AWSMobile SDK for Unity

デベロッパーガイド

Table of Contents

-AWSMobile SDK for Unity がAWS SDK for .NET。このガイドでは、Mobile SDK for Unity のアーカイブバージョンを参照します。詳細については、「」を参照してください。とはAWSMobile SDK for Unity？ (p. 1)

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。

とはAWSMobile SDK for Unity？

-AWSMobile SDK for Unity は、AWS SDK for .NET。詳細については、「AWS SDK for .NET デベロッパーガイド」を参照してください。

このガイドは更新されなくなりました。Mobile SDK for Unity のアーカイブバージョンを参照しています。

関連ガイドとトピック

- フロントエンドおよびモバイルアプリの開発には、AWS Amplify。
- の使用に関する特別な考慮事項についてはAWS SDK for .NETUnity アプリについては、を参照してください。Unity サポートに関する特別な考慮事項のAWS SDK for .NETデベロッパーガイド。
- 参考のために、アーカイブされたバージョンのAWSMobile SDK for UnityGitHub で。

アーカイブされた参照コンテンツ

アーカイブされた Mobile SDK for Unity には .NET クラス群が含まれており、Unity で作成されたゲームで Unity で記述されたゲームを活用することができます。AWSのサービス。Mobile SDK for Unity で記述されたアプリケーションは、iOS または Android デバイスで実行することができます。

サポート対象AWSサービスには、以下が含まれます。

- Amazon Cognito
- Amazon DynamoDB
- AWS Identity and Access Management (IAM)
- Amazon Kinesis Data Streams
- AWS Lambda
- Amazon Mobile Analytics
- Amazon Simple Email Service (Amazon SES)
- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Queue Service (Amazon SQS)
- Amazon Simple Storage Service (Amazon S3)

これらのサービスでは、ユーザーの認証、プレーヤーデータおよびゲームデータの保存、クラウド内へのオブジェクトの保存、プッシュ通知の送信、使用データの収集を行うことができます。

Compatibility

Mobile SDK for Unity v3 は、Unity バージョン 4.6 以降と互換性があります。

Mobile SDK for Unity では、プロジェクトに組み込む際にコードを変更する必要がなくなりました。これらの変更の詳細については、「」を参照してください。の改良点AWSMobile SDK for Unityフロントエンド Web & モバイルのブログを参照してください。

Mobile SDK for Unity のダウンロード

Unity 用モバイル SDK を.zip ファイルとしてダウンロードすることもできますここ:。

Mobile SDK for Unity には何が含まれていますか。

Mobile SDK for Unity の NuGet パッケージ、サンプル、その他のファイルの完全なリストは、「」を参照してください。AWS SDK for .NETGitHub で。

AWS Mobile SDK for Unity のセットアップ

AWS Mobile SDK for Unity の使用を開始するには、SDK をセットアップして新しいプロジェクトを作成したり、SDK を既存のプロジェクトと統合することができます。また、SDK の仕組みを確認するために、クローンを作成して、サンプルを実行することもできます。

Prerequisites

AWS Mobile SDK for Unity を使用するには、以下が必要になります。

- AWS アカウント
- Unity バージョン 4.x または 5.x (Unity 4.6.4p4 または Unity 5.0.1p3 以降が必要です。iOS 上で実行するアプリケーションを作成する場合は、64 ビット)

前提条件を完了した後、開始するには次の手順を実行する必要があります。

1. AWS Mobile SDK for Unity のダウンロード。
2. AWS Mobile SDK for Unity の設定。
3. Amazon Cognito を使用して AWS 認証情報を取得します。

ステップ 1: AWS Mobile SDK for Unity のダウンロード

最初に、AWS Mobile SDK for Unity をダウンロードします。SDK の各パッケージは、パッケージの名前に基づいて、対応する AWS のサービスを使用する必要があります。たとえば、aws-unity-sdk-dynamodb-2.1.0.0.unitypackage パッケージは、AWS DynamoDB サービスの呼び出しに使用されます。すべてのパッケージをインポートすることも、使用するパッケージだけをインポートすることもできます。

1. Unity エディタを開き、新しい空のプロジェクトを作成し、デフォルト設定を使用します。
2. [Assets (アセット)] > [Import Package (パッケージのインポート)] > [Custom Package (カスタムパッケージ)] を選択します。
3. [Import (インポート)] パッケージダイアログで、使用する .unitypackage ファイルに移動して選択します。
4. [Importing Package (パッケージをインポート)] ダイアログで、すべての項目が選択されていることを確認し、[Import (インポート)] をクリックします。

ステップ 2: AWS Mobile SDK for Unity の設定

シーンの作成

AWS Mobile SDK for Unity を使用して作業する場合、モノ動作クラスの `Start` または `Awake`メソッドに次のコード行を含めることで開始できます。

```
UnityInitializer.AttachToGameObject(this.gameObject);
```

[File] メニューから [New Scene] を選択してシーンを作成します。

AWS SDK for Unity には、サポートしている各 AWS サービスのクライアントクラスが含まれています。これらのクライアントは、awsconfig.xml という名前のファイルを使用して設定されます。次のセクションでは、awsconfig.xml ファイルで最も一般的に使用される設定について説明します。これらの設定の詳細については、Unity SDK API リファレンスを参照してください。

デフォルトの AWS サービスリージョンの設定

すべてのサービスクライアントのデフォルトのリージョンを設定するには:

```
<aws region="us-west-2" />
```

これにより、Unity SDK のすべてのサービスクライアントのデフォルトのリージョンが設定されます。この設定は、次のように、サービスクライアントのインスタント作成時に、リージョンを明示的に指定すると上書きできます。

```
IAmazonS3 s3Client = new AmazonS3Client(<credentials>,RegionEndpoint.USEast1);
```

ログ情報の記録

次のように、ログ設定を指定します。

```
<logging logTo="UnityLogger"
         logResponses="Always"
         logMetrics="true"
         logMetricsFormat="JSON" />
```

この設定は、Unity のログを設定するために使用されます。`UnityLogger` にログを記録すると、フレームワーク内部で Debug Log に出力されます。HTTP レスポンスをログする場合は、logResponses フラグを設定します。値は、Always、Never、または OnError のいずれかです。また、logMetrics プロパティを使用して HTTP リクエストのパフォーマンスメトリクスをログに記録したり、LogMetricsFormat プロパティを使用してログフォーマットを指定したり、有効な値を JSON または標準とすることもできます。

次の例では、awsconfig.xml ファイルで最も一般的に使用される設定について説明します。特定のサービス設定の詳細については、以下のサービスセクションを参照してください。

```
<?xml version="1.0" encoding="utf-8"?>
<aws region="us-west-2"
    <logging logTo="UnityLogger"
             logResponses="Always"
             logMetrics="true"
             logMetricsFormat="JSON" />
/>
```

link.xml ファイルの操作

SDK は、プラットフォーム固有のコンポーネントに対してリフレクションを使用します。IL2CPP スクリプトバックエンドを使用している場合、iOS では常に `strip bytecode` が有効になっているため、次のエントリを使用してアセンブリルートに `link.xml` ファイルを作成する必要があります。

```
<linker>
<!-- if you are using AWSConfigs.HttpClient.UnityWebRequest option-->
<assembly fullname="UnityEngine">
    <type fullname="UnityEngine.Networking.UnityWebRequest" preserve="all" />
    <type fullname="UnityEngine.Networking.UploadHandlerRaw" preserve="all" />
    <type fullname="UnityEngine.Networking.UploadHandler" preserve="all" />
    <type fullname="UnityEngine.Networking.DownloadHandler" preserve="all" />
    <type fullname="UnityEngine.Networking.DownloadHandlerBuffer" preserve="all" />
</assembly>
<assembly fullname="mscorlib">
    <namespace fullname="System.Security.Cryptography" preserve="all"/>
</assembly>
<assembly fullname="System">
    <namespace fullname="System.Security.Cryptography" preserve="all"/>
</assembly>
<assembly fullname="AWSSDK.Core" preserve="all"/>
<assembly fullname="AWSSDK.CognitoIdentity" preserve="all"/>
<assembly fullname="AWSSDK.SecurityToken" preserve="all"/>
add more services that you need here...
</linker>
```

ステップ 3: Amazon Cognito を使用してアイデンティティプール ID を取得する

モバイルアプリケーションで AWS サービスを使用するには、Amazon Cognito Identity を使用して ID プールの ID を取得する必要があります。Amazon Cognito を使用して ID プールの ID を取得すると、アプリケーションにプライベートな認証情報を埋め込むことなく、AWS サービスにアクセスできるようになります。また、アクセス権限を設定して、ユーザーがアクセスできる AWS サービスを管理することができます。

Amazon Cognito の使用を開始するには、ID プールを作成する必要があります。ID プールはアカウントに固有のユーザー ID データのストアです。各 ID プールには、アプリケーションのユーザーがアクセスできる AWS サービスを指定するために使用できる、設定可能な IAM ロールがあります。通常、開発者はアプリケーションごとに 1 つの ID プールを使用します。ID プールの詳細については、Amazon Cognito 開発者ガイドを参照してください。

アプリケーション用の ID プールを作成するには:

1. Amazon Cognito コンソールにログインし、[新しい ID プールの作成] をクリックします。
2. ID プールの名前を入力し、未認証 ID へのアクセスを有効にするチェックボックスをオンにします。[プールの作成] をクリックして、ID プールを作成します。
3. [Allow (許可)] をクリックして、ID プールに関連付けられた 2 つのデフォルトロール (1 つは未認証ユーザー用、もう 1 つは認証されたユーザー用) を作成します。これらのデフォルトのロールは、Cognito Sync および Mobile Analytics への ID プールアクセスを提供します。

次のページには、Cognito Identity と Unity アプリケーションを簡単に統合できるように、認証プロバイダーを作成するコードが表示されます。認証情報プロバイダーオブジェクトを、使用している AWS クライアントのコンストラクタに渡します。コードは次のようになります。

```
CognitoAWSCredentials credentials = new CognitoAWSCredentials (
    "IDENTITY_POOL_ID", // Identity Pool ID
    RegionEndpoint.USEast1 // Region
);
```

次のステップ

- 開始方法: ReadAWS Mobile SDK for Unity の使用開始 (p. 7)SDK に含まれているサービスの詳細な概要を得るには、
- デモを実行する:「Our」サンプルの Unity アプリケーションの一般的なユースケースについて説明します。このサンプルアプリを実行して、前述の &SDK for Unity をセットアップするには、各サンプルの README ファイルに含まれる指示に従います。
- API リファレンスを読む: を表示するAPI リファレンスAWS Mobile SDK for Unity のことです。
- 質問をしましょう: に質問を投稿するAWS Mobile SDK ForumまたはGithub で課題を開く。

AWS Mobile SDK for Unity の使用開始

このページでは、AWS Mobile SDK for Unity の各 AWS サービスの概要と、Unity サンプルの設定方法について説明します。以下のサービスの使用を開始する前に、「AWS Mobile SDK for Unity のセットアップ (p. 3)」ページのすべての手順を完了する必要があります。

Amazon Cognito ID

AWS を呼び出すには、必ず AWS 認証情報が必要です。アプリに AWS 認証情報を提供するには、アプリケーションへの認証情報をハードコードするのではなく、Amazon Cognito ID を使用することをお勧めします。「AWS Mobile SDK for Unity をセットアップする (p. 3)」の手順に従って、Amazon Cognito 経由で AWS 認証情報を取得します。

Cognito では、パブリックログインプロバイダー (Amazon、Facebook、Twitter、Google など) や、OpenID Connect をサポートするプロバイダーを使用して、ユーザーを認証することができます。また、Cognito では、未認証ユーザーのアクセスを設定することもできます。Cognito では、Identity and Access Management (IAM) ロールを使用して指定した制限付きアクセス権に一時的な認証情報を提供します。Cognito は、IAM ロールに関連付けられた新しい ID プールを作成して構成されます。IAM ロールは、アプリケーションでアクセスできるリソースとサービスを指定します。

Cognito ID の使用を開始するには、Amazon Cognito 開発者ガイドを参照してください。

Amazon Cognito Sync

Cognito Sync では、ユーザーの好みやゲームの状態などのエンドユーザーのデータを AWS クラウドに簡単に保存できるため、これらのデータはユーザーが使用しているデバイスに関係なく利用可能になります。Cognito はこのデータをローカルに保存することもできるため、インターネット接続が利用できなくてもアプリを動作させることができます。インターネット接続が利用可能になると、アプリとローカルデータをクラウドに同期できます。

Cognito Sync の使用を開始するには、Amazon Cognito 開発者ガイドを参照してください。

CognitoSyncManager サンプルを使用する

[Project (プロジェクト)] ペインで、[Assets (アセット)]/[AWSSDK]/[examples (例)]/[CognitoSync] に移動して、右側のペインの [CognitoSync] を選択してシーンを開きます。

サンプルを実行するには、エディタ画面上部の再生ボタンをクリックします。アプリが実行されると、いくつかのプレーヤー情報を入力できるテキストボックスとボタンが表示されます。この下には、プレーヤーの情報をローカルに保存したり、ローカルプレーヤー情報を Cognito クラウドと同期させたり、Cognito Cloud からプレーヤー情報を更新したり、ローカルプレーヤー情報を削除する一連のボタンがあります。各ボタンを押して操作を実行します。サンプルゲーム画面の上部にフィードバックが表示されます。

CognitoSyncManager サンプルを設定するには、Cognito ID プールの ID を指定する必要があります。この値を指定するには、Unity エディタで、[Heirarchy] ペインの [SyncManager] を選択し、[Inspector Pane] の [IDENTITY_POOL_ID] テキストボックスに入力します

Note

CognitoSyncManager サンプルには、Facebook の ID プロバイダーを使用して「USE_FACEBOOK_LOGIN」マクロを検索する方法を示すコードが含まれています。Facebook SDK for Unity を使用する必要があります。詳細については、Facebook SDK for Unity を参照してください。

DynamoDB

Amazon DynamoDB は、拡張性と可用性に優れた、費用効果の高い、高速な非リレーショナルデータベースサービスです。DynamoDB により、データストレージに対して低いレイテンシーと予測可能なパフォーマンスを維持しながら、従来の拡張性の限界を排除できます。

AWS SDK for Unity は、DynamoDB を操作するための低レベルライブラリと高レベルライブラリの両方を提供します。高レベルライブラリには、DynamoDB Object Mapper が含まれており、クライアント側のクラスを DynamoDB テーブルにマップできます。さまざまな作成、読み取り、更新、および削除 (CRUD) 操作を実行し、さらにクエリを実行します。DynamoDB Object Mapper を使用すると、オブジェクトをクラウドに格納する簡単で読みやすいコードを記述できます。

DynamoDB の詳細については、DynamoDB 開発者ガイドを参照してください。

Unity アプリケーションから DynamoDB を使用する方法の詳細については、「Amazon DynamoDB (p. 21)」を参照してください。

DynamoDB サンプルの使用

[Project (プロジェクト)] ペインで、[Assets (アセット)]/[AWSSDK/] [examples (例)]/[DynamoDB] に移動します。このサンプルは、以下のシーンで構成されています。

- DynamoDBExample - アプリの最初のシーン
- LowLevelDynamoDbExample - 低レベルの DynamoDBD API を使用した例
- TableQueryAndScanExample - クエリの実行方法の例
- HighLevelExample- 高レベルの DynamoDB API を使用した例

Build Settings ダイアログ (File.Build Settings を選択して開く) を使用して、これらのシーンをビルドに追加します (上に表示されている順番で)。このサンプルの 4 つのテーブルを作成します。ProductCatalog、フォーラム、スレッド、返信。

サンプルを実行するには、エディタ画面上部の再生ボタンをクリックします。アプリが実行されると、いくつかのボタンが表示されます。

- 低レベルテーブルのオペレーション - テーブルの作成、リスト作成、更新、記述、および削除の方法を示します。
- 中間レベル Query&Scan のオペレーション - クエリを実行する方法を示します。
- 高レベル Object Mapper - オブジェクトを作成、更新、削除する方法を示します。

Mobile Analytics

Amazon Mobile Analytics を使用すると、顧客の行動を追跡し、メトリクスを集約して、データを視覚化できるほか、有意義なパターンを特定することができます。AWS SDK for Unity では、Amazon Mobile

Analytics サービスとの統合を提供します。Mobile Analytics の詳細については、Mobile Analytics ユーザーガイドを参照してください。Unity アプリケーションから Mobile Analytics を使用する方法の詳細については、「Amazon Mobile Analytics (p. 14)」を参照してください。

Mobile Analytics の設定

Mobile Analytics では、awsconfig.xml で設定可能な設定の一部を定義します。

```
<mobileAnalytics sessionTimeout = "5"
                 maxDBSize = "5242880"
                 dbWarningThreshold = "0.9"
                 maxRequestSize = "102400"
                 allowUseDataNetwork = "false"/>
```

- sessionTimeout - アプリケーションがバックグラウンドに移行してからセッションが終了するまでの時間間隔。
- maxDBSize - SQLIte データベースのサイズ。データベースの最大サイズに達すると、それ以降のイベントは削除されます。
- dbWarningThreshold - データベースのサイズ制限。この値に達すると、警告ログが生成されます。
- maxRequestSize - HTTP リクエストで Mobile Analytics サービスに送信する必要がある、バイト単位のリクエストの最大サイズ。
- allowUseDataNetwork - データネットワーク上でセッションイベントを送信するかどうかを指定するブール値。

Mobile Analytics サンプルを使用する

[Project (プロジェクト)] ペインで、[Assets (アセット)]/[AWSSDK]/[examples (例)]/[Mobile Analytics] に移動して、右側のペインの [Amazon Mobile Analytics Sample (Amazon Mobile Analytics のサンプル)] を選択してシーンを開きます。サンプルを使用するには、Amazon Mobile Analytics コンソールを使用してアプリを追加する必要があります。Mobile Analytics コンソールを使用する方法の詳細については、Amazon Mobile Analytics ユーザーガイドを参照してください。

実行する前にサンプルを設定するには、次の手順に従います。

1. [AmazonMobileAnalyticsSample] ゲームオブジェクトを選択します。
2. [App Id (アプリケーション ID)] フィールドに、アプリ ID (Amazon Mobile Analytics コンソールで作成済み) を指定します。
3. [Cognito Identity Pool Id (Cognito ID プール ID)] フィールドに Cognito ID プール ID (Amazon Cognito コンソールを使用して作成済み) を指定します。
4. 認証されたロールと未認証ロールに Mobile Analytics サービスにアクセスする権限があることを確認します。IAM ロールにポリシーを適用する方法の詳細については、「ロールの管理」を参照してください。

サンプルアプリケーションを実行する場合、イベントはすぐにバックエンドサービスに送信されない可能性があることに注意してください。バックグラウンドスレッドはイベントをローカルでバッファリングした後、一定の間隔 (デフォルト値は 60 秒) で Amazon Mobile Analytics バックエンドにバッチで送信して、ゲームのパフォーマンスに悪影響を与えないようにします。データに対して Amazon Mobile Analytics の実行の処理が複雑なため、イベントおよび対応するレポートは、最初の送信から最大 60 分経過するまで AWS コンソールに表示されないことがあります。

Amazon Mobile Analytics が提供するレポートの詳細については、「レポートとモバイルメトリクス」を参照してください。

Amazon S3

Amazon Simple Storage Service (Amazon S3) を使用すると、開発者および IT チームは、耐久性および耐障害性が高く、セキュアなオブジェクトストレージを使用できるようになります。Unity では、S3 を使用して、ゲームで使用されている画像、ビデオ、音楽、その他のデータを保存、一覧表示、取得することができます。

S3 の詳細については、「Amazon S3」および「S3 の使用開始」を参照してください。

Unity アプリケーションから S3 を使用する方法の詳細については、「Amazon Simple Storage Service (S3) (p. 17)」を参照してください。

S3 Default Signature の設定

デフォルトの S3 署名は、次のように構成されています。

```
<s3 useSignatureVersion4="true" />
```

これは、S3 リクエストに対して署名バージョン 4 を使用する必要があるかどうかを指定するために使用されます。

S3 サンプル の使用

[Project (プロジェクト)] ペインで、[Assets (アセット)]/[AWSSDK]/[examples (例)]/[S3] に移動して、右側のペインの [S3Example] シーンを選択してシーンを開きます。このサンプルは、バケットの一覧表示、バケット内のオブジェクトの一覧表示、オブジェクトのバケットへのポスト、およびバケットからのオブジェクトのダウンロード方法を示しています。実行する前にサンプルを設定するには、次の手順に従います。

1. [Hierarchy (階層)] ペインで [S3] ゲームオブジェクトを選択します。
2. [Inspector (インスペクター)] ペインで、[S3BucketName] と [SampleFileName] の値を入力します。S3BucketName はサンプルで使用されるバケットの名前で、S3SampleFileName はサンプルが指定された S3 バケットにアップロードされるファイルの名前です。
3. 認証されたロールと未認証ロールに、アカウントの S3 バケットにアクセスする権限があることを確認します。IAM ロールにポリシーを適用する方法の詳細については、「ロールの管理」を参照してください。

サンプルを実行するには、エディタ画面上部の再生ボタンをクリックします。アプリが実行されると、いくつかのボタンが表示されます。

- Get Objects - AWS アカウントのすべてのバケット内のすべてのオブジェクトのリストを取得します。
- Get Buckets - AWS アカウントのすべてのバケット内のすべてのオブジェクトのリストを取得します。
- Post Object- 指定された S3 バケットにオブジェクトをアップロードします。
- Delete Object- 指定された S3 バケットからオブジェクトをすべて削除します。

サンプルゲーム画面の上部にフィードバックが表示されます。

Amazon Simple Notification Service

Amazon Simple Notification Service は、高速かつ柔軟な完全マネージド型のプッシュ通知サービスです。このサービスを使用すると、個々のメッセージを送信したり、多数の受信者にメッセージをファンアウト

したりできます。Amazon Simple Notification Service により、簡単かつコスト効率の高い方法で、モバイルデバイスユーザーおよびメール受信者にプッシュ通知を送信したり、他の分散サービスにメッセージを送信したりできます。Amazon Simple Notification Service を開始するには、「Amazon Simple Notification Service (p. 26)」を参照してください。

AWS Lambda

AWS Lambda は、リクエストまたはイベントに応答してコードを実行するコンピューティングサービスです。コンピューティングリソースを自動的に管理して、新規情報に迅速に対応するアプリケーションを容易に構築できるようにします。AWS Lambda 関数は、モバイル、IoT、ウェブアプリから直接呼び出して、同期的にレスポンスを返すことができるため、インフラストラクチャをプロビジョニングまたは管理することなく、モバイルアプリ向けにスケーラブルかつセキュアで可用性が高いバックエンドを容易に作成することができます。詳細については、「AWS Lambda (p. 32)」を参照してください。

Amazon Cognito ID

Amazon Cognito ID とは

Amazon Cognito ID を使用すると、ユーザーに固有の ID を作成し、Amazon S3 や Amazon DynamoDB などの AWS リソースへの安全なアクセスを認証できます。Amazon Cognito ID は、パブリック ID プロバイダー (Amazon、Facebook、Twitter/Digits、Google、OpenID Connect と互換性のあるプロバイダー) および未認証 ID をサポートします。また、Cognito は、開発者が認証した ID をサポートします。引き続き Amazon Cognito Sync を使用してユーザーデータを同期します。AWS リソースにアクセスすることにより、独自の既存の認証プロセスを通じてユーザー登録や認証ができます。

Cognito ID の詳細については、Amazon Cognito 開発者ガイドを参照してください。

Cognito 認証リージョンの可用性の詳細については、「Amazon Cognito ID リージョンの可用性」を参照してください。

パブリックプロバイダーを使用してユーザーの認証

Amazon、Facebook、Twitter/Digits、Google などのパブリック ID プロバイダーを使用してユーザーを認証する方法については、Amazon Cognito 開発者ガイドの「外部プロバイダー」を参照してください。

開発者が認証した ID の使用

開発者が認証した ID の詳細については、Amazon Cognito 開発者ガイドの「開発者が認証した ID」を参照してください。

Amazon Cognito Sync

Cognito Sync は、アプリケーション関連のユーザーデータのデバイス間の同期を有効にする、AWS サービスとクライアントライブラリです。Cognito Sync API を使用すると、デバイス間でユーザーデータを同期することができます。Cognito Sync をアプリで使用するには、プロジェクトに AWS Mobile SDK for Unity 含む必要があります。

Amazon Cognito Sync をアプリケーションに統合する方法については、Amazon Cognito Sync 開発者ガイドを参照してください。

Amazon Mobile Analytics

Amazon Mobile Analytics を使用すると、顧客の行動を追跡し、メトリクスを集約して、データを視覚化できるほか、有意義なパターンを特定することができます。Mobile Analytics の詳細については、「AWS Mobile Analytics」を参照してください。

Amazon Mobile Analytics の統合

以下のセクションでは、Mobile Analytics をアプリに統合する方法について説明します。

Mobile Analytics コンソールでアプリを作成する

Amazon Mobile Analytics コンソールに移動して、アプリを作成します。`appId` 値は、後で必要になるため記録しておきます。

> Note
>
> コンソールでの操作方法については、Amazon Mobile Analytics ユーザーガイドを参照してください。

Mobile Analytics コンソールでアプリを作成する場合は、Cognito ID プールの ID を指定する必要があります。新しい Cognito ID プールと ID を作成する方法については、Cognito ID 開発者ガイドを参照してください。

Mobile Analytics をアプリケーションに統合する

Unity から Mobile Analytics にアクセスするには、次のステートメントを使用します。

```
using Amazon.MobileAnalytics.MobileAnalyticsManager;
using Amazon.CognitoIdentity;
```

Amazon Cognito を使用するベストプラクティスは、一時的な AWS 認証情報をアプリケーションに提供することです。アプリは、これらの認証情報を使用して AWS リソースにアクセスします。認証情報プロバイダーを作成するには、「Amazon Cognito ID (p. 12)」の手順に従います。

次の情報を使用して、MobileAnalyticsManager をインスタンス化します。

- cognitoIdentityPoolId - アプリの Cognito ID プールの ID
- cognitoRegion - Cognito ID プールのリージョン (例: 「RegionEndpoint.USEast1」)
- region - Mobile Analytics サービスのリージョン (例: 「RegionEndpoint.USEast1」)
- appId - アプリを追加時に Mobile Analytics コンソールによって生成される値

MobileAnalyticsClientContextConfig を使用して、以下のコードスニペットで示されている **`MobileAnalyticsManager`** インスタンスを初期化します。

```
// Initialize the MobileAnalyticsManager
void Start()
{
    // ...
    analyticsManager = MobileAnalyticsManager.GetOrCreateInstance(
```

```
        new CognitoAWSCredentials(<cognitoIdentityPoolId>, <cognitoRegion>),
        <region>,
        <appId>);
    // ...
}
```

Note

アプリ ID は、アプリの作成ウィザードで生成されます。これらの値はいずれも、Mobile Analytics コンソールの値と一致する必要があります。

`appId` は、Mobile Analytics コンソールのデータをグループ化するために使用されます。Mobile Analytics コンソールでアプリ作成後にアプリ ID を検索するには、Mobile Analytics コンソールに移動して、画面右上隅の歯車アイコンをクリックします。これにより、登録済みのアプリとアプリ ID がすべて表示されたアプリの管理ページが表示されます。

収益化イベントの記録

SDK for Unity には、`MonetizationEvent` クラスが用意されています。このクラスを使用して、モバイルアプリケーション内の購入を追跡する収益化イベントを生成できます。収益化イベントの作成方法を示すコードスニペットは、以下のとおりです。

```
// Create the monetization event object
MonetizationEvent monetizationEvent = new MonetizationEvent();

// Set the details of the monetization event
monetizationEvent.Quantity = 3.0;
monetizationEvent.ItemPrice = 1.99;
monetizationEvent.ProductId = "ProductId123";
monetizationEvent.ItemPriceFormatted = "$1.99";
monetizationEvent.Store = "Your-App-Store";
monetizationEvent.TransactionId = "TransactionId123";
monetizationEvent.Currency = "USD";

// Record the monetiziation event
analyticsManager.RecordEvent(monetizationEvent);
```

カスタムイベントの記録

Mobile Analytics では、カスタムイベントを定義できます。カスタムイベントは全体的に定義されています。そのため、アプリやゲーム固有のユーザーアクションを追跡しやすくなります。カスタムイベントの詳細については、「カスタムイベント」を参照してください。この例では、ゲームアプリとして、ユーザーがレベルを達成するとイベントを記録します。新しい `AmazonMobileAnalyticsEvent` インスタンスを作成して、「LevelComplete」イベントを作成します。

```
CustomEvent customEvent = new CustomEvent("LevelComplete");

// Add attributes
customEvent.AddAttribute("LevelName","Level1");
customEvent.AddAttribute("CharacterClass","Warrior");
customEvent.AddAttribute("Successful","True");

// Add metrics
customEvent.AddMetric("Score",12345);
customEvent.AddMetric("TimeInLevel",64);

// Record the event
analyticsManager.RecordEvent(customEvent);
```

セッションの記録

アプリケーションがフォーカスを失った場合は、セッションを一時停止することができます。`OnApplicationFocus` で、アプリが一時停止中かどうかを確認します。次のコードスニペットに示すように、一時停止中の場合は `PauseSession`、それ以外の場合は `ResumeSession` を呼び出します。

```
void OnApplicationFocus(bool focus)
{
    if(focus)
    {
        analyticsManager.ResumeSession();
    }
    else
    {
        analyticsManager.PauseSession();
    }
}
```

デフォルトでは、5 秒以内にアプリから焦点を切り替えて戻した場合、セッションは再開されます。ユーザーが焦点から離れて 5 秒以上経過した場合は、新しいセッションが作成されます。この設定は、awsconfig.xml ファイルで設定できます。詳細については、「AWS Mobile SDK for Unity の開始方法 (p. 7)」の「Mobile Analytics の設定」セクションを参照してください。

Amazon Simple Storage Service (S3)

Amazon Simple Storage Service (Amazon S3) を使用すると、開発者および IT チームは、耐久性および耐障害性が高く、セキュアなオブジェクトストレージを使用できるようになります。Unity の開発者は、S3 を活用して、ゲームで使用されるアセットを動的にロードすることができます。これにより、最初から迅速にアプリストアからのダウンロードを行うことができます。

S3 の詳細については、「Amazon S3」を参照してください。

AWS S3 リージョンの可用性の詳細については、「AWS サービスリージョンの可用性」を参照してください。

> Note
>
> このドキュメントの一部サンプルでは、ResultText と呼ばれる可変のテキストボックスを使用して、トレース出力を表示していることを前提としています。

S3 バケットの作成と設定

Amazon S3 は、特定のリージョンに存在するクラウドストレージコンテナである Amazon S3 バケットにリソースを保存します。Amazon S3 バケットの名前は、それぞれグローバルに一意である必要があります。Amazon S3 コンソールを使用して、バケットを作成します。

S3 バケットの作成

1. Amazon S3 コンソールにサインインし、[バケットを作成する] をクリックします。
2. バケット名を入力してリージョンを選択したら [作成] をクリックします。

S3 のアクセス許可を設定

デフォルトの IAM ロールのポリシーでは、Amazon Mobile Analytics および Amazon Cognito Sync へのアクセス権がアプリケーションに付与されています。Cognito ID プールから Amazon S3 にアクセスできるようにするには、ID プールのロールを変更する必要があります。

1. Identity and Access Management コンソールに移動し、左側のペインの [ロール] をクリックします。
2. 検索ボックスに ID プールの名前を入力します。2 つのロール (未認証ユーザーと認証済みユーザー) が表示されます。
3. 未認証ユーザーのロールをクリックします (unauth を ID プール名に追加)。
4. [ロールポリシーの作成] をクリックして [Policy Generator] を選択し、[選択] をクリックします。
5. [アクセス許可の編集] ページで、以下のイメージに示す設定を入力し、Amazon リソースネーム (ARN) を実際の名前に置き換えます。S3 バケットの ARN は、`arn:aws:s3:::examplebucket/*` のように表示され、バケットを配置するリージョンとバケットの名前で構成されます。以下に示す設定では、ID プールから指定したバケットのすべてのアクションにフルアクセスを付与しています。

1. [ステートメントを追加] ボタンをクリックし、[次のステップ] をクリックします。
2. ウィザードに、生成した設定が表示されます。[ポリシーの適用] をクリックします。

S3 に対するアクセス権の付与の詳細については、「Amazon S3 バケットへアクセス権を付与する」を参照してください。

コンソールからファイルをアップロードする

テストファイルをバケットにアップロードするには、以下のように行います。

1. S3 コンソールのバケットビューで、[アップロード] をクリックします。
2. [ファイルを追加] をクリックし、アップロードするテストファイルを選択します。このチュートリアルでは、イメージ (`myImage.jpg`) をアップロードしていることを前提としています。
3. テストイメージを選択した状態で、[Start Upload (アップロードを開始)] をクリックします。

(オプション) S3 リクエストの署名バージョンを設定する

Amazon S3 とのすべてのやり取りは認証されるか匿名で行われます。AWS では、署名バージョン 4 または署名バージョン 2 のアルゴリズムを使用して、サービスへの呼び出しを認証します。

2014 年 1 月以降に作成されたすべての新しい AWS リージョンでは、署名バージョン 4 のみをサポートしています。ただし、以前のリージョンの多くは、現在も署名バージョン 4 および署名バージョン 2 のリクエストに対応しています。

バケットが、このページの署名バージョン 2 のリクエストをサポートしていないリージョンのいずれかにある場合は、AWSConfigsS3.UseSignatureVersion4 プロパティを「true」に設定する必要があります。

AWS 署名バージョンの詳細については、「リクエストの認証 (AWS 署名バージョン 4)」を参照してください。

Amazon S3 クライアントの作成

Amazon S3 を使用するには、まず、以前に作成した CognitoAWSCredentials インスタンスを参照する AmazonS3Client インスタンスを作成する必要があります。

```
AmazonS3Client S3Client = new AmazonS3Client (credentials);
```

`AmazonS3Client` クラスは、高レベルの S3 API のエントリポイントです。

バケットの一覧表示

AWS アカウントのバケットを一覧表示するには、次のサンプルコードに示すように、`AmazonS3Client.ListBucketsAsync` メソッドを呼び出します。

```
// ResultText is a label used for displaying status information
ResultText.text = "Fetching all the Buckets";
 Client.ListBucketsAsync(new ListBucketsRequest(), (responseObject) =>
 {
     ResultText.text += "\n";
     if (responseObject.Exception == null)
     {
         ResultText.text += "Got Response \nPrinting now \n";
         responseObject.Response.Buckets.ForEach((s3b) =>
```

```
        {
            ResultText.text += string.Format("bucket = {0}, created date = {1} \n",
            s3b.BucketName, s3b.CreationDate);
        });
    }
    else
    {
        ResultText.text += "Got Exception \n";
    }
});
```

オブジェクトのリスト化

バケットのオブジェクトをすべて一覧表示するには、次のサンプルコードに示すように、AmazonS3Client.ListObjectsAsync メソッドを呼び出します。

```
// ResultText is a label used for displaying status information
ResultText.text = "Fetching all the Objects from " + S3BucketName;

var request = new ListObjectsRequest()
{
    BucketName = S3BucketName
};

Client.ListObjectsAsync(request, (responseObject) =>
{
    ResultText.text += "\n";
    if (responseObject.Exception == null)
    {
        ResultText.text += "Got Response \nPrinting now \n";
        responseObject.Response.S3Objects.ForEach((o) =>
        {
            ResultText.text += string.Format("{0}\n", o.Key);
        });
    }
    else
    {
        ResultText.text += "Got Exception \n";
    }
});
```

オブジェクトのダウンロード

オブジェクトをダウンロードするには、バケット名およびキーを指定しながら GetObjectRequest を作成して、Client.GetObjectAsync に対する呼び出しにオブジェクトを渡します。

```
private void GetObject()
{
    ResultText.text = string.Format("fetching {0} from bucket {1}",
    SampleFileName, S3BucketName);
    Client.GetObjectAsync(S3BucketName, SampleFileName, (responseObj) =>
    {
       string data = null;
       var response = responseObj.Response;
       if (response.ResponseStream != null)
       {
          using (StreamReader reader = new StreamReader(response.ResponseStream))
```

```
            {
               data = reader.ReadToEnd();
            }

            ResultText.text += "\n";
            ResultText.text += data;
        }
    });
}
```

GetObjectAsync は、GetObjectRequest のインスタンス、コールバック、AsyncOptions インスタンスを受け取ります。コールバックの型はである必要があります。`AmazonServiceCallback<GetObjectRequest, GetObjectResponse>`。AsyncOptions インスタンスはオプションです。指定した場合、メインスレッドでコールバックが実行されるかどうかを決定します。

オブジェクトのアップロード

オブジェクトをアップロードするには、オブジェクトをストリームに書き込み、新しい PostObjectRequest を作成して、バケット名およびストリームデータを指定します。

AWS SDK for Unity では、WWW HTTP クライアントを使用します。このクライアントは、HTTP PUT オペレーションをサポートしていません。S3 バケットにオブジェクトをアップロードするためには、以下に示すように、S3 のブラウザを使用する必要があります。

```
public void PostObject(string fileName)
{
    ResultText.text = "Retrieving the file";

    var stream = new FileStream(Application.persistentDataPath +
    Path.DirectorySeparatorChar + fileName,
    FileMode.Open, FileAccess.Read, FileShare.Read);

    ResultText.text += "\nCreating request object";
    var request = new PostObjectRequest()
    {
        Bucket = S3BucketName,
        Key = fileName,
        InputStream = stream,
        CannedACL = S3CannedACL.Private
    };

    ResultText.text += "\nMaking HTTP post call";

    Client.PostObjectAsync(request, (responseObj) =>
    {
        if (responseObj.Exception == null)
        {
            ResultText.text += string.Format("\nobject {0} posted to bucket {1}",
            responseObj.Request.Key, responseObj.Request.Bucket);
        }
        else
        {
            ResultText.text += "\nException while posting the result object";
            ResultText.text += string.Format("\n receieved error {0}",
            responseObj.Response.HttpStatusCode.ToString());
        }
    });
}
```

Amazon DynamoDB

Amazon DynamoDB は、拡張性と可用性に優れた、費用効果の高い、高速な非リレーショナルデータベースサービスです。DynamoDB により、データストレージに対して低いレイテンシーと予測可能なパフォーマンスを維持しながら、従来の拡張性の限界を排除できます。DynamoDB に関する詳細については、「Amazon DynamoDB」を参照してください。

AWS Mobile SDK for Unity には、DynamoDB を操作するための高レベルのライブラリが用意されています。低レベルの DynamoDB API に対してリクエストすることもできますが、ほとんどのユースケースにおいて、ハイレベルのライブラリが推奨されています。特に、AmazonDynamoDBClient は高レベルライブラリの便利な部分です。このクラスを使用することで、さまざまな (CRUD) 操作の作成、読み取り、更新、削除、およびクエリの実行を行うことができます。

> Note
>
> このドキュメントの一部サンプルでは、ResultText と呼ばれる可変のテキストボックスを使用して、トレース出力を表示していることを前提としています。

Amazon DynamoDB の統合

Unity アプリケーションで DynamoDB を使用するには、プロジェクトに Unity SDK を追加する必要があります。追加していない場合は、SDK for Unity をダウンロードし、「AWS Mobile SDK for Unity のセットアップ (p. 3)」の手順に従います。アプリケーションに一時的な AWS 認証情報を提供するには、Amazon Cognito ID を使用することをおすすめします。これらの認証情報を使用すると、アプリから AWS サービスおよびリソースにアクセスできるようになります。

DynamoDB をアプリケーションで使用するには、適切なアクセス許可を設定する必要があります。次の IAM ポリシーでは、ユーザーは特定の DynamoDB テーブルの項目を削除、取得、挿入、スキャン、更新することができます。このポリシーは、ARN によって識別されます。

```
{
"Statement": [{
    "Effect": "Allow",
    "Action": [
        "dynamodb:DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:UpdateItem"
    ],
    "Resource": "arn:aws:dynamodb:us-west-2:123456789012:table/MyTable"
}]
}
```

このポリシーは、Cognito ID プールに割り当てられているロールに適用されますが、**`Resource`** 値を DynamoDB テーブルの適切な ARN に置き換える必要があります。Cognito によって、自動的に新しい ID プールのロールが作成されると、IAM コンソールで、このロールにポリシーを適用することができます。

アプリケーションのニーズに応じて、許可されているアクションを追加または削除する必要があります。IAM ポリシーの詳細については、「IAM の使用」を参照してください。DynamoDB 固有のポリシーの詳細については、「IAM を使用して DynamoDB リソースへのアクセスをコントロールする」を参照してください。

DynamoDB テーブルの作成

これでアクセス許可と認証情報がセットアップされたため、アプリケーションの DynamoDB テーブルを作成しましょう。テーブルを作成するには、DynamoDB コンソールに移動し、次の手順に従って操作します。

1. [テーブルの作成] をクリックします。
2. テーブルの名前に `Bookstore` と入力します。
3. プライマリキーのタイプとして、[ハッシュ] を選択します。
4. [数値] を選択し、ハッシュ属性名の `id` を入力します。[次へ] をクリックします。
5. [次へ] を再度クリックして、インデックスの追加をスキップします。
6. 読み込みキャパシティーを `10`、書き込みキャパシティーを `5` に設定します。[次へ] をクリックします。
7. 通知 E メールを入力したら、[次へ] をクリックしてスループットアラームを作成します。
8. [Create] をクリックします。DynamoDB にデータベースが作成されます。

DynamoDB クライアントの作成

アプリで DynamoDB テーブルを操作するには、クライアントが必要です。デフォルトの DynamodDB クライアントは、次のように作成できます。

```
var credentials = new CognitoAWSCredentials(IDENTITY_POOL_ID, RegionEndpoint.USEast1);
AmazonDynamoDBClient client = new AmazonDynamoDBClient(credentials);
DynamoDBContext Context = new DynamoDBContext(client);
```

AmazonDynamoDBClient クラスは、DynamoDB API のエントリポイントです。このクラスには、他のアプリケーション間で、テーブルを作成、記述、更新、削除するインスタンスメソッドがあります。コンテキストによって、クライアントの抽象化レイヤーが追加され、オブジェクト永続性モデルなどの追加の機能を使用できるようになります。

テーブルの説明

DynamoDB テーブルの説明を取得するには、次のコードを使用します。

```
resultText.text +=("\n*** Retrieving table information ***\n");
       var request = new DescribeTableRequest
       {
           TableName = @"ProductCatalog"
       };
       Client.DescribeTableAsync(request, (result) =>
       {
               if (result.Exception != null)
               {
                       resultText.text += result.Exception.Message;
                       Debug.Log(result.Exception);
                       return;
               }
               var response = result.Response;
               TableDescription description = response.Table;
               resultText.text += ("Name: " + description.TableName + "\n");
               resultText.text += ("# of items: " + description.ItemCount + "\n");
               resultText.text += ("Provision Throughput (reads/sec): " +
```

```
                description.ProvisionedThroughput.ReadCapacityUnits + "\n");
            resultText.text += ("Provision Throughput (reads/sec): " +
                description.ProvisionedThroughput.WriteCapacityUnits + "\n");

        }, null);
    }
```

この例では、クライアントと DescribeTableRequest オブジェクトを作成し、テーブルの名前を **`TableName`** プロパティに割り当て、リクエストオブジェクトを AmazonDynamoDBClient オブジェクトの DescribeTableAsync メソッドに渡します。また、DescribeTableAsync では、非同期操作が完了すると委任されます。

> Note
>
> AmazonDynamoDBClient の非同期メソッドでは、非同期操作が完了すると必ず、委任が呼び出されます。

オブジェクトの保存

オブジェクトを DynamoDB に保存するには、AmazonDynamoDBClient オブジェクトの SaveAsync<T> メソッドを使用します。ここで、T は、保存するオブジェクトのタイプです。

データベースを「Bookstore」としたので、そのテーマに沿って、本に関する属性を記録するデータモデルを実装します。データモデルを定義するクラスは、次のとおりです。

```
[DynamoDBTable("ProductCatalog")]
    public class Book
    {
        [DynamoDBHashKey]   // Hash key.
        public int Id { get; set; }
        [DynamoDBProperty]
        public string Title { get; set; }
        [DynamoDBProperty]
        public string ISBN { get; set; }
        [DynamoDBProperty("Authors")]    // Multi-valued (set type) attribute.
        public List<string> BookAuthors { get; set; }
    }
```

もちろん、実店舗の書店アプリケーションでは、作者や価格などのフィールドを追加する必要があります。Book クラスは [DynamoDBTable] 属性で指定されており、この属性によって、Book が書き込まれるタイプのデータベーステーブルオブジェクトが定義されます。Book クラスの各インスタンスのキーは、[DynamoDBHashKey] 属性を使用して特定されます。プロパティは、[DynamoDBProperty] 属性を使用して識別されているため、プロパティが書き込まれるデータベーステーブルの列を指定します。データモデルを定義して、Book オブジェクトを作成、取得、更新、および削除するメソッドを書き込むことができます。

ボットを作成する

```
private void PerformCreateOperation()
{
    Book myBook = new Book
    {
        Id = bookID,
        Title = "object persistence-AWS SDK for.NET SDK-Book 1001",
        ISBN = "111-1111111001",
        BookAuthors = new List<string> { "Author 1", "Author 2" },
```

```
    };

    // Save the book.
    Context.SaveAsync(myBook,(result)=>{
        if(result.Exception == null)
            resultText.text += @"book saved";
    });
}
```

本を取得する

```
private void RetrieveBook()
{
    this.displayMessage += "\n*** Load book**\n";
    Context.LoadAsync<Book>(bookID,
                            (AmazonDynamoResult<Book> result) =>
    {
        if (result.Exception != null)
        {

            this.displayMessage += ("LoadAsync error" +result.Exception.Message);
            Debug.LogException(result.Exception);
            return;
        }
        _retrievedBook = result.Response;
        this.displayMessage += ("Retrieved Book: " +
                                "\nId=" + _retrievedBook.Id +
                                "\nTitle=" + _retrievedBook.Title +
                                "\nISBN=" + _retrievedBook.ISBN);
        string authors = "";
        foreach(string author in _retrievedBook.BookAuthors)
            authors += author + ",";
        this.displayMessage += "\nBookAuthor= "+ authors;
        this.displayMessage += ("\nDimensions= "+ _retrievedBook.Dimensions.Length + " X "
 +
                                _retrievedBook.Dimensions.Height + " X " +
                                _retrievedBook.Dimensions.Thickness);

    }, null);
}
```

本を更新する

```
private void PerformUpdateOperation()
{
    // Retrieve the book.
    Book bookRetrieved = null;
    Context.LoadAsync<Book>(bookID,(result)=>
    {
        if(result.Exception == null )
        {
            bookRetrieved = result.Result as Book;
            // Update few properties.
            bookRetrieved.ISBN = "222-2222221001";
            // Replace existing authors list with this
            bookRetrieved.BookAuthors = new List<string> { "Author 1", "Author x" };
            Context.SaveAsync<Book>(bookRetrieved,(res)=>
            {
```

```
                if(res.Exception == null)
                    resultText.text += ("\nBook updated");
            });
        }
    });
}
```

本を削除する

```
private void PerformDeleteOperation()
{
    // Delete the book.
    Context.DeleteAsync<Book>(bookID,(res)=>
    {
        if(res.Exception == null)
        {
            Context.LoadAsync<Book>(bookID,(result)=>
            {
                 Book deletedBook = result.Result;
                 if(deletedBook==null)
                    resultText.text += ("\nBook is deleted");
            });
        }
    });
}
```

Amazon Simple Notification Service

Amazon Simple Notification Service (SNS) と Unity SDK を使用すると、モバイルプッシュ通知を受信できる iOS と Android アプリを構築できます。SNS の詳細については、「Amazon Simple Notification Service」を参照してください。

このトピックでは、Amazon SNS を通じてモバイルプッシュ通知を受け取るための、AWS SDK for Unity sample app の SNSExample.unity の設定について説明します。

SNSExample.unity サンプルを使用して、iOS と Android アプリの両方を作成できます。iOS と Android の設定手順が異なります。ターゲットとするプラットフォームの適切なセクションをお読みください。

Prerequisites

このソリューションの使用の前提条件として以下が必要です。

SNS のアクセス許可を設定

Cognito Identity Pool を作成すると、2 つの IAM ロールが生成されます。

- Cognito/_<Identity-Pool-Name>Auth_DefaultRole - 認証されたユーザーに対するデフォルトの IAM ロール
- Cognito/_<Identity-Pool-Name>Unauth_DefaultRole - 未認証ユーザーに対するデフォルトの IAM ロール

Amazon SNS サービスへのアクセス権限をこれらのロールに追加する必要があります。目的:

1. IAM コンソールを参照し、設定する IAM ロールを選択します。
2. [ポリシーのアタッチ] をクリック後、AmazonSNSFullAccess ポリシーを選択し、[ポリシーのアタッチ] をクリックします。

> Note
>
> 本番環境では、AmazonSNSFullAccess を使用することはお勧めできませんが、ここでは使用して迅速に稼働させます。IAM ロールのアクセス許可を指定する方法については、「IAM ロールのアクセス許可の概要」を参照してください。

iOS 前提条件

- Apple iOS 開発者プログラムのメンバーシップ
- 署名 ID を生成する
- プッシュ通知用に設定されたプロビジョニングプロファイルを作成する

プッシュ通知を受け取るには、物理デバイスでアプリを実行する必要があります。デバイスでアプリを実行するには、Apple iOS Developer Program メンバーシップのメンバーシップが必要です。メンバーシップを取得したら、Xcode を使用して署名 ID を生成できます。詳細については、Apple の App ディストリビューションクイックスタートドキュメントを参照してください。次に、プッシュ通知用に設定されたプ

ロビジョニングプロファイルの詳細が必要になります。Apple の Configuring Push Notifications ドキュメントを参照してください。

Android の前提条件

- Android SDK のインストール
- JDK のインストール
- android-support-v4.jar
- google-play-services.jar

Unity Sample App for iOS の設定

Unity エディタを開き、新しいプロジェクトを作成します。[Assets (アセット)]/[Import Package (パッケージのインポート)]/[Custom Package (カスタムパッケージ)] を選択し、aws-unity-sdk-sns-2.0.0.1.unity パッケージを選択して、AWS SDK for Unity パッケージをインポートします。[Importing Package (パッケージをインポート)] ダイアログで、すべての項目が選択されていることを確認し、[Import (インポート)] をクリックします。

Unity の設定

Unity プロジェクトを設定するには、次の手順を実行します。

1. [Project (プロジェクト)] ペインで、[Assets (アセット)]/[AWSSDK]/[examples (例)] に移動し、SNSExample シーンを開きます。
2. [Hierarchy (階層)] ペインで、SNSExample を選択します。
3. [Inspector (インスペクター)] ペインで、Cognito ID プールの ID を指定します。
4. [iOS Platform Application ARN (iOS プラットフォームアプリケーション ARN)] のテキストボックスがあることを確認します。この情報は後で生成します。
5. [Build Settings] ダイアログで [File/Build Settings] を選択し、[Scenes in Build] リストボックスの下にある [Add Current] ボタンをクリックして、現在のシーンをビルドに追加します。
6. [Platform (プラットフォーム)] で [iOS] を選択し、[Player Settings... (プレイヤー設定...)] ボタンをクリックします。Unity エディタの [Inspector (インスペクター)] ペインで、iPhone アイコンをクリックしたら、[Identification (識別)] セクションまでスクロールして、[Bundle Identifier (バンドル識別子)] を指定します。

iOS 設定

iOS 固有の設定を構成するようにサンプルを設定するには、次の手順を実行します。

1. ウェブブラウザで Apple 開発者メンバーセンターに移動し、[Certificates、Identifiers & Profiles (証明書、識別子、プロファイル)] をクリックします。
2. [iOS Apps] の [Identifiers] をクリック後、その ウェブページの右上隅の + ボタンをクリックして新しい iOS App ID を追加し、App ID の説明を入力します。
3. [Add ID Suffix (ID サフィックスを追加)] セクションまで下にスクロールし、[Explicit App ID (明示的な App ID)] を選択して、バンドル ID を入力します。
4. [App Services (アプリサービス)] セクションまで下にスクロールし、[Push Notifications (プッシュ通知)] を選択します。
5. [Continue (続行)] ボタンをクリックします。
6. [Submit (送信)] ボタンをクリックします。

7. [Done (完了)] ボタンをクリックします。
8. 作成した App ID を選択し、[Edit (編集)] ボタンをクリックします。
9. [Push Notifications (プッシュ通知)] セクションまで下にスクロールします。
10. [Development SSL Certificate] の [Create Certificate] ボタンを選択します。
11. 手順に従って、証明書署名リクエスト (CSR) を作成してアップロードし、Apple Notification Service (APNS) との通信に使用する SSL 証明書をダウンロードします。
12. [Certificates, Identifiers & Profiles] ウェブページに戻り、[Provisioning Profiles] の下の [All] をクリックします。
13. 右上隅の + ボタンをクリックし、新しいプロビジョニングプロファイルを追加します。
14. [iOS App Development (iOS App の開発)] を選択し、[Continue (続行)] ボタンをクリックします。
15. App ID を選択し、[Continue (続行)] ボタンをクリックします。
16. 開発者の証明書を選択して、[Continue (続行)] ボタンをクリックします。
17. デバイスを選択し、[Continue (続行)] ボタンをクリックします。
18. プロファイル名を入力し、[Generate (生成)] ボタンをクリックします。
19. プロビジョニングプロファイルをダウンロードしてダブルクリックし、インストールします。

新しいプロファイルを追加した後、Xcode のプロビジョニングプロファイルを更新する必要がある場合があります。Xcode で:

1. [Xcode]/[Preferences (設定)] メニュー項目を選択します。
2. [Accounts (アカウント)] タブを選択後、Apple ID を選択し、[View Details (詳細を表示)] をクリックします。
3. プロビジョニングプロファイルを更新し、新しいプロファイルが表示されていることを確認するには、ダイアログの左下にある更新ボタンをクリックします。

SNS 構成

1. KeyChain アクセスアプリを実行し、画面の左下にある [My Certificates (自分の証明書)] を選択します。生成した SSL 証明書を右クリックして APNS に接続し、[Export (エクスポート)] を選択すると、ファイル名とパスワードを指定して証明書を保護するように求められます。証明書は、P12 ファイルに保存されます。
2. ウェブブラウザで SNS コンソールに移動し、画面左側の [アプリケーション] をクリックします。
3. [プラットフォームアプリケーションの作成] をクリックして、新しい SNS プラットフォームアプリケーションを作成します。
4. アプリケーション名を入力します。
5. Apple Push Notification Service Sandbox (APNS_SANDBOX) を選択して、[プッシュ通知プラットフォーム] を選択します。
6. [ファイルの選択] をクリックし、SSL 証明書をエクスポートした際に作成した P12 ファイルを選択します。
7. SSL 証明書をエクスポートした際に指定したパスワードを入力し、[認証情報をファイルから読み込み] をクリックします。
8. [プラットフォームアプリケーションの作成] をクリックします。
9. 作成したプラットフォームアプリケーションを選択して、アプリケーション ARN をコピーします。
10. Unity Editor でプロジェクトに戻り、[Hierarchy] ペインの [Inspector] ペインで [SNSExample] を選択し、[iOS Platform Application ARN] というテキストボックスに Platform Application ARN を貼り付けます。
11. [File (ファイル)]/[Build Settings (ビルド設定)] を選択し、[Build (ビルド)] ボタンをクリックすると Xcode プロジェクトが作成されます。

Xcode の使用

1. Xcode プロジェクトを開き、プロジェクトナビゲータでプロジェクトを選択します。
2. バンドルの識別子が正しく設定されていることを確認します。
3. Apple Developer Account が [Team (チーム)] で指定されていることを確認します。これは、プロビジョニングプロファイルを有効にするために必要です。
4. プロジェクトをビルドし、デバイス上で実行します。
5. [Register for Notification (通知に登録)] をタップし、[OK] をタップして通知を許可すると、アプリはデバイストークンを表示します。

SNS コンソールで [アプリケーション] をクリックし、プラットフォームアプリケーションを選択してから、[プラットフォームエンドポイントの作成] をクリックして、アプリで表示されるデバイストークンを入力します。

この時点で、アプリ、APNS、NSN は完全に設定されています。プラットフォームアプリケーションを選択してエンドポイントを選択して、[エンドポイントへの発行] をクリックすると、デバイスにプッシュ通知を送信できます。

Unity Sample (iOS)

このサンプルでは、CognitoAWSCredentials インスタンスを作成して、アプリが AWS サービスを呼び出すことを許可する一時的に限定的な認証情報を生成します。また、AmazonSimpleNotificationServiceClient のインスタンスを作成して SNS と通信します。アプリは、[Register for Notification (通知に登録)] と [Unregister (登録解除)] の 2 つのボタンを表示します。

[Register for Notifications (通知に登録)] ボタンをタップすると、`RegisterDevice()` メソッドが呼び出されます。`RegisterDevice()` は `UnityEngine.iOS.NotificationServices.RegisterForNotifications` を呼び出し、これによって使用される通知タイプ (アラート、サウンド、バッジ) が指定されます。また、デバイストークンを取得するために APNS への非同期呼び出しを行います。コールバックが定義されていないため、`CheckForDeviceToken` が繰り返し呼び出され (最大 10 回)、デバイストークンをチェックします。

トークンが検索されると、`AmazonSimpleNotificationServiceClient.CreatePlatformEndpointAsync()` が呼び出されて、SNS プラットフォームアプリケーションのエンドポイントが作成されます。

サンプルはプッシュ通知を受信するように設定されました。SNS コンソールに移動して、ページの左側にある [アプリケーション] をクリックします。プラットフォームアプリケーションを選択した後エンドポイントを選択して、[エンドポイントへの発行] をクリックします。使用するエンドポイントを選択し、[エンドポイントへの発行] をクリックします。テキストボックスにテキストメッセージを入力し、[メッセージの発行] をクリックしてメッセージを発行します。

Unity Sample App for Android の設定

Unity エディタを開き、新しいプロジェクトを作成します。[Assets (アセット)]/[Import Package (パッケージのインポート)]/[Custom Package (カスタムパッケージ)] を選択し、aws-unity-sdk-sns-2.0.0.1.unity パッケージを選択して、AWS SDK for Unity パッケージをインポートします。[Importing Package (パッケージをインポート)] ダイアログで、すべての項目が選択されていることを確認し、[Import (インポート)] をクリックします。

Unity の設定

Unity プロジェクトを設定するには、次の手順を実行します。

1. [Project (プロジェクト)] ペインで、[Assets (アセット)]/[AWSSDK]/[examples (例)] に移動し、SNSExample シーンを開きます。
2. [Hierarchy (階層)] ペインで、SNSExample を選択します。
3. [Inspector (インスペクター)] ペインで、Cognito ID プールの ID を指定します。
4. [Android Platform Application ARN (Android プラットフォームアプリケーション ARN)] と [Google Console Project ID (Google コンソールプロジェクト ID)] のテキストボックスがあることを確認します。この情報は後で生成します。
5. [Build Settings] ダイアログで [File/Build Settings] を選択し、[Scenes in Build] リストボックスの下にある [Add Current] ボタンをクリックして、現在のシーンをビルドに追加します。
6. [Platform (プラットフォーム)] で [Android] を選択し、[Player Settings… (プレイヤー設定…)] ボタンをクリックします。Unity エディタの [Inspector (インスペクター)] ペインで、Android アイコンをクリックしたら、[Identification (識別)] セクションまでスクロールして、[Bundle Identifier (バンドル識別子)] を指定します。
7. android-support-v4.jar and google-play-services.jar を、[Project] ペインの [Assets/Plugins/Android] ディレクトリにコピーします。

android-support-v4.jar の場所の詳細については、Android Support Library Setup を参照してください。google-play-services.jar の検索方法の詳細については、Google APIs for Android Setup を参照してください。

Android の設定

まず、新しい Google API プロジェクトを追加します。

1. ウェブブラウザで、Google 開発者コンソールに移動し、[Create Project (プロジェクトの作成)] を選択します。
2. [New Project (新規プロジェクト)] ボックスにプロジェクト名を入力し、プロジェクト番号をメモしたら (後で使用します)、[Create (作成)] をクリックします。

次に、プロジェクトの Google クラウド メッセージング (GCM) サービスを有効にします。

1. Google 開発者コンソールでは、新しいプロジェクトがすでに選択されています。選択されていない場合は、ページの上部にあるドロップダウンメニューからプロジェクトを選択します。
2. ページ左側のサイドバーから [APIs & auth (API と Auth)] を選択します。
3. 検索ボックスに「Google Cloud Messaging for Android」と入力し、[Google Cloud Messaging for Android] リンクをクリックします。
4. [Enable API (API を有効化)] をクリックします。

最後に、API キーを取得します。

1. Google 開発者コンソールで、[APIs & auth (API と Auth)]>[Credentials (認証情報)] の順に選択します。
2. [Public API access (パブリック API アクセス)] で、[Create new key (新しいキーを作成)] をクリックします。
3. [Create a new key (新しいキーの作成)] ダイアログで、[Server key (サーバーキー)] をクリックします。
4. 表示されたダイアログボックスで [Create (作成)] を選択し、表示された API キーをコピーします。

API キーは、後で認証を実行するために使用します。

SNS 構成

1. ウェブブラウザで SNS コンソールに移動し、画面左側の [アプリケーション] をクリックします。

2. [プラットフォームアプリケーションの作成] をクリックして、新しい SNS プラットフォームアプリケーションを作成します。
3. アプリケーション名を入力します。
4. [Push notification platform] で [Google Cloud Messaging (GCM)] を選択します。
5. [API キー] と表示されているテキストボックスに API キーを貼り付けます。
6. [プラットフォームアプリケーションの作成] をクリックします。
7. 作成したプラットフォームアプリケーションを選択して、アプリケーション ARN をコピーします。
8. Unity エディタでプロジェクトに戻り、[Hierarchy] ペインの [Inspector] ペインで [SNSExample] を選択し、プラットフォームアプリケーション ARN を [Android Platform Application ARN] というテキストボックスに貼り付け、プロジェクト番号を [Google Console Project ID] というテキストボックスに入力します。
9. Android デバイスをコンピュータに接続後、[File (ファイル)]/[Build Settings (ビルド設定)] を選択して、[Build and Run (ビルドして実行)] をクリックします。

Unity Sample (Android)

このサンプルでは、CognitoAWSCredentials インスタンスを作成して、アプリが AWS サービスを呼び出すことを許可する一時的に限定的な認証情報を生成します。また、AmazonSimpleNotificationServiceClient のインスタンスを作成して SNS と通信します。

アプリは、[Register for Notification (通知に登録)] と [Unregister (登録解除)] の 2 つのボタンを表示します。[Register for Notifications (通知に登録)] ボタンをタップすると、`RegisterDevice()` メソッドが呼び出されます。`RegisterDevice()` は `GCM.Register` を呼び出し、これによって GCM でアプリが登録されます。GCM はサンプルコード内で定義されたクラスです。これは、GCM でアプリを登録するための非同期呼び出しを行います。

コールバックが呼び出される
と、`AmazonSimpleNotificationServiceClient.CreatePlatformEndpointAsync` が呼び出されて、SNS メッセージを受信するためのプラットフォームエンドポイントが作成されます。

サンプルはプッシュ通知を受信するように設定されました。SNS コンソールに移動して、ページの左側にある [アプリケーション] をクリックします。プラットフォームアプリケーションを選択した後エンドポイントを選択して、[エンドポイントへの発行] をクリックします。使用するエンドポイントを選択し、[エンドポイントへの発行] をクリックします。テキストボックスにテキストメッセージを入力し、[メッセージの発行] をクリックしてメッセージを発行します。

AWS Lambda

AWS Lambda は、リクエストまたはイベントに応答してコードを実行するコンピューティングサービスです。コンピューティングリソースを自動的に管理して、新規情報に迅速に対応するアプリケーションを容易に構築できるようにします。AWS Lambda 関数は、モバイル、IoT、ウェブアプリから直接呼び出して、同期的にレスポンスを返すことができるため、インフラストラクチャをプロビジョニングまたは管理することなく、モバイルアプリ向けにスケーラブルかつセキュアで可用性が高いバックエンドを容易に作成することができます。

AWS Lambda は、次のいずれかに応答して Lambda 関数を実行できます。

- 離散型アップデートの分割のようなイベント (Amazon S3 または CloudWatch アラートのオブジェクト作成イベントなど)、またはアップデートのストリーミング (ウェブサイトのクリックストリームや接続されたデバイスからの出力など)。
- カスタムアプリケーションからの JSON の入力または HTTPS コマンド。

AWS Lambda は必要に応じてコードを実行し、1 日あたり数個のリクエストから 1 秒あたり数千のリクエストまで自動的にスケーリングします。これらの機能により、Lambda を使って Amazon S3 や Amazon DynamoDB などの AWS サービス用のトリガーを簡単に構築し、Amazon Kinesis に保存されたストリーミングデータを処理し、AWS のスケール、パフォーマンス、およびセキュリティで運用される独自のバックエンドを作成できます。

AWS Lambda の仕組みについては、「」を参照してください。AWS Lambda: 仕組み。

Permissions

Lambda 関数に関するアクセス権限は 2 種類あります。

- 実行アクセス権限 — アカウントの他の AWS リソースにアクセスするために Lambda 関数が必要とするアクセス権限。これらのアクセス権限は、実行ロールと呼ばれる IAM ロールを作成して付与します。
- 呼び出しアクセス権限 — Lambda 関数と通信するためにイベントソースが必要とするアクセス権限。呼び出しモデル (プッシュまたはプルモデル) に応じて、実行ロールまたはリソースポリシー (Lambda 関数に関連付けられたアクセスポリシー) のいずれかを使用して、これらのアクセス権限を付与できます。

プロジェクトのセットアップ

AWS Lambda のアクセス許可を設定する

1. AWS IAM コンソールを開きます。
2. このカスタムポリシーをロールにアタッチします。これにより、アプリケーションで AWS Lambda を呼び出すことができます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": [
        "lambda:*"
      ],
      "Resource": "*"
    }
  ]
}
```

新しい実行ロールを作成する

このロールは、次のステップで作成する Lambda 関数に適用され、この関数を使用してアクセスできる AWS リソースを決定します。

1. AWS IAM コンソールを開きます。
2. [ロール] をクリックします。
3. [新しいロールの作成] をクリックします。
4. 画面上の指示に従って、Lambda 関数でアクセスする必要のあるサービスとそのポリシーを選択します。たとえば、Lambda 関数で S3 バケットを作成する場合は、S3 への書き込みアクセスがあるポリシーが必要になります。
5. [ロールの作成] をクリックします。

AWS Lambda での関数の作成

1. AWS Lambda コンソールを開きます。
2. [Create a Lambda function (Lambda 関数の作成)] をクリックします。
3. [スキップ] をクリックして設計図の作成をスキップします。
4. 次の画面で、独自の関数を設定します。関数名、説明を入力し、ランタイムを選択します。選択したランタイムに基づき、画面上の指示に従います。新しく作成したロールを関数に割り当て、実行権限を指定します。
5. 完了したら、[次へ] をクリックします。
6. [関数の作成] をクリックします。

Lambda クライアントを作成する

```
var credentials = new CognitoAWSCredentials(IDENTITY_POOL_ID, RegionEndpoint.USEast1);
var Client = new AmazonLambdaClient(credentials, RegionEndpoint.USEast1);
```

リクエストオブジェクトを作成する

呼び出しタイプおよび関数名を指定するリクエストオブジェクトを作成します。

```
var request = new InvokeRequest()
{
    FunctionName = "hello-world",
    Payload = "{\"key1\" : \"Hello World!\"}",
    InvocationType = InvocationType.RequestResponse
};
```

Lambda 関数を呼び出す

呼び出して、リクエストオブジェクトを渡すには、以下のように行います。

```
Client.InvokeAsync(request, (result) =>
{
    if (result.Exception == null)
    {
        Debug.Log(Encoding.ASCII.GetString(result.Response.Payload.ToArray()));
    }
    else
    {
        Debug.LogError(result.Exception);
    }
});
```

Troubleshooting

AWS SDK for Unity で使用する Unity.WWW クラスの制限により、AWS サービス呼び出し中に問題が発生しても、詳細なエラーメッセージは返りません。このトピックでは、このような問題をトラブルシューティングする方法について説明します。

IAM ロールに必要なアクセス許可が付与されていることを確認する

アプリから AWS サービスを呼び出す際は、Cognito ID プールの ID を使用します。プールの各 ID は、IAM (Identity and Access Management) ロールに関連付いています。ロールには、1 つ以上のポリシーが関連付けられており、そのロールに割り当てられたユーザーがアクセスできる AWS リソースを指定しています。デフォルトでは、2 種類のロール (未認証ユーザー用と認証済みユーザー用) が作成されています。既存のポリシーファイルを変更するか、新しいポリシーファイルをアプリで必要なアクセス許可に関連付ける必要があります。認証済みユーザーと未認証ユーザーがいずれもアプリケーションにアクセスできるようにするには、アプリで必要な AWS リソースにアクセスするためのアクセス許可を両方のロールに付与する必要があります。

以下のポリシーファイルは、S3 バケットへのアクセス権限を付与する方法を示しています。

```
{
  "Statement": [
    {
      "Action": [
        "s3:AbortMultipartUpload",
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::MYBUCKETNAME/*",
      "Principal": "*"
    }
  ]
}
```

以下のポリシーファイルは、DynamoDB データベースへのアクセス権限を付与する方法を示しています。

```
{
"Statement": [{
    "Effect": "Allow",
    "Action": [
        "dynamodb:DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:UpdateItem"
    ],
    "Resource": "arn:aws:dynamodb:us-west-2:123456789012:table/MyTable"
}]
}
```

ポリシーの指定に関する詳細については、「IAM ポリシー」を参照してください。

HTTP プロキシデバッガーを使用する

アプリで呼び出す AWS サービスに HTTP または HTTPS エンドポイントがある場合、HTTP/HTTPS プロキシデバッガーを使用してリクエストおよびレスポンスを表示し、動作の詳細を確認できます。利用できる HTTP プロキシデバッガーは多数あります。

- Charles - OSX 用のウェブデバッギングプロキシ
- Fiddler - Windows 用のウェブデバッギングプロキシ

> **Important**
>
> Cognito 認証情報プロバイダーには、Charles ウェブデバッギングプロキシが実行されていると正常に動作しないという既知の問題があります。

Charles と Fiddler ではいずれも、一部の設定で、SSL 暗号化トラフィックを表示できるようにする必要があります。これらのツールの詳細については、該当するドキュメントをお読み下さい。ウェブデバッギングプロキシを使用しており、暗号化トラフィックを表示するように設定できない場合は、aws_endpoints_json ファイル (AWSUnitySDK/AWSCore/Resources にあります) を開き、デバッグが必要な AWS サービスの HTTP タグを true に設定します。