

Entwicklerhandbuch

# AWS Mobiles SDK for Unity



# AWS Mobiles SDK for Unity: Entwicklerhandbuch

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

Die Marken und Handelsmarken von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, die geeignet ist, Kunden irrezuführen oder Amazon in irgendeiner Weise herabzusetzen oder zu diskreditieren. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

# Table of Contents

.....	vi
Was ist das AWS Mobile SDK for Unity? .....	1
Verwandte Leitfäden und Themen .....	1
Archivierter Referenzinhalt .....	1
Kompatibilität .....	2
Laden Sie das Mobile SDK for Unity herunter .....	2
Was ist im Mobile SDK for Unity enthalten? .....	2
Set Up the AWS Mobile SDK for Unity .....	3
Voraussetzungen .....	3
Schritt 1: Herunterladen von AWS Mobile SDK for Unity .....	3
Schritt 2: Konfigurieren von AWS Mobile SDK for Unity .....	4
Erstellen einer Szene .....	4
Festlegen der Standardregion für den AWS-Service .....	4
Festlegen der Protokolleinstellungen .....	4
Arbeiten mit der Datei link.xml .....	5
Schritt 3: Abrufen der Identitäten-Pool-ID mit Amazon Cognito .....	6
Nächste Schritte .....	7
Erste Schritte mit AWS Mobile SDK for Unity .....	8
Amazon Cognito Identity .....	8
Amazon Cognito Sync .....	8
Das CognitoSyncManager Beispiel verwenden .....	9
Dynamo DB .....	9
Verwenden des DynamoDB-Beispiels .....	10
Mobile Analytics .....	10
Konfigurieren von Mobile Analytics .....	11
Verwenden des Mobile Analytics-Beispiels .....	11
Amazon S3 .....	12
Konfigurieren der S3-Standardsignatur .....	12
Verwenden des S3-Beispiels .....	12
Amazon Simple Notification Service .....	13
AWS Lambda .....	13
Amazon Cognito Identity .....	15
Was ist Amazon Cognito Identity? .....	15
Verwenden eines öffentlichen Anbieters zum Authentifizieren von Benutzern .....	15

Verwenden entwicklerauthentifizierter Identitäten .....	15
Amazon Cognito Sync .....	16
Amazon Mobile Analytics .....	17
Integrieren von Amazon Mobile Analytics .....	17
Erstellen einer Anwendung in der Mobile Analytics-Konsole .....	17
Integrieren von Mobile Analytics in die App .....	17
Aufzeichnen von Monetarisierungsereignissen .....	18
Aufzeichnen benutzerdefinierter Ereignisse .....	19
Aufzeichnen von Sitzungen .....	20
Amazon Simple Storage Service (S3) .....	21
Erstellen und Konfigurieren eines S3-Bucket .....	21
Erstellen eines S3-Bucket .....	21
Festlegen von Berechtigungen für S3 .....	21
Hochladen von Dateien mit der Konsole .....	22
(Optional) Konfigurieren der Signature Version for S3-Anforderungen .....	23
Erstellen des Amazon S3-Clients .....	23
Auflisten von Buckets .....	23
Auflisten von Objekten .....	24
Herunterladen eines Objekts .....	25
Hochladen eines Objekts .....	25
Amazon DynamoDB .....	27
Integrieren von Amazon DynamoDB .....	27
Erstellen einer DynamoDB-Tabelle .....	28
Erstellen eines DynamoDB-Clients .....	29
Beschreiben einer Tabelle .....	29
Speichern eines Objekts .....	30
Erstellen eines Book-Objekts .....	31
Abrufen eines Book-Objekts .....	31
Aktualisieren eines Book-Objekts .....	32
Löschen eines Book-Objekts .....	33
Amazon Simple Notification Service .....	34
Voraussetzungen .....	3
Festlegen von Berechtigungen für SNS .....	34
iOS-Voraussetzungen .....	35
Android-Voraussetzungen .....	35
Konfigurieren der Unity-Beispiel-App für iOS .....	35

Unity-Konfiguration .....	35
iOS-Konfiguration .....	36
SNS-Konfiguration .....	37
Verwenden von Xcode .....	38
Unity Sample (iOS) .....	38
Konfigurieren der Unity-Beispiel-App für Android .....	39
Unity-Konfiguration .....	39
Android-Konfiguration .....	40
SNS-Konfiguration .....	41
Unity-Beispiel (Android) .....	41
AWS Lambda .....	43
Berechtigungen .....	43
Projekteinrichtung .....	44
Festlegen von Berechtigungen für AWS Lambda .....	44
Erstellen einer neuen Ausführungsrolle .....	44
Erstellen einer Funktion in AWS Lambda .....	45
Erstellen eines Lambda-Clients .....	45
Erstellen eines Anforderungsobjekts .....	45
Aufrufen der Lambda-Funktion .....	45
Fehlersuche .....	47
Sicherstellen, dass die IAM-Rolle über die erforderlichen Berechtigungen verfügt .....	47
Verwenden eines HTTP-Proxy-Debugger .....	48

Das AWS Mobile SDK for Unity ist jetzt in der enthaltenen AWS SDK for .NET. Dieses Handbuch bezieht sich auf die archivierte Version des Mobile SDK for Unity. Weitere Informationen finden Sie unter [Was ist das AWS Mobile SDK for Unity?](#).

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.

# Was ist das AWS Mobile SDK for Unity?

Das AWS Mobile SDK for Unity ist jetzt in der enthaltenen SDK for .NET. Weitere Informationen finden Sie im [AWS SDK for .NET -Entwicklerhandbuch](#).

Dieses Handbuch wird nicht mehr aktualisiert — es bezieht sich auf die archivierte Version des Mobile SDK for Unity.

## Verwandte Leitfäden und Themen

- Für die Entwicklung von Front-End- und mobilen Apps empfehlen wir die Verwendung von [AWS Amplify](#).
- Besondere Hinweise zur Verwendung von AWS SDK for .NET für Ihre Unity-Apps finden Sie unter [Besondere Überlegungen zum Unity-Support](#) im AWS SDK for .NET Entwicklerhandbuch.
- Zu Referenzzwecken finden Sie die archivierte Version des [AWS Mobile SDK for Unity](#) unter GitHub.

## Archivierter Referenzinhalt

Das archivierte Mobile SDK for Unity enthält eine Reihe von .NET-Klassen, mit denen Spiele, die mit Unity geschrieben wurden, AWS Dienste nutzen können. Mit dem Mobile SDK for Unity geschriebene Anwendungen können auf iOS- oder Android-Geräten ausgeführt werden.

Zu den unterstützten AWS Diensten gehören:

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

- [Amazon Simple Storage Service \(Amazon-S3\)](#)

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

## Kompatibilität

Das Mobile SDK for Unity v3 ist mit Unity-Versionen 4.6 und höher kompatibel.

In der neuesten Version des Mobile SDK for Unity wurden Verbesserungen eingeführt, sodass Sie Ihren Code möglicherweise ändern müssen, wenn Sie ihn in Ihr Projekt integrieren. Weitere Informationen zu diesen Änderungen finden Sie unter [Verbesserungen im AWS Mobile SDK for Unity](#) im Front-End Web & Mobile-Blog.

## Laden Sie das Mobile SDK for Unity herunter

Sie können das Mobile SDK for Unity auch als ZIP-Datei [hier](#) herunterladen.

## Was ist im Mobile SDK for Unity enthalten?

Die vollständige Liste der NuGet Pakete, Beispiele und anderen Dateien im Mobile SDK for Unity finden Sie [AWS SDK for .NET](#) unter GitHub.



# Set Up the AWS Mobile SDK for Unity

Um die Arbeit mit AWS Mobile SDK for Unity zu beginnen, können Sie das SDK einrichten und ein neues Projekt erstellen oder das SDK in ein vorhandenes Projekt zu integrieren. Sie können auch die [Beispiele](#) klonen und ausführen, um eine Vorstellung von der Funktionsweise des SDKs zu erhalten.

## Voraussetzungen

Damit Sie AWS Mobile SDK for Unity verwenden können, müssen folgende Voraussetzungen erfüllt sein:

- [Ein AWS-Konto](#)
- Unity Version 4.x oder 5.x (Unity 4.6.4p4 oder Unity 5.0.1p3 wird benötigt, wenn Sie Anwendungen für die 64-Bit-Version von iOS schreiben möchten)

Wenn die Voraussetzungen erfüllt sind, müssen Sie folgende Schritte durchführen:

1. Laden Sie AWS Mobile SDK for Unity herunter.
2. Konfigurieren Sie AWS Mobile SDK for Unity.
3. Rufen Sie AWS-Anmeldeinformationen mit Amazon Cognito ab.

## Schritt 1: Herunterladen von AWS Mobile SDK for Unity

Laden Sie zunächst [AWS Mobile SDK for Unity herunter](#). Jedes Paket im SDK wird für den entsprechenden AWS-Service – basierend auf dem Namen des Pakets – benötigt. Beispielsweise wird das Paket `aws-unity-sdk-dynamodb -2.1.0.0.unitypackage` verwendet, um den AWS DynamoDB-Service aufzurufen. Sie können alle Pakete oder nur diejenigen importieren, die Sie nutzen möchten.

1. Öffnen Sie den Unity-Editor und erstellen Sie mit den Standardeinstellungen ein neues leeres Projekt.
2. Wählen Sie Assets > Import Package > Custom Package.
3. Navigieren Sie im Dialogfeld Import package zu den zu verwendenden .unitypackage-Dateien und wählen Sie diese aus.
4. Prüfen Sie im Dialogfeld Importing package, ob alle relevanten Elemente ausgewählt sind. Klicken Sie dann auf Import.

## Schritt 2: Konfigurieren von AWS Mobile SDK for Unity

### Erstellen einer Szene

Sie können die Arbeit mit AWS Mobile SDK for Unity beginnen, indem Sie die folgende Codezeile in die Methode `Start` oder `Awake` der Klasse mit eindimensionalem Verhalten einfügen:

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

Erstellen Sie die Szene, indem Sie `New Scene` im Menü `File` wählen.

AWS SDK for Unity enthält Client-Klassen für jeden unterstützten AWS-Service. Diese Clients sind so konfiguriert, dass eine Datei namens `awsconfig.xml` verwendet wird. Der folgende Abschnitt beschreibt die am häufigsten in der Datei `awsconfig.xml` verwendeten Einstellungen. Weitere Informationen zu diesen Einstellungen finden Sie unter [Unity SDK API Reference](#).

### Festlegen der Standardregion für den AWS-Service

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

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

Dadurch wird die Standardregion für alle Service-Clients im Unity SDK festgelegt. Diese Einstellung kann überschrieben werden, indem Sie die Region explizit zum Zeitpunkt der Erstellung einer Instance des Service-Clients angeben, z. B.:

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

### Festlegen der Protokolleinstellungen

Protokolleinstellungen werden wie folgt angegeben:

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

Diese Einstellung wird verwendet, um die Protokollierung in Unity zu konfigurieren. Wenn Sie sich bei `UnityLogger` anmelden, schreibt das Framework die Ausgabe intern in die Debug-Protokolle. Wenn

Sie HTTP-Antworten protokollieren möchten, setzen Sie das LogResponses-Flag. Die Werte können Always, Never oder sein.OnError Sie können auch Leistungsmetriken für HTTP-Anfragen mithilfe der LogMetrics-Eigenschaft protokollieren. Das Protokollformat kann mithilfe der LogMetricsFormat Eigenschaft angegeben werden, gültige Werte sind JSON oder Standard.

Das folgende Beispiel zeigt die am häufigsten in der Datei awsconfig.xml verwendeten Einstellungen. Weitere Informationen zu bestimmten Service-Einstellungen finden Sie im Service-Abschnitt unten:

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

## Arbeiten mit der Datei link.xml

Das SDK verwendet für plattformspezifische Komponenten die Spiegelung. Wenn Sie das IL2 CPP-Skripting-Backend verwenden, strip bytecode ist es unter iOS immer aktiviert, Sie benötigen also eine link.xml Datei in Ihrem Assembly-Stammverzeichnis mit den folgenden Einträgen:

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

&lt;/linker&gt;

## Schritt 3: Abrufen der Identitäten-Pool-ID mit Amazon Cognito

Um AWS-Services in einer mobilen Anwendung zu nutzen, müssen Sie die Identitäten-Pool-ID mit Amazon Cognito Identity abrufen. Wenn Sie Amazon Cognito zum Abrufen der Identitäten-Pool-ID verwenden, kann die App auf AWS-Services zugreifen, ohne dass Sie Ihre persönlichen Anmeldeinformationen in die Anwendung einbetten müssen. Außerdem ermöglicht dieses Vorgehen das Festlegen der Zugriffsrechte, um zu steuern, auf welche AWS-Services die Benutzer zugreifen dürfen.

Zur Vorbereitung der Nutzung von Amazon Cognito müssen Sie einen Identitäten-Pool erstellen. Ein Identitäten-Pool ist eine Speicher von Benutzer-Identitätsdaten, die speziell für Ihr Konto gelten. Jeder Identitäten-Pool verfügt über konfigurierbare IAM-Rollen, mit denen Sie festlegen können, auf welche AWS-Services Benutzer der Anwendung zugreifen können. Üblicherweise verwenden Entwickler einen Identitäten-Pool pro Anwendung. Weitere Informationen zu Identitäten-Pools erhalten Sie im [Entwicklerhandbuch von Amazon Cognito](#).

Erstellen Sie einen neuen Identitäten-Pool für die Anwendung wie folgt:

1. Melden Sie sich bei der [Amazon Cognito-Konsole](#) an und klicken Sie auf Create new identity pool.
2. Geben Sie einen Namen für den Identitäten-Pool ein und aktivieren Sie das Kontrollkästchen, um den Zugriff für nicht authentifizierte Identitäten zu aktivieren. Klicken Sie zum Erstellen des Identitäten-Pools auf Create Pool (Pool erstellen).
3. Klicken Sie auf Allow (Zulassen), 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 den Zugriff auf Cognito Sync und Mobile Analytics.

Auf der nächsten Seite wird Code gezeigt, der einen Anmeldeinformationsanbieter erstellt, damit Sie Cognito Identity ganz einfach in die Unity-Anwendung integrieren können. Das Anmeldeinformationsanbieter-Objekt übergeben Sie an den Konstruktor des verwendeten AWS-Clients. Der Code sieht folgendermaßen aus:

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

```
);
```

## Nächste Schritte

- Erste Schritte: Lesen Sie [Erste Schritte mit AWS Mobile SDK for Unity](#), um sich einen ausführlicheren Überblick über die im SDK enthaltenen Services zu verschaffen.
- Ausführen der Demos: Sehen Sie sich unsere [Unity-Beispielanwendungen](#) an, die typische Anwendungsfälle abdecken. Richten Sie zum Ausführen der Beispiel-Apps SDK for Unity wie oben beschrieben ein. Befolgen Sie dann die Anweisungen in den README-Dateien der jeweiligen Beispiele.
- Lesen der API-Referenz: Zeigen Sie die [API-Referenz](#) für AWS Mobile SDK for Unity an.
- Fragen: Veröffentlichen Sie Fragen in den [AWS Mobile SDK-Foren](#) oder [öffnen Sie auf Github ein Problemticket](#).

# Erste Schritte mit AWS Mobile SDK for Unity

Diese Seite bietet einen Überblick zu den verschiedenen AWS-Services in AWS Mobile SDK for Unity und enthält Anleitungen zum Einrichten von Unity-Beispielen. Befolgen Sie alle Anweisungen auf der Seite [Set Up the AWS Mobile SDK for Unity](#), bevor Sie mit der Nutzung der Services unten beginnen.

## Amazon Cognito Identity

Alle AWS-Aufrufe setzen AWS-Anmeldeinformationen voraus. Statt Anmeldeinformationen in den Apps fest zu codieren, empfehlen wir die Verwendung von [Amazon Cognito Identity](#), um AWS-Anmeldeinformationen an die Anwendung zu übergeben. Befolgen Sie die Anweisungen unter [Set Up the AWS Mobile SDK for Unity](#), um AWS-Anmeldeinformationen über Amazon Cognito abzurufen.

Mit Cognito können Sie Benutzer zudem unter Verwendung öffentlicher Anmeldeanbieter wie Amazon, Facebook, Twitter und Google sowie von Anbietern 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 neuen 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 finden Sie im [Entwicklerhandbuch von Amazon Cognito](#).

## Amazon Cognito Sync

[Cognito Sync](#) erleichtert es, Endbenutzerdaten wie Benutzereinstellungen oder Spielstände in der AWS Cloud zu speichern, sodass diese Daten Benutzern unabhängig vom verwendeten Gerät bereitgestellt werden können. Cognito kann diese Daten lokal speichern, damit Apps auch funktionieren, wenn keine Internet-Verbindung verfügbar ist. Wenn eine Internet-Verbindung verfügbar wird, kann die App deren lokale Daten mit der Cloud synchronisieren.

Informationen zu den ersten Schritten mit Cognito Sync finden Sie im [Entwicklerhandbuch von Amazon Cognito](#).

## Das CognitoSyncManager Beispiel verwenden

Navigieren Sie im Bereich Projekt zu Assets//examples AWSSDK/CognitoSync, und wählen Sie auf der rechten Seite des Bereichs die Szene aus, um die CognitoSyncSzene zu öffnen.

Klicken Sie zum Ausführen des Beispiels oben im Editorbildschirm auf die Wiedergabeschaltfläche. Wenn die App ausgeführt wird, werden einige Textfelder und Schaltflächen angezeigt, mit deren Hilfe Sie Daten zu den Spielern erfassen können. Darunter befinden sich einige Schaltflächen, mit denen Spielerdaten lokal gespeichert, lokale Spielerdaten mit der Cognito Cloud synchronisiert, Spielerdaten aus der Cognito Cloud aktualisiert und die lokalen Spielerdaten gelöscht werden können. Klicken Sie auf die Schaltflächen, um die betreffende Operation auszuführen. Das Beispiel zeigt die Rückmeldung auf dem Spielbildschirm an.

Um das CognitoSyncManager Beispiel zu konfigurieren, müssen Sie eine Cognito Identity Pool-ID angeben. Um diesen Wert anzugeben, wählen Sie ihn im Unity-Editor SyncManagerim Bereich Hierarchie aus und geben Sie ihn in das Textfeld IDENTITY\_POOL\_ID im Inspektorfenster ein.

### Note

Das CognitoSyncManager Beispiel enthält Code, der veranschaulicht, wie Sie den Facebook-Identitätsanbieter verwenden und nach dem Makro „USE\_FACEBOOK\_LOGIN“ suchen. Dies setzt die Verwendung von Facebook SDK for Unity voraus. Weitere Informationen finden Sie unter [Facebook SDK for Unity](#).

## Dynamo DB

[Amazon DynamoDB](#) ist ein schneller, hochskalierbarer, hochverfügbarer, wirtschaftlicher, nicht relationaler Datenbank-Service. Mit DynamoDB werden Einschränkungen der Skalierbarkeit des Datenspeichers eliminiert, die Latenz wird niedrig gehalten und die Leistung ist vorhersehbar.

AWS SDK für Unity stellt Low-Level- und High-Level-Bibliotheken für das Arbeiten mit DynamoDB bereit. Die High-Level-Bibliothek enthält DynamoDB Object Mapper. Mit dieser Komponente können Sie Client-seitige Klassen DynamoDB-Tabellen zuordnen, verschiedene Operationen zum Erstellen, Lesen, Aktualisieren und Löschen (sogenannte CRUD-Operationen) durchführen und Abfragen ausführen. Mit DynamoDB Object Mapper können Sie einfachen, lesbaren Code schreiben, der Objekte in der Cloud speichert.

Weitere Informationen über DynamoDB finden Sie im [Entwicklerhandbuch von DynamoDB](#).

Weitere Informationen über die Nutzung von Dynamo DB in Unity-Anwendungen erhalten Sie unter [Amazon DynamoDB](#).

## Verwenden des DynamoDB-Beispiels

Navigieren Sie im Bereich Projekt zu Assets//examples AWSSDK/DynamoDB. Dieses Beispiel besteht aus folgenden Szenen:

- Dynamo DBExample — die erste Szene der App
- LowLevelDynamoDbExample - Beispiel mit Low-Level-DynamoDBD-API
- TableQueryAndScanExample - Beispiel, das zeigt, wie Abfragen ausgeführt werden
- HighLevelExample - Beispiel mit dynamoDB-API auf hoher Ebene

Fügen Sie diese Szenen in der oben angegebenen Reihenfolge in den Build ein. Verwenden Sie dazu das Dialogfeld "Build Settings" (wählen Sie zum Öffnen "File.Build Settings"). In diesem Beispiel werden vier Tabellen erstellt: ProductCatalog, Forum, Thread, Reply.

Klicken Sie zum Ausführen des Beispiels oben im Editorbildschirm auf die Wiedergabeschaltfläche. Wenn die Anwendung ausgeführt wird, zeigt sie einige Schaltflächen an:

- Low Level Table Operations: zeigt, wie Tabellen erstellt, aufgelistet, aktualisiert, beschrieben und gelöscht werden.
- Mid Level Query & Scan Operations: illustriert die Ausführung von Abfragen.
- High Level Object Mapper: zeigt, wie Objekte erstellt, aktualisiert und gelöscht werden.

## Mobile Analytics

Mit [Amazon Mobile Analytics](#) können Sie das Kundenverhalten verfolgen, Daten generieren, Kennzahlen zusammenstellen, Datenvisualisierungen generieren und aussagekräftige Muster identifizieren. AWS SDK für Unity unterstützt die Integration mit dem Service Amazon Mobile Analytics. Weitere Informationen zu Mobile Analytics erhalten Sie im [Benutzerhandbuch von Mobile Analytics](#). Weitere Informationen zur Verwendung von Mobile Analytics in Unity-Anwendungen erhalten Sie unter [Amazon Mobile Analytics](#).



## Konfigurieren von Mobile Analytics

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

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

- `sessionTimeout`: Dieses Zeitintervall gibt an, wann eine Sitzung beendet werden kann, nachdem die Anwendung in den Hintergrund verschoben wurde.
- `max DBSize` — Das ist die Größe der SQLite Datenbank. Wenn die Datenbank die maximale Größe erreicht, werden alle weiteren Ereignisse ignoriert.
- `dbWarningThreshold` - Dies ist die Obergrenze für die Größe der Datenbank. Sobald sie erreicht ist, werden Warnprotokolle generiert.
- `maxRequestSize` - Dies ist die maximale Größe der Anfrage in Byte, die in einer HTTP-Anfrage an den mobilen Analysedienst übertragen werden sollte.
- `allowUseDataNetzwerk` — Ein boolescher Wert, der angibt, ob die Sitzungsereignisse im Datennetzwerk gesendet werden.

## Verwenden des Mobile Analytics-Beispiels

Navigieren Sie im Bereich Projekt zu `Assets//examples AWSSDK/Mobile Analytics` und wählen Sie auf der rechten Seite des Bereichs die Amazon Mobile Analytics Analytics-Beispielszene aus, um die Szene zu öffnen. Zur Verwendung des Beispiels müssen Sie Ihre App mit der [Amazon Mobile Analytics-Konsole](#) hinzufügen. Weitere Informationen zur Verwendung der Mobile Analytics-Konsole finden Sie im [Benutzerhandbuch von Amazon Mobile Analytics](#).

Befolgen Sie diese Schritte zum Konfigurieren des Beispiels, bevor Sie es ausführen:

1. Wählen Sie das `AmazonMobileAnalyticsSample` Spielobjekt aus.
2. Geben Sie die App-ID (erstellt in der [Amazon Mobile Analytics-Konsole](#)) in das Feld „App Id (App-ID)“ ein.
3. Geben Sie Ihre Cognito-Identitäten-Pool-ID (erstellt mit der [Amazon Cognito-Konsole](#)) in das Feld „Cognito Identity Pool Id (Cognito-Identitäten-Pool-ID)“ ein.

4. Stellen Sie sicher, dass Ihre authentifizierten und nicht authentifizierten Rollen über Zugriffsrechte für den Service Mobile Analytics verfügen. Weitere Informationen zum Anwenden von Richtlinien auf IAM-Rollen finden Sie unter [Managing Roles](#).

Bedenken Sie bei Ausführung der Beispielanwendung, dass Ereignisse möglicherweise nicht sofort zum Backend-Service übertragen werden. Ein Hintergrund-Thread puffert Ereignisse lokal und sendet sie stapelweise in regelmäßigen Intervallen (der Standardwert ist 60 Sekunden) an das Amazon Mobile Analytics-Backend, damit die Ausführung des Spiels nicht beeinträchtigt wird. Aufgrund der komplexen Verarbeitung Ihrer Daten durch Amazon Mobile Analytics kann es nach dem Übertragen bis zu 60 Minuten dauern, bis Ereignisse und entsprechende Berichte in der AWS-Konsole angezeigt werden.

Weitere Informationen über die von Amazon Mobile Analytics bereitgestellten Berichte finden Sie unter [Report and Mobile Metrics](#).

## Amazon S3

Amazon Simple Storage Service (Amazon S3) stellt für Entwickler und IT-Teams einen sicheren, dauerhaften und hochskalierbaren Objektspeicher bereit. Von Unity aus können Sie S3 verwenden, um Bilder, Videos, Musik und andere von Ihren Spielen verwendete Daten zu speichern, aufzulisten und abzurufen.

Weitere Informationen zu S3 erhalten Sie unter [Amazon S3](#) und [Erste Schritte mit S3](#).

Weitere Informationen über die Nutzung von S3 in Unity-Anwendungen erhalten Sie unter [Amazon Simple Storage Service \(S3\)](#).

## Konfigurieren der S3-Standardsignatur

Die S3-Standardsignatur wird folgendermaßen konfiguriert:

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

So wird angegeben, ob Signaturversion 4 für S3-Anforderungen zu verwenden ist.

## Verwenden des S3-Beispiels

Navigieren Sie im Projektbereich zu Assets//examples AWSSDK/S3 und wählen Sie auf der rechten Seite des Bereichs die Szene S3Example aus, um die Szene zu öffnen. Das Beispiel zeigt, wie Sie

Buckets und Objekte in einem Bucket auflisten, Objekte an einen Bucket senden und Objekte aus einem Bucket herunterladen. Befolgen Sie diese Schritte zum Konfigurieren des Beispiels, bevor Sie es ausführen:

1. Wählen Sie das S3-Spielobjekt im Bereich Hierarchy.
2. Geben Sie im Inspektorbereich Werte für S3 BucketName und ein SampleFileName.  
S3 BucketName ist der Name des Buckets, der vom Sample verwendet wird, und S3 SampleFileName ist der Name der Datei, die das Sample in den angegebenen S3-Bucket hochladen wird.
3. Stellen Sie sicher, dass Ihre authentifizierten und nicht authentifizierten Rollen über Zugriffsrechte für S3-Buckets in Ihrem Konto verfügen. Weitere Informationen zum Anwenden von Richtlinien auf IAM-Rollen finden Sie unter [Managing Roles](#).

Klicken Sie zum Ausführen des Beispiels oben im Editorbildschirm auf die Wiedergabeschaltfläche. Wenn die Anwendung ausgeführt wird, zeigt sie einige Schaltflächen an:

- Get Objects: Ruft eine Liste aller Objekte in allen Buckets in Ihrem AWS-Konto ab.
- Get Buckets: Ruft eine Liste aller Buckets in Ihrem AWS-Konto ab.
- Post Object: Lädt ein Objekt in einen angegebenen S3-Bucket hoch.
- Delete Object: Löscht alle Objekte aus einem angegebenen S3-Bucket.

Das Beispiel zeigt die Rückmeldung auf dem Spielbildschirm an.

## Amazon Simple Notification Service

Amazon Simple Notification Service 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 Amazon Simple Notification Service erhalten Sie unter [Amazon Simple Notification Service](#).

## AWS Lambda

AWS Lambda ist ein Datenverarbeitungsservice, der Ihren Code als Reaktion auf Anforderungen oder Ereignisse ausführt und automatisch die Datenverarbeitungsressourcen für Sie verwaltet.

Dadurch ist es einfach, Anwendungen zu erstellen, die schnell auf neue Informationen reagieren. AWS Lambda-Funktionen können direkt von Mobilgeräte-, IoT- und Web-Apps aufgerufen werden. Sie senden synchron eine Antwort zurück und erleichtern so das Erstellen skalierbarer und sicherer Backends mit hoher Verfügbarkeit für mobile Apps, ohne dass Infrastruktur bereitgestellt oder verwaltet werden muss. Weitere Informationen finden Sie unter [AWS Lambda](#).

# Amazon Cognito Identity

## Was ist Amazon Cognito Identity?

Mit Amazon Cognito Identity können Sie eindeutige Identitäten für Benutzer erstellen und diese für den sicheren Zugriff auf AWS-Ressourcen wie Amazon S3 oder Amazon DynamoDB authentifizieren. Amazon Cognito Identity unterstützt öffentliche Identitätsanbieter – Amazon, Facebook, Twitter/Digits, Google und alle OpenID Connect-kompatiblen Anbieter – sowie nicht authentifizierte Identitäten. Cognito unterstützt zudem entwicklerauthentifizierte Identitäten, sodass Sie Benutzer über einen eigenen Backend-Authentifizierungsprozess registrieren und authentifizieren und weiterhin [Amazon Cognito Sync](#) verwenden können, um Benutzerdaten zu synchronisieren und auf AWS-Ressourcen zuzugreifen.

Weitere Informationen zu Cognito Identity erhalten Sie im [Entwicklerhandbuch von Amazon Cognito](#).

Weitere Informationen zur regionalen Verfügbarkeit von Cognito Authentication erhalten Sie unter [Amazon Cognito Identity Region Availability](#).

## Verwenden eines öffentlichen Anbieters zum Authentifizieren von Benutzern

Informationen zur Nutzung öffentlicher Identitätsanbieter wie Amazon, Facebook, Twitter/Digits oder Google für die Authentifizierung von Benutzern erhalten Sie unter [External Providers](#) im Entwicklerhandbuch von Amazon Cognito.

## Verwenden entwicklerauthentifizierter Identitäten

Weitere Informationen zu entwicklerauthentifizierten Identitäten erhalten Sie unter [Entwicklerauthentifizierte Identitäten](#) im Entwicklerhandbuch von Amazon Cognito.

# 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 Benutzerdaten gerätübergreifend zu synchronisieren. Um Cognito Sync in der App zu verwenden, müssen Sie AWS Mobile SDK for Unity in das Projekt aufnehmen.

Anweisungen zum Integrieren von Amazon Cognito Sync in die Anwendung finden Sie im [Entwicklerhandbuch von Amazon Cognito](#).

# Amazon Mobile Analytics

Mit Amazon Mobile Analytics können Sie das Kundenverhalten verfolgen, Daten generieren, Kennzahlen zusammenstellen, Datenvisualisierungen generieren und aussagekräftige Muster identifizieren. Weitere Informationen zu Mobile Analytics erhalten Sie unter [AWS Mobile Analytics](#).

## Integrieren von Amazon Mobile Analytics

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-Konsole](#) und erstellen Sie eine App. Notieren Sie sich den appId-Wert, da Sie ihn später benötigen werden.

#### Note

Weitere Informationen zum Arbeiten in der Konsole finden Sie im [Benutzerhandbuch von Amazon Mobile Analytics](#).

Beim Erstellen einer App in der Mobile Analytics-Konsole müssen Sie eine Cognito-Identitäten-Pool-ID angeben. Informationen zum Erstellen eines neuen Cognito-Identitäten-Pools und zum Generieren einer ID finden Sie im [Entwicklerhandbuch von Cognito Identity](#).

## Integrieren von Mobile Analytics in die App

Um mit Unity auf Mobile Analytics zuzugreifen, benötigen Sie die folgenden Verwendungsanweisungen:

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

Eine bewährte Methode besteht darin, Amazon Cognito zu verwenden, um temporäre AWS-Anmeldeinformationen für die Anwendung bereitzustellen. Diese Anmeldeinformationen ermöglichen der Anwendung den Zugriff auf die AWS-Ressourcen. Befolgen Sie zum Erstellen eines Anmeldeinformationsanbieters die Anweisungen unter [Amazon Cognito-Identität](#).

Instanzieren Sie eine MobileAnalyticsManager Instanz mit den folgenden Informationen:

- `cognitoIdentityPoolId` — Die ID des Cognito Identity Pools für Ihre App
- `cognitoRegion` — Die Region für Ihren Cognito Identity Pool, zum Beispiel „`RegionEndpoint USEast1`“
- `region` — Die Region für den Mobile Analytics Analytics-Dienst, zum Beispiel „`RegionEndpoint. USEast1`“
- `appId`: Der von der Mobile Analytics-Konsole beim Hinzufügen der App generierte Wert

Verwenden Sie die `MobileAnalyticsClientContextConfig`, um eine **`MobileAnalyticsManager`** Instanz zu initialisieren, wie im folgenden Codeausschnitt gezeigt:

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

#### Note

Die App-ID wird bei Ausführung des Assistenten zur App-Erstellung generiert. Beide Werte müssen denen in der Mobile Analytics-Konsole entsprechen.

Die `appId` 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. Daraufhin wird die Seite App Management angezeigt, auf der alle registrierten Apps und ihre Apps aufgelistet sind. IDs

## Aufzeichnen von Monetarisierungsereignissen

SDK for Unity 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 monetization 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 und Sie ein Ereignis aufzeichnen möchten, wenn ein Benutzer einen Level abschließt. Erstellen Sie ein Ereignis „LevelComplete“, indem Sie eine neue `AmazonMobileAnalyticsEvent` Instanz 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

Wenn die Anwendung den Fokus abgibt, kann die Sitzung angehalten werden. Prüfen Sie in `OnApplicationFocus`, ob die App angehalten wurde. Wenn dies der Fall ist, rufen Sie `PauseSession` und andernfalls `ResumeSession` auf (siehe folgenden Code-Ausschnitt):

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

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 Datei `awsconfig.xml` konfiguriert werden. Weitere Informationen finden Sie im Abschnitt "Konfigurieren von Mobile Analytics" in [Erste Schritte mit AWS Mobile SDK for Unity](#).

# Amazon Simple Storage Service (S3)

Amazon Simple Storage Service (Amazon S3) stellt für Entwickler und IT-Teams einen sicheren, dauerhaften und hochskalierbaren Objektspeicher bereit. Unity-Entwickler können S3 nutzen, um von ihren Spielen genutzte Komponenten dynamisch zu laden. Dies kann das erstmalige Herunterladen von Spielen aus App-Stores beschleunigen.

Weitere Informationen über S3 erhalten Sie unter [Amazon S3](#).

Weitere Informationen zur regionalen Verfügbarkeit von AWS S3 finden Sie in der [AWS-Tabelle der Regionen](#).

## Note

Einige Beispiele in diesem Dokument setzen die Verwendung einer Textfeldvariablen voraus, die aufgerufen wird, ResultText um die Trace-Ausgabe anzuzeigen.

## Erstellen und Konfigurieren eines S3-Bucket

Amazon S3 speichert Ressourcen 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-Konsole](#) zum Erstellen eines Bucket verwenden.

### Erstellen eines S3-Bucket

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 auf **Create Role Policy**, wählen Sie **Policy Generator** und klicken Sie dann auf **Select**.
5. Geben Sie auf der Seite **Edit Permissions** die in der folgenden Abbildung gezeigten Einstellungen ein. Ersetzen Sie dabei den Amazon-Ressourcennamen (ARN) durch den eigenen Namen. 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.

## Edit Permissions

The policy generator enables you to create policies that control access to Amazon Web Services (AWS) products and resources. For more information about creating policies, see [Overview of Policies](#) in Using AWS Identity and Access Management.

**Effect** Allow ☒ Deny ☐

**AWS Service** Amazon S3

**Actions** All Actions Selected

**Amazon Resource Name (ARN)** arn:aws:s3:::examplebucket/\*

[Add Conditions \(optional\)](#)

**Add Statement**

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](#).

## Hochladen von Dateien mit der Konsole

Gehen Sie zum Hochladen einer Testdatei in den Bucket wie folgt vor:

1. Klicken Sie in der Bucket-Ansicht der S3-Konsole auf **Upload**.

2. Klicken Sie auf Add Files und wählen Sie eine hochzuladende Testdatei. Für dieses Tutorial nehmen wir an, dass Sie ein Bild namens `myImage.jpg` hochladen.
3. Klicken Sie bei ausgewähltem Testbild auf Start Upload.

## (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-Anfragen nicht [unterstützt](#), müssen Sie AWSConfigs S3 einstellen. UseSignatureVersionEigenschaft 4 auf „true“.

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

## Erstellen des Amazon S3-Clients

Um Amazon S3 verwenden zu können, müssen wir zunächst eine AmazonS3Client-Instanz erstellen, die einen Verweis auf die AWSCredentials Cognito-Instanz verwendet, die Sie zuvor erstellt haben:

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

Die Klasse AmazonS3Client ist der Eintrittspunkt für die S3-High-Level-API.

## Auflisten von Buckets

Listen Sie die Buckets in einem AWS-Konto durch Aufrufen der Methode AmazonS3Client.ListBucketsAsync wie im folgenden Beispiel-Code auf:

```
// ResultText is a label used for displaying status information
ResultText.text = "Fetching all the Buckets";
Client.ListBucketsAsync(new ListBucketsRequest(), (responseObject) =>
```

```
{
    ResultText.text += "\n";
    if (responseObject.Exception == null)
    {
        ResultText.text += "Got Response \nPrinting now \n";
        responseObject.Response.Buckets.ForEach((s3b) =>
        {
            ResultText.text += string.Format("bucket = {0}, created date = {1} \n",
                s3b.BucketName, s3b.CreationDate);
        });
    }
    else
    {
        ResultText.text += "Got Exception \n";
    }
});
```

## Auflisten von Objekten

Listen Sie alle Objekte in einem Bucket durch Aufrufen der Methode `AmazonS3Client.ListObjectsAsync` wie im folgenden Beispiel-Code auf:

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

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

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

```
{
    ResultText.text += "Got Exception \n";
}
});
```

## Herunterladen eines Objekts

Um ein Objekt herunterzuladen, erstellen Sie ein `GetObjectRequest`, geben Sie den Bucket-Namen und den Schlüssel an und übergeben Sie das Objekt an einen Aufruf an den Client. `GetObjectAsync`:

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

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

`GetObjectAsync` nimmt eine Instanz von `GetObjectRequest`, einen Callback und eine `AsyncOptions` Instanz. Der Rückruf muss vom Typ `AmazonServiceCallback<GetObjectRequest, GetObjectResponse>` sein. Die `AsyncOptions` Instanz ist optional. Wenn sie angegeben wird, ermittelt sie, ob der Rückruf für den Haupt-Thread ausgeführt wird.

## Hochladen eines Objekts

Um ein Objekt hochzuladen, schreiben Sie Ihr Objekt in einen Stream, erstellen Sie einen neuen `PostObjectRequest` und geben Sie den Schlüssel, den Bucket-Namen und die Stream-Daten an.

Das AWS SDK for Unity verwendet den WWW-HTTP-Client, der die HTTP-PUT-Operation nicht unterstützt. Zum Hochladen eines Objekts in den S3-Bucket müssen Sie – wie unten dargestellt – S3 Browser Post verwenden.

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

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

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

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

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



# Amazon DynamoDB

[Amazon DynamoDB](#) ist ein schneller, hochskalierbarer, hochverfügbarer, wirtschaftlicher, nicht relationaler Datenbank-Service. Mit DynamoDB werden Einschränkungen der Skalierbarkeit des Datenspeichers eliminiert, die Latenz wird niedrig gehalten und die Leistung ist vorhersehbar. Informationen zu DynamoDB erhalten Sie unter [Amazon DynamoDB](#).

AWS Mobile SDK for Unity stellt eine High-Level-Bibliothek für das Arbeiten mit DynamoDB bereit. Sie können Anforderungen auch direkt an die DynamoDB-Low-Level-API richten. Für die meisten Anwendungsfälle wird die Verwendung der High-Level-Bibliothek empfohlen. Das 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.

## Note

Einige Beispiele in diesem Dokument setzen die Verwendung einer Textfeldvariablen voraus, die aufgerufen wird, ResultText um die Trace-Ausgabe anzuzeigen.

## Integrieren von Amazon DynamoDB

Um DynamoDB in einer Unity-Anwendung verwenden zu können, müssen Sie das Unity SDK in das Projekt aufnehmen. Wenn dies noch nicht geschehen ist, [laden Sie das SDK for Unity](#) herunter und befolgen Sie die Anweisungen unter [Set Up the AWS Mobile SDK for Unity](#). Wir empfehlen die Verwendung von Amazon Cognito Identity, um temporäre AWS-Anmeldeinformationen für Anwendungen bereitzustellen. Mit diesen Anmeldeinformationen kann eine App auf AWS-Services und -Ressourcen zugreifen.

Soll DynamoDB in einer Anwendung verwendet werden, müssen Sie die richtigen Berechtigungen festlegen. Die folgende IAM-Richtlinie ermöglicht Benutzern das Löschen, Abrufen, Schreiben, Scannen und Aktualisieren von Elementen in einer DynamoDB-Tabelle, die über den [ARN](#) angegeben 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"
}]
}
```

Diese Richtlinie sollte auf die Rollen angewendet werden, die dem Cognito-Identitäten-Pool zugewiesen sind. Sie müssen jedoch den Wert für **Resource** durch den ARN der relevanten DynamoDB-Tabelle ersetzen. Cognito erstellt automatisch eine Rolle für den neuen Identitäten-Pool und Sie können in der [IAM-Konsole](#) Richtlinien auf diese Rolle anwenden.

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 erhalten Sie unter [Using IAM to Control Access to DynamoDB Resources](#).

## Erstellen einer DynamoDB-Tabelle

Nachdem wir nun die Berechtigungen und Anmeldeinformationen eingerichtet haben, können wir eine DynamoDB-Tabelle für die Anwendung erstellen. Rufen Sie zum Erstellen einer Tabelle die [DynamoDB-Konsole](#) auf und befolgen Sie diese Schritte:

1. Klicken Sie auf Create Table.
2. Geben Sie Bookstore als Namen der Tabelle ein.
3. Wählen Sie Hash als Typ des Primärschlüssels.
4. Wählen Sie Number und geben Sie id als Hash-Attributnamen ein. Klicken Sie auf Weiter.
5. Klicken Sie erneut auf Continue, um das Hinzufügen von Indizes zu überspringen.
6. Legen Sie die Lesekapazität auf 10 und die Schreibkapazität auf 5 fest. Klicken Sie auf Weiter.
7. Geben Sie eine Benachrichtigungs-E-Mail ein und klicken Sie auf Continue, um Durchsatzalarme zu erstellen.
8. Klicken Sie auf Create. DynamoDB erstellt die Datenbank.

# Erstellen eines DynamoDB-Clients

Damit die Anwendung mit einer DynamoDB-Tabelle interagieren kann, benötigen wir einen Client. Wir können folgendermaßen einen DynamoDB-Standard-Client erstellen:

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

Die `AmazonDynamoDBClient` Klasse ist der Einstiegspunkt für die DynamoDB-API. Die Klasse stellt neben anderen Operationen Instance-Methoden zum Erstellen, Beschreiben, Aktualisieren und Löschen von Tabellen bereit. Der Kontext ergänzt eine weitere Abstraktionsebene über dem Client und ermöglicht die Verwendung zusätzlicher Funktionen, z. B. des Object Persistence-Modells.

## Beschreiben einer Tabelle

Mit dem folgenden Code können wir eine Beschreibung der DynamoDB-Tabelle abrufen:

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

```
    }, null);  
}
```

In diesem Beispiel erstellen wir einen Client und ein DescribeTableRequest Objekt, weisen der **TableName**Eigenschaft den Namen unserer Tabelle zu und übergeben dann das Anforderungsobjekt an die DescribeTableAsync Methode für das Objekt. AmazonDynamoDBClient DescribeTableAsync benötigt auch einen Delegaten, der aufgerufen wird, wenn der asynchrone Vorgang abgeschlossen ist.

#### Note

Alle asynchronen Methoden auf den AmazonDynamoDBClient Take-Delegaten, die aufgerufen werden, wenn der asynchrone Vorgang abgeschlossen ist.

## Speichern eines Objekts

Um ein Objekt in DynamoDB zu speichern, verwenden Sie die SaveAsync <T>Methode des AmazonDynamoDBClient Objekts, wobei T der Objekttyp ist, den Sie speichern.

Wir haben die Datenbank „Bookstore“ genannt und implementieren dementsprechend ein Datenmodell, das buchspezifische Attribute aufzeichnet. Dies sind die Klassen, die das Datenmodell definieren.

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

Für eine echte Buchhandlungsanwendung würden wir natürlich weitere Felder beispielsweise für Autor und Preis benötigen. Die Book-Klasse ist mit dem [DynamoDBTable] -Attribut versehen. Dieses

definiert die Datenbanktabelle, in die Objekte des Typs Book geschrieben werden. Der Schlüssel für jede Instanz der Book-Klasse wird mithilfe des [Dynamo DBHash Key] -Attributs identifiziert. Eigenschaften werden mit dem [DynamoDBProperty] -Attribut identifiziert. Diese geben die Spalte in der Datenbanktabelle an, in die die Eigenschaft geschrieben wird. Sobald das Modell vorliegt, können wir einige Methoden zum Erstellen, Abrufen, Aktualisieren und Löschen von Book-Objekten schreiben.

## Erstellen eines Book-Objekts

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

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

## Abrufen eines Book-Objekts

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

```

    }
    _retrievedBook = result.Response;
    this.displayMessage += ("Retrieved Book: " +
                           "\nId=" + _retrievedBook.Id +
                           "\nTitle=" + _retrievedBook.Title +
                           "\nISBN=" + _retrievedBook.ISBN);

    string authors = "";
    foreach(string author in _retrievedBook.BookAuthors)
        authors += author + ",";
    this.displayMessage += "\nBookAuthor= " + authors;
    this.displayMessage += ("\nDimensions= " + _retrievedBook.Dimensions.Length + "
X " +
                           _retrievedBook.Dimensions.Height + " X " +
                           _retrievedBook.Dimensions.Thickness);

    }, null);
}

```

## Aktualisieren eines Book-Objekts

```

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

```

## Löschen eines Book-Objekts

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

# Amazon Simple Notification Service

Unter Verwendung von Amazon Simple Notification Service (SNS) und Unity SDK können Sie iOS- und Android-Apps schreiben, die mobile Push-Benachrichtigungen empfangen. Weitere Informationen zu SNS erhalten Sie unter [Amazon Simple Notification Service](#).

Dieses Thema führt Sie durch die Konfiguration der SNSExample Beispiel-App.unity, des AWS-SDK für Unity, für den Empfang mobiler Push-Benachrichtigungen über Amazon SNS.

Mit dem SNSExample .unity-Beispiel können Sie sowohl iOS- als auch Android-Apps erstellen. Die Konfigurationsschritte unterscheiden sich zwischen iOS und Android. Lesen Sie unten den Abschnitt zur Zielplattform.

## Voraussetzungen

Die folgenden Voraussetzungen sind für die Verwendung dieser Lösung erforderlich.

### Festlegen von Berechtigungen für SNS

Wenn Sie einen Cognito-Identitäten-Pool erstellen, werden zwei IAM-Rollen generiert:

- Cognito/\_ <Identity-Pool-Name>Auth\_ DefaultRole — die Standard-IAM-Rolle für authentifizierte Benutzer
- Cognito/\_ <Identity-Pool-Name>Unauth\_ DefaultRole — die Standard-IAM-Rolle für nicht authentifizierte Benutzer

Sie müssen diesen Rollen Berechtigungen für den Zugriff auf den Service Amazon SNS hinzufügen. So gehen Sie vor:

1. Navigieren Sie zur [IAM-Konsole](#) und wählen Sie die zu konfigurierende IAM-Rolle.
2. Klicken Sie auf Richtlinie anhängen, wählen Sie die Amazon SNSFull Access-Richtlinie aus und klicken Sie auf Richtlinie anhängen.

#### Note

Die Verwendung von Amazon SNSFull Access in einer Produktionsumgebung wird nicht empfohlen. Wir verwenden es hier, damit Sie schnell loslegen können. Weitere Informationen



über das Festlegen von Berechtigungen für eine IAM-Rolle erhalten Sie unter [Overview of IAM Role Permissions](#).

## iOS-Voraussetzungen

- Mitgliedschaft im Apple iOS Developer Program
- Generieren einer Signaturidentität
- Erstellen eines für Push-Benachrichtigungen konfigurierten Bereitstellungsprofils

Sie müssen die App auf einem physischen Gerät ausführen, um Push-Benachrichtigungen zu empfangen. Zum Ausführen der App auf einem Gerät benötigen Sie eine [Apple iOS Developer Program Membership](#). Sobald Sie eine Mitgliedschaft besitzen, können Sie mit Xcode eine Signaturidentität generieren. Weitere Informationen enthält die Dokumentation [App Distribution Quick Start](#) von Apple. Nun benötigen Sie ein Bereitstellungsprofil, das für Push-Benachrichtigungen konfiguriert ist. Weitere Informationen enthält die Apple-Dokumentation [Configuring Push Notifications](#).

## Android-Voraussetzungen

- Installieren des Android-SDK
- Installieren des JDK
- android-support-v4.jar
- google-play-services.jar

## Konfigurieren der Unity-Beispiel-App für iOS

Öffnen Sie den Unity-Editor und erstellen Sie ein neues Projekt. Importieren Sie das Paket AWS SDK for Unity, indem Sie Assets/Import Package/Custom Package und dann aws-unity-sdk-sns-2.0.0.1.unitypackage auswählen. Stellen Sie sicher, dass alle Elemente im Dialogfeld Importing Package ausgewählt sind. Klicken Sie dann auf Import.

## Unity-Konfiguration

Führen Sie zum Konfigurieren des Unity-Projekts die folgenden Schritte durch:

1. Navigieren Sie im Bereich Projekt zu Assets//AWSSDKexamples und öffnen Sie die Szene. SNSEExample
2. Wählen Sie im Hierarchiebereich die Option aus SNSEExample.
3. Geben Sie im Bereich Inspector die Cognito-Identitäten-Pool-ID an.
4. Beachten Sie das Textfeld für iOS Platform Application ARN. Die betreffenden Informationen werden Sie später generieren.
5. Wählen Sie File/Build Settings und klicken Sie im Dialogfeld Build Settings auf die Schaltfläche Add Current unter dem Listenfeld Scenes in Build, um die aktuelle Szene in den Build einzufügen.
6. Wählen Sie unter Platform (Plattform) die Option iOS aus und klicken Sie auf die Schaltfläche Player Settings... (Playereinstellungen...). Klicken Sie im Bereich Inspector des Unity-Editors auf das iPhone-Symbol. Scrollen Sie dann nach unten zum Bereich Identification (Identifizierung) und geben Sie einen Bundle Identifier (Bundle-ID) an.

## iOS-Konfiguration

Führen Sie die folgenden Schritte zum Konfigurieren des Beispiels mit iOS-spezifischen Einstellungen durch:

1. Navigieren Sie in einem Web-Browser zu [Apple Developer Member Center](#) und klicken Sie auf Certificates, Identifiers & Profiles.
2. Klicken Sie unter iOS Apps auf Identifiers, klicken Sie oben rechts auf der Web-Seite auf die Plus-Schaltfläche, um eine neue iOS-App-ID hinzuzufügen, und geben Sie eine Beschreibung für die App-ID ein.
3. Scrollen Sie nach unten zum Bereich Add ID Suffix, wählen Sie Explicit App ID und geben Sie die Bundle-ID ein.
4. Scrollen Sie nach unten zum Bereich App Services und wählen Sie Push Notifications.
5. Klicken Sie auf die Schaltfläche Continue.
6. Klicken Sie auf die Schaltfläche Submit.
7. Klicken Sie auf die Schaltfläche Done.
8. Wählen Sie die gerade erstellte App-ID und klicken Sie dann auf die Schaltfläche Edit.
9. Scrollen Sie nach unten zum Bereich Push Notifications.
10. Klicken Sie auf die Schaltfläche Create Certificate unter Development SSL Certificate.

11. 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.
12. Sobald Sie wieder zurück auf der Webseite Certificates, Identifiers & Profiles sind, klicken Sie auf All unter Provisioning Profiles.
13. Klicken Sie auf die Plus-Schaltfläche oben rechts, um ein neues Bereitstellungsprofil hinzuzufügen.
14. Wählen Sie iOS App Development und klicken Sie auf die Schaltfläche Continue.
15. Wählen Sie die App-ID und klicken Sie auf die Schaltfläche Continue.
16. Wählen Sie das Entwicklerzertifikat und klicken Sie auf die Schaltfläche Continue.
17. Wählen Sie das Gerät und klicken Sie auf die Schaltfläche Continue.
18. Geben Sie einen Profilnamen ein und klicken Sie auf die Schaltfläche Generate.
19. Laden Sie die Bereitstellungsdatei herunter und doppelklicken Sie dann auf die Datei, um das Bereitstellungsprofil zu installieren.

Sie müssen die Bereitstellungsprofile in Xcode aktualisieren, nachdem ein neues Bereitstellungsprofil hinzugefügt wurde. In Xcode:

1. Wählen Sie die Menüoption Xcode/Preferences.
2. Wählen Sie die Registerkarte Accounts, wählen Sie die Apple-ID und klicken Sie auf View Details.
3. Klicken Sie unten links im Dialogfeld auf die Schaltfläche zum Aktualisieren der Bereitstellungsprofile und prüfen Sie, ob das neue Bereitstellungsprofil angezeigt wird.

## SNS-Konfiguration

1. Führen Sie die KeyChain Access-App aus, wählen Sie unten links auf dem Bildschirm Meine Zertifikate aus, klicken Sie mit der rechten Maustaste auf das SSL-Zertifikat, das Sie für die Verbindung mit APNS generiert haben, und wählen Sie 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. Rufen Sie in einem Web-Browser die [SNS-Konsole](#) auf und klicken Sie auf der linken Seite des Bildschirms auf Applications.
3. Klicken Sie auf Create platform application, um eine Anwendung für die SNS-Plattform zu erstellen.
4. Geben Sie einen Application Name ein.

5. Wählen Sie Apple Push Notification Service Sandbox (APNS\_SANDBOX) für Push notification platform.
6. Klicken Sie auf Choose File und wählen Sie die P12-Datei, 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.
10. Kehren Sie im Unity-Editor zu Ihrem Projekt zurück, wählen Sie es SNSExample im Hierarchiebereich und im Inspektorbereich aus und fügen Sie den Platform Application ARN in das Textfeld iOS Platform Application ARN ein.
11. Wählen Sie File/Build Settings und klicken Sie auf die Schaltfläche Build, um ein Xcode-Projekt zu erstellen.

## Verwenden von Xcode

1. Öffnen Sie das Xcode-Projekt und wählen Sie das Projekt in Project Navigator aus.
2. Überprüfen der richtigen Einstellung der Bundle-ID
3. Überprüfen Sie, ob Ihr Apple Developer-Konto in Team angegeben ist. Das ist erforderlich, damit das Bereitstellungsprofil wirksam wird.
4. Erstellen Sie das Projekt und führen Sie es auf dem Gerät aus.
5. Tippen Sie auf Register for Notification und dann auf OK, um Benachrichtigungen zuzulassen. Die App zeigt das Geräte-Token an.

Klicken Sie in der [SNS-Konsole](#) auf Applications, wählen Sie die Plattformanwendung und klicken Sie auf Create Platform Endpoint. Geben Sie das von der App angezeigte Geräte-Token ein.

An diesem Punkt sind App, APNS und NSN vollständig konfiguriert. Sie können die Plattformanwendung und dann den Endpunkt wählen. Klicken Sie auf Publish to endpoint, um eine Push-Benachrichtigung an das Gerät zu senden.

## Unity Sample (iOS)

Das Beispiel erstellt eine AWSCredentials Cognito-Instanz, um temporäre Anmeldeinformationen mit begrenztem Umfang zu generieren, die es der App ermöglichen, AWS-Services aufzurufen.

Außerdem wird eine Instanz für die Kommunikation mit `AmazonSimpleNotificationServiceClient` SNS erstellt. Die App zeigt zwei Schaltflächen an, die mit `Register for Notification` und `Unregister` beschriftet sind.

Beim Tippen auf die Schaltfläche `Register for Notifications` wird die `RegisterDevice()` Methode aufgerufen. `RegisterDevice()` ruft `UnityEngine.iOS.NotificationServices.RegisterForNotifications`, auf, die angibt, welche Benachrichtigungstypen (Warnung, Sound oder Stempel) verwendet werden. Außerdem erfolgt ein asynchroner Aufruf von APNS, um ein Geräte-Token abzurufen. Da kein Rückruf definiert ist, wird `CheckForDeviceToken` wiederholt (bis zu 10-mal) aufgerufen, um auf das Geräte-Token zu prüfen.

Wenn ein Token abgerufen wird, wird `AmazonSimpleNotificationServiceClient.CreatePlatformEndpointAsync()` aufgerufen, um einen Endpunkt für die SNS-Plattformanwendung zu erstellen.

Das Beispiel ist jetzt so konfiguriert, dass Push-Benachrichtigungen empfangen werden. Sie können zur [SNS-Konsole](#) navigieren, links auf der Seite auf `Applications` klicken und die Plattformanwendung wählen. Wählen Sie einen Endpunkt und klicken Sie auf `Publish to endpoint`. Wählen Sie den zu verwendenden Endpunkt und klicken Sie auf `Publish to Endpoint`. Geben Sie eine Textnachricht in das Textfeld ein und klicken Sie auf `Publish message` (Nachricht veröffentlichen), um eine Nachricht zu veröffentlichen.

## Konfigurieren der Unity-Beispiel-App für Android

Öffnen Sie den Unity-Editor und erstellen Sie ein neues Projekt. Importieren Sie das Paket AWS SDK for Unity, indem Sie `Assets/Import Package/Custom Package` und dann `aws-unity-sdk-sns-2.0.0.1.unitypackage` auswählen. Stellen Sie sicher, dass alle Elemente im Dialogfeld `Importing Package` ausgewählt sind. Klicken Sie dann auf `Import`.

## Unity-Konfiguration

Führen Sie zum Konfigurieren des Unity-Projekts die folgenden Schritte durch:

1. Navigieren Sie im Bereich `Projekt` zu `Assets//AWSSDKexamples` und öffnen Sie die Szene. `SNSExample`
2. Wählen Sie im Hierarchiebereich die Option aus `SNSExample`.
3. Geben Sie im Bereich `Inspector` die `Cognito-Identitäten-Pool-ID` an.

4. Beachten Sie die Textfelder für Android Platform Application ARN und Google Console Project ID. Die betreffenden Informationen werden Sie später generieren.
5. Wählen Sie File/Build Settings und klicken Sie im Dialogfeld Build Settings auf die Schaltfläche Add Current unter dem Listenfeld Scenes in Build, um die aktuelle Szene in den Build einzufügen.
6. Wählen Sie unter Platform (Plattform) die Option Android aus und klicken Sie auf die Schaltfläche Player Settings... (Playereinstellungen...). Klicken Sie im Bereich Inspector des Unity-Editors auf das Android-Symbol. Scrollen Sie dann nach unten zum Bereich Identification (Identifizierung) und geben Sie einen Bundle Identifier (Bundle-ID) an.
7. Kopieren Sie android-support-v 4.jar und google-play-services .jar in das Verzeichnis Assets/Plugins/Android im Bereich Projekt.

Weitere Informationen darüber, wo Sie android-support-v 4.jar finden, finden Sie unter [Einrichtung der Android-Supportbibliothek](#). Weitere Informationen zum Auffinden von google-play-services .jar finden Sie unter [Google APIs für Android Setup](#).

## Android-Konfiguration

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

1. Navigieren Sie in einem Webbrowser zur [Google Developers-Konsole](#) und klicken Sie auf Create Project.
2. Geben Sie einen Projektnamen in das Feld New Project ein. Notieren Sie sich die (später benötigte) Projektnummer und klicken Sie auf Create.

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

1. In der Google Developers-Konsole sollte das neue Projekt bereits ausgewählt sein. Ist das nicht der Fall, wählen Sie es in der Dropdown-Liste oben auf der Seite aus.
2. Wählen Sie in der Seitenleiste links auf der Seite die Option APIs & auth aus.
3. Geben Sie in das Suchfeld „Google Cloud Messaging for Android“ ein und klicken Sie darunter 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 Console APIs & auth > Credentials 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 resultierenden Dialogfeld auf Create und kopieren Sie den angezeigten API-Schlüssel.

Sie werden den API-Schlüssel später zur Authentifizierung verwenden.

## SNS-Konfiguration

1. Rufen Sie in einem Web-Browser die [SNS-Konsole](#) auf und klicken Sie auf der linken Seite des Bildschirms auf Applications.
2. Klicken Sie auf Create platform application, um eine Anwendung für die SNS-Plattform zu erstellen.
3. Geben Sie einen Application Name ein.
4. Wählen Sie Google Cloud Messaging (GCM) für Push notification platform.
5. Fügen Sie den API-Schlüssel in das Textfeld API key ein.
6. Klicken Sie auf Create platform application.
7. Wählen Sie die soeben erstellte Plattformanwendung und kopieren Sie den ARN der Anwendung.
8. Kehren Sie im Unity-Editor zu Ihrem Projekt zurück, wählen Sie es SNSExampleim Hierarchiebereich im Inspektorbereich aus und fügen Sie den Platform Application ARN in das Textfeld Android Platform Application ARN und Ihre Projektnummer in das Textfeld Google Console Project ID ein.
9. Verbinden Sie das Android-Gerät mit dem Computer. Wählen Sie File/Build Settings und klicken Sie auf Build and Run.

## Unity-Beispiel (Android)

Das Beispiel erstellt eine AWSCredentials Cognito-Instanz, um temporäre Anmeldeinformationen mit begrenztem Umfang zu generieren, die es der App ermöglichen, AWS-Services aufzurufen. Außerdem wird eine Instanz für die Kommunikation mit AmazonSimpleNotificationServiceClient SNS erstellt.

Die App zeigt zwei Schaltflächen an, die mit Register for Notification und Unregister beschriftet sind. Beim Tippen auf die Schaltfläche Register for Notifications wird die RegisterDevice() Methode aufgerufen. RegisterDevice() ruft GCM.Register, auf, die wiederum die App bei GCM

registriert. GCM ist eine im Beispielcode definierte Klasse. Sie führt einen asynchronen Aufruf durch, um die App bei GCM zu registrieren.

Wenn der Rückruf aufgerufen wird, wird `AmazonSimpleNotificationServiceClient.CreatePlatformEndpointAsync` zum Erstellen eines Plattformendpunkts für den Empfang von SNS-Nachrichten aufgerufen.

Das Beispiel ist jetzt so konfiguriert, dass Push-Benachrichtigungen empfangen werden. Sie können zur [SNS-Konsole](#) navigieren, links auf der Seite auf Applications klicken und die Plattformanwendung wählen. Wählen Sie einen Endpunkt und klicken Sie auf Publish to endpoint. Wählen Sie den zu verwendenden Endpunkt und klicken Sie auf Publish to Endpoint. Geben Sie eine Textnachricht in das Textfeld ein und klicken Sie auf Publish message (Nachricht veröffentlichen), um eine Nachricht zu veröffentlichen.



# AWS Lambda

AWS Lambda ist ein Datenverarbeitungsservice, der Ihren Code als Reaktion auf Anforderungen oder Ereignisse ausführt und automatisch die Datenverarbeitungsressourcen für Sie verwaltet. Dadurch ist es einfach, Anwendungen zu erstellen, die schnell auf neue Informationen reagieren. AWS Lambda-Funktionen können direkt von Mobilgeräte-, IoT- und Web-Apps aufgerufen werden. Sie senden synchron eine Antwort zurück und erleichtern so das Erstellen skalierbarer und sicherer Backends mit hoher Verfügbarkeit für mobile Apps, ohne dass Infrastruktur bereitgestellt oder verwaltet werden muss.

AWS Lambda kann Lambda-Funktionen als Reaktion auf einen der folgenden Umstände ausführen:

- Ereignisse, wie z. B. diskrete Updates (z. B. von Objekten erstellte Ereignisse in Amazon S3 oder CloudWatch Benachrichtigungen) oder Streaming-Updates (z. B. Website-Clickstreams oder Ausgaben von verbundenen Geräten).
- JSON-Eingaben oder HTTPS-Befehle von Ihren benutzerdefinierten Anwendungen.

AWS Lambda führt Ihren Code nur bei Bedarf aus und skaliert automatisch – von einigen Anforderungen pro Tag bis zu Tausenden pro Sekunde. Dank dieser Funktionen können Sie Lambda verwenden, um auf einfache Weise Trigger für AWS-Services wie Amazon S3 und Amazon DynamoDB zu erstellen, in Amazon Kinesis gespeicherte Streaming-Daten verarbeiten oder ein eigenes Backend zu erstellen, das in AWS-Größenordnungen sowie mit AWS-Leistung und -Sicherheit arbeitet.

Weitere Informationen zur Funktionsweise von AWS Lambda erhalten Sie unter [AWS Lambda: How It Works](#).

## Berechtigungen

Es gibt zwei Arten von Berechtigungen im Zusammenhang mit Lambda-Funktionen:

- **Ausführungsberechtigungen:** Die Berechtigungen, die Ihre Lambda-Funktion für den Zugriff auf andere AWS-Ressourcen in Ihrem Konto benötigt. Sie erteilen diese Berechtigungen, indem Sie eine IAM-Rolle erstellen, die auch als Ausführungsrolle bezeichnet wird.
- **Aufrufberechtigungen:** Die Berechtigungen, die die Ereignisquelle für die Kommunikation mit Ihrer Lambda-Funktion benötigt. Je nach Aufrufmodell (Push- oder Pull-Modell) können Sie diese

Berechtigungen unter Verwendung der Ausführungsrolle oder der Ressourcenrichtlinien (die der Lambda-Funktion zugeordnete Zugriffsrichtlinie) gewähren.

## Projekteinrichtung

### Festlegen von Berechtigungen für AWS Lambda

1. Öffnen Sie die [AWS IAM-Konsole](#).
2. Ordnen Sie diese benutzerdefinierte Richtlinie Ihren Rollen zu, damit die Anwendung Aufrufe an AWS Lambda richten kann.

#### JSON

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

### Erstellen einer neuen Ausführungsrolle

Diese Rolle gilt für die Lambda-Funktion, die Sie im nächsten Schritt erstellen. Sie bestimmt, auf welche AWS-Ressourcen die Funktion zugreifen kann.

1. Öffnen Sie die [AWS IAM-Konsole](#).
2. Klicken Sie auf Rollen.
3. Klicken Sie auf Create New Roles.
4. Befolgen Sie die Anweisungen auf dem Bildschirm, um die Services und die zugehörigen Richtlinien auszuwählen, auf die Ihre Lambda-Funktion Zugriff benötigt. Wenn Sie beispielsweise

möchten, dass Ihre Lambda-Funktion einen S3-Bucket erstellt, benötigt die Richtlinie Schreibzugriff auf S3.

5. Klicken Sie auf Create Role.

## Erstellen einer Funktion in AWS Lambda

1. Öffnen Sie die [AWS Lambda-Konsole](#).
2. Klicken Sie auf Create a Lambda function.
3. Klicken Sie auf Skip, um das Erstellen einer Vorlage auszulassen.
4. Konfigurieren Sie Ihre eigene Funktion im nächsten Bildschirm. Geben Sie den Funktionsnamen und eine Beschreibung ein und wählen Sie die Laufzeit. Befolgen Sie die Anweisungen auf dem Bildschirm basierend auf der gewählten Laufzeit. Legen Sie die Ausführungsberechtigungen fest, indem Sie die neu erstellte Ausführungsrolle Ihrer Funktion zuweisen.
5. Klicken Sie auf Next, wenn Sie fertig sind.
6. Klicken Sie auf Create Function.

## Erstellen eines Lambda-Clients

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

## Erstellen eines Anforderungsobjekts

Erstellen Sie ein Anforderungsobjekt, um den Aufruftyp und den Funktionsnamen anzugeben:

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

## Aufrufen der Lambda-Funktion

Rufen Sie "invoke" unter Übergabe des Anforderungsobjekts auf:

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

# Fehlersuche

Aufgrund von Beschränkungen der Klasse `Unity.WWW`, die von AWS SDK for Unity verwendet wird, werden keine detaillierten Fehlermeldungen zurückgegeben, wenn ein Problem beim Aufrufen eines AWS-Services auftritt. In diesem Thema werden einige Ideen zur Fehlerbehebung bei solchen Problemen beschrieben.

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

Beim Aufrufen von AWS-Services verwendet die App eine Identität aus einem Cognito-Identitäten-Pool. Jede Identität im Pool ist einer IAM-Rolle (Identity and Access Management) zugeordnet. Der Rolle ist mindestens eine Richtliniendatei zugeordnet, die angibt, auf welche AWS-Ressourcen die der Rolle zugewiesenen Benutzer Zugriff erhalten. Standardmäßig werden 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 OSX
- [Fiddler](#): Web-Debugging-Proxyfidd für Windows

### Important

Es gibt ein bekanntes Problem mit dem Cognito-Anmeldeinformationsanbieter, wenn der Web-Debugging-Proxy Charles ausgeführt wird, der ein ordnungsgemäßes Funktionieren des Anmeldeinformationsanbieters verhindert.

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 für die Anzeige von verschlüsseltem Datenverkehr konfiguriert werden kann, öffnen Sie die Datei

`aws_endpoints_json` (befindet sich unter `AWSUnitySDK/AWSCore/Resources`) und setzen Sie das HTTP-Tag für den AWS-Service, den Sie debuggen müssen, auf `true`