Guida per sviluppatori Xamarin

AWS Mobile SDK

AWS Mobile SDK: Guida per sviluppatori Xamarin

Table of Contents

L'SDKAWS Mobile per Xamarin è ora incluso inAWS SDK for .NET. Questa guida fa riferimento alla versione archiviata di Mobile SDK per Xamarin.

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.

Cos'è l'SDK AWS per .NET e Xamarin

L'SDK AWS Mobile per Xamarin è incluso in AWS SDK for .NET. Per ulteriori informazioni, consulta la Guida per gli sviluppatori di AWS SDK for .NET.

Questa guida non è più aggiornata: fa riferimento alla versione archiviata di Mobile SDK per Xamarin.

Guide e argomenti correlati

- Per lo sviluppo di app front-end e mobili, consigliamo di utilizzare AWS Amplify.
- Per considerazioni speciali sull'utilizzo delle app XamarinAWS SDK for .NET per le tue app, consulta Considerazioni speciali per il supporto di Xamarin nella Guida per gliAWS SDK for .NET sviluppatori.
- A scopo di riferimento, puoi trovare la versione archiviata di AWSMobile SDK per Xamarin su GitHub.

Contenuto di riferimento archiviato

L'SDK for .NET and Xamarin AWS per .NET

- Xamarin
- Xamarin
- Windows Phone Silverlight
- Windows RT 8.1
- Windows Phone 8.1

Le app mobili scritte utilizzando l'SDK AWS Mobile per .NET e Xamarin chiamano API di piattaforma native in modo da avere l'aspetto delle applicazioni native. Le librerie.NET nell'SDK forniscono wrapper C# per le API REST di AWS.

Cosa è incluso nell'SDK AWS per .NET e Xamarin

I servizi AWS supportati attualmente includono, a titolo esemplificativo ma non esaustivo:

- Amazon Cognito

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

Questi servizi consentono di autenticare gli utenti, salvare dati di giocatori e giochi, salvare oggetti nel cloud, ricevere notifiche push e raccogliere e analizzare i dati di utilizzo.

L'SDK AWS Mobile per .NET e Xamarin consente inoltre di utilizzare la maggior parte dei servizi AWS supportati dall'AWS SDK for .NET. I servizi AWS specifici per lo sviluppo mobile sono spiegati in questa guida per gli sviluppatori. Per ulteriori informazioni sull'AWS SDK for .NET, consulta:

- Guida introduttiva
- Guida per gli sviluppatori dell'SDK AWS per .NET
- Riferimento all'API AWS

compatibilità

L'SDK AWS Mobile per .NET e Xamarin viene fornito come Portable Class Library (PCL). Il Support PCL è stato aggiunto in Xamarin.Android 4.10.1 e Xamarin.iOS 7.0.4. I progetti Portable Library sono integrati in Visual Studio.

IDE

Per ulteriori informazioni sull'utilizzo degli IDE con la versione archiviata di Xamarin SDK, vedereConfigurazione dell'SDK AWS Mobile per.NET e Xamarin.

Come posso ottenere l'SDK AWS per .NET e Xamarin

Per ottenere l'SDK AWS Mobile per .NET e Xamarin, consulta Configurazione dell'SDK AWS Mobile per .NET e Xamarin. L'SDK AWS Mobile per .NET e Xamarin è distribuito come NuGet pacchetti. Puoi trovare un elenco completo dei pacchetti di servizi AWS nei pacchetti SDK AWS NuGet o nel GitHubrepository AWS SDK for .NET.

Informazioni sui servizi mobili AWS

Amazon Cognito Identity

Tutte le chiamate effettuate ad AWS richiedono credenziali AWS. Anziché codificare le credenziali nelle app, ti consigliamo di utilizzare Amazon Cognito Identity per fornire credenziali AWS alla tua applicazione. Segui le istruzioni in Configurazione dell'SDK AWS Mobile per .NET e Xamarin per ottenere le credenziali AWS tramite Amazon Cognito.

Cognito consente inoltre di autenticare gli utenti utilizzando provider di accesso pubblici come Amazon, Facebook, Twitter e Google, nonché provider che supportano OpenID Connect. Cognito funziona anche con utenti non autenticati. Cognito fornisce credenziali temporanee con diritti di accesso limitati specificate con un ruolo IAM (Identity and Access Management). Cognito è configurato tramite la creazione di un pool di identità associato a un ruolo IAM. Il ruolo IAM specifica le risorse/i servizi a cui l'app può accedere.

Per iniziare a usare Cognito Identity, consulta la pagina relativa alla configurazione di AWS Mobile Web

Per ulteriori informazioni su Cognito Identity, consulta Amazon Cognito Identity.

Amazon Cognito Sync

Cognito Sync è una libreria di servizi e client AWS che consente la sincronizzazione tra più dispositivi di dati dell'utente relativi all'applicazione. Puoi utilizzare l'API Cognito Sync per sincronizzare i dati del profilo utente tra dispositivi e provider di accesso: Amazon, Facebook, Google e il tuo provider di identità personalizzato.

Per iniziare a usare Cognito Sync, vedi Sincronizzare i dati utente con Cognito Sync.

Per ulteriori informazioni su Cognito Sync, consulta Amazon Cognito Sync.

Mobile Analytics

Amazon Mobile Analytics ti consente di raccogliere, visualizzare e comprendere l'utilizzo delle app per dispositivi mobili. I report sono disponibili per le metriche su utenti attivi, sessioni, fidelizzazione, entrate in-app ed eventi personalizzati e possono essere filtrati per piattaforma e intervallo di date. Amazon Mobile Analytics è progettato per adattarsi alla tua azienda e può raccogliere ed elaborare miliardi di eventi da molti milioni di endpoint.

Per iniziare a usare Mobile Analytics, consulta Monitoraggio dei dati di utilizzo delle app con Amazon Mobile Analytics.

Per ulteriori informazioni su Mobile Analytics, consulta Amazon Mobile Analytics.

Dynamo DB

Amazon DynamoDB è un servizio di database non relazionale, conveniente, veloce e altamente scalabile e disponibile. DynamoDB rimuove le tradizionali limitazioni di scalabilità sullo storage dei dati mantenendo una bassa latenza e prestazioni prevedibili.

Per iniziare a usare Dynamo DB, consulta Archiviare e recuperare dati con DynamoDB.

Per ulteriori informazioni su Dynamo DB, consulta Amazon DynamoDB.

Amazon Simple Notification Service

Amazon Simple Notification Service (SNS) è un servizio di notifica push veloce, flessibile e completamente gestito che consente di inviare singoli messaggi o di estrarre messaggi a un numero elevato di destinatari. Amazon Simple Notification Service rende semplice ed economico l'invio di notifiche push agli utenti di dispositivi mobili, ai destinatari delle e-mail o persino l'invio di messaggi ad altri servizi distribuiti.

Per iniziare a usare SNS per Xamarin iOS, consulta Ricevere notifiche push tramite SNS (Xamarin iOS).

Per iniziare a usare SNS per Xamarin Android, consulta Ricevere notifiche push tramite SNS (Xamarin Android).

Per ulteriori informazioni su SNS, consulta Amazon Simple Notification Service (SNS).

Configurazione dell'SDK AWS Mobile per.NET e Xamarin

Puoi configurare AWS Mobile SDK for .NET and Xamarin e iniziare a creare un nuovo progetto oppure integrare l'SDK con un progetto esistente. È inoltre possibile clonare ed eseguire il filecampioniper avere un'idea di come funziona l'SDK. Segui questi passaggi per configurare e iniziare a utilizzare l'SDK AWS Mobile per.NET e Xamarin.

Prerequisiti

Per poter utilizzare l'SDK AWS Mobile per.NET e Xamarin, devi eseguire le seguenti operazioni:

- Creazione di unUn account AWS.
- InstallaXamarin.

Dopo aver completato i prerequisiti:

1. Ottieni credenziali AWS utilizzando Amazon Cognito.
2. Imposta le autorizzazioni richieste per ogni servizio AWS che utilizzerai nella tua applicazione.
3. Crea un nuovo progetto nel tuo IDE.
4. Installare l'SDK AWS Mobile per.NET e Xamarin.
5. Configurazione dell'SDK AWS Mobile per.NET e Xamarin.

Fase 1: Ottieni le credenziali AWS

Per effettuare chiamate ad AWS nella tua applicazione, devi prima ottenere le credenziali AWS. Puoi farlo utilizzando Amazon Cognito, un servizio AWS che consente alla tua applicazione di accedere ai servizi nell'SDK senza dover incorporare le credenziali AWS private nell'applicazione.

Per iniziare a utilizzare Amazon Cognito, devi creare un pool di identità. Un pool di identità è un archivio di informazioni specifico per il tuo account ed è identificato da un ID univoco del pool di identità simile al seguente. :

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

1. Accedi alConsole Amazon Cognito, scegliGestisci le identità federate die quindi scegliereCrea un nuovo pool di identità di.

2. Inserisci un nome per il pool di identità e seleziona la casella di controllo per abilitare l'accesso alle identità non autenticate. ScegliereCrea poolper creare il pool di identità.
3. SceglioreAbilitaper creare i due ruoli predefiniti associati al pool di identità, uno per gli utenti non autenticati e uno per gli utenti autenticati. Questi ruoli predefiniti forniscono a Amazon Cognito Sync e Amazon Mobile Analytics l'accesso al pool di identità.

In genere, si utilizza un solo pool di identità per applicazione.

Una volta creato il pool di identità, ottieni le credenziali AWS creando un`CognitoAWSCredentials`object (passando il tuo ID del pool di identità) e quindi passandolo al costruttore di un client AWS come segue. :

```
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
);
```

Fase 2: Impostazione delle autorizzazioni

Devi impostare le autorizzazioni per ogni servizio AWS da utilizzare nell'applicazione. Innanzitutto, devi capire come AWS visualizza gli utenti della tua applicazione.

Quando qualcuno utilizza la tua applicazione e effettua chiamate ad AWS, AWS assegna all'utente un'identità. Il pool di identità creato nel passaggio 1 è il punto in cui AWS memorizza queste identità. Esistono due tipi di identità: autenticata e non autenticata. Le identità autenticate appartengono agli utenti autenticati da un provider di accesso pubblico (ad esempio Facebook, Amazon, Google). Le identità non autenticate appartengono a utenti guest.

Ogni identità è associata a unAWS Identity and Access ManagementRuolo . Nel passaggio 1 sono stati creati due ruoli IAM, uno per gli utenti autenticati e uno per gli utenti non autenticati. A ogni ruolo IAM sono associate una o più policy che specificano a quali servizi AWS le identità assegnate a quel

ruolo possono accedere. Ad esempio, la seguente policy di esempio concede l'accesso a un bucket Amazon S3. :

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

Per impostare le autorizzazioni per i servizi AWS che si desidera utilizzare nell'applicazione, modificare la policy associata ai ruoli.

1. Passa aIAM Console e scegli Ruoli. Digitare il nome del pool di identità nella casella di ricerca. Scegliere il ruolo IAM che si desidera configurare. Se l'applicazione consente utenti autenticati e non autenticati, è necessario concedere le autorizzazioni per entrambi i ruoli.
2. Fare clic suCollegamento della policy, seleziona la policy desiderata, quindi fai clic suCollegamento della policy. Le politiche predefinite per i ruoli IAM creati forniscono l'accesso ad Amazon Cognito Sync e Mobile Analytics.

Per ulteriori informazioni sulla creazione di policy o per scegliere tra un elenco di policy esistenti, consultaPolicy IAM.

Fase 3: Creazione di un nuovo progetto

Windows

È possibile utilizzare Visual Studio per sviluppare l'applicazione.

OS X

È possibile utilizzare Visual Studio per sviluppare le applicazioni. Lo sviluppo iOS con Xamarin richiede l'accesso a un Mac per eseguire l'app. Per ulteriori informazioni, consultaInstallazione di Xamarin.iOS su Windows.

> **Note**
>
> L'IDE commerciale multipiattaformaCavaliere daJetBrainsinclude il supporto Xamarin su piattaforme Windows e Mac.

Fase 4: Installare l'SDK AWS Mobile per.NET e Xamarin

Windows

Opzione 1: Installare utilizzando la Console di Gestione pacchetti

L'SDK AWS Mobile per .NET e Xamarin è costituito da un set di assembly .NET. Per installare AWS Mobile SDK for .NET and Xamarin, esegui il comando install-package per ciascun pacchetto nella console di Package Manager. Ad esempio, per installare Cognito Identity, eseguire quanto segue. :

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

I pacchetti AWS Core Runtime e Amazon Cognito Identity sono necessari per tutti i progetti. Di seguito è riportato un elenco completo dei nomi di pacchetti per ciascun servizio.

Servizio	Package name (Nome pacchetto)
Runtime AWS Core	AWSSDK.Core
Amazon Cognito Sync	AWSSDK.CognitoSync
Amazon Cognito Identity	AWSSDK.CognitoIdentity
Amazon DynamoDB	AWSSDK.dynamoDBv2
Amazon Mobile Analytics	AWSSDK.MobileAnalytics

Servizio	Package name (Nome pacchetto)
Amazon S3	AWSSDK.S3
Amazon SNS	AWSSDK.SimpleNotificationService (Servizio)

Per includere un pacchetto prerelease, includi `-Pre` argomento della riga di comando durante l'installazione del pacchetto come segue. :

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

È possibile trovare un elenco completo dei pacchetti di servizi AWS all'indirizzoConfigurazione dei pacchetti AWS SDK suNuGeto alSDK AWS per .NETGitHubRepository.

Opzione 2: Installa utilizzando il tuo IDE

In Visual Studio

1. Fare clic sul progetto, quindi fare clic suManage (Gestione)NuGetPacchetti.
2. Cerca il nome del pacchetto che vuoi aggiungere al tuo progetto. Per includere il preleasingNuGetpacchetti, scegliPreleasing di inclusione. È possibile trovare un elenco completo dei pacchetti di servizi AWS all'indirizzoConfigurazione dei pacchetti AWS SDK suNuGet.
3. Scegliere il pacchetto e quindi scegliere Install (Installa).

Mac (OS X)

In Visual Studio

1. Fai clic con il pulsante destro del mouse sulla cartella pacchetti, quindiAggiungi pacchetti.
2. Cerca il nome del pacchetto che vuoi aggiungere al tuo progetto. Per includere il preleasingNuGetpacchetti, scegliMostra pacchetti pre-release. È possibile trovare un elenco completo dei pacchetti di servizi AWS all'indirizzoConfigurazione dei pacchetti AWS SDK suNuGet.
3. Seleziona la casella di controllo accanto al pacchetto desiderato, quindi scegliAggiungi pacchetto.

> **Important**
>
> Se stai sviluppando utilizzando una libreria di classi portatili, devi aggiungere anche laAWSSDK.CoreNuGetpacchetto per tutti i progetti derivanti dalla Portable Class Library.

Fase 5: Configurazione dell'SDK AWS Mobile per.NET e Xamarin

Imposta la registrazione

È possibile impostare le impostazioni di registrazione utilizzando il`Amazon.AWSConfigs`classe e`Amazon.Util.LoggingConfig`classe. Sono disponibili nella sezione`AWSSdk.Core`assembly, disponibile tramite il gestore di pacchetti Nuget in Visual Studio. È possibile inserire il codice delle impostazioni di registrazione nel`OnCreate`metodo nel`MainActivity.cs`file per app Android o`AppDelegate.cs`file per app iOS. Ti consigliamo inoltre di aggiungere`using Amazon`e`using Amazon.Util`istruzioni per i file.cs.

Configurare le impostazioni di registrazione come indicato di seguito. :

```
var loggingConfig = AWSConfigs.LoggingConfig;
loggingConfig.LogMetrics = true;
loggingConfig.LogResponses = ResponseLoggingOption.Always;
loggingConfig.LogMetricsFormat = LogMetricsFormatOption.JSON;
loggingConfig.LogTo = LoggingOptions.SystemDiagnostics;
```

Quando accedi aSystemDiagnostics, il framework stampa internamente l'output su System.Console. Se si desidera registrare le risposte HTTP, impostare il`LogResponses`bandiera. I valori possono essere Sempre, Mai oOnError.

È inoltre possibile registrare le metriche delle prestazioni per le richieste HTTP utilizzando il`LogMetrics`proprietà. Il formato di log può essere specificato utilizzando`LogMetricsFormat`proprietà. I valori validi sono JSON o standard.

Impostazione dell'endpoint della regione

Configurazione dell'area predefinita per tutti i client di servizio come segue. :

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

Imposta l'area predefinita per tutti i client di servizio nell'SDK. È possibile ignorare questa impostazione specificando esplicitamente la regione al momento della creazione di un'istanza del client di servizio, come indicato di seguito. :

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

Configurazione delle impostazioni proxy HTTP

Se la rete è dietro un proxy, è possibile configurare le impostazioni proxy per le richieste HTTP come indicato di seguito.

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

Corretto per l'inclinazione dell'ora

Questa proprietà determina se l'SDK deve correggere l'inclinazione dell'orologio client determinando l'ora corretta del server e riemettendo la richiesta con l'ora corretta.

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

Questo campo viene impostato se una chiamata di servizio ha generato un'eccezione e l'SDK ha determinato che esiste una differenza tra l'ora locale e il server.

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

Per ulteriori informazioni su clock skew, consultaCorrezione dell'inclinazione dell'orologiosul Blog AWS.

Fasi successive

Ora che hai configurato l'SDK AWS Mobile per.NET e Xamarin, puoi:

- Nozioni di base su. LeggiNozioni di base su AWS SDK for .NET and Xamarinper istruzioni di avvio rapido su come utilizzare e configurare i servizi nell'SDK AWS Mobile per .NET e Xamarin.

- Esplora gli argomenti del servizio. Scopri di più su ogni servizio e come funziona nell'SDK AWS Mobile per.NET e Xamarin.
- Eseguire le demo. Visualizza il nostroApplicazioni Xamarin di esempioche dimostrano casi d'uso comuni. Per eseguire le app di esempio, configurare AWS Mobile SDK for .NET and Xamarin come descritto in precedenza, quindi seguire le istruzioni contenute nei file README dei singoli campioni.
- Scopri le API. Visualizzare il|sdk-xamarin-ref|_.
- Fai domande: Pubblica domande sulForum AWS Mobile SDKoapri un problema suGitHub.

Guida introduttiva all'SDK AWS Mobile per.NET e Xamarin

AWS Mobile SDK for .NET and Xamarin fornisce le librerie, i campioni e la documentazione necessaria per chiamare i servizi AWS dalle applicazioni Xamarin.

È necessario completare tutte le istruzioni riportate inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinprima di iniziare a utilizzare i servizi riportati di seguito.

Questi argomenti introdutti illustreranno le seguenti operazioni:

Argomenti

- Archivia e recupera file con Amazon S3
- Sincronizza dati utente con Cognito Sync
- Archivia e recupera dati con DynamoDB
- Gestione dei dati sull'utilizzo delle app con Amazon Mobile Analytics
- Ricevi notifiche push con SNS (Xamarin iOS)
- Ricevi notifiche push con SNS (Xamarin Android)

Per informazioni su altri SDK AWS Mobile, vediSDK AWS Mobile.

Archivia e recupera file con Amazon S3

Amazon Simple Storage Service (Amazon S3) offre agli sviluppatori di dispositivi mobili uno storage di oggetti sicuro, durevole e altamente scalabile. Amazon S3 è di facile utilizzo e include un'interfaccia Web service intuitiva che consente di archiviare e recuperare qualsiasi quantità di dati ovunque nel Web.

Il tutorial che segue illustra come integrare l'S3TransferUtility, un'utilità di alto livello per l'utilizzo di S3 con la tua app. Per ulteriori informazioni sull'utilizzo di S3 dalle applicazioni Xamarin, consultaAmazon Simple Storage Service (S3).

Configurazione progetto

Prerequisiti

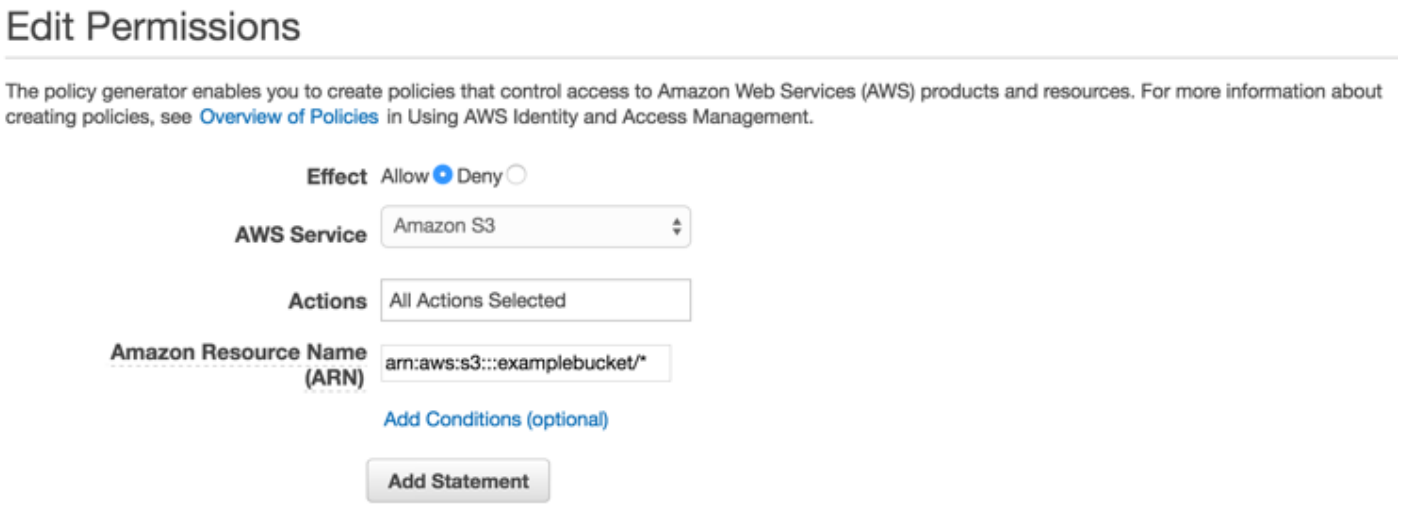
È necessario compilare tutte le istruzioni riportate nelConfigurazione dell'SDK AWS Mobile per.NET e Xamarinprima di iniziare questo tutorial.

Questo tutorial presuppone inoltre che tu abbia già creato un bucket S3. Per creare un bucket S3, visita ilConsole AWS S3.

Impostazione delle autorizzazioni per S3

La policy del ruolo IAM predefinita concede all'applicazione l'accesso ad Amazon Mobile Analytics e Amazon Cognito Sync. Affinché il pool di identità Cognito acceda ad Amazon S3, devi modificare i ruoli del pool di identità.

1. Accedi aIdentity and Access Management Consolee clicca suRuolinel riquadro sinistro.
2. Digitare il nome del pool di identità nella casella di ricerca. Vengono elencati due ruoli, relativi, rispettivamente, agli utenti autenticati e non.
3. Fare clic sul ruolo degli utenti non autenticati (non verrà aggiunto l'autenticazione al nome del pool di identità).
4. Fare clic suCreazione di policy di ruolo, selezionaGeneratore di policye quindi fai clic suSeleziona.
5. SulModifica delle autorizzazioniinserisci le impostazioni mostrate nell'immagine seguente, sostituendo l'Amazon Resource Name (ARN) con il tuo. L'ARN di un bucket S3 ha l'aspetto di`arn:aws:s3:::examplebucket/*`ed è composto dalla regione in cui si trova il bucket e dal nome del bucket. Le impostazioni mostrate di seguito daranno al tuo pool di identità pieno l'accesso a tutte le azioni per il bucket specificato.



1. Fai clic sull'icona della barra degli strumentiAggiungi istruzionepulsante e poi fai clicFase successiva.
2. La procedura guidata mostrerà la configurazione generata. Fare clic suApplica policy.

Per ulteriori informazioni sulla concessione dell'accesso a S3, consultaConcessione dell'accesso a un bucket Amazon S3.

InserisciNuGetPacchetto per S3 per il tuo progetto

Seguire il passaggio 4 delle istruzioni riportate inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper aggiungere l'S3NuGetConfigurazione del progetto.

(facoltativo) Configurare la versione Signature per le richieste S3

Ogni interazione con Amazon S3 è autenticata o anonima. AWS utilizza gli algoritmi Signature Version 4 o Signature Version 2 per autenticare le chiamate al servizio.

Tutte le nuove regioni AWS create dopo gennaio 2014 supportano solo Signature Version 4. Tuttavia, molte regioni precedenti supportano ancora le richieste Signature Version 4 e Signature Version 2.

Se il bucket si trova in una delle regioni che non supportano le richieste Signature Version 2 come elencato suquesta pagina, devi impostare ilAWSConfigsS3.UseSignatureproprietà Version4 su «true» in questo modo:

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

Per ulteriori informazioni sulle versioni di AWS Signature, consultaAutenticazione delle richieste (AWS Signature Version 4).

Inizializza l'S3TransferUtilityClient

Crea un client S3, passandolo al tuo oggetto credenziali AWS, quindi passa il client S3 all'utilità di trasferimento, in questo modo:

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

Caricamento di un file su Amazon S3

Per caricare un file su S3, chiama`Upload`sull'oggetto Transfer Utility, passando i seguenti parametri:

- `file`- Nome stringa del file che desideri caricare

- `bucketName`- Nome stringa del bucket S3 per archiviare il file

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

Il codice sopra presuppone che ci sia un file nella directory Environment.SpecialFolder.ApplicationData. I caricamenti utilizzano automaticamente la funzionalità di caricamento multiparte di S3 su file di grandi dimensioni per migliorare il throughput.

Scaricare un file da Amazon S3

Per scaricare un file da S3, chiama`Download`sull'oggetto Transfer Utility, passando i seguenti parametri:

- `file`- Nome stringa del file che desideri scaricare
- `bucketName`- Nome stringa del bucket S3 da cui scaricare il file
- `key`- Una stringa che rappresenta il nome dell'oggetto S3 (in questo caso un file) da scaricare

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

Per ulteriori informazioni sull'accesso ad Amazon S3 da un'applicazione Xamarin, consultaAmazon Simple Storage Service (S3).

Sincronizza dati utente con Cognito Sync

Amazon Cognito Sync agevola il salvataggio di dati utente dei dispositivi mobili, come preferenze delle app o stato dei giochi nel cloud AWS senza scrivere codice di back-end o gestire infrastrutture. Puoi salvare i dati localmente sui dispositivi degli utenti, affinché le applicazioni funzionino anche se i dispositivi sono offline. L'utente potrà inoltre sincronizzare i dati tra i diversi dispositivi, in modo da mantenere la stessa esperienza sull'applicazione, indipendentemente dal tipo di dispositivo utilizzato.

Il tutorial che segue illustra come integrare Synchronics con l'app.

Configurazione progetto

Prerequisiti

È necessario seguire le istruzioni riportate alConfigurazione dell'SDK AWS Mobile per.NET e Xamarinprima di iniziare questo tutorial.

Concedere l'accesso alle risorse di Cognito Sync

I criteri predefiniti associati ai ruoli non autenticati e autenticati creati durante l'installazione garantiscono all'applicazione l'accesso a Cognito Sync. Non è richiesta alcuna configurazione aggiuntiva.

InserisciNuGetPacchetto per Cognito Sync al tuo progetto

Seguire il passaggio 4 delle istruzioni riportate inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper aggiungere il CognitoSyncManager NuGetpacchetto per il progetto.

InizializzazioneCognitoSyncManager

Bisogna passare il provider di credenziali Amazon Cognito inizializzato al costruttore `CognitoSyncManager`:

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

Sincronizzazione dei dati utente

Per sincronizzare i dati utente non autenticati:

1. Creazione di un set di dati
2. Aggiungere i dati utente al set di dati.
3. Sincronizza il set di dati con il cloud.

Creare un set di dati

Creare un'istanza di `Dataset`. LaopenOrCreateIl metodo del set di dati viene utilizzato per creare un nuovo set di dati o aprire un'istanza esistente di un set di dati memorizzato localmente sul dispositivo:

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

Aggiunta di dati utente al set di dati

I dati utente vengono aggiunti sotto forma di coppie chiave/valore:

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

Funzione di set di dati di Cognito come dizionari, con i valori accessibili tramite chiave:

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

Sincronizzazione set di dati

Per sincronizzare il set di dati, chiama il suo metodo di sincronizzazione:

```
dataset.SynchronizeAsync();

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

Tutti i dati scritti sui set di dati verranno memorizzati localmente fino alla sincronizzazione del set di dati. Il codice in questa sezione presuppone che tu stia utilizzando un'identità Cognito non autenticata, quindi quando i dati dell'utente vengono sincronizzati con il cloud verranno memorizzati per dispositivo. Al dispositivo è associato un ID dispositivo. Quando i dati utente vengono sincronizzati con il cloud, verranno associati all'ID del dispositivo.

Per ulteriori informazioni su Cognito Sync, vediAmazon Cognito Sync.

Archivia e recupera dati con DynamoDB

Amazon DynamoDB è un servizio di database non relazionale, conveniente, veloce e altamente scalabile e disponibile. DynamoDB rimuove le tradizionali limitazioni di scalabilità dello storage dei dati, mantenendo una bassa latenza e prestazioni prevedibili.

L'esercitazione seguente illustra come integrare DynamoDB Object Persistence Model con l'app, che memorizza gli oggetti in DynamoDB.

Configurazione progetto

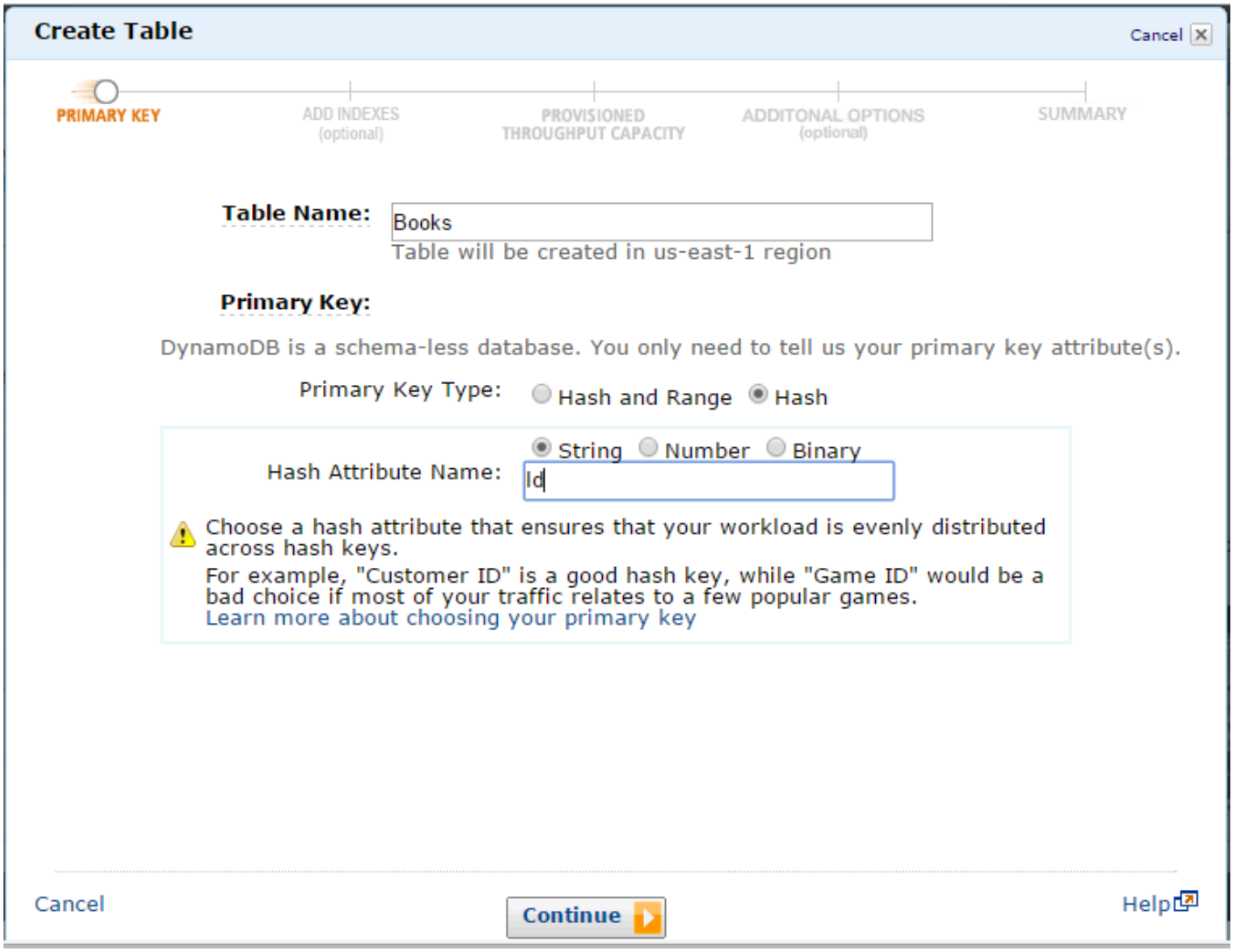
Prerequisiti

È necessario seguire le istruzioni riportate alla barra degli strumentiConfigurazione dell'SDK AWS Mobile per.NET e Xamarinprima di iniziare questo tutorial.

Creazione di una tabella DynamoDB

Prima di poter leggere e scrivere dati su un database DynamoDB, devi creare una tabella. Quando si crea una tabella, è necessario specificare la chiave primaria. La chiave primaria è costituita da un attributo hash e da un attributo range facoltativo. Per ulteriori informazioni su come vengono utilizzati gli attributi primari e dell'intervallo, consulta.Utilizzo delle tabelle.

1. Passa alla barra degli strumentiConsole DynamoDBe clicca suCreare tabella. Viene visualizzata la procedura guidata Crea tabella.
2. Specificare il nome della tabella, il tipo di chiave primaria (Hash) e il nome dell'attributo hash («Id») come mostrato di seguito, quindi fare clic suContinua:

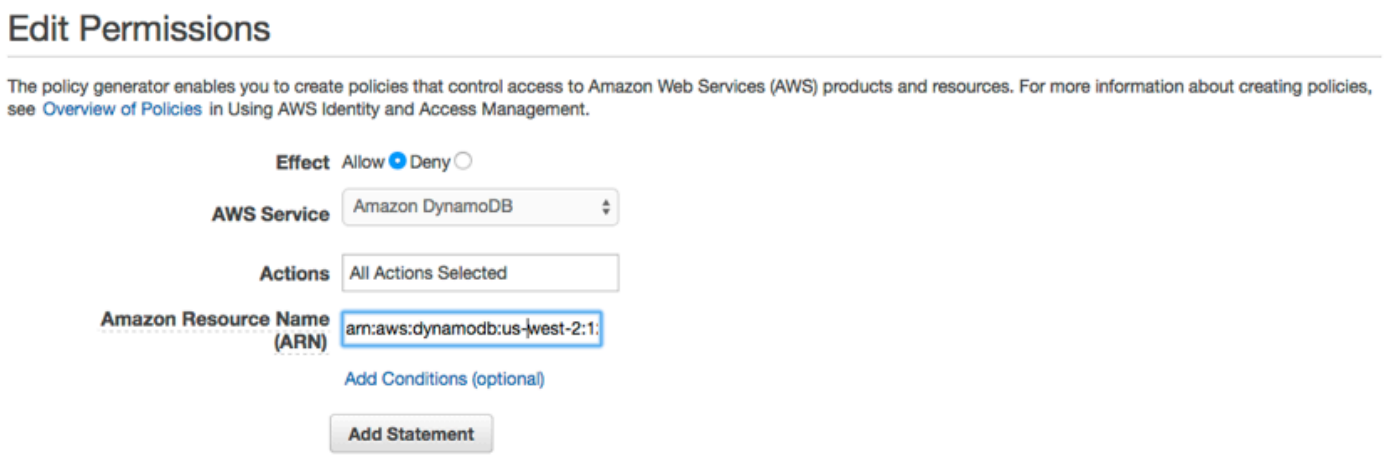


3. Lasciare vuoti i campi di modifica nella schermata successiva e fai clic suContinua.
4. Accettare i valori predefiniti perunità di capacità in letturaeUnità di capacità in scritturae clicca suContinua.
5. Nella schermata successiva, inserisci il tuo indirizzo email nellInvia notifica a:casella di testo e clickContinua. Viene visualizzata la schermata di revisione.
6. Fai clic su Create (Crea). Per creare il tavolo potrebbero essere necessari alcuni minuti.

Impostazione delle autorizzazioni per DynamoDB

Affinché il tuo pool di identità acceda ad Amazon DynamoDB, devi modificare i ruoli del pool di identità.

1. Passare alla .Identity and Access Management Consolee clicca suRuoliNel riquadro sinistro. Cerca il nome del pool di identità — verranno elencati due ruoli, uno per gli utenti non autenticati e uno per gli utenti autenticati.
2. Fare clic sul ruolo degli utenti non autenticati (verrà aggiunto «unauth» al nome del pool di identità) e fai clic suCreate policy per i ruoli.
3. Selezionageneratore di policye clicca suSeleziona.
4. SulModifica delle autorizzazioni, inserisci le impostazioni mostrate nell'immagine seguente. L'Amazon Resource Name (ARN) di una tabella DynamoDB ha l'aspettoarn:aws:dynamodb:us-west-2:123456789012:table/Booksed è composto dalla regione in cui si trova la tabella, dal numero di account AWS del proprietario e dal nome della tabella nel formatotable/Books. Per ulteriori informazioni su come specificare gli ARN, consulta.Amazon Resource Names for DynamoDB.



5. Fare clic suAggiungi istruzione, quindi fai clic suFase successiva. La procedura guidata mostrerà la configurazione generata.
6. Fare clic suApplica policy.

InserisciNuGetpacchetto per DynamoDB to Your Project

Seguire il passaggio 4 delle istruzioni riportate inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper aggiungere DynamoDBNuGetConfigurazione del progetto.

InizializzazioneAmazonDynamoDBClient

Passa il provider di credenziali Amazon Cognito inizializzato e la tua regione allaAmazonDynamoDBcostruttore, quindi passa il client al DynamoDBContext:

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

Creazione di una classe

Per scrivere una riga nella tabella, definire una classe per contenere i dati della riga. La classe deve inoltre contenere proprietà che contengono i dati dell'attributo per la riga e verrà mappata alla tabella DynamoDB creata nella console. La seguente dichiarazione di classe illustra tale classe:

```
[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; }
}
```

Salvataggio di una voce

Per salvare un elemento, creare innanzitutto un oggetto:

```
Book songOfIceAndFire = new Book()
{
    Id=1,
    Title="Game Of Thrones",
    ISBN="978-0553593716",
    Price=4,
    PageCount="819",
    Author="GRRM"
};
```

Quindi salvalo:

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

Per aggiornare una riga, modificare l'istanza del`DDTableRow`classe e call`AWSDynamoObjectMapper.save()`Come illustrato sopra.

Recupero di una voce

Recupera un elemento utilizzando una chiave primaria:

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

Aggiornamento di una voce

Per aggiornare una voce:

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

Eliminazione di una voce

Per eliminare una voce:

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

Per ulteriori informazioni sull'accesso a DynamoDB da un'applicazione Xamarin, vedereAmazon DynamoDB.

Gestione dei dati sull'utilizzo delle app con Amazon Mobile Analytics

Amazon Mobile Analytics ti consente di misurare l'utilizzo delle app e le entrate delle app. Tracciando le tendenze chiave come utenti nuovi e restituiti, ricavi delle app, fidelizzazione degli utenti ed eventi personalizzati di comportamento in-app, puoi prendere decisioni basate sui dati per aumentare il coinvolgimento e la monetizzazione per la tua app.

Il tutorial che segue illustra come integrare Mobile Analytics con l'app.

Configurazione progetto

Prerequisiti

Configurazione di tutte le istruzioni riportate allaConfigurazione dell'SDK AWS Mobile per.NET e Xamarinprima di iniziare questo tutorial.

Crea un'app nella console di Mobile Analytics

Accedi aConsole Amazon Mobile AnalyticsCreare un'app. Nota:`appId`valore, come ne avrai bisogno in un secondo momento. Quando crei un'app in Mobile Analytics Console, dovrai specificare l'ID del pool di identità. Per istruzioni su come creare un pool di identità, consulta la paginaConfigurazione dell'SDK AWS Mobile per.NET e Xamarin.

Per ulteriori informazioni sulle operazioni con la console, consulta laGuida per l'utente di Amazon Mobile Analytics.

Impostazione delle autorizzazioni per Mobile Analytics

Il criterio predefinito associato ai ruoli creati durante l'installazione garantisce all'applicazione l'accesso a Mobile Analytics. Non è richiesta alcuna configurazione aggiuntiva.

InserisciNuGetPacchetto per Mobile Analytics per il tuo progetto

Seguire il passaggio 4 delle istruzioni riportate inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper aggiungere Mobile AnalyticsNuGetPacchetto per il progetto.

Configurazione Mobile Analytics

Mobile Analytics definisce alcune impostazioni che possono essere configurate nel file awsconfig.xml:

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

- AllowUseDataNetwork- Un valore booleano che specifica se gli eventi di sessione vengono inviati sulla rete dati.

- DBWarningThreshold- Questo è il limite alle dimensioni del database che, una volta raggiunto, genererà registri di avviso.
- MaxDBSize - Questa è la dimensione del database SQLite. Quando il database raggiunge la dimensione massima, vengono eliminati eventuali eventi aggiuntivi.
- MaxRequestDimensione - Questa è la dimensione massima della richiesta in byte che deve essere trasmessa in una richiesta HTTP al servizio di analisi mobile.
- SessionTimeout- Questo l'intervallo di tempo dopo che un'applicazione passa in background e quando la sessione può essere terminata.

Le impostazioni mostrate sopra sono i valori predefiniti per ogni elemento di configurazione.

InizializzazioneMobileAnalyticsManager

Inizializzazione diMobileAnalyticsManager, chiamaGetOrCreateInstancesul tuo`MobileAnalyticsManager`, passando le credenziali AWS, la tua regione, l'ID dell'applicazione Mobile Analytics e l'oggetto di configurazione opzionale:

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

Gestione di eventi di traccia

Android Xamarin

Sovrascrivi l'attività`OnPause()`e`OnResume()`metodi per registrare eventi di sessione.

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

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

```
    base.OnPause();
}
```

Questo deve essere implementato per ogni attività della tua applicazione.

Xamarin iOS

Nel tuoAppDelegate.cs:

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

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

Per ulteriori informazioni su Mobile Analytics, consultaAmazon Mobile Analytics.

Ricevi notifiche push con SNS (Xamarin iOS)

Questo documento spiega come inviare notifiche push a un'applicazione Xamarin iOS utilizzando Amazon Simple Notification Service (SNS) e AWS Mobile SDK for .NET and Xamarin.

Configurazione progetto

Prerequisiti

È necessario seguire le istruzioni riportate allaConfigurazione dell'SDK AWS Mobile per.NET e Xamarinprima di iniziare questo tutorial.

Imposta autorizzazioni per SNS

Segui il passaggio 2 inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper allegare la politica menzionata di seguito ai ruoli della tua candidatura. Ciò darà alla tua applicazione le autorizzazioni appropriate per accedere a SNS:

1. Passa allaConsole IAMe seleziona il ruolo IAM da configurare.

2. Fare clic suCollegamento della policy, seleziona AmazonSNSFullAccesspolicy e cliccaCollegamento della policy.

> **Warning**
>
> Utilizzo di AmazonSNSFullAccessNon è raccomandato in un ambiente di produzione. Lo usiamo qui per permetterti di iniziare a usare rapidamente. Per ulteriori informazioni sulla definizione delle autorizzazioni per un ruolo IAM, consultaPanoramica delle autorizzazioni del ruolo IAM.

Ottieni l'iscrizione al programma Apple iOS Developer

Dovrai eseguire l'app su un dispositivo fisico per ricevere notifiche push. Per eseguire la tua app su un dispositivo, devi avere un abbonamento allIscrizione al programma per sviluppatori Apple iOS. Quando disponi di un abbonamento, puoi utilizzare Xcode per generare un'identità di firma. Per ulteriori informazioni, consulta AppleQuick Start Distributiondocumentazione.

Crea un certificato iOS

In primo luogo, devi creare un certificato iOS. Quindi, è necessario creare un profilo di provisioning configurato per le notifiche push. A questo proposito:

1. Passa allaCentro membri Apple Developer, fare clic suCertificati, identificatori e profili.
2. Fare clic suIdentificatori disottoApp iOS, fai clic sul pulsante più nell'angolo in alto a destra della pagina Web per aggiungere un nuovo ID app iOS e inserisci una descrizione dell'ID app.
3. Scorrere fino alla barra degli strumentiAggiungi suffisso IDsezione e selezionaID app esplicitoe inserisci il tuo identificatore del bundle.
4. Scorrere fino alla barra degli strumentiApp Servicessezione e selezionaNotifiche push.
5. Fai clic su Continue (Continua).
6. Fare clic su Submit (Invia).
7. Fare clic suFatto.
8. Seleziona l'ID app appena creato e fai clic suModificare.
9. Scorrere fino alla barra degli strumentiNotifiche pushsezione. Fare clic suCreazione di certificatosottoSviluppo del certificato SSL.

10. Segui le istruzioni per creare una richiesta di firma del certificato (CSR), carica la richiesta e scarica un certificato SSL che verrà utilizzato per comunicare con Apple Notification Service (APNS).
11. Torna allaCertificati, identificatori e profili(Certificato creato). Fare clic suAll (Tutti)sottoProfili di provisioning.
12. Fare clic sul pulsante più nell'angolo in alto a destra per aggiungere un nuovo profilo di provisioning.
13. SelezionaSviluppo di app iOS, quindi fai clic suContinua.
14. Selezionare l'ID dell'app, quindi fare clic suContinua.
15. Selezionare il certificato dello sviluppatore, quindi fare clic suContinua.
16. Selezionare il dispositivo, quindi fare clic suContinua.
17. Immettere un nome del profilo, quindi fare clic suGenera.
18. Scarica e fai doppio clic sul file di provisioning per installare il profilo di provisioning.

Per ulteriori informazioni sul provisioning di un profilo configurato per le notifiche push, consultaConfigurazione delle notifiche pushdocumentazione.

Utilizzare il certificato per creare ARN della piattaforma in SNS Console

1. EseguireKeyChainaccedi all'app, selezionaI miei certificatinella parte inferiore sinistra dello schermo, quindi fare clic con il pulsante destro del mouse sul certificato SSL generato per connettersi a APNS e selezionareEsportazione. Ti verrà richiesto di specificare un nome per il file e una password per proteggere il certificato. Il certificato verrà salvato in un file P12.
2. Passa allaConsole SNSe fai clic suApplicazionisul lato sinistro dello schermo.
3. Fare clic suCreazione di applicazioni per piattaformeCreazione di una nuova applicazione di piattaforma SNS.
4. Immettere unNome applicazione.
5. SelezionaSviluppo AppleperPiattaforma di notifiche push.
6. Fare clic suSelezionare Filee seleziona il file P12 creato quando hai esportato il certificato SSL.
7. Immetti la password specificata all'esportazione del certificato SSL e fai clic suCarica le credenziali da file.
8. Fare clic suCreazione di applicazioni per piattaforme.
9. Seleziona l'applicazione Platform appena creata e copia l'ARN dell'applicazione. Questo ARN sarà necessario nei prossimi passaggi.

InserisciNuGetPacchetto per SNS per il tuo progetto

Seguire il passaggio 4 delle istruzioni riportate in[Configurazione dell'SDK AWS Mobile per.NET e Xamarin](#)per aggiungere Amazon Simple Notification ServiceNuGetpacchetto per il tuo progetto.

Creazione di un client SNS

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

Registra la tua applicazione per le notifiche remote

Per registrare una domanda, chiamaRegisterForRemoteNotificationssul tuo oggetto UIApplication, come mostrato di seguito. Inserire il seguente codice inAppDelegate.cs, inserendo l'ARN dell'applicazione della piattaforma dove richiesto di seguito:

```
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 */
    });
  }
```

```
}
```

Invia un messaggio dalla console SNS al tuo endpoint

1. Passa allaSNS Console > Applicazioni.
2. Seleziona l'applicazione della piattaforma, seleziona un endpoint e fai clic suPubblicazione su endpoint.
3. Digita un messaggio di testo nella casella di testo e fai clic suPubblicazione di messaggiPer pubblicare un messaggio.

Ricevi notifiche push con SNS (Xamarin Android)

Il tutorial spiega come inviare notifiche push a un'applicazione Android Xamarin utilizzando Amazon Simple Notification Service (SNS) e AWS Mobile SDK for .NET and Xamarin.

Configurazione progetto

Prerequisiti

È necessario seguire le istruzioni riportate alConfigurazione dell'SDK AWS Mobile per.NET e Xamarinprima di iniziare questo tutorial.

Impostazione delle autorizzazioni per SNS

Segui il passaggio 2 inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper allegare la politica menzionata di seguito ai ruoli della tua candidatura. Ciò darà alla tua applicazione le autorizzazioni appropriate per accedere a SNS:

1. Passa alConsole IAMe seleziona il ruolo IAM da configurare.
2. Fare clic suCollegamento della policy, seleziona AmazonSNSFullAccesspolicy e cliccaCollegamento della policy.

> **Warning**
>
> Usare AmazonSNSFullAccessnon è raccomandato in un ambiente di produzione. Lo usiamo qui per permetterti di iniziare a usare rapidamente. Per ulteriori informazioni sulla definizione delle autorizzazioni per un ruolo IAM, consultaPanoramica delle autorizzazioni del ruolo IAM.

Abilita notifiche push su Google Cloud

Innanzitutto, aggiungi un nuovo progetto Google API:

1. Passa alConsole per sviluppatori di Google.
2. Fare clic suCreazione di un progetto.
3. NellaNuovo progettocasella, inserisci il nome di un progetto, prendi nota dell'ID del progetto (ne avrai bisogno in seguito) e fai clic suCreate.

Successivamente, abilita il servizio Google Cloud Messaging (GCM) per il progetto:

1. NellaConsole per sviluppatori di Google, il nuovo progetto dovrebbe già essere selezionato. In caso contrario, selezionalo nel menu a discesa nella parte superiore della pagina.
2. SelezionaAutenticazione delle API e dell'autenticazionedalla barra laterale sul lato sinistro della pagina.
3. Nella casella di ricerca, digita «Google Cloud Messaging for Android» e fai clic suGoogle Cloud Messaging per Androidcollegamento.
4. Fare clic suAbilitazione dell'API.

Infine, ottieni una chiave API:

1. Nella Google Developers Console, selezionaAutenticazione delle API e dell'autenticazione >Credenziali.
2. UNDERAccesso all'API pubblica, clickCreazione di una nuova chiave.
3. NellaCreazione di una nuova chiavefinestra di dialogo, fai clicChiave server.
4. Nella finestra di dialogo risultante, fai clic suCreatee copia la chiave API visualizzata. Utilizzerai questa chiave API per eseguire l'autenticazione in un secondo momento.

Utilizzare l'ID progetto per creare un ARN della piattaforma in SNS Console

1. Passa alConsole SNS.
2. Fare clic suApplicazionisul lato sinistro dello schermo.
3. Fare clic suCreazione di applicazioni piattaformaper creare una nuova applicazione di piattaforma SNS.
4. Specificare unNome applicazione.

5. SelezionaGoogle Cloud Messaging (GCM)perPiattaforma di notifiche push.
6. Incolla la chiave dell'API nella casella di testo con etichettaChiave API.
7. Fare clic suCreazione di applicazioni piattaforma.
8. Seleziona l'applicazione Platform appena creata e copia l'ARN dell'applicazione.

InserisciNuGetPacchetto per SNS per il tuo progetto

Seguire il passaggio 4 delle istruzioni riportate inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper aggiungere Amazon Simple Notification ServiceNuGetpacchetto per il tuo progetto.

Creazione di un client SNS

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

Registra la tua applicazione per le notifiche remote

Per registrarsi alle notifiche remote su Android, è necessario creare unBroadcastReceiverche può ricevere messaggi Google Cloud. Modificare il nome del pacchetto qui sotto dove richiesto:

```
[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);
      }
}
```

Di seguito è riportato il servizio che riceve la notifica push dalBroadcastReceivere visualizza la notifica sulla barra delle notifiche del dispositivo:

```
[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());
  }
}
```

Invia un messaggio dalla console SNS al tuo endpoint

1. Passa alSNS Console > Applicazioni.
2. Seleziona l'applicazione della piattaforma, seleziona un endpoint e fai clic suPubblica su endpoint.
3. Digita un messaggio di testo nella casella di testo e fai clic suPubblica di messaggiper pubblicare un messaggio.

Amazon Cognito Identity

Che cos'è Amazon Cognito?

Amazon Cognito Identity ti permette di creare identità univoche per i tuoi utenti e autenticarli con provider di identità. Con un'identità, puoi ottenere delle credenziali AWS temporanee con privilegi limitati per sincronizzare i dati con Amazon Cognito Sync o accedere direttamente ad altri servizi AWS. Amazon Cognito Identity supporta provider di identità pubbliche: Amazon, Facebook e Google, nonché identità non autenticate. Questa funzione supporta anche identità non autenticate per gli sviluppatori e ti consente di registrare e di autenticare gli utenti tramite il tuo processo di autenticazione di back-end.

Per ulteriori informazioni su Cognito Identity, consultare ilGuida per sviluppatori di Amazon Cognito.

Per informazioni sulla disponibilità di una regione di autenticazione di Cognito, consultareDisponibilità di una regione AWS Service.

Utilizzo di un provider pubblico per autenticare gli utenti

Utilizzando Amazon Cognito Identity, puoi creare identità univoche per i tuoi utenti e autenticarli per accedere in modo sicuro alle risorse AWS come Amazon S3 o Amazon DynamoDB. Amazon Cognito Identity supporta provider di identità pubbliche: Amazon, Facebook, Twitter/Digits, Google o qualsiasi provider compatibile con OpenID Connect, nonché identità non autenticate.

Per informazioni sull'utilizzo di provider di identità pubblica come Amazon, Facebook, Twitter/Digits o Google per autenticare gli utenti, consulta la sezioneProvider esterninella Guida per sviluppatori di Amazon Cognito.

Utilizzo delle identità autenticate dagli sviluppatori

Amazon Cognito supporta identità autenticate dagli sviluppatori oltre alla federazione delle identità Web tramite Facebook, Google e Amazon. Grazie alle identità autenticate dagli sviluppatori, puoi registrare e autenticare gli utenti tramite il tuo processo di autenticazione esistente, continuando a utilizzareAmazon Cognito Syncper sincronizzare i dati utente e accedere alle risorse AWS. L'utilizzo di identità autenticate dagli sviluppatori prevede l'interazione tra il dispositivo dell'utente finale, il back-end per l'autenticazione e Amazon Cognito.

Per informazioni sulle identità autenticate dagli sviluppatori, consulta la sezioneIdentità autenticate tramite gli sviluppatorinella Guida per sviluppatori di Amazon Cognito.

Amazon Cognito Sync

Che cos'è Amazon Cognito Sync?

Cognito Sync è un servizio AWS e una libreria client che consente la sincronizzazione tra più dispositivi di dati dell'utente (ad esempio punteggi di gioco, preferenze dell'utente, stato del gioco). È possibile utilizzare l'API Cognito Sync per sincronizzare i dati utente tra i dispositivi. Per utilizzare Cognito Sync nella tua app, devi includere | nel progetto.

Per istruzioni su come integrare Amazon Cognito Sync nella tua applicazione, consultaAmazon Cognito Sync.

Amazon Mobile Analytics

Amazon Mobile Analyticsè un servizio per raccogliere, visualizzare, conoscere ed estrarre dati sull'utilizzo di app su larga scala. Mobile Analytics acquisisce facilmente sia i dati dei dispositivi standard sia gli eventi personalizzati ed elabora i report per te automaticamente. Oltre ai report aggregati elencati di seguito, è anche possibile configurare i dati affinché siano esportati automaticamente a Redshift e S3 per un'ulteriore analisi.

Utilizzando Amazon Mobile Analytics, puoi monitorare i comportamenti dei clienti, aggregare metriche, generare visualizzazioni di dati e identificare modelli significativi.

Concetti chiave

Tipi di report

Mobile Analytics fornisce i seguenti report in Mobile Analytics Console:

- Daily Active Users (DAU, Utenti attivi giornalieri), Monthly Active Users (MAU, Utenti attivi mensili) e New Users (Nuovi utenti)
- Sticky Factor (Fattore sticky) (DAU diviso MAU)
- Session Count (Conteggio sessioni) e Average Sessions per Daily Active User (Sessioni medie per utente attivo giornaliero)
- Average Revenue per Daily Active User (ARPPDAU, Fatturato medio per utente attivo giornaliero a pagamento).
- Day 1, 3, and 7 Retention (Conservazione giorno 1, 3 e 7) e Week 1, 2, and 3 Retention (Conservazione settimana 1, 2 e 3)
- Eventi personalizzati

Questi report vengono forniti tramite sei schede di report nella console:

- Panoramica— Tieni traccia di nove report preselezionati in unsimple-to-reviewdashboard per avere una rapida idea di coinvolgimento: MAU, DAU, Nuovi utenti, Sessioni giornaliere, Sticky Factor, 1 giorno di conservazione, ARPDAU, Daily Paying Users (Utenti giornalieri a pagamento), ARPPDAU.
- Utenti attiviMonitora il numero degli utenti che interagiscono con la tua app giornalmente e mensilmente e monitora il fattore adesivo per valutare coinvolgimento, interesse e monetizzazione.

- Sessioni— Monitora la frequenza con cui la tua app viene utilizzata in una specifica data e quella con cui i singoli utenti aprono l'app in una determinata giornata.
- RetentionMonitora la frequenza con cui i clienti tornano a utilizzare la tua app su base giornaliera e settimanale.
- Revenue (Fatturato)Monitora le tendenze di fatturato in-app per identificare le aree di miglioramento dal punto di vista della monet
- Eventi personalizzati— Tiene traccia delle azioni personalizzate definite dall'utente specifiche dell'app.

Per ulteriori informazioni sui report di Mobile Analytics e sul lavoro nella console di Mobile Analytics, consulta la sezionePanoramica dei report della console Mobile Analyticsnella Guida per gli sviluppatori di Mobile Analytics.

Configurazione progetto

Prerequisiti

Per utilizzare Mobile Analytics nella tua applicazione, devi aggiungere l'SDK al tuo progetto. A tale scopo, segui le istruzioni riportate diConfigurazione dell'SDK AWS Mobile per.NET e Xamarin.

Configurazione Mobile Analytics

Mobile Analytics definisce alcune impostazioni che possono essere configurate nel file awsconfig.xml:

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

- SessionTimeout- Se l'app rimane in background per un tempo superiore alSessionTimeoutquindi il client Mobile Analytics termina la sessione corrente e viene creata una nuova sessione quando l'app torna in primo piano. Raccomandiamo di utilizzare valori compresi tra 5 e 10. Il valore predefinito è 5.
- MaxDBSize- La dimensione massima del database (in byte) utilizzata per l'archiviazione locale degli eventi. Se la dimensione del database supera questo valore, gli eventi aggiuntivi verranno

ignorati. Si consiglia di utilizzare valori che vanno da 1 MB a 10 MB. Il valore predefinito è 5242880 (5 MB).

- DBWarningThreashold- La soglia di avviso. I valori validi variano tra 0 e 1. Se i valori superano la soglia, verranno generati i registri di avviso. Il valore predefinito è 0,9.
- MaxRequestDimensioni- La dimensione massima di una richiesta HTTP inviata al servizio Mobile Analytics. Il valore è specificato in byte e può variare tra 1-512 KB. Il valore predefinito è 102400 (100 KB). Non utilizzare valori superiori a 512 KB, questo potrebbe causare il rifiuto della richiesta HTTP.
- AllowUseDataNetwork- Un valore che indica se la chiamata di servizio è consentita su una rete dati cellulare. Utilizzare questa opzione con cautela in quanto ciò potrebbe aumentare l'utilizzo dei dati del cliente.

Le impostazioni mostrate sopra sono i valori predefiniti per ogni elemento di configurazione.

Integrazione di Mobile Analytics con l'applicazione

Le sezioni seguenti spiegano come integrare Mobile Analytics con la tua app.

Crea un'app nella console di Mobile Analytics

Accedi aConsole Amazon Mobile Analyticse crea un'app. Nota:`appId`come è necessario in seguito. Quando crei un'app in Mobile Analytics Console, dovrai specificare l'ID del pool di identità. Per istruzioni su come creare un pool di identità, consultaConfigurazione dell'SDK AWS Mobile per.NET e Xamarin.

Per ulteriori informazioni sulle operazioni con Mobile Analytics Console, consulta laPanoramica dei report della console Mobile Analyticsnella Guida per gli sviluppatori di Mobile Analytics.

Creazione di unMobileAnalyticsClient manager

Inizializzazione diMobileAnalyticsManager, chiamaGetOrCreateInstancesul tuo`MobileAnalyticsManager`, passando le credenziali AWS, la tua regione, l'ID dell'applicazione Mobile Analytics e l'oggetto di configurazione opzionale:

```
// Initialize the MobileAnalyticsManager
analyticsManager = MobileAnalyticsManager.GetOrCreateInstance(
  cognitoCredentials,
  RegionEndpoint.USEast1,
```

```
  APP_ID,
  config
);
```

LaAPP_IDviene generato per te durante la procedura guidata di creazione dell'app. Entrambi questi valori devono corrispondere a quelli di Mobile Analytics Console. LaAPP_IDviene utilizzato per raggruppare i dati nella console di Mobile Analytics. Per trovare il tuo ID app dopo aver creato l'app nella console di Mobile Analytics, accedi a Mobile Analytics Console, fai clic sull'icona a forma di ingranaggio nell'angolo in alto a destra dello schermo. Viene visualizzata la pagina Gestione app che elenca tutte le app registrate e i relativi ID app.

Registrazione di eventi di monetizzazione

L'SDK AWS Mobile per.NET e Xamarin fornisce la`MonetizationEvent`class, che consente di generare eventi di monetizzazione per monitorare gli acquisti effettuati all'interno di applicazioni mobili. Il seguente frammento di codice mostra come creare un evento di monetizzazione:

```
// 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);
```

Registrazione di eventi personalizzati

Mobile Analytics consente di definire eventi personalizzati. Gli eventi personalizzati sono definiti interamente da te e ti aiutano a monitorare le azioni degli utenti specifiche della tua app o del tuo gioco. Per ulteriori informazioni sugli eventi personalizzati, consultaEventi personalizzati.

Per questo esempio, diremo che la nostra app è un gioco e che vogliamo registrare un evento quando un utente completa un livello. Crea un»LevelComplete» evento creando un nuovo`AmazonMobileAnalyticsEvent`istanza:

```
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);
```

Sessioni di registrazione

Xamarin iOS

Quando l'applicazione perde lo stato attivo è possibile sospendere la sessione. Per le app iOS, nelAppDelegatefile.cs, sovrascrivere`DidEnterBackground`e`WillEnterForeground`per chiamare`MobileAnalyticsManager.PauseSesione`e`MobileAnalyticsManager.ResumeSession`cor mostrato nel seguente frammento:

```
public override void DidEnterBackground(UIApplication application)
{
  // ...
  _manager.PauseSesson();
  // ...
}

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

Xamarin per Android

Per le app Android chiama`MobileAnalyticsManager.PauseSesionnellaOnPause()` metodo e`MobileAnalyticsManager.ResumeSessionnellaOnResume()` metodo come mostrato nel seguente frammento di codice:

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

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

Per impostazione predefinita, se l'utente passa lo stato attivo dall'app per meno di 5 secondi e torna all'app, la sessione verrà ripresa. Se l'utente allontana lo stato attivo dall'app per 5 secondi o più, verrà creata una nuova sessione. Questa impostazione è configurabile nel file di configurazione aws_mobile_analytics.json impostando la proprietà «SESSION_DELTA» sul numero di secondi di attesa prima di creare una nuova sessione.

Amazon Simple Storage Service (S3)

Che cos'è S3?

Amazon Simple Storage Service (Amazon S3)offre agli sviluppatori uno storage di oggetti sicuro, durevole e altamente scalabile. Amazon S3 è facile da usare e include un'interfaccia Web service intuitiva che consente di archiviare e recuperare qualsiasi quantità di dati ovunque nel Web. Con Amazon S3 paghi solo per lo storage effettivamente utilizzato. Non è prevista una tariffa minima e non viene applicato alcun costo di configurazione.

Amazon S3 fornisce storage di oggetti conveniente per un'ampia varietà di casi d'uso, tra cui applicazioni cloud, distribuzione di contenuti, backup e archiviazione, disaster recovery e analisi dei big data.

Per informazioni sulla disponibilità di AWS S3 Region, consultaDisponibilità di una regione AWS Service.

Concetti chiave

Bucket

Ogni oggetto archiviato in Amazon S3 risiede in un bucket. I bucket consentono di raggruppare oggetti correlati nello stesso modo in cui si utilizza una directory per raggruppare i file in un file system. I bucket hanno proprietà, come le autorizzazioni di accesso e lo stato del controllo delle versioni, ed è possibile specificare l'area in cui si desidera che risiedano.

Per ulteriori informazioni su bucket S3, consultaUtilizzo dei bucketnella Guida per gli sviluppatori S3.

Oggetti

Gli oggetti sono dati archiviati in Amazon S3. Ogni oggetto risiede in un bucket creato in una regione AWS specifica.

Gli oggetti archiviati in una regione non la lasciano mai a meno che non vengano trasferiti esplicitamente in un'altra regione. Ad esempio, gli oggetti archiviati nella regione UE (Irlanda) non lasceranno mai tale regione. Gli oggetti archiviati in una regione Amazon S3 rimangono fisicamente in tale regione. Amazon S3 non conserva copie né esegue lo spostamento di tali copie in altre

regioni. Sarà tuttavia possibile accedere agli oggetti da qualsiasi posizione, purché si disponga delle autorizzazioni necessarie.

Gli oggetti possono essere rappresentati da qualsiasi tipo di file: immagini, dati di backup, filmati e così via. La dimensione massima per un oggetto è di 5 TB. Un bucket può avere un numero illimitato di oggetti.

Prima di poter caricare un oggetto in Amazon S3, è necessario disporre delle autorizzazioni di scrittura in un bucket. Per ulteriori informazioni sull'impostazione delle autorizzazioni del bucket, consultaModifica delle autorizzazioni del bucketnella Guida per gli sviluppatori S3.

Per ulteriori informazioni sugli oggetti S3, consultaUtilizzo degli oggettinella Guida per gli sviluppatori S3.

Metadata degli oggetti

Ciascun oggetto in Amazon S3 dispone di un set di coppie chiave-valore che ne rappresenta i metadati. Esistono due tipi di metadata:

- Metadati di sistema— Talvolta elaborato da Amazon S3, ad esempio Content-Type e Content-Length.
- Metadati utente— Non elaborato mai da Amazon S3. I metadata degli utenti vengono archiviati e restituiti con l'oggetto. La dimensione massima dei metadata degli utenti è di 2 KB. Sia le chiavi, sia i relativi valori devono inoltre essere conformi agli standard US-ASCII.

Per ulteriori informazioni sui metadati dell'oggetto S3, consultaModifica dei metadata dell'oggetto.

Configurazione progetto

Prerequisiti

Per utilizzare Amazon S3 nella tua applicazione, devi aggiungere l'SDK al tuo progetto. A tale scopo, segui le istruzioni riportate inConfigurazione dell'SDK AWS Mobile per.NET e Xamarin.

Creare un bucket S3

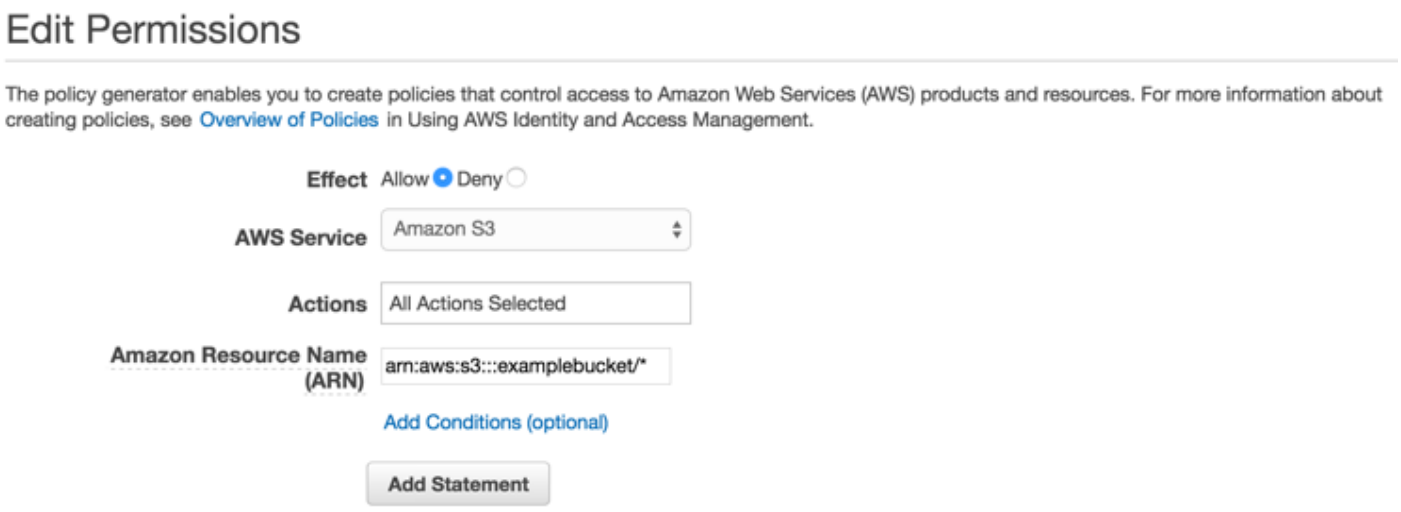
Amazon S3 memorizza le risorse della tua applicazione nei bucket Amazon S3, ovvero contenitori di storage cloud che vivono in uno specificoregione. Ciascun bucket Amazon S3 deve avere un nome univoco globale. Puoi utilizzare il pluginConsole Amazon S3per creare un bucket.

1. Accedi allaConsole Amazon S3e fai clic suCrea bucket.
2. Immettere un nome bucket, selezionare una regione e fare clic suCreate.

Impostazione delle autorizzazioni per S3

La policy del ruolo IAM predefinita concede alla tua applicazione l'accesso ad Amazon Mobile Analytics e Amazon Cognito Sync. Affinché il pool di identità Cognito acceda ad Amazon S3, devi modificare i ruoli del pool di identità.

1. Passa allaConsole di Identity and Access Managemente fai clic suRuolinel riquadro sinistro.
2. Digitare il nome del pool di identità nella casella di ricerca. Vengono elencati due ruoli, relativi, rispettivamente, agli utenti autenticati e non.
3. Fare clic sul ruolo degli utenti non autenticati (verrà aggiunto non autenticato al nome del pool di identità).
4. Fare clic suCreazione di policy di ruolo, selezionaGeneratore di policye quindi fai clic suSeleziona.
5. SulModifica delle autorizzazioni(ARN), inserisci le impostazioni mostrate nella seguente immagine, sostituendo l'Amazon Resource Name (ARN) con il tuo. L'ARN di un bucket S3 sembra`arn:aws:s3:::examplebucket/*`ed è composto dalla regione in cui si trova il bucket e dal nome del bucket. Le impostazioni mostrate di seguito daranno al tuo pool di identità pieno l'accesso a tutte le azioni per il bucket specificato.



1. Fai clic sull'icona della barra degli strumentiAggiungi istruzionepulsante e poi fai clicFase successiva.
2. La procedura guidata mostrerà la configurazione generata. Fare clic suApplica policy.

Per ulteriori informazioni sulla concessione dell'accesso a S3, consultaConcessione dell'accesso a un bucket Amazon S3.

(facoltativo) Configurare la versione Signature per le richieste S3

Ogni interazione con Amazon S3 è autenticata o anonima. AWS utilizza gli algoritmi Signature Version 4 o Signature Version 2 per autenticare le chiamate al servizio.

Tutte le nuove regioni AWS create dopo gennaio 2014 supportano solo Signature Version 4. Tuttavia, molte regioni precedenti supportano ancora le richieste Signature Version 4 e Signature Version 2.

Se il bucket si trova in una delle regioni che non supportano le richieste Signature Version 2 come elencato suquesta pagina, devi impostareAWSConfigsS3.UseSignatureproprietà Version4 su «true» in questo modo:

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

Per ulteriori informazioni sulle versioni di AWS Signature, consultaAutenticazione delle richieste (AWS Signature Version 4).

Integrazione di S3 con l'applicazione

Sono disponibili due modi per interagire con S3 nella tua applicazione Xamarin. I due metodi sono approfonditi nei seguenti argomenti:

Utilizzo di S3 Transfer Utility

S3 Transfer Utility semplifica il caricamento e il download di file su S3 dall'applicazione Xamarin.

Inizializzare ilTransferUtility

Crea un client S3, passandolo al tuo oggetto credenziali AWS, quindi passa il client S3 all'utilità di trasferimento, in questo modo:

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

(facoltativo) Configurare ilTransferUtility

Sono disponibili tre proprietà opzionali che è possibile configurare:

- ConcurrentServiceRichieste- Determina il numero di thread attivi o il numero di richieste Web asincrone simultanee che verranno utilizzate per caricare/scaricare il file. Il valore predefinito è 10.
- MinSizeBeforePartCaricamento- Ottiene o imposta la dimensione minima della parte per caricare le parti in byte. Il valore predefinito è 16 MB. La riduzione delle dimensioni minime della parte fa sì che i caricamenti di più parti vengano suddivisi in un numero maggiore di parti più piccole. L'impostazione di questo valore troppo basso ha un effetto negativo sulle velocità di trasferimento, causando ulteriore latenza e comunicazione di rete per ogni parte.
- NumberOfUploadThreads- Ottiene o imposta il numero di thread in esecuzione. Questa proprietà determina quanti thread attivi verranno utilizzati per caricare il file. Il valore predefinito è 10 thread.

Per configurare S3TransferUtilityclient, crea un oggetto di configurazione, imposta le tue proprietà e passa l'oggetto al tuoTransferUtilitycostruttore in questo modo:

```
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);
```

Download di un file

Per scaricare un file da S3, chiamare`Download`sull'oggetto Transfer Utility, passando i seguenti parametri:

- `file`- Nome stringa del file da scaricare
- `bucketName`- Nome stringa del bucket S3 da cui si desidera scaricare il file
- `key`- Una stringa che rappresenta il nome dell'oggetto S3 (in questo caso un file) da scaricare

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

Caricamento di un file

Per caricare un file su S3, chiama`Upload`sull'oggetto Transfer Utility, passando i seguenti parametri:

- `file`- Nome stringa del file da caricare
- `bucketName`- Nome stringa del bucket S3 per archiviare il file

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

Il codice precedente presuppone che ci sia un file nella directory Environment.SpecialFolder.ApplicationData. I caricamenti utilizzano automaticamente la funzionalità di caricamento multiparte di S3 su file di grandi dimensioni per migliorare il throughput.

Utilizzo delle API Service Level S3

Oltre all'utilizzo di S3TransferUtility, puoi anche interagire con S3 utilizzando le API S3 di basso livello.

Inizializzazione del client Amazon S3

Per utilizzare Amazon S3, dobbiamo innanzitutto creare un'istanza AmazonS3Client che richieda un riferimento a CognitoAWSCredentialsistanza che hai creato in precedenza e la tua regione:

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

Download di un file

Per scaricare un file da S3:

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

Caricamento di un file

Per caricare un file in S3:

```
// 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);
```

Eliminazione di una voce

Per eliminare una voce in 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);
```

Eliminazione di più elementi

Per eliminare più oggetti da un bucket utilizzando una singola richiesta HTTP:

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

È possibile specificare fino a 1000 chiavi.

Creazione di un elenco di bucket

Per restituire una lista di tutti i bucket di proprietà del mittente autenticato della richiesta:

```
// 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);
}
```

Elenco di oggetti

È possibile restituire alcuni o tutti (fino a 1000) gli oggetti memorizzati nel bucket S3. Per farlo, devi avere accesso in lettura al 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);
  }
}
```

Ottenere una regione del bucket

Per ottenere la regione in cui risiede un secchio:

```
// 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);
```

Ottenere una politica del bucket

Per ottenere la politica di un secchio:

```
// 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

Che cos'è Amazon DynamoDB?

Amazon DynamoDBè un servizio di database non relazionale, altamente scalabile e altamente scalabile. DynamoDB rimuove le tradizionali limitazioni di scalabilità dello storage dei dati, mantenendo una bassa latenza e prestazioni prevedibili.

Concetti chiave

I concetti del modello di dati DynamoDB includono tabelle, item e attributi.

Tabelle

In Amazon DynamoDB, un database è una raccolta di tabelle. Una tabella è una raccolta di voci e ogni item è una raccolta di attributi.

In un database relazionale, una tabella ha uno schema predefinito come il nome della tabella, la chiave primaria, l'elenco dei nomi delle colonne e i relativi tipi di dati. Tutti i record memorizzati nella tabella devono avere lo stesso set di colonne. Al contrario, DynamoDB richiede solo che una tabella abbia una chiave primaria, ma non richiede di definire in anticipo tutti i nomi degli attributi e i tipi di dati.

Per ulteriori informazioni sull'utilizzo delle tabelle, consultaUtilizzo di tabelle in DynamoDB.

Elementi e attributi

I singoli elementi di una tabella DynamoDB possono avere un numero qualsiasi di attributi, anche se esiste un limite di 400 KB per le dimensioni dell'articolo. La dimensione di un item è data dalla somma delle lunghezze dei nomi e dei valori dei relativi attributi (lunghezze binarie e UTF-8).

Ogni attributo in un elemento è una coppia nome-valore. Un attributo può essere impostato a valore singolo o multivalore. Ad esempio, un elemento del libro può avere attributi di titolo e autori. Ogni libro ha un titolo ma può avere molti autori. L'attributo multivalore è un insieme; i valori duplicati non sono consentiti.

Ad esempio, si consideri di archiviare un catalogo di prodotti in DynamoDB. È possibile creare una tabella,ProductCatalog, con l'attributo Id come chiave primaria. La chiave primaria identifica in modo univoco ciascun item, in modo che nessun prodotto nella tabella possa avere lo stesso ID.

Per ulteriori informazioni sull'utilizzo di item, consultaUtilizzo di item in DynamoDB.

Tipi di dati

Amazon DynamoDB supporta i seguenti tipi di dati:

- Tipi scalari— Number, String, Binary, Booleano e Null.
- Tipi multivalore— Set di stringhe, set di numeri e set binario.
- Tipi di documento— Elenco e mappa.

Per ulteriori informazioni su tipi di dati scalari, tipi di dati a più valori e tipi di dati di documento, vedereTipi di dati DynamoDB.

Chiave primaria

Quando crei una tabella, oltre al nome della tabella, è necessario specificare la chiave primaria della tabella. La chiave primaria identifica in modo univoco ciascun item della tabella, in modo che nessun item possa avere la stessa chiave. DynamoDB supporta i due tipi di chiavi primarie seguenti:

- Chiave hash: La chiave primaria è costituita da un attributo, un attributo hash. DynamoDB crea un indice di hash non ordinato su questo attributo chiave primaria. Ogni item nella tabella è identificato univocamente dal valore della chiave hash.
- Hash e Range Key: La chiave primaria è composta da due attributi. Il primo attributo è l'attributo hash e il secondo è l'attributo range. DynamoDB crea un indice hash non ordinato sull'attributo della chiave primaria hash e un indice di intervallo ordinato sull'attributo della chiave primaria di intervallo. Ogni item nella tabella è identificato univocamente dalla combinazione dei valori della chiave hash e intervallo. È possibile che due elementi abbiano lo stesso valore della chiave hash, ma questi due elementi devono avere valori di chiave di intervallo diversi.

Indici secondari

Quando si crea una tabella con una chiave hash e una chiave di intervallo, è possibile definire, a tua piacenza, uno o più indici secondari in quella tabella. Un indice secondario consente di eseguire query sui dati nella tabella utilizzando una chiave alternativa, oltre alle query sulla chiave primaria.

DynamoDB supporta due tipi di indici secondari: indici secondari locali e indici secondari globali.

- Indice secondario locale: Indice con la stessa chiave hash della tabella ma con una chiave di intervallo diversa.
- Indice secondario globale: Indice con una chiave hash e una chiave di intervallo che possono essere differenti da quelle presenti sul tavolo.

È possibile definire fino a 5 indici secondari globali e 5 indici secondari locali per tabella. Per ulteriori informazioni, consultaMiglioramento dell'accesso ai dati tramite gli indici secondari in DynamoDBnella Guida per gli sviluppatori di DynamoDB.

Effettua query e scansione

Oltre a utilizzare le chiavi primarie per accedere agli elementi, Amazon DynamoDB fornisce anche due API per la ricerca dei dati: Effettua query e scansione. Ti consigliamo di leggereLinee guida per eseguire query e scansioninella DynamoDB Developer Guide per familiarizzare con alcune best practice.

Query

Un'operazione di query consente di trovare le voci in una tabella o un indice secondario utilizzando solo i valori degli attributi della chiave primaria. È necessario fornire un nome di attributo chiave hash e un valore distinto da cercare. Facoltativamente, puoi fornire un nome e un valore dell'attributo della chiave di intervallo e utilizzare un operatore di confronto per perfezionare i risultati della ricerca.

Per ulteriori query di esempio, consulta:

- Uso del modello di documento
- Utilizzo del modello di persistenza degli oggetti
- Utilizzo delle API di livello di servizio DynamoDB

Per ulteriori informazioni su Query, consultaQuerynella Guida per gli sviluppatori di DynamoDB.

Scan

Un'operazione di scansione legge ogni elemento in una tabella o in un indice secondario. Per impostazione predefinita, un'operazione di scansione restituisce tutti gli attributi dei dati per ogni item nella tabella o nell'indice. Puoi utilizzare il pluginProjectionExpressionparametro in modo che Scan restituisca solo alcuni attributi, piuttosto che tutti.

Per le scansioni di esempio, vedere:

- Uso del modello di documento
- Utilizzo del modello di persistenza degli oggetti
- Utilizzo delle API di livello di servizio DynamoDB

Per ulteriori informazioni su Scan, consultaScannella Guida per gli sviluppatori di DynamoDB.

Configurazione progetto

Prerequisiti

Per utilizzare DynamoDB nella tua applicazione, devi aggiungere l'SDK al tuo progetto. A tale scopo, segui le istruzioni inConfigurazione dell'SDK AWS Mobile per.NET e Xamarin.

Creazione di una tabella DynamoDB

Per creare una tabella, vai allaConsole DynamoDBe segui la procedura riportata di seguito.

1. Fare clic su Create Table (Crea tabella).
2. Immettere il nome della tabella.
3. SelezionaHashcome tipo di chiave primaria.
4. Selezionare un tipo e immettere un valore per il nome dell'attributo hash. Fai clic su Continue (Continua).
5. SulADD INDEXpage, se si prevede di utilizzare indici secondari globali, impostareTipo di indicea «Indice secondario globale» e sottoChiave di indice hash, immettere un valore per l'indice secondario. Ciò consentirà di eseguire query e scansionare utilizzando sia l'indice primario che l'indice secondario. Fare clic suAggiungi indice alla tabella, quindi fai clic suContinua. Per saltare l'utilizzo di indici secondari globali, fare clic suContinua.
6. Imposta la capacità di lettura e scrittura sui livelli desiderati. Per ulteriori informazioni sulla configurazione della capacità, consultaThroughput fornito in Amazon DynamoDB. Fai clic su Continue (Continua).
7. Nella schermata successiva, inserisci un'e-mail di notifica per creare allarmi di throughput, se lo desideri. Fai clic su Continue (Continua).
8. Nella pagina di riepilogo, fai clic suCreate. DynamoDB creerà il database.

Impostazione delle autorizzazioni per DynamoDB

Per utilizzare DynamoDB in un'applicazione, è necessario impostare le autorizzazioni corrette. Il seguente criterio IAM consente all'utente di eliminare, ottenere, inserire, eseguire query, scansionare e aggiornare gli elementi in una tabella DynamoDB specifica, identificata daARN:

```
{
  "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"
    }
  ]
}
```

È possibile modificare le policy nellaConsole IAM. È necessario aggiungere o rimuovere le azioni consentite in base alle esigenze della tua app.

Per ulteriori informazioni sulle policy IAM, consulta la sezione relativa all'utilizzo di IAM.

Per ulteriori informazioni sulle policy specifiche di DynamoDB, consultaUtilizzo di IAM per controllare l'accesso alle risorse DynamoDBnella Guida per gli sviluppatori di DynamoDB.

Integrazione di DynamoDB con la tua applicazione

L'SDK AWS Mobile per.NET e Xamarin offre una libreria di alto livello per lavorare con DynamoDB. È inoltre possibile effettuare richieste direttamente contro l'API DynamoDB di basso livello, ma per la maggior parte dei casi d'uso è consigliata la libreria di alto livello. LaAmazonDynamoDBClient è una parte particolarmente utile della libreria di alto livello. Utilizzando questa classe, puoi eseguire varie operazioni di creazione, lettura, aggiornamento ed eliminazione (CRUD) ed eseguire query.

L'SDK AWS Mobile per.NET e Xamarin consente di effettuare chiamate utilizzando le API dell'AWS SDK for .NET funzionare con DynamoDB. Tutte le API sono disponibili nelAWSSDK.dll. Per ulteriori informazioni sul download dell'AWS SDK for .NET, consultaSDK AWS per .NET.

Esistono vari modo per interagire con DynamoDB nella tua applicazione Xamarin:

- Modello di documento: Questa API fornisce classi wrapper intorno all'API DyanMoDB di basso livello per semplificare ulteriormente le attività di programmazione. La tabella e il documento sono le classi chiave del wrapper. È possibile utilizzare il modello di documento per le operazioni dei dati, ad esempio creare, recuperare, aggiornare ed eliminare elementi. L'API è disponibile in Amazon.dynamoDB.DocumentModelspazio dei nomi.
- Modello di persistenza degli oggetti: L'API di persistenza degli oggetti consente di mappare le classi lato client alle tabelle DynamoDB. Ogni istanza dell'oggetto viene quindi mappata a un elemento nelle tabelle corrispondenti. La classe DynamoDBContext in questa API fornisce metodi per salvare gli oggetti lato client in una tabella, recuperare gli elementi come oggetti ed eseguire query e scansioni. È possibile utilizzare il modello di persistenza oggetti per le operazioni dei dati, ad esempio creare, recuperare, aggiornare ed eliminare elementi. È necessario prima creare le tabelle utilizzando l'API del client di servizio e quindi utilizzare il modello di persistenza degli oggetti per mappare le classi alle tabelle. L'API è disponibile in Amazon.dynamoDB.DataModelspazio dei nomi.
- Client API di servizio: Questa è l'API a livello di protocollo che mappa strettamente all'API DynamoDB. È possibile utilizzare questa API di basso livello per tutte le operazioni di tabelle e elementi, come la creazione, l'aggiornamento, l'eliminazione di tabelle e elementi. È inoltre possibile eseguire query e scansionare le tabelle. Questa API è disponibile nello spazio dei nomi Amazon.dynamoDB.

Questi tre modelli sono approfonditi nei seguenti argomenti:

Uso del modello di documento

Il modello Document fornisce classi wrapper attorno all'API .NET di basso livello. La tabella e il documento sono le classi chiave del wrapper. È possibile utilizzare il modello di documento per creare, recuperare, aggiornare ed eliminare gli elementi. Per creare, aggiornare ed eliminare le tabelle, è necessario utilizzare l'API di basso livello. Per istruzioni su come utilizzare l'API di basso livello, consultaUtilizzo delle API di livello di servizio DynamoDB. L'API di basso livello è disponibile su Amazon.dynamoDB.DocumentModelspazio dei nomi.

Per ulteriori informazioni sul modello di documento, consultaModello di documento .NET.

Creazione di un client DynamoDB

Per creare un client DynamoDB:

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

Operazioni CRUD

Salvataggio di una voce

Creazione di una voce:

```
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";
```

Salva una voce in una tabella DynamoDB:

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

Recupero di una voce

Per recuperare un articolo:

```
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);
}
```

Aggiornamento di una voce

Per aggiornare una voce:

```
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);
}
```

Per aggiornare una voce in modo condizionale:

```
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);
}
```

Eliminazione di una voce

Per eliminare una voce:

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

```
  await books.DeleteItemAsync(id);
}
```

Esegui query e scansioni

Per interrogare e recuperare tutti i libri il cui autore è «Mark Twain»:

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

Il codice di esempio di scansione riportato di seguito restituisce tutti i libri nella nostra tabella:

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

Utilizzo del modello di persistenza degli oggetti

L'SDK AWS Mobile per .NET e Xamarin offre un modello di persistenza degli oggetti che permette di mappare le classi lato client a una tabella DynamoDB. Ogni istanza dell'oggetto viene quindi mappata a un elemento nella tabella corrispondente. Per salvare gli oggetti lato client in una tabella, il modello

di persistenza degli oggetti fornisce la classe DynamoDBContext, un punto di ingresso a DynamoDB. Questa classe fornisce una connessione a DynamoDB e permette di accedere alle tabelle, eseguire varie operazioni CRUD ed eseguire query.

Il modello di persistenza degli oggetti non fornisce un'API per creare, aggiornare o eliminare tabelle. Esso fornisce solo operazioni di dati. Per creare, aggiornare ed eliminare tabelle, è necessario utilizzare l'API di basso livello. Per istruzioni sull'utilizzo dell'API di basso livello, consultaUtilizzo delle API di livello di servizio DynamoDB.

Panoramica

Il modello di persistenza degli oggetti fornisce un insieme di attributi per mappare le classi lato client alle tabelle e le proprietà o i campi agli attributi di tabella. Il modello di persistenza degli oggetti supporta sia la mappatura esplicita che quella predefinita tra le proprietà della classe e gli attributi di tabella.

- Mappatura esplicita: Per mappare una proprietà a una chiave primaria, è necessario utilizzare DynamoDBHashKeye DynamoDBRangeKeyAttributi del modello di persistenza oggetto. Inoltre, per gli attributi della chiave non primaria, se il nome di una proprietà nella classe e l'attributo tabella corrispondente a cui si desidera mappare non sono gli stessi, è necessario definire la mappatura aggiungendo esplicitamente l'attributo DynamoDBProperty.
- Mappatura predefinita- Per impostazione predefinita, il modello di persistenza degli oggetti mappa le proprietà della classe agli attributi con lo stesso nome nella tabella.

Non è necessario mappare ogni singola proprietà della classe. Queste proprietà vengono identificate aggiungendo l'attributo DynamoDBIgnore. Il salvataggio e il recupero di un'istanza di un oggetto ometterebbe qualsiasi proprietà contrassegnata con questo attributo.

Tipi di dati supportati

Il modello di persistenza degli oggetti supporta un insieme di tipi di dati .NET primitivi e tipi di dati arbitrari. Il modello supporta i seguenti tipi di dati primitivi:

- bool
- byte
- char
- DateTime

- decimale, doppio, float
- Int 16, Int32, Int64
- SByte
- string
- Uint16, Uint32, Uint64

Il modello Object Persistence supporta anche i tipi di insieme .NET con le seguenti limitazioni:

- Il tipo di raccolta deve implementare l'interfaccia iCollection.
- Il tipo di raccolta deve essere composto dai tipi primitivi supportati. Ad esempio, iCollection<string>, iCollection<bool>.
- Il tipo di raccolta deve fornire un costruttore senza parametri.

Per ulteriori informazioni sul modello di persistenza degli oggetti, consultaModello di persistenza degli oggetti .NET.

Creazione di un client DynamoDB

Per creare un client DynamoDB

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

Operazioni CRUD

Salva un oggetto

Creazione di un oggetto:

```
[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
};
```

Salva un oggetto in una tabella DynamoDB:

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

Recupero di un oggetto

Per recuperare un oggetto:

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

Aggiornamento di un oggetto

Per aggiornare un oggetto:

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

Eliminazione di un oggetto

Per eliminare un oggetto:

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

Eseguire query e scansioni

Per interrogare e recuperare tutti i libri il cui autore è «Charles Dickens»:

```
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());
  });
}
```

Il codice di esempio di scansione riportato di seguito restituisce tutti i libri nella nostra tabella:

```
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());
  });
}
```

Utilizzo delle API DynamoDB Service Level

Le API di livello di servizio di Dynamo ti consentono di creare, aggiornare ed eliminare tabelle. Puoi anche eseguire operazioni tipiche di creazione, lettura, aggiornamento ed eliminazione (CRUD) sugli item di una tabella utilizzando questa API.

Creazione di un client DynamoDB

Per creare un client DynamoDB

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

Operazioni CRUD

Salvataggio di una voce

Per salvare una voce in una tabella DynamoDB:

```
// 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);
```

Recupero di una voce

Per recuperare un articolo:

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

Aggiornamento di una voce

Per aggiornare una voce:

```
// 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);
```

Eliminazione di una voce

Per eliminare una voce:

```
// 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);
```

Esecuzione query e scansione

Per interrogare e recuperare tutti i libri il cui autore è «Mark Twain»:

```
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);
    });
```

```
  }
}
```

Il codice di esempio di scansione riportato di seguito restituisce tutti i libri nella nostra tabella:

```
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)

Utilizzando SNS e l'SDK AWS Mobile per.NET e Xamarin, è possibile scrivere applicazioni in grado di ricevere notifiche push per dispositivi mobili. Per informazioni su SNS, consultaAmazon Simple Notification Service.

Concetti chiave

Amazon SNS consente alle applicazioni e agli utenti finali su diversi dispositivi di ricevere notifiche tramite notifica Mobile Push (Apple, Google e Kindle Fire Devices), HTTP/HTTPS, Email/Email-JSON, SMS o Amazon Simple Queue Service (SQS) o funzioni AWS Lambda. SNS consente di inviare singoli messaggi o messaggi di fan-out a un numero elevato di destinatari iscritti a un singolo argomento.

Argomenti

Un argomento è un «punto di accesso» per consentire ai destinatari di sottoscrivere dinamicamente copie identiche della stessa notifica. Un argomento può supportare le consegne a più tipi di endpoint, ad esempio, è possibile raggruppare i destinatari iOS, Android e SMS.

Sottoscrizioni

Per ricevere i messaggi pubblicati in un argomento devi effettuare la sottoscrizione di un endpoint all'argomento specificato. Un endpoint è un'app per dispositivi mobili, un server Web, un indirizzo e-mail o una coda Amazon SQS in grado di ricevere messaggi di notifica da Amazon SNS. Una volta effettuata la sottoscrizione di un endpoint a un argomento, e dopo che la sottoscrizione è stata confermata, l'endpoint riceverà tutti i messaggi pubblicati nell'argomento specificato.

Pubblicazione

Quando pubblichi su un argomento, SNS fornisce copie formattate in modo appropriato del messaggio a ciascun abbonato di tale argomento. Per le notifiche push mobili, puoi pubblicare direttamente sull'endpoint o sottoscrivere l'endpoint a un argomento.

Configurazione rapida

Prerequisiti

Per utilizzare SNS nella tua applicazione, devi aggiungere l'SDK al tuo progetto. A tale scopo, segui le istruzioni inConfigurazione dell'SDK AWS Mobile per.NET e Xamarin.

Impostazione delle autorizzazioni per SNS

Per informazioni sull'impostazione delle autorizzazioni per SNS, consultaGestione degli accessi agli argomenti di Amazon SNS.

InserisciNuGetPacchetto per SNS per il tuo progetto

Seguire il passaggio 4 delle istruzioni riportate inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper aggiungere Amazon Simple Notification ServiceNuGetpacchetto per il progetto.

Integrazione di SNS con l'applicazione

Esistono molti modi per interagire con SNS nell'applicazione Xamarin:

Invia notifiche push (Xamarin Android)

Questo documento spiega come inviare notifiche push a un'applicazione Android Xamarin utilizzando Amazon Simple Notification Service (SNS) e AWS Mobile SDK for .NET and Xamarin.

Configurazione progetto

Prerequisiti

È necessario seguire le istruzioni riportate alConfigurazione dell'SDK AWS Mobile per.NET e Xamarinprima di iniziare questo tutorial.

Impostazione delle autorizzazioni per SNS

Segui il passaggio 2 inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper allegare la politica menzionata di seguito ai ruoli della tua candidatura. Ciò darà alla tua applicazione le autorizzazioni appropriate per accedere a SNS:

1. Passa allaConsole IAMe seleziona il ruolo IAM da configurare.

2. Fare clic sucollegamento di policy, seleziona AmazonSNSFullAccesspolicy e cliccacollegamento di policy.

> **Warning**
>
> Usare AmazonSNSFullAccessnon è raccomandato in un ambiente di produzione. Lo usiamo qui per permetterti di iniziare a usare rapidamente. Per ulteriori informazioni sulla definizione delle autorizzazioni per un ruolo IAM, consultaPanoramica delle autorizzazioni del ruolo IAM.

Abilita notifiche push su Google Cloud

Innanzitutto, aggiungi un nuovo progetto Google API:

1. Passa allaConsole per sviluppatori di Google.
2. Fare clic suCreazione di un progetto.
3. NellaNuovo progettocasella, inserisci il nome di un progetto, prendi nota dell'ID del progetto (ne avrai bisogno in seguito) e fai clic suCreate.

Successivamente, abilita il servizio Google Cloud Messaging (GCM) per il progetto:

1. NellaConsole per sviluppatori di Google, il nuovo progetto dovrebbe già essere selezionato. In caso contrario, selezionalo nel menu a discesa nella parte superiore della pagina.
2. SelezionaAPI e autenticazionedalla barra laterale sul lato sinistro della pagina.
3. Nella casella di ricerca, digita «Google Cloud Messaging for Android» e fai clic suGoogle Cloud Messaging per Androidcollegamento.
4. Fare clic suAbilitazione dell'API.

Infine, ottieni una chiave API:

1. Nella Google Developers Console, selezionaAPI e autenticazione >Credenziali.
2. UNDERAccesso pubblico all'API, fare clic suCreazione di una nuova chiave.
3. NellaCreazione di una nuova chiavefinestra di dialogo, fai clicChiave server.
4. Nella finestra di dialogo risultante, fai clic suCreatee copia la chiave API visualizzata. Utilizzerai questa chiave API per eseguire l'autenticazione in un secondo momento.

Utilizzare l'ID progetto per creare un ARN della piattaforma in SNS Console

1. Passa allaConsole SNS.
2. Fare clic suApplicazionisul lato sinistro dello schermo.
3. Fare clic suCreare applicazione piattaformaper creare una nuova applicazione SNS platform.
4. Specificare unNome applicazione.
5. SelezionaGoogle Cloud Messaging (GCM)perPiattaforma di notifiche push.
6. Incolla la chiave API nella casella di testo con l'etichettaChiave API.
7. Fare clic suCreare applicazione piattaforma.
8. Seleziona l'applicazione Platform appena creata e copia l'ARN dell'applicazione.

InserisciNuGetPacchetto per SNS per il tuo progetto

Seguire il passaggio 4 delle istruzioni riportate inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper aggiungere Amazon Simple Notification ServiceNuGetpacchetto per il tuo progetto.

Creare un client SNS

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

Registra la tua applicazione per le notifiche remote

Per registrarsi alle notifiche remote su Android, è necessario creare unBroadcastReceiverche può ricevere messaggi Google Cloud. Modificare il nome del pacchetto qui sotto dove richiesto:

```
[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);
      }
}
```

Di seguito è riportato il servizio che riceve la notifica push dalBroadcastReceivere visualizza la notifica sulla barra delle notifiche del dispositivo:

```
[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());
  }
}
```

Invia un messaggio dalla console SNS al tuo endpoint

1. Passa allaSNS Console > Applicazioni.
2. Seleziona l'applicazione della piattaforma, seleziona un endpoint e fai clic suPubblicazione su endpoint.
3. Digita un messaggio di testo nella casella di testo e fai clic suPubblicazione di messaggiper pubblicare un messaggio.

Invia notifiche push (Xamarin iOS)

Questo documento spiega come inviare notifiche push a un'applicazione Xamarin iOS utilizzando Amazon Simple Notification Service (SNS) e AWS Mobile SDK for .NET and Xamarin.

Configurazione progetto

Prerequisiti

È necessario seguire le istruzioni riportate allaConfigurazione dell'SDK AWS Mobile per.NET e Xamarinprima di iniziare questo tutorial.

Impostazione delle autorizzazioni per SNS

Segui il passaggio 2 inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper allegare la politica menzionata di seguito ai ruoli della tua candidatura. Ciò darà alla tua applicazione le autorizzazioni appropriate per accedere a SNS:

1. Accedi aConsole IAMe seleziona il ruolo IAM da configurare.
2. Fare clic sucollegamento di policy, seleziona AmazonSNSFullAccesspolicy e cliccacollegamento di policy.

> **Warning**
>
> Usare AmazonSNSFullAccessNon è raccomandato in un ambiente di produzione. Lo usiamo qui per permetterti di diventare operativi in tempi brevi. Per ulteriori informazioni sulla definizione delle autorizzazioni per un ruolo IAM, consultaPanoramica delle autorizzazioni dei ruoli IAM.

Ottieni l'iscrizione al programma Apple iOS Developer

Dovrai eseguire l'app su un dispositivo fisico per ricevere notifiche push. Per eseguire la tua app su un dispositivo, devi avere un abbonamento alIscrizione al programma per sviluppatori Apple iOS. Quando disponi di un abbonamento, puoi utilizzare Xcode per generare un'identità di firma. Per ulteriori informazioni, consulta AppleQuick Start di distribuzione appdocumentazione.

Crea un certificato iOS

Innanzitutto, devi creare un certificato iOS. Quindi, è necessario creare un profilo di provisioning configurato per le notifiche push. A questo proposito:

1. Accedi aCentro membri Apple Developer, clicca suCertificati, identificatori e profili.

2. Fare clic suIdentificatori disottoApp iOS, fai clic sul pulsante più nell'angolo in alto a destra della pagina Web per aggiungere un nuovo ID app iOS e Immetti una descrizione dell'ID app.
3. Scorrere in basso fino allaAggiunta di suffisso IDsezione e selezionaID app esplicitoe inserisci il tuo identificatore del bundle.
4. Scorrere in basso fino allaApp Servicessezione e selezionaNotifiche push.
5. Fai clic su Continue (Continua).
6. Fare clic su Submit (Invia).
7. Fare clic suFatto.
8. Seleziona l'ID app appena creato e fai clic suModificare.
9. Scorrere in basso fino allaNotifiche pushsezione. Fare clic suCreazione di certificatosottoCertificato SSL di sviluppo.
10. Segui le istruzioni per creare una richiesta di firma del certificato (CSR), carica la richiesta e scarica un certificato SSL che verrà utilizzato per comunicare con Apple Notification Service (APNS).
11. Torna allaCertificati, identificatori e profili(Certificato creato). Fare clic suAll (Tutti)sottoProfili di provisioning.
12. Fare clic sul pulsante più nell'angolo in alto a destra per aggiungere un nuovo profilo di provisioning.
13. SelezionaSviluppo di app iOS, quindi fai clic suContinua.
14. Selezionare l'ID dell'app e quindi fare clic suContinua.
15. Selezionare il certificato dello sviluppatore, quindi fare clic suContinua.
16. Selezionare il dispositivo, quindi fare clic suContinua.
17. Immettere un nome di profilo, quindi fare clic suGenera.
18. Scarica e fai doppio clic sul file di provisioning per installare il profilo di provisioning.

Per ulteriori informazioni sul provisioning di un profilo configurato per le notifiche push, consulta AppleConfigurazione delle notifiche pushdocumentazione.

Usa certificato per creare ARN della piattaforma in SNS Console

1. EseguireKeyChainaccedi all'app, selezionaI miei certificatinella parte inferiore sinistra dello schermo, quindi fare clic con il pulsante destro del mouse sul certificato SSL generato per

connettersi a APNS e selezionareEsportazione. Ti verrà richiesto di specificare un nome per il file e una password per proteggere il certificato. Il certificato verrà salvato in un file P12.

2. Accedi aConsole SNSe fai clic suApplicazionisul lato sinistro dello schermo.
3. Fare clic suCreazione di applicazioni per piattaformecreare una nuova applicazione di piattaforma SNS.
4. Immettere unNome applicazione.
5. SelezionaSviluppo AppleperPiattaforma di notifiche push.
6. Fare clic suSelezionare Filee seleziona il file P12 creato quando hai esportato il certificato SSL.
7. Immetti la password specificata all'esportazione del certificato SSL e fai clic suCarica credenziali da file.
8. Fare clic suCreazione di applicazioni per piattaforme.
9. Seleziona l'applicazione Platform appena creata e copia l'ARN dell'applicazione. Questo ARN sarà necessario nei prossimi passaggi.

InserisciNuGetPacchetto per SNS per il tuo progetto

Seguire il passaggio 4 delle istruzioni riportate inConfigurazione dell'SDK AWS Mobile per.NET e Xamarinper aggiungere Amazon Simple Notification ServiceNuGetpacchetto per il tuo progetto.

Creazione di un client SNS

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

Registra la tua applicazione per le notifiche remote

Per registrare una domanda, chiama RegisterForRemoteNotifications sull'oggetto UIApplication, come mostrato di seguito. Inserisci il seguente codice inAppDelegate.cs, inserendo l'ARN dell'applicazione della piattaforma dove richiesto di seguito:

```
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 */
    });
  }
}
```

Invia un messaggio dalla console SNS al tuo endpoint

1. Accedi aSNS Console > Applicazioni.
2. Seleziona l'applicazione della piattaforma, seleziona un endpoint e fai clic suPubblicazione su endpoint.
3. Digita un messaggio di testo nella casella di testo e fai clic suPubblicazione di messaggiper pubblicare un messaggio.

Inviare e ricevere notifiche SMS

Puoi usare Amazon Simple Notification Service (Amazon SNS) per inviare e ricevere notifiche SMS (Short Message Service) verso cellulari e smartphone abilitati per SMS (Short Message Service) verso cellulari e smartphone abilitati per SMS.

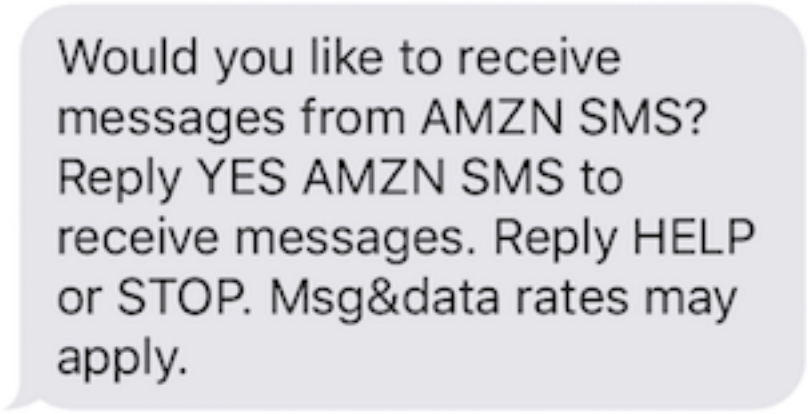
> **Note**
>
> Le notifiche via SMS sono attualmente supportate per numeri di telefono negli Stati Uniti. I messaggi via SMS possono essere inviati solo da argomenti creati nella regione Stati Uniti

orientali (Virginia settentrionale). Tuttavia, puoi pubblicare messaggi in argomenti creati nella regione Stati Uniti orientali (Virginia settentrionale) da qualsiasi altra regione.

Creazione di un argomento

Per creare un argomento:

1. Nella console Amazon SNS, fai clic suCreare un nuovo argomento. Viene visualizzata la finestra di dialogo Crea nuovo argomento.
2. Nella casella Topic name (Nome argomento) inserisci un nome per l'argomento.
3. Nella casella Nome visualizzato digitare un nome visualizzato. All'argomento deve essere assegnato un nome visualizzato perché i primi dieci (10) caratteri del nome visualizzato vengono utilizzati come parte iniziale del prefisso del messaggio di testo. Il nome visualizzato immesso apparirà nel messaggio di conferma che SNS invia all'utente (il nome visualizzato di seguito è «AMZN SMS»).



1. Fai clic su Create topic (Crea argomento). Il nuovo argomento viene visualizzato nella pagina Topics (Argomenti).
2. Seleziona il nuovo argomento, quindi fai clic sul relativo ARN. Viene visualizzata la pagina Topic Details (Dettagli argomento).
3. Copia l'argomento ARN, come ne avrai bisogno quando ti iscrivi a un argomento nel passaggio successivo.

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

Sottoscrizione a un argomento utilizzando il protocollo SMS

Crea un client SNS, passando il tuo oggetto credenziali e la regione del tuo pool di identità:

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

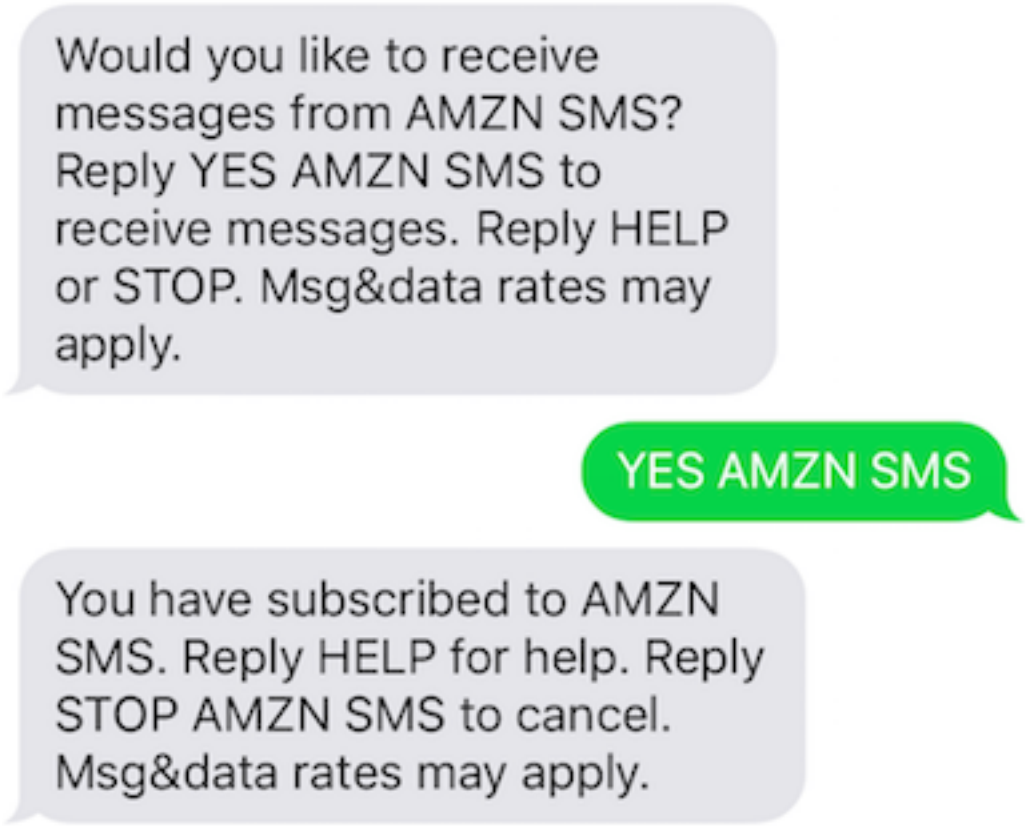
Per effettuare la sottoscrizione a un argomento, richiama`SubscribeAsync`e passalo l'ARN dell'argomento a cui vuoi iscriverti, il protocollo («sms») e il numero di telefono:

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

Riceverai un arn di sottoscrizione nell'oggetto di risposta di sottoscrizione. Il tuo arn di sottoscrizione ha la seguente struttura:

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

Quando un dispositivo sottoscrive un argomento, SNS invierà un messaggio di conferma al dispositivo e l'utente dovrà confermare di voler ricevere notifiche, come mostrato di seguito:

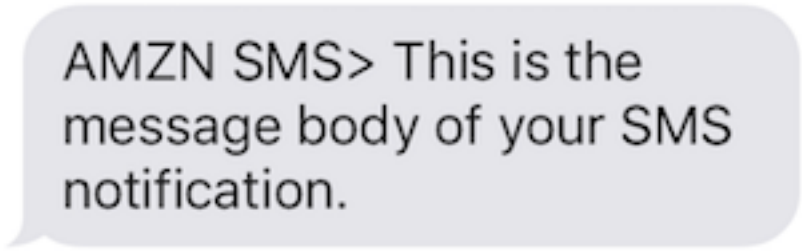


Dopo che l'utente si è iscritto all'argomento, riceverà messaggi SMS quando li pubblicherai su quell'argomento.

Pubblicazione di un messaggio

Per pubblicare un messaggio in un argomento:

1. Accedere alla Console di gestione AWS e aprire laConsole Amazon SNS.
2. Nel riquadro di navigazione sinistro, fai clic su Topics (Argomenti) e seleziona l'argomento in cui effettuare la pubblicazione.
3. Fare clic suPubblicazione in argomento.
4. Nella casella Subject (Oggetto) digita un oggetto.
5. Nella casella Messaggio (Messaggio) digita un messaggio. Amazon SNS invia il testo immesso nella casella Messaggio agli abbonati SMS a meno che tu non inserisca anche testo nella casella Oggetto. Poiché Amazon SNS include un prefisso del nome visualizzato con tutti i messaggi SMS inviati, la somma del prefisso del nome visualizzato e del payload del messaggio non possono superare i 140 caratteri ASCII o 70 caratteri Unicode. Amazon SNS tronca i messaggi che superano questi limiti.
6. Fai clic su Publish Message (Pubblica messaggio). Amazon SNS visualizza una finestra di dialogo di conferma. Il messaggio SMS viene visualizzato sul dispositivo abilitato per SMS, come mostrato di seguito.



Invio di messaggi a endpoint HTTP/HTTPS

Puoi utilizzare Amazon SNS per inviare messaggi di notifica a uno o più endpoint HTTP o HTTPS. Di seguito è riportato il procedimento:

1. Per configurare l'endpoint la ricezione di messaggi Amazon SNS.
2. Sottoscrivere un endpoint HTTP/HTTPS a un argomento.

3. Confermare la sottoscrizione.
4. Pubblica una notifica nell'argomento. Amazon SNS provvederà a inviare una richiesta HTTP POST per consegnare il contenuto della notifica all'endpoint dotato di sottoscrizione.

Configurare il tuo endpoint HTTP/HTTPS per ricevere messaggi Amazon SNS

Seguire le istruzioni riportate nel passaggio 1 diInvio di messaggi Amazon SNS a endpoint HTTP/HTTPSper configurare l'endpoint.

Sottoscrizione dell'endpoint HTTP/HTTPS all'argomento Amazon SNS

Crea un client SNS, passando il tuo oggetto credenziali e la regione del tuo pool di identità:

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

Per inviare messaggi a un endpoint HTTP o HTTPS tramite un argomento, devi effettuare la sottoscrizione dell'endpoint all'argomento Amazon SNS. L'endpoint deve essere specificato tramite il relativo URL:

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

Conferma della sottoscrizione a

A seguito della sottoscrizione di un endpoint, Amazon SNS invierà a tale endpoint un messaggio di conferma della sottoscrizione Il codice sull'endpoint deve recuperare il`SubscribeURL`valore dal messaggio di conferma dell'abbonamento e visita la posizione specificata dal`SubscribeURL`se stesso o rendilo disponibile per te in modo da poter visitare manualmente il`SubscribeURL`(ad esempio, se si utilizza un browser Web).

Amazon SNS non invierà messaggi all'endpoint fino alla conferma della sottoscrizione. Quando visiterai `SubscribeURL`, otterrai una risposta con un documento XML contenente un elemento `SubscriptionArn` che specifica l'ARN della sottoscrizione.

Invio di messaggi all'endpoint HTTP/HTTPS

Puoi inviare un messaggio alle sottoscrizioni di un argomento pubblicando nell'argomento. RichiamoPublishAsyncepassalo l'argomento ARN e il tuo messaggio.

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

Risoluzione dei problemi di SNS

Utilizzo dello stato di spedizione in Amazon SNS Console

La console Amazon SNS contiene una funzione Stato di consegna che ti consente di raccogliere feedback sui tentativi di recapito riusciti e non riusciti dei tuoi messaggi alle piattaforme di notifica push mobile (Apple (APNS), Google (GCM), Amazon (ADM), Windows (WNS e MPNS) e Baidu.

Fornisce anche altre informazioni importanti come i tempi di soggiorno in Amazon SNS. Queste informazioni vengono acquisite in AmazonCloudWatchGruppo di log creato automaticamente da Amazon SNS quando questa funzione è abilitata tramite la console Amazon SNS o tramite le API Amazon SNS.

Per istruzioni su come usare la funzione Stato di consegna, consultaUtilizzo della funzione Stato di consegna di Amazon SNSsul blog di AWS Mobile.

Best practice per l'utilizzo dell'SDK AWS Mobile per.NET e Xamarin

Esistono solo alcuni fondamentali e best practice che sono utili da sapere quando si utilizza AWS Mobile SDK for .NET and Xamarin.

- Usa Amazon Cognito per ottenere credenziali AWS anziché codificare le credenziali nella tua applicazione. Se codifica le credenziali nella tua applicazione, potresti finire per esporle al pubblico, consentendo ad altri di effettuare chiamate ad AWS utilizzando le tue credenziali. Per istruzioni su come utilizzare Amazon Cognito per ottenere le credenziali AWS, consulta la sezioneConfigurazione dell'SDK AWS Mobile per.NET e Xamarin.
- Per le best practice da seguire per l'utilizzo di S3, consultaquesto articolo sul blog AWS.
- Per le best practice da seguire per l'utilizzo di DynamoDB, consultaBest practice DynamoDBnella Guida per gli sviluppatori di DynamoDB.

Siamo sempre alla ricerca di aiutare i nostri clienti ad avere successo e feedback graditi, quindi non esitate apost sui forum AWSoapri un problema suGitHub.

Documentazione della libreria di AWS Service

Ogni servizio in AWS Mobile SDK for .NET and Xamarin ha una guida per sviluppatori separata e un riferimento all'API del servizio che fornisce informazioni aggiuntive che potresti trovare utili.

Amazon Cognito Identity

- Guida per sviluppatori di Cognito
- Cognito riferimento dell'API del servizio di identità

Amazon Cognito Sync

- Guida per sviluppatori di Cognito
- Service API Cognito Sync

Amazon Mobile Analytics

- Guida per sviluppatori di Mobile Analytics
- Documentazione di riferimento dell'API Mobile Analytics

Amazon S3

- Guida per gli sviluppatori S3
- Guida alle operazioni di base di S3
- Service API S3 Service

Amazon DynamoDB

- Guida per gli sviluppatori di DynamoDB
- Guida alle procedure di base di DynamoDB
- Documentazione di riferimento dell'API di DynamoDB

Amazon Simply Notification Service (SNS)

- Guida per gli sviluppatori di SNS
- Service API SNS Service

Altri link utili

- Glossario AWS dei termini
- Informazioni su AWS Credentials

Risoluzione dei problemi

In questo argomento vengono descritte alcune idee per la risoluzione dei problemi che potrebbero verificarsi quando si utilizza l'SDK AWS Mobile per .NET e Xamarin.

Assicurati che il ruolo IAM abbia le autorizzazioni richieste

Quando chiami i servizi AWS, la tua app dovrebbe utilizzare un'identità proveniente da un pool di identità Cognito. Ogni identità del pool è associata a un ruolo IAM (Identity and Access Management).

A un ruolo sono associati uno o più file di criteri che specificano le risorse AWS a cui gli utenti assegnati al ruolo hanno accesso. Per impostazione predefinita, vengono creati due ruoli per pool di identità: uno per gli utenti autenticati e uno per gli utenti non autenticati.

Dovrai modificare il file dei criteri esistente o associare un nuovo file di criteri alle autorizzazioni richieste dall'app. Se la tua app consente utenti autenticati e non autenticati, a entrambi i ruoli devono essere concesse le autorizzazioni per l'accesso alle risorse AWS di cui l'app ha bisogno.

Il seguente file policy mostra come concedere l'accesso a un bucket S3:

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

Il seguente file di criteri mostra come concedere l'accesso a un database DynamoDB:

```
{
   "Statement":[
```

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

Per ulteriori informazioni su come specificare le policy;, consultaPolicy IAM.

Utilizzo di un debugger proxy HTTP

Se il servizio AWS che stai chiamando l'app ha un endpoint HTTP o HTTPS, puoi utilizzare un debugger proxy HTTP/HTTPS per visualizzare le richieste e le risposte per ottenere maggiori informazioni su ciò che sta accadendo. Sono disponibili diversi debugger proxy HTTP, come:

- Carlo- un proxy di debug web per Windows e OSX
- Violinista- un proxyfidd di debug web per Windows

Sia Charles che Fiddler richiedono una certa configurazione per poter visualizzare il traffico crittografato SSL, leggi la documentazione di questi strumenti per ulteriori informazioni. Se si utilizza un proxy di debug Web che non può essere configurato per visualizzare il traffico crittografato, aprire il file aws_endpoints_json e impostare il tag HTTP per il servizio AWS necessario per il debug su true.

Cronologia dei documenti

Nella tabella seguente vengono descritte importanti modifiche alla documentazione rispetto all'ultima versione dell'SDK AWS Mobile per .NET e Xamarin.

- Versione API: 27-08-2015
- Ultimo aggiornamento della documentazione: 20-02-2012

Modifica	Versione API	Descrizione	Data di rilascio
Archiviato	27-08-2015	LaAWSMobile SDK per Xamarin è incluso nelAWS SDK for .NET. Questa guida fa riferimento alla versione archiviat a dell'SDK Mobile per Xamarin.	20-02-2012
Versione GA di	27-08-2015	Versione GA di	27-08-2015
Versione beta	28-07-2015	Versione beta	28/07/2012