Entwicklerhandbuch von Xamarin

AWS SDK für Mobilgeräte

AWS SDK für Mobilgeräte: Entwicklerhandbuch von Xamarin

Table of Contents

Was ist das AWS Mobile SDK for .NET and Xamarin?

AWS Mobile SDK for .NET and Xamarin stellt eine Reihe von .NET-Bibliotheken sowie Code-Beispiele und Dokumentation bereit, um Entwickler beim Erstellen vernetzter Mobilgeräteanwendungen zu unterstützen:

- Xamarin iOS
- Xamarin Android
- Windows Phone Silverlight
- Windows RT 8.1
- Windows Phone 8.1

Mobile Apps, die mit der geschrieben wurden, AWS Mobile SDK for .NET and Xamarin rufen die native Plattform APIs auf, damit sie das Look&Feel nativer Anwendungen annehmen. Die .NET-Bibliotheken im SDK stellen C#-Wrapper für den AWS REST APIsbereit.

Was ist in enthalten?AWS Mobile SDK for .NET and Xamarin?

Zu den unterstützten AWS-Services zählen derzeit (nicht exklusiv):

- Amazon Cognito
- Amazon S3
- Amazon DynamoDB
- Amazon Mobile Analytics
- Amazon Simple Notification Service

Mit diesen Services können Sie Benutzer authentifizieren, Spieler- und Spieldaten speichern, Objekte in der Cloud speichern, Push-Benachrichtigungen empfangen und Nutzungsdaten erfassen und analysieren.

Mit AWS Mobile SDK for .NET and Xamarin können Sie außerdem die meisten AWS-Services nutzen, die von AWS SDK for. NET unterstützt werden. Die AWS-Services für die Entwicklung für Mobilgeräte werden in diesem Entwicklerhandbuch erläutert. Weitere Informationen zum AWS SDK for .NET finden Sie unter:

- AWS SDK for .NET – Handbuch Erste Schritte
- AWS SDK for .NET Entwicklerhandbuch
- AWS SDK for .NET – API-Referenz

Compatability

AWS Mobile SDK for .NET and Xamarin wird als PCL (Portable Class Library) ausgeliefert. Xamarin.Android 4.10.1, Xamarin.iOS 7.0.4 und Xamarin Studio 4.2 wurden um die Unterstützung von PCL ergänzt. Portable Library-Projekte werden in Xamarin Studio für OS X automatisch aktiviert und in Visual Studio 2013 integriert.

IDEs

- Windows: Sie können Visual Studio oder Xamarin Studio zum Entwickeln der Anwendung verwenden.
- Mac: Sie müssen die Xamarin Studio-IDE (Integrated Development Environment, integrierte Entwicklungsumgebung) zur Entwicklung der Anwendungen verwenden. Die iOS-Entwicklung mit Xamarin setzt Zugriff auf einen Mac voraus, auf dem die App ausgeführt werden kann. Weitere Informationen erhalten Sie unter Installing Xamarin.iOS on Windows.

Wie erhalte ich AWS Mobile SDK for .NET and Xamarin?

Informationen zum Erhalt von AWS Mobile SDK for .NET and Xamarin finden Sie unter Einrichten von AWS Mobile SDK for .NET and Xamarin. Der AWS Mobile SDK for .NET and Xamarin wird als NuGet Pakete verteilt. Eine vollständige Liste der AWS-Servicepakete finden Sie unter AWS SDK packages on NuGet oder im AWS SDK for .NET Github Repository.

Über AWS Mobile Services

Amazon Cognito-Identität

Alle AWS-Aufrufe setzen AWS-Anmeldeinformationen voraus. Anstatt Ihre Anmeldeinformationen in Ihren Apps fest zu codieren, empfehlen wir Ihnen, Amazon Cognito Identity zu verwenden, um AWS-Anmeldeinformationen für Ihre Anwendung bereitzustellen. Befolgen Sie die Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, um AWS-Anmeldeinformationen über Amazon Cognito abzurufen.

Mit Cognito können Sie Benutzer auch über öffentliche Anmeldeanbieter wie Amazon, Facebook, Twitter und Google sowie über Anbieter authentifizieren, die OpenID Connect unterstützen. Cognito funktioniert auch mit nicht authentifizierten Benutzern. Cognito stellt temporäre Anmeldeinformationen mit eingeschränkten Zugriffsrechten bereit, die Sie mit einer Identity and Access Management (IAM)-Rolle angeben. Cognito wird durch Erstellung eines Identitäten-Pools konfiguriert, der einer IAM-Rolle zugeordnet ist. Die IAM-Rolle gibt die Ressourcen und Services an, auf die die App zugreifen kann.

Informationen zu den ersten Schritten mit Cognito Identity erhalten Sie unter Einrichten von AWS Mobile SDK for .NET and Xamarin.

Weitere Informationen zu Cognito Identity erhalten Sie unter Amazon Cognito Identity.

Amazon Cognito Sync

Cognito Sync ist ein AWS-Service und eine Client-Bibliothek, mit der die geräteübergreifende Synchronisierung von anwendungsbezogenen Benutzerdaten möglich ist. Sie können die Cognito Sync-API verwenden, um Benutzerprofildaten geräteübergreifend und über Anmeldungsanbieter wie Amazon, Facebook und Google sowie benutzerdefinierte Identitätsanbieter zu synchronisieren.

Informationen zu den ersten Schritten mit Cognito Sync erhalten Sie unter Synchronisieren von Benutzerdaten mit Cognito Sync.

Weitere Informationen zu Cognito Sync erhalten Sie unter Amazon Cognito Sync.

Mobile Analytics

Mit Amazon Mobile Analytics können Sie Daten zur Nutzung Ihrer Mobilgeräte-Apps erfassen, visualisieren und analysieren. Berichte können für Kennzahlen zu aktiven Benutzern, Sitzungen, Bindung, In-App-Umsätzen und benutzerdefinierten Ereignissen erstellt und nach Plattform und Datumsbereich gefiltert werden. Amazon Mobile Analytics wächst mit Ihrem Unternehmen und kann pro Tag Milliarden Ereignisse von Millionen Endpunkten erfassen und verarbeiten.

Informationen zu den ersten Schritten mit Mobile Analytics erhalten Sie unter Verfolgen von App-Nutzungsdaten mit Amazon Mobile Analytics.

Weitere Informationen zu Mobile Analytics erhalten Sie unter Amazon Mobile Analytics.

Dynamo DB

Amazon DynamoDB ist ein schneller, hochgradig skalierbarer, hochverfügbarer, wirtschaftlicher, nicht relationaler Datenbankservice. DynamoDBMit werden Einschränkungen der

Datenspeicherungsskalierbarkeit eliminiert, die Latenz niedrig gehalten und die Leistung ist vorhersehbar.

Informationen zu den ersten Schritten mit Dynamo DB finden Sie unter Speichern und Abrufen von Daten mit DynamoDB .

Weitere Informationen zu Dynamo DB finden Sie unter Amazon DynamoDB .

Amazon Simple Notification Service

Amazon Simple Notification Service (SNS) ist ein schneller, flexibler und vollständig verwalteter Push-Benachrichtigungsdienst, über den Sie einzelne Nachrichten oder Rundsendungen an eine große Zahl von Empfängern senden können. Amazon Simple Notification Service ermöglicht das einfache und kostengünstige Senden von Push-Benachrichtigungen an Benutzer mobiler Geräte, E-Mail-Empfänger und sogar an andere verteilte Services.

Informationen zu den ersten Schritten mit SNS für Xamarin iOS erhalten Sie unter Empfangen von Push-Benachrichtigungen mit SNS (Xamarin iOS).

Informationen zu den ersten Schritten mit SNS für Xamarin Android erhalten Sie unter Empfangen von Push-Benachrichtigungen mit SNS (Xamarin Android).

Weitere Informationen zu SNS erhalten Sie unter Amazon Simple Notification Service (SNS).

-EinrichtungAWS Mobile SDK for .NET and Xamarin

Sie können AWS Mobile SDK for .NET and Xamarin einrichten und mit der Erstellung eines neuen Projekts beginnen oder das SDK in ein vorhandenes Projekt integrieren. Sie können die Beispiele auch klonen und ausführen, um eine Vorstellung davon zu erhalten, wie das SDK funktioniert. Befolgen Sie diese Schritte, um einzurichten und zu verwenden.AWS Mobile SDK for .NET and Xamarin.

Prerequisites

Vor Verwendung von AWS Mobile SDK for .NET and Xamarin müssen Sie die folgenden Arbeiten durchführen:

- Erstellen Sie ein AWS-Konto.
- Installieren Sie die Xamarin-Plattform.
- Windows-Benutzer: Xamarin: Getting Started for Windows
- Mac-Benutzer: Xamarin: Getting Started for Mac. (vollständig).

Sobald die Voraussetzungen erfüllt sind:

1. Rufen Sie AWS-Anmeldeinformationen mit ab.Amazon Cognito.
2. Legen Sie die erforderlichen Berechtigungen für jeden in der App verwendeten AWS-Service fest.
3. Erstellen Sie in der IDE ein neues Projekt.
4. Installieren des AWS Mobile SDK for .NET and Xamarin.
5. Konfigurieren der AWS Mobile SDK for .NET and Xamarin.

Schritt 1: Abrufen von AWS-Anmeldeinformationen

Um in der App AWS aufrufen zu können, müssen Sie zunächst AWS-Anmeldeinformationen abrufen. Hierzu wird Amazon Cognito verwendet, ein AWS-Service, mit dem die Anwendung auf die Services im SDK zugreifen kann, ohne private AWS-Anmeldeinformationen in die Anwendung einzubetten.

Um Amazon Cognito nutzen zu können, müssen Sie einen Identitäten-Pool erstellen. Ein Identitäten-Pool ist ein Speicher für kontenspezifische Daten. Er wird mittels einer eindeutigen Identitäten-Pool-ID identifiziert, die folgendermaßen aufgebaut ist.

```
"us-east-1:00000000-0000-0000-0000-000000000000"
```

1. Melden Sie sich bei der Amazon Cognito-Konsole an, wählen Sie Verbundidentitäten verwalten und wählen Sie anschließend Create new identity pool (Neuen Identitätenpool erstellen).
2. Geben Sie einen Namen für den Identitäten-Pool ein und aktivieren Sie das Kontrollkästchen, um den Zugriff für nicht authentifizierte Identitäten zu aktivieren. Wählen Sie Create Pool (Pool erstellen) aus, um Ihren Identitäten-Pool zu erstellen.
3. Wählen Sie Allow (Zulassen) aus, um die zwei Ihrem Identitäten-Pool zugeordneten Standardrollen zu erstellen: eine für nicht authentifizierte Benutzer und eine für authentifizierte Benutzer. Diese Standardrollen ermöglichen dem Identitäten-Pool Zugriff auf Amazon Cognito Sync und Amazon Mobile Analytics.

Üblicherweise verwenden Sie nur einen Identitäten-Pool pro Anwendung.

Nach dem Erstellen des Identitäten-Pool rufen Sie AWS-Anmeldeinformationen ab, indem Sie ein `CognitoAWSCredentials`-Objekt (durch Übergabe der Identitäten-Pool-ID) erstellen und es dann folgendermaßen dem Konstruktor eines AWS-Clients übergeben:

```
CognitoAWSCredentials credentials = new CognitoAWSCredentials (
    "us-east-1:00000000-0000-0000-0000-000000000000", // Your identity pool ID
    RegionEndpoint.USEast1 // Region
);

// Example for |MA|
analyticsManager = MobileAnalyticsManager.GetOrCreateInstance(
  credentials,
  RegionEndpoint.USEast1, // Region
  APP_ID // app id
);
```

Schritt 2: Festlegen der Berechtigungen

Sie müssen die Berechtigungen für jeden in der Anwendung zu verwendenden AWS-Service festlegen. Zunächst müssen Sie verstehen, wie AWS die Benutzer der Anwendung sieht.

Wenn jemand Ihre Anwendung verwendet und AWS-Aufrufe auslöst, weist AWS dem betreffenden Benutzer eine Identität zu. In dem in Schritt 1 erstellten Identitäten-Pool speichert AWS diese Identitäten. Es gibt zwei Arten von Identitäten: authentifizierte und nicht authentifizierte.

Authentifizierte Identitäten gehören Benutzern, die von einem öffentlichen Login-Anbieter (z. B. Facebook, Amazon, Google) authentifiziert wurden. Nicht authentifizierte Identitäten gehören Gastbenutzern.

Jede Identität ist einer AWS Identity and Access Management-Rolle zugeordnet. In Schritt 1 haben Sie zwei IAM-Rollen erstellt, eine für authentifizierte Benutzer und eine für nicht authentifizierter Benutzer. Jeder IAM-Rolle ist mindestens eine Richtlinie zugeordnet, die angibt, auf welche AWS-Services die Identitäten zugreifen können, die der Rolle zugewiesen sind. Mit der folgenden Beispielrichtlinie gewähren Sie Zugriff auf einen Amazon S3-Bucket.

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

Um Berechtigungen für die AWS-Services festzulegen, auf die in der Anwendung zugegriffen werden soll, ändern Sie einfach die den Rollen zugeordnete Richtlinie.

1. Navigieren Sie zur IAM-Konsole und wählen Sie "Roles (Rollen)". Geben Sie den Identitäten-Pool-Namen in das Suchfeld ein. Wählen Sie die IAM-Rolle aus, die Sie konfigurieren möchten. Wenn die Anwendung sowohl authentifizierte als auch nicht authentifizierte Benutzer zulässt, müssen Sie beiden Rollen Berechtigungen gewähren.
2. Klicken Sie auf Attach Policy (Richtlinie anfügen), wählen Sie die gewünschte Richtlinie und klicken Sie dann auf Attach Policy (Richtlinien anfügen). Die Standardrichtlinien für die von Ihnen erstellten IAM-Rollen gewähren Zugriff auf Amazon Cognito Sync und Mobile Analytics.

Weitere Informationen zum Erstellen von Richtlinien oder zum Auswählen von Richtlinien in einer Liste vorhandener Richtlinien erhalten Sie unter IAM-Richtlinien.

Schritt 3: Erstellen eines neuen -Projekts

Windows

Sie können Visual Studio oder Xamarin Studio zum Entwickeln der Anwendung verwenden.

OS X

Sie müssen die Xamarin Studio-IDE (Integrated Development Environment, integrierte Entwicklungsumgebung) zur Entwicklung der Anwendungen verwenden. Die iOS-Entwicklung mit Xamarin setzt Zugriff auf einen Mac voraus, auf dem die App ausgeführt werden kann. Weitere Informationen erhalten Sie unter Installing Xamarin.iOS on Windows.

Schritt 4: Installieren der AWS Mobile SDK for .NET and Xamarin

Windows

Option 1: Installieren mithilfe der Package Manager-Konsole

AWS Mobile SDK for .NET and Xamarin besteht aus einer Reihe von .NET-Komponenten. Führen Sie zum Installieren von AWS Mobile SDK for .NET and Xamarin den Befehl "install-package" für jedes Paket in der Package Manager-Konsole aus. Führen Sie zum Installieren von Cognito Identity beispielsweise Folgendes aus:

```
Install-Package AWSSDK.CognitoIdentity
```

Die Pakete AWS Core Runtime und Amazon Cognito Identity werden für alle Projekte benötigt. Nachstehend finden Sie eine vollständige Liste der Paketnamen für jeden Service.

Service	Package name
AWS Core Runtime	AWSSDK.Core
Amazon Cognito Sync	AWSSDK.CognitoSync
Amazon Cognito Identity	AWSSDK.CognitoIdentity
Amazon DynamoDB	AWSSDK.DynamoDBv2

Service	Package name
Amazon Mobile Analytics	AWSSDK.MobileAnalytics
Amazon S3	AWSSDK.S3
Amazon SNS	AWSSDK.SimpleNotificationService

Um ein Beta-Paket einzuschließen, fügen Sie beim Installieren des Pakets das Befehlszeilenargument `-Pre` ein:

```
Install-Package AWSSDK.CognitoSync -Pre
```

Eine vollständige Liste der AWS-Servicepakete finden Sie unter AWS SDK packages on NuGet oder im AWS SDK for .NET Github Repository.

Option 2: Installieren unter Verwendung der IDE

In Visual Studio

1. Klicken Sie mit der rechten Maustaste auf das Projekt und klicken Sie dann auf Manage Packages (NuGetPakete verwalten).
2. Suchen Sie nach dem Namen des Pakets, das Sie dem Projekt hinzufügen möchten. Um die NuGet prelease-Pakete einzuschließen, wählen Sie Include Prelease (Prelease einschließen) aus. Eine vollständige Liste der AWS-Service-Pakete finden Sie unter AWS SDK-Pakete unter NuGet .
3. Wählen sie das Paket und dann Installieren.

In Xamarin Studio

1. Klicken Sie mit der rechten Maustaste auf den Paketordner und wählen Sie dann Add Packages (Pakete hinzufügen).
2. Suchen Sie nach dem Namen des Pakets, das Sie dem Projekt hinzufügen möchten. Um die NuGet prelease-Pakete einzuschließen, wählen Sie Show pre-release packages (Vorversionspakete anzeigen) aus. Eine vollständige Liste der AWS-Service-Pakete finden Sie unter AWS SDK-Pakete unter NuGet .
3. Aktivieren Sie das Kontrollkästchen neben dem gewünschten Paket und wählen Sie Add Package (Paket hinzufügen).

Mac (OS X)

In Xamarin Studio

1. Klicken Sie mit der rechten Maustaste auf den Paketordner und wählen Sie dann Add Packages (Pakete hinzufügen).
2. Suchen Sie nach dem Namen des Pakets, das Sie dem Projekt hinzufügen möchten. Um die NuGet prelease-Pakete einzuschließen, wählen Sie Show pre-release packages (Vorversionspakete anzeigen) aus. Eine vollständige Liste der AWS-Service-Pakete finden Sie unter AWS SDK-Pakete unter NuGet .
3. Aktivieren Sie das Kontrollkästchen neben dem gewünschten Paket und wählen Sie Add Package (Paket hinzufügen).

> **Important**
>
> Wenn Sie eine Entwicklung mit einer Portable Class Library durchführen, müssen Sie das NuGet AWSSDK.Core-Paket auch allen Projekten hinzufügen, die aus der Portable Class Library abgeleitet werden.

Schritt 5: Konfigurieren von AWS Mobile SDK for .NET and Xamarin

Festlegen der Protokollierung

Sie legen Einstellungen für die Protokollierung mit den Klassen `Amazon.AWSConfigs` und `Amazon.Util.LoggingConfig` fest. Sie finden diese Klassen in der Komponente `AWSSdk.Core`, die über Nuget Package Manager in Visual Studio verfügbar ist. Sie können den Code, der zur Protokollierung der Einstellungen verwendet wird, in der Methode `OnCreate` der Datei `MainActivity.cs` für Android-Apps oder der Datei `AppDelegate.cs` für iOS-Apps platzieren. Sie sollten außerdem die Anweisungen `using Amazon` und `using Amazon.Util` in die CS-Dateien einfügen.

Konfigurieren Sie die Protokolleinstellungen wie folgt:

```
var loggingConfig = AWSConfigs.LoggingConfig;
loggingConfig.LogMetrics = true;
```

```
loggingConfig.LogResponses = ResponseLoggingOption.Always;
loggingConfig.LogMetricsFormat = LogMetricsFormatOption.JSON;
loggingConfig.LogTo = LoggingOptions.SystemDiagnostics;
```

Wenn Sie sich bei SystemDiagnosticsanmelden, druckt das Framework die Ausgabe intern an die System.Console. Wenn Sie HTTP-Antworten protokollieren möchten, setzen Sie das Flag `LogResponses` Die Werte können Always (Immer), Never (Nie) oder OnErrorsein.

Sie können außerdem Leistungskennzahlen zu HTTP-Anforderungen protokollieren, indem Sie die Eigenschaft `LogMetrics` verwenden. Das Protokollformat können Sie mit der Eigenschaft `LogMetricsFormat` angegeben. Gültige Werte sind "JSON" und "standard".

Festlegen des Regionsendpunkts

Konfigurieren Sie die Standardregion für alle Service-Clients wie folgt:

```
AWSConfigs.AWSRegion="us-east-1";
```

Dadurch wird die Standardregion für alle Service-Clients im SDK festgelegt. Sie können diese Einstellung überschreiben, indem Sie wie folgt die Region explizit zum Zeitpunkt der Erstellung einer Instance des Service-Clients angeben:

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

Konfigurieren der HTTP-Proxy-Einstellungen

Wenn Ihr Netzwerk hinter einem Proxy liegt, können Sie die Proxy-Einstellungen für die HTTP-Anforderungen wie folgt konfigurieren.

```
var proxyConfig = AWSConfigs.ProxyConfig;
proxyConfig.Host = "localhost";
proxyConfig.Port = 80;
proxyConfig.Username = "<username>";
proxyConfig.Password = "<password>";
```

Korrektur für Taktversatz

Diese Eigenschaft bestimmt, ob das SDK den Client-Taktversatz korrigieren soll, indem die richtige Server-Zeit ermittelt und die Anforderung mit der richtigen Zeit erneut ausgegeben wird.

```
AWSConfigs.CorrectForClockSkew = true;
```

Dieses Feld wird eingestellt, wenn ein Service-Aufruf eine Exception ausgelöst und das SDK festgestellt hat, dass die lokale Zeit und die Server-Zeit differieren.

```
var offset = AWSConfigs.ClockOffset;
```

Weitere Informationen zum Taktversatz finden Sie unter Clock-skew Correction im AWS-Blog.

Nächste Schritte

Nachdem Sie AWS Mobile SDK for .NET and Xamarin eingerichtet haben, können Sie folgende Operationen ausführen:

- Erste Schritte. Lesen Sie Erste Schritte mit AWS Mobile SDK for .NET and Xamarin für kurzgefasste Anleitungen zur Verwendung und zum Konfigurieren der Services in AWS Mobile SDK for .NET and Xamarin.
- Erkunden Sie die Service-Themen. Erfahren Sie mehr über die einzelnen Services und deren Funktionsweise in AWS Mobile SDK for .NET and Xamarin.
- Führen Sie die Demos aus. Sehen Sie sich unsere Xamarin-Beispielanwendungen an, die häufige Anwendungsfälle veranschaulichen. Richten Sie zum Ausführen der Beispiel-Apps AWS Mobile SDK for .NET and Xamarin wie zuvor beschrieben ein. Befolgen Sie dann die Anweisungen in den README-Dateien der jeweiligen Beispiele.
- Erfahren Sie mehr über das APIs. Zeigen Sie |sdk-xamarin-ref|_ an.
- Fragen: Veröffentlichen Sie Fragen in den AWS Mobile SDK-Foren oder öffnen Sie auf Github ein Problemticket.

Erste Schritte mit dem AWS Mobile SDK for .NET and Xamarin

AWS Mobile SDK for .NET and Xamarin stellt die Bibliotheken, Beispiele und Dokumentation bereit, die zum Aufrufen von AWS-Services aus Xamarin-Anwendungen benötigt werden.

Befolgen Sie alle Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, bevor Sie die folgenden Services verwenden können.

Diese "Erste Schritte"-Themen führen Sie durch Folgendes:

Themen

- Speichern und Abrufen von Daten mit Amazon S3
- Synchronisieren von Benutzerdaten mit Cognito Sync
- Speichern und Abrufen von Daten mit DynamoDB
- Verfolgen von App-Nutzungsdaten mit Amazon Mobile Analytics
- Empfangen von Push-Benachrichtigungen mit SNS (Xamarin iOS)
- Empfangen von Push-Benachrichtigungen mit SNS (Xamarin Android)

Weitere Informationen zu anderen AWS Mobile SDKsfinden Sie unter AWS Mobile SDK.

Speichern und Abrufen von Daten mit Amazon S3

Amazon Simple Storage Service (Amazon S3) stellt Entwicklern von Mobilgeräte-Apps sicheren, dauerhaften und hochskalierbaren Objektspeicher bereit. Amazon S3 ist bedienungsfreundlich und mit einer einfachen Web-Service-Schnittstelle zum Speichern und Abrufen beliebiger Datenmengen von jedem Ort über das Internet ausgestattet.

Im folgenden Tutorial wird erläutert, wie Sie den S3 TransferUtilityintegrieren, ein High-Level-Dienstprogramm für die Verwendung von S3 mit Ihrer App. Weitere Informationen über die Nutzung von S3 in Xamarin-Anwendungen erhalten Sie unter Amazon Simple Storage Service (S3).

Projekteinrichtung

Prerequisites

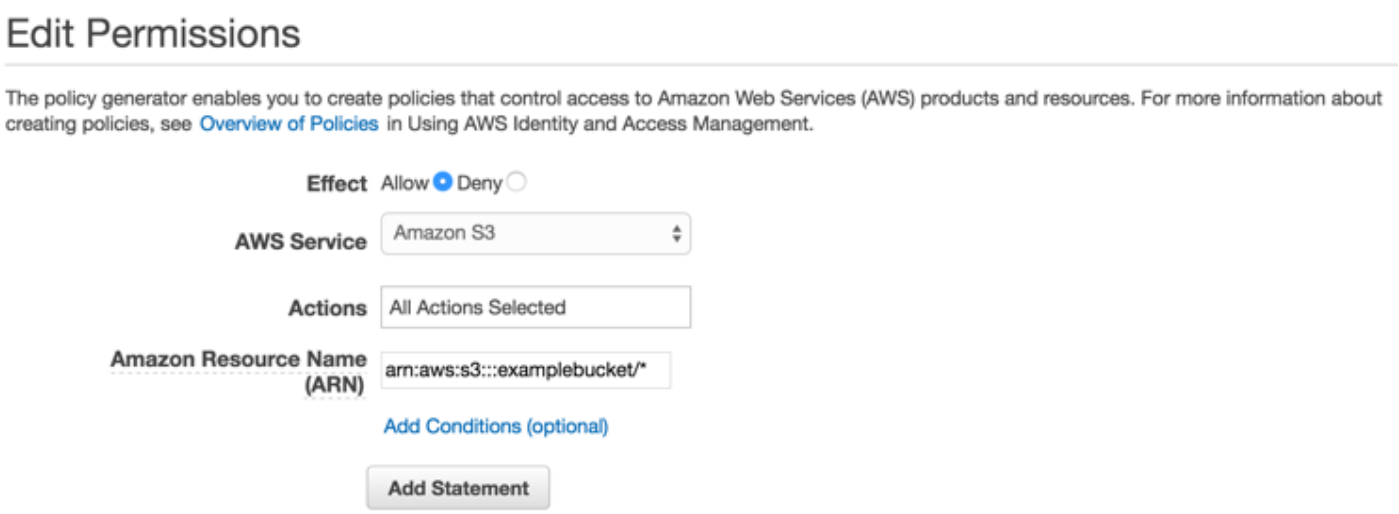
Befolgen Sie alle Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, bevor Sie mit diesem Tutorial beginnen.

In diesem Tutorial wird davon ausgegangen, dass Sie bereits einen S3-Bucket erstellt haben. Rufen Sie zum Erstellen eines S3-Bucket die S3 AWS-Konsole. auf.

Festlegen von Berechtigungen für S3

Die IAM-Standardrollenrichtlinie gewährt der Anwendung Zugriff auf Amazon Mobile Analytics und Amazon Cognito Sync. Damit der Cognito-Identitäten-Pool auf Amazon S3 zugreift, müssen Sie die Rollen des Identitäten-Pools modifizieren.

1. Navigieren Sie zur Identity and Access Management Console und klicken Sie im linken Bereich auf Roles.
2. Geben Sie den Identitäten-Pool-Namen in das Suchfeld ein. Es werden zwei Rollen aufgelistet: eine für nicht authentifizierte Benutzer und eine für authentifizierte Benutzer.
3. Klicken Sie auf die Rolle für nicht authentifizierte Benutzer (an den Identitäten-Pool-Namen ist "unauth" angehängt).
4. Klicken Sie aufCreate Role Policy, wählen Sie Policy Generator und klicken Sie dann auf Select.
5. Geben Sie auf der Seite Edit Permissions (Berechtigungen bearbeiten) die in der folgenden Abbildung gezeigten Einstellungen ein und ersetzen Sie den Amazon-Ressourcennamen (ARN) durch Ihren eigenen. Der ARN eines S3-Bucket entspricht `arn:aws:s3:::examplebucket/*` und besteht aus der Region, in der sich der Bucket befindet, sowie dem Namen des Bucket. Die unten gezeigten Einstellungen gewähren dem Identitäten-Pool vollen Zugriff auf alle Aktionen für den angegebenen Bucket.



1. Klicken Sie auf die Schaltfläche Add Statement und anschließend auf Next Step.
2. Der Assistent zeigt die von Ihnen generierte Konfiguration. Klicken Sie auf Apply Policy.

Weitere Informationen zum Gewähren des Zugriffs auf S3 erhalten Sie unter Granting Access to an Amazon S3 Bucket.

Hinzufügen eines NuGet Pakets für S3 zu Ihrem Projekt

Befolgen Sie Schritt 4 der Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, um das S3 NuGet Ihrem Projekt hinzuzufügen.

(Optional) Konfigurieren der Signature Version for S3-Anforderungen

Jede Interaktion mit Amazon S3 erfolgt entweder authentifiziert oder anonym. AWS verwendet die Signature Version 4- oder Signature Version 2-Algorithmen zum Authentifizieren von Aufrufen des Services.

Alle neuen, nach Januar 2014 erstellten AWS-Regionen unterstützen nur Signature Version 4. Viele ältere Regionen unterstützen jedoch weiterhin Signature Version 4- und Signature Version 2-Anforderungen.

Wenn sich Ihr Bucket in einer der Regionen befindet, die Signature Version 2-Anforderungen nicht unterstützt, wie auf dieser Seite aufgeführt, müssen Sie die AWSConfigsS3.UseSignatureVersion4 Eigenschaft wie folgt auf „true“ setzen:

```
AWSConfigsS3.UseSignatureVersion4 = true;
```

Weitere Informationen zu AWS Signature-Versionen erhalten Sie unter Authenticating Requests (AWS Signature Version 4).

Initialisieren des S3TransferUtility

Erstellen Sie einen S3-Client, übergeben Sie die AWS-Anmeldeinformationen und dann den S3-Client an das Dienstprogramm für die Datenübertragung, z. B.:

```
var s3Client = new AmazonS3Client(credentials,region);
var transferUtility = new TransferUtility(s3Client);
```

Hochladen einer Datei nach Amazon S3

Um eine Datei nach S3 hochzuladen, rufen Sie `Upload` für das Transfer Utility-Objekt auf und übergeben dabei die folgenden Parameter:

- `file`: Name der hochzuladenden Datei als String
- `bucketName`: Name des S3-Bucket, in dem die Datei gespeichert werden soll, als String

```
transferUtility.Upload(
  Path.Combine(Environment.SpecialFolder.ApplicationData,"file"),
  "bucketName"
);
```

Im Code oben wird davon ausgegangen, dass sich eine Datei im Verzeichnis Environment.SpecialFolder.ApplicationData befindet. Beim Hochladen wird automatisch die S3-Funktion für mehrteiliges Hochladen großer Dateien verwendet, um den Durchsatz zu verbessern.

Herunterladen einer Datei aus Amazon S3

Um eine Datei aus S3 herunterzuladen, rufen Sie `Download` für das Transfer Utility-Objekt auf und übergeben dabei die folgenden Parameter:

- `file`: Name der herunterzuladenden Datei als String
- `bucketName`: Name des S3-Bucket, aus dem die Datei heruntergeladen werden soll

- `key`: String, der den Namen des herunterzuladenden S3-Objekts (in diesem Fall eine Datei) angibt

```
transferUtility.Download(
  Path.Combine(Environment.SpecialFolder.ApplicationData,"file"),
  "bucketName",
  "key"
);
```

Weitere Informationen zum Zugriff auf Amazon S3 aus einer Xamarin-Anwendung erhalten Sie unter Amazon Simple Storage Service (S3).

Synchronisieren von Benutzerdaten mit Cognito Sync

Mit Amazon Cognito Sync können Sie mobile Benutzerdaten (wie App-Benutzereinstellungen oder Spielstände) einfach in der AWS Cloud speichern, ohne dafür Backend-Code zu schreiben oder eine Infrastruktur zu verwalten. Sie können Daten lokal auf den Geräten der Benutzer speichern. So funktionieren Ihre Anwendungen auch dann, wenn die Geräte offline sind. Außerdem können Sie die Daten über die Geräte eines Benutzers synchronisieren und ein konsistentes App-Erlebnis schaffen – unabhängig vom verwendeten Gerät.

Das Tutorial unten erläutert, wie Sync in die App integriert werden kann.

Projekteinrichtung

Prerequisites

Befolgen Sie alle Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, bevor Sie mit diesem Tutorial beginnen.

Gewähren des Zugriffs auf Cognito Sync-Ressourcen

Die Standardrichtlinie, die den im Rahmen der Einrichtung erstellten Rollen für nicht authentifizierte und authentifizierte Benutzer zugeordnet ist, gewährt der Anwendung Zugriff auf Cognito Sync. Weitere Konfigurationsarbeiten sind nicht erforderlich.

Hinzufügen eines NuGet Pakets für Cognito Sync zu Ihrem Projekt

Befolgen Sie Schritt 4 der Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, um das SyncManager NuGet Cognito-Paket Ihrem Projekt hinzuzufügen.

Initialisieren des CognitoSyncManager

Übergeben Sie den initialisierten Amazon Cognito-Anmeldeinformationsanbieter an den `CognitoSyncManager`-Konstruktor:

```
CognitoSyncManager syncManager = new CognitoSyncManager (
    credentials,
    new AmazonCognitoSyncConfig {
        RegionEndpoint = RegionEndpoint.USEast1 // Region
    }
);
```

Synchronisieren von Benutzerdaten

Synchronisieren Sie die Daten nicht authentifizierter Benutzer wie folgt:

1. Erstellen Sie ein Dataset.
2. Fügen Sie dem Dataset Benutzerdaten hinzu.
3. Synchronisieren Sie das Dataset mit der Cloud.

Erstellen eines Dataset

Erstellen Sie eine Instance von `Dataset`. Die openOrCreateDataset -Methode wird verwendet, um ein neues Dataset zu erstellen oder eine vorhandene Instance eines lokal auf dem Gerät gespeicherten Datasets zu öffnen:

```
Dataset dataset = syncManager.OpenOrCreateDataset("myDataset");
```

Hinzufügen von Benutzerdaten zum Dataset

Benutzerdaten werden in Form von Schlüssel-Wert-Paaren hinzugefügt:

```
dataset.OnSyncSuccess += SyncSuccessCallback;
dataset.Put("myKey", "myValue");
```

Cognito-Datasets fungieren als Wörterbücher, auf deren Werte mit einem Schlüssel zugegriffen werden kann:

```
string myValue = dataset.Get("myKey");
```

Synchronisieren von Datasets

Rufen Sie zum Synchronisieren eines Dataset die Synchronisierungsmethode auf:

```
dataset.SynchronizeAsync();

void SyncSuccessCallback(object sender, SyncSuccessEventArgs e) {
  // Your handler code here
}
```

Alle in Datasets geschriebenen Daten werden lokal gespeichert, bis das Dataset synchronisiert wird. Beim Code in diesem Abschnitt ist unterstellt, dass Sie eine nicht authentifizierte Cognito-Identität verwenden, damit die mit der Cloud synchronisierten Benutzerdaten auf jedem Gerät gespeichert werden. Dem Gerät ist eine Geräte-ID zugeordnet. Wenn die Benutzerdaten in die Cloud synchronisiert werden, werden sie mit dieser Geräte-ID verknüpft.

Weitere Informationen zu Cognito Sync erhalten Sie unter Amazon Cognito Sync.

Speichern und Abrufen von Daten mit DynamoDB

Amazon DynamoDB ist ein schneller, hochgradig skalierbarer, hochverfügbarer, wirtschaftlicher, nicht relationaler Datenbankservice. DynamoDBMit werden Einschränkungen der Datenspeicherungsskalierbarkeit eliminiert, die Latenz niedrig gehalten und die Leistung ist vorhersehbar.

Im folgenden Tutorial wird erläutert, wie Sie das DynamoDB Object Persistence-Modell in Ihre App integrieren, in der Objekte in gespeichert DynamoDBwerden.

Projekteinrichtung

Prerequisites

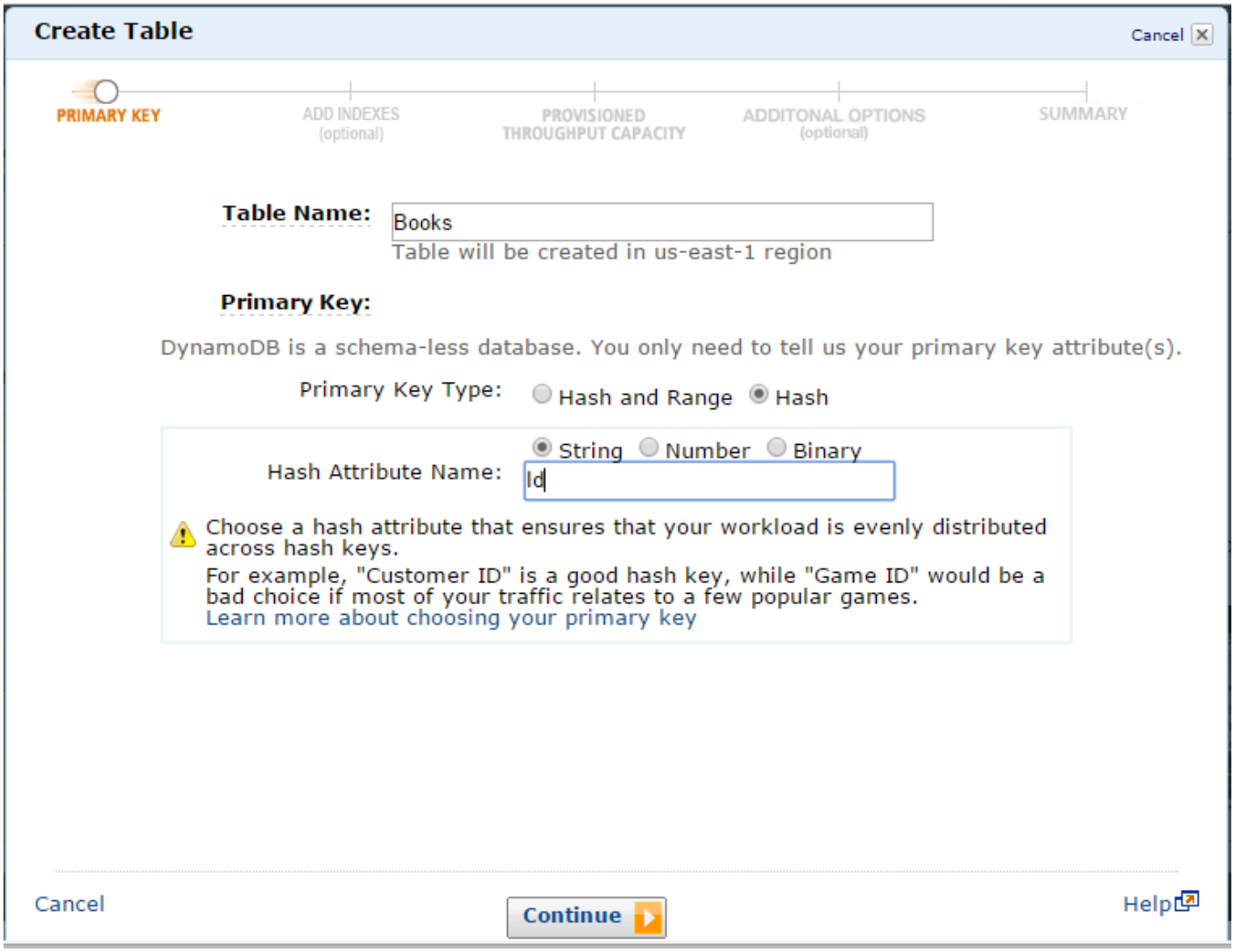
Befolgen Sie alle Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, bevor Sie mit diesem Tutorial beginnen.

Erstellen einer DynamoDB Tabelle

Bevor Sie Daten in einer DynamoDB Datenbank lesen und schreiben können, müssen Sie eine Tabelle erstellen. Beim Erstellen der Tabelle müssen Sie den Primärschlüssel angeben. Der Primärschlüssel besteht aus einem Hash-Attribut und einem optionalen Bereichsattribut. Weitere

Informationen zur Verwendung primärer Attribute und Bereichsattribute finden Sie unter Arbeiten mit Tabellen.

1. Navigieren Sie zur DynamoDB -Konsole und klicken Sie auf Create Table (Tabelle erstellen). Der Assistent "Create Table" wird angezeigt.
2. Geben Sie den Tabellennamen, den Typ des Primärschlüssels (Hash) und den Namen des Hash-Attributs („Id“) wie unten dargestellt ein. Klicken Sie dann auf Continue (Weiter):



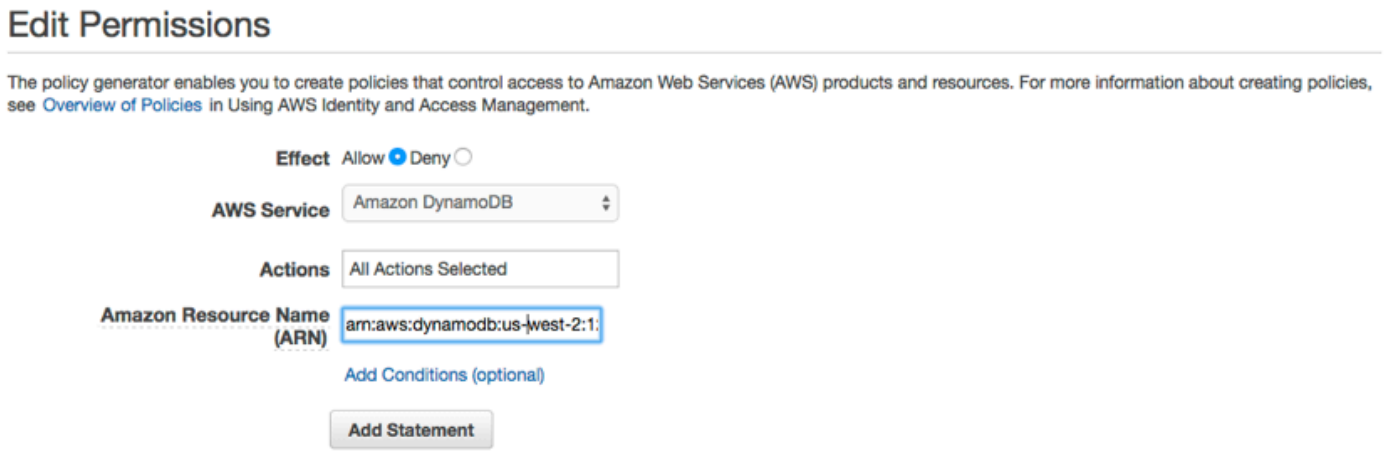
3. Lassen Sie die Felder im nächsten Bildschirm leer und klicken Sie auf Continue (Weiter).
4. Übernehmen Sie die Standardwerte für Read Capacity Units (Lesekapazitätseinheiten) und Write Capacity Units (Schreibkapazitätseinheiten) und klicken Sie auf Continue (Weiter).
5. Geben Sie im nächsten Bildschirm Ihre E-Mail-Adresse in das Textfeld Send notification to: (Benachrichtigung senden an:) ein und klicken Sie auf Continue (Weiter). Der Zusammenfassungsbildschirm wird angezeigt.

6. Klicken Sie auf Create. Es kann einige Minuten dauern, bis die Tabelle erstellt wurde.

Festlegen von -Berechtigungen für DynamoDB

Damit Ihr Identitäten-Pool auf Amazon zugreifen DynamoDBkann, müssen Sie die Rollen des Identitäten-Pools ändern.

1. Navigieren Sie zur Identity and Access Management Console und klicken Sie im linken Bereich auf Roles. Suchen Sie Ihren Identitäten-Pool-Namen. Aufgelistet werden zwei Rollen, eine für nicht authentifizierte Benutzer, die andere für authentifizierte Benutzer.
2. Klicken Sie auf die Rolle für nicht authentifizierte Benutzer (an den Identitäten-Pool-Namen ist „unauth“ angehängt) und dann auf Create Role Policy (Rollenrichtlinie erstellen).
3. Wählen Sie Policy Generator (Richtliniengenerator) und klicken Sie auf Select (Auswählen).
4. Geben Sie auf der Seite Edit Permissions (Berechtigungen bearbeiten) die in der folgenden Abbildung gezeigten Einstellungen ein. Der Amazon-Ressourcenname (ARN) einer DynamoDB Tabelle sieht aus wie `arn:aws:dynamodb:us-west-2:123456789012:table/Books` und besteht aus der Region, in der sich die Tabelle befindet, der AWS-Kontonummer des Besitzers und dem Namen der Tabelle im Format `table/Books`. Weitere Informationen zum Angeben von ARNsfinden Sie unter Amazon-Ressourcennamen für DynamoDB .



5. Klicken Sie auf Add Statement (Anweisung hinzufügen) und anschließend auf Next Step (Nächster Schritt). Der Assistent zeigt die generierte Konfiguration.
6. Klicken Sie auf Apply Policy.

Hinzufügen NuGet des Pakets für DynamoDB zu Ihrem Projekt

Befolgen Sie Schritt 4 der Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, um das -DynamoDBNuGetPaket Ihrem Projekt hinzuzufügen.

Initialisieren AmazonDynamoDBClient

Übergeben Sie Ihren initialisierten Amazon Cognito und Ihre Region an den `AmazonDynamoDB` Konstruktor und dann den Client an den DynamoDBContext:

```
var client = new AmazonDynamoDBClient(credentials,region);
DynamoDBContext context = new DynamoDBContext(client);
```

Erstellen einer Klasse

Um eine Zeile in die Tabelle zu schreiben, definieren Sie eine Klasse, die die Zeilendaten aufnimmt. Die Klasse sollte auch Eigenschaften enthalten, die die Attributdaten für die Zeile enthalten und der in der -Konsole erstellten DynamoDB Tabelle zugeordnet werden. Die folgende Klassendeklaration zeigt eine solche Klasse:

```
[DynamoDBTable("Books")]
public class Book
{
    [DynamoDBHashKey]    // Hash key.
    public int Id { get; set; }
    public string Title { get; set; }
    public string ISBN { get; set; }
    public int Price { get; set; }
    public string PageCount { get; set; }
    public string Author{ get; set; }
}
```

Speichern eines Elements

Erstellen Sie zum Speichern eines Elements zunächst ein Objekt:

```
Book songOfIceAndFire = new Book()
{
    Id=1,
    Title="Game Of Thrones",
    ISBN="978-0553593716",
```

```
    Price=4,
    PageCount="819",
    Author="GRRM"
};
```

Speichern Sie es dann:

```
context.Save(songOfIceAndFire);
```

Ändern Sie zum Aktualisieren einer Zeile die Instance der `DDTableRow`-Klasse und rufen Sie wie oben gezeigt `AWSDynamoObjectMapper.save()` auf.

Abrufen eines Elements

Abrufen eines Elements mit einem Primärschlüssel:

```
Book retrievedBook = context.Load<Book>(1);
```

Aktualisieren eines Elements

Aktualisieren Sie ein Element wie folgt:

```
Book retrievedBook = context.Load<Book>(1);
retrievedBook.ISBN = "978-0553593716";
context.Save(retrievedBook);
```

Löschen eines Elements

So löschen Sie einen Artikel:

```
Book retrievedBook = context.Load<Book>(1);
context.Delete(retrievedBook);
```

Weitere Informationen zum Zugriff auf DynamoDB über eine Xamarin-Anwendung finden Sie unter Amazon DynamoDB .

Verfolgen von App-Nutzungsdaten mit Amazon Mobile Analytics

Mit Amazon Mobile Analytics können Sie App-Nutzung und App-Umsatz messen. Indem Sie wichtige Trends – beispielsweise neue und zurückkehrende Nutzer, App-Umsatz, Nutzerbindung – und

Ereignisse zu speziellen Verhaltensweisen in der App nachverfolgen, können Sie Entscheidungen auf Basis entsprechender Daten treffen und so die Bindung und Monetarisierung mit der App steigern.

Das Tutorial unten erläutert, wie Mobile Analytics in die App integriert werden kann.

Projekteinrichtung

Prerequisites

Befolgen Sie alle Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, bevor Sie mit diesem Tutorial beginnen.

Erstellen einer Anwendung in der Mobile Analytics-Konsole

Rufen Sie die Amazon Mobile Analytics auf und erstellen Sie eine App. Notieren Sie sich den `appId`-Wert, da Sie ihn später benötigen werden. Beim Erstellen einer App in der Mobile Analytics-Konsole müssen Sie eine Identitäten-Pool-ID angeben. Anweisungen zum Erstellen eines Identitäten-Pools erhalten Sie unter Einrichten von AWS Mobile SDK for .NET and Xamarin.

Weitere Informationen zum Arbeiten in der Konsole finden Sie im Benutzerhandbuch von Amazon Mobile Analytics.

Festlegen von Berechtigungen für Mobile Analytics

Die Standardrichtlinie, die den im Rahmen der Einrichtung erstellten Rollen zugeordnet ist, gewährt Zugriff auf Mobile Analytics. Weitere Konfigurationsarbeiten sind nicht erforderlich.

Hinzufügen eines NuGet Pakets für Mobile Analytics zu Ihrem Projekt

Befolgen Sie Schritt 4 der Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, um das Mobile Analytics-NuGetPaket Ihrem Projekt hinzuzufügen.

Konfigurieren der Mobile Analytics-Einstellungen

Mobile Analytics definiert einige Einstellungen, die in der Datei awsconfig.xml konfiguriert werden können:

```
var config = new MobileAnalyticsManagerConfig();
config.AllowUseDataNetwork = true;
config.DBWarningThreshold = 0.9f;
```

```
config.MaxDBSize = 5242880;
config.MaxRequestSize = 102400;
config.SessionTimeout = 5;
```

- AllowUseDataNetwork – Ein boolescher Wert, der angibt, ob die Sitzungsereignisse im Datennetzwerk gesendet werden.
- DBWarningThreshold – Dies ist die Begrenzung für die Größe der Datenbank, die, sobald sie erreicht ist, Warnprotokolle generiert.
- MaxDBSize – Dies ist die Größe der SQLIte Datenbank. Wenn die Datenbank die maximale Größe erreicht, werden alle weiteren Ereignisse ignoriert.
- MaxRequestSize – Dies ist die maximale Größe der Anforderung in Byte, die in einer HTTP-Anforderung an den Mobile Analytics Service übertragen werden soll.
- SessionTimeout – Dies ist das Zeitintervall, nachdem eine Anwendung in den Hintergrund gewechselt ist und wann die Sitzung beendet werden kann.

Die oben gezeigten Einstellungen sind Standardwerte für die betreffenden Konfigurationselemente.

Initialisieren MobileAnalyticsManager

Um Ihre zu MobileAnalyticsManagerinitialisieren, rufen Sie GetOrCreateInstance auf Ihrer auf `MobileAnalyticsManager`und übergeben Ihre AWS-Anmeldeinformationen, Ihre Region, Ihre Mobile Analytics und Ihr optionales Konfigurationsobjekt:

```
var manager = MobileAnalyticsManager.GetOrCreateInstance(
  "APP_ID",
  "Credentials",
  "RegionEndPoint",
  config
);
```

Verfolgen von Sitzungsereignissen

Xamarin Android

Überschreiben Sie die Methoden `OnPause()` und `OnResume()` der Aktivität so, dass sie Ereignisse aufzeichnen.

```
protected override void OnResume()
```

```
{
    manager.ResumeSession();
    base.OnResume();
}

protected override void OnPause()
{
    manager.PauseSession();
    base.OnPause();
}
```

Dies muss für jede Aktivität in der Anwendung implementiert werden.

Xamarin iOS

In Ihrer AppDelegate.cs:

```
public override void DidEnterBackground(UIApplication application)
{
    manager.PauseSession();
}

public override void WillEnterForeground(UIApplication application)
{
    manager.ResumeSession();
}
```

Weitere Informationen zu Mobile Analytics erhalten Sie unter Amazon Mobile Analytics.

Empfangen von Push-Benachrichtigungen mit SNS (Xamarin iOS)

In diesem Dokument wird erläutert, wie Push-Benachrichtigungen mit Amazon Simple Notification Service (SNS) und an eine Xamarin-iOS-Anwendung gesendet werden.AWS Mobile SDK for .NET and Xamarin.

Projekteinrichtung

Prerequisites

Befolgen Sie alle Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, bevor Sie mit diesem Tutorial beginnen.

Festlegen von Berechtigungen für SNS

Befolgen Sie Schritt 2 in Einrichten von AWS Mobile SDK for .NET and Xamarin, um die unten angegebene Richtlinie an die Rollen Ihrer Anwendung anzufügen. Die Anwendung erhält so die erforderlichen Berechtigungen für den Zugriff auf SNS:

1. Navigieren Sie zur IAM-Konsole und wählen Sie die IAM-Rolle aus, die Sie konfigurieren möchten.
2. Klicken Sie auf Attach Policy (Richtlinie anfügen), wählen Sie die AmazonSNSFullAccess Richtlinie aus und klicken Sie auf Attach Policy (Richtlinie anfügen).

> **Warning**
>
> Die Verwendung von AmazonSNSFullAccess wird in einer Produktionsumgebung nicht empfohlen. Wir verwenden die Richtlinie hier, damit Sie schnell mit der Arbeit beginnen können. Weitere Informationen über das Festlegen von Berechtigungen für eine IAM-Rolle erhalten Sie unter Overview of IAM Role Permissions.

Anfordern der Mitgliedschaft im Apple iOS Developer Program

Sie müssen die App auf einem physischen Gerät ausführen, um Push-Benachrichtigungen zu empfangen. Zum Ausführen der App auf einem Gerät benötigen Sie eine Mitgliedschaft im Apple iOS Developer Program. Sobald Sie eine Mitgliedschaft besitzen, können Sie mit Xcode eine Signaturidentität generieren. Weitere Informationen finden Sie in der App Distribution Quick Start-Dokumentation von Apple.

Erstellen eines iOS-Zertifikats

Zunächst müssen Sie ein iOS-Zertifikat erstellen. Anschließend müssen Sie ein Bereitstellungsprofil erstellen, das für Push-Benachrichtigungen konfiguriert ist. Hierzu gehen Sie wie folgt vor:

1. Navigieren Sie zum Apple Developer Member Center und klicken Sie auf Certificates, Identifiers & Profiles (Zertifikate, IDs und Profile).
2. Klicken Sie unter iOS Apps auf Identifiers, klicken Sie auf das Plus-Symbol in der oberen rechten Ecke der Webseite, um eine neue iOS-App-ID hinzuzufügen, und geben Sie eine App-ID-Beschreibung ein.
3. Scrollen Sie nach unten zum Abschnitt Add ID Suffix (ID-Suffix hinzufügen), wählen Sie Explicit App ID (Explizite App-ID) aus und geben Sie Ihre Bundle-ID ein.

4. Scrollen Sie nach unten zum Bereich App Services und wählen Sie Push Notifications.
5. Klicken Sie auf Weiter.
6. Klicken Sie auf Submit.
7. Klicken Sie auf Done (Fertig).
8. Wählen Sie die gerade erstellte App-ID aus und klicken Sie dann auf Edit (Bearbeiten).
9. Scrollen Sie nach unten zum Abschnitt Push Notifications (Push-Benachrichtigungen). Klicken Sie auf Create Certificate (Zertifkat erstellen) unter Development SSL Certificate (Entwicklung-SSL-Zertifikat).
10. Befolgen Sie die Anweisungen zum Erstellen einer CSR-Anforderung (Certificate Signing Request), laden Sie die Anforderung hoch und laden Sie ein SSL-Zertifikat herunter, das für die Kommunikation mit Apple Notification Service (APNS) verwendet wird.
11. Kehren Sie zur Seite Certificates, Identifiers & Profiles (Zertifikate, IDs und Profile) zurück. Klicken Sie auf All unter Provisioning Profiles.
12. Klicken Sie auf die Plus-Schaltfläche oben rechts, um ein neues Bereitstellungsprofil hinzuzufügen.
13. Wählen Sie iOS App Development (Entwicklung von iOS-Anwendungen) und klicken Sie dann auf Continue (Weiter).
14. Wählen Sie die App-ID und klicken Sie dann auf Continue (Weiter).
15. Wählen Sie Ihr Entwicklerzertifikat und klicken Sie dann auf Continue (Weiter).
16. Wählen Sie das Gerät und klicken Sie dann auf Continue (Weiter).
17. Geben Sie einen Profilnamen ein und klicken Sie dann auf Generate (Erstellen).
18. Laden Sie die Bereitstellungsdatei herunter und doppelklicken Sie dann auf die Datei, um das Bereitstellungsprofil zu installieren.

Weitere Informationen zur Bereitstellung eines Profils, das für Push-Benachrichtigungen konfiguriert ist, finden Sie in der Apple-Dokumentation Configuring Push Notifications.

Verwenden eines Zertifikats zum Erstellen eines Plattform-ARN in der SNS-Konsole

1. Führen Sie die KeyChain Zugriffs-App aus, wählen Sie unten links auf dem Bildschirm My Certificates (Meine Zertifikate) aus und klicken Sie dann mit der rechten Maustaste auf das SSL-Zertifikat, das Sie generiert haben, um eine Verbindung mit APNS herzustellen. Wählen Sie dann Export (Exportieren) aus. Sie werden aufgefordert, einen Namen für die Datei und ein Passwort zum Schutz des Zertifikats anzugeben. Das Zertifikat wird in einer P12-Datei gespeichert.

2. Navigieren Sie zur SNS-Konsole und klicken Sie auf der linken Seite des Bildschirms auf Applications (Anwendungen).
3. Klicken Sie auf Create platform application (Plattformanwendung erstellen), um eine neue SNS-Plattformanwendung zu erstellen.
4. Geben Sie einen Application Name. ein.
5. Wählen Sie Apple Development für Push notification platform (Push-Benachrichtigungsplattform).
6. Klicken Sie auf Choose File (Datei auswählen) und wählen Sie die P12 aus, die Sie beim Exportieren des SSL-Zertifikats erstellt haben.
7. Geben Sie das beim Exportieren des SSL-Zertifikats eingegebene Passwort ein und klicken Sie auf Load Credentials From File.
8. Klicken Sie auf Create platform application.
9. Wählen Sie die soeben erstellte Plattformanwendung und kopieren Sie den ARN der Anwendung. Sie werden den Namen in kommenden Schritten benötigen.

Hinzufügen eines NuGet Pakets für SNS zu Ihrem Projekt

Befolgen Sie Schritt 4 der Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, um das Amazon Simple Notification NuGet Service-Paket Ihrem Projekt hinzuzufügen.

Erstellen eines SNS-Clients

```
var snsClient = new AmazonSimpleNotificationServiceClient(credentials, region);
```

Registrieren der Anwendung für Remote-Benachrichtigungen

Um eine Anwendung zu registrieren, rufen Sie RegisterForRemoteNotifications für Ihr UIApplication -Objekt auf, wie unten gezeigt. Fügen Sie den folgenden Code in das AppDelegate.cs, Einfügen Ihres Plattformanwendungs-ARN ein, wenn Sie unten dazu aufgefordert werden:

```
public override bool FinishedLaunching(UIApplication app, NSDictionary options) {
// do something
var pushSettings = UIUserNotificationSettings.GetSettingsForTypes (
  UIUserNotificationType.Alert |
  UIUserNotificationType.Badge |
  UIUserNotificationType.Sound,
```

```
  null
);
app.RegisterUserNotifications(pushSettings);
app.RegisterForRemoteNotifications();
// do something
  return true;
}

public override void RegisteredForRemoteNotifications(UIApplication application, NSData
 token) {
  var deviceToken = token.Description.Replace("<", "").Replace(">", "").Replace(" ",
 "");
  if (!string.IsNullOrEmpty(deviceToken)) {
    //register with SNS to create an endpoint ARN
    var response = await SnsClient.CreatePlatformEndpointAsync(
    new CreatePlatformEndpointRequest {
      Token = deviceToken,
      PlatformApplicationArn = "YourPlatformArn" /* insert your platform application
 ARN here */
    });
  }
}
```

Senden einer Nachricht mit der SNS-Konsole an den Endpunkt

1. Navigieren Sie zu SNS Console (SNS-Konsole) > Applications (Anwendungen).
2. Wählen Sie die Plattformanwendung, dann einen Endpunkt und klicken Sie schließlich auf Publish to endpoint (In Endpunkt veröffentlichen).
3. Geben Sie eine Textnachricht in das Textfeld ein und klicken Sie auf Publish message (Nachricht veröffentlichen), um eine Nachricht zu veröffentlichen.

Empfangen von Push-Benachrichtigungen mit SNS (Xamarin Android)

Das Tutorial erläutert, wie Push-Benachrichtigungen mit Amazon Simple Notification Service (SNS) und an eine Xamarin-Android-Anwendung gesendet werden.AWS Mobile SDK for .NET and Xamarin.

Projekteinrichtung

Prerequisites

Befolgen Sie alle Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, bevor Sie mit diesem Tutorial beginnen.

Festlegen von Berechtigungen für SNS

Befolgen Sie Schritt 2 in Einrichten von AWS Mobile SDK for .NET and Xamarin, um die unten angegebene Richtlinie an die Rollen Ihrer Anwendung anzufügen. Die Anwendung erhält so die erforderlichen Berechtigungen für den Zugriff auf SNS:

1. Navigieren Sie zur IAM-Konsole und wählen Sie die IAM-Rolle aus, die Sie konfigurieren möchten.
2. Klicken Sie auf Attach Policy, wählen Sie die AmazonSNSFullAccess Richtlinie aus und klicken Sie auf Attach Policy.

> **Warning**
>
> Die Verwendung von AmazonSNSFullAccess wird in einer Produktionsumgebung nicht empfohlen. Wir verwenden die Richtlinie hier, damit Sie schnell mit der Arbeit beginnen können. Weitere Informationen über das Festlegen von Berechtigungen für eine IAM-Rolle erhalten Sie unter Overview of IAM Role Permissions.

Aktivieren von Push-Benachrichtigungen in der Google Cloud

Fügen Sie zunächst ein neues Google-API-Projekt hinzu:

1. Navigieren Sie zur Google Developers-Konsole.
2. Klicken Sie auf Create Project (Projekt erstellen).
3. Geben Sie einen Projektnamen in das Feld New Project (Neues Projekt) ein. Notieren Sie sich die (später benötigte) Projekt-ID und klicken Sie auf Create (Erstellen).

Aktivieren Sie nun den Service Google Cloud Messaging (GCM) für das Projekt:

1. In der Google Developers-Konsole sollte Ihr neues Projekt bereits ausgewählt sein. Wählen Sie es andernfalls im Dropdown-Menü oben auf der Seite aus.

2. Wählen Sie APIs in der Seitenleiste auf der linken Seite der Seite & auth aus.
3. Geben Sie in das Suchfeld „Google Cloud Messaging for Android“ ein und klicken Sie auf den Link Google Cloud Messaging for Android.
4. Klicken Sie auf Enable API.

Rufen Sie schließlich einen API-Schlüssel ab:

1. Wählen Sie in der Google Developers-Konsole APIs & auth > Credentials (Anmeldeinformationen) aus.
2. Klicken Sie unter Public API access auf Create new key.
3. Klicken Sie im Dialogfeld Create a new key auf Server key.
4. Klicken Sie im daraufhin angezeigten Dialogfeld auf Create (Erstellen) und kopieren Sie den API-Schlüssel. Sie werden diesen API-Schlüssel später zur Authentifizierung verwenden.

Verwenden der Projekt-ID zum Erstellen eines Plattform-ARN in der SNS-Konsole

1. Rufen Sie die SNS-Konsole. auf.
2. Klicken Sie auf der linken Seite des Bildschirms auf Applications (Anwendungen).
3. Klicken Sie auf Create platform application (Plattformanwendung erstellen), um eine neue SNS-Plattformanwendung zu erstellen.
4. Geben Sie einen Application Name. ein.
5. Wählen Sie Google Cloud Messaging (GCM) für Push notification platform (Push-Benachrichtigungsplattform).
6. Fügen Sie den API-Schlüssel in das Textfeld API key. ein.
7. Klicken Sie auf Create platform application.
8. Wählen Sie die soeben erstellte Plattformanwendung und kopieren Sie den ARN der Anwendung.

Hinzufügen eines NuGet Pakets für SNS zu Ihrem Projekt

Befolgen Sie Schritt 4 der Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, um das Amazon Simple Notification NuGet Service-Paket Ihrem Projekt hinzuzufügen.

Erstellen eines SNS-Clients

```
var snsClient = new AmazonSimpleNotificationServiceClient(credentials, region);
```

Registrieren der Anwendung für Remote-Benachrichtigungen

Um sich für Remote-Benachrichtigungen auf Android zu registrieren, müssen Sie einen BroadcastReceiver erstellen, der Google Cloud-Nachrichten empfangen kann. Ändern Sie den Paketnamen unten, wenn Sie dazu aufgefordert werden:

```
[BroadcastReceiver(Permission = "com.google.android.c2dm.permission.SEND")]
[IntentFilter(new string[] {
      "com.google.android.c2dm.intent.RECEIVE"
}, Categories = new string[] {
      "com.amazonaws.sns" /* change to match your package */
})]
[IntentFilter(new string[] {
      "com.google.android.c2dm.intent.REGISTRATION"
}, Categories = new string[] {
      "com.amazonaws.sns" /* change to match your package */
})]
[IntentFilter(new string[] {
      "com.google.android.gcm.intent.RETRY"
}, Categories = new string[] {
      "com.amazonaws.sns" /* change to match your package */
})]
public class GCMBroadcastReceiver: BroadcastReceiver {
      const string TAG = "PushHandlerBroadcastReceiver";
      public override void OnReceive(Context context, Intent intent) {
              GCMIntentService.RunIntentInService(context, intent);
              SetResult(Result.Ok, null, null);
      }
}

[BroadcastReceiver]
[IntentFilter(new[] {
      Android.Content.Intent.ActionBootCompleted
})]
public class GCMBootReceiver: BroadcastReceiver {
      public override void OnReceive(Context context, Intent intent) {
              GCMIntentService.RunIntentInService(context, intent);
              SetResult(Result.Ok, null, null);
```

```
      }
}
```

Nachfolgend finden Sie den Service, der die Push-Benachrichtigung von der empfängt BroadcastReceiver und die Benachrichtigung in der Benachrichtigungsleiste des Geräts anzeigt:

```
[Service]
 public class GCMIntentService: IntentService {
  static PowerManager.WakeLock sWakeLock;
  static object LOCK = new object();

  public static void RunIntentInService(Context context, Intent intent) {
    lock(LOCK) {
      if (sWakeLock == null) {
        // This is called from BroadcastReceiver, there is no init.
        var pm = PowerManager.FromContext(context);
        sWakeLock = pm.NewWakeLock(
        WakeLockFlags.Partial, "My WakeLock Tag");
      }
    }

    sWakeLock.Acquire();
    intent.SetClass(context, typeof(GCMIntentService));
    context.StartService(intent);
  }

  protected override void OnHandleIntent(Intent intent) {
    try {
      Context context = this.ApplicationContext;
      string action = intent.Action;

      if (action.Equals("com.google.android.c2dm.intent.REGISTRATION")) {
        HandleRegistration(intent);
      } else if (action.Equals("com.google.android.c2dm.intent.RECEIVE")) {
        HandleMessage(intent);
      }
    } finally {
      lock(LOCK) {
        //Sanity check for null as this is a public method
        if (sWakeLock != null) sWakeLock.Release();
      }
    }
  }
```

```
 private void HandleRegistration(Intent intent) {
   string registrationId = intent.GetStringExtra("registration_id");
   string error = intent.GetStringExtra("error");
   string unregistration = intent.GetStringExtra("unregistered");

   if (string.IsNullOrEmpty(error)) {
     var response = await SnsClient.CreatePlatformEndpointAsync(new
CreatePlatformEndpointRequest {
       Token = registrationId,
       PlatformApplicationArn = "YourPlatformArn" /* insert your platform application
ARN here */
     });
   }
 }

 private void HandleMessage(Intent intent) {
   string message = string.Empty;
   Bundle extras = intent.Extras;
   if (!string.IsNullOrEmpty(extras.GetString("message"))) {
     message = extras.GetString("message");
   } else {
     message = extras.GetString("default");
   }

   Log.Info("Messages", "message received = " + message);
   ShowNotification(this, "SNS Push", message);
   //show the message

 }

 public void ShowNotification(string contentTitle,
 string contentText) {
   // Intent
   Notification.Builder builder = new Notification.Builder(this)
     .SetContentTitle(contentTitle)
     .SetContentText(contentText)
     .SetDefaults(NotificationDefaults.Sound | NotificationDefaults.Vibrate)
     .SetSmallIcon(Resource.Drawable.Icon)
     .SetSound(RingtoneManager.GetDefaultUri(RingtoneType.Notification));

   // Get the notification manager:
   NotificationManager notificationManager =
this.GetSystemService(Context.NotificationService) as NotificationManager;
```

```
    notificationManager.Notify(1001, builder.Build());
  }
}
```

Senden einer Nachricht mit der SNS-Konsole an den Endpunkt

1. Navigieren Sie zu SNS Console (SNS-Konsole) > Applications (Anwendungen).
2. Wählen Sie die Plattformanwendung, dann einen Endpunkt und klicken Sie schließlich auf Publish to endpoint (In Endpunkt veröffentlichen).
3. Geben Sie eine Textnachricht in das Textfeld ein und klicken Sie auf Publish message (Nachricht veröffentlichen), um eine Nachricht zu veröffentlichen.

Amazon Cognito-Identität

Was ist Amazon Cognito Identity?

Mit Amazon Cognito Identity können Sie eindeutige Identitäten für Benutzer erstellen und sie mithilfe von Identitätsanbietern authentifizieren. Eine Identität ermöglicht es, temporäre AWS-Anmeldeinformationen mit eingeschränkten Berechtigungen für die Synchronisierung von Daten mit Amazon Cognito Sync oder den direkten Zugriff auf andere AWS-Services abzurufen. Amazon Cognito Identity unterstützt öffentliche Identitätsanbieter – Amazon, Facebook und Google – sowie nicht authentifizierte Identitäten. Es werden außerdem entwicklerauthentifizierte Identitäten unterstützt, sodass Sie Benutzer mit eigenen Backend-Authentifizierungsprozessen registrieren und authentifizieren können.

Weitere Informationen zu Cognito Identity erhalten Sie im Entwicklerhandbuch von Amazon Cognito.

Weitere Informationen zur regionalen Verfügbarkeit von Cognito Authentication finden Sie in der AWS-Tabelle der Regionen.

Verwenden eines öffentlichen Anbieters zum Authentifizieren von Benutzern

Mit Amazon Cognito Identity können Sie eindeutige Identitäten für Ihre Benutzer erstellen und diese für den sicheren Zugriff auf Ihre AWS-Ressourcen wie Amazon S3 oder Amazon DynamoDBauthentifizieren. Amazon Cognito Identity unterstützt öffentliche — IdentitätsanbieterAmazon, Facebook, Twitter/Digits, Google oder beliebige OpenID Connect-kompatible —Anbietersowie nicht authentifizierte Identitäten.

Informationen zur Verwendung öffentlicher Identitätsanbieter wie Amazon, Facebook, Twitter/Digits oder Google zum Authentifizieren von Benutzern finden Sie unter Externe Anbieter im Amazon Cognito.

Verwenden entwicklerauthentifizierter Identitäten

Amazon Cognito unterstützt ergänzend zum Web-Identitätsverbund über Facebook, Google und Amazon entwicklerauthentifizierte Identitäten. Mit entwicklerauthentifizierten Identitäten können Sie Benutzer über Ihren eigenen vorhandenen Authentifizierungsprozess registrieren und authentifizieren, während Sie Amazon Cognito Sync verwenden, um Benutzerdaten zu synchronisieren und auf AWS-Ressourcen zuzugreifen. Die Verwendung von

entwicklerauthentifizierten Identitäten beinhaltet die Interaktion zwischen dem Endbenutzergerät, dem Backend für die Authentifizierung und Amazon Cognito.

Weitere Informationen zu entwicklerauthentifizierten Identitäten finden Sie unter Entwicklerauthentifizierte Identitäten im Amazon Cognito.

Amazon Cognito Sync

Was ist Amazon Cognito Sync?

Cognito Sync ist eine AWS-Service- und Client-Bibliothek, die das geräteübergreifende Synchronisieren von Benutzerdaten (z. B. Spielstände, Benutzereinstellungen, Spielstatus) ermöglicht. Sie können die Cognito Sync-API verwenden, um Benutzerdaten gerätübergreifend zu synchronisieren. Um Cognito Sync in der App zu verwenden, müssen Sie | in das Projekt aufnehmen.

Anweisungen zum Integrieren von Amazon Cognito Sync in Ihre App finden Sie im Amazon Cognito Sync Entwicklerhandbuch.

Amazon Mobile Analytics

Amazon Mobile Analytics ist ein Service für die Erfassung, Visualisierung, Analyse und Extraktion von App-Nutzungsdaten in großem Umfang. Mobile Analytics erfasst problemlos Standardgerätedaten und benutzerdefinierte Ereignisse und stellt automatisch Berichte für Sie zusammen. Zusätzlich zu den unten aufgeführten zusammenfassenden Berichten können Sie das automatische Exportieren der Daten nach Redshift und S3 zur weiteren Analyse einrichten.

Mit Amazon Mobile Analytics können Sie das Kundenverhalten verfolgen, Daten generieren, Kennzahlen zusammenstellen, Datenvisualisierungen generieren und aussagekräftige Muster identifizieren.

Die wichtigsten Konzepte

Berichtstypen

Im Auslieferungszustand stellt Mobile Analytics die folgenden Berichte in der Mobile Analytics-Konsole bereit:

- Aktive Benutzer pro Tag (Daily Active Users, DAU) und pro Monat (Monthly Active Users, MAU) sowie neue Benutzer
- Treuefaktor (DAU geteilt durch MAU)
- Sitzungsanzahl und mittlere Anzahl Sitzungen pro aktivem Benutzer pro Tag
- Mittlerer Ertrag pro DAU (Average Revenue per Daily Active User, ARPDAU) und mittlerer Ertrag pro bezahlendem DAU (Average Revenue per Daily Paying Active User, ARPPDAU)
- Kundenbindung für 1, 3 und 7 Tage sowie für 1, 2 und 3 Wochen
- Benutzerdefinierte Ereignisse

Diese Berichte werden über sechs Berichtsregisterkarten in der Konsole bereitgestellt:

- Overview (Übersicht) – Verfolgen Sie neun vordefinierte Berichte in einem übersichtlichen Dashboard, um sich schnell einen Eindruck über die Nutzung zu verschaffen: MAU, DAU, New Users (Neue Benutzer), Daily Sessions (Tägliche Sitzungen), Sticky Factor (Treuefaktor), 1-Day Retention (Kundenbindung für einen Tag), ARPDAU, Daily Paying Users (Täglich zahlende Benutzer), ARPPDAU.

- Active Users (Aktive Benutzer) – Verfolgen Sie nach, wie viele Benutzer sich täglich und monatlich mit Ihrer App beschäftigen, und überwachen Sie den Treuefaktor, um Bindung, Attraktivität und Monetarisierung einschätzen zu können.
- Sessions (Sitzungen) – Verfolgen Sie, wie oft Ihre App an einem bestimmten Tag verwendet wird und wie oft jeder Benutzer Ihre App im Lauf eines Tages öffnet.
- Retention (Kundenbindung) – Verfolgen Sie auf täglicher und wöchentlicher Basis die Rate, mit der Kunden zur App zurückkehren.
- Revenue (Umsatz) – Verfolgen Sie die In-App-Umsatz-Trends, um Bereiche zu identifizieren, in denen die Monetarisierung verbessert werden sollte.
- Custom events (Benutzerdefinierte Ereignisse) – Verfolgen Sie benutzerdefinierte, für die App spezifische Benutzeraktionen.

Weitere Informationen zu Mobile Analytics und zum Arbeiten in der Mobile Analytics finden Sie in der Mobile Mobile Analytics unter Berichtsübersicht im Mobile Analytics.

Projekteinrichtung

Prerequisites

Um Mobile Analytics in der Anwendung verwenden zu können, müssen Sie das SDK in das Projekt aufnehmen. Befolgen Sie zu diesem Zweck die Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin.

Konfigurieren der Mobile Analytics-Einstellungen

Mobile Analytics definiert einige Einstellungen, die in der Datei awsconfig.xml konfiguriert werden können:

```
var config = new MobileAnalyticsManagerConfig();
config.AllowUseDataNetwork = true;
config.DBWarningThreshold = 0.9f;
config.MaxDBSize = 5242880;
config.MaxRequestSize = 102400;
config.SessionTimeout = 5;
```

- SessionTimeout – Wenn die App eine Zeit im Hintergrund bleibt, die größer ist als der SessionTimeout , beendet der Mobile Analytics die aktuelle Sitzung und es wird eine neue Sitzung

erstellt, wenn die App wieder in den Vordergrund kommt. Wir empfehlen die Verwendung von Werten zwischen 5 und 10. Der Standardwert lautet 5.

- MaxDBSize – Die maximale Größe der Datenbank (in Bytes), die für die lokale Speicherung von Ereignissen verwendet wird. Wenn die Datenbankgröße diesen Wert überschreitet, werden weitere Ereignisse ignoriert. Wir empfehlen die Verwendung von Werten zwischen 1 MB und 10 MB. Der Standardwert ist 5 242 880 (5 MB).
- DBWarningThreashold – Der Warnschwellenwert. Gültige Werte liegen zwischen 0 und 1. Wenn die Werte den Grenzwert überschreiten, werden Warnungsprotokolle erstellt. Der Standardwert lautet 0.9.
- MaxRequestSize – Die maximale Größe einer HTTP-Anforderung an den Mobile Analytics. Der Wert wird in Byte angegeben und kann zwischen 1 KB und 512 KB betragen. Der Standardwert lautet 102 400 (100 KB). Verwenden Sie keine Werte, die größer als 512 KB sind, da dies dazu führen kann, dass der Service die HTTP-Anforderung ablehnt.
- AllowUseDataNetwork – Ein Wert, der angibt, ob ein Service-Aufruf über ein Mobilfunknetz zulässig ist. Verwenden Sie diese Option mit Vorsicht, da sie zu erhöhter Datennutzung der Kunden führen kann.

Die oben gezeigten Einstellungen sind Standardwerte für die betreffenden Konfigurationselemente.

Integrieren von Mobile Analytics in die Anwendung

Die folgenden Abschnitte erläutern, wie Mobile Analytics in die App integriert werden kann.

Erstellen einer Anwendung in der Mobile Analytics-Konsole

Navigieren Sie zur Amazon Mobile Analytics und erstellen Sie eine App. Notieren Sie sich den `appId`-Wert, da Sie ihn später benötigen werden. Beim Erstellen einer App in der Mobile Analytics-Konsole müssen Sie eine Identitäten-Pool-ID angeben. Anweisungen zum Erstellen eines Identitäten-Pools erhalten Sie unter Einrichten von AWS Mobile SDK for .NET and Xamarin.

Weitere Informationen zum Arbeiten in der Mobile Analytics finden Sie unter Mobile Mobile Analytics Console Reports Overview im Mobile Analytics.

Erstellen eines MobileAnalyticsManager Clients

Um Ihre zu MobileAnalyticsManagerinitialisieren, rufen Sie GetOrCreateInstance auf Ihrer auf `MobileAnalyticsManager`und übergeben Ihre AWS-Anmeldeinformationen, Ihre Region, Ihre Mobile Analytics und Ihr optionales Konfigurationsobjekt:

```
// Initialize the MobileAnalyticsManager
analyticsManager = MobileAnalyticsManager.GetOrCreateInstance(
  cognitoCredentials,
  RegionEndpoint.USEast1,
  APP_ID,
  config
);
```

Die `APP_ID` wird bei Ausführung des Assistenten zur App-Erstellung generiert. Beide Werte müssen denen in der Mobile Analytics-Konsole entsprechen. Die `APP_ID` wird verwendet, um die Daten in der Mobile Analytics-Konsole zu gruppieren. Sie können die App-ID nach dem Erstellen der App in der Mobile Analytics-Konsole ermitteln, indem Sie zur Mobile Analytics-Konsole navigieren und oben rechts auf dem Bildschirm auf das Zahnradsymbol klicken. Dadurch wird die Seite App Management (App-Verwaltung) angezeigt, auf der alle registrierten Apps und deren App aufgelistet IDssind.

Aufzeichnen von Monetarisierungsereignissen

AWS Mobile SDK for .NET and Xamarin stellt die Klasse `MonetizationEvent` bereit, mit der Sie Monetarisierungsereignisse generieren können, um in der Mobilgeräte-Anwendung getätigte Käufe zu verfolgen. Der folgende Code-Ausschnitt zeigt, wie ein Monetarisierungsereignis erstellt wird:

```
// 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);
```

Aufzeichnen benutzerdefinierter Ereignisse

Mit Mobile Analytics können Sie benutzerdefinierte Ereignisse definieren. Benutzerdefinierte Ereignisse werden vollständig von Ihnen definiert. Sie dienen der Verfolgung von Benutzeraktionen, die für die jeweilige App oder das Spiel spezifisch sind. Weitere Informationen zu benutzerdefinierten Ereignissen erhalten Sie unter Custom-Events.

In diesem Beispiel gehen wir davon aus, dass es sich bei der App um ein Spiel handelt. Wir möchten als Ereignis aufzeichnen, wenn ein Benutzer einen Level abschließt. Erstellen Sie ein „LevelComplete“-Ereignis, indem Sie eine neue `AmazonMobileAnalyticsEvent`-Instance erstellen:

```
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);
```

Aufzeichnen von Sitzungen

Xamarin iOS

Wenn die Anwendung den Fokus abgibt, kann die Sitzung angehalten werden. Für iOS-Apps überschreiben Sie in der AppDelegate.cs Datei `DidEnterBackground` und , `WillEnterForeground` um `MobileAnalyticsManager.PauseSesion` und `MobileAnalyticsManager.ResumeSession` aufzurufen, wie im folgenden Codeausschnitt gezeigt:

```
public override void DidEnterBackground(UIApplication application)
{
```

```
  // ...
  _manager.PauseSesson();
  // ...
}

public override void WillEnterForeground(UIApplication application)
{
  // ...
  _manager.ResumeSession();
  // ...
}
```

Xamarin Android

Für Android-Apps rufen Sie `MobileAnalyticsManager.PauseSesion` in der OnPause()-Methode und `MobileAnalyticsManager.ResumeSession` in der OnResume()-Methode auf, wie im folgenden Codeausschnitt gezeigt:

```
protected override void OnResume()
{
  _manager.ResumeSession();
  base.OnResume();
}

protected override void OnPause()
{
  _manager.PauseSession();
  base.OnPause();
}
```

Wenn der Benutzer den Fokus von der App entfernt und für weniger als 5 Sekunden einem anderen Prozess übergibt, bevor er wieder der App zugewiesen wird, wird die Sitzung fortgesetzt. Übergibt der Benutzer den Fokus für 5 Sekunden oder mehr an einen anderen Prozess, wird eine neue Sitzung erstellt. Diese Einstellung kann in der Konfigurationsdatei aws_mobile_analytics.json konfiguriert werden, indem der Eigenschaft "SESSION_DELTA" die Anzahl der Sekunden zugewiesen wird, die bis zum Erstellen einer neuen Sitzung verstreichen muss.

Amazon Simple Storage Service (S3)

Was ist S3?

Amazon Simple Storage Service (Amazon S3) stellt Entwicklern sicheren, dauerhaften und hochskalierbaren Objektspeicher bereit. Amazon S3 ist bedienungsfreundlich und mit einer einfachen Web-Service-Schnittstelle zum Speichern und Abrufen beliebiger Datenmengen von jedem Ort über das Internet ausgestattet. Mit Amazon S3 zahlen Sie nur für den Speicherplatz, den Sie tatsächlich nutzen. Es fallen weder Mindestgebühren noch Einrichtungskosten an.

Amazon S3 stellt kostengünstigen Objektspeicher für eine Vielzahl an Anwendungsfällen bereit, z. B. Cloud-Anwendungen, Inhaltsverteilung, Sicherung und Archivierung, Wiederherstellung im Katastrophenfall und Big Data-Analyse.

Weitere Informationen zur regionalen Verfügbarkeit von AWS S3 finden Sie in der AWS-Tabelle der Regionen.

Die wichtigsten Konzepte

Bucket

Alle in Amazon S3 gespeicherten Objekte werden in einem Bucket aufbewahrt. Sie können Buckets verwenden, um verwandte Objekte auf ähnliche Weise zu gruppieren, wie Sie ein Verzeichnis verwenden, um Dateien in einem Dateisystem zu gruppieren. Buckets besitzen Eigenschaften wie Zugriffsberechtigungen und Versioning-Status. Außerdem können Sie die Region angeben, in der sich die Buckets befinden sollen.

Weitere Informationen zu S3 finden Sie unter Arbeiten mit Buckets im S3.

Objects

Objekte sind die Daten, die Sie in Amazon S3 speichern. Jedes Objekt befindet sich in einem Bucket, den Sie in einer bestimmten AWS-Region erstellen.

In einer Region gespeicherte Objekte verbleiben so lange in der Region, bis sie explizit in eine andere Region verschoben werden. In der Region EU (Irland) gespeicherte Objekte verlassen diese Region nicht. Die in einer Amazon S3-Region gespeicherten Objekte verbleiben physisch in dieser Region. Amazon S3 speichert keine Kopien und verschiebt sie nicht in eine andere Region. Sie

können jedoch von beliebigen Orten aus auf die Objekte zugreifen, sofern Sie über die erforderlichen Berechtigungen verfügen.

Die Objekte können von jedem Dateityp sein: Bilder, Sicherungsdaten, Filme, usw. Ein Objekt kann bis zu 5 TB groß sein. Ein Bucket kann eine unbegrenzte Anzahl von Objekten aufnehmen.

Bevor Sie ein Objekt nach Amazon S3 hochladen können, benötigen Sie Schreibberechtigungen für einen Bucket. Weitere Informationen zum Festlegen von Bucket-Berechtigungen finden Sie unter Bearbeiten von Bucket-Berechtigungen im S3.

Weitere Informationen zu S3 finden Sie unter Arbeiten mit Objekten im S3.

Objekt-Metadaten

Jedes Objekt in Amazon S3 weist eine Gruppe von Schlüssel-Wert-Paaren auf, die seine Metadaten repräsentieren. Es gibt zwei Typen von Metadaten:

- System-Metadaten: Gelegentlich von Amazon S3 verarbeitet, z. B. Content-Type und Content-Length.
- Benutzer-Metadaten: Niemals von Amazon S3 verarbeitet. Benutzer-Metadaten werden mit dem Objekt gespeichert und mit ihm zurückgegeben. Die maximale Größe für Benutzer-Metadaten beträgt 2 KB. Sowohl die Schlüssel als auch deren Werte müssen US-ASCII-Standards entsprechen.

Weitere Informationen über S3-Objekt-Metadaten erhalten Sie unter Bearbeiten von Objekt-Metadaten.

Projekteinrichtung

Prerequisites

Um Amazon S3 in der Anwendung verwenden zu können, müssen Sie das SDK in das Projekt aufnehmen. Befolgen Sie zu diesem Zweck die Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin.

Erstellen eines S3-Bucket

Amazon S3 speichert die Ressourcen der Anwendung in Amazon S3-Buckets, also in Cloud-Speicher-Containern, die sich in einer bestimmten Region. befinden. Jeder Amazon S3-Bucket muss

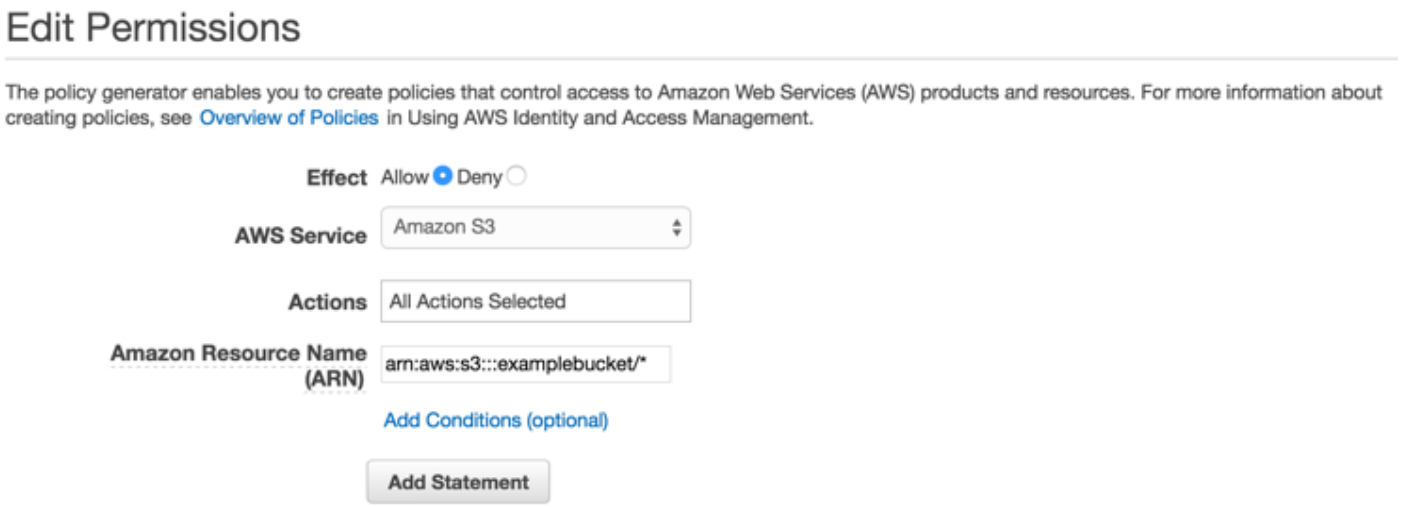
einen global eindeutigen Namen tragen. Sie können die Amazon S3 verwenden, um einen Bucket zu erstellen.

1. Melden Sie sich bei der Amazon S3-Konsole an und klicken Sie auf Create Bucket.
2. Geben Sie einen Bucket-Namen ein, wählen Sie eine Region und klicken Sie auf Create.

Festlegen von Berechtigungen für S3

Die IAM-Standardrollenrichtlinie gewährt der Anwendung Zugriff auf Amazon Mobile Analytics und Amazon Cognito Sync. Damit der Cognito-Identitäten-Pool auf Amazon S3 zugreift, müssen Sie die Rollen des Identitäten-Pools modifizieren.

1. Navigieren Sie zur Identity and Access Management Console und klicken Sie im linken Bereich auf Roles.
2. Geben Sie den Identitäten-Pool-Namen in das Suchfeld ein. Es werden zwei Rollen aufgelistet: eine für nicht authentifizierte Benutzer und eine für authentifizierte Benutzer.
3. Klicken Sie auf die Rolle für nicht authentifizierte Benutzer (an den Identitäten-Pool-Namen ist "unauth" angehängt).
4. Klicken Sie aufCreate Role Policy, wählen Sie Policy Generator und klicken Sie dann auf Select.
5. Geben Sie auf der Seite Edit Permissions (Berechtigungen bearbeiten) die in der folgenden Abbildung gezeigten Einstellungen ein und ersetzen Sie den Amazon-Ressourcennamen (ARN) durch Ihren eigenen. Der ARN eines S3-Bucket entspricht `arn:aws:s3:::examplebucket/*` und besteht aus der Region, in der sich der Bucket befindet, sowie dem Namen des Bucket. Die unten gezeigten Einstellungen gewähren dem Identitäten-Pool vollen Zugriff auf alle Aktionen für den angegebenen Bucket.



1. Klicken Sie auf die Schaltfläche Add Statement und anschließend auf Next Step.
2. Der Assistent zeigt die von Ihnen generierte Konfiguration. Klicken Sie auf Apply Policy.

Weitere Informationen zum Gewähren des Zugriffs auf S3 erhalten Sie unter Granting Access to an Amazon S3 Bucket.

(Optional) Konfigurieren der Signature Version for S3-Anforderungen

Jede Interaktion mit Amazon S3 erfolgt entweder authentifiziert oder anonym. AWS verwendet die Signature Version 4- oder Signature Version 2-Algorithmen zum Authentifizieren von Aufrufen des Services.

Alle neuen, nach Januar 2014 erstellten AWS-Regionen unterstützen nur Signature Version 4. Viele ältere Regionen unterstützen jedoch weiterhin Signature Version 4- und Signature Version 2-Anforderungen.

Wenn sich Ihr Bucket in einer der Regionen befindet, die Signature Version 2-Anforderungen nicht unterstützt, wie auf dieser Seite aufgeführt, müssen Sie die AWSConfigsS3.UseSignatureVersion4 Eigenschaft wie folgt auf „true“ setzen:

```
AWSConfigsS3.UseSignatureVersion4 = true;
```

Weitere Informationen zu AWS Signature-Versionen erhalten Sie unter Authenticating Requests (AWS Signature Version 4).

Integrieren von S3 in die Anwendung

Es gibt zwei Möglichkeiten für die Interaktion mit S3 in einer Xamarin-Anwendung. Diese beiden Methoden werden in den folgenden Themen ausführlich behandelt:

Verwenden von S3 Transfer Utility

S3 Transfer Utility erleichtert das Hoch- und Herunterladen von Dateien nach bzw. aus S3 mit der Xamarin-Anwendung.

Initialisieren des TransferUtility

Erstellen Sie einen S3-Client, übergeben Sie die AWS-Anmeldeinformationen und dann den S3-Client an das Dienstprogramm für die Datenübertragung, z. B.:

```
var s3Client = new AmazonS3Client(credentials,region);
var transferUtility = new TransferUtility(s3Client);
```

(optional) Konfigurieren der TransferUtility

Es gibt drei optionale Eigenschaften, die Sie konfigurieren können:

- ConcurrentServiceRequests – Bestimmt, wie viele aktive Threads oder die Anzahl der gleichzeitigen asynchronen Webanforderungen zum Hochladen/Herunterladen der Datei verwendet werden. Der Standardwert lautet 10.
- MinSizeBeforePartUpload – Ruft die minimale Teilegröße für das Hochladen von Teilen in Bytes ab oder legt sie fest. Der Standardwert ist 16 MB. Durch Absenken der minimalen Komponentengröße werden mehrteilige Hochladeoperationen in eine größere Zahl kleinerer Komponenten aufgeteilt. Wenn dieser Wert zu niedrig eingestellt und dadurch die Übertragungsgeschwindigkeit beeinträchtigt wird, verursacht dies zusätzliche Latenz und Netzwerkkommunikation für jede Komponente.
- NumberOfUploadThreads – Ruft die Anzahl der ausgeführten Threads ab oder legt sie fest. Diese Eigenschaft bestimmt, wie viele aktive Threads verwendet werden, um die Datei hochzuladen. Der Standardwert lautet 10 Threads.

Um den S3 TransferUtility zu konfigurieren, erstellen Sie ein Konfigurationsobjekt, legen Sie Ihre Eigenschaften fest und übergeben Sie das Objekt wie folgt an Ihren TransferUtility Konstruktor:

```
var config = new TransferUtilityConfig();

config.ConcurrentServiceRequests = 10;
config.MinSizeBeforePartUpload=16*1024*1024;
config.NumberOfUploadThreads=10;

var s3Client = new AmazonS3Client(credentials);
var utility = new TransferUtility(s3Client,config);
```

Herunterladen einer Datei

Um eine Datei aus S3 herunterzuladen, rufen Sie `Download` für das Transfer Utility-Objekt auf und übergeben dabei die folgenden Parameter:

- `file`: Name der herunterzuladenden Datei als String

- `bucketName`: Name des S3-Bucket, aus dem die Datei heruntergeladen werden soll
- `key`: String, der den Namen des herunterzuladenden S3-Objekts (in diesem Fall eine Datei) angibt

```
transferUtility.Download(
  Path.Combine(Environment.SpecialFolder.ApplicationData,"file"),
  "bucketName",
  "key"
);
```

Hochladen einer Datei

Um eine Datei nach S3 hochzuladen, rufen Sie `Upload` für das Transfer Utility-Objekt auf und übergeben dabei die folgenden Parameter:

- `file`: Name der hochzuladenden Datei als String
- `bucketName`: Name des S3-Bucket, in dem die Datei gespeichert werden soll, als String

```
transferUtility.Upload(
  Path.Combine(Environment.SpecialFolder.ApplicationData,"file"),
  "bucketName"
);
```

Im Code oben wird davon ausgegangen, dass sich eine Datei im Verzeichnis Environment.SpecialFolder.ApplicationData befindet. Beim Hochladen wird automatisch die S3-Funktion für mehrteiliges Hochladen großer Dateien verwendet, um den Durchsatz zu verbessern.

Verwenden des Service S3 APIs

Zusätzlich zur Verwendung des S3 können TransferUtilitySie auch mithilfe des S3 mit S3 APIsinteragieren.

Initialisieren des Amazon S3-Clients

Um Amazon S3 verwenden zu können, müssen wir zunächst eine AmazonS3Client Instance erstellen, die eine Referenz auf die zuvor erstellte CognitoAWSCredentials Instance und Ihre Region verwendet:

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

Herunterladen einer Datei

Laden Sie wie folgt eine Datei aus S3 herunter:

```
// Create a GetObject request
GetObjectRequest request = new GetObjectRequest
{
    BucketName = "SampleBucket",
    Key = "Item1"
};

// Issue request and remember to dispose of the response
using (GetObjectResponse response = client.GetObject(request))
{
    using (StreamReader reader = new StreamReader(response.ResponseStream))
    {
        string contents = reader.ReadToEnd();
        Console.WriteLine("Object - " + response.Key);
        Console.WriteLine(" Version Id - " + response.VersionId);
        Console.WriteLine(" Contents - " + contents);
    }
}
```

Hochladen einer Datei

Laden Sie wie folgt eine Datei nach S3 hoch:

```
// Create a client
AmazonS3Client client = new AmazonS3Client();

// Create a PutObject request
PutObjectRequest request = new PutObjectRequest
{
    BucketName = "SampleBucket",
    Key = "Item1",
    FilePath = "contents.txt"
};

// Put object
```

```
PutObjectResponse response = client.PutObject(request);
```

Löschen eines Elements

Löschen Sie wie folgt ein Element aus S3:

```
// Create a client
AmazonS3Client client = new AmazonS3Client();

// Create a DeleteObject request
DeleteObjectRequest request = new DeleteObjectRequest
{
    BucketName = "SampleBucket",
    Key = "Item1"
};

// Issue request
client.DeleteObject(request);
```

Löschen mehrerer Elemente

Löschen Sie mehrere Objekte wie folgt mit einer HTTP-Anforderung aus einem Bucket:

```
// Create a client
AmazonS3Client client = new AmazonS3Client();

// Create a DeleteObject request
DeleteObjectsRequest request = new DeleteObjectsRequest
{
    BucketName = "SampleBucket",
    Objects = new List<KeyVersion>
    {
        new KeyVersion() {Key = "Item1"},
        // Versioned item
        new KeyVersion() { Key = "Item2", VersionId =
 "Rej8CiBxcZKVK81cLr39j27Y5FVXghDK", },
        // Item in subdirectory
        new KeyVersion() { Key = "Logs/error.txt"}
    }
};

try
```

```
{
    // Issue request
    DeleteObjectsResponse response = client.DeleteObjects(request);
}
catch (DeleteObjectsException doe)
{
    // Catch error and list error details
    DeleteObjectsResponse errorResponse = doe.Response;

    foreach (DeletedObject deletedObject in errorResponse.DeletedObjects)
    {
        Console.WriteLine("Deleted item " + deletedObject.Key);
    }
    foreach (DeleteError deleteError in errorResponse.DeleteErrors)
    {
        Console.WriteLine("Error deleting item " + deleteError.Key);
        Console.WriteLine(" Code - " + deleteError.Code);
        Console.WriteLine(" Message - " + deleteError.Message);
    }
}
```

Sie können bis zu 1 000 Schlüssel angeben.

Auflisten von Buckets

Geben Sie wie folgt eine Liste aller Buckets zurück, deren Eigentümer der authentifizierte Absender der Anforderung ist:

```
// Create a client
AmazonS3Client client = new AmazonS3Client();

// Issue call
ListBucketsResponse response = client.ListBuckets();

// View response data
Console.WriteLine("Buckets owner - {0}", response.Owner.DisplayName);
foreach (S3Bucket bucket in response.Buckets)
{
    Console.WriteLine("Bucket {0}, Created on {1}", bucket.BucketName,
 bucket.CreationDate);
}
```

Auflisten von Objekten

Sie können einige oder alle (bis zu 1 000) der Objekte im S3-Bucket zurückgeben. Zu diesem Zweck benötigen Sie Lesezugriff auf den Bucket.

```
// Create a GetObject request
GetObjectRequest request = new GetObjectRequest
{
  BucketName = "SampleBucket",
  Key = "Item1"
};

// Issue request and remember to dispose of the response
using (GetObjectResponse response = client.GetObject(request))
{
  using (StreamReader reader = new StreamReader(response.ResponseStream))
  {
      string contents = reader.ReadToEnd();
      Console.WriteLine("Object - " + response.Key);
      Console.WriteLine(" Version Id - " + response.VersionId);
      Console.WriteLine(" Contents - " + contents);
  }
}
```

Ermitteln der Region eines Bucket

Ermitteln Sie die Region, in der ein Bucket sich befindet, wie folgt:

```
// Create a client
AmazonS3Client client = new AmazonS3Client();

// Construct request
GetBucketLocationRequest request = new GetBucketLocationRequest
{
    BucketName = "SampleBucket"
};

// Issue call
GetBucketLocationResponse response = client.GetBucketLocation(request);

// View response data
Console.WriteLine("Bucket location - {0}", response.Location);
```

Ermitteln der Richtlinie eines Bucket

So ermitteln Sie die Richtlinie eines Bucket:

```
// Create a client
AmazonS3Client client = new AmazonS3Client();

// Construct request
GetBucketPolicyRequest getRequest = new GetBucketPolicyRequest
{
   BucketName = "SampleBucket"
};
string policy = client.GetBucketPolicy(getRequest).Policy;

Console.WriteLine(policy);
Debug.Assert(policy.Contains("BasicPerms"));
```

Amazon – DynamoDB

Was ist Amazon DynamoDB?

Amazon DynamoDB ist ein schneller, hochgradig skalierbarer Service für nicht relationale Datenbanken. DynamoDBMit werden Einschränkungen der Datenspeicherungsskalierbarkeit eliminiert, die Latenz niedrig gehalten und die Leistung ist vorhersehbar.

Die wichtigsten Konzepte

Die DynamoDB Datenmodellkonzepte umfassen Tabellen, Elemente und Attribute.

Tables

In Amazon ist DynamoDBeine Datenbank eine Sammlung von Tabellen. Eine Tabelle ist eine Sammlung von Elementen und jedes Element wiederum eine Sammlung von Attributen.

In einer relationalen Datenbank hat eine Tabelle ein vordefiniertes Schema, z. B. Tabellenname, Primärschlüssel, Liste der Spaltennamen sowie Datentypen der Spalten. Alle Datensätze in der Tabelle müssen über dieselben Spalten verfügen. Im Gegensatz dazu erfordert DynamoDB nur, dass eine Tabelle über einen Primärschlüssel verfügt, aber nicht alle Attributnamen und Datentypen im Voraus definieren muss.

Weitere Informationen zum Arbeiten mit Tabellen finden Sie unter Arbeiten mit Tabellen in DynamoDB .

Elemente und Attribute

Einzelne Elemente in einer DynamoDB Tabelle können eine beliebige Anzahl von Attributen haben, obwohl die Elementgröße auf 400 KB beschränkt ist. Die Elementgröße wird gebildet aus der Länge der Attributnamen und -werte (Binär- und UTF-8-Längen).

Jedes Attribut in einem Element ist ein Name-Wert-Paar. Ein Attribut kann einen Wert oder mehrere Werten aufweisen. So kann ein Buchelement beispielsweise Attribute für Titel und Autor aufweisen. Jedes Buch hat einen Titel, kann jedoch mehrere Autoren haben. Das Attribut für mehrere Werte ist eine Gruppe, Duplikatwerte sind nicht zulässig.

Nehmen wir als Beispiel, einen Katalog von Produkten in zu DynamoDBspeichern. Sie können eine Tabelle erstellen, ProductCatalog, mit dem ID-Attribut als Primärschlüssel. Der Primärschlüssel

identifiziert jedes Element eindeutig, es können also nicht zwei Produkte in der Tabelle dieselbe ID aufweisen.

Weitere Informationen zum Arbeiten mit Elementen finden Sie unter Arbeiten mit Elementen im DynamoDB .

Datentypen

Amazon DynamoDB unterstützt die folgenden Datentypen:

- Skalare Typen: Number, String, Binary, Boolean und Null.
- Typen für mehrere Werte: String Set, Number Set und Binary Set.
- Dokumenttypen: List und Map.

Weitere Informationen zu skalaren Datentypen, Datentypen mit mehreren Werten und Dokumentdatentypen finden Sie unter DynamoDB -Datentypen.

Primärschlüssel

Wenn Sie eine Tabelle erstellen, müssen Sie außer dem Tabellennamen auch den Primärschlüssel der Tabelle angeben. Der Primärschlüssel identifiziert jedes Element in der Tabelle eindeutig, es gibt also nicht zwei Elemente mit identischem Schlüsselwert. DynamoDB unterstützt die folgenden zwei Typen von Primärschlüsseln:

- Hash-Schlüssel: Der Primärschlüssel besteht aus einem Attribut, einem Hash-Attribut. DynamoDB erstellt einen ungeordneten Hash-Index für dieses Primärschlüsselattribut. Jedes Element in der Tabelle wird eindeutig anhand seines Hash-Schlüsselwerts identifiziert.
- Hash- und Bereichsschlüssel: Der Primärschlüssel besteht aus zwei Attributen. Das erste Attribut ist das Hash-Attribut, das zweite das Bereichsattribut. DynamoDB erstellt einen ungeordneten Hash-Index auf dem Hash-Primärschlüsselattribut und einen sortierten Bereichsindex auf dem Bereichs-Primärschlüsselattribut. Jedes Element in der Tabelle wird durch die Kombination der Hash- und Bereichsschlüsselwerte eindeutig identifiziert. Zwei Elemente können denselben Hash-Schlüsselwert, müssen aber unterschiedliche Bereichsschlüsselwerte aufweisen.

Sekundäre Indizes

Wenn Sie eine Tabelle mit einem Hash- und Bereichsschlüssel erstellen, können Sie optional einzelne oder mehrere Sekundärindizes für die betreffende Tabelle definieren. Ein Sekundärindex

ermöglicht – ergänzend zu Abfragen über den Primärschlüssel – das Abfragen der Daten in der Tabelle über einen alternativen Schlüssel.

DynamoDB unterstützt zwei Arten von sekundären Indizes: lokale sekundäre Indizes und globale sekundäre Indizes.

- Lokaler Sekundärindex: Ein Index mit demselben Hash-Schlüssel wie die Tabelle, jedoch einem anderen Bereichsschlüssel.
- Globaler Sekundärindex: Ein Index mit einem Hash- und Bereichsschlüssel, der sich von den Schlüsseln der Tabelle unterscheiden kann.

Sie können bis zu 5 globale Sekundärindizes und 5 lokale Sekundärindizes pro Tabelle definieren. Weitere Informationen finden Sie unter Verbessern des Datenzugriffs mit sekundären Indizes in DynamoDB im -DynamoDBEntwicklerhandbuch.

Query und Scan

Zusätzlich zur Verwendung von Primärschlüsseln für den Zugriff auf Elemente bietet Amazon DynamoDB auch zwei APIs für die Datensuche: Query und Scan. Wir empfehlen, dass Sie Richtlinien für Abfragen und Scans im DynamoDB -Entwicklerhandbuch lesen, um sich mit einigen bewährten Methoden vertraut zu machen.

Query

Eine Query-Operation sucht Elemente ausschließlich unter Verwendung von Primärschlüssel-Attributwerten in einer Tabelle oder einem Sekundärindex. Sie müssen einen Hash-Schlüsselattributnamen und einen Wert angeben, der gesucht werden soll. Sie können optional den Namen und den Wert eines Bereichsschlüsselattributs angeben und einen Vergleichsoperator verwenden, um die Suchergebnisse zu verfeinern.

Beispielabfragen finden Sie unter:

- Verwenden des Document-Modells
- Verwenden des Object Persistence-Modells
- Verwenden des DynamoDB Service Levels APIs

Weitere Informationen zu Query finden Sie unter Query im -DynamoDBEntwicklerhandbuch.

Scan

Eine Scan-Operation liest jedes Element in einer Tabelle oder einem Sekundärindex. Standardmäßig gibt eine Scan-Operation für jedes Element in der Tabelle oder im Index alle Datenattribute zurück. Sie können den ProjectionExpression Parameter verwenden, sodass Scan nur einige der Attribute und nicht alle zurückgibt.

Beispielscans finden Sie unter:

- Verwenden des Document-Modells
- Verwenden des Object Persistence-Modells
- Verwenden des DynamoDB Service Levels APIs

Weitere Informationen zu Scan finden Sie unter Scan im -DynamoDBEntwicklerhandbuch.

Projekteinrichtung

Prerequisites

Um DynamoDB in Ihrer Anwendung zu verwenden, müssen Sie das SDK zu Ihrem Projekt hinzufügen. Befolgen Sie zu diesem Zweck die Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin.

Erstellen einer DynamoDB Tabelle

Gehen Sie zum Erstellen einer Tabelle zur -DynamoDBKonsole und führen Sie die folgenden Schritte aus:

1. Klicken Sie auf Create Table.
2. Geben Sie den Namen der Tabelle ein.
3. Wählen Sie Hash als Primärschlüsseltyp aus.
4. Wählen Sie einen Typ und geben Sie einen Wert für den Hash-Attributnamen ein. Klicken Sie auf Weiter.
5. Wenn Sie auf der Seite Add Indexes globale sekundäre Indizes verwenden möchten, legen Sie Index Type auf „Global Secondary Index“ fest und geben Sie unter Index Hash Key einen

Wert für den sekundären Index ein. So können Sie Primärindex und Sekundärindex abfragen und scannen. Klicken Sie auf Add Index To Table (Index zur Tabelle hinzufügen) und dann auf Continue (Weiter). Klicken Sie auf Continue (Weiter)., wenn Sie keine globalen Sekundärindizes verwenden wollen.

6. Stellen Sie für Lese- und Schreibkapazität die gewünschten Größen ein. Weitere Informationen zum Konfigurieren der Kapazität finden Sie unter Bereitgestellter Durchsatz in Amazon DynamoDB . Klicken Sie auf Continue (Weiter).
7. Geben Sie im nächsten Bildschirm eine Benachrichtigungs-E-Mail ein, um ggf. Durchsatzalarme zu erstellen. Klicken Sie auf Weiter.
8. Klicken Sie auf der Übersichtsseite auf Create (Erstellen). DynamoDB erstellt Ihre Datenbank.

Festlegen von -Berechtigungen für DynamoDB

Um DynamoDB in einer Anwendung zu verwenden, müssen Sie die richtigen Berechtigungen festlegen. Mit der folgenden IAM-Richtlinie kann der Benutzer Elemente in einer bestimmten DynamoDB Tabelle, die durch den ARN identifiziert wird, löschen, abrufen, ablegen, abfragen, scannen und aktualisieren:

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

Sie können Richtlinien in der IAM-Konsole. modifizieren. Sie sollten zulässige Aktionen basierend auf den Anforderungen der App hinzufügen oder entfernen.

Weitere Informationen zu IAM-Richtlinien erhalten Sie unter Using IAM.

Weitere Informationen zu DynamoDB-spezifischen Richtlinien finden Sie unter Using IAM to Control Access to DynamoDB Resources im -DynamoDBEntwicklerhandbuch.

Integrieren von DynamoDB in Ihre Anwendung

Das AWS Mobile SDK for .NET and Xamarin bietet eine allgemeine Bibliothek für die Arbeit mit DynamoDB. Sie können auch Anforderungen direkt an die DynamoDB Low-Level-API stellen, aber für die meisten Anwendungsfälle wird die High-Level-Bibliothek empfohlen. Der AmazonDynamoDBClient ist ein besonders nützlicher Teil der High-Level-Bibliothek. Mit dieser Klasse können Sie verschiedene CRUD-Operationen (Erstellen, Lesen, Aktualisieren, Löschen) sowie Abfragen ausführen.

Das AWS Mobile SDK for .NET and Xamarin ermöglicht es Ihnen, Aufrufe mit APIs über das AWS SDK for .NET zu tätigen, um mit zu DynamoDBarbeiten. Alle -APIsDateien sind in der Datei AWSSDK.dll verfügbar. Weitere Informationen zum Herunterladen von AWS SDK for .NET erhalten Sie unter AWS SDK for .NET.

Es gibt drei Möglichkeiten, mit DynamoDB in Ihrer Xamarin-Anwendung zu interagieren:

- Dokumentmodell: Diese API stellt Wrapper-Klassen für die DyanmoDB Low-Level-API bereit, um Ihre Programmieraufgaben weiter zu vereinfachen. Die wichtigsten Wrapper-Klassen sind Table und Document. Sie können das Dokumentmodell für Datenoperationen wie das Erstellen, Abrufen, Aktualisieren und Löschen von Elementen verwenden. Die API ist im Namespace Amazon.DynamoDB.DocumentModel verfügbar.
- Object Persistence-Modell: Mit der Object Persistence-API können Sie den DynamoDB -Tabellen clientseitige Klassen zuordnen. Die einzelnen Objekt-Instances werden anschließend einem Element in den entsprechenden Tabellen zugeordnet. Die DynamoDBContext -Klasse in dieser API bietet Methoden, mit denen Sie clientseitige Objekte in einer Tabelle speichern, Elemente als -Objekte abrufen und Abfragen und Scans durchführen können. Sie können das Object Persistence-Modell für Datenoperationen wie das Erstellen, Abrufen, Aktualisieren und Löschen von Elementen verwenden. Sie müssen zunächst die Tabellen mit der Service-Client-API erstellen und dann das Objektpersistenzmodell verwenden, um die Klassen den Tabellen zuzuordnen. Die API ist im Namespace Amazon.DynamoDB.DataModel verfügbar.
- Service-Client-API: Dies ist die API auf Protokollebene, die der DynamoDB API eng zugeordnet ist. Sie können diese Low-Level-API für alle Tabellen- und Elementoperationen wie das Erstellen, Aktualisieren und Löschen von Tabellen und Elementen verwenden. Sie können Tabellen außerdem abfragen und scannen. Diese API ist im Namespace Amazon.DynamoDB namespace verfügbar.

Diese drei Modelle werden in den folgenden Themen ausführlich behandelt:

Verwenden des Document-Modells

Das Dokumentmodell stellt Wrapper-Klassen für die .NET-Low-Level-API bereit. Die wichtigsten Wrapper-Klassen sind Table und Document. Sie können das Dokumentmodell verwenden, um Elemente zu erstellen, abzurufen, zu aktualisieren und zu löschen. Zum Erstellen, Aktualisieren und Löschen von Tabellen müssen Sie die Low-Level-API verwenden. Anweisungen zur Verwendung der Low-Level-API finden Sie unter Verwenden des DynamoDB Service APIsLevels . Die Low-Level-API ist im Namespace DynamoDB.DocumentModel verfügbar.

Weitere Informationen zum Dokumentmodell erhalten Sie unter .NET: Dokumentmodell.

Erstellen eines DynamoDB Clients

So erstellen Sie einen DynamoDB Client:

```
var client = new AmazonDynamoDBClient(credentials,region);
DynamoDBContext context = new DynamoDBContext(client);
```

CRUD-Operationen

Speichern eines Elements

Erstellen Sie ein Element wie folgt:

```
Table table = Table.LoadTable(client, "Books");
id = Guid.NewGuid().ToString();
var books = new Document();
books["Id"] = id;
books["Author"] = "Mark Twain";
books["Title"] = "Adventures of Huckleberry Finn";
books["ISBN"] = "112-111111";
books["Price"] = "10";
```

Speichern eines Elements in einer DynamoDB Tabelle:

```
var book = await table.PutItemAsync(books);
```

Abrufen eines Elements

Rufen Sie ein Element wie folgt ab:

```
public async Task GetItemAsync(AWSCredentials credentials, RegionEndpoint region)
{
      var client = new AmazonDynamoDBClient(credentials, region);
      Table books = Table.LoadTable(client, "Books");
      var book = await books.GetItemAsync(id);
}
```

Aktualisieren eines Elements

Aktualisieren Sie ein Element wie folgt:

```
public async Task UpdateItemAttributesAsync(AWSCredentials credentials, RegionEndpoint
 region)
{
      var book = new Document();
      book["Id"] = id;
      book["PageCount"] = "200";
      var client = new AmazonDynamoDBClient(credentials, region);
      Table books = Table.LoadTable(client, "Books");
      Document updatedBook = await books.UpdateItemAsync(book);
}
```

Aktualisieren Sie ein Element wie folgt bedingungsabhängig:

```
public async Task UpdateItemConditionallyAsync(AWSCredentials credentials,
 RegionEndpoint region) {
      var book = new Document();
      book["Id"] = id;
      book["Price"] = "30";

      // For conditional price update, creating a condition expression.
      Expression expr = new Expression();
      expr.ExpressionStatement = "Price = :val";
      expr.ExpressionAttributeValues[":val"] = 10.00;

      var client = new AmazonDynamoDBClient(credentials, region);
      Table books = Table.LoadTable(client, "Books");

      Document updatedBook = await books.UpdateItemAsync(book);
```

```
}
```

Löschen eines Elements

So löschen Sie einen Artikel:

```
public async Task DeleteItemAsync(AWSCredentials credentials, RegionEndpoint region)
{
  var client = new AmazonDynamoDBClient(credentials, region);
  Table books = Table.LoadTable(client, "Books");
  await books.DeleteItemAsync(id);
}
```

Query und Scan

Gehen Sie zum Abfragen und Abrufen aller Bücher, deren Autor „Mark Twain“ ist, wie folgt vor:

```
public async Task QueryAsync(AWSCredentials credentials, RegionEndpoint region) {
  var client = new AmazonDynamoDBClient(credentials, region);
  Table books = Table.LoadTable(client, "Books");
  var search = books.Query(new QueryOperationConfig() {
   IndexName = "Author-Title-index",
   Filter = new QueryFilter("Author", QueryOperator.Equal, "Mark Twain")
  });
  Console.WriteLine("ScanAsync: printing query response");
  var documents = await search.GetRemainingAsync();
  documents.ForEach((d) = > {
   PrintDocument(d);
  });
}
```

Der Scan-Beispiel-Code unten gibt alle Bücher in der Tabelle zurück:

```
public async Task ScanAsync(AWSCredentials credentials, RegionEndpoint region) {
      var client = new AmazonDynamoDBClient(credentials, region);
      Table books = Table.LoadTable(client, "Books");
      var search = books.Scan(new ScanOperationConfig() {
              ConsistentRead = true
      });
      Console.WriteLine("ScanAsync: printing scan response");
      var documents = await search.GetRemainingAsync();
      documents.ForEach((d) = > {
```

```
                PrintDocument(d);
        });
}
```

Verwenden des Object Persistence-Modells

Das AWS Mobile SDK for .NET and Xamarin bietet ein Object Persistence-Modell, mit dem Sie Ihre clientseitigen Klassen einer DynamoDB -Tabelle zuordnen können. Jede Objekt-Instance wird dann einem Element in der entsprechenden Tabelle zugeordnet. Um Ihre clientseitigen Objekte in einer Tabelle zu speichern, stellt das Object Persistence-Modell die DynamoDBContext Klasse bereit, einen Eintrittspunkt für DynamoDB. Diese Klasse bietet Ihnen eine Verbindung zu DynamoDB und ermöglicht Ihnen den Zugriff auf Tabellen, die Ausführung verschiedener CRUD-Operationen und die Ausführung von Abfragen.

Das Object Persistence-Modell stellt keine API zum Erstellen, Aktualisieren und Löschen von Tabellen bereit. Es stellt ausschließlich Datenoperationen bereit. Zum Erstellen, Aktualisieren und Löschen von Tabellen müssen Sie die Low-Level-API verwenden. Anweisungen zur Verwendung der Low-Level-API finden Sie unter Verwenden des DynamoDB Service APIsLevels .

Overview

Das Object Persistence-Modell stellt eine Reihe von Attributen bereit, mit deren Hilfe Client-seitige Klassen Tabellen und Eigenschaften/Felder Tabellenattributen zugeordnet werden können. Das Object Persistence-Modell unterstützt die explizite und die Standardzuordnung zwischen Klasseneigenschaften und Tabellenattributen.

- Explizite Zuweisung: Um eine Eigenschaft einem Primärschlüssel zuzuweisen, müssen Sie die Attribute DynamoDBHashKey und DynamoDBRangeKey Object Persistence-Modell verwenden. Außerdem gilt für Nicht-Primärschlüsselattribute, dass Sie die Zuweisung definieren müssen, indem Sie explizit das Attribut DynamoDBProperty hinzufügen, wenn der Name der Eigenschaft in Ihrer Klasse und das entsprechende Tabellenattribut, dem Sie ihn zuordnen möchten, nicht identisch sind.
- Standardzuordnung: Standardmäßig ordnet das Object Persistence-Modell die Klasseneigenschaften Attributen mit identischem Namen in der Tabelle zu.

Sie müssen nicht jede Klasseneigenschaft zuordnen. Sie erkennen diese Eigenschaften, indem Sie das DynamoDBIgnore-Attribut hinzufügen. Beim Speichern und Abrufen einer Instance eines Objekts werden mit diesem Attribut markierte Eigenschaften ausgelassen.

Unterstützte Datentypen

Das Object Persistence-Modell unterstützt eine Reihe primitiver .NET-Datentypen, Sammlungen sowie beliebige Datentypen. Das Modell unterstützt die folgenden primitiven Datentypen.

- bool
- Byte
- char
- DateTime
- decimal, double, float
- Int16, Int32, Int64
- SByte
- string
- UInt16, UInt32, UInt64

Das Object Persistence-Modell unterstützt außerdem die .NET-Sammlungstypen. Dabei gelten folgende Einschränkungen:

- Der Sammlungstyp muss eine ICollection Schnittstelle implementieren.
- Der Sammlungstyp muss aus unterstützten primitiven Datentypen gebildet werden. Beispiel: ICollection <string>, ICollection <bool>.
- Der Sammlungstyp muss einen parameterlosen Konstruktor bereitstellen.

Weitere Informationen zum Object Persistence-Modell erhalten Sie unter .NET Object Persistence-Modell.

Erstellen eines DynamoDB Clients

So erstellen Sie einen DynamoDB Client:

```
var client = new AmazonDynamoDBClient(credentials,region);
DynamoDBContext context = new DynamoDBContext(client);
```

CRUD-Operationen

Speichern eines Objekts

Erstellen eines Objekts:

```
[DynamoDBTable("Books")]
public class Book {
  [DynamoDBHashKey] // Hash key.
  public string Id {
    get;
    set;
  }

  [DynamoDBGlobalSecondaryIndexHashKey]
  public string Author {
    get;
    set;
  }

  [DynamoDBGlobalSecondaryIndexRangeKey]
  public string Title {
    get;
    set;
  }
  public string ISBN {
    get;
    set;
  }
  public int Price {
    get;
    set;
  }
  public string PageCount {
    get;
    set;
  }
}

Book myBook = new Book
{
    Id = id,
    Author = "Charles Dickens",
```

```
    Title = "Oliver Twist",
    ISBN = "111-1111111001",
    Price = 10,
    PageCount = 300
};
```

Speichern eines Objekts in einer DynamoDB Tabelle:

```
context.Save(myBook);
```

Abrufen eines Objekts

Rufen Sie ein Objekt wie folgt ab:

```
Book retrievedBook = context.Load<Book>(1);
```

Aktualisieren eines Objekts

Aktualisieren Sie ein Objekt wie folgt:

```
Book retrievedBook = context.Load<Book>(1);
retrievedBook.ISBN = "111-1111111001";
context.Save(retrievedBook);
```

Objekte löschen

So löschen Sie ein Objekt:

```
Book retrievedBook = context.Load<Book>(1);
context.Delete(retrievedBook);
```

Query und Scan

Gehen Sie zum Abfragen und Abrufen aller Bücher, deren Autor „Charles Dickens“ ist, wie folgt vor:

```
public async Task QueryAsync(AWSCredentials credentials, RegionEndpoint region) {
  var client = new AmazonDynamoDBClient(credentials, region);
  DynamoDBContext context = new DynamoDBContext(client);

  var search = context.FromQueryAsync < Book > (new
 Amazon.DynamoDBv2.DocumentModel.QueryOperationConfig() {
```

```
    IndexName = "Author-Title-index",
    Filter = new Amazon.DynamoDBv2.DocumentModel.QueryFilter("Author",
 Amazon.DynamoDBv2.DocumentModel.QueryOperator.Equal, "Charles Dickens")
  });

  Console.WriteLine("items retrieved");

  var searchResponse = await search.GetRemainingAsync();
  searchResponse.ForEach((s) = > {
    Console.WriteLine(s.ToString());
  });
}
```

Der Scan-Beispiel-Code unten gibt alle Bücher in der Tabelle zurück:

```
public async Task ScanAsync(AWSCredentials credentials, RegionEndpoint region) {
  var client = new AmazonDynamoDBClient(credentials, region);
  DynamoDBContext context = new DynamoDBContext(client);

  var search = context.FromScanAsync < Book > (new
 Amazon.DynamoDBv2.DocumentModel.ScanOperationConfig() {
   ConsistentRead = true
  });

  Console.WriteLine("items retrieved");

  var searchResponse = await search.GetRemainingAsync();
  searchResponse.ForEach((s) = > {
   Console.WriteLine(s.ToString());
  });
}
```

Verwenden des DynamoDB Service-Levels APIs

Mit dem Dynamo Service Level APIs können Sie Tabellen erstellen, aktualisieren und löschen. Sie können zudem typische CRUD-Operationen (Erstellen, Lesen, Aktualisieren, Löschen) für Elemente in einer Tabelle ausführen, die diese API verwendet.

Erstellen eines DynamoDB Clients

So erstellen Sie einen DynamoDB Client:

```
AmazonDynamoDBClient client = new AmazonDynamoDBClient(credentials,region);
```

CRUD-Operationen

Speichern eines Elements

So speichern Sie ein Element in einer DynamoDB Tabelle:

```
// Create a client
AmazonDynamoDBClient client = new AmazonDynamoDBClient(credentials,region);

// Define item attributes
Dictionary<string, AttributeValue> attributes = new Dictionary<string,
 AttributeValue>();

// Author is hash-key
attributes["Author"] = new AttributeValue { S = "Mark Twain" };
attributes["Title"] = new AttributeValue { S = "The Adventures of Tom Sawyer" };
attributes["PageCount"] = new AttributeValue { N = "275" };
attributes["Price"] = new AttributeValue{N = "10.00"};
attributes["Id"] = new AttributeValue{N="10"};
attributes["ISBN"] = new AttributeValue{S="111-1111111"};

// Create PutItem request
PutItemRequest request = new PutItemRequest
{
    TableName = "Books",
    Item = attributes
};

// Issue PutItem request
var response = await client.PutItemAsync(request);
```

Abrufen eines Elements

Rufen Sie ein Element wie folgt ab:

```
// Create a client
AmazonDynamoDBClient client = new AmazonDynamoDBClient(credentials,region);

Dictionary<string, AttributeValue> key = new Dictionary<string, AttributeValue>
```

```
{
    { "Id", new AttributeValue { N = "10" } }
};

// Create GetItem request
GetItemRequest request = new GetItemRequest
{
    TableName = "Books",
    Key = key,
};

// Issue request
var result = await client.GetItemAsync(request);

// View response
Console.WriteLine("Item:");
Dictionary<string, AttributeValue> item = result.Item;
foreach (var keyValuePair in item)
{
    Console.WriteLine("Author := {0}", item["Author"]);
    Console.WriteLine("Title := {0}", item["Title"]);
    Console.WriteLine("Price:= {0}", item["Price"]);
    Console.WriteLine("PageCount := {0}", item["PageCount"]);
}
```

Aktualisieren eines Elements

Aktualisieren Sie ein Element wie folgt:

```
// Create a client
AmazonDynamoDBClient client = new AmazonDynamoDBClient(credentials,region);

Dictionary<string, AttributeValue> key = new Dictionary<string, AttributeValue>
{
    { "Id", new AttributeValue { N = "10" } }
};

// Define attribute updates
Dictionary<string, AttributeValueUpdate> updates = new Dictionary<string,
 AttributeValueUpdate>();
// Add a new string to the item's Genres SS attribute
updates["Genres"] = new AttributeValueUpdate()
{
```

```
    Action = AttributeAction.ADD,
    Value = new AttributeValue { SS = new List<string> { "Bildungsroman" } }
};

// Create UpdateItem request
UpdateItemRequest request = new UpdateItemRequest
{
    TableName = "Books",
    Key = key,
    AttributeUpdates = updates
};

// Issue request
var response = await client.UpdateItemAsync(request);
```

Löschen eines Elements

So löschen Sie einen Artikel:

```
// Create a client
AmazonDynamoDBClient client = new AmazonDynamoDBClient(credentials,region);

Dictionary<string, AttributeValue> key = new Dictionary<string, AttributeValue>
{
  { "Id", new AttributeValue { N = "10" } }
};

// Create DeleteItem request
DeleteItemRequest request = new DeleteItemRequest
{
  TableName = "Books",
  Key = key
};

// Issue request
var response = await client.DeleteItemAsync(request);
```

Query und Scan

Gehen Sie zum Abfragen und Abrufen aller Bücher, deren Autor „Mark Twain“ ist, wie folgt vor:

```
public void Query(AWSCredentials credentials, RegionEndpoint region) {
```

```
  using(var client = new AmazonDynamoDBClient(credentials, region)) {
    var queryResponse = await client.QueryAsync(new QueryRequest() {
      TableName = "Books",
      IndexName = "Author-Title-index",
      KeyConditionExpression = "Author = :v_Id",
      ExpressionAttributeValues = new Dictionary < string, AttributeValue > {
        {
          ":v_Id", new AttributeValue {
            S = "Mark Twain"
          }
        }
      }
    });
    queryResponse.Items.ForEach((i) = > {
      Console.WriteLine(i["Title"].S);
    });

  }
}
```

Der Scan-Beispiel-Code unten gibt alle Bücher in der Tabelle zurück:

```
public void Scan(AWSCredentials credentials, RegionEndpoint region) {
      using(var client = new AmazonDynamoDBClient(credentials, region)) {
              var queryResponse = client.Scan(new ScanRequest() {
                      TableName = "Books"
              });
              queryResponse.Items.ForEach((i) = > {
                      Console.WriteLine(i["Title"].S);
              });
      }
}
```

Amazon Simple Notification Service (SNS)

Mit SNS und AWS Mobile SDK for .NET and Xamarin können Sie Anwendungen schreiben, die mobile Push-Benachrichtigungen empfangen können. Weitere Informationen zu SNS erhalten Sie unter Amazon Simple Notification Service.

Die wichtigsten Konzepte

Amazon SNS ermöglicht Anwendungen und Endbenutzern mit unterschiedlichen Geräten den Empfang von Benachrichtigungen via Mobile Push (Apple, Google und Kindle Fire), HTTP/HTTPS, E-Mail/E-Mail-JSON, SMS oder Amazon Simple Queue Service (SQS)-Warteschlangen oder AWS Lambda-Funktionen. Mit SNS können Sie individuelle Nachrichten oder Rundsendungen an eine große Zahl von Empfängern senden, die ein einzelnes Thema abonniert haben.

Topics

Ein Thema ist ein „Zugriffspunkt“, der es Empfängern erlaubt, identische Kopien derselben Benachrichtigung zu abonnieren. Ein Thema kann Übermittlungen an mehrere Endpunkttypen unterstützen. So können Sie beispielsweise iOS-, Android- und SMS-Empfänger gruppieren.

Subscriptions

Zum Abrufen der für ein Thema veröffentlichten Nachrichten müssen Sie einen Endpunkt für das Thema abonnieren. Ein Endpunkt ist eine mobile App, ein Web-Server, eine E-Mail-Adresse oder eine Amazon SQS-Warteschlange, die Benachrichtigungen von Amazon SNS empfangen kann. Sobald Sie einen Endpunkt für ein Thema abonnieren und das Abonnement bestätigt wurde, wird der Endpunkt alle Nachrichten empfangen, die zu diesem Thema veröffentlicht werden.

Publishing

Wenn Sie für ein Thema veröffentlichen, übermittelt SNS Kopien der Nachricht mit geeigneter Formatierung an jeden Abonnenten des Themas. Für Mobile Push-Benachrichtigungen können Sie direkt für den Endpunkt veröffentlichen oder den Endpunkt für ein Thema abonnieren.

Projekteinrichtung

Prerequisites

Um SNS in der Anwendung verwenden zu können, müssen Sie das SDK in das Projekt aufnehmen. Befolgen Sie zu diesem Zweck die Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin.

Festlegen von Berechtigungen für SNS

Weitere Informationen zum Festlegen von Berechtigungen für SNS erhalten Sie unter Übersicht über die Verwaltung von Zugriffsberechtigungen für Ihre Amazon SNS-Themen.

Hinzufügen eines NuGet Pakets für SNS zu Ihrem Projekt

Befolgen Sie Schritt 4 der Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, um das Amazon Simple Notification NuGet Service-Paket Ihrem Projekt hinzuzufügen.

Integrieren von SNS mit der Anwendung

Es gibt mehrere Möglichkeiten für die Interaktion mit SNS in einer Xamarin-Anwendung:

Senden von Push-Benachrichtigungen (Xamarin Android)

In diesem Dokument wird erläutert, wie Push-Benachrichtigungen mit Amazon Simple Notification Service (SNS) und an eine Xamarin-Android-Anwendung gesendet werden.AWS Mobile SDK for .NET and Xamarin.

Projekteinrichtung

Prerequisites

Befolgen Sie alle Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, bevor Sie mit diesem Tutorial beginnen.

Festlegen von Berechtigungen für SNS

Befolgen Sie Schritt 2 in Einrichten von AWS Mobile SDK for .NET and Xamarin, um die unten angegebene Richtlinie an die Rollen Ihrer Anwendung anzufügen. Die Anwendung erhält so die erforderlichen Berechtigungen für den Zugriff auf SNS:

1. Navigieren Sie zur IAM-Konsole und wählen Sie die IAM-Rolle aus, die Sie konfigurieren möchten.
2. Klicken Sie auf Attach Policy, wählen Sie die AmazonSNSFullAccess Richtlinie aus und klicken Sie auf Attach Policy.

> **Warning**
>
> Die Verwendung von AmazonSNSFullAccess wird in einer Produktionsumgebung nicht empfohlen. Wir verwenden die Richtlinie hier, damit Sie schnell mit der Arbeit beginnen können. Weitere Informationen über das Festlegen von Berechtigungen für eine IAM-Rolle erhalten Sie unter Overview of IAM Role Permissions.

Aktivieren von Push-Benachrichtigungen in der Google Cloud

Fügen Sie zunächst ein neues Google-API-Projekt hinzu:

1. Navigieren Sie zur Google Developers-Konsole.
2. Klicken Sie auf Create Project (Projekt erstellen).
3. Geben Sie einen Projektnamen in das Feld New Project (Neues Projekt) ein. Notieren Sie sich die (später benötigte) Projekt-ID und klicken Sie auf Create (Erstellen).

Aktivieren Sie nun den Service Google Cloud Messaging (GCM) für das Projekt:

1. In der Google Developers-Konsole sollte Ihr neues Projekt bereits ausgewählt sein. Wählen Sie es andernfalls im Dropdown-Menü oben auf der Seite aus.
2. Wählen Sie APIs in der Seitenleiste auf der linken Seite der Seite & auth aus.
3. Geben Sie in das Suchfeld „Google Cloud Messaging for Android“ ein und klicken Sie auf den Link Google Cloud Messaging for Android.
4. Klicken Sie auf Enable API.

Rufen Sie schließlich einen API-Schlüssel ab:

1. Wählen Sie in der Google Developers-Konsole APIs & auth > Credentials (Anmeldeinformationen) aus.
2. Klicken Sie unter Public API access auf Create new key.

3. Klicken Sie im Dialogfeld Create a new key auf Server key.
4. Klicken Sie im daraufhin angezeigten Dialogfeld auf Create (Erstellen) und kopieren Sie den API-Schlüssel. Sie werden diesen API-Schlüssel später zur Authentifizierung verwenden.

Verwenden der Projekt-ID zum Erstellen eines Plattform-ARN in der SNS-Konsole

1. Rufen Sie die SNS-Konsole. auf.
2. Klicken Sie auf der linken Seite des Bildschirms auf Applications (Anwendungen).
3. Klicken Sie auf Create platform application (Plattformanwendung erstellen), um eine neue SNS-Plattformanwendung zu erstellen.
4. Geben Sie einen Application Name. ein.
5. Wählen Sie Google Cloud Messaging (GCM) für Push notification platform (Push-Benachrichtigungsplattform).
6. Fügen Sie den API-Schlüssel in das Textfeld API key. ein.
7. Klicken Sie auf Create platform application.
8. Wählen Sie die soeben erstellte Plattformanwendung und kopieren Sie den ARN der Anwendung.

Hinzufügen eines NuGet Pakets für SNS zu Ihrem Projekt

Befolgen Sie Schritt 4 der Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, um das Amazon Simple Notification NuGet Service-Paket Ihrem Projekt hinzuzufügen.

Erstellen eines SNS-Clients

```
var snsClient = new AmazonSimpleNotificationServiceClient(credentials, region);
```

Registrieren der Anwendung für Remote-Benachrichtigungen

Um sich für Remote-Benachrichtigungen auf Android zu registrieren, müssen Sie einen BroadcastReceiver erstellen, der Google Cloud-Nachrichten empfangen kann. Ändern Sie den Paketnamen unten, wenn Sie dazu aufgefordert werden:

```
[BroadcastReceiver(Permission = "com.google.android.c2dm.permission.SEND")]
[IntentFilter(new string[] {
      "com.google.android.c2dm.intent.RECEIVE"
```

```
}, Categories = new string[] {
      "com.amazonaws.sns" /* change to match your package */
})]
[IntentFilter(new string[] {
      "com.google.android.c2dm.intent.REGISTRATION"
}, Categories = new string[] {
      "com.amazonaws.sns" /* change to match your package */
})]
[IntentFilter(new string[] {
      "com.google.android.gcm.intent.RETRY"
}, Categories = new string[] {
      "com.amazonaws.sns" /* change to match your package */
})]
public class GCMBroadcastReceiver: BroadcastReceiver {
      const string TAG = "PushHandlerBroadcastReceiver";
      public override void OnReceive(Context context, Intent intent) {
              GCMIntentService.RunIntentInService(context, intent);
              SetResult(Result.Ok, null, null);
      }
}

[BroadcastReceiver]
[IntentFilter(new[] {
      Android.Content.Intent.ActionBootCompleted
})]
public class GCMBootReceiver: BroadcastReceiver {
      public override void OnReceive(Context context, Intent intent) {
              GCMIntentService.RunIntentInService(context, intent);
              SetResult(Result.Ok, null, null);
      }
}
```

Nachfolgend finden Sie den Service, der die Push-Benachrichtigung von der empfängt BroadcastReceiver und die Benachrichtigung in der Benachrichtigungsleiste des Geräts anzeigt:

```
[Service]
 public class GCMIntentService: IntentService {
  static PowerManager.WakeLock sWakeLock;
  static object LOCK = new object();

  public static void RunIntentInService(Context context, Intent intent) {
    lock(LOCK) {
      if (sWakeLock == null) {
```

```
        // This is called from BroadcastReceiver, there is no init.
        var pm = PowerManager.FromContext(context);
        sWakeLock = pm.NewWakeLock(
        WakeLockFlags.Partial, "My WakeLock Tag");
      }
    }

    sWakeLock.Acquire();
    intent.SetClass(context, typeof(GCMIntentService));
    context.StartService(intent);
  }

  protected override void OnHandleIntent(Intent intent) {
    try {
      Context context = this.ApplicationContext;
      string action = intent.Action;

      if (action.Equals("com.google.android.c2dm.intent.REGISTRATION")) {
        HandleRegistration(intent);
      } else if (action.Equals("com.google.android.c2dm.intent.RECEIVE")) {
        HandleMessage(intent);
      }
    } finally {
      lock(LOCK) {
        //Sanity check for null as this is a public method
        if (sWakeLock != null) sWakeLock.Release();
      }
    }
  }

  private void HandleRegistration(Intent intent) {
    string registrationId = intent.GetStringExtra("registration_id");
    string error = intent.GetStringExtra("error");
    string unregistration = intent.GetStringExtra("unregistered");

    if (string.IsNullOrEmpty(error)) {
      var response = await SnsClient.CreatePlatformEndpointAsync(new
CreatePlatformEndpointRequest {
        Token = registrationId,
        PlatformApplicationArn = "YourPlatformArn" /* insert your platform application
ARN here */
      });
    }
  }
```

```
  private void HandleMessage(Intent intent) {
    string message = string.Empty;
    Bundle extras = intent.Extras;
    if (!string.IsNullOrEmpty(extras.GetString("message"))) {
      message = extras.GetString("message");
    } else {
      message = extras.GetString("default");
    }

    Log.Info("Messages", "message received = " + message);
    ShowNotification(this, "SNS Push", message);
    //show the message

  }

  public void ShowNotification(string contentTitle,
  string contentText) {
    // Intent
    Notification.Builder builder = new Notification.Builder(this)
      .SetContentTitle(contentTitle)
      .SetContentText(contentText)
      .SetDefaults(NotificationDefaults.Sound | NotificationDefaults.Vibrate)
      .SetSmallIcon(Resource.Drawable.Icon)
      .SetSound(RingtoneManager.GetDefaultUri(RingtoneType.Notification));

    // Get the notification manager:
    NotificationManager notificationManager =
 this.GetSystemService(Context.NotificationService) as NotificationManager;

    notificationManager.Notify(1001, builder.Build());
  }
}
```

Senden einer Nachricht mit der SNS-Konsole an den Endpunkt

1. Navigieren Sie zu SNS Console (SNS-Konsole) > Applications (Anwendungen).
2. Wählen Sie die Plattformanwendung, dann einen Endpunkt und klicken Sie schließlich auf Publish to endpoint (In Endpunkt veröffentlichen).
3. Geben Sie eine Textnachricht in das Textfeld ein und klicken Sie auf Publish message (Nachricht veröffentlichen), um eine Nachricht zu veröffentlichen.

Senden von Push-Benachrichtigungen (Xamarin iOS)

In diesem Dokument wird erläutert, wie Push-Benachrichtigungen mit Amazon Simple Notification Service (SNS) und an eine Xamarin-iOS-Anwendung gesendet werden.AWS Mobile SDK for .NET and Xamarin.

Projekteinrichtung

Prerequisites

Befolgen Sie alle Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, bevor Sie mit diesem Tutorial beginnen.

Festlegen von Berechtigungen für SNS

Befolgen Sie Schritt 2 in Einrichten von AWS Mobile SDK for .NET and Xamarin, um die unten angegebene Richtlinie an die Rollen Ihrer Anwendung anzufügen. Die Anwendung erhält so die erforderlichen Berechtigungen für den Zugriff auf SNS:

1. Navigieren Sie zur IAM-Konsole und wählen Sie die IAM-Rolle aus, die Sie konfigurieren möchten.
2. Klicken Sie auf Attach Policy, wählen Sie die AmazonSNSFullAccess Richtlinie aus und klicken Sie auf Attach Policy.

> **Warning**
>
> Die Verwendung von AmazonSNSFullAccess wird in einer Produktionsumgebung nicht empfohlen. Wir verwenden die Richtlinie hier, damit Sie schnell mit der Arbeit beginnen können. Weitere Informationen über das Festlegen von Berechtigungen für eine IAM-Rolle erhalten Sie unter Overview of IAM Role Permissions.

Anfordern der Mitgliedschaft im Apple iOS Developer Program

Sie müssen die App auf einem physischen Gerät ausführen, um Push-Benachrichtigungen zu empfangen. Zum Ausführen der App auf einem Gerät benötigen Sie eine Mitgliedschaft im Apple iOS Developer Program. Sobald Sie eine Mitgliedschaft besitzen, können Sie mit Xcode eine Signaturidentität generieren. Weitere Informationen finden Sie in der Dokumentation App Distribution Quick Start von Apple.

Erstellen eines iOS-Zertifikats

Zunächst müssen Sie ein iOS-Zertifikat erstellen. Anschließend müssen Sie ein Bereitstellungsprofil erstellen, das für Push-Benachrichtigungen konfiguriert ist. Hierzu gehen Sie wie folgt vor:

1. Navigieren Sie zum Apple Developer Member Center und klicken Sie auf Certificates, Identifiers & Profiles (Zertifikate, IDs und Profile).
2. Klicken Sie unter iOS Apps auf Identifiers, klicken Sie auf das Plus-Symbol in der oberen rechten Ecke der Webseite, um eine neue iOS-App-ID hinzuzufügen, und geben Sie eine App-ID-Beschreibung ein.
3. Scrollen Sie nach unten zum Abschnitt Add ID Suffix (ID-Suffix hinzufügen), wählen Sie Explicit App ID (Explizite App-ID) aus und geben Sie Ihre Bundle-ID ein.
4. Scrollen Sie nach unten zum Bereich App Services und wählen Sie Push Notifications.
5. Klicken Sie auf Weiter.
6. Klicken Sie auf Submit.
7. Klicken Sie auf Done (Fertig).
8. Wählen Sie die gerade erstellte App-ID aus und klicken Sie dann auf Edit (Bearbeiten).
9. Scrollen Sie nach unten zum Abschnitt Push Notifications (Push-Benachrichtigungen). Klicken Sie auf Create Certificate (Zertifkat erstellen) unter Development SSL Certificate (Entwicklung-SSL-Zertifikat).
10. Befolgen Sie die Anweisungen zum Erstellen einer CSR-Anforderung (Certificate Signing Request), laden Sie die Anforderung hoch und laden Sie ein SSL-Zertifikat herunter, das für die Kommunikation mit Apple Notification Service (APNS) verwendet wird.
11. Kehren Sie zur Seite Certificates, Identifiers & Profiles (Zertifikate, IDs und Profile) zurück. Klicken Sie auf All unter Provisioning Profiles.
12. Klicken Sie auf die Plus-Schaltfläche oben rechts, um ein neues Bereitstellungsprofil hinzuzufügen.
13. Wählen Sie iOS App Development (Entwicklung von iOS-Anwendungen) und klicken Sie dann auf Continue (Weiter).
14. Wählen Sie die App-ID und klicken Sie dann auf Continue (Weiter).
15. Wählen Sie Ihr Entwicklerzertifikat und klicken Sie dann auf Continue (Weiter).
16. Wählen Sie das Gerät und klicken Sie dann auf Continue (Weiter).
17. Geben Sie einen Profilnamen ein und klicken Sie dann auf Generate (Erstellen).

18. Laden Sie die Bereitstellungsdatei herunter und doppelklicken Sie dann auf die Datei, um das Bereitstellungsprofil zu installieren.

Weitere Informationen zur Bereitstellung eines Profils, das für Push-Benachrichtigungen konfiguriert ist, finden Sie in der Apple-Dokumentation Configuring Push Notifications.

Verwenden eines Zertifikats zum Erstellen eines Plattform-ARN in der SNS-Konsole

1. Führen Sie die KeyChain Zugriffs-App aus, wählen Sie unten links auf dem Bildschirm My Certificates (Meine Zertifikate) aus und klicken Sie dann mit der rechten Maustaste auf das SSL-Zertifikat, das Sie generiert haben, um eine Verbindung mit APNS herzustellen. Wählen Sie dann Export (Exportieren) aus. Sie werden aufgefordert, einen Namen für die Datei und ein Passwort zum Schutz des Zertifikats anzugeben. Das Zertifikat wird in einer P12-Datei gespeichert.
2. Navigieren Sie zur SNS-Konsole und klicken Sie auf der linken Seite des Bildschirms auf Applications (Anwendungen).
3. Klicken Sie auf Create platform application (Plattformanwendung erstellen), um eine neue SNS-Plattformanwendung zu erstellen.
4. Geben Sie einen Application Name. ein.
5. Wählen Sie Apple Development für Push notification platform (Push-Benachrichtigungsplattform).
6. Klicken Sie auf Choose File (Datei auswählen) und wählen Sie die P12 aus, die Sie beim Exportieren des SSL-Zertifikats erstellt haben.
7. Geben Sie das beim Exportieren des SSL-Zertifikats eingegebene Passwort ein und klicken Sie auf Load Credentials From File.
8. Klicken Sie auf Create platform application.
9. Wählen Sie die soeben erstellte Plattformanwendung und kopieren Sie den ARN der Anwendung. Sie werden den Namen in kommenden Schritten benötigen.

Hinzufügen eines NuGet Pakets für SNS zu Ihrem Projekt

Befolgen Sie Schritt 4 der Anweisungen unter Einrichten von AWS Mobile SDK for .NET and Xamarin, um das Amazon Simple Notification NuGet Service-Paket Ihrem Projekt hinzuzufügen.

Erstellen eines SNS-Clients

```
var snsClient = new AmazonSimpleNotificationServiceClient(credentials, region);
```

Registrieren der Anwendung für Remote-Benachrichtigungen

Um eine Anwendung zu registrieren, rufen Sie RegisterForRemoteNotifications für Ihr UIApplication -Objekt auf, wie unten gezeigt. Fügen Sie den folgenden Code in das AppDelegate.cs, Einfügen Ihres Plattformanwendungs-ARN ein, wenn Sie unten dazu aufgefordert werden:

```
public override bool FinishedLaunching(UIApplication app, NSDictionary options) {
// do something
var pushSettings = UIUserNotificationSettings.GetSettingsForTypes (
  UIUserNotificationType.Alert |
  UIUserNotificationType.Badge |
  UIUserNotificationType.Sound,
  null
);
app.RegisterUserNotifications(pushSettings);
app.RegisterForRemoteNotifications();
// do something
  return true;
}

public override void RegisteredForRemoteNotifications(UIApplication application, NSData
 token) {
  var deviceToken = token.Description.Replace("<", "").Replace(">", "").Replace(" ",
 "");
  if (!string.IsNullOrEmpty(deviceToken)) {
    //register with SNS to create an endpoint ARN
    var response = await SnsClient.CreatePlatformEndpointAsync(
    new CreatePlatformEndpointRequest {
      Token = deviceToken,
      PlatformApplicationArn = "YourPlatformArn" /* insert your platform application
 ARN here */
    });
  }
}
```

Senden einer Nachricht mit der SNS-Konsole an den Endpunkt

1. Navigieren Sie zu SNS Console (SNS-Konsole) > Applications (Anwendungen).
2. Wählen Sie die Plattformanwendung, dann einen Endpunkt und klicken Sie schließlich auf Publish to endpoint (In Endpunkt veröffentlichen).

3. Geben Sie eine Textnachricht in das Textfeld ein und klicken Sie auf Publish message (Nachricht veröffentlichen), um eine Nachricht zu veröffentlichen.

Senden und Empfangen von SMS-Benachrichtigungen

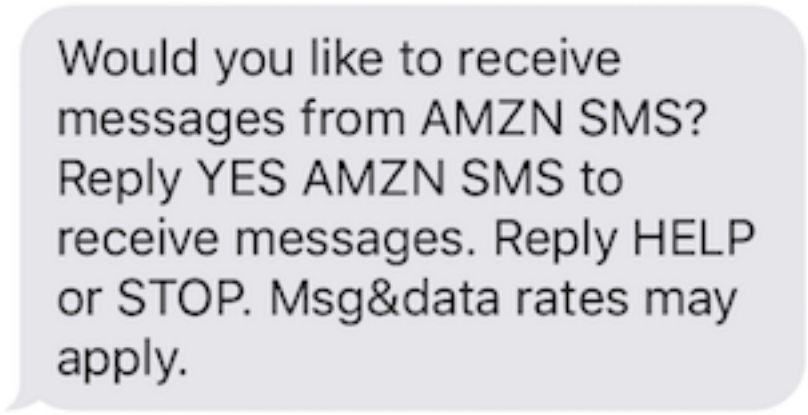
Sie können Amazon Simple Notification Service (Amazon SNS) verwenden, um SMS-Benachrichtigungen (Short Message Service) an SMS-fähige Mobiltelefone und -Smartphones zu senden.

> **Note**
>
> SMS-Benachrichtigungen werden derzeit für Telefonnummern in den USA unterstützt. SMS-Nachrichten können nur zu Themen gesendet werden, die in der Region USA Ost (Nord-Virginia) erstellt wurden. Sie können jedoch Nachrichten für Themen veröffentlichen, die Sie ausgehend von einer anderen Region in der Region USA Ost (Nord-Virginia) erstellen.

Erstellen eines Themas

Erstellen Sie ein Thema wie folgt:

1. Klicken Sie in der Amazon SNS-Konsole auf Create new topic (Neues Thema erstellen). Das Dialogfeld "Create New Topic" wird angezeigt.
2. Geben Sie in das Feld "Topic name" einen Namen für das Thema ein.
3. Geben Sie in das Feld "Display name" einen Anzeigenamen ein. Dem Thema muss ein Anzeigenamen zugewiesen werden, weil die ersten zehn (10) Zeichen des Anzeigenamens als Anfang des Textnachrichtenpräfixes verwendet werden. Der Anzeigename, den Sie eingeben, erscheint in der Bestätigungsnachricht, die SNS an den Benutzer sendet (unten lautet der Anzeigename „AMZN SMS“).



1. Klicken Sie auf Create topic (Thema erstellen). Das neue Thema wird auf der Seite "Topics" angezeigt.
2. Wählen Sie das neue Thema aus und klicken Sie dann auf den ARN des Themas. Die Seite "Topic Details" wird angezeigt.
3. Kopieren Sie den ARN des Themas, den Sie im nächsten Schritt benötigen, wenn Sie ein Thema abonnieren.

```
arn:aws:sns:us-west-2:111122223333:MyTopic
```

Abonnieren eines Themas mit dem SMS-Protokoll

Erstellen Sie einen SNS-Client. Übergeben Sie dabei das Anmeldeinformationsobjekt und die Region des Identitäten-Pools:

```
var snsClient = new AmazonSimpleNotificationServiceClient(credentials, region);
```

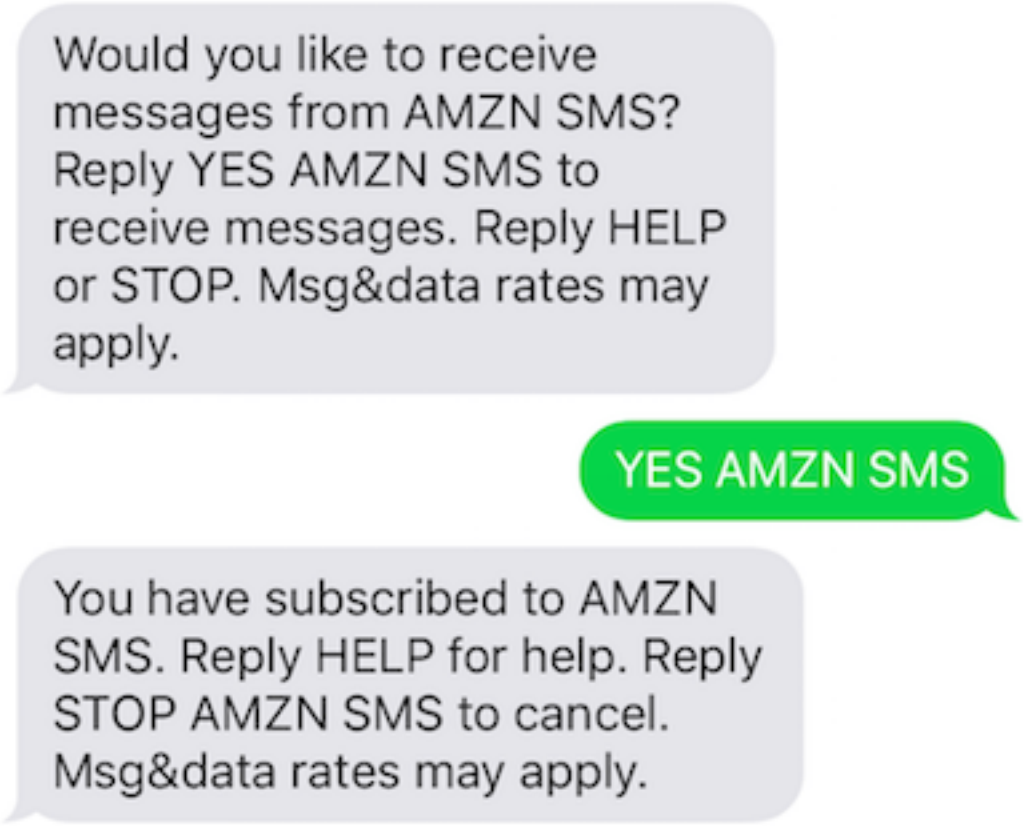
Rufen Sie zum Abonnieren eines Themas `SubscribeAsync` auf und übergeben Sie den ARN des Themas, das Sie abonnieren möchten, das Protokoll („sms“) und die Telefonnummer:

```
var response = await snsClient.SubscribeAsync(topicArn, "sms", "1234567890");
```

Sie empfangen im Antwortobjekt zum Abonnement einen Abonnement-ARN. Der Abonnement-ARN sieht wie folgt aus:

```
arn:aws:sns:us-west-2:123456789012:MyTopic:6b0e71bd-7e97-4d97-80ce-4a0994e55286
```

Wenn ein Gerät ein Thema abonniert, sendet SNS eine Bestätigung an das Gerät. Benutzer müssen bestätigen, dass sie Benachrichtigungen empfangen möchten (siehe unten):



Nachdem Benutzer das Thema abonniert haben, empfangen sie SMS-Nachrichten, sobald Sie diese für das Thema veröffentlichen.

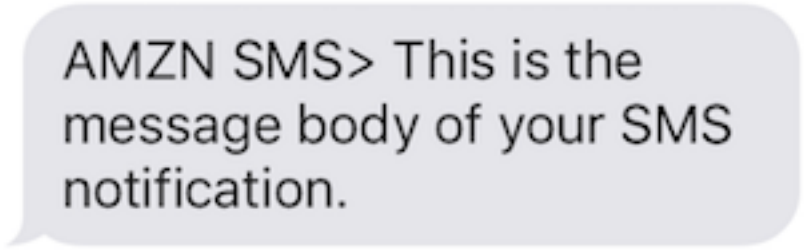
Veröffentlichen einer Nachricht

Veröffentlichen Sie wie folgt eine Nachricht für ein Thema:

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon SNS-Konsole.
2. Klicken Sie im linken Navigationsbereich auf Topics (Themen) und wählen Sie dann das Thema aus, zu dem Sie veröffentlichen möchten.
3. Klicken Sie auf Publish to topic (In einem Thema veröffentlichen).
4. Geben Sie in das Feld "Subject" einen Betreff ein.
5. Geben Sie in das Feld "Message" eine Nachricht ein. Amazon SNS sendet von Ihnen in das Feld "Message" eingegebenen Text an SMS-Abonnenten, es sei denn, Sie geben auch Text in das Feld "Subject" ein. Da Amazon SNS ein Anzeigenamenspräfix mit allen SMS-Nachrichten enthält, die Sie senden, darf die Länge von Anzeigenamenspräfix plus Nachrichteninhalt 140 ASCII-Zeichen

oder 70 Unicode-Zeichen nicht überschreiten. Amazon SNS kürzt Nachrichten, die diese Grenze überschreiten.

6. Klicken Sie auf Publish message (Nachricht veröffentlichen). Amazon SNS zeigt ein Bestätigungsdialogfeld an. Die SMS-Nachricht wird wie unten dargestellt auf dem SMS-fähigen Gerät angezeigt.



Senden von Nachrichten an HTTP/HTTPS-Endpunkte

Sie können Amazon SNS verwenden, um Benachrichtigungen an einzelne oder mehrere HTTP- oder HTTPS-Endpunkte zu senden. Der Prozess läuft folgendermaßen ab:

1. Konfigurieren Sie den Endpunkt für den Empfang von Amazon SNS-Nachrichten.
2. Abonnieren Sie einen HTTP/HTTPS-Endpunkt für ein Thema.
3. Bestätigen Sie das Abonnement.
4. Veröffentlichen Sie eine Benachrichtigung für das Thema. Amazon SNS sendet dann eine HTTP-POST-Anforderung mit dem Inhalt der Benachrichtigung an den abonnierten Endpunkt.

Konfigurieren von HTTP/HTTPS-Endpunkt zum Empfangen von Amazon SNS-Nachrichten

Befolgen Sie die Anweisungen in Schritt 1 unter Senden von Amazon SNS an HTTP-/HTTPS-Endpunkte, um Ihren Endpunkt zu konfigurieren.

Abonnieren des HTTP/HTTPS-Endpunkts für das Amazon SNS-Thema

Erstellen Sie einen SNS-Client. Übergeben Sie dabei das Anmeldeinformationsobjekt und die Region des Identitäten-Pools:

```
var snsClient = new AmazonSimpleNotificationServiceClient(credentials, region);
```

Um Nachrichten über ein Thema an einen HTTP- oder HTTPS-Endpunkt zu senden, müssen Sie den Endpunkt für das Amazon SNS-Thema abonnieren. Sie geben den Endpunkt über seine URL an:

```
var response = await snsClient.SubscribeAsync(
  "topicArn",
  "http",  /* "http" or "https" */
  "endpointUrl" /* endpoint url beginning with http or https */
);
```

Bestätigen des Abonnements

Nachdem Sie einen Endpunkt abonniert haben, sendet Amazon SNS eine Abonnement-Bestätigungsnachricht an den Endpunkt. Der Code am Endpunkt muss den Wert `SubscribeURL` aus der Abonnement-Bestätigungsnachricht abrufen und entweder die in `SubscribeURL` angegebene Position direkt aufrufen oder so verfügbar machen, dass Sie `SubscribeURL` manuell (z. B. mit einem Web-Browser) aufrufen können.

Amazon SNS sendet keine Nachrichten an den Endpunkt, bis das Abonnement bestätigt wird. Wenn Sie `SubscribeURL` aufrufen, enthält die Antwort ein XML-Dokument, das seinerseits ein `SubscriptionArn`-Element enthält, mit dem der ARN für das Abonnement angegeben wird.

Senden von Nachrichten an den HTTP/HTTPS-Endpunkt

Sie können eine Nachricht an die Abonnements eines Themas senden, indem Sie für das Thema veröffentlichen. Rufen Sie `PublishAsync` auf und übergeben Sie der Methode den Themen-ARN und die Nachricht.

```
var response = await snsClient.PublishAsync(topicArn, "This is your message");
```

SNS-Fehlerbehebung

Verwenden des Übermittlungsstatus in der Amazon SNS-Konsole

Die Amazon SNS-Konsole enthält eine "Delivery Status"-Funktion, mit der Sie Rückmeldungen zu erfolgreichen und fehlgeschlagenen Übermittlungsversuchen von Nachrichten an mobile Push-Benachrichtigungsplattformen (Apple (APNS), Google (GCM), Amazon (ADM), Windows (WNS und MPNS) und Baidu) erfassen können.

Die Funktion stellt auch weitere wichtige Informationen wie die dwell-Zeiten in Amazon SNS bereit. Diese Informationen werden in einer Amazon CloudWatch Log-Gruppe erfasst, die automatisch von Amazon SNS erstellt wird, wenn diese Funktion über die Amazon SNS oder über die Amazon SNS aktiviert APIswird.

Anweisungen zur Verwendung der Funktion Delivery Status (Zustellungsstatus) finden Sie unter Verwenden der Funktion Delivery Status (Zustellungsstatus) von Amazon SNS im AWS Mobile Blog.

Bewährte Methoden zur Nutzung von AWS Mobile SDK for .NET and Xamarin

Es gibt einige wenige Grundlagen und bewährte Methoden, deren Kenntnis bei Verwendung von hilfreich ist.AWS Mobile SDK for .NET and Xamarin.

- Verwenden Sie Amazon Cognito, um AWS-Anmeldeinformationen abzurufen, statt die Anmeldeinformationen fest in der Anwendung zu codieren. Wenn Sie die Anmeldeinformationen fest in der Anwendung codieren, werden diese möglicherweise öffentlich, sodass Dritte AWS unter Verwendung Ihrer Anmeldeinformationen aufrufen können. Anweisungen zur Verwendung von Amazon Cognito zum Abrufen von AWS-Anmeldeinformationen erhalten Sie unter Einrichten von AWS Mobile SDK for .NET and Xamarin.
- Bewährte Methoden zur Verwendung von S3 erhalten Sie in diesem Artikel im AWS-Blog.
- Bewährte Methoden zur Verwendung von DynamoDBfinden Sie unter DynamoDB Bewährte Methoden im -DynamoDBEntwicklerhandbuch.

Wir streben stets danach, den Erfolg unserer Kunden sicherzustellen und begrüßen deshalb Ihre Rückmeldungen in Form von Veröffentlichungen in den AWS-Foren oder indem Sie ein Problemticket auf Github öffnen.

Bibliothek der AWS Service-Dokumentation

Zu jedem Service in AWS Mobile SDK for .NET and Xamarin gibt es ein eigenes Entwicklerhandbuch und eine API-Referenz mit weiterführenden Informationen, die hilfreich sein können.

Amazon Cognito-Identität

- Entwicklerhandbuch von Amazon Cognito
- Cognito Identity Service – API-Referenz

Amazon Cognito Sync

- Entwicklerhandbuch von Amazon Cognito
- Cognito Sync Service – API-Referenz

Amazon Mobile Analytics

- Entwicklerhandbuch von Mobile Analytics
- Mobile Analytics Service – API-Referenz

Amazon S3

- Entwicklerhandbuch von S3
- S3 – Handbuch Erste Schritte
- S3 Service – API-Referenz

Amazon – DynamoDB

- DynamoDB Entwicklerhandbuch
- DynamoDB Handbuch „Erste Schritte“
- DynamoDB Service-API-Referenz

Amazon Simple Notification Service (SNS)

- Entwicklerhandbuch von SNS
- SNS Service – API-Referenz

Andere nützliche Links

- AWS-Glossar
- Über AWS-Anmeldeinformationen

Troubleshooting

Dieses Thema beschreibt einige Ideen für die Behandlung von Problemen, die bei Verwendung von auftreten können.AWS Mobile SDK for .NET and Xamarin.

Sicherstellen, dass die IAM-Rolle über die erforderlichen Berechtigungen verfügt

Beim Aufrufen von AWS-Services sollte die App eine Identität aus einem Cognito-Identitäten-Pool verwenden. Jede Identität im Pool ist einer IAM-Rolle (Identity and Access Management) zugeordnet.

Einer Rolle ist mindestens eine Richtliniendatei zugeordnet, die angibt, auf welche AWS-Ressourcen die der Rolle zugewiesenen Benutzer Zugriff erhalten. Standardmäßig werden pro Identitäten-Pool zwei Rollen erstellt: eine für authentifizierte Benutzer und eine für nicht authentifizierte Benutzer.

Sie müssen entweder die vorhandene Richtlinie ändern oder eine neue Richtliniendatei mit den Berechtigungen zuordnen, die von der App benötigt werden. Wenn die App authentifizierte und nicht authentifizierte Benutzer zulässt, müssen beiden Rollen Berechtigungen für den Zugriff auf die von der Anwendung benötigten AWS-Ressourcen gewährt werden.

Die folgende Richtliniendatei zeigt, wie Zugriff auf einen S3-Bucket gewährt wird:

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

Die folgende Richtliniendatei zeigt, wie Zugriff auf eine DynamoDB Datenbank gewährt wird:

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

Weitere Informationen zum Angeben der Richtlinien finden Sie unter IAM-Richtlinien.

Verwenden eines HTTP-Proxy-Debugger

Wenn der von der App aufgerufene AWS-Service einen HTTP- oder HTTPS-Endpunkt hat, können Sie einen HTTP/HTTPS-Proxy-Debugger verwenden, um die Anforderungen und die Antworten anzuzeigen und Einblicke in das Geschehen zu erlangen. Es stehen verschiedene HTTP-Proxy-Debugger zur Verfügung, z. B.:

- Charles: Web-Debugging-Proxy für Windows und OSX
- Fiddler: Web-Debugging-Proxyfidd für Windows

Sowohl Charles als auch Fiddler müssen konfiguriert werden, damit SSL-verschlüsselter Datenverkehr angezeigt werden kann. Weitere Informationen enthält die Dokumentation des betreffenden Dienstprogramms. Wenn Sie einen Web-Debugging-Proxy verwenden, der nicht zum Anzeigen verschlüsselten Datenverkehrs konfiguriert werden kann, öffnen Sie die Datei aws_endpoints_json und weisen dem HTTP-Tag für den zu debuggenden AWS-Service "true" zu.

Dokumentverlauf

In der folgenden Tabelle werden wichtige Änderungen an der Dokumentation seit der letzten Veröffentlichung von beschrieben.AWS Mobile SDK for .NET and Xamarin.

- API-Version: 2015-08-27
- Letzte Aktualisierung der Dokumentation: 2015-08-27

Änderung	API-Version	Beschreibung	Datum der Veröffentlichung
GA-Veröffentlichung	27.08.2015	GA-Veröffentlichung	27.08.2015
Beta-Veröffentlichung	28.07.2015	Beta-Veröffentlichung	rn2015-07-28

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.