group_id [String] The ID of the security group to delete.
# @return [Boolean] true if the security group was deleted; otherwise, false.
# @example
#   exit 1 unless security_group_deleted?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'sg-030a858e078f1b9EX'
#   )
def security_group_deleted?(ec2_client, security_group_id)
  ec2_client.delete_security_group(group_id: security_group_id)
  puts "Deleted security group '#{security_group_id}'."
  return true
rescue StandardError => e
  puts "Error deleting security group: #{e.message}"
  return false
end

# Example usage:
def run_me
  group_name = ""
  description = ""
  vpc_id = ""
  ip_protocol_http = ""
  from_port_http = ""
  to_port_http = ""
  cidr_ip_range_http = ""
  ip_protocol_ssh = ""
  from_port_ssh = ""
  to_port_ssh = ""
  cidr_ip_range_ssh = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-security-group.rb " \
      "GROUP_NAME DESCRIPTION VPC_ID IP_PROTOCOL_1 FROM_PORT_1 TO_PORT_1 " \
      "CIDR_IP_RANGE_1 IP_PROTOCOL_2 FROM_PORT_2 TO_PORT_2 " \
      "CIDR_IP_RANGE_2 REGION"
    puts "Example: ruby ec2-ruby-example-security-group.rb " \
      "my-security-group 'This is my security group.' vpc-6713dfEX " \
      "tcp 80 80 '0.0.0.0/0' tcp 22 22 '0.0.0.0/0' us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    group_name = "my-security-group"
    description = "This is my security group."
```

```
  vpc_id = "vpc-6713dfEX"
  ip_protocol_http = "tcp"
  from_port_http = "80"
  to_port_http = "80"
  cidr_ip_range_http = "0.0.0.0/0"
  ip_protocol_ssh = "tcp"
  from_port_ssh = "22"
  to_port_ssh = "22"
  cidr_ip_range_ssh = "0.0.0.0/0"
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  group_name = ARGV[0]
  description = ARGV[1]
  vpc_id = ARGV[2]
  ip_protocol_http = ARGV[3]
  from_port_http = ARGV[4]
  to_port_http = ARGV[5]
  cidr_ip_range_http = ARGV[6]
  ip_protocol_ssh = ARGV[7]
  from_port_ssh = ARGV[8]
  to_port_ssh = ARGV[9]
  cidr_ip_range_ssh = ARGV[10]
  region = ARGV[11]
end

security_group_id = ""
security_group_exists = false
ec2_client = Aws::EC2::Client.new(region: region)

puts "Attempting to create security group..."
security_group_id = create_security_group(
  ec2_client,
  group_name,
  description,
  vpc_id
)
if security_group_id == "Error"
  puts "Could not create security group. Skipping this step."
else
  security_group_exists = true
end
```

```
  if security_group_exists
    puts "Attempting to add inbound rules to security group..."
    unless security_group_ingress_authorized?(
      ec2_client,
      security_group_id,
      ip_protocol_http,
      from_port_http,
      to_port_http,
      cidr_ip_range_http
    )
      puts "Could not add inbound HTTP rule to security group. " \
        "Skipping this step."
    end

    unless security_group_ingress_authorized?(
      ec2_client,
      security_group_id,
      ip_protocol_ssh,
      from_port_ssh,
      to_port_ssh,
      cidr_ip_range_ssh
    )
      puts "Could not add inbound SSH rule to security group. " \
        "Skipping this step."
    end
  end

  puts "\nInformation about available security groups:"
  describe_security_groups(ec2_client)

  if security_group_exists
    puts "\nAttempting to delete security group..."
    unless security_group_deleted?(ec2_client, security_group_id)
      puts "Could not delete security group. You must delete it yourself."
    end
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考CreateSecurityGroup中的。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        oo_result = lo_ec2->createsecuritygroup(                 " oo_result is
 returned for testing purposes. "
          iv_description = 'Security group example'
          iv_groupname = iv_security_group_name
          iv_vpcid = iv_vpc_id
        ).
        MESSAGE 'Security group created.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK CreateSecurityGroup中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭`CreateSnapshot`配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用`CreateSnapshot`。

CLI

AWS CLI

建立快照

此範例指令會建立磁碟區的快照，其磁碟區識別碼為，vol-1234567890abcdef0並提供識別快照的簡短描述。

命令：

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description "This is
 my root volume snapshot"
```

輸出：

```
{
    "Description": "This is my root volume snapshot",
    "Tags": [],
    "Encrypted": false,
    "VolumeId": "vol-1234567890abcdef0",
    "State": "pending",
    "VolumeSize": 8,
    "StartTime": "2018-02-28T21:06:01.000Z",
    "Progress": "",
    "OwnerId": "012345678910",
    "SnapshotId": "snap-066877671789bd71b"
}
```

使用標籤建立快照

此範例指令會建立快照並套用兩個標籤：目的 = 產品和成本中心 = 123。

命令：

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0
 --description 'Prod backup' --tag-specifications
 'ResourceType=snapshot,Tags=[{Key=purpose,Value=prod},
{Key=costcenter,Value=123}]'
```

輸出：

```
{
    "Description": "Prod backup",
    "Tags": [
        {
            "Value": "prod",
            "Key": "purpose"
```

```
        },
        {
            "Value": "123",
            "Key": "costcenter"
        }
    ],
    "Encrypted": false,
    "VolumeId": "vol-1234567890abcdef0",
    "State": "pending",
    "VolumeSize": 8,
    "StartTime": "2018-02-28T21:06:06.000Z",
    "Progress": "",
    "OwnerId": "012345678910",
    "SnapshotId": "snap-09ed24a70bc19bbe4"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateSnapshot中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會建立指定磁碟區的快照。

```
New-EC2Snapshot -VolumeId vol-12345678 -Description "This is a test"
```

輸出：

```
DataEncryptionKeyId :
Description         : This is a test
Encrypted           : False
KmsKeyId            :
OwnerAlias          :
OwnerId             : 123456789012
Progress            :
SnapshotId          : snap-12345678
StartTime           : 12/22/2015 1:28:42 AM
State               : pending
StateMessage        :
Tags                : {}
VolumeId            : vol-12345678
VolumeSize          : 20
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateSnapshot式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateSpotDatafeedSubscription配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateSpotDatafeedSubscription。

CLI

AWS CLI

建立競價型執行個體資料饋送

下列create-spot-datafeed-subscription範例會建立競價型執行個體資料饋送。

```
aws ec2 create-spot-datafeed-subscription \
    --bucket my-bucket \
    --prefix spot-data-feed
```

輸出：

```
{
    "SpotDatafeedSubscription": {
        "Bucket": "my-bucket",
        "OwnerId": "123456789012",
        "Prefix": "spot-data-feed",
        "State": "Active"
    }
}
```

資料饋送存放在您指定的 Amazon S3 儲存貯體中。此資料饋送的檔案名稱具有下列格式。

```
my-bucket.s3.amazonaws.com/spot-data-feed/123456789012.YYYY-MM-DD-
HH.n.abcd1234.gz
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的競價型執行個體資料饋送。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateSpotDatafeedSubscription中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會建立 Spot 執行個體資料饋送。

```
New-EC2SpotDatafeedSubscription -Bucket my-s3-bucket -Prefix spotdata
```

輸出：

```
Bucket  : my-s3-bucket
Fault   :
OwnerId : 123456789012
Prefix  : spotdata
State   : Active
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateSpotDatafeedSubscription式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateSubnet配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateSubnet。

CLI

AWS CLI

範例 1：建立僅具有 IPv4 CIDR 區塊的子網路

以下 create-subnet 範例會在指定 VPC 內建立具有指定 IPv4 CIDR 區塊的子網路。

```
aws ec2 create-subnet \
    --vpc-id vpc-081ec835f3EXAMPLE \
    --cidr-block 10.0.0.0/24 \
    --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv4-only-
subnet}]
```

輸出：

```
{
    "Subnet": {
        "AvailabilityZone": "us-west-2a",
        "AvailabilityZoneId": "usw2-az2",
        "AvailableIpAddressCount": 251,
        "CidrBlock": "10.0.0.0/24",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-0e99b93155EXAMPLE",
        "VpcId": "vpc-081ec835f3EXAMPLE",
        "OwnerId": "123456789012",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [],
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-ipv4-only-subnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0e99b93155EXAMPLE"
    }
}
```

範例 2：建立具有 IPv4 和 IPv6 CIDR 區塊的子網路

以下 `create-subnet` 範例會在指定 VPC 內建立具有指定 IPv4 和 IPv6 CIDR 區塊的子網路。

```
aws ec2 create-subnet \
    --vpc-id vpc-081ec835f3EXAMPLE \
    --cidr-block 10.0.0.0/24 \
    --ipv6-cidr-block 2600:1f16:cfe:3660::/64 \
    --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv4-ipv6-
subnet}]
```

輸出：

```
{
    "Subnet": {
        "AvailabilityZone": "us-west-2a",
        "AvailabilityZoneId": "usw2-az2",
        "AvailableIpAddressCount": 251,
```

```
        "CidrBlock": "10.0.0.0/24",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-0736441d38EXAMPLE",
        "VpcId": "vpc-081ec835f3EXAMPLE",
        "OwnerId": "123456789012",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [
            {
                "AssociationId": "subnet-cidr-assoc-06c5f904499fcc623",
                "Ipv6CidrBlock": "2600:1f13:cfe:3660::/64",
                "Ipv6CidrBlockState": {
                    "State": "associating"
                }
            }
        ],
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-ipv4-ipv6-subnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0736441d38EXAMPLE"
    }
}
```

範例 3：建立僅具有 IPv6 CIDR 區塊的子網路

以下 `create-subnet` 範例會在指定 VPC 內建立具有指定 IPv6 CIDR 區塊的子網路。

```
aws ec2 create-subnet \
    --vpc-id vpc-081ec835f3EXAMPLE \
    --ipv6-native \
    --ipv6-cidr-block 2600:1f16:115:200::/64 \
    --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv6-only-
subnet}]
```

輸出：

```
{
    "Subnet": {
```

```
        "AvailabilityZone": "us-west-2a",
        "AvailabilityZoneId": "usw2-az2",
        "AvailableIpAddressCount": 0,
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-03f720e7deEXAMPLE",
        "VpcId": "vpc-081ec835f3EXAMPLE",
        "OwnerId": "123456789012",
        "AssignIpv6AddressOnCreation": true,
        "Ipv6CidrBlockAssociationSet": [
            {
                "AssociationId": "subnet-cidr-assoc-01ef639edde556709",
                "Ipv6CidrBlock": "2600:1f13:cfe:3660::/64",
                "Ipv6CidrBlockState": {
                    "State": "associating"
                }
            }
        ],
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-ipv6-only-subnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-03f720e7deEXAMPLE"
    }
}
```

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 和子網路。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateSubnet中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會使用指定的 CIDR 建立子網路。

```
New-EC2Subnet -VpcId vpc-12345678 -CidrBlock 10.0.0.0/24
```

輸出：

```
AvailabilityZone        : us-west-2c
AvailableIpAddressCount : 251
CidrBlock               : 10.0.0.0/24
DefaultForAz            : False
MapPublicIpOnLaunch     : False
State                   : pending
SubnetId                : subnet-1a2b3c4d
Tag                     : {}
VpcId                   : vpc-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateSubnet式參考中的。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
require "aws-sdk-ec2"

# Creates a subnet within a virtual private cloud (VPC) in
# Amazon Virtual Private Cloud (Amazon VPC) and then tags
# the subnet.
#
# Prerequisites:
#
# - A VPC in Amazon VPC.
#
# @param ec2_resource [Aws::EC2::Resource] An initialized
#   Amazon Elastic Compute Cloud (Amazon EC2) resource object.
# @param vpc_id [String] The ID of the VPC for the subnet.
# @param cidr_block [String] The IPv4 CIDR block for the subnet.
# @param availability_zone [String] The ID of the Availability Zone
#   for the subnet.
# @param tag_key [String] The key portion of the tag for the subnet.
# @param tag_vlue [String] The value portion of the tag for the subnet.
```

```
# @return [Boolean] true if the subnet was created and tagged;
#   otherwise, false.
# @example
#   exit 1 unless subnet_created_and_tagged?(
#     Aws::EC2::Resource.new(region: 'us-west-2'),
#     'vpc-6713dfEX',
#     '10.0.0.0/24',
#     'us-west-2a',
#     'my-key',
#     'my-value'
#   )
def subnet_created_and_tagged?(
  ec2_resource,
  vpc_id,
  cidr_block,
  availability_zone,
  tag_key,
  tag_value
)
  subnet = ec2_resource.create_subnet(
    vpc_id: vpc_id,
    cidr_block: cidr_block,
    availability_zone: availability_zone
  )
  subnet.create_tags(
    tags: [
      {
        key: tag_key,
        value: tag_value
      }
    ]
  )
  puts "Subnet created with ID '#{subnet.id}' in VPC with ID '#{vpc_id}' " \
    "and CIDR block '#{cidr_block}' in availability zone " \
    "'#{availability_zone}' and tagged with key '#{tag_key}' and " \
    "value '#{tag_value}'."
  return true
rescue StandardError => e
  puts "Error creating or tagging subnet: #{e.message}"
  return false
end

# Example usage:
def run_me
```

```
vpc_id = ""
cidr_block = ""
availability_zone = ""
tag_key = ""
tag_value = ""
region = ""
# Print usage information and then stop.
if ARGV[0] == "--help" || ARGV[0] == "-h"
  puts "Usage:   ruby ec2-ruby-example-create-subnet.rb " \
    "VPC_ID CIDR_BLOCK AVAILABILITY_ZONE TAG_KEY TAG_VALUE REGION"
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  puts "Example: ruby ec2-ruby-example-create-subnet.rb " \
    "vpc-6713dfEX 10.0.0.0/24 us-west-2a my-key my-value us-west-2"
  exit 1
# If no values are specified at the command prompt, use these default values.
elsif ARGV.count.zero?
  vpc_id = "vpc-6713dfEX"
  cidr_block = "10.0.0.0/24"
  availability_zone = "us-west-2a"
  tag_key = "my-key"
  tag_value = "my-value"
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  vpc_id = ARGV[0]
  cidr_block = ARGV[1]
  availability_zone = ARGV[2]
  tag_key = ARGV[3]
  tag_value = ARGV[4]
  region = ARGV[5]
end

ec2_resource = Aws::EC2::Resource.new(region: region)

if subnet_created_and_tagged?(
  ec2_resource,
  vpc_id,
  cidr_block,
  availability_zone,
  tag_key,
  tag_value
)
  puts "Subnet created and tagged."
```

```
  else
    puts "Subnet not created or not tagged."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考CreateSubnet中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateTags配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateTags。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::Tag nameTag;
    nameTag.SetKey("Name");
    nameTag.SetValue(instanceName);

    Aws::EC2::Model::CreateTagsRequest createRequest;
    createRequest.AddResources(instanceID);
    createRequest.AddTags(nameTag);

    Aws::EC2::Model::CreateTagsOutcome createOutcome = ec2Client.CreateTags(
            createRequest);
    if (!createOutcome.IsSuccess()) {
        std::cerr << "Failed to tag ec2 instance " << instanceID <<
```

```
                " with name " << instanceName << ":" <<
                createOutcome.GetError().GetMessage() << std::endl;
        return false;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考CreateTags中的。

CLI

AWS CLI

範例 1：將標籤新增至資源

以下 create-tags 範例會將標籤 Stack=production 新增至指定的映像，或覆寫 AMI 的現有標籤，其中標籤金鑰為 Stack。

```
aws ec2 create-tags \
    --resources ami-1234567890abcdef0 \
    --tags Key=Stack,Value=production
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的主題標題。

範例 2：若要將標籤新增至多個資源

下列 create-tags 範例會為 AMI 和執行個體新增 (或覆寫) 兩個標籤。其中一個標籤具有索引鍵 (webserver)，但沒有值 (值會設定為空白字串)。另一個標籤則有一個索引鍵 (stack) 和一個值 (Production)。

```
aws ec2 create-tags \
    --resources ami-1a2b3c4d i-1234567890abcdef0 \
    --tags Key=webserver,Value=   Key=stack,Value=Production
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的主題標題。

範例 3：若要新增包含特殊字元的標籤

下列 create-tags 範例會為執行個體新增標籤 [Group]=test。中括號 ([和]) 是特殊字元，必須將其逸出。下列範例也會使用每個環境適用的行接續字元。

如果您使用 Windows，請以雙引號 (") 括住具有特殊字元的元素，然後在每個雙引號字元前面加上反斜線 (\)，如下所示：

```
aws ec2 create-tags ^
    --resources i-1234567890abcdef0 ^
    --tags Key=\"[Group]\",Value=test
```

如果您使用的是 Windows PowerShell，請將具有特殊字元的值以雙引號 (「) 括住元素，在每個雙引號字元前加上反斜線 (\)，然後用單引號 (') 圍住整個索引鍵和值結構，如下所示：

```
aws ec2 create-tags `
    --resources i-1234567890abcdef0 `
    --tags 'Key=\"[Group]\",Value=test'
```

如果您是使用 Linux 或 OS X，請以雙引號 (") 括住具有特殊字元的值，然後使用單引號 (') 括住整個索引鍵和值結構，如下所示：

```
aws ec2 create-tags \
    --resources i-1234567890abcdef0 \
    --tags 'Key="[Group]",Value=test'
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的主題標題。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateTags中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將單一標籤新增至指定的資源。標籤鍵是「行李通」，標籤值為 'myTagValue'。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
New-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag"; Value="myTagValue" }
```

範例 2：此範例會更新指定的標籤，或將指定的標籤新增至指定的資源。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
New-EC2Tag -Resource i-12345678 -Tag @( @{ Key="myTag"; Value="newTagValue" },
 @{ Key="test"; Value="anotherTagValue" } )
```

範例 3：在 PowerShell 版本 2 中，您必須使用新物件來建立標籤參數的標籤。

```
$tag = New-Object Amazon.EC2.Model.Tag
```

```
$tag.Key = "myTag"
$tag.Value = "myTagValue"

New-EC2Tag -Resource i-12345678 -Tag $tag
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateTags式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateVolume配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateVolume。

CLI

AWS CLI

若要建立空的一般用途 SSD (gp2) 磁碟區

下列create-volume範例會在指定的可用區域中建立 80 GiB 一般用途 SSD (gp2) 磁碟區。請注意，目前的「區域」必須是us-east-1，或者您可以加入--region參數以指定指令的「區域」(Region)。

```
aws ec2 create-volume \
    --volume-type gp2 \
    --size 80 \
    --availability-zone us-east-1a
```

輸出：

```
{
    "AvailabilityZone": "us-east-1a",
    "Tags": [],
    "Encrypted": false,
    "VolumeType": "gp2",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "creating",
    "Iops": 240,
    "SnapshotId": "",
    "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "Size": 80
```

```
}
```

如果您未指定磁碟區類型，則預設磁碟區類型為gp2。

```
aws ec2 create-volume \
    --size 80 \
    --availability-zone us-east-1a
```

範例 2：從快照建立佈建 IOPS SSD (io1) 磁碟區

下列create-volume範例會使用指定的快照，在指定的可用區域中建立具有 1000 個已佈建 IOPS IOPS 的佈建 IOPS SSD (io1) 磁碟區。

```
aws ec2 create-volume \
    --volume-type io1 \
    --iops 1000 \
    --snapshot-id snap-066877671789bd71b \
    --availability-zone us-east-1a
```

輸出：

```
{
    "AvailabilityZone": "us-east-1a",
    "Tags": [],
    "Encrypted": false,
    "VolumeType": "io1",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "creating",
    "Iops": 1000,
    "SnapshotId": "snap-066877671789bd71b",
    "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "Size": 500
}
```

範例 3：建立加密磁碟區

下列create-volume範例會使用 EBS 加密的預設 CMK 建立加密磁碟區。如果預設為停用加密，您必須依照下列方式指定--encrypted參數。

```
aws ec2 create-volume \
    --size 80 \
    --encrypted \
```

```
    --availability-zone us-east-1a
```

輸出：

```
{
    "AvailabilityZone": "us-east-1a",
    "Tags": [],
    "Encrypted": true,
    "VolumeType": "gp2",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "creating",
    "Iops": 240,
    "SnapshotId": "",
    "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "Size": 80
}
```

如果預設為啟用加密，下列範例指令會建立加密的磁碟區，即使沒有--encrypted參數也是如此。

```
aws ec2 create-volume \
    --size 80 \
    --availability-zone us-east-1a
```

如果您使用--kms-key-id參數來指定客戶管理的 CMK，即使預設已啟用加密，也必須指定--encrypted參數。

```
aws ec2 create-volume \
    --volume-type gp2 \
    --size 80 \
    --encrypted \
    --kms-key-id 0ea3fef3-80a7-4778-9d8c-1c0c6EXAMPLE \
    --availability-zone us-east-1a
```

範例 4：建立含有標籤的磁碟區

下列create-volume範例會建立磁碟區並新增兩個標籤。

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
```

```
    --size 80 \
    --tag-specifications
 'ResourceType=volume,Tags=[{Key=purpose,Value=production},{Key=cost-
center,Value=cc123}]'
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateVolume中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會建立指定的磁碟區。

```
New-EC2Volume -Size 50 -AvailabilityZone us-west-2a -VolumeType gp2
```

輸出：

```
Attachments      : {}
AvailabilityZone : us-west-2a
CreateTime       : 12/22/2015 1:42:07 AM
Encrypted        : False
Iops             : 150
KmsKeyId         :
Size             : 50
SnapshotId       :
State            : creating
Tags             : {}
VolumeId         : vol-12345678
VolumeType       : gp2
```

範例 2：此範例要求會建立磁碟區，並套用含有堆疊金鑰和生產值的標籤。

```
$tag = @{ Key="stack"; Value="production" }

$tagspec = new-object Amazon.EC2.Model.TagSpecification
$tagspec.ResourceType = "volume"
$tagspec.Tags.Add($tag)

New-EC2Volume -Size 80 -AvailabilityZone "us-west-2a" -TagSpecification $tagspec
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateVolume式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateVpc配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateVpc。

CLI

AWS CLI

範例 1：建立 VPC

以下 create-vpc 範例會建立具有指定 IPv4 CIDR 區塊和名稱標籤的 VPC。

```
aws ec2 create-vpc \
    --cidr-block 10.0.0.0/16 \
    --tag-specifications ResourceType=vpc,Tags=[{Key=Name,Value=MyVpc}]
```

輸出：

```
{
    "Vpc": {
        "CidrBlock": "10.0.0.0/16",
        "DhcpOptionsId": "dopt-5EXAMPLE",
        "State": "pending",
        "VpcId": "vpc-0a60eb65b4EXAMPLE",
        "OwnerId": "123456789012",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-07501b79ecEXAMPLE",
                "CidrBlock": "10.0.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false,
        "Tags": [
            {
                "Key": "Name",
```

```
                "Value": MyVpc"
            }
        ]
    }
}
```

範例 2：建立具有專用租用的 VPC

以下 create-vpc 範例會建立具有指定 IPv4 CIDR 區塊和專用租用的 VPC。

```
aws ec2 create-vpc \
    --cidr-block 10.0.0.0/16 \
    --instance-tenancy dedicated
```

輸出：

```
{
    "Vpc": {
        "CidrBlock": "10.0.0.0/16",
        "DhcpOptionsId": "dopt-19edf471",
        "State": "pending",
        "VpcId": "vpc-0a53287fa4EXAMPLE",
        "OwnerId": "111122223333",
        "InstanceTenancy": "dedicated",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-00b24cc1c2EXAMPLE",
                "CidrBlock": "10.0.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false
    }
}
```

範例 3：建立具有 IPv6 CIDR 區塊的 VPC

以下 create-vpc 範例會建立具有 Amazon 提供之 IPv6 CIDR 區塊的 VPC。

```
aws ec2 create-vpc \
```

```
    --cidr-block 10.0.0.0/16 \
    --amazon-provided-ipv6-cidr-block
```

輸出：

```
{
    "Vpc": {
        "CidrBlock": "10.0.0.0/16",
        "DhcpOptionsId": "dopt-dEXAMPLE",
        "State": "pending",
        "VpcId": "vpc-0fc5e3406bEXAMPLE",
        "OwnerId": "123456789012",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-068432c60bEXAMPLE",
                "Ipv6CidrBlock": "",
                "Ipv6CidrBlockState": {
                    "State": "associating"
                },
                "Ipv6Pool": "Amazon",
                "NetworkBorderGroup": "us-west-2"
            }
        ],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-0669f8f9f5EXAMPLE",
                "CidrBlock": "10.0.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false
    }
}
```

範例 4：使用來自 IPAM 集區的 CIDR 建立 VPC

以下 `create-vpc` 範例會使用來自 Amazon VPC IP 位址管理器 (IPAM) 集區的 CIDR 建立 VPC。

Linux 和 macOS：

```
aws ec2 create-vpc \
    --ipv4-ipam-pool-id ipam-pool-0533048da7d823723 \
    --tag-specifications
 ResourceType=vpc,Tags='[{Key=Environment,Value="Preprod"},
{Key=Owner,Value="Build Team"}]'
```

Windows：

```
aws ec2 create-vpc ^
    --ipv4-ipam-pool-id ipam-pool-0533048da7d823723 ^
    --tag-specifications
 ResourceType=vpc,Tags=[{Key=Environment,Value="Preprod"},{Key=Owner,Value="Build
 Team"}]
```

輸出：

```
{
    "Vpc": {
        "CidrBlock": "10.0.1.0/24",
        "DhcpOptionsId": "dopt-2afccf50",
        "State": "pending",
        "VpcId": "vpc-010e1791024eb0af9",
        "OwnerId": "123456789012",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-0a77de1d803226d4b",
                "CidrBlock": "10.0.1.0/24",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false,
        "Tags": [
            {
                "Key": "Environment",
                "Value": "Preprod"
            },
            {
                "Key": "Owner",
```

```
                "Value": "Build Team"
            }
        ]
    }
}
```

如需詳細資訊，請參閱《Amazon VPC IPAM 使用者指南》中的建立使用 IPAM 集區 CIDR的 VPC。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateVpc中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會使用指定的 CIDR 建立 VPC。Amazon VPC 也會為虛 VPC 私人雲端建立下列項目：預設 DHCP 選項集、主路由表和預設網路 ACL。

```
New-EC2VPC -CidrBlock 10.0.0.0/16
```

輸出：

```
CidrBlock       : 10.0.0.0/16
DhcpOptionsId   : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault       : False
State           : pending
Tags            : {}
VpcId           : vpc-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateVpc式參考中的。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
require "aws-sdk-ec2"

# Creates a virtual private cloud (VPC) in
# Amazon Virtual Private Cloud (Amazon VPC) and then tags
# the VPC.
#
# @param ec2_resource [Aws::EC2::Resource] An initialized
#   Amazon Elastic Compute Cloud (Amazon EC2) resource object.
# @param cidr_block [String] The IPv4 CIDR block for the subnet.
# @param tag_key [String] The key portion of the tag for the VPC.
# @param tag_value [String] The value portion of the tag for the VPC.
# @return [Boolean] true if the VPC was created and tagged;
#   otherwise, false.
# @example
#   exit 1 unless vpc_created_and_tagged?(
#     Aws::EC2::Resource.new(region: 'us-west-2'),
#     '10.0.0.0/24',
#     'my-key',
#     'my-value'
#   )
def vpc_created_and_tagged?(
  ec2_resource,
  cidr_block,
  tag_key,
  tag_value
)
  vpc = ec2_resource.create_vpc(cidr_block: cidr_block)

  # Create a public DNS by enabling DNS support and DNS hostnames.
  vpc.modify_attribute(enable_dns_support: { value: true })
  vpc.modify_attribute(enable_dns_hostnames: { value: true })

  vpc.create_tags(tags: [{ key: tag_key, value: tag_value }])

  puts "Created VPC with ID '#{vpc.id}' and tagged with key " \
    "'#{tag_key}' and value '#{tag_value}'."
  return true
rescue StandardError => e
  puts "#{e.message}"
  return false
end
```

```
# Example usage:
def run_me
  cidr_block = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-create-vpc.rb " \
      "CIDR_BLOCK TAG_KEY TAG_VALUE REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-create-vpc.rb " \
      "10.0.0.0/24 my-key my-value us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    cidr_block = "10.0.0.0/24"
    tag_key = "my-key"
    tag_value = "my-value"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    cidr_block = ARGV[0]
    tag_key = ARGV[1]
    tag_value = ARGV[2]
    region = ARGV[3]
  end

  ec2_resource = Aws::EC2::Resource.new(region: region)

  if vpc_created_and_tagged?(
    ec2_resource,
    cidr_block,
    tag_key,
    tag_value
  )
    puts "VPC created and tagged."
  else
    puts "VPC not created or not tagged."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考CreateVpc中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateVpcEndpoint配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateVpcEndpoint。

CLI

AWS CLI

範例 1：建立閘道端點

下列create-vpc-endpoint範例會在該us-east-1區域的 VPC vpc-1a2b3c4d 和 Amazon S3 之間建立閘道 VPC 端點，並將路由表rtb-11aa22bb與端點建立關聯。

```
aws ec2 create-vpc-endpoint \
    --vpc-id vpc-1a2b3c4d \
    --service-name com.amazonaws.us-east-1.s3 \
    --route-table-ids rtb-11aa22bb
```

輸出：

```
{
    "VpcEndpoint": {
        "PolicyDocument": "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Sid\":
\"\",\"Effect\":\"Allow\",\"Principal\":\"\*\",\"Action\":\"\*\",\"Resource\":
\"\*\"}]}",
        "VpcId": "vpc-1a2b3c4d",
        "State": "available",
        "ServiceName": "com.amazonaws.us-east-1.s3",
        "RouteTableIds": [
            "rtb-11aa22bb"
        ],
        "VpcEndpointId": "vpc-1a2b3c4d",
        "CreationTimestamp": "2015-05-15T09:40:50Z"
    }
```

```
}
```

如需詳細資訊，請參閱AWS PrivateLink 指南中的建立閘道端點。

範例 2：建立介面端點

下列create-vpc-endpoint範例會在該地區的 VPC vpc-1a2b3c4d 和 Amazon S3 之間建立一個介面虛擬私人雲端端點。us-east-1此命令會在子網路中建立端點subnet-1a2b3c4d，將其與安全性群組建立關聯sg-1a2b3c4d，並新增含有索引鍵為「Service」且值為「S3」的標籤。

```
aws ec2 create-vpc-endpoint \
    --vpc-id vpc-1a2b3c4d \
    --vpc-endpoint-type Interface \
    --service-name com.amazonaws.us-east-1.s3 \
    --subnet-ids subnet-7b16de0c \
    --security-group-id sg-1a2b3c4d \
    --tag-specifications ResourceType=vpc-endpoint,Tags=[{Key=service,Value=S3}]
```

輸出：

```
{
    "VpcEndpoint": {
        "VpcEndpointId": "vpce-1a2b3c4d5e6f1a2b3",
        "VpcEndpointType": "Interface",
        "VpcId": "vpc-1a2b3c4d",
        "ServiceName": "com.amazonaws.us-east-1.s3",
        "State": "pending",
        "RouteTableIds": [],
        "SubnetIds": [
            "subnet-1a2b3c4d"
        ],
        "Groups": [
            {
                "GroupId": "sg-1a2b3c4d",
                "GroupName": "default"
            }
        ],
        "PrivateDnsEnabled": false,
        "RequesterManaged": false,
        "NetworkInterfaceIds": [
            "eni-0b16f0581c8ac6877"
```

```
        ],
        "DnsEntries": [
            {
                "DnsName": "*.vpce-1a2b3c4d5e6f1a2b3-9hnenorg.s3.us-
east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "*.vpce-1a2b3c4d5e6f1a2b3-9hnenorg-us-east-1c.s3.us-
east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            }
        ],
        "CreationTimestamp": "2021-03-05T14:46:16.030000+00:00",
        "Tags": [
            {
                "Key": "service",
                "Value": "S3"
            }
        ],
        "OwnerId": "123456789012"
    }
}
```

如需詳細資訊，請參閱《使用指南》中的〈建立介面端點〉 AWS PrivateLink。

範例 3：建立閘道 Load Balancer 端點

下列create-vpc-endpoint範例會在 VPC vpc-111122223333aabbc 和使用閘道 Load Balancer 設定的服務之間建立閘道 Load Balancer 端點。

```
aws ec2 create-vpc-endpoint \
    --service-name com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123 \
    --vpc-endpoint-type GatewayLoadBalancer \
    --vpc-id vpc-111122223333aabbc \
    --subnet-ids subnet-0011aabbcc2233445
```

輸出：

```
{
    "VpcEndpoint": {
        "VpcEndpointId": "vpce-aabbaabbaabbaabba",
```

```
        "VpcEndpointType": "GatewayLoadBalancer",
        "VpcId": "vpc-111122223333aabbc",
        "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123",
        "State": "pending",
        "SubnetIds": [
            "subnet-0011aabbcc2233445"
        ],
        "RequesterManaged": false,
        "NetworkInterfaceIds": [
            "eni-01010120203030405"
        ],
        "CreationTimestamp": "2020-11-11T08:06:03.522Z",
        "OwnerId": "123456789012"
    }
}
```

如需詳細資訊，請參閱《使用指南》中的「閘道 Load Balancer」端點 AWS PrivateLink。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateVpcEndpoint中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例為 VPC vpc -0fc1ff23f45b678eb 中的服務建立一個新的虛擬私人雲端端點

```
New-EC2VpcEndpoint -ServiceName com.amazonaws.eu-west-1.s3 -VpcId
 vpc-0fc1ff23f45b678eb
```

輸出：

```
ClientToken VpcEndpoint
----------- -----------
            Amazon.EC2.Model.VpcEndpoint
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateVpcEndpoint式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateVpnConnection配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateVpnConnection。

CLI

AWS CLI

範例 1：使用動態路由建立 VPN 連線

下列create-vpn-connection範例會在指定的虛擬私有閘道與指定的客戶閘道之間建立 VPN 連線，並將標籤套用至 VPN 連線。輸出包括客戶閘道裝置的組態資訊 (XML 格式)。

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --customer-gateway-id cgw-001122334455aabbc \
    --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
    --tag-specification 'ResourceType=vpn-connection,Tags=[{Key=Name,Value=BGP-
VPN}]'
```

輸出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "...configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-123123123123abcab",
        "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": false,
            "LocalIpv4NetworkCidr": "0.0.0.0/0",
            "RemoteIpv4NetworkCidr": "0.0.0.0/0",
            "TunnelInsideIpVersion": "ipv4",
            "TunnelOptions": [
                {},
                {}
            ]
        },
        "Routes": [],
```

```
        "Tags": [
             {
                "Key": "Name",
                "Value": "BGP-VPN"
            }
        ]
    }
}
```

如需詳細資訊，請參閱 AWS 網站間 VPN 使用者指南中的 AWS Site-to-Site VPN 運作方式。

範例 2：使用靜態路由建立 VPN 連線

下列create-vpn-connection範例會在指定的虛擬私有閘道與指定的客戶閘道之間建立 VPN 連線。這些選項指定靜態路由。輸出包括客戶閘道裝置的組態資訊 (XML 格式)。

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --customer-gateway-id cgw-001122334455aabbc \
    --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
    --options "{\"StaticRoutesOnly\":true}"
```

輸出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "..configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-123123123123abcab",
        "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": true,
            "LocalIpv4NetworkCidr": "0.0.0.0/0",
            "RemoteIpv4NetworkCidr": "0.0.0.0/0",
            "TunnelInsideIpVersion": "ipv4",
            "TunnelOptions": [
                {},
                {}
            ]
        },
```

```
        "Routes": [],
        "Tags": []
    }
}
```

如需詳細資訊，請參閱 AWS 網站間 VPN 使用者指南中的 AWS Site-to-Site VPN 運作方式。

範例 3：建立 VPN 連線，並在 CIDR 和預先共用金鑰中指定您自己的連線

下列create-vpn-connection範例會建立 VPN 連線，並為每個通道指定內部 IP 位址 CIDR 區塊和自訂預先共用金鑰。指定的值會在CustomerGatewayConfiguration資訊中傳回。

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --customer-gateway-id cgw-001122334455aabbc \
    --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
    --options
 TunnelOptions='[{TunnelInsideCidr=169.254.12.0/30,PreSharedKey=ExamplePreSharedKey1},
{TunnelInsideCidr=169.254.13.0/30,PreSharedKey=ExamplePreSharedKey2}]'
```

輸出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "..configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-123123123123abcab",
        "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": false,
            "LocalIpv4NetworkCidr": "0.0.0.0/0",
            "RemoteIpv4NetworkCidr": "0.0.0.0/0",
            "TunnelInsideIpVersion": "ipv4",
            "TunnelOptions": [
                {
                    "OutsideIpAddress": "203.0.113.3",
                    "TunnelInsideCidr": "169.254.12.0/30",
                    "PreSharedKey": "ExamplePreSharedKey1"
                },
                {
```

```
                    "OutsideIpAddress": "203.0.113.5",
                    "TunnelInsideCidr": "169.254.13.0/30",
                    "PreSharedKey": "ExamplePreSharedKey2"
                }
            ]
        },
        "Routes": [],
        "Tags": []
    }
}
```

如需詳細資訊，請參閱 AWS 網站間 VPN 使用者指南中的 AWS Site-to-Site VPN 運作方式。

範例 4：建立支援 IPv6 流量的 VPN 連線

下列create-vpn-connection範例會建立 VPN 連線，以支援指定傳輸閘道與指定客戶閘道之間的 IPv6 流量。兩個通道的通道選項都指定 AWS 必須起始 IKE 交涉。

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --transit-gateway-id tgw-12312312312312312 \
    --customer-gateway-id cgw-001122334455aabbc \
    --options TunnelInsideIpVersion=ipv6,TunnelOptions=[{StartupAction=start},
{StartupAction=start}]
```

輸出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "..configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-11111111122222222",
        "TransitGatewayId": "tgw-12312312312312312",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": false,
            "LocalIpv6NetworkCidr": "::/0",
            "RemoteIpv6NetworkCidr": "::/0",
            "TunnelInsideIpVersion": "ipv6",
            "TunnelOptions": [
                {
```

```
                    "OutsideIpAddress": "203.0.113.3",
                    "StartupAction": "start"
                },
                {
                    "OutsideIpAddress": "203.0.113.5",
                    "StartupAction": "start"
                }
            ]
        },
        "Routes": [],
        "Tags": []
    }
}
```

如需詳細資訊，請參閱 AWS 網站間 VPN 使用者指南中的 AWS Site-to-Site VPN 運作方式。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateVpnConnection中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會在指定的虛擬私有閘道與指定的客戶閘道之間建立 VPN 連線。輸出包含網路管理員所需的組態資訊 (XML 格式)。

```
New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
 vgw-1a2b3c4d
```

輸出：

```
CustomerGatewayConfiguration : [XML document]
CustomerGatewayId            : cgw-1a2b3c4d
Options                      :
Routes                       : {}
State                        : pending
Tags                         : {}
Type                         :
VgwTelemetry                 : {}
VpnConnectionId              : vpn-12345678
VpnGatewayId                 : vgw-1a2b3c4d
```

範例 2：此範例會建立 VPN 連線，並擷取具有指定名稱的檔案中的組態。

```
(New-EC2VpnConnection -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
 vgw-1a2b3c4d).CustomerGatewayConfiguration | Out-File C:\path\vpn-
configuration.xml
```

範例 3：此範例會在指定的虛擬私有閘道與指定的客戶閘道之間建立具有靜態路由的 VPN 連線。

```
New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
 vgw-1a2b3c4d -Options_StaticRoutesOnly $true
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateVpnConnection式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateVpnConnectionRoute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateVpnConnectionRoute。

CLI

AWS CLI

建立 VPN 連線的靜態路由

此範例會為指定的 VPN 連線建立靜態路由。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 create-vpn-connection-route --vpn-connection-id vpn-40f41529 --
destination-cidr-block 11.12.0.0/16
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateVpnConnectionRoute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會為指定的 VPN 連線建立指定的靜態路由。

```
New-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock
 11.12.0.0/16
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateVpnConnectionRoute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateVpnGateway配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateVpnGateway。

CLI

AWS CLI

建立虛擬私有閘道

此範例會建立虛擬私有閘道。

命令：

```
aws ec2 create-vpn-gateway --type ipsec.1
```

輸出：

```
{
    "VpnGateway": {
        "AmazonSideAsn": 64512,
        "State": "available",
        "Type": "ipsec.1",
        "VpnGatewayId": "vgw-9a4cacf3",
        "VpcAttachments": []
    }
}
```

使用特定亞馬遜端 ASN 建立虛擬私有閘道

此範例會建立虛擬私有閘道，並為 BGP 工作階段的 Amazon 端指定自主系統編號 (ASN)。

命令：

```
aws ec2 create-vpn-gateway --type ipsec.1 --amazon-side-asn 65001
```

輸出：

```
{
    "VpnGateway": {
        "AmazonSideAsn": 65001,
        "State": "available",
        "Type": "ipsec.1",
        "VpnGatewayId": "vgw-9a4cacf3",
        "VpcAttachments": []
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateVpnGateway中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會建立指定的虛擬私有閘道。

```
New-EC2VpnGateway -Type ipsec.1
```

輸出：

```
AvailabilityZone :
State            : available
Tags             : {}
Type             : ipsec.1
VpcAttachments   : {}
VpnGatewayId     : vgw-1a2b3c4d
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateVpnGateway式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteCustomerGateway配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteCustomerGateway。

CLI

AWS CLI

若要刪除客戶閘道

此範例會刪除指定的客戶閘道。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-customer-gateway --customer-gateway-id cgw-0e11f167
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteCustomerGateway中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會刪除指定的客戶閘道。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2CustomerGateway (DeleteCustomerGateway)" on
 Target "cgw-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteCustomerGateway式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteDhcpOptions配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteDhcpOptions。

CLI

AWS CLI

若要刪除 DHCP 選項集

此範例會刪除指定的 DHCP 選項集。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-dhcp-options --dhcp-options-id dopt-d9070ebb
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteDhcpOptions中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會刪除指定的 DHCP 選項集。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2DhcpOption (DeleteDhcpOptions)" on Target
 "dopt-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteDhcpOptions式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteFlowLogs配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteFlowLogs。

CLI

AWS CLI

若要刪除流程記錄

下列delete-flow-logs範例會刪除指定的流程記錄檔。

```
aws ec2 delete-flow-logs --flow-log-id fl-11223344556677889
```

輸出：

```
{
    "Unsuccessful": []
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteFlowLogs中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會移除指定的 FlowLogId FL-01a2b3456a789c01

```
Remove-EC2FlowLog -FlowLogId fl-01a2b3456a789c01
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2FlowLog (DeleteFlowLogs)" on target
 "fl-01a2b3456a789c01".
[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 指令程DeleteFlowLogs式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteInternetGateway配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteInternetGateway。

CLI

AWS CLI

若要刪除網際網路閘道

下列delete-internet-gateway範例會刪除指定的網際網路閘道。

```
aws ec2 delete-internet-gateway \
    --internet-gateway-id igw-0d0fb496b3EXAMPLE
```

此命令不會產生輸出。

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的網際網路閘道。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteInternetGateway中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會刪除指定的網際網路閘道。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2InternetGateway (DeleteInternetGateway)" on
 Target "igw-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteInternetGateway式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteKeyPair配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteKeyPair。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Delete an Amazon EC2 key pair.
    /// </summary>
    /// <param name="keyPairName">The name of the key pair to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteKeyPair(string keyPairName)
    {
        try
        {
            await _amazonEC2.DeleteKeyPairAsync(new
DeleteKeyPairRequest(keyPairName)).ConfigureAwait(false);
            return true;
        }
        catch (Exception ex)
        {
```

```
            Console.WriteLine($"Couldn't delete the key pair because:
 {ex.Message}");
            return false;
        }
    }

    /// <summary>
    /// Delete the temporary file where the key pair information was saved.
    /// </summary>
    /// <param name="tempFileName">The path to the temporary file.</param>
    public void DeleteTempFile(string tempFileName)
    {
        if (File.Exists(tempFileName))
        {
            File.Delete(tempFileName);
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DeleteKeyPair中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
###############################################################################
# function ec2_delete_keypair
#
# This function deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#
# And:
#       0 - If successful.
#       1 - If it fails.
```

```
################################################################################
function ec2_delete_keypair() {
  local key_pair_name response

  local option OPTARG # Required to use getopts command in a function.
  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_keypair"
    echo "Deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair."
    echo "  -n key_pair_name - A key pair name."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) key_pair_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -n parameter."
    usage
    return 1
  fi

  response=$(aws ec2 delete-key-pair \
    --key-name "$key_pair_name") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-key-pair operation failed.$response"
    return 1
  }

  return 0
```

```
}
```

此範例中使用的公用程式函數。

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
########################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
```

```
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteKeyPair中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   Aws::EC2::EC2Client ec2Client(clientConfiguration);
   Aws::EC2::Model::DeleteKeyPairRequest request;

   request.SetKeyName(keyPairName);
   const Aws::EC2::Model::DeleteKeyPairOutcome outcome =
ec2Client.DeleteKeyPair(
           request);

   if (!outcome.IsSuccess()) {
       std::cerr << "Failed to delete key pair " << keyPairName <<
                 ":" << outcome.GetError().GetMessage() << std::endl;
   }
   else {
       std::cout << "Successfully deleted key pair named " << keyPairName <<
                 std::endl;
   }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考DeleteKeyPair中的。

CLI

AWS CLI

刪除金鑰對

下列delete-key-pair範例會刪除指定的 key pair。

```
aws ec2 delete-key-pair \
    --key-name my-key-pair
```

輸出：

```
{
    "Return": true,
    "KeyPairId": "key-03c8d3aceb53b507"
}
```

若要取得更多資訊，請參閱《指AWS 令行介面使用指南》中的〈建立和刪除金鑰配對〉。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteKeyPair中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    public static void deleteKeys(Ec2Client ec2, String keyPair) {
        try {
            DeleteKeyPairRequest request = DeleteKeyPairRequest.builder()
                    .keyName(keyPair)
                    .build();

            ec2.deleteKeyPair(request);
            System.out.println("Successfully deleted key pair named " + keyPair);
```

```
        } catch (Ec2Exception 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DeleteKeyPair中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { DeleteKeyPairCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new DeleteKeyPairCommand({
    KeyName: "KEY_PAIR_NAME",
  });

  try {
    await client.send(command);
    console.log("Successfully deleted key pair.");
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DeleteKeyPair中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun deleteKeys(keyPair: String?) {
    val request =
        DeleteKeyPairRequest {
            keyName = keyPair
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.deleteKeyPair(request)
        println("Successfully deleted key pair named $keyPair")
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK DeleteKeyPair中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會刪除指定的 key pair。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2KeyPair -KeyName my-key-pair
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2KeyPair (DeleteKeyPair)" on Target "my-key-pair".
```

```
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteKeyPair式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class KeyPairWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair
 actions."""

    def __init__(self, ec2_resource, key_file_dir, key_pair=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param key_file_dir: The folder where the private key information is
 stored.
                             This should be a secure folder.
        :param key_pair: A Boto3 KeyPair object. This is a high-level object that
                         wraps key pair actions.
        """
        self.ec2_resource = ec2_resource
        self.key_pair = key_pair
        self.key_file_path = None
        self.key_file_dir = key_file_dir

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource, tempfile.TemporaryDirectory())
```

```
    def delete(self):
        """
        Deletes a key pair.
        """
        if self.key_pair is None:
            logger.info("No key pair to delete.")
            return

        key_name = self.key_pair.name
        try:
            self.key_pair.delete()
            self.key_pair = None
        except ClientError as err:
            logger.error(
                "Couldn't delete key %s. Here's why: %s : %s",
                key_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DeleteKeyPair中的 Python (博托 3) API 參考。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        lo_ec2->deletekeypair( iv_keyname = iv_key_name ).
        MESSAGE 'Amazon EC2 key pair deleted.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
```

```
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK DeleteKeyPair中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteLaunchTemplate配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteLaunchTemplate。

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：

- 建置及管理彈性服務

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
/// <summary>
/// Delete a launch template by name.
/// </summary>
/// <param name="templateName">The name of the template to delete.</param>
/// <returns>Async task.</returns>
public async Task DeleteTemplateByName(string templateName)
{
    try
    {
        await _amazonEc2.DeleteLaunchTemplateAsync(
            new DeleteLaunchTemplateRequest()
            {
                LaunchTemplateName = templateName
```

```
                });
        }
        catch (AmazonClientException)
        {
            Console.WriteLine($"Unable to delete template {templateName}.");
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DeleteLaunchTemplate中的。

CLI

AWS CLI

刪除啟動範本

此範例會刪除指定的啟動範本。

命令：

```
aws ec2 delete-launch-template --launch-template-id lt-0abcd290751193123
```

輸出：

```
{
  "LaunchTemplate": {
      "LatestVersionNumber": 2,
      "LaunchTemplateId": "lt-0abcd290751193123",
      "LaunchTemplateName": "TestTemplate",
      "DefaultVersionNumber": 2,
      "CreatedBy": "arn:aws:iam::123456789012:root",
      "CreateTime": "2017-11-23T16:46:25.000Z"
  }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteLaunchTemplate中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
  await client.send(
    new DeleteLaunchTemplateCommand({
      LaunchTemplateName: NAMES.launchTemplateName,
    }),
  );
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DeleteLaunchTemplate中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
```

```
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
```

```
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete launch template
{self.launch_template_name}: {err}."
                )
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DeleteLaunchTemplate中的 Python (博托 3) API 參考。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteNetworkAcl配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteNetworkAcl。

CLI

AWS CLI

刪除網路 ACL 的步驟

此範例會刪除指定的網路 ACL。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-network-acl --network-acl-id acl-5fb85d36
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteNetworkAcl中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會刪除指定的網路 ACL。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2NetworkAcl -NetworkAclId acl-12345678
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkAcl (DeleteNetworkAcl)" on Target
 "acl-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteNetworkAcl式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteNetworkAclEntry配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteNetworkAclEntry。

CLI

AWS CLI

刪除網路 ACL 項目

此範例會從指定的網路 ACL 刪除輸入規則編號 100。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-
number 100
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteNetworkAclEntry中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會從指定的網路 ACL 移除指定的規則。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber
 100
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkAclEntry (DeleteNetworkAclEntry)" on
 Target "acl-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteNetworkAclEntry式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteNetworkInterface配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteNetworkInterface。

CLI

AWS CLI

刪除網路介面

此範例會刪除指定的網路介面。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-network-interface --network-interface-id eni-e5aa89a3
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteNetworkInterface中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會刪除指定的網路介面。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkInterface (DeleteNetworkInterface)" on
 Target "eni-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteNetworkInterface式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeletePlacementGroup配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeletePlacementGroup。

CLI

AWS CLI

刪除放置群組

此範例指令會刪除指定的放置群組。

命令：

```
aws ec2 delete-placement-group --group-name my-cluster
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeletePlacementGroup中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會刪除指定的放置群組。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2PlacementGroup -GroupName my-placement-group
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2PlacementGroup (DeletePlacementGroup)" on Target
 "my-placement-group".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeletePlacementGroup式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteRoute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteRoute。

CLI

AWS CLI

刪除路線

這個例子從指定的路由表中刪除指定的路由。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-route --route-table-id rtb-22574640 --destination-cidr-block
 0.0.0.0/0
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteRoute中的。

PowerShell

適用的工具 PowerShell

實施例 1：這個例子從指定的路由表中刪除指定的路由。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Route (DeleteRoute)" on Target "rtb-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteRoute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteRouteTable配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteRouteTable。

CLI

AWS CLI

刪除路由表

這個例子刪除指定的路由表。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-route-table --route-table-id rtb-22574640
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteRouteTable中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會刪除指定的路由表格。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2RouteTable (DeleteRouteTable)" on Target
 "rtb-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteRouteTable式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteSecurityGroup配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteSecurityGroup。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Delete an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteSecurityGroup(string groupId)
    {
        var response = await _amazonEC2.DeleteSecurityGroupAsync(new
DeleteSecurityGroupRequest { GroupId = groupId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DeleteSecurityGroup中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
#############################################################################
# function ec2_delete_security_group
#
# This function deletes an Amazon Elastic Compute Cloud (Amazon EC2) security
 group.
```

```
#
# Parameters:
#       -i security_group_id - The ID of the security group to delete.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_delete_security_group() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_security_group"
    echo "Deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -i security_group_id - The ID of the security group to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) security_group_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -i parameter."
    usage
    return 1
  fi
```

```
  response=$(aws ec2 delete-security-group --group-id "$security_group_id" --
output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-security-group operation failed.$response"
    return 1
  }

  return 0
}
```

此範例中使用的公用程式函數。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
```

```
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteSecurityGroup中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DeleteSecurityGroupRequest request;

    request.SetGroupId(securityGroupID);
    auto outcome = ec2Client.DeleteSecurityGroup(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to delete security group " << securityGroupID <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted security group " << securityGroupID <<
                  std::endl;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考DeleteSecurityGroup中的。

CLI

AWS CLI

[EC2-Classic] 刪除安全群組

此範例會刪除名為 MySecurityGroup 的安全群組。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-security-group --group-name MySecurityGroup
```

[EC2-VPC] 刪除安全群組

此範例會刪除 ID 為 sg-903004f8 的安全群組。請注意，EC2-VPC 的安全群組不能按名稱引用。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-security-group --group-id sg-903004f8
```

如需詳細資訊，請參閱《AWS 命令行介面使用者指南》中的「使用安全群組」。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteSecurityGroup中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    public static void deleteEC2SecGroup(Ec2Client ec2, String groupId) {
        try {
            DeleteSecurityGroupRequest request =
DeleteSecurityGroupRequest.builder()
```

```
                    .groupId(groupId)
                    .build();

            ec2.deleteSecurityGroup(request);
            System.out.println("Successfully deleted security group with Id " +
 groupId);

        } catch (Ec2Exception 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DeleteSecurityGroup中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { DeleteSecurityGroupCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new DeleteSecurityGroupCommand({
    GroupId: "GROUP_ID",
  });

  try {
    await client.send(command);
    console.log("Security group deleted successfully.");
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DeleteSecurityGroup中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun deleteEC2SecGroup(groupIdVal: String) {
    val request =
        DeleteSecurityGroupRequest {
            groupId = groupIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.deleteSecurityGroup(request)
        println("Successfully deleted Security Group with id $groupIdVal")
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK DeleteSecurityGroup中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會刪除 EC2-VPC 的指定安全性群組。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2SecurityGroup -GroupId sg-12345678
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2SecurityGroup (DeleteSecurityGroup)" on Target
 "sg-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

範例 2：此範例會刪除 EC2-Classic 的指定安全性群組。

```
Remove-EC2SecurityGroup -GroupName my-security-group -Force
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteSecurityGroup式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class SecurityGroupWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) security group
 actions."""

    def __init__(self, ec2_resource, security_group=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param security_group: A Boto3 SecurityGroup object. This is a high-level
 object
                               that wraps security group actions.
        """
        self.ec2_resource = ec2_resource
        self.security_group = security_group
```

```
    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def delete(self):
        """
        Deletes the security group.
        """
        if self.security_group is None:
            logger.info("No security group to delete.")
            return

        group_id = self.security_group.id
        try:
            self.security_group.delete()
        except ClientError as err:
            logger.error(
                "Couldn't delete security group %s. Here's why: %s: %s",
                group_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DeleteSecurityGroup中的 Python (博托 3) API 參考。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        lo_ec2->deletesecuritygroup( iv_groupid = iv_security_group_id ).
        MESSAGE 'Security group deleted.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK DeleteSecurityGroup中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteSnapshot配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteSnapshot。

CLI

AWS CLI

刪除快照

此範例命令會刪除快照 ID 為 snap-1234567890abcdef0 的快照。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteSnapshot中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會刪除指定的快照。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2Snapshot -SnapshotId snap-12345678
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Snapshot (DeleteSnapshot)" on target
 "snap-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteSnapshot式參考中的。

Rust

適用於 Rust 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
async fn delete_snapshot(client: &Client, id: &str) -> Result<(), Error> {
    client.delete_snapshot().snapshot_id(id).send().await?;

    println!("Deleted");

    Ok(())
}
```

- 如需 API 的詳細資訊，請參閱 AWS SDK DeleteSnapshot中的 Rust API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteSpotDatafeedSubscription配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteSpotDatafeedSubscription。

CLI

AWS CLI

取消競價型執行個體資料饋送訂閱

此範例命令會刪除該帳戶的 Spot 資料饋送訂閱。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-spot-datafeed-subscription
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteSpotDatafeedSubscription中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會刪除您的 Spot 執行個體資料饋送。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2SpotDatafeedSubscription
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2SpotDatafeedSubscription
 (DeleteSpotDatafeedSubscription)" on Target "".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteSpotDatafeedSubscription式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteSubnet配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteSubnet。

CLI

AWS CLI

若要刪除子網路

此範例會刪除指定的子網路。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-subnet --subnet-id subnet-9d4a7b6c
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteSubnet中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會刪除指定的子網路。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2Subnet -SubnetId subnet-1a2b3c4d
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Subnet (DeleteSubnet)" on Target
 "subnet-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteSubnet式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**DeleteTags**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteTags。

CLI

AWS CLI

範例 1：若要從資源刪除標籤

下列delete-tags範例會刪除指定影像Stack=Test中的標籤。當您同時指定值和索引鍵名稱時，只有在標籤的值與指定值相符時，才會刪除標籤。

```
aws ec2 delete-tags \
    --resources ami-1234567890abcdef0 \
    --tags Key=Stack,Value=Test
```

指定標籤的值是可選的。下列delete-tags範例會purpose從指定的執行個體刪除含有索引鍵名稱的標籤，而不論標籤的標籤值為何。

```
aws ec2 delete-tags \
    --resources i-1234567890abcdef0 \
    --tags Key=purpose
```

如果您將空字串指定為標籤值，則只有當標籤的值為空字串時，才會刪除標籤。下列delete-tags範例會指定空字串做為要刪除之標籤的標籤值。

```
aws ec2 delete-tags \
    --resources i-1234567890abcdef0 \
    --tags Key=Name,Value=
```

範例 2：若要從多個資源刪除標籤

下列delete-tags範例會從執行個體和 AMI 中刪除標籤 `目的 =test``。如前面的範例所示，您可以省略指令中的標籤值。

```
aws ec2 delete-tags \
    --resources i-1234567890abcdef0 ami-1234567890abcdef0 \
    --tags Key=Purpose
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteTags中的。

PowerShell

用於的工具 PowerShell

範例 1：無論標籤值為何，本範例都會從指定的資源刪除指定的標籤。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag" } -Force
```

範例 2：此範例會從指定的資源刪除指定的標籤，但前提是在標籤值相符時才會刪除。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag";Value="myTagValue" } -
Force
```

範例 3：此範例會從指定的資源刪除指定的標籤，而不論標籤值為何。

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

範例 4：此範例會從指定的資源刪除指定的標籤，但前提是在標籤值相符時才會刪除。

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"
$tag.Value = "myTagValue"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteTags式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteVolume配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteVolume。

CLI

AWS CLI

刪除磁碟區

此範例指令會刪除磁碟區 ID 為的可用磁碟區vol-049df61146c4d7901。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-volume --volume-id vol-049df61146c4d7901
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteVolume中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例分離指定的磁碟區。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2Volume -VolumeId vol-12345678
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Volume (DeleteVolume)" on target
 "vol-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteVolume式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteVpc配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteVpc。

CLI

AWS CLI

若要刪除虛擬私人雲端

此範例會刪除指定的 VPC。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-vpc --vpc-id vpc-a01106c2
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteVpc中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會刪除指定的 VPC。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2Vpc -VpcId vpc-12345678
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Vpc (DeleteVpc)" on Target "vpc-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteVpc式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteVpnConnection配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteVpnConnection。

CLI

AWS CLI

若要刪除 VPN 連線

此範例會刪除指定的 VPN 連線。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-vpn-connection --vpn-connection-id vpn-40f41529
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteVpnConnection中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會刪除指定的 VPN 連線。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2VpnConnection -VpnConnectionId vpn-12345678
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnConnection (DeleteVpnConnection)" on Target
 "vpn-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteVpnConnection式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteVpnConnectionRoute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteVpnConnectionRoute。

CLI

AWS CLI

從 VPN 連線刪除靜態路由

此範例會從指定的 VPN 連線刪除指定的靜態路由。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-vpn-connection-route --vpn-connection-id vpn-40f41529 --
destination-cidr-block 11.12.0.0/16
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteVpnConnectionRoute中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會從指定的 VPN 連線移除指定的靜態路由。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock
 11.12.0.0/16
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnConnectionRoute (DeleteVpnConnectionRoute)" on
 Target "vpn-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteVpnConnectionRoute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteVpnGateway配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteVpnGateway。

CLI

AWS CLI

刪除虛擬私有閘道

此範例會刪除指定的虛擬私有閘道。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 delete-vpn-gateway --vpn-gateway-id vgw-9a4cacf3
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteVpnGateway中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會刪除指定的虛擬私有閘道。除非您同時指定了 Force 參數，否則系統會在作業繼續之前提示您確認。

```
Remove-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnGateway (DeleteVpnGateway)" on Target
 "vgw-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeleteVpnGateway式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeregisterImage配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeregisterImage。

CLI

AWS CLI

若要取消註冊 AMI

此範例會取消註冊指定的 AMI。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 deregister-image --image-id ami-4fa54026
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeregisterImage中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會取消註冊指定 AMI。

```
Unregister-EC2Image -ImageId ami-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DeregisterImage式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeAccountAttributes配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeAccountAttributes。

CLI

AWS CLI

描述您 AWS 帳戶的所有屬性

此範例說明您 AWS 帳戶的屬性。

命令：

```
aws ec2 describe-account-attributes
```

輸出：

```
{
    "AccountAttributes": [
        {
            "AttributeName": "vpc-max-security-groups-per-interface",
            "AttributeValues": [
                {
                    "AttributeValue": "5"
                }
            ]
        },
        {
            "AttributeName": "max-instances",
            "AttributeValues": [
                {
                    "AttributeValue": "20"
                }
            ]
        },
        {
            "AttributeName": "supported-platforms",
            "AttributeValues": [
                {
                    "AttributeValue": "EC2"
                },
                {
                    "AttributeValue": "VPC"
                }
            ]
        },
        {
            "AttributeName": "default-vpc",
            "AttributeValues": [
                {
                    "AttributeValue": "none"
                }
```

```
            ]
        },
        {
            "AttributeName": "max-elastic-ips",
            "AttributeValues": [
                {
                    "AttributeValue": "5"
                }
            ]
        },
        {
            "AttributeName": "vpc-max-elastic-ips",
            "AttributeValues": [
                {
                    "AttributeValue": "5"
                }
            ]
        }
    ]
}
```

描述 AWS 帳戶的單一屬性

此範例說明您 AWS 帳戶的supported-platforms屬性。

命令：

```
aws ec2 describe-account-attributes --attribute-names supported-platforms
```

輸出：

```
{
    "AccountAttributes": [
        {
            "AttributeName": "supported-platforms",
            "AttributeValues": [
                {
                    "AttributeValue": "EC2"
                },
                {
                    "AttributeValue": "VPC"
                }
```

```
            ]
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeAccountAttributes](#)中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明您是否可以將例證啟動到該區域中的 EC2-典型和 EC2-VPC 中，或只啟動至 EC2-VPC。

```
(Get-EC2AccountAttribute -AttributeName supported-platforms).AttributeValues
```

輸出：

```
AttributeValue
--------------
EC2
VPC
```

範例 2：此範例說明您的預設 VPC，如果您的區域中沒有預設 VPC，則為「none」。

```
(Get-EC2AccountAttribute -AttributeName default-vpc).AttributeValues
```

輸出：

```
AttributeValue
--------------
vpc-12345678
```

範例 3：此範例說明您可以執行的隨需執行個體數目上限。

```
(Get-EC2AccountAttribute -AttributeName max-instances).AttributeValues
```

輸出：

```
AttributeValue
--------------
20
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeAccountAttributes式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeAddresses配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeAddresses。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DescribeAddressesRequest request;
    auto outcome = ec2Client.DescribeAddresses(request);
    if (outcome.IsSuccess()) {
        std::cout << std::left << std::setw(20) << "InstanceId" <<
                  std::setw(15) << "Public IP" << std::setw(10) << "Domain" <<
                  std::setw(30) << "Allocation ID" << std::setw(25) <<
                  "NIC ID" << std::endl;

        const auto &addresses = outcome.GetResult().GetAddresses();
        for (const auto &address: addresses) {
            Aws::String domainString =
                    Aws::EC2::Model::DomainTypeMapper::GetNameForDomainType(
                            address.GetDomain());

            std::cout << std::left << std::setw(20) <<
```

```
                    address.GetInstanceId() << std::setw(15) <<
                    address.GetPublicIp() << std::setw(10) << domainString <<
                    std::setw(30) << address.GetAllocationId() << std::setw(25)
                    << address.GetNetworkInterfaceId() << std::endl;
        }
    }
    else {
        std::cerr << "Failed to describe Elastic IP addresses:" <<
                  outcome.GetError().GetMessage() << std::endl;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考DescribeAddresses中的。

CLI

AWS CLI

範例 1：擷取有關您所有彈性 IP 地址的詳細資訊

以下 `describe addresses` 範例顯示有關您彈性 IP 地址的詳細資訊。

```
aws ec2 describe-addresses
```

輸出：

```
{
    "Addresses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PublicIp": "198.51.100.0",
            "PublicIpv4Pool": "amazon",
            "Domain": "standard"
        },
        {
            "Domain": "vpc",
            "PublicIpv4Pool": "amazon",
            "InstanceId": "i-1234567890abcdef0",
            "NetworkInterfaceId": "eni-12345678",
            "AssociationId": "eipassoc-12345678",
            "NetworkInterfaceOwnerId": "123456789012",
            "PublicIp": "203.0.113.0",
```

```
            "AllocationId": "eipalloc-12345678",
            "PrivateIpAddress": "10.0.1.241"
        }
    ]
}
```

範例 2：擷取有關 EC2-VPC 適用之彈性 IP 地址的詳細資訊

下列 describe-addresses 範例會顯示彈性 IP 地址的詳細資訊，以便搭配 VPC 中的執行個體使用。

```
aws ec2 describe-addresses \
    --filters "Name=domain,Values=vpc"
```

輸出：

```
{
    "Addresses": [
        {
            "Domain": "vpc",
            "PublicIpv4Pool": "amazon",
            "InstanceId": "i-1234567890abcdef0",
            "NetworkInterfaceId": "eni-12345678",
            "AssociationId": "eipassoc-12345678",
            "NetworkInterfaceOwnerId": "123456789012",
            "PublicIp": "203.0.113.0",
            "AllocationId": "eipalloc-12345678",
            "PrivateIpAddress": "10.0.1.241"
        }
    ]
}
```

範例 3：擷取有關透過配置 ID 所指定的彈性 IP 地址的詳細資訊

下列 describe-addresses 範例顯示具有指定配置 ID (已與 EC2-VPC 中的執行個體建立關聯) 的彈性 IP 地址的詳細資訊。

```
aws ec2 describe-addresses \
    --allocation-ids eipalloc-282d9641
```

輸出：

```
{
    "Addresses": [
        {
            "Domain": "vpc",
            "PublicIpv4Pool": "amazon",
            "InstanceId": "i-1234567890abcdef0",
            "NetworkInterfaceId": "eni-1a2b3c4d",
            "AssociationId": "eipassoc-123abc12",
            "NetworkInterfaceOwnerId": "1234567891012",
            "PublicIp": "203.0.113.25",
            "AllocationId": "eipalloc-282d9641",
            "PrivateIpAddress": "10.251.50.12"
        }
    ]
}
```

範例 4：擷取有關透過其 VPC 私有 IP 地址所指定的彈性 IP 地址的詳細資訊

下列 describe-addresses 範例針對已與 EC2-VPC 中特定私有 IP 位址建立關聯的彈性 IP 地址，顯示詳細資訊。

```
aws ec2 describe-addresses \
    --filters "Name=private-ip-address,Values=10.251.50.12"
```

範例 5：擷取有關 EC2-Classic 中彈性 IP 地址的詳細資訊

下列 describe-addresses 範例會顯示彈性 IP 地址的詳細資訊，以便用於 EC2-Classic。

```
aws ec2 describe-addresses \
    --filters "Name=domain,Values=standard"
```

輸出：

```
{
    "Addresses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PublicIp": "203.0.110.25",
            "PublicIpv4Pool": "amazon",
            "Domain": "standard"
        }
```

```
    ]
}
```

範例 6：擷取有關透過其公有 IP 地址所指定的彈性 IP 地址的詳細資訊

下列 `describe-addresses` 範例顯示具有值 `203.0.110.25` (已與 EC2-Classic 中的執行個體建立關聯) 的彈性 IP 地址的詳細資訊。

```
aws ec2 describe-addresses \
    --public-ips 203.0.110.25
```

輸出：

```
{
    "Addresses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PublicIp": "203.0.110.25",
            "PublicIpv4Pool": "amazon",
            "Domain": "standard"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeAddresses中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { DescribeAddressesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";
```

```
export const main = async () => {
  const command = new DescribeAddressesCommand({
    // You can omit this property to show all addresses.
    AllocationIds: ["ALLOCATION_ID"],
  });

  try {
    const { Addresses } = await client.send(command);
    const addressList = Addresses.map((address) => ` • ${address.PublicIp}`);
    console.log("Elastic IP addresses:");
    console.log(addressList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DescribeAddresses中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明 EC2-Classic 中執行個體的指定彈性 IP 位址。

```
Get-EC2Address -AllocationId eipalloc-12345678
```

輸出：

```
AllocationId            : eipalloc-12345678
AssociationId           : eipassoc-12345678
Domain                  : vpc
InstanceId              : i-87654321
NetworkInterfaceId      : eni-12345678
NetworkInterfaceOwnerId : 12345678
PrivateIpAddress        : 10.0.2.172
PublicIp                : 198.51.100.2
```

範例 2：此範例說明 VPC 中執行個體的彈性 IP 位址。此語法需要 PowerShell 版本 3 或更新版本。

```
Get-EC2Address -Filter @{ Name="domain";Values="vpc" }
```

範例 3：此範例說明 EC2-Classic 中執行個體的指定彈性 IP 位址。

```
Get-EC2Address -PublicIp 203.0.113.17
```

輸出：

```
AllocationId            :
AssociationId           :
Domain                  : standard
InstanceId              : i-12345678
NetworkInterfaceId      :
NetworkInterfaceOwnerId :
PrivateIpAddress        :
PublicIp                : 203.0.113.17
```

範例 4：此範例說明 EC2-Classic 中執行個體的彈性 IP 位址。此語法需要 PowerShell 版本 3 或更新版本。

```
Get-EC2Address -Filter @{ Name="domain";Values="standard" }
```

範例 5：此範例說明您的所有彈性 IP 位址。

```
Get-EC2Address
```

範例 6：此範例會針對篩選器中提供的執行個體 ID 傳回公用和私有 IP

```
Get-EC2Address -Region eu-west-1 -Filter @{Name="instance-
id";Values="i-0c12d3f4f567ffb89"} | Select-Object PrivateIpAddress, PublicIp
```

輸出：

```
PrivateIpAddress PublicIp
---------------- --------
10.0.0.99        63.36.5.227
```

範例 7：此範例會擷取所有具有其配置識別碼、關聯識別碼和執行個體 ID 的彈性 IP

```
Get-EC2Address -Region eu-west-1 | Select-Object InstanceId, AssociationId,
 AllocationId, PublicIp
```

輸出：

```
InstanceId          AssociationId              AllocationId
 PublicIp
----------          -------------              ------------
 --------
                                               eipalloc-012e3b456789e1fad
 17.212.120.178
i-0c123dfd3415bac67 eipassoc-0e123456bb7890bdb eipalloc-01cd23ebf45f7890c
 17.212.124.77
                                               eipalloc-012345678eeabcfad
 17.212.225.7
i-0123d405c67e89a0c eipassoc-0c123b456783966ba eipalloc-0123cdd456a8f7892
 37.216.52.173
i-0f1bf2f34c5678d09 eipassoc-0e12934568a952d96 eipalloc-0e1c23e4d5e6789e4
 37.218.222.278
i-012e3cb4df567e8aa eipassoc-0d1b2fa4d67d03810 eipalloc-0123f456f78a01b58
 37.210.82.27
i-0123bcf4b567890e1 eipassoc-01d2345f678903fb1 eipalloc-0e1db23cfef5c45c7
 37.215.222.270
```

範例 8：此範例擷取符合標籤金鑰「類別」且值為「Prod」的 EC2 IP 位址清單

```
Get-EC2Address -Filter @{Name="tag:Category";Values="Prod"}
```

輸出：

```
AllocationId            : eipalloc-0123f456f81a01b58
AssociationId           : eipassoc-0d1b23a456d103810
CustomerOwnedIp         :
CustomerOwnedIpv4Pool   :
Domain                  : vpc
InstanceId              : i-012e3cb4df567e1aa
NetworkBorderGroup      : eu-west-1
NetworkInterfaceId      : eni-0123f41d5a60d5f40
NetworkInterfaceOwnerId : 123456789012
PrivateIpAddress        : 192.168.1.84
PublicIp                : 34.250.81.29
```

```
PublicIpv4Pool          : amazon
Tags                    : {Category, Name}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeAddresses式參考中的。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        oo_result = lo_ec2->describeaddresses( ) .                      "
 oo_result is returned for testing purposes. "
        DATA(lt_addresses) = oo_result->get_addresses( ).
        MESSAGE 'Retrieved information about Elastic IP addresses.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK DescribeAddresses中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeAvailabilityZones配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeAvailabilityZones。

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：

- 建置及管理彈性服務

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
            new DescribeAvailabilityZonesRequest());
        return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DescribeAvailabilityZones中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
```

```
    Aws::EC2::Model::DescribeAvailabilityZonesRequest describe_request;
    auto describe_outcome =
ec2Client.DescribeAvailabilityZones(describe_request);

    if (describe_outcome.IsSuccess()) {
        std::cout << std::left <<
                  std::setw(32) << "ZoneName" <<
                  std::setw(20) << "State" <<
                  std::setw(32) << "Region" << std::endl;

        const auto &zones =
                describe_outcome.GetResult().GetAvailabilityZones();

        for (const auto &zone: zones) {
            Aws::String stateString =

Aws::EC2::Model::AvailabilityZoneStateMapper::GetNameForAvailabilityZoneState(
                            zone.GetState());
            std::cout << std::left <<
                      std::setw(32) << zone.GetZoneName() <<
                      std::setw(20) << stateString <<
                      std::setw(32) << zone.GetRegionName() << std::endl;
        }
    }
    else {
        std::cerr << "Failed to describe availability zones:" <<
                  describe_outcome.GetError().GetMessage() << std::endl;
        result = false;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考DescribeAvailabilityZones中的。

CLI

AWS CLI

描述您的可用區域

下列範例 `describe-availability-zones` 針對可供您使用的可用區域顯示詳細資訊。回應包含僅適用於目前區域的可用區域。在這個範例中，它預設在 `us-west-2` (奧勒岡) 區域使用設定檔。

```
aws ec2 describe-availability-zones
```

輸出：

```
{
    "AvailabilityZones": [
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2a",
            "ZoneId": "usw2-az1",
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2b",
            "ZoneId": "usw2-az2",
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2c",
            "ZoneId": "usw2-az3",
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2d",
            "ZoneId": "usw2-az4",
```

```
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opted-in",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2-lax-1a",
            "ZoneId": "usw2-lax1-az1",
            "GroupName": "us-west-2-lax-1",
            "NetworkBorderGroup": "us-west-2-lax-1"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeAvailabilityZones](#)中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明目前可用區域的可用區域。

```
Get-EC2AvailabilityZone
```

輸出：

```
Messages    RegionName    State        ZoneName
--------    ----------    -----        --------
{}          us-west-2     available    us-west-2a
{}          us-west-2     available    us-west-2b
{}          us-west-2     available    us-west-2c
```

範例 2：此範例說明處於受損狀態的任何可用區域。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
Get-EC2AvailabilityZone -Filter @{ Name="state";Values="impaired" }
```

實施例 3：在 PowerShell 版本 2 中，您必須使用新對象創建過濾器。

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = "impaired"

Get-EC2AvailabilityZone -Filter $filter
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeAvailabilityZones式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
```

```
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def get_availability_zones(self):
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get availability zones: {err}.")
        else:
            return zones
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeAvailabilityZones中的 Python (博托 3) API 參考。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        oo_result = lo_ec2->describeavailabilityzones( ) .
 " oo_result is returned for testing purposes. "
        DATA(lt_zones) = oo_result->get_availabilityzones( ).
        MESSAGE 'Retrieved information about Availability Zones.' TYPE 'I'.

      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK DescribeAvailabilityZones中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭`DescribeBundleTasks`配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用`DescribeBundleTasks`。

CLI

AWS CLI

描述您的套裝軟體工作

此範例說明所有套裝軟體工作。

命令：

```
aws ec2 describe-bundle-tasks
```

輸出：

```
{
  "BundleTasks": [
    {
      "UpdateTime": "2015-09-15T13:26:54.000Z",
      "InstanceId": "i-1234567890abcdef0",
      "Storage": {
        "S3": {
            "Prefix": "winami",
            "Bucket": "bundletasks"
        }
      },
      "State": "bundling",
      "StartTime": "2015-09-15T13:24:35.000Z",
      "Progress": "3%",
      "BundleId": "bun-2a4e041c"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeBundleTasks中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的套裝軟體工作。

```
Get-EC2BundleTask -BundleId bun-12345678
```

範例 2：此範例說明狀態為「完成」或「失敗」的套裝軟體工作。

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "complete", "failed" )
```

```
Get-EC2BundleTask -Filter $filter
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeBundleTasks式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeCapacityReservations配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeCapacityReservations。

CLI

AWS CLI

範例 1：描述一或多個容量保留

下列describe-capacity-reservations範例顯示目前 AWS 區域中所有容量保留的詳細資訊。

```
aws ec2 describe-capacity-reservations
```

輸出：

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
            "EndDateType": "unlimited",
            "AvailabilityZone": "eu-west-1a",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "EphemeralStorage": false,
            "CreateDate": "2019-08-16T09:03:18.000Z",
            "AvailableInstanceCount": 1,
            "InstancePlatform": "Linux/UNIX",
            "TotalInstanceCount": 1,
            "State": "active",
            "Tenancy": "default",
```

```
            "EbsOptimized": true,
            "InstanceType": "a1.medium"
        },
        {
            "CapacityReservationId": "cr-abcdEXAMPLE9876ef ",
            "EndDateType": "unlimited",
            "AvailabilityZone": "eu-west-1a",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "EphemeralStorage": false,
            "CreateDate": "2019-08-07T11:34:19.000Z",
            "AvailableInstanceCount": 3,
            "InstancePlatform": "Linux/UNIX",
            "TotalInstanceCount": 3,
            "State": "cancelled",
            "Tenancy": "default",
            "EbsOptimized": true,
            "InstanceType": "m5.large"
        }
    ]
}
```

範例 2：描述一或多個容量保留

下列describe-capacity-reservations範例顯示有關指定容量保留的詳細資訊。

```
aws ec2 describe-capacity-reservations \
    --capacity-reservation-ids cr-1234abcd56EXAMPLE
```

輸出：

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-1234abcd56EXAMPLE",
            "EndDateType": "unlimited",
            "AvailabilityZone": "eu-west-1a",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "EphemeralStorage": false,
            "CreateDate": "2019-08-16T09:03:18.000Z",
            "AvailableInstanceCount": 1,
            "InstancePlatform": "Linux/UNIX",
```

```
            "TotalInstanceCount": 1,
            "State": "active",
            "Tenancy": "default",
            "EbsOptimized": true,
            "InstanceType": "a1.medium"
        }
    ]
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的檢視容量保留。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeCapacityReservations中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明區域的一或多個「產能預留」

```
Get-EC2CapacityReservation -Region eu-west-1
```

輸出：

```
AvailabilityZone       : eu-west-1b
AvailableInstanceCount : 2
CapacityReservationId  : cr-0c1f2345db6f7cdba
CreateDate             : 3/28/2019 9:29:41 AM
EbsOptimized           : True
EndDate                : 1/1/0001 12:00:00 AM
EndDateType            : unlimited
EphemeralStorage       : False
InstanceMatchCriteria  : open
InstancePlatform       : Windows
InstanceType           : m4.xlarge
State                  : active
Tags                   : {}
Tenancy                : default
TotalInstanceCount     : 2
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeCapacityReservations式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeCustomerGateways配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeCustomerGateways。

CLI

AWS CLI

描述您的客戶閘道

此範例說明您的客戶閘道。

命令：

```
aws ec2 describe-customer-gateways
```

輸出：

```
{
    "CustomerGateways": [
        {
            "CustomerGatewayId": "cgw-b4dc3961",
            "IpAddress": "203.0.113.12",
            "State": "available",
            "Type": "ipsec.1",
            "BgpAsn": "65000"
        },
        {
            "CustomerGatewayId": "cgw-0e11f167",
            "IpAddress": "12.1.2.3",
            "State": "available",
            "Type": "ipsec.1",
            "BgpAsn": "65534"
        }
    ]
}
```

描述特定客戶閘道

此範例說明指定的客戶閘道。

命令：

```
aws ec2 describe-customer-gateways --customer-gateway-ids cgw-0e11f167
```

輸出：

```
{
    "CustomerGateways": [
        {
            "CustomerGatewayId": "cgw-0e11f167",
            "IpAddress": "12.1.2.3",
            "State": "available",
            "Type": "ipsec.1",
            "BgpAsn": "65534"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeCustomerGateways](#)中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的客戶閘道。

```
Get-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

輸出：

```
BgpAsn            : 65534
CustomerGatewayId : cgw-1a2b3c4d
IpAddress         : 203.0.113.12
State             : available
Tags              : {}
Type              : ipsec.1
```

範例 2：此範例說明其狀態為擱置中或可用的任何客戶閘道。

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
```

```
$filter.Values = @( "pending", "available" )

Get-EC2CustomerGateway -Filter $filter
```

範例 3：此範例說明所有客戶閘道。

```
Get-EC2CustomerGateway
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeCustomerGateways式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeDhcpOptions配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeDhcpOptions。

CLI

AWS CLI

範例 1：描述您的 DHCP 選項

下列describe-dhcp-options範例會擷取有關 DHCP 選項的詳細資料。

```
aws ec2 describe-dhcp-options
```

輸出：

```
{
    "DhcpOptions": [
        {
            "DhcpConfigurations": [
                {
                    "Key": "domain-name",
                    "Values": [
                        {
                            "Value": "us-east-2.compute.internal"
                        }
                    ]
```

```
                },
                {
                    "Key": "domain-name-servers",
                    "Values": [
                        {
                            "Value": "AmazonProvidedDNS"
                        }
                    ]
                }
            ],
            "DhcpOptionsId": "dopt-19edf471",
            "OwnerId": "111122223333"
        },
        {
            "DhcpConfigurations": [
                {
                    "Key": "domain-name",
                    "Values": [
                        {
                            "Value": "us-east-2.compute.internal"
                        }
                    ]
                },
                {
                    "Key": "domain-name-servers",
                    "Values": [
                        {
                            "Value": "AmazonProvidedDNS"
                        }
                    ]
                }
            ],
            "DhcpOptionsId": "dopt-fEXAMPLE",
            "OwnerId": "111122223333"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS VPC 使用指南》中的〈使用 DHCP 選項集〉。

範例 2：描述您的 DHCP 選項並篩選輸出

下列`describe-dhcp-options`範例說明您的 DHCP 選項，並使用篩選器，僅傳回網域名稱伺服器的 DHCP 選項。`example.com`此範例使用`--query`參數僅顯示輸出中的組態資訊和 ID。

```
aws ec2 describe-dhcp-options \
    --filters Name=key,Values=domain-name-servers Name=value,Values=example.com \
    --query "DhcpOptions[*].[DhcpConfigurations,DhcpOptionsId]"
```

輸出：

```
[
    [
        [
            {
                "Key": "domain-name",
                "Values": [
                    {
                        "Value": "example.com"
                    }
                ]
            },
            {
                "Key": "domain-name-servers",
                "Values": [
                    {
                        "Value": "172.16.16.16"
                    }
                ]
            }
        ],
        "dopt-001122334455667ab"
    ]
]
```

如需詳細資訊，請參閱《AWS VPC 使用指南》中的〈使用 DHCP 選項集〉。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeDhcpOptions中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會列出您的 DHCP 選項集。

```
Get-EC2DhcpOption
```

輸出：

```
DhcpConfigurations                    DhcpOptionsId    Tag
------------------                    -------------    ---
{domain-name, domain-name-servers}    dopt-1a2b3c4d    {}
{domain-name, domain-name-servers}    dopt-2a3b4c5d    {}
{domain-name-servers}                 dopt-3a4b5c6d    {}
```

範例 2：此範例取得指定 DHCP 選項集的組態詳細資料。

```
(Get-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d).DhcpConfigurations
```

輸出：

```
Key                  Values
---                  ------
domain-name          {abc.local}
domain-name-servers  {10.0.0.101, 10.0.0.102}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeDhcpOptions式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeFlowLogs配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeFlowLogs。

CLI

AWS CLI

範例 1：描述所有流程記錄

下列describe-flow-logs範例會顯示所有流程記錄的詳細資料。

```
aws ec2 describe-flow-logs
```

輸出：

```
{
    "FlowLogs": [
        {
            "CreationTime": "2018-02-21T13:22:12.644Z",
            "DeliverLogsPermissionArn": "arn:aws:iam::123456789012:role/flow-
logs-role",
            "DeliverLogsStatus": "SUCCESS",
            "FlowLogId": "fl-aabbccdd112233445",
            "MaxAggregationInterval": 600,
            "FlowLogStatus": "ACTIVE",
            "LogGroupName": "FlowLogGroup",
            "ResourceId": "subnet-12345678901234567",
            "TrafficType": "ALL",
            "LogDestinationType": "cloud-watch-logs",
            "LogFormat": "${version} ${account-id} ${interface-id} ${srcaddr}
 ${dstaddr} ${srcport} ${dstport} ${protocol} ${packets} ${bytes} ${start} ${end}
 ${action} ${log-status}"
        },
        {
            "CreationTime": "2020-02-04T15:22:29.986Z",
            "DeliverLogsStatus": "SUCCESS",
            "FlowLogId": "fl-01234567890123456",
            "MaxAggregationInterval": 60,
            "FlowLogStatus": "ACTIVE",
            "ResourceId": "vpc-00112233445566778",
            "TrafficType": "ACCEPT",
            "LogDestinationType": "s3",
            "LogDestination": "arn:aws:s3:::my-flow-log-bucket/custom",
            "LogFormat": "${version} ${vpc-id} ${subnet-id} ${instance-id}
 ${interface-id} ${account-id} ${type} ${srcaddr} ${dstaddr} ${srcport}
 ${dstport} ${pkt-srcaddr} ${pkt-dstaddr} ${protocol} ${bytes} ${packets}
 ${start} ${end} ${action} ${tcp-flags} ${log-status}"
        }
    ]
}
```

範例 2：描述流程記錄的子集

下列`describe-flow-logs`範例使用篩選器，僅顯示 Amazon Logs 中指定日誌群組中的流程 CloudWatch 日誌的詳細資訊。

```
aws ec2 describe-flow-logs \
```

```
    --filter "Name=log-group-name,Values=MyFlowLogs"
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeFlowLogs中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明具有日誌目標類型 's3' 的一個或多個流程日誌

```
Get-EC2FlowLog -Filter @{Name="log-destination-type";Values="s3"}
```

輸出：

```
CreationTime              : 2/25/2019 9:07:36 PM
DeliverLogsErrorMessage   :
DeliverLogsPermissionArn  :
DeliverLogsStatus         : SUCCESS
FlowLogId                 : fl-01b2e3d45f67f8901
FlowLogStatus             : ACTIVE
LogDestination            : arn:aws:s3:::my-bucket-dd-tata
LogDestinationType        : s3
LogGroupName              :
ResourceId                : eni-01d2dda3456b7e890
TrafficType               : ALL
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeFlowLogs式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeHostReservationOfferings配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeHostReservationOfferings。

CLI

AWS CLI

說明專用主機保留項目

此範例說明可供購買之 M4 執行個體系列的專用主機保留項目。

命令：

```
aws ec2 describe-host-reservation-offerings --filter Name=instance-
family,Values=m4
```

輸出：

```
{
  "OfferingSet": [
      {
          "HourlyPrice": "1.499",
          "OfferingId": "hro-03f707bf363b6b324",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "UpfrontPrice": "0.000",
          "Duration": 31536000
      },
      {
          "HourlyPrice": "1.045",
          "OfferingId": "hro-0ef9181cabdef7a02",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "UpfrontPrice": "0.000",
          "Duration": 94608000
      },
      {
          "HourlyPrice": "0.714",
          "OfferingId": "hro-04567a15500b92a51",
          "InstanceFamily": "m4",
          "PaymentOption": "PartialUpfront",
          "UpfrontPrice": "6254.000",
          "Duration": 31536000
      },
      {
          "HourlyPrice": "0.484",
          "OfferingId": "hro-0d5d7a9d23ed7fbfe",
          "InstanceFamily": "m4",
          "PaymentOption": "PartialUpfront",
          "UpfrontPrice": "12720.000",
          "Duration": 94608000
      },
```

```
      {
          "HourlyPrice": "0.000",
          "OfferingId": "hro-05da4108ca998c2e5",
          "InstanceFamily": "m4",
          "PaymentOption": "AllUpfront",
          "UpfrontPrice": "23913.000",
          "Duration": 94608000
      },
      {
          "HourlyPrice": "0.000",
          "OfferingId": "hro-0a9f9be3b95a3dc8f",
          "InstanceFamily": "m4",
          "PaymentOption": "AllUpfront",
          "UpfrontPrice": "12257.000",
          "Duration": 31536000
      }
  ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeHostReservationOfferings](#)中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明可針對指定篩選器「執行個體系列」購買的專用主機保留區，其中 PaymentOption 為 " NoUpfront

```
Get-EC2HostReservationOffering -Filter @{Name="instance-family";Values="m4"} |
 Where-Object PaymentOption -eq NoUpfront
```

輸出：

```
CurrencyCode   :
Duration       : 94608000
HourlyPrice    : 1.307
InstanceFamily : m4
OfferingId     : hro-0c1f234567890d9ab
PaymentOption  : NoUpfront
UpfrontPrice   : 0.000

CurrencyCode   :
```

```
Duration       : 31536000
HourlyPrice    : 1.830
InstanceFamily : m4
OfferingId     : hro-04ad12aaaf34b5a67
PaymentOption  : NoUpfront
UpfrontPrice   : 0.000
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeHostReservationOfferings式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeHosts配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeHosts。

CLI

AWS CLI

檢視專用主機的詳細資訊

下列describe-hosts範例會顯示您 AWS 帳戶中available專用主機的詳細資料。

```
aws ec2 describe-hosts --filter "Name=state,Values=available"
```

輸出：

```
{
    "Hosts": [
        {
            "HostId": "h-07879acf49EXAMPLE",
            "Tags": [
                {
                    "Value": "production",
                    "Key": "purpose"
                }
            ],
            "HostProperties": {
                "Cores": 48,
```

```
                "TotalVCpus": 96,
                "InstanceType": "m5.large",
                "Sockets": 2
            },
            "Instances": [],
            "State": "available",
            "AvailabilityZone": "eu-west-1a",
            "AvailableCapacity": {
                "AvailableInstanceCapacity": [
                    {
                        "AvailableCapacity": 48,
                        "InstanceType": "m5.large",
                        "TotalCapacity": 48
                    }
                ],
                "AvailableVCpus": 96
            },
            "HostRecovery": "on",
            "AllocationTime": "2019-08-19T08:57:44.000Z",
            "AutoPlacement": "off"
        }
    ]
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的檢視專用主機。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeHosts中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例傳回 EC2 主機詳細資訊

```
Get-EC2Host
```

輸出：

```
AllocationTime    : 3/23/2019 4:55:22 PM
AutoPlacement     : off
AvailabilityZone  : eu-west-1b
AvailableCapacity : Amazon.EC2.Model.AvailableCapacity
```

```
ClientToken       :
HostId            : h-01e23f4cd567890f1
HostProperties    : Amazon.EC2.Model.HostProperties
HostReservationId :
Instances         : {}
ReleaseTime       : 1/1/0001 12:00:00 AM
State             : available
Tags              : {}
```

範例 2：此範例會 AvailableInstanceCapacity 針對主機進行查詢

```
Get-EC2Host -HostId h-01e23f4cd567899f1 | Select-Object -ExpandProperty
 AvailableCapacity | Select-Object -expand AvailableInstanceCapacity
```

輸出：

```
AvailableCapacity InstanceType TotalCapacity
----------------- ------------ -------------
11                m4.xlarge    11
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeHosts式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeIamInstanceProfileAssociations配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeIamInstanceProfileAssociations。

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：

- 建置及管理彈性服務

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
            new DescribeIamInstanceProfileAssociationsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("instance-id", new List<string>() { instanceId })
                },
            });
        return response.IamInstanceProfileAssociations[0];
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DescribeIamInstanceProfileAssociations中的。

CLI

AWS CLI

描述 IAM 執行個體設定檔關聯

此範例描述所有 IAM 執行個體設定檔關聯。

命令：

```
aws ec2 describe-iam-instance-profile-associations
```

輸出：

```
{
  "IamInstanceProfileAssociations": [
      {
          "InstanceId": "i-09eb09efa73ec1dee",
          "State": "associated",
          "AssociationId": "iip-assoc-0db249b1f25fa24b8",
          "IamInstanceProfile": {
              "Id": "AIPAJVQN4F5WVLGCJDRGM",
              "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"
          }
      },
      {
          "InstanceId": "i-0402909a2f4dffd14",
          "State": "associating",
          "AssociationId": "iip-assoc-0d1ec06278d29f44a",
          "IamInstanceProfile": {
              "Id": "AGJAJVQN4F5WVLGCJABCM",
              "Arn": "arn:aws:iam::123456789012:instance-profile/user1-role"
          }
      }
   ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeIamInstanceProfileAssociations中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DescribeIamInstanceProfileAssociations中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
```

```
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def get_instance_profile(self, instance_id):
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instance profile association for instance
{instance_id}: {err}"
            )
        else:
```

```
            return response["IamInstanceProfileAssociations"][0]
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeIamInstanceProfileAssociations中的 Python (博托 3) API 參考。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeIdFormat配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeIdFormat。

CLI

AWS CLI

範例 1：描述資源的 ID 格式

下列describe-id-format範例說明安全性群組的 ID 格式。

```
aws ec2 describe-id-format \
    --resource security-group
```

在下列範例輸出中，Deadline值表示此資源類型從短 ID 格式永久切換為長 ID 格式的截止日期在 2018 年 8 月 15 日 00:00 UTC 到期。

```
{
    "Statuses": [
        {
            "Deadline": "2018-08-15T00:00:00.000Z",
            "Resource": "security-group",
            "UseLongIds": true
        }
    ]
}
```

範例 2：描述所有資源的 ID 格式

下列`describe-id-format`範例說明所有資源類型的 ID 格式。所有支援短 ID 格式的資源類型都會切換為使用長 ID 格式。

```
aws ec2 describe-id-format
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeIdFormat](#)中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定資源類型的 ID 格式。

```
Get-EC2IdFormat -Resource instance
```

輸出：

```
Resource       UseLongIds
--------       ----------
instance       False
```

範例 2：此範例說明支援較長 ID 之所有資源類型的 ID 格式。

```
Get-EC2IdFormat
```

輸出：

```
Resource       UseLongIds
--------       ----------
reservation    False
instance       False
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程[DescribeIdFormat](#)式參考中的。

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 開發套件建立 Amazon EC2 資源](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**DescribeIdentityIdFormat**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeIdentityIdFormat。

CLI

AWS CLI

描述 IAM 角色的 ID 格式

下列describe-identity-id-format範例說明 AWS 帳戶EC2Role中由 IAM 角色建立的執行個體所收到的 ID 格式。

```
aws ec2 describe-identity-id-format \
    --principal-arn arn:aws:iam::123456789012:role/my-iam-role \
    --resource instance
```

下列輸出指出此角色建立的執行個體會接收長 ID 格式的 ID。

```
{
    "Statuses": [
        {
            "Deadline": "2016-12-15T00:00:00Z",
            "Resource": "instance",
            "UseLongIds": true
        }
    ]
}
```

描述 IAM 使用者的 ID 格式

下列describe-identity-id-format範例說明 IAM 使用者AdminUser在您 AWS 帳戶中建立的快照所接收到的 ID 格式。

```
aws ec2 describe-identity-id-format \
    --principal-arn arn:aws:iam::123456789012:user/AdminUser \
    --resource snapshot
```

輸出表示此使用者建立的快照會收到長 ID 格式的 ID。

```
{
```

```
    "Statuses": [
        {
            "Deadline": "2016-12-15T00:00:00Z",
            "Resource": "snapshot",
            "UseLongIds": true
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeIdentityIdFormat中的。

PowerShell

用於的工具 PowerShell

示例 1：此示例返回給定角色的資源「圖像」的 ID 格式

```
Get-EC2IdentityIdFormat -PrincipalArn arn:aws:iam::123456789511:role/JDBC -
Resource image
```

輸出：

```
Deadline             Resource UseLongIds
--------             -------- ----------
8/2/2018 11:30:00 PM image    True
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeIdentityIdFormat式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeImageAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeImageAttribute。

CLI

AWS CLI

描述 AMI 的啟動權限

此範例說明指定 AMI 的啟動權限。

命令：

```
aws ec2 describe-image-attribute --image-id ami-5731123e --attribute
 launchPermission
```

輸出：

```
{
    "LaunchPermissions": [
        {
            "UserId": "123456789012"
        }
    ],
    "ImageId": "ami-5731123e",
}
```

描述 AMI 的產品代碼

此範例說明指定 AMI 的產品代碼。請注意，此 AMI 沒有產品代碼。

命令：

```
aws ec2 describe-image-attribute --image-id ami-5731123e --attribute productCodes
```

輸出：

```
{
    "ProductCodes": [],
    "ImageId": "ami-5731123e",
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeImageAttribute中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例取得指定 AMI 的說明。

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute description
```

輸出：

```
BlockDeviceMappings : {}
Description         : My image description
ImageId             : ami-12345678
KernelId            :
LaunchPermissions   : {}
ProductCodes        : {}
RamdiskId           :
SriovNetSupport     :
```

範例 2：此範例會取得指定 AMI 的啟動權限。

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

輸出：

```
BlockDeviceMappings : {}
Description         :
ImageId             : ami-12345678
KernelId            :
LaunchPermissions   : {all}
ProductCodes        : {}
RamdiskId           :
SriovNetSupport     :
```

範例 3：此範例會測試是否已啟用增強型網路。

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute sriovNetSupport
```

輸出：

```
BlockDeviceMappings : {}
Description         :
ImageId             : ami-12345678
KernelId            :
LaunchPermissions   : {}
ProductCodes        : {}
RamdiskId           :
```

```
SriovNetSupport     : simple
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeImageAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeImages配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeImages。

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：

- 開始使用執行個體

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
#############################################################################
# function ec2_describe_images
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 images.
#
# Parameters:
#       -i image_ids - A space-separated  list of image IDs (optional).
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
```

```
###############################################################################
function ec2_describe_images() {
  local image_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_images"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 images."
    echo "  -i image_ids - A space-separated list of image IDs (optional)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) image_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$image_ids" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--image-ids" $image_ids)
  fi

  response=$(aws ec2 describe-images \
    "${aws_cli_args[@]}" \
    --query 'Images[*].[Description,Architecture,ImageId]' \
    --output text) || {
    aws_cli_error_log ${?}
```

```
    errecho "ERROR: AWS reports describe-images operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

此範例中使用的公用程式函數。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
```

```
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeImages中的。

CLI

AWS CLI

範例 1：描述 AMI

下列 describe-images 範例描述指定區域中的指定 AMI。

```
aws ec2 describe-images \
    --region us-east-1 \
    --image-ids ami-1234567890EXAMPLE
```

輸出：

```
{
    "Images": [
        {
            "VirtualizationType": "hvm",
            "Description": "Provided by Red Hat, Inc.",
            "PlatformDetails": "Red Hat Enterprise Linux",
            "EnaSupport": true,
            "Hypervisor": "xen",
            "State": "available",
            "SriovNetSupport": "simple",
            "ImageId": "ami-1234567890EXAMPLE",
            "UsageOperation": "RunInstances:0010",
            "BlockDeviceMappings": [
```

```
                {
                    "DeviceName": "/dev/sda1",
                    "Ebs": {
                        "SnapshotId": "snap-111222333444aaabb",
                        "DeleteOnTermination": true,
                        "VolumeType": "gp2",
                        "VolumeSize": 10,
                        "Encrypted": false
                    }
                }
            ],
            "Architecture": "x86_64",
            "ImageLocation": "123456789012/RHEL-8.0.0_HVM-20190618-x86_64-1-
Hourly2-GP2",
            "RootDeviceType": "ebs",
            "OwnerId": "123456789012",
            "RootDeviceName": "/dev/sda1",
            "CreationDate": "2019-05-10T13:17:12.000Z",
            "Public": true,
            "ImageType": "machine",
            "Name": "RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-GP2"
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon Machine Images (AMI)。

範例 2：根據篩選條件描述 AMI

以下 `describe-images` 範例描述 Amazon 所提供，且受 Amazon EBS 支援的 Windows AMI。

```
aws ec2 describe-images \
    --owners amazon \
    --filters "Name=platform,Values=windows" "Name=root-device-type,Values=ebs"
```

如需 `describe-images` 的輸出範例，請參閱範例 1。

如需使用篩選條件的其他範例，請參閱《Amazon EC2 使用者指南》中的列出與篩選您的資源。

範例 3：根據標籤描述 AMI

下列 describe-images 範例描述具有標籤 Type=Custom 的所有 AMI。此範例使用 --query 參數，僅顯示 AMI ID。

```
aws ec2 describe-images \
    --filters "Name=tag:Type,Values=Custom" \
    --query 'Images[*].[ImageId]' \
    --output text
```

輸出：

```
ami-1234567890EXAMPLE
ami-0abcdef1234567890
```

如需使用標籤篩選條件的其他範例，請參閱《Amazon EC2 使用者指南》中的使用標籤。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeImages中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { paginateDescribeImages } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// List at least the first i386 image available for EC2 instances.
export const main = async () => {
  // The paginate function is a wrapper around the base command.
  const paginator = paginateDescribeImages(
    // Without limiting the page size, this call can take a long time. pageSize
 is just sugar for
    // the MaxResults property in the base command.
    { client, pageSize: 25 },
    {
```

```
      // There are almost 70,000 images available. Be specific with your
 filtering
      // to increase efficiency.
      // See https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/clients/
client-ec2/interfaces/describeimagescommandinput.html#filters
      Filters: [{ Name: "architecture", Values: ["x86_64"] }],
    },
  );

  try {
    const arm64Images = [];
    for await (const page of paginator) {
      if (page.Images.length) {
        arm64Images.push(...page.Images);
        // Once we have at least 1 result, we can stop.
        if (arm64Images.length >= 1) {
          break;
        }
      }
    }
    console.log(arm64Images);
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DescribeImages中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的 AMI。

```
Get-EC2Image -ImageId ami-12345678
```

輸出：

```
Architecture        : x86_64
BlockDeviceMappings : {/dev/xvda}
CreationDate        : 2014-10-20T00:56:28.000Z
```

```
Description         : My image
Hypervisor          : xen
ImageId             : ami-12345678
ImageLocation       : 123456789012/my-image
ImageOwnerAlias     :
ImageType           : machine
KernelId            :
Name                : my-image
OwnerId             : 123456789012
Platform            :
ProductCodes        : {}
Public              : False
RamdiskId           :
RootDeviceName      : /dev/xvda
RootDeviceType      : ebs
SriovNetSupport     : simple
State               : available
StateReason         :
Tags                : {Name}
VirtualizationType  : hvm
```

範例 2：此範例說明您擁有的 AMI。

```
Get-EC2Image -owner self
```

範例 3：此範例說明執行 Microsoft 視窗伺服器的公用 AMI。

```
Get-EC2Image -Filter @{ Name="platform"; Values="windows" }
```

範例 4：此範例說明「us-west-2」區域中的所有公用 AMI。

```
Get-EC2Image -Region us-west-2
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeImages式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
                          wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def get_images(self, image_ids):
        """
        Gets information about Amazon Machine Images (AMIs) from a list of AMI
 IDs.

        :param image_ids: The list of AMIs to look up.
        :return: A list of Boto3 Image objects that represent the requested AMIs.
        """
        try:
```

```
            images = list(self.ec2_resource.images.filter(ImageIds=image_ids))
        except ClientError as err:
            logger.error(
                "Couldn't get images. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return images
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeImages中的 Python (博托 3) API 參考。

Rust

適用於 Rust 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
// A simple Rust program to list all Amazon images in the currently configured
 region.
#[tokio::main]
async fn main() -> Result<(), Error> {
    let config = aws_config::load_from_env().await;
    let client = Client::new(&config);

    let resp = client.describe_images().owners("amazon").send().await?;

    println!("AWS SDK for Rust v{}", PKG_VERSION);
    println!("Describing Amazon Machine Images (AMIs):");

    let mut images: Vec<_> = resp
        .images()
        .iter()
```

```
        .filter(|i| {
            i.description()
                .filter(|i| i.contains("Amazon Linux AMI 2023"))
                .is_some()
        })
        .collect();
    images.sort_by(|a, b| a.description.cmp(&b.description));

    if images.is_empty() {
        println!("No images found.");
        return Ok(());
    }

    for image in images {
        let id = image.image_id().unwrap_or_default();
        let description = image.description().unwrap_or_default();

        println!("{id}: {description}");
    }

    Ok(())
}
```

- 如需 API 的詳細資訊，請參閱 AWS SDK DescribeImages中的 Rust API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeImportImageTasks配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeImportImageTasks。

CLI

AWS CLI

監視匯入影像工作的步驟

下列describe-import-image-tasks範例會檢查指定匯入映像工作的狀態。

```
aws ec2 describe-import-image-tasks \
```

```
    --import-task-ids import-ami-1234567890abcdef0
```

正在進行的匯入影像工作的輸出。

```
{
    "ImportImageTasks": [
        {
            "ImportTaskId": "import-ami-1234567890abcdef0",
            "Progress": "28",
            "SnapshotDetails": [
                {
                    "DiskImageSize": 705638400.0,
                    "Format": "ova",
                    "Status": "completed",
                    "UserBucket": {
                        "S3Bucket": "my-import-bucket",
                        "S3Key": "vms/my-server-vm.ova"
                    }
                }
            ],
            "Status": "active",
            "StatusMessage": "converting"
        }
    ]
}
```

已完成之匯入映像工作的輸出。產生的 AMI 的識別碼由提供ImageId。

```
{
    "ImportImageTasks": [
        {
            "ImportTaskId": "import-ami-1234567890abcdef0",
            "ImageId": "ami-1234567890abcdef0",
            "SnapshotDetails": [
                {
                    "DiskImageSize": 705638400.0,
                    "Format": "ova",
                    "SnapshotId": "snap-1234567890abcdef0"
                    "Status": "completed",
                    "UserBucket": {
                        "S3Bucket": "my-import-bucket",
                        "S3Key": "vms/my-server-vm.ova"
                    }
```

```
                }
            ],
            "Status": "completed"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeImportImageTasks中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的影像匯入工作。

```
Get-EC2ImportImageTask -ImportTaskId import-ami-hgfedcba
```

輸出：

```
Architecture    : x86_64
Description     : Windows Image 2
Hypervisor      :
ImageId         : ami-1a2b3c4d
ImportTaskId    : import-ami-hgfedcba
LicenseType     : AWS
Platform        : Windows
Progress        :
SnapshotDetails : {/dev/sda1}
Status          : completed
StatusMessage   :
```

範例 2：此範例說明所有映像匯入工作。

```
Get-EC2ImportImageTask
```

輸出：

```
Architecture    :
Description     : Windows Image 1
Hypervisor      :
```

```
ImageId         :
ImportTaskId    : import-ami-abcdefgh
LicenseType     : AWS
Platform        : Windows
Progress        :
SnapshotDetails : {}
Status          : deleted
StatusMessage   : User initiated task cancelation

Architecture    : x86_64
Description     : Windows Image 2
Hypervisor      :
ImageId         : ami-1a2b3c4d
ImportTaskId    : import-ami-hgfedcba
LicenseType     : AWS
Platform        : Windows
Progress        :
SnapshotDetails : {/dev/sda1}
Status          : completed
StatusMessage   :
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeImportImageTasks式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeImportSnapshotTasks配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeImportSnapshotTasks。

CLI

AWS CLI

監視匯入快照工作

下列describe-import-snapshot-tasks範例會檢查指定匯入快照工作的狀態。

```
aws ec2 describe-import-snapshot-tasks \
    --import-task-ids import-snap-1234567890abcdef0
```

正在進行的匯入快照工作的輸出：

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "My server VMDK",
            "ImportTaskId": "import-snap-1234567890abcdef0",
            "SnapshotTaskDetail": {
                "Description": "My server VMDK",
                "DiskImageSize": "705638400.0",
                "Format": "VMDK",
                "Progress": "42",
                "Status": "active",
                "StatusMessage": "downloading/converting",
                "UserBucket": {
                    "S3Bucket": "my-import-bucket",
                    "S3Key": "vms/my-server-vm.vmdk"
                }
            }
        }
    ]
}
```

已完成之匯入快照工作的輸出。所產生快照的識別碼由提供SnapshotId。

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "My server VMDK",
            "ImportTaskId": "import-snap-1234567890abcdef0",
            "SnapshotTaskDetail": {
                "Description": "My server VMDK",
                "DiskImageSize": "705638400.0",
                "Format": "VMDK",
                "SnapshotId": "snap-1234567890abcdef0"
                "Status": "completed",
                "UserBucket": {
                    "S3Bucket": "my-import-bucket",
                    "S3Key": "vms/my-server-vm.vmdk"
                }
            }
        }
    ]
```

```
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeImportSnapshotTasks中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的快照匯入工作。

```
Get-EC2ImportSnapshotTask -ImportTaskId import-snap-abcdefgh
```

輸出：

```
Description             ImportTaskId               SnapshotTaskDetail

-----------------       --------------------       ------------------

Disk Image Import 1     import-snap-abcdefgh
 Amazon.EC2.Model.SnapshotTaskDetail
```

範例 2：此範例說明所有快照匯入工作。

```
Get-EC2ImportSnapshotTask
```

輸出：

```
Description             ImportTaskId               SnapshotTaskDetail

-----------------       --------------------       ------------------

Disk Image Import 1     import-snap-abcdefgh
 Amazon.EC2.Model.SnapshotTaskDetail
Disk Image Import 2     import-snap-hgfedcba
 Amazon.EC2.Model.SnapshotTaskDetail
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeImportSnapshotTasks式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配 AWS 開發套件或 CLI 使用 DescribeInstanceAttribute

下列程式碼範例會示範如何使用DescribeInstanceAttribute。

CLI

AWS CLI

說明執行個體類型

此範例說明指定執行個體的執行個體類型。

命令：

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 instanceType
```

輸出：

```
{
    "InstanceId": "i-1234567890abcdef0"
    "InstanceType": {
        "Value": "t1.micro"
    }
}
```

描述屬 disableApiTermination 性

此範例說明指定執行個體的disableApiTermination屬性。

命令：

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 disableApiTermination
```

輸出：

```
{
```

```
"InstanceId": "i-1234567890abcdef0"
    "DisableApiTermination": {
        "Value": "false"
    }
}
```

說明執行個體的區塊裝置對應

此範例說明指定執行個體的blockDeviceMapping屬性。

命令：

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 blockDeviceMapping
```

輸出：

```
{
    "InstanceId": "i-1234567890abcdef0"
    "BlockDeviceMappings": [
        {
            "DeviceName": "/dev/sda1",
            "Ebs": {
                "Status": "attached",
                "DeleteOnTermination": true,
                "VolumeId": "vol-049df61146c4d7901",
                "AttachTime": "2013-05-17T22:42:34.000Z"
            }
        },
        {
            "DeviceName": "/dev/sdf",
            "Ebs": {
                "Status": "attached",
                "DeleteOnTermination": false,
                "VolumeId": "vol-049df61146c4d7901",
                "AttachTime": "2013-09-10T23:07:00.000Z"
            }
        }
    ],
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeInstanceAttribute中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定執行個體的執行個體類型。

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute instanceType
```

輸出：

```
InstanceType                     : t2.micro
```

範例 2：此範例說明是否為指定的執行個體啟用增強型聯網。

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

輸出：

```
SriovNetSupport                  : simple
```

範例 3：此範例說明指定執行個體的安全性群組。

```
(Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute groupSet).Groups
```

輸出：

```
GroupId
-------
sg-12345678
sg-45678901
```

範例 4：此範例說明是否針對指定的執行個體啟用 EBS 最佳化。

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

輸出：

```
EbsOptimized                     : False
```

範例 5：此範例說明指定執行個體的 disableApiTermination " 屬性。

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute disableApiTermination
```

輸出：

```
DisableApiTermination            : False
```

範例 6：此範例說明指定執instanceInitiatedShutdown行個體的「行為」屬性。

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute
 instanceInitiatedShutdownBehavior
```

輸出：

```
InstanceInitiatedShutdownBehavior : stop
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeInstanceAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**DescribeInstanceStatus**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeInstanceStatus。

CLI

AWS CLI

描述執行個體的狀態

下列 describe-instance-status 範例會描述指定執行個體的目前狀態。

```
aws ec2 describe-instance-status \
    --instance-ids i-1234567890abcdef0
```

輸出：

```
{
    "InstanceStatuses": [
```

```
        {
            "InstanceId": "i-1234567890abcdef0",
            "InstanceState": {
                "Code": 16,
                "Name": "running"
            },
            "AvailabilityZone": "us-east-1d",
            "SystemStatus": {
                "Status": "ok",
                "Details": [
                    {
                        "Status": "passed",
                        "Name": "reachability"
                    }
                ]
            },
            "InstanceStatus": {
                "Status": "ok",
                "Details": [
                    {
                        "Status": "passed",
                        "Name": "reachability"
                    }
                ]
            }
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的監控您的執行個體狀態。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeInstanceStatus中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定執行處理的狀態。

```
Get-EC2InstanceStatus -InstanceId i-12345678
```

輸出：

```
AvailabilityZone : us-west-2a
Events           : {}
InstanceId       : i-12345678
InstanceState    : Amazon.EC2.Model.InstanceState
Status           : Amazon.EC2.Model.InstanceStatusSummary
SystemStatus     : Amazon.EC2.Model.InstanceStatusSummary
```

```
$status = Get-EC2InstanceStatus -InstanceId i-12345678
$status.InstanceState
```

輸出：

```
Code    Name
----    ----
16      running
```

```
$status.Status
```

輸出：

```
Details           Status
-------           ------
{reachability}    ok
```

```
$status.SystemStatus
```

輸出：

```
Details           Status
-------           ------
{reachability}    ok
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeInstanceStatus式參考中的。

Rust

適用於 Rust 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
async fn show_all_events(client: &Client) -> Result<(), Error> {
    let resp = client.describe_regions().send().await.unwrap();

    for region in resp.regions.unwrap_or_default() {
        let reg: &'static str =
 Box::leak(Box::from(region.region_name().unwrap()));
        let region_provider =
 RegionProviderChain::default_provider().or_else(reg);
        let config = aws_config::from_env().region(region_provider).load().await;
        let new_client = Client::new(&config);

        let resp = new_client.describe_instance_status().send().await;

        println!("Instances in region {}:", reg);
        println!();

        for status in resp.unwrap().instance_statuses() {
            println!(
                "  Events scheduled for instance ID: {}",
                status.instance_id().unwrap_or_default()
            );
            for event in status.events() {
                println!("    Event ID:     {}",
 event.instance_event_id().unwrap());
                println!("    Description:  {}", event.description().unwrap());
                println!("    Event code:   {}", event.code().unwrap().as_ref());
                println!();
            }
        }
    }

    Ok(())
```

```
}
```

- 如需 API 的詳細資訊，請參閱 AWS SDK DescribeInstanceStatus中的 Rust API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeInstanceTypes配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeInstanceTypes。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Describe the instance types available.
    /// </summary>
    /// <returns>A list of instance type information.</returns>
    public async Task<List<InstanceTypeInfo>>
DescribeInstanceTypes(ArchitectureValues architecture)
    {
        var request = new DescribeInstanceTypesRequest();

        var filters = new List<Filter>
            { new Filter("processor-info.supported-architecture", new
List<string> { architecture.ToString() }) };
        filters.Add(new Filter("instance-type", new() { "*.micro", "*.small" }));
```

```
        request.Filters = filters;
        var instanceTypes = new List<InstanceTypeInfo>();

        var paginator = _amazonEC2.Paginators.DescribeInstanceTypes(request);
        await foreach (var instanceType in paginator.InstanceTypes)
        {
            instanceTypes.Add(instanceType);
        }
        return instanceTypes;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DescribeInstanceTypes中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
##############################################################################
# ec2_describe_instance_types
#
# This function describes EC2 instance types filtered by processor architecture
# and optionally by instance type. It takes the following arguments:
#
# -a, --architecture ARCHITECTURE  Specify the processor architecture (e.g.,
 x86_64)
# -t, --type INSTANCE_TYPE         Comma-separated list of instance types (e.g.,
 t2.micro)
# -h, --help                       Show the usage help
#
# The function prints the instance type and supported architecture for each
# matching instance type.
##############################################################################
function ec2_describe_instance_types() {
  local architecture=""
```

```
  local instance_types=""

  # bashsupport disable=BP5008
  function usage() {
    echo "Usage: ec2_describe_instance_types [-a|--architecture ARCHITECTURE] [-
t|--type INSTANCE_TYPE] [-h|--help]"
    echo "  -a, --architecture ARCHITECTURE  Specify the processor architecture
 (e.g., x86_64)"
    echo "  -t, --type INSTANCE_TYPE         Comma-separated list of instance
 types (e.g., t2.micro)"
    echo "  -h, --help                       Show this help message"
  }

  while [[ $# -gt 0 ]]; do
    case "$1" in
      -a | --architecture)
        architecture="$2"
        shift 2
        ;;
      -t | --type)
        instance_types="$2"
        shift 2
        ;;
      -h | --help)
        usage
        return 0
        ;;
      *)
        echo "Unknown argument: $1"
        return 1
        ;;
    esac
  done

  if [[ -z "$architecture" ]]; then
    errecho "Error: Architecture not specified."
    usage
    return 1
  fi

  if [[ -z "$instance_types" ]]; then
    errecho "Error: Instance type not specified."
    usage
    return 1
```

```
  fi

  local tmp_json_file="temp_ec2.json"
  echo -n '[
    {
      "Name": "processor-info.supported-architecture",
      "Values": [' >"$tmp_json_file"

  local items
  IFS=',' read -ra items <<<"$architecture"
  local array_size
  array_size=${#items[@]}
  for i in $(seq 0 $((array_size - 1))); do
    echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
      echo -n ',' >>"$tmp_json_file"
    fi
  done
  echo -n ']},
    {
    "Name": "instance-type",
      "Values": [' >>"$tmp_json_file"
  IFS=',' read -ra items <<<"$instance_types"
  local array_size
  array_size=${#items[@]}
  for i in $(seq 0 $((array_size - 1))); do
    echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
      echo -n ',' >>"$tmp_json_file"
    fi
  done

  echo -n ']}]' >>"$tmp_json_file"

  local response
  response=$(aws ec2 describe-instance-types --filters file://"$tmp_json_file" \
    --query 'InstanceTypes[*].[InstanceType]' --output text)

  local error_code=$?

  rm "$tmp_json_file"

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
```

```
    echo "ERROR: AWS reports describe-instance-types operation failed."
    return 1
  fi

  echo "$response"
  return 0
}
```

此範例中使用的公用程式函數。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
```

```
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeInstanceTypes](#)中的。

CLI

AWS CLI

範例 1：描述執行個體類型

下列 `describe-instance-types` 範例顯示指定執行個體類型的詳細資訊。

```
aws ec2 describe-instance-types \
    --instance-types t2.micro
```

輸出：

```
{
    "InstanceTypes": [
        {
            "InstanceType": "t2.micro",
            "CurrentGeneration": true,
            "FreeTierEligible": true,
            "SupportedUsageClasses": [
                "on-demand",
                "spot"
            ],
            "SupportedRootDeviceTypes": [
                "ebs"
            ],
            "BareMetal": false,
            "Hypervisor": "xen",
            "ProcessorInfo": {
```

```
                "SupportedArchitectures": [
                    "i386",
                    "x86_64"
                ],
                "SustainedClockSpeedInGhz": 2.5
            },
            "VCpuInfo": {
                "DefaultVCpus": 1,
                "DefaultCores": 1,
                "DefaultThreadsPerCore": 1,
                "ValidCores": [
                    1
                ],
                "ValidThreadsPerCore": [
                    1
                ]
            },
            "MemoryInfo": {
                "SizeInMiB": 1024
            },
            "InstanceStorageSupported": false,
            "EbsInfo": {
                "EbsOptimizedSupport": "unsupported",
                "EncryptionSupport": "supported"
            },
            "NetworkInfo": {
                "NetworkPerformance": "Low to Moderate",
                "MaximumNetworkInterfaces": 2,
                "Ipv4AddressesPerInterface": 2,
                "Ipv6AddressesPerInterface": 2,
                "Ipv6Supported": true,
                "EnaSupport": "unsupported"
            },
            "PlacementGroupInfo": {
                "SupportedStrategies": [
                    "partition",
                    "spread"
                ]
            },
            "HibernationSupported": false,
            "BurstablePerformanceSupported": true,
            "DedicatedHostsSupported": false,
            "AutoRecoverySupported": true
        }
```

```
    ]
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的執行個體類型。

範例 2：若要篩選可用的執行個體類型

您可以指定篩選條件，將結果範圍限制為具有特定特性的執行個體類型。下列 `describe-instance-types` 範例列出支援休眠的執行個體類型。

```
aws ec2 describe-instance-types \
    --filters Name=hibernation-supported,Values=true --query
 'InstanceTypes[*].InstanceType'
```

輸出：

```
[
    "m5.8xlarge",
    "r3.large",
    "c3.8xlarge",
    "r5.large",
    "m4.4xlarge",
    "c4.large",
    "m5.xlarge",
    "m4.xlarge",
    "c3.large",
    "c4.8xlarge",
    "c4.4xlarge",
    "c5.xlarge",
    "c5.12xlarge",
    "r5.4xlarge",
    "c5.4xlarge"
]
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的執行個體類型。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeInstanceTypes中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   // Get a list of instance types.
   public static String getInstanceTypes(Ec2Client ec2) {
       String instanceType;
       try {
           DescribeInstanceTypesRequest typesRequest =
DescribeInstanceTypesRequest.builder()
                .maxResults(10)
                .build();

           DescribeInstanceTypesResponse response =
ec2.describeInstanceTypes(typesRequest);
           List<InstanceTypeInfo> instanceTypes = response.instanceTypes();
           for (InstanceTypeInfo type : instanceTypes) {
               System.out.println("The memory information of this type is " +
type.memoryInfo().sizeInMiB());
               System.out.println("Network information is " +
type.networkInfo().toString());
               System.out.println("Instance type is " +
type.instanceType().toString());
               instanceType = type.instanceType().toString();
               if (instanceType.compareTo("t2.2xlarge")  == 0){
                   return instanceType;
               }
           }

       } catch (SsmException e) {
           System.err.println(e.getMessage());
           System.exit(1);
       }
       return "";
   }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考DescribeInstanceTypes中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import {
  paginateDescribeInstanceTypes,
  DescribeInstanceTypesCommand,
} from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// List at least the first arm64 EC2 instance type available.
export const main = async () => {
  // The paginate function is a wrapper around the underlying command.
  const paginator = paginateDescribeInstanceTypes(
    // Without limiting the page size, this call can take a long time. pageSize
 is just sugar for
    // the MaxResults property in the underlying command.
    { client, pageSize: 25 },
    {
      Filters: [
        { Name: "processor-info.supported-architecture", Values: ["x86_64"] },
        { Name: "free-tier-eligible", Values: ["true"] },
      ],
    }
  );

  try {
    const instanceTypes = [];

    for await (const page of paginator) {
      if (page.InstanceTypes.length) {
        instanceTypes.push(...page.InstanceTypes);
```

```
        // When we have at least 1 result, we can stop.
        if (instanceTypes.length >= 1) {
          break;
        }
      }
    }
    console.log(instanceTypes);
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DescribeInstanceTypes中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
// Get a list of instance types.
suspend fun getInstanceTypesSc(): String {
    var instanceType = ""
    val filterObs = ArrayList<Filter>()
    val filter =
        Filter {
            name = "processor-info.supported-architecture"
            values = listOf("arm64")
        }

    filterObs.add(filter)
    val typesRequest =
        DescribeInstanceTypesRequest {
            filters = filterObs
            maxResults = 10
```

```
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeInstanceTypes(typesRequest)
        response.instanceTypes?.forEach { type ->
            println("The memory information of this type is
 ${type.memoryInfo?.sizeInMib}")
            println("Maximum number of network cards is
 ${type.networkInfo?.maximumNetworkCards}")
            instanceType = type.instanceType.toString()
        }
        return instanceType
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK DescribeInstanceTypes中的 Kotlin API 參考。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
                         wraps instance actions.
        """
        self.ec2_resource = ec2_resource
```

```
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def get_instance_types(self, architecture):
        """
        Gets instance types that support the specified architecture and are
designated
        as either 'micro' or 'small'. When an instance is created, the instance
type
        you specify must support the architecture of the AMI you use.

        :param architecture: The kind of architecture the instance types must
support,
                             such as 'x86_64'.
        :return: A list of instance types that support the specified architecture
                 and are either 'micro' or 'small'.
        """
        try:
            inst_types = []
            it_paginator = self.ec2_resource.meta.client.get_paginator(
                "describe_instance_types"
            )
            for page in it_paginator.paginate(
                Filters=[
                    {
                        "Name": "processor-info.supported-architecture",
                        "Values": [architecture],
                    },
                    {"Name": "instance-type", "Values": ["*.micro", "*.small"]},
                ]
            ):
                inst_types += page["InstanceTypes"]
        except ClientError as err:
            logger.error(
                "Couldn't get instance types. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

```
        else:
            return inst_types
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeInstanceTypes中的 Python (博托 3) API 參考。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeInstances。

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：

- 建置及管理彈性服務
- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Get information about existing EC2 images.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task DescribeInstances()
    {
        // List all EC2 instances.
        await GetInstanceDescriptions();
```

```
        string tagName = "IncludeInList";
        string tagValue = "Yes";
        await GetInstanceDescriptionsFiltered(tagName, tagValue);
    }

    /// <summary>
    /// Get information for all existing Amazon EC2 instances.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task GetInstanceDescriptions()
    {
        Console.WriteLine("Showing all instances:");
        var paginator = _amazonEC2.Paginators.DescribeInstances(new
DescribeInstancesRequest());

        await foreach (var response in paginator.Responses)
        {
            foreach (var reservation in response.Reservations)
            {
                foreach (var instance in reservation.Instances)
                {
                    Console.Write($"Instance ID: {instance.InstanceId}");
                    Console.WriteLine($"\tCurrent State: {instance.State.Name}");
                }
            }
        }
    }

    /// <summary>
    /// Get information about EC2 instances filtered by a tag name and value.
    /// </summary>
    /// <param name="tagName">The name of the tag to filter on.</param>
    /// <param name="tagValue">The value of the tag to look for.</param>
    /// <returns>Async task.</returns>
    public async Task GetInstanceDescriptionsFiltered(string tagName, string
tagValue)
    {
        // This tag filters the results of the instance list.
        var filters = new List<Filter>
        {
            new Filter
            {
                Name = $"tag:{tagName}",
                Values = new List<string>
```

```
                {
                    tagValue,
                },
            },
        };
        var request = new DescribeInstancesRequest
        {
            Filters = filters,
        };

        Console.WriteLine("\nShowing instances with tag: \"IncludeInList\" set to
\"Yes\".");
        var paginator = _amazonEC2.Paginators.DescribeInstances(request);

        await foreach (var response in paginator.Responses)
        {
            foreach (var reservation in response.Reservations)
            {
                foreach (var instance in reservation.Instances)
                {
                    Console.Write($"Instance ID: {instance.InstanceId} ");
                    Console.WriteLine($"\tCurrent State: {instance.State.Name}");
                }
            }
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DescribeInstances中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
#############################################################################
# function ec2_describe_instances
```

```
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID of the instance to describe (optional).
#       -q query - The query to filter the response (optional).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_instances() {
  local instance_id query response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_instances"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_id - The ID of the instance to describe (optional)."
    echo "  -q query - The query to filter the response (optional)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:q:h" option; do
    case "${option}" in
      i) instance_id="${OPTARG}" ;;
      q) query="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$instance_id" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--instance-ids" $instance_id)
  fi

  local query_arg=""
  if [[ -n "$query" ]]; then
    query_arg="--query '$query'"
  else
    query_arg="--query Reservations[*].Instances[*].
[InstanceId,ImageId,InstanceType,KeyName,VpcId,PublicIpAddress,State.Name]"
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 describe-instances \
    "${aws_cli_args[@]}" \
    $query_arg \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-instances operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

此範例中使用的公用程式函數。

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}
```

```
##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeInstances中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DescribeInstancesRequest request;
    bool header = false;
    bool done = false;
    while (!done) {
        auto outcome = ec2Client.DescribeInstances(request);
        if (outcome.IsSuccess()) {
            if (!header) {
                std::cout << std::left <<
                          std::setw(48) << "Name" <<
                          std::setw(20) << "ID" <<
                          std::setw(25) << "Ami" <<
                          std::setw(15) << "Type" <<
                          std::setw(15) << "State" <<
                          std::setw(15) << "Monitoring" << std::endl;
                header = true;
            }

            const std::vector<Aws::EC2::Model::Reservation> &reservations =
                    outcome.GetResult().GetReservations();

            for (const auto &reservation: reservations) {
                const std::vector<Aws::EC2::Model::Instance> &instances =
                        reservation.GetInstances();
                for (const auto &instance: instances) {
                    Aws::String instanceStateString =

Aws::EC2::Model::InstanceStateNameMapper::GetNameForInstanceStateName(
                                    instance.GetState().GetName());

                    Aws::String typeString =
```

```
Aws::EC2::Model::InstanceTypeMapper::GetNameForInstanceType(
                                    instance.GetInstanceType());

                    Aws::String monitorString =

Aws::EC2::Model::MonitoringStateMapper::GetNameForMonitoringState(
                                    instance.GetMonitoring().GetState());
                    Aws::String name = "Unknown";

                    const std::vector<Aws::EC2::Model::Tag> &tags =
instance.GetTags();
                    auto nameIter = std::find_if(tags.cbegin(), tags.cend(),
                                                 [](const Aws::EC2::Model::Tag
&tag) {
                                                         return tag.GetKey() ==
"Name";
                                                 });
                    if (nameIter != tags.cend()) {
                        name = nameIter->GetValue();
                    }
                    std::cout <<
                              std::setw(48) << name <<
                              std::setw(20) << instance.GetInstanceId() <<
                              std::setw(25) << instance.GetImageId() <<
                              std::setw(15) << typeString <<
                              std::setw(15) << instanceStateString <<
                              std::setw(15) << monitorString << std::endl;
                }
            }

            if (!outcome.GetResult().GetNextToken().empty()) {
                request.SetNextToken(outcome.GetResult().GetNextToken());
            }
            else {
                done = true;
            }
        }
        else {
            std::cerr << "Failed to describe EC2 instances:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考DescribeInstances中的。

CLI

AWS CLI

範例 1：描述執行個體

下列 describe-instances 範例會描述指定的執行個體。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0
```

輸出：

```
{
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-0abcdef1234567890",
                    "InstanceId": "i-1234567890abcdef0",
                    "InstanceType": "t3.nano",
                    "KeyName": "my-key-pair",
                    "LaunchTime": "2022-11-15T10:48:59+00:00",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-2a",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
                    "PrivateIpAddress": "10-0-0-157",
                    "ProductCodes": [],
                    "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
                    "PublicIpAddress": "34.253.223.13",
```

```
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
                    "StateTransitionReason": "",
                    "SubnetId": "subnet-04a636d18e83cfacb",
                    "VpcId": "vpc-1234567890abcdef0",
                    "Architecture": "x86_64",
                    "BlockDeviceMappings": [
                        {
                            "DeviceName": "/dev/xvda",
                            "Ebs": {
                                "AttachTime": "2022-11-15T10:49:00+00:00",
                                "DeleteOnTermination": true,
                                "Status": "attached",
                                "VolumeId": "vol-02e6ccdca7de29cf2"
                            }
                        }
                    ],
                    "ClientToken": "1234abcd-1234-abcd-1234-d46a8903e9bc",
                    "EbsOptimized": true,
                    "EnaSupport": true,
                    "Hypervisor": "xen",
                    "IamInstanceProfile": {
                        "Arn": "arn:aws:iam::111111111111:instance-profile/
AmazonSSMRoleForInstancesQuickSetup",
                        "Id": "11111111111111111111"
                    },
                    "NetworkInterfaces": [
                        {
                            "Association": {
                                "IpOwnerId": "amazon",
                                "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
                                "PublicIp": "34.253.223.13"
                            },
                            "Attachment": {
                                "AttachTime": "2022-11-15T10:48:59+00:00",
                                "AttachmentId": "eni-attach-1234567890abcdefg",
                                "DeleteOnTermination": true,
                                "DeviceIndex": 0,
                                "Status": "attached",
                                "NetworkCardIndex": 0
                            },
```

```
                            "Description": "",
                            "Groups": [
                                {
                                    "GroupName": "launch-wizard-146",
                                    "GroupId": "sg-1234567890abcdefg"
                                }
                            ],
                            "Ipv6Addresses": [],
                            "MacAddress": "00:11:22:33:44:55",
                            "NetworkInterfaceId": "eni-1234567890abcdefg",
                            "OwnerId": "104024344472",
                            "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
                            "PrivateIpAddress": "10-0-0-157",
                            "PrivateIpAddresses": [
                                {
                                    "Association": {
                                        "IpOwnerId": "amazon",
                                        "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
                                        "PublicIp": "34.253.223.13"
                                    },
                                    "Primary": true,
                                    "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
                                    "PrivateIpAddress": "10-0-0-157"
                                }
                            ],
                            "SourceDestCheck": true,
                            "Status": "in-use",
                            "SubnetId": "subnet-1234567890abcdefg",
                            "VpcId": "vpc-1234567890abcdefg",
                            "InterfaceType": "interface"
                        }
                    ],
                    "RootDeviceName": "/dev/xvda",
                    "RootDeviceType": "ebs",
                    "SecurityGroups": [
                        {
                            "GroupName": "launch-wizard-146",
                            "GroupId": "sg-1234567890abcdefg"
                        }
                    ],
                    "SourceDestCheck": true,
```

```
                    "Tags": [
                        {
                            "Key": "Name",
                            "Value": "my-instance"
                        }
                    ],
                    "VirtualizationType": "hvm",
                    "CpuOptions": {
                        "CoreCount": 1,
                        "ThreadsPerCore": 2
                    },
                    "CapacityReservationSpecification": {
                        "CapacityReservationPreference": "open"
                    },
                    "HibernationOptions": {
                        "Configured": false
                    },
                    "MetadataOptions": {
                        "State": "applied",
                        "HttpTokens": "optional",
                        "HttpPutResponseHopLimit": 1,
                        "HttpEndpoint": "enabled",
                        "HttpProtocolIpv6": "disabled",
                        "InstanceMetadataTags": "enabled"
                    },
                    "EnclaveOptions": {
                        "Enabled": false
                    },
                    "PlatformDetails": "Linux/UNIX",
                    "UsageOperation": "RunInstances",
                    "UsageOperationUpdateTime": "2022-11-15T10:48:59+00:00",
                    "PrivateDnsNameOptions": {
                        "HostnameType": "ip-name",
                        "EnableResourceNameDnsARecord": true,
                        "EnableResourceNameDnsAAAARecord": false
                    },
                    "MaintenanceOptions": {
                        "AutoRecovery": "default"
                    }
                }
            ],
            "OwnerId": "111111111111",
            "ReservationId": "r-1234567890abcdefg"
        }
```

```
    ]
}
```

範例 2：篩選具有指定類型的執行個體

下列 describe-instances 範例會使用篩選條件，將結果範圍限定為指定類型的執行個體。

```
aws ec2 describe-instances \
    --filters Name=instance-type,Values=m5.large
```

如需輸出範例，請參閱範例 1。

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的使用 CLI 列出和篩選。

範例 3：篩選具有指定類型和可用區域的執行個體

下列 describe-instances 範例會使用多個篩選條件，將結果範圍限定為指定可用區域中具有指定類型的執行個體。

```
aws ec2 describe-instances \
    --filters Name=instance-type,Values=t2.micro,t3.micro Name=availability-
zone,Values=us-east-2c
```

如需輸出範例，請參閱範例 1。

範例 4：使用 JSON 檔案篩選具有指定類型和可用區域的執行個體

下列 describe-instances 範例會使用 JSON 輸入檔案來執行與先前範例相同的篩選條件。若篩選條件變得更複雜，便可更輕鬆地在 JSON 檔案中指定這些條件。

```
aws ec2 describe-instances \
    --filters file://filters.json
```

filters.json 的內容：

```
[
    {
        "Name": "instance-type",
        "Values": ["t2.micro", "t3.micro"]
    },
    {
```

```
        "Name": "availability-zone",
        "Values": ["us-east-2c"]
    }
]
```

如需輸出範例，請參閱範例 1。

範例 5：篩選具有指定 Owner 標籤的執行個體

下列 describe-instances 範例會使用標籤篩選條件，將結果範圍限定為其標籤具有指定標籤索引鍵 (Owner) 的執行個體，不論標籤值為何。

```
aws ec2 describe-instances \
    --filters "Name=tag-key,Values=Owner"
```

如需輸出範例，請參閱範例 1。

範例 6：篩選具有指定 my-team 標籤值的執行個體

下列 describe-instances 範例會使用標籤篩選條件，將結果範圍限定為其標籤具有指定標籤值 (my-team) 的執行個體，不論標籤索引鍵為何。

```
aws ec2 describe-instances \
    --filters "Name=tag-value,Values=my-team"
```

如需輸出範例，請參閱範例 1。

範例 7：篩選具有指定 Owner 標籤和 my-team 值的執行個體

下列 describe-instances 範例會使用標籤篩選條件，將結果範圍限定為具有指定標籤 (Owner=my-team) 的執行個體。

```
aws ec2 describe-instances \
    --filters "Name=tag:Owner,Values=my-team"
```

如需輸出範例，請參閱範例 1。

範例 8：僅顯示所有執行個體的執行個體和子網路 ID

下列 describe-instances 範例會使用 --query 參數，以 JSON 格式僅顯示所有執行個體的執行個體和子網路 ID。

Linux 和 macOS：

```
aws ec2 describe-instances \
    --query 'Reservations[*].Instances[*].{Instance:InstanceId,Subnet:SubnetId}'
 \
    --output json
```

Windows：

```
aws ec2 describe-instances ^
    --query "Reservations[*].Instances[*].{Instance:InstanceId,Subnet:SubnetId}"
 ^
    --output json
```

輸出：

```
[
    {
        "Instance": "i-057750d42936e468a",
        "Subnet": "subnet-069beee9b12030077"
    },
    {
        "Instance": "i-001efd250faaa6ffa",
        "Subnet": "subnet-0b715c6b7db68927a"
    },
    {
        "Instance": "i-027552a73f021f3bd",
        "Subnet": "subnet-0250c25a1f4e15235"
    }
    ...
]
```

範例 9：篩選指定類型的執行個體，並僅顯示其執行個體 ID

下列 `describe-instances` 範例會使用篩選條件，將結果範圍限定為指定類型的執行個體以及 `--query` 參數，以便僅顯示執行個體 ID。

```
aws ec2 describe-instances \
    --filters "Name=instance-type,Values=t2.micro" \
    --query "Reservations[*].Instances[*].[InstanceId]" \
    --output text
```

輸出：

```
i-031c0dc19de2fb70c
i-00d8bff789a736b75
i-0b715c6b7db68927a
i-0626d4edd54f1286d
i-00b8ae04f9f99908e
i-0fc71c25d2374130c
```

範例 10：篩選指定類型的執行個體，並僅顯示其執行個體 ID、可用區域以及指定標籤值

下列 describe-instances 範例會針對其標籤具有名稱 tag-key 的執行個體，以表格格式顯示執行個體 ID、可用區域以及 Name 標籤的值。

Linux 和 macOS：

```
aws ec2 describe-instances \
    --filters Name=tag-key,Values=Name \
    --query 'Reservations[*].Instances[*].
{Instance:InstanceId,AZ:Placement.AvailabilityZone,Name:Tags[?Key==`Name`]|
[0].Value}' \
    --output table
```

Windows：

```
aws ec2 describe-instances ^
    --filters Name=tag-key,Values=Name ^
    --query "Reservations[*].Instances[*].
{Instance:InstanceId,AZ:Placement.AvailabilityZone,Name:Tags[?Key=='Name']|
[0].Value}" ^
    --output table
```

輸出：

```
-------------------------------------------------------------
|                    DescribeInstances                      |
+--------------+-----------------------+--------------------+
|      AZ      |       Instance        |        Name        |
+--------------+-----------------------+--------------------+
|  us-east-2b  |  i-057750d42936e468a  |  my-prod-server    |
|  us-east-2a  |  i-001efd250faaa6ffa  |  test-server-1     |
```

```
|  us-east-2a  |  i-027552a73f021f3bd  |  test-server-2     |
+--------------+-----------------------+--------------------+
```

範例 11：描述分區放置群組中的執行個體

下列 describe-instances 範例會描述指定的執行個體。輸出包含執行個體的放置資料，其中包括執行個體的放置群組名稱和分區號碼。

```
aws ec2 describe-instances \
    --instance-ids i-0123a456700123456 \
    --query "Reservations[*].Instances[*].Placement"
```

輸出：

```
[
    [
        {
            "AvailabilityZone": "us-east-1c",
            "GroupName": "HDFS-Group-A",
            "PartitionNumber": 3,
            "Tenancy": "default"
        }

    ]
]
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的描述放置群組中的執行個體。

範例 12：篩選為具有指定放置群組和分區號碼的執行個體

下列 describe-instances 範例會將結果篩選為僅顯示具有指定放置群組和分割區號碼的執行個體。

```
aws ec2 describe-instances \
    --filters "Name=placement-group-name,Values=HDFS-Group-A" "Name=placement-
partition-number,Values=7"
```

以下內容僅顯示輸出的相關資訊。

```
"Instances": [
```

```
    {
        "InstanceId": "i-0123a456700123456",
        "InstanceType": "r4.large",
        "Placement": {
            "AvailabilityZone": "us-east-1c",
            "GroupName": "HDFS-Group-A",
            "PartitionNumber": 7,
            "Tenancy": "default"
        }
    },
    {
        "InstanceId": "i-9876a543210987654",
        "InstanceType": "r4.large",
        "Placement": {
            "AvailabilityZone": "us-east-1c",
            "GroupName": "HDFS-Group-A",
            "PartitionNumber": 7,
            "Tenancy": "default"
        }
    ],
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的描述放置群組中的執行個體。

範例 13：篩選出設定為允許從執行個體中繼資料存取標籤的執行個體

下列 describe-instances 範例會將結果篩選為僅顯示設定為允許從執行個體中繼資料存取執行個體標籤的執行個體。

```
aws ec2 describe-instances \
    --filters "Name=metadata-options.instance-metadata-tags,Values=enabled" \
    --query "Reservations[*].Instances[*].InstanceId" \
    --output text
```

以下內容顯示預期的輸出。

```
i-1234567890abcdefg
i-abcdefg1234567890
i-11111111aaaaaaaaa
i-aaaaaaaa111111111
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的使用執行個體中繼資料中的執行個體標籤。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeInstances中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ec2.Ec2Client;
import software.amazon.awssdk.services.ec2.model.DescribeInstancesRequest;
import software.amazon.awssdk.services.ec2.model.Ec2Exception;
import software.amazon.awssdk.services.ec2.paginators.DescribeInstancesIterable;

/**
 * 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 DescribeInstances {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        Ec2Client ec2 = Ec2Client.builder()
            .region(region)
            .build();

        describeEC2Instances(ec2);
        ec2.close();
    }

    public static void describeEC2Instances(Ec2Client ec2) {
        try {
            DescribeInstancesRequest request = DescribeInstancesRequest.builder()
                .maxResults(10)
```

```
                .build();

            DescribeInstancesIterable instancesIterable =
 ec2.describeInstancesPaginator(request);
            instancesIterable.stream()
                .flatMap(r -> r.reservations().stream())
                .flatMap(reservation -> reservation.instances().stream())
                .forEach(instance -> {
                    System.out.println("Instance Id is " +
 instance.instanceId());
                    System.out.println("Image id is " + instance.imageId());
                    System.out.println("Instance type is " +
 instance.instanceType());
                    System.out.println("Instance state name is " +
 instance.state().name());
                    System.out.println("Monitoring information is " +
 instance.monitoring().state());
                });

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorCode());
            System.exit(1);
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考DescribeInstances中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { DescribeInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";
```

```
// List all of your EC2 instances running with x86_64 architecture that were
// launched this month.
export const main = async () => {
  const d = new Date();
  const year = d.getFullYear();
  const month = `0${d.getMonth() + 1}`.slice(-2);
  const launchTimePattern = `${year}-${month}-*`;
  const command = new DescribeInstancesCommand({
    Filters: [
      { Name: "architecture", Values: ["x86_64"] },
      { Name: "instance-state-name", Values: ["running"] },
      {
        Name: "launch-time",
        Values: [launchTimePattern],
      },
    ],
  });

  try {
    const { Reservations } = await client.send(command);
    const instanceList = Reservations.reduce((prev, current) => {
      return prev.concat(current.Instances);
    }, []);

    console.log(instanceList);
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DescribeInstances中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun describeEC2Instances() {
    val request =
        DescribeInstancesRequest {
            maxResults = 6
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeInstances(request)
        response.reservations?.forEach { reservation ->
            reservation.instances?.forEach { instance ->
                println("Instance Id is ${instance.instanceId}")
                println("Image id is ${instance.imageId}")
                println("Instance type is ${instance.instanceType}")
                println("Instance state name is ${instance.state?.name}")
                println("monitoring information is
 ${instance.monitoring?.state}")
            }
        }
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK DescribeInstances中的 Kotlin API 參考。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的執行個體。

```
(Get-EC2Instance -InstanceId i-12345678).Instances
```

輸出：

```
AmiLaunchIndex        : 0
Architecture          : x86_64
BlockDeviceMappings   : {/dev/sda1}
ClientToken           : TleEy1448154045270
EbsOptimized          : False
Hypervisor            : xen
IamInstanceProfile    : Amazon.EC2.Model.IamInstanceProfile
ImageId               : ami-12345678
```

```
InstanceId            : i-12345678
InstanceLifecycle     :
InstanceType          : t2.micro
KernelId              :
KeyName               : my-key-pair
LaunchTime            : 12/4/2015 4:44:40 PM
Monitoring            : Amazon.EC2.Model.Monitoring
NetworkInterfaces     : {ip-10-0-2-172.us-west-2.compute.internal}
Placement             : Amazon.EC2.Model.Placement
Platform              : Windows
PrivateDnsName        : ip-10-0-2-172.us-west-2.compute.internal
PrivateIpAddress      : 10.0.2.172
ProductCodes          : {}
PublicDnsName         :
PublicIpAddress       :
RamdiskId             :
RootDeviceName        : /dev/sda1
RootDeviceType        : ebs
SecurityGroups        : {default}
SourceDestCheck       : True
SpotInstanceRequestId :
SriovNetSupport       :
State                 : Amazon.EC2.Model.InstanceState
StateReason           :
StateTransitionReason :
SubnetId              : subnet-12345678
Tags                  : {Name}
VirtualizationType    : hvm
VpcId                 : vpc-12345678
```

範例 2：此範例說明目前區域中的所有執行個體，並依保留分組。若要查看執行個體詳細資料，請展開每個保留物件中的「執行個體

```
Get-EC2Instance
```

輸出：

```
GroupNames    : {}
Groups        : {}
Instances     : {}
OwnerId       : 123456789012
RequesterId   : 226008221399
ReservationId : r-c5df370c
```

```
GroupNames    : {}
Groups        : {}
Instances     : {}
OwnerId       : 123456789012
RequesterId   : 854251627541
ReservationId : r-63e65bab
...
```

範例 3：此範例說明如何使用篩選器查詢 VPC 特定子網路中的 EC2 執行個體。

```
(Get-EC2Instance -Filter @{Name="vpc-id";Values="vpc-1a2bc34d"},@{Name="subnet-
id";Values="subnet-1a2b3c4d"}).Instances
```

輸出：

```
InstanceId          InstanceType Platform PrivateIpAddress PublicIpAddress
 SecurityGroups SubnetId        VpcId
----------          ------------ -------- ---------------- --------------
 -------------- --------        -----
i-01af...82cf180e19 t2.medium    Windows  10.0.0.98                      ...
         subnet-1a2b3c4d vpc-1a2b3c4d
i-0374...7e9d5b0c45 t2.xlarge    Windows  10.0.0.53                      ...
         subnet-1a2b3c4d vpc-1a2b3c4d
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeInstances式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class InstanceWrapper:
```

```
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
that
                            wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def display(self, indent=1):
        """
        Displays information about an instance.

        :param indent: The visual indent to apply to the output.
        """
        if self.instance is None:
            logger.info("No instance to display.")
            return

        try:
            self.instance.load()
            ind = "\t" * indent
            print(f"{ind}ID: {self.instance.id}")
            print(f"{ind}Image ID: {self.instance.image_id}")
            print(f"{ind}Instance type: {self.instance.instance_type}")
            print(f"{ind}Key name: {self.instance.key_name}")
            print(f"{ind}VPC ID: {self.instance.vpc_id}")
            print(f"{ind}Public IP: {self.instance.public_ip_address}")
            print(f"{ind}State: {self.instance.state['Name']}")
        except ClientError as err:
            logger.error(
```

```
                "Couldn't display your instance. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeInstances中的 Python (博托 3) API 參考。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
require "aws-sdk-ec2"

# @param ec2_resource [Aws::EC2::Resource] An initialized EC2 resource object.
# @example
#   list_instance_ids_states(Aws::EC2::Resource.new(region: 'us-west-2'))
def list_instance_ids_states(ec2_resource)
  response = ec2_resource.instances
  if response.count.zero?
    puts "No instances found."
  else
    puts "Instances -- ID, state:"
    response.each do |instance|
      puts "#{instance.id}, #{instance.state.name}"
    end
  end
rescue StandardError => e
  puts "Error getting information about instances: #{e.message}"
end
```

```
# Example usage:
def run_me
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-get-all-instance-info.rb REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-get-all-instance-info.rb us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    region = ARGV[0]
  end
  ec2_resource = Aws::EC2::Resource.new(region: region)
  list_instance_ids_states(ec2_resource)
end

run_me if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考DescribeInstances中的。

Rust

適用於 Rust 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
async fn show_state(client: &Client, ids: Option<Vec<String>>) -> Result<(),
 Error> {
    let resp = client
        .describe_instances()
        .set_instance_ids(ids)
        .send()
```

```
        .await?;

    for reservation in resp.reservations() {
        for instance in reservation.instances() {
            println!("Instance ID: {}", instance.instance_id().unwrap());
            println!(
                "State:       {:?}",
                instance.state().unwrap().name().unwrap()
            );
            println!();
        }
    }

    Ok(())
}
```

- 如需 API 的詳細資訊，請參閱 AWS SDK DescribeInstances中的 Rust API 參考資料。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        oo_result = lo_ec2->describeinstances( ) .                      "
oo_result is returned for testing purposes. "

        " Retrieving details of EC2 instances. "
        DATA: lv_istance_id    TYPE /aws1/ec2string,
              lv_status        TYPE /aws1/ec2instancestatename,
              lv_instance_type TYPE /aws1/ec2instancetype,
              lv_image_id      TYPE /aws1/ec2string.
        LOOP AT oo_result->get_reservations( ) INTO DATA(lo_reservation).
          LOOP AT lo_reservation->get_instances( ) INTO DATA(lo_instance).
            lv_istance_id = lo_instance->get_instanceid( ).
            lv_status =  lo_instance->get_state( )->get_name( ).
```

```
            lv_instance_type = lo_instance->get_instancetype( ).
            lv_image_id = lo_instance->get_imageid( ).
          ENDLOOP.
        ENDLOOP.
        MESSAGE 'Retrieved information about EC2 instances.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK DescribeInstances中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**DescribeInternetGateways**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeInternetGateways。

CLI

AWS CLI

描述網際網路閘道

下列describe-internet-gateways範例說明指定的網際網路閘道。

```
aws ec2 describe-internet-gateways \
    --internet-gateway-ids igw-0d0fb496b3EXAMPLE
```

輸出：

```
{
    "InternetGateways": [
        {
            "Attachments": [
                {
                    "State": "available",
                    "VpcId": "vpc-0a60eb65b4EXAMPLE"
                }
```

```
            ],
            "InternetGatewayId": "igw-0d0fb496b3EXAMPLE",
            "OwnerId": "123456789012",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "my-igw"
                }
            ]
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的網際網路閘道。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeInternetGateways中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的網際網路閘道。

```
Get-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

輸出：

```
Attachments       InternetGatewayId     Tags
-----------       -----------------     ----
{vpc-1a2b3c4d}    igw-1a2b3c4d          {}
```

範例 2：此範例說明您所有的網際網路閘道。

```
Get-EC2InternetGateway
```

輸出：

```
Attachments       InternetGatewayId     Tags
-----------       -----------------     ----
{vpc-1a2b3c4d}    igw-1a2b3c4d          {}
{}                igw-2a3b4c5d          {}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeInternetGateways式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeKeyPairs配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeKeyPairs。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
/// <summary>
/// Get information about an Amazon EC2 key pair.
/// </summary>
/// <param name="keyPairName">The name of the key pair.</param>
/// <returns>A list of key pair information.</returns>
public async Task<List<KeyPairInfo>> DescribeKeyPairs(string keyPairName)
{
    var request = new DescribeKeyPairsRequest();
    if (!string.IsNullOrEmpty(keyPairName))
    {
        request = new DescribeKeyPairsRequest
        {
            KeyNames = new List<string> { keyPairName }
        };
    }
```

```
        var response = await _amazonEC2.DescribeKeyPairsAsync(request);
        return response.KeyPairs.ToList();
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DescribeKeyPairs中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
###############################################################################
# function ec2_describe_key_pairs
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 key pairs.
#
# Parameters:
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_key_pairs() {
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_key_pairs"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
 pairs."
    echo "  -h - Display help."
    echo ""
  }
```

```
  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws ec2 describe-key-pairs \
    --query 'KeyPairs[*].[KeyName, KeyFingerprint]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-key-pairs operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

此範例中使用的公用程式函數。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}
```

```
##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeKeyPairs中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);
Aws::EC2::Model::DescribeKeyPairsRequest request;

auto outcome = ec2Client.DescribeKeyPairs(request);
if (outcome.IsSuccess()) {
    std::cout << std::left <<
              std::setw(32) << "Name" <<
              std::setw(64) << "Fingerprint" << std::endl;

    const std::vector<Aws::EC2::Model::KeyPairInfo> &key_pairs =
            outcome.GetResult().GetKeyPairs();
    for (const auto &key_pair: key_pairs) {
        std::cout << std::left <<
                  std::setw(32) << key_pair.GetKeyName() <<
                  std::setw(64) << key_pair.GetKeyFingerprint() << std::endl;
    }
}
else {
    std::cerr << "Failed to describe key pairs:" <<
              outcome.GetError().GetMessage() << std::endl;
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考DescribeKeyPairs中的。

CLI

AWS CLI

顯示金鑰對

下列 `describe-key-pairs` 範例顯示指定金鑰對的相關資訊。

```
aws ec2 describe-key-pairs \
    --key-names my-key-pair
```

輸出：

```
{
    "KeyPairs": [
        {
            "KeyPairId": "key-0b94643da6EXAMPLE",
            "KeyFingerprint":
 "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f",
            "KeyName": "my-key-pair",
            "KeyType": "rsa",
            "Tags": [],
            "CreateTime": "2022-05-27T21:51:16.000Z"
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的描述公有金鑰。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeKeyPairs中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    public static void describeKeys(Ec2Client ec2) {
        try {
            DescribeKeyPairsResponse response = ec2.describeKeyPairs();
            response.keyPairs().forEach(keyPair -> System.out.printf(
                    "Found key pair with name %s " +
                            "and fingerprint %s",
                    keyPair.keyName(),
                    keyPair.keyFingerprint()));
```

```
        } catch (Ec2Exception 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DescribeKeyPairs中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { DescribeKeyPairsCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new DescribeKeyPairsCommand({});

  try {
    const { KeyPairs } = await client.send(command);
    const keyPairList = KeyPairs.map(
      (kp) => ` • ${kp.KeyPairId}: ${kp.KeyName}`,
    ).join("\n");
    console.log("The following key pairs were found in your account:");
    console.log(keyPairList);
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DescribeKeyPairs中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun describeEC2Keys() {
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeKeyPairs(DescribeKeyPairsRequest {})
        response.keyPairs?.forEach { keyPair ->
            println("Found key pair with name ${keyPair.keyName} and fingerprint
 ${ keyPair.keyFingerprint}")
        }
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK DescribeKeyPairs中的 Kotlin API 參考。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的 key pair。

```
Get-EC2KeyPair -KeyName my-key-pair
```

輸出：

```
KeyFingerprint                                                  KeyName
--------------                                                  -------
1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f my-key-pair
```

範例 2：此範例說明您所有的金鑰配對。

```
Get-EC2KeyPair
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeKeyPairs式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class KeyPairWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair
 actions."""

    def __init__(self, ec2_resource, key_file_dir, key_pair=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param key_file_dir: The folder where the private key information is
 stored.
                             This should be a secure folder.
        :param key_pair: A Boto3 KeyPair object. This is a high-level object that
                         wraps key pair actions.
        """
        self.ec2_resource = ec2_resource
        self.key_pair = key_pair
        self.key_file_path = None
        self.key_file_dir = key_file_dir

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource, tempfile.TemporaryDirectory())


    def list(self, limit):
```

```
        """
        Displays a list of key pairs for the current account.

        :param limit: The maximum number of key pairs to list.
        """
        try:
            for kp in self.ec2_resource.key_pairs.limit(limit):
                print(f"Found {kp.key_type} key {kp.name} with fingerprint:")
                print(f"\t{kp.key_fingerprint}")
        except ClientError as err:
            logger.error(
                "Couldn't list key pairs. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeKeyPairs中的 Python (博托 3) API 參考。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        oo_result = lo_ec2->describekeypairs( ) .                       "
 oo_result is returned for testing purposes. "
        DATA(lt_key_pairs) = oo_result->get_keypairs( ).
        MESSAGE 'Retrieved information about key pairs.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
```

```
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK DescribeKeyPairs中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeNetworkAcls配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeNetworkAcls。

CLI

AWS CLI

描述您的網路 ACL

下列describe-network-acls範例會擷取有關網路 ACL 的詳細資料。

```
aws ec2 describe-network-acls
```

輸出：

```
{
    "NetworkAcls": [
        {
            "Associations": [
                {
                    "NetworkAclAssociationId": "aclassoc-0c1679dc41EXAMPLE",
                    "NetworkAclId": "acl-0ea1f54ca7EXAMPLE",
                    "SubnetId": "subnet-0931fc2fa5EXAMPLE"
                }
            ],
            "Entries": [
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
```

```
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32767
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": false,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": false,
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32767
                }
            ],
            "IsDefault": true,
            "NetworkAclId": "acl-0ea1f54ca7EXAMPLE",
            "Tags": [],
            "VpcId": "vpc-06e4ab6c6cEXAMPLE",
            "OwnerId": "111122223333"
        },
        {
            "Associations": [],
            "Entries": [
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
                {
                    "Egress": true,
                    "Ipv6CidrBlock": "::/0",
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 101
```

```
            },
            {
                "CidrBlock": "0.0.0.0/0",
                "Egress": true,
                "Protocol": "-1",
                "RuleAction": "deny",
                "RuleNumber": 32767
            },
            {
                "Egress": true,
                "Ipv6CidrBlock": "::/0",
                "Protocol": "-1",
                "RuleAction": "deny",
                "RuleNumber": 32768
            },
            {
                "CidrBlock": "0.0.0.0/0",
                "Egress": false,
                "Protocol": "-1",
                "RuleAction": "allow",
                "RuleNumber": 100
            },
            {
                "Egress": false,
                "Ipv6CidrBlock": "::/0",
                "Protocol": "-1",
                "RuleAction": "allow",
                "RuleNumber": 101
            },
            {
                "CidrBlock": "0.0.0.0/0",
                "Egress": false,
                "Protocol": "-1",
                "RuleAction": "deny",
                "RuleNumber": 32767
            },
            {
                "Egress": false,
                "Ipv6CidrBlock": "::/0",
                "Protocol": "-1",
                "RuleAction": "deny",
                "RuleNumber": 32768
            }
        ],
```

```
            "IsDefault": true,
            "NetworkAclId": "acl-0e2a78e4e2EXAMPLE",
            "Tags": [],
            "VpcId": "vpc-03914afb3eEXAMPLE",
            "OwnerId": "111122223333"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS VPC 使用者指南》中的〈網路 ACL〉。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeNetworkAcls中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的網路 ACL。

```
Get-EC2NetworkAcl -NetworkAclId acl-12345678
```

輸出：

```
Associations : {aclassoc-1a2b3c4d}
Entries      : {Amazon.EC2.Model.NetworkAclEntry,
 Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
Tags         : {Name}
VpcId        : vpc-12345678
```

範例 2：此範例說明指定網路 ACL 的規則。

```
(Get-EC2NetworkAcl -NetworkAclId acl-12345678).Entries
```

輸出：

```
CidrBlock    : 0.0.0.0/0
Egress       : True
IcmpTypeCode :
PortRange    :
Protocol     : -1
```

```
RuleAction   : deny
RuleNumber   : 32767

CidrBlock    : 0.0.0.0/0
Egress       : False
IcmpTypeCode :
PortRange    :
Protocol     : -1
RuleAction   : deny
RuleNumber   : 32767
```

範例 3：此範例說明您所有的網路 ACL。

```
Get-EC2NetworkAcl
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeNetworkAcls式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeNetworkInterfaceAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeNetworkInterfaceAttribute。

CLI

AWS CLI

描述網路介面的附件屬性

此範例命令描述指定之網路介面的attachment屬性。

命令：

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
 --attribute attachment
```

輸出：

```
{
  "NetworkInterfaceId": "eni-686ea200",
```

```
  "Attachment": {
      "Status": "attached",
      "DeviceIndex": 0,
      "AttachTime": "2015-05-21T20:02:20.000Z",
      "InstanceId": "i-1234567890abcdef0",
      "DeleteOnTermination": true,
      "AttachmentId": "eni-attach-43348162",
      "InstanceOwnerId": "123456789012"
  }
}
```

描述網路介面的描述屬性

此範例命令描述指定之網路介面的description屬性。

命令：

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
 --attribute description
```

輸出：

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "Description": {
      "Value": "My description"
  }
}
```

描述網路介面的 GroupSet 屬性

此範例命令描述指定之網路介面的groupSet屬性。

命令：

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
 --attribute groupSet
```

輸出：

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "Groups": [
```

```
      {
          "GroupName": "my-security-group",
          "GroupId": "sg-903004f8"
      }
  ]
}
```

描述網路介面的 sourceDestCheck 屬性

此範例命令描述指定之網路介面的sourceDestCheck屬性。

命令：

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
 --attribute sourceDestCheck
```

輸出：

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "SourceDestCheck": {
      "Value": true
  }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeNetworkInterfaceAttribute中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的網路介面。

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute
 Attachment
```

輸出：

```
Attachment         : Amazon.EC2.Model.NetworkInterfaceAttachment
```

範例 2：此範例說明指定的網路介面。

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute
 Description
```

輸出：

```
Description        : My description
```

範例 3：此範例說明指定的網路介面。

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute
 GroupSet
```

輸出：

```
Groups             : {my-security-group}
```

範例 4：此範例說明指定的網路介面。

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute
 SourceDestCheck
```

輸出：

```
SourceDestCheck    : True
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeNetworkInterfaceAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeNetworkInterfaces配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeNetworkInterfaces。

CLI

AWS CLI

描述您的網路介面

此範例說明您所有的網路介面。

命令：

```
aws ec2 describe-network-interfaces
```

輸出：

```
{
  "NetworkInterfaces": [
      {
          "Status": "in-use",
          "MacAddress": "02:2f:8f:b0:cf:75",
          "SourceDestCheck": true,
          "VpcId": "vpc-a01106c2",
          "Description": "my network interface",
          "Association": {
              "PublicIp": "203.0.113.12",
              "AssociationId": "eipassoc-0fbb766a",
              "PublicDnsName": "ec2-203-0-113-12.compute-1.amazonaws.com",
              "IpOwnerId": "123456789012"
          },
          "NetworkInterfaceId": "eni-e5aa89a3",
          "PrivateIpAddresses": [
              {
                  "PrivateDnsName": "ip-10-0-1-17.ec2.internal",
                  "Association": {
                      "PublicIp": "203.0.113.12",
                      "AssociationId": "eipassoc-0fbb766a",
                      "PublicDnsName":
 "ec2-203-0-113-12.compute-1.amazonaws.com",
                      "IpOwnerId": "123456789012"
                  },
                  "Primary": true,
                  "PrivateIpAddress": "10.0.1.17"
              }
          ],
          "RequesterManaged": false,
          "Ipv6Addresses": [],
          "PrivateDnsName": "ip-10-0-1-17.ec2.internal",
          "AvailabilityZone": "us-east-1d",
          "Attachment": {
              "Status": "attached",
```

```
            "DeviceIndex": 1,
            "AttachTime": "2013-11-30T23:36:42.000Z",
            "InstanceId": "i-1234567890abcdef0",
            "DeleteOnTermination": false,
            "AttachmentId": "eni-attach-66c4350a",
            "InstanceOwnerId": "123456789012"
        },
        "Groups": [
            {
                "GroupName": "default",
                "GroupId": "sg-8637d3e3"
            }
        ],
        "SubnetId": "subnet-b61f49f0",
        "OwnerId": "123456789012",
        "TagSet": [],
        "PrivateIpAddress": "10.0.1.17"
    },
    {
        "Status": "in-use",
        "MacAddress": "02:58:f5:ef:4b:06",
        "SourceDestCheck": true,
        "VpcId": "vpc-a01106c2",
        "Description": "Primary network interface",
        "Association": {
            "PublicIp": "198.51.100.0",
            "IpOwnerId": "amazon"
        },
        "NetworkInterfaceId": "eni-f9ba99bf",
        "PrivateIpAddresses": [
            {
                "Association": {
                    "PublicIp": "198.51.100.0",
                    "IpOwnerId": "amazon"
                },
                "Primary": true,
                "PrivateIpAddress": "10.0.1.149"
            }
        ],
        "RequesterManaged": false,
        "Ipv6Addresses": [],
        "AvailabilityZone": "us-east-1d",
        "Attachment": {
            "Status": "attached",
```

```
                "DeviceIndex": 0,
                "AttachTime": "2013-11-30T23:35:33.000Z",
                "InstanceId": "i-0598c7d356eba48d7",
                "DeleteOnTermination": true,
                "AttachmentId": "eni-attach-1b9db777",
                "InstanceOwnerId": "123456789012"
            },
            "Groups": [
                {
                    "GroupName": "default",
                    "GroupId": "sg-8637d3e3"
                }
            ],
            "SubnetId": "subnet-b61f49f0",
            "OwnerId": "123456789012",
            "TagSet": [],
            "PrivateIpAddress": "10.0.1.149"
        }
    ]
}
```

此範例說明具有含索引鍵Purpose和值之標籤的網路介面Prod。

命令：

```
aws ec2 describe-network-interfaces --filters Name=tag:Purpose,Values=Prod
```

輸出：

```
{
    "NetworkInterfaces": [
        {
            "Status": "available",
            "MacAddress": "12:2c:bd:f9:bf:17",
            "SourceDestCheck": true,
            "VpcId": "vpc-8941ebec",
            "Description": "ProdENI",
            "NetworkInterfaceId": "eni-b9a5ac93",
            "PrivateIpAddresses": [
                {
                    "PrivateDnsName": "ip-10-0-1-55.ec2.internal",
                    "Primary": true,
                    "PrivateIpAddress": "10.0.1.55"
```

```
                },
                {
                    "PrivateDnsName": "ip-10-0-1-117.ec2.internal",
                    "Primary": false,
                    "PrivateIpAddress": "10.0.1.117"
                }
            ],
            "RequesterManaged": false,
            "PrivateDnsName": "ip-10-0-1-55.ec2.internal",
            "AvailabilityZone": "us-east-1d",
            "Ipv6Addresses": [],
            "Groups": [
                {
                    "GroupName": "MySG",
                    "GroupId": "sg-905002f5"
                }
            ],
            "SubnetId": "subnet-31d6c219",
            "OwnerId": "123456789012",
            "TagSet": [
                {
                    "Value": "Prod",
                    "Key": "Purpose"
                }
            ],
            "PrivateIpAddress": "10.0.1.55"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeNetworkInterfaces](#)中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的網路介面。

```
Get-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

輸出：

```
Association        :
Attachment         : Amazon.EC2.Model.NetworkInterfaceAttachment
AvailabilityZone   : us-west-2c
Description        :
Groups             : {my-security-group}
MacAddress         : 0a:e9:a6:19:4c:7f
NetworkInterfaceId : eni-12345678
OwnerId            : 123456789012
PrivateDnsName     : ip-10-0-0-107.us-west-2.compute.internal
PrivateIpAddress   : 10.0.0.107
PrivateIpAddresses : {ip-10-0-0-107.us-west-2.compute.internal}
RequesterId        :
RequesterManaged   : False
SourceDestCheck    : True
Status             : in-use
SubnetId           : subnet-1a2b3c4d
TagSet             : {}
VpcId              : vpc-12345678
```

範例 2：此範例說明您的所有網路介面。

```
Get-EC2NetworkInterface
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeNetworkInterfaces式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**DescribePlacementGroups**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribePlacementGroups。

CLI

AWS CLI

說明放置群組

此範例指令會說明您所有的置放群組。

命令：

```
aws ec2 describe-placement-groups
```

輸出：

```
{
    "PlacementGroups": [
        {
            "GroupName": "my-cluster",
            "State": "available",
            "Strategy": "cluster"
        },
        ...
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribePlacementGroups中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的置放群組。

```
Get-EC2PlacementGroup -GroupName my-placement-group
```

輸出：

```
GroupName             State        Strategy
---------             -----        --------
my-placement-group    available    cluster
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribePlacementGroups式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribePrefixLists配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribePrefixLists。

CLI

AWS CLI

描述前綴列表

此範例會列出該地區的所有可用前置字元清單。

命令：

```
aws ec2 describe-prefix-lists
```

輸出：

```
{
  "PrefixLists": [
    {
      "PrefixListName": "com.amazonaws.us-east-1.s3",
      "Cidrs": [
        "54.231.0.0/17"
      ],
      "PrefixListId": "pl-63a5400a"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribePrefixLists中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會擷取 AWS 服務 區域的前置碼清單格式中可用

```
Get-EC2PrefixList
```

輸出：

```
Cidrs                                          PrefixListId PrefixListName
-----                                          ------------ --------------
{52.94.5.0/24, 52.119.240.0/21, 52.94.24.0/23} pl-6fa54006  com.amazonaws.eu-
west-1.dynamodb
```

```
{52.218.0.0/17, 54.231.128.0/19}                pl-6da54004  com.amazonaws.eu-
west-1.s3
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribePrefixLists式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeRegions配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeRegions。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);

Aws::EC2::Model::DescribeRegionsRequest request;
auto outcome = ec2Client.DescribeRegions(request);
bool result = true;
if (outcome.IsSuccess()) {
    std::cout << std::left <<
              std::setw(32) << "RegionName" <<
              std::setw(64) << "Endpoint" << std::endl;

    const auto &regions = outcome.GetResult().GetRegions();
    for (const auto &region: regions) {
        std::cout << std::left <<
                  std::setw(32) << region.GetRegionName() <<
                  std::setw(64) << region.GetEndpoint() << std::endl;
    }
}
else {
```

```
        std::cerr << "Failed to describe regions:" <<
                  outcome.GetError().GetMessage() << std::endl;
        result = false;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考DescribeRegions中的。

CLI

AWS CLI

範例 1：描述所有已啟用的區域

以下 `describe-regions` 範例說明為您帳戶啟用的所有區域。

```
aws ec2 describe-regions
```

輸出：

```
{
    "Regions": [
        {
            "Endpoint": "ec2.eu-north-1.amazonaws.com",
            "RegionName": "eu-north-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-south-1.amazonaws.com",
            "RegionName": "ap-south-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-3.amazonaws.com",
            "RegionName": "eu-west-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-2.amazonaws.com",
            "RegionName": "eu-west-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
```

```
            "Endpoint": "ec2.eu-west-1.amazonaws.com",
            "RegionName": "eu-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-3.amazonaws.com",
            "RegionName": "ap-northeast-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-2.amazonaws.com",
            "RegionName": "ap-northeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-1.amazonaws.com",
            "RegionName": "ap-northeast-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.sa-east-1.amazonaws.com",
            "RegionName": "sa-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ca-central-1.amazonaws.com",
            "RegionName": "ca-central-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-southeast-1.amazonaws.com",
            "RegionName": "ap-southeast-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-southeast-2.amazonaws.com",
            "RegionName": "ap-southeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-central-1.amazonaws.com",
            "RegionName": "eu-central-1",
            "OptInStatus": "opt-in-not-required"
        },
```

```
        {
            "Endpoint": "ec2.us-east-1.amazonaws.com",
            "RegionName": "us-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-2.amazonaws.com",
            "RegionName": "us-east-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-1.amazonaws.com",
            "RegionName": "us-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-2.amazonaws.com",
            "RegionName": "us-west-2",
            "OptInStatus": "opt-in-not-required"
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的區域 (Region) 和區域 (Zone)。

範例 2：說明使用名稱包含特定字串的端點，且已啟用的區域

下列 `describe-regions` 範例會描述您已啟用，且其端點中具有字串「美國」(us) 的所有區域。

```
aws ec2 describe-regions \
    --filters "Name=endpoint,Values=*us*"
```

輸出：

```
{
    "Regions": [
        {
            "Endpoint": "ec2.us-east-1.amazonaws.com",
            "RegionName": "us-east-1"
        },
        {
            "Endpoint": "ec2.us-east-2.amazonaws.com",
```

```
            "RegionName": "us-east-2"
        },
        {
            "Endpoint": "ec2.us-west-1.amazonaws.com",
            "RegionName": "us-west-1"
        },
        {
            "Endpoint": "ec2.us-west-2.amazonaws.com",
            "RegionName": "us-west-2"
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的區域 (Region) 和區域 (Zone)。

範例 3：描述所有區域

下列 `describe-regions` 範例會描述所有可用的區域，包括已停用的區域。

```
aws ec2 describe-regions \
    --all-regions
```

輸出：

```
{
    "Regions": [
        {
            "Endpoint": "ec2.eu-north-1.amazonaws.com",
            "RegionName": "eu-north-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-south-1.amazonaws.com",
            "RegionName": "ap-south-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-3.amazonaws.com",
            "RegionName": "eu-west-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-2.amazonaws.com",
```

```
            "RegionName": "eu-west-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-1.amazonaws.com",
            "RegionName": "eu-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-3.amazonaws.com",
            "RegionName": "ap-northeast-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.me-south-1.amazonaws.com",
            "RegionName": "me-south-1",
            "OptInStatus": "not-opted-in"
        },
        {
            "Endpoint": "ec2.ap-northeast-2.amazonaws.com",
            "RegionName": "ap-northeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-1.amazonaws.com",
            "RegionName": "ap-northeast-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.sa-east-1.amazonaws.com",
            "RegionName": "sa-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ca-central-1.amazonaws.com",
            "RegionName": "ca-central-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-east-1.amazonaws.com",
            "RegionName": "ap-east-1",
            "OptInStatus": "not-opted-in"
        },
        {
```

```
            "Endpoint": "ec2.ap-southeast-1.amazonaws.com",
            "RegionName": "ap-southeast-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-southeast-2.amazonaws.com",
            "RegionName": "ap-southeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-central-1.amazonaws.com",
            "RegionName": "eu-central-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-1.amazonaws.com",
            "RegionName": "us-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-2.amazonaws.com",
            "RegionName": "us-east-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-1.amazonaws.com",
            "RegionName": "us-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-2.amazonaws.com",
            "RegionName": "us-west-2",
            "OptInStatus": "opt-in-not-required"
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的區域 (Region) 和區域 (Zone)。

範例 4：僅列出區域名稱

下列 `describe-regions` 範例會使用 `--query` 參數來篩選輸出，並以文字形式僅傳回區域 (Region) 的名稱。

```
aws ec2 describe-regions \
    --all-regions \
    --query "Regions[].{Name:RegionName}" \
    --output text
```

輸出：

```
eu-north-1
ap-south-1
eu-west-3
eu-west-2
eu-west-1
ap-northeast-3
ap-northeast-2
me-south-1
ap-northeast-1
sa-east-1
ca-central-1
ap-east-1
ap-southeast-1
ap-southeast-2
eu-central-1
us-east-1
us-east-2
us-west-1
us-west-2
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的區域 (Region) 和區域 (Zone)。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeRegions中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { DescribeRegionsCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new DescribeRegionsCommand({
    // By default this command will not show regions that require you to opt-in.
    // When AllRegions true even the regions that require opt-in will be
 returned.
    AllRegions: true,
    // You can omit the Filters property if you want to get all regions.
    Filters: [
      {
        Name: "region-name",
        // You can specify multiple values for a filter.
        // You can also use '*' as a wildcard. This will return all
        // of the regions that start with `us-east-`.
        Values: ["ap-southeast-4"],
      },
    ],
  });

  try {
    const { Regions } = await client.send(command);
    const regionsList = Regions.map((reg) => ` • ${reg.RegionName}`);
    console.log("Found regions:");
    console.log(regionsList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DescribeRegions中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明可供您使用的區域。

```
Get-EC2Region
```

輸出：

```
Endpoint                            RegionName
--------                            ----------
ec2.eu-west-1.amazonaws.com         eu-west-1
ec2.ap-southeast-1.amazonaws.com    ap-southeast-1
ec2.ap-southeast-2.amazonaws.com    ap-southeast-2
ec2.eu-central-1.amazonaws.com      eu-central-1
ec2.ap-northeast-1.amazonaws.com    ap-northeast-1
ec2.us-east-1.amazonaws.com         us-east-1
ec2.sa-east-1.amazonaws.com         sa-east-1
ec2.us-west-1.amazonaws.com         us-west-1
ec2.us-west-2.amazonaws.com         us-west-2
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeRegions式參考中的。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
require "aws-sdk-ec2"

# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @example
#   list_regions_endpoints(Aws::EC2::Client.new(region: 'us-west-2'))
def list_regions_endpoints(ec2_client)
  result = ec2_client.describe_regions
  # Enable pretty printing.
  max_region_string_length = 16
  max_endpoint_string_length = 33
  # Print header.
  print "Region"
  print " " * (max_region_string_length - "Region".length)
  print "  Endpoint\n"
```

```
  print "-" * max_region_string_length
  print "  "
  print "-" * max_endpoint_string_length
  print "\n"
  # Print Regions and their endpoints.
  result.regions.each do |region|
    print region.region_name
    print " " * (max_region_string_length - region.region_name.length)
    print "  "
    print region.endpoint
    print "\n"
  end
end

# Displays a list of Amazon Elastic Compute Cloud (Amazon EC2)
# Availability Zones available to you depending on the AWS Region
# of the Amazon EC2 client.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @example
#   list_availability_zones(Aws::EC2::Client.new(region: 'us-west-2'))
def list_availability_zones(ec2_client)
  result = ec2_client.describe_availability_zones
  # Enable pretty printing.
  max_region_string_length = 16
  max_zone_string_length = 18
  max_state_string_length = 9
  # Print header.
  print "Region"
  print " " * (max_region_string_length - "Region".length)
  print "  Zone"
  print " " * (max_zone_string_length - "Zone".length)
  print "  State\n"
  print "-" * max_region_string_length
  print "  "
  print "-" * max_zone_string_length
  print "  "
  print "-" * max_state_string_length
  print "\n"
  # Print Regions, Availability Zones, and their states.
  result.availability_zones.each do |zone|
    print zone.region_name
    print " " * (max_region_string_length - zone.region_name.length)
    print "  "
```

```
    print zone.zone_name
    print " " * (max_zone_string_length - zone.zone_name.length)
    print "  "
    print zone.state
    # Print any messages for this Availability Zone.
    if zone.messages.count.positive?
      print "\n"
      puts "  Messages for this zone:"
      zone.messages.each do |message|
        print "    #{message.message}\n"
      end
    end
    print "\n"
  end
end

# Example usage:
def run_me
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-regions-availability-zones.rb REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-regions-availability-zones.rb us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    region = ARGV[0]
  end

  ec2_client = Aws::EC2::Client.new(region: region)

  puts "AWS Regions for Amazon EC2 that are available to you:"
  list_regions_endpoints(ec2_client)
  puts "\n\nAmazon EC2 Availability Zones that are available to you for AWS
 Region '#{region}':"
  list_availability_zones(ec2_client)
end

run_me if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考DescribeRegions中的。

Rust

適用於 Rust 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
async fn show_regions(client: &Client) -> Result<(), Error> {
    let rsp = client.describe_regions().send().await?;

    println!("Regions:");
    for region in rsp.regions() {
        println!("  {}", region.region_name().unwrap());
    }

    Ok(())
}
```

- 如需 API 的詳細資訊，請參閱 AWS SDK DescribeRegions中的 Rust API 參考資料。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
```

```
        oo_result = lo_ec2->describeregions( ) .                        "
 oo_result is returned for testing purposes. "
        DATA(lt_regions) = oo_result->get_regions( ).
        MESSAGE 'Retrieved information about Regions.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK DescribeRegions中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeRouteTables配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeRouteTables。

CLI

AWS CLI

描述您的路由表

以下describe-route-tables示例檢索有關路由表的詳細信息

```
aws ec2 describe-route-tables
```

輸出：

```
{
    "RouteTables": [
        {
            "Associations": [
                {
                    "Main": true,
                    "RouteTableAssociationId": "rtbassoc-0df3f54e06EXAMPLE",
                    "RouteTableId": "rtb-09ba434c1bEXAMPLE"
                }
```

```
            ],
            "PropagatingVgws": [],
            "RouteTableId": "rtb-09ba434c1bEXAMPLE",
            "Routes": [
                {
                    "DestinationCidrBlock": "10.0.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                },
                {
                    "DestinationCidrBlock": "0.0.0.0/0",
                    "NatGatewayId": "nat-06c018cbd8EXAMPLE",
                    "Origin": "CreateRoute",
                    "State": "blackhole"
                }
            ],
            "Tags": [],
            "VpcId": "vpc-0065acced4EXAMPLE",
            "OwnerId": "111122223333"
        },
        {
            "Associations": [
                {
                    "Main": true,
                    "RouteTableAssociationId": "rtbassoc-9EXAMPLE",
                    "RouteTableId": "rtb-a1eec7de"
                }
            ],
            "PropagatingVgws": [],
            "RouteTableId": "rtb-a1eec7de",
            "Routes": [
                {
                    "DestinationCidrBlock": "172.31.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                },
                {
                    "DestinationCidrBlock": "0.0.0.0/0",
                    "GatewayId": "igw-fEXAMPLE",
                    "Origin": "CreateRoute",
                    "State": "active"
                }
```

```
            ],
            "Tags": [],
            "VpcId": "vpc-3EXAMPLE",
            "OwnerId": "111122223333"
        },
        {
            "Associations": [
                {
                    "Main": false,
                    "RouteTableAssociationId": "rtbassoc-0b100c28b2EXAMPLE",
                    "RouteTableId": "rtb-07a98f76e5EXAMPLE",
                    "SubnetId": "subnet-0d3d002af8EXAMPLE"
                }
            ],
            "PropagatingVgws": [],
            "RouteTableId": "rtb-07a98f76e5EXAMPLE",
            "Routes": [
                {
                    "DestinationCidrBlock": "10.0.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                },
                {
                    "DestinationCidrBlock": "0.0.0.0/0",
                    "GatewayId": "igw-06cf664d80EXAMPLE",
                    "Origin": "CreateRoute",
                    "State": "active"
                }
            ],
            "Tags": [],
            "VpcId": "vpc-0065acced4EXAMPLE",
            "OwnerId": "111122223333"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS VPC 使用指南》中的〈使用路由表〉。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeRouteTables中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明您的所有路由表。

```
Get-EC2RouteTable
```

輸出：

```
DestinationCidrBlock    : 10.0.0.0/16
DestinationPrefixListId :
GatewayId               : local
InstanceId              :
InstanceOwnerId         :
NetworkInterfaceId      :
Origin                  : CreateRouteTable
State                   : active
VpcPeeringConnectionId  :

DestinationCidrBlock    : 0.0.0.0/0
DestinationPrefixListId :
GatewayId               : igw-1a2b3c4d
InstanceId              :
InstanceOwnerId         :
NetworkInterfaceId      :
Origin                  : CreateRoute
State                   : active
VpcPeeringConnectionId  :
```

範例 2：此範例會傳回指定路由資料表的詳細資訊。

```
Get-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

範例 3：此範例說明指定 VPC 的路由表。

```
Get-EC2RouteTable -Filter @{ Name="vpc-id"; Values="vpc-1a2b3c4d" }
```

輸出：

```
Associations    : {rtbassoc-12345678}
PropagatingVgws : {}
```

```
Routes          : {, }
RouteTableId    : rtb-1a2b3c4d
Tags            : {}
VpcId           : vpc-1a2b3c4d
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeRouteTables式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**DescribeScheduledInstanceAvailability**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeScheduledInstanceAvailability。

CLI

AWS CLI

描述可用的排程

此範例說明從指定日期開始，每週在星期日發生的排程。

命令：

```
aws ec2 describe-scheduled-instance-availability --recurrence
 Frequency=Weekly,Interval=1,OccurrenceDays=[1] --first-slot-start-time-range
 EarliestTime=2016-01-31T00:00:00Z,LatestTime=2016-01-31T04:00:00Z
```

輸出：

```
{
  "ScheduledInstanceAvailabilitySet": [
    {
        "AvailabilityZone": "us-west-2b",
        "TotalScheduledInstanceHours": 1219,
        "PurchaseToken": "eyJ2IjoiMSIsInMiOjEsImMiOi...",
        "MinTermDurationInDays": 366,
        "AvailableInstanceCount": 20,
        "Recurrence": {
            "OccurrenceDaySet": [
```

```
                1
            ],
            "Interval": 1,
            "Frequency": "Weekly",
            "OccurrenceRelativeToEnd": false
        },
        "Platform": "Linux/UNIX",
        "FirstSlotStartTime": "2016-01-31T00:00:00Z",
        "MaxTermDurationInDays": 366,
        "SlotDurationInHours": 23,
        "NetworkPlatform": "EC2-VPC",
        "InstanceType": "c4.large",
        "HourlyPrice": "0.095"
    },
    ...
  ]
}
```

若要縮小結果範圍，您可以新增指定作業系統、網路和執行個體類型的篩選器。

命令：

-篩選器名稱 = 平台，值 = Linux/UNIX 名稱 = 網路平台，值 =EC2-VPC 名稱 = 執行個體類型，值 =C4. 大

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeScheduledInstanceAvailability中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明從指定日期開始，每週在星期日發生的排程。

```
Get-EC2ScheduledInstanceAvailability -Recurrence_Frequency
 Weekly -Recurrence_Interval 1 -Recurrence_OccurrenceDay 1 -
FirstSlotStartTimeRange_EarliestTime 2016-01-31T00:00:00Z -
FirstSlotStartTimeRange_LatestTime 2016-01-31T04:00:00Z
```

輸出：

```
AvailabilityZone            : us-west-2b
AvailableInstanceCount      : 20
FirstSlotStartTime          : 1/31/2016 8:00:00 AM
```

```
HourlyPrice                 : 0.095
InstanceType                : c4.large
MaxTermDurationInDays       : 366
MinTermDurationInDays       : 366
NetworkPlatform             : EC2-VPC
Platform                    : Linux/UNIX
PurchaseToken               : eyJ2IjoiMSIsInMiOjEsImMiOi...
Recurrence                  : Amazon.EC2.Model.ScheduledInstanceRecurrence
SlotDurationInHours         : 23
TotalScheduledInstanceHours : 1219

...
```

範例 2：若要縮小結果範圍，您可以新增篩選條件，例如作業系統、網路和執行個體類型。

```
-Filter @{ Name="platform";Values="Linux/UNIX" },@{ Name="network-
platform";Values="EC2-VPC" },@{ Name="instance-type";Values="c4.large" }
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程[DescribeScheduledInstanceAvailability](https://docs.aws.amazon.com/powershell/latest/reference)式參考中的。

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 開發套件建立 Amazon EC2 資源](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/developing-ec2.html)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭`DescribeScheduledInstances`配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用`DescribeScheduledInstances`。

CLI

AWS CLI

說明您的排程執行個體

此範例說明指定的排程執行個體。

命令：

```
aws ec2 describe-scheduled-instances --scheduled-instance-ids
 sci-1234-1234-1234-1234-123456789012
```

輸出：

```
{
  "ScheduledInstanceSet": [
      {
          "AvailabilityZone": "us-west-2b",
          "ScheduledInstanceId": "sci-1234-1234-1234-1234-123456789012",
          "HourlyPrice": "0.095",
          "CreateDate": "2016-01-25T21:43:38.612Z",
          "Recurrence": {
              "OccurrenceDaySet": [
                  1
              ],
              "Interval": 1,
              "Frequency": "Weekly",
              "OccurrenceRelativeToEnd": false,
              "OccurrenceUnit": ""
          },
          "Platform": "Linux/UNIX",
          "TermEndDate": "2017-01-31T09:00:00Z",
          "InstanceCount": 1,
          "SlotDurationInHours": 32,
          "TermStartDate": "2016-01-31T09:00:00Z",
          "NetworkPlatform": "EC2-VPC",
          "TotalScheduledInstanceHours": 1696,
          "NextSlotStartTime": "2016-01-31T09:00:00Z",
          "InstanceType": "c4.large"
      }
  ]
}
```

此範例說明您所有的排程執行個體。

命令：

```
aws ec2 describe-scheduled-instances
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeScheduledInstances](#)中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的排程執行個體。

```
Get-EC2ScheduledInstance -ScheduledInstanceId
 sci-1234-1234-1234-1234-123456789012
```

輸出：

```
AvailabilityZone            : us-west-2b
CreateDate                  : 1/25/2016 1:43:38 PM
HourlyPrice                 : 0.095
InstanceCount               : 1
InstanceType                : c4.large
NetworkPlatform             : EC2-VPC
NextSlotStartTime           : 1/31/2016 1:00:00 AM
Platform                    : Linux/UNIX
PreviousSlotEndTime         :
Recurrence                  : Amazon.EC2.Model.ScheduledInstanceRecurrence
ScheduledInstanceId         : sci-1234-1234-1234-1234-123456789012
SlotDurationInHours         : 32
TermEndDate                 : 1/31/2017 1:00:00 AM
TermStartDate               : 1/31/2016 1:00:00 AM
TotalScheduledInstanceHours : 1696
```

範例 2：此範例說明您所有的排程執行個體。

```
Get-EC2ScheduledInstance
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeScheduledInstances式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**DescribeSecurityGroups**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeSecurityGroups。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Retrieve information for an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupId">The Id of the Amazon EC2 security group.</param>
    /// <returns>A list of security group information.</returns>
    public async Task<List<SecurityGroup>> DescribeSecurityGroups(string groupId)
    {
        var request = new DescribeSecurityGroupsRequest();
        var groupIds = new List<string> { groupId };
        request.GroupIds = groupIds;

        var response = await _amazonEC2.DescribeSecurityGroupsAsync(request);
        return response.SecurityGroups;
    }

    /// <summary>
    /// Display the information returned by the call to
    /// DescribeSecurityGroupsAsync.
    /// </summary>
    /// <param name="securityGroup">A list of security group information.</param>
    public void DisplaySecurityGroupInfoAsync(SecurityGroup securityGroup)
    {
        Console.WriteLine($"{securityGroup.GroupName}");
        Console.WriteLine("Ingress permissions:");
        securityGroup.IpPermissions.ForEach(permission =>
        {
            Console.WriteLine($"\tFromPort: {permission.FromPort}");
            Console.WriteLine($"\tIpProtocol: {permission.IpProtocol}");

            Console.Write($"\tIpv4Ranges: ");
            permission.Ipv4Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIp} "); });
```

```
            Console.WriteLine($"\n\tIpv6Ranges:");
            permission.Ipv6Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIpv6} "); });

            Console.Write($"\n\tPrefixListIds: ");
            permission.PrefixListIds.ForEach(id => Console.Write($"{id.Id} "));

            Console.WriteLine($"\n\tTo Port: {permission.ToPort}");
        });
        Console.WriteLine("Egress permissions:");
        securityGroup.IpPermissionsEgress.ForEach(permission =>
        {
            Console.WriteLine($"\tFromPort: {permission.FromPort}");
            Console.WriteLine($"\tIpProtocol: {permission.IpProtocol}");

            Console.Write($"\tIpv4Ranges: ");
            permission.Ipv4Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIp} "); });

            Console.WriteLine($"\n\tIpv6Ranges:");
            permission.Ipv6Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIpv6} "); });

            Console.Write($"\n\tPrefixListIds: ");
            permission.PrefixListIds.ForEach(id => Console.Write($"{id.Id} "));

            Console.WriteLine($"\n\tTo Port: {permission.ToPort}");
        });
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DescribeSecurityGroups中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
##############################################################################
# function ec2_describe_security_groups
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 security groups.
#
# Parameters:
#       -g security_group_id - The ID of the security group to describe
 (optional).
#
# And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function ec2_describe_security_groups() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_security_groups"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 security groups."
    echo "  -g security_group_id - The ID of the security group to describe
 (optional)."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "g:h" option; do
    case "${option}" in
      g) security_group_id="${OPTARG}" ;;
```

```
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local query="SecurityGroups[*].[GroupName, GroupId, VpcId, IpPermissions[*].
[IpProtocol, FromPort, ToPort, IpRanges[*].CidrIp]]"

  if [[ -n "$security_group_id" ]]; then
    response=$(aws ec2 describe-security-groups --group-ids "$security_group_id"
 --query "${query}" --output text)
  else
    response=$(aws ec2 describe-security-groups --query "${query}" --output text)
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports describe-security-groups operation failed.
$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

此範例中使用的公用程式函數。

```
###############################################################################
# function errecho
#
```

```
# This function outputs everything sent to it to STDERR (standard error output).
#########################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#########################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#########################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeSecurityGroups](#)中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);
Aws::EC2::Model::DescribeSecurityGroupsRequest request;

if (!groupID.empty()) {
    request.AddGroupIds(groupID);
}

Aws::String nextToken;
do {
    if (!nextToken.empty()) {
        request.SetNextToken(nextToken);
    }

    auto outcome = ec2Client.DescribeSecurityGroups(request);
    if (outcome.IsSuccess()) {
        std::cout << std::left <<
                  std::setw(32) << "Name" <<
                  std::setw(30) << "GroupId" <<
                  std::setw(30) << "VpcId" <<
                  std::setw(64) << "Description" << std::endl;

        const std::vector<Aws::EC2::Model::SecurityGroup> &securityGroups =
                outcome.GetResult().GetSecurityGroups();

        for (const auto &securityGroup: securityGroups) {
            std::cout << std::left <<
                      std::setw(32) << securityGroup.GetGroupName() <<
                      std::setw(30) << securityGroup.GetGroupId() <<
                      std::setw(30) << securityGroup.GetVpcId() <<
                      std::setw(64) << securityGroup.GetDescription() <<
                      std::endl;
        }
```

```
        }
        else {
            std::cerr << "Failed to describe security groups:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        nextToken = outcome.GetResult().GetNextToken();
    } while (!nextToken.empty());
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考DescribeSecurityGroups中的。

CLI

AWS CLI

範例 1：描述安全群組

下列 describe-security-groups 範例會描述指定的安全群組。

```
aws ec2 describe-security-groups \
    --group-ids sg-903004f8
```

輸出：

```
{
    "SecurityGroups": [
        {
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "UserIdGroupPairs": [],
                    "PrefixListIds": []
                }
            ],
            "Description": "My security group",
            "Tags": [
```

```
                {
                    "Value": "SG1",
                    "Key": "Name"
                }
            ],
            "IpPermissions": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [],
                    "UserIdGroupPairs": [
                        {
                            "UserId": "123456789012",
                            "GroupId": "sg-903004f8"
                        }
                    ],
                    "PrefixListIds": []
                },
                {
                    "PrefixListIds": [],
                    "FromPort": 22,
                    "IpRanges": [
                        {
                            "Description": "Access from NY office",
                            "CidrIp": "203.0.113.0/24"
                        }
                    ],
                    "ToPort": 22,
                    "IpProtocol": "tcp",
                    "UserIdGroupPairs": []
                    }
            ],
            "GroupName": "MySecurityGroup",
            "VpcId": "vpc-1a2b3c4d",
            "OwnerId": "123456789012",
            "GroupId": "sg-903004f8",
        }
    ]
}
```

範例 2：描述具有特定規則的安全群組

下列範`describe-security-groups`例會使用篩選器，將結果限定為具有允許 SSH 流量 (連接埠 22) 規則的安全性群組，以及允許來自所有位址的流量 (`0.0.0.0/0`) 的規則。此範例使用

--query 參數，僅顯示安全群組的名稱。安全群組必須符合所有篩選條件才能在結果中傳回；不過，單一規則不需要符合所有篩選條件。例如，輸出會傳回一個安全群組，其中包含允許來自特定 IP 地址之 SSH 流量的一個規則，以及允許來自所有地址之 HTTP 流量的另一個規則。

```
aws ec2 describe-security-groups \
    --filters Name=ip-permission.from-port,Values=22 Name=ip-permission.to-
port,Values=22 Name=ip-permission.cidr,Values='0.0.0.0/0' \
    --query "SecurityGroups[*].[GroupName]" \
    --output text
```

輸出：

```
default
my-security-group
web-servers
launch-wizard-1
```

範例 3：根據標籤描述安全群組

下列 describe-security-groups 範例會使用篩選條件，將結果範圍限定為在安全群組名稱中加入 test，且具有標籤 Test=To-delete 的安全群組。此範例使用 --query 參數，僅顯示安全群組的名稱和 ID。

```
aws ec2 describe-security-groups \
    --filters Name=group-name,Values=*test* Name=tag:Test,Values=To-delete \
    --query "SecurityGroups[*].{Name:GroupName,ID:GroupId}"
```

輸出：

```
[
    {
        "Name": "testfornewinstance",
        "ID": "sg-33bb22aa"
    },
    {
        "Name": "newgrouptest",
        "ID": "sg-1a2b3c4d"
    }
]
```

如需使用標籤篩選條件的其他範例，請參閱《Amazon EC2 使用者指南》中的使用標籤。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeSecurityGroups中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   public static void describeSecurityGroups(Ec2Client ec2, String groupId) {
       try {
           DescribeSecurityGroupsRequest request =
DescribeSecurityGroupsRequest.builder()
               .groupIds(groupId)
               .build();

           // Use a paginator.
           DescribeSecurityGroupsIterable listGroups =
ec2.describeSecurityGroupsPaginator(request);
           listGroups.stream()
               .flatMap(r -> r.securityGroups().stream())
               .forEach(group -> System.out
                   .println(" Group id: " +group.groupId() + " group name = " +
group.groupName()));

       } catch (Ec2Exception 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DescribeSecurityGroups中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { DescribeSecurityGroupsCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// Log the details of a specific security group.
export const main = async () => {
  const command = new DescribeSecurityGroupsCommand({
    GroupIds: ["SECURITY_GROUP_ID"],
  });

  try {
    const { SecurityGroups } = await client.send(command);
    console.log(JSON.stringify(SecurityGroups, null, 2));
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DescribeSecurityGroups中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun describeEC2SecurityGroups(groupId: String) {
    val request =
        DescribeSecurityGroupsRequest {
            groupIds = listOf(groupId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->

        val response = ec2.describeSecurityGroups(request)
        response.securityGroups?.forEach { group ->
            println("Found Security Group with id ${group.groupId}, vpc id
 ${group.vpcId} and description ${group.description}")
        }
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK DescribeSecurityGroups中的 Kotlin API 參考。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明 VPC 的指定安全性群組。使用屬於 VPC 的安全群組時，您必須使用安全性群組 ID (-GroupId 參數)，而不是名稱 (-GroupName 參數) 來參照群組。

```
Get-EC2SecurityGroup -GroupId sg-12345678
```

輸出：

```
Description         : default VPC security group
GroupId             : sg-12345678
GroupName           : default
IpPermissions       : {Amazon.EC2.Model.IpPermission}
IpPermissionsEgress : {Amazon.EC2.Model.IpPermission}
OwnerId             : 123456789012
Tags                : {}
VpcId               : vpc-12345678
```

範例 2：此範例說明 EC2-Classic 的指定安全性群組。使用 EC2-Classic 的安全性群組時，您可以使用群組名稱 (-GroupName 參數) 或群組識別碼 (-GroupId 參數) 來參照安全性群組。

```
Get-EC2SecurityGroup -GroupName my-security-group
```

輸出：

```
Description         : my security group
GroupId             : sg-45678901
GroupName           : my-security-group
IpPermissions       : {Amazon.EC2.Model.IpPermission,
 Amazon.EC2.Model.IpPermission}
IpPermissionsEgress : {}
OwnerId             : 123456789012
Tags                : {}
VpcId               :
```

範例 3：此範例會擷取 vpc -0fc1ff23456b789eb 的所有安全性群組

```
Get-EC2SecurityGroup -Filter @{Name="vpc-id";Values="vpc-0fc1ff23456b789eb"}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeSecurityGroups式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class SecurityGroupWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) security group
 actions."""

    def __init__(self, ec2_resource, security_group=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
```

```
                            is used to create additional high-level objects
                            that wrap low-level Amazon EC2 service actions.
        :param security_group: A Boto3 SecurityGroup object. This is a high-level
object
                               that wraps security group actions.
        """
        self.ec2_resource = ec2_resource
        self.security_group = security_group

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def describe(self):
        """
        Displays information about the security group.
        """
        if self.security_group is None:
            logger.info("No security group to describe.")
            return

        try:
            print(f"Security group: {self.security_group.group_name}")
            print(f"\tID: {self.security_group.id}")
            print(f"\tVPC: {self.security_group.vpc_id}")
            if self.security_group.ip_permissions:
                print(f"Inbound permissions:")
                pp(self.security_group.ip_permissions)
        except ClientError as err:
            logger.error(
                "Couldn't get data for security group %s. Here's why: %s: %s",
                self.security_group.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeSecurityGroups中的 Python (博托 3) API 參考。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        DATA lt_group_ids TYPE /aws1/
cl_ec2groupidstrlist_w=>tt_groupidstringlist.
        APPEND NEW /aws1/cl_ec2groupidstrlist_w( iv_value = iv_group_id ) TO
 lt_group_ids.
        oo_result = lo_ec2->describesecuritygroups( it_groupids = lt_group_ids ).
         " oo_result is returned for testing purposes. "
        DATA(lt_security_groups) = oo_result->get_securitygroups( ).
        MESSAGE 'Retrieved information about security groups.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK DescribeSecurityGroups中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭`DescribeSnapshotAttribute`配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用`DescribeSnapshotAttribute`。

CLI

AWS CLI

說明快照的快照屬性

下列describe-snapshot-attribute範例會列出與其共用快照的帳戶。

```
aws ec2 describe-snapshot-attribute \
    --snapshot-id snap-01234567890abcedf \
    --attribute createVolumePermission
```

輸出：

```
{
    "SnapshotId": "snap-01234567890abcedf",
    "CreateVolumePermissions": [
        {
            "UserId": "123456789012"
        }
    ]
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端使用者指南中的共用 Amazon EBS 快照。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeSnapshotAttribute中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定快照的指定屬性。

```
Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute ProductCodes
```

輸出：

```
CreateVolumePermissions    ProductCodes    SnapshotId
-----------------------    ------------    ----------
{}                         {}              snap-12345678
```

範例 2：此範例說明指定快照的指定屬性。

```
(Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
 CreateVolumePermission).CreateVolumePermissions
```

輸出：

```
Group    UserId
-----    ------
all
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeSnapshotAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeSnapshots配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeSnapshots。

CLI

AWS CLI

範例 1：描述快照

下列 describe-snapshots 範例會描述指定的快照。

```
aws ec2 describe-snapshots \
    --snapshot-ids snap-1234567890abcdef0
```

輸出：

```
{
    "Snapshots": [
        {
            "Description": "This is my snapshot",
            "Encrypted": false,
            "VolumeId": "vol-049df61146c4d7901",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2019-02-28T21:28:32.000Z",
            "Progress": "100%",
            "OwnerId": "012345678910",
```

```
            "SnapshotId": "snap-01234567890abcdef",
            "Tags": [
                {
                    "Key": "Stack",
                    "Value": "test"
                }
            ]
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EBS 加密。

範例 2：根據篩選條件描述快照

下列範describe-snapshots例會使用篩選條件，將結果範圍限定為您 AWS 帳戶所擁有且處於pending狀態的快照。此範例使用 --query 參數，僅顯示快照 ID 和快照啟動時間。

```
aws ec2 describe-snapshots \
    --owner-ids self \
    --filters Name=status,Values=pending \
    --query "Snapshots[*].{ID:SnapshotId,Time:StartTime}"
```

輸出：

```
[
    {
        "ID": "snap-1234567890abcdef0",
        "Time": "2019-08-04T12:48:18.000Z"
    },
    {
        "ID": "snap-066877671789bd71b",
        "Time": "2019-08-04T02:45:16.000Z
    },
    ...
]
```

下列 describe-snapshots 範例會使用篩選條件，將結果範圍限制為從指定磁碟區建立的快照。此範例使用 --query 參數，僅顯示快照 ID。

```
aws ec2 describe-snapshots \
```

```
    --filters Name=volume-id,Values=049df61146c4d7901 \
    --query "Snapshots[*].[SnapshotId]" \
    --output text
```

輸出：

```
snap-1234567890abcdef0
snap-08637175a712c3fb9
...
```

如需使用篩選條件的其他範例，請參閱《Amazon EC2 使用者指南》中的列出與篩選您的資源。

範例 3：根據標籤描述快照

下列 describe-snapshots 範例會使用標籤篩選條件，將結果範圍設定為具有標籤 Stack=Prod 的快照。

```
aws ec2 describe-snapshots \
    --filters Name=tag:Stack,Values=prod
```

如需 describe-snapshots 的輸出範例，請參閱範例 1。

如需使用標籤篩選條件的其他範例，請參閱《Amazon EC2 使用者指南》中的使用標籤。

範例 4：根據年齡描述快照

下列describe-snapshots範例會使用 JMESPath 運算式來描述您的 AWS 帳戶在指定日期之前建立的所有快照。此範例僅顯示快照 ID。

```
aws ec2 describe-snapshots \
    --owner-ids 012345678910 \
    --query "Snapshots[?(StartTime<='2020-03-31')].[SnapshotId]"
```

如需使用篩選條件的其他範例，請參閱《Amazon EC2 使用者指南》中的列出與篩選您的資源。

範例 5：僅檢視封存的快照

以下 describe-snapshots 範例只列出儲存在封存層中的快照。

```
aws ec2 describe-snapshots \
    --filters "Name=storage-tier,Values=archive"
```

輸出：

```
{
    "Snapshots": [
        {
            "Description": "Snap A",
            "Encrypted": false,
            "VolumeId": "vol-01234567890aaaaaa",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2021-09-07T21:00:00.000Z",
            "Progress": "100%",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-01234567890aaaaaa",
            "StorageTier": "archive",
            "Tags": []
        },
    ]
}
```

如需詳細資訊，請參閱《Amazon Elastic Compute Cloud 使用者指南》中的檢視封存的快照。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeSnapshots中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的快照。

```
Get-EC2Snapshot -SnapshotId snap-12345678
```

輸出：

```
DataEncryptionKeyId :
Description         : Created by CreateImage(i-1a2b3c4d) for ami-12345678 from
 vol-12345678
Encrypted           : False
```

```
KmsKeyId            :
OwnerAlias          :
OwnerId             : 123456789012
Progress            : 100%
SnapshotId          : snap-12345678
StartTime           : 10/23/2014 6:01:28 AM
State               : completed
StateMessage        :
Tags                : {}
VolumeId            : vol-12345678
VolumeSize          : 8
```

範例 2：此範例說明具有「Name」標籤的快照集。

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" }
```

範例 3：此範例說明具有「Name」標籤且值為 'TestValue' 的快照集。

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" -and
 $_.Tags.Value -eq "TestValue" }
```

範例 4：此範例說明您的所有快照。

```
Get-EC2Snapshot -Owner self
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeSnapshots式參考中的。

Rust

適用於 Rust 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

顯示快照的狀態。

```
async fn show_state(client: &Client, id: &str) -> Result<(), Error> {
    let resp = client
        .describe_snapshots()
        .filters(Filter::builder().name("snapshot-id").values(id).build())
        .send()
        .await?;

    println!(
        "State: {}",
        resp.snapshots().first().unwrap().state().unwrap().as_ref()
    );

    Ok(())
}
```

```
async fn show_snapshots(client: &Client) -> Result<(), Error> {
    // "self" represents your account ID.
    // You can list the snapshots for any account by replacing
    // "self" with that account ID.
    let resp = client.describe_snapshots().owner_ids("self").send().await?;
    let snapshots = resp.snapshots();
    let length = snapshots.len();

    for snapshot in snapshots {
        println!(
            "ID:          {}",
            snapshot.snapshot_id().unwrap_or_default()
        );
        println!(
            "Description: {}",
            snapshot.description().unwrap_or_default()
        );
        println!("State:       {}", snapshot.state().unwrap().as_ref());
        println!();
    }

    println!();
    println!("Found {} snapshot(s)", length);
    println!();

    Ok(())
}
```

- 如需 API 的詳細資訊，請參閱 AWS SDK DescribeSnapshots中的 Rust API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeSpotDatafeedSubscription配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeSpotDatafeedSubscription。

CLI

AWS CLI

說明帳戶的競價型執行個體資料饋送訂閱

此範例命令說明帳戶的資料饋送。

命令：

```
aws ec2 describe-spot-datafeed-subscription
```

輸出：

```
{
    "SpotDatafeedSubscription": {
        "OwnerId": "123456789012",
        "Prefix": "spotdata",
        "Bucket": "my-s3-bucket",
        "State": "Active"
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeSpotDatafeedSubscription中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明您的 Spot 執行個體資料饋送。

```
Get-EC2SpotDatafeedSubscription
```

輸出：

```
Bucket  : my-s3-bucket
Fault   :
OwnerId : 123456789012
Prefix  : spotdata
State   : Active
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程[DescribeSpotDatafeedSubscription](式參考中的。)

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 開發套件建立 Amazon EC2 資源](。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。)

搭**DescribeSpotFleetInstances**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeSpotFleetInstances。

CLI

AWS CLI

說明與 Spot 叢集相關聯的競價型執行個體

此範例命令會列出與指定 Spot 叢集相關聯的 Spot 執行個體。

命令：

```
aws ec2 describe-spot-fleet-instances --spot-fleet-request-id sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

輸出：

```
{
  "ActiveInstances": [
      {
          "InstanceId": "i-1234567890abcdef0",
          "InstanceType": "m3.medium",
          "SpotInstanceRequestId": "sir-08b93456"
```

```
        },
        ...
    ],
    "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeSpotFleetInstances中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明與指定競價型叢集請求相關聯的執行個體。

```
Get-EC2SpotFleetInstance -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

輸出：

```
InstanceId    InstanceType    SpotInstanceRequestId
----------    ------------    ---------------------
i-f089262a    c3.large        sir-12345678
i-7e8b24a4    c3.large        sir-87654321
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeSpotFleetInstances式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeSpotFleetRequestHistory配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeSpotFleetRequestHistory。

CLI

AWS CLI

描述 Spot 艦隊歷史

此範例命令會傳回指定競價型叢集在指定時間開始的歷史記錄。

命令：

```
aws ec2 describe-spot-fleet-request-history --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE --start-time 2015-05-26T00:00:00Z
```

下列範例輸出顯示 Spot 叢集的兩個 Spot 執行個體成功啟動。

輸出：

```
{
  "HistoryRecords": [
      {
          "Timestamp": "2015-05-26T23:17:20.697Z",
          "EventInformation": {
              "EventSubType": "submitted"
          },
          "EventType": "fleetRequestChange"
      },
      {
          "Timestamp": "2015-05-26T23:17:20.873Z",
          "EventInformation": {
              "EventSubType": "active"
          },
          "EventType": "fleetRequestChange"
      },
      {
          "Timestamp": "2015-05-26T23:21:21.712Z",
          "EventInformation": {
              "InstanceId": "i-1234567890abcdef0",
              "EventSubType": "launched"
          },
          "EventType": "instanceChange"
      },
      {
          "Timestamp": "2015-05-26T23:21:21.816Z",
          "EventInformation": {
              "InstanceId": "i-1234567890abcdef1",
              "EventSubType": "launched"
          },
          "EventType": "instanceChange"
      }
  ],
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
```

```
  "NextToken": "CpHNsscimcV5oH7bSbub03CI2Qms5+ypNpNm
+53MNlR0YcXAkp0xFlfKf91yVxSExmbtma3awYxMFzNA663ZskT0AHtJ6TCb2Z8bQC2EnZgyELbymtWPfpZ1ZbauV
+P+TfGlWxWWB/Vr5dk5d4LfdgA/DRAHUrYgxzrEXAMPLE=",
  "StartTime": "2015-05-26T00:00:00Z"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeSpotFleetRequestHistory中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定競價型叢集請求的歷史記錄。

```
Get-EC2SpotFleetRequestHistory -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -StartTime 2015-12-26T00:00:00Z
```

輸出：

```
HistoryRecords     : {Amazon.EC2.Model.HistoryRecord,
 Amazon.EC2.Model.HistoryRecord...}
LastEvaluatedTime  : 12/26/2015 8:29:11 AM
NextToken          :
SpotFleetRequestId : sfr-088bc5f1-7e7b-451a-bd13-757f10672b93
StartTime          : 12/25/2015 8:00:00 AM
```

```
(Get-EC2SpotFleetRequestHistory -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -StartTime 2015-12-26T00:00:00Z).HistoryRecords
```

輸出：

```
EventInformation                    EventType            Timestamp
----------------                    ---------            ---------
Amazon.EC2.Model.EventInformation    fleetRequestChange    12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation    fleetRequestChange    12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation    fleetRequestChange    12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation    launched              12/26/2015 8:25:34 AM
Amazon.EC2.Model.EventInformation    launched              12/26/2015 8:25:05 AM
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeSpotFleetRequestHistory式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeSpotFleetRequests配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeSpotFleetRequests。

CLI

AWS CLI

描述您的 Spot 叢集請求

此範例說明您所有的 Spot 叢集請求。

命令：

```
aws ec2 describe-spot-fleet-requests
```

輸出：

```
{
  "SpotFleetRequestConfigs": [
      {
          "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
          "SpotFleetRequestConfig": {
              "TargetCapacity": 20,
              "LaunchSpecifications": [
                  {
                      "EbsOptimized": false,
                      "NetworkInterfaces": [
                          {
                              "SubnetId": "subnet-a61dafcf",
                              "DeviceIndex": 0,
                              "DeleteOnTermination": false,
                              "AssociatePublicIpAddress": true,
                              "SecondaryPrivateIpAddressCount": 0
                          }
                      ],
                      "InstanceType": "cc2.8xlarge",
                      "ImageId": "ami-1a2b3c4d"
                  },
                  {
```

```
                    "EbsOptimized": false,
                    "NetworkInterfaces": [
                        {
                            "SubnetId": "subnet-a61dafcf",
                            "DeviceIndex": 0,
                            "DeleteOnTermination": false,
                            "AssociatePublicIpAddress": true,
                            "SecondaryPrivateIpAddressCount": 0
                        }
                    ],
                    "InstanceType": "r3.8xlarge",
                    "ImageId": "ami-1a2b3c4d"
                }
            ],
            "SpotPrice": "0.05",
            "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
        },
        "SpotFleetRequestState": "active"
    },
    {
        "SpotFleetRequestId": "sfr-306341ed-9739-402e-881b-ce47bEXAMPLE",
        "SpotFleetRequestConfig": {
            "TargetCapacity": 20,
            "LaunchSpecifications": [
                {
                    "EbsOptimized": false,
                    "NetworkInterfaces": [
                        {
                            "SubnetId": "subnet-6e7f829e",
                            "DeviceIndex": 0,
                            "DeleteOnTermination": false,
                            "AssociatePublicIpAddress": true,
                            "SecondaryPrivateIpAddressCount": 0
                        }
                    ],
                    "InstanceType": "m3.medium",
                    "ImageId": "ami-1a2b3c4d"
                }
            ],
            "SpotPrice": "0.05",
            "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
        },
        "SpotFleetRequestState": "active"
    }
```

```
  ]
}
```

描述競價型叢集請求

此範例說明指定的 Spot 叢集請求。

命令：

```
aws ec2 describe-spot-fleet-requests --spot-fleet-request-ids sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

輸出：

```
{
  "SpotFleetRequestConfigs": [
      {
          "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
          "SpotFleetRequestConfig": {
              "TargetCapacity": 20,
              "LaunchSpecifications": [
                  {
                      "EbsOptimized": false,
                      "NetworkInterfaces": [
                          {
                              "SubnetId": "subnet-a61dafcf",
                              "DeviceIndex": 0,
                              "DeleteOnTermination": false,
                              "AssociatePublicIpAddress": true,
                              "SecondaryPrivateIpAddressCount": 0
                          }
                      ],
                      "InstanceType": "cc2.8xlarge",
                      "ImageId": "ami-1a2b3c4d"
                  },
                  {
                      "EbsOptimized": false,
                      "NetworkInterfaces": [
                          {
                              "SubnetId": "subnet-a61dafcf",
                              "DeviceIndex": 0,
                              "DeleteOnTermination": false,
```

```
                            "AssociatePublicIpAddress": true,
                            "SecondaryPrivateIpAddressCount": 0
                        }
                    ],
                    "InstanceType": "r3.8xlarge",
                    "ImageId": "ami-1a2b3c4d"
                }
            ],
            "SpotPrice": "0.05",
            "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
        },
        "SpotFleetRequestState": "active"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeSpotFleetRequests中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的 Spot 叢集請求。

```
Get-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE | format-list
```

輸出：

```
ConfigData            : Amazon.EC2.Model.SpotFleetRequestConfigData
CreateTime            : 12/26/2015 8:23:33 AM
SpotFleetRequestId    : sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
SpotFleetRequestState : active
```

範例 2：此範例說明您所有的 Spot 叢集請求。

```
Get-EC2SpotFleetRequest
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeSpotFleetRequests式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeSpotInstanceRequests配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeSpotInstanceRequests。

CLI

AWS CLI

範例 1：描述競價型執行個體請求

下列describe-spot-instance-requests範例說明指定的 Spot 執行個體請求。

```
aws ec2 describe-spot-instance-requests \
    --spot-instance-request-ids sir-08b93456
```

輸出：

```
{
    "SpotInstanceRequests": [
        {
            "CreateTime": "2018-04-30T18:14:55.000Z",
            "InstanceId": "i-1234567890abcdef1",
            "LaunchSpecification": {
                "InstanceType": "t2.micro",
                "ImageId": "ami-003634241a8fcdec0",
                "KeyName": "my-key-pair",
                "SecurityGroups": [
                    {
                        "GroupName": "default",
                        "GroupId": "sg-e38f24a7"
                    }
                ],
                "BlockDeviceMappings": [
                    {
                        "DeviceName": "/dev/sda1",
                        "Ebs": {
                            "DeleteOnTermination": true,
                            "SnapshotId": "snap-0e54a519c999adbbd",
                            "VolumeSize": 8,
                            "VolumeType": "standard",
```

```
                            "Encrypted": false
                        }
                    }
                ],
                "NetworkInterfaces": [
                    {
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "SubnetId": "subnet-049df61146c4d7901"
                    }
                ],
                "Placement": {
                    "AvailabilityZone": "us-east-2b",
                    "Tenancy": "default"
                },
                "Monitoring": {
                    "Enabled": false
                }
            },
            "LaunchedAvailabilityZone": "us-east-2b",
            "ProductDescription": "Linux/UNIX",
            "SpotInstanceRequestId": "sir-08b93456",
            "SpotPrice": "0.010000"
            "State": "active",
            "Status": {
                "Code": "fulfilled",
                "Message": "Your Spot request is fulfilled.",
                "UpdateTime": "2018-04-30T18:16:21.000Z"
            },
            "Tags": [],
            "Type": "one-time",
            "InstanceInterruptionBehavior": "terminate"
        }
    ]
}
```

範例 2：根據篩選器描述 Spot 執行個體請求

下列範`describe-spot-instance-requests`例會使用篩選器，將結果範圍限定為具有指定可用區域中指定執行個體類型的 Spot 執行個體請求。此範例使用`--query`參數僅顯示例證 ID。

```
aws ec2 describe-spot-instance-requests \
```

```
    --filters Name=launch.instance-type,Values=m3.medium Name=launched-
availability-zone,Values=us-east-2a \
    --query "SpotInstanceRequests[*].[InstanceId]" \
    --output text
```

輸出：

```
i-057750d42936e468a
i-001efd250faaa6ffa
i-027552a73f021f3bd
...
```

如需使用篩選器的其他範例，請參閱 Amazon 彈性運算雲端使用者指南中的列出和篩選資源。

範例 3：根據標籤描述 Spot 執行個體請求

下列範例使用標籤篩選器，將結果範圍限定為具有標籤的競價型執行個體請求。

```
aws ec2 describe-spot-instance-requests \
    --filters Name=tag:cost-center,Values=cc123
```

如需 的輸出範例，請參閱範例 1。

如需使用標籤篩選條件的其他範例，請參閱《Amazon EC2 使用者指南》中的使用標籤。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的 Spot 執行個體請求。

```
Get-EC2SpotInstanceRequest -SpotInstanceRequestId sir-12345678
```

輸出：

```
ActualBlockHourlyPrice   :
AvailabilityZoneGroup    :
BlockDurationMinutes     : 0
CreateTime               : 4/8/2015 2:51:33 PM
```

```
Fault                     :
InstanceId                : i-12345678
LaunchedAvailabilityZone  : us-west-2b
LaunchGroup               :
LaunchSpecification       : Amazon.EC2.Model.LaunchSpecification
ProductDescription        : Linux/UNIX
SpotInstanceRequestId     : sir-12345678
SpotPrice                 : 0.020000
State                     : active
Status                    : Amazon.EC2.Model.SpotInstanceStatus
Tags                      : {Name}
Type                      : one-time
```

範例 2：此範例說明您所有的 Spot 執行個體請求。

```
Get-EC2SpotInstanceRequest
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeSpotInstanceRequests式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeSpotPriceHistory配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeSpotPriceHistory。

CLI

AWS CLI

描述現貨價格歷史

此範例指令會傳回一月特定日期 m1.xlarge 執行個體的現貨價格歷史記錄。

命令：

```
aws ec2 describe-spot-price-history --instance-types m1.xlarge --start-time
 2014-01-06T07:08:09 --end-time 2014-01-06T08:09:10
```

輸出：

```
{
  "SpotPriceHistory": [
          {
              "Timestamp": "2014-01-06T07:10:55.000Z",
              "ProductDescription": "SUSE Linux",
              "InstanceType": "m1.xlarge",
              "SpotPrice": "0.087000",
              "AvailabilityZone": "us-west-1b"
          },
          {
              "Timestamp": "2014-01-06T07:10:55.000Z",
              "ProductDescription": "SUSE Linux",
              "InstanceType": "m1.xlarge",
              "SpotPrice": "0.087000",
              "AvailabilityZone": "us-west-1c"
          },
          {
              "Timestamp": "2014-01-06T05:42:36.000Z",
              "ProductDescription": "SUSE Linux (Amazon VPC)",
              "InstanceType": "m1.xlarge",
              "SpotPrice": "0.087000",
              "AvailabilityZone": "us-west-1a"
      },
      ...
}
```

用於描述 LinuX/Unix Amazon VPC 的現貨價格歷史記錄

此範例指令會傳回一月特定日期 m1.xlarge、Linux/Unix Amazon VPC 執行個體的現貨價格歷史記錄。

命令：

```
aws ec2 describe-spot-price-history --instance-types m1.xlarge --product-
description "Linux/UNIX (Amazon VPC)" --start-time 2014-01-06T07:08:09 --end-time
 2014-01-06T08:09:10
```

輸出：

```
{
  "SpotPriceHistory": [
      {
```

```
            "Timestamp": "2014-01-06T04:32:53.000Z",
            "ProductDescription": "Linux/UNIX (Amazon VPC)",
            "InstanceType": "m1.xlarge",
            "SpotPrice": "0.080000",
            "AvailabilityZone": "us-west-1a"
        },
        {
            "Timestamp": "2014-01-05T11:28:26.000Z",
            "ProductDescription": "Linux/UNIX (Amazon VPC)",
            "InstanceType": "m1.xlarge",
            "SpotPrice": "0.080000",
            "AvailabilityZone": "us-west-1c"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeSpotPriceHistory中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會取得指定執行個體類型和可用區域的 Spot 價格歷史記錄中的最後 10 個項目。請注意，為-AvailabilityZone 參數指定的值必須對提供給指令程式之-Region 參數的區域值有效 (未顯示在範例中)，或在命令介面中設定為預設值。此範例指令假設環境中已設定「us-west-2 的預設區域。

```
Get-EC2SpotPriceHistory -InstanceType c3.large -AvailabilityZone us-west-2a -
MaxResult 10
```

輸出：

```
AvailabilityZone   : us-west-2a
InstanceType       : c3.large
Price              : 0.017300
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp          : 12/25/2015 7:39:49 AM

AvailabilityZone   : us-west-2a
InstanceType       : c3.large
Price              : 0.017200
ProductDescription : Linux/UNIX (Amazon VPC)
```

```
Timestamp          : 12/25/2015 7:38:29 AM

AvailabilityZone   : us-west-2a
InstanceType       : c3.large
Price              : 0.017300
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp          : 12/25/2015 6:57:13 AM
...
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeSpotPriceHistory式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeSubnets配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeSubnets。

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：

- 建置及管理彈性服務

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
/// <summary>
/// Get all the subnets for a Vpc in a set of availability zones.
/// </summary>
/// <param name="vpcId">The Id of the Vpc.</param>
/// <param name="availabilityZones">The list of availability zones.</param>
/// <returns>The collection of subnet objects.</returns>
```

```
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        var subnets = new List<Subnet>();
        var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
            new DescribeSubnetsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("vpc-id", new List<string>() { vpcId}),
                    new ("availability-zone", availabilityZones),
                    new ("default-for-az", new List<string>() { "true" })
                }
            });

        // Get the entire list using the paginator.
        await foreach (var subnet in subnetPaginator.Subnets)
        {
            subnets.Add(subnet);
        }

        return subnets;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考[DescribeSubnets](#)中的。

CLI

AWS CLI

範例 1：描述所有子網路

以下 `describe-subnets` 範例顯示子網路的詳細資訊。

```
aws ec2 describe-subnets
```

輸出：

```
{
    "Subnets": [
        {
            "AvailabilityZone": "us-east-1d",
```

```
            "AvailabilityZoneId": "use1-az2",
            "AvailableIpAddressCount": 4089,
            "CidrBlock": "172.31.80.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": false,
            "MapCustomerOwnedIpOnLaunch": true,
            "State": "available",
            "SubnetId": "subnet-0bb1c79de3EXAMPLE",
            "VpcId": "vpc-0ee975135dEXAMPLE",
            "OwnerId": "111122223333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "CustomerOwnedIpv4Pool:": 'pool-2EXAMPLE',
            "SubnetArn": "arn:aws:ec2:us-east-2:111122223333:subnet/
subnet-0bb1c79de3EXAMPLE",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        },
        {
            "AvailabilityZone": "us-east-1d",
            "AvailabilityZoneId": "use1-az2",
            "AvailableIpAddressCount": 4089,
            "CidrBlock": "172.31.80.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true,
            "MapCustomerOwnedIpOnLaunch": false,
            "State": "available",
            "SubnetId": "subnet-8EXAMPLE",
            "VpcId": "vpc-3EXAMPLE",
            "OwnerId": "1111222233333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "MySubnet"
                }
            ],
```

```
            "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-8EXAMPLE",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        }
    ]
}
```

如需詳細資訊，請參閱《AWS VPC 使用者指南》中的使用 VPC 和子網路。

範例 2：描述特定 VPC 的子網路

下列 `describe-subnets` 範例會使用篩選條件來擷取指定 VPC 得子網路詳細資訊。

```
aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=vpc-3EXAMPLE"
```

輸出：

```
{
    "Subnets": [
        {
            "AvailabilityZone": "us-east-1d",
            "AvailabilityZoneId": "use1-az2",
            "AvailableIpAddressCount": 4089,
            "CidrBlock": "172.31.80.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true,
            "MapCustomerOwnedIpOnLaunch": false,
            "State": "available",
            "SubnetId": "subnet-8EXAMPLE",
            "VpcId": "vpc-3EXAMPLE",
            "OwnerId": "1111222233333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "Tags": [
                {
```

```
                    "Key": "Name",
                    "Value": "MySubnet"
                }
            ],
            "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-8EXAMPLE",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        }
    ]
}
```

如需詳細資訊，請參閱《AWS VPC 使用者指南》中的使用 VPC 和子網路。

範例 3：描述具有特定標籤的子網路

下列 describe-subnets 範例會使用篩選條件來擷取這些子網路 (其中包含標籤 CostCenter=123 和 --query 參數) 的詳細資訊，以顯示具有此標籤之子網路的子網路 ID。

```
aws ec2 describe-subnets \
    --filters "Name=tag:CostCenter,Values=123" \
    --query "Subnets[*].SubnetId" \
    --output text
```

輸出：

```
subnet-0987a87c8b37348ef
subnet-02a95061c45f372ee
subnet-03f720e7de2788d73
```

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用 VPC 和子網路。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeSubnets中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
const client = new EC2Client({});
const { Subnets } = await client.send(
  new DescribeSubnetsCommand({
    Filters: [
      { Name: "vpc-id", Values: [state.defaultVpc] },
      { Name: "availability-zone", Values: state.availabilityZoneNames },
      { Name: "default-for-az", Values: ["true"] },
    ],
  }),
);
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DescribeSubnets中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的子網路。

```
Get-EC2Subnet -SubnetId subnet-1a2b3c4d
```

輸出：

```
AvailabilityZone        : us-west-2c
AvailableIpAddressCount : 251
CidrBlock               : 10.0.0.0/24
DefaultForAz            : False
MapPublicIpOnLaunch     : False
State                   : available
SubnetId                : subnet-1a2b3c4d
```

```
Tags                    : {}
VpcId                   : vpc-12345678
```

範例 2：此範例說明您的所有子網路。

```
Get-EC2Subnet
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeSubnets式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
```

```
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def get_subnets(self, vpc_id, zones):
        """
        Gets the default subnets in a VPC for a specified list of Availability
Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        try:
            response = self.ec2_client.describe_subnets(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )
            subnets = response["Subnets"]
            log.info("Found %s subnets for the specified zones.", len(subnets))
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get subnets: {err}")
        else:
```

```
            return subnets
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeSubnets中的 Python (博托 3) API 參考。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeTags配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeTags。

CLI

AWS CLI

範例 1：描述單一資源的所有標籤

下列describe-tags範例說明指定執行個體的標籤。

```
aws ec2 describe-tags \
    --filters "Name=resource-id,Values=i-1234567890abcdef8"
```

輸出：

```
{
    "Tags": [
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Test",
            "Key": "Stack"
        },
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Beta Server",
            "Key": "Name"
        }
```

```
    ]
}
```

範例 2：描述資源類型的所有標籤

下列describe-tags範例說明磁碟區的標籤。

```
aws ec2 describe-tags \
    --filters "Name=resource-type,Values=volume"
```

輸出：

```
{
    "Tags": [
        {
            "ResourceType": "volume",
            "ResourceId": "vol-1234567890abcdef0",
            "Value": "Project1",
            "Key": "Purpose"
        },
        {
            "ResourceType": "volume",
            "ResourceId": "vol-049df61146c4d7901",
            "Value": "Logs",
            "Key": "Purpose"
        }
    ]
}
```

範例 3：描述所有標籤

下列describe-tags範例說明所有資源的標籤。

```
aws ec2 describe-tags
```

範例 4：根據標籤鍵描述資源的標籤

下列describe-tags範例說明具有金鑰標籤的資源標籤Stack。

```
aws ec2 describe-tags \
    --filters Name=key,Values=Stack
```

輸出：

```
{
    "Tags": [
        {
            "ResourceType": "volume",
            "ResourceId": "vol-027552a73f021f3b",
            "Value": "Production",
            "Key": "Stack"
        },
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Test",
            "Key": "Stack"
        }
    ]
}
```

範例 5：根據標籤索引鍵和標籤值描述資源標籤

下列describe-tags範例說明具有標籤之資源的標籤Stack=Test。

```
aws ec2 describe-tags \
    --filters Name=key,Values=Stack Name=value,Values=Test
```

輸出：

```
{
    "Tags": [
        {
            "ResourceType": "image",
            "ResourceId": "ami-3ac336533f021f3bd",
            "Value": "Test",
            "Key": "Stack"
        },
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Test",
            "Key": "Stack"
        }
```

```
    ]
}
```

下列describe-tags範例會使用替代語法來描述具有標籤的資源Stack=Test。

```
aws ec2 describe-tags \
    --filters "Name=tag:Stack,Values=Test"
```

下列describe-tags範例說明所有執行個體的標籤 (含索引鍵Purpose且沒有值) 的標籤。

```
aws ec2 describe-tags \
    --filters "Name=resource-type,Values=instance" "Name=key,Values=Purpose"
 "Name=value,Values="
```

輸出：

```
{
    "Tags": [
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef5",
            "Value": null,
            "Key": "Purpose"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeTags中的。

PowerShell

用於的工具 PowerShell

例 1：此示例獲取資源類型「圖像」的標籤

```
Get-EC2Tag -Filter @{Name="resource-type";Values="image"}
```

輸出：

```
Key         ResourceId            ResourceType Value
```

```
---          ----------            ------------ -----
Name         ami-0a123b4ccb567a8ea image        Win7-Imported
auto-delete ami-0a123b4ccb567a8ea image        never
```

範例 2：此範例會擷取所有資源的所有標籤，並依資源類型將它們分組

```
Get-EC2Tag | Group-Object resourcetype
```

輸出：

```
Count Name                      Group
----- ----                      -----
    9 subnet                    {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
   53 instance                  {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
    3 route-table               {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
    5 security-group            {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
   30 volume                    {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
    1 internet-gateway          {Amazon.EC2.Model.TagDescription}
    3 network-interface         {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
    4 elastic-ip                {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription}
    1 dhcp-options              {Amazon.EC2.Model.TagDescription}
    2 image                     {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription}
    3 vpc                       {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
```

範例 3：此範例顯示所有標籤為「自動刪除」的資源，且指定區域的值為「no」

```
Get-EC2Tag -Region eu-west-1 -Filter @{Name="tag:auto-delete";Values="no"}
```

輸出：

```
Key         ResourceId            ResourceType Value
---         ----------            ------------ -----
auto-delete i-0f1bce234d5dd678b   instance     no
auto-delete vol-01d234aa5678901a2 volume       no
auto-delete vol-01234bfb5def6f7b8 volume       no
auto-delete vol-01ccb23f4c5e67890 volume       no
```

示例 4：此示例獲取標籤為「自動刪除」的所有資源，其值為「no」，並在下一個管道中進一步過濾器以僅解析「實例」資源類型，並最終為每個實例資源創建 ThisInstance「」標籤，其值為實例 ID 本身

```
Get-EC2Tag -Region eu-west-1 -Filter @{Name="tag:auto-delete";Values="no"}
 | Where-Object ResourceType -eq "instance" | ForEach-Object {New-EC2Tag -
ResourceId $_.ResourceId -Tag @{Key="ThisInstance";Value=$_.ResourceId}}
```

範例 5：此範例會擷取所有執行處理資源的標籤以及「名稱」鍵，並以表格格式顯示這些標籤

```
Get-EC2Tag -Filter @{Name="resource-
type";Values="instance"},@{Name="key";Values="Name"} | Select-Object ResourceId,
 @{Name="Name-Tag";Expression={$PSItem.Value}} | Format-Table -AutoSize
```

輸出：

```
ResourceId          Name-Tag
----------          --------
i-012e3cb4df567e1aa jump1
i-01c23a45d6fc7a89f repro-3
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeTags式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeVolumeAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVolumeAttribute。

CLI

AWS CLI

描述磁碟區屬性

此範例指令描述具有 ID 之磁碟區的autoEnableIo屬性vol-049df61146c4d7901。

命令：

```
aws ec2 describe-volume-attribute --volume-id vol-049df61146c4d7901 --attribute
 autoEnableIO
```

輸出：

```
{
    "AutoEnableIO": {
        "Value": false
    },
    "VolumeId": "vol-049df61146c4d7901"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeVolumeAttribute中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定磁碟區的指定屬性。

```
Get-EC2VolumeAttribute -VolumeId vol-12345678 -Attribute AutoEnableIO
```

輸出：

```
AutoEnableIO    ProductCodes    VolumeId
------------    ------------    --------
False           {}              vol-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeVolumeAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeVolumeStatus配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVolumeStatus。

CLI

AWS CLI

描述單一磁碟區的狀態

此範例指令描述磁碟區的狀態vol-1234567890abcdef0。

命令：

```
aws ec2 describe-volume-status --volume-ids vol-1234567890abcdef0
```

輸出：

```
{
    "VolumeStatuses": [
        {
            "VolumeStatus": {
                "Status": "ok",
                "Details": [
                    {
                        "Status": "passed",
                        "Name": "io-enabled"
                    },
                    {
                        "Status": "not-applicable",
                        "Name": "io-performance"
                    }
                ]
            },
            "AvailabilityZone": "us-east-1a",
            "VolumeId": "vol-1234567890abcdef0",
            "Actions": [],
            "Events": []
        }
    ]
```

```
}
```

描述受損磁碟區的狀態

此範例指令描述所有受損磁碟區的狀態。在此範例輸出中，沒有受損的磁碟區。

命令：

```
aws ec2 describe-volume-status --filters Name=volume-
status.status,Values=impaired
```

輸出：

```
{
    "VolumeStatuses": []
}
```

如果磁碟區的狀態檢查失敗 (狀態受損)，請參閱 Amazon EC2 使用者指南中的使用受損的磁碟區。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeVolumeStatus中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定磁碟區的狀態。

```
Get-EC2VolumeStatus -VolumeId vol-12345678
```

輸出：

```
Actions          : {}
AvailabilityZone : us-west-2a
Events           : {}
VolumeId         : vol-12345678
VolumeStatus     : Amazon.EC2.Model.VolumeStatusInfo
```

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus
```

輸出：

```
Details                         Status
-------                         ------
{io-enabled, io-performance}    ok
```

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus.Details
```

輸出：

```
Name                            Status
----                            ------
io-enabled                      passed
io-performance                  not-applicable
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeVolumeStatus式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeVolumes配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVolumes。

CLI

AWS CLI

範例 1：描述磁碟區

下列describe-volumes範例說明目前區域中指定的磁碟區。

```
aws ec2 describe-volumes \
    --volume-ids vol-049df61146c4d7901 vol-1234567890abcdef0
```

輸出：

```
{
    "Volumes": [
        {
```

```
            "AvailabilityZone": "us-east-1a",
            "Attachments": [
                {
                    "AttachTime": "2013-12-18T22:35:00.000Z",
                    "InstanceId": "i-1234567890abcdef0",
                    "VolumeId": "vol-049df61146c4d7901",
                    "State": "attached",
                    "DeleteOnTermination": true,
                    "Device": "/dev/sda1"
                }
            ],
            "Encrypted": true,
            "KmsKeyId": "arn:aws:kms:us-east-2a:123456789012:key/8c5b2c63-
b9bc-45a3-a87a-5513eEXAMPLE,
            "VolumeType": "gp2",
            "VolumeId": "vol-049df61146c4d7901",
            "State": "in-use",
            "Iops": 100,
            "SnapshotId": "snap-1234567890abcdef0",
            "CreateTime": "2019-12-18T22:35:00.084Z",
            "Size": 8
        },
        {
            "AvailabilityZone": "us-east-1a",
            "Attachments": [],
            "Encrypted": false,
            "VolumeType": "gp2",
            "VolumeId": "vol-1234567890abcdef0",
            "State": "available",
            "Iops": 300,
            "SnapshotId": "",
            "CreateTime": "2020-02-27T00:02:41.791Z",
            "Size": 100
        }
    ]
}
```

範例 2：說明附加至特定執行個體的磁碟區

下列`describe-volumes`範例說明所有磁碟區都附加至指定的執行個體，並在執行個體終止時設定為 delete。

```
aws ec2 describe-volumes \
```

```
    --region us-east-1 \
    --filters Name=attachment.instance-id,Values=i-1234567890abcdef0
 Name=attachment.delete-on-termination,Values=true
```

如需 describe-volumes 的輸出範例，請參閱範例 1。

範例 3：說明特定可用區域中的可用磁碟區

下列describe-volumes範例說明所有狀態為available且位於指定可用區域中的磁碟區。

```
aws ec2 describe-volumes \
    --filters Name=status,Values=available Name=availability-zone,Values=us-
east-1a
```

如需 describe-volumes 的輸出範例，請參閱範例 1。

範例 4：根據標籤描述磁碟區

下列describe-volumes範例說明所有具有標籤索引鍵Name和開頭值的磁碟區Test。然後使用僅顯示磁碟區標籤和 ID 的查詢來篩選輸出。

```
aws ec2 describe-volumes \
    --filters Name=tag:Name,Values=Test* \
    --query "Volumes[*].{ID:VolumeId,Tag:Tags}"
```

輸出：

```
[
    {
       "Tag": [
           {
               "Value": "Test2",
               "Key": "Name"
           }
       ],
       "ID": "vol-1234567890abcdef0"
   },
   {
       "Tag": [
           {
               "Value": "Test1",
               "Key": "Name"
```

```
            }
        ],
        "ID": "vol-049df61146c4d7901"
    }
]
```

如需使用標籤篩選條件的其他範例，請參閱《Amazon EC2 使用者指南》中的使用標籤。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeVolumes中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的 EBS 磁碟區。

```
Get-EC2Volume -VolumeId vol-12345678
```

輸出：

```
Attachments      : {}
AvailabilityZone : us-west-2c
CreateTime       : 7/17/2015 4:35:19 PM
Encrypted        : False
Iops             : 90
KmsKeyId         :
Size             : 30
SnapshotId       : snap-12345678
State            : in-use
Tags             : {}
VolumeId         : vol-12345678
VolumeType       : standard
```

範例 2：此範例說明狀態為「可用」的 EBS 磁碟區。

```
Get-EC2Volume -Filter @{ Name="status"; Values="available" }
```

輸出：

```
Attachments      : {}
AvailabilityZone : us-west-2c
```

```
CreateTime       : 12/21/2015 2:31:29 PM
Encrypted        : False
Iops             : 60
KmsKeyId         :
Size             : 20
SnapshotId       : snap-12345678
State            : available
Tags             : {}
VolumeId         : vol-12345678
VolumeType       : gp2
...
```

範例 3：此範例說明所有 EBS 磁碟區。

```
Get-EC2Volume
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeVolumes式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeVpcAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVpcAttribute。

CLI

AWS CLI

描述屬 enableDnsSupport 性

此範例說明enableDnsSupport屬性。此屬性指出 VPC 是否已啟用 DNS 解析。如果此屬性為 true，Amazon DNS 伺服器會將您的執行個體的 DNS 主機名稱解析為對應的 IP 地址；否則將不會進行解析。

命令：

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute enableDnsSupport
```

輸出：

```
{
    "VpcId": "vpc-a01106c2",
    "EnableDnsSupport": {
        "Value": true
    }
}
```

描述屬 enableDnsHostnames 性

此範例說明enableDnsHostnames屬性。此屬性指出在 VPC 中啟動的執行個體是否取得 DNS 主機名稱。如果此屬性為 true，該 VPC 中的執行個體會取得 DNS 主機名稱；否則將不會取得。

命令：

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute
 enableDnsHostnames
```

輸出：

```
{
    "VpcId": "vpc-a01106c2",
    "EnableDnsHostnames": {
        "Value": true
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeVpcAttribute中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明 'enableDnsSupport' 屬性。

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsSupport
```

輸出：

```
EnableDnsSupport
```

```
----------------
True
```

範例 2：此範例說明 'enableDnsHostnames' 屬性。

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsHostnames
```

輸出：

```
EnableDnsHostnames
------------------
True
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeVpcAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeVpcClassicLink配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVpcClassicLink。

CLI

AWS CLI

描述 V ClassicLink PC 的狀態

這個範例會列出 vpc 的 ClassicLink 狀態。

命令：

```
aws ec2 describe-vpc-classic-link --vpc-id vpc-88888888
```

輸出：

```
{
  "Vpcs": [
    {
      "ClassicLinkEnabled": true,
```

```
      "VpcId": "vpc-88888888",
      "Tags": [
        {
          "Value": "classiclinkvpc",
          "Key": "Name"
        }
      ]
    }
  ]
}
```

此範例僅列出針對類別連結啟用的 VPC (的篩選器值設定`is-classic-link-enabled`為)。`true`

命令：

```
aws ec2 describe-vpc-classic-link --filter "Name=is-classic-link-
enabled,Values=true"
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeVpcClassicLink](#)中的。

PowerShell

用於的工具 PowerShell

範例 1：上述範例會傳回所有 VPC 及其區 ClassicLinkEnabled 域狀態

```
Get-EC2VpcClassicLink -Region eu-west-1
```

輸出：

```
ClassicLinkEnabled Tags   VpcId
------------------ ----   -----
False              {Name} vpc-0fc1ff23f45b678eb
False              {}     vpc-01e23c4a5d6db78e9
False              {Name} vpc-0123456b078b9d01f
False              {}     vpc-12cf3b4f
False              {Name} vpc-0b12d3456a7e8901d
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程[DescribeVpcClassicLink](#)式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeVpcClassicLinkDnsSupport配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVpcClassicLinkDnsSupport。

CLI

AWS CLI

若要說明虛擬私人雲端的 ClassicLink DNS 支援

此範例說明所有 VPC 的 ClassicLink DNS 支援狀態。

命令：

```
aws ec2 describe-vpc-classic-link-dns-support
```

輸出：

```
{
  "Vpcs": [
    {
      "VpcId": "vpc-88888888",
      "ClassicLinkDnsSupported": true
    },
    {
      "VpcId": "vpc-1a2b3c4d",
      "ClassicLinkDnsSupported": false
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeVpcClassicLinkDnsSupport中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明 eu-west-1 地區的 VPC 的 ClassicLink DNS 支援狀態

```
Get-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

輸出：

```
ClassicLinkDnsSupported VpcId
----------------------- -----
False                   vpc-0b12d3456a7e8910d
False                   vpc-12cf3b4f
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeVpcClassicLinkDnsSupport式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeVpcEndpointServices配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVpcEndpointServices。

CLI

AWS CLI

範例 1：描述所有 VPC 端點服務

下列 "describe-vpc-endpoint-services" 範例會列出某個 AWS 區域的所有 VPC 端點服務。

```
aws ec2 describe-vpc-endpoint-services
```

輸出：

```
{
    "ServiceDetails": [
        {
            "ServiceType": [
                {
                    "ServiceType": "Gateway"
                }
            ],
            "AcceptanceRequired": false,
            "ServiceName": "com.amazonaws.us-east-1.dynamodb",
```

```
            "VpcEndpointPolicySupported": true,
            "Owner": "amazon",
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e",
                "us-east-1f"
            ],
            "BaseEndpointDnsNames": [
                "dynamodb.us-east-1.amazonaws.com"
            ]
        },
        {
            "ServiceType": [
                {
                    "ServiceType": "Interface"
                }
            ],
            "PrivateDnsName": "ec2.us-east-1.amazonaws.com",
            "ServiceName": "com.amazonaws.us-east-1.ec2",
            "VpcEndpointPolicySupported": false,
            "Owner": "amazon",
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e",
                "us-east-1f"
            ],
            "AcceptanceRequired": false,
            "BaseEndpointDnsNames": [
                "ec2.us-east-1.vpce.amazonaws.com"
            ]
        },
        {
            "ServiceType": [
                {
                    "ServiceType": "Interface"
                }
            ],
            "PrivateDnsName": "ssm.us-east-1.amazonaws.com",
```

```
            "ServiceName": "com.amazonaws.us-east-1.ssm",
            "VpcEndpointPolicySupported": true,
            "Owner": "amazon",
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e"
            ],
            "AcceptanceRequired": false,
            "BaseEndpointDnsNames": [
                "ssm.us-east-1.vpce.amazonaws.com"
            ]
        }
    ],
    "ServiceNames": [
        "com.amazonaws.us-east-1.dynamodb",
        "com.amazonaws.us-east-1.ec2",
        "com.amazonaws.us-east-1.ec2messages",
        "com.amazonaws.us-east-1.elasticloadbalancing",
        "com.amazonaws.us-east-1.kinesis-streams",
        "com.amazonaws.us-east-1.s3",
        "com.amazonaws.us-east-1.ssm"
    ]
}
```

如需詳細資訊，請參閱的使用指南中的檢視可用 AWS 服務名稱 AWS PrivateLink。

範例 2：描述有關端點服務的詳細資料

下列 "describe-vpc-endpoint-services" 範例列出 Amazon S3 介面端點伺服器的詳細資訊

```
aws ec2 describe-vpc-endpoint-services \
    --filter "Name=service-type,Values=Interface" Name=service-
name,Values=com.amazonaws.us-east-1.s3
```

輸出：

```
{
    "ServiceDetails": [
        {
            "ServiceName": "com.amazonaws.us-east-1.s3",
```

```
            "ServiceId": "vpce-svc-081d84efcdEXAMPLE",
            "ServiceType": [
                {
                    "ServiceType": "Interface"
                }
            ],
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e",
            "us-east-1f"
            ],
            "Owner": "amazon",
            "BaseEndpointDnsNames": [
                "s3.us-east-1.vpce.amazonaws.com"
            ],
            "VpcEndpointPolicySupported": true,
            "AcceptanceRequired": false,
            "ManagesVpcEndpoints": false,
            "Tags": []
        }
    ],
    "ServiceNames": [
        "com.amazonaws.us-east-1.s3"
    ]
}
```

如需詳細資訊，請參閱的使用指南中的檢視可用 AWS 服務名稱 AWS PrivateLink。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeVpcEndpointServices中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明具有指定篩選器的 EC2 VPC 端點服務，在本例中為 .amazonaws.eu 西部 1.ecs。此外，它還擴展了 ServiceDetails 屬性並顯示詳細信息

```
Get-EC2VpcEndpointService -Region eu-west-1 -MaxResult 5 -Filter @{Name="service-
name";Values="com.amazonaws.eu-west-1.ecs"} | Select-Object -ExpandProperty
 ServiceDetails
```

輸出：

```
AcceptanceRequired         : False
AvailabilityZones          : {eu-west-1a, eu-west-1b, eu-west-1c}
BaseEndpointDnsNames       : {ecs.eu-west-1.vpce.amazonaws.com}
Owner                      : amazon
PrivateDnsName             : ecs.eu-west-1.amazonaws.com
ServiceName                : com.amazonaws.eu-west-1.ecs
ServiceType                : {Amazon.EC2.Model.ServiceTypeDetail}
VpcEndpointPolicySupported : False
```

範例 2：此範例擷取所有 EC2 VPC 端點服務，並傳回 ServiceNames 相符的「ssm」

```
Get-EC2VpcEndpointService -Region eu-west-1 | Select-Object -ExpandProperty
 Servicenames | Where-Object { -match "ssm"}
```

輸出：

```
com.amazonaws.eu-west-1.ssm
com.amazonaws.eu-west-1.ssmmessages
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程[DescribeVpcEndpointServices](式參考中的。)

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 開發套件建立 Amazon EC2 資源](。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。)

搭`DescribeVpcEndpoints`配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用`DescribeVpcEndpoints`。

CLI

AWS CLI

描述您的 VPC 端點

下列`describe-vpc-endpoints`範例會顯示所有 VPC 端點的詳細資料。

```
aws ec2 describe-vpc-endpoints
```

輸出：


```
{
    "VpcEndpoints": [
        {
            "PolicyDocument": "{\"Version\":\"2008-10-17\",\"Statement\":
[{\"Effect\":\"Allow\",\"Principal\":\"*\",\"Action\":\"*\",\"Resource\":\"*
\"}]}",
            "VpcId": "vpc-aabb1122",
            "NetworkInterfaceIds": [],
            "SubnetIds": [],
            "PrivateDnsEnabled": true,
            "State": "available",
            "ServiceName": "com.amazonaws.us-east-1.dynamodb",
            "RouteTableIds": [
                "rtb-3d560345"
            ],
            "Groups": [],
            "VpcEndpointId": "vpce-032a826a",
            "VpcEndpointType": "Gateway",
            "CreationTimestamp": "2017-09-05T20:41:28Z",
            "DnsEntries": [],
            "OwnerId": "123456789012"
        },
        {
            "PolicyDocument": "{\n  \"Statement\": [\n    {\n      \"Action\":
 \"*\", \n      \"Effect\": \"Allow\", \n      \"Principal\": \"*\", \n
 \"Resource\": \"*\"\n    }\n  ]\n}",
            "VpcId": "vpc-1a2b3c4d",
            "NetworkInterfaceIds": [
                "eni-2ec2b084",
                "eni-1b4a65cf"
            ],
            "SubnetIds": [
                "subnet-d6fcaa8d",
                "subnet-7b16de0c"
            ],
            "PrivateDnsEnabled": false,
            "State": "available",
            "ServiceName": "com.amazonaws.us-east-1.elasticloadbalancing",
            "RouteTableIds": [],
            "Groups": [
                {
                    "GroupName": "default",
```

```
                    "GroupId": "sg-54e8bf31"
                }
            ],
            "VpcEndpointId": "vpce-0f89a33420c1931d7",
            "VpcEndpointType": "Interface",
            "CreationTimestamp": "2017-09-05T17:55:27.583Z",
            "DnsEntries": [
                {
                    "HostedZoneId": "Z7HUB22UULQXV",
                    "DnsName": "vpce-0f89a33420c1931d7-
bluzidnv.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
                },
                {
                    "HostedZoneId": "Z7HUB22UULQXV",
                    "DnsName": "vpce-0f89a33420c1931d7-bluzidnv-us-
east-1b.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
                },
                {
                    "HostedZoneId": "Z7HUB22UULQXV",
                    "DnsName": "vpce-0f89a33420c1931d7-bluzidnv-us-
east-1a.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
                }
            ],
            "OwnerId": "123456789012"
        },
        {
            "VpcEndpointId": "vpce-aabbaabbaabbaabba",
            "VpcEndpointType": "GatewayLoadBalancer",
            "VpcId": "vpc-111122223333aabbc",
            "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-
svc-123123a1c43abc123",
            "State": "available",
            "SubnetIds": [
                "subnet-0011aabbcc2233445"
            ],
            "RequesterManaged": false,
            "NetworkInterfaceIds": [
                "eni-01010120203030405"
            ],
            "CreationTimestamp": "2020-11-11T08:06:03.522Z",
            "Tags": [],
            "OwnerId": "123456789012"
        }
    ]
```

```
}
```

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 端點。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeVpcEndpoints中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明 eu-west-1 區域的一或多個 VPC 端點。然後將輸出傳送到下一個命令，該命令選擇 VpcEndpointId 屬性並將數組 VPC ID 作為字符串數組返回

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object -ExpandProperty
 VpcEndpointId
```

輸出：

```
vpce-01a2ab3f4f5cc6f7d
vpce-01d2b345a6787890b
vpce-0012e34d567890e12
vpce-0c123db4567890123
```

範例 2：此範例說明 eu-west-1 區域的所有 vpc 端點，並選取 VpcEndpointId VpcId、ServiceName 和 PrivateDnsEnabled 屬性以表格格式呈現

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object VpcEndpointId, VpcId,
 ServiceName, PrivateDnsEnabled | Format-Table -AutoSize
```

輸出：

```
VpcEndpointId          VpcId                ServiceName
 PrivateDnsEnabled
-------------          -----                -----------
 -----------------
vpce-02a2ab2f2f2cc2f2d vpc-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ssm
             True
vpce-01d1b111a1114561b vpc-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ec2
             True
vpce-0011e23d45167e838 vpc-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ec2messages
             True
```

```
vpce-0c123db4567890123 vpc-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ssmmessages
            True
```

範例 3：此範例會將虛擬私人雲端端點 vpce-01a2ab3f4f5cc6f7d 的政策文件匯出至 json 檔案

```
Get-EC2VpcEndpoint -Region eu-west-1 -VpcEndpointId vpce-01a2ab3f4f5cc6f7d |
 Select-Object -expand PolicyDocument | Out-File vpce_policyDocument.json
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeVpcEndpoints式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeVpcs配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVpcs。

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：

- 建置及管理彈性服務

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
    public async Task<Vpc> GetDefaultVpc()
    {
```

```
        var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
            new DescribeVpcsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("is-default", new List<string>() { "true" })
                }
            });
        return vpcResponse.Vpcs[0];
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DescribeVpcs中的。

CLI

AWS CLI

範例 1：描述所有 VPC

下列 `describe-vpcs` 範例會擷取有關您 VPC 的詳細資訊。

```
aws ec2 describe-vpcs
```

輸出：

```
{
    "Vpcs": [
        {
            "CidrBlock": "30.1.0.0/16",
            "DhcpOptionsId": "dopt-19edf471",
            "State": "available",
            "VpcId": "vpc-0e9801d129EXAMPLE",
            "OwnerId": "111122223333",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-062c64cfafEXAMPLE",
                    "CidrBlock": "30.1.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
```

```
                }
            ],
            "IsDefault": false,
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Not Shared"
                }
            ]
        },
        {
            "CidrBlock": "10.0.0.0/16",
            "DhcpOptionsId": "dopt-19edf471",
            "State": "available",
            "VpcId": "vpc-06e4ab6c6cEXAMPLE",
            "OwnerId": "222222222222",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-00b17b4eddEXAMPLE",
                    "CidrBlock": "10.0.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": false,
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Shared VPC"
                }
            ]
        }
    ]
}
```

範例 2：描述指定的 VPC

下列 `describe-vpcs` 範例會擷取指定 VPC 的詳細資訊。

```
aws ec2 describe-vpcs \
    --vpc-ids vpc-06e4ab6c6cEXAMPLE
```

輸出：

```
{
    "Vpcs": [
        {
            "CidrBlock": "10.0.0.0/16",
            "DhcpOptionsId": "dopt-19edf471",
            "State": "available",
            "VpcId": "vpc-06e4ab6c6cEXAMPLE",
            "OwnerId": "111122223333",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-00b17b4eddEXAMPLE",
                    "CidrBlock": "10.0.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": false,
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Shared VPC"
                }
            ]
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeVpcs中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    const client = new EC2Client({});
    const { Vpcs } = await client.send(
      new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }],
      }),
    );
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DescribeVpcs中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的 VPC。

```
Get-EC2Vpc -VpcId vpc-12345678
```

輸出：

```
CidrBlock       : 10.0.0.0/16
DhcpOptionsId   : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault       : False
State           : available
Tags            : {Name}
VpcId           : vpc-12345678
```

範例 2：此範例說明預設 VPC (每個區域只能有一個)。如果您的帳戶在此區域支援 EC2-Classic，則沒有預設 VPC。

```
Get-EC2Vpc -Filter @{Name="isDefault"; Values="true"}
```

輸出：

```
CidrBlock       : 172.31.0.0/16
DhcpOptionsId   : dopt-12345678
InstanceTenancy : default
IsDefault       : True
State           : available
Tags            : {}
```

```
VpcId           : vpc-45678901
```

範例 3：此範例說明符合指定篩選器的 VPC (亦即，具有符合值 '10.0.0.0/16' 且狀態為「可用」狀態的 CIDR)。

```
Get-EC2Vpc -Filter @{Name="cidr";
 Values="10.0.0.0/16"},@{Name="state";Values="available"}
```

範例 4：此範例說明您的所有 VPC。

```
Get-EC2Vpc
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeVpcs式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
```

```
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def get_default_vpc(self):
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get default VPC: {err}")
        else:
            return response["Vpcs"][0]
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeVpcs中的 Python (博托 3) API 參考。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeVpnConnections配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVpnConnections。

CLI

AWS CLI

範例 1：描述您的 VPN 連線

下列describe-vpn-connections範例說明您所有的 Site-to-Site VPN 連線。

```
aws ec2 describe-vpn-connections
```

輸出：

```
{
    "VpnConnections": [
        {
            "CustomerGatewayConfiguration": "...configuration information...",
            "CustomerGatewayId": "cgw-01234567abcde1234",
            "Category": "VPN",
            "State": "available",
            "Type": "ipsec.1",
            "VpnConnectionId": "vpn-1122334455aabbccd",
            "TransitGatewayId": "tgw-00112233445566aab",
            "Options": {
                "EnableAcceleration": false,
                "StaticRoutesOnly": true,
                "LocalIpv4NetworkCidr": "0.0.0.0/0",
                "RemoteIpv4NetworkCidr": "0.0.0.0/0",
                "TunnelInsideIpVersion": "ipv4"
            },
            "Routes": [],
            "Tags": [
```

```
                {
                    "Key": "Name",
                    "Value": "CanadaVPN"
                }
            ],
            "VgwTelemetry": [
                {
                    "AcceptedRouteCount": 0,
                    "LastStatusChange": "2020-07-29T10:35:11.000Z",
                    "OutsideIpAddress": "203.0.113.3",
                    "Status": "DOWN",
                    "StatusMessage": ""
                },
                {
                    "AcceptedRouteCount": 0,
                    "LastStatusChange": "2020-09-02T09:09:33.000Z",
                    "OutsideIpAddress": "203.0.113.5",
                    "Status": "UP",
                    "StatusMessage": ""
                }
            ]
        }
    ]
}
```

如需詳細資訊，請參閱 AWS 網站間 VPN 使用者指南中的 AWS Site-to-Site VPN 運作方式。

範例 2：描述您可用的 VPN 連線

下列`describe-vpn-connections`範例說明狀態為的 Site-to-Site VPN 連線。`available`

```
aws ec2 describe-vpn-connections \
    --filters "Name=state,Values=available"
```

如需詳細資訊，請參閱 AWS 網站間 VPN 使用者指南中的 AWS Site-to-Site VPN 運作方式。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeVpnConnections中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的 VPN 連線。

```
Get-EC2VpnConnection -VpnConnectionId vpn-12345678
```

輸出：

```
CustomerGatewayConfiguration : [XML document]
CustomerGatewayId            : cgw-1a2b3c4d
Options                      : Amazon.EC2.Model.VpnConnectionOptions
Routes                       : {Amazon.EC2.Model.VpnStaticRoute}
State                        : available
Tags                         : {}
Type                         : ipsec.1
VgwTelemetry                 : {Amazon.EC2.Model.VgwTelemetry,
 Amazon.EC2.Model.VgwTelemetry}
VpnConnectionId              : vpn-12345678
VpnGatewayId                 : vgw-1a2b3c4d
```

範例 2：此範例說明任何狀態為擱置中或可用的 VPN 連線。

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnConnection -Filter $filter
```

範例 3：此範例說明您所有的 VPN 連線。

```
Get-EC2VpnConnection
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeVpnConnections式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeVpnGateways配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVpnGateways。

CLI

AWS CLI

描述您的虛擬私有閘道

此範例說明您的虛擬私有閘道。

命令：

```
aws ec2 describe-vpn-gateways
```

輸出：

```
{
    "VpnGateways": [
        {
            "State": "available",
            "Type": "ipsec.1",
            "VpnGatewayId": "vgw-f211f09b",
            "VpcAttachments": [
                {
                    "State": "attached",
                    "VpcId": "vpc-98eb5ef5"
                }
            ]
        },
        {
            "State": "available",
            "Type": "ipsec.1",
            "VpnGatewayId": "vgw-9a4cacf3",
            "VpcAttachments": [
                {
                    "State": "attaching",
                    "VpcId": "vpc-a01106c2"
                }
            ]
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DescribeVpnGateways](#)中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例說明指定的虛擬私有閘道。

```
Get-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

輸出：

```
AvailabilityZone :
State            : available
Tags             : {}
Type             : ipsec.1
VpcAttachments   : {vpc-12345678}
VpnGatewayId     : vgw-1a2b3c4d
```

範例 2：此範例說明任何狀態為擱置中或可用的虛擬私有閘道。

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnGateway -Filter $filter
```

範例 3：此範例說明您所有的虛擬私有閘道。

```
Get-EC2VpnGateway
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeVpnGateways式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DetachInternetGateway配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DetachInternetGateway。

CLI

AWS CLI

從 VPC 中分離網際網路閘道

下列detach-internet-gateway範例會將指定的網際網路閘道與特定 VPC 分離。

```
aws ec2 detach-internet-gateway \
    --internet-gateway-id igw-0d0fb496b3EXAMPLE \
    --vpc-id vpc-0a60eb65b4EXAMPLE
```

此命令不會產生輸出。

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的網際網路閘道。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DetachInternetGateway中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會將指定的網際網路閘道從指定的 VPC 卸離。

```
Dismount-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d -VpcId vpc-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DetachInternetGateway式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DetachNetworkInterface配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DetachNetworkInterface。

CLI

AWS CLI

從執行個體中分離網路介面

此範例會從指定的執行個體分離指定的網路介面。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 detach-network-interface --attachment-id eni-attach-66c4350a
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DetachNetworkInterface中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例會移除網路介面與執行個體之間的指定附件。

```
Dismount-EC2NetworkInterface -AttachmentId eni-attach-1a2b3c4d -Force
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DetachNetworkInterface式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DetachVolume配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DetachVolume。

CLI

AWS CLI

從執行個體卸離磁碟區

此範例指令會將磁碟區 (vol-049df61146c4d7901) 與其所連接的例證分離出來。

命令：

```
aws ec2 detach-volume --volume-id vol-1234567890abcdef0
```

輸出：

```
{
    "AttachTime": "2014-02-27T19:23:06.000Z",
    "InstanceId": "i-1234567890abcdef0",
    "VolumeId": "vol-049df61146c4d7901",
    "State": "detaching",
    "Device": "/dev/sdb"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DetachVolume中的。

PowerShell

用於的工具 PowerShell

範例 1：此範例分離指定的磁碟區。

```
Dismount-EC2Volume -VolumeId vol-12345678
```

輸出：

```
AttachTime          : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device              : /dev/sdh
InstanceId          : i-1a2b3c4d
State               : detaching
VolumeId            : vol-12345678
```

範例 2：您也可以指定執行個體 ID 和裝置名稱，以確保分離正確的磁碟區。

```
Dismount-EC2Volume -VolumeId vol-12345678 -InstanceId i-1a2b3c4d -Device /dev/sdh
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DetachVolume式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DetachVpnGateway配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DetachVpnGateway。

CLI

AWS CLI

將虛擬私有閘道從 VPC 中斷連結

此範例會將指定的虛擬私人閘道從指定的 VPC 卸離。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 detach-vpn-gateway --vpn-gateway-id vgw-9a4cacf3 --vpc-id vpc-a01106c2
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DetachVpnGateway中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的虛擬私人閘道從指定的 VPC 卸離。

```
Dismount-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d -VpcId vpc-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DetachVpnGateway式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DisableVgwRoutePropagation配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DisableVgwRoutePropagation。

CLI

AWS CLI

停用路由傳輸

此範例會停用指定的虛擬私人閘道，將靜態路由傳播至指定的路由資料表。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 disable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-id
 vgw-9a4cacf3
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DisableVgwRoutePropagation中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例停用 VGW 將路由自動傳輸至指定的路由表。

```
Disable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DisableVgwRoutePropagation式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DisableVpcClassicLink配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DisableVpcClassicLink。

CLI

AWS CLI

若要停 ClassicLink 用 VPC

此範例會停 ClassicLink 用虛擬電腦。

命令：

```
aws ec2 disable-vpc-classic-link --vpc-id vpc-88888888
```

輸出：

```
{
  "Return": true
```

```
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DisableVpcClassicLink中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會停用 vpc -01e23c4a VpcClassicLink 5d6db78e9 的 EC2。它返回真或假

```
Disable-EC2VpcClassicLink -VpcId vpc-01e23c4a5d6db78e9
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DisableVpcClassicLink式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DisableVpcClassicLinkDnsSupport配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DisableVpcClassicLinkDnsSupport。

CLI

AWS CLI

若要停用 VPC 雲端的 ClassicLink DNS 支援

此範例會停用的 ClassicLink DNS 支援vpc-88888888。

命令：

```
aws ec2 disable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

輸出：

```
{
  "Return": true
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DisableVpcClassicLinkDnsSupport中的。

PowerShell

適用的工具 PowerShell

範例 1：這個範例會停用對虛擬電腦的 ClassicLink DNS 支援

```
Disable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DisableVpcClassicLinkDnsSupport式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DisassociateAddress配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DisassociateAddress。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Disassociate an Elastic IP address from an EC2 instance.
    /// </summary>
    /// <param name="associationId">The association Id.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DisassociateIp(string associationId)
    {
```

```
        var response = await _amazonEC2.DisassociateAddressAsync(
            new DisassociateAddressRequest { AssociationId = associationId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DisassociateAddress中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
###############################################################################
# function ec2_disassociate_address
#
# This function disassociates an Elastic IP address from an Amazon Elastic
 Compute Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a association_id - The association ID that represents the association of
 the Elastic IP address with an instance.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_disassociate_address() {
  local association_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_disassociate_address"
    echo "Disassociates an Elastic IP address from an Amazon Elastic Compute
 Cloud (Amazon EC2) instance."
```

```
    echo "  -a association_id - The association ID that represents the
 association of the Elastic IP address with an instance."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
      a) association_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$association_id" ]]; then
    errecho "ERROR: You must provide an association ID with the -a parameter."
    return 1
  fi

  response=$(aws ec2 disassociate-address \
    --association-id "$association_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports disassociate-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}
```

此範例中使用的公用程式函數。

```
###############################################################################
```

```
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[DisassociateAddress](#)中的。

CLI

AWS CLI

在 EC2-Classic 中取消彈性 IP 地址的關聯

此範例會在 EC2-Classic 中取消彈性 IP 地址與執行個體的關聯。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 disassociate-address --public-ip 198.51.100.0
```

在 EC2-VPC 中取消彈性 IP 地址的關聯

此範例會在 VPC 中取消彈性 IP 地址與執行個體的關聯。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 disassociate-address --association-id eipassoc-2bebb745
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DisassociateAddress中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    public static void disassociateAddress(Ec2Client ec2, String associationId) {
        try {
            DisassociateAddressRequest addressRequest =
DisassociateAddressRequest.builder()
                    .associationId(associationId)
                    .build();
```

```
            ec2.disassociateAddress(addressRequest);
            System.out.println("You successfully disassociated the address!");

        } catch (Ec2Exception 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DisassociateAddress中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { DisassociateAddressCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// Disassociate an Elastic IP address from an instance.
export const main = async () => {
  const command = new DisassociateAddressCommand({
    // You can also use PublicIp, but that is for EC2 classic which is being
 retired.
    AssociationId: "ASSOCIATION_ID",
  });

  try {
    await client.send(command);
    console.log("Successfully disassociated address");
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考DisassociateAddress中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun disassociateAddressSc(associationIdVal: String?) {
    val addressRequest =
        DisassociateAddressRequest {
            associationId = associationIdVal
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.disassociateAddress(addressRequest)
        println("You successfully disassociated the address!")
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK DisassociateAddress中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會取消指定彈性 IP 位址與 VPC 中指定執行個體的關聯。

```
Unregister-EC2Address -AssociationId eipassoc-12345678
```

範例 2：此範例會取消 EC2-Classic 中指定彈性 IP 位址與指定執行個體的關聯。

```
Unregister-EC2Address -PublicIp 203.0.113.17
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DisassociateAddress式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
 actions."""

    def __init__(self, ec2_resource, elastic_ip=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param elastic_ip: A Boto3 VpcAddress object. This is a high-level object
 that
                           wraps Elastic IP actions.
        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def disassociate(self):
        """
        Removes an association between an Elastic IP address and an instance.
 When the
        association is removed, the instance is assigned a new public IP address.
```

```
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to disassociate.")
            return

        try:
            self.elastic_ip.association.delete()
        except ClientError as err:
            logger.error(
                "Couldn't disassociate Elastic IP %s from its instance. Here's
why: %s: %s",
                self.elastic_ip.allocation_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DisassociateAddress中的 Python (博托 3) API 參考。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DisassociateRouteTable配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DisassociateRouteTable。

CLI

AWS CLI

取消路由表的關聯

此範例會取消指定路由表與指定子網路的關聯。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 disassociate-route-table --association-id rtbassoc-781d0d1a
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DisassociateRouteTable中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會移除路由表和子網路之間的指定關聯。

```
Unregister-EC2RouteTable -AssociationId rtbassoc-1a2b3c4d
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DisassociateRouteTable式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭EnableVgwRoutePropagation配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用EnableVgwRoutePropagation。

CLI

AWS CLI

啟用路由傳輸

此範例可讓指定的虛擬私有閘道將靜態路由傳播至指定的路由表。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 enable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-id
 vgw-9a4cacf3
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考EnableVgwRoutePropagation中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例可讓指定的 VGW 自動將路由傳播至指定的路由表。

```
Enable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程EnableVgwRoutePropagation式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭EnableVolumeIo配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用EnableVolumeIo。

CLI

AWS CLI

若要啟用磁碟區的 I/O

此範例會在磁碟區上啟用 I/O vol-1234567890abcdef0。

命令：

```
aws ec2 enable-volume-io --volume-id vol-1234567890abcdef0
```

輸出：

```
{
  "Return": true
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考EnableVolumeIo中的。

PowerShell

適用的工具 PowerShell

範例 1：如果停用 I/O 作業，此範例會啟用指定磁碟區的 I/O 作業。

```
Enable-EC2VolumeIO -VolumeId vol-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程EnableVolumeIo式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭EnableVpcClassicLink配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用EnableVpcClassicLink。

CLI

AWS CLI

若要啟用虛 VPC ClassicLink

這個範例會啟用 vpc -88888。 ClassicLink

命令：

```
aws ec2 enable-vpc-classic-link --vpc-id vpc-88888888
```

輸出：

```
{
  "Return": true
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考EnableVpcClassicLink中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會針對下列項目啟用虛擬私家電腦 vpc -0123456b789b0d12f ClassicLink

```
Enable-EC2VpcClassicLink -VpcId vpc-0123456b789b0d12f
```

輸出：

```
True
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程EnableVpcClassicLink式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭EnableVpcClassicLinkDnsSupport配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用EnableVpcClassicLinkDnsSupport。

CLI

AWS CLI

若要啟用 VPC 雲端的 ClassicLink DNS 支援

此範例會啟用的 ClassicLink DNS 支援vpc-88888888。

命令：

```
aws ec2 enable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

輸出：

```
{
  "Return": true
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考EnableVpcClassicLinkDnsSupport中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例可讓 vpc -0b12d3456a7e8910d 支援下列項目的 DNS 主機名稱解析 ClassicLink

```
Enable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程EnableVpcClassicLinkDnsSupport式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭GetConsoleOutput配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用GetConsoleOutput。

CLI

AWS CLI

示例 1：獲取控制台輸出

下列get-console-output範例會取得指定 Linux 執行個體的主控台輸出。

```
aws ec2 get-console-output \
    --instance-id i-1234567890abcdef0
```

輸出：

```
{
    "InstanceId": "i-1234567890abcdef0",
    "Timestamp": "2013-07-25T21:23:53.000Z",
    "Output": "..."
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的執行個體主控台輸出。

範例 2：取得最新的主控台輸出

下列get-console-output範例會取得指定 Linux 執行個體的最新主控台輸出。

```
aws ec2 get-console-output \
    --instance-id i-1234567890abcdef0 \
    --latest \
    --output text
```

輸出：

```
i-1234567890abcdef0 [    0.000000] Command line: root=LABEL=/ console=tty1
 console=ttyS0 selinux=0 nvme_core.io_timeout=4294967295
[    0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point
 registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
```

```
[    0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
...
Cloud-init v. 0.7.6 finished at Wed, 09 May 2018 19:01:13 +0000. Datasource
 DataSourceEc2.  Up 21.50 seconds
Amazon Linux AMI release 2018.03
Kernel 4.14.26-46.32.amzn1.x
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的執行個體主控台輸出。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考GetConsoleOutput中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會取得指定 Linux 執行個體的主控台輸出。控制台輸出被編碼。

```
Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456
```

輸出：

```
InstanceId          Output
----------          ------
i-0e194d3c47c123637 WyAgICAwLjAwMDAwMF0gQ29tbW...bGU9dHR5UzAgc2Vs
```

範例 2：此範例會將編碼的主控台輸出儲存在變數中，然後將其解碼。

```
$Output_encoded = (Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456).Output
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($Output_encoded
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程GetConsoleOutput式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭GetHostReservationPurchasePreview配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用GetHostReservationPurchasePreview。

CLI

AWS CLI

取得專用主機保留區的購買預覽

此範例提供您帳戶中指定專用主機之指定專用主機保留項目的成本預覽。

命令：

```
aws ec2 get-host-reservation-purchase-preview --offering-id hro-03f707bf363b6b324
 --host-id-set h-013abcd2a00cbd123
```

輸出：

```
{
  "TotalHourlyPrice": "1.499",
  "Purchase": [
      {
          "HourlyPrice": "1.499",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "HostIdSet": [
              "h-013abcd2a00cbd123"
          ],
          "UpfrontPrice": "0.000",
          "Duration": 31536000
      }
  ],
  "TotalUpfrontPrice": "0.000"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考GetHostReservationPurchasePreview中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會預覽保留項目購買，其組態與您的專用主機 h-01e23f4cd567890f1 相符的組態

```
Get-EC2HostReservationPurchasePreview -OfferingId hro-0c1f23456789d0ab -HostIdSet
 h-01e23f4cd567890f1
```

輸出：

```
CurrencyCode Purchase TotalHourlyPrice TotalUpfrontPrice
------------ -------- ---------------- -----------------
             {}       1.307            0.000
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程GetHostReservationPurchasePreview式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭GetPasswordData配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用GetPasswordData。

CLI

AWS CLI

取得加密的密碼

此範例會取得加密的密碼。

命令：

```
aws ec2 get-password-data --instance-id i-1234567890abcdef0
```

輸出：

```
{
    "InstanceId": "i-1234567890abcdef0",
    "Timestamp": "2013-08-07T22:18:38.000Z",
    "PasswordData": "gSlJFq+VpcZXqy+iktxMF6NyxQ4qCrT4+gaOuNOenX1MmgXPTj7XEXAMPLE
UQ+YeFfb+L1U4C4AKv652Ux1iRB3CPTYP7WmU3TUnhsuBd+p6LVk7T2lKUml6OXbk6WPW1VYYm/TRPB1
e1DQ7PY4an/DgZT4mwcpRFigzhniQgDDeO1InvSDcwoUTwNs0Y1S8ouri2W4n5GNlriM3Q0AnNVelVz/
53TkDtxbNoU606M1gK9zUWSxqEgwvbV2j8c5rP0WCuaMWSFl4ziDu4bd7q+4RSyi8NUsVWnKZ4aEZffu
DPGzKrF5yLlf3etP2L4ZR6CvG7K1hx7VKOQVN32Dajw=="
```

```
}
```

取得解密的密碼

此範例會取得解密的密碼。

命令：

```
aws ec2 get-password-data --instance-id  i-1234567890abcdef0 --priv-launch-key C:
\Keys\MyKeyPair.pem
```

輸出：

```
{
    "InstanceId": "i-1234567890abcdef0",
    "Timestamp": "2013-08-30T23:18:05.000Z",
    "PasswordData": "&ViJ652e*u"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考GetPasswordData中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會解密 Amazon EC2 指派給指定 Windows 執行個體之管理員帳戶的密碼。指定 pem 檔案時，會自動假設-Decrypto 參數的設定。

```
Get-EC2PasswordData -InstanceId i-12345678 -PemFile C:\path\my-key-pair.pem
```

輸出：

```
mYZ(PA9?C)Q
```

範例 2：(PowerShell 僅限 Windows) 檢查執行個體以判斷用來啟動執行個體的金鑰配對名稱，然後嘗試在 Visual Studio 的 AWS Toolkit 的組態存放區中尋找對應的金鑰配對資料。如果找到金鑰配對資料，則會解密密碼。

```
Get-EC2PasswordData -InstanceId i-12345678 -Decrypt
```

輸出：

```
mYZ(PA9?C)Q
```

範例 3：傳回執行個體的加密密碼資料。

```
Get-EC2PasswordData -InstanceId i-12345678
```

輸出：

```
iVz3BAK/WAXV.....dqt8WeMA==
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程GetPasswordData式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ImportImage配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ImportImage。

CLI

AWS CLI

將虛擬機器映像檔匯入為 AMI

下列import-image範例會匯入指定的 OVA。

```
aws ec2 import-image \
  --disk-containers Format=ova,UserBucket="{S3Bucket=my-import-bucket,S3Key=vms/
my-server-vm.ova}"
```

輸出：

```
{
    "ImportTaskId": "import-ami-1234567890abcdef0",
```

```
    "Progress": "2",
    "SnapshotDetails": [
        {
            "DiskImageSize": 0.0,
            "Format": "ova",
            "UserBucket": {
                "S3Bucket": "my-import-bucket",
                "S3Key": "vms/my-server-vm.ova"
            }
        }
    ],
    "Status": "active",
    "StatusMessage": "pending"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ImportImage中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例將單一磁碟虛擬機器映像從指定的 Amazon S3 儲存貯體匯入具有冪等權杖的 Amazon EC2。此範例要求存在預設名稱為 'vmimport' 的 VM Import 服務角色，其政策允許 Amazon EC2 存取指定儲存貯體，如 VM Import 預測網站主題中所述。若要使用自訂角色，請使用**-RoleName**參數指定角色名稱。

```
$container = New-Object Amazon.EC2.Model.ImageDiskContainer
$container.Format="VMDK"
$container.UserBucket = New-Object Amazon.EC2.Model.UserBucket
$container.UserBucket.S3Bucket = "myVirtualMachineImages"
$container.UserBucket.S3Key = "Win_2008_Server_Standard_SP2_64-bit-disk1.vmdk"

$parms = @{
    "ClientToken"="idempotencyToken"
    "Description"="Windows 2008 Standard Image Import"
    "Platform"="Windows"
    "LicenseType"="AWS"
}

Import-EC2Image -DiskContainer $container @parms
```

輸出：

```
Architecture    :
Description     : Windows 2008 Standard Image
Hypervisor      :
ImageId         :
ImportTaskId    : import-ami-abcdefgh
LicenseType     : AWS
Platform        : Windows
Progress        : 2
SnapshotDetails : {}
Status          : active
StatusMessage   : pending
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ImportImage式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**ImportKeyPair**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ImportKeyPair。

CLI

AWS CLI

匯入公開金鑰

首先，使用您選擇的工具生成 key pair。例如，使用這個 SSH 凱基命令：

命令：

```
ssh-keygen -t rsa -C "my-key" -f ~/.ssh/my-key
```

輸出：

```
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ec2-user/.ssh/my-key.
Your public key has been saved in /home/ec2-user/.ssh/my-key.pub.
```

```
...
```

這個範例命令會匯入指定的公開金鑰。

命令：

```
aws ec2 import-key-pair --key-name "my-key" --public-key-material fileb://~/.ssh/
my-key.pub
```

輸出：

```
{
  "KeyName": "my-key",
  "KeyFingerprint": "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ImportKeyPair中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例將公開金鑰匯入 EC2。第一行會將公開金鑰檔案 (*.pub) 的內容儲存在變數**$publickey**中。接下來，範例會將公開金鑰檔案的 UTF8 格式轉換為 Base64 編碼字串，並將轉換後的字串儲存在變數中。**$pkbase64**在最後一行中，轉換後的公鑰被導入到 EC2。指令程式會傳回金鑰指紋和名稱做為結果。

```
$publickey=[Io.File]::ReadAllText("C:\Users\TestUser\.ssh\id_rsa.pub")
$pkbase64 =
 [System.Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes($publickey))
Import-EC2KeyPair -KeyName Example-user-key -PublicKey $pkbase64
```

輸出：

```
KeyFingerprint                                  KeyName
--------------                                  -------
do:d0:15:8f:79:97:12:be:00:fd:df:31:z3:b1:42:z1 Example-user-key
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ImportKeyPair式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ImportSnapshot配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ImportSnapshot。

CLI

AWS CLI

匯入快照

下列import-snapshot範例會將指定的磁碟匯入為快照集。

```
aws ec2 import-snapshot \
    --description "My server VMDK" \
    --disk-container Format=VMDK,UserBucket={S3Bucket=my-import-bucket,S3Key=vms/
my-server-vm.vmdk}
```

輸出：

```
{
    "Description": "My server VMDK",
    "ImportTaskId": "import-snap-1234567890abcdef0",
    "SnapshotTaskDetail": {
        "Description": "My server VMDK",
        "DiskImageSize": "0.0",
        "Format": "VMDK",
        "Progress": "3",
        "Status": "active",
        "StatusMessage": "pending"
        "UserBucket": {
            "S3Bucket": "my-import-bucket",
            "S3Key": "vms/my-server-vm.vmdk"
        }
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ImportSnapshot中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將格式為「VMDK」的虛擬機器磁碟映像匯入 Amazon EBS 快照。此範例需要具有預設名稱 'vmimport' 的 VM Import 服務角色，其中包含允許 Amazon EC2 存取指定儲存貯體的政策，如 http://docs.aws.amazon.com/ AWSEC 2/ WindowsGuide 最新//VM .html **VM Import Prequisites** 主題中所述。ImportPrerequisites若要使用自訂角色，請使用**-RoleName**參數指定角色名稱。

```
$parms = @{
    "ClientToken"="idempotencyToken"
    "Description"="Disk Image Import"
    "DiskContainer_Description" = "Data disk"
    "DiskContainer_Format" = "VMDK"
    "DiskContainer_S3Bucket" = "myVirtualMachineImages"
    "DiskContainer_S3Key" = "datadiskimage.vmdk"
}

Import-EC2Snapshot @parms
```

輸出：

```
Description            ImportTaskId               SnapshotTaskDetail

-----------------      --------------------       ------------------

Disk Image Import      import-snap-abcdefgh
 Amazon.EC2.Model.SnapshotTaskDetail
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ImportSnapshot式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**ModifyCapacityReservation**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifyCapacityReservation。

CLI

AWS CLI

範例 1：變更現有容量保留所保留的執行個體數目

下列modify-capacity-reservation範例會變更容量保留保留容量的執行個體數目。

```
aws ec2 modify-capacity-reservation \
    --capacity-reservation-id cr-1234abcd56EXAMPLE \
    --instance-count 5
```

輸出：

```
{
    "Return": true
}
```

範例 2：變更現有產能保留的結束日期與時間

下列modify-capacity-reservation範例會修改現有的容量保留區，使其在指定的日期和時間結束。

```
aws ec2 modify-capacity-reservation \
    --capacity-reservation-id cr-1234abcd56EXAMPLE \
    --end-date-type limited \
    --end-date 2019-08-31T23:59:59Z
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的修改容量保留。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifyCapacityReservation中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將安裝計數變更為 1 來修改 CapacityReservationId cr-0c1f2345db6f7cdba

```
Edit-EC2CapacityReservation -CapacityReservationId cr-0c1f2345db6f7cdba -
InstanceCount 1
```

輸出：

```
True
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifyCapacityReservation式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifyHosts配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifyHosts。

CLI

AWS CLI

範例 1：啟用專用主機的自動放置功能

下列modify-hosts範例會為專用主機啟用自動放置功能，以便接受任何符合其執行個體類型組態的未鎖定目標執行個體啟動。

```
aws ec2 modify-hosts \
    --host-id h-06c2f189b4EXAMPLE \
    --auto-placement on
```

輸出：

```
{
    "Successful": [
        "h-06c2f189b4EXAMPLE"
    ],
    "Unsuccessful": []
}
```

範例 2：啟用專用主機的主機復原

下列modify-hosts範例會啟用指定之專用主機的主機復原。

```
aws ec2 modify-hosts \
```

```
    --host-id h-06c2f189b4EXAMPLE \
    --host-recovery on
```

輸出：

```
{
    "Successful": [
        "h-06c2f189b4EXAMPLE"
    ],
    "Unsuccessful": []
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端 Linux 執行個體使用者指南中的修改專用主機自動放置。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifyHosts中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例將專用主機的 AutoPlacement 設定修改為關閉

```
Edit-EC2Host -HostId h-03e09f8cd681609f3 -AutoPlacement off
```

輸出：

```
Successful            Unsuccessful
----------            ------------
{h-01e23f4cd567890f3} {}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifyHosts式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifyIdFormat配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifyIdFormat。

CLI

AWS CLI

若要啟用資源的較長 ID 格式

下列modify-id-format範例會啟用instance資源類型的較長 ID 格式。

```
aws ec2 modify-id-format \
    --resource instance \
    --use-long-ids
```

若要停用資源的較長 ID 格式

下列modify-id-format範例會停用instance資源類型的較長 ID 格式。

```
aws ec2 modify-id-format \
    --resource instance \
    --no-use-long-ids
```

下列modify-id-format範例會為選擇加入期間內的所有支援資源類型啟用較長的 ID 格式。

```
aws ec2 modify-id-format \
    --resource all-current \
    --use-long-ids
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifyIdFormat中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會為指定的資源類型啟用較長的 ID 格式。

```
Edit-EC2IdFormat -Resource instance -UseLongId $true
```

範例 2：此範例會停用指定資源類型的較長 ID 格式。

```
Edit-EC2IdFormat -Resource instance -UseLongId $false
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifyIdFormat式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifyImageAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifyImageAttribute。

CLI

AWS CLI

範例 1：公開 AMI

下列modify-instance-attribute範例會將指定的 AMI 設為公用。

```
aws ec2 modify-image-attribute \
    --image-id ami-5731123e \
    --launch-permission "Add=[{Group=all}]"
```

此命令不會產生輸出。

示例 2：將 AMI 設為私有

下列modify-instance-attribute範例會將指定的 AMI 設為私有。

```
aws ec2 modify-image-attribute \
    --image-id ami-5731123e \
    --launch-permission "Remove=[{Group=all}]"
```

此命令不會產生輸出。

範例 3：將啟動權限授與 AWS 帳戶

下列modify-instance-attribute範例會授與指定 AWS 帳戶的啟動權限。

```
aws ec2 modify-image-attribute \
    --image-id ami-5731123e \
```

```
    --launch-permission "Add=[{UserId=123456789012}]"
```

此命令不會產生輸出。

範例 4：從 AWS 帳戶移除啟動權限

下列modify-instance-attribute範例會從指定 AWS 帳戶移除啟動權限。

```
aws ec2 modify-image-attribute \
    --image-id ami-5731123e \
    --launch-permission "Remove=[{UserId=123456789012}]"
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifyImageAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會更新指定 AMI 的描述。

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Description "New description"
```

示例 2：此示例使 AMI 公開（例如，所 AWS 帳戶 以任何人都可以使用它）。

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType add -UserGroup all
```

範例 3：此範例會將 AMI 設為私有 (例如，只有擁有者的您才能使用)。

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType remove -UserGroup all
```

範例 4：此範例會將啟動權限授與指定的 AWS 帳戶。

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType add -UserId 111122223333
```

範例 5：此範例會移除指定的啟動權限 AWS 帳戶。

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType remove -UserId 111122223333
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifyImageAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifyInstanceAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifyInstanceAttribute。

CLI

AWS CLI

範例 1：修改例證類型

下列modify-instance-attribute範例會修改指定執行個體的執行個體類型。執行個體必須處於 stopped 狀態。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --instance-type "{\"Value\": \"m1.small\"}"
```

此命令不會產生輸出。

範例 2：在執行個體上啟用增強型聯網

下列modify-instance-attribute範例會為指定的執行個體啟用增強型聯網。執行個體必須處於 stopped 狀態。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --sriov-net-support simple
```

此命令不會產生輸出。

範例 3：修改 sourceDestCheck 屬性

下列modify-instance-attribute範例會將指定執行個體的sourceDestCheck屬性設定為true。執行個體必須位於 VPC 中。

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --source-
dest-check "{\"Value\": true}"
```

此命令不會產生輸出。

範例 4：修改根磁碟區的 deleteOnTermination 屬性

下列modify-instance-attribute範例會將指定 Amazon EBS 後端執行個體的根磁碟區deleteOnTermination屬性設定為。false依預設，此屬性適true用於根磁碟區。

命令：

```
aws ec2 modify-instance-attribute \
  --instance-id i-1234567890abcdef0 \
  --block-device-mappings "[{\"DeviceName\": \"/dev/sda1\",\"Ebs\":
{\"DeleteOnTermination\":false}}]"
```

此命令不會產生輸出。

範例 5：修改附加至執行個體的使用者資料

下列modify-instance-attribute範例會新增檔案的內容，UserData.txt做 UserData 為指定執行個體的。

原始檔案內容UserData.txt：

```
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

檔案的內容必須以 base64 編碼。第一個命令將文本文件轉換為 base64 並將其保存為新文件。

Linux /macOS 版本的命令：

```
base64 UserData.txt > UserData.base64.txt
```

此命令不會產生輸出。

視窗版本的命令：

```
certutil -encode UserData.txt tmp.b64 && findstr /v /c:- tmp.b64 >
 UserData.base64.txt
```

輸出：

```
Input Length = 67
Output Length = 152
CertUtil: -encode command completed successfully.
```

現在，您可以在下面的 CLI 命令中引用該文件：

```
aws ec2 modify-instance-attribute \
    --instance-id=i-09b5a14dbca622e76 \
    --attribute userData --value file://UserData.base64.txt
```

此命令不會產生輸出。

如需詳細資訊，請參閱 EC2 使用者指南中的使用者資料和 AWS CLI。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifyInstanceAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會修改指定執行個體的執行個體類型。

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -InstanceType m3.medium
```

範例 2：這個範例會指定「simple」作為單一根 I/O 虛擬化 (SR-IOV) 網路支援參數的值，以啟用指定執行個體的增強型聯網，-.. SriovNetSupport

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SriovNetSupport "simple"
```

範例 3：此範例會修改指定執行個體的安全性群組。執行個體必須位於 VPC 中。您必須指定每個安全群組的 ID，而不是名稱。

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -Group @( "sg-12345678",
 "sg-45678901" )
```

範例 4：此範例會針對指定的執行個體啟用 EBS I/O 最佳化。並非所有執行個體類型都可以使用此功能。使用 EBS 優化執行個體時需支付額外的使用費。

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -EbsOptimized $true
```

範例 5：此範例會啟用指定執行個體的來源/目標檢查。若要讓 NAT 執行個體執行 NAT，值必須是「錯誤」。

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SourceDestCheck $true
```

範例 6：此範例會停用指定執行個體的終止。

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -DisableApiTermination $true
```

範例 7：此範例會變更指定的執行處理，以便在執行個體啟動關閉時終止該執行個體。

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -
InstanceInitiatedShutdownBehavior terminate
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifyInstanceAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifyInstanceCreditSpecification配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifyInstanceCreditSpecification。

CLI

AWS CLI

修改執行處理 CPU 使用率的評分選項

此範例會將指定區域中指定執行個體的 CPU 使用率評分選項修改為「無限制」。有效的信用選項為「標準」和「無限制」。

命令：

```
aws ec2 modify-instance-credit-specification --instance-credit-specification
 "InstanceId=i-1234567890abcdef0,CpuCredits=unlimited"
```

輸出：

```
{
  "SuccessfulInstanceCreditSpecifications": [
    {
      "InstanceId": "i-1234567890abcdef0"
    }
  ],
  "UnsuccessfulInstanceCreditSpecifications": []
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifyInstanceCreditSpecification中的。

PowerShell

適用的工具 PowerShell

範例 1：這樣可以啟用 T2 無限制的積分，例如 i-01234567890 位元。

```
$Credit = New-Object -TypeName
 Amazon.EC2.Model.InstanceCreditSpecificationRequest
$Credit.InstanceId = "i-01234567890abcdef"
$Credit.CpuCredits = "unlimited"
Edit-EC2InstanceCreditSpecification -InstanceCreditSpecification $Credit
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifyInstanceCreditSpecification式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifyNetworkInterfaceAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifyNetworkInterfaceAttribute。

CLI

AWS CLI

修改網路介面的附件屬性

此範例指令會修改指定網路介面的attachment屬性。

命令：

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
attachment AttachmentId=eni-attach-43348162,DeleteOnTermination=false
```

修改網路介面的描述屬性

此範例指令會修改指定網路介面的description屬性。

命令：

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
description "My description"
```

若要修改網路介面的 GroupSet 屬性

此範例指令會修改指定網路介面的groupSet屬性。

命令：

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
groups sg-903004f8 sg-1a2b3c4d
```

修改網路介面的 sourceDestCheck 屬性

此範例指令會修改指定網路介面的sourceDestCheck屬性。

命令：

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
no-source-dest-check
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifyNetworkInterfaceAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會修改指定的網路介面，以便在終止時刪除指定的附件。

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -
Attachment_AttachmentId eni-attach-1a2b3c4d -Attachment_DeleteOnTermination $true
```

範例 2：此範例會修改指定網路介面的描述。

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Description
 "my description"
```

範例 3：此範例會修改指定網路介面的安全性群組。

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Groups
 sg-1a2b3c4d
```

範例 4：此範例會停用指定網路介面的來源/目標檢查。

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -
SourceDestCheck $false
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifyNetworkInterfaceAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifyReservedInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifyReservedInstances。

CLI

AWS CLI

修改預留執行個體

此範例命令會將預留執行個體移至相同區域中的另一個可用區域。

命令：

```
aws ec2 modify-reserved-instances --reserved-instances-ids b847fa93-
e282-4f55-b59a-1342f5bd7c02 --target-configurations AvailabilityZone=us-
west-1c,Platform=EC2-Classic,InstanceCount=10
```

輸出：

```
{
  "ReservedInstancesModificationId": "rimod-d3ed4335-b1d3-4de6-ab31-0f13aaf46687"
}
```

修改預留執行個體的網路平台

此範例命令會將 EC2 傳統預留執行個體轉換為 EC2-VPC。

命令：

```
aws ec2 modify-reserved-instances --reserved-instances-ids f127bd27-
edb7-44c9-a0eb-0d7e09259af0 --target-configurations AvailabilityZone=us-
west-1c,Platform=EC2-VPC,InstanceCount=5
```

輸出：

```
{
  "ReservedInstancesModificationId": "rimod-82fa9020-668f-4fb6-945d-61537009d291"
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的修改預留執行個體。

修改預留執行個體的執行個體大小

這個範例命令會修改在美國西部 1c 中具有 10 個 m1.small Linux/Unix 執行個體的預留執行個體，使 8 m1.small 執行個體變成 2 個 m1.large 執行個體，而其餘的 2 m1.small 會變成相同可用區域中的 1 m1.medium 執行個體。命令：

```
aws ec2 modify-reserved-instances --reserved-instances-
ids 1ba8e2e3-3556-4264-949e-63ee671405a9 --target-
configurations AvailabilityZone=us-west-1c,Platform=EC2-
Classic,InstanceCount=2,InstanceType=m1.large AvailabilityZone=us-
west-1c,Platform=EC2-Classic,InstanceCount=1,InstanceType=m1.medium
```

輸出：

```
{
    "ReservedInstancesModificationId": "rimod-acc5f240-080d-4717-
b3e3-1c6b11fa00b6"
}
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的修改保留的執行個體大小。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifyReservedInstances中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會修改指定預留執行個體的可用區域、執行個體計數和平台。

```
$config = New-Object Amazon.EC2.Model.ReservedInstancesConfiguration
$config.AvailabilityZone = "us-west-2a"
$config.InstanceCount = 1
$config.Platform = "EC2-VPC"

Edit-EC2ReservedInstance `
-ReservedInstancesId @("FE32132D-70D5-4795-B400-AE435EXAMPLE",
 "0CC556F3-7AB8-4C00-B0E5-98666EXAMPLE") `
-TargetConfiguration $config
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifyReservedInstances式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifySnapshotAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifySnapshotAttribute。

CLI

AWS CLI

範例 1：修改快照屬性

下列modify-snapshot-attribute範例會更新指定快照的createVolumePermission屬性，移除指定使用者的磁碟區權限。

```
aws ec2 modify-snapshot-attribute \
    --snapshot-id snap-1234567890abcdef0 \
    --attribute createVolumePermission \
    --operation-type remove \
    --user-ids 123456789012
```

範例 2：將快照集設為公開

下列modify-snapshot-attribute範例會將指定的快照集設為公用。

```
aws ec2 modify-snapshot-attribute \
    --snapshot-id snap-1234567890abcdef0 \
    --attribute createVolumePermission \
    --operation-type add \
    --group-names all
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifySnapshotAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會透過設定其 CreateVolumePermission 屬性，使指定的快照集成為公開狀態。

```
Edit-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
 CreateVolumePermission -OperationType Add -GroupName all
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifySnapshotAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifySpotFleetRequest配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifySpotFleetRequest。

CLI

AWS CLI

若要修改競價型叢集請求

此範例命令會更新指定競價型叢集請求的目標容量。

命令：

```
aws ec2 modify-spot-fleet-request --target-capacity 20 --spot-fleet-request-id
 sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

輸出：

```
{
    "Return": true
}
```

此範例命令會減少指定 Spot 叢集請求的目標容量，而不會因此終止任何 Spot 執行個體。

命令：

```
aws ec2 modify-spot-fleet-request --target-capacity 10 --excess-capacity-
termination-policy NoTermination --spot-fleet-request-ids sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

輸出：

```
{
    "Return": true
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifySpotFleetRequest中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會更新指定競價型叢集請求的目標容量。

```
Edit-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -TargetCapacity 10
```

輸出：

```
True
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifySpotFleetRequest式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifySubnetAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifySubnetAttribute。

CLI

AWS CLI

若要變更子網路的公用 IPv4 定址行為

此範例會修改子網路 1a2b3c4d，以指定所有啟動至此子網路的執行個體都會指派一個公用 IPv4 位址。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --map-public-ip-on-
launch
```

若要變更子網路的 IPv6 定址行為

此範例會修改子網路 1a2b3c4d，以指定所有啟動至此子網路的執行個體都會指派子網路範圍內的 IPv6 位址。

命令：

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --assign-ipv6-
address-on-creation
```

如需詳細資訊，請參閱AWS 虛擬私有雲使用者指南中的 VPC 中的 IP 定址。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifySubnetAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會為指定的子網路啟用公用 IP 位址。

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $true
```

範例 2：此範例會停用指定子網路的公用 IP 位址。

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $false
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifySubnetAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifyVolumeAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifyVolumeAttribute。

CLI

AWS CLI

修改體積塊屬性的步驟

此範例會vol-1234567890abcdef0將 ID 的磁碟區autoEnableIo屬性設定為true。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 modify-volume-attribute --volume-id vol-1234567890abcdef0 --auto-enable-
io
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifyVolumeAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會修改指定磁碟區的指定屬性。由於資料可能不一致，磁碟區的 I/O 作業會在暫停後自動恢復。

```
Edit-EC2VolumeAttribute -VolumeId vol-12345678 -AutoEnableIO $true
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifyVolumeAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ModifyVpcAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ModifyVpcAttribute。

CLI

AWS CLI

修改 enableDnsSupport 屬性的步驟

此範例會修改enableDnsSupport屬性。此屬性指出 VPC 是否已啟用 DNS 解析。如果此屬性為 true，Amazon DNS 伺服器會將您的執行個體的 DNS 主機名稱解析為對應的 IP 地址；否則將不會進行解析。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-support "{\"Value
\":false}"
```

修改 enableDnsHostnames 屬性的步驟

此範例會修改enableDnsHostnames屬性。此屬性指出在 VPC 中啟動的執行個體是否取得 DNS 主機名稱。如果此屬性為 true，該 VPC 中的執行個體會取得 DNS 主機名稱；否則將不會取得。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-hostnames
 "{\"Value\":false}"
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ModifyVpcAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例啟用對指定 VPC 之 DNS 主機名稱的支援。

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $true
```

範例 2：此範例停用對指定 VPC 之 DNS 主機名稱的支援。

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $false
```

範例 3：此範例啟用指定 VPC 的 DNS 解析支援。

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $true
```

範例 4：此範例停用對指定 VPC 之 DNS 解析的支援。

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $false
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ModifyVpcAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭MonitorInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用MonitorInstances。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::MonitorInstancesRequest request;
    request.AddInstanceIds(instanceId);
    request.SetDryRun(true);

    auto dry_run_outcome = ec2Client.MonitorInstances(request);
    if (dry_run_outcome.IsSuccess()) {
        std::cerr
                << "Failed dry run to enable monitoring on instance. A dry run
should trigger an error."
                <<
                std::endl;
        return false;
    }
    else if (dry_run_outcome.GetError().GetErrorType()
             != Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cerr << "Failed dry run to enable monitoring on instance " <<
                  instanceId << ": " << dry_run_outcome.GetError().GetMessage()
<<
                  std::endl;
        return false;
    }

    request.SetDryRun(false);
    auto monitorInstancesOutcome = ec2Client.MonitorInstances(request);
    if (!monitorInstancesOutcome.IsSuccess()) {
        std::cerr << "Failed to enable monitoring on instance " <<
                  instanceId << ": " <<
                  monitorInstancesOutcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully enabled monitoring on instance " <<
```

```
                instanceId << std::endl;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考MonitorInstances中的。

CLI

AWS CLI

啟用執行個體的詳細監控

此範例命令會啟用指定執行個體的詳細監控。

命令：

```
aws ec2 monitor-instances --instance-ids i-1234567890abcdef0
```

輸出：

```
{
  "InstanceMonitorings": [
      {
          "InstanceId": "i-1234567890abcdef0",
          "Monitoring": {
              "State": "pending"
          }
      }
  ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考MonitorInstances中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { MonitorInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// Turn on detailed monitoring for the selected instance.
// By default, metrics are sent to Amazon CloudWatch every 5 minutes.
// For a cost you can enable detailed monitoring which sends metrics every
 minute.
export const main = async () => {
  const command = new MonitorInstancesCommand({
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { InstanceMonitorings } = await client.send(command);
    const instancesBeingMonitored = InstanceMonitorings.map(
      (im) =>
        ` • Detailed monitoring state for ${im.InstanceId} is
 ${im.Monitoring.State}.`,
    );
    console.log("Monitoring status:");
    console.log(instancesBeingMonitored.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考MonitorInstances中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會啟用指定執行處理的詳細監視。

```
Start-EC2InstanceMonitoring -InstanceId i-12345678
```

輸出：

```
InstanceId    Monitoring
----------    ----------
i-12345678    Amazon.EC2.Model.Monitoring
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程MonitorInstances式參考中的。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
    APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
 lt_instance_ids.

    "Perform dry run"
    TRY.
        " DryRun is set to true. This checks for the required permissions to
 monitor the instance without actually making the request. "
        lo_ec2->monitorinstances(
          it_instanceids = lt_instance_ids
          iv_dryrun = abap_true
        ).
```

```
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        " If the error code returned is `DryRunOperation`, then you have the
 required permissions to monitor this instance. "
        IF lo_exception->av_err_code = 'DryRunOperation'.
          MESSAGE 'Dry run to enable detailed monitoring completed.' TYPE 'I'.
          " DryRun is set to false to enable detailed monitoring. "
          lo_ec2->monitorinstances(
            it_instanceids = lt_instance_ids
            iv_dryrun = abap_false
          ).
          MESSAGE 'Detailed monitoring enabled.' TYPE 'I'.
          " If the error code returned is `UnauthorizedOperation`, then you don't
 have the required permissions to monitor this instance. "
        ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
          MESSAGE 'Dry run to enable detailed monitoring failed. User does not
 have the permissions to monitor the instance.' TYPE 'E'.
        ELSE.
          DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
          MESSAGE lv_error TYPE 'E'.
        ENDIF.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK MonitorInstances中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭MoveAddressToVpc配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用MoveAddressToVpc。

CLI

AWS CLI

若要將地址移動到 EC2-VPC，請執行

此範例會將彈性 IP 位址 54.123.4.56 移至 EC2-VPC 平台。

命令：

```
aws ec2 move-address-to-vpc --public-ip 54.123.4.56
```

輸出：

```
{
  "Status": "MoveInProgress"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考MoveAddressToVpc中的。

PowerShell

適用的工具 PowerShell

範例 1：本範例將公有 IP 位址為 12.345.67.89 的 EC2 執行個體移至美國東部 (維吉尼亞北部) 區域的 EC2-VPC 平台。

```
Move-EC2AddressToVpc -PublicIp 12.345.67.89 -Region us-east-1
```

範例 2：此範例會將命令的結果Move-EC2AddressToVpc傳送至指Get-EC2Instance令程式。命Get-EC2Instance令會取得由執行個體 ID 指定的執行個體，然後傳回執行個體的公用 IP 位址屬性。

```
(Get-EC2Instance -Instance i-12345678).Instances.PublicIpAddress | Move-
EC2AddressToVpc
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程MoveAddressToVpc式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭PurchaseHostReservation配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用PurchaseHostReservation。

CLI

AWS CLI

購買專用主機保留

此範例會針對您帳戶中指定的專用主機購買指定的專用主機保留項目。

命令：

```
aws ec2 purchase-host-reservation --offering-id hro-03f707bf363b6b324 --host-id-
set h-013abcd2a00cbd123
```

輸出：

```
{
  "TotalHourlyPrice": "1.499",
  "Purchase": [
      {
          "HourlyPrice": "1.499",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "HostIdSet": [
              "h-013abcd2a00cbd123"
          ],
          "HostReservationId": "hr-0d418a3a4ffc669ae",
          "UpfrontPrice": "0.000",
          "Duration": 31536000
      }
  ],
  "TotalUpfrontPrice": "0.000"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考PurchaseHostReservation中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例購買保留項目提供的保留項目 Hro-0c1f23456789d0ab，其組態與您的專用主機相符的組態

```
New-EC2HostReservation -OfferingId hro-0c1f23456789d0ab HostIdSet
 h-01e23f4cd567890f1
```

輸出：

```
ClientToken       :
CurrencyCode      :
Purchase          : {hr-0123f4b5d67bedc89}
TotalHourlyPrice  : 1.307
TotalUpfrontPrice : 0.000
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程PurchaseHostReservation式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭PurchaseScheduledInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用PurchaseScheduledInstances。

CLI

AWS CLI

購買排程執行個體

此範例會購買排程執行個體。

命令：

```
aws ec2 purchase-scheduled-instances --purchase-requests file://purchase-
request.json
```

購買請求：

```
[
    {
        "PurchaseToken": "eyJ2IjoiMSIsInMiOjEsImMiOi...",
        "InstanceCount": 1
    }
```

```
]
```

輸出：

```
{
  "ScheduledInstanceSet": [
      {
          "AvailabilityZone": "us-west-2b",
          "ScheduledInstanceId": "sci-1234-1234-1234-1234-123456789012",
          "HourlyPrice": "0.095",
          "CreateDate": "2016-01-25T21:43:38.612Z",
          "Recurrence": {
              "OccurrenceDaySet": [
                  1
              ],
              "Interval": 1,
              "Frequency": "Weekly",
              "OccurrenceRelativeToEnd": false,
              "OccurrenceUnit": ""
          },
          "Platform": "Linux/UNIX",
          "TermEndDate": "2017-01-31T09:00:00Z",
          "InstanceCount": 1,
          "SlotDurationInHours": 32,
          "TermStartDate": "2016-01-31T09:00:00Z",
          "NetworkPlatform": "EC2-VPC",
          "TotalScheduledInstanceHours": 1696,
          "NextSlotStartTime": "2016-01-31T09:00:00Z",
          "InstanceType": "c4.large"
      }
  ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考PurchaseScheduledInstances中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例購買排程執行個體。

```
$request = New-Object Amazon.EC2.Model.PurchaseRequest
```

```
$request.InstanceCount = 1
$request.PurchaseToken = "eyJ2IjoiMSIsInMiOjEsImMiOi..."
New-EC2ScheduledInstancePurchase -PurchaseRequest $request
```

輸出：

```
AvailabilityZone            : us-west-2b
CreateDate                  : 1/25/2016 1:43:38 PM
HourlyPrice                 : 0.095
InstanceCount               : 1
InstanceType                : c4.large
NetworkPlatform             : EC2-VPC
NextSlotStartTime           : 1/31/2016 1:00:00 AM
Platform                    : Linux/UNIX
PreviousSlotEndTime         :
Recurrence                  : Amazon.EC2.Model.ScheduledInstanceRecurrence
ScheduledInstanceId         : sci-1234-1234-1234-1234-123456789012
SlotDurationInHours         : 32
TermEndDate                 : 1/31/2017 1:00:00 AM
TermStartDate               : 1/31/2016 1:00:00 AM
TotalScheduledInstanceHours : 1696
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程PurchaseScheduledInstances式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭RebootInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用RebootInstances。

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：

- 建置及管理彈性服務

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Reboot EC2 instances.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the instances that will be
rebooted.</param>
    /// <returns>Async task.</returns>
    public async Task RebootInstances(string ec2InstanceId)
    {
        var request = new RebootInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        var response = await _amazonEC2.RebootInstancesAsync(request);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine("Instances successfully rebooted.");
        }
        else
        {
            Console.WriteLine("Could not reboot one or more instances.");
        }
    }
```

取代執行個體的設定檔，重新開機，然後重新啟動 Web 伺服器。

```
    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
```

```
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
    {
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
        // Allow time before resetting.
        Thread.Sleep(25000);
        var instanceReady = false;
        var retries = 5;
        while (retries-- > 0 && !instanceReady)
        {
            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(10000);

            var instancesPaginator =
_amazonSsm.Paginators.DescribeInstanceInformation(
                new DescribeInstanceInformationRequest());
            // Get the entire list using the paginator.
            await foreach (var instance in
instancesPaginator.InstanceInformationList)
            {
                instanceReady = instance.InstanceId == instanceId;
                if (instanceReady)
                {
                    break;
                }
            }
        }
```

```
        Console.WriteLine($"Sending restart command to instance {instanceId}");
        await _amazonSsm.SendCommandAsync(
            new SendCommandRequest()
            {
                InstanceIds = new List<string>() { instanceId },
                DocumentName = "AWS-RunShellScript",
                Parameters = new Dictionary<string, List<string>>()
                {
                        {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
                }
            });
        Console.WriteLine($"Restarted the web server on instance {instanceId}");
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考RebootInstances中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::RebootInstancesRequest request;
    request.AddInstanceIds(instanceId);
    request.SetDryRun(true);

    auto dry_run_outcome = ec2Client.RebootInstances(request);
    if (dry_run_outcome.IsSuccess()) {
        std::cerr
                << "Failed dry run to reboot on instance. A dry run should
trigger an error."
                <<
                std::endl;
        return false;
```

```
    }
    else if (dry_run_outcome.GetError().GetErrorType()
             != Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to reboot instance " << instanceId << ": "
                  << dry_run_outcome.GetError().GetMessage() << std::endl;
        return false;
    }

    request.SetDryRun(false);
    auto outcome = ec2Client.RebootInstances(request);
    if (!outcome.IsSuccess()) {
        std::cout << "Failed to reboot instance " << instanceId << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully rebooted instance " << instanceId <<
                  std::endl;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考RebootInstances中的。

CLI

AWS CLI

重新啟動 Amazon EC2 執行個體

此範例會重新啟動指定執行個體。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 reboot-instances --instance-ids i-1234567890abcdef5
```

如需詳細資訊，請參閱《Amazon Elastic Compute Cloud 使用者指南》中的「重新啟動您的執行個體」。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考RebootInstances中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { RebootInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new RebootInstancesCommand({
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    await client.send(command);
    console.log("Instance rebooted successfully.");
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考RebootInstances中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會重新啟動指定的執行個體。

```
Restart-EC2Instance -InstanceId i-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程RebootInstances式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
```

```
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def replace_instance_profile(
        self, instance_id, new_instance_profile_name, profile_association_id
    ):
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to update.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)
            inst_ready = False
            tries = 0
```

```
            while not inst_ready:
                if tries % 6 == 0:
                    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                    log.info(
                        "Rebooting instance %s and waiting for it to to be
ready.",
                        instance_id,
                    )
                tries += 1
                time.sleep(10)
                response = self.ssm_client.describe_instance_information()
                for info in response["InstanceInformationList"]:
                    if info["InstanceId"] == instance_id:
                        inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件RebootInstances中的 Python (博托 3) API 參考。

Rust

適用於 Rust 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
async fn reboot_instance(client: &Client, id: &str) -> Result<(), Error> {
    println!("Rebooting instance.");

    client.reboot_instances().instance_ids(id).send().await?;

    client
        .wait_until_instance_stopped()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await?;
    let wait_status_ok = client
        .wait_until_instance_status_ok()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await;

    match wait_status_ok {
        Ok(_) => println!("Rebooted instance {id}, it is started with status
 OK."),
        Err(err) => return Err(err.into()),
    }

    Ok(())
}
```

- 如需 API 的詳細資訊，請參閱 AWS SDK RebootInstances中的 Rust API 參考資料。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
```

```
    APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
 lt_instance_ids.

    "Perform dry run"
    TRY.
        " DryRun is set to true. This checks for the required permissions to
 reboot the instance without actually making the request. "
        lo_ec2->rebootinstances(
          it_instanceids = lt_instance_ids
          iv_dryrun = abap_true
        ).
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        " If the error code returned is `DryRunOperation`, then you have the
 required permissions to reboot this instance. "
        IF lo_exception->av_err_code = 'DryRunOperation'.
          MESSAGE 'Dry run to reboot instance completed.' TYPE 'I'.
          " DryRun is set to false to make a reboot request. "
          lo_ec2->rebootinstances(
             it_instanceids = lt_instance_ids
             iv_dryrun = abap_false
           ).
          MESSAGE 'Instance rebooted.' TYPE 'I'.
          " If the error code returned is `UnauthorizedOperation`, then you don't
 have the required permissions to reboot this instance. "
        ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
          MESSAGE 'Dry run to reboot instance failed. User does not have
 permissions to reboot the instance.' TYPE 'E'.
        ELSE.
          DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
          MESSAGE lv_error TYPE 'E'.
        ENDIF.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK RebootInstances中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭RegisterImage配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用RegisterImage。

CLI

AWS CLI

示例 1：使用清單文件註冊 AMI

下列register-image範例會使用 Amazon S3 中指定的資訊清單檔案註冊 AMI。

```
aws ec2 register-image \
    --name my-image \
    --image-location my-s3-bucket/myimage/image.manifest.xml
```

輸出：

```
{
    "ImageId": "ami-1234567890EXAMPLE"
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon Machine Images (AMI)。

範例 2：使用根裝置的快照註冊 AMI

下列register-image範例會使用指定的 EBS 根磁碟區快照作為裝置/dev/xvda來註冊 AMI。該區塊裝置對應還包括一個空的 100 GiB EBS 磁碟區做為裝置。/dev/xvdf

```
aws ec2 register-image \
    --name my-image \
    --root-device-name /dev/xvda \
    --block-device-mappings DeviceName=/dev/
xvda,Ebs={SnapshotId=snap-0db2cf683925d191f} DeviceName=/dev/
xvdf,Ebs={VolumeSize=100}
```

輸出：

```
{
    "ImageId": "ami-1a2b3c4d5eEXAMPLE"
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon Machine Images (AMI)。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考RegisterImage中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例使用 Amazon S3 中指定的資訊清單檔案註冊 AMI。

```
Register-EC2Image -ImageLocation my-s3-bucket/my-web-server-ami/
image.manifest.xml -Name my-web-server-ami
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程RegisterImage式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭RejectVpcPeeringConnection配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用RejectVpcPeeringConnection。

CLI

AWS CLI

拒絕 VPC 對等連線

此範例拒絕指定的 VPC 對等連線要求。

命令：

```
aws ec2 reject-vpc-peering-connection --vpc-peering-connection-id pcx-1a2b3c4d
```

輸出：

```
{
    "Return": true
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考RejectVpcPeeringConnection中的。

PowerShell

適用的工具 PowerShell

範例 1：上述範例拒絕請 VpcPeering 求識別碼

```
Deny-EC2VpcPeeringConnection -VpcPeeringConnectionId pcx-01a2b3ce45fe67eb8
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程RejectVpcPeeringConnection式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ReleaseAddress配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ReleaseAddress。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Release an Elastic IP address.
    /// </summary>
    /// <param name="allocationId">The allocation Id of the Elastic IP address.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> ReleaseAddress(string allocationId)
```

```
    {
        var request = new ReleaseAddressRequest
        {
            AllocationId = allocationId
        };

        var response = await _amazonEC2.ReleaseAddressAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考ReleaseAddress中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
###############################################################################
# function ec2_release_address
#
# This function releases an Elastic IP address from an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to
 release.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_release_address() {
  local allocation_id response

  # Function to display usage information
```

```
  function usage() {
    echo "function ec2_release_address"
    echo "Releases an Elastic IP address from an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 release."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
  fi

  response=$(aws ec2 release-address \
    --allocation-id "$allocation_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports release-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}
```

此範例中使用的公用程式函數。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ReleaseAddress中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
Aws::EC2::EC2Client ec2(clientConfiguration);

Aws::EC2::Model::ReleaseAddressRequest request;
request.SetAllocationId(allocationID);

auto outcome = ec2.ReleaseAddress(request);
if (!outcome.IsSuccess()) {
    std::cerr << "Failed to release Elastic IP address " <<
              allocationID << ":" << outcome.GetError().GetMessage() <<
              std::endl;
}
else {
    std::cout << "Successfully released Elastic IP address " <<
              allocationID << std::endl;
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考ReleaseAddress中的。

CLI

AWS CLI

釋出適用於 EC2-Classic 的彈性 IP 地址

此範例會釋出彈性 IP 地址，以便與 EC2-Classic 中的執行個體搭配使用。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 release-address --public-ip 198.51.100.0
```

釋出適用於 EC2-VPC 的彈性 IP 地址

此範例會釋出彈性 IP 地址，以便與 VPC 中的執行個體搭配使用。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 release-address --allocation-id eipalloc-64d5890a
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ReleaseAddress中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    public static void releaseEC2Address(Ec2Client ec2, String allocId) {
        try {
            ReleaseAddressRequest request = ReleaseAddressRequest.builder()
                    .allocationId(allocId)
                    .build();

            ec2.releaseAddress(request);
            System.out.println("Successfully released Elastic IP address " +
allocId);
        } catch (Ec2Exception 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ReleaseAddress中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { ReleaseAddressCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new ReleaseAddressCommand({
    // You can also use PublicIp, but that is for EC2 classic which is being
 retired.
    AllocationId: "ALLOCATION_ID",
  });

  try {
    await client.send(command);
    console.log("Successfully released address.");
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考ReleaseAddress中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun releaseEC2AddressSc(allocId: String?) {
    val request =
        ReleaseAddressRequest {
            allocationId = allocId
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.releaseAddress(request)
        println("Successfully released Elastic IP address $allocId")
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK ReleaseAddress中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會針對 VPC 中的執行個體釋放指定的彈性 IP 位址。

```
Remove-EC2Address -AllocationId eipalloc-12345678 -Force
```

範例 2：此範例會針對 EC2-Classic 中的執行個體發行指定的彈性 IP 位址。

```
Remove-EC2Address -PublicIp 198.51.100.2 -Force
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ReleaseAddress式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
 actions."""

    def __init__(self, ec2_resource, elastic_ip=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param elastic_ip: A Boto3 VpcAddress object. This is a high-level object
 that
                           wraps Elastic IP actions.
        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def release(self):
        """
        Releases an Elastic IP address. After the Elastic IP address is released,
        it can no longer be used.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to release.")
            return
```

```
        try:
            self.elastic_ip.release()
        except ClientError as err:
            logger.error(
                "Couldn't release Elastic IP address %s. Here's why: %s: %s",
                self.elastic_ip.allocation_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件ReleaseAddress中的 Python (博托 3) API 參考。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
# Releases an Elastic IP address from an
# Amazon Elastic Compute Cloud (Amazon EC2) instance.
#
# Prerequisites:
#
# - An Amazon EC2 instance with an associated Elastic IP address.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param allocation_id [String] The ID of the allocation corresponding to
#   the Elastic IP address.
# @return [Boolean] true if the Elastic IP address was released;
#   otherwise, false.
# @example
#   exit 1 unless elastic_ip_address_released?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
```

```
#     'eipalloc-04452e528a66279EX'
#   )
def elastic_ip_address_released?(ec2_client, allocation_id)
  ec2_client.release_address(allocation_id: allocation_id)
  return true
rescue StandardError => e
  puts("Error releasing Elastic IP address: #{e.message}")
  return false
end
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考ReleaseAddress中的。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    TRY.
        lo_ec2->releaseaddress( iv_allocationid = iv_allocation_id ).
        MESSAGE 'Elastic IP address released.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK ReleaseAddress中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ReleaseHosts配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ReleaseHosts。

CLI

AWS CLI

從您的帳戶釋放專用主機

從您的帳戶釋放專用主機。主機上的執行個體必須先停止或終止，才能釋放主機。

命令：

```
aws ec2 release-hosts --host-id=h-0029d6e3cacf1b3da
```

輸出：

```
{
    "Successful":  [
        "h-0029d6e3cacf1b3da"
         ],
  "Unsuccessful": []

 }
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ReleaseHosts中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會釋出指定的主機識別碼 h-0

```
Remove-EC2Host -HostId h-0badafd1dcb2f3456
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Host (ReleaseHosts)" on target
 "h-0badafd1dcb2f3456".
```

```
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y

Successful            Unsuccessful
----------            ------------
{h-0badafd1dcb2f3456} {}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ReleaseHosts式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ReplaceIamInstanceProfileAssociation配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ReplaceIamInstanceProfileAssociation。

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：

- 建置及管理彈性服務

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   /// <summary>
   /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
   /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
   /// used to restart the Python web server.
   /// </summary>
   /// <param name="instanceId">The Id of the instance to update.</param>
```

```
   /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
   /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
   /// <returns>Async task.</returns>
   public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
   {
       await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
               new ReplaceIamInstanceProfileAssociationRequest()
               {
                   AssociationId = associationId,
                   IamInstanceProfile = new IamInstanceProfileSpecification()
                   {
                       Name = credsProfileName
                   }
               });
       // Allow time before resetting.
       Thread.Sleep(25000);
       var instanceReady = false;
       var retries = 5;
       while (retries-- > 0 && !instanceReady)
       {
           await _amazonEc2.RebootInstancesAsync(
               new RebootInstancesRequest(new List<string>() { instanceId }));
           Thread.Sleep(10000);

           var instancesPaginator =
_amazonSsm.Paginators.DescribeInstanceInformation(
               new DescribeInstanceInformationRequest());
           // Get the entire list using the paginator.
           await foreach (var instance in
instancesPaginator.InstanceInformationList)
           {
               instanceReady = instance.InstanceId == instanceId;
               if (instanceReady)
               {
                   break;
               }
           }
       }
       Console.WriteLine($"Sending restart command to instance {instanceId}");
       await _amazonSsm.SendCommandAsync(
           new SendCommandRequest()
```

```
            {
                InstanceIds = new List<string>() { instanceId },
                DocumentName = "AWS-RunShellScript",
                Parameters = new Dictionary<string, List<string>>()
                {
                        {"commands", new List<string>() { "cd / && sudo python3
 server.py 80" }}
                }
            });
        Console.WriteLine($"Restarted the web server on instance {instanceId}");
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考[ReplaceIamInstanceProfileAssociation](#)中的。

CLI

AWS CLI

取代執行個體的 IAM 執行個體設定檔

此範例會取代以與名為 `AdminRole` 的 IAM 執行個體設定檔之關聯 `iip-assoc-060bae234aac2e7fa` 所表示的 IAM 執行個體設定檔。

```
aws ec2 replace-iam-instance-profile-association \
    --iam-instance-profile Name=AdminRole \
    --association-id iip-assoc-060bae234aac2e7fa
```

輸出：

```
{
    "IamInstanceProfileAssociation": {
        "InstanceId": "i-087711ddaf98f9489",
        "State": "associating",
        "AssociationId": "iip-assoc-0b215292fab192820",
        "IamInstanceProfile": {
            "Id": "AIPAJLNLDX3AMYZNWYYAY",
            "Arn": "arn:aws:iam::123456789012:instance-profile/AdminRole"
        }
    }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ReplaceIamInstanceProfileAssociation中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      ec2Client.send(
        new ReplaceIamInstanceProfileAssociationCommand({
          AssociationId: state.instanceProfileAssociationId,
          IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
        }),
      ),
    );
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考ReplaceIamInstanceProfileAssociation中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

此範例會取代執行中執行個體的執行個體設定檔、重新啟動執行個體，並在執行個體啟動後傳送命令至執行個體。

```
class AutoScaler:
```

```
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def replace_instance_profile(
```

```
        self, instance_id, new_instance_profile_name, profile_association_id
    ):
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to update.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)
            inst_ready = False
            tries = 0
            while not inst_ready:
                if tries % 6 == 0:
                    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                    log.info(
                        "Rebooting instance %s and waiting for it to to be
ready.",
                        instance_id,
                    )
                tries += 1
                time.sleep(10)
                response = self.ssm_client.describe_instance_information()
                for info in response["InstanceInformationList"]:
                    if info["InstanceId"] == instance_id:
```

```
                        inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件ReplaceIamInstanceProfileAssociation中的 Python (博托 3) API 參考。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ReplaceNetworkAclAssociation配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ReplaceNetworkAclAssociation。

CLI

AWS CLI

取代與子網路相關聯的網路 ACL

此範例會將指定的網路 ACL 與指定網路 ACL 關聯的子網路產生關聯。

命令：

```
aws ec2 replace-network-acl-association --association-id aclassoc-e5b95c8c --
network-acl-id acl-5fb85d36
```

輸出：

```
{
```

```
    "NewAssociationId": "aclassoc-3999875b"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ReplaceNetworkAclAssociation中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的網路 ACL 與指定網路 ACL 關聯的子網路產生關聯。

```
Set-EC2NetworkAclAssociation -NetworkAclId acl-12345678 -AssociationId
 aclassoc-1a2b3c4d
```

輸出：

```
aclassoc-87654321
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ReplaceNetworkAclAssociation式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**ReplaceNetworkAclEntry**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ReplaceNetworkAclEntry。

CLI

AWS CLI

取代網路 ACL 項目

此範例會取代指定網路 ACL 的項目。新的規則 100 允許將 UDP 連接埠 53 (DNS) 上 203.0.113.12/24 的流量輸入至任何相關聯的子網路。

命令：

```
aws ec2 replace-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-
number 100 --protocol udp --port-range From=53,To=53 --cidr-block 203.0.113.12/24
 --rule-action allow
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ReplaceNetworkAclEntry中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會取代指定網路 ACL 的指定項目。新規則允許從指定位址到任何關聯子網路的輸入流量。

```
Set-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber 100
 -Protocol 17 -PortRange_From 53 -PortRange_To 53 -CidrBlock 203.0.113.12/24 -
RuleAction allow
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ReplaceNetworkAclEntry式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ReplaceRoute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ReplaceRoute。

CLI

AWS CLI

取代路線

此範例會取代指定路由表格中的指定路由。新路由符合指定的 CIDR，並將流量傳送到指定的虛擬私有閘道。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 replace-route --route-table-id rtb-22574640 --destination-cidr-block
 10.0.0.0/16 --gateway-id vgw-9a4cacf3
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ReplaceRoute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會取代指定路由表的指定路由。新路由會將指定的流量傳送到指定的虛擬私有閘道。

```
Set-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 10.0.0.0/24 -
GatewayId vgw-1a2b3c4d
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ReplaceRoute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ReplaceRouteTableAssociation配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ReplaceRouteTableAssociation。

CLI

AWS CLI

取代與子網路相關聯的路由表

此範例會將指定的路由表與指定路由表關聯的子網路產生關聯。

命令：

```
aws ec2 replace-route-table-association --association-id rtbassoc-781d0d1a --
route-table-id rtb-22574640
```

輸出：

```
{
    "NewAssociationId": "rtbassoc-3a1f0f58"
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ReplaceRouteTableAssociation中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將指定的路由表與指定路由表關聯的子網路產生關聯。

```
Set-EC2RouteTableAssociation -RouteTableId rtb-1a2b3c4d -AssociationId
 rtbassoc-12345678
```

輸出：

```
rtbassoc-87654321
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ReplaceRouteTableAssociation式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ReportInstanceStatus配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ReportInstanceStatus。

CLI

AWS CLI

回報執行個體的狀態回饋

此範例命令會報告指定執行個體的狀態回饋。

命令：

```
aws ec2 report-instance-status --instances i-1234567890abcdef0 --status impaired
 --reason-codes unresponsive
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ReportInstanceStatus中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會報告指定執行個體的狀態回饋。

```
Send-EC2InstanceStatus -Instance i-12345678 -Status impaired -ReasonCode
 unresponsive
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ReportInstanceStatus式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭RequestSpotFleet配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用RequestSpotFleet。

CLI

AWS CLI

以最低價格在子網路中要求競價型叢集

此範例命令會建立一個 Spot 叢集請求，其中兩個啟動規格僅因子網路而異。Spot 叢集以最低的價格啟動指定子網路中的執行個體。如果執行個體是在預設 VPC 中啟動，則依預設會收到公用 IP 位址。如果執行個體是在非預設的 VPC 中啟動，則其預設不會接收公有 IP 地址。

請注意，您無法在 Spot 叢集請求中從相同可用區域指定不同的子網路。

命令：

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
```

```
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "SubnetId": "subnet-1a2b3c4d, subnet-3c4d5e6f",
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
      }
  ]
}
```

輸出：

```
{
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

以最低的價格在可用區域中要求競價型叢集

此範例命令會建立具有兩個啟動規格的 Spot 叢集請求，這些規格僅因可用區域而異。Spot 叢集以最低的價格啟動指定可用區域中的執行個體。如果您的帳戶僅支援 EC2-VPC，Amazon EC2 會在可用區域的預設子網路中啟動競價型執行個體。如果您的帳戶支援 EC2-Classic，Amazon EC2 會在可用區域的 EC2-Classic 中啟動執行個體。

命令：

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
```

```
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "Placement": {
              "AvailabilityZone": "us-west-2a, us-west-2b"
          },
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
      }
  ]
}
```

在子網路中啟動 Spot 執行個體並為其指派公有 IP 位址

此範例命令會將公用位址指派給在非預設 VPC 中啟動的執行個體。請注意，當您指定網路介面時，必須使用網路介面加入子網路識別碼和安全群組識別碼。

命令：

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "InstanceType": "m3.medium",
          "NetworkInterfaces": [
```

```
            {
                "DeviceIndex": 0,
                "SubnetId": "subnet-1a2b3c4d",
                "Groups": [ "sg-1a2b3c4d" ],
                "AssociatePublicIpAddress": true
            }
        ],
        "IamInstanceProfile": {
            "Arn": "arn:aws:iam::880185128111:instance-profile/my-iam-role"
        }
    }
  ]
}
```

使用多元化配置策略請求 Spot 車隊

此範例命令會建立 Spot 叢集請求，使用多元化配置策略啟動 30 個執行個體。啟動規格會因執行個體類型而異。Spot 叢集會在啟動規格中分配執行個體，以便每種類型有 10 個執行個體。

命令：

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 30,
  "AllocationStrategy": "diversified",
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c4.2xlarge",
          "SubnetId": "subnet-1a2b3c4d"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "m3.2xlarge",
          "SubnetId": "subnet-1a2b3c4d"
      },
      {
          "ImageId": "ami-1a2b3c4d",
```

```
            "InstanceType": "r3.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        }
    ]
}
```

如需詳細資訊，請參閱 Amazon 彈性運算雲端使用者指南中的競價型叢集請求。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考RequestSpotFleet中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會以指定執行個體類型的最低價格在可用區域中建立 Spot 叢集請求。如果您的帳戶僅支援 EC2-VPC，Spot 叢集會在價格最低的可用區域中啟動具有預設子網路的執行個體。如果您的帳戶支援 EC2-Classic，Spot 叢集會在價格最低的可用區域中啟動 EC2-Classic 中的執行個體。請注意，您支付的價格不會超過請求的指定現貨價格。

```
$sg = New-Object Amazon.EC2.Model.GroupIdentifier
$sg.GroupId = "sg-12345678"
$lc = New-Object Amazon.EC2.Model.SpotFleetLaunchSpecification
$lc.ImageId = "ami-12345678"
$lc.InstanceType = "m3.medium"
$lc.SecurityGroups.Add($sg)
Request-EC2SpotFleet -SpotFleetRequestConfig_SpotPrice 0.04 `
-SpotFleetRequestConfig_TargetCapacity 2 `
-SpotFleetRequestConfig_IamFleetRole arn:aws:iam::123456789012:role/my-spot-
fleet-role `
-SpotFleetRequestConfig_LaunchSpecification $lc
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程RequestSpotFleet式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭RequestSpotInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用RequestSpotInstances。

CLI

AWS CLI

要求競價型執行個體

此範例命令會針對指定可用區域中的五個執行個體建立一次性 Spot 執行個體請求。如果您的帳戶僅支援 EC2-VPC，Amazon EC2 會在指定可用區域的預設子網路中啟動執行個體。如果您的帳戶支援 EC2-Classic，Amazon EC2 會在指定可用區域的 EC2-Classic 中啟動執行個體。

命令：

```
aws ec2 request-spot-instances --spot-price "0.03" --instance-count 5 --type
 "one-time" --launch-specification file://specification.json
```

規格:

```
{
  "ImageId": "ami-1a2b3c4d",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": [ "sg-1a2b3c4d" ],
  "InstanceType": "m3.medium",
  "Placement": {
    "AvailabilityZone": "us-west-2a"
  },
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

輸出：

```
{
  "SpotInstanceRequests": [
      {
          "Status": {
              "UpdateTime": "2014-03-25T20:54:21.000Z",
              "Code": "pending-evaluation",
              "Message": "Your Spot request has been submitted for review, and is
 pending evaluation."
          },
          "ProductDescription": "Linux/UNIX",
```

```
            "SpotInstanceRequestId": "sir-df6f405d",
            "State": "open",
            "LaunchSpecification": {
                "Placement": {
                    "AvailabilityZone": "us-west-2a"
                },
                "ImageId": "ami-1a2b3c4d",
                "KeyName": "my-key-pair",
                "SecurityGroups": [
                    {
                        "GroupName": "my-security-group",
                        "GroupId": "sg-1a2b3c4d"
                    }
                ],
                "Monitoring": {
                    "Enabled": false
                },
                "IamInstanceProfile": {
                    "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
                },
                "InstanceType": "m3.medium"
            },
            "Type": "one-time",
            "CreateTime": "2014-03-25T20:54:20.000Z",
            "SpotPrice": "0.050000"
        },
        ...
    ]
}
```

此範例命令會針對指定子網路中的五個執行個體建立一次性 Spot 執行個體請求。Amazon EC2 會在指定子網中啟動執行個體。如果 VPC 是非預設 VPC，則依預設，執行個體不會收到公用 IP 位址。

命令：

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 5 --type
 "one-time" --launch-specification file://specification.json
```

規格:

```
{
```

```
  "ImageId": "ami-1a2b3c4d",
  "SecurityGroupIds": [ "sg-1a2b3c4d" ],
  "InstanceType": "m3.medium",
  "SubnetId": "subnet-1a2b3c4d",
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

輸出：

```
{
  "SpotInstanceRequests": [
      {
          "Status": {
             "UpdateTime": "2014-03-25T22:21:58.000Z",
             "Code": "pending-evaluation",
             "Message": "Your Spot request has been submitted for review, and is
 pending evaluation."
          },
          "ProductDescription": "Linux/UNIX",
          "SpotInstanceRequestId": "sir-df6f405d",
          "State": "open",
          "LaunchSpecification": {
             "Placement": {
                 "AvailabilityZone": "us-west-2a"
             }
             "ImageId": "ami-1a2b3c4d"
             "SecurityGroups": [
                 {
                     "GroupName": "my-security-group",
                     "GroupID": "sg-1a2b3c4d"
                 }
             ]
             "SubnetId": "subnet-1a2b3c4d",
             "Monitoring": {
                 "Enabled": false
             },
             "IamInstanceProfile": {
                 "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
             },
             "InstanceType": "m3.medium",
         },
```

```
            "Type": "one-time",
            "CreateTime": "2014-03-25T22:21:58.000Z",
            "SpotPrice": "0.050000"
        },
        ...
    ]
}
```

此範例會將公用 IP 位址指派給您在非預設 VPC 中啟動的 Spot 執行個體。請注意，當您指定網路介面時，必須使用網路介面加入子網路識別碼和安全群組識別碼。

命令：

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 1 --type
 "one-time" --launch-specification file://specification.json
```

規格:

```
{
  "ImageId": "ami-1a2b3c4d",
  "KeyName": "my-key-pair",
  "InstanceType": "m3.medium",
  "NetworkInterfaces": [
    {
      "DeviceIndex": 0,
      "SubnetId": "subnet-1a2b3c4d",
      "Groups": [ "sg-1a2b3c4d" ],
      "AssociatePublicIpAddress": true
    }
  ],
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考RequestSpotInstances中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例要求指定子網路中的一次性 Spot 執行個體。請注意，必須為包含指定子網路的 VPC 建立安全性群組，並且必須使用網路介面透過 ID 指定。當您指定網路介面時，必須使用網路介面加入子網路 ID。

```
$n = New-Object Amazon.EC2.Model.InstanceNetworkInterfaceSpecification
$n.DeviceIndex = 0
$n.SubnetId = "subnet-12345678"
$n.Groups.Add("sg-12345678")
Request-EC2SpotInstance -InstanceCount 1 -SpotPrice 0.050 -Type one-time `
-IamInstanceProfile_Arn arn:aws:iam::123456789012:instance-profile/my-iam-role `
-LaunchSpecification_ImageId ami-12345678 `
-LaunchSpecification_InstanceType m3.medium `
-LaunchSpecification_NetworkInterface $n
```

輸出：

```
ActualBlockHourlyPrice   :
AvailabilityZoneGroup    :
BlockDurationMinutes     : 0
CreateTime               : 12/26/2015 7:44:10 AM
Fault                    :
InstanceId               :
LaunchedAvailabilityZone :
LaunchGroup              :
LaunchSpecification      : Amazon.EC2.Model.LaunchSpecification
ProductDescription       : Linux/UNIX
SpotInstanceRequestId    : sir-12345678
SpotPrice                : 0.050000
State                    : open
Status                   : Amazon.EC2.Model.SpotInstanceStatus
Tags                     : {}
Type                     : one-time
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程RequestSpotInstances式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配 AWS 開發套件或 CLI 使用 ResetImageAttribute

下列程式碼範例會示範如何使用 ResetImageAttribute。

CLI

AWS CLI

若要重設啟動權限屬性

此範例會將指定 AMI 的 launchPermission 屬性重設為預設值。依預設，AMI 是私人的。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 reset-image-attribute --image-id ami-5731123e --attribute
 launchPermission
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ResetImageAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會將「啟動權限」屬性重設為預設值。依預設，AMI 是私人的。

```
Reset-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ResetImageAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配 AWS 開發套件或 CLI 使用 ResetInstanceAttribute

下列程式碼範例會示範如何使用 ResetInstanceAttribute。

CLI

AWS CLI

重設 sourceDestCheck 屬性的步驟

此範例會重設指定執行個體的sourceDestCheck屬性。執行個體必須位於 VPC 中。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 sourceDestCheck
```

若要重設核心屬性

此範例會重設指定執行個體的kernel屬性。執行個體必須處於 stopped 狀態。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 kernel
```

若要重設磁碟屬性

此範例會重設指定執行個體的ramdisk屬性。執行個體必須處於 stopped 狀態。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 ramdisk
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ResetInstanceAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會重設指定執行個體的 sriovNetSupport " 屬性。

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

範例 2：此範例會重設指定執行個體的「最佳化」屬性。

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

範例 3：此範例會重設指定執行個體的 sourceDestCheck " 屬性。

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sourceDestCheck
```

範例 4：此範例會重設指定執行個體的 disableApiTermination " 屬性。

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute
 disableApiTermination
```

範例 5：此範例會重設指定執instanceInitiatedShutdown行個體的「行為」屬性。

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute
 instanceInitiatedShutdownBehavior
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ResetInstanceAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**ResetNetworkInterfaceAttribute**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用`ResetNetworkInterfaceAttribute`。

CLI

AWS CLI

重設網路介面屬性

下列`reset-network-interface-attribute`範例會將來源/目的地檢查屬性的值重設為。`true`

```
aws ec2 reset-network-interface-attribute \
```

```
    --network-interface-id eni-686ea200 \
    --source-dest-check
```

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ResetNetworkInterfaceAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會重設指定網路介面的來源/目標檢查。

```
Reset-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -
SourceDestCheck
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ResetNetworkInterfaceAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ResetSnapshotAttribute配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ResetSnapshotAttribute。

CLI

AWS CLI

重設快照屬性

此範例會重設快照的建立磁碟區權限snap-1234567890abcdef0。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 reset-snapshot-attribute --snapshot-id snap-1234567890abcdef0 --attribute
 createVolumePermission
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ResetSnapshotAttribute中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會重設指定快照的指定屬性。

```
Reset-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
 CreateVolumePermission
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程ResetSnapshotAttribute式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭RevokeSecurityGroupEgress配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用RevokeSecurityGroupEgress。

CLI

AWS CLI

範例 1：移除允許輸出流量到特定位址範圍的規則

下列範revoke-security-group-egress例命令會移除授與 TCP 連接埠 80 上指定位址範圍存取權的規則。

```
aws ec2 revoke-security-group-egress \
    --group-id sg-026c12253ce15eff7 \
    --ip-permissions
 [{IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges=[{CidrIp=10.0.0.0/16}]
```

此命令不會產生輸出。

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的安全群組。

範例 2：移除允許輸出流量至特定安全性群組的規則

下列revoke-security-group-egress範例命令會移除授與 TCP 連接埠 80 上指定安全性群組存取權的規則。

```
aws ec2 revoke-security-group-egress \
    --group-id sg-026c12253ce15eff7 \
    --ip-permissions '[{"IpProtocol": "tcp", "FromPort": 443, "ToPort":
 443,"UserIdGroupPairs": [{"GroupId": "sg-06df23a01ff2df86d"}]}]'
```

此命令不會產生輸出。

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的安全群組。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考RevokeSecurityGroupEgress中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會針對 EC2-VPC 移除指定安全性群組的規則。這會撤銷 TCP 連接埠 80 上指定 IP 位址範圍的存取權。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
$ip = @{ IpProtocol="tcp"; FromPort="80"; ToPort="80";
 IpRanges="203.0.113.0/24" }
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

範例 2：在 PowerShell 版本 2 中，您必須使用新物件來建立 IpPermission 物件。

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 80
$ip.ToPort = 80
$ip.IpRanges.Add("203.0.113.0/24")
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

範例 3：此範例會撤銷 TCP 連接埠 80 上指定來源安全性群組的存取權。

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission
 @( @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; UserIdGroupPairs=$ug } )
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程RevokeSecurityGroupEgress式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭RevokeSecurityGroupIngress配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用RevokeSecurityGroupIngress。

CLI

AWS CLI

範例 1：若要從安全性群組移除規則

下列範revoke-security-group-ingress例會從預設 VPC 的指定安全性群組中移除203.0.113.0/24位址範圍的 TCP 連接埠 22 存取權。

```
aws ec2 revoke-security-group-ingress \
    --group-name mySecurityGroup
    --protocol tcp \
    --port 22 \
    --cidr 203.0.113.0/24
```

如果此命令成功，則不會產生輸出。

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的安全群組。

範例 2：使用 IP 權限集移除規則

下列revoke-security-group-ingress範例會使用ip-permissions參數移除允許 ICMP 訊息的輸入規則 Destination Unreachable: Fragmentation Needed and Don't Fragment was Set (類型 3，程式碼 4)。

```
aws ec2 revoke-security-group-ingress \
    --group-id sg-026c12253ce15eff7 \
    --ip-permissions
 IpProtocol=icmp,FromPort=3,ToPort=4,IpRanges=[{CidrIp=0.0.0.0/0}]
```

如果此命令成功，則不會產生輸出。

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的安全群組。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考[RevokeSecurityGroupIngress](#)中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會針對 EC2-VPC 的指定安全性群組，從指定的位址範圍撤銷 TCP 連接埠 22 的存取權。請注意，您必須使用安全性群組識別碼而非安全群組名稱來識別 EC2-VPC 的安全性群組。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";
 IpRanges="203.0.113.0/24" }
Revoke-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission $ip
```

範例 2：在 PowerShell 版本 2 中，您必須使用新物件來建立 IpPermission 物件。

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 22
$ip.ToPort = 22
$ip.IpRanges.Add("203.0.113.0/24")

Revoke-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission $ip
```

範例 3：此範例會從指定的 EC2-Classic 安全性群組的指定位址範圍撤銷 TCP 連接埠 22 的存取權。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";
 IpRanges="203.0.113.0/24" }

Revoke-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission $ip
```

實施例 4：在 PowerShell 版本 2 中，您必須使用新對象創建對 IpPermission 象。

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 22
$ip.ToPort = 22
$ip.IpRanges.Add("203.0.113.0/24")
```

```
Revoke-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission $ip
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程RevokeSecurityGroupIngress式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭RunInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用RunInstances。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Create and run an EC2 instance.
    /// </summary>
    /// <param name="ImageId">The image Id of the image used as a basis for the
    /// EC2 instance.</param>
    /// <param name="instanceType">The instance type of the EC2 instance to
create.</param>
    /// <param name="keyName">The name of the key pair to associate with the
    /// instance.</param>
    /// <param name="groupId">The Id of the Amazon EC2 security group that will
be
    /// allowed to interact with the new EC2 instance.</param>
    /// <returns>The instance Id of the new EC2 instance.</returns>
```

```
    public async Task<string> RunInstances(string imageId, string instanceType,
 string keyName, string groupId)
    {
        var request = new RunInstancesRequest
        {
            ImageId = imageId,
            InstanceType = instanceType,
            KeyName = keyName,
            MinCount = 1,
            MaxCount = 1,
            SecurityGroupIds = new List<string> { groupId }
        };
        var response = await _amazonEC2.RunInstancesAsync(request);
        return response.Reservation.Instances[0].InstanceId;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考RunInstances中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
###############################################################################
# function ec2_run_instances
#
# This function launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i image_id - The ID of the Amazon Machine Image (AMI) to use.
#       -t instance_type - The instance type to use (e.g., t2.micro).
#       -k key_pair_name - The name of the key pair to use.
#       -s security_group_id - The ID of the security group to use.
#       -c count - The number of instances to launch (default: 1).
```

```
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_run_instances() {
  local image_id instance_type key_pair_name security_group_id count response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_run_instances"
    echo "Launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i image_id - The ID of the Amazon Machine Image (AMI) to use."
    echo "  -t instance_type - The instance type to use (e.g., t2.micro)."
    echo "  -k key_pair_name - The name of the key pair to use."
    echo "  -s security_group_id - The ID of the security group to use."
    echo "  -c count - The number of instances to launch (default: 1)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:t:k:s:c:h" option; do
    case "${option}" in
      i) image_id="${OPTARG}" ;;
      t) instance_type="${OPTARG}" ;;
      k) key_pair_name="${OPTARG}" ;;
      s) security_group_id="${OPTARG}" ;;
      c) count="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1
```

```
 if [[ -z "$image_id" ]]; then
   errecho "ERROR: You must provide an Amazon Machine Image (AMI) ID with the -i
parameter."
   usage
   return 1
 fi

 if [[ -z "$instance_type" ]]; then
   errecho "ERROR: You must provide an instance type with the -t parameter."
   usage
   return 1
 fi

 if [[ -z "$key_pair_name" ]]; then
   errecho "ERROR: You must provide a key pair name with the -k parameter."
   usage
   return 1
 fi

 if [[ -z "$security_group_id" ]]; then
   errecho "ERROR: You must provide a security group ID with the -s parameter."
   usage
   return 1
 fi

 if [[ -z "$count" ]]; then
   count=1
 fi

 response=$(aws ec2 run-instances \
   --image-id "$image_id" \
   --instance-type "$instance_type" \
   --key-name "$key_pair_name" \
   --security-group-ids "$security_group_id" \
   --count "$count" \
   --query 'Instances[*].[InstanceId]' \
   --output text) || {
   aws_cli_error_log ${?}
   errecho "ERROR: AWS reports run-instances operation failed.$response"
   return 1
 }

 echo "$response"
```

```
  return 0
}
```

此範例中使用的公用程式函數。

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
########################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
```

```
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考RunInstances中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::RunInstancesRequest runRequest;
    runRequest.SetImageId(amiId);
    runRequest.SetInstanceType(Aws::EC2::Model::InstanceType::t1_micro);
    runRequest.SetMinCount(1);
    runRequest.SetMaxCount(1);

    Aws::EC2::Model::RunInstancesOutcome runOutcome = ec2Client.RunInstances(
            runRequest);
    if (!runOutcome.IsSuccess()) {
        std::cerr << "Failed to launch EC2 instance " << instanceName <<
                  " based on ami " << amiId << ":" <<
                  runOutcome.GetError().GetMessage() << std::endl;
        return false;
    }

    const Aws::Vector<Aws::EC2::Model::Instance> &instances =
runOutcome.GetResult().GetInstances();
    if (instances.empty()) {
        std::cerr << "Failed to launch EC2 instance " << instanceName <<
                  " based on ami " << amiId << ":" <<
```

```
                  runOutcome.GetError().GetMessage() << std::endl;
        return false;
    }

    instanceID = instances[0].GetInstanceId();
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考RunInstances中的。

CLI

AWS CLI

範例 1：在預設子網路中啟動執行個體

下列 `run-instances` 範例會在目前區域的預設子網路中啟動 `t2.micro` 類型的單一執行個體，並將其與該區域中預設 VPC 的預設子網路建立關聯。如果您不打算使用 SSH (Linux) 或 RDP (Windows) 連接至執行個體，則金鑰對為選用項目。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --key-name MyKeyPair
```

輸出：

```
{
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-0abcdef1234567890",
            "InstanceId": "i-1231231230abcdef0",
            "InstanceType": "t2.micro",
            "KeyName": "MyKeyPair",
            "LaunchTime": "2018-05-10T08:05:20.000Z",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-east-2a",
                "GroupName": "",
                "Tenancy": "default"
```

```
            },
            "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
            "PrivateIpAddress": "10.0.0.157",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "StateTransitionReason": "",
            "SubnetId": "subnet-04a636d18e83cfacb",
            "VpcId": "vpc-1234567890abcdef0",
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "",
            "EbsOptimized": false,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
                        "AttachTime": "2018-05-10T08:05:20.000Z",
                        "AttachmentId": "eni-attach-0e325c07e928a0405",
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "Status": "attaching"
                    },
                    "Description": "",
                    "Groups": [
                        {
                            "GroupName": "MySecurityGroup",
                            "GroupId": "sg-0598c7d356eba48d7"
                        }
                    ],
                    "Ipv6Addresses": [],
                    "MacAddress": "0a:ab:58:e0:67:e2",
                    "NetworkInterfaceId": "eni-0c0a29997760baee7",
                    "OwnerId": "123456789012",
                    "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
                    "PrivateIpAddress": "10.0.0.157",
                    "PrivateIpAddresses": [
                        {
                            "Primary": true,
                            "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
```

```
                            "PrivateIpAddress": "10.0.0.157"
                        }
                    ],
                    "SourceDestCheck": true,
                    "Status": "in-use",
                    "SubnetId": "subnet-04a636d18e83cfacb",
                    "VpcId": "vpc-1234567890abcdef0",
                    "InterfaceType": "interface"
                }
            ],
            "RootDeviceName": "/dev/xvda",
            "RootDeviceType": "ebs",
            "SecurityGroups": [
                {
                    "GroupName": "MySecurityGroup",
                    "GroupId": "sg-0598c7d356eba48d7"
                }
            ],
            "SourceDestCheck": true,
            "StateReason": {
                "Code": "pending",
                "Message": "pending"
            },
            "Tags": [],
            "VirtualizationType": "hvm",
            "CpuOptions": {
                "CoreCount": 1,
                "ThreadsPerCore": 1
            },
            "CapacityReservationSpecification": {
                "CapacityReservationPreference": "open"
            },
            "MetadataOptions": {
                "State": "pending",
                "HttpTokens": "optional",
                "HttpPutResponseHopLimit": 1,
                "HttpEndpoint": "enabled"
            }
        }
    ],
    "OwnerId": "123456789012",
    "ReservationId": "r-02a3f596d91211712"
}
```

範例 2：在非預設的子網路中啟動執行個體，並新增公有 IP 地址

下列 run-instances 範例會針對您要在非預設的子網路中啟動的執行個體請求公有 IP 地址。執行個體已與指定的安全群組建立關聯。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --subnet-id subnet-08fc749671b2d077c \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --associate-public-ip-address \
    --key-name MyKeyPair
```

如需 run-instances 的輸出範例，請參閱範例 1。

範例 3：啟動具有額外磁碟區的執行個體

下列 run-instances 範例會使用 mapping.json 中指定的區塊型裝置映射，在啟動時連接額外的磁碟區。區塊型裝置映射可以指定 EBS 磁碟區、執行個體儲存體磁碟區，或同時指定 EBS 磁碟區和執行個體儲存體磁碟區。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --subnet-id subnet-08fc749671b2d077c \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --key-name MyKeyPair \
    --block-device-mappings file://mapping.json
```

mapping.json 的內容。此範例會為 /dev/sdh 新增空的 EBS 磁碟區 (大小為 100 GiB)。

```
[
    {
        "DeviceName": "/dev/sdh",
        "Ebs": {
            "VolumeSize": 100
        }
    }
]
```

mapping.json 的內容。此範例會將 ephemeral1 新增為執行個體儲存體磁碟區。

```
[
    {
        "DeviceName": "/dev/sdc",
        "VirtualName": "ephemeral1"
    }
]
```

如需 run-instances 的輸出範例，請參閱範例 1。

如需區塊型裝置映射的詳細資訊，請參閱中的《Amazon EC2 使用者指南》的區塊型裝置映射。

範例 4：啟動執行個體並在建立時新增標籤

下列 run-instances 範例會將索引鍵為 webserver、值為 production 的標籤新增至執行個體。命令也會為任何建立的 EBS 磁碟區 (此範例中為根磁碟區) 套用鍵為 cost-center，值為 cc123 的標籤。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --count 1 \
    --subnet-id subnet-08fc749671b2d077c \
    --key-name MyKeyPair \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --tag-specifications
 'ResourceType=instance,Tags=[{Key=webserver,Value=production}]'
 'ResourceType=volume,Tags=[{Key=cost-center,Value=cc123}]'
```

如需 run-instances 的輸出範例，請參閱範例 1。

範例 5：使用使用者資料啟動執行個體

下列 run-instances 範例會將使用者資料傳遞至名為 my_script.txt 的檔案中，且該檔案包含您執行個體的組態指令碼。指令碼會在啟動時執行。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --count 1 \
    --subnet-id subnet-08fc749671b2d077c \
    --key-name MyKeyPair \
```

```
    --security-group-ids sg-0b0384b66d7d692f9 \
    --user-data file://my_script.txt
```

如需 run-instances 的輸出範例，請參閱範例 1。

如需執行個體使用者資料的詳細資訊，請參閱《Amazon EC2 使用者指南》中的使用執行個體使用者資料。

範例 6：啟動爆量效能執行個體

下列 run-instances 範例會啟動具有 unlimited 積分選項的 t2.micro 執行個體。啟動 T2 執行個體時，如果您未指定 --credit-specification，預設值為 standard 積分選項。啟動 T3 執行個體時，預設值為 unlimited 積分選項。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --count 1 \
    --subnet-id subnet-08fc749671b2d077c \
    --key-name MyKeyPair \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --credit-specification CpuCredits=unlimited
```

如需 run-instances 的輸出範例，請參閱範例 1。

如需爆量效能執行個體的詳細資訊，請參閱《Amazon EC2 使用者指南》中的爆量效能執行個體。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考RunInstances中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.ec2.Ec2Client;
import software.amazon.awssdk.services.ec2.model.InstanceType;
import software.amazon.awssdk.services.ec2.model.RunInstancesRequest;
import software.amazon.awssdk.services.ec2.model.RunInstancesResponse;
import software.amazon.awssdk.services.ec2.model.Tag;
import software.amazon.awssdk.services.ec2.model.CreateTagsRequest;
import software.amazon.awssdk.services.ec2.model.Ec2Exception;

/**
 * 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 code example requires an AMI value. You can learn more about this value
 * by reading this documentation topic:
 *
 * https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/AMIs.html
 */
public class CreateInstance {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                   <name> <amiId>

                Where:
                   name - An instance name value that you can obtain from the AWS
 Console (for example, ami-xxxxxx5c8b987b1a0).\s
                   amiId - An Amazon Machine Image (AMI) value that you can
 obtain from the AWS Console (for example, i-xxxxxx2734106d0ab).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String name = args[0];
        String amiId = args[1];
        Region region = Region.US_EAST_1;
```

```
        Ec2Client ec2 = Ec2Client.builder()
            .region(region)
            .build();

        String instanceId = createEC2Instance(ec2, name, amiId);
        System.out.println("The Amazon EC2 Instance ID is " + instanceId);
        ec2.close();
    }

    public static String createEC2Instance(Ec2Client ec2, String name, String
amiId) {
        RunInstancesRequest runRequest = RunInstancesRequest.builder()
            .imageId(amiId)
            .instanceType(InstanceType.T1_MICRO)
            .maxCount(1)
            .minCount(1)
            .build();

        // Use a waiter to wait until the instance is running.
        System.out.println("Going to start an EC2 instance using a waiter");
        RunInstancesResponse response = ec2.runInstances(runRequest);
        String instanceIdVal = response.instances().get(0).instanceId();
        ec2.waiter().waitUntilInstanceRunning(r -> r.instanceIds(instanceIdVal));
        Tag tag = Tag.builder()
            .key("Name")
            .value(name)
            .build();

        CreateTagsRequest tagRequest = CreateTagsRequest.builder()
            .resources(instanceIdVal)
            .tags(tag)
            .build();

        try {
            ec2.createTags(tagRequest);
            System.out.printf("Successfully started EC2 Instance %s based on AMI
%s", instanceIdVal, amiId);
            return instanceIdVal;

        } catch (Ec2Exception 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RunInstances中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { RunInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// Create a new EC2 instance.
export const main = async () => {
  const command = new RunInstancesCommand({
    // Your key pair name.
    KeyName: "KEY_PAIR_NAME",
    // Your security group.
    SecurityGroupIds: ["SECURITY_GROUP_ID"],
    // An x86_64 compatible image.
    ImageId: "ami-0001a0d1a04bfcc30",
    // An x86_64 compatible free-tier instance type.
    InstanceType: "t1.micro",
    // Ensure only 1 instance launches.
    MinCount: 1,
    MaxCount: 1,
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
```

```
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考RunInstances中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun createEC2Instance(
    name: String,
    amiId: String,
): String? {
    val request =
        RunInstancesRequest {
            imageId = amiId
            instanceType = InstanceType.T1Micro
            maxCount = 1
            minCount = 1
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.runInstances(request)
        val instanceId = response.instances?.get(0)?.instanceId
        val tag =
            Tag {
                key = "Name"
                value = name
            }

        val requestTags =
            CreateTagsRequest {
                resources = listOf(instanceId.toString())
                tags = listOf(tag)
            }
```

```
        ec2.createTags(requestTags)
        println("Successfully started EC2 Instance $instanceId based on AMI
 $amiId")
        return instanceId
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK RunInstances中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會在 EC2-Classic 或預設 VPC 中啟動指定 AMI 的單一執行個體。

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -InstanceType
 m3.medium -KeyName my-key-pair -SecurityGroup my-security-group
```

範例 2：此範例會在 VPC 中啟動指定 AMI 的單一執行個體。

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -SubnetId
 subnet-12345678 -InstanceType t2.micro -KeyName my-key-pair -SecurityGroupId
 sg-12345678
```

範例 3：若要新增 EBS 磁碟區或執行個體儲存磁碟區，請定義區塊裝置對應並將其新增至指令。此範例會新增執行個體儲存磁碟區。

```
$bdm = New-Object Amazon.EC2.Model.BlockDeviceMapping
$bdm.VirtualName = "ephemeral0"
$bdm.DeviceName = "/dev/sdf"

New-EC2Instance -ImageId ami-12345678 -BlockDeviceMapping $bdm ...
```

範例 4：若要指定其中一個目前的視窗 AMI，請使用Get-EC2ImageByName取得其 AMI 識別碼。此範例會從目前的基礎 AMI 啟動執行個體 (適用於視窗伺服器 2016 年)。

```
$ami = Get-EC2ImageByName WINDOWS_2016_BASE

New-EC2Instance -ImageId $ami.ImageId ...
```

範例 5：將執行個體啟動至指定的專用主機環境。

```
New-EC2Instance -ImageId ami-1a2b3c4d -InstanceType m4.large -KeyName my-key-pair
 -SecurityGroupId sg-1a2b3c4d  -AvailabilityZone us-west-1a -Tenancy host -HostID
 h-1a2b3c4d5e6f1a2b3
```

示例 6：此請求啟動兩個實例，並將帶有 Web 服務器密鑰和生產值的標籤應用於實例。此要求也會將含有成本中心金鑰和值 cc123 的標籤套用至所建立的磁碟區 (在本例中為每個執行個體的根磁碟區)。

```
$tag1 = @{ Key="webserver"; Value="production" }
$tag2 = @{ Key="cost-center"; Value="cc123" }

$tagspec1 = new-object Amazon.EC2.Model.TagSpecification
$tagspec1.ResourceType = "instance"
$tagspec1.Tags.Add($tag1)

$tagspec2 = new-object Amazon.EC2.Model.TagSpecification
$tagspec2.ResourceType = "volume"
$tagspec2.Tags.Add($tag2)

New-EC2Instance -ImageId "ami-1a2b3c4d" -KeyName "my-key-pair" -MaxCount 2 -
InstanceType "t2.large" -SubnetId "subnet-1a2b3c4d" -TagSpecification $tagspec1,
$tagspec2
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程RunInstances式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""
```

```
   def __init__(self, ec2_resource, instance=None):
       """
       :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
resource
                            is used to create additional high-level objects
                            that wrap low-level Amazon EC2 service actions.
       :param instance: A Boto3 Instance object. This is a high-level object
that
                          wraps instance actions.
       """
       self.ec2_resource = ec2_resource
       self.instance = instance

   @classmethod
   def from_resource(cls):
       ec2_resource = boto3.resource("ec2")
       return cls(ec2_resource)


   def create(self, image, instance_type, key_pair, security_groups=None):
       """
       Creates a new EC2 instance. The instance starts immediately after
       it is created.

       The instance is created in the default VPC of the current account.

       :param image: A Boto3 Image object that represents an Amazon Machine
Image (AMI)
                     that defines attributes of the instance that is created.
The AMI
                     defines things like the kind of operating system and the
type of
                     storage used by the instance.
       :param instance_type: The type of instance to create, such as 't2.micro'.
                             The instance type defines things like the number of
CPUs and
                             the amount of memory.
       :param key_pair: A Boto3 KeyPair or KeyPairInfo object that represents
the key
                        pair that is used to secure connections to the instance.
       :param security_groups: A list of Boto3 SecurityGroup objects that
represents the
```

```
                                security groups that are used to grant access to
the
                                instance. When no security groups are specified,
the
                                default security group of the VPC is used.
        :return: A Boto3 Instance object that represents the newly created
instance.
        """
        try:
            instance_params = {
                "ImageId": image.id,
                "InstanceType": instance_type,
                "KeyName": key_pair.name,
            }
            if security_groups is not None:
                instance_params["SecurityGroupIds"] = [sg.id for sg in
security_groups]
            self.instance = self.ec2_resource.create_instances(
                **instance_params, MinCount=1, MaxCount=1
            )[0]
            self.instance.wait_until_running()
        except ClientError as err:
            logging.error(
                "Couldn't create instance with image %s, instance type %s, and
key %s. "
                "Here's why: %s: %s",
                image.id,
                instance_type,
                key_pair.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.instance
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件RunInstances中的 Python (博托 3) API 參考。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    " Create tags for resource created during instance launch. "
    DATA lt_tagspecifications TYPE /aws1/
cl_ec2tagspecification=>tt_tagspecificationlist.
    DATA ls_tagspecifications LIKE LINE OF lt_tagspecifications.
    ls_tagspecifications =  NEW /aws1/cl_ec2tagspecification(
      iv_resourcetype = 'instance'
      it_tags = VALUE /aws1/cl_ec2tag=>tt_taglist(
        ( NEW /aws1/cl_ec2tag( iv_key = 'Name' iv_value = iv_tag_value ) )
      )
    ).
    APPEND ls_tagspecifications TO lt_tagspecifications.

    TRY.
        " Create/launch Amazon Elastic Compute Cloud (Amazon EC2) instance. "
        oo_result = lo_ec2->runinstances(                           " oo_result
 is returned for testing purposes. "
          iv_imageid = iv_ami_id
          iv_instancetype = 't2.micro'
          iv_maxcount = 1
          iv_mincount = 1
          it_tagspecifications = lt_tagspecifications
          iv_subnetid = iv_subnet_id
        ).
        MESSAGE 'EC2 instance created.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK RunInstances中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭RunScheduledInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用RunScheduledInstances。

CLI

AWS CLI

啟動排程執行個體

此範例會在 VPC 中啟動指定的排程執行個體。

命令：

```
aws ec2 run-scheduled-instances --scheduled-instance-id
 sci-1234-1234-1234-1234-123456789012 --instance-count 1 --launch-specification
 file://launch-specification.json
```

啟動規格. JSON：

```
{
  "ImageId": "ami-12345678",
  "KeyName": "my-key-pair",
  "InstanceType": "c4.large",
  "NetworkInterfaces": [
    {
        "DeviceIndex": 0,
        "SubnetId": "subnet-12345678",
        "AssociatePublicIpAddress": true,
        "Groups": ["sg-12345678"]
    }
  ],
  "IamInstanceProfile": {
      "Name": "my-iam-role"
  }
}
```

輸出：

```
{
  "InstanceIdSet": [
      "i-1234567890abcdef0"
  ]
}
```

此範例會在 EC2-Classic 中啟動指定的排程執行個體。

命令：

```
aws ec2 run-scheduled-instances --scheduled-instance-id
 sci-1234-1234-1234-1234-123456789012 --instance-count 1 --launch-specification
 file://launch-specification.json
```

啟動規格. JSON：

```
{
  "ImageId": "ami-12345678",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": ["sg-12345678"],
  "InstanceType": "c4.large",
  "Placement": {
    "AvailabilityZone": "us-west-2b"
  }
  "IamInstanceProfile": {
      "Name": "my-iam-role"
  }
}
```

輸出：

```
{
  "InstanceIdSet": [
      "i-1234567890abcdef0"
  ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考RunScheduledInstances中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會啟動指定的排程執行個體。

```
New-EC2ScheduledInstance -ScheduledInstanceId
 sci-1234-1234-1234-1234-123456789012 -InstanceCount 1 `
-IamInstanceProfile_Name my-iam-role `
-LaunchSpecification_ImageId ami-12345678 `
-LaunchSpecification_InstanceType c4.large `
-LaunchSpecification_SubnetId subnet-12345678`
-LaunchSpecification_SecurityGroupId sg-12345678
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程RunScheduledInstances式參考中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭StartInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用StartInstances。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
```

```
    /// Start an EC2 instance.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the Amazon EC2 instance
    /// to start.</param>
    /// <returns>Async task.</returns>
    public async Task StartInstances(string ec2InstanceId)
    {
        var request = new StartInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        var response = await _amazonEC2.StartInstancesAsync(request);

        if (response.StartingInstances.Count > 0)
        {
            var instances = response.StartingInstances;
            instances.ForEach(i =>
            {
                Console.WriteLine($"Successfully started the EC2 instance with
instance ID: {i.InstanceId}.");
            });
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考StartInstances中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
#############################################################################
# function ec2_start_instances
#
```

```
# This function starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to start (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_start_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_start_instances"
    echo "Starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to start (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
```

```
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 start-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports start-instances operation failed with $response."
    return 1
  }

  return 0
}
```

此範例中使用的公用程式函數。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
```

```
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考StartInstances中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::StartInstancesRequest start_request;
    start_request.AddInstanceIds(instanceId);
    start_request.SetDryRun(true);

    auto dry_run_outcome = ec2Client.StartInstances(start_request);
    if (dry_run_outcome.IsSuccess()) {
        std::cerr
```

```
                << "Failed dry run to start instance. A dry run should trigger an
error."
                << std::endl;
        return false;
    }
    else if (dry_run_outcome.GetError().GetErrorType() !=
             Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to start instance " << instanceId << ": "
                  << dry_run_outcome.GetError().GetMessage() << std::endl;
        return false;
    }

    start_request.SetDryRun(false);
    auto start_instancesOutcome = ec2Client.StartInstances(start_request);

    if (!start_instancesOutcome.IsSuccess()) {
        std::cout << "Failed to start instance " << instanceId << ": " <<
                  start_instancesOutcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully started instance " << instanceId <<
                  std::endl;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考StartInstances中的。

CLI

AWS CLI

啟動 Amazon EC2 執行個體

此範例會啟動指定且受 Amazon EBS 支援的執行個體。

命令：

```
aws ec2 start-instances --instance-ids i-1234567890abcdef0
```

輸出：

```
{
```

```
    "StartingInstances": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "CurrentState": {
                "Code": 0,
                "Name": "pending"
            },
            "PreviousState": {
                "Code": 80,
                "Name": "stopped"
            }
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon Elastic Compute Cloud 使用者指南》中的「停止和啟動執行個體」。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考StartInstances中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    public static void startInstance(Ec2Client ec2, String instanceId) {
        Ec2Waiter ec2Waiter = Ec2Waiter.builder()
                .overrideConfiguration(b -> b.maxAttempts(100))
                .client(ec2)
                .build();

        StartInstancesRequest request = StartInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        System.out.println("Use an Ec2Waiter to wait for the instance to run.
 This will take a few minutes.");
```

```
        ec2.startInstances(request);
        DescribeInstancesRequest instanceRequest =
 DescribeInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        WaiterResponse<DescribeInstancesResponse> waiterResponse =
 ec2Waiter.waitUntilInstanceRunning(instanceRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Successfully started instance " + instanceId);
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考StartInstances中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { StartInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new StartInstancesCommand({
    // Use DescribeInstancesCommand to find InstanceIds
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { StartingInstances } = await client.send(command);
    const instanceIdList = StartingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Starting instances:");
    console.log(instanceIdList.join("\n"));
```

```
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考StartInstances中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun startInstanceSc(instanceId: String) {
    val request =
        StartInstancesRequest {
            instanceIds = listOf(instanceId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.startInstances(request)
        println("Waiting until instance $instanceId starts. This will take a few
 minutes.")
        ec2.waitUntilInstanceRunning {
            // suspend call
            instanceIds = listOf(instanceId)
        }
        println("Successfully started instance $instanceId")
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK StartInstances中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會啟動指定的執行個體。

```
Start-EC2Instance -InstanceId i-12345678
```

輸出：

```
CurrentState                      InstanceId    PreviousState
------------                      ----------    -------------
Amazon.EC2.Model.InstanceState    i-12345678    Amazon.EC2.Model.InstanceState
```

範例 2：此範例會啟動指定的執行個體。

```
@("i-12345678", "i-76543210") | Start-EC2Instance
```

範例 3：此範例會啟動目前已停止的執行個體集。傳回的執行個體物件Get-EC2Instance會傳送至Start-EC2Instance。此範例使用的語法需要 PowerShell 版本 3 或更高版本。

```
(Get-EC2Instance -Filter @{ Name="instance-state-name";
 Values="stopped"}).Instances | Start-EC2Instance
```

範例 4：在 PowerShell 版本 2 中，您必須使用新物件來建立篩選器參數的篩選器。

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "instance-state-name"
$filter.Values = "stopped"

(Get-EC2Instance -Filter $filter).Instances | Start-EC2Instance
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程StartInstances式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
                          wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def start(self):
        """
        Starts an instance and waits for it to be in a running state.

        :return: The response to the start request.
        """
        if self.instance is None:
            logger.info("No instance to start.")
            return
```

```
        try:
            response = self.instance.start()
            self.instance.wait_until_running()
        except ClientError as err:
            logger.error(
                "Couldn't start instance %s. Here's why: %s: %s",
                self.instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件StartInstances中的 Python (博托 3) API 參考。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
require "aws-sdk-ec2"

# Attempts to start an Amazon Elastic Compute Cloud (Amazon EC2) instance.
#
# Prerequisites:
#
# - The Amazon EC2 instance.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param instance_id [String] The ID of the instance.
# @return [Boolean] true if the instance was started; otherwise, false.
# @example
```

```
#   exit 1 unless instance_started?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'i-123abc'
#   )
def instance_started?(ec2_client, instance_id)
  response = ec2_client.describe_instance_status(instance_ids: [instance_id])

  if response.instance_statuses.count.positive?
    state = response.instance_statuses[0].instance_state.name
    case state
    when "pending"
      puts "Error starting instance: the instance is pending. Try again later."
      return false
    when "running"
      puts "The instance is already running."
      return true
    when "terminated"
      puts "Error starting instance: " \
        "the instance is terminated, so you cannot start it."
      return false
    end
  end

  ec2_client.start_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_running, instance_ids: [instance_id])
  puts "Instance started."
  return true
rescue StandardError => e
  puts "Error starting instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-start-instance-i-123abc.rb " \
      "INSTANCE_ID REGION "
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-start-instance-i-123abc.rb " \
      "i-123abc us-west-2"
    exit 1
```

```
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    instance_id = "i-123abc"
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    instance_id = ARGV[0]
    region = ARGV[1]
  end

  ec2_client = Aws::EC2::Client.new(region: region)

  puts "Attempting to start instance '#{instance_id}' " \
    "(this might take a few minutes)..."
  unless instance_started?(ec2_client, instance_id)
    puts "Could not start instance."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考StartInstances中的。

Rust

適用於 Rust 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
async fn start_instance(client: &Client, id: &str) -> Result<(), Error> {
    // start_instance has no unique errors to handle.
    client.start_instances().instance_ids(id).send().await?;

    println!("Waiting for instance to be running");

    let wait_for_running = client
```

```
        .wait_until_instance_running()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await;

    match wait_for_running {
        Ok(_) => println!("Instance is running"),
        Err(err) => match err {
            WaiterError::ExceededMaxWait(exceeded) => {
                println!(
                    "Exceeded max time waiting for instance to start. Exceeded
 {}s by {}s.",
                    exceeded.max_wait().as_secs(),
                    (exceeded.elapsed() - exceeded.max_wait()).as_secs()
                );
                return Ok(());
            }
            _ => return Err(err.into()),
        },
    }

    println!("Started instance.");

    Ok(())
}
```

- 如需 API 的詳細資訊，請參閱 AWS SDK StartInstances中的 Rust API 參考資料。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
```

```
    APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
 lt_instance_ids.

    "Perform dry run"
    TRY.
        " DryRun is set to true. This checks for the required permissions to
 start the instance without actually making the request. "
        lo_ec2->startinstances(
          it_instanceids = lt_instance_ids
          iv_dryrun = abap_true
        ).
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        " If the error code returned is `DryRunOperation`, then you have the
 required permissions to start this instance. "
        IF lo_exception->av_err_code = 'DryRunOperation'.
          MESSAGE 'Dry run to start instance completed.' TYPE 'I'.
          " DryRun is set to false to start instance. "
          oo_result = lo_ec2->startinstances(           " oo_result is returned
 for testing purposes. "
            it_instanceids = lt_instance_ids
            iv_dryrun = abap_false
          ).
          MESSAGE 'Successfully started the EC2 instance.' TYPE 'I'.
          " If the error code returned is `UnauthorizedOperation`, then you don't
 have the required permissions to start this instance. "
        ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
          MESSAGE 'Dry run to start instance failed. User does not have
 permissions to start the instance.' TYPE 'E'.
        ELSE.
          DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
          MESSAGE lv_error TYPE 'E'.
        ENDIF.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK StartInstances中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭StopInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用StopInstances。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
/// <summary>
/// Stop an EC2 instance.
/// </summary>
/// <param name="ec2InstanceId">The instance Id of the EC2 instance to
/// stop.</param>
/// <returns>Async task.</returns>
public async Task StopInstances(string ec2InstanceId)
{
    // In addition to the list of instance Ids, the
    // request can also include the following properties:
    //     Force      When true, forces the instances to
    //                stop but you must check the integrity
    //                of the file system. Not recommended on
    //                Windows instances.
    //     Hibernate  When true, hibernates the instance if the
    //                instance was enabled for hibernation when
    //                it was launched.
    var request = new StopInstancesRequest
    {
        InstanceIds = new List<string> { ec2InstanceId },
    };
```

```
        var response = await _amazonEC2.StopInstancesAsync(request);

        if (response.StoppingInstances.Count > 0)
        {
            var instances = response.StoppingInstances;
            instances.ForEach(i =>
            {
                Console.WriteLine($"Successfully stopped the EC2 Instance " +
                                  $"with InstanceID: {i.InstanceId}.");
            });
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考StopInstances中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
###############################################################################
# function ec2_stop_instances
#
# This function stops one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to stop (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_stop_instances() {
  local instance_ids
```

```
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_stop_instances"
    echo "Stops one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to stop (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 stop-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports stop-instances operation failed with $response."
    return 1
  }

  return 0
```

```
}
```

此範例中使用的公用程式函數。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
```

```
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考StopInstances中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::StopInstancesRequest request;
    request.AddInstanceIds(instanceId);
    request.SetDryRun(true);

    auto dry_run_outcome = ec2Client.StopInstances(request);
    if (dry_run_outcome.IsSuccess()) {
        std::cerr
                << "Failed dry run to stop instance. A dry run should trigger an
error."
                << std::endl;
        return false;
    }
    else if (dry_run_outcome.GetError().GetErrorType() !=
             Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to stop instance " << instanceId << ": "
                  << dry_run_outcome.GetError().GetMessage() << std::endl;
        return false;
    }

    request.SetDryRun(false);
    auto outcome = ec2Client.StopInstances(request);
    if (!outcome.IsSuccess()) {
        std::cout << "Failed to stop instance " << instanceId << ": " <<
```

```
                outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully stopped instance " << instanceId <<
                std::endl;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考StopInstances中的。

CLI

AWS CLI

範例 1：停止 Amazon EC2 執行個體

下列 `stop-instances` 範例會停止指定且受 Amazon EBS 支援的執行個體。

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0
```

輸出：

```
{
    "StoppingInstances": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon Elastic Compute Cloud 使用者指南》中的停止和啟動執行個體。

範例 2：讓 Amazon EC2 執行個體休眠

下列 `stop-instances` 範例會在 Amazon EBS 已啟用休眠並符合休眠必要條件時，讓該執行個體進入休眠。執行個體進入休眠狀態之後，即停止運作。

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0 \
    --hibernate
```

輸出：

```
{
    "StoppingInstances": [
        {
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "InstanceId": "i-1234567890abcdef0",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon Elastic Compute Cloud 使用者指南》中的讓您的隨需 Linux 執行個體休眠。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考StopInstances中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    public static void stopInstance(Ec2Client ec2, String instanceId) {
        Ec2Waiter ec2Waiter = Ec2Waiter.builder()
                .overrideConfiguration(b -> b.maxAttempts(100))
                .client(ec2)
                .build();
        StopInstancesRequest request = StopInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        System.out.println("Use an Ec2Waiter to wait for the instance to stop.
This will take a few minutes.");
        ec2.stopInstances(request);
        DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        WaiterResponse<DescribeInstancesResponse> waiterResponse =
ec2Waiter.waitUntilInstanceStopped(instanceRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Successfully stopped instance " + instanceId);
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考StopInstances中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { StopInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
```

```
  const command = new StopInstancesCommand({
    // Use DescribeInstancesCommand to find InstanceIds
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { StoppingInstances } = await client.send(command);
    const instanceIdList = StoppingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Stopping instances:");
    console.log(instanceIdList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考StopInstances中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun stopInstanceSc(instanceId: String) {
    val request =
        StopInstancesRequest {
            instanceIds = listOf(instanceId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.stopInstances(request)
        println("Waiting until instance $instanceId stops. This will take a few
 minutes.")
        ec2.waitUntilInstanceStopped {
            // suspend call
```

```
            instanceIds = listOf(instanceId)
        }
        println("Successfully stopped instance $instanceId")
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK StopInstances中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會停止指定的執行個體。

```
Stop-EC2Instance -InstanceId i-12345678
```

輸出：

```
CurrentState                      InstanceId    PreviousState
------------                      ----------    -------------
Amazon.EC2.Model.InstanceState    i-12345678    Amazon.EC2.Model.InstanceState
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程StopInstances式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
```

```
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
that
                            wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def stop(self):
        """
        Stops an instance and waits for it to be in a stopped state.

        :return: The response to the stop request.
        """
        if self.instance is None:
            logger.info("No instance to stop.")
            return

        try:
            response = self.instance.stop()
            self.instance.wait_until_stopped()
        except ClientError as err:
            logger.error(
                "Couldn't stop instance %s. Here's why: %s: %s",
                self.instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件StopInstances中的 Python (博托 3) API 參考。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
require "aws-sdk-ec2"

# Prerequisites:
#
# - The Amazon EC2 instance.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param instance_id [String] The ID of the instance.
# @return [Boolean] true if the instance was stopped; otherwise, false.
# @example
#   exit 1 unless instance_stopped?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'i-123abc'
#   )
def instance_stopped?(ec2_client, instance_id)
  response = ec2_client.describe_instance_status(instance_ids: [instance_id])

  if response.instance_statuses.count.positive?
    state = response.instance_statuses[0].instance_state.name
    case state
    when "stopping"
      puts "The instance is already stopping."
      return true
    when "stopped"
      puts "The instance is already stopped."
      return true
    when "terminated"
      puts "Error stopping instance: " \
        "the instance is terminated, so you cannot stop it."
```

```
      return false
    end
  end

  ec2_client.stop_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_stopped, instance_ids: [instance_id])
  puts "Instance stopped."
  return true
rescue StandardError => e
  puts "Error stopping instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-stop-instance-i-123abc.rb " \
      "INSTANCE_ID REGION "
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-start-instance-i-123abc.rb " \
      "i-123abc us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    instance_id = "i-123abc"
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    instance_id = ARGV[0]
    region = ARGV[1]
  end

  ec2_client = Aws::EC2::Client.new(region: region)

  puts "Attempting to stop instance '#{instance_id}' " \
    "(this might take a few minutes)..."
  unless instance_stopped?(ec2_client, instance_id)
    puts "Could not stop instance."
  end
end
```

```
run_me if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考StopInstances中的。

Rust

適用於 Rust 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
async fn stop_instance(client: &Client, id: &str) -> Result<(), Error> {
    client.stop_instances().instance_ids(id).send().await?;

    println!("Stopping instance...");

    let wait = client
        .wait_until_instance_stopped()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await;

    match wait {
        Ok(_) => {
            println!("Stopped instance.");
        }
        Err(err) => match err {
            WaiterError::ExceededMaxWait(exceeded) => {
                println!(
                    "Exceeded max time waiting for instance to stop. Exceeded {}s
 by {}s",
                    exceeded.max_wait().as_secs(),
                    (exceeded.elapsed() - exceeded.max_wait()).as_secs()
                )
            }
            _ => return Err(err.into()),
        },
```

```
    };
    Ok(())
}
```

- 如需 API 的詳細資訊，請參閱 AWS SDK StopInstances中的 Rust API 參考資料。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
    APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
 lt_instance_ids.

    "Perform dry run"
    TRY.
        " DryRun is set to true. This checks for the required permissions to stop
 the instance without actually making the request. "
        lo_ec2->stopinstances(
          it_instanceids = lt_instance_ids
          iv_dryrun = abap_true
        ).
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        " If the error code returned is `DryRunOperation`, then you have the
 required permissions to stop this instance. "
        IF lo_exception->av_err_code = 'DryRunOperation'.
          MESSAGE 'Dry run to stop instance completed.' TYPE 'I'.
          " DryRun is set to false to stop instance. "
          oo_result = lo_ec2->stopinstances(           " oo_result is returned
 for testing purposes. "
            it_instanceids = lt_instance_ids
            iv_dryrun = abap_false
          ).
```

```
          MESSAGE 'Successfully stopped the EC2 instance.' TYPE 'I'.
          " If the error code returned is `UnauthorizedOperation`, then you don't
 have the required permissions to stop this instance. "
        ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
          MESSAGE 'Dry run to stop instance failed. User does not have
 permissions to stop the instance.' TYPE 'E'.
        ELSE.
          DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
          MESSAGE lv_error TYPE 'E'.
        ENDIF.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱 AWS SDK StopInstances中的 SAP ABAP API 參考資料。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭TerminateInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用TerminateInstances。

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：

- 開始使用執行個體

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Terminate an EC2 instance.
```

```
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the EC2 instance
    /// to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task<List<InstanceStateChange>> TerminateInstances(string
 ec2InstanceId)
    {
        var request = new TerminateInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId }
        };

        var response = await _amazonEC2.TerminateInstancesAsync(request);
        return response.TerminatingInstances;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考TerminateInstances中的。

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
#############################################################################
# function ec2_terminate_instances
#
# This function terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances using the AWS CLI.
#
# Parameters:
#       -i instance_ids - A space-separated list of instance IDs.
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
################################################################################
function ec2_terminate_instances() {
  local instance_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_terminate_instances"
    echo "Terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_ids - A space-separated list of instance IDs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Check if instance ID is provided
  if [[ -z "${instance_ids}" ]]; then
    echo "Error: Missing required instance IDs parameter."
    usage
    return 1
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 terminate-instances \
    "--instance-ids" $instance_ids \
    --query 'TerminatingInstances[*].[InstanceId,CurrentState.Name]' \
    --output text) || {
```

```
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports terminate-instances operation failed.$response"
    return 1
  }

  return 0
}
```

此範例中使用的公用程式函數。

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
########################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
```

```
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考TerminateInstances中的。

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::TerminateInstancesRequest request;
    request.SetInstanceIds({instanceID});

    Aws::EC2::Model::TerminateInstancesOutcome outcome =
            ec2Client.TerminateInstances(request);
    if (outcome.IsSuccess()) {
        std::cout << "Ec2 instance '" << instanceID <<
                  "' was terminated." << std::endl;
    }
    else {
        std::cerr << "Failed to terminate ec2 instance " << instanceID <<
                  ", " <<
                  outcome.GetError().GetMessage() << std::endl;
        return false;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考TerminateInstances中的。

CLI

AWS CLI

若要終止 Amazon EC2 執行個體

此範例會終止指定執行個體。

命令：

```
aws ec2 terminate-instances --instance-ids i-1234567890abcdef0
```

輸出：

```
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

如需詳細資訊，請參閱《AWS 命令行介面使用者指南》中的「Amazon EC2 執行個體」。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考TerminateInstances中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    public static void terminateEC2(Ec2Client ec2, String instanceId) {
        try {
            Ec2Waiter ec2Waiter = Ec2Waiter.builder()
                    .overrideConfiguration(b -> b.maxAttempts(100))
                    .client(ec2)
                    .build();

            TerminateInstancesRequest ti = TerminateInstancesRequest.builder()
                    .instanceIds(instanceId)
                    .build();

            System.out.println("Use an Ec2Waiter to wait for the instance to
terminate. This will take a few minutes.");
            ec2.terminateInstances(ti);
            DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
                    .instanceIds(instanceId)
                    .build();

            WaiterResponse<DescribeInstancesResponse> waiterResponse = ec2Waiter
                    .waitUntilInstanceTerminated(instanceRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Successfully started instance " + instanceId);
            System.out.println(instanceId + " is terminated!");

        } catch (Ec2Exception 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TerminateInstances中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { TerminateInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new TerminateInstancesCommand({
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { TerminatingInstances } = await client.send(command);
    const instanceList = TerminatingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Terminating instances:");
    console.log(instanceList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考TerminateInstances中的。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
suspend fun terminateEC2(instanceID: String) {
    val request =
        TerminateInstancesRequest {
            instanceIds = listOf(instanceID)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.terminateInstances(request)
        response.terminatingInstances?.forEach { instance ->
            println("The ID of the terminated instance is
 ${instance.instanceId}")
        }
    }
}
```

- 有關 API 的詳細信息，請參閱 AWS SDK TerminateInstances中的 Kotlin API 參考。

PowerShell

適用的工具 PowerShell

範例 1：此範例會終止指定的執行個體 (執行個體可能正在執行中或處於「已停止」狀態)。指令程式會在繼續之前提示確認；請使用-Force 參數來隱藏提示。

```
Remove-EC2Instance -InstanceId i-12345678
```

輸出：

```
CurrentState                    InstanceId    PreviousState
------------                    ----------    -------------
```

```
Amazon.EC2.Model.InstanceState    i-12345678    Amazon.EC2.Model.InstanceState
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程TerminateInstances式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
                         wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def terminate(self):
        """
        Terminates an instance and waits for it to be in a terminated state.
```

```
        """
        if self.instance is None:
            logger.info("No instance to terminate.")
            return

        instance_id = self.instance.id
        try:
            self.instance.terminate()
            self.instance.wait_until_terminated()
            self.instance = None
        except ClientError as err:
            logging.error(
                "Couldn't terminate instance %s. Here's why: %s: %s",
                instance_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件TerminateInstances中的 Python (博托 3) API 參考。

Ruby

適用於 Ruby 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
require "aws-sdk-ec2"

# Prerequisites:
#
# - The Amazon EC2 instance.
#
```

```
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param instance_id [String] The ID of the instance.
# @return [Boolean] true if the instance was terminated; otherwise, false.
# @example
#   exit 1 unless instance_terminated?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'i-123abc'
#   )
def instance_terminated?(ec2_client, instance_id)
  response = ec2_client.describe_instance_status(instance_ids: [instance_id])

  if response.instance_statuses.count.positive? &&
    response.instance_statuses[0].instance_state.name == "terminated"

    puts "The instance is already terminated."
    return true
  end

  ec2_client.terminate_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_terminated, instance_ids: [instance_id])
  puts "Instance terminated."
  return true
rescue StandardError => e
  puts "Error terminating instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-terminate-instance-i-123abc.rb " \
      "INSTANCE_ID REGION "
   # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-terminate-instance-i-123abc.rb " \
      "i-123abc us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    instance_id = "i-123abc"
    region = "us-west-2"
```

```
  # Otherwise, use the values as specified at the command prompt.
  else
    instance_id = ARGV[0]
    region = ARGV[1]
  end

  ec2_client = Aws::EC2::Client.new(region: region)

  puts "Attempting to terminate instance '#{instance_id}' " \
    "(this might take a few minutes)..."
  unless instance_terminated?(ec2_client, instance_id)
    puts "Could not terminate instance."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Ruby API 參考TerminateInstances中的。

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭UnassignPrivateIpAddresses配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用UnassignPrivateIpAddresses。

CLI

AWS CLI

從網路介面取消指派次要私人 IP 位址

此範例會從指定的網路介面取消指定的私有 IP 位址。如果命令成功，則不會傳回任何輸出。

命令：

```
aws ec2 unassign-private-ip-addresses --network-interface-id eni-e5aa89a3 --
private-ip-addresses 10.0.0.82
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考UnassignPrivateIpAddresses中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例會從指定的網路介面取消指定的私有 IP 位址。

```
Unregister-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -PrivateIpAddress
 10.0.0.82
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程[UnassignPrivateIpAddresses](#)式參考中的。

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 開發套件建立 Amazon EC2 資源](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭UnmonitorInstances配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用UnmonitorInstances。

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：

- [開始使用執行個體](#)

C++

適用於 C++ 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在[AWS 設定和執行程式碼範例儲存庫](#)。

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::UnmonitorInstancesRequest unrequest;
    unrequest.AddInstanceIds(instanceId);
    unrequest.SetDryRun(true);

    auto undryRunOutcome = ec2Client.UnmonitorInstances(unrequest);
```

```
    if (undryRunOutcome.IsSuccess()) {
        std::cerr
                << "Failed dry run to disable monitoring on instance. A dry run
should trigger an error."
                <<
                std::endl;
        return false;
    }
    else if (undryRunOutcome.GetError().GetErrorType() !=
             Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to disable monitoring on instance " <<
                  instanceId << ": " << undryRunOutcome.GetError().GetMessage()
<<
                  std::endl;
        return false;
    }

    unrequest.SetDryRun(false);
    auto unmonitorInstancesOutcome = ec2Client.UnmonitorInstances(unrequest);
    if (!unmonitorInstancesOutcome.IsSuccess()) {
        std::cout << "Failed to disable monitoring on instance " << instanceId
                  << ": " << unmonitorInstancesOutcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully disable monitoring on instance " <<
                  instanceId << std::endl;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for C++ API 參考[UnmonitorInstances](#)中的。

CLI

AWS CLI

停用執行個體的詳細監控

這個範例命令會停用指定執行個體的詳細監控。

命令：

```
aws ec2 unmonitor-instances --instance-ids i-1234567890abcdef0
```

輸出：

```
{
  "InstanceMonitorings": [
      {
          "InstanceId": "i-1234567890abcdef0",
          "Monitoring": {
              "State": "disabling"
          }
      }
  ]
}
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考UnmonitorInstances中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import { UnmonitorInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new UnmonitorInstancesCommand({
    InstanceIds: ["i-09a3dfe7ae00e853f"],
  });

  try {
    const { InstanceMonitorings } = await client.send(command);
    const instanceMonitoringsList = InstanceMonitorings.map(
      (im) =>
        ` • Detailed monitoring state for ${im.InstanceId} is
 ${im.Monitoring.State}.`,
    );
```

```
    console.log("Monitoring status:");
    console.log(instanceMonitoringsList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考UnmonitorInstances中的。

PowerShell

適用的工具 PowerShell

範例 1：此範例停用指定執行個體的詳細監視。

```
Stop-EC2InstanceMonitoring -InstanceId i-12345678
```

輸出：

```
InstanceId    Monitoring
----------    ----------
i-12345678    Amazon.EC2.Model.Monitoring
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程UnmonitorInstances式參考中的。

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 開發套件建立 Amazon EC2 資源。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 開發套件的 Amazon EC2 案例

下列程式碼範例說明如何使用 AWS 開發套件在 Amazon EC2 中實作常見案例。這些案例會展示如何在 Amazon EC2 中呼叫多個函數來完成特定任務。每個案例都包含一個連結 GitHub，您可以在其中找到如何設定和執行程式碼的指示。

範例

- 使用 AWS SDK 建置及管理彈性服務
- 使用開 AWS 發套件開始使用 Amazon EC2 執行個體

使用 AWS SDK 建置及管理彈性服務

下列程式碼範例示範如何建立負載平衡的 Web 服務，以傳回書籍、影片和歌曲建議。此範例顯示服務如何回應失故障，以及如何在發生故障時重組服務以提高復原能力。

- 使用 Amazon EC2 Auto Scaling 群組根據啟動範本建立 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，並將執行個體數量保持在指定範圍內。
- 使用 Elastic Load Balancing 處理和分發 HTTP 請求。
- 監控 Auto Scaling 群組中執行個體的運作狀態，並且只將請求轉送給運作良好的執行個體。
- 在每個 EC2 執行個體上執行一個 Python Web 伺服器來處理 HTTP 請求。Web 伺服器會回應建議和運作狀態檢查。
- 使用 Amazon DynamoDB 資料表模擬建議服務。
- 透過更新 AWS Systems Manager 參數來控制 Web 伺服器對要求和健康狀態檢查的回應。

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

在命令提示中執行互動式案例。

```
static async Task Main(string[] args)
{
    _configuration = new ConfigurationBuilder()
        .SetBasePath(Directory.GetCurrentDirectory())
        .AddJsonFile("settings.json") // Load settings from .json file.
        .AddJsonFile("settings.local.json",
            true) // Optionally, load local settings.
        .Build();


    // Set up dependency injection for the AWS services.
    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<IAmazonIdentityManagementService>()
                    .AddAWSService<IAmazonDynamoDB>()
                    .AddAWSService<IAmazonElasticLoadBalancingV2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddAWSService<IAmazonAutoScaling>()
                    .AddAWSService<IAmazonEC2>()
                    .AddTransient<AutoScalerWrapper>()
                    .AddTransient<ElasticLoadBalancerWrapper>()
                    .AddTransient<SmParameterWrapper>()
                    .AddTransient<Recommendations>()
                    .AddSingleton<IConfiguration>(_configuration)
            )
            .Build();

        ServicesSetup(host);
        ResourcesSetup();

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Resilient Architecture Example
Scenario.");
            Console.WriteLine(new string('-', 80));
            await Deploy(true);

            Console.WriteLine("Now let's begin the scenario.");
            Console.WriteLine(new string('-', 80));
            await Demo(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Finally, let's clean up our resources.");
            Console.WriteLine(new string('-', 80));

            await DestroyResources(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Resilient Architecture Example Scenario is
complete.");
```

```
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
            await DestroyResources(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Setup any common resources, also used for integration testing.
    /// </summary>
    public static void ResourcesSetup()
    {
        _httpClient = new HttpClient();
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _elasticLoadBalancerWrapper =
host.Services.GetRequiredService<ElasticLoadBalancerWrapper>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
        _recommendations = host.Services.GetRequiredService<Recommendations>();
        _autoScalerWrapper =
host.Services.GetRequiredService<AutoScalerWrapper>();
        _smParameterWrapper =
host.Services.GetRequiredService<SmParameterWrapper>();
    }

    /// <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> Deploy(bool interactive)
    {
```

```
        var protocol = "HTTP";
        var port = 80;
        var sshPort = 22;

        Console.WriteLine(
            "\nFor this demo, we'll use the AWS SDK for .NET to create several
AWS resources\n" +
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n" +
            "against various kinds of failures.\n\n" +
            "Some of the resources create by this demo are:\n");

        Console.WriteLine(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations.");
        Console.WriteLine(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server.");
        Console.WriteLine(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones.");
        Console.WriteLine(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to start deploying
resources.");
        if (interactive)
            Console.ReadLine();

        // Create and populate the DynamoDB table.
        var databaseTableName = _configuration["databaseName"];
        var recommendationsPath = Path.Join(_configuration["resourcePath"],
            "recommendations_objects.json");
        Console.WriteLine($"Creating and populating a DynamoDB table named
{databaseTableName}.");
        await _recommendations.CreateDatabaseWithName(databaseTableName);
        await _recommendations.PopulateDatabase(databaseTableName,
recommendationsPath);
        Console.WriteLine(new string('-', 80));

        // Create the EC2 Launch Template.

        Console.WriteLine(
```

```
            $"Creating an EC2 launch template that runs
'server_startup_script.sh' when an instance starts.\n"
            + "\nThis script starts a Python web server defined in the
`server.py` script. The web server\n"
            + "listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.\n"
            + "For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            + "run a web server, such as Apache, with least-privileged
credentials.");
        Console.WriteLine(
            "\nThe template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            + "permissions to access the DynamoDB recommendation table and
Systems Manager parameters\n"
            + "that control the flow of the demo.");

        var startupScriptPath = Path.Join(_configuration["resourcePath"],
            "server_startup_script.sh");
        var instancePolicyPath = Path.Join(_configuration["resourcePath"],
            "instance_policy.json");
        await _autoScalerWrapper.CreateTemplate(startupScriptPath,
instancePolicyPath);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            + "Availability Zone.\n");
        var zones = await _autoScalerWrapper.DescribeAvailabilityZones();
        await _autoScalerWrapper.CreateGroupOfSize(3,
_autoScalerWrapper.GroupName, zones);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            + "HTTP requests. You can see these instances in the console or
continue with the demo.\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue.");
        if (interactive)
            Console.ReadLine();
```

```
        Console.WriteLine("Creating variables that control the flow of the
 demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine(
            "\nCreating an Elastic Load Balancing target group and load balancer.
 The target group\n"
            + "defines how the load balancer connects to instances. The load
 balancer provides a\n"
            + "single endpoint where clients connect and dispatches requests to
 instances in the group.");

        var defaultVpc = await _autoScalerWrapper.GetDefaultVpc();
        var subnets = await
 _autoScalerWrapper.GetAllVpcSubnetsForZones(defaultVpc.VpcId, zones);
        var subnetIds = subnets.Select(s => s.SubnetId).ToList();
        var targetGroup = await
 _elasticLoadBalancerWrapper.CreateTargetGroupOnVpc(_elasticLoadBalancerWrapper.TargetGro
 protocol, port, defaultVpc.VpcId);

        await
 _elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
 subnetIds, targetGroup);
        await
 _autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
 targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
 _elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
 _elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);

        if (!loadBalancerAccess)
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
 that the port is open...");

            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
 _autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
```

```
            var portIsOpen =
_autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
ipString);
            var sshPortIsOpen =
_autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
ipString);

            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
your default VPC must\n"
                    + "allows access from this computer. You can either add it
automatically from this\n"
                    + "example or add it yourself using the AWS Management
Console.\n");

                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
inbound traffic from your computer's IP address?"))
                {
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
inbound SSH traffic for debugging from your computer's IP address?"))
                {
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
ipString);
                }
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
```

```
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        else
        {
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
```

```
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
                          "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
        Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
        Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Let's reinstate the recommendation service.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
        Console.WriteLine(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
```

```
            "access the DynamoDB recommendation table.\n"
        );
        await _autoScalerWrapper.CreateInstanceProfileWithName(
            _autoScalerWrapper.BadCredsPolicyName,
            _autoScalerWrapper.BadCredsRoleName,
            _autoScalerWrapper.BadCredsProfileName,
            ssmOnlyPolicy,
            new List<string> { "AmazonSSMManagedInstanceCore" }
        );
        var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
        var badInstanceId = instances.First();
        var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
        Console.WriteLine(
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");
```

```
        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");
```

```
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");

        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );

        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
resources? (y/n) "))
        {
            await
_elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
            await
_elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
_autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
_autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
```

```
            await
_autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
            );
            await
_recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        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;
    }
```

建立包裝 Auto Scaling 和 Amazon EC2 動作的類別。

```
/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;

    private readonly string _instanceType = "";
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
```

```
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";

    public string GroupName => _groupName;
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;

        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
        _amiParam = configuration["amiParam"];

        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
```

```
    }

    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
    /// An instance's associated profile defines a role that is assumed by the
    /// instance.The role has attached policies that specify the AWS permissions
granted to
    /// clients that run on the instance.
    /// </summary>
    /// <param name="policyName">Name to use for the policy.</param>
    /// <param name="roleName">Name to use for the role.</param>
    /// <param name="profileName">Name to use for the profile.</param>
    /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
    /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
    /// <returns>The Arn of the profile.</returns>
    public async Task<string> CreateInstanceProfileWithName(
        string policyName,
        string roleName,
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                    "\"Version\": \"2012-10-17\"," +
                                    "\"Statement\": [{" +
                                         "\"Effect\": \"Allow\"," +
                                         "\"Principal\": {" +
                                         "\"Service\": [" +
                                             "\"ec2.amazonaws.com\"" +
                                         "]" +
                                         "}," +
                                    "\"Action\": \"sts:AssumeRole\"" +
                                    "}]" +
                                "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
```

```
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
```

```
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
```

```
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }

    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
    {
        try
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
```

```
    }

    /// <summary>
    /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
    /// The launch template specifies a Bash script in its user data field that
runs after
    /// the instance is started. This script installs the Python packages and
starts a Python
    /// web server on the instance.
    /// </summary>
    /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
    /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
    /// <returns>The template object.</returns>
    public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
    {
        await CreateKeyPair(_keyPairName);
        await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName, _instanceProfileName, instancePolicyPath);

        var startServerText = await File.ReadAllTextAsync(startupScriptPath);
        var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(startServerText);

        var amiLatest = await _amazonSsm.GetParameterAsync(
            new GetParameterRequest() { Name = _amiParam });
        var amiId = amiLatest.Parameter.Value;
        var launchTemplateResponse = await _amazonEc2.CreateLaunchTemplateAsync(
            new CreateLaunchTemplateRequest()
            {
                LaunchTemplateName = _launchTemplateName,
                LaunchTemplateData = new RequestLaunchTemplateData()
                {
                    InstanceType = _instanceType,
                    ImageId = amiId,
                    IamInstanceProfile =
                        new

LaunchTemplateIamInstanceProfileSpecificationRequest()
                        {
                            Name = _instanceProfileName
                        },
```

```
                    KeyName = _keyPairName,
                    UserData = System.Convert.ToBase64String(plainTextBytes)
                }
            });
        return launchTemplateResponse.LaunchTemplate;

    }

    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
 Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
            new DescribeAvailabilityZonesRequest());
        return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
                        new
 Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
```

```
                    },
                MaxSize = groupSize,
                MinSize = groupSize
            });
        Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
size {groupSize}.");
    }
    catch (EntityAlreadyExistsException)
    {
        Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
    }
}

/// <summary>
/// Get the default VPC for the account.
/// </summary>
/// <returns>The default VPC object.</returns>
public async Task<Vpc> GetDefaultVpc()
{
    var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
        new DescribeVpcsRequest()
        {
            Filters = new List<Amazon.EC2.Model.Filter>()
            {
                new ("is-default", new List<string>() { "true" })
            }
        });
    return vpcResponse.Vpcs[0];
}

/// <summary>
/// Get all the subnets for a Vpc in a set of availability zones.
/// </summary>
/// <param name="vpcId">The Id of the Vpc.</param>
/// <param name="availabilityZones">The list of availability zones.</param>
/// <returns>The collection of subnet objects.</returns>
public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
{
    var subnets = new List<Subnet>();
    var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
        new DescribeSubnetsRequest()
        {
```

```
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("vpc-id", new List<string>() { vpcId}),
                    new ("availability-zone", availabilityZones),
                    new ("default-for-az", new List<string>() { "true" })
                }
            });

        // Get the entire list using the paginator.
        await foreach (var subnet in subnetPaginator.Subnets)
        {
            subnets.Add(subnet);
        }

        return subnets;
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonClientException)
        {
            Console.WriteLine($"Unable to delete template {templateName}.");
        }
    }

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
```

```
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
```

```
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
            new DescribeIamInstanceProfileAssociationsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("instance-id", new List<string>() { instanceId })
                },
            });
        return response.IamInstanceProfileAssociations[0];
    }

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
```

```
   /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
   /// used to restart the Python web server.
   /// </summary>
   /// <param name="instanceId">The Id of the instance to update.</param>
   /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
   /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
   /// <returns>Async task.</returns>
   public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
   {
       await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
               new ReplaceIamInstanceProfileAssociationRequest()
               {
                   AssociationId = associationId,
                   IamInstanceProfile = new IamInstanceProfileSpecification()
                   {
                       Name = credsProfileName
                   }
               });
       // Allow time before resetting.
       Thread.Sleep(25000);
       var instanceReady = false;
       var retries = 5;
       while (retries-- > 0 && !instanceReady)
       {
           await _amazonEc2.RebootInstancesAsync(
               new RebootInstancesRequest(new List<string>() { instanceId }));
           Thread.Sleep(10000);

           var instancesPaginator =
_amazonSsm.Paginators.DescribeInstanceInformation(
               new DescribeInstanceInformationRequest());
           // Get the entire list using the paginator.
           await foreach (var instance in
instancesPaginator.InstanceInformationList)
           {
               instanceReady = instance.InstanceId == instanceId;
               if (instanceReady)
               {
                   break;
               }
```

```
            }
        }
        Console.WriteLine($"Sending restart command to instance {instanceId}");
        await _amazonSsm.SendCommandAsync(
            new SendCommandRequest()
            {
                InstanceIds = new List<string>() { instanceId },
                DocumentName = "AWS-RunShellScript",
                Parameters = new Dictionary<string, List<string>>()
                {
                        {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
                }
            });
        Console.WriteLine($"Restarted the web server on instance {instanceId}");
    }

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
```

```
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
    {
```

```
        var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }


    /// <summary>
    /// Get the default security group for a specified Vpc.
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
```

```
                }
            });
        return groupResponse.SecurityGroups[0];
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
    /// This can be done by allowing ingress from this computer's IP address.
    /// In some situations, such as connecting from a corporate network, you must
instead specify
    /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
    /// If you do, be sure to remove public access when you're done.
    /// </summary>
    /// <param name="vpc">The group to check.</param>
    /// <param name="port">The port to verify.</param>
    /// <param name="ipAddress">This computer's IP address.</param>
    /// <returns>True if the ip address is allowed on the group.</returns>
    public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
    {
        var portIsOpen = false;
        foreach (var ipPermission in group.IpPermissions)
        {
            if (ipPermission.FromPort == port)
            {
                foreach (var ipRange in ipPermission.Ipv4Ranges)
                {
                    var cidr = ipRange.CidrIp;
                    if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                    {
                        portIsOpen = true;
                    }
                }

                if (ipPermission.PrefixListIds.Any())
                {
                    portIsOpen = true;
                }

                if (!portIsOpen)
                {
```

```
                    Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                      "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
                }
                else
                {
                    break;
                }
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
                    }
                }
            });
```

```
    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }
}
```

建立包裝 Elastic Load Balancing 動作的類別。

```
/// <summary>
/// Encapsulates Elastic Load Balancer actions.
/// </summary>
public class ElasticLoadBalancerWrapper
{
    private readonly IAmazonElasticLoadBalancingV2 _amazonElasticLoadBalancingV2;
    private string? _endpoint = null;
    private readonly string _targetGroupName = "";
    private readonly string _loadBalancerName = "";
    HttpClient _httpClient = new();

    public string TargetGroupName => _targetGroupName;
    public string LoadBalancerName => _loadBalancerName;

    /// <summary>
    /// Constructor for the Elastic Load Balancer wrapper.
```

```
    /// </summary>
    /// <param name="amazonElasticLoadBalancingV2">The injected load balancing v2
client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public ElasticLoadBalancerWrapper(
        IAmazonElasticLoadBalancingV2 amazonElasticLoadBalancingV2,
        IConfiguration configuration)
    {
        _amazonElasticLoadBalancingV2 = amazonElasticLoadBalancingV2;
        var prefix = configuration["resourcePrefix"];
        _targetGroupName = prefix + "-tg";
        _loadBalancerName = prefix + "-lb";
    }

    /// <summary>
    /// Get the HTTP Endpoint of a load balancer by its name.
    /// </summary>
    /// <param name="loadBalancerName">The name of the load balancer.</param>
    /// <returns>The HTTP endpoint.</returns>
    public async Task<string> GetEndpointForLoadBalancerByName(string
loadBalancerName)
    {
        if (_endpoint == null)
        {
            var endpointResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { loadBalancerName }
                    });
            _endpoint = endpointResponse.LoadBalancers[0].DNSName;
        }

        return _endpoint;
    }

    /// <summary>
    /// Return the GET response for an endpoint as text.
    /// </summary>
    /// <param name="endpoint">The endpoint for the request.</param>
    /// <returns>The request response.</returns>
    public async Task<string> GetEndPointResponse(string endpoint)
    {
        var endpointResponse = await _httpClient.GetAsync($"http://{endpoint}");
```

```
        var textResponse = await endpointResponse.Content.ReadAsStringAsync();
        return textResponse!;
    }

    /// <summary>
    /// Get the target health for a group by name.
    /// </summary>
    /// <param name="groupName">The name of the group.</param>
    /// <returns>The collection of health descriptions.</returns>
    public async Task<List<TargetHealthDescription>>
CheckTargetHealthForGroup(string groupName)
    {
        List<TargetHealthDescription> result = null!;
        try
        {
            var groupResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                    new DescribeTargetGroupsRequest()
                    {
                        Names = new List<string>() { groupName }
                    });
            var healthResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetHealthAsync(
                    new DescribeTargetHealthRequest()
                    {
                        TargetGroupArn =
groupResponse.TargetGroups[0].TargetGroupArn
                    });
            ;
            result = healthResponse.TargetHealthDescriptions;
        }
        catch (TargetGroupNotFoundException)
        {
            Console.WriteLine($"Target group {groupName} not found.");
        }
        return result;
    }

    /// <summary>
    /// Create an Elastic Load Balancing target group. The target group specifies
how the load balancer forwards
    /// requests to instances in the group and how instance health is checked.
    ///
```

```
    /// To speed up this demo, the health check is configured with shortened
 times and lower thresholds. In production,
    /// you might want to decrease the sensitivity of your health checks to avoid
 unwanted failures.
    /// </summary>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="protocol">The protocol, such as HTTP.</param>
    /// <param name="port">The port to use to forward requests, such as 80.</
param>
    /// <param name="vpcId">The Id of the Vpc in which the load balancer
 exists.</param>
    /// <returns>The new TargetGroup object.</returns>
    public async Task<TargetGroup> CreateTargetGroupOnVpc(string groupName,
 ProtocolEnum protocol, int port, string vpcId)
    {
        var createResponse = await
 _amazonElasticLoadBalancingV2.CreateTargetGroupAsync(
            new CreateTargetGroupRequest()
            {
                Name = groupName,
                Protocol = protocol,
                Port = port,
                HealthCheckPath = "/healthcheck",
                HealthCheckIntervalSeconds = 10,
                HealthCheckTimeoutSeconds = 5,
                HealthyThresholdCount = 2,
                UnhealthyThresholdCount = 2,
                VpcId = vpcId
            });
        var targetGroup = createResponse.TargetGroups[0];
        return targetGroup;
    }

    /// <summary>
    /// Create an Elastic Load Balancing load balancer that uses the specified
 subnets
    /// and forwards requests to the specified target group.
    /// </summary>
    /// <param name="name">The name for the new load balancer.</param>
    /// <param name="subnetIds">Subnets for the load balancer.</param>
    /// <param name="targetGroup">Target group for forwarded requests.</param>
    /// <returns>The new LoadBalancer object.</returns>
    public async Task<LoadBalancer> CreateLoadBalancerAndListener(string name,
 List<string> subnetIds, TargetGroup targetGroup)
```

```
    {
        var createLbResponse = await
_amazonElasticLoadBalancingV2.CreateLoadBalancerAsync(
            new CreateLoadBalancerRequest()
            {
                Name = name,
                Subnets = subnetIds
            });
        var loadBalancerArn = createLbResponse.LoadBalancers[0].LoadBalancerArn;

        // Wait for load balancer to be available.
        var loadBalancerReady = false;
        while (!loadBalancerReady)
        {
            try
            {
                var describeResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                        new DescribeLoadBalancersRequest()
                        {
                            Names = new List<string>() { name }
                        });

                var loadBalancerState =
describeResponse.LoadBalancers[0].State.Code;

                loadBalancerReady = loadBalancerState ==
LoadBalancerStateEnum.Active;
            }
            catch (LoadBalancerNotFoundException)
            {
                loadBalancerReady = false;
            }
            Thread.Sleep(10000);
        }
        // Create the listener.
        await _amazonElasticLoadBalancingV2.CreateListenerAsync(
            new CreateListenerRequest()
            {
                LoadBalancerArn = loadBalancerArn,
                Protocol = targetGroup.Protocol,
                Port = targetGroup.Port,
                DefaultActions = new List<Action>()
```

```
                {
                    new Action()
                    {
                        Type = ActionTypeEnum.Forward,
                        TargetGroupArn = targetGroup.TargetGroupArn
                    }
                }
            });
        return createLbResponse.LoadBalancers[0];
    }

    /// <summary>
    /// Verify this computer can successfully send a GET request to the
    /// load balancer endpoint.
    /// </summary>
    /// <param name="endpoint">The endpoint to check.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyLoadBalancerEndpoint(string endpoint)
    {
        var success = false;
        var retries = 3;
        while (!success && retries > 0)
        {
            try
            {
                var endpointResponse = await _httpClient.GetAsync($"http://
{endpoint}");
                Console.WriteLine($"Response: {endpointResponse.StatusCode}.");

                if (endpointResponse.IsSuccessStatusCode)
                {
                    success = true;
                }
                else
                {
                    retries = 0;
                }
            }
            catch (HttpRequestException)
            {
                Console.WriteLine("Connection error, retrying...");
                retries--;
                Thread.Sleep(10000);
            }
```

```
        }

        return success;
    }

    /// <summary>
    /// Delete a load balancer by its specified name.
    /// </summary>
    /// <param name="name">The name of the load balancer to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteLoadBalancerByName(string name)
    {
        try
        {
            var describeLoadBalancerResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { name }
                    });
            var lbArn =
describeLoadBalancerResponse.LoadBalancers[0].LoadBalancerArn;
            await _amazonElasticLoadBalancingV2.DeleteLoadBalancerAsync(
                new DeleteLoadBalancerRequest()
                {
                    LoadBalancerArn = lbArn
                }
            );
        }
        catch (LoadBalancerNotFoundException)
        {
            Console.WriteLine($"Load balancer {name} not found.");
        }
    }

    /// <summary>
    /// Delete a TargetGroup by its specified name.
    /// </summary>
    /// <param name="groupName">Name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTargetGroupByName(string groupName)
    {
        var done = false;
        while (!done)
```

```
        {
            try
            {
                var groupResponse =
                    await
 _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                        new DescribeTargetGroupsRequest()
                        {
                            Names = new List<string>() { groupName }
                        });

                var targetArn = groupResponse.TargetGroups[0].TargetGroupArn;
                await _amazonElasticLoadBalancingV2.DeleteTargetGroupAsync(
                    new DeleteTargetGroupRequest() { TargetGroupArn =
 targetArn });
                Console.WriteLine($"Deleted load balancing target group
 {groupName}.");
                done = true;
            }
            catch (TargetGroupNotFoundException)
            {
                Console.WriteLine(
                    $"Target group {groupName} not found, could not delete.");
                done = true;
            }
            catch (ResourceInUseException)
            {
                Console.WriteLine("Target group not yet released, waiting...");
                Thread.Sleep(10000);
            }
        }
    }
}
```

建立使用 DynamoDB 模擬建議服務的類別。

```
/// <summary>
/// Encapsulates a DynamoDB table to use as a service that recommends books,
 movies, and songs.
/// </summary>
public class Recommendations
{
```

```
    private readonly IAmazonDynamoDB _amazonDynamoDb;
    private readonly DynamoDBContext _context;
    private readonly string _tableName;

    public string TableName => _tableName;

    /// <summary>
    /// Constructor for the Recommendations service.
    /// </summary>
    /// <param name="amazonDynamoDb">The injected DynamoDb client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public Recommendations(IAmazonDynamoDB amazonDynamoDb, IConfiguration
configuration)
    {
        _amazonDynamoDb = amazonDynamoDb;
        _context = new DynamoDBContext(_amazonDynamoDb);
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Create the DynamoDb table with a specified name.
    /// </summary>
    /// <param name="tableName">The name for the table.</param>
    /// <returns>True when ready.</returns>
    public async Task<bool> CreateDatabaseWithName(string tableName)
    {
        try
        {
            Console.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
                AttributeDefinitions = new List<AttributeDefinition>()
                    {
                        new AttributeDefinition()
                        {
                            AttributeName = "MediaType",
                            AttributeType = ScalarAttributeType.S
                        },
                        new AttributeDefinition()
                        {
                            AttributeName = "ItemId",
                            AttributeType = ScalarAttributeType.N
                        }
```

```
                    },
                KeySchema = new List<KeySchemaElement>()
                    {
                        new KeySchemaElement()
                        {
                            AttributeName = "MediaType",
                            KeyType = KeyType.HASH
                        },
                        new KeySchemaElement()
                        {
                            AttributeName = "ItemId",
                            KeyType = KeyType.RANGE
                        }
                    },
                ProvisionedThroughput = new ProvisionedThroughput()
                {
                    ReadCapacityUnits = 5,
                    WriteCapacityUnits = 5
                }
            };
            await _amazonDynamoDb.CreateTableAsync(createRequest);

            // Wait until the table is ACTIVE and then report success.
            Console.Write("\nWaiting for table to become active...");

            var request = new DescribeTableRequest
            {
                TableName = tableName
            };

            TableStatus status;
            do
            {
                Thread.Sleep(2000);

                var describeTableResponse = await
_amazonDynamoDb.DescribeTableAsync(request);
                status = describeTableResponse.Table.TableStatus;

                Console.Write(".");
            }
            while (status != "ACTIVE");

            return status == TableStatus.ACTIVE;
```

```
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
            return false;
        }
    }

    /// <summary>
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }

    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
                new DeleteTableRequest() { TableName = tableName });
```

```
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

建立包裝 Systems Manager 動作的類別。

```
/// <summary>
/// Encapsulates Systems Manager parameter operations. This example uses these
 parameters
/// to drive the demonstration of resilient architecture, such as failure of a
 dependency or
/// how the service responds to a health check.
/// </summary>
public class SmParameterWrapper
{
    private readonly IAmazonSimpleSystemsManagement
 _amazonSimpleSystemsManagement;

    private readonly string _tableParameter = "doc-example-resilient-
architecture-table";
    private readonly string _failureResponseParameter = "doc-example-resilient-
architecture-failure-response";
    private readonly string _healthCheckParameter = "doc-example-resilient-
architecture-health-check";
    private readonly string _tableName = "";

    public string TableParameter => _tableParameter;
    public string TableName => _tableName;
    public string HealthCheckParameter => _healthCheckParameter;
    public string FailureResponseParameter => _failureResponseParameter;

    /// <summary>
    /// Constructor for the SmParameterWrapper.
    /// </summary>
    /// <param name="amazonSimpleSystemsManagement">The injected Simple Systems
 Management client.</param>
    /// <param name="configuration">The injected configuration.</param>
```

```
    public SmParameterWrapper(IAmazonSimpleSystemsManagement
 amazonSimpleSystemsManagement, IConfiguration configuration)
    {
        _amazonSimpleSystemsManagement = amazonSimpleSystemsManagement;
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Reset the Systems Manager parameters to starting values for the demo.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task Reset()
    {
        await this.PutParameterByName(_tableParameter, _tableName);
        await this.PutParameterByName(_failureResponseParameter, "none");
        await this.PutParameterByName(_healthCheckParameter, "shallow");
    }

    /// <summary>
    /// Set the value of a named Systems Manager parameter.
    /// </summary>
    /// <param name="name">The name of the parameter.</param>
    /// <param name="value">The value to set.</param>
    /// <returns>Async task.</returns>
    public async Task PutParameterByName(string name, string value)
    {
        await _amazonSimpleSystemsManagement.PutParameterAsync(
            new PutParameterRequest() { Name = name, Value = value, Overwrite =
 true });
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for .NET API 參考》中的下列主題。
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer

- CreateTargetGroup
- DeleteAutoScalingGroup
- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

在命令提示中執行互動式案例。

```
public class Main {
```

```
    public static final String fileName = "C:\\AWS\\resworkflow\
\recommendations.json"; // Modify file location.
    public static final String tableName = "doc-example-recommendation-service";
    public static final String startScript = "C:\\AWS\\resworkflow\
\server_startup_script.sh"; // Modify file location.
    public static final String policyFile = "C:\\AWS\\resworkflow\
\instance_policy.json"; // Modify file location.
    public static final String ssmJSON = "C:\\AWS\\resworkflow\
\ssm_only_policy.json"; // Modify file location.
    public static final String failureResponse = "doc-example-resilient-
architecture-failure-response";
    public static final String healthCheck = "doc-example-resilient-architecture-
health-check";
    public static final String templateName = "doc-example-resilience-template";
    public static final String roleName = "doc-example-resilience-role";
    public static final String policyName = "doc-example-resilience-pol";
    public static final String profileName = "doc-example-resilience-prof";

    public static final String badCredsProfileName = "doc-example-resilience-
prof-bc";

    public static final String targetGroupName = "doc-example-resilience-tg";
    public static final String autoScalingGroupName = "doc-example-resilience-
group";
    public static final String lbName = "doc-example-resilience-lb";
    public static final String protocol = "HTTP";
    public static final int port = 80;

    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws IOException,
 InterruptedException {
        Scanner in = new Scanner(System.in);
        Database database = new Database();
        AutoScaler autoScaler = new AutoScaler();
        LoadBalancer loadBalancer = new LoadBalancer();

        System.out.println(DASHES);
        System.out.println("Welcome to the demonstration of How to Build and
 Manage a Resilient Service!");
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("A - SETUP THE RESOURCES");
        System.out.println("Press Enter when you're ready to start deploying
resources.");
        in.nextLine();
        deploy(loadBalancer);
        System.out.println(DASHES);
        System.out.println(DASHES);
        System.out.println("B - DEMO THE RESILIENCE FUNCTIONALITY");
        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        demo(loadBalancer);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("C - DELETE THE RESOURCES");
        System.out.println("""
                This concludes the demo of how to build and manage a resilient
service.
                To keep things tidy and to avoid unwanted charges on your
account, we can clean up all AWS resources
                that were created for this demo.
                """);

        System.out.println("\n Do you want to delete the resources (y/n)? ");
        String userInput = in.nextLine().trim().toLowerCase(); // Capture user
input

        if (userInput.equals("y")) {
            // Delete resources here
            deleteResources(loadBalancer, autoScaler, database);
            System.out.println("Resources deleted.");
        } else {
            System.out.println("""
                    Okay, we'll leave the resources intact.
                    Don't forget to delete them when you're done with them or you
might incur unexpected charges.
                    """);
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The example has completed. ");
        System.out.println("\n Thanks for watching!");
        System.out.println(DASHES);
```

```
    }

    // Deletes the AWS resources used in this example.
    private static void deleteResources(LoadBalancer loadBalancer, AutoScaler
autoScaler, Database database)
            throws IOException, InterruptedException {
        loadBalancer.deleteLoadBalancer(lbName);
        System.out.println("*** Wait 30 secs for resource to be deleted");
        TimeUnit.SECONDS.sleep(30);
        loadBalancer.deleteTargetGroup(targetGroupName);
        autoScaler.deleteAutoScaleGroup(autoScalingGroupName);
        autoScaler.deleteRolesPolicies(policyName, roleName, profileName);
        autoScaler.deleteTemplate(templateName);
        database.deleteTable(tableName);
    }

    private static void deploy(LoadBalancer loadBalancer) throws
InterruptedException, IOException {
        Scanner in = new Scanner(System.in);
        System.out.println(
                """
                        For this demo, we'll use the AWS SDK for Java (v2) to
create several AWS resources
                        to set up a load-balanced web service endpoint and
explore some ways to make it resilient
                        against various kinds of failures.

                        Some of the resources create by this demo are:
                        \t* A DynamoDB table that the web service depends on to
provide book, movie, and song recommendations.
                        \t* An EC2 launch template that defines EC2 instances
that each contain a Python web server.
                        \t* An EC2 Auto Scaling group that manages EC2 instances
across several Availability Zones.
                        \t* An Elastic Load Balancing (ELB) load balancer that
targets the Auto Scaling group to distribute requests.
                        """);

        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("Creating and populating a DynamoDB table named " +
tableName);
        Database database = new Database();
        database.createTable(tableName, fileName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.
                This script starts a Python web server defined in the `server.py`
script. The web server
                listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.
                For demo purposes, this server is run as the root user. In
production, the best practice is to
                run a web server, such as Apache, with least-privileged
credentials.

                The template also defines an IAM policy that each instance uses
to assume a role that grants
                permissions to access the DynamoDB recommendation table and
Systems Manager parameters
                that control the flow of the demo.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        templateCreator.createTemplate(policyFile, policyName, profileName,
startScript, templateName, roleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different Availability Zone.");
        System.out.println("*** Wait 30 secs for the VPC to be created");
        TimeUnit.SECONDS.sleep(30);
        AutoScaler autoScaler = new AutoScaler();
        String[] zones = autoScaler.createGroup(3, templateName,
autoScalingGroupName);

        System.out.println("""
                At this point, you have EC2 instances created. Once each instance
starts, it listens for
```

```
                HTTP requests. You can see these instances in the console or
continue with the demo.
                Press Enter when you're ready to continue.
                """);

        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating variables that control the flow of the
demo.");
        ParameterHelper paramHelper = new ParameterHelper();
        paramHelper.reset();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an Elastic Load Balancing target group and load
balancer. The target group
                defines how the load balancer connects to instances. The load
balancer provides a
                single endpoint where clients connect and dispatches requests to
instances in the group.
                """);

        String vpcId = autoScaler.getDefaultVPC();
        List<Subnet> subnets = autoScaler.getSubnets(vpcId, zones);
        System.out.println("You have retrieved a list with " + subnets.size() + "
subnets");
        String targetGroupArn = loadBalancer.createTargetGroup(protocol, port,
vpcId, targetGroupName);
        String elbDnsName = loadBalancer.createLoadBalancer(subnets,
targetGroupArn, lbName, port, protocol);
        autoScaler.attachLoadBalancerTargetGroup(autoScalingGroupName,
targetGroupArn);
        System.out.println("Verifying access to the load balancer endpoint...");
        boolean wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
```

```
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);

                // Read the response content.
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }

            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
```

```
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
```

```
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                       """);
       paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

       System.out.println(
               """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                       """);
       demoChoices(loadBalancer);

       System.out.println(
               """
                       Instead of failing when the recommendation service fails,
the web service can return a static response.
                       While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                       """);
       paramHelper.put(paramHelper.failureResponse, "static");

       System.out.println("""
               Now, sending a GET request to the load balancer endpoint returns
a static response.
               The service still reports as healthy because health checks are
still shallow.
               """);
       demoChoices(loadBalancer);

       System.out.println("Let's reinstate the recommendation service.");
       paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

       System.out.println("""
               Let's also substitute bad credentials for one of the instances in
the target group so that it can't
               access the DynamoDB recommendation table. We will get an instance
id value.
               """);
```

```
        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();

        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);

        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
                """);

        System.out.println("""
```

```
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);

        paramHelper.put(paramHelper.healthCheck, "deep");

        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
                """);

        demoChoices(loadBalancer);
```

```
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        demoChoices(loadBalancer);
        paramHelper.reset();
    }

    public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        String[] actions = {
                "Send a GET request to the load balancer endpoint.",
                "Check the health of load balancer targets.",
                "Go to the next part of the demo."
        };
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("-".repeat(88));
            System.out.println("See the current state of the service by selecting
one of the following choices:");
            for (int i = 0; i < actions.length; i++) {
                System.out.println(i + ": " + actions[i]);
            }

            try {
                System.out.print("\nWhich action would you like to take? ");
                int choice = scanner.nextInt();
                System.out.println("-".repeat(88));

                switch (choice) {
                    case 0 -> {
                        System.out.println("Request:\n");
                        System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                        CloseableHttpClient httpClient =
HttpClients.createDefault();

                        // Create an HTTP GET request to the ELB.
                        HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                        // Execute the request and get the response.
```

```
                        HttpResponse response = httpClient.execute(httpGet);
                        int statusCode =
response.getStatusLine().getStatusCode();
                        System.out.println("HTTP Status Code: " + statusCode);

                        // Display the JSON response
                        BufferedReader reader = new BufferedReader(
                                new
InputStreamReader(response.getEntity().getContent()));
                        StringBuilder jsonResponse = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null) {
                            jsonResponse.append(line);
                        }
                        reader.close();

                        // Print the formatted JSON response.
                        System.out.println("Full Response:\n");
                        System.out.println(jsonResponse.toString());

                        // Close the HTTP client.
                        httpClient.close();

                    }
                    case 1 -> {
                        System.out.println("\nChecking the health of load
balancer targets:\n");
                        List<TargetHealthDescription> health =
loadBalancer.checkTargetHealth(targetGroupName);
                        for (TargetHealthDescription target : health) {
                            System.out.printf("\tTarget %s on port %d is %s%n",
target.target().id(),
                                    target.target().port(),
target.targetHealth().stateAsString());
                        }
                        System.out.println("""
                                Note that it can take a minute or two for the
health check to update
                                after changes are made.
                                """);
                    }
                    case 2 -> {
                        System.out.println("\nOkay, let's move on.");
                        System.out.println("-".repeat(88));
```

```
                        return; // Exit the method when choice is 2
                    }
                    default -> System.out.println("You must choose a value
 between 0-2. Please select again.");
                }

            } catch (java.util.InputMismatchException e) {
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

建立包裝 Auto Scaling 和 Amazon EC2 動作的類別。

```
public class AutoScaler {

    private static Ec2Client ec2Client;
    private static AutoScalingClient autoScalingClient;
    private static IamClient iamClient;

    private static SsmClient ssmClient;

    private IamClient getIAMClient() {
        if (iamClient == null) {
            iamClient = IamClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return iamClient;
    }

    private SsmClient getSSMClient() {
        if (ssmClient == null) {
            ssmClient = SsmClient.builder()
                    .region(Region.US_EAST_1)
```

```
                    .build();
        }
        return ssmClient;
    }

    private Ec2Client getEc2Client() {
        if (ec2Client == null) {
            ec2Client = Ec2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ec2Client;
    }

    private AutoScalingClient getAutoScalingClient() {
        if (autoScalingClient == null) {
            autoScalingClient = AutoScalingClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return autoScalingClient;
    }

    /**
     * Terminates and instances in an EC2 Auto Scaling group. After an instance
is
     * terminated, it can no longer be accessed.
     */
    public void terminateInstance(String instanceId) {
        TerminateInstanceInAutoScalingGroupRequest terminateInstanceIRequest =
TerminateInstanceInAutoScalingGroupRequest
                .builder()
                .instanceId(instanceId)
                .shouldDecrementDesiredCapacity(false)
                .build();


getAutoScalingClient().terminateInstanceInAutoScalingGroup(terminateInstanceIRequest);
        System.out.format("Terminated instance %s.", instanceId);
    }

    /**
     * Replaces the profile associated with a running instance. After the profile
is
```

```
     * replaced, the instance is rebooted to ensure that it uses the new profile.
     * When
     * the instance is ready, Systems Manager is used to restart the Python web
     * server.
     */
    public void replaceInstanceProfile(String instanceId, String
newInstanceProfileName, String profileAssociationId)
            throws InterruptedException {
        // Create an IAM instance profile specification.
        software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
iamInstanceProfile =
software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
                .builder()
                .name(newInstanceProfileName) // Make sure
'newInstanceProfileName' is a valid IAM Instance Profile
                                              // name.
                .build();

        // Replace the IAM instance profile association for the EC2 instance.
        ReplaceIamInstanceProfileAssociationRequest replaceRequest =
ReplaceIamInstanceProfileAssociationRequest
                .builder()
                .iamInstanceProfile(iamInstanceProfile)
                .associationId(profileAssociationId) // Make sure
'profileAssociationId' is a valid association ID.
                .build();

        try {
            getEc2Client().replaceIamInstanceProfileAssociation(replaceRequest);
            // Handle the response as needed.
        } catch (Ec2Exception e) {
            // Handle exceptions, log, or report the error.
            System.err.println("Error: " + e.getMessage());
        }
        System.out.format("Replaced instance profile for association %s with
profile %s.", profileAssociationId,
                newInstanceProfileName);
        TimeUnit.SECONDS.sleep(15);
        boolean instReady = false;
        int tries = 0;

        // Reboot after 60 seconds
        while (!instReady) {
            if (tries % 6 == 0) {
```

```
                getEc2Client().rebootInstances(RebootInstancesRequest.builder()
                        .instanceIds(instanceId)
                        .build());
                System.out.println("Rebooting instance " + instanceId + " and
waiting for it to be ready.");
            }
            tries++;
            try {
                TimeUnit.SECONDS.sleep(10);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            DescribeInstanceInformationResponse informationResponse =
getSSMClient().describeInstanceInformation();
            List<InstanceInformation> instanceInformationList =
informationResponse.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
```

```
                .fromPort(Integer.parseInt(port))
                .build();

        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }

    /**
     * Detaches a role from an instance profile, detaches policies from the role,
     * and deletes all the resources.
     */
    public void deleteInstanceProfile(String roleName, String profileName) {
        try {
            software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                    .builder()
                    .instanceProfileName(profileName)
                    .build();

            GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
            String name = response.instanceProfile().instanceProfileName();
            System.out.println(name);

            RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .roleName(roleName)
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(profileRequest);
            DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .build();

            getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
            System.out.println("Deleted instance profile " + profileName);

            DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            // List attached role policies.
            ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
                    .listAttachedRolePolicies(role -> role.roleName(roleName));
            List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
            for (AttachedPolicy attachedPolicy : attachedPolicies) {
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");

        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();


getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
        System.out.println(groupName + " was deleted.");
    }
```

```
   /*
    * Verify the default security group of the specified VPC allows ingress from
    * this
    * computer. This can be done by allowing ingress from this computer's IP
    * address. In some situations, such as connecting from a corporate network,
you
    * must instead specify a prefix list ID. You can also temporarily open the
port
    * to
    * any IP address while running this example. If you do, be sure to remove
    * public
    * access when you're done.
    *
    */
   public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
       boolean portIsOpen = false;
       GroupInfo groupInfo = new GroupInfo();
       try {
           Filter filter = Filter.builder()
                   .name("group-name")
                   .values("default")
                   .build();

           Filter filter1 = Filter.builder()
                   .name("vpc-id")
                   .values(VPC)
                   .build();

           DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                   .filters(filter, filter1)
                   .build();

           DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                   .describeSecurityGroups(securityGroupsRequest);
           String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
           groupInfo.setGroupName(securityGroup);

           for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
```

```
                System.out.println("Found security group: " +
secGroup.groupId());

                for (IpPermission ipPermission : secGroup.ipPermissions()) {
                    if (ipPermission.fromPort() == port) {
                        System.out.println("Found inbound rule: " +
ipPermission);
                        for (IpRange ipRange : ipPermission.ipRanges()) {
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
                        }
                    }
                }
            }

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }

    /*
     * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
```

```
    * Scaling group.
    * The target group specifies how the load balancer forward requests to the
    * instances
    * in the group.
    */
   public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
       try {
           AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                   .autoScalingGroupName(asGroupName)
                   .targetGroupARNs(targetGroupARN)
                   .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
           System.out.println("Attached load balancer to " + asGroupName);

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

   // Creates an EC2 Auto Scaling group with the specified size.
   public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

       // Get availability zones.

software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
               .builder()
               .build();

       DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
       List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()

.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
               .collect(Collectors.toList());
```

```
        String availabilityZones = String.join(",", availabilityZoneNames);
        LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
                .launchTemplateName(templateName)
                .version("$Default")
                .build();

        String[] zones = availabilityZones.split(",");
        CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
                .launchTemplate(specification)
                .availabilityZones(zones)
                .maxSize(groupSize)
                .minSize(groupSize)
                .autoScalingGroupName(autoScalingGroupName)
                .build();

        try {
            getAutoScalingClient().createAutoScalingGroup(groupRequest);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
        return zones;
    }

    public String getDefaultVPC() {
        // Define the filter.
        Filter defaultFilter = Filter.builder()
                .name("is-default")
                .values("true")
                .build();

        software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
                .builder()
                .filters(defaultFilter)
                .build();

        DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
        return response.vpcs().get(0).vpcId();
```

```
    }

    // Gets the default subnets in a VPC for a specified list of Availability
Zones.
    public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
        List<Subnet> subnets = null;
        Filter vpcFilter = Filter.builder()
                .name("vpc-id")
                .values(vpcId)
                .build();

        Filter azFilter = Filter.builder()
                .name("availability-zone")
                .values(availabilityZones)
                .build();

        Filter defaultForAZ = Filter.builder()
                .name("default-for-az")
                .values("true")
                .build();

        DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
                .filters(vpcFilter, azFilter, defaultForAZ)
                .build();

        DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
        subnets = response.subnets();
        return subnets;
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());
```

```
        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
            return instanceId;
        }
        return "";
    }

    // Gets data about the profile associated with an instance.
    public String getInstanceProfile(String instanceId) {
        Filter filter = Filter.builder()
                .name("instance-id")
                .values(instanceId)
                .build();

        DescribeIamInstanceProfileAssociationsRequest associationsRequest =
DescribeIamInstanceProfileAssociationsRequest
                .builder()
                .filters(filter)
                .build();

        DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
                .describeIamInstanceProfileAssociations(associationsRequest);
        return response.iamInstanceProfileAssociations().get(0).associationId();
    }

    public void deleteRolesPolicies(String policyName, String roleName, String
InstanceProfile) {
        ListPoliciesRequest listPoliciesRequest =
ListPoliciesRequest.builder().build();
        ListPoliciesResponse listPoliciesResponse =
getIAMClient().listPolicies(listPoliciesRequest);
        for (Policy policy : listPoliciesResponse.policies()) {
            if (policy.policyName().equals(policyName)) {
                // List the entities (users, groups, roles) that are attached to
the policy.

software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
listEntitiesRequest =
software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                        .builder()
                        .policyArn(policy.arn())
                        .build();
```

```
                ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                        .listEntitiesForPolicy(listEntitiesRequest);
                if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
                        || !listEntitiesResponse.policyRoles().isEmpty()) {
                    // Detach the policy from any entities it is attached to.
                    DetachRolePolicyRequest detachPolicyRequest =
 DetachRolePolicyRequest.builder()
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
 role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
 DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();

                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }

        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
 ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
 iamClient.listInstanceProfilesForRole(listRolesRequest);
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
 listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
 RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
```

```
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
            System.out.println("Role " + roleName + " removed from instance
 profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

建立包裝 Elastic Load Balancing 動作的類別。

```
public class LoadBalancer {
    public ElasticLoadBalancingV2Client elasticLoadBalancingV2Client;

    public ElasticLoadBalancingV2Client getLoadBalancerClient() {
        if (elasticLoadBalancingV2Client == null) {
            elasticLoadBalancingV2Client = ElasticLoadBalancingV2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }

        return elasticLoadBalancingV2Client;
    }

    // Checks the health of the instances in the target group.
    public List<TargetHealthDescription> checkTargetHealth(String
 targetGroupName) {
        DescribeTargetGroupsRequest targetGroupsRequest =
 DescribeTargetGroupsRequest.builder()
                .names(targetGroupName)
                .build();
```

```
        DescribeTargetGroupsResponse tgResponse =
getLoadBalancerClient().describeTargetGroups(targetGroupsRequest);

        DescribeTargetHealthRequest healthRequest =
DescribeTargetHealthRequest.builder()

.targetGroupArn(tgResponse.targetGroups().get(0).targetGroupArn())
                .build();

        DescribeTargetHealthResponse healthResponse =
getLoadBalancerClient().describeTargetHealth(healthRequest);
        return healthResponse.targetHealthDescriptions();
    }

    // Gets the HTTP endpoint of the load balancer.
    public String getEndpoint(String lbName) {
        DescribeLoadBalancersResponse res = getLoadBalancerClient()
                .describeLoadBalancers(describe -> describe.names(lbName));
        return res.loadBalancers().get(0).dnsName();
    }

    // Deletes a load balancer.
    public void deleteLoadBalancer(String lbName) {
        try {
            // Use a waiter to delete the Load Balancer.
            DescribeLoadBalancersResponse res = getLoadBalancerClient()
                    .describeLoadBalancers(describe -> describe.names(lbName));
            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()

.loadBalancerArns(res.loadBalancers().get(0).loadBalancerArn())
                    .build();

            getLoadBalancerClient().deleteLoadBalancer(
                    builder ->
builder.loadBalancerArn(res.loadBalancers().get(0).loadBalancerArn()));
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancersDeleted(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
```

```
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(lbName + " was deleted.");
    }

    // Deletes the target group.
    public void deleteTargetGroup(String targetGroupName) {
        try {
            DescribeTargetGroupsResponse res = getLoadBalancerClient()
                    .describeTargetGroups(describe ->
describe.names(targetGroupName));
            getLoadBalancerClient()
                    .deleteTargetGroup(builder ->
builder.targetGroupArn(res.targetGroups().get(0).targetGroupArn()));
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(targetGroupName + " was deleted.");
    }

    // Verify this computer can successfully send a GET request to the load
balancer
    // endpoint.
    public boolean verifyLoadBalancerEndpoint(String elbDnsName) throws
IOException, InterruptedException {
        boolean success = false;
        int retries = 3;
        CloseableHttpClient httpClient = HttpClients.createDefault();

        // Create an HTTP GET request to the ELB.
        HttpGet httpGet = new HttpGet("http://" + elbDnsName);
        try {
            while ((!success) && (retries > 0)) {
                // Execute the request and get the response.
                HttpResponse response = httpClient.execute(httpGet);
                int statusCode = response.getStatusLine().getStatusCode();
                System.out.println("HTTP Status Code: " + statusCode);
                if (statusCode == 200) {
                    success = true;
                } else {
                    retries--;
                    System.out.println("Got connection error from load balancer
endpoint, retrying...");
```

```
                    TimeUnit.SECONDS.sleep(15);
                }
            }

        } catch (org.apache.http.conn.HttpHostConnectException e) {
            System.out.println(e.getMessage());
        }

        System.out.println("Status.." + success);
        return success;
    }

    /*
     * Creates an Elastic Load Balancing target group. The target group specifies
     * how
     * the load balancer forward requests to instances in the group and how
instance
     * health is checked.
     */
    public String createTargetGroup(String protocol, int port, String vpcId,
String targetGroupName) {
        CreateTargetGroupRequest targetGroupRequest =
CreateTargetGroupRequest.builder()
                .healthCheckPath("/healthcheck")
                .healthCheckTimeoutSeconds(5)
                .port(port)
                .vpcId(vpcId)
                .name(targetGroupName)
                .protocol(protocol)
                .build();

        CreateTargetGroupResponse targetGroupResponse =
getLoadBalancerClient().createTargetGroup(targetGroupRequest);
        String targetGroupArn =
targetGroupResponse.targetGroups().get(0).targetGroupArn();
        String targetGroup =
targetGroupResponse.targetGroups().get(0).targetGroupName();
        System.out.println("The " + targetGroup + " was created with ARN" +
targetGroupArn);
        return targetGroupArn;
    }

    /*
     * Creates an Elastic Load Balancing load balancer that uses the specified
```

```
     * subnets
     * and forwards requests to the specified target group.
     */
    public String createLoadBalancer(List<Subnet> subnetIds, String
targetGroupARN, String lbName, int port,
            String protocol) {
        try {
            List<String> subnetIdStrings = subnetIds.stream()
                    .map(Subnet::subnetId)
                    .collect(Collectors.toList());

            CreateLoadBalancerRequest balancerRequest =
CreateLoadBalancerRequest.builder()
                    .subnets(subnetIdStrings)
                    .name(lbName)
                    .scheme("internet-facing")
                    .build();

            // Create and wait for the load balancer to become available.
            CreateLoadBalancerResponse lsResponse =
getLoadBalancerClient().createLoadBalancer(balancerRequest);
            String lbARN = lsResponse.loadBalancers().get(0).loadBalancerArn();

            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()
                    .loadBalancerArns(lbARN)
                    .build();

            System.out.println("Waiting for Load Balancer " + lbName + " to
become available.");
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancerAvailable(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Load Balancer " + lbName + " is available.");

            // Get the DNS name (endpoint) of the load balancer.
            String lbDNSName = lsResponse.loadBalancers().get(0).dnsName();
            System.out.println("*** Load Balancer DNS Name: " + lbDNSName);

            // Create a listener for the load balance.
            Action action = Action.builder()
```

```
                    .targetGroupArn(targetGroupARN)
                    .type("forward")
                    .build();

            CreateListenerRequest listenerRequest =
 CreateListenerRequest.builder()

 .loadBalancerArn(lsResponse.loadBalancers().get(0).loadBalancerArn())
                    .defaultActions(action)
                    .port(port)
                    .protocol(protocol)
                    .defaultActions(action)
                    .build();

            getLoadBalancerClient().createListener(listenerRequest);
            System.out.println("Created listener to forward traffic from load
 balancer " + lbName + " to target group "
                    + targetGroupARN);

            // Return the load balancer DNS name.
            return lbDNSName;

        } catch (ElasticLoadBalancingV2Exception e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

建立使用 DynamoDB 模擬建議服務的類別。

```
public class Database {

    private static DynamoDbClient dynamoDbClient;

    public static DynamoDbClient getDynamoDbClient() {
        if (dynamoDbClient == null) {
            dynamoDbClient = DynamoDbClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return dynamoDbClient;
```

```
    }

    // Checks to see if the Amazon DynamoDB table exists.
    private boolean doesTableExist(String tableName) {
        try {
            // Describe the table and catch any exceptions.
            DescribeTableRequest describeTableRequest =
DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            getDynamoDbClient().describeTable(describeTableRequest);
            System.out.println("Table '" + tableName + "' exists.");
            return true;

        } catch (ResourceNotFoundException e) {
            System.out.println("Table '" + tableName + "' does not exist.");
        } catch (DynamoDbException e) {
            System.err.println("Error checking table existence: " +
e.getMessage());
        }
        return false;
    }

    /*
     * Creates a DynamoDB table to use a recommendation service. The table has a
     * hash key named 'MediaType' that defines the type of media recommended,
such
     * as
     * Book or Movie, and a range key named 'ItemId' that, combined with the
     * MediaType,
     * forms a unique identifier for the recommended item.
     */
    public void createTable(String tableName, String fileName) throws IOException
{
        // First check to see if the table exists.
        boolean doesExist = doesTableExist(tableName);
        if (!doesExist) {
            DynamoDbWaiter dbWaiter = getDynamoDbClient().waiter();
            CreateTableRequest createTableRequest = CreateTableRequest.builder()
                    .tableName(tableName)
                    .attributeDefinitions(
                            AttributeDefinition.builder()
                                    .attributeName("MediaType")
```

```
                                .attributeType(ScalarAttributeType.S)
                                .build(),
                        AttributeDefinition.builder()
                                .attributeName("ItemId")
                                .attributeType(ScalarAttributeType.N)
                                .build())
                .keySchema(
                        KeySchemaElement.builder()
                                .attributeName("MediaType")
                                .keyType(KeyType.HASH)
                                .build(),
                        KeySchemaElement.builder()
                                .attributeName("ItemId")
                                .keyType(KeyType.RANGE)
                                .build())
                .provisionedThroughput(
                        ProvisionedThroughput.builder()
                                .readCapacityUnits(5L)
                                .writeCapacityUnits(5L)
                                .build())
                .build();

        getDynamoDbClient().createTable(createTableRequest);
        System.out.println("Creating table " + tableName + "...");

        // Wait until the Amazon DynamoDB table is created.
        DescribeTableRequest tableRequest = DescribeTableRequest.builder()
                .tableName(tableName)
                .build();

        WaiterResponse<DescribeTableResponse> waiterResponse =
dbWaiter.waitUntilTableExists(tableRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Table " + tableName + " created.");

        // Add records to the table.
        populateTable(fileName, tableName);
    }
}

public void deleteTable(String tableName) {
    getDynamoDbClient().deleteTable(table -> table.tableName(tableName));
    System.out.println("Table " + tableName + " deleted.");
}
```

```
    // Populates the table with data located in a JSON file using the DynamoDB
    // enhanced client.
    public void populateTable(String fileName, String tableName) throws
 IOException {
        DynamoDbEnhancedClient enhancedClient = DynamoDbEnhancedClient.builder()
                .dynamoDbClient(getDynamoDbClient())
                .build();
        ObjectMapper objectMapper = new ObjectMapper();
        File jsonFile = new File(fileName);
        JsonNode rootNode = objectMapper.readTree(jsonFile);

        DynamoDbTable<Recommendation> mappedTable =
 enhancedClient.table(tableName,
                TableSchema.fromBean(Recommendation.class));
        for (JsonNode currentNode : rootNode) {
            String mediaType = currentNode.path("MediaType").path("S").asText();
            int itemId = currentNode.path("ItemId").path("N").asInt();
            String title = currentNode.path("Title").path("S").asText();
            String creator = currentNode.path("Creator").path("S").asText();

            // Create a Recommendation object and set its properties.
            Recommendation rec = new Recommendation();
            rec.setMediaType(mediaType);
            rec.setItemId(itemId);
            rec.setTitle(title);
            rec.setCreator(creator);

            // Put the item into the DynamoDB table.
            mappedTable.putItem(rec); // Add the Recommendation to the list.
        }
        System.out.println("Added all records to the " + tableName);
    }
}
```

建立包裝 Systems Manager 動作的類別。

```
public class ParameterHelper {

    String tableName = "doc-example-resilient-architecture-table";
    String dyntable = "doc-example-recommendation-service";
```

```
    String failureResponse = "doc-example-resilient-architecture-failure-
response";
    String healthCheck = "doc-example-resilient-architecture-health-check";

    public void reset() {
        put(dyntable, tableName);
        put(failureResponse, "none");
        put(healthCheck, "shallow");
    }

    public void put(String name, String value) {
        SsmClient ssmClient = SsmClient.builder()
                .region(Region.US_EAST_1)
                .build();

        PutParameterRequest parameterRequest = PutParameterRequest.builder()
                .name(name)
                .value(value)
                .overwrite(true)
                .type("String")
                .build();

        ssmClient.putParameter(parameterRequest);
        System.out.printf("Setting demo parameter %s to '%s'.", name, value);
    }
}
```

- 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。
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate

- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

在命令提示中執行互動式案例。

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0


import {
  Scenario,
```

```
  parseScenarioArgs,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";

/**
 * The workflow steps are split into three stages:
 *   - deploy
 *   - demo
 *   - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios);
}
```

建立步驟以部署所有資源。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
```

```
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {
```

```
            AttributeName: "ItemId",
            AttributeType: "N",
          },
        ],
        KeySchema: [
          {
            AttributeName: "MediaType",
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
      new BatchWriteItemCommand({
        RequestItems: {
          [NAMES.tableName]: recommendations.map((item) => ({
            PutRequest: { Item: item },
          })),
        },
      }),
```

```
    );
  }),
  new ScenarioOutput(
    "populatedTable",
    MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "creatingKeyPair",
    MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioAction("createKeyPair", async () => {
    const client = new EC2Client({});
    const { KeyMaterial } = await client.send(
      new CreateKeyPairCommand({
        KeyName: NAMES.keyPairName,
      }),
    );

    writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
  }),
  new ScenarioOutput(
    "createdKeyPair",
    MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioOutput(
    "creatingInstancePolicy",
    MESSAGES.creatingInstancePolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    ),
  ),
  new ScenarioAction("createInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const {
      Policy: { Arn },
    } = await client.send(
      new CreatePolicyCommand({
        PolicyName: NAMES.instancePolicyName,
        PolicyDocument: readFileSync(
          join(RESOURCES_PATH, "instance_policy.json"),
        ),
      }),
    );
    state.instancePolicyArn = Arn;
```

```
  }),
  new ScenarioOutput("createdInstancePolicy", (state) =>
    MESSAGES.createdInstancePolicy
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
  ),
  new ScenarioOutput(
    "creatingInstanceRole",
    MESSAGES.creatingInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
        ),
      }),
    );
  }),
  new ScenarioOutput(
    "createdInstanceRole",
    MESSAGES.createdInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioOutput(
    "attachingPolicyToRole",
    MESSAGES.attachingPolicyToRole
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
  ),
  new ScenarioAction("attachPolicyToRole", async (state) => {
    const client = new IAMClient({});
    await client.send(
      new AttachRolePolicyCommand({
        RoleName: NAMES.instanceRoleName,
        PolicyArn: state.instancePolicyArn,
      }),
```

```
    );
  }),
  new ScenarioOutput(
    "attachedPolicyToRole",
    MESSAGES.attachedPolicyToRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioOutput(
    "creatingInstanceProfile",
    MESSAGES.creatingInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    ),
  ),
  new ScenarioAction("createInstanceProfile", async (state) => {
    const client = new IAMClient({});
    const {
      InstanceProfile: { Arn },
    } = await client.send(
      new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
    state.instanceProfileArn = Arn;

    await waitUntilInstanceProfileExists(
      { client },
      { InstanceProfileName: NAMES.instanceProfileName },
    );
  }),
  new ScenarioOutput("createdInstanceProfile", (state) =>
    MESSAGES.createdInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
  ),
  new ScenarioOutput(
    "addingRoleToInstanceProfile",
    MESSAGES.addingRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
```

```
    return client.send(
      new AddRoleToInstanceProfileCommand({
        RoleName: NAMES.instanceRoleName,
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
  }),
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
    );
    const ec2Client = new EC2Client({});
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
      // snippet-end:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
```

```
      NAMES.launchTemplateName,
    ),
  ),
  new ScenarioOutput(
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    ),
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
            Version: "$Default",
          },
          MinSize: 3,
          MaxSize: 3,
        }),
      ),
    );
  }),
  new ScenarioOutput(
    "createdAutoScalingGroup",
    /**
     * @param {{ availabilityZoneNames: string[] }} state
     */
    (state) =>
      MESSAGES.createdAutoScalingGroup
        .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
        .replace(
          "${AVAILABILITY_ZONE_NAMES}",
          state.availabilityZoneNames.join(", "),
        ),
```

```
  ),
  new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
    type: "confirm",
  }),
  new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
  new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
  new ScenarioAction("getVpc", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DescribeVpcs]
    const client = new EC2Client({});
    const { Vpcs } = await client.send(
      new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DescribeVpcs]
    state.defaultVpc = Vpcs[0].VpcId;
  }),
  new ScenarioOutput("gotVpc", (state) =>
    MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
  ),
  new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
  new ScenarioAction("getSubnets", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DescribeSubnets]
    const client = new EC2Client({});
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DescribeSubnets]
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
```

```
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateTargetGroup]
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateTargetGroup]
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
```

```
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
  new ScenarioAction("createListener", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateListener]
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateListener]
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
```

```
      state.loadBalancerListenerArn,
    ),
  ),
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.AttachTargetGroup]
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.AttachTargetGroup]
  }),
  new ScenarioOutput(
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];
```

```
      /**
       * @type {string}
       */
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
      const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
        ({ IpRanges }) =>
          IpRanges.some(
            ({ CidrIp }) =>
              CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
          ),
      )
        .filter(({ IpProtocol }) => IpProtocol === "tcp")
        .filter(({ FromPort }) => FromPort === 80);

      state.myIpRules = myIpRules;
    },
  ),
  new ScenarioOutput(
    "verifiedInboundPort",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return MESSAGES.foundIpRules.replace(
          "${IP_RULES}",
          JSON.stringify(state.myIpRules, null, 2),
        );
      } else {
        return MESSAGES.noIpRules;
      }
    },
  ),
  new ScenarioInput(
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      } else {
        return MESSAGES.noIpRules;
```

```
      }
    },
    { type: "confirm" },
  ),
  new ScenarioAction(
    "addInboundRule",
    /**
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }

      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
          ToPort: 80,
          IpProtocol: "tcp",
        }),
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    } else {
      return false;
    }
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
```

```
      state.verifyEndpointError = e;
    }
  }),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
];
```

建立步驟以執行示範。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { readFileSync } from "node:fs";
import { join } from "node:path";

import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
```

```
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
```

```
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetGroups]
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetGroups]

  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  state.targetHealthDescriptions = TargetHealthDescriptions;
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,
```

```
  {
    whileConfig: {
      whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
      output: getHealthCheckResult,
    },
  },
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
```

```
    "brokenDependencyConfirmation",
    MESSAGES.demoBrokenDependencyConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("brokenDependency", async (state) => {
    if (!state.brokenDependencyConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      state.badTableName = `fake-table-${Date.now()}`;
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmTableNameKey,
          Value: state.badTableName,
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testBrokenDependency", (state) =>
    MESSAGES.demoTestBrokenDependency.replace(
      "${TABLE_NAME}",
      state.badTableName,
    ),
  ),
  ...statusSteps,
  new ScenarioInput(
    "staticResponseConfirmation",
    MESSAGES.demoStaticResponseConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("staticResponse", async (state) => {
    if (!state.staticResponseConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmFailureResponseKey,
          Value: "static",
          Overwrite: true,
          Type: "String",
        }),
```

```
      );
    }
  }),
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioAction(
    "badCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      // snippet-start:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      const ec2Client = new EC2Client({});
```

```
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      // snippet-start:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]

      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );

        const instance = InstanceInformationList.find(
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });
```

```
      await ssmClient.send(
        new SendCommandCommand({
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
  ),
  loadBalancerLoop,
  new ScenarioInput(
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
```

```
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "killInstanceConfirmation",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    (state) =>
      MESSAGES.demoKillInstanceConfirmation.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
    { type: "confirm" },
  ),
  new ScenarioAction("killInstanceExit", (state) => {
    if (!state.killInstanceConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction(
    "killInstance",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    async (state) => {
      const client = new AutoScalingClient({});
      await client.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: state.targetInstance.InstanceId,
          ShouldDecrementDesiredCapacity: false,
        }),
      );
    },
  ),
  new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
    type: "confirm",
  }),
  new ScenarioAction("failOpenExit", (state) => {
```

```
    if (!state.failOpenConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("failOpen", () => {
    const client = new SSMClient({});
    return client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: `fake-table-${Date.now()}`,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "resetTableConfirmation",
    MESSAGES.demoResetTableConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("resetTableExit", (state) => {
    if (!state.resetTableConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("resetTable", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];
```

```
async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "ssm_only_policy.json"),
      ),
    }),
  );
  await iamClient.send(
    new CreateRoleCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      AssumeRolePolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Principal: { Service: "ec2.amazonaws.com" },
            Action: "sts:AssumeRole",
          },
        ],
      }),
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: Policy.Arn,
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
    }),
  );
  // snippet-start:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
```

```
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

建立步驟以銷毀所有資源。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
```

```
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    } else {
      return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
    }
```

```
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    } else {
      return MESSAGES.deletedKeyPair.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    }
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);

      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
      } else {
        await client.send(
          new DetachRolePolicyCommand({
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
          }),
        );
      }
    } catch (e) {
```

```
      state.detachPolicyFromRoleError = e;
    }
  }),
  new ScenarioOutput("detachedPolicyFromRole", (state) => {
    if (state.detachPolicyFromRoleError) {
      console.error(state.detachPolicyFromRoleError);
      return MESSAGES.detachPolicyFromRoleError
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.detachedPolicyFromRole
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
  }),
  new ScenarioAction("deleteInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const policy = await findPolicy(NAMES.instancePolicyName);

    if (!policy) {
      state.deletePolicyError = new Error(
        `Policy ${NAMES.instancePolicyName} not found.`,
      );
    } else {
      return client.send(
        new DeletePolicyCommand({
          PolicyArn: policy.Arn,
        }),
      );
    }
  }),
  new ScenarioOutput("deletePolicyResult", (state) => {
    if (state.deletePolicyError) {
      console.error(state.deletePolicyError);
      return MESSAGES.deletePolicyError.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    } else {
      return MESSAGES.deletedPolicy.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    }
```

```
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.removedRoleFromInstanceProfile
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
```

```
      );
    } else {
      return MESSAGES.deletedInstanceRole.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    } else {
      return MESSAGES.deletedInstanceProfile.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
```

```
    }
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    } else {
      return MESSAGES.deletedAutoScalingGroup.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    } else {
      return MESSAGES.deletedLaunchTemplate.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
  }),
```

```
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    } else {
      return MESSAGES.deletedLoadBalancer.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
    const client = new ElasticLoadBalancingV2Client({});
    try {
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );
```

```
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      client.send(
        new DeleteTargetGroupCommand({
          TargetGroupArn: TargetGroups[0].TargetGroupArn,
        }),
      ),
    );
  } catch (e) {
    state.deleteLoadBalancerTargetGroupError = e;
  }
  // snippet-end:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
}),
new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
  if (state.deleteLoadBalancerTargetGroupError) {
    console.error(state.deleteLoadBalancerTargetGroupError);
    return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  } else {
    return MESSAGES.deletedLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  }
}),
new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
  try {
    const client = new IAMClient({});
    await client.send(
      new RemoveRoleFromInstanceProfileCommand({
        InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        RoleName: NAMES.ssmOnlyRoleName,
      }),
    );
  } catch (e) {
    state.detachSsmOnlyRoleFromProfileError = e;
  }
}),
new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
  if (state.detachSsmOnlyRoleFromProfileError) {
    console.error(state.detachSsmOnlyRoleFromProfileError);
    return MESSAGES.detachSsmOnlyRoleFromProfileError
```

```
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    } else {
      return MESSAGES.detachedSsmOnlyRoleFromProfile
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyCustomRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
      console.error(state.detachSsmOnlyCustomRolePolicyError);
      return MESSAGES.detachSsmOnlyCustomRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    } else {
      return MESSAGES.detachedSsmOnlyCustomRolePolicy
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
  }),
  new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
        }),
      );
    } catch (e) {
```

```
      state.detachSsmOnlyAWSRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
    if (state.detachSsmOnlyAWSRolePolicyError) {
      console.error(state.detachSsmOnlyAWSRolePolicyError);
      return MESSAGES.detachSsmOnlyAWSRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    } else {
      return MESSAGES.detachedSsmOnlyAWSRolePolicy
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    }
  }),
  new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
```

```
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DeletePolicyCommand({
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyPolicy.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    }
  }),
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    } else {
```

```
      return MESSAGES.deletedSsmOnlyRole.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
}

/**
 * @param {string} groupName
 */
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
    await client.send(
      new DeleteAutoScalingGroupCommand({
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    } else {
      console.log(err.name);
      throw err;
    }
  }
}

/**
```

```
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for JavaScript API 參考》中的下列主題。
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile

- CreateLaunchTemplate
- CreateListener
- CreateLoadBalancer
- CreateTargetGroup
- DeleteAutoScalingGroup
- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

在命令提示中執行互動式案例。

```
class Runner:
    def __init__(
        self, resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    ):
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

    def deploy(self):
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        print(
            "\nFor this demo, we'll use the AWS SDK for Python (Boto3) to create
 several AWS resources\n"
            "to set up a load-balanced web service endpoint and explore some ways
 to make it resilient\n"
            "against various kinds of failures.\n\n"
            "Some of the resources create by this demo are:\n"
        )
        print(
            "\t* A DynamoDB table that the web service depends on to provide
 book, movie, and song recommendations."
        )
        print(
            "\t* An EC2 launch template that defines EC2 instances that each
 contain a Python web server."
        )
        print(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
 several Availability Zones."
        )
        print(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
 Auto Scaling group to distribute requests."
```

```
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to start deploying resources.")

        print(
            f"Creating and populating a DynamoDB table named
'{self.recommendation.table_name}'."
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)
        print("-" * 88)

        print(
            f"Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.\n"
            f"This script starts a Python web server defined in the `server.py`
script. The web server\n"
            f"listens to HTTP requests on port 80 and responds to requests to '/'
and to '/healthcheck'.\n"
            f"For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            f"run a web server, such as Apache, with least-privileged
credentials.\n"
        )
        print(
            f"The template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            f"permissions to access the DynamoDB recommendation table and Systems
Manager parameters\n"
            f"that control the flow of the demo.\n"
        )
        self.autoscaler.create_template(startup_script, instance_policy)
        print("-" * 88)

        print(
            f"Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            f"Availability Zone."
        )
        zones = self.autoscaler.create_group(3)
        print("-" * 88)
        print(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
```

```
            "HTTP requests. You can see these instances in the console or
continue with the demo."
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to continue.")

        print(f"Creating variables that control the flow of the demo.\n")
        self.param_helper.reset()

        print(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            "defines how the load balancer connects to instances. The load
balancer provides a\n"
            "single endpoint where clients connect and dispatches requests to
instances in the group.\n"
        )
        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.protocol, self.port, vpc["VpcId"]
        )
        self.loadbalancer.create_load_balancer(
            [subnet["SubnetId"] for subnet in subnets], target_group
        )
        self.autoscaler.attach_load_balancer_target_group(target_group)
        print(f"Verifying access to the load balancer endpoint...")
        lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if not lb_success:
            print(
                "Couldn't connect to the load balancer, verifying that the port
is open..."
            )
            current_ip_address = requests.get(
                "http://checkip.amazonaws.com"
            ).text.strip()
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
                print(
```

```
                    "For this example to work, the default security group for
 your default VPC must\n"
                    "allows access from this computer. You can either add it
 automatically from this\n"
                    "example or add it yourself using the AWS Management Console.
\n"
                )
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound traffic on port {self.port} from your computer's IP
 address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
 from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
            lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if lb_success:
            print("Your load balancer is ready. You can access it by browsing to:
\n")
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        else:
            print(
                "Couldn't get a successful response from the load balancer
 endpoint. Troubleshoot by\n"
                "manually verifying that your VPC and security group are
 configured correctly and that\n"
                "you can successfully make a GET request to the load balancer
 endpoint:\n"
            )
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        print("-" * 88)
```

```
        q.ask("Press Enter when you're ready to continue with the demo.")

    def demo_choices(self):
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
        ]
        choice = 0
        while choice != 2:
            print("-" * 88)
            print(
                "\nSee the current state of the service by selecting one of the
following choices:\n"
            )
            choice = q.choose("\nWhich action would you like to take? ", actions)
            print("-" * 88)
            if choice == 0:
                print("Request:\n")
                print(f"GET http://{self.loadbalancer.endpoint()}")
                response = requests.get(f"http://{self.loadbalancer.endpoint()}")
                print("\nResponse:\n")
                print(f"{response.status_code}")
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                print("\nChecking the health of load balancer targets:\n")
                health = self.loadbalancer.check_target_health()
                for target in health:
                    state = target["TargetHealth"]["State"]
                    print(
                        f"\tTarget {target['Target']['Id']} on port
{target['Target']['Port']} is {state}"
                    )
                    if state != "healthy":
                        print(
                            f"\t\t{target['TargetHealth']['Reason']}:
{target['TargetHealth']['Description']}\n"
                        )
                print(
                    f"\nNote that it can take a minute or two for the health
check to update\n"
                    f"after changes are made.\n"
                )
```

```
            elif choice == 2:
                print("\nOkay, let's move on.")
                print("-" * 88)

    def demo(self):
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        print("\nResetting parameters to starting values for demo.\n")
        self.param_helper.reset()

        print(
            "\nThis part of the demonstration shows how to toggle different parts
of the system\n"
            "to create situations where the web service fails, and shows how
using a resilient\n"
            "architecture can keep the web service running in spite of these
failures."
        )
        print("-" * 88)

        print(
            "At the start, the load balancer endpoint returns recommendations and
reports that all targets are healthy."
        )
        self.demo_choices()

        print(
            f"The web service running on the EC2 instances gets recommendations
by querying a DynamoDB table.\n"
            f"The table name is contained in a Systems Manager parameter named
'{self.param_helper.table}'.\n"
            f"To simulate a failure of the recommendation service, let's set this
parameter to name a non-existent table.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nNow, sending a GET request to the load balancer endpoint returns a
failure code. But, the service reports as\n"
            "healthy to the load balancer because shallow health checks don't
check for failure of the recommendation service."
        )
        self.demo_choices()

        print(
```

```
            f"Instead of failing when the recommendation service fails, the web
service can return a static response.\n"
            f"While this is not a perfect solution, it presents the customer with
a somewhat better experience than failure.\n"
        )
        self.param_helper.put(self.param_helper.failure_response, "static")
        print(
            f"\nNow, sending a GET request to the load balancer endpoint returns
a static response.\n"
            f"The service still reports as healthy because health checks are
still shallow.\n"
        )
        self.demo_choices()

        print("Let's reinstate the recommendation service.\n")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)
        print(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n"
            "access the DynamoDB recommendation table.\n"
        )
        self.autoscaler.create_instance_profile(
            ssm_only_policy,
            self.autoscaler.bad_creds_policy_name,
            self.autoscaler.bad_creds_role_name,
            self.autoscaler.bad_creds_profile_name,
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
        print(
            f"\nReplacing the profile for instance {bad_instance_id} with a
profile that contains\n"
            f"bad credentials...\n"
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        print(
```

```
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n"
            "depending on which instance is selected by the load balancer.\n"
        )
        self.demo_choices()

        print(
            "\nLet's implement a deep health check. For this demo, a deep health
check tests whether\n"
            "the web service can access the DynamoDB table that it depends on for
recommendations. Note that\n"
            "the deep health check is only for ELB routing and not for Auto
Scaling instance health.\n"
            "This kind of deep health check is not recommended for Auto Scaling
instance health, because it\n"
            "risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.\n"
        )
        print(
            "By implementing deep health checks, the load balancer can detect
when one of the instances is failing\n"
            "and take that instance out of rotation.\n"
        )
        self.param_helper.put(self.param_helper.health_check, "deep")
        print(
            f"\nNow, checking target health indicates that the instance with bad
credentials ({bad_instance_id})\n"
            f"is unhealthy. Note that it might take a minute or two for the load
balancer to detect the unhealthy \n"
            f"instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because\n"
            "the load balancer takes unhealthy instances out of its rotation.\n"
        )
        self.demo_choices()

        print(
            "\nBecause the instances in this demo are controlled by an auto
scaler, the simplest way to fix an unhealthy\n"
            "instance is to terminate it and let the auto scaler start a new
instance to replace it.\n"
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        print(
```

```
            "\nEven while the instance is terminating and the new instance is
starting, sending a GET\n"
            "request to the web service continues to get a successful
recommendation response because\n"
            "the load balancer routes requests to the healthy instances. After
the replacement instance\n"
            "starts and reports as healthy, it is included in the load balancing
rotation.\n"
            "\nNote that terminating and replacing an instance typically takes
several minutes, during which time you\n"
            "can see the changing health check status until the new instance is
running and healthy.\n"
        )
        self.demo_choices()

        print(
            "\nIf the recommendation service fails now, deep health checks mean
all instances report as unhealthy.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nWhen all instances are unhealthy, the load balancer continues to
route requests even to\n"
            "unhealthy instances, allowing them to fail open and return a static
response rather than fail\n"
            "closed and report failure to the customer."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self):
        print(
            "This concludes the demo of how to build and manage a resilient
service.\n"
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n"
            "that were created for this demo."
        )
        if q.ask("Do you want to clean up all demo resources? (y/n) ",
q.is_yesno):
            self.loadbalancer.delete_load_balancer()
            self.loadbalancer.delete_target_group()
            self.autoscaler.delete_group()
            self.autoscaler.delete_key_pair()
```

```
            self.autoscaler.delete_template()
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            self.recommendation.destroy()
        else:
            print(
                "Okay, 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."
            )


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
 are\n"
        "deployed, the demo is run, and resources are destroyed.",
    )
    parser.add_argument(
        "--resource_path",
        default="../../../workflows/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
 policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    print("-" * 88)
    print(
        "Welcome to the demonstration of How to Build and Manage a Resilient
 Service!"
    )
    print("-" * 88)

    prefix = "doc-example-resilience"
    recommendation = RecommendationService.from_client(
        "doc-example-recommendation-service"
    )
```

```
    autoscaler = AutoScaler.from_client(prefix)
    loadbalancer = LoadBalancer.from_client(prefix)
    param_helper = ParameterHelper.from_client(recommendation.table_name)
    runner = Runner(
        args.resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
 "destroy"]
    for action in actions:
        if action == "deploy":
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    print("-" * 88)
    print("Thanks for watching!")
    print("-" * 88)


if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

建立包裝 Auto Scaling 和 Amazon EC2 動作的類別。

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
```

```
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from Boto3 clients.

        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        """
        as_client = boto3.client("autoscaling")
        ec2_client = boto3.client("ec2")
        ssm_client = boto3.client("ssm")
        iam_client = boto3.client("iam")
        return cls(
            resource_prefix,
```

```
            "t3.micro",
            "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
            as_client,
            ec2_client,
            ssm_client,
            iam_client,
        )

    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
```

```
        }
        with open(policy_file) as file:
            instance_policy_doc = file.read()

        policy_arn = None
        try:
            pol_response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=instance_policy_doc
            )
            policy_arn = pol_response["Policy"]["Arn"]
            log.info("Created policy with ARN %s.", policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Policy %s already exists, nothing to do.", policy_name)
                list_pol_response = self.iam_client.list_policies(Scope="Local")
                for pol in list_pol_response["Policies"]:
                    if pol["PolicyName"] == policy_name:
                        policy_arn = pol["Arn"]
                        break
            if policy_arn is None:
                raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")

        try:
            self.iam_client.create_role(
                RoleName=role_name,
AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            self.iam_client.attach_role_policy(RoleName=role_name,
PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info("Created role %s and attached policy %s.", role_name,
policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Role %s already exists, nothing to do.", role_name)
            else:
                raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

        try:
```

```
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create profile {profile_name} and attach it to
role\n"
                    f"{role_name}: {err}"
                )
        return profile_arn


    def get_instance_profile(self, instance_id):
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
        except ClientError as err:
```

```
            raise AutoScalerError(
                f"Couldn't get instance profile association for instance
{instance_id}: {err}"
            )
        else:
            return response["IamInstanceProfileAssociations"][0]


    def replace_instance_profile(
        self, instance_id, new_instance_profile_name, profile_association_id
    ):
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to update.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)
            inst_ready = False
            tries = 0
            while not inst_ready:
                if tries % 6 == 0:
                    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                    log.info(
```

```
                        "Rebooting instance %s and waiting for it to to be
ready.",
                        instance_id,
                    )
                tries += 1
                time.sleep(10)
                response = self.ssm_client.describe_instance_information()
                for info in response["InstanceInformationList"]:
                    if info["InstanceId"] == instance_id:
                        inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )


    def delete_instance_profile(self, profile_name, role_name):
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
```

```
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete instance profile {profile_name} or detach "
                    f"policies and delete role {role_name}: {err}"
                )


    def create_key_pair(self, key_pair_name):
        """
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        :return: The newly created key pair.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't create key pair {key_pair_name}:
{err}")


    def delete_key_pair(self):
        """
        Deletes a key pair.

        :param key_pair_name: The name of the key pair to delete.
```

```
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )
        except FileNotFoundError:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
        except PermissionError:
            log.info(
                "Inadequate permissions to delete key pair %s.",
self.key_pair_name
            )
        except Exception as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )


    def create_template(self, server_startup_script_file, instance_policy_file):
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
```

```
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )
            with open(server_startup_script_file) as file:
                start_server_script = file.read()
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
            template = lt_response["LaunchTemplate"]
            log.info(
                "Created launch template %s for AMI %s on %s.",
                self.launch_template_name,
                ami_id,
                self.inst_type,
            )
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.AlreadyExistsException"
            ):
                log.info(
                    "Launch template %s already exists, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create launch template
{self.launch_template_name}: {err}."
                )
```

```
        return template


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete launch template
{self.launch_template_name}: {err}."
                )


    def get_availability_zones(self):
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get availability zones: {err}.")
        else:
```

```
            return zones


    def create_group(self, group_size):
        """
        Creates an EC2 Auto Scaling group with the specified size.

        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        zones = []
        try:
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
                MaxSize=group_size,
            )
            log.info(
                "Created EC2 Auto Scaling group %s with availability zones %s.",
                self.launch_template_name,
                zones,
            )
        except ClientError as err:
            if err.response["Error"]["Code"] == "AlreadyExists":
                log.info(
                    "EC2 Auto Scaling group %s already exists, nothing to do.",
                    self.group_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create EC2 Auto Scaling group {self.group_name}:
{err}"
                )
        return zones
```

```
    def get_instances(self):
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: Data about the instances.
        """
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instances for Auto Scaling group
{self.group_name}: {err}"
            )
        else:
            return instance_ids


    def terminate_instance(self, instance_id):
        """
        Terminates and instances in an EC2 Auto Scaling group. After an instance
is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id, ShouldDecrementDesiredCapacity=False
            )
            log.info("Terminated instance %s.", instance_id)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't terminate instance {instance_id}:
{err}")

    def attach_load_balancer_target_group(self, lb_target_group):
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
```

```
        The target group specifies how the load balancer forward requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't attach load balancer target group
{lb_target_group['TargetGroupName']}\n"
                f"to auto scaling group {self.group_name}"
            )


    def _try_terminate_instance(self, inst_id):
        stopping = False
        log.info(f"Stopping {inst_id}.")
        while not stopping:
            try:
                self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                    InstanceId=inst_id, ShouldDecrementDesiredCapacity=True
                )
                stopping = True
            except ClientError as err:
                if err.response["Error"]["Code"] == "ScalingActivityInProgress":
                    log.info("Scaling activity in progress for %s. Waiting...",
inst_id)
                    time.sleep(10)
                else:
                    raise AutoScalerError(f"Couldn't stop instance {inst_id}:
{err}.")

    def _try_delete_group(self):
```

```
        """
        Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
        the function waits and retries until the group is successfully deleted.
        """
        stopped = False
        while not stopped:
            try:
                self.autoscaling_client.delete_auto_scaling_group(
                    AutoScalingGroupName=self.group_name
                )
                stopped = True
                log.info("Deleted EC2 Auto Scaling group %s.", self.group_name)
            except ClientError as err:
                if (
                    err.response["Error"]["Code"] == "ResourceInUse"
                    or err.response["Error"]["Code"] ==
"ScalingActivityInProgress"
                ):
                    log.info(
                        "Some instances are still running. Waiting for them to
stop..."
                    )
                    time.sleep(10)
                else:
                    raise AutoScalerError(
                        f"Couldn't delete group {self.group_name}: {err}."
                    )

    def delete_group(self):
        """
        Terminates all instances in the group, deletes the EC2 Auto Scaling
group.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=self.group_name, MinSize=0
                )
```

```
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self._try_terminate_instance(inst_id)
                self._try_delete_group()
            else:
                log.info("No groups found named %s, nothing to do.",
 self.group_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't delete group {self.group_name}:
 {err}.")


    def get_default_vpc(self):
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get default VPC: {err}")
        else:
            return response["Vpcs"][0]


    def verify_inbound_port(self, vpc, port, ip_address):
        """
        Verify the default security group of the specified VPC allows ingress
 from this
        computer. This can be done by allowing ingress from this computer's IP
        address. In some situations, such as connecting from a corporate network,
 you
        must instead specify a prefix list ID. You can also temporarily open the
 port to
        any IP address while running this example. If you do, be sure to remove
 public
        access when you're done.

        :param vpc: The VPC used by this example.
        :param port: The port to verify.
```

```
        :param ip_address: This computer's IP address.
        :return: The default security group of the specific VPC, and a value that
indicates
                 whether the specified port is open.
        """
        try:
            response = self.ec2_client.describe_security_groups(
                Filters=[
                    {"Name": "group-name", "Values": ["default"]},
                    {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
                ]
            )
            sec_group = response["SecurityGroups"][0]
            port_is_open = False
            log.info("Found default security group %s.", sec_group["GroupId"])
            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info("Found inbound rule: %s", ip_perm)
                    for ip_range in ip_perm["IpRanges"]:
                        cidr = ip_range.get("CidrIp", "")
                        if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                            port_is_open = True
                    if ip_perm["PrefixListIds"]:
                        port_is_open = True
                    if not port_is_open:
                        log.info(
                            "The inbound rule does not appear to be open to
either this computer's IP\n"
                            "address of %s, to all IP addresses (0.0.0.0/0), or
to a prefix list ID.",
                            ip_address,
                        )
                    else:
                        break
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't verify inbound rule for port {port} for VPC
{vpc['VpcId']}: {err}"
            )
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id, port, ip_address):
```

```
        """
        Add an ingress rule to the specified security group that allows access on
the
        specified port from the specified IP address.

        :param sec_group_id: The ID of the security group to modify.
        :param port: The port to open.
        :param ip_address: The IP address that is granted access.
        """
        try:
            self.ec2_client.authorize_security_group_ingress(
                GroupId=sec_group_id,
                CidrIp=f"{ip_address}/32",
                FromPort=port,
                ToPort=port,
                IpProtocol="tcp",
            )
            log.info(
                "Authorized ingress to %s on port %s from %s.",
                sec_group_id,
                port,
                ip_address,
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't authorize ingress to {sec_group_id} on port {port}
from {ip_address}: {err}"
            )


    def get_subnets(self, vpc_id, zones):
        """
        Gets the default subnets in a VPC for a specified list of Availability
Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        try:
            response = self.ec2_client.describe_subnets(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
```

```
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )
            subnets = response["Subnets"]
            log.info("Found %s subnets for the specified zones.", len(subnets))
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get subnets: {err}")
        else:
            return subnets
```

建立包裝 Elastic Load Balancing 動作的類別。

```
class LoadBalancer:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, target_group_name, load_balancer_name, elb_client):
        """
        :param target_group_name: The name of the target group associated with
 the load balancer.
        :param load_balancer_name: The name of the load balancer.
        :param elb_client: A Boto3 Elastic Load Balancing client.
        """
        self.target_group_name = target_group_name
        self.load_balancer_name = load_balancer_name
        self.elb_client = elb_client
        self._endpoint = None


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from a Boto3 client.

        :param resource_prefix: The prefix to give to AWS resources created by
 this class.
        """
        elb_client = boto3.client("elbv2")
        return cls(f"{resource_prefix}-tg", f"{resource_prefix}-lb", elb_client)
```

```
    def endpoint(self):
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        if self._endpoint is None:
            try:
                response = self.elb_client.describe_load_balancers(
                    Names=[self.load_balancer_name]
                )
                self._endpoint = response["LoadBalancers"][0]["DNSName"]
            except ClientError as err:
                raise LoadBalancerError(
                    f"Couldn't get the endpoint for load balancer
{self.load_balancer_name}: {err}"
                )
        return self._endpoint


    def create_target_group(self, protocol, port, vpc_id):
        """
        Creates an Elastic Load Balancing target group. The target group
specifies how
        the load balancer forward requests to instances in the group and how
instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
times and
        lower thresholds. In production, you might want to decrease the
sensitivity of
        your health checks to avoid unwanted failures.

        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
            response = self.elb_client.create_target_group(
                Name=self.target_group_name,
                Protocol=protocol,
                Port=port,
```

```
                HealthCheckPath="/healthcheck",
                HealthCheckIntervalSeconds=10,
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info("Created load balancing target group %s.",
self.target_group_name)
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't create load balancing target group
{self.target_group_name}: {err}"
            )
        else:
            return target_group


    def delete_target_group(self):
        """
        Deletes the target group.
        """
        done = False
        while not done:
            try:
                response = self.elb_client.describe_target_groups(
                    Names=[self.target_group_name]
                )
                tg_arn = response["TargetGroups"][0]["TargetGroupArn"]
                self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
                log.info(
                    "Deleted load balancing target group %s.",
self.target_group_name
                )
                done = True
            except ClientError as err:
                if err.response["Error"]["Code"] == "TargetGroupNotFound":
                    log.info(
                        "Load balancer target group %s not found, nothing to
do.",
                        self.target_group_name,
                    )
                    done = True
```

```
                elif err.response["Error"]["Code"] == "ResourceInUse":
                    log.info(
                        "Target group not yet released from load balancer,
waiting..."
                    )
                    time.sleep(10)
                else:
                    raise LoadBalancerError(
                        f"Couldn't delete load balancing target group
{self.target_group_name}: {err}"
                    )


   def create_load_balancer(self, subnet_ids, target_group):
       """
       Creates an Elastic Load Balancing load balancer that uses the specified
subnets
       and forwards requests to the specified target group.

       :param subnet_ids: A list of subnets to associate with the load balancer.
       :param target_group: An existing target group that is added as a listener
to the
                            load balancer.
       :return: Data about the newly created load balancer.
       """
       try:
           response = self.elb_client.create_load_balancer(
               Name=self.load_balancer_name, Subnets=subnet_ids
           )
           load_balancer = response["LoadBalancers"][0]
           log.info("Created load balancer %s.", self.load_balancer_name)
           waiter = self.elb_client.get_waiter("load_balancer_available")
           log.info("Waiting for load balancer to be available...")
           waiter.wait(Names=[self.load_balancer_name])
           log.info("Load balancer is available!")
           self.elb_client.create_listener(
               LoadBalancerArn=load_balancer["LoadBalancerArn"],
               Protocol=target_group["Protocol"],
               Port=target_group["Port"],
               DefaultActions=[
                   {
                       "Type": "forward",
                       "TargetGroupArn": target_group["TargetGroupArn"],
                   }
```

```
                ],
            )
            log.info(
                "Created listener to forward traffic from load balancer %s to
target group %s.",
                self.load_balancer_name,
                target_group["TargetGroupName"],
            )
        except ClientError as err:
            raise LoadBalancerError(
                f"Failed to create load balancer {self.load_balancer_name}"
                f"and add a listener for target group
{target_group['TargetGroupName']}: {err}"
            )
        else:
            self._endpoint = load_balancer["DNSName"]
            return load_balancer


    def delete_load_balancer(self):
        """
        Deletes a load balancer.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[self.load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[self.load_balancer_name])
        except ClientError as err:
            if err.response["Error"]["Code"] == "LoadBalancerNotFound":
                log.info(
                    "Load balancer %s does not exist, nothing to do.",
                    self.load_balancer_name,
                )
            else:
                raise LoadBalancerError(
                    f"Couldn't delete load balancer {self.load_balancer_name}:
{err}"
                )
```

```
    def verify_load_balancer_endpoint(self):
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.
        """
        success = False
        retries = 3
        while not success and retries > 0:
            try:
                lb_response = requests.get(f"http://{self.endpoint()}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    success = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
                )
                retries -= 1
                time.sleep(10)
        return success

    def check_target_health(self):
        """
        Checks the health of the instances in the target group.

        :return: The health status of the target group.
        """
        try:
            tg_response = self.elb_client.describe_target_groups(
                Names=[self.target_group_name]
            )
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            raise LoadBalancerError(
```

```
                f"Couldn't check health of {self.target_group_name} targets:
 {err}"
            )
        else:
            return health_response["TargetHealthDescriptions"]
```

建立使用 DynamoDB 模擬建議服務的類別。

```
class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
 movies,
    and songs.
    """

    def __init__(self, table_name, dynamodb_client):
        """
        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    @classmethod
    def from_client(cls, table_name):
        """
        Creates this class from a Boto3 client.

        :param table_name: The name of the DynamoDB recommendations table.
        """
        ddb_client = boto3.client("dynamodb")
        return cls(table_name, ddb_client)

    def create(self):
        """
        Creates a DynamoDB table to use a recommendation service. The table has a
        hash key named 'MediaType' that defines the type of media recommended,
 such as
```

```
        Book or Movie, and a range key named 'ItemId' that, combined with the
MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
            log.info("Creating table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be do.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file):
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
```

```
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
 batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
 data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
 {err}"
            )

    def destroy(self):
        """
        Deletes the recommendations table.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                log.info("Table %s does not exist, nothing to do.",
 self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

建立包裝 Systems Manager 動作的類別。

```
class ParameterHelper:
    """
    Encapsulates Systems Manager parameters. This example uses these parameters
 to drive
    the demonstration of resilient architecture, such as failure of a dependency
 or
    how the service responds to a health check.
```

```
    """

    table = "doc-example-resilient-architecture-table"
    failure_response = "doc-example-resilient-architecture-failure-response"
    health_check = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name, ssm_client):
        """
        :param table_name: The name of the DynamoDB table that is used as a
recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name

    @classmethod
    def from_client(cls, table_name):
        ssm_client = boto3.client("ssm")
        return cls(table_name, ssm_client)

    def reset(self):
        """
        Resets the Systems Manager parameters to starting values for the demo.
        These are the name of the DynamoDB recommendation table, no response when
a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name, value):
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting demo parameter %s to '%s'.", name, value)
```

```
        except ClientError as err:
            raise ParameterHelperError(
                f"Couldn't set parameter {name} to {value}: {err}"
            )
```

- 如需 API 的詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考資料》中的下列主題。
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances
  - ReplaceIamInstanceProfileAssociation

- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用開 AWS 發套件開始使用 Amazon EC2 執行個體

下列程式碼範例示範如何：

- 建立金鑰對和安全群組。
- 選取 Amazon Machine Image (AMI) 和相容的執行個體類型，然後建立執行個體。
- 停止並重新啟動執行個體。
- 將彈性 IP 地址與您的執行個體建立關聯.
- 使用 SSH 連線至執行個體，然後清理資源。

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

在命令提示中執行案例。

```
/// <summary>
/// Show Amazon Elastic Compute Cloud (Amazon EC2) Basics actions.
/// </summary>
public class EC2Basics
{
    /// <summary>
    /// Perform the actions defined for the Amazon EC2 Basics scenario.
    /// </summary>
    /// <param name="args">Command line arguments.</param>
    /// <returns>A Task object.</returns>
```

```
    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon EC2 and Amazon Simple Systems
        // Management Service.
        using var host =
Microsoft.Extensions.Hosting.Host.CreateDefaultBuilder(args)
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonEC2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddTransient<EC2Wrapper>()
                    .AddTransient<SsmWrapper>()
            )
            .Build();

        // Now the client is available for injection.
        var ec2Client = host.Services.GetRequiredService<IAmazonEC2>();
        var ec2Methods = new EC2Wrapper(ec2Client);

        var ssmClient =
host.Services.GetRequiredService<IAmazonSimpleSystemsManagement>();
        var ssmMethods = new SsmWrapper(ssmClient);
        var uiMethods = new UiMethods();

        var uniqueName = Guid.NewGuid().ToString();
        var keyPairName = "mvp-example-key-pair" + uniqueName;
        var groupName = "ec2-scenario-group" + uniqueName;
        var groupDescription = "A security group created for the EC2 Basics
scenario.";

        // Start the scenario.
        uiMethods.DisplayOverview();
        uiMethods.PressEnter();

        // Create the key pair.
        uiMethods.DisplayTitle("Create RSA key pair");
        Console.Write("Let's create an RSA key pair that you can be use to ");
        Console.WriteLine("securely connect to your EC2 instance.");
        var keyPair = await ec2Methods.CreateKeyPair(keyPairName);

        // Save key pair information to a temporary file.
        var tempFileName = ec2Methods.SaveKeyPair(keyPair);

        Console.WriteLine($"Created the key pair: {keyPair.KeyName} and saved it
to: {tempFileName}");
```

```
        string? answer;
        do
        {
            Console.Write("Would you like to list your existing key pairs? ");
            answer = Console.ReadLine();
        } while (answer!.ToLower() != "y" && answer.ToLower() != "n");

        if (answer == "y")
        {
            // List existing key pairs.
            uiMethods.DisplayTitle("Existing key pairs");

            // Passing an empty string to the DescribeKeyPairs method will return
            // a list of all existing key pairs.
            var keyPairs = await ec2Methods.DescribeKeyPairs("");
            keyPairs.ForEach(kp =>
            {
                Console.WriteLine($"{kp.KeyName} created at: {kp.CreateTime}
Fingerprint: {kp.KeyFingerprint}");
            });
        }
        uiMethods.PressEnter();

        // Create the security group.
        Console.WriteLine("Let's create a security group to manage access to your
instance.");
        var secGroupId = await ec2Methods.CreateSecurityGroup(groupName,
groupDescription);
        Console.WriteLine("Let's add rules to allow all HTTP and HTTPS inbound
traffic and to allow SSH only from your current IP address.");

        uiMethods.DisplayTitle("Security group information");
        var secGroups = await ec2Methods.DescribeSecurityGroups(secGroupId);

        Console.WriteLine($"Created security group {groupName} in your default
VPC.");
        secGroups.ForEach(group =>
        {
            ec2Methods.DisplaySecurityGroupInfoAsync(group);
        });
        uiMethods.PressEnter();

        Console.WriteLine("Now we'll authorize the security group we just created
so that it can");
```

```
        Console.WriteLine("access the EC2 instances you create.");
        var success = await ec2Methods.AuthorizeSecurityGroupIngress(groupName);

        secGroups = await ec2Methods.DescribeSecurityGroups(secGroupId);
        Console.WriteLine($"Now let's look at the permissions again.");
        secGroups.ForEach(group =>
        {
            ec2Methods.DisplaySecurityGroupInfoAsync(group);
        });
        uiMethods.PressEnter();

        // Get list of available Amazon Linux 2 Amazon Machine Images (AMIs).
        var parameters = await ssmMethods.GetParametersByPath("/aws/service/ami-
amazon-linux-latest");

        List<string> imageIds = parameters.Select(param => param.Value).ToList();

        var images = await ec2Methods.DescribeImages(imageIds);

        var i = 1;
        images.ForEach(image =>
        {
            Console.WriteLine($"\t{i++}\t{image.Description}");
        });

        int choice;
        bool validNumber = false;

        do
        {
            Console.Write("Please select an image: ");
            var selImage = Console.ReadLine();
            validNumber = int.TryParse(selImage, out choice);
        } while (!validNumber);

        var selectedImage = images[choice - 1];

        // Display available instance types.
        uiMethods.DisplayTitle("Instance Types");
        var instanceTypes = await
 ec2Methods.DescribeInstanceTypes(selectedImage.Architecture);

        i = 1;
        instanceTypes.ForEach(instanceType =>
```

```
        {
            Console.WriteLine($"\t{i++}\t{instanceType.InstanceType}");
        });

        do
        {
            Console.Write("Please select an instance type: ");
            var selImage = Console.ReadLine();
            validNumber = int.TryParse(selImage, out choice);
        } while (!validNumber);

        var selectedInstanceType = instanceTypes[choice - 1].InstanceType;

        // Create an EC2 instance.
        uiMethods.DisplayTitle("Creating an EC2 Instance");
        var instanceId = await ec2Methods.RunInstances(selectedImage.ImageId,
 selectedInstanceType, keyPairName, secGroupId);
        Console.Write("Waiting for the instance to start.");
        var isRunning = false;
        do
        {
            isRunning = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Running);
        } while (!isRunning);

        uiMethods.PressEnter();

        var instance = await ec2Methods.DescribeInstance(instanceId);
        uiMethods.DisplayTitle("New Instance Information");
        ec2Methods.DisplayInstanceInformation(instance);

        Console.WriteLine("\nYou can use SSH to connect to your instance. For
 example:");
        Console.WriteLine($"\tssh -i {tempFileName} ec2-
user@{instance.PublicIpAddress}");

        uiMethods.PressEnter();

        Console.WriteLine("Now we'll stop the instance and then start it again to
 see what's changed.");

        await ec2Methods.StopInstances(instanceId);
        var hasStopped = false;
        do
```

```
        {
            hasStopped = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Stopped);
        } while (!hasStopped);

        Console.WriteLine("\nThe instance has stopped.");

        Console.WriteLine("Now let's start it up again.");
        await ec2Methods.StartInstances(instanceId);
        Console.Write("Waiting for instance to start. ");

        isRunning = false;
        do
        {
            isRunning = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Running);
        } while (!isRunning);

        Console.WriteLine("\nLet's see what changed.");

        instance = await ec2Methods.DescribeInstance(instanceId);
        uiMethods.DisplayTitle("New Instance Information");
        ec2Methods.DisplayInstanceInformation(instance);

        Console.WriteLine("\nNotice the change in the SSH information:");
        Console.WriteLine($"\tssh -i {tempFileName} ec2-
user@{instance.PublicIpAddress}");

        uiMethods.PressEnter();

        Console.WriteLine("Now we will stop the instance again. Then we will
 create and associate an");
        Console.WriteLine("Elastic IP address to use with our instance.");

        await ec2Methods.StopInstances(instanceId);
        hasStopped = false;
        do
        {
            hasStopped = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Stopped);
        } while (!hasStopped);

        Console.WriteLine("\nThe instance has stopped.");
        uiMethods.PressEnter();
```

```
        uiMethods.DisplayTitle("Allocate Elastic IP address");
        Console.WriteLine("You can allocate an Elastic IP address and associate
 it with your instance\nto keep a consistent IP address even when your instance
 restarts.");
        var allocationId = await ec2Methods.AllocateAddress();
        Console.WriteLine("Now we will associate the Elastic IP address with our
 instance.");
        var associationId = await ec2Methods.AssociateAddress(allocationId,
 instanceId);

        // Start the instance again.
        Console.WriteLine("Now let's start the instance again.");
        await ec2Methods.StartInstances(instanceId);
        Console.Write("Waiting for instance to start. ");

        isRunning = false;
        do
        {
            isRunning = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Running);
        } while (!isRunning);

        Console.WriteLine("\nLet's see what changed.");

        instance = await ec2Methods.DescribeInstance(instanceId);
        uiMethods.DisplayTitle("Instance information");
        ec2Methods.DisplayInstanceInformation(instance);

        Console.WriteLine("\nHere is the SSH information:");
        Console.WriteLine($"\tssh -i {tempFileName} ec2-
user@{instance.PublicIpAddress}");

        Console.WriteLine("Let's stop and start the instance again.");
        uiMethods.PressEnter();

        await ec2Methods.StopInstances(instanceId);

        hasStopped = false;
        do
        {
            hasStopped = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Stopped);
        } while (!hasStopped);
```

```
        Console.WriteLine("\nThe instance has stopped.");

        Console.WriteLine("Now let's start it up again.");
        await ec2Methods.StartInstances(instanceId);
        Console.Write("Waiting for instance to start. ");

        isRunning = false;
        do
        {
            isRunning = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Running);
        } while (!isRunning);

        instance = await ec2Methods.DescribeInstance(instanceId);
        uiMethods.DisplayTitle("New Instance Information");
        ec2Methods.DisplayInstanceInformation(instance);
        Console.WriteLine("Note that the IP address did not change this time.");
        uiMethods.PressEnter();

        uiMethods.DisplayTitle("Clean up resources");

        Console.WriteLine("Now let's clean up the resources we created.");

        // Terminate the instance.
        Console.WriteLine("Terminating the instance we created.");
        var stateChange = await ec2Methods.TerminateInstances(instanceId);

        // Wait for the instance state to be terminated.
        var hasTerminated = false;
        do
        {
            hasTerminated = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Terminated);
        } while (!hasTerminated);

        Console.WriteLine($"\nThe instance {instanceId} has been terminated.");
        Console.WriteLine("Now we can disassociate the Elastic IP address and
release it.");

        // Disassociate the Elastic IP address.
        var disassociated = ec2Methods.DisassociateIp(associationId);

        // Delete the Elastic IP address.
```

```
        var released = ec2Methods.ReleaseAddress(allocationId);

        // Delete the security group.
        Console.WriteLine($"Deleting the Security Group: {groupName}.");
        success = await ec2Methods.DeleteSecurityGroup(secGroupId);
        if (success)
        {
            Console.WriteLine($"Successfully deleted {groupName}.");
        }

        // Delete the RSA key pair.
        Console.WriteLine($"Deleting the key pair: {keyPairName}");
        await ec2Methods.DeleteKeyPair(keyPairName);
        Console.WriteLine("Deleting the temporary file with the key
 information.");
        ec2Methods.DeleteTempFile(tempFileName);
        uiMethods.PressEnter();

        uiMethods.DisplayTitle("EC2 Basics Scenario completed.");
        uiMethods.PressEnter();
    }
}
```

定義包裝 EC2 動作的類別。

```
/// <summary>
/// Methods of this class perform Amazon Elastic Compute Cloud (Amazon EC2).
/// </summary>
public class EC2Wrapper
{
    private readonly IAmazonEC2 _amazonEC2;

    public EC2Wrapper(IAmazonEC2 amazonService)
    {
        _amazonEC2 = amazonService;
    }

    /// <summary>
    /// Allocate an Elastic IP address.
    /// </summary>
    /// <returns>The allocation Id of the allocated address.</returns>
    public async Task<string> AllocateAddress()
```

```
    {
        var request = new AllocateAddressRequest();

        var response = await _amazonEC2.AllocateAddressAsync(request);
        return response.AllocationId;
    }

    /// <summary>
    /// Associate an Elastic IP address to an EC2 instance.
    /// </summary>
    /// <param name="allocationId">The allocation Id of an Elastic IP address.</
param>
    /// <param name="instanceId">The instance Id of the EC2 instance to
    /// associate the address with.</param>
    /// <returns>The association Id that represents
    /// the association of the Elastic IP address with an instance.</returns>
    public async Task<string> AssociateAddress(string allocationId, string
 instanceId)
    {
        var request = new AssociateAddressRequest
        {
            AllocationId = allocationId,
            InstanceId = instanceId
        };

        var response = await _amazonEC2.AssociateAddressAsync(request);
        return response.AssociationId;
    }

    /// <summary>
    /// Authorize the local computer ingress to EC2 instances associated
    /// with the virtual private cloud (VPC) security group.
    /// </summary>
    /// <param name="groupName">The name of the security group.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AuthorizeSecurityGroupIngress(string groupName)
    {
        // Get the IP address for the local computer.
        var ipAddress = await GetIpAddress();
        Console.WriteLine($"Your IP address is: {ipAddress}");
        var ipRanges = new List<IpRange> { new IpRange { CidrIp =
 $"{ipAddress}/32" } };
        var permission = new IpPermission
        {
```

```
            Ipv4Ranges = ipRanges,
            IpProtocol = "tcp",
            FromPort = 22,
            ToPort = 22
        };
        var permissions = new List<IpPermission> { permission };
        var response = await _amazonEC2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest(groupName, permissions));
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Authorize the local computer for ingress to
    /// the Amazon EC2 SecurityGroup.
    /// </summary>
    /// <returns>The IPv4 address of the computer running the scenario.</returns>
    private static async Task<string> GetIpAddress()
    {
        var httpClient = new HttpClient();
        var ipString = await httpClient.GetStringAsync("https://
checkip.amazonaws.com");

        // The IP address is returned with a new line
        // character on the end. Trim off the whitespace and
        // return the value to the caller.
        return ipString.Trim();
    }

    /// <summary>
    /// Create an Amazon EC2 key pair.
    /// </summary>
    /// <param name="keyPairName">The name for the new key pair.</param>
    /// <returns>The Amazon EC2 key pair created.</returns>
    public async Task<KeyPair?> CreateKeyPair(string keyPairName)
    {
        var request = new CreateKeyPairRequest
        {
            KeyName = keyPairName,
        };

        var response = await _amazonEC2.CreateKeyPairAsync(request);

        if (response.HttpStatusCode == HttpStatusCode.OK)
        {
```

```
            var kp = response.KeyPair;
            return kp;
        }
        else
        {
            Console.WriteLine("Could not create key pair.");
            return null;
        }
    }

    /// <summary>
    /// Save KeyPair information to a temporary file.
    /// </summary>
    /// <param name="keyPair">The name of the key pair.</param>
    /// <returns>The full path to the temporary file.</returns>
    public string SaveKeyPair(KeyPair keyPair)
    {
        var tempPath = Path.GetTempPath();
        var tempFileName = $"{tempPath}\\{Path.GetRandomFileName()}";
        var pemFileName = Path.ChangeExtension(tempFileName, "pem");

        // Save the key pair to a file in a temporary folder.
        using var stream = new FileStream(pemFileName, FileMode.Create);
        using var writer = new StreamWriter(stream);
        writer.WriteLine(keyPair.KeyMaterial);

        return pemFileName;
    }

    /// <summary>
    /// Create an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupName">The name for the new security group.</param>
    /// <param name="groupDescription">A description of the new security group.</
param>
    /// <returns>The group Id of the new security group.</returns>
    public async Task<string> CreateSecurityGroup(string groupName, string
 groupDescription)
    {
        var response = await _amazonEC2.CreateSecurityGroupAsync(
            new CreateSecurityGroupRequest(groupName, groupDescription));

        return response.GroupId;
    }
```

```
    /// <summary>
    /// Create a new Amazon EC2 VPC.
    /// </summary>
    /// <param name="cidrBlock">The CIDR block for the new security group.</
param>
    /// <returns>The VPC Id of the new VPC.</returns>
    public async Task<string?> CreateVPC(string cidrBlock)
    {

        try
        {
            var response = await _amazonEC2.CreateVpcAsync(new CreateVpcRequest
            {
                CidrBlock = cidrBlock,
            });

            Vpc vpc = response.Vpc;
            Console.WriteLine($"Created VPC with ID: {vpc.VpcId}.");
            return vpc.VpcId;
        }
        catch (AmazonEC2Exception ex)
        {
            Console.WriteLine($"Couldn't create VPC because: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Delete an Amazon EC2 key pair.
    /// </summary>
    /// <param name="keyPairName">The name of the key pair to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteKeyPair(string keyPairName)
    {
        try
        {
            await _amazonEC2.DeleteKeyPairAsync(new
 DeleteKeyPairRequest(keyPairName)).ConfigureAwait(false);
            return true;
        }
        catch (Exception ex)
        {
```

```
            Console.WriteLine($"Couldn't delete the key pair because:
{ex.Message}");
            return false;
        }
    }

    /// <summary>
    /// Delete the temporary file where the key pair information was saved.
    /// </summary>
    /// <param name="tempFileName">The path to the temporary file.</param>
    public void DeleteTempFile(string tempFileName)
    {
        if (File.Exists(tempFileName))
        {
            File.Delete(tempFileName);
        }
    }

    /// <summary>
    /// Delete an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteSecurityGroup(string groupId)
    {
        var response = await _amazonEC2.DeleteSecurityGroupAsync(new
DeleteSecurityGroupRequest { GroupId = groupId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Delete an Amazon EC2 VPC.
    /// </summary>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteVpc(string vpcId)
    {
        var request = new DeleteVpcRequest
        {
            VpcId = vpcId,
        };

        var response = await _amazonEC2.DeleteVpcAsync(request);

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
```

```
    }

    /// <summary>
    /// Get information about existing Amazon EC2 images.
    /// </summary>
    /// <returns>A list of image information.</returns>
    public async Task<List<Image>> DescribeImages(List<string>? imageIds)
    {
        var request = new DescribeImagesRequest();
        if (imageIds is not null)
        {
            // If the imageIds list is not null, add the list
            // to the request object.
            request.ImageIds = imageIds;
        }

        var response = await _amazonEC2.DescribeImagesAsync(request);
        return response.Images;
    }

    /// <summary>
    /// Display the information returned by DescribeImages.
    /// </summary>
    /// <param name="images">The list of image information to display.</param>
    public void DisplayImageInfo(List<Image> images)
    {
        images.ForEach(image =>
        {
            Console.WriteLine($"{image.Name} Created on: {image.CreationDate}");
        });

    }

    /// <summary>
    /// Get information about an Amazon EC2 instance.
    /// </summary>
    /// <param name="instanceId">The instance Id of the EC2 instance.</param>
    /// <returns>An EC2 instance.</returns>
    public async Task<Instance> DescribeInstance(string instanceId)
    {
        var response = await _amazonEC2.DescribeInstancesAsync(
            new DescribeInstancesRequest { InstanceIds = new List<string>
{ instanceId } });
        return response.Reservations[0].Instances[0];
```

```
    }

    /// <summary>
    /// Display EC2 instance information.
    /// </summary>
    /// <param name="instance">The instance Id of the EC2 instance.</param>
    public void DisplayInstanceInformation(Instance instance)
    {
        Console.WriteLine($"ID: {instance.InstanceId}");
        Console.WriteLine($"Image ID: {instance.ImageId}");
        Console.WriteLine($"{instance.InstanceType}");
        Console.WriteLine($"Key Name: {instance.KeyName}");
        Console.WriteLine($"VPC ID: {instance.VpcId}");
        Console.WriteLine($"Public IP: {instance.PublicIpAddress}");
        Console.WriteLine($"State: {instance.State.Name}");
    }

    /// <summary>
    /// Get information about existing EC2 images.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task DescribeInstances()
    {
        // List all EC2 instances.
        await GetInstanceDescriptions();

        string tagName = "IncludeInList";
        string tagValue = "Yes";
        await GetInstanceDescriptionsFiltered(tagName, tagValue);
    }

    /// <summary>
    /// Get information for all existing Amazon EC2 instances.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task GetInstanceDescriptions()
    {
        Console.WriteLine("Showing all instances:");
        var paginator = _amazonEC2.Paginators.DescribeInstances(new
DescribeInstancesRequest());

        await foreach (var response in paginator.Responses)
        {
            foreach (var reservation in response.Reservations)
```

```
            {
                foreach (var instance in reservation.Instances)
                {
                    Console.Write($"Instance ID: {instance.InstanceId}");
                    Console.WriteLine($"\tCurrent State: {instance.State.Name}");
                }
            }
        }
    }

    /// <summary>
    /// Get information about EC2 instances filtered by a tag name and value.
    /// </summary>
    /// <param name="tagName">The name of the tag to filter on.</param>
    /// <param name="tagValue">The value of the tag to look for.</param>
    /// <returns>Async task.</returns>
    public async Task GetInstanceDescriptionsFiltered(string tagName, string
tagValue)
    {
        // This tag filters the results of the instance list.
        var filters = new List<Filter>
        {
            new Filter
            {
                Name = $"tag:{tagName}",
                Values = new List<string>
                {
                    tagValue,
                },
            },
        };
        var request = new DescribeInstancesRequest
        {
            Filters = filters,
        };

        Console.WriteLine("\nShowing instances with tag: \"IncludeInList\" set to
\"Yes\".");
        var paginator = _amazonEC2.Paginators.DescribeInstances(request);

        await foreach (var response in paginator.Responses)
        {
            foreach (var reservation in response.Reservations)
            {
```

```
                foreach (var instance in reservation.Instances)
                {
                    Console.Write($"Instance ID: {instance.InstanceId} ");
                    Console.WriteLine($"\tCurrent State: {instance.State.Name}");
                }
            }
        }
    }

    /// <summary>
    /// Describe the instance types available.
    /// </summary>
    /// <returns>A list of instance type information.</returns>
    public async Task<List<InstanceTypeInfo>>
 DescribeInstanceTypes(ArchitectureValues architecture)
    {
        var request = new DescribeInstanceTypesRequest();

        var filters = new List<Filter>
            { new Filter("processor-info.supported-architecture", new
 List<string> { architecture.ToString() }) };
        filters.Add(new Filter("instance-type", new() { "*.micro", "*.small" }));

        request.Filters = filters;
        var instanceTypes = new List<InstanceTypeInfo>();

        var paginator = _amazonEC2.Paginators.DescribeInstanceTypes(request);
        await foreach (var instanceType in paginator.InstanceTypes)
        {
            instanceTypes.Add(instanceType);
        }
        return instanceTypes;
    }

    /// <summary>
    /// Display the instance type information returned by
 DescribeInstanceTypesAsync.
    /// </summary>
    /// <param name="instanceTypes">The list of instance type information.</
param>
    public void DisplayInstanceTypeInfo(List<InstanceTypeInfo> instanceTypes)
    {
        instanceTypes.ForEach(type =>
        {
```

```
            Console.WriteLine($"{type.InstanceType}\t{type.MemoryInfo}");
        });
    }

    /// <summary>
    /// Get information about an Amazon EC2 key pair.
    /// </summary>
    /// <param name="keyPairName">The name of the key pair.</param>
    /// <returns>A list of key pair information.</returns>
    public async Task<List<KeyPairInfo>> DescribeKeyPairs(string keyPairName)
    {
        var request = new DescribeKeyPairsRequest();
        if (!string.IsNullOrEmpty(keyPairName))
        {
            request = new DescribeKeyPairsRequest
            {
                KeyNames = new List<string> { keyPairName }
            };
        }
        var response = await _amazonEC2.DescribeKeyPairsAsync(request);
        return response.KeyPairs.ToList();
    }


    /// <summary>
    /// Retrieve information for an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupId">The Id of the Amazon EC2 security group.</param>
    /// <returns>A list of security group information.</returns>
    public async Task<List<SecurityGroup>> DescribeSecurityGroups(string groupId)
    {
        var request = new DescribeSecurityGroupsRequest();
        var groupIds = new List<string> { groupId };
        request.GroupIds = groupIds;

        var response = await _amazonEC2.DescribeSecurityGroupsAsync(request);
        return response.SecurityGroups;
    }

    /// <summary>
    /// Display the information returned by the call to
    /// DescribeSecurityGroupsAsync.
    /// </summary>
    /// <param name="securityGroup">A list of security group information.</param>
```

```
    public void DisplaySecurityGroupInfoAsync(SecurityGroup securityGroup)
    {
        Console.WriteLine($"{securityGroup.GroupName}");
        Console.WriteLine("Ingress permissions:");
        securityGroup.IpPermissions.ForEach(permission =>
        {
            Console.WriteLine($"\tFromPort: {permission.FromPort}");
            Console.WriteLine($"\tIpProtocol: {permission.IpProtocol}");

            Console.Write($"\tIpv4Ranges: ");
            permission.Ipv4Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIp} "); });

            Console.WriteLine($"\n\tIpv6Ranges:");
            permission.Ipv6Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIpv6} "); });

            Console.Write($"\n\tPrefixListIds: ");
            permission.PrefixListIds.ForEach(id => Console.Write($"{id.Id} "));

            Console.WriteLine($"\n\tTo Port: {permission.ToPort}");
        });
        Console.WriteLine("Egress permissions:");
        securityGroup.IpPermissionsEgress.ForEach(permission =>
        {
            Console.WriteLine($"\tFromPort: {permission.FromPort}");
            Console.WriteLine($"\tIpProtocol: {permission.IpProtocol}");

            Console.Write($"\tIpv4Ranges: ");
            permission.Ipv4Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIp} "); });

            Console.WriteLine($"\n\tIpv6Ranges:");
            permission.Ipv6Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIpv6} "); });

            Console.Write($"\n\tPrefixListIds: ");
            permission.PrefixListIds.ForEach(id => Console.Write($"{id.Id} "));

            Console.WriteLine($"\n\tTo Port: {permission.ToPort}");
        });
    }
```

```
    /// <summary>
    /// Disassociate an Elastic IP address from an EC2 instance.
    /// </summary>
    /// <param name="associationId">The association Id.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DisassociateIp(string associationId)
    {
        var response = await _amazonEC2.DisassociateAddressAsync(
            new DisassociateAddressRequest { AssociationId = associationId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Retrieve a list of available Amazon Linux images.
    /// </summary>
    /// <returns>A list of image information.</returns>
    public async Task<List<Image>> GetEC2AmiList()
    {
        var filter = new Filter { Name = "architecture", Values = new
List<string> { "x86_64" } };
        var filters = new List<Filter> { filter };
        var response = await _amazonEC2.DescribeImagesAsync(new
DescribeImagesRequest { Filters = filters });
        return response.Images;
    }

    /// <summary>
    /// Reboot EC2 instances.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the instances that will be
rebooted.</param>
    /// <returns>Async task.</returns>
    public async Task RebootInstances(string ec2InstanceId)
    {
        var request = new RebootInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        var response = await _amazonEC2.RebootInstancesAsync(request);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine("Instances successfully rebooted.");
        }
```

```
        else
        {
            Console.WriteLine("Could not reboot one or more instances.");
        }
    }

    /// <summary>
    /// Release an Elastic IP address.
    /// </summary>
    /// <param name="allocationId">The allocation Id of the Elastic IP address.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> ReleaseAddress(string allocationId)
    {
        var request = new ReleaseAddressRequest
        {
            AllocationId = allocationId
        };

        var response = await _amazonEC2.ReleaseAddressAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Create and run an EC2 instance.
    /// </summary>
    /// <param name="ImageId">The image Id of the image used as a basis for the
    /// EC2 instance.</param>
    /// <param name="instanceType">The instance type of the EC2 instance to
 create.</param>
    /// <param name="keyName">The name of the key pair to associate with the
    /// instance.</param>
    /// <param name="groupId">The Id of the Amazon EC2 security group that will
 be
    /// allowed to interact with the new EC2 instance.</param>
    /// <returns>The instance Id of the new EC2 instance.</returns>
    public async Task<string> RunInstances(string imageId, string instanceType,
 string keyName, string groupId)
    {
        var request = new RunInstancesRequest
        {
            ImageId = imageId,
            InstanceType = instanceType,
            KeyName = keyName,
```

```
            MinCount = 1,
            MaxCount = 1,
            SecurityGroupIds = new List<string> { groupId }
        };
        var response = await _amazonEC2.RunInstancesAsync(request);
        return response.Reservation.Instances[0].InstanceId;
    }


    /// <summary>
    /// Start an EC2 instance.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the Amazon EC2 instance
    /// to start.</param>
    /// <returns>Async task.</returns>
    public async Task StartInstances(string ec2InstanceId)
    {
        var request = new StartInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        var response = await _amazonEC2.StartInstancesAsync(request);

        if (response.StartingInstances.Count > 0)
        {
            var instances = response.StartingInstances;
            instances.ForEach(i =>
            {
                Console.WriteLine($"Successfully started the EC2 instance with
instance ID: {i.InstanceId}.");
            });
        }
    }

    /// <summary>
    /// Stop an EC2 instance.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the EC2 instance to
    /// stop.</param>
    /// <returns>Async task.</returns>
    public async Task StopInstances(string ec2InstanceId)
    {
        // In addition to the list of instance Ids, the
```

```
        // request can also include the following properties:
        //     Force      When true, forces the instances to
        //                stop but you must check the integrity
        //                of the file system. Not recommended on
        //                Windows instances.
        //     Hibernate  When true, hibernates the instance if the
        //                instance was enabled for hibernation when
        //                it was launched.
        var request = new StopInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        var response = await _amazonEC2.StopInstancesAsync(request);

        if (response.StoppingInstances.Count > 0)
        {
            var instances = response.StoppingInstances;
            instances.ForEach(i =>
            {
                Console.WriteLine($"Successfully stopped the EC2 Instance " +
                                  $"with InstanceID: {i.InstanceId}.");
            });
        }
    }

    /// <summary>
    /// Terminate an EC2 instance.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the EC2 instance
    /// to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task<List<InstanceStateChange>> TerminateInstances(string
ec2InstanceId)
    {
        var request = new TerminateInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId }
        };

        var response = await _amazonEC2.TerminateInstancesAsync(request);
        return response.TerminatingInstances;
    }
```

```
    /// <summary>
    /// Wait until an EC2 instance is in a specified state.
    /// </summary>
    /// <param name="instanceId">The instance Id.</param>
    /// <param name="stateName">The state to wait for.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitForInstanceState(string instanceId,
 InstanceStateName stateName)
    {
        var request = new DescribeInstancesRequest
        {
            InstanceIds = new List<string> { instanceId }
        };

        // Wait until the instance is running.
        var hasState = false;
        do
        {
            // Wait 5 seconds.
            Thread.Sleep(5000);

            // Check for the desired state.
            var response = await _amazonEC2.DescribeInstancesAsync(request);
            var instance = response.Reservations[0].Instances[0];
            hasState = instance.State.Name == stateName;
            Console.Write(". ");
        } while (!hasState);

        return hasState;
    }

}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for .NET API 參考》中的下列主題。
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup
  - DeleteKeyPair

- DeleteSecurityGroup
- DescribeImages
- DescribeInstanceTypes
- DescribeInstances
- DescribeKeyPairs
- DescribeSecurityGroups
- DisassociateAddress
- ReleaseAddress
- RunInstances
- StartInstances
- StopInstances
- TerminateInstances
- UnmonitorInstances

Bash

AWS CLI 與 Bash 腳本

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

在命令提示中執行互動式案例。

```
###############################################################################
# function get_started_with_ec2_instances
#
# Runs an interactive scenario that shows how to get started using EC2 instances.
#
#     "EC2 access" permissions are needed to run this code.
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
```

```
################################################################################
function get_started_with_ec2_instances() {
  # Requires version 4 for mapfile.
  local required_version=4.0

  # Get the current Bash version
  # Check if BASH_VERSION is set
  local current_version
  if [[ -n "$BASH_VERSION" ]]; then
    # Convert BASH_VERSION to a number for comparison
    current_version=$BASH_VERSION
  else
    # Get the current Bash version using the bash command
    current_version=$(bash --version | head -n 1 | awk '{ print $4 }')
  fi

  # Convert version strings to numbers for comparison
  local required_version_num current_version_num
  required_version_num=$(echo "$required_version" | awk -F. '{ print ($1 * 10000)
 + ($2 * 100) + $3 }')
  current_version_num=$(echo "$current_version" | awk -F. '{ print ($1 * 10000) +
 ($2 * 100) + $3 }')

  # Compare versions
  if ((current_version_num < required_version_num)); then
    echo "Error: This script requires Bash version $required_version or higher."
    echo "Your current Bash version is number is $current_version."
    exit 1
  fi

  {
    if [ "$EC2_OPERATIONS_SOURCED" != "True" ]; then

      source ./ec2_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the Amazon Elastic Compute Cloud (Amazon EC2) get started with
 instances demo."
  echo_repeat "*" 88
  echo

  echo "Let's create an RSA key pair that you can be use to securely connect to "
```

```
  echo "your EC2 instance."

  echo -n "Enter a unique name for your key: "
  get_input
  local key_name
  key_name=$get_input_result

  local temp_dir
  temp_dir=$(mktemp -d)
  local key_file_name="$temp_dir/${key_name}.pem"

  if ec2_create_keypair -n "${key_name}" -f "${key_file_name}"; then
    echo "Created a key pair $key_name and saved the private key to
$key_file_name"
    echo
  else
    errecho "The key pair failed to create. This demo will exit."
    return 1
  fi

  chmod 400 "${key_file_name}"

  if yes_no_input "Do you want to list some of your key pairs? (y/n) "; then
    local keys_and_fingerprints
    keys_and_fingerprints="$(ec2_describe_key_pairs)" && {
      local image_name_and_id
      while IFS=$'\n' read -r image_name_and_id; do
        local entries
        IFS=$'\t' read -ra entries <<<"$image_name_and_id"
        echo "Found rsa key ${entries[0]} with fingerprint:"
        echo "     ${entries[1]}"
      done <<<"$keys_and_fingerprints"

    }
  fi

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "Let's create a security group to manage access to your instance."
  echo -n "Enter a unique name for your security group: "
  get_input
  local security_group_name
  security_group_name=$get_input_result
```

```
  local security_group_id
  security_group_id=$(ec2_create_security_group -n "$security_group_name" \
    -d "Security group for EC2 instance") || {
    errecho "The security failed to create. This demo will exit."
    clean_up "$key_name" "$key_file_name"
    return 1
  }

  echo "Security group created with ID $security_group_id"
  echo

  local public_ip
  public_ip=$(curl -s http://checkip.amazonaws.com)

  echo "Let's add a rule to allow SSH only from your current IP address."
  echo "Your public IP address is $public_ip"
  echo -n "press return to add this rule to your security group."
  get_input

  if ! ec2_authorize_security_group_ingress -g "$security_group_id" -i
"$public_ip" -p tcp -f 22 -t 22; then
    errecho "The security group rules failed to update. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  fi

  echo "Security group rules updated"

  local security_group_description
  security_group_description="$(ec2_describe_security_groups -g
"${security_group_id}")" || {
    errecho "Failed to describe security groups. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  }

  mapfile -t parameters <<<"$security_group_description"
  IFS=$'\t' read -ra entries <<<"${parameters[0]}"
  echo "Security group: ${entries[0]}"
  echo "    ID: ${entries[1]}"
  echo "    VPC: ${entries[2]}"
  echo "Inbound permissions:"
  IFS=$'\t' read -ra entries <<<"${parameters[1]}"
  echo "    IpProtocol: ${entries[0]}"
```

```
  echo "    FromPort: ${entries[1]}"
  echo "    ToPort: ${entries[2]}"
  echo "    CidrIp: ${parameters[2]}"

  local parameters
  parameters="$(ssm_get_parameters_by_path -p "/aws/service/ami-amazon-linux-
latest")" || {
    errecho "Failed to get parameters. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1

  }

  local image_ids=""
  mapfile -t parameters <<<"$parameters"
  for image_name_and_id in "${parameters[@]}"; do
    IFS=$'\t' read -ra values <<<"$image_name_and_id"
    if [[ "${values[0]}" == *"amzn2"* ]]; then
      image_ids+="${values[1]} "
    fi
  done

  local images
  images="$(ec2_describe_images -i "$image_ids")" || {
    errecho "Failed to describe images. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1

  }

  new_line_and_tab_to_list "$images"
  local images=("${list_result[@]}")

  # Get the size of the array
  local images_count=${#images[@]}

  if ((images_count == 0)); then
    errecho "No images found. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  fi

  echo_repeat "*" 88
  echo_repeat "*" 88
```

```
  echo "Let's create an instance from an Amazon Linux 2 AMI. Here are some
 options:"
  for ((i = 0; i < images_count; i += 3)); do
    echo "$(((i / 3) + 1)) - ${images[$i]}"
  done

  integer_input "Please enter the number of the AMI you want to use: " 1
 "$((images_count / 3))"
  local choice=$get_input_result
  choice=$(((choice - 1) * 3))

  echo "Great choice."
  echo

  local architecture=${images[$((choice + 1))]}
  local image_id=${images[$((choice + 2))]}
  echo "Here are some instance types that support the ${architecture}
 architecture of the image:"
  response="$(ec2_describe_instance_types -a "${architecture}" -t
 "*.micro,*.small")" || {
    errecho "Failed to describe instance types. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  }

  local instance_types
  mapfile -t instance_types <<<"$response"

  # Get the size of the array
  local instance_types_count=${#instance_types[@]}

  echo "Here are some options:"
  for ((i = 0; i < instance_types_count; i++)); do
    echo "$((i + 1)) - ${instance_types[$i]}"
  done

  integer_input "Which one do you want to use? " 1 "${#instance_types[@]}
"
  choice=$get_input_result
  local instance_type=${instance_types[$((choice - 1))]}
  echo "Another great choice."
  echo
```

```
  echo "Creating your instance and waiting for it to start..."
  local instance_id
  instance_id=$(ec2_run_instances -i "$image_id" -t "$instance_type" -k
"$key_name" -s "$security_group_id") || {
    errecho "Failed to run instance. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  }

  ec2_wait_for_instance_running -i "$instance_id"
  echo "Your instance is ready:"
  echo

  local instance_details
  instance_details="$(ec2_describe_instances -i "${instance_id}")"

  echo
  print_instance_details "${instance_details}"

  local public_ip
  public_ip=$(echo "${instance_details}" | awk '{print $6}')
  echo
  echo "You can use SSH to connect to your instance"
  echo "If the connection attempt times out, you might have to manually update
the SSH ingress rule"
  echo "for your IP address in the AWS Management Console."
  connect_to_instance "$key_file_name" "$public_ip"

  echo -n "Press Enter when you're ready to continue the demo: "
  get_input

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "Let's stop and start your instance to see what changes."
  echo "Stopping your instance and waiting until it's stopped..."
  ec2_stop_instances -i "$instance_id"
  ec2_wait_for_instance_stopped -i "$instance_id"

  echo "Your instance is stopped. Restarting..."

  ec2_start_instances -i "$instance_id"
  ec2_wait_for_instance_running -i "$instance_id"
```

```
  echo "Your instance is running again."
  local instance_details
  instance_details="$(ec2_describe_instances -i "${instance_id}")"

  print_instance_details "${instance_details}"

  public_ip=$(echo "${instance_details}" | awk '{print $6}')

  echo "Every time your instance is restarted, its public IP address changes"
  connect_to_instance "$key_file_name" "$public_ip"

  echo -n "Press Enter when you're ready to continue the demo: "
  get_input

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "You can allocate an Elastic IP address and associate it with your
instance"
  echo "to keep a consistent IP address even when your instance restarts."

  local result
  result=$(ec2_allocate_address -d vpc) || {
    errecho "Failed to allocate an address. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id"
    return 1
  }

  local elastic_ip allocation_id
  elastic_ip=$(echo "$result" | awk '{print $1}')
  allocation_id=$(echo "$result" | awk '{print $2}')

  echo "Allocated static Elastic IP address: $elastic_ip"

  local association_id
  association_id=$(ec2_associate_address -i "$instance_id" -a "$allocation_id")
|| {
    errecho "Failed to associate an address. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id"
"$allocation_id"
    return 1
  }

  echo "Associated your Elastic IP with your instance."
```

```
  echo "You can now use SSH to connect to your instance by using the Elastic IP."
  connect_to_instance "$key_file_name" "$elastic_ip"

  echo -n "Press Enter when you're ready to continue the demo: "
  get_input

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "Let's stop and start your instance to see what changes."
  echo "Stopping your instance and waiting until it's stopped..."
  ec2_stop_instances -i "$instance_id"
  ec2_wait_for_instance_stopped -i "$instance_id"

  echo "Your instance is stopped. Restarting..."

  ec2_start_instances -i "$instance_id"
  ec2_wait_for_instance_running -i "$instance_id"

  echo "Your instance is running again."
  local instance_details
  instance_details="$(ec2_describe_instances -i "${instance_id}")"

  print_instance_details "${instance_details}"

  echo "Because you have associated an Elastic IP with your instance, you can"
  echo "connect by using a consistent IP address after the instance restarts."
  connect_to_instance "$key_file_name" "$elastic_ip"

  echo -n "Press Enter when you're ready to continue the demo: "
  get_input

  echo_repeat "*" 88
  echo_repeat "*" 88

  if yes_no_input "Do you want to delete the resources created in this demo: (y/
n) "; then
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id" \
      "$allocation_id" "$association_id"
  else
    echo "The following resources were not deleted."
    echo "Key pair: $key_name"
    echo "Key file: $key_file_name"
    echo "Security group: $security_group_id"
```

```
    echo "Instance: $instance_id"
    echo "Elastic IP address: $elastic_ip"
  fi
}

################################################################################
# function clean_up
#
# This function cleans up the created resources.
#     $1 - The name of the ec2 key pair to delete.
#     $2 - The name of the key file to delete.
#     $3 - The ID of the security group to delete.
#     $4 - The ID of the instance to terminate.
#     $5 - The ID of the elastic IP address to release.
#     $6 - The ID of the elastic IP address to disassociate.
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
################################################################################
function clean_up() {
  local result=0
  local key_pair_name=$1
  local key_file_name=$2
  local security_group_id=$3
  local instance_id=$4
  local allocation_id=$5
  local association_id=$6

  if [ -n "$association_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_disassociate_address -a "$association_id"); then
      echo "Disassociated elastic IP address with ID $association_id"
    else
      errecho "The elastic IP address disassociation failed."
      result=1
    fi
  fi

  if [ -n "$allocation_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_release_address -a "$allocation_id"); then
      echo "Released elastic IP address with ID $allocation_id"
    else
```

```
      errecho "The elastic IP address release failed."
      result=1
    fi
  fi

  if [ -n "$instance_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_terminate_instances -i "$instance_id"); then
      echo "Started terminating instance with ID $instance_id"

      ec2_wait_for_instance_terminated -i "$instance_id"
    else
      errecho "The instance terminate failed."
      result=1
    fi
  fi

  if [ -n "$security_group_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_delete_security_group -i "$security_group_id"); then
      echo "Deleted security group with ID $security_group_id"
    else
      errecho "The security group delete failed."
      result=1
    fi
  fi

  if [ -n "$key_pair_name" ]; then
    # bashsupport disable=BP2002
    if (ec2_delete_keypair -n "$key_pair_name"); then
      echo "Deleted key pair named $key_pair_name"
    else
      errecho "The key pair delete failed."
      result=1
    fi
  fi

  if [ -n "$key_file_name" ]; then
    rm -f "$key_file_name"
  fi

  return $result
}
```

```
###############################################################################
# function ssm_get_parameters_by_path
#
# This function retrieves one or more parameters from the AWS Systems Manager
 Parameter Store
# by specifying a parameter path.
#
# Parameters:
#       -p parameter_path - The path of the parameter(s) to retrieve.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ssm_get_parameters_by_path() {
  local parameter_path response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ssm_get_parameters_by_path"
    echo "Retrieves one or more parameters from the AWS Systems Manager Parameter
 Store by specifying a parameter path."
    echo "  -p parameter_path - The path of the parameter(s) to retrieve."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "p:h" option; do
    case "${option}" in
      p) parameter_path="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1
```

```
  if [[ -z "$parameter_path" ]]; then
    errecho "ERROR: You must provide a parameter path with the -p parameter."
    usage
    return 1
  fi

  response=$(aws ssm get-parameters-by-path \
    --path "$parameter_path" \
    --query "Parameters[*].[Name, Value]" \
    --output text) || {
    aws_cli_error_log $?
    errecho "ERROR: AWS reports get-parameters-by-path operation failed.
$response"
    return 1
  }

  echo "$response"

  return 0
}

###############################################################################
# function print_instance_details
#
# This function prints the details of an Amazon Elastic Compute Cloud (Amazon
 EC2) instance.
#
# Parameters:
#       instance_details - The instance details in the format "InstanceId ImageId
 InstanceType KeyName VpcId PublicIpAddress State.Name".
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function print_instance_details() {
  local instance_details="$1"

  if [[ -z "${instance_details}" ]]; then
    echo "Error: Missing required instance details argument."
    return 1
  fi

  local instance_id image_id instance_type key_name vpc_id public_ip state
```

```
  instance_id=$(echo "${instance_details}" | awk '{print $1}')
  image_id=$(echo "${instance_details}" | awk '{print $2}')
  instance_type=$(echo "${instance_details}" | awk '{print $3}')
  key_name=$(echo "${instance_details}" | awk '{print $4}')
  vpc_id=$(echo "${instance_details}" | awk '{print $5}')
  public_ip=$(echo "${instance_details}" | awk '{print $6}')
  state=$(echo "${instance_details}" | awk '{print $7}')

  echo "    ID: ${instance_id}"
  echo "    Image ID: ${image_id}"
  echo "    Instance type: ${instance_type}"
  echo "    Key name: ${key_name}"
  echo "    VPC ID: ${vpc_id}"
  echo "    Public IP: ${public_ip}"
  echo "    State: ${state}"

  return 0
}

###############################################################################
# function connect_to_instance
#
# This function displays the public IP address of an Amazon Elastic Compute Cloud
 (Amazon EC2) instance and prompts the user to connect to the instance via SSH.
#
# Parameters:
#       $1 - The name of the key file used to connect to the instance.
#       $2 - The public IP address of the instance.
#
# Returns:
#       None
###############################################################################
function connect_to_instance() {
  local key_file_name="$1"
  local public_ip="$2"

  # Validate the input parameters
  if [[ -z "$key_file_name" ]]; then
    echo "ERROR: You must provide a key file name as the first argument." >&2
    return 1
  fi

  if [[ -z "$public_ip" ]]; then
```

```
    echo "ERROR: You must provide a public IP address as the second argument."
 >&2
    return 1
  fi

  # Display the public IP address and connection command
  echo "To connect, run the following command:"
  echo "    ssh -i ${key_file_name} ec2-user@${public_ip}"

  # Prompt the user to connect to the instance
  if yes_no_input "Do you want to connect now? (y/n) "; then
    echo "After you have connected, you can return to this example by typing
 'exit'"
    ssh -i "${key_file_name}" ec2-user@"${public_ip}"
  fi
}

##############################################################################
# function get_input
#
# This function gets user input from the command line.
#
# Outputs:
#   User input to stdout.
#
# Returns:
#       0
##############################################################################
function get_input() {

  if [ -z "${mock_input+x}" ]; then
    read -r get_input_result
  else

    if [ "$mock_input_array_index" -lt ${#mock_input_array[@]} ]; then
      get_input_result="${mock_input_array[$mock_input_array_index]}"
      # bashsupport disable=BP2001
      # shellcheck disable=SC2206
      ((mock_input_array_index++))
      echo -n "$get_input_result"
    else
      echo "MOCK_INPUT_ARRAY has no more elements" 1>&2
      return 1
    fi
```

```
  fi

  return 0
}

################################################################################
# function yes_no_input
#
# This function requests a yes/no answer from the user, following to a prompt.
#
# Parameters:
#       $1 - The prompt.
#
# Returns:
#       0 - If yes.
#       1 - If no.
################################################################################
function yes_no_input() {
  if [ -z "$1" ]; then
    echo "Internal error yes_no_input"
    return 1
  fi

  local index=0
  local response="N"
  while [[ $index -lt 10 ]]; do
    index=$((index + 1))
    echo -n "$1"
    if ! get_input; then
      return 1
    fi
    response=$(echo "$get_input_result" | tr '[:upper:]' '[:lower:]')
    if [ "$response" = "y" ] || [ "$response" = "n" ]; then
      break
    else
      echo -e "\nPlease enter or 'y' or 'n'."
    fi
  done

  echo

  if [ "$response" = "y" ]; then
    return 0
  else
```

```
    return 1
  fi
}

################################################################################
# function integer_input
#
# This function prompts the user to enter an integer within a specified range
# and validates the input.
#
# Parameters:
#       $1 - The prompt message to display to the user.
#       $2 - The minimum value of the accepted range.
#       $3 - The maximum value of the accepted range.
#
# Returns:
#       The valid integer input from the user.
#       If the input is invalid or out of range, the function will continue
#       prompting the user until a valid input is provided.
################################################################################
function integer_input() {
  local prompt="$1"
  local min_value="$2"
  local max_value="$3"
  local input=""

  while true; do
    # Display the prompt message and wait for user input
    echo -n "$prompt"

    if ! get_input; then
      return 1
    fi

    input="$get_input_result"

    # Check if the input is a valid integer
    if [[ "$input" =~ ^-?[0-9]+$ ]]; then
      # Check if the input is within the specified range
      if ((input >= min_value && input <= max_value)); then
        return 0
      else
        echo "Error: Input, $input, must be between $min_value and $max_value."
      fi
```

```
    else
      echo "Error: Invalid input- $input. Please enter an integer."
    fi
  done
}
################################################################################
# function new_line_and_tab_to_list
#
# This function takes a string input containing newlines and tabs, and
# converts it into a list (array) of elements.
#
# Parameters:
#       $1 - The input string containing newlines and tabs.
#
# Returns:
#       The resulting list (array) is stored in the global variable
#       'list_result'.
################################################################################
function new_line_and_tab_to_list() {
  local input=$1
  export list_result

  list_result=()
  mapfile -t lines <<<"$input"
  local line
  for line in "${lines[@]}"; do
    IFS=$'\t' read -ra parameters <<<"$line"
    list_result+=("${parameters[@]}")
  done
}

################################################################################
# function echo_repeat
#
# This function prints a string 'n' times to stdout.
#
# Parameters:
#       $1 - The string.
#       $2 - Number of times to print the string.
#
# Outputs:
#   String 'n' times to stdout.
#
# Returns:
```

```
#       0
################################################################################
function echo_repeat() {
  local end=$2
  for ((i = 0; i < end; i++)); do
    echo -n "$1"
  done
  echo
}
```

此案例中使用的 DynamoDB 函數。

```
################################################################################
# function ec2_create_keypair
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or
 2048-bit RSA key pair
# and writes it to a file.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#       -f file_path - File to store the key pair.
#
# And:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function ec2_create_keypair() {
  local key_pair_name file_path response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_create_keypair"
    echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or 2048-
bit RSA key pair"
    echo " and writes it to a file."
    echo "  -n key_pair_name - A key pair name."
    echo "  -f file_path - File to store the key pair."
    echo ""
  }
```

```
  # Retrieve the calling parameters.
  while getopts "n:f:h" option; do
    case "${option}" in
      n) key_pair_name="${OPTARG}" ;;
      f) file_path="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

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

  if [[ -z "$file_path" ]]; then
    errecho "ERROR: You must provide a file path with the -f parameter."
    usage
    return 1
  fi

  response=$(aws ec2 create-key-pair \
    --key-name "$key_pair_name" \
    --query 'KeyMaterial' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  }

  if [[ -n "$file_path" ]]; then
    echo "$response" >"$file_path"
  fi

  return 0
```

```
}

###############################################################################
# function ec2_describe_key_pairs
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 key pairs.
#
# Parameters:
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_key_pairs() {
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_key_pairs"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
 pairs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response
```

```
  response=$(aws ec2 describe-key-pairs \
    --query 'KeyPairs[*].[KeyName, KeyFingerprint]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-key-pairs operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

###############################################################################
# function ec2_create_security_group
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) security
 group.
#
# Parameters:
#       -n security_group_name - The name of the security group.
#       -d security_group_description - The description of the security group.
#
# Returns:
#       The ID of the created security group, or an error message if the
 operation fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_create_security_group() {
  local security_group_name security_group_description response

  # Function to display usage information
  function usage() {
    echo "function ec2_create_security_group"
    echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -n security_group_name - The name of the security group."
    echo "  -d security_group_description - The description of the security
 group."
    echo ""
  }
```

```
 # Parse the command-line arguments
 while getopts "n:d:h" option; do
   case "${option}" in
     n) security_group_name="${OPTARG}" ;;
     d) security_group_description="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 # Validate the input parameters
 if [[ -z "$security_group_name" ]]; then
   errecho "ERROR: You must provide a security group name with the -n
parameter."
   return 1
 fi

 if [[ -z "$security_group_description" ]]; then
   errecho "ERROR: You must provide a security group description with the -d
parameter."
   return 1
 fi

 # Create the security group
 response=$(aws ec2 create-security-group \
   --group-name "$security_group_name" \
   --description "$security_group_description" \
   --query "GroupId" \
   --output text) || {
   aws_cli_error_log ${?}
   errecho "ERROR: AWS reports create-security-group operation failed."
   errecho "$response"
   return 1
 }
```

```
  echo "$response"
  return 0
}

###############################################################################
# function ec2_describe_security_groups
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 security groups.
#
# Parameters:
#       -g security_group_id - The ID of the security group to describe
 (optional).
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_security_groups() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_security_groups"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 security groups."
    echo "  -g security_group_id - The ID of the security group to describe
 (optional)."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "g:h" option; do
    case "${option}" in
      g) security_group_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
```

```
        ;;
    esac
  done
  export OPTIND=1

  local query="SecurityGroups[*].[GroupName, GroupId, VpcId, IpPermissions[*].
[IpProtocol, FromPort, ToPort, IpRanges[*].CidrIp]]"

  if [[ -n "$security_group_id" ]]; then
    response=$(aws ec2 describe-security-groups --group-ids "$security_group_id"
 --query "${query}" --output text)
  else
    response=$(aws ec2 describe-security-groups --query "${query}" --output text)
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports describe-security-groups operation failed.
$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function ec2_authorize_security_group_ingress
#
# This function authorizes an ingress rule for an Amazon Elastic Compute Cloud
 (Amazon EC2) security group.
#
# Parameters:
#       -g security_group_id - The ID of the security group.
#       -i ip_address - The IP address or CIDR block to authorize.
#       -p protocol - The protocol to authorize (e.g., tcp, udp, icmp).
#       -f from_port - The start of the port range to authorize.
#       -t to_port - The end of the port range to authorize.
#
# And:
#       0 - If successful.
```

```
#       1 - If it fails.
###############################################################################
function ec2_authorize_security_group_ingress() {
  local security_group_id ip_address protocol from_port to_port response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_authorize_security_group_ingress"
    echo "Authorizes an ingress rule for an Amazon Elastic Compute Cloud (Amazon
 EC2) security group."
    echo "  -g security_group_id - The ID of the security group."
    echo "  -i ip_address - The IP address or CIDR block to authorize."
    echo "  -p protocol - The protocol to authorize (e.g., tcp, udp, icmp)."
    echo "  -f from_port - The start of the port range to authorize."
    echo "  -t to_port - The end of the port range to authorize."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "g:i:p:f:t:h" option; do
    case "${option}" in
      g) security_group_id="${OPTARG}" ;;
      i) ip_address="${OPTARG}" ;;
      p) protocol="${OPTARG}" ;;
      f) from_port="${OPTARG}" ;;
      t) to_port="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -g parameter."
    usage
    return 1
```

```
  fi

  if [[ -z "$ip_address" ]]; then
    errecho "ERROR: You must provide an IP address or CIDR block with the -i
 parameter."
    usage
    return 1
  fi

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

  if [[ -z "$from_port" ]]; then
    errecho "ERROR: You must provide a start port with the -f parameter."
    usage
    return 1
  fi

  if [[ -z "$to_port" ]]; then
    errecho "ERROR: You must provide an end port with the -t parameter."
    usage
    return 1
  fi

  response=$(aws ec2 authorize-security-group-ingress \
    --group-id "$security_group_id" \
    --cidr "${ip_address}/32" \
    --protocol "$protocol" \
    --port "$from_port-$to_port" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports authorize-security-group-ingress operation
 failed.$response"
    return 1
  }

  return 0
}

################################################################################
# function ec2_describe_images
```

```
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 images.
#
# Parameters:
#       -i image_ids - A space-separated  list of image IDs (optional).
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_images() {
  local image_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_images"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 images."
    echo "  -i image_ids - A space-separated list of image IDs (optional)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) image_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local aws_cli_args=()
```

```
  if [[ -n "$image_ids" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--image-ids" $image_ids)
  fi

  response=$(aws ec2 describe-images \
    "${aws_cli_args[@]}" \
    --query 'Images[*].[Description,Architecture,ImageId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-images operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

################################################################################
# ec2_describe_instance_types
#
# This function describes EC2 instance types filtered by processor architecture
# and optionally by instance type. It takes the following arguments:
#
# -a, --architecture ARCHITECTURE  Specify the processor architecture (e.g.,
 x86_64)
# -t, --type INSTANCE_TYPE         Comma-separated list of instance types (e.g.,
 t2.micro)
# -h, --help                       Show the usage help
#
# The function prints the instance type and supported architecture for each
# matching instance type.
################################################################################
function ec2_describe_instance_types() {
  local architecture=""
  local instance_types=""

  # bashsupport disable=BP5008
  function usage() {
    echo "Usage: ec2_describe_instance_types [-a|--architecture ARCHITECTURE] [-
t|--type INSTANCE_TYPE] [-h|--help]"
```

```
    echo "  -a, --architecture ARCHITECTURE  Specify the processor architecture
(e.g., x86_64)"
    echo "  -t, --type INSTANCE_TYPE         Comma-separated list of instance
types (e.g., t2.micro)"
    echo "  -h, --help                       Show this help message"
  }

  while [[ $# -gt 0 ]]; do
    case "$1" in
      -a | --architecture)
        architecture="$2"
        shift 2
        ;;
      -t | --type)
        instance_types="$2"
        shift 2
        ;;
      -h | --help)
        usage
        return 0
        ;;
      *)
        echo "Unknown argument: $1"
        return 1
        ;;
    esac
  done

  if [[ -z "$architecture" ]]; then
    errecho "Error: Architecture not specified."
    usage
    return 1
  fi

  if [[ -z "$instance_types" ]]; then
    errecho "Error: Instance type not specified."
    usage
    return 1
  fi

  local tmp_json_file="temp_ec2.json"
  echo -n '[
    {
      "Name": "processor-info.supported-architecture",
```

```
    "Values": [' >"$tmp_json_file"

  local items
  IFS=',' read -ra items <<<"$architecture"
  local array_size
  array_size=${#items[@]}
  for i in $(seq 0 $((array_size - 1))); do
    echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
      echo -n ',' >>"$tmp_json_file"
    fi
  done
  echo -n ']},
    {
    "Name": "instance-type",
      "Values": [' >>"$tmp_json_file"
  IFS=',' read -ra items <<<"$instance_types"
  local array_size
  array_size=${#items[@]}
  for i in $(seq 0 $((array_size - 1))); do
    echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
      echo -n ',' >>"$tmp_json_file"
    fi
  done

  echo -n ']}]' >>"$tmp_json_file"

  local response
  response=$(aws ec2 describe-instance-types --filters file://"$tmp_json_file" \
    --query 'InstanceTypes[*].[InstanceType]' --output text)

  local error_code=$?

  rm "$tmp_json_file"

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    echo "ERROR: AWS reports describe-instance-types operation failed."
    return 1
  fi

  echo "$response"
  return 0
```

```
}

###############################################################################
# function ec2_run_instances
#
# This function launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i image_id - The ID of the Amazon Machine Image (AMI) to use.
#       -t instance_type - The instance type to use (e.g., t2.micro).
#       -k key_pair_name - The name of the key pair to use.
#       -s security_group_id - The ID of the security group to use.
#       -c count - The number of instances to launch (default: 1).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_run_instances() {
  local image_id instance_type key_pair_name security_group_id count response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_run_instances"
    echo "Launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i image_id - The ID of the Amazon Machine Image (AMI) to use."
    echo "  -t instance_type - The instance type to use (e.g., t2.micro)."
    echo "  -k key_pair_name - The name of the key pair to use."
    echo "  -s security_group_id - The ID of the security group to use."
    echo "  -c count - The number of instances to launch (default: 1)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:t:k:s:c:h" option; do
    case "${option}" in
      i) image_id="${OPTARG}" ;;
      t) instance_type="${OPTARG}" ;;
      k) key_pair_name="${OPTARG}" ;;
```

```
      s) security_group_id="${OPTARG}" ;;
      c) count="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$image_id" ]]; then
    errecho "ERROR: You must provide an Amazon Machine Image (AMI) ID with the -i
parameter."
    usage
    return 1
  fi

  if [[ -z "$instance_type" ]]; then
    errecho "ERROR: You must provide an instance type with the -t parameter."
    usage
    return 1
  fi

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -k parameter."
    usage
    return 1
  fi

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -s parameter."
    usage
    return 1
  fi

  if [[ -z "$count" ]]; then
    count=1
  fi
```

```
  response=$(aws ec2 run-instances \
    --image-id "$image_id" \
    --instance-type "$instance_type" \
    --key-name "$key_pair_name" \
    --security-group-ids "$security_group_id" \
    --count "$count" \
    --query 'Instances[*].[InstanceId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports run-instances operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

###############################################################################
# function ec2_describe_instances
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID of the instance to describe (optional).
#       -q query - The query to filter the response (optional).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_instances() {
  local instance_id query response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_instances"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_id - The ID of the instance to describe (optional)."
    echo "  -q query - The query to filter the response (optional)."
```

```
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:q:h" option; do
    case "${option}" in
      i) instance_id="${OPTARG}" ;;
      q) query="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$instance_id" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--instance-ids" $instance_id)
  fi

  local query_arg=""
  if [[ -n "$query" ]]; then
    query_arg="--query '$query'"
  else
    query_arg="--query Reservations[*].Instances[*].
[InstanceId,ImageId,InstanceType,KeyName,VpcId,PublicIpAddress,State.Name]"
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 describe-instances \
    "${aws_cli_args[@]}" \
    $query_arg \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-instances operation failed.$response"
```

```
    return 1
  }

  echo "$response"

  return 0
}

###############################################################################
# function ec2_stop_instances
#
# This function stops one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to stop (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_stop_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_stop_instances"
    echo "Stops one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to stop (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
```

```
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 stop-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports stop-instances operation failed with $response."
    return 1
  }

  return 0
}

###############################################################################
# function ec2_start_instances
#
# This function starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to start (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_start_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.
```

```
  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_start_instances"
    echo "Starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to start (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 start-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports start-instances operation failed with $response."
    return 1
  }

  return 0
}
```

```
###############################################################################
# function ec2_allocate_address
#
# This function allocates an Elastic IP address for use with Amazon Elastic
 Compute Cloud (Amazon EC2) instances in a specific AWS Region.
#
# Parameters:
#       -d domain - The domain for the Elastic IP address (either 'vpc' or
 'standard').
#
# Returns:
#       The allocated Elastic IP address, or an error message if the operation
 fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_allocate_address() {
  local domain response

  # Function to display usage information
  function usage() {
    echo "function ec2_allocate_address"
    echo "Allocates an Elastic IP address for use with Amazon Elastic Compute
 Cloud (Amazon EC2) instances in a specific AWS Region."
    echo "  -d domain - The domain for the Elastic IP address (either 'vpc' or
 'standard')."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "d:h" option; do
    case "${option}" in
      d) domain="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
```

```
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$domain" ]]; then
    errecho "ERROR: You must provide a domain with the -d parameter (either 'vpc'
 or 'standard')."
    return 1
  fi

  if [[ "$domain" != "vpc" && "$domain" != "standard" ]]; then
    errecho "ERROR: Invalid domain value. Must be either 'vpc' or 'standard'."
    return 1
  fi

  # Allocate the Elastic IP address
  response=$(aws ec2 allocate-address \
    --domain "$domain" \
    --query "[PublicIp,AllocationId]" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports allocate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}

################################################################################
# function ec2_associate_address
#
# This function associates an Elastic IP address with an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to
 associate.
#       -i instance_id - The ID of the EC2 instance to associate the Elastic IP
 address with.
#
```

```
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_associate_address() {
  local allocation_id instance_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_associate_address"
    echo "Associates an Elastic IP address with an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 associate."
    echo "  -i instance_id - The ID of the EC2 instance to associate the Elastic
 IP address with."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:i:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
      i) instance_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
  fi
```

```
  if [[ -z "$instance_id" ]]; then
    errecho "ERROR: You must provide an instance ID with the -i parameter."
    return 1
  fi

  # Associate the Elastic IP address
  response=$(aws ec2 associate-address \
    --allocation-id "$allocation_id" \
    --instance-id "$instance_id" \
    --query "AssociationId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports associate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}

################################################################################
# function ec2_disassociate_address
#
# This function disassociates an Elastic IP address from an Amazon Elastic
 Compute Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a association_id - The association ID that represents the association of
 the Elastic IP address with an instance.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#
################################################################################
function ec2_disassociate_address() {
  local association_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_disassociate_address"
    echo "Disassociates an Elastic IP address from an Amazon Elastic Compute
 Cloud (Amazon EC2) instance."
```

```
    echo "  -a association_id - The association ID that represents the
 association of the Elastic IP address with an instance."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
      a) association_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$association_id" ]]; then
    errecho "ERROR: You must provide an association ID with the -a parameter."
    return 1
  fi

  response=$(aws ec2 disassociate-address \
    --association-id "$association_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports disassociate-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}

################################################################################
# function ec2_release_address
#
# This function releases an Elastic IP address from an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
```

```
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to
 release.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_release_address() {
  local allocation_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_release_address"
    echo "Releases an Elastic IP address from an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 release."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
```

```
  fi

  response=$(aws ec2 release-address \
    --allocation-id "$allocation_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports release-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}

###############################################################################
# function ec2_terminate_instances
#
# This function terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances using the AWS CLI.
#
# Parameters:
#       -i instance_ids - A space-separated list of instance IDs.
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_terminate_instances() {
  local instance_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_terminate_instances"
    echo "Terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_ids - A space-separated list of instance IDs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
```

```
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Check if instance ID is provided
  if [[ -z "${instance_ids}" ]]; then
    echo "Error: Missing required instance IDs parameter."
    usage
    return 1
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 terminate-instances \
    "--instance-ids" $instance_ids \
    --query 'TerminatingInstances[*].[InstanceId,CurrentState.Name]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports terminate-instances operation failed.$response"
    return 1
  }

  return 0
}

################################################################################
# function ec2_delete_security_group
#
# This function deletes an Amazon Elastic Compute Cloud (Amazon EC2) security
 group.
#
# Parameters:
#       -i security_group_id - The ID of the security group to delete.
#
# And:
```

```
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_delete_security_group() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_security_group"
    echo "Deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -i security_group_id - The ID of the security group to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) security_group_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -i parameter."
    usage
    return 1
  fi

  response=$(aws ec2 delete-security-group --group-id "$security_group_id" --
output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-security-group operation failed.$response"
    return 1
  }
```

```
  return 0
}

################################################################################
# function ec2_delete_keypair
#
# This function deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#
# And:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function ec2_delete_keypair() {
  local key_pair_name response

  local option OPTARG # Required to use getopts command in a function.
  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_keypair"
    echo "Deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair."
    echo "  -n key_pair_name - A key pair name."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) key_pair_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1
```

```
  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -n parameter."
    usage
    return 1
  fi

  response=$(aws ec2 delete-key-pair \
    --key-name "$key_pair_name") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-key-pair operation failed.$response"
    return 1
  }

  return 0
}
```

此案例中使用的公用程式函數。

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
```

```
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DescribeImages
  - DescribeInstanceTypes
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups
  - DisassociateAddress
  - ReleaseAddress
  - RunInstances

- StartInstances
- StopInstances
- TerminateInstances
- UnmonitorInstances

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
/**
 * 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 example performs the following tasks:
 *
 * 1. Creates an RSA key pair and saves the private key data as a .pem file.
 * 2. Lists key pairs.
 * 3. Creates a security group for the default VPC.
 * 4. Displays security group information.
 * 5. Gets a list of Amazon Linux 2 AMIs and selects one.
 * 6. Gets more information about the image.
 * 7. Gets a list of instance types that are compatible with the selected AMI's
 * architecture.
 * 8. Creates an instance with the key pair, security group, AMI, and an
 * instance type.
 * 9. Displays information about the instance.
 * 10. Stops the instance and waits for it to stop.
 * 11. Starts the instance and waits for it to start.
 * 12. Allocates an Elastic IP address and associates it with the instance.
```

```
 * 13. Displays SSH connection info for the instance.
 * 14. Disassociates and deletes the Elastic IP address.
 * 15. Terminates the instance and waits for it to terminate.
 * 16. Deletes the security group.
 * 17. Deletes the key pair.
 */
public class EC2Scenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws InterruptedException {

        final String usage = """

                Usage:
                   <keyName> <fileName> <groupName> <groupDesc> <vpcId>

                Where:
                   keyName -  A key pair name (for example, TestKeyPair).\s
                   fileName -  A file name where the key information is written
 to.\s
                   groupName - The name of the security group.\s
                   groupDesc - The description of the security group.\s
                   vpcId - A VPC Id value. You can get this value from the AWS
 Management Console.\s
                   myIpAddress - The IP address of your development machine.\s

                """;

        if (args.length != 6) {
            System.out.println(usage);
            System.exit(1);
        }

        String keyName = args[0];
        String fileName = args[1];
        String groupName = args[2];
        String groupDesc = args[3];
        String vpcId = args[4];
        String myIpAddress = args[5];

        Region region = Region.US_WEST_2;
        Ec2Client ec2 = Ec2Client.builder()
                .region(region)
```

```
                .build();

        SsmClient ssmClient = SsmClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon EC2 example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("1. Create an RSA key pair and save the private key
material as a .pem file.");
        createKeyPair(ec2, keyName, fileName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. List key pairs.");
        describeKeys(ec2);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Create a security group.");
        String groupId = createSecurityGroup(ec2, groupName, groupDesc, vpcId,
myIpAddress);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Display security group info for the newly created
security group.");
        describeSecurityGroups(ec2, groupId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. Get a list of Amazon Linux 2 AMIs and selects one
with amzn2 in the name.");
        String instanceId = getParaValues(ssmClient);
        System.out.println("The instance Id is " + instanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. Get more information about an amzn2 image.");
        String amiValue = describeImage(ec2, instanceId);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("7. Get a list of instance types.");
        String instanceType = getInstanceTypes(ec2);
        System.out.println("The instance type is " + instanceType);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Create an instance.");
        String newInstanceId = runInstance(ec2, instanceType, keyName, groupName,
amiValue);
        System.out.println("The instance Id is " + newInstanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("9. Display information about the running instance.
");
        String ipAddress = describeEC2Instances(ec2, newInstanceId);
        System.out.println("You can SSH to the instance using this command:");
        System.out.println("ssh -i " + fileName + "ec2-user@" + ipAddress);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("10. Stop the instance and use a waiter.");
        stopInstance(ec2, newInstanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("11. Start the instance and use a waiter.");
        startInstance(ec2, newInstanceId);
        ipAddress = describeEC2Instances(ec2, newInstanceId);
        System.out.println("You can SSH to the instance using this command:");
        System.out.println("ssh -i " + fileName + "ec2-user@" + ipAddress);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("12. Allocate an Elastic IP address and associate it
with the instance.");
        String allocationId = allocateAddress(ec2);
        System.out.println("The allocation Id value is " + allocationId);
        String associationId = associateAddress(ec2, newInstanceId,
allocationId);
        System.out.println("The associate Id value is " + associationId);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("13. Describe the instance again.");
        ipAddress = describeEC2Instances(ec2, newInstanceId);
        System.out.println("You can SSH to the instance using this command:");
        System.out.println("ssh -i " + fileName + "ec2-user@" + ipAddress);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("14. Disassociate and release the Elastic IP
address.");
        disassociateAddress(ec2, associationId);
        releaseEC2Address(ec2, allocationId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("15. Terminate the instance and use a waiter.");
        terminateEC2(ec2, newInstanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("16. Delete the security group.");
        deleteEC2SecGroup(ec2, groupId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("17. Delete the key.");
        deleteKeys(ec2, keyName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("You successfully completed the Amazon EC2
scenario.");
        System.out.println(DASHES);
        ec2.close();
    }

    public static void deleteEC2SecGroup(Ec2Client ec2, String groupId) {
        try {
            DeleteSecurityGroupRequest request =
DeleteSecurityGroupRequest.builder()
                    .groupId(groupId)
                    .build();
```

```
            ec2.deleteSecurityGroup(request);
            System.out.println("Successfully deleted security group with Id " +
groupId);

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void terminateEC2(Ec2Client ec2, String instanceId) {
        try {
            Ec2Waiter ec2Waiter = Ec2Waiter.builder()
                    .overrideConfiguration(b -> b.maxAttempts(100))
                    .client(ec2)
                    .build();

            TerminateInstancesRequest ti = TerminateInstancesRequest.builder()
                    .instanceIds(instanceId)
                    .build();

            System.out.println("Use an Ec2Waiter to wait for the instance to
terminate. This will take a few minutes.");
            ec2.terminateInstances(ti);
            DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
                    .instanceIds(instanceId)
                    .build();

            WaiterResponse<DescribeInstancesResponse> waiterResponse = ec2Waiter
                    .waitUntilInstanceTerminated(instanceRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Successfully started instance " + instanceId);
            System.out.println(instanceId + " is terminated!");

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKeys(Ec2Client ec2, String keyPair) {
        try {
            DeleteKeyPairRequest request = DeleteKeyPairRequest.builder()
```

```
                    .keyName(keyPair)
                    .build();

            ec2.deleteKeyPair(request);
            System.out.println("Successfully deleted key pair named " + keyPair);

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void releaseEC2Address(Ec2Client ec2, String allocId) {
        try {
            ReleaseAddressRequest request = ReleaseAddressRequest.builder()
                    .allocationId(allocId)
                    .build();

            ec2.releaseAddress(request);
            System.out.println("Successfully released Elastic IP address " +
allocId);
        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void disassociateAddress(Ec2Client ec2, String associationId) {
        try {
            DisassociateAddressRequest addressRequest =
DisassociateAddressRequest.builder()
                    .associationId(associationId)
                    .build();

            ec2.disassociateAddress(addressRequest);
            System.out.println("You successfully disassociated the address!");

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

```
   public static String associateAddress(Ec2Client ec2, String instanceId,
String allocationId) {
       try {
           AssociateAddressRequest associateRequest =
AssociateAddressRequest.builder()
                   .instanceId(instanceId)
                   .allocationId(allocationId)
                   .build();

           AssociateAddressResponse associateResponse =
ec2.associateAddress(associateRequest);
           return associateResponse.associationId();

       } catch (Ec2Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
       return "";
   }

   public static String allocateAddress(Ec2Client ec2) {
       try {
           AllocateAddressRequest allocateRequest =
AllocateAddressRequest.builder()
                   .domain(DomainType.VPC)
                   .build();

           AllocateAddressResponse allocateResponse =
ec2.allocateAddress(allocateRequest);
           return allocateResponse.allocationId();

       } catch (Ec2Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
       return "";
   }

   public static void startInstance(Ec2Client ec2, String instanceId) {
       Ec2Waiter ec2Waiter = Ec2Waiter.builder()
               .overrideConfiguration(b -> b.maxAttempts(100))
               .client(ec2)
               .build();
```

```
        StartInstancesRequest request = StartInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        System.out.println("Use an Ec2Waiter to wait for the instance to run.
This will take a few minutes.");
        ec2.startInstances(request);
        DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        WaiterResponse<DescribeInstancesResponse> waiterResponse =
ec2Waiter.waitUntilInstanceRunning(instanceRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Successfully started instance " + instanceId);
    }

    public static void stopInstance(Ec2Client ec2, String instanceId) {
        Ec2Waiter ec2Waiter = Ec2Waiter.builder()
                .overrideConfiguration(b -> b.maxAttempts(100))
                .client(ec2)
                .build();
        StopInstancesRequest request = StopInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        System.out.println("Use an Ec2Waiter to wait for the instance to stop.
This will take a few minutes.");
        ec2.stopInstances(request);
        DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        WaiterResponse<DescribeInstancesResponse> waiterResponse =
ec2Waiter.waitUntilInstanceStopped(instanceRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Successfully stopped instance " + instanceId);
    }

    public static String describeEC2Instances(Ec2Client ec2, String
newInstanceId) {
        try {
```

```
            String pubAddress = "";
            boolean isRunning = false;
            DescribeInstancesRequest request = DescribeInstancesRequest.builder()
                    .instanceIds(newInstanceId)
                    .build();

            while (!isRunning) {
                DescribeInstancesResponse response =
ec2.describeInstances(request);
                String state =
response.reservations().get(0).instances().get(0).state().name().name();
                if (state.compareTo("RUNNING") == 0) {
                    System.out.println("Image id is " +
response.reservations().get(0).instances().get(0).imageId());
                    System.out.println(
                            "Instance type is " +
response.reservations().get(0).instances().get(0).instanceType());
                    System.out.println(
                            "Instance state is " +
response.reservations().get(0).instances().get(0).state().name());
                    pubAddress =
response.reservations().get(0).instances().get(0).publicIpAddress();
                    System.out.println("Instance address is " + pubAddress);
                    isRunning = true;
                }
            }
            return pubAddress;
        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    public static String runInstance(Ec2Client ec2, String instanceType, String
keyName, String groupName,
            String amiId) {
        try {
            RunInstancesRequest runRequest = RunInstancesRequest.builder()
                    .instanceType(instanceType)
                    .keyName(keyName)
                    .securityGroups(groupName)
                    .maxCount(1)
                    .minCount(1)
```

```
                    .imageId(amiId)
                    .build();

            System.out.println("Going to start an EC2 instance using a waiter");
            RunInstancesResponse response = ec2.runInstances(runRequest);
            String instanceIdVal = response.instances().get(0).instanceId();
            ec2.waiter().waitUntilInstanceRunning(r ->
r.instanceIds(instanceIdVal));
            System.out.println("Successfully started EC2 instance " +
instanceIdVal + " based on AMI " + amiId);
            return instanceIdVal;

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    // Get a list of instance types.
    public static String getInstanceTypes(Ec2Client ec2) {
        String instanceType;
        try {
            DescribeInstanceTypesRequest typesRequest =
DescribeInstanceTypesRequest.builder()
                .maxResults(10)
                .build();

            DescribeInstanceTypesResponse response =
ec2.describeInstanceTypes(typesRequest);
            List<InstanceTypeInfo> instanceTypes = response.instanceTypes();
            for (InstanceTypeInfo type : instanceTypes) {
                System.out.println("The memory information of this type is " +
type.memoryInfo().sizeInMiB());
                System.out.println("Network information is " +
type.networkInfo().toString());
                System.out.println("Instance type is " +
type.instanceType().toString());
                instanceType = type.instanceType().toString();
                if (instanceType.compareTo("t2.2xlarge")  == 0){
                    return instanceType;
                }
            }
```

```
        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    // Display the Description field that corresponds to the instance Id value.
    public static String describeImage(Ec2Client ec2, String instanceId) {
        try {
            DescribeImagesRequest imagesRequest = DescribeImagesRequest.builder()
                    .imageIds(instanceId)
                    .build();

            DescribeImagesResponse response = ec2.describeImages(imagesRequest);
            System.out.println("The description of the first image is " +
response.images().get(0).description());
            System.out.println("The name of the first image is " +
response.images().get(0).name());

            // Return the image Id value.
            return response.images().get(0).imageId();

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    // Get the Id value of an instance with amzn2 in the name.
    public static String getParaValues(SsmClient ssmClient) {
        try {
            GetParametersByPathRequest parameterRequest =
GetParametersByPathRequest.builder()
                    .path("/aws/service/ami-amazon-linux-latest")
                    .build();

            GetParametersByPathIterable responses =
ssmClient.getParametersByPathPaginator(parameterRequest);
            for
(software.amazon.awssdk.services.ssm.model.GetParametersByPathResponse
response : responses) {
                System.out.println("Test " + response.nextToken());
```

```
                List<Parameter> parameterList = response.parameters();
                for (Parameter para : parameterList) {
                    System.out.println("The name of the para is: " +
para.name());
                    System.out.println("The type of the para is: " +
para.type());
                    if (filterName(para.name())) {
                        return para.value();
                    }
                }
            }

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    // Return true if the name has amzn2 in it. For example:
    // /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-arm64-gp2
    private static boolean filterName(String name) {
        String[] parts = name.split("/");
        String myValue = parts[4];
        return myValue.contains("amzn2");
    }

    public static void describeSecurityGroups(Ec2Client ec2, String groupId) {
        try {
            DescribeSecurityGroupsRequest request =
DescribeSecurityGroupsRequest.builder()
                .groupIds(groupId)
                .build();

            // Use a paginator.
            DescribeSecurityGroupsIterable listGroups =
ec2.describeSecurityGroupsPaginator(request);
            listGroups.stream()
                .flatMap(r -> r.securityGroups().stream())
                .forEach(group -> System.out
                    .println(" Group id: " +group.groupId() + " group name = " +
group.groupName()));

        } catch (Ec2Exception e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static String createSecurityGroup(Ec2Client ec2, String groupName,
String groupDesc, String vpcId,
            String myIpAddress) {
        try {
            CreateSecurityGroupRequest createRequest =
CreateSecurityGroupRequest.builder()
                    .groupName(groupName)
                    .description(groupDesc)
                    .vpcId(vpcId)
                    .build();

            CreateSecurityGroupResponse resp =
ec2.createSecurityGroup(createRequest);
            IpRange ipRange = IpRange.builder()
                    .cidrIp(myIpAddress + "/0")
                    .build();

            IpPermission ipPerm = IpPermission.builder()
                    .ipProtocol("tcp")
                    .toPort(80)
                    .fromPort(80)
                    .ipRanges(ipRange)
                    .build();

            IpPermission ipPerm2 = IpPermission.builder()
                    .ipProtocol("tcp")
                    .toPort(22)
                    .fromPort(22)
                    .ipRanges(ipRange)
                    .build();

            AuthorizeSecurityGroupIngressRequest authRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                    .groupName(groupName)
                    .ipPermissions(ipPerm, ipPerm2)
                    .build();

            ec2.authorizeSecurityGroupIngress(authRequest);
```

```
            System.out.println("Successfully added ingress policy to security
group " + groupName);
            return resp.groupId();

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void describeKeys(Ec2Client ec2) {
        try {
            DescribeKeyPairsResponse response = ec2.describeKeyPairs();
            response.keyPairs().forEach(keyPair -> System.out.printf(
                    "Found key pair with name %s " +
                            "and fingerprint %s",
                    keyPair.keyName(),
                    keyPair.keyFingerprint()));

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void createKeyPair(Ec2Client ec2, String keyName, String
fileName) {
        try {
            CreateKeyPairRequest request = CreateKeyPairRequest.builder()
                    .keyName(keyName)
                    .build();

            CreateKeyPairResponse response = ec2.createKeyPair(request);
            String content = response.keyMaterial();
            BufferedWriter writer = new BufferedWriter(new FileWriter(fileName));
            writer.write(content);
            writer.close();
            System.out.println("Successfully created key pair named " + keyName);

        } catch (Ec2Exception | IOException 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》中的下列主題。
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DescribeImages
  - DescribeInstanceTypes
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups
  - DisassociateAddress
  - ReleaseAddress
  - RunInstances
  - StartInstances
  - StopInstances
  - TerminateInstances
  - UnmonitorInstances

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

在命令提示中執行互動式案例。

```
import { mkdtempSync, writeFileSync, rmSync } from "fs";
import { tmpdir } from "os";
import { join } from "path";
import { get } from "http";

import {
  AllocateAddressCommand,
  AssociateAddressCommand,
  AuthorizeSecurityGroupIngressCommand,
  CreateKeyPairCommand,
  CreateSecurityGroupCommand,
  DeleteKeyPairCommand,
  DeleteSecurityGroupCommand,
  DescribeInstancesCommand,
  DescribeKeyPairsCommand,
  DescribeSecurityGroupsCommand,
  DisassociateAddressCommand,
  EC2Client,
  paginateDescribeImages,
  paginateDescribeInstanceTypes,
  ReleaseAddressCommand,
  RunInstancesCommand,
  StartInstancesCommand,
  StopInstancesCommand,
  TerminateInstancesCommand,
  waitUntilInstanceStatusOk,
  waitUntilInstanceStopped,
  waitUntilInstanceTerminated,
} from "@aws-sdk/client-ec2";
import { paginateGetParametersByPath, SSMClient } from "@aws-sdk/client-ssm";

import { wrapText } from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { Prompter } from "@aws-doc-sdk-examples/lib/prompter.js";

const ec2Client = new EC2Client();
const ssmClient = new SSMClient();

const prompter = new Prompter();
const confirmMessage = "Continue?";
const tmpDirectory = mkdtempSync(join(tmpdir(), "ec2-scenario-tmp"));

const createKeyPair = async (keyPairName) => {
```

```
  // Create a key pair in Amazon EC2.
  const { KeyMaterial, KeyPairId } = await ec2Client.send(
    // A unique name for the key pair. Up to 255 ASCII characters.
    new CreateKeyPairCommand({ KeyName: keyPairName }),
  );

  // Save the private key in a temporary location.
  writeFileSync(`${tmpDirectory}/${keyPairName}.pem`, KeyMaterial, {
    mode: 0o400,
  });

  return KeyPairId;
};

const describeKeyPair = async (keyPairName) => {
  const command = new DescribeKeyPairsCommand({
    KeyNames: [keyPairName],
  });
  const { KeyPairs } = await ec2Client.send(command);
  return KeyPairs[0];
};

const createSecurityGroup = async (securityGroupName) => {
  const command = new CreateSecurityGroupCommand({
    GroupName: securityGroupName,
    Description: "A security group for the Amazon EC2 example.",
  });
  const { GroupId } = await ec2Client.send(command);
  return GroupId;
};

const allocateIpAddress = async () => {
  const command = new AllocateAddressCommand({});
  const { PublicIp, AllocationId } = await ec2Client.send(command);
  return { PublicIp, AllocationId };
};

const getLocalIpAddress = () => {
  return new Promise((res, rej) => {
    get("http://checkip.amazonaws.com", (response) => {
      let data = "";
      response.on("data", (chunk) => (data += chunk));
      response.on("end", () => res(data.trim()));
    }).on("error", (err) => {
```

```
      rej(err);
    });
  });
};

const authorizeSecurityGroupIngress = async (securityGroupId) => {
  const ipAddress = await getLocalIpAddress();
  const command = new AuthorizeSecurityGroupIngressCommand({
    GroupId: securityGroupId,
    IpPermissions: [
      {
        IpProtocol: "tcp",
        FromPort: 22,
        ToPort: 22,
        IpRanges: [{ CidrIp: `${ipAddress}/32` }],
      },
    ],
  });

  await ec2Client.send(command);
  return ipAddress;
};

const describeSecurityGroup = async (securityGroupName) => {
  const command = new DescribeSecurityGroupsCommand({
    GroupNames: [securityGroupName],
  });
  const { SecurityGroups } = await ec2Client.send(command);

  return SecurityGroups[0];
};

const getAmznLinux2AMIs = async () => {
  const AMIs = [];
  for await (const page of paginateGetParametersByPath(
    {
      client: ssmClient,
    },
    { Path: "/aws/service/ami-amazon-linux-latest" },
  )) {
    page.Parameters.forEach((param) => {
      if (param.Name.includes("amzn2")) {
        AMIs.push(param.Value);
      }
```

```
    });
  }

  const imageDetails = [];

  for await (const page of paginateDescribeImages(
    { client: ec2Client },
    { ImageIds: AMIs },
  )) {
    imageDetails.push(...(page.Images || []));
  }

  const choices = imageDetails.map((image, index) => ({
    name: `${image.ImageId} - ${image.Description}`,
    value: index,
  }));

  /**
   * @type {number}
   */
  const selectedIndex = await prompter.select({
    message: "Select an image.",
    choices,
  });

  return imageDetails[selectedIndex];
};

/**
 * @param {import('@aws-sdk/client-ec2').Image} imageDetails
 */
const getCompatibleInstanceTypes = async (imageDetails) => {
  const paginator = paginateDescribeInstanceTypes(
    { client: ec2Client, pageSize: 25 },
    {
      Filters: [
        {
          Name: "processor-info.supported-architecture",
          Values: [imageDetails.Architecture],
        },
        { Name: "instance-type", Values: ["*.micro", "*.small"] },
      ],
    },
  );
```

```
  const instanceTypes = [];

  for await (const page of paginator) {
    if (page.InstanceTypes.length) {
      instanceTypes.push(...(page.InstanceTypes || []));
    }
  }

  const choices = instanceTypes.map((type, index) => ({
    name: `${type.InstanceType} - Memory:${type.MemoryInfo.SizeInMiB}`,
    value: index,
  }));

  /**
   * @type {number}
   */
  const selectedIndex = await prompter.select({
    message: "Select an instance type.",
    choices,
  });
  return instanceTypes[selectedIndex];
};

const runInstance = async ({
  keyPairName,
  securityGroupId,
  imageId,
  instanceType,
}) => {
  const command = new RunInstancesCommand({
    KeyName: keyPairName,
    SecurityGroupIds: [securityGroupId],
    ImageId: imageId,
    InstanceType: instanceType,
    MinCount: 1,
    MaxCount: 1,
  });

  const { Instances } = await ec2Client.send(command);
  await waitUntilInstanceStatusOk(
    { client: ec2Client },
    { InstanceIds: [Instances[0].InstanceId] },
  );
```

```
  return Instances[0].InstanceId;
};

const describeInstance = async (instanceId) => {
  const command = new DescribeInstancesCommand({
    InstanceIds: [instanceId],
  });

  const { Reservations } = await ec2Client.send(command);
  return Reservations[0].Instances[0];
};

const displaySSHConnectionInfo = ({ publicIp, keyPairName }) => {
  return `ssh -i ${tmpDirectory}/${keyPairName}.pem ec2-user@${publicIp}`;
};

const stopInstance = async (instanceId) => {
  const command = new StopInstancesCommand({ InstanceIds: [instanceId] });
  await ec2Client.send(command);
  await waitUntilInstanceStopped(
    { client: ec2Client },
    { InstanceIds: [instanceId] },
  );
};

const startInstance = async (instanceId) => {
  const startCommand = new StartInstancesCommand({ InstanceIds: [instanceId] });
  await ec2Client.send(startCommand);
  await waitUntilInstanceStatusOk(
    { client: ec2Client },
    { InstanceIds: [instanceId] },
  );
  return await describeInstance(instanceId);
};

const associateAddress = async ({ allocationId, instanceId }) => {
  const command = new AssociateAddressCommand({
    AllocationId: allocationId,
    InstanceId: instanceId,
  });

  const { AssociationId } = await ec2Client.send(command);
  return AssociationId;
};
```

```
const disassociateAddress = async (associationId) => {
  const command = new DisassociateAddressCommand({
    AssociationId: associationId,
  });
  try {
    await ec2Client.send(command);
  } catch (err) {
    console.warn(
      `Failed to disassociated address with association id: ${associationId}`,
      err,
    );
  }
};

const releaseAddress = async (allocationId) => {
  const command = new ReleaseAddressCommand({
    AllocationId: allocationId,
  });

  try {
    await ec2Client.send(command);
    console.log(`Address with allocation ID ${allocationId} released.\n`);
  } catch (err) {
    console.log(
      `Failed to release address with allocation id: ${allocationId}.`,
      err,
    );
  }
};

const restartInstance = async (instanceId) => {
  console.log("Stopping instance.");
  await stopInstance(instanceId);
  console.log("Instance stopped.");
  console.log("Starting instance.");
  const { PublicIpAddress } = await startInstance(instanceId);
  return PublicIpAddress;
};

const terminateInstance = async (instanceId) => {
  const command = new TerminateInstancesCommand({
    InstanceIds: [instanceId],
  });
```

```
  try {
    await ec2Client.send(command);
    await waitUntilInstanceTerminated(
      { client: ec2Client },
      { InstanceIds: [instanceId] },
    );
    console.log(`Instance with ID ${instanceId} terminated.\n`);
  } catch (err) {
    console.warn(`Failed to terminate instance ${instanceId}.`, err);
  }
};

const deleteSecurityGroup = async (securityGroupId) => {
  const command = new DeleteSecurityGroupCommand({
    GroupId: securityGroupId,
  });

  try {
    await ec2Client.send(command);
    console.log(`Security group ${securityGroupId} deleted.\n`);
  } catch (err) {
    console.warn(`Failed to delete security group ${securityGroupId}.`, err);
  }
};

const deleteKeyPair = async (keyPairName) => {
  const command = new DeleteKeyPairCommand({
    KeyName: keyPairName,
  });

  try {
    await ec2Client.send(command);
    console.log(`Key pair ${keyPairName} deleted.\n`);
  } catch (err) {
    console.warn(`Failed to delete key pair ${keyPairName}.`, err);
  }
};

const deleteTemporaryDirectory = () => {
  try {
    rmSync(tmpDirectory, { recursive: true });
    console.log(`Temporary directory ${tmpDirectory} deleted.\n`);
  } catch (err) {
```

```
    console.warn(`Failed to delete temporary directory ${tmpDirectory}.`, err);
  }
};

export const main = async () => {
  const keyPairName = "ec2-scenario-key-pair";
  const securityGroupName = "ec2-scenario-security-group";

  let securityGroupId, ipAllocationId, publicIp, instanceId, associationId;

  console.log(wrapText("Welcome to the Amazon EC2 basic usage scenario."));

  try {
    // Prerequisites
    console.log(
      "Before you launch an instance, you'll need a few things:",
      "\n - A Key Pair",
      "\n - A Security Group",
      "\n - An IP Address",
      "\n - An AMI",
      "\n - A compatible instance type",
      "\n\n I'll go ahead and take care of the first three, but I'll need your
 help for the rest.",
    );

    await prompter.confirm({ message: confirmMessage });

    await createKeyPair(keyPairName);
    securityGroupId = await createSecurityGroup(securityGroupName);
    const { PublicIp, AllocationId } = await allocateIpAddress();
    ipAllocationId = AllocationId;
    publicIp = PublicIp;
    const ipAddress = await authorizeSecurityGroupIngress(securityGroupId);

    const { KeyName } = await describeKeyPair(keyPairName);
    const { GroupName } = await describeSecurityGroup(securityGroupName);
    console.log(`# created the key pair ${KeyName}.\n`);
    console.log(
      `# created the security group ${GroupName}`,
      `and allowed SSH access from ${ipAddress} (your IP).\n`,
    );
    console.log(`# allocated ${publicIp} to be used for your EC2 instance.\n`);

    await prompter.confirm({ message: confirmMessage });
```

```
   // Creating the instance
   console.log(wrapText("Create the instance."));
   console.log(
     "You get to choose which image you want. Select an amazon-linux-2 image
from the following:",
   );
   const imageDetails = await getAmznLinux2AMIs();
   const instanceTypeDetails = await getCompatibleInstanceTypes(imageDetails);
   console.log("Creating your instance. This can take a few seconds.");
   instanceId = await runInstance({
     keyPairName,
     securityGroupId,
     imageId: imageDetails.ImageId,
     instanceType: instanceTypeDetails.InstanceType,
   });
   const instanceDetails = await describeInstance(instanceId);
   console.log(`# instance ${instanceId}.\n`);
   console.log(instanceDetails);
   console.log(
     `\nYou should now be able to SSH into your instance from another
terminal:`,
     `\n${displaySSHConnectionInfo({
       publicIp: instanceDetails.PublicIpAddress,
       keyPairName,
     })}`,
   );

   await prompter.confirm({ message: confirmMessage });

   // Understanding the IP address.
   console.log(wrapText("Understanding the IP address."));
   console.log(
     "When you stop and start an instance, the IP address will change. I'll
restart your",
     "instance for you. Notice how the IP address changes.",
   );
   const ipAddressAfterRestart = await restartInstance(instanceId);
   console.log(
     `\n Instance started. The IP address changed from
${instanceDetails.PublicIpAddress} to ${ipAddressAfterRestart}`,
     `\n${displaySSHConnectionInfo({
       publicIp: ipAddressAfterRestart,
       keyPairName,
```

```
      })}`,
    );
    await prompter.confirm({ message: confirmMessage });
    console.log(
      `If you want to the IP address to be static, you can associate an
allocated`,
      `IP address to your instance. I allocated ${publicIp} for you earlier, and
now I'll associate it to your instance.`,
    );
    associationId = await associateAddress({
      allocationId: ipAllocationId,
      instanceId,
    });
    console.log(
      "Done. Now you should be able to SSH using the new IP.\n",
      `${displaySSHConnectionInfo({ publicIp, keyPairName })}`,
    );
    await prompter.confirm({ message: confirmMessage });
    console.log(
      "I'll restart the server again so you can see the IP address remains the
same.",
    );
    const ipAddressAfterAssociated = await restartInstance(instanceId);
    console.log(
      `Done. Here's your SSH info. Notice the IP address hasn't changed.`,
      `\n${displaySSHConnectionInfo({
        publicIp: ipAddressAfterAssociated,
        keyPairName,
      })}`,
    );
    await prompter.confirm({ message: confirmMessage });
  } catch (err) {
    console.error(err);
  } finally {
    // Clean up.
    console.log(wrapText("Clean up."));
    console.log("Now I'll clean up all of the stuff I created.");
    await prompter.confirm({ message: confirmMessage });
    console.log("Cleaning up. Some of these steps can take a bit of time.");
    await disassociateAddress(associationId);
    await terminateInstance(instanceId);
    await releaseAddress(ipAllocationId);
    await deleteSecurityGroup(securityGroupId);
    deleteTemporaryDirectory();
```

```
    await deleteKeyPair(keyPairName);
    console.log(
      "Done cleaning up. Thanks for staying until the end!",
      "If you have any feedback please use the feedback button in the docs",
      "or create an issue on GitHub.",
    );
  }
};
```

- 如需 API 詳細資訊，請參閱《AWS SDK for JavaScript API 參考》中的下列主題。
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DescribeImages
  - DescribeInstanceTypes
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups
  - DisassociateAddress
  - ReleaseAddress
  - RunInstances
  - StartInstances
  - StopInstances
  - TerminateInstances
  - UnmonitorInstances

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
/**
 Before running this Kotlin 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-kotlin/latest/developer-guide/setup.html

 This Kotlin example performs the following tasks:

 1. Creates an RSA key pair and saves the private key data as a .pem file.
 2. Lists key pairs.
 3. Creates a security group for the default VPC.
 4. Displays security group information.
 5. Gets a list of Amazon Linux 2 AMIs and selects one.
 6. Gets more information about the image.
 7. Gets a list of instance types that are compatible with the selected AMI's
 architecture.
 8. Creates an instance with the key pair, security group, AMI, and an instance
 type.
 9. Displays information about the instance.
 10. Stops the instance and waits for it to stop.
 11. Starts the instance and waits for it to start.
 12. Allocates an Elastic IP address and associates it with the instance.
 13. Displays SSH connection info for the instance.
 14. Disassociates and deletes the Elastic IP address.
 15. Terminates the instance.
 16. Deletes the security group.
 17. Deletes the key pair.
 */

val DASHES = String(CharArray(80)).replace("\u0000", "-")
```

```
suspend fun main(args: Array<String>) {
    val usage = """
        Usage:
            <keyName> <fileName> <groupName> <groupDesc> <vpcId> <myIpAddress>

        Where:
            keyName - A key pair name (for example, TestKeyPair).
            fileName - A file name where the key information is written to.
            groupName - The name of the security group.
            groupDesc - The description of the security group.
            vpcId - A VPC ID. You can get this value from the AWS Management
 Console.
            myIpAddress - The IP address of your development machine.

"""

    if (args.size != 6) {
        println(usage)
        exitProcess(0)
    }

    val keyName = args[0]
    val fileName = args[1]
    val groupName = args[2]
    val groupDesc = args[3]
    val vpcId = args[4]
    val myIpAddress = args[5]
    var newInstanceId: String? = ""

    println(DASHES)
    println("Welcome to the Amazon EC2 example scenario.")
    println(DASHES)

    println(DASHES)
    println("1. Create an RSA key pair and save the private key material as
 a .pem file.")
    createKeyPairSc(keyName, fileName)
    println(DASHES)

    println(DASHES)
    println("2. List key pairs.")
    describeEC2KeysSc()
    println(DASHES)
```

```
   println(DASHES)
   println("3. Create a security group.")
   val groupId = createEC2SecurityGroupSc(groupName, groupDesc, vpcId,
myIpAddress)
   println(DASHES)

   println(DASHES)
   println("4. Display security group info for the newly created security
group.")
   describeSecurityGroupsSc(groupId.toString())
   println(DASHES)

   println(DASHES)
   println("5. Get a list of Amazon Linux 2 AMIs and select one with amzn2 in
the name.")
   val instanceId = getParaValuesSc()
   if (instanceId == "") {
       println("The instance Id value isn't valid.")
       exitProcess(0)
   }
   println("The instance Id is $instanceId.")
   println(DASHES)

   println(DASHES)
   println("6. Get more information about an amzn2 image and return the AMI
value.")
   val amiValue = instanceId?.let { describeImageSc(it) }
   if (instanceId == "") {
       println("The instance Id value is invalid.")
       exitProcess(0)
   }
   println("The AMI value is $amiValue.")
   println(DASHES)

   println(DASHES)
   println("7. Get a list of instance types.")
   val instanceType = getInstanceTypesSc()
   println(DASHES)

   println(DASHES)
   println("8. Create an instance.")
   if (amiValue != null) {
       newInstanceId = runInstanceSc(instanceType, keyName, groupName, amiValue)
```

```
        println("The instance Id is $newInstanceId")
    }
    println(DASHES)

    println(DASHES)
    println("9. Display information about the running instance. ")
    var ipAddress = describeEC2InstancesSc(newInstanceId)
    println("You can SSH to the instance using this command:")
    println("ssh -i " + fileName + "ec2-user@" + ipAddress)
    println(DASHES)

    println(DASHES)
    println("10.  Stop the instance.")
    if (newInstanceId != null) {
        stopInstanceSc(newInstanceId)
    }
    println(DASHES)

    println(DASHES)
    println("11.  Start the instance.")
    if (newInstanceId != null) {
        startInstanceSc(newInstanceId)
    }
    ipAddress = describeEC2InstancesSc(newInstanceId)
    println("You can SSH to the instance using this command:")
    println("ssh -i " + fileName + "ec2-user@" + ipAddress)
    println(DASHES)

    println(DASHES)
    println("12. Allocate an Elastic IP address and associate it with the
instance.")
    val allocationId = allocateAddressSc()
    println("The allocation Id value is $allocationId")
    val associationId = associateAddressSc(newInstanceId, allocationId)
    println("The associate Id value is $associationId")
    println(DASHES)

    println(DASHES)
    println("13. Describe the instance again.")
    ipAddress = describeEC2InstancesSc(newInstanceId)
    println("You can SSH to the instance using this command:")
    println("ssh -i " + fileName + "ec2-user@" + ipAddress)
    println(DASHES)
```

```
    println(DASHES)
    println("14. Disassociate and release the Elastic IP address.")
    disassociateAddressSc(associationId)
    releaseEC2AddressSc(allocationId)
    println(DASHES)

    println(DASHES)
    println("15. Terminate the instance and use a waiter.")
    if (newInstanceId != null) {
        terminateEC2Sc(newInstanceId)
    }
    println(DASHES)

    println(DASHES)
    println("16. Delete the security group.")
    if (groupId != null) {
        deleteEC2SecGroupSc(groupId)
    }
    println(DASHES)

    println(DASHES)
    println("17. Delete the key pair.")
    deleteKeysSc(keyName)
    println(DASHES)

    println(DASHES)
    println("You successfully completed the Amazon EC2 scenario.")
    println(DASHES)
}

suspend fun deleteKeysSc(keyPair: String) {
    val request =
        DeleteKeyPairRequest {
            keyName = keyPair
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.deleteKeyPair(request)
        println("Successfully deleted key pair named $keyPair")
    }
}

suspend fun deleteEC2SecGroupSc(groupIdVal: String) {
    val request =
        DeleteSecurityGroupRequest {
```

```
            groupId = groupIdVal
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.deleteSecurityGroup(request)
        println("Successfully deleted security group with Id $groupIdVal")
    }
}

suspend fun terminateEC2Sc(instanceIdVal: String) {
    val ti =
        TerminateInstancesRequest {
            instanceIds = listOf(instanceIdVal)
        }
    println("Wait for the instance to terminate. This will take a few minutes.")
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.terminateInstances(ti)
        ec2.waitUntilInstanceTerminated {
            // suspend call
            instanceIds = listOf(instanceIdVal)
        }
        println("$instanceIdVal is terminated!")
    }
}

suspend fun releaseEC2AddressSc(allocId: String?) {
    val request =
        ReleaseAddressRequest {
            allocationId = allocId
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.releaseAddress(request)
        println("Successfully released Elastic IP address $allocId")
    }
}

suspend fun disassociateAddressSc(associationIdVal: String?) {
    val addressRequest =
        DisassociateAddressRequest {
            associationId = associationIdVal
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.disassociateAddress(addressRequest)
        println("You successfully disassociated the address!")
```

```
    }
}

suspend fun associateAddressSc(
    instanceIdVal: String?,
    allocationIdVal: String?,
): String? {
    val associateRequest =
        AssociateAddressRequest {
            instanceId = instanceIdVal
            allocationId = allocationIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val associateResponse = ec2.associateAddress(associateRequest)
        return associateResponse.associationId
    }
}

suspend fun allocateAddressSc(): String? {
    val allocateRequest =
        AllocateAddressRequest {
            domain = DomainType.Vpc
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val allocateResponse = ec2.allocateAddress(allocateRequest)
        return allocateResponse.allocationId
    }
}

suspend fun startInstanceSc(instanceId: String) {
    val request =
        StartInstancesRequest {
            instanceIds = listOf(instanceId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.startInstances(request)
        println("Waiting until instance $instanceId starts. This will take a few
 minutes.")
        ec2.waitUntilInstanceRunning {
            // suspend call
            instanceIds = listOf(instanceId)
        }
```

```
        println("Successfully started instance $instanceId")
    }
}

suspend fun stopInstanceSc(instanceId: String) {
    val request =
        StopInstancesRequest {
            instanceIds = listOf(instanceId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.stopInstances(request)
        println("Waiting until instance $instanceId stops. This will take a few
 minutes.")
        ec2.waitUntilInstanceStopped {
            // suspend call
            instanceIds = listOf(instanceId)
        }
        println("Successfully stopped instance $instanceId")
    }
}

suspend fun describeEC2InstancesSc(newInstanceId: String?): String {
    var pubAddress = ""
    var isRunning = false
    val request =
        DescribeInstancesRequest {
            instanceIds = listOf(newInstanceId.toString())
        }

    while (!isRunning) {
        Ec2Client { region = "us-west-2" }.use { ec2 ->
            val response = ec2.describeInstances(request)
            val state =
                response.reservations
                    ?.get(0)
                    ?.instances
                    ?.get(0)
                    ?.state
                    ?.name
                    ?. value
            if (state != null) {
                if (state.compareTo("running") == 0) {
```

```
                    println("Image id is
 ${response.reservations!!.get(0).instances?.get(0)?.imageId}")
                    println("Instance type is
 ${response.reservations!!.get(0).instances?.get(0)?.instanceType}")
                    println("Instance state is
 ${response.reservations!!.get(0).instances?.get(0)?.state}")
                    pubAddress =
                        response.reservations!!
                            .get(0)
                            .instances
                            ?.get(0)
                            ?.publicIpAddress
                            .toString()
                    println("Instance address is $pubAddress")
                    isRunning = true
                }
            }
        }
    }
    return pubAddress
}

suspend fun runInstanceSc(
    instanceTypeVal: String,
    keyNameVal: String,
    groupNameVal: String,
    amiIdVal: String,
): String {
    val runRequest =
        RunInstancesRequest {
            instanceType = InstanceType.fromValue(instanceTypeVal)
            keyName = keyNameVal
            securityGroups = listOf(groupNameVal)
            maxCount = 1
            minCount = 1
            imageId = amiIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.runInstances(runRequest)
        val instanceId = response.instances?.get(0)?.instanceId
        println("Successfully started EC2 Instance $instanceId based on AMI
 $amiIdVal")
        return instanceId.toString()
```

```
    }
}

// Get a list of instance types.
suspend fun getInstanceTypesSc(): String {
    var instanceType = ""
    val filterObs = ArrayList<Filter>()
    val filter =
        Filter {
            name = "processor-info.supported-architecture"
            values = listOf("arm64")
        }

    filterObs.add(filter)
    val typesRequest =
        DescribeInstanceTypesRequest {
            filters = filterObs
            maxResults = 10
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeInstanceTypes(typesRequest)
        response.instanceTypes?.forEach { type ->
            println("The memory information of this type is
 ${type.memoryInfo?.sizeInMib}")
            println("Maximum number of network cards is
 ${type.networkInfo?.maximumNetworkCards}")
            instanceType = type.instanceType.toString()
        }
        return instanceType
    }
}

// Display the Description field that corresponds to the instance Id value.
suspend fun describeImageSc(instanceId: String): String? {
    val imagesRequest =
        DescribeImagesRequest {
            imageIds = listOf(instanceId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeImages(imagesRequest)
        println("The description of the first image is
 ${response.images?.get(0)?.description}")
```

```
        println("The name of the first image is
 ${response.images?.get(0)?.name}")

        // Return the image Id value.
        return response.images?.get(0)?.imageId
    }
}

// Get the Id value of an instance with amzn2 in the name.
suspend fun getParaValuesSc(): String? {
    val parameterRequest =
        GetParametersByPathRequest {
            path = "/aws/service/ami-amazon-linux-latest"
        }

    SsmClient { region = "us-west-2" }.use { ssmClient ->
        val response = ssmClient.getParametersByPath(parameterRequest)
        response.parameters?.forEach { para ->
            println("The name of the para is: ${para.name}")
            println("The type of the para is: ${para.type}")
            println("")
            if (para.name?.let { filterName(it) } == true) {
                return para.value
            }
        }
    }
    return ""
}

fun filterName(name: String): Boolean {
    val parts = name.split("/").toTypedArray()
    val myValue = parts[4]
    return myValue.contains("amzn2")
}

suspend fun describeSecurityGroupsSc(groupId: String) {
    val request =
        DescribeSecurityGroupsRequest {
            groupIds = listOf(groupId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeSecurityGroups(request)
        for (group in response.securityGroups!!) {
```

```
            println("Found Security Group with id " + group.groupId.toString() +
 " and group VPC " + group.vpcId)
        }
    }
}

suspend fun createEC2SecurityGroupSc(
    groupNameVal: String?,
    groupDescVal: String?,
    vpcIdVal: String?,
    myIpAddress: String?,
): String? {
    val request =
        CreateSecurityGroupRequest {
            groupName = groupNameVal
            description = groupDescVal
            vpcId = vpcIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val resp = ec2.createSecurityGroup(request)
        val ipRange =
            IpRange {
                cidrIp = "$myIpAddress/0"
            }

        val ipPerm =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 80
                fromPort = 80
                ipRanges = listOf(ipRange)
            }

        val ipPerm2 =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 22
                fromPort = 22
                ipRanges = listOf(ipRange)
            }

        val authRequest =
            AuthorizeSecurityGroupIngressRequest {
```

```
                groupName = groupNameVal
                ipPermissions = listOf(ipPerm, ipPerm2)
            }
        ec2.authorizeSecurityGroupIngress(authRequest)
        println("Successfully added ingress policy to Security Group
 $groupNameVal")
        return resp.groupId
    }
}

suspend fun describeEC2KeysSc() {
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeKeyPairs(DescribeKeyPairsRequest {})
        response.keyPairs?.forEach { keyPair ->
            println("Found key pair with name ${keyPair.keyName} and fingerprint
 ${ keyPair.keyFingerprint}")
        }
    }
}

suspend fun createKeyPairSc(
    keyNameVal: String,
    fileNameVal: String,
) {
    val request =
        CreateKeyPairRequest {
            keyName = keyNameVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.createKeyPair(request)
        val content = response.keyMaterial
        if (content != null) {
            File(fileNameVal).writeText(content)
        }
        println("Successfully created key pair named $keyNameVal")
    }
}
```

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

- AuthorizeSecurityGroupIngress
- CreateKeyPair
- CreateSecurityGroup
- DeleteKeyPair
- DeleteSecurityGroup
- DescribeImages
- DescribeInstanceTypes
- DescribeInstances
- DescribeKeyPairs
- DescribeSecurityGroups
- DisassociateAddress
- ReleaseAddress
- RunInstances
- StartInstances
- StopInstances
- TerminateInstances
- UnmonitorInstances

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

在命令提示中執行互動式案例。

```
class Ec2InstanceScenario:
    """Runs an interactive scenario that shows how to get started using EC2
 instances."""
```

```
    def __init__(self, inst_wrapper, key_wrapper, sg_wrapper, eip_wrapper,
ssm_client):
        """
        :param inst_wrapper: An object that wraps instance actions.
        :param key_wrapper: An object that wraps key pair actions.
        :param sg_wrapper: An object that wraps security group actions.
        :param eip_wrapper: An object that wraps Elastic IP actions.
        :param ssm_client: A Boto3 AWS Systems Manager client.
        """
        self.inst_wrapper = inst_wrapper
        self.key_wrapper = key_wrapper
        self.sg_wrapper = sg_wrapper
        self.eip_wrapper = eip_wrapper
        self.ssm_client = ssm_client

    @demo_func
    def create_and_list_key_pairs(self):
        """
        1. Creates an RSA key pair and saves its private key data as a .pem file
in secure
           temporary storage. The private key data is deleted after the example
completes.
        2. Lists the first five key pairs for the current account.
        """
        print(
            "Let's create an RSA key pair that you can be use to securely connect
to "
            "your EC2 instance."
        )
        key_name = q.ask("Enter a unique name for your key: ", q.non_empty)
        self.key_wrapper.create(key_name)
        print(
            f"Created a key pair {self.key_wrapper.key_pair.key_name} and saved
the "
            f"private key to {self.key_wrapper.key_file_path}.\n"
        )
        if q.ask("Do you want to list some of your key pairs? (y/n) ",
q.is_yesno):
            self.key_wrapper.list(5)

    @demo_func
    def create_security_group(self):
        """
        1. Creates a security group for the default VPC.
```

```
        2. Adds an inbound rule to allow SSH. The SSH rule allows only
           inbound traffic from the current computer's public IPv4 address.
        3. Displays information about the security group.

        This function uses 'http://checkip.amazonaws.com' to get the current
public IP
        address of the computer that is running the example. This method works in
most
        cases. However, depending on how your computer connects to the internet,
you
        might have to manually add your public IP address to the security group
by using
        the AWS Management Console.
        """
        print("Let's create a security group to manage access to your instance.")
        sg_name = q.ask("Enter a unique name for your security group: ",
q.non_empty)
        security_group = self.sg_wrapper.create(
            sg_name, "Security group for example: get started with instances."
        )
        print(
            f"Created security group {security_group.group_name} in your default
"
            f"VPC {security_group.vpc_id}.\n"
        )

        ip_response = urllib.request.urlopen("http://checkip.amazonaws.com")
        current_ip_address = ip_response.read().decode("utf-8").strip()
        print("Let's add a rule to allow SSH only from your current IP address.")
        print(f"Your public IP address is {current_ip_address}.")
        q.ask("Press Enter to add this rule to your security group.")
        response = self.sg_wrapper.authorize_ingress(current_ip_address)
        if response["Return"]:
            print("Security group rules updated.")
        else:
            print("Couldn't update security group rules.")
        self.sg_wrapper.describe()

    @demo_func
    def create_instance(self):
        """
        1. Gets a list of Amazon Linux 2 AMIs from AWS Systems Manager.
Specifying the
```

```
            '/aws/service/ami-amazon-linux-latest' path returns only the latest
 AMIs.
        2. Gets and displays information about the available AMIs and lets you
 select one.
        3. Gets a list of instance types that are compatible with the selected
 AMI and
           lets you select one.
        4. Creates an instance with the previously created key pair and security
 group,
           and the selected AMI and instance type.
        5. Waits for the instance to be running and then displays its
 information.
        """
        ami_paginator = self.ssm_client.get_paginator("get_parameters_by_path")
        ami_options = []
        for page in ami_paginator.paginate(Path="/aws/service/ami-amazon-linux-
latest"):
            ami_options += page["Parameters"]
        amzn2_images = self.inst_wrapper.get_images(
            [opt["Value"] for opt in ami_options if "amzn2" in opt["Name"]]
        )
        print(
            "Let's create an instance from an Amazon Linux 2 AMI. Here are some
 options:"
        )
        image_choice = q.choose(
            "Which one do you want to use? ", [opt.description for opt in
 amzn2_images]
        )
        print("Great choice!\n")

        print(
            f"Here are some instance types that support the "
            f"{amzn2_images[image_choice].architecture} architecture of the
 image:"
        )
        inst_types = self.inst_wrapper.get_instance_types(
            amzn2_images[image_choice].architecture
        )
        inst_type_choice = q.choose(
            "Which one do you want to use? ", [it["InstanceType"] for it in
 inst_types]
        )
        print("Another great choice.\n")
```

```
        print("Creating your instance and waiting for it to start...")
        self.inst_wrapper.create(
            amzn2_images[image_choice],
            inst_types[inst_type_choice]["InstanceType"],
            self.key_wrapper.key_pair,
            [self.sg_wrapper.security_group],
        )
        print(f"Your instance is ready:\n")
        self.inst_wrapper.display()

        print("You can use SSH to connect to your instance.")
        print(
            "If the connection attempt times out, you might have to manually
update "
            "the SSH ingress rule for your IP address in the AWS Management
Console."
        )
        self._display_ssh_info()

    def _display_ssh_info(self):
        """
        Displays an SSH connection string that can be used to connect to a
running
        instance.
        """
        print("To connect, open another command prompt and run the following
command:")
        if self.eip_wrapper.elastic_ip is None:
            print(
                f"\tssh -i {self.key_wrapper.key_file_path} "
                f"ec2-user@{self.inst_wrapper.instance.public_ip_address}"
            )
        else:
            print(
                f"\tssh -i {self.key_wrapper.key_file_path} "
                f"ec2-user@{self.eip_wrapper.elastic_ip.public_ip}"
            )
        q.ask("Press Enter when you're ready to continue the demo.")

    @demo_func
    def associate_elastic_ip(self):
        """
        1. Allocates an Elastic IP address and associates it with the instance.
```

```
        2. Displays an SSH connection string that uses the Elastic IP address.
        """
        print(
            "You can allocate an Elastic IP address and associate it with your
instance\n"
            "to keep a consistent IP address even when your instance restarts."
        )
        elastic_ip = self.eip_wrapper.allocate()
        print(f"Allocated static Elastic IP address: {elastic_ip.public_ip}.")
        self.eip_wrapper.associate(self.inst_wrapper.instance)
        print(f"Associated your Elastic IP with your instance.")
        print(
            "You can now use SSH to connect to your instance by using the Elastic
IP."
        )
        self._display_ssh_info()

    @demo_func
    def stop_and_start_instance(self):
        """
        1. Stops the instance and waits for it to stop.
        2. Starts the instance and waits for it to start.
        3. Displays information about the instance.
        4. Displays an SSH connection string. When an Elastic IP address is
associated
           with the instance, the IP address stays consistent when the instance
stops
           and starts.
        """
        print("Let's stop and start your instance to see what changes.")
        print("Stopping your instance and waiting until it's stopped...")
        self.inst_wrapper.stop()
        print("Your instance is stopped. Restarting...")
        self.inst_wrapper.start()
        print("Your instance is running.")
        self.inst_wrapper.display()
        if self.eip_wrapper.elastic_ip is None:
            print(
                "Every time your instance is restarted, its public IP address
changes."
            )
        else:
            print(
```

```
                "Because you have associated an Elastic IP with your instance,
you can \n"
                "connect by using a consistent IP address after the instance
restarts."
            )
        self._display_ssh_info()

    @demo_func
    def cleanup(self):
        """
        1. Disassociate and delete the previously created Elastic IP.
        2. Terminate the previously created instance.
        3. Delete the previously created security group.
        4. Delete the previously created key pair.
        """
        print("Let's clean everything up. This example created these resources:")
        print(f"\tElastic IP: {self.eip_wrapper.elastic_ip.allocation_id}")
        print(f"\tInstance: {self.inst_wrapper.instance.id}")
        print(f"\tSecurity group: {self.sg_wrapper.security_group.id}")
        print(f"\tKey pair: {self.key_wrapper.key_pair.name}")
        if q.ask("Ready to delete these resources? (y/n) ", q.is_yesno):
            self.eip_wrapper.disassociate()
            print("Disassociated the Elastic IP from the instance.")
            self.eip_wrapper.release()
            print("Released the Elastic IP.")
            print("Terminating the instance and waiting for it to terminate...")
            self.inst_wrapper.terminate()
            print("Instance terminated.")
            self.sg_wrapper.delete()
            print("Deleted security group.")
            self.key_wrapper.delete()
            print("Deleted key pair.")

    def run_scenario(self):
        logging.basicConfig(level=logging.INFO, format="%(levelname)s:
%(message)s")

        print("-" * 88)
        print(
            "Welcome to the Amazon Elastic Compute Cloud (Amazon EC2) get started
with instances demo."
        )
        print("-" * 88)
```

```
        self.create_and_list_key_pairs()
        self.create_security_group()
        self.create_instance()
        self.stop_and_start_instance()
        self.associate_elastic_ip()
        self.stop_and_start_instance()
        self.cleanup()

        print("\nThanks for watching!")
        print("-" * 88)


if __name__ == "__main__":
    try:
        scenario = Ec2InstanceScenario(
            InstanceWrapper.from_resource(),
            KeyPairWrapper.from_resource(),
            SecurityGroupWrapper.from_resource(),
            ElasticIpWrapper.from_resource(),
            boto3.client("ssm"),
        )
        scenario.run_scenario()
    except Exception:
        logging.exception("Something went wrong with the demo.")
```

定義包裝金鑰對動作的類別。

```
class KeyPairWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair
 actions."""

    def __init__(self, ec2_resource, key_file_dir, key_pair=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param key_file_dir: The folder where the private key information is
 stored.
                             This should be a secure folder.
        :param key_pair: A Boto3 KeyPair object. This is a high-level object that
                         wraps key pair actions.
```

```
        """
        self.ec2_resource = ec2_resource
        self.key_pair = key_pair
        self.key_file_path = None
        self.key_file_dir = key_file_dir

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource, tempfile.TemporaryDirectory())


    def create(self, key_name):
        """
        Creates a key pair that can be used to securely connect to an EC2
instance.
        The returned key pair contains private key information that cannot be
retrieved
        again. The private key data is stored as a .pem file.

        :param key_name: The name of the key pair to create.
        :return: A Boto3 KeyPair object that represents the newly created key
pair.
        """
        try:
            self.key_pair = self.ec2_resource.create_key_pair(KeyName=key_name)
            self.key_file_path = os.path.join(
                self.key_file_dir.name, f"{self.key_pair.name}.pem"
            )
            with open(self.key_file_path, "w") as key_file:
                key_file.write(self.key_pair.key_material)
        except ClientError as err:
            logger.error(
                "Couldn't create key %s. Here's why: %s: %s",
                key_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.key_pair


    def list(self, limit):
```

```
        """
        Displays a list of key pairs for the current account.

        :param limit: The maximum number of key pairs to list.
        """
        try:
            for kp in self.ec2_resource.key_pairs.limit(limit):
                print(f"Found {kp.key_type} key {kp.name} with fingerprint:")
                print(f"\t{kp.key_fingerprint}")
        except ClientError as err:
            logger.error(
                "Couldn't list key pairs. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete(self):
        """
        Deletes a key pair.
        """
        if self.key_pair is None:
            logger.info("No key pair to delete.")
            return

        key_name = self.key_pair.name
        try:
            self.key_pair.delete()
            self.key_pair = None
        except ClientError as err:
            logger.error(
                "Couldn't delete key %s. Here's why: %s : %s",
                key_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

定義包裝安全群組動作的類別。

```
class SecurityGroupWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) security group
 actions."""

    def __init__(self, ec2_resource, security_group=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param security_group: A Boto3 SecurityGroup object. This is a high-level
 object
                               that wraps security group actions.
        """
        self.ec2_resource = ec2_resource
        self.security_group = security_group

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def create(self, group_name, group_description):
        """
        Creates a security group in the default virtual private cloud (VPC) of
 the
        current account.

        :param group_name: The name of the security group to create.
        :param group_description: The description of the security group to
 create.
        :return: A Boto3 SecurityGroup object that represents the newly created
 security group.
        """
        try:
            self.security_group = self.ec2_resource.create_security_group(
                GroupName=group_name, Description=group_description
            )
        except ClientError as err:
            logger.error(
                "Couldn't create security group %s. Here's why: %s: %s",
```

```
                group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.security_group


    def authorize_ingress(self, ssh_ingress_ip):
        """
        Adds a rule to the security group to allow access to SSH.

        :param ssh_ingress_ip: The IP address that is granted inbound access to
connect
                               to port 22 over TCP, used for SSH.
        :return: The response to the authorization request. The 'Return' field of
the
                 response indicates whether the request succeeded or failed.
        """
        if self.security_group is None:
            logger.info("No security group to update.")
            return

        try:
            ip_permissions = [
                {
                    # SSH ingress open to only the specified IP address.
                    "IpProtocol": "tcp",
                    "FromPort": 22,
                    "ToPort": 22,
                    "IpRanges": [{"CidrIp": f"{ssh_ingress_ip}/32"}],
                }
            ]
            response = self.security_group.authorize_ingress(
                IpPermissions=ip_permissions
            )
        except ClientError as err:
            logger.error(
                "Couldn't authorize inbound rules for %s. Here's why: %s: %s",
                self.security_group.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
```

```
            raise
        else:
            return response


    def describe(self):
        """
        Displays information about the security group.
        """
        if self.security_group is None:
            logger.info("No security group to describe.")
            return

        try:
            print(f"Security group: {self.security_group.group_name}")
            print(f"\tID: {self.security_group.id}")
            print(f"\tVPC: {self.security_group.vpc_id}")
            if self.security_group.ip_permissions:
                print(f"Inbound permissions:")
                pp(self.security_group.ip_permissions)
        except ClientError as err:
            logger.error(
                "Couldn't get data for security group %s. Here's why: %s: %s",
                self.security_group.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete(self):
        """
        Deletes the security group.
        """
        if self.security_group is None:
            logger.info("No security group to delete.")
            return

        group_id = self.security_group.id
        try:
            self.security_group.delete()
        except ClientError as err:
            logger.error(
                "Couldn't delete security group %s. Here's why: %s: %s",
```

```
                group_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

定義包裝執行個體動作的類別。

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
                          wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def create(self, image, instance_type, key_pair, security_groups=None):
        """
        Creates a new EC2 instance. The instance starts immediately after
        it is created.

        The instance is created in the default VPC of the current account.
```

```
        :param image: A Boto3 Image object that represents an Amazon Machine
Image (AMI)
                      that defines attributes of the instance that is created.
The AMI
                      defines things like the kind of operating system and the
type of
                      storage used by the instance.
        :param instance_type: The type of instance to create, such as 't2.micro'.
                              The instance type defines things like the number of
CPUs and
                              the amount of memory.
        :param key_pair: A Boto3 KeyPair or KeyPairInfo object that represents
the key
                         pair that is used to secure connections to the instance.
        :param security_groups: A list of Boto3 SecurityGroup objects that
represents the
                                security groups that are used to grant access to
the
                                instance. When no security groups are specified,
the
                                default security group of the VPC is used.
        :return: A Boto3 Instance object that represents the newly created
instance.
        """
        try:
            instance_params = {
                "ImageId": image.id,
                "InstanceType": instance_type,
                "KeyName": key_pair.name,
            }
            if security_groups is not None:
                instance_params["SecurityGroupIds"] = [sg.id for sg in
security_groups]
            self.instance = self.ec2_resource.create_instances(
                **instance_params, MinCount=1, MaxCount=1
            )[0]
            self.instance.wait_until_running()
        except ClientError as err:
            logging.error(
                "Couldn't create instance with image %s, instance type %s, and
key %s. "
                "Here's why: %s: %s",
                image.id,
                instance_type,
```

```
                key_pair.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.instance


    def display(self, indent=1):
        """
        Displays information about an instance.

        :param indent: The visual indent to apply to the output.
        """
        if self.instance is None:
            logger.info("No instance to display.")
            return

        try:
            self.instance.load()
            ind = "\t" * indent
            print(f"{ind}ID: {self.instance.id}")
            print(f"{ind}Image ID: {self.instance.image_id}")
            print(f"{ind}Instance type: {self.instance.instance_type}")
            print(f"{ind}Key name: {self.instance.key_name}")
            print(f"{ind}VPC ID: {self.instance.vpc_id}")
            print(f"{ind}Public IP: {self.instance.public_ip_address}")
            print(f"{ind}State: {self.instance.state['Name']}")
        except ClientError as err:
            logger.error(
                "Couldn't display your instance. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def terminate(self):
        """
        Terminates an instance and waits for it to be in a terminated state.
        """
        if self.instance is None:
            logger.info("No instance to terminate.")
```

```
            return

        instance_id = self.instance.id
        try:
            self.instance.terminate()
            self.instance.wait_until_terminated()
            self.instance = None
        except ClientError as err:
            logging.error(
                "Couldn't terminate instance %s. Here's why: %s: %s",
                instance_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def start(self):
        """
        Starts an instance and waits for it to be in a running state.

        :return: The response to the start request.
        """
        if self.instance is None:
            logger.info("No instance to start.")
            return

        try:
            response = self.instance.start()
            self.instance.wait_until_running()
        except ClientError as err:
            logger.error(
                "Couldn't start instance %s. Here's why: %s: %s",
                self.instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response


    def stop(self):
        """
```

```
        Stops an instance and waits for it to be in a stopped state.

        :return: The response to the stop request.
        """
        if self.instance is None:
            logger.info("No instance to stop.")
            return

        try:
            response = self.instance.stop()
            self.instance.wait_until_stopped()
        except ClientError as err:
            logger.error(
                "Couldn't stop instance %s. Here's why: %s: %s",
                self.instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response


    def get_images(self, image_ids):
        """
        Gets information about Amazon Machine Images (AMIs) from a list of AMI
IDs.

        :param image_ids: The list of AMIs to look up.
        :return: A list of Boto3 Image objects that represent the requested AMIs.
        """
        try:
            images = list(self.ec2_resource.images.filter(ImageIds=image_ids))
        except ClientError as err:
            logger.error(
                "Couldn't get images. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return images
```

```
   def get_instance_types(self, architecture):
       """
       Gets instance types that support the specified architecture and are
designated
       as either 'micro' or 'small'. When an instance is created, the instance
type
       you specify must support the architecture of the AMI you use.

       :param architecture: The kind of architecture the instance types must
support,
                            such as 'x86_64'.
       :return: A list of instance types that support the specified architecture
                and are either 'micro' or 'small'.
       """
       try:
           inst_types = []
           it_paginator = self.ec2_resource.meta.client.get_paginator(
               "describe_instance_types"
           )
           for page in it_paginator.paginate(
               Filters=[
                   {
                       "Name": "processor-info.supported-architecture",
                       "Values": [architecture],
                   },
                   {"Name": "instance-type", "Values": ["*.micro", "*.small"]},
               ]
           ):
               inst_types += page["InstanceTypes"]
       except ClientError as err:
           logger.error(
               "Couldn't get instance types. Here's why: %s: %s",
               err.response["Error"]["Code"],
               err.response["Error"]["Message"],
           )
           raise
       else:
           return inst_types
```

定義包裝彈性 IP 動作的類別。

```
class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
 actions."""

    def __init__(self, ec2_resource, elastic_ip=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param elastic_ip: A Boto3 VpcAddress object. This is a high-level object
 that
                           wraps Elastic IP actions.
        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def allocate(self):
        """
        Allocates an Elastic IP address that can be associated with an Amazon EC2
        instance. By using an Elastic IP address, you can keep the public IP
 address
        constant even when you restart the associated instance.

        :return: The newly created Elastic IP object. By default, the address is
 not
                 associated with any instance.
        """
        try:
            response =
 self.ec2_resource.meta.client.allocate_address(Domain="vpc")
            self.elastic_ip =
 self.ec2_resource.VpcAddress(response["AllocationId"])
        except ClientError as err:
            logger.error(
                "Couldn't allocate Elastic IP. Here's why: %s: %s",
```

```
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.elastic_ip


    def associate(self, instance):
        """
        Associates an Elastic IP address with an instance. When this association
is
        created, the Elastic IP's public IP address is immediately used as the
public
        IP address of the associated instance.

        :param instance: A Boto3 Instance object. This is a high-level object
that wraps
                         Amazon EC2 instance actions.
        :return: A response that contains the ID of the association.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to associate.")
            return

        try:
            response = self.elastic_ip.associate(InstanceId=instance.id)
        except ClientError as err:
            logger.error(
                "Couldn't associate Elastic IP %s with instance %s. Here's why:
%s: %s",
                self.elastic_ip.allocation_id,
                instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        return response


    def disassociate(self):
        """
        Removes an association between an Elastic IP address and an instance.
When the
```

```
        association is removed, the instance is assigned a new public IP address.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to disassociate.")
            return

        try:
            self.elastic_ip.association.delete()
        except ClientError as err:
            logger.error(
                "Couldn't disassociate Elastic IP %s from its instance. Here's
why: %s: %s",
                self.elastic_ip.allocation_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def release(self):
        """
        Releases an Elastic IP address. After the Elastic IP address is released,
        it can no longer be used.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to release.")
            return

        try:
            self.elastic_ip.release()
        except ClientError as err:
            logger.error(
                "Couldn't release Elastic IP address %s. Here's why: %s: %s",
                self.elastic_ip.allocation_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 如需 API 的詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考資料》中的下列主題。
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DescribeImages
  - DescribeInstanceTypes
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups
  - DisassociateAddress
  - ReleaseAddress
  - RunInstances
  - StartInstances
  - StopInstances
  - TerminateInstances
  - UnmonitorInstances

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 開發套件建立 Amazon EC2 資源。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 Amazon 監控亞馬遜 EC2 API 請求 CloudWatch

您可以使用 Amazon 監控 Amazon EC2 API 請求 CloudWatch，Amazon 會收集原始資料並將其處理為可讀且近乎即時的指標。這些指標提供一種簡單的方法來追蹤一段時間內 Amazon EC2 API 操作的使用情況和結果。此資訊可讓您更清楚瞭解 Web 應用程式的執行方式，並可讓您識別和診斷各種問題。您也可以設定警示來監控特定閾值，並在達到這些閾值時傳送通知或採取特定動作。

如需有關的詳細資訊 CloudWatch，請參閱 Amazon CloudWatch 使用者指南。

> **Important**
>
> Amazon EC2 API 指標是一項選擇加入功能。您必須要求存取此功能。如需詳細資訊，請參閱 the section called “啟用 Amazon EC2 API 指標”。

目錄

啟用 Amazon EC2 API 指標

請使用下列程序為您的要求存取此功能 AWS 帳戶。

要求存取此功能

1. 打開AWS Support 中心。
2. 選擇建立案例。
3. 選擇 帳戶和帳單。
4. 在「服務」中，選擇「一般信息」和「入門」。
5. 在「類別」中，選擇「使用 AWS 與服務」。
6. 選擇 Next step: Additional information (下一步：其他資訊)。

7. 對於 Subject (主旨)，請輸入 **Request access to Amazon EC2 API metrics**。
8. 對於 Description (說明)，輸入 **Please grant my account access to Amazon EC2 API metrics. Related page: https://docs.aws.amazon.com/AWSEC2/latest/APIReference/monitor.html**。還包括您需要訪問的地區。
9. 選擇下一步驟：立即解決或聯絡我們。
10. 在 [聯絡我們] 索引標籤上，選擇您偏好的聯絡語言和聯絡方式。
11. 選擇提交。

Amazon EC2 API 指標和維度

指標

Amazon EC2 API 指標包含在AWS/EC2/API命名空間中。下表列出可用於 Amazon EC2 API 請求的指標。

指標	描述
ClientErrors	因用戶端錯誤而導致的失敗 API 要求數目。 這些錯誤通常是由用戶端執行的動作所造成，例如在要求中指定不正確或無效的參數，或是代表無權使用動作或資源的使用者使用動作或資源。 單位：計數
RequestLimitExceeded	您的帳戶超過 Amazon EC2 API 允許的最大請求率的次數。 Amazon EC2 API 請求會受到限制，以協助維持服務的效能。如果您的請求已被限制，則會Client.RequestLimitExceeded 收到錯誤。 單位：計數
ServerErrors	因內部伺服器錯誤而導致的失敗 API 要求數目。 這些錯誤通常是由 AWS 伺服器端錯誤、例外狀況或失敗所造成。

指標	描述
	單位：計數
SuccessfulCalls	成功的 API 要求數目。 單位：計數

維度

您可以跨所有 EC2 API 動作篩選 Amazon EC2 指標資料。如需維度的詳細資訊，請參閱 Amazon CloudWatch 概念。

指標資料保留

Amazon EC2 API 指標會以 1 分鐘的間隔傳送到 CloudWatch 。 CloudWatch 如下所示保留測量結果資料：

- 含少於 60 秒期間 (1 分鐘) 的資料點可供使用 15 天。
- 期間為 300 秒（5 分鐘）的資料點可使用 63 天。
- 期間為 3600 秒（1 小時）的資料點可使用 455 天（15 個月）。

監控代表您提出的要求

代表您的 AWS 服務發出的 API 請求 (例如透過服務連結角色發出的請求) 不會計入您的 API 節流限制，也不會為您的帳戶傳送指標至 Amazon CloudWatch 。無法使用監視這些要求 CloudWatch。

第三方服務供應商代表您發出的 API 請求會計入您的 API 節流限制，並且會將指標傳送給 Amazon 給您 CloudWatch 的帳戶。您可以使用來監視這些要求 CloudWatch。

帳單

適用標準 CloudWatch 定價和費用。使用 Amazon EC2 API 指標不收取額外費用。如需詳細資訊，請參閱 Amazon CloudWatch 定價。

與 Amazon 合作 CloudWatch

內容

檢視 CloudWatch 量度

使用下列程序來檢視 Amazon EC2 API 指標。

必要條件

您必須為您的帳戶啟用對 Amazon EC2 API 指標的存取權。如需詳細資訊，請參閱 the section called “啟用 Amazon EC2 API 指標”。

使用主控台檢視 Amazon EC2 API 指標

1. 請在以下位置開啟 CloudWatch 主控台。 https://console.aws.amazon.com/cloudwatch/
2. 在導覽窗格中，選擇「量度」，「所有量度」。
3. 在「瀏覽」標籤上，選擇 EC2/API 度量命名空間。
4. 若要檢視指標，請選取指標維度。

使用命令列檢視 Amazon EC2 API 指標

請使用以下其中一個命令：

- 清單量度 ()AWS CLI

```
aws cloudwatch list-metrics --namespace "AWS/EC2/API"
```

- 獲得-CW () MetricListAWS Tools for Windows PowerShell

```
Get-CWMetricList -Namespace "AWS/EC2/API"
```

建立 CloudWatch 鬧鐘

您可以建立 CloudWatch 警示，在警示狀態變更時傳送 Amazon SNS 訊息。警示會在您指定的期間監看單一指標。它會根據在數個期間內指定臨界值相對於指定臨界值的量度值，傳送通知給 SNS 主題。

例如，您可以建立警示來監控因伺服器端錯誤而失敗的 DescribeInstances API 要求數目。當 DescribeInstances API 要求失敗次數在 5 分鐘內達到 10 個伺服器端錯誤閾值時，下列警示會傳送電子郵件通知。

必要條件

您必須為您的帳戶啟用對 Amazon EC2 API 指標的存取權。如需詳細資訊，請參閱 the section called “啟用 Amazon EC2 API 指標”。

若要針對 Amazon EC2 DescribeInstances API 請求伺服器錯誤建立警示

1. 請在以下位置開啟 CloudWatch 主控台。 https://console.aws.amazon.com/cloudwatch/
2. 在導覽窗格中，選擇 Alarms (警示)、All alarms (所有警示)。
3. 選擇 Create alarm (建立警示)。
4. 選擇 「選取測量結果」，然後指定下列項目：
   a. 選擇 EC2/API。
   b. 選擇「每個動作量度」。
   c. 選取與ServerErrors測量結果名稱位於DescribeInstances相同資料列旁邊的核取方塊。
   d. 選擇選取指標。
5. Specify metric and conditions (指定指標和條件) 頁面隨即出現，顯示您所選取指標和統計資料的圖形及其他資訊。
   a. 在「公制」下，指定下列項目：
      i. 在 Statistic (統計資料) 中選擇 Sum (總和)。
      ii. 對於「期間」，確認已選取 5 分鐘。
   b. 在 Conditions (條件) 下，指定以下內容：
      i. 對於閾值類型，選擇靜態。
      ii. 對於「無論何時」 ServerErrors ，請選擇「大於/等於 >=」。
      iii. 對於比... 」中，輸入 10。

c. 選擇下一步。

6. Configure actions (設定動作) 頁面隨即顯示。

   - 在「通知」下，指定下列項目：

     i. 對於 A lam 狀態觸發，請選擇「在警報中」。

     ii. 對於選取 SNS 主題，選擇選取現有 SNS 主題或建立新主題，然後完成通知的必要欄位。

     iii. 選擇下一步。

7. 這時系統顯示「添加名稱和描述」頁。

   a. 在 [警報名稱] 中，輸入鬧鐘的名稱。名稱只能包含 ASCII 字元。

   b. 對於鬧鐘說明，請輸入警報的選擇性說明。

   c. 選擇下一步。

8. 這時系統顯示「預覽和創建」頁。確認資訊正確無誤，然後選擇 [建立警示]。

如需詳細資訊，請參閱 Amazon 使用 CloudWatch 者指南中的使用 Amazon CloudWatch 警示。

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。