Guide du développeur

AWS SDK mobile pour Unity

AWS SDK mobile pour Unity: Guide du développeur

Table of Contents

Le SDK AWS mobile pour Unity est désormais inclus dans le AWS SDK pour .NET. Ce guide fait référence à la version archivée du SDK mobile pour Unity. Pour de plus amples informations, consultez Qu'est-ce que le SDK AWS mobile pour Unity ?.

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.

Qu'est-ce que le SDK AWS mobile pour Unity ?

Le SDK AWS mobile pour Unity est désormais inclus dans le SDK pour .NET. Pour plus d'informations, consultez le Manuel du développeur AWS SDK pour .NET .

Ce guide n'est plus mis à jour ; il fait référence à la version archivée du SDK mobile pour Unity.

Guides et sujets connexes

- Pour le développement d'applications frontales et mobiles, nous vous recommandons d'utiliser AWS Amplify.
- Pour les considérations particulières relatives à l'utilisation de AWS SDK pour .NET pour vos applications Unity, consultez la section Considérations spéciales relatives au support Unity dans le guide du AWS SDK pour .NET développeur.
- À titre de référence, vous pouvez trouver la version archivée du SDK AWS mobile pour Unity GitHub sur.

Contenu de référence archivé

Le SDK mobile archivé pour Unity contient un ensemble de classes .NET qui permettent aux jeux écrits avec Unity d' AWS utiliser des services. Les applications écrites avec le SDK mobile pour Unity peuvent être exécutées sur des appareils iOS ou Android.

Les AWS services pris en charge incluent :

- Amazon Cognito
- Amazon DynamoDB
- Gestion des identités et des accès AWS (JE SUIS)
- 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)

Ces services vous permettent d'authentifier les utilisateurs, d'enregistrer les données relatives aux joueurs et aux jeux, de conserver des objets dans le cloud, de recevoir des notifications Send et de collecter et d'analyser les données d'utilisation.

Compatibilité

Le SDK mobile pour Unity v3 est compatible avec les versions Unity 4.6 et supérieures.

La dernière version du SDK mobile pour Unity a introduit des améliorations qui peuvent vous obliger à modifier votre code lorsque vous l'intégrez à votre projet. Pour plus d'informations sur ces modifications, consultez la section Améliorations apportées au SDK AWS mobile pour Unity sur le blog Front-End Web & Mobile.

Téléchargez le SDK mobile pour Unity

Vous pouvez également télécharger le SDK mobile pour Unity sous forme de fichier .zip ici.

Qu'est-ce qui est inclus dans le SDK mobile pour Unity ?

Pour obtenir la liste complète des NuGet packages, des exemples et des autres fichiers du SDK mobile pour Unity, AWS SDK pour .NETreportez-vous GitHub à la section suivante.

Configuration du kit SDK AWS Mobile pour Unity

Pour débuter avec le kit SDK AWS Mobile pour Unity, vous pouvez configurer le kit SDK et commencer à créer un nouveau projet, ou bien intégrer le kit SDK à un projet existant. Vous pouvez également cloner et exécuter les exemples pour comprendre comment fonctionne le kit de développement logiciel.

Prérequis

Avant de pouvoir utiliser le kit SDK AWS Mobile pour Unity, vous avez besoin des éléments suivants :

- Un compte AWS
- Unity version 4.x ou 5.x (si vous souhaitez écrire des applications qui s'exécutent sous iOS 64 bits, Unity 4.6.4p4 ou 5.0.1p3 est requis)

Une fois les conditions requises respectées, vous devez effectuer les opérations suivantes :

1. Télécharger le kit SDK AWS Mobile pour Unity.
2. Configurer le kit SDK AWS Mobile pour Unity.
3. Obtenir les informations d'identification AWS à l'aide d'Amazon Cognito.

Étape 1 : Télécharger le kit SDK AWS Mobile pour Unity

Pour commencer, téléchargez le kit SDK AWS Mobile pour Unity. Chaque package du kit SDK est requis pour pouvoir utiliser le service AWS correspondant (en fonction du nom du package). Par exemple, le package aws-unity-sdk-dynamodb -2.1.0.0.unitypackage est utilisé pour appeler le service AWS DynamoDB. Vous pouvez importer tous les packages ou seulement ceux dont vous avez besoin.

1. Ouvrez l'éditeur Unity et créez un projet vide en gardant les paramètres par défaut.
2. Sélectionnez Assets > Import Package > Custom Package.
3. Dans la boîte de dialogue Import, accédez aux fichiers .unitypackage à utiliser et sélectionnez-les.
4. Dans la boîte de dialogue Importing, assurez-vous que tous les éléments sont sélectionnés et cliquez sur Import.

Étape 2 : Configurer le kit SDK AWS Mobile pour Unity

Créer une scène

Lorsque vous utilisez le kit SDK AWS Mobile pour Unity, vous pouvez commencer en incluant la ligne de code suivante dans la méthode `Start` ou `Awake` de votre classe de comportement Mono :

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

Créez votre scène en cliquant sur New Scene dans le menu File.

Le kit SDK AWS pour Unity contient des classes client pour chaque service AWS pris en charge. Ces clients sont configurés à l'aide d'un fichier nommé awsconfig.xml. La section suivante décrit les paramètres plus couramment utilisés dans le fichier awsconfig.xml. Pour plus d'informations sur ces paramètres, consultez la section Référence d'API du kit SDK Unity.

Définir la région par défaut des services AWS

Pour configurer la région par défaut pour tous les clients du service :

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

Ce paramètre définit la région par défaut pour tous les clients de service figurant dans le kit SDK Unity. Ce paramètre peut être remplacé en spécifiant explicitement la région au moment de la création d'une instance du client de service, comme suit :

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

Définir les informations de journalisation

Les paramètres de journalisation sont spécifiés comme suit :

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

Ce paramètre est utilisé pour configurer la journalisation dans Unity. Quand vous consignez des données dans `UnityLogger`, le framework imprime les données de sortie en interne dans les

journaux de débogage. Si vous souhaitez enregistrer les réponses HTTP, définissez l'indicateur LogResponses. Les valeurs peuvent être Always, Never ou. OnError Vous pouvez également enregistrer les mesures de performance des requêtes HTTP à l'aide de la propriété LogMetrics, le format du journal peut être spécifié à l'aide de la LogMetricsFormat propriété, les valeurs valides sont JSON ou standard.

L'exemple suivant illustre les paramètres les plus couramment utilisés dans le fichier awsconfig.xml. Pour plus d'informations sur certains paramètres de service spécifiques, consultez la section ci-dessous :

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

Utilisation du fichier link.xml

Le kit SDK utilise la réflexion pour les composants spécifiques à chaque plate-forme. Si vous utilisez le backend de script IL2 CPP, `strip bytecode` il est toujours activé sur iOS. Vous devez donc avoir un `link.xml` fichier dans votre racine d'assemblage avec les entrées suivantes :

```
<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>
```

Étape 3 : Obtenir l'ID du groupe d'identités avec Amazon Cognito

Pour utiliser les services AWS dans votre application mobile, vous devez obtenir l'ID du groupe d'identités à l'aide d'Amazon Cognito Identity. L'utilisation d'Amazon Cognito pour connaître l'ID du groupe d'identités permet à votre application d'accéder aux services AWS sans avoir à y intégrer vos informations d'identification privées. Cette approche vous permet également de définir des autorisations pour contrôler les services AWS auxquels vos utilisateurs ont accès.

Pour commencer avec Amazon Cognito, vous devez créer un groupe d'identités. Un groupe d'identités stocke les données d'identité utilisateur spécifiques à votre compte. Chaque groupe d'identités a des rôles IAM configurables que vous pouvez utiliser pour spécifier les services AWS auxquels les utilisateurs de votre application ont accès. En général, les développeurs utilisent un groupe d'identités par application. Pour en savoir plus sur les groupes d'identités, consultez le manuel du développeur Amazon Cognito.

Pour créer un pool d'identités pour votre application :

1. Connectez-vous à la console Amazon Cognito et cliquez sur Créer un groupe d'identités.
2. Entrez un nom pour votre pool d'identités, puis cochez la case autorisant l'accès pour les identités non authentifiées. Cliquez sur Create Pool pour créer votre pool d'identités.
3. Cliquez sur Autoriser pour créer les deux rôles par défaut associés à votre groupe d'identités : un pour les utilisateurs non authentifiés et un pour les utilisateurs authentifiés. Ces rôles par défaut fournissent un accès à Cognito Sync et Mobile Analytics pour le pool d'identités.

La page suivante affiche un code qui crée un fournisseur d'informations d'identification, afin que vous puissiez facilement intégrer Cognito Identity à votre application Unity. Transmettez l'objet du fournisseur d'informations d'identification au constructeur du client AWS que vous utilisez. Ce code se présente sous la forme suivante :

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

Étapes suivantes

- Commencer : lisez l'article Prise en main du kit SDK AWS Mobile pour Unity pour connaître en détail les services inclus dans le kit SDK.
- Exécuter les démonstrations : référez-vous à nos exemples d'applications Unity, qui illustrent quelques cas d'utilisation courants. Pour exécuter les exemples d'applications, configurez le kit SDK pour Unity comme décrit ci-dessus, puis suivez les instructions contenues dans les fichiers README de chaque exemple.
- Lire la référence de l'API : consultez la référence de l'API correspondant au kit SDK AWS Mobile pour Unity.
- Questions : publiez vos questions sur les forums du kit SDK AWS Mobile ou ouvrez un problème sur Github.

Prise en main du kit SDK AWS Mobile pour Unity

Cette page fournit une vue d'ensemble de chaque service AWS du kit SDK AWS Mobile pour Unity, ainsi que des instructions de configuration relatives aux exemples Unity. Avant de commencer à utiliser les services ci-dessous, vous devez suivre toutes les instructions de la page Configurer le kit SDK AWS Mobile pour Unity.

Amazon Cognito Identity

Tous les appels renvoyant vers AWS nécessitent la saisie d'informations d'identification AWS. Au lieu de coder en dur vos informations d'identification dans vos applications, nous vous recommandons d'utiliser Amazon Cognito Identity pour fournir les informations d'identification AWS à votre application. Suivez les instructions indiquée à la page Configurer le kit SDK AWS Mobile pour Unity pour obtenir les informations d'identification AWS via Amazon Cognito.

Cognito vous permet également d'authentifier les utilisateurs à l'aide de fournisseurs de connexion publics comme Amazon, Facebook, Twitter et Google, ainsi que des fournisseurs prenant en charge OpenID Connect. Cognito fonctionne également avec des utilisateurs non authentifiés. Cognito fournit des informations d'identification temporaires avec des droits d'accès limités, que vous spécifiez avec un rôle IAM (Identity and Access Management). Vous configurez Cognito en créant un groupe d'identités qui est associé à un rôle IAM. Ce rôle spécifie les ressources/services auxquels votre application peut accéder.

Pour prendre en main Cognito Identity, consultez le manuel du développeur Amazon Cognito.

Amazon Cognito Sync

Cognito Sync facilite l'enregistrement des données utilisateur telles que les préférences utilisateur ou l'état du jeu sur le cloud AWS, afin qu'elles soient à la disposition des utilisateurs, quel que soit l'appareil qu'ils utilisent. Cognito peut également enregistrer ces données localement, ce qui permet à vos applications de fonctionner même en l'absence de connexion Internet. Lorsqu'une connexion Internet est disponible, vos applications peuvent synchroniser leurs données locales vers le cloud.

Pour prendre en main Cognito Sync, consultez le manuel du développeur Amazon Cognito.

Utilisation de l' CognitoSyncManageréchantillon

Dans le volet Projet, accédez à Assets AWSSDK//examples/CognitoSync, puis dans la partie droite du volet, sélectionnez la CognitoSyncscène pour ouvrir la scène.

Pour exécuter l'exemple, cliquez sur le bouton de lecture en haut de l'écran de l'éditeur. Lorsque l'application s'exécute, elle affiche plusieurs zones de texte et boutons qui vous permettent d'entrer des informations sur le lecteur. Ci-dessous, vous trouverez une série de boutons permettant d'enregistrer localement les informations sur le lecteur, de synchroniser ces informations locales avec le cloud Cognito, de les actualiser à partir du cloud Cognito et de les supprimer. Appuyez sur chaque bouton pour effectuer une opération. L'exemple affiche un commentaire en haut de l'écran de jeu.

Pour configurer l' CognitoSyncManager exemple, vous devez spécifier un ID de pool d'identités Cognito. Pour spécifier cette valeur, dans l'éditeur Unity, sélectionnez-la SyncManagerdans le volet Hiérarchie et saisissez-la dans la zone de texte IDENTITY_POOL_ID du volet Inspector.

> **Note**
>
> L' CognitoSyncManager exemple contient du code qui illustre comment utiliser le fournisseur d'identité Facebook, recherchez la macro « USE_FACEBOOK_LOGIN ». Pour cette opération, le kit SDK Facebook pour Unity est nécessaire. Pour plus d'informations, consultez la page SDK Facebook pour Unity.

Dynamo DB

Amazon DynamoDB est un service de base de données non relationnelle rapide, économique, très évolutif et hautement disponible. DynamoDB permet de s'affranchir des limites habituelles du dimensionnement de stockage de données, tout en conservant une faible latence et des performances prévisibles.

Le kit SDK AWS pour Unity fournit des bibliothèques de bas niveau et de haut niveau pour l'utilisation de DynamoDB. La bibliothèque de haut niveau inclut l'outil de mappage d'objet de DynamoDB, qui vous permet de mapper des classes côté client aux tables DynamoDB, d'effectuer diverses opérations de création, de lecture, de mise à jour et de suppression, et d'exécuter des requêtes. A l'aide de cet outil, vous pouvez écrire du code simple et accessible, qui stocke des objets dans le cloud.

Pour plus d'informations sur DynamoDB, consultez le manuel du développeur DynamoDB.

Pour plus d'informations sur l'utilisation de Dynamo DB à partir d'applications Unity, consultez la page Amazon DynamoDB.

Utilisation de l'exemple DynamoDB

Dans le volet Projet, accédez à Assets//examples AWSSDK/DynamoDB. Cet exemple regroupe les scènes suivantes :

- Dynamo DBExample : la scène initiale de l'application
- LowLevelDynamoDbExample - exemple d'utilisation de l'API DynamoDBD de bas niveau
- TableQueryAndScanExample - exemple montrant comment effectuer des requêtes
- HighLevelExample - exemple d'utilisation de l'API DynamoDB de haut niveau

Ajoutez ces scènes à la composition (dans l'ordre d'apparition ci-dessus) à l'aide de la boîte de dialogue Build Settings (Paramètres de composition) (ouverte en sélectionnant File.Build Settings (Paramètres File.Build)). Cet exemple crée quatre tables : Forum ProductCatalog, Fil, Réponse.

Pour exécuter l'exemple, cliquez sur le bouton de lecture en haut de l'écran de l'éditeur. Lorsque l'application s'exécute, elle affiche un certain nombre de boutons :

- Low Level Table Operations (Opérations de table de bas niveau) : indique comment créer, répertorier, mettre à jour, décrire et supprimer des tables.
- Mid Level Query & Scan Operations (Opérations de requête et d'analyse de niveau intermédiaire) : indique comment exécuter des requêtes.
- High Level Object Mapper (Mappeur d'objet de haut niveau) : indique comment créer, mettre à jour et supprimer des objets.

Mobile Analytics

Avec Amazon Mobile Analytics, vous pouvez suivre les comportements des clients, regrouper des mesures, générer des visualisations de données et identifier des tendances pertinentes. Le kit SDK AWS pour Unity assure l'intégration avec le service Amazon Mobile Analytics. Pour plus d'informations sur Mobile Analytics, consultez la page manuel de l'utilisateur Mobile Analytics. Pour plus d'informations sur l'utilisation de Mobile Analytics à partir d'applications Unity, consultez la page Amazon Mobile Analytics.

Configuration de Mobile Analytics

Mobile Analytics définit certains paramètres qui peuvent être configurés dans le fichier awsconfig.xml :

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

- sessionTimeout : laps de temps au terme duquel la session ouverte est arrêtée après qu'une application passe en arrière-plan.
- max DBSize - Il s'agit de la taille de la SQLIte base de données. Lorsque la base de données atteint sa taille maximale, tous les nouveaux événements sont ignorés.
- dbWarningThreshold - Il s'agit de la limite de taille de la base de données qui, une fois atteinte, générera des journaux d'avertissement.
- maxRequestSize - Il s'agit de la taille maximale de la demande en octets qui doit être transmise dans une requête HTTP au service d'analyse mobile.
- allowUseDataRéseau : booléen qui indique si les événements de session sont envoyés sur le réseau de données.

Utilisation de l'exemple Mobile Analytics

Dans le volet Projet, accédez à Assets/AWSSDK/examples/Mobile Analytics, puis dans la partie droite du volet, sélectionnez la scène Amazon Mobile Analytics Sample pour ouvrir la scène. Pour utiliser l'exemple, vous devez ajouter votre application à l'aide de la console Amazon Mobile Analytics. Pour plus d'informations sur l'utilisation de la console Mobile Analytics, consultez le manuel de l'utilisateur Amazon Mobile Analytics.

Veuillez procéder comme suit pour configurer l'exemple avant l'exécution :

1. Sélectionnez l'objet AmazonMobileAnalyticsSample du jeu.
2. Spécifiez votre ID d'application (créé dans la console Amazon Mobile Analytics) dans le champ « App Id ».
3. Spécifiez votre ID de groupe d'identités Cognito (créé à l'aide de la console Amazon Cognito) dans le champ « Cognito Identity Pool Id ».

4. Vérifiez que vos rôles authentifiés et non authentifiés disposent d'autorisations pour accéder au service Mobile Analytics. Pour plus d'informations sur l'application d'une stratégie aux rôles IAM, consultez la page Gestion des rôles.

Lorsque vous exécutez l'exemple d'application, il se peut que les événements ne soient pas transmis immédiatement au service backend. Un thread d'arrière-plan place les événements dans le tampon local et les envoie par lots au système backend Amazon Mobile Analytics à intervalles réguliers (la valeur par défaut est 60 secondes) afin de maintenir les performances du jeu. En raison de la complexité du traitement effectué par Amazon Mobile Analytics sur vos données, il se peut que les événements soumis et les rapports correspondants ne soient visibles dans la console AWS qu'après un délai pouvant atteindre 60 minutes après l'envoi initial.

Pour plus d'informations sur les rapports fournis par Amazon Mobile Analytics, consultez la page Rapports et mesures mobiles.

Amazon S3

Amazon Simple Storage Service (Amazon S3) offre aux développeurs et aux équipes informatiques un espace de stockage d'objets sécurisé, durable et hautement évolutif. Depuis Unity, vous pouvez utiliser S3 pour stocker, afficher et récupérer des images, des vidéos, des musiques et d'autres données utilisées par vos jeux.

Pour plus d'informations sur S3, consultez les pages Amazon S3 et Mise en route sur S3.

Pour plus d'informations sur l'utilisation de S3 à partir d'applications Unity, consultez la page Amazon Simple Storage Service (S3).

Configuration de la signature par défaut de S3

La signature de S3 par défaut est configurée comme suit :

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

Elle est utilisée pour spécifier si vous devez utiliser Signature Version 4 pour les requêtes S3.

Utilisation de l'exemple S3

Dans le volet Projet, accédez à Assets/AWSSDK/examples/S3, et dans le côté droit du volet, sélectionnez la scène S3Example pour ouvrir la scène. L'exemple illustre comment afficher les

compartiments ou les objets d'un compartiment, publier des objets dans un compartiment et télécharger des objets d'un compartiment. Veuillez procéder comme suit pour configurer l'exemple avant l'exécution :

1. Sélectionnez l'objet de jeu S3 dans le volet Hierarchy.
2. Dans le volet Inspector, entrez les valeurs pour S3 BucketName et SampleFileName. S3 BucketName est le nom du compartiment utilisé par l'échantillon et S3 SampleFileName est le nom du fichier que l'échantillon téléchargera dans le compartiment S3 spécifié.
3. Vérifiez que vos rôles authentifiés et non authentifiés disposent d'autorisations pour accéder aux compartiments S3 dans votre compte. Pour plus d'informations sur l'application d'une stratégie aux rôles IAM, consultez la page Gestion des rôles.

Pour exécuter l'exemple, cliquez sur le bouton de lecture en haut de l'écran de l'éditeur. Lorsque l'application s'exécute, elle affiche un certain nombre de boutons :

- Get Objects : permet d'obtenir la liste de tous les objets inclus dans tous les compartiments de votre compte AWS.
- Get Buckets : permet d'obtenir la liste de tous les compartiments de votre compte AWS.
- Post Object : charge un objet dans le compartiment S3 spécifié.
- Delete Object : supprime tous les objets du compartiment S3 spécifié.

L'exemple affiche un commentaire en haut de l'écran de jeu.

Amazon Simple Notification Service

Amazon Simple Notification Service est un service de notification Push rapide, flexible et entièrement géré, qui vous permet d'envoyer des messages individuels ou de diffuser des messages à un grand nombre de destinataires. Amazon Simple Notification Service permet d'envoyer des notifications Push de manière simple et économique à des utilisateurs d'appareils mobiles ou titulaires d'adresses e-mail, et même d'envoyer des messages à d'autres services distribués. Pour démarrer avec Amazon Simple Notification Service, consultez la page Amazon Simple Notification Service.

AWS Lambda

AWS Lambda est un service de calcul qui exécute votre code en réponse à des demandes ou à des événements, et gère automatiquement les ressources de calcul pour vous, facilitant ainsi

le développement d'applications capables de réagir rapidement aux nouvelles informations. Les fonctions AWS Lambda peuvent être appelées directement à partir d'applications mobiles, IoT et Web. Comme elles envoient une réponse de manière synchrone, il est facile de créer des systèmes backend évolutifs, sécurisés et hautement disponibles pour vos applications mobiles sans avoir besoin de mettre en service ou de gérer l'infrastructure. Pour plus d'informations, consultez AWS Lambda.

Amazon Cognito Identity

Qu'est-ce qu'Amazon Cognito Identity ?

A l'aide d'Amazon Cognito Identity, vous pouvez créer des identités uniques pour vos utilisateurs et les authentifier pour qu'ils bénéficient d'un accès sécurisé à vos ressources AWS, comme Amazon S3 ou Amazon DynamoDB. Amazon Cognito Identity prend en charge les fournisseurs d'identité publics (Amazon, Facebook, Twitter/Digits, Google ou tout autre fournisseur compatible avec OpenID Connect) ainsi que les identités non authentifiées. Cognito prend également en charge les identités authentifiées par les développeurs, qui vous permettent d'inscrire et d'authentifier les utilisateurs par l'intermédiaire de votre propre processus d'authentification backend, tout en utilisant Amazon Cognito Sync pour synchroniser les données utilisateur et l'accès aux ressources AWS.

Pour en savoir plus sur Cognito Identity, consultez le Manuel du développeur Amazon Cognito.

Pour plus d'informations sur la disponibilité des régions pour l'authentification Cognito, consultez la page Disponibilité des régions Amazon Cognito Identity.

Utilisation d'un fournisseur public pour authentifier les utilisateurs

Pour plus d'informations sur l'utilisation de fournisseurs d'identité publics comme Amazon, Facebook, Twitter/Digits ou Google pour l'authentification des utilisateurs, consultez la page Fournisseurs d'identité externes dans le manuel du développeur Amazon Cognito.

Utilisation d'identités authentifiées par le développeur

Pour plus d'informations sur les identités authentifiées par les développeurs, consultez Identités authentifiées par le développeur dans le manuel du développeur Amazon Cognito.

Amazon Cognito Sync

Cognito Sync est un service AWS et une bibliothèque client qui permet la synchronisation des données utilisateur liées à une application sur différents appareils. Vous pouvez utiliser l'API Cognito Sync pour synchroniser les données utilisateur sur tous les appareils. Pour utiliser Cognito Sync dans votre application, vous devez inclure le kit SDK AWS Mobile pour Unity dans votre projet.

Pour plus d'informations sur la façon d'intégrer Amazon Cognito Sync dans votre application, consultez la page Manuel du développeur Amazon Cognito Sync.

Amazon Mobile Analytics

Avec Amazon Mobile Analytics, vous pouvez suivre les comportements des clients, regrouper des mesures, générer des visualisations de données et identifier des tendances pertinentes. Pour plus d'informations sur Mobile Analytics, consultez la page AWS Mobile Analytics.

Intégration d'Amazon Mobile Analytics

Les sections ci-dessous expliquent comment intégrer Mobile Analytics à votre application.

Créer une application dans la console Mobile Analytics

Accédez à la console Amazon Mobile Analytics et créez une application. Notez la valeur `appId` car vous en aurez besoin par la suite.

> **Note**
>
> Pour en savoir plus sur le fonctionnement de la console, consultez le manuel de l'utilisateur Amazon Mobile Analytics.

Lorsque vous créez une application dans la console Mobile Analytics, vous devez spécifier l'ID de votre groupe d'identités Cognito. Pour créer un nouveau groupe d'identités Cognito et générer un ID, veuillez consulter le manuel du développeur Cognito Identity.

Intégrer Mobile Analytics dans votre application

Pour accéder à Mobile Analytics à partir de Unity, vous devrez utiliser les éléments suivants à l'aide d'instructions :

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

Il est recommandé d'utiliser Amazon Cognito pour fournir des informations d'identification AWS temporaires à votre application. Ces identifiants permettent à l'application d'accéder à vos ressources AWS. Pour créer un fournisseur d'informations d'identification, suivez les instructions fournies à la section Amazon Cognito Identity.

Instanciez une MobileAnalyticsManager instance avec les informations suivantes :

- cognitoIdentityPoolId : ID du pool d'identités Cognito pour votre application
- CognitoRegion - La région de votre pool d'identités Cognito, par exemple «. RegionEndpoint USEast1»
- région - La région du service Mobile Analytics, par exemple «RegionEndpoint. USEast1»
- appId : valeur générée par la console Mobile Analytics lorsque vous ajoutez une application

Utilisez le MobileAnalyticsClientContextConfig pour initialiser une **`MobileAnalyticsManager`**instance, comme indiqué dans l'extrait de code suivant :

```
// Initialize the MobileAnalyticsManager
void Start()
{
    // ...
    analyticsManager = MobileAnalyticsManager.GetOrCreateInstance(
        new CognitoAWSCredentials(<cognitoIdentityPoolId>, <cognitoRegion>),
        <region>,
        <appId>);
    // ...
}
```

> **Note**
>
> L'ID d'application est généré pour vous par l'Assistant de création d'applications. Ces deux valeurs doivent correspondre à celles de la console Mobile Analytics.

L'`appId` permet de grouper vos données dans la console Mobile Analytics. Après avoir créé une application dans la console Mobile Analytics, vous pouvez trouver son ID en accédant à la console Mobile Analytics, puis en cliquant sur l'icône en forme de roue dentée dans le coin supérieur droit de l'écran. Cela affichera la page de gestion des applications qui répertorie toutes les applications enregistrées et leur application IDs.

Enregistrer les événements de monétisation

Le kit SDK pour Unity fournit la classe `MonetizationEvent`, qui vous permet de générer des événements de monétisation pour suivre les achats effectués au sein d'applications mobiles. L'extrait de code suivant montre comment créer un événement de monétisation :

```
// 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);
```

Enregistrer les événements personnalisés

Mobile Analytics vous permet de définir des événements personnalisés. Vous définissez entièrement les événements personnalisés. Ils vous aident à suivre les actions utilisateur propres à votre application ou à votre jeu. Pour plus d'informations sur les événements personnalisés, consultez la page Evénements personnalisés. Pour cet exemple, supposons que votre application est un jeu, et que vous voulez enregistrer un événement lorsqu'un utilisateur termine un niveau. Créez un événement « LevelComplete » en créant une nouvelle `AmazonMobileAnalyticsEvent` instance :

```
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);
```

Sessions d'enregistrement

Lorsque l'application passe à l'arrière-plan, vous pouvez mettre la session en pause. Dans `OnApplicationFocus`, vérifiez si l'application est en pause. Si tel est le cas, appelez `PauseSession`, sinon appelez `ResumeSession` comme illustré dans l'extrait de code suivant :

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

Par défaut, si l'utilisateur sort de l'application pendant moins de 5 secondes, puis y revient, sa session reprend. Si l'utilisateur quitte l'application pendant 5 secondes ou plus, une nouvelle session est créée. Ce paramètre peut être configuré dans le fichier awsconfig.xml. Pour plus d'informations, reportez-vous à la section Configuration de Mobile Analytics de la page Prise en main du kit SDK AWS Mobile pour Unity.

Amazon Simple Storage Service (S3)

Amazon Simple Storage Service (Amazon S3) offre aux développeurs et aux équipes informatiques un espace de stockage d'objets sécurisé, durable et hautement évolutif. Les développeurs Unity peuvent tirer parti de S3 pour charger dynamiquement les ressources utilisées par leurs jeux. Cela peut permettre de télécharger plus rapidement les jeux depuis les magasins d'applications.

Pour plus d'informations sur S3, consultez Amazon S3.

Pour en savoir plus sur la disponibilité des régions pour AWS S3, consultez la page Disponibilité des régions pour les services AWS.

> **Note**
>
> Certains des exemples présentés dans ce document supposent l'utilisation d'une variable de zone de texte appelée ResultText pour afficher le résultat du suivi.

Créer et configurer un compartiment S3

Amazon S3 stocke les ressources dans des compartiments Amazon S3, qui sont des conteneurs de stockage cloud qui résident dans une région spécifique. Chaque compartiment Amazon S3 doit avoir un nom unique dans le monde entier. Vous pouvez utiliser la console Amazon S3 pour créer un compartiment.

Création d'un compartiment S3

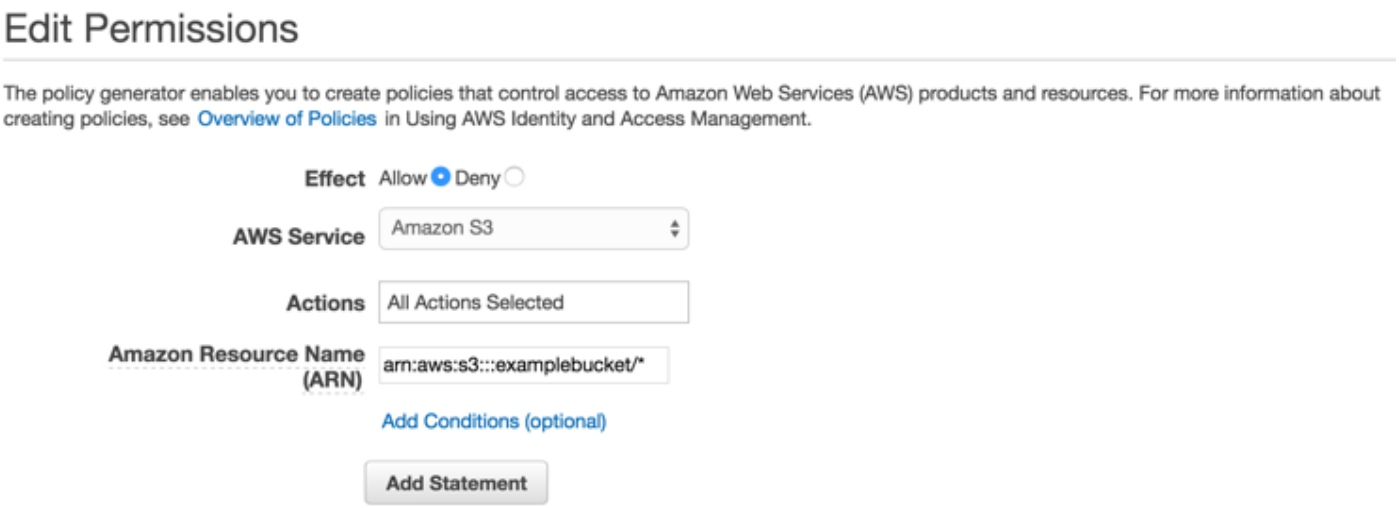
1. Connectez-vous à la console Amazon S3, puis cliquez sur Créer un compartiment.
2. Saisissez le nom du compartiment, sélectionnez une région, puis cliquez sur Créer.

Définir les autorisations pour S3

La stratégie de rôle IAM par défaut autorise votre application à accéder à Amazon Mobile Analytics et Amazon Cognito Sync. Pour que votre pool d'identités Cognito puisse accéder à Amazon S3, vous devez modifier les rôles de ce pool d'identités.

1. Accédez à la console Identity and Access Management et cliquez sur Rôles dans le volet de gauche.

2. Entrez le nom du pool d'identités dans la zone de recherche. Deux rôles seront répertoriés : un premier pour les utilisateurs authentifiés et un second pour les utilisateurs non authentifiés.
3. Cliquez sur le rôle pour les utilisateurs non authentifiés (unauth sera ajouté au nom du pool d'identités).
4. Cliquez sur Créer une stratégie de rôle, sélectionnez Générateur de stratégies, puis cliquez sur Sélectionner.
5. Sur la page Modifier les autorisations, entrez les paramètres affichés dans l'image suivante, en remplaçant le nom de ressource Amazon (ARN) par le vôtre. L'ARN d'un compartiment S3 ressemble à ceci : `arn:aws:s3:::examplebucket/*`. Il se compose de la région dans laquelle se trouve le compartiment et du nom du compartiment. Les paramètres ci-dessous accordent au pool d'identités complet l'accès à toutes les actions pour le compartiment spécifié.



1. Cliquez sur le bouton Ajouter une instruction, puis sur Étape suivante.
2. L'Assistant vous indiquera la configuration qui aura été générée. Cliquez sur Apply Policy.

Pour plus d'informations sur l'octroi d'accès à S3, consultez Octroi d'accès à un compartiment Amazon S3.

Charger les fichiers de la console

Pour charger un fichier test dans votre compartiment :

1. Dans la console S3, dans la vue du compartiment, cliquez sur Charger.

2. Cliquez sur Ajouter des fichiers et sélectionnez un fichier de test à charger. Pour ce didacticiel, nous allons supposer que vous chargez une image nommée `myImage.jpg`.
3. Une fois l'image de test sélectionnée, cliquez sur Start Upload.

(facultatif) Configurer la version de Signature pour les requêtes S3

Chaque interaction avec Amazon S3 est authentifiée ou anonyme. AWS utilise les algorithmes Signature Version 4 ou Signature Version 2 pour authentifier les appels au service.

Toutes les nouvelles régions AWS créées après janvier 2014 prennent en charge Signature Version 4 uniquement. Cependant, de nombreuses régions plus anciennes prennent encore en charge les demandes de Signature Version 4 et Signature Version 2.

Si votre compartiment se trouve dans l'une des régions qui ne prennent pas en charge les demandes Signature version 2 répertoriées sur cette page, vous devez configurer le AWSConfigs S3. UseSignatureVersion4 propriétés pour « vrai ».

Pour plus d'informations sur les versions d'AWS Signature, consultez la page Demandes d'authentification (AWS Signature Version 4).

Créer le client Amazon S3

Pour utiliser Amazon S3, nous devons d'abord créer une instance AmazonS3Client qui fait référence à l'instance Cognito AWSCredentials que vous avez créée précédemment :

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

La classe `AmazonS3Client` est le point d'entrée vers l'API S3 de haut niveau.

Etablir une liste des compartiments

Pour afficher les compartiments dans un compte AWS, appelez la méthode `AmazonS3Client.ListBucketsAsync`, comme illustré dans l'exemple de code suivant :

```
// 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";
    }
});
```

Affichage de la liste des objets

Pour afficher tous les objets d'un compartiment, appelez la méthode `AmazonS3Client.ListObjectsAsync`, comme illustré dans l'exemple de code suivant :

```
// 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";
    }
});
```

Téléchargement d'un objet

Pour télécharger un objet, créez-en un GetObjectRequest en spécifiant le nom et la clé du compartiment et transmettez l'objet à un appel au client. GetObjectAsync:

```
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 prend une instance de GetObjectRequest, un rappel et une AsyncOptions instance. Le rappel doit être de type :`AmazonServiceCallback<GetObjectRequest, GetObjectResponse>`. L' AsyncOptions instance est facultative. Si elle est spécifiée, elle détermine si le rappel est exécuté sur le thread principal.

Chargement d'un objet

Pour télécharger un objet, inscrivez-le dans un flux, créez-en un nouveau PostObjectRequest et spécifiez la clé, le nom du compartiment et les données du flux.

Le kit SDK AWS pour Unity utilise le client HTTP WWW qui ne prend pas en charge l'opération HTTP PUT. Afin de charger un objet dans votre compartiment S3, vous devez utiliser le Browser Post de S3, comme indiqué ci-dessous.

```
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 est un service de base de données non relationnelle rapide, économique, très évolutif et hautement disponible. DynamoDB permet de s'affranchir des limites habituelles du dimensionnement de stockage de données, tout en conservant une faible latence et des performances prévisibles. Pour plus d'informations sur DynamoDB, consultez la page Amazon DynamoDB.

Le kit SDK AWS Mobile pour Unity fournit une bibliothèque de haut niveau pour l'utilisation de DynamoDB. Vous pouvez également envoyer des requêtes directement à l'API DynamoDB de bas niveau, mais pour la plupart des cas d'utilisation, il est recommandée d'utiliser la bibliothèque de haut niveau. AmazonDynamoDBClient Il s'agit d'une partie particulièrement utile de la bibliothèque de haut niveau. Cette classe vous permet d'effectuer diverses opérations de création, de lecture, de mise à jour et de suppression, et d'exécuter des requêtes.

> **Note**
>
> Certains des exemples présentés dans ce document supposent l'utilisation d'une variable de zone de texte appelée ResultText pour afficher le résultat du suivi.

Intégration d'Amazon DynamoDB

Pour utiliser DynamoDB dans une application Unity, vous devez ajouter le kit SDK pour Unity dans votre projet. Si ce n'est pas encore fait, téléchargez le kit SDK pour Unity et suivez les instructions fournies à la section Configurer le kit SDK AWS Mobile pour Unity. Nous vous conseillons d'utiliser Amazon Cognito Identity pour fournir des informations d'identification AWS temporaires à vos applications. Ces informations permettent à votre application d'accéder aux ressources et services AWS.

Pour utiliser DynamoDB dans une application, vous devez définir les autorisations appropriées. La stratégie IAM suivante permet à l'utilisateur de supprimer, d'obtenir, de placer, d'analyser et de mettre à jour des éléments dans une table DynamoDB, qui est identifiée par son ARN :

```
{
"Statement": [{
    "Effect": "Allow",
```

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

Cette stratégie doit être appliquée aux rôles assignés au groupe d'identités Cognito, mais vous devez remplacer la valeur **`Resource`** par l'ARN approprié pour votre table DynamoDB. Cognito crée automatiquement un rôle pour votre nouveau groupe d'identités. Vous pouvez lui appliquer des stratégies via la console IAM.

Ajoutez ou supprimez ensuite des actions autorisées selon les besoins de votre application. Pour plus d'informations sur les stratégies IAM, consultez Utilisation d'IAM. Pour en savoir plus sur les stratégies DynamoDB, consultez la page Utilisation d'IAM pour contrôler l'accès aux ressources DynamoDB.

Créer une table DynamoDB

Maintenant que les autorisations et les informations d'identification sont définies, nous allons créer une table DynamoDB pour notre application. Pour créer une table, accédez à la console DynamoDB et procédez comme suit :

1. Cliquez sur Créer une table.
2. Entrez `Bookstore` comme nom de la table.
3. Sélectionnez Hachage comme type de clé primaire.
4. Sélectionnez Nombre et entrez `id` comme nom d'attribut de hachage. Cliquez sur Continuer.
5. Cliquez à nouveau sur Continuer pour ne pas ajouter d'index.
6. Définissez la capacité en lecture à `10` et la capacité en écriture à `5`. Cliquez sur Continuer.
7. Saisissez un e-mail de notification et cliquez sur Continuer pour créer des alarmes relatives au débit.
8. Cliquez sur Create. DynamoDB crée alors votre base de données.

Créer un client DynamoDB

Pour que notre application interagisse avec une table DynamoDB, nous avons besoin d'un client. Nous pouvons créer un client DynamodDB par défaut comme suit :

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

La AmazonDynamo DBClient classe est le point d'entrée de l'API DynamoDB. La classe fournit des méthodes d'instanciation pour créer, décrire, mettre à jour et supprimer des tables, entre autres opérations. Le contexte ajoute une nouvelle couche d'abstraction au client et vous permet d'utiliser des fonctionnalités supplémentaires, telles que le modèle de persistance des objets.

Décrire une table

Pour obtenir la description de notre table DynamoDB, nous pouvons utiliser le code suivant :

```
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);
```

```
    }
```

Dans cet exemple, nous créons un client et un DescribeTableRequest objet, attribuons le nom de notre table à la **`TableName`**propriété, puis transmettons l'objet de requête à la DescribeTableAsync méthode de l' AmazonDynamoDBClient objet. DescribeTableAsync prend également un délégué qui sera appelé lorsque l'opération asynchrone sera terminée.

> **Note**
>
> Toutes les méthodes asynchrones des délégués de AmazonDynamo DBClient prise qui sont appelées lorsque l'opération asynchrone est terminée.

Enregistrer un objet

Pour enregistrer un objet dans DynamoDB, utilisez SaveAsync <T>la méthode de l'objet, où T est AmazonDynamo DBClient le type d'objet que vous enregistrez.

Nous avons appelé notre base de données « Bookstore » (« Librairie »), et en accord avec ce thème, nous allons mettre en place un modèle de données qui enregistre les attributs liés aux livres. Voici les classes qui définissent notre modèle de données.

```
[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; }
    }
```

Bien entendu, pour une application de librairie réelle, nous aurions besoin de champs supplémentaires pour renseigner des données comme l'auteur et le prix. La classe Book est décorée avec l'attribut [DynamoDBTable], qui définit les objets de table de base de données du type Book dans lesquels seront écrits les objets. La clé de chaque instance de la classe Book est identifiée à l'aide de l'attribut [Dynamo DBHash Key]. Les propriétés sont identifiées par l'attribut

[DynamoDBProperty], qui indique la colonne de la table de base de données dans laquelle la propriété sera écrite. Une fois le modèle en place, nous pouvons enregistrer certaines méthodes pour créer, extraire, mettre à jour et supprimer des objets Livre.

Créer un livre

```
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";
    });
}
```

Récupérer un livre

```
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);
}
```

Mettre à jour un livre

```
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");
            });
        }
   });
}
```

Supprimer un livre

```
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) et le kit SDK Unity, vous permettent d'écrire des applications iOS et Android qui peuvent recevoir des notifications push mobile. Pour en savoir plus sur SNS, consultez la section Amazon Simple Notification Service.

Cette rubrique explique comment configurer le kit SDK AWS pour l'exemple d'application Unity SNSExample, .unity, afin de recevoir des notifications push mobiles via Amazon SNS.

Vous pouvez créer des applications iOS et Android à l'aide de l'exemple SNSExample .unity. Les étapes de configuration d'iOS et Android sont différentes, veuillez lire la section appropriée ci-dessous pour la plateforme que vous ciblez.

Prérequis

Les conditions préalables suivantes sont requises pour utiliser cette solution.

Définir les autorisations pour SNS

Lorsque vous créez un pool d'identités Cognito, deux rôles IAM sont générés :

- Cognito/_ <Identity-Pool-Name>Auth_ DefaultRole : rôle IAM par défaut pour les utilisateurs authentifiés
- Cognito/_ <Identity-Pool-Name>Unauth_ : rôle IAM par défaut pour les DefaultRole utilisateurs non authentifiés

Vous devez ajouter des autorisations d'accès au service Amazon SNS à ces rôles. Pour cela :

1. Accédez à la console IAM et sélectionnez le rôle IAM à configurer.
2. Cliquez sur Attacher la politique, sélectionnez la politique Amazon SNSFull Access et cliquez sur Attacher la politique.

> **Note**
>
> L'utilisation SNSFull d'Amazon Access n'est pas recommandée dans un environnement de production. Nous l'utilisons ici pour vous permettre d'être rapidement opérationnel. Pour

en savoir plus sur la définition d'autorisations pour un rôle IAM, consultez Présentation des autorisations des rôles IAM.

Prérequis iOS

- Adhésion au programme Apple iOS pour les développeurs
- Générer une identité de signature
- Créer un profil de mise en service configuré pour les notifications push

Pour recevoir des notifications Push, vous devez exécuter votre application sur un appareil physique. Pour exécuter votre application sur un appareil, vous devez adhérer au programme Apple iOS pour les développeurs. Une fois que vous disposez d'un abonnement, vous pouvez utiliser Xcode pour générer une identité de signature. Pour en savoir plus, consultez la documentation App Distribution Quick Start publiée par Apple. Ensuite, vous aurez besoin d'un profil de mise en service configuré pour les notifications push. Pour de plus amples informations, veuillez consulter la documentation Configuring Push Notifications publiée par Apple.

Prérequis Android

- Installer le kit SDK Android
- Installer le JDK
- android-support-v4.jar
- google-play-services.jar

Configuration de l'exemple d'application Unity pour iOS

Ouvrez l'éditeur Unity et créez un projet. Importez le package AWS SDK for Unity en sélectionnant Assets/Import Package/Custom Package et en sélectionnant aws-unity-sdk-sns -2.0.0.1.unitypackage. Assurez-vous que tous les éléments de la boîte de dialogue Importing Package sont sélectionnés et cliquez sur Import.

Configuration Unity

Effectuez les étapes suivantes pour configurer le projet Unity :

1. Dans le volet Projet, accédez à Assets/AWSSDK/examples et ouvrez la SNSExample scène.
2. Dans le volet Hiérarchie, sélectionnez SNSExample.
3. Dans le volet Inspector, spécifiez votre ID de pool d'identités Cognito.
4. Comme vous pouvez le remarquer, le volet contient une zone de texte iOS Platform Application ARN. Vous générerez ces informations plus tard.
5. Sélectionnez File/Build Settings, dans la boîte de dialogue Build Settings, cliquez sur le bouton Add Current sous la zone de liste Scenes in Build pour ajouter la scène actuelle à la version.
6. Sous Platform, sélectionnez iOS et cliquez sur le bouton Player Settings... dans le volet Inspector de l'éditeur Unity, cliquez sur l'icône iPhone et faites défiler vers le bas jusqu'à la section Identification et spécifiez un identifiant de groupe.

Configuration iOS

Effectuez les étapes suivantes pour configurer l'exemple afin de définir des paramètres spécifiques à iOS :

1. Dans un navigateur web, accédez au site Apple Developer Member Center, puis cliquez sur Certificates, Identifiers & Profiles.
2. Cliquez sur Identifiers sous iOS Apps, cliquez sur le bouton Plus situé dans l'angle supérieur droit de la page pour ajouter un nouvel identifiant iOS App ID, puis saisissez une description de l'ID d'application.
3. Faites défiler la page jusqu'à la section Add ID Suffix, puis choisissez Explicit App ID et saisissez votre identifiant de groupe.
4. Faites défiler jusqu'à la section App Services et sélectionnez Push Notifications.
5. Cliquez sur le bouton Continue.
6. Cliquez sur le bouton Submit.
7. Cliquez sur le bouton Done.
8. Sélectionnez l'ID d'application que vous venez de créer, puis cliquez sur le bouton Edit.
9. Faites défiler jusqu'à la section Push Notifications.
10. Cliquez sur le bouton Create Certificate sous Development SSL Certificate.
11. Suivez les instructions pour créer une demande de signature de certificat (CSR, Certificate Signing Request), chargez la demande et téléchargez un certificat SSL qui sera utilisé pour communiquer avec Apple Notification Service (APNS).

12. De retour dans la page web Certificates, Identifiers & Profiles, cliquez sur All sous Provisioning Profiles.
13. Cliquez sur le bouton Plus dans l'angle supérieur droit pour ajouter un nouveau profil de mise en service.
14. Sélectionnez iOS App Development et cliquez sur le bouton Continue.
15. Sélectionnez votre ID d'application et cliquez sur le bouton Continue.
16. Sélectionnez votre certificat de développeur et cliquez sur le bouton Continue.
17. Sélectionnez votre appareil et cliquez sur le bouton Continue.
18. Saisissez un nom de profil et cliquez sur le bouton Generate.
19. Téléchargez le fichier de mise en service et double-cliquez dessus pour installer le profil de mise en service.

Vous devrez peut-être actualiser les profils de mise en service dans Xcode après avoir ajouté un nouveau profil. Dans Xcode :

1. Sélectionnez l'élément de menu Xcode/Preferences.
2. Sélectionnez l'onglet Accounts, puis votre ID Apple et cliquez sur View Details.
3. Cliquez sur le bouton Refresh dans le coin inférieur gauche de la boîte de dialogue pour actualiser vos profils de mise en service et vous assurer que votre nouveau profil s'affiche.

Configuration SNS

1. Lancez l'application KeyChain d'accès, sélectionnez Mes certificats dans le coin inférieur gauche de l'écran, cliquez avec le bouton droit sur le certificat SSL que vous avez généré pour vous connecter à APNS et sélectionnez Exporter. Vous serez invité à spécifier un nom pour le fichier et un mot de passe pour protéger le certificat. Le certificat sera enregistré dans un fichier P12.
2. Dans un navigateur web, accédez à la console SNS et cliquez sur Applications sur le côté gauche de l'écran.
3. Cliquez sur Créer une application de plate-forme pour créer une nouvelle application de plateforme SNS.
4. Saisissez le nom de l'application.
5. Sélectionnez Apple Push Notification Service Sandbox (APNS_SANDBOX) pour Plate-forme de notifications push.

6. Cliquez sur Choisir un fichier et sélectionnez le fichier P12 que vous avez créé lorsque vous avez exporté votre certificat SSL.
7. Entrez le mot de passe vous avez spécifié lorsque vous avez exporté le certificat SSL et cliquez sur Charger à partir du fichier.
8. Cliquez sur Créer une application de plate-forme.
9. Sélectionnez l'application de plateforme que vous venez de créer et copiez l'ARN de l'application.
10. Revenez à votre projet dans l'éditeur Unity, sélectionnez-le SNSExampledans le volet Hiérarchie, dans le volet Inspector et collez l'ARN de l'application de plate-forme dans la zone de texte intitulée ARN de l'application de plate-forme iOS.
11. Sélectionnez File/Build Settings, puis cliquez sur le bouton Build pour créer un projet Xcode.

Utilisation de Xcode

1. Ouvrez le projet Xcode et sélectionnez le projet dans le navigateur de projet.
2. Vérifiez que l'identifiant de groupe est défini correctement
3. Vérifiez que votre compte de développeur Apple est spécifié dans l'équipe - cette action est obligatoire pour que votre profil de mise en service prenne effet.
4. Générez le projet et exécutez-le sur votre appareil.
5. Appuyez sur Register for Notification, puis sur OK pour autoriser les notifications, l'application affichera votre jeton d'appareil

Dans la console SNS, cliquez sur Applications, sélectionnez votre application de plateforme, cliquez sur Créer un point de terminaison de plate-forme, et saisissez le jeton d'appareil affiché sur l'application.

A ce stade, votre application, APNS et NSN sont entièrement configurés. Vous pouvez sélectionner votre application de plateforme, choisir votre point de terminaison, puis cliquer sur Publier sur le point de terminaison pour envoyer une notification push à votre appareil.

Exemple Unity (iOS)

L'exemple crée une AWSCredentials instance Cognito pour générer des informations d'identification temporaires à portée limitée qui permettent à l'application d'appeler les services AWS. Il crée également une instance de AmazonSimpleNotificationServiceClient pour communiquer avec SNS. L'application affiche deux boutons intitulés Register for Notification et Unregister.

Lorsque vous appuyez sur le bouton Register for Notifications, la méthode `RegisterDevice()` est appelée. `RegisterDevice()` appelle`UnityEngine.iOS.NotificationServices.RegisterForNotifications`, qui spécifie les types de notification (alerte, son ou badge) qui seront utilisés. Elle effectue aussi un appel asynchrone à APNS pour obtenir un jeton d'appareil. Parce qu'aucun rappel n'est défini, `CheckForDeviceToken` est appelée plusieurs fois (jusqu'à 10 fois) pour vérifier le jeton de l'appareil.

Lorsqu'un jeton est récupéré `AmazonSimpleNotificationServiceClient.CreatePlatformEndpointAsync()` est appelée pour créer un point de terminaison pour l'application de plateforme SNS.

L'exemple est maintenant configuré pour recevoir des notifications push. Vous pouvez accéder à la console SNS, cliquer sur Applications sur le côté gauche de la page, sélectionner votre application de plateforme, choisir un point de terminaison et cliquer sur Publier sur le point de terminaison. Sélectionnez le point de terminaison à utiliser et cliquez sur Publier sur le point de terminaison. Pour publier un message, saisissez un message texte dans la zone de texte et cliquez sur Publier le message.

Configuration de l'exemple d'application Unity pour Android

Ouvrez l'éditeur Unity et créez un projet. Importez le package AWS SDK for Unity en sélectionnant Assets/Import Package/Custom Package et en sélectionnant aws-unity-sdk-sns -2.0.0.1.unitypackage. Assurez-vous que tous les éléments de la boîte de dialogue Importing Package sont sélectionnés et cliquez sur Import.

Configuration Unity

Effectuez les étapes suivantes pour configurer le projet Unity :

1. Dans le volet Projet, accédez à Assets/AWSSDK/examples et ouvrez la SNSExample scène.
2. Dans le volet Hiérarchie, sélectionnez SNSExample.
3. Dans le volet Inspector, spécifiez votre ID de pool d'identités Cognito.
4. Comme vous pouvez le remarquer, le volet contient des zones de texte Android Platform Application ARN et Google Console Project ID. Vous générerez ces informations plus tard.
5. Sélectionnez File/Build Settings, dans la boîte de dialogue Build Settings, cliquez sur le bouton Add Current sous la zone de liste Scenes in Build pour ajouter la scène actuelle à la version.

6. Sous Platform sélectionnez Android et cliquez sur le bouton Player Settings... dans le volet Inspector de l'éditeur Unity, cliquez sur l'icône Android et faites défiler vers le bas jusqu'à la section Identification et spécifiez un identifiant de groupe.
7. Copiez les fichiers android-support-v 4.jar et google-play-services .jar dans le répertoire Assets/Plugins/Android du volet Projet.

Pour plus d'informations sur l'emplacement du android-support-v fichier 4.jar, consultez la section Configuration de la bibliothèque de support Android. Pour plus d'informations sur la recherche du google-play-services fichier .jar, consultez la section Configuration de Google APIs pour Android.

Configuration Android

Tout d'abord, ajoutez un nouveau projet d'API Google :

1. Dans un navigateur web, accédez à la console Google Developers, cliquez sur Google APIs for Android Setup.
2. Dans la zone New Project, entrez un nom de projet, notez son numéro (vous en aurez besoin plus tard) et cliquez sur Create.

Ensuite, activez le service de messagerie de Google Cloud (GCM) pour le projet :

1. Dans la console Google Developers, votre nouveau projet doit être déjà sélectionné, sinon, sélectionnez-le dans le menu déroulant en haut de la page.
2. Sélectionnez APIs & auth dans la barre latérale sur le côté gauche de la page.
3. Dans la zone de recherche, tapez « Google Cloud Messaging pour Android » et cliquez sur le lien Google Cloud Messaging pour Android ci-dessous.
4. Cliquez sur Enable API.

Enfin, obtention d'une clé API :

1. Dans la console Google Developers, sélectionnez APIs & auth > Identifiants.
2. Sous Public API access, cliquez sur Create new key.
3. Dans la boîte de dialogue Create new key, cliquez sur Server Key.
4. Dans la boîte de dialogue qui s'affiche, cliquez sur Create, puis copiez la clé d'API affichée.

Vous utiliserez la clé API pour effectuer l'authentification par la suite.

Configuration SNS

1. Dans un navigateur web, accédez à la console SNS et cliquez sur Applications sur le côté gauche de l'écran.
2. Cliquez sur Créer une application de plate-forme pour créer une nouvelle application de plateforme SNS.
3. Saisissez le nom de l'application
4. Sélectionnez Google Cloud Messaging (GCM) comme plateforme de notification push
5. Collez la clé de l'API dans la zone de texte API key.
6. Cliquez sur Create platform application
7. Sélectionnez l'application de plateforme que vous venez de créer et copiez l'ARN de l'application.
8. Revenez à votre projet dans l'éditeur Unity, sélectionnez-le SNSExampledans le volet Hierarchy, dans le volet Inspector et collez l'ARN de l'application de plateforme dans la zone de texte intitulée Android Platform Application ARN et votre numéro de projet dans la zone de texte intitulée ID de projet de la console Google.
9. Connectez votre appareil Android à votre ordinateur, sélectionnez File/Build Settings et cliquez sur Build and Run.

Exemple Unity (Android)

L'exemple crée une AWSCredentials instance Cognito pour générer des informations d'identification temporaires à portée limitée qui permettent à l'application d'appeler les services AWS. Il crée également une instance de AmazonSimpleNotificationServiceClient pour communiquer avec SNS.

L'application affiche deux boutons intitulés Register for Notification et Unregister. Lorsque vous appuyez sur le bouton Register for Notifications, la méthode `RegisterDevice()` est appelée. `RegisterDevice()` appelle`GCM.Register`, qui inscrit l'application auprès de GCM. GCM est une classe définie dans l'exemple de code. Elle effectue un appel asynchrone pour inscrire l'application auprès de GCM.

Lorsque le rappel est appelé, `AmazonSimpleNotificationServiceClient.CreatePlatformEndpointAsync` est appelée pour créer un point de terminaison de plateforme afin de recevoir des messages SNS.

L'exemple est maintenant configuré pour recevoir des notifications push. Vous pouvez accéder à la console SNS, cliquer sur Applications sur le côté gauche de la page, sélectionner votre application de plateforme, choisir un point de terminaison et cliquer sur Publier sur le point de terminaison. Sélectionnez le point de terminaison à utiliser et cliquez sur Publier sur le point de terminaison. Pour publier un message, saisissez un message texte dans la zone de texte et cliquez sur Publier le message.

AWS Lambda

AWS Lambda est un service de calcul qui exécute votre code en réponse à des demandes ou à des événements, et gère automatiquement les ressources de calcul pour vous, facilitant ainsi le développement d'applications capables de réagir rapidement aux nouvelles informations. Les fonctions AWS Lambda peuvent être appelées directement à partir d'applications mobiles, IoT et Web. Comme elles envoient une réponse de manière synchrone, il est facile de créer des systèmes backend évolutifs, sécurisés et hautement disponibles pour vos applications mobiles sans avoir besoin de mettre en service ou de gérer l'infrastructure.

AWS Lambda peut exécuter vos fonctions Lambda en réponse à l'un des éléments suivants :

- Des événements, tels que des mises à jour discrètes (par exemple, des événements créés par des objets dans Amazon S3 ou des CloudWatch alertes) ou des mises à jour en continu (par exemple, des flux de clics sur des sites Web ou des résultats provenant d'appareils connectés).
- Entrées JSON ou commandes HTTPS à partir de vos applications personnalisées.

AWS Lambda exécute le code uniquement lorsque cela est nécessaire et s'adapte automatiquement, qu'il s'agisse de quelques requêtes par jour ou de milliers de requêtes par seconde. Avec ces fonctionnalités, vous pouvez utiliser Lambda pour créer facilement des déclencheurs pour des services AWS tels qu'Amazon S3 et Amazon DynamoDB, pour traiter les données de diffusion stockées dans Amazon Kinesis ou pour créer vos propres services backend, tout en bénéficiant du dimensionnement, des performances et de la sécurité d'AWS.

Pour en savoir plus sur le fonctionnement d'AWS Lambda, consultez la page Fonctionnement d'AWS Lambda.

Permissions

Il existe deux types d'autorisations associées à des fonctions Lambda :

- Autorisations d'exécution : autorisations dont votre fonction Lambda a besoin pour accéder aux autres ressources AWS de votre compte. Pour accorder ces autorisations, créez un rôle IAM, dit rôle d'exécution.
- Autorisations d'appel : autorisations dont la source de l'événement a besoin pour communiquer avec votre fonction Lambda. En fonction du modèle d'appel (modèle « push » ou « pull »), vous

pouvez accorder ces autorisations à l'aide de stratégies de rôle d'exécution ou de ressource (stratégie d'accès associée à votre fonction Lambda).

Configuration du projet

Définir des autorisations pour AWS Lambda

1. Ouvrez la console AWS IAM.
2. Attachez cette stratégie personnalisée à vos rôles, ce qui permet à votre application d'effectuer des appels vers AWS Lambda.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:*"
      ],
      "Resource": "*"
    }
  ]
}
```

Créer un nouveau rôle d'exécution

Ce rôle s'applique à la fonction Lambda que vous allez créer à l'étape suivante et détermine les ressources AWS auxquelles cette fonction peut accéder.

1. Ouvrez la console AWS IAM.
2. Cliquez sur Rôles.
3. Cliquez sur Créer un rôle.

4. Suivez les instructions à l'écran pour sélectionner les services et les stratégies correspondantes auxquelles votre fonction Lambda doit accéder. Par exemple, si vous voulez que votre fonction Lambda crée un compartiment S3, votre stratégie doit disposer d'un accès en écriture à S3.
5. Cliquez sur Créer un rôle.

Création d'une fonction dans AWS Lambda

1. Ouvrez la console AWS Lambda.
2. Cliquez sur Créer une fonction Lambda.
3. Cliquez sur Ignorer pour éviter la création d'un plan.
4. Configurez votre propre fonction sur l'écran suivant. Entrez le nom de la fonction, une description, puis choisissez l'environnement d'exécution. Suivez les instructions à l'écran en fonction de votre environnement d'exécution. Spécifiez les autorisations d'exécution en affectant le rôle d'exécution nouvellement créé à votre fonction.
5. Cliquez ensuite sur Suivant.
6. Cliquez sur Créer une fonction.

Créer un client Lambda

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

Créer un objet de requête

Créez un objet de requête pour spécifier le type d'appel et le nom de la fonction :

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

Appeler votre fonction Lambda

Effectuez l'appel, en indiquant l'objet de la requête :

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

Dépannage

En raison des limites de la classe Unity.WWW utilisées par le kit AWS SDK pour Unity, les messages d'erreur détaillés ne sont pas retournés lorsqu'un problème se produit lors de l'appel d'un service AWS. Cette rubrique décrit quelques idées pour le dépannage de ce type de problème.

Vérifier que le rôle IAM dispose des autorisations requises

Lorsque vous appelez les services AWS, votre application utilise l'identité d'un pool d'identités Cognito. Chaque identité du groupe est associée à un rôle IAM (Identity and Access Management). Le rôle est associé à un ou plusieurs fichiers de stratégie qui désignent les ressources AWS auxquelles ont accès les utilisateurs assignés à ce rôle. Par défaut, deux rôles sont créés : un premier pour les utilisateurs authentifiés et un second pour les utilisateurs non authentifiés. Vous aurez besoin de modifier le fichier de stratégie existant ou d'associer un nouveau fichier de stratégie aux autorisations requises par votre application. Si votre application autorise les utilisateurs authentifiés et non authentifiés, les deux rôles doivent disposer des autorisations pour l'accès aux ressources AWS dont votre application a besoin.

Le fichier de stratégie suivant montre comment accorder l'accès à un compartiment S3 :

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

Le fichier de stratégie suivant montre comment accorder l'accès à une base de données DynamoDB :

```
{
```

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

Pour plus d'informations sur la spécification de stratégies, consultez Stratégies IAM.

Utilisation d'un débogueur proxy HTTP

Si le service AWS que votre application appelle dispose d'un point de terminaison HTTP ou HTTPS, vous pouvez utiliser un débogueur de proxy HTTP/HTTPS pour afficher les requêtes et les réponses afin de mieux comprendre ce qui se passe. Il existe un certain nombre de débogueurs proxy HTTP tels que :

- Charles : proxy de débogage web pour OSX
- Fiddler : proxy de débogage web pour Windows

> **Important**
>
> Un problème a déjà été détecté avec le fournisseur d'informations d'identification Cognito lorsque vous exécutez le proxy de débogage web Charles qui empêche le fournisseur d'informations d'identification de fonctionner correctement.

Charles et Fiddler nécessitent tous les deux un peu de travail de configuration pour pouvoir afficher le trafic SSL chiffré. C'est pourquoi il est préférable de lire la documentation sur ces outils pour plus d'informations. Si vous utilisez un proxy de débogage Web qui ne peut pas être configuré pour afficher le trafic chiffré, ouvrez le fichier aws_endpoints_json (situé dans AWSUnitySDK/AWSCore/Resources) et définissez la balise HTTP du service AWS que vous devez déboguer sur true