Guia do desenvolvedor do Xamarin

AWS Mobile SDK

AWS Mobile SDK: Guia do desenvolvedor do Xamarin

Table of Contents

O SDK AWS móvel para Xamarin agora está incluído no. AWS SDK para .NET Este guia faz referência à versão arquivada do Mobile SDK para Xamarin.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.

O que é o AWS Mobile SDK para .NET e Xamarin?

O SDK AWS móvel para Xamarin está incluído no. SDK para .NET Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para .NET .

Este guia não é mais atualizado. Ele faz referência à versão arquivada do Mobile SDK para Xamarin.

Guias e tópicos relacionados

- Para desenvolvimento de aplicativos front-end e móveis, recomendamos o uso da AWS Amplify.
- Para considerações especiais sobre o uso do AWS SDK para .NET em seus aplicativos Xamarin, consulte Considerações especiais sobre o suporte do Xamarin no Guia do desenvolvedor.AWS SDK para .NET
- Para fins de referência, você pode encontrar a versão arquivada do SDK AWS móvel para Xamarin em. GitHub

Conteúdo de referência arquivado

O AWS Mobile SDK arquivado para .NET e Xamarin fornece um conjunto de bibliotecas .NET, exemplos de código e documentação para ajudar os desenvolvedores a criar aplicativos móveis conectados para:

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

Aplicativos móveis criados usando o AWS Mobile SDK para .NET e Xamarin APIs chamam de plataforma nativa para que tenham a aparência de aplicativos nativos. As bibliotecas.NET no SDK fornecem wrappers de C# em todo o AWS REST. APIs

O que está incluído no AWS Mobile SDK para .NET e Xamarin?

Os serviços da AWS compatíveis no momento incluem, sem limitação:

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

Esses serviços permitem autenticar usuários, salvar dados de jogadores e jogos, salvar objetos na nuvem, receber notificações por push, e coletar e analisar dados de uso.

O AWS Mobile SDK para .NET e Xamarin também permite que você use a maioria dos serviços da AWS compatíveis com o AWS SDK para .NET. Os serviços da AWS de desenvolvimento móvel são explicados neste guia do desenvolvedor. Para obter mais informações sobre o AWS SDK para .NET, consulte:

- Guia de conceitos básicos do AWS SDK para .NET
- Guia do desenvolvedor do AWS SDK para .NET
- Referência de API do AWS SDK para .NET

Compatibilidade

O AWS Mobile SDK para .NET e Xamarin é fornecido como uma biblioteca de classes portátil (PCL). O suporte à PCL foi adicionado no Xamarin.Android 4.10.1 e no Xamarin.iOS 7.0.4. Os projetos da biblioteca portátil são incorporados ao Visual Studio.

IDEs

Para obter mais informações sobre como usar IDEs com a versão arquivada do SDK do Xamarin, consulte. Configurar o AWS Mobile SDK para .NET e Xamarin

Como obtenho o AWS Mobile SDK para .NET e Xamarin?

Para obter o AWS Mobile SDK para .NET e Xamarin, consulte Configurar o AWS Mobile SDK para .NET e Xamarin. O AWS Mobile SDK para .NET e Xamarin NuGet é distribuído como pacotes. Você pode encontrar uma lista completa dos pacotes de serviços da AWS em Pacotes do AWS SDK em NuGet ou no AWS SDK for GitHub .NET Repository.

Sobre o AWS Mobile Services

Identidade do Amazon Cognito

Todas as chamadas feitas para a AWS precisam das credenciais da AWS. Em vez de codificar suas credenciais nos aplicativos, recomendamos que utilize o Amazon Cognito Identity para fornecer credenciais da AWS ao aplicativo. Siga as instruções em Configuração do AWS Mobile SDK para .NET e Xamarin para obter credenciais da AWS por meio do Amazon Cognito.

O Cognito permite também a autenticação de usuários que usam provedores públicos de login, como a Amazon, o Facebook, o Twitter e o Google, além de provedores compatíveis com o OpenID Connect. O Cognito também funciona com usuários não autenticados. O Cognito fornece credenciais temporárias, com direitos limitados de acesso especificados por você, usando uma função do Identity and Access Management (IAM). O Cognito é configurado por meio da criação de um grupo de identidades associado a uma função do IAM. A função do IAM especifica o que resources/services seu aplicativo pode acessar.

Para começar a usar o Cognito Identity, consulte Configuração do AWS Mobile SDK para .NET e Xamarin.

Para saber mais sobre o Cognito Identity, consulte Amazon Cognito Identity.

Amazon Cognito Sync

O Cognito Sync é um serviço da AWS e uma biblioteca de clientes que permite a sincronização dos dados de usuário relacionados a aplicativo entre dispositivos. Você pode usar a API do Cognito Sync para sincronizar dados de perfil de usuário entre dispositivos e entre provedores de login (Amazon, Facebook, Google e seu próprio provedor de identidade personalizada).

Para começar a usar o Cognito Sync, consulte Sincronizar dados do usuário com o Cognito Sync.

Para obter mais informações sobre o Cognito Sync, consulte Amazon Cognito Sync.

Mobile Analytics

O Amazon Mobile Analytics permite coletar, visualizar e compreender o uso dos aplicativos móveis. Há relatórios disponíveis para métricas sobre usuários ativos, sessões, retenção, receita de aplicativo e eventos personalizados. Eles podem ser filtrados por plataforma e intervalo de datas. O Amazon Mobile Analytics foi desenvolvido para ser dimensionado junto com seu negócio, e pode coletar e processar bilhões de eventos de milhões de endpoints.

Para começar a usar o Mobile Analytics, consulte Rastreio de dados de uso do aplicativo com o Amazon Mobile Analytics.

Para obter mais informações sobre o Mobile Analytics, consulte Amazon Mobile Analytics.

Dynamo DB

O Amazon DynamoDB é um serviço de banco de dados rápido, altamente disponível, altamente escalável, econômico e não relacional. O DynamoDB remove limitações de escalabilidade tradicionais sobre armazenamento de dados, mantendo, ao mesmo tempo, a baixa latência e o desempenho previsível.

Para começar a usar o Dynamo DB, consulte Armazene e recupere dados com o DynamoDB.

Para obter mais informações sobre o Dynamo DB, consulte Amazon DynamoDB.

Amazon Simple Notification Service

O Amazon Simple Notification Service (SNS) é um serviço de notificação por push rápido, flexível e totalmente gerenciado que permite enviar mensagens individuais ou encaminhá-las para um grande número de destinatários. Com o Amazon Simple Notification Service, é simples e econômico enviar notificações por push para usuários de dispositivos móveis, destinatários de e-mail ou até mesmo enviar mensagens a outros serviços distribuídos.

Para começar a usar o SNS para Xamarin iOS, consulte Receba notificações por push usando o SNS (Xamarin iOS).

Para começar a usar o SNS para Xamarin Android, consulte Receba notificações por push usando o SNS (Xamarin Android).

Para obter mais informações sobre o SNS, consulte Amazon Simple Notification Service (SNS)

Configurar o AWS Mobile SDK para .NET e Xamarin

Você pode configurar o AWS Mobile SDK para .NET e Xamarin e começar a criar um novo projeto ou integrar o SDK a um projeto existente. Você também pode clonar e executar os exemplos para ter uma ideia de como funciona o SDK. Siga estas etapas para configurar e começar a usar o AWS Mobile SDK para .NET e Xamarin.

Pré-requisitos

Antes de usar o AWS Mobile SDK para .NET e Xamarin, você precisa fazer o seguinte:

- Crie uma conta da AWS.
- Instale o Xamarin.

Após concluir os pré-requisitos:

1. Obtenha credenciais da AWS usando o Amazon Cognito.
2. Defina as permissões necessárias para cada serviço da AWS usado no aplicativo.
3. Crie um novo projeto no IDE.
4. Instale o AWS Mobile SDK para .NET e Xamarin.
5. Configure o AWS Mobile SDK para .NET e Xamarin.

Etapa 1: Obter as credenciais da AWS

Para fazer chamadas para a AWS no aplicativo, primeiro obtenha as credenciais da AWS. Faça isso usando o Amazon Cognito, um serviço da AWS que permite que o aplicativo acesse os serviços do SDK sem precisar incorporar as credenciais privadas da AWS ao aplicativo.

Para começar a usar o Amazon Cognito, é necessário criar um banco de identidades. Um grupo de identidades é um armazenamento de informações específico da sua conta e identificado por um ID de grupo de identidades com a seguinte aparência:

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

1. Inicie a sessão no console do Amazon Cognito, selecione Manage Federated Identities (Gerenciar identidades federadas) e selecione Create new identity pool (Criar grupo de identidades).

2. Insira um nome para o grupo de identidades e marque a caixa de seleção para habilitar o acesso a identidades não autenticadas. Selecione Create Pool (Criar grupo) para criar o grupo de identidades.
3. Selecione Allow (Permitir) para criar as duas funções padrão associadas ao grupo de identidades, uma para usuários não autenticados e outra para usuários autenticados. Esses perfis padrão oferecem ao banco de identidades acesso à sincronização do Amazon Cognito e ao Amazon Mobile Analytics.

Normalmente, você só usa um grupo de identidades por aplicativo.

Após criar o grupo de identidades, você obterá as credenciais da AWS criando um objeto `CognitoAWSCredentials` (inserindo o ID do grupo de identidades) e inserindo-o no construtor de um cliente da AWS, da seguinte forma:

```
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
);
```

Etapa 2: Definir permissões

Defina permissões para cada serviço da AWS a ser usado no aplicativo. Primeiro, você precisa entender como a AWS visualiza os usuários do aplicativo.

Quando alguém usa o aplicativo e faz chamadas para a AWS, ela atribui uma identidade a esse usuário. O grupo de identidades criado na Etapa 1 é o local onde a AWS armazena essas identidades. Existem dois tipos de identidades: autenticados e não autenticados. As identidades autenticadas pertencem a usuários que serão autenticados por meio de um provedor de login público (por exemplo, Facebook, Amazon, Google). As identidades não autenticadas pertencem a usuários convidados.

Cada identidade está associada a uma AWS Identity and Access Management função. Na Etapa 1, você criou dois perfis do IAM, um para usuários autenticados e outro para usuários não autenticados. Cada perfil do IAM tem uma ou mais políticas anexadas a ele que especificam quais serviços da AWS as identidades atribuídas a esse perfil podem acessar. Por exemplo, a política de exemplo a seguir concede acesso a um bucket do Amazon S3:

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

Para definir permissões para os serviços da AWS que você deseja usar no aplicativo, modifique a política anexada aos perfis.

1. Acesse o console do IAM e selecione Roles (Funções). Digite o nome do grupo de identidades na caixa de pesquisa. Escolha o perfil do IAM que você deseja configurar. Se o aplicativo permitir usuários autenticados e não autenticados, você precisará conceder permissões para ambas as funções.
2. Clique em Attach Policy (Associar política), selecione a política desejada e clique em Attach Policy (Associar política). As políticas padrão dos perfis do IAM que você criou fornecem acesso ao Amazon Cognito e ao Mobile Analytics.

Para obter mais informações sobre como criar políticas ou escolher um item em uma lista de políticas, consulte Políticas do IAM.

Etapa 3: Criar um novo projeto da

Windows

Você pode usar o Visual Studio para desenvolver seu aplicativo.

OS X

Você pode usar o Visual Studio para desenvolver aplicativos. O desenvolvimento para iOS usando o Xamarin requer acesso a um Mac para executar o aplicativo. Para obter mais informações, consulte Como instalar o Xamarin.iOS no Windows.

> **Note**
>
> O IDE Rider comercial multiplataforma da JetBrains inclui suporte ao Xamarin nas plataformas Windows e Mac.

Etapa 4: Instalar o AWS Mobile SDK para .NET e Xamarin

Windows

Opção 1: Instalar usando o console do Package Manager

O AWS Mobile SDK para .NET e Xamarin consiste em um conjunto de assemblies .NET. Para instalar o AWS Mobile SDK para .NET e Xamarin, execute o comando install-package para cada pacote no console do Package Manager. Por exemplo, para instalar o Cognito Identity, execute o seguinte:

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

Os pacotes do AWS Core Runtime e do Amazon Cognito Identity são necessários em todos os projetos. Veja a seguir uma lista completa de nomes de pacotes para cada serviço.

Serviço	Nome do pacote
AWS Core Runtime	AWSSDK.Núcleo
Amazon Cognito Sync	AWSSDK.CognitoSync

Serviço	Nome do pacote
Identidade do Amazon Cognito	AWSSDK.CognitoIdentity
Amazon DynamoDB	AWSSDK.Dínamo DBv2
Amazon Mobile Analytics	AWSSDK.MobileAnalytics
Amazon S3	AWSSDK.S3
Amazon SNS	AWSSDK.SimpleNotificationService

Para incluir um pacote de pré-lançamento, inclua o argumento de linha de comando `-Pre` durante a instalação do pacote, da seguinte forma:

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

Você pode encontrar uma lista completa dos pacotes de serviços da AWS em Pacotes do AWS SDK em NuGet ou no AWS SDK for GitHub .NET Repository.

Opção 2: Instalar usando o IDE

No Visual Studio

1. Clique com o botão direito do mouse no projeto e, em seguida, clique em Gerenciar NuGet pacotes.
2. Procure o nome do pacote que você deseja adicionar ao projeto. Para incluir os NuGet pacotes de pré-lançamento, escolha Incluir pré-lançamento. Você pode encontrar uma lista completa dos pacotes de serviços da AWS em Pacotes do SDK da AWS em NuGet.
3. Escolha o pacote e, em seguida, escolha Install.

Mac (OS X)

No Visual Studio

1. Clique com o botão direito do mouse na pasta de pacotes e clique em Add Packages (Adicionar pacotes).

2. Procure o nome do pacote que você deseja adicionar ao projeto. Para incluir os pacotes de pré-lançamento, escolha Mostrar NuGet pacotes de pré-lançamento. Você pode encontrar uma lista completa dos pacotes de serviços da AWS em Pacotes do SDK da AWS em NuGet.
3. Marque a caixa de seleção ao lado do pacote desejado e selecione Add Package (Adicionar pacote).

> **Important**
>
> Se você estiver desenvolvendo usando uma Biblioteca de Classes Portátil, você também deve adicionar o AWSSDK NuGet pacote.Core a todos os projetos derivados da Biblioteca de Classes Portátil.

Etapa 5: Configurar o AWS Mobile SDK para .NET e Xamarin

Definição do registro

Definas as configurações de registro usando as classes `Amazon.AWSConfigs` e `Amazon.Util.LoggingConfig`. Você as encontrará no assembly `AWSSdk.Core`, disponível por meio do Nuget Package Manager no Visual Studio. Insira o código das configurações de registro no método `OnCreate` do arquivo `MainActivity.cs` para aplicativos Android ou do arquivo `AppDelegate.cs` para aplicativos iOS. Você também deve adicionar as instruções `using Amazon` e `using Amazon.Util` aos arquivos .cs.

Defina as configurações de registro da seguinte maneira:

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

Quando você faz login SystemDiagnostics, a estrutura imprime internamente a saída no System.Console. Se você deseja registrar respostas HTTP, defina o sinalizador `LogResponses`. Os valores podem ser Sempre, Nunca ou OnError.

Você também pode registrar as métricas de desempenho das solicitações HTTP usando a propriedade `LogMetrics`. O formato do log pode ser especificado através da propriedade `LogMetricsFormat`. Os valores válidos são JSON ou standard.

Definição do endpoint da região

Configure a região padrão para todos os clientes do serviço da seguinte maneira:

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

Isso define a região padrão para todos os clientes de serviços no SDK. Você pode substituir essa configuração especificando explicitamente a região no momento da criação de uma instância do cliente de serviço, da seguinte maneira:

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

Definição das configurações de proxy HTTP

Se a rede estiver atrás de um proxy, você poderá definir as configurações de proxy das solicitações HTTP da seguinte maneira.

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

Correção da distorção do relógio

Esta propriedade determina se o SDK corrigirá a distorção de relógio do cliente determinando a hora de servidor correta e emitindo novamente a solicitação com a hora correta.

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

Este campo será definido se uma chamada de serviço tiver resultado em uma exceção e o SDK tiver determinado que há uma diferença entre a hora local e a hora do servidor.

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

Para saber mais sobre distorção de relógio, consulte Correção de distorção de relógio no Blog da AWS.

Próximas etapas

Agora que configurou o AWS Mobile SDK para .NET e Xamarin, você pode:

- Comece a usar. Leia Conceitos básicos do AWS Mobile SDK para .NET e Xamarin para obter instruções de início rápido sobre como usar e configurar os serviços no AWS Mobile SDK para .NET e Xamarin.
- Explore os tópicos de serviço. Saiba mais sobre cada serviço e como ele funciona no AWS Mobile SDK para .NET e Xamarin.
- Execute as demonstrações. Visualize nossos aplicativos Xamarin de exemplo que demonstram casos de uso comuns. Para executar os aplicativos de exemplo, configure o AWS Mobile SDK para .NET e Xamarin conforme descrito anteriormente e, depois, siga as instruções contidas nos arquivos README de cada exemplo.
- Aprenda APIs o. Veja o | sdk-xamarin-ref |_.
- Faça perguntas: publique perguntas nos fóruns do AWS Mobile SDK ou abra um problema no GitHub.

Primeiros passos no AWS Mobile SDK para .NET e Xamarin

O AWS Mobile SDK para .NET e Xamarin fornece a documentação, as bibliotecas e os exemplos necessários para chamar serviços da AWS nos aplicativos Xamarin.

Você deve concluir todas as instruções fornecidas em Configuração do AWS Mobile SDK para .NET e Xamarin antes de começar a usar os serviços a seguir.

Esses tópicos de conceitos básicos orientarão você pelas seguintes tarefas:

Tópicos

- Armazenar e recuperar arquivos com o Amazon S3
- Sincronizar dados do usuário com Cognito Sync
- Armazene e Recupere Dados com o DynamoDB
- Rastreio de dados de uso do aplicativo com o Amazon Mobile Analytics
- Receber notificações por push usando o SNS (Xamarin iOS)
- Receba notificações por push usando o SNS (Xamarin Android)

Para obter informações sobre outros dispositivos móveis da AWS SDKs, consulte AWS Mobile SDK.

Armazenar e recuperar arquivos com o Amazon S3

O Amazon Simple Storage Service (Amazon S3) oferece aos desenvolvedores de dispositivos móveis e às equipes de TI um armazenamento de objetos seguro, durável e altamente escalável. O Amazon S3 é fácil de usar, com uma interface simples de serviços da web para armazenar e recuperar qualquer quantidade de dados de qualquer lugar da web.

O tutorial abaixo explica como integrar o S3 TransferUtility, um utilitário de alto nível para usar o S3 com seu aplicativo. Para obter mais informações sobre o uso do S3 nos aplicativos Xamarin, consulte o Amazon Simple Storage Service (S3).

Configuração do projeto

Pré-requisitos

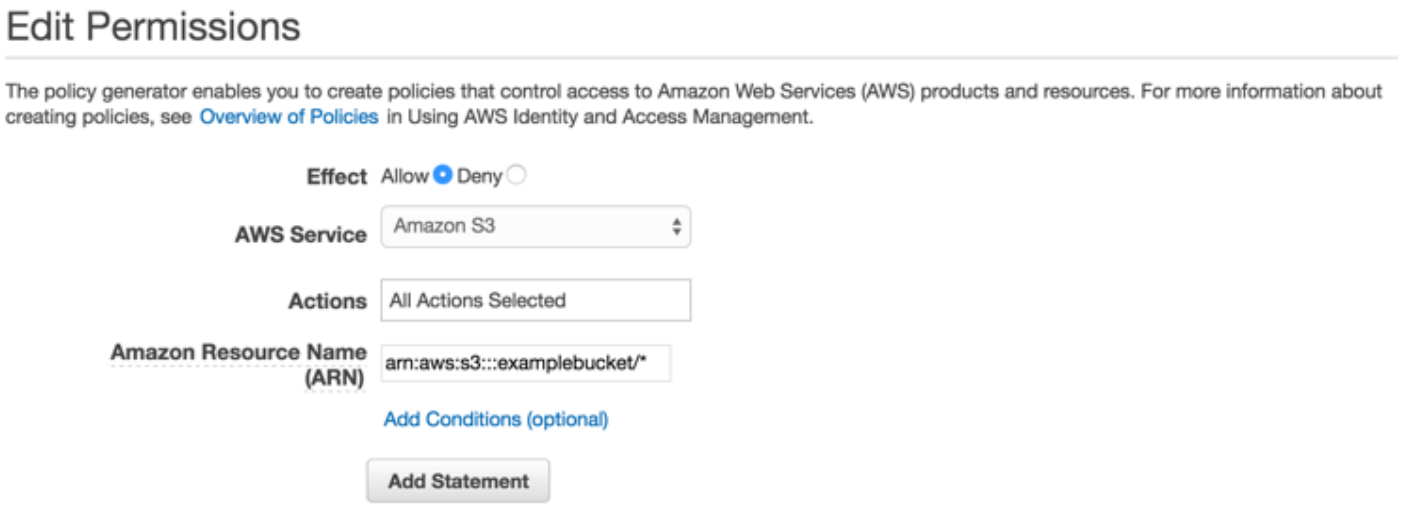
É necessário concluir todas as instruções na Configuração do AWS Mobile SDK para .NET e Xamarin antes de iniciar este tutorial.

Este tutorial também pressupõe que você já tenha criado um bucket do S3. Para criar um bucket do S3, acesse o console do AWS S3.

Definir permissões para o S3

A política padrão da função do IAM concede ao aplicativo acesso ao Amazon Mobile Analytics e ao Amazon Cognito Sync. Para que o grupo de identidades do Cognito acesse o Amazon S3, você deve modificar as funções do grupo de identidades.

1. Acesse o console do Identity and Access Management e clique em Roles (Funções) no painel à esquerda.
2. Digite o nome do grupo de identidades na caixa de pesquisa. Duas funções serão listadas: uma para os usuários autenticados e outra para os usuários não autenticados.
3. Clique na função para usuários não autenticados (ela terá "unauth" anexado ao nome do grupo de identidades).
4. Clique em Create Role Policy (Criar política de função), selecione Policy Generator (Gerador de políticas) e, em seguida, clique em Select (Selecionar).
5. Na página Edit Permissions (Editar permissões), insira as configurações mostradas na imagem a seguir, substituindo o Nome de recurso da Amazon (ARN) pelo seu nome. O ARN do bucket do S3 é semelhante ao `arn:aws:s3:::examplebucket/*` e composto pela região na qual o bucket está localizado e pelo nome do bucket. As configurações mostradas abaixo concederão ao grupo de identidades acesso total a todas as ações do bucket especificado.



1. Clique no botão Add Statement (Adicionar instrução) e em Next Step (Próxima etapa).
2. O assistente mostrará a você a configuração gerada. Clique em Apply Policy (Aplicar política).

Para obter mais informações sobre como conceder acesso ao S3, consulte Conceder acesso a um bucket do Amazon S3.

Adicione o NuGet Package for S3 ao seu projeto

Siga a Etapa 4 das instruções em Configurar o AWS Mobile SDK para .NET e Xamarin para NuGet adicionar o pacote S3 ao seu projeto.

(opcional) Configuração da versão de assinatura para solicitações do S3

Cada interação com o Amazon S3 é autenticada ou anônima. A AWS usa os algoritmos do Signature versão 4 ou Signature versão 2 para autenticar chamadas para o serviço.

Todas as novas regiões da AWS criadas após janeiro de 2014 são compatíveis apenas com o Signature versão 4. No entanto, muitas regiões mais antigas ainda oferecem suporte às solicitações do Signature versão 4 e do Signature versão 2.

Se seu bucket estiver em uma das regiões que não oferecem suporte às solicitações do Signature versão 2, conforme listado nesta página, você deverá definir o AWSConfigs S3. UseSignatureVersion4 propriedades para “verdadeiro” assim:

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

Para obter mais informações sobre as versões do AWS Signature, consulte Solicitações de autenticação (AWS Signature Version 4).

Inicializar o cliente S3 TransferUtility

Crie um cliente S3, transmitindo seu objeto de credenciais da AWS e, em seguida, transmita o cliente S3 para o utilitário de transferência, da seguinte forma:

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

Para fazer upload de um arquivo ao Amazon S3

Para fazer upload de um arquivo para o S3, chame `Upload` no objeto Transfer Utility, transmitindo os seguintes parâmetros:

- `file`: o nome da string do arquivo que você deseja fazer upload

- `bucketName`: o nome da string do bucket do S3 para armazenar o arquivo

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

O código acima pressupõe que há um arquivo no diretório Environment. SpecialFolder. ApplicationData. Os uploads usam automaticamente a funcionalidade multipart upload do S3 em arquivos grandes para aumentar a taxa de transferência.

Como fazer o download de um arquivo do Amazon S3

Para fazer o download de um arquivo do S3, chame `Download` no objeto Transfer Utility, transmitindo os seguintes parâmetros:

- `file`: o nome da string do arquivo que você deseja fazer download
- `bucketName`: o nome do bucket do S3 do qual você deseja fazer download do arquivo
- `key`: uma string representa o nome do objeto do S3 (neste caso, um arquivo) para download

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

Para obter mais informações sobre como acessar o Amazon S3 em um aplicativo Xamarin, consulte Amazon Simple Storage Service (S3).

Sincronizar dados do usuário com Cognito Sync

O Amazon Cognito Sync facilita salvar os dados móveis do usuário, como preferências de aplicativo ou estado do jogo na Nuvem AWS sem gravar qualquer código de back-end nem gerenciar qualquer infraestrutura. Você pode salvar dados localmente nos dispositivos dos usuários permitindo que os aplicativos funcionem até mesmo quando os dispositivos estiverem offline. Você também pode sincronizar os dados nos dispositivos de um usuário para que sua experiência de aplicativo seja consistente, independentemente do dispositivo usado.

O tutorial abaixo explica como integrar o Sync com seu aplicativo.

Configuração do projeto

Pré-requisitos

É necessário concluir todas as instruções na Configuração do AWS Mobile SDK para .NET e Xamarin antes de iniciar este tutorial.

Conceda acesso aos seus recursos do Cognito Sync

A política padrão associada às funções não autenticadas e autenticadas, que você criou durante a configuração, concede ao seu aplicativo acesso ao Cognito Sync. Nenhuma outra configuração é necessária.

Adicione o NuGet Package for Cognito Sync ao seu projeto

Siga a Etapa 4 das instruções em Configurar o AWS Mobile SDK para .NET e Xamarin para SyncManager NuGet adicionar o pacote Cognito ao seu projeto.

Inicialize o CognitoSyncManager

Transmita o seu provedor de credenciais do Amazon Cognito inicializado para o construtor `CognitoSyncManager`:

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

Sincronização de dados do usuário

Para sincronizar dados do usuário não autenticados:

1. Crie um conjunto de dados.
2. Adicione os dados do usuário ao conjunto de dados.
3. Sincronize o conjunto de dados com a nuvem.

Crie um conjunto de dados.

Crie uma instância de `Dataset`. O método openOrCreate Dataset é usado para criar um novo conjunto de dados ou abrir uma instância existente de um conjunto de dados armazenado localmente no dispositivo:

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

Adicione os dados do usuário ao conjunto de dados.

Os dados do usuário são adicionados na forma de key/value pares:

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

Os conjuntos de dados do Cognito funcionam como dicionários, com valores acessíveis por chave:

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

Sincronizar o conjunto de dados

Para sincronizar um conjunto de dados, chame o método de sincronização:

```
dataset.SynchronizeAsync();

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

Todos os dados gravados em conjuntos de dados serão armazenados localmente até que o conjunto de dados esteja sincronizado. O código nessa seção pressupõe que você esteja usando uma identidade do Cognito não autenticada e, por isso, quando os dados do usuário estiverem sincronizados com a nuvem, eles serão armazenados por dispositivo. O dispositivo tem um ID do dispositivo associado a ele. Quando os dados do usuário estiverem sincronizados com a nuvem, eles serão associados a esse ID do dispositivo.

Para obter mais informações sobre o Cognito Sync, consulte Amazon Cognito Sync.

Armazene e Recupere Dados com o DynamoDB

O Amazon DynamoDB é um serviço de banco de dados rápido, altamente disponível, altamente escalável, econômico e não relacional. O DynamoDB remove limitações de escalabilidade tradicionais sobre armazenamento de dados, mantendo, ao mesmo tempo, a baixa latência e o desempenho previsível.

O tutorial a seguir explica como integrar o DynamoDB Object Persistence Model com o seu aplicativo, que armazena objetos no DynamoDB.

Configuração do projeto

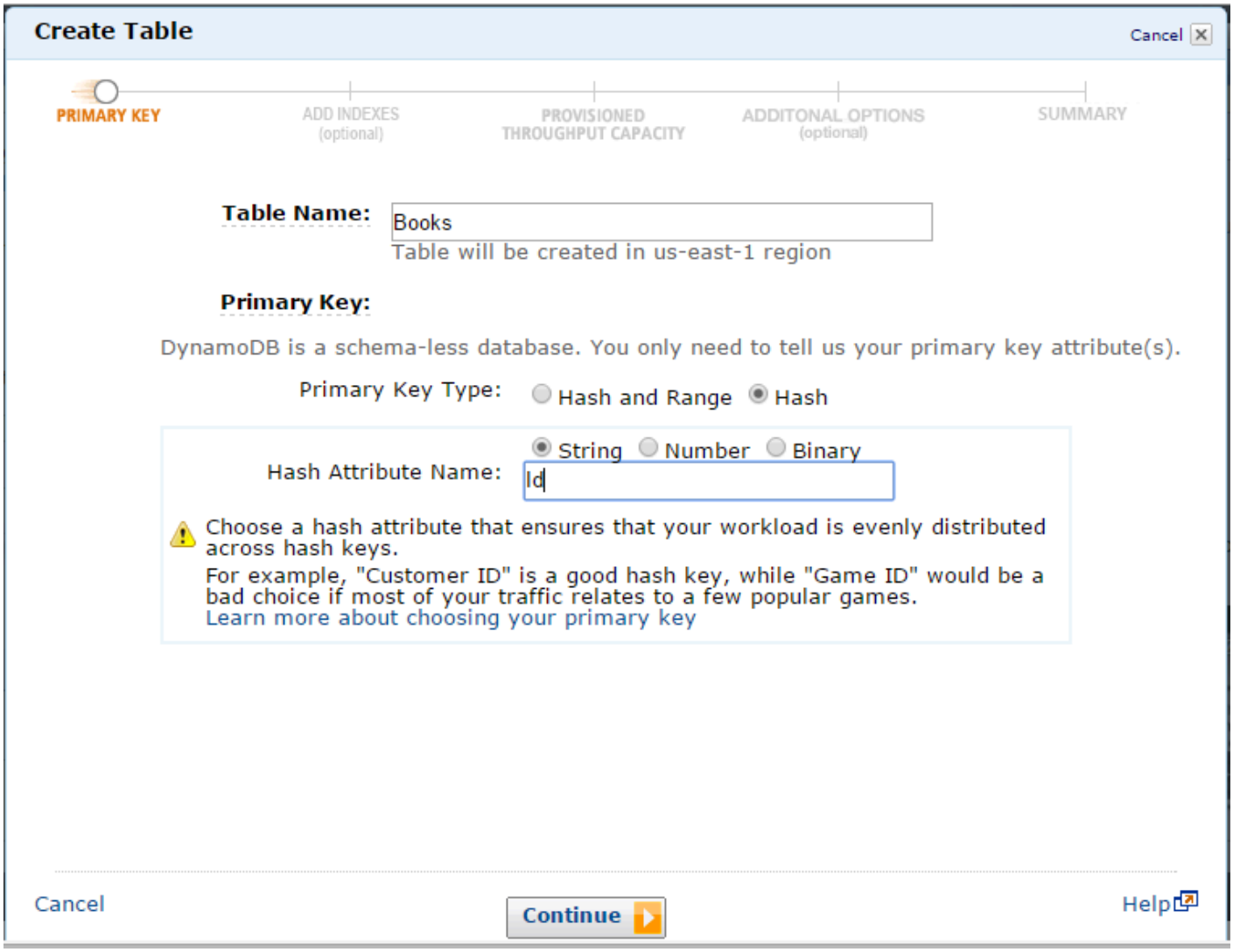
Pré-requisitos

É necessário concluir todas as instruções na Configuração do AWS Mobile SDK para .NET e Xamarin antes de iniciar este tutorial.

Criação de uma tabela do DynamoDB

Antes de ler e gravar dados em um banco de dados do DynamoDB, você deve criar uma tabela. Ao criar uma tabela, é preciso especificar a chave primária. A chave primária é composta por um atributo de hash e um atributo de intervalo. Para obter mais informações sobre como os atributos principal e de intervalo são usados, consulte Trabalho com tabelas.

1. Acesse o console do DynamoDB e clique em Create Table (Criar tabela). O assistente Criar Tabela será exibido.
2. Especifique o nome da tabela, o tipo de chave primária (Hash), e o nome do atributo de hash ("Id"), conforme mostrado abaixo e clique em Continue (Continuar):

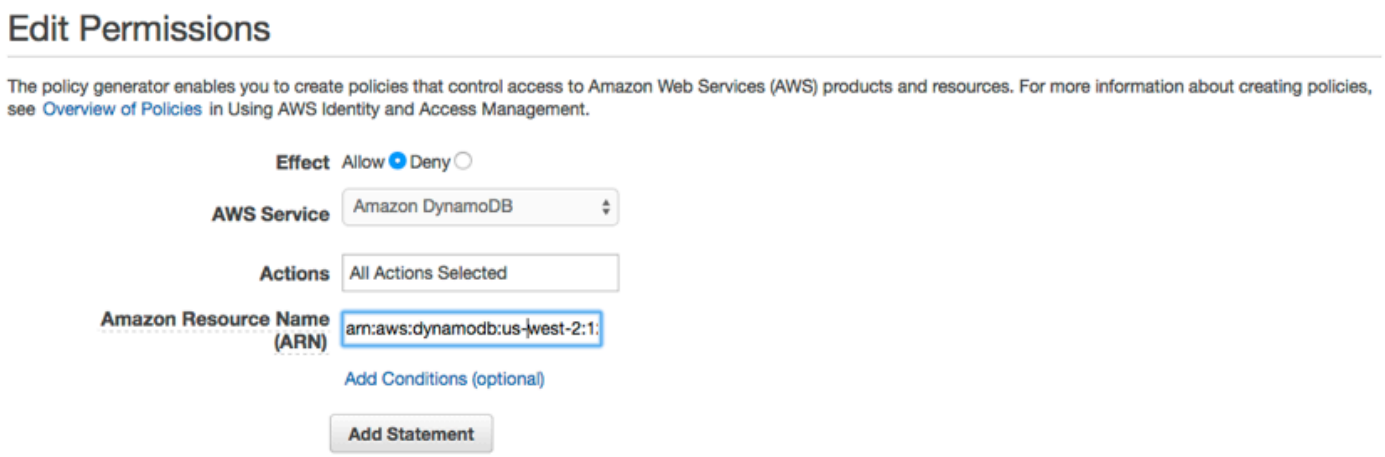


3. Deixe em branco os campos de edição da tela seguinte e clique em Continue (Continuar).
4. Aceite os valores padrão para Read Capacity Units (Unidades de capacidade de leitura) e Write Capacity Units (Unidades de capacidade de gravação) e clique em Continue (Continuar).
5. Na próxima tela, insira seu endereço de e-mail na caixa de texto Send notification to: (Enviar notificação para:) e clique em Continue (Continuar). A tela de revisão será exibida.
6. Clique em Criar. Pode demorar alguns minutos para a criação de sua tabela.

Definição de permissões para DynamoDB

Para que o seu grupo de identidades acesse o Amazon DynamoDB, você deve modificar as funções do grupo de identidades.

1. Navegue até o console do Identity and Access Management e clique em Roles (Funções) no painel à esquerda. Pesquise o nome do seu grupo de identidades. Duas funções serão listadas: uma para os usuários autenticados e outra para os não autenticados.
2. Clique na função para usuários não autenticados (terá "unauth" anexado ao nome do seu grupo de identidades) e clique em Create Role Policy (Criar política de função).
3. Selecione Policy Generator (Gerador de políticas_ e clique em Select (Selecionar).
4. Na página Edit Permissions (Editar permissões), insira as configurações mostradas na imagem a seguir. O nome de recurso da Amazon (ARN) de uma tabela do DynamoDB se parece com `arn:aws:dynamodb:us-west-2:123456789012:table/Books` e é composto pela região em que a tabela está localizada, pelo proprietário do número da conta da AWS e pelo nome da tabela no formato `table/Books`. Para obter mais informações sobre a especificação ARNs, consulte Amazon Resource Names for DynamoDB.



5. Clique em Add statement (Adicionar instrução)e Next Step (Próxima etapa). O assistente mostrará a você a configuração gerada.
6. Clique em Apply Policy (Aplicar política).

Adicione o NuGet pacote do DynamoDB ao seu projeto

Siga a etapa 4 das instruções em Configurar o AWS Mobile SDK para .NET e Xamarin para NuGet adicionar o pacote do DynamoDB ao seu projeto.

Inicializar AmazonDynamo DBClient

Passe seu provedor de credenciais inicializado do Amazon Cognito e sua região para `AmazonDynamoDB` o construtor e, em seguida, passe o cliente para o Dynamo: DBContext

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

Crie uma classe

Para gravar uma linha na tabela, defina uma classe para armazenar seus dados de linha. A classe também deve conter propriedades que contenham o atributo de dados referente à linha e que será mapeado para o DynamoDB Table criado no console. A declaração de classe a seguir ilustra a referida 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; }
}
```

Salvar um item

Para salvar um item, crie um objeto primeiro:

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

Em seguida, salve:

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

Para atualizar uma linha, modifique a instância da classe `DDTableRow` e convoque `AWSDynamoObjectMapper.save()`, conforme mostrado acima.

Recuperação de um item

Recupere um item usando sua chave primária:

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

Atualização de um item

Para atualizar um item:

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

Exclusão de um item

Para excluir um item:

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

Para obter mais informações sobre como acessar o DynamoDB em um aplicativo Xamarin, consulte Amazon DynamoDB.

Rastreio de dados de uso do aplicativo com o Amazon Mobile Analytics

O Amazon Mobile Analytics permite que você meça o uso e a receita do aplicativo. Ao rastrear tendências importantes, como usuários novos e usuários que voltam a usar o aplicativo, receita do aplicativo, retenção do usuário e eventos personalizados de comportamento do aplicativo, você poderá tomar decisões orientadas a dados para aumentar o envolvimento e a monetização do aplicativo.

O tutorial abaixo explica como integrar o Mobile Analytics ao aplicativo.

Configuração do projeto

Pré-requisitos

É necessário concluir todas as instruções na Configuração do AWS Mobile SDK para .NET e Xamarin antes de iniciar este tutorial.

Criação de um aplicativo no console do Mobile Analytics

Acesse o console do Amazon Mobile Analytics e crie um aplicativo. Observe o valor `appId`, pois ele será necessário em um passo posterior. Quando você estiver criando um aplicativo no console do Mobile Analytics, precisará especificar o ID do grupo de identidades. Para receber as instruções sobre como criar um grupo de identidades, consulte Configuração do AWS Mobile SDK para .NET e Xamarin.

Para saber mais informações sobre como trabalhar no console, consulte o Guia do usuário do Amazon Mobile Analytics.

Definição de permissões para Mobile Analytics

A política padrão associada às funções criadas durante a configuração concede ao seu aplicativo acesso ao Mobile Analytics. Nenhuma outra configuração é necessária.

Adicione o NuGet Package for Mobile Analytics ao seu projeto

Siga a Etapa 4 das instruções em Configurar o AWS Mobile SDK para .NET e Xamarin para adicionar o pacote Mobile NuGet Analytics ao seu projeto.

Definição das configurações do Mobile Analytics

O Mobile Analytics define algumas configurações que podem ser especificadas no arquivo awsconfig.xml:

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

- AllowUseDataNetwork - Um booleano que especifica se os eventos da sessão são enviados na rede de dados.
- DBWarningLimite - Esse é o limite do tamanho do banco de dados que, uma vez atingido, gerará registros de aviso.
- Máximo DBSize - Esse é o tamanho do SQLIte banco de dados. Quando o banco de dados atingir o tamanho máximo, todos os eventos adicionais serão removidos.
- MaxRequestSize - Esse é o tamanho máximo da solicitação em bytes que deve ser transmitida em uma solicitação HTTP para o serviço de análise móvel.
- SessionTimeout - É o intervalo de tempo após um aplicativo entrar em segundo plano e quando a sessão pode ser encerrada.

As configurações mostrados acima são os valores padrão de cada item de configuração.

Inicializar MobileAnalyticsManager

Para inicializar seu MobileAnalyticsManager, GetOrCreateInstance ligue para você`MobileAnalyticsManager`, informe suas credenciais da AWS, sua região, seu ID do aplicativo Mobile Analytics e seu objeto de configuração opcional:

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

Rastreio dos eventos de sessão

Xamarin Android

Substitua os métodos `OnPause()` e `OnResume()` da atividade para registrar eventos de sessão.

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

protected override void OnPause()
```

```
{
    manager.PauseSession();
    base.OnPause();
}
```

Isso precisa ser implementado em cada atividade do aplicativo.

Xamarin iOS

Em seu AppDelegate .cs:

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

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

Para obter mais informações sobre o Mobile Analytics, consulte Amazon Mobile Analytics.

Receber notificações por push usando o SNS (Xamarin iOS)

Este documento explica como enviar notificações por push a um aplicativo Xamarin iOS usando o Amazon Simple Notification Service (SNS) e o AWS Mobile SDK para .NET e Xamarin.

Configuração do projeto

Pré-requisitos

É necessário concluir todas as instruções na Configuração do AWS Mobile SDK para .NET e Xamarin antes de iniciar este tutorial.

Definir permissões para o SNS

Siga a etapa 2 em Configuração do AWS Mobile SDK para .NET e Xamarin para associar a política mencionada abaixo aos perfis do aplicativo. Isso concederá aplicativo as permissões adequadas para acessar o SNS:

1. Acesse o console do IAM e selecione a função do IAM a ser configurada.

2. Clique em Anexar política, selecione a política Amazon SNSFull Access e clique em Anexar política.

> **Warning**
>
> O uso do Amazon SNSFull Access não é recomendado em um ambiente de produção. Nós o utilizamos aqui para que você possa avançar rapidamente. Para obter mais informações sobre como especificar permissões para uma perfil do IAM, consulte Visão geral das permissões de perfis do IAM.

Obtenção da associação ao Apple iOS Developer Program

Você precisará executar o aplicativo em um dispositivo físico para receber notificações por push. Para executar o aplicativo em um dispositivo, é preciso ter uma associação ao Apple iOS Developer Program. Quando você tiver uma associação, poderá usar o Xcode para gerar uma identidade de assinatura. Para obter mais informações, consulte a documentação App Distribution Quick Start da Apple.

Criação de um certificado iOS

Primeiro, você precisa criar um certificado iOS. Em seguida, você precisará criar um perfil de provisionamento configurado para notificações por push. Para fazer isso:

1. Acesse o Apple Developer Member Center e clique em Certificates, Identifiers & Profiles (Certificados, identificadores e perfis).
2. Clique em Identifiers (Identificadores), em iOS Apps (Aplicativos iOS), clique no sinal de adição no canto superior direito da página da web para adicionar um novo ID do aplicativo iOS e insira uma descrição para o ID do aplicativo.
3. Role a tela para baixo até a seção Add ID Suffix (Adicionar sufixo ao ID) e selecione Explicit App ID (ID explícito do aplicativo) e insira o identificador do pacote.
4. Role para baixo até a seção App Services (Serviços do aplicativo) e, em seguida, selecione Push Notifications (Notificações por push).
5. Clique em Continue.
6. Clique em Enviar.
7. Clique em Done (Concluído).

8. Selecione o ID do aplicativo que você acabou de criar e clique em Edit (Editar).
9. Role para baixo até a seção Push Notifications (Notificações por push). Clique em Create Certificate (Criar certificado), em Development SSL Certificate (Certificado SSL para desenvolvimento).
10. Siga as instruções para criar a Certificate Signing Request (CSR - Solicitação de assinatura de certificado), fazer o upload da solicitação e fazer o download de um certificado SSL que será usado para se comunicar com o Apple Notification Service (APNS).
11. Retorne à página Certificates, Identifiers & Profiles (Certificados, identificadores e perfis). Clique em All (Tudo), em Provisioning Profiles (Perfis de provisionamento).
12. Clique no botão de adição no canto superior direito para adicionar um novo perfil de provisionamento.
13. Selecione iOS App Development (Desenvolvimento de aplicativo iOS) e clique em Continue (Continuar).
14. Selecione o ID do aplicativo e clique em Continue (Continuar).
15. Selecione o certificado do desenvolvedor e clique em Continue (Continuar).
16. Selecione o dispositivo e clique em Continue (Continuar).
17. Insira um nome de perfil e clique em Generate (Gerar).
18. Faça o download e clique duas vezes no arquivo de provisão para instalar o perfil de provisionamento.

Para obter mais informações sobre o provisionamento de um perfil configurado para receber notificações por push, consulte a documentação Configuração de notificações por push da Apple.

Uso do certificado para criar um Nome de região da Amazon (ARN) de plataforma no console do SNS

1. Execute o aplicativo de KeyChain acesso, selecione Meus certificados no canto inferior esquerdo da tela e clique com o botão direito do mouse no certificado SSL que você gerou para se conectar ao APNS e selecione Exportar. Você será solicitado a especificar um nome para o arquivo e uma senha para proteger o certificado. O certificado será salvo em um arquivo P12.
2. Acesse o console do SNS e clique em Applications (Aplicativos) no lado esquerdo da tela.
3. Clique em Create platform application (Criar aplicativo de plataforma) para criar um novo aplicativo para a plataforma SNS.
4. Insira um Application Name (Nome de aplicativo).

5. Selecione Apple Development (Desenvolvimento da Apple) em Push notification platform (Plataforma de notificação por push).
6. Clique em Choose File (Selecionar arquivo) e selecione o arquivo P12 criado ao exportar o certificado SSL.
7. Insira a senha especificada durante a exportação do certificado SSL e clique em Load Credentials From File (Carregar credenciais a partir do arquivo).
8. Clique em Create platform application (Criar aplicativo de plataforma).
9. Selecione o aplicativo de plataforma que você acabou de criar e copie o Nome de região da Amazon (ARN) do aplicativo. Você precisará dele nas etapas subsequentes.

Adicione o NuGet Package for SNS ao seu projeto

Siga a Etapa 4 das instruções em Configurar o AWS Mobile SDK para .NET e Xamarin para adicionar o pacote NuGet Amazon Simple Notification Service ao seu projeto.

Criação de um cliente SNS

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

Registro do aplicativo para notificações remotas

Para cadastrar um aplicativo, chame RegisterForRemoteNotifications seu UIApplication objeto, conforme mostrado abaixo. Coloque o código a seguir em AppDelegate .cs, inserindo o ARN do aplicativo da plataforma conforme solicitado abaixo:

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

Envio de uma mensagem do console do SNS para o endpoint

1. Acesse SNS Console (Console do SNS) > Applications (Aplicativos).
2. Selecione o aplicativo de plataforma, selecione um endpoint e clique em Publish to endpoint (Publicar no endpoint).
3. Digite uma mensagem de texto na caixa de texto e clique em Publish message (Publicar mensagem) para publicar uma mensagem.

Receba notificações por push usando o SNS (Xamarin Android)

O tutorial explica como enviar notificações por push a um aplicativo Xamarin Android usando o Amazon Simple Notification Service (SNS) e o AWS Mobile SDK para .NET e Xamarin.

Configuração do projeto

Pré-requisitos

É necessário concluir todas as instruções na Configuração do AWS Mobile SDK para .NET e Xamarin antes de iniciar este tutorial.

Definir permissões para o SNS

Siga a etapa 2 em Configuração do AWS Mobile SDK para .NET e Xamarin para associar a política mencionada abaixo aos perfis do aplicativo. Isso concederá aplicativo as permissões adequadas para acessar o SNS:

1. Acesse o console do IAM e selecione a função do IAM a ser configurada.
2. Clique em Anexar política, selecione a política Amazon SNSFull Access e clique em Anexar política.

> **Warning**
>
> O uso do Amazon SNSFull Access não é recomendado em um ambiente de produção. Nós o utilizamos aqui para que você possa avançar rapidamente. Para obter mais informações sobre como especificar permissões para uma perfil do IAM, consulte Visão geral das permissões de perfis do IAM.

Habilitar notificações por push no Google Cloud

Primeiro, adicione um novo projeto do Google API:

1. Acesse o console do Google Developers.
2. Clique em Create Project (Criar projeto).
3. Na caixa New Project (Novo projeto), insira um nome de projeto, anote o ID do projeto (você precisará dele mais tarde) e clique em Create (Criar).

Em seguida, habilite o serviço Google Cloud Messaging (GCM) do seu projeto:

1. No console do Google Developers, o novo projeto já estará selecionado. Se não estiver, selecione-o no menu suspenso na parte superior da página.
2. Selecione APIs e auth na barra lateral no lado esquerdo da página.
3. Na caixa de pesquisa, digite "Google Cloud Messaging para Android" e clique no link Google Cloud Messaging for Android (Google Cloud Messaging para Android).
4. Clique em Enable API (Permitir API).

Por fim, obtenha uma chave de API:

1. No Google Developers Console, selecione APIs & auth > Credenciais.
2. Em Public API access (Acesso à API pública), clique em Create new key (Criar nova chave).

3. Na caixa de diálogo Create a new key (Criar uma nova chave), clique em Server key (Chave do servidor).
4. Na caixa de diálogo resultante, clique em Create (Criar) e copie a chave da API exibida. Você a usará para realizar a autenticação posteriormente.

Uso do ID do projeto para criar um Nome de região da Amazon (ARN) de plataforma console do SNS

1. Acesse o console do SNS.
2. Clique em Applications (Aplicativos) no lado esquerdo da tela.
3. Clique em Create platform application (Criar aplicativo de plataforma) para criar um novo aplicativo para a plataforma SNS.
4. Insira um Application Name (Nome de aplicativo).
5. Selecione Google Cloud Messaging (GCM) em Push notification platform (Plataforma de notificação por push).
6. Cole a chave da API na caixa de texto intitulada API key (Chave da API).
7. Clique em Create platform application (Criar aplicativo de plataforma).
8. Selecione o aplicativo de plataforma que você acabou de criar e copie o Nome de região da Amazon (ARN) do aplicativo.

Adicione o NuGet Package for SNS ao seu projeto

Siga a Etapa 4 das instruções em Configurar o AWS Mobile SDK para .NET e Xamarin para adicionar o pacote NuGet Amazon Simple Notification Service ao seu projeto.

Criação de um cliente SNS

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

Registro do aplicativo para notificações remotas

Para se registrar para receber notificações remotas no Android, você precisará criar uma BroadcastReceiver que possa receber mensagens do Google Cloud. Altere o nome do pacote abaixo quando solicitado:

```
[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);
      }
}
```

Abaixo está o serviço que recebe a notificação push do BroadcastReceiver e exibe a notificação na barra de notificação do 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());
  }
}
```

Envio de uma mensagem do console do SNS para o endpoint

1. Acesse SNS Console (Console do SNS) > Applications (Aplicativos).
2. Selecione o aplicativo de plataforma, selecione um endpoint e clique em Publish to endpoint (Publicar no endpoint).

3. Digite uma mensagem de texto na caixa de texto e clique em Publish message (Publicar mensagem) para publicar uma mensagem.

Amazon Cognito Identity

O que é o Amazon Cognito Identity?

O Amazon Cognito Identity permite a criação de identidades exclusivas para os usuários e autenticá-los com provedores de identidade. Com uma identidade, você pode obter credenciais da AWS temporárias e de privilégio limitado para sincronizar dados com a sincronização do Amazon Cognito, ou acessar diretamente outros serviços da AWS. O Amazon Cognito Identity é compatível com provedores de identidade públicos, como o Amazon, o Facebook e o Google, além de identidades não autenticadas. Ele é também compatível com as identidades autenticadas do desenvolvedor, que permitem a você registrar e autenticar usuários por meio de seu próprio processo de autenticação de backend.

Para obter mais informações sobre o Cognito Identity, consulte o Guia do desenvolvedor do Amazon Cognito.

Para obter informações sobre a disponibilidade da região de autenticação do Cognito, consulte a Disponibilidade de regiões do serviço da AWS.

Usando um provedor público para autenticar usuários

Usando o Amazon Cognito Identity, é possível criar identidades exclusivas para os seus usuários e autentique-os para proteger o acesso aos seus recursos da AWS, como o Amazon S3 ou o Amazon DynamoDB. O Amazon Cognito Identity é compatível com provedores de identidade públicos, como o Amazon, o Facebook e o Google, ou qualquer provedor compatível com o OpenID Connect, além de identidades não autenticadas.

Para obter informações sobre como usar provedores públicos de identidade como o Amazon, Facebook, Twitter/Digits, ou o Google para autenticar usuários, consulte os Provedores externos no guia de Desenvolvedor do Amazon Cognito.

Uso de Identidades autenticadas pelo desenvolvedor

O Amazon Cognito oferece suporte às identidades autenticadas pelo desenvolvedor, além da federação de identidades da web por meio do Facebook, do Google e da Amazon. Com as identidades autenticadas pelos desenvolvedores, você pode registrar e autenticar usuários por meio do processo de autenticação existente, sem deixar de usar o Amazon Cognito Sync para sincronizar

os dados do usuário e acessar os recursos da AWS. O uso de identidades autenticadas pelo desenvolvedor engloba a interação entre o dispositivo do usuário final, o backend para autenticação e o Amazon Cognito.

Para obter informações sobre as identidades autenticadas do desenvolvedor, consulte as Identidades autenticadas pelo desenvolvedor no Guia do desenvolvedor do Amazon Cognito.

Amazon Cognito Sync

O que é o Amazon Cognito Sync?

O Cognito Sync é um serviço da AWS e a biblioteca de clientes que permite a sincronização dos dados do usuário entre dispositivos (por exemplo, resultados de jogos, preferências do usuário, estado do jogo). Você pode usar a API do Cognito Sync para sincronizar os dados do usuário entre dispositivos. Para usar o Cognito Sync no aplicativo, inclua o | no projeto.

Para obter instruções sobre como integrar o Amazon Cognito Sync ao seu aplicativo, consulte Guia do desenvolvedor do Amazon Cognito Sync.

Amazon Mobile Analytics

O Amazon Mobile Analytics é um serviço para coletar, visualizar, compreender e extrair dados de uso do aplicativo em escala. O Mobile Analytics captura facilmente dados do dispositivo padrão e eventos personalizados, além de calcular automaticamente relatórios em seu nome. Além dos relatórios agregados listados abaixo, também é possível configurar os dados a serem exportados automaticamente para o Redshift e o S3, para análise posterior.

Com o uso do Amazon Mobile Analytics, é possível acompanhar os comportamentos dos clientes, agregar métricas, gerar visualizações de dados, além de identificar padrões significativos.

Principais conceitos

Tipos de relatórios

O Mobile Analytics oferece as seguintes alternativas de relatórios no Mobile Analytics Console:

- Daily Active Users (Usuário ativo diariamente - DAU), Monthly Active Users (Usuário ativo mensalmente - MAU) e novos usuários
- Fator de adesão (DAU divididos pelos MAU)
- Número de sessões e Média de sessões para cada Usuário ativo diariamente
- Receita média para de Usuário ativo diariamente (ARPDAU) e Receita média de cada Usuário ativo e pagante diariamente (ARPPDAU)
- Retenção de um, três e sete dias e Retenção de uma, duas e três semanas
- Eventos personalizados

Esses relatórios são fornecidos por meio de seis abas de relatórios no console:

- Visão geral — Acompanhe nove relatórios pré-selecionados em um simple-to-review painel para ter uma ideia rápida do engajamento: MAU, DAU, novos usuários, sessões diárias, fator fixo, retenção de 1 dia, ARPDAU, usuários pagantes diários, ARPPDAU.
- Usuários ativos – Rastreie o número de usuários que interagem com seu aplicativo diariamente e mensalmente, além de monitorar o fator de adesão em relação ao engajamento, envolvimento, à conquista de novos usuários e à monetização.

- Sessões – Acompanhe a frequência com que seu aplicativo foi usado em determinado dia e a frequência com que cada usuário abre o seu aplicativo no decorrer de um dia.
- Retenção – Acompanhe a taxa de retorno dos clientes em relação ao aplicativo, diária e semanalmente.
- Receita – Acompanhe as tendências de receitas no aplicativo para identificar as áreas onde a monetização pode melhorar.
- Eventos personalizados – Acompanhe as ações personalizadas e definidas pelo usuário, específicas para o seu aplicativo.

Para saber mais sobre os relatórios do Mobile Analytics e sobre como trabalhar no console do Mobile Analytics, consulte a Visão geral dos relatórios do console do Mobile Analytics no Guia do desenvolvedor do Mobile Analytics.

Configuração do projeto

Pré-requisitos

Para usar o Mobile Analytics em seu aplicativo, você precisará adicionar o SDK ao seu projeto. Para fazer isso, siga as instruções em Configuração do AWS Mobile SDK para .NET e Xamarin.

Definição das configurações do Mobile Analytics

O Mobile Analytics define algumas configurações que podem ser especificadas no arquivo awsconfig.xml:

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

- SessionTimeout- Se o aplicativo permanecer em segundo plano por um tempo maior do que SessionTimeout o cliente do Mobile Analytics encerrará a sessão atual e uma nova sessão será criada quando o aplicativo voltar ao primeiro plano. Recomendamos o uso de valores que variam de 5 a 10. O valor padrão é 5.

- Máximo DBSize - O tamanho máximo do banco de dados (em bytes) usado para armazenamento local de eventos. Se o tamanho do banco de dados exceder esse valor, os eventos adicionais serão ignorados. Recomendamos o uso de valores que variam de 1MB a 10MB. O valor padrão é 5242880. (5MB).
- DBWarningLimite - O limite de aviso. Os valores válidos variam entre 0 – 1. Se os valores excederem o limite, serão gerados logs de aviso. O valor padrão é 0.9.
- MaxRequestSize- O tamanho máximo de uma solicitação HTTP feita ao serviço Mobile Analytics. O valor é especificado em bytes e pode variar entre 1 e 512KB. O valor padrão é os 102400 (100KB). Não use valores superiores a 512 KB. Isso pode fazer com que o serviço rejeite a solicitação HTTP.
- AllowUseDataNetwork- Um valor que indica se a chamada de serviço é permitida em uma rede de dados celular. Use essa opção com cautela, pois ela pode aumentar o uso de dados do cliente.

As configurações mostrados acima são os valores padrão de cada item de configuração.

Integração do Mobile Analytics com o seu aplicativo

As seções abaixo explicam como integrar o Mobile Analytics ao aplicativo.

Criação de um aplicativo no console do Mobile Analytics

Acesse o console do Amazon Mobile Analytics e crie um aplicativo. Observe o valor `appId`, pois ele será necessário em um passo posterior. Quando você estiver criando um aplicativo no console do Mobile Analytics, precisará especificar o ID do grupo de identidades. Para receber as instruções sobre como criar um grupo de identidades, consulte Configuração do AWS Mobile SDK para .NET e Xamarin.

Para saber mais sobre como trabalhar no Console do Mobile Analytics, consulte a Visão geral dos relatórios do console do Mobile Analytics no Guia do desenvolvedor do Mobile Analytics.

Crie um MobileAnalyticsManager cliente

Para inicializar seu MobileAnalyticsManager, GetOrCreateInstance ligue para você`MobileAnalyticsManager`, informe suas credenciais da AWS, sua região, seu ID do aplicativo Mobile Analytics e seu objeto de configuração opcional:

```
// Initialize the MobileAnalyticsManager
analyticsManager = MobileAnalyticsManager.GetOrCreateInstance(
```

```
  cognitoCredentials,
  RegionEndpoint.USEast1,
  APP_ID,
  config
);
```

O `APP_ID` será gerado para você durante o assistente de criação de aplicativos. Os dois valores devem corresponder aos usados no console do Mobile Analytics. O `APP_ID` é usado para agrupar seus dados no console do Mobile Analytics. Para localizar o ID d aplicativo após criar o aplicativo no console do Mobile Analytics, navegue até o console do Mobile Analytics e clique no ícone de engrenagem no canto superior direito da tela. Isso exibirá a página de gerenciamento de aplicativos, que lista todos os aplicativos registrados e seus aplicativos IDs.

Registro de eventos de monetização

O AWS Mobile SDK para .NET e Xamarin fornece a classe `MonetizationEvent`, o que permite gerar eventos de monetização para rastrear as compras feitas em aplicativos móveis. O trecho de código a seguir mostra como criar um evento de monetização:

```
// 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);
```

Registro de eventos personalizados

O Mobile Analytics permite que você defina eventos personalizados. Os eventos personalizados são definidos inteiramente por você; eles ajudam a rastrear ações dos usuários específicas do seu aplicativo ou jogo. Para obter mais informações sobre os eventos personalizados, consulte Eventos personalizados.

Neste exemplo, digamos que o nosso aplicativo seja um jogo e que queremos registrar um evento quando um determinado usuário concluir um nível. Crie um evento “LevelComplete” criando uma nova `AmazonMobileAnalyticsEvent` instância:

```
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);
```

Registro de sessões

Xamarin iOS

Quando o aplicativo perde o foco, é possível pausar a sessão. Para aplicativos iOS, no AppDelegate arquivo.cs, substitua `DidEnterBackground` e `WillEnterForeground` para chamar `MobileAnalyticsManager.PauseSesion` e `MobileAnalyticsManager.ResumeSession` conforme mostrado no seguinte trecho:

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

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

Xamarin Android

Para aplicativos Android, chame `MobileAnalyticsManager.PauseSesion` o método OnPause () e `MobileAnalyticsManager.ResumeSession` o método OnResume () conforme mostrado no seguinte trecho de código:

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

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

Por padrão, se o usuário tirar o foco do aplicativo por menos de cinco segundos e retornar novamente ao aplicativo, a sessão será retomada. Se o usuário tirar o foco do aplicativo por cinco segundos ou mais, uma nova sessão será criada. Esta configuração é configurável no arquivo de configuração aws_mobile_analytics.json “SESSION_DELTA”, ao definir a propriedade referente ao número de segundos de espera antes de criar uma nova sessão.

Amazon Simple Storage Service (S3)

O que é o S3?

O Amazon Simple Storage Service (Amazon S3) oferece aos desenvolvedores um armazenamento de objetos seguro, duradouro e altamente dimensionável. O Amazon S3 é fácil de usar, com uma interface simples de serviços da web para armazenar e recuperar qualquer quantidade de dados de qualquer lugar da web. Com o Amazon S3, você paga apenas pelo armazenamento realmente utilizado. Não há taxa mínima nem custo de configuração.

O Amazon S3 oferece um armazenamento de objetos econômico para uma ampla gama de casos de uso, incluindo aplicativos de nuvem, distribuição de conteúdo, backup e arquivamento, recuperação de desastres e análise de big data.

Para obter informações sobre a disponibilidade de regiões do AWS S3, consulte Disponibilidade de regiões do Serviço da AWS.

Principais conceitos

Bucket

Cada objeto armazenado no Amazon S3 reside em um bucket. Você pode usar buckets para agrupar objetos relacionados da mesma forma como usa um diretório para agrupar arquivos em um sistema de arquivos. Os buckets têm propriedades, como permissões de acesso e status de versionamento, e você pode especificar a região em que eles devem residir.

Para saber mais sobre os buckets do S3, consulte Trabalho com buckets no Guia do desenvolvedor do S3.

Objetos

Os objetos são os dados que você armazena no Amazon S3. Cada objeto reside em um bucket criado em uma região específica da AWS.

Os objetos armazenados em uma região nunca saem dela, a não ser que você os transfira explicitamente para outra região. Por exemplo, os objetos armazenados na região da UE (Irlanda) nunca saem dela. Os objetos armazenados em uma região do Amazon S3 permanecem fisicamente nessa região. O Amazon S3 não mantém cópias nem as move para nenhuma outra região.

No entanto, você pode acessar os objetos de qualquer lugar, desde que tenha as permissões necessárias.

Os objetos podem ser qualquer tipo de arquivo: imagens, dados de backup, filmes etc. Um objeto pode ter até 5 TB. Você pode ter um número ilimitado de objetos em um bucket.

Para que possa fazer upload de um objeto no Amazon S3, você deverá ter permissões de gravação em um bucket. Para obter mais informações sobre como definir permissões de bucket, consulte Editar permissões do bucket no Guia do desenvolvedor do S3.

Para saber mais sobre os objetos do S3, consulte Trabalho com objetos no Guia do desenvolvedor do S3.

Metadados do objeto

Cada objeto no Amazon S3 tem um conjunto de pares de chave-valor que representa seus metadados. Existem dois tipos de metadados:

- Metadados de sistema – Processados ocasionalmente pelo Amazon S3; por exemplo, Content-Type e Content-Length.
- Metadados de usuário – Nunca processado pelo Amazon S3. Os metadados do usuário são armazenados com o objeto e retornados com ele. O tamanho máximo dos metadados do usuário é 2 KB; e tanto as chaves como os seus valores devem estar em conformidade com os padrões US-ASCII.

Para saber mais sobre os metadados de objeto do S3, consulte Editar metadados de objeto.

Configuração do projeto

Pré-requisitos

Para usar o Amazon S3 no aplicativo, você precisará adicionar o SDK ao projeto. Para fazer isso, siga as instruções em Configuração do AWS Mobile SDK para .NET e Xamarin.

Criar um bucket do S3

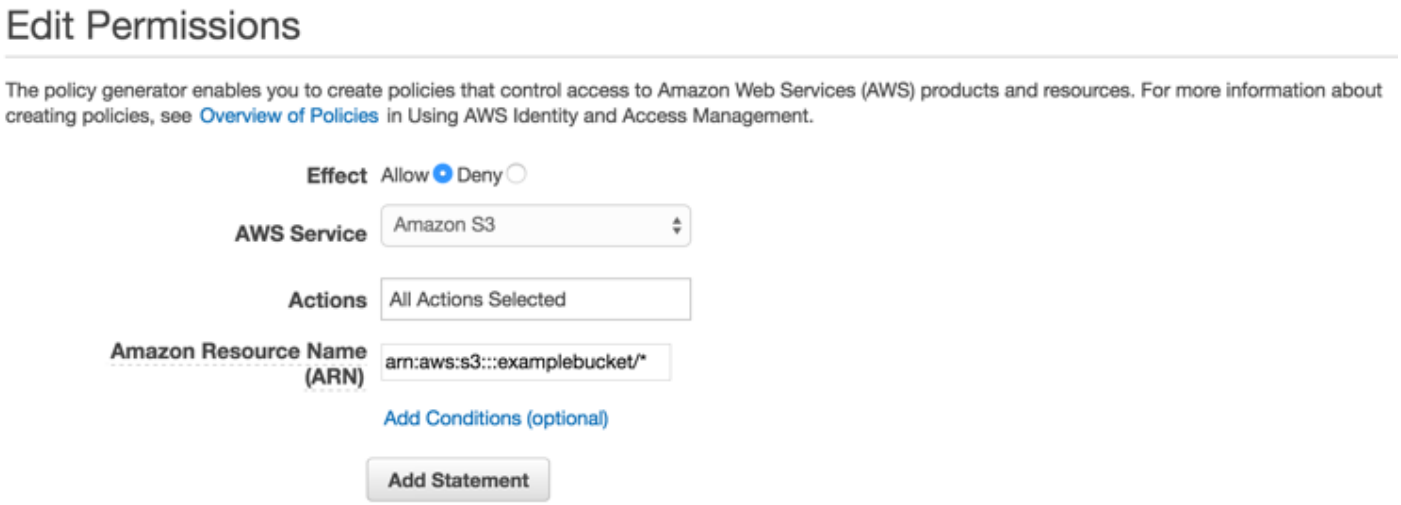
O Amazon S3 armazena os recursos do aplicativo em buckets do Amazon S3, contêineres de armazenamento na nuvem que residem em uma região específica. Cada bucket do Amazon S3 deve ter um nome globalmente exclusivo. Você pode usar o console do Amazon S3 para criar um bucket.

1. Faça login no console do Amazon S3 e clique em Create Bucket (Criar bucket).
2. Insira um nome de bucket, selecione uma região e clique em Create (Criar).

Definir permissões para o S3

A política padrão da função do IAM concede ao aplicativo acesso ao Amazon Mobile Analytics e ao Amazon Cognito Sync. Para que o grupo de identidades do Cognito acesse o Amazon S3, você deve modificar as funções do grupo de identidades.

1. Acesse o console do Identity and Access Management e clique em Roles (Funções) no painel à esquerda.
2. Digite o nome do grupo de identidades na caixa de pesquisa. Duas funções serão listadas: uma para os usuários autenticados e outra para os usuários não autenticados.
3. Clique na função para usuários não autenticados (ela terá "unauth" anexado ao nome do grupo de identidades).
4. Clique em Create Role Policy (Criar política de função), selecione Policy Generator (Gerador de políticas) e, em seguida, clique em Select (Selecionar).
5. Na página Edit Permissions (Editar permissões), insira as configurações mostradas na imagem a seguir, substituindo o Nome de recurso da Amazon (ARN) pelo seu nome. O ARN do bucket do S3 é semelhante ao `arn:aws:s3:::examplebucket/*` e composto pela região na qual o bucket está localizado e pelo nome do bucket. As configurações mostradas abaixo concederão ao grupo de identidades acesso total a todas as ações do bucket especificado.



1. Clique no botão Add Statement (Adicionar instrução) e em Next Step (Próxima etapa).

2. O assistente mostrará a você a configuração gerada. Clique em Apply Policy (Aplicar política).

Para obter mais informações sobre como conceder acesso ao S3, consulte Conceder acesso a um bucket do Amazon S3.

(opcional) Configuração da versão de assinatura para solicitações do S3

Cada interação com o Amazon S3 é autenticada ou anônima. A AWS usa os algoritmos do Signature versão 4 ou Signature versão 2 para autenticar chamadas para o serviço.

Todas as novas regiões da AWS criadas após janeiro de 2014 são compatíveis apenas com o Signature versão 4. No entanto, muitas regiões mais antigas ainda oferecem suporte às solicitações do Signature versão 4 e do Signature versão 2.

Se seu bucket estiver em uma das regiões que não oferecem suporte às solicitações do Signature versão 2, conforme listado nesta página, você deverá definir o AWSConfigs S3. UseSignatureVersion4 propriedades para “verdadeiro” assim:

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

Para obter mais informações sobre as versões do AWS Signature, consulte Solicitações de autenticação (AWS Signature Version 4).

Integração do S3 ao aplicativo

Há duas maneiras de interagir com o S3 no aplicativo Xamarin. Esses dois métodos são descritos detalhadamente nos tópicos a seguir:

Uso do S3 Transfer Utility

O S3 Transfer Utility facilita o envio e o download de arquivos para o S3 a partir do aplicativo Xamarin.

Inicialize o TransferUtility

Crie um cliente S3, transmitindo seu objeto de credenciais da AWS e, em seguida, transmita o cliente S3 para o utilitário de transferência, da seguinte forma:

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

```
var transferUtility = new TransferUtility(s3Client);
```

(opcional) Configure o TransferUtility

Há três propriedades opcionais que você pode configurar:

- ConcurrentServiceRequests- Determina quantos threads ativos ou o número de solicitações web assíncronas simultâneas serão usadas no arquivo. upload/download O valor padrão é 10.
- MinSizeBeforePartUpload- Obtém ou define o tamanho mínimo da peça para fazer upload de peças em bytes. O padrão é 16 MB. A redução do tamanho mínimo da parte produz multipart uploads para serem divididos em um número maior de partes menores. Definir esse valor muito baixo tem um efeito negativo nas velocidades de transferência, produzindo latência extra e comunicação de rede para cada parte.
- NumberOfUploadThreads- Obtém ou define o número de threads em execução. Essa propriedade determina o número de threads ativos que serão usados para fazer o upload do arquivo. O valor padrão é 10 threads.

Para configurar o TransferUtility cliente S3, crie um objeto de configuração, defina suas propriedades e passe o objeto para o seu TransferUtility construtor da seguinte forma:

```
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);
```

Fazer download de um arquivo

Para fazer o download de um arquivo do S3, chame `Download` no objeto Transfer Utility, transmitindo os seguintes parâmetros:

- `file`: o nome da string do arquivo que você deseja fazer download
- `bucketName`: o nome do bucket do S3 do qual você deseja fazer download do arquivo
- `key`: uma string representa o nome do objeto do S3 (neste caso, um arquivo) para download

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

Fazer upload de um arquivo

Para fazer upload de um arquivo para o S3, chame `Upload` no objeto Transfer Utility, transmitindo os seguintes parâmetros:

- `file`: o nome da string do arquivo que você deseja fazer upload
- `bucketName`: o nome da string do bucket do S3 para armazenar o arquivo

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

O código acima pressupõe que há um arquivo no diretório Environment. SpecialFolder. ApplicationData. Os uploads usam automaticamente a funcionalidade multipart upload do S3 em arquivos grandes para aumentar a taxa de transferência.

Usando o nível de serviço S3 APIs

Além de usar o S3 TransferUtility, você também pode interagir com o S3 usando o S3 de baixo nível. APIs

Inicialização do cliente Amazon S3

Para usar o Amazon S3, primeiro precisamos criar uma instância do AmazonS3Client que faça referência à instância do Cognito que você criou anteriormente e à sua regiãoAWSCredentials :

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

Fazer download de um arquivo

Para fazer download de um arquivo no 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);
    }
}
```

Fazer upload de um arquivo

Para fazer upload de um arquivo para o 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);
```

Exclusão de um item

Para excluir um item no 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);
```

Exclusão de vários itens

Para excluir vários objetos em um bucket usando uma única solicitação 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);
    }
}
```

Você pode especificar até 1000 chaves.

Listar buckets

Para retornar uma lista de todos os buckets pertencentes ao remetente autenticado da solicitação:

```
// 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);
}
```

Listar objetos

Você pode retornar alguns ou todos os (até 1000) objetos armazenados no bucket do S3. Para fazer isso, você deve ter acesso de leitura ao 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);
  }
}
```

Obtenção da região de um bucket

Para obter a região em que um bucket reside:

```
// 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);
```

Obtenção da política de um bucket

Para obter a política de um bucket:

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

// Construct request
```

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

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

Amazon DynamoDB

O que é o Amazon DynamoDB?

O Amazon DynamoDB é um serviço de banco de dados não relacional rápido e altamente dimensionável. O DynamoDB remove limitações de escalabilidade tradicionais sobre armazenamento de dados, mantendo, ao mesmo tempo, a baixa latência e o desempenho previsível.

Principais conceitos

Os conceitos do modelo de dados do DynamoDB incluem tabelas, itens e atributos.

Tabelas

No Amazon DynamoDB, um banco de dados é uma coleção de tabelas. Uma tabela é uma coleção de itens, e cada item é uma coleção de atributos.

Em um banco de dados relacional, uma tabela tem um schema predefinido, como nome da tabela, chave primária, lista de seus nomes de coluna e seus tipos de dados. Todos os registros armazenados na tabela devem ter o mesmo conjunto de colunas. Por outro lado, o DynamoDB só requer que uma tabela tenha uma chave primária, mas não requer que você defina todos os nomes de atributo e tipos de dados com antecedência.

Para saber mais sobre como trabalhar com tabelas, consulte Trabalhar com tabelas no DynamoDB.

Itens e atributos

Os itens individuais em uma tabela do DynamoDB pode ter qualquer número de atributos, embora haja um limite de 400 KB para o tamanho do item. Um tamanho de item é a soma dos tamanhos de seus nomes e valores de atributo (tamanhos binários e UTF-8).

Cada atributo em um item é um par de nome-valor. Um atributo pode ser um conjunto de valor único ou multivalor. Por exemplo, um item de livro pode ter os atributos title e authors. Cada livro tem um título, mas pode ter vários autores. O atributo multivalor é um conjunto; não são permitidos valores duplicados.

Por exemplo, é recomendável armazenar um catálogo de produtos no DynamoDB. Você pode criar uma tabela, ProductCatalog, com o atributo Id como chave primária. A chave primária identifica exclusivamente cada item, para que não haja dois produtos na tabela com o mesmo ID.

Para saber mais sobre como trabalhar com itens, consulte Trabalhar com itens no DynamoDB.

Tipos de dados

O Amazon DynamoDB é compatível com os seguintes tipos de dados:

- Tipos escalares – Número, string, binário, booleano e nulo.
- Tipos de vários valores – Conjunto de strings, conjunto de números e conjunto binário.
- Tipos de documento – Lista e mapa.

Para obter mais informações sobre tipos de dados escalares, tipos de dados de vários valores e tipos de dados de documento, consulte Tipos de dados do DynamoDB.

Chave primária

Ao criar uma tabela, além do nome dela, você deve especificar a chave primária da tabela. A chave primária identifica exclusivamente cada item na tabela, de modo que não possa haver dois itens com a mesma chave. O DynamoDB é compatível com os seguintes tipos de chaves primárias:

- Chave hash: a chave primária é composta por um atributo, um atributo de hash. O DynamoDB cria um índice de hash não ordenado neste atributo de chave primária. Cada item da tabela é identificado exclusivamente por seu valor de chave de hash.
- Chave hash e chave de intervalo: a chave primária é composta por dois atributos. O primeiro atributo é o atributo de hash; o segundo atributo é o atributo de intervalo. O DynamoDB cria um índice de hash não ordenado no atributo de chave primária de hash e um índice de intervalo classificado no atributo de chave primária de intervalo. Cada item da tabela é identificado exclusivamente pela combinação de seus valores de chave de hash e de intervalo. Dois itens podem ter o mesmo valor de chave de hash, mas eles devem ter valores de chave de intervalo diferentes.

Índices secundários

Quando você criar uma tabela com uma chave de hash e uma chave de intervalo, poderá definir um ou mais índices secundários nessa tabela, se desejar. Um índice secundário permite consultar os dados na tabela usando uma chave alternativa, além de consultas com base na chave primária.

O DynamoDB oferece suporte a dois tipos de índices secundários: índices secundários locais e índices secundários globais.

- Índice secundário local: um índice que possui a mesma chave hash que a tabela, mas uma chave de intervalo diferente.
- Índice secundário global: um índice com uma chave hash e uma chave de intervalo que podem ser diferentes das contidas na tabela.

Você pode definir até cinco índices secundários globais e cinco índices secundários locais por tabela. Para obter mais informações, consulte Como melhorar o acesso a dados com índices secundários no DynamoDB no Guia do desenvolvedor do DynamoDB.

Consulta e verificação

Além de usar chaves primárias para acessar itens, o Amazon DynamoDB também fornece APIs duas para pesquisar os dados: consulta e digitalização. É recomendável que você leia Orientações para consulta e verificação no Guia do desenvolvedor do DynamoDB Developer Guide para se familiarizar com algumas práticas recomendadas.

Consulta

A operação Query localiza os itens em uma tabela ou um índice secundário usando somente valores de atributo de chave primária. Você deve fornecer um nome de atributo de chave de hash e um valor distinto a serem procurados. Se desejar, você pode fornecer um nome de atributo de chave de intervalo e um valor, e usar um operador de comparação para refinar os resultados da pesquisa.

Para consultas de exemplo, consulte:

- Uso do modelo de documento
- Uso do modelo de persistência de objeto
- Usando o nível de serviço do DynamoDB APIs

Para obter mais informações sobre consultas, consulte Consulta no Guia do desenvolvedor do DynamoDB.

Verificar

Uma operação Scan lê cada item de uma tabela ou de um índice secundário. Por padrão, uma operação Scan retorna todos os atributos de dados de cada item na tabela ou no índice. Você pode usar o ProjectionExpression parâmetro para que Scan retorne apenas alguns dos atributos, em vez de todos eles.

Para verificações de exemplo, consulte:

- Uso do modelo de documento
- Uso do modelo de persistência de objeto
- Usando o nível de serviço do DynamoDB APIs

Para obter mais informações sobre verificações, consulte Verificação no Guia do desenvolvedor do DynamoDB.

Configuração do projeto

Pré-requisitos

Para usar o DynamoDB em seu aplicativo, você precisará adicionar o SDK ao projeto. Para fazer isso, siga as instruções em Configuração do AWS Mobile SDK para .NET e Xamarin.

Criação de uma tabela do DynamoDB

Para criar uma tabela, acesse o console do DynamoDB e siga estas etapas:

1. Clique em Create Table (Criar tabela).
2. Informe o nome da tabela.
3. Selecione Hash como tipo de chave primária.
4. Selecione um tipo e insira um valor para o nome do atributo de hash. Clique em Continue.
5. Na página Add Indexes (Adicionar índices), se você pretende usar os índices secundários globais, defina Index Type (Tipo de índice) como "Global Secondary Index" (Índice secundário global) e, em Index Hash Key (Chave hash do índice), insira uma valor para o índice secundário. Isso permitirá que você faça operações de consulta e verificação usando os índices primário e secundário. Clique em Add Index To Table (Adicionar índice à tabela) e em Continue (Continuar). Para ignorar o uso de índices secundários globais, clique em Continue (Continuar).
6. Defina a capacidade de leitura e gravação para os níveis desejados. Para obter mais informações sobre como configurar a capacidade, consulte Throughput provisionado no Amazon DynamoDB. Clique em Continue.
7. Na próxima tela, insira um e-mail de notificação para criar alarmes de taxa de transferência, se desejado. Clique em Continue.

8. Na página de resumo, clique em Create (Criar). O DynamoDB criará seu banco de dados.

Definição de permissões para DynamoDB

Para usar o DynamoDB em um aplicativo, defina as permissões corretas. A seguinte política do IAM permite que o usuário exclua, obtenha, insira, consulte, verifique e atualize itens em uma tabela específica do DynamoDB, que é identificada pelo ARN:

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

Você pode modificar as políticas no console do IAM. Você deve adicionar ou remover ações permitidas com base nas necessidades do seu aplicativo.

Para saber mais sobre as políticas do IAM, consulte Usar o IAM.

Para saber mais sobre as políticas específicas do DynamoDB, consulte Usar o IAM para controlar o acesso a recursos do DynamoDB no Guia do desenvolvedor do DynamoDB.

Integração do DynamoDB ao aplicativo

O AWS Mobile SDK para .NET e Xamarin fornece uma biblioteca de alto nível para trabalhar com o DynamoDB. Você também pode fazer solicitações diretamente com base na API de nível inferior do DynamoDB, mas, na maioria dos casos de uso, é recomendável o uso da biblioteca de nível superior. AmazonDynamoDBClient É uma parte especialmente útil da biblioteca de alto nível.

Usando essa classe, você pode realizar várias operações de criação, leitura, atualização e exclusão (CRUD) e executar consultas.

O AWS Mobile SDK para .NET e Xamarin permite que você APIs faça chamadas usando o AWS SDK para.NET para trabalhar com o DynamoDB. Todos APIs eles estão disponíveis no AWSSDK domínio.dll. Para obter mais informações sobre como fazer download do AWS SDK para .NET, consulte AWS SDK para .NET.

Há três formas de interagir com o DynamoDB no aplicativo Xamarin:

- Modelo de documento: esta API oferece classes wrapper sobre a API de nível inferior do DynamoDB para simplificar ainda mais as tarefas de programação. Table e Document são as principais classes wrapper. Você pode usar o modelo de documento nas operações de dados, como criar, recuperar, atualizar e excluir itens. A API está disponível no Amazon.DynamoDB. DocumentModel namespace.
- Modelo de persistência de objeto: a API Object Persistence permite mapear classes do lado do cliente para as tabelas do DynamoDB. Em seguida, cada instância de objeto é mapeada para um item nas tabelas correspondentes. A DBContext classe Dynamo nessa API fornece métodos para você salvar objetos do lado do cliente em uma tabela, recuperar itens como objetos e realizar consultas e escaneamentos. Você pode usar o modelo de persistência de objeto nas operações de dados, como criar, recuperar, atualizar e excluir itens. Primeiro, crie as tabelas usando a API Service Client e, em seguida, use o modelo de persistência de objeto para mapear as classes para as tabelas. A API está disponível no Amazon.DynamoDB. DataModel namespace.
- API Service Client: esta é a API de nível de protocolo que é mapeada estreitamente para a API do DynamoDB. Você pode usar essa API de nível inferior em todas as operações de tabela e item, como criar, atualizar e excluir tabela e itens. Você também pode consultar e verificar as tabelas. Esta API está disponível no namespace Amazon.DynamoDB.

Esses três modelos são descritos detalhadamente nos tópicos a seguir:

Uso do modelo de documento

O modelo de documentos oferece cursos em todo o nível inferior de API do .NET. Table e Document são as principais classes wrapper. Você pode usar o modelo de documento para criar, recuperar, atualizar e excluir itens. Para criar, atualizar e excluir tabelas, você deve usar a API de nível inferior. Para obter instruções sobre como usar a API de baixo nível, consulte Como usar o nível de serviço

do DynamoDB. APIs A API de baixo nível está disponível no Amazon.dynamoDB. DocumentModel namespace.

Para saber mais sobre o Modelo de documento, consulte Modelo de documento do .NET.

Criação de um cliente DynamoDB

Para criar um cliente DynamoDB:

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

Operações de CRUD

Salvar um item

Crie um item:

```
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";
```

Salve um item em uma tabela do DynamoDB:

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

Recuperação de um item

Para recuperar um item:

```
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);
}
```

Atualização de um item

Para atualizar um item:

```
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);
}
```

Para atualizar um item de forma condicional:

```
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);
}
```

Exclusão de um item

Para excluir um item:

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

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

Consulta e verificação

Para consultar e recuperar todos os livros cujo autor seja "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);
  });
}
```

O código de exemplo de verificação abaixo retorna todos os livros de nossa tabela:

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

Uso do modelo de persistência de objetos

O AWS Mobile SDK para .NET e Xamarin fornece um modelo de persistência de objetos que permite mapear classes do cliente para uma tabela do DynamoDB. Cada instância do objeto, então, mapeia para um item na tabela correspondente. Para salvar seus objetos do lado do cliente em uma tabela,

o modelo de persistência de objetos fornece a DBContext classe Dynamo, um ponto de entrada para o DynamoDB. Esta classe fornece uma conexão ao DynamoDB e permite que você acesse tabelas, execute várias operações CRUD e realize consultas.

O modelo de persistência de objetos não fornece uma API para criar, atualizar ou excluir tabelas. Ele fornece apenas operações de dados. Para criar, atualizar e excluir tabelas, você deve usar a API de nível inferior. Para obter instruções sobre como usar a API de baixo nível, consulte Como usar o nível de serviço do DynamoDB. APIs

Visão geral do

O modelo de persistência de objetos fornece um conjunto de atributos para mapear classes do lado do cliente para tabelas e properties/fields atributos de tabela. O modelo de persistência de objetos é compatível com o mapeamento explícito e o padrão entre as propriedades das classes e os atributos da tabela.

- Mapeamento explícito: para mapear uma propriedade para uma chave primária, você deve usar os atributos do modelo Dynamo DBHash Key e Dynamo DBRange Key Object Persistence. Além disso, para os atributos de chave não primária, se o nome de uma propriedade em sua classe e o atributo de tabela correspondente para o qual você deseja mapeá-la não forem iguais, você deverá definir o mapeamento adicionando explicitamente o atributo do DBProperty Dynamo.
- Mapeamento padrão – Por padrão, o modelo de persistência de objetos mapeia as propriedades de classe para os atributos com o mesmo nome na tabela.

Você não precisa mapear todas as propriedades de classe. Você identifica essas propriedades adicionando o DBIgnore atributo do Dynamo. Salvar e recuperar uma instância de um objeto pode omitir qualquer propriedade marcada com este atributo.

Tipos de dados compatíveis

O modelo de persistência de objeto é compatível com um conjunto de tipos de dados primitivos do .NET, com a coleta dos mesmos e com os tipos de dados arbitrários. O modelo é compatível com os seguintes tipos de dados primitivos.

- bool
- byte
- char
- DateTime

- decimal, duplo, float
- Int16, Int32, Int64
- SByte
- string
- UInt16, UInt32, UInt64

O modelo de persistência de objetos .NET também é compatível com os tipos de coleta que contêm as seguintes limitações:

- O tipo de coleção deve implementar a ICollection interface.
- O tipo de Coleta deve ser composto por tipos primitivos compatíveis. Por exemplo, ICollection<string>, ICollection<bool>.
- O tipo de coleta deve fornecer um construtor sem parâmetro.

Para obter mais informações sobre o modelo de persistência de objetos, consulte o Modelo de persistência de objetos do .NET.

Criação de um cliente DynamoDB

Para criar um cliente DynamoDB:

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

Operações de CRUD

Salvar um objeto

Criar um objeto:

```
[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
};
```

Salve um objeto em uma tabela do DynamoDB:

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

Recuperar um objeto

Para recuperar um objeto:

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

Atualizar um objeto

Para atualizar um objeto:

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

Excluir um objeto

Para excluir um objeto:

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

Consulta e verificação

Para consultar e recuperar todos os livros cujo autor seja "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());
  });
}
```

O código de exemplo de verificação abaixo retorna todos os livros de nossa tabela:

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

Usando o nível de serviço do DynamoDB APIs

O nível de serviço do Dynamo APIs permite que você crie, atualize e exclua tabelas. Você também pode realizar operações típicas de criação, leitura, atualização e exclusão (CRUD) nos itens de uma tabela usando essa API.

Criação de um cliente DynamoDB

Para criar um cliente DynamoDB:

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

Operações de CRUD

Salvar um item

Para salvar um item em uma tabela do 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);
```

Recuperação de um item

Para recuperar um item:

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

Atualização de um item

Para atualizar um item:

```
// 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);
```

Exclusão de um item

Para excluir um item:

```
// 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);
```

Consulta e verificação

Para consultar e recuperar todos os livros cujo autor seja "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);
    });
```

```
  }
}
```

O código de exemplo de verificação abaixo retorna todos os livros de nossa tabela:

```
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)

Ao usar o SNS e o AWS Mobile SDK, você pode criar aplicativos que podem receber notificações móveis por push. Para obter mais informações sobre o SNS, consulte Amazon Simple Notification Service

Principais conceitos

O Amazon SNS permite que aplicativos e usuários finais em diferentes dispositivos recebam notificações por meio de notificações push móveis (dispositivos Apple, Google e Kindle Fire), HTTP/HTTPS, Email/Email -JSON, SMS ou filas Amazon Simple Queue Service (SQS) ou funções do AWS Lambda. O SNS permite enviar mensagens individuais ou distribuir mensagens para um grande número de destinatários inscritos em um único tópico.

Tópicos

Um tópico é um "ponto de acesso" que permite aos destinatários se inscrever dinamicamente para receber cópias idênticas da mesma notificação. Um tópico pode dar suporte a entregas para vários tipos de endpoints, por exemplo, é possível agrupar destinatários de iOS, Android e SMS.

Assinaturas

Para receber mensagens publicadas em um tópico, você precisa inscrever um endpoint no tópico em questão. Um endpoint é um aplicativo móvel, servidor da web, endereço de e-mail ou uma fila do Amazon SQS que pode receber mensagens de notificação do Amazon SNS. Quando você inscrever um endpoint em um tópico e a inscrição for confirmada, o endpoint receberá todas as mensagens publicadas nesse tópico.

Publicação

Quando você publica em um tópico, o SNS entrega cópias adequadamente formatadas da sua mensagem para cada assinante desse tópico. Para notificações por push móvel, você pode publicar diretamente no endpoint ou inscrever o endpoint em um tópico.

Configuração do projeto

Pré-requisitos

Para usar o SNS em seu aplicativo, você precisará adicionar o SDK ao seu projeto. Para fazer isso, siga as instruções em Configuração do AWS Mobile SDK para .NET e Xamarin.

Definir permissões para o SNS

Para obter informações sobre a configuração de permissões para o SNS, consulte Gerenciar o acesso aos seus tópicos do Amazon SNS.

Adicione o NuGet Package for SNS ao seu projeto

Siga a Etapa 4 das instruções em Configurar o AWS Mobile SDK para .NET e Xamarin para adicionar o pacote NuGet Amazon Simple Notification Service ao seu projeto.

Integração do SNS ao seu aplicativo

Há muitas maneiras de interagir com o SNS em seu aplicativo Xamarin:

Enviar notificações por push (Xamarin Android)

Este documento explica como enviar notificações por push a um aplicativo Xamarin Android usando o Amazon Simple Notification Service (SNS) e o AWS Mobile SDK para .NET e Xamarin.

Configuração do projeto

Pré-requisitos

É necessário concluir todas as instruções na Configuração do AWS Mobile SDK para .NET e Xamarin antes de iniciar este tutorial.

Definir permissões para o SNS

Siga a etapa 2 em Configuração do AWS Mobile SDK para .NET e Xamarin para associar a política mencionada abaixo aos perfis do aplicativo. Isso concederá aplicativo as permissões adequadas para acessar o SNS:

1. Acesse o console do IAM e selecione a função do IAM a ser configurada.

2. Clique em Anexar política, selecione a política Amazon SNSFull Access e clique em Anexar política.

> **Warning**
>
> O uso do Amazon SNSFull Access não é recomendado em um ambiente de produção. Nós o utilizamos aqui para que você possa avançar rapidamente. Para obter mais informações sobre como especificar permissões para uma perfil do IAM, consulte Visão geral das permissões de perfis do IAM.

Habilitar notificações por push no Google Cloud

Primeiro, adicione um novo projeto do Google API:

1. Acesse o console do Google Developers.
2. Clique em Create Project (Criar projeto).
3. Na caixa New Project (Novo projeto), insira um nome de projeto, anote o ID do projeto (você precisará dele mais tarde) e clique em Create (Criar).

Em seguida, habilite o serviço Google Cloud Messaging (GCM) do seu projeto:

1. No console do Google Developers, o novo projeto já estará selecionado. Se não estiver, selecione-o no menu suspenso na parte superior da página.
2. Selecione APIs e auth na barra lateral no lado esquerdo da página.
3. Na caixa de pesquisa, digite "Google Cloud Messaging para Android" e clique no link Google Cloud Messaging for Android (Google Cloud Messaging para Android).
4. Clique em Enable API (Permitir API).

Por fim, obtenha uma chave de API:

1. No Google Developers Console, selecione APIs & auth > Credenciais.
2. Em Public API access (Acesso à API pública), clique em Create new key (Criar nova chave).
3. Na caixa de diálogo Create a new key (Criar uma nova chave), clique em Server key (Chave do servidor).

4. Na caixa de diálogo resultante, clique em Create (Criar) e copie a chave da API exibida. Você a usará para realizar a autenticação posteriormente.

Uso do ID do projeto para criar um Nome de região da Amazon (ARN) de plataforma console do SNS

1. Acesse o console do SNS.
2. Clique em Applications (Aplicativos) no lado esquerdo da tela.
3. Clique em Create platform application (Criar aplicativo de plataforma) para criar um novo aplicativo para a plataforma SNS.
4. Insira um Application Name (Nome de aplicativo).
5. Selecione Google Cloud Messaging (GCM) em Push notification platform (Plataforma de notificação por push).
6. Cole a chave da API na caixa de texto intitulada API key (Chave da API).
7. Clique em Create platform application (Criar aplicativo de plataforma).
8. Selecione o aplicativo de plataforma que você acabou de criar e copie o Nome de região da Amazon (ARN) do aplicativo.

Adicione o NuGet Package for SNS ao seu projeto

Siga a Etapa 4 das instruções em Configurar o AWS Mobile SDK para .NET e Xamarin para adicionar o pacote NuGet Amazon Simple Notification Service ao seu projeto.

Criação de um cliente SNS

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

Registro do aplicativo para notificações remotas

Para se registrar para receber notificações remotas no Android, você precisará criar uma BroadcastReceiver que possa receber mensagens do Google Cloud. Altere o nome do pacote abaixo quando solicitado:

```
[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);
      }
}
```

Abaixo está o serviço que recebe a notificação push do BroadcastReceiver e exibe a notificação na barra de notificação do 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());
  }
}
```

Envio de uma mensagem do console do SNS para o endpoint

1. Acesse SNS Console (Console do SNS) > Applications (Aplicativos).
2. Selecione o aplicativo de plataforma, selecione um endpoint e clique em Publish to endpoint (Publicar no endpoint).
3. Digite uma mensagem de texto na caixa de texto e clique em Publish message (Publicar mensagem) para publicar uma mensagem.

Envio de notificações por push (Xamarin iOS)

Este documento explica como enviar notificações por push a um aplicativo Xamarin iOS usando o Amazon Simple Notification Service (SNS) e o AWS Mobile SDK para .NET e Xamarin.

Configuração do projeto

Pré-requisitos

É necessário concluir todas as instruções na Configuração do AWS Mobile SDK para .NET e Xamarin antes de iniciar este tutorial.

Definir permissões para o SNS

Siga a etapa 2 em Configuração do AWS Mobile SDK para .NET e Xamarin para associar a política mencionada abaixo aos perfis do aplicativo. Isso concederá aplicativo as permissões adequadas para acessar o SNS:

1. Acesse o console do IAM e selecione a função do IAM a ser configurada.
2. Clique em Anexar política, selecione a política Amazon SNSFull Access e clique em Anexar política.

> **Warning**
>
> O uso do Amazon SNSFull Access não é recomendado em um ambiente de produção. Nós o utilizamos aqui para que você possa avançar rapidamente. Para obter mais informações sobre como especificar permissões para uma perfil do IAM, consulte Visão geral das permissões de perfis do IAM.

Obtenção da associação ao Apple iOS Developer Program

Você precisará executar o aplicativo em um dispositivo físico para receber notificações por push. Para executar o aplicativo em um dispositivo, é preciso ter uma associação ao Apple iOS Developer Program. Quando você tiver uma associação, poderá usar o Xcode para gerar uma identidade de assinatura. Para obter mais informações, consulte a documentação App Distribution Quick Start da Apple.

Criação de um certificado iOS

Primeiro, você precisa criar um certificado iOS. Em seguida, você precisará criar um perfil de provisionamento configurado para notificações por push. Para fazer isso:

1. Acesse o Apple Developer Member Center e clique em Certificates, Identifiers & Profiles (Certificados, identificadores e perfis).
2. Clique em Identifiers (Identificadores), em iOS Apps (Aplicativos iOS), clique no sinal de adição no canto superior direito da página da web para adicionar um novo ID do aplicativo iOS e insira uma descrição para o ID do aplicativo.
3. Role a tela para baixo até a seção Add ID Suffix (Adicionar sufixo ao ID) e selecione Explicit App ID (ID explícito do aplicativo) e insira o identificador do pacote.
4. Role para baixo até a seção App Services (Serviços do aplicativo) e, em seguida, selecione Push Notifications (Notificações por push).
5. Clique em Continue.
6. Clique em Enviar.
7. Clique em Done (Concluído).
8. Selecione o ID do aplicativo que você acabou de criar e clique em Edit (Editar).
9. Role para baixo até a seção Push Notifications (Notificações por push). Clique em Create Certificate (Criar certificado), em Development SSL Certificate (Certificado SSL para desenvolvimento).
10. Siga as instruções para criar a Certificate Signing Request (CSR - Solicitação de assinatura de certificado), fazer o upload da solicitação e fazer o download de um certificado SSL que será usado para se comunicar com o Apple Notification Service (APNS).
11. Retorne à página Certificates, Identifiers & Profiles (Certificados, identificadores e perfis). Clique em All (Tudo), em Provisioning Profiles (Perfis de provisionamento).
12. Clique no botão de adição no canto superior direito para adicionar um novo perfil de provisionamento.
13. Selecione iOS App Development (Desenvolvimento de aplicativo iOS) e clique em Continue (Continuar).
14. Selecione o ID do aplicativo e clique em Continue (Continuar).
15. Selecione o certificado do desenvolvedor e clique em Continue (Continuar).
16. Selecione o dispositivo e clique em Continue (Continuar).
17. Insira um nome de perfil e clique em Generate (Gerar).

18. Faça o download e clique duas vezes no arquivo de provisão para instalar o perfil de provisionamento.

Para obter mais informações sobre o provisionamento de um perfil configurado para receber notificações por push, consulte a documentação Configurar notificações por push da Apple.

Uso do certificado para criar um Nome de região da Amazon (ARN) de plataforma no console do SNS

1. Execute o aplicativo de KeyChain acesso, selecione Meus certificados no canto inferior esquerdo da tela e clique com o botão direito do mouse no certificado SSL que você gerou para se conectar ao APNS e selecione Exportar. Você será solicitado a especificar um nome para o arquivo e uma senha para proteger o certificado. O certificado será salvo em um arquivo P12.
2. Acesse o console do SNS e clique em Applications (Aplicativos) no lado esquerdo da tela.
3. Clique em Create platform application (Criar aplicativo de plataforma) para criar um novo aplicativo para a plataforma SNS.
4. Insira um Application Name (Nome de aplicativo).
5. Selecione Apple Development (Desenvolvimento da Apple) em Push notification platform (Plataforma de notificação por push).
6. Clique em Choose File (Selecionar arquivo) e selecione o arquivo P12 criado ao exportar o certificado SSL.
7. Insira a senha especificada durante a exportação do certificado SSL e clique em Load Credentials From File (Carregar credenciais a partir do arquivo).
8. Clique em Create platform application (Criar aplicativo de plataforma).
9. Selecione o aplicativo de plataforma que você acabou de criar e copie o Nome de região da Amazon (ARN) do aplicativo. Você precisará dele nas etapas subsequentes.

Adicione o NuGet Package for SNS ao seu projeto

Siga a Etapa 4 das instruções em Configurar o AWS Mobile SDK para .NET e Xamarin para adicionar o pacote NuGet Amazon Simple Notification Service ao seu projeto.

Criação de um cliente SNS

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

Registro do aplicativo para notificações remotas

Para cadastrar um aplicativo, chame RegisterForRemoteNotifications seu UIApplication objeto, conforme mostrado abaixo. Coloque o código a seguir em AppDelegate .cs, inserindo o ARN do aplicativo da plataforma conforme solicitado abaixo:

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

Envio de uma mensagem do console do SNS para o endpoint

1. Acesse SNS Console (Console do SNS) > Applications (Aplicativos).
2. Selecione o aplicativo de plataforma, selecione um endpoint e clique em Publish to endpoint (Publicar no endpoint).

3. Digite uma mensagem de texto na caixa de texto e clique em Publish message (Publicar mensagem) para publicar uma mensagem.

Enviar e receber notificações SMS

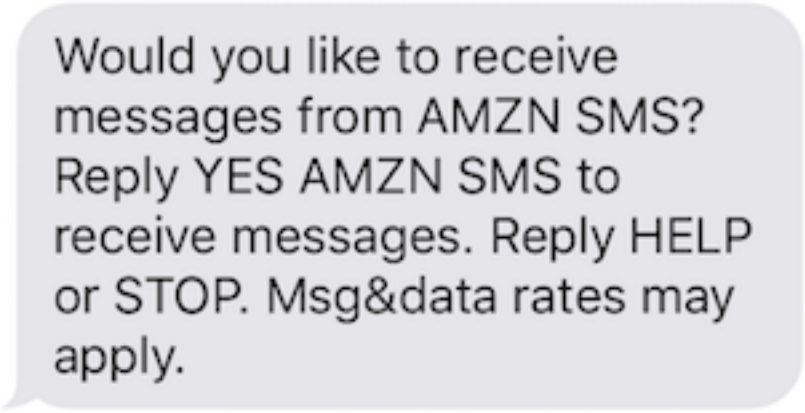
Você pode usar o Amazon SNS (Amazon Simple Notification Service) para enviar e receber notificações SMS (Short Message Service) para smartphones e celulares habilitados para SMS.

> **Note**
>
> As notificações SMS atualmente podem ser enviadas para números de telefone nos Estados Unidos. As mensagens SMS só podem ser enviadas de tópicos criados na região Leste dos EUA (Norte da Virgínia). No entanto, você pode publicar mensagens em tópicos que você cria na região Leste dos EUA (Norte da Virgínia) a partir de qualquer outra região.

Criar um tópico

Para criar um tópico:

1. No console do Amazon SNS, clique em Create new topic (Criar outro tópico). A caixa de diálogo Create new topic é exibida.
2. Na caixa Topic name, digite um nome para o tópico.
3. Na caixa Display name, digite um nome de exibição. O tópico precisa ter um nome de exibição atribuído a ele, pois os primeiros dez (10) caracteres do nome de exibição são usados como a parte inicial do prefixo da mensagem de texto. O nome de exibição que você inserir será exibido na mensagem de confirmação que o SNS envia ao usuário (o nome de exibição abaixo é "AMZN SMS").



1. Clique em Create topic (Criar tópico). O novo tópico é exibido na página de Topics.
2. Selecione o novo tópico e clique no ARN do tópico. A página Topic Details é exibida.
3. Copie o ARN do tópico, pois ele será necessário quando você inscrever um tópico na próxima etapa.

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

Inscrever-se em um tópico usando o protocolo SMS

Crie um cliente SNS, transmitindo seu objeto de credenciais e a região do seu grupo de identidades:

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

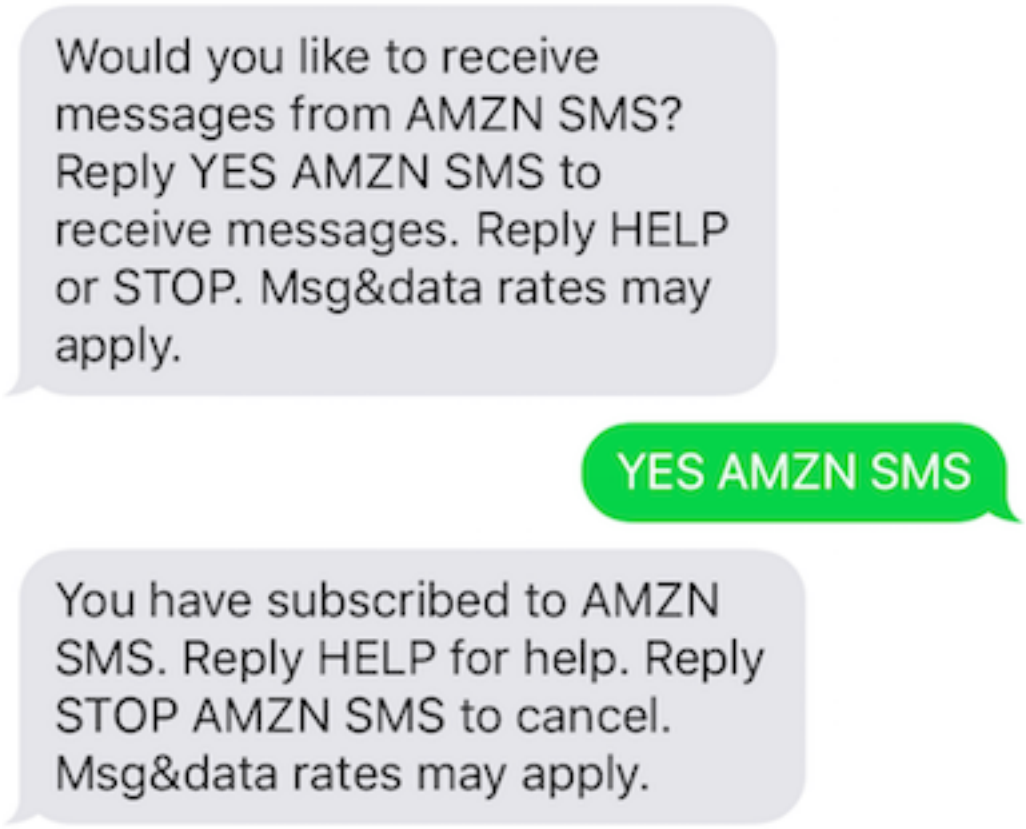
Para inscrever um tópico, invoque `SubscribeAsync` e transmita o ARN do tópico que você deseja inscrever, o protocolo ("sms") e o número de telefone:

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

Você receberá um arn de inscrição no objeto de resposta da inscrição. Seu arn de inscrição é semelhante a:

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

Quando um dispositivo se inscreve em um tópico, o SNS envia uma mensagem de confirmação para o dispositivo, e o usuário tem que confirmar que deseja receber notificações, como mostrado abaixo:



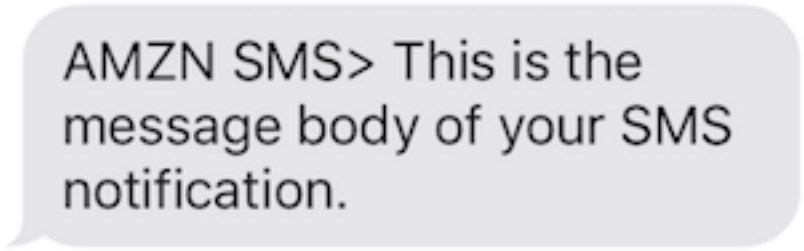
Depois que o usuário se inscrever no tópico, ele receberá mensagens SMS quando você publicá-las nesse tópico.

Publicar uma mensagem

Para publicar uma mensagem em um tópico:

1. Faça login no Console de Gerenciamento da AWS e abra o console do Amazon SNS.
2. No painel de navegação à esquerda, clique em Topics (Tópicos) e selecione o tópico no qual você deseja publicar.
3. Clique em Publish to topic (Publicar no tópico).
4. Na caixa Subject, digite um assunto.
5. Na caixa de Message, digite uma mensagem. O Amazon SNS envia o texto que você inseriu na caixa Message para os assinantes de SMS, salvo se você também inserir texto na caixa Subject. Como o Amazon SNS inclui um prefixo do nome de exibição com todas as mensagens SMS que você envia, a soma do prefixo do nome de exibição e a carga da mensagem não podem exceder 140 caracteres ASCII ou 70 caracteres Unicode. O Amazon SNS trunca mensagens que excedam esses limites.

6. Clique em Publish message (Publicar mensagem). O Amazon SNS exibe uma caixa de diálogo de confirmação. A mensagem SMS é exibida no seu dispositivo habilitado para SMS, como mostrado abaixo.



Enviar mensagens para HTTP/HTTPS endpoints

Use o Amazon SNS para enviar mensagens de notificação a um ou mais endpoints HTTP ou HTTPS. O processo é o seguinte:

1. Configure o endpoint para receber mensagens do Amazon SNS.
2. Inscreva um HTTP/HTTPS endpoint em um tópico.
3. Confirme sua assinatura.
4. Publique uma notificação no tópico. O Amazon SNS enviará uma solicitação HTTP POST enviando o conteúdo da notificação ao endpoint inscrito.

Configure seu HTTP/HTTPS endpoint para receber mensagens do Amazon SNS

Siga as instruções na Etapa 1 do envio de mensagens do Amazon SNS para HTTP/HTTPS endpoints para configurar seu endpoint.

Inscreva seu HTTP/HTTPS endpoint em seu tópico do Amazon SNS

Crie um cliente SNS, transmitindo seu objeto de credenciais e a região do seu grupo de identidades:

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

Para enviar mensagens a um endpoint HTTP ou HTTPS por meio de um tópico, inscreva o endpoint no tópico do Amazon SNS. Você especifica o endpoint usando seu URL:

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

Confirmação da assinatura

Depois que você se inscrever em um endpoint, o Amazon SNS enviará uma mensagem de confirmação de assinatura ao endpoint. O código no endpoint deve recuperar o valor `SubscribeURL` na mensagem de confirmação de assinatura e acessar o local especificado pelo próprio `SubscribeURL` ou disponibilizá-lo para que você possa acessar o `SubscribeURL` manualmente (por exemplo, se você estiver usando um navegador da web).

O Amazon SNS não enviará mensagens ao endpoint até que a assinatura seja confirmada. Ao acessar o `SubscribeURL`, a resposta conterá um documento XML com um elemento `SubscriptionArn` que especifica o ARN da inscrição.

Enviar mensagens para o HTTP/HTTPS endpoint

É possível enviar uma mensagem para inscrições em um tópico publicando no tópico. Invoque `PublishAsync` e transmita a ele o Nome de região da Amazon (ARN) do tópico e sua mensagem.

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

Solução de problemas do SNS

Uso do status de entrega no console do Amazon SNS

O console do Amazon SNS contém um recurso de status de entrega que permite coletar feedback sobre tentativas de entrega bem-sucedidas e malsucedidas de suas mensagens para plataformas de notificação por push móveis (Apple (APNS), Google (GCM), Amazon (ADM), Windows (WNS e MPNS) e Baidu).

Ele também fornece outras informações importantes, como os tempos de permanência no Amazon SNS. Essas informações são capturadas em um grupo Amazon CloudWatch Log que é criado automaticamente pelo Amazon SNS quando esse recurso é ativado por meio do console do Amazon SNS ou do Amazon SNS. APIs

Para obter instruções sobre como usar o recurso de status de entrega, consulte Uso do recurso de status de entrega do Amazon SNS no blog do AWS Mobile.

Práticas recomendadas para o uso do AWS Mobile SDK para .NET e Xamarin

Há somente alguns fundamentos e práticas recomendadas que você deve conhecer ao usar o AWS Mobile SDK para .NET e Xamarin.

- Use o Amazon Cognito para obter credenciais da AWS, em vez de incorporar suas credenciais ao aplicativo. Se você codificar suas credenciais no aplicativo, pode acabar expondo-as ao público, permitindo que outras pessoas façam chamadas para a AWS usando suas credenciais. Para obter instruções sobre como usar o Amazon Cognito para obter credenciais da AWS, consulte Configuração do AWS Mobile SDK para .NET e Xamarin.
- Para ver as práticas recomendadas para o uso do S3, consulte este artigo no Blog da AWS.
- Para ver as práticas recomendadas para o uso do DynamoDB, consulte Práticas recomendadas do DynamoDB no Guia do desenvolvedor do DynamoDB.

Estamos sempre procurando ajudar nossos clientes a serem bem-sucedidos e receber feedback. Portanto, sinta-se à vontade para publicar nos fóruns da AWS ou abrir uma edição sobre GitHub.

Biblioteca da documentação sobre o Serviço da AWS

Cada serviço no AWS Mobile SDK para .NET e Xamarin tem um guia do desenvolvedor independente e dispõe de uma referência de API e de serviços, que fornecem informações adicionais que podem ser úteis para você.

Identidade do Amazon Cognito

- Guia do desenvolvedor do Cognito
- Referência de API do serviço Cognito Identity

Amazon Cognito Sync

- Guia do desenvolvedor do Cognito
- Referência de API do serviço Cognito Sync

Amazon Mobile Analytics

- Guia do desenvolvedor do Mobile Analytics
- Referência de API do serviço Mobile Analytics

Amazon S3

- Guia do desenvolvedor do S3
- Guia de conceitos básicos do S3
- Referência de API do serviço S3

Amazon DynamoDB

- Guia do desenvolvedor do DynamoDB
- Guia de conceitos básicos do DynamoDB:
- Referência de API do serviço DynamoDB

Amazon Simply Notification Service (SNS)

- Guia do desenvolvedor do SNS
- Referência de API do serviço SNS

Outros Links úteis

- Glossário de termos da AWS
- Sobre as credenciais da AWS

Solução de problemas

Este tópico descreve algumas ideias para solucionar possíveis problemas ao usar o AWS Mobile SDK para .NET e Xamarin.

Verificação da existência de permissões necessárias na função do IAM

Ao chamar os serviços da AWS, seu aplicativo deve estar usando uma identidade de um grupo de identidades do Cognito. Cada identidade do grupo é associada a uma função do IAM (Identity and Access Management).

Uma função tem um ou mais arquivos de política associados a ela, que especificam a quais recursos da AWS os usuários atribuídos à função terão acesso. Por padrão, duas funções serão criadas para cada grupo de identidade: uma para os usuários autenticados e outra para os não autenticados.

Você precisará modificar o arquivo de política existente ou associar um novo arquivo de política com as permissões necessárias ao aplicativo. Se o aplicativo permitir usuários autenticados e não autenticados, as duas funções deverão receber permissões para acessar os recursos da AWS necessários ao aplicativo.

O arquivo de política a seguir mostra como conceder acesso a um bucket do S3:

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

O arquivo de política a seguir mostra como conceder acesso a um banco de dados do 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"
      }
   ]
}
```

Para obter mais informações sobre como especificar políticas, consulte Políticas do IAM.

Uso de um depurador de proxy HTTP

Se o serviço da AWS que seu aplicativo está chamando tiver um endpoint HTTP ou HTTPS, você poderá usar um depurador de HTTP/HTTPS proxy para visualizar as solicitações e as respostas e obter mais informações sobre o que está ocorrendo. Há uma série depuradores de proxy HTTP disponíveis, como:

- Charles – um proxy de depuração da web para Windows e OSX
- Fiddler – um proxyfidd de depuração da web para Windows

Charles e Fiddler exigem que algumas configurações possam exibir o tráfego criptografado por SSL. Leia a documentação dessas ferramentas para obter mais informações. Se você estiver usando um proxy de depuração da web que não possa ser configurado para exibir o tráfego criptografado, abra o arquivo aws_endpoints_json e defina a tag HTTP referente ao Serviço da AWS necessária para depurar como verdadeiro.

Histórico do documento

A tabela a seguir descreve as mudanças importantes na documentação desde o último lançamento do AWS Mobile SDK para .NET e Xamarin.

- Versão da API: 27-08-2015
- Última atualização da documentação: 23/02/2021

Alteração	Versão da API	Descrição	Data de lançamento
Arquivado	27/08/2015	O SDK AWS móvel para Xamarin está incluído no. AWS SDK para .NET Este guia faz referência à versão arquivada do Mobile SDK para Xamarin.	2021-02-23
Lançamento do GA	27/08/2015	Lançamento do GA	27/08/2015
Versão beta	28/07/2015	Versão beta	rn28-07-2015