ユーザーガイド

AWS Toolkit for Visual Studio

AWS Toolkit for Visual Studio: ユーザーガイド

Table of Contents

AWS Toolkit for Visual Studio

これは、AWS Toolkit for Visual Studio のユーザーガイドです。AWS Toolkit for VS Code をお探しの場合、AWS Toolkit for Visual Studio Code用ユーザーガイドを参照してください。

Toolkit for Visual Studio とは何か

AWS Toolkit for Visual Studio は Visual Studio IDE のプラグインで、Amazon Web Services を使用する.NET アプリケーションの開発、デバッグ、デプロイを簡単に行えます。Toolkit for Visual Studio は Visual Studio バージョン 2019 以降でサポートされています。キットのダウンロードとインストール方法の詳細については、このユーザーガイドの「インストールとセットアップ」トピックを参照してください。

> **Note**
>
> Toolkit for Visual Studio は、Visual Studio 2008、2010、2012、2013、2015、2017 バージョン向けにもリリースされました。ただし、これらのバージョンはサポートされていません。詳細については、このユーザーガイドの「インストールとセットアップ」トピックを参照してください。

Toolkit for Visual Studio には、開発作業を向上させるための、以下の機能が含まれています。

AWS エクスプローラー

IDE の「表示」 AWS メニューから利用できるエクスプローラツールウィンドウでは、Visual Studio IDE AWS 内から多くのサービスを操作できます。サポートされているデータサービスには、アマゾンシンプルストレージサービス (Amazon S3)、Amazon SimpleDB、アマゾンシンプル通知サービス (Amazon SNS)、Amazon Simple Queue Service (Amazon SQS)、アマゾンが含まれます。 CloudFront AWS Explorer では、Amazon Elastic Compute Cloud (Amazon EC2) 管理、 AWS Identity and Access Management (IAM) ユーザーおよびポリシー管理、サーバーレスアプリケーションと機能のデプロイ、 AWS Lambda およびへのウェブアプリケーションのデプロイにもアクセスできます。 AWS Elastic Beanstalk AWS CloudFormation

認証情報とリージョンの管理

AWS Explorer AWS は複数のアカウント (IAM ユーザーアカウントを含む) とリージョンをサポートしており、表示されるビューをあるアカウントから別のアカウントに簡単に変更したり、さまざまなリージョンのリソースやサービスを表示および管理したりできます。

Amazon EC2

AWS エクスプローラーから、利用可能な Amazon マシンイメージ (AMI) を表示し、それらの AMI から Amazon EC2 インスタンスを作成し、Windows リモートデスクトップを使用してそれらのインスタンスに接続できます。 AWS Explorer では、キーペアやセキュリティグループの作成や管理などのサポート機能も利用できます。

AWS Lambda

Lambda を使用して、サーバーレス .NET Core C# 関数やサーバーレスアプリケーションをホストすることができます。設計図を使用して、新しいサーバーレスプロジェクトを迅速に作成し、サーバーレスアプリケーションの開発を簡単に行うことができます。

AWS CodeCommit

CodeCommit Visual Studio チームエクスプローラーと統合されています。これにより、保持されているリポジトリを複製して作成したり CodeCommit、IDE 内からソースコードの変更を処理したりすることが容易になります。

Amazon DynamoDB

DynamoDB は、拡張性と可用性に優れた、費用効果の高い、高速な非リレーショナルデータベースサービスです。Toolkit for Visual Studio には、開発の場面で Amazon DynamoDB と連携する機能が用意されています。Toolkit for Visual Studio を使用すると、DynamoDB テーブルの属性を作成および編集したり、テーブルのスキャンオペレーションを実行したりできます。

Amazon S3

ドラッグアンドドロップによって、コンテンツを Amazon S3 バケットにすばやく簡単にアップロードしたり、Amazon S3 からコンテンツをダウンロードできます。バケット内のオブジェクトに、アクセス許可、メタデータ、タグを簡単に設定することもできます。

Amazon RDS

AWS エクスプローラーは、Visual Studio で Amazon RDS アセットを作成および管理するのに役立ちます。Microsoft SQL Server を使用する Amazon RDS インスタンスを Visual Studio の Server Explorer に追加することもできます。

AWS Elastic Beanstalk

Elastic Beanstalk を使用して .NET ウェブアプリケーションプロジェクトを AWSにデプロイすることができます。IDE 内から、アプリケーションを、単一インスタンス環境にデプロイしたり、完全に負荷分散され、自動スケーリングされる環境にデプロイしたりすることができます。また Visual Studio の中から、アプリケーションの新しいバージョンを迅速かつ簡単にデプロイすることもできます。アプリケーションが Amazon RDS の SQL Server を使用している場合、デプロイウィザードを使って、Elastic Beanstalk のアプリケーション環境と Amazon RDS のデータベースインスタンス間の接続をセットアップすることもできます。Toolkit for Visual Studio には、スタンドアロンのコマンドラインデプロイツールも含まれています。デプロイツールを使用して、デプロイをビルド処理の自動化に含めることができます。またはデプロイを Visual Studio の外部で他のスクリプトシナリオに含めることもできます。

AWS CloudFormation

Toolkit for Visual Studio を使用して、 IntelliSense エディターと構文の強調表示をサポートする AWS CloudFormation JSON 形式のテンプレートを編集できます。 AWS CloudFormation テンプレートには、アプリケーションをホストするためにインスタンス化するリソースを記述します。次に IDE 内からテンプレートをデプロイします。 AWS CloudFormationテンプレートに記述されているリソースがプロビジョニングされ、デベロッパーはアプリケーションの機能の開発に注力できます。

AWS Identity and Access Management (IAM)

AWS Explorer から IAM ユーザー、ロール、ポリシーを作成し、ユーザーにポリシーをアタッチできます。

関連情報

課題を開いたり、現在未解決の課題を表示したりするには、https://github.com/aws/ aws-toolkit-visual-studio /issuesにアクセスしてください。

Visual Studio の詳細については、https://visualstudio.microsoft.com/vs/ にアクセスしてください。

Amazon Q と Amazon CodeWhisperer

Amazon Q とは

2024 年 4 月 30 日現在、Amazon CodeWhisperer は Amazon Q Developer の一部になりました。これにはインラインコードの提案とセキュリティスキャンが含まれます。

での Amazon Q デベロッパーの操作の詳細については AWS Toolkit for Visual Studio、「Amazon Q デベロッパーユーザーガイド」のIDEs での Amazon Q デベロッパー」トピックを参照してください。Amazon Q のプランと料金の詳細については、Amazon Q の料金ガイドを参照してください。

Toolkit for Visual Studio のダウンロード

Toolkit for Visual Studio は IDE の Visual Studio Marketplace からダウンロード、インストール、セットアップできます。詳細な手順については、このユーザーガイドの「開始方法」トピックの「AWS Toolkit for Visual Studio のインストール」セクションを参照してください。

Visual Studio Marketplace からツールキットをダウンロードする

ウェブブラウザで AWS Visual Studio ダウンロードサイトを開いて、Toolkit for Visual Studio のインストールファイルをダウンロードします。

AWS のその他のIDE ツールキット

Toolkit for Visual Studio に加えて、 は IDE Toolkit for VS Code と AWSも提供しています JetBrains。

AWS Toolkit for Visual Studio Code へのリンク

- このリンクにアクセスして、VS Code Marketplace から AWS Toolkit for Visual Studio Code をダウンロードできます。
- AWS Toolkit for Visual Studio Code の詳細については、「AWS Toolkit for Visual Studio Code ユーザーガイド」を参照してください。

AWS Toolkit for JetBrains へのリンク

- このリンクから Marketplace から をダウンロードAWS Toolkit for JetBrainsします JetBrains 。
- AWS Toolkit for JetBrains の詳細については、「AWS Toolkit for JetBrains ユーザーガイド」を参照してください。

開始方法

AWS Toolkit for Visual Studio は、AWS のサービスとリソースを Visual Studio 統合開発環境 (IDE) から直接利用できるようにします。

使用開始をサポートするために、以下のトピックでは AWS Toolkit for Visual Studio のインストール、セットアップ、設定方法について説明します。

トピック

のインストールとセットアップ AWS Toolkit for Visual Studio

次のトピックでは、 AWS Toolkit for Visual Studioのダウンロード、インストール、セットアップ、アンインストールの方法について説明します。

トピック

前提条件

サポートされているバージョンの AWS Toolkit for Visual Studioをセットアップするための前提条件を次に示します。

- Visual Studio 19 またはそれ以降のリリース
- Windows 10 以降のリリース
- Windows と Visual Studio への管理者アクセス
- アクティブな AWS IAM 認証情報

> **Note**
>
> AWS Toolkit for Visual Studio サポートされていないバージョンは、Visual Studio 2008、2010、2012、2013、2015、2017 でご利用いただけます。サポートされていないバージョンをダウンロードするには、AWS Toolkit for Visual Studio のランディングページに移動し、ダウンロードリンクのリストから目的のバージョンを選択します。
> IAM 認証情報の詳細を確認したり、アカウントにサインアップしたりするには、AWS コンソールゲートウェイにアクセスしてください。

をインストールする AWS Toolkit for Visual Studio

をインストールするには AWS Toolkit for Visual Studio、次の手順から使用している Visual Studio のバージョンを確認し、必要な手順を実行します。 AWS Toolkit for Visual Studio のすべてのバージョンのダウンロードリンクは、AWS Toolkit for Visual Studioランディングページにあります。

> **Note**
>
> のインストール中に問題が発生した場合は AWS Toolkit for Visual Studio、本ガイドの「インストールに関する問題のトラブルシューティング」を参照してください。

Visual Studio 2022 AWS Toolkit for Visual Studio 版のをインストールします。

Visual Studio から AWS Toolkit for Visual Studio 2022 をインストールするには、以下の手順を実行します。

1. [Main menu] から [Extensions] に移動し、[Manage Extensions] を選択します。
2. 検索ボックスで AWSを検索します。
3. 該当するバージョンの Visual Studio 2022 の [Download] ボタンを選択し、インストールプロンプトに従います。

> **Note**
>
> インストールプロセスを完了するには、Visual Studio を手動で閉じて再起動しなければならない場合があります。

4. ダウンロードとインストールが完了したら、[表示] メニューから [AWS エクスプローラー] AWS Toolkit for Visual Studio を選択してを開くことができます。

Visual Studio 2019 AWS Toolkit for Visual Studio 用のをインストールします。

Visual Studio から AWS Toolkit for Visual Studio 2019 をインストールするには、次の手順を実行します。

1. [Main menu] から [Extensions] に移動し、[Manage Extensions] を選択します。
2. 検索ボックスで AWSを検索します。
3. Visual Studio 2017 と 2019 の [Download] ボタンを選択し、プロンプトに従います。

> **Note**
>
> インストールプロセスを完了するには、Visual Studio を手動で閉じて再起動しなければならない場合があります。

4. ダウンロードとインストールが完了したら、[表示] メニューから [AWS エクスプローラー] AWS Toolkit for Visual Studio を選択してを開くことができます。

をアンインストールします。 AWS Toolkit for Visual Studio

をアンインストールするには AWS Toolkit for Visual Studio、次の手順から使用している Visual Studio のバージョンを確認し、必要な手順を実行します。

Visual Studio 2022 AWS Toolkit for Visual Studio 用のをアンインストールします。

Visual Studio から AWS Toolkit for Visual Studio 2022 をアンインストールするには、次の手順を実行します。

1. [Main menu] から [Extensions] に移動し、[Manage Extensions] を選択します。
2. [Manage Extensions] ナビゲーションメニューから、[Installed] 見出しを展開します。
3. AWS Toolkit for Visual Studio 2022 拡張機能を探し、[Uninstall] ボタンを選択します。

> **Note**
>
> ナビゲーションメニューの [インストール済み] AWS Toolkit for Visual Studio セクションにが表示されない場合は、Visual Studio の再起動が必要な場合があります。

4. 画面上のプロンプトに従ってアンインストールプロセスを完了します。

Visual Studio 2019 AWS Toolkit for Visual Studio 用をアンインストールします。

AWS Toolkit for Visual Studio 2019 を Visual Studio からアンインストールするには、次の手順を実行します。

1. [Main menu] から [ツール] に移動し、[Manage Extensions] を選択します。
2. [Manage Extensions] ナビゲーションメニューから、[Installed] 見出しを展開します。
3. AWS Toolkit for Visual Studio 2019 拡張機能を探し、[Uninstall] ボタンを選択します。
4. 画面上のプロンプトに従ってアンインストールプロセスを完了します。

Visual Studio 2017 AWS Toolkit for Visual Studio 用をアンインストールします。

Visual Studio で AWS Toolkit for Visual Studio 2017 をアンインストールするには、次の手順を実行します。

1. [Main] メニューから [ツール] に移動し、[Extensions and Updates] を選択します。
2. [Extensions and Updates] ナビゲーションメニューから、[Installed] 見出しを展開します。
3. AWS Toolkit for Visual Studio 2017 拡張機能を探し、[Uninstall]ボタンを選択します。
4. 画面上のプロンプトに従ってアンインストールプロセスを完了します。

Visual Studio 2013 または 2015 AWS Toolkit for Visual Studio 用のをアンインストールします。

AWS Toolkit for Visual Studio 2013 または 2015 をアンインストールするには、次の手順を実行します。

1. Windows のコントロールパネルから [プログラムと機能] を開きます。

> **Note**
>
> Windows のコマンドプロンプトまたは Windows の [実行] ダイアログから実行すると、[プログラムと機能] をすぐに開くことができます。

2. インストールされたプログラムのリストから、[AWS Tools for Windows] のコンテキスト (右クリック) メニューを開きます。
3. [Uninstall] を選択し、プロンプトに従ってアンインストールプロセスを完了します。

> **Note**
>
> [Samples] ディレクトリはアンインストールプロセル中に削除されません。このディレクトリは、サンプルを変更した場合に保持されます。このディレクトリは手動で削除する必要があります。

への接続 AWS

ほとんどの Amazon Web Services (AWS) サービスとリソースは、 AWS アカウントを通じて管理されます。 AWS アカウントは を使用する必要はありませんが AWS Toolkit for Visual Studio、 ツールキットの機能は接続なしで制限されます。

以前に別の AWS サービス (など AWS Command Line Interface) を通じて AWS アカウントと認証を設定したことがある場合、Toolkit for Visual Studio は認証情報を自動的に検出します。

前提条件

を初めて使用する場合、 AWS またはアカウントを作成していない場合は、Toolkit for Visual Studio を AWS アカウントに接続するための 3 つの主要なステップがあります。

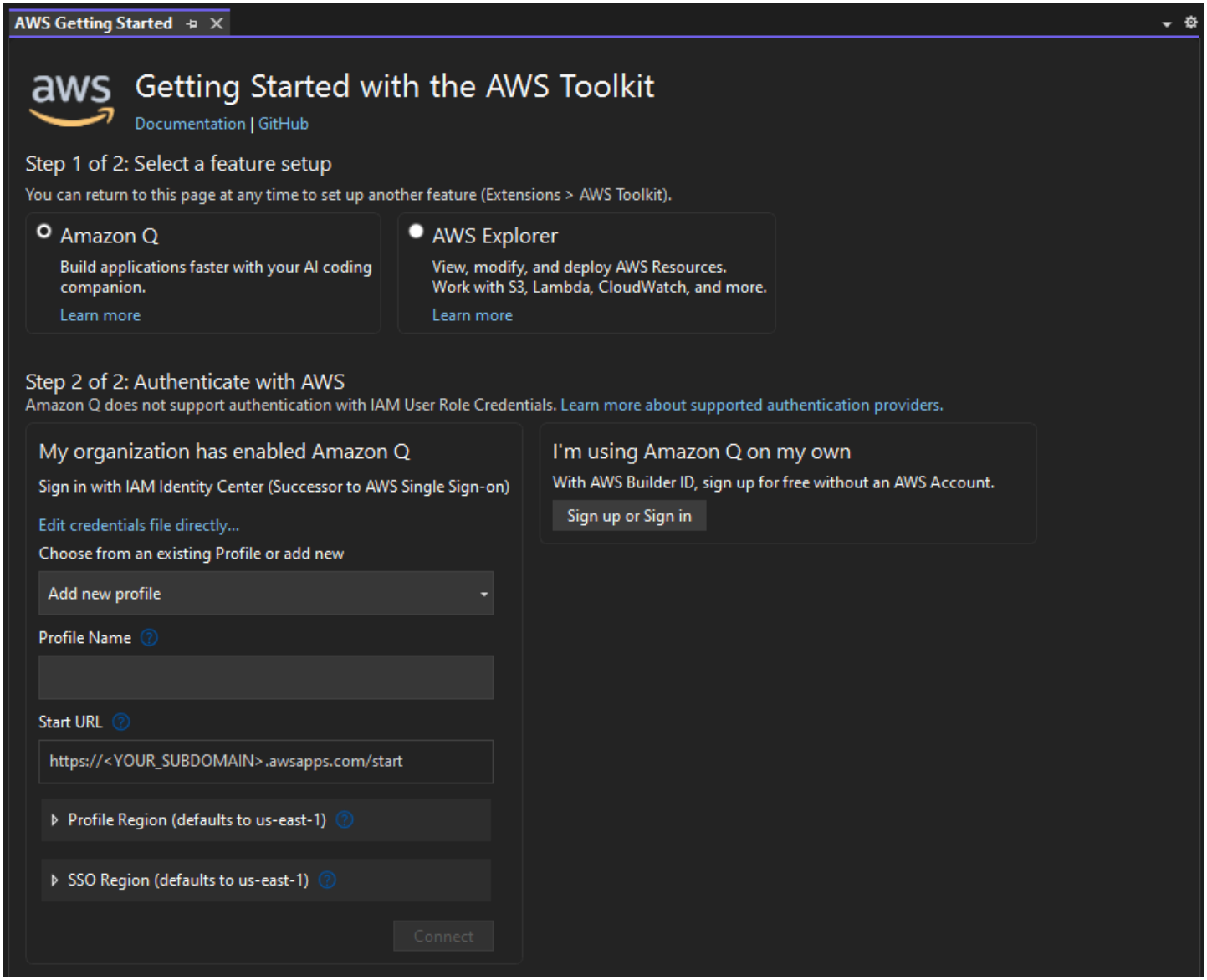
1. AWS アカウントにサインアップする: サインアップポータル AWS からアカウントにサインアップできます。 AWS新しい AWS アカウントの設定の詳細については、「セットアップユーザーガイド」の「概要」トピックを参照してください。 AWS
2. 認証のセットアップ: Toolkit for Visual Studio から AWS アカウントで認証するには、主に 3 つの方法があります。これらの方法の詳細については、本ユーザーガイドの「認証とアクセス」トピックを参照してください。

3. Toolkit AWS から を使用して認証する: このユーザーガイドの以下のセクションの手順を完了することで、 Toolkit から AWS アカウントに接続できます。

Toolkit AWS から に接続する

Toolkit for Visual Studio から AWS アカウントに接続するには、次の手順を実行して AWS Toolkit ユーザーインターフェイス (接続 UI) の開始方法を開きます。

1. Visual Studio のメインメニューから Extensions を展開し、AWS Toolkit を展開します。
2. AWS ツールキット メニューから 開始方法 を選択します。
3. Visual Studio で AWS Toolkit 接続 UI の開始方法が開きます。



次の表は、各機能と互換性のある認証方法を示しています。3 つの認証方法 AWS IAM Identity Center、 AWS Identity and Access Management 認証情報、および AWS Builder ID のそれぞれの詳細については、このユーザーガイドの「認証とアクセス」の目次を参照してください。

> **Note**
>
> Toolkit for Visual Studio CodeCatalyst で を使用する場合、現在必要なのは、サードパーティーのリポジトリのクローンを作成するときに AWS Builder ID を使用して認証することだけです。

Amazon Q Developer	AWS Explorer	Amazon CodeCatalyst
AWS ☍Builder ID	AWS ☒Builder ID	AWS ☍Builder ID
☑ IAM アイデンティティセンター	☑ IAM アイデンティティセンター	☒ IAM アイデンティティセンター
AWS ☒IAM 認証情報	AWS 「IAM 認証情報」	AWS ☒IAM 認証情報

Amazon Q デベロッパーの認証

Amazon Q Developer の使用を開始するには、 AWS IAM Identity Center または AWS Builder ID の認証情報を使用して認証して接続します。

以下の手順では、 AWS アカウントを使用して Toolkit の認証と接続を行う方法について説明します。

IAM アイデンティティセンターで認証して接続する

1. AWS ツールキット接続 UI の開始方法から、Amazon Q デベロッパーリングを選択して Amazon Q デベロッパー認証オプションを展開します。

   > **Note**
   >
   > 認証情報が保存されていない場合は、ステップ 3 に進み、IAM Identity Center 認証情報を追加または更新します。

2. 「My organization has enabled Amazon Q Developer」セクションから、「Select from an existing Profile」を展開するか、新しいドロップダウンメニューを追加して、保存した認証情報のリストから選択します。
3. プロファイルタイプドロップダウンメニューから、 AWS IAM Identity Center
4. プロファイル名のテキストフィールドに、認証に使用する IAM Identity Center プロファイル**`Profile Name`**の を入力します。
5. URL テキストの開始 フィールドに、IAM Identity Center 認証情報にアタッチ**`Start URL`**されている を入力します。

6. プロファイルリージョン (デフォルトは us-east-1) ドロップダウンメニューから、認証に使用する IAM Identity Center ユーザープロファイルで定義されているプロファイルリージョンを選択します。
7. SSO リージョン (デフォルトは us-east-1) ドロップダウンメニューから、IAM Identity Center 認証情報で定義されている SSO リージョンを選択し、Connect ボタンを選択して IAM Identity Center で AWS ログインダイアログを開きます。
8. AWS 「IAM Identity Center でログイン」ダイアログで「ブラウザに進む」ボタンを選択し、デフォルトのウェブブラウザで AWS リクエストサイトの承認を開きます。
9. IDE のセキュリティコードがウェブブラウザに表示されるリクエスト確認コード AWS の承認と一致することを確認し、送信と続行ボタンを選択して続行します。
10. デフォルトのウェブブラウザのプロンプトに従って、認証プロセスが完了すると通知が送信され、ブラウザを閉じ、Visual Studio に戻ります。

AWS Builder ID を使用して認証して接続する

1. AWS ツールキット接続 UI の開始方法から、Amazon Q デベロッパーのリングを選択して Amazon Q デベロッパー認証オプションを展開します。
2. 自分のセクションで Amazon Q Developer を使用しています。「サインアップ」または「サインイン」ボタンを選択して、 AWS 「 Builder ID でログイン」ダイアログを開きます。
3. ブラウザに進む ボタンを選択して、デフォルトのウェブブラウザでリクエストサイトの承認を開きます AWS 。
4. IDE のセキュリティコードがウェブブラウザに表示されるリクエスト確認コード AWS の承認と一致することを確認し、送信と続行ボタンを選択して続行します。
5. デフォルトのウェブブラウザのプロンプトに従って、認証プロセスが完了すると通知が送信され、ブラウザを閉じ、Visual Studio に戻ります。

AWS Explorer の認証

ツールキットから AWS Explorer の使用を開始するには、IAM Identity Center 認証情報または IAM 認証情報を使用して認証し、接続します。

以下の手順では、 AWS アカウントを使用して Toolkit の認証と接続を行う方法について説明します。

IAM アイデンティティセンターで認証して接続する

1. AWS ツールキット接続 UI の開始方法 から Explorer AWS のリングを選択して、Amazon Q Developer 認証オプションを展開します。
2. [**Profile Type**] ドロップダウンメニューから [AWS IAM Identity Center] を選択します。
3. プロファイル名のテキストフィールドに、使用する IAM Identity Center プロファイル**Profile Name**の を入力します。
4. 「URL テキストの開始」フィールドに、IAM Identity Center 認証情報にアタッチ**Start URL**されている を入力します。
5. プロファイルリージョン (デフォルトは us-east-1) ドロップダウンメニューから、認証する IAM Identity Center ユーザープロファイルで定義されているプロファイルリージョンを選択します。
6. SSO リージョン (デフォルトは us-east-1) ドロップダウンメニューから、IAM Identity Center 認証情報で定義されている SSO リージョンを選択します。
7. ブラウザに進むボタンを選択して、デフォルトのウェブブラウザでAWS リクエストサイトの承認を開きます。
8. IDE のセキュリティコードがウェブブラウザに表示されるリクエスト確認コードAWS の承認と一致していることを確認し、送信と続行ボタンを選択して続行します。
9. デフォルトのウェブブラウザのプロンプトに従って、認証プロセスが完了すると通知が送信され、ブラウザを閉じ、Visual Studio に戻ります。

IAM 認証情報で認証して接続する

1. AWS ツールキット接続 UI の開始方法 から Explorer のAWS リングを選択して、Amazon Q Developer 認証オプションを展開します。
2. **Profile Type** ドロップダウンメニューから、IAM ユーザーロール を選択します。
3. プロファイル名のテキストフィールドに、認証するプロファイル**Profile Name**の を入力します。
4. アクセスキー ID テキストフィールドに、認証するプロファイル**Access Key ID**の を入力します。
5. シークレットキーテキストフィールドに、認証するプロファイル**Secret Key**の を入力します。
6. ストレージの場所 (デフォルトは共有認証情報ファイル) ドロップダウンメニューから、認証情報を共有認証情報ファイルまたは.NET 暗号化保存済み のどちらで保存するかを指定します。

7. プロファイルリージョン (デフォルトは us-east-1) ドロップダウンメニューから、認証するプロファイルにアタッチされているプロファイルリージョンを選択します。

のインストールに関する問題のトラブルシューティング AWS Toolkit for Visual Studio

次の情報は、AWS Toolkit for Visual Studioをインストールする際によく発生するインストールの問題を解決するためのものです。

AWS Toolkit for Visual Studioのインストール中にエラーが発生した場合や、インストールが完了したかどうか不明な場合は、次の各セクションの情報を確認してください。

ビジュアルスタジオの管理者権限

AWS Toolkit for Visual Studioこの拡張機能には、AWSすべてのサービスと機能にアクセスできるようにするための管理者権限が必要です。

ローカル管理者権限を持っている場合、管理者権限が Visual Studio インスタンスに直接適用されない可能性があります。

管理者権限で Visual Studio をローカルで起動するには:

1. Windows から、Visual Studio アプリケーションランチャー (アイコン) を探します。
2. Visual Studio アイコンのコンテキストメニュー (右クリック) を開いて、コンテキストメニューを開きます。
3. コンテキストメニューから [管理者として実行] を選択します。

管理者権限で Visual Studio をリモートで起動するには:

1. Windows から、Visual Studio のリモートインスタンスへの接続に使用しているアプリケーションのアプリケーションランチャーを探します。
2. アプリケーションのコンテキストメニュー (右クリック) を開いて、コンテキストメニューを開きます。
3. コンテキストメニューから [管理者として実行] を選択します。

> **Note**
>
> プログラムをローカルで起動する場合でも、リモートで接続する場合でも、Windows から管理者資格情報の確認を求められる場合があります。

インストールログの取得

前述の「管理者権限」セクションの手順を完了し、管理者権限で Visual Studio を実行または接続していることが確認できた場合は、インストールログファイルを取得すると他の問題の診断に役立ちます。

`.vsix`をファイルから手動でインストールしてインストールログファイルを生成するには、次の手順を実行します。AWS Toolkit for Visual Studio

1. AWS Toolkit for Visual Studioランディングページから、ダウンロードリンクをクリックして、`.vsix`AWS Toolkit for Visual Studioインストールするバージョンのファイルを保存します。
2. Visual Studio のメインメニューから、「ツール」ヘッダーを展開し、「コマンドライン」サブメニューを展開して、「Visual Studio 開発者用コマンドプロンプト」を選択します。
3. Visual Studio の開発者用コマンドプロンプトから、`vsixinstaller`次の形式のコマンドを入力します。

   ```
   vsixinstaller /logFile:[file path to log file] [file path to Toolkit installation file]
   ```

4. `[file path to log file]`インストールログを作成するディレクトリのファイル名とフルファイルパスに置き換えてください。`vsixinstaller`指定したファイルパスとファイル名を使用したコマンドの例は、次のようになります。

   ```
   vsixinstaller /logFile:C:\Users\Documents\install-log.txt [file path to AWSToolkitPackage.vsix]
   ```

5. `[file path to Toolkit installation file]`が置かれているディレクトリのフルファイルパスに置き換えてください。`AWSToolkitPackage.vsix`

   Toolkit `vsixinstaller` インストールファイルへのフルファイルパスを含むコマンドの例は、次のようになります。

```
vsixinstaller /logFile:[file path to log file] C:\Users\Downloads
\AWSToolkitPackage.vsix
```

6. ファイル名とパスが正しいことを確認してから、vsixinstallerコマンドを実行します。

   vsixinstallerコマンド全体の例は次のようになります。

   ```
   vsixinstaller /logFile:C:\Users\Documents\install-log.txt C:\Users
   \Downloads\AWSToolkitPackage.vsix
   ```

さまざまな Visual Studio 拡張機能のインストール

インストールログファイルを取得してもインストールプロセスが失敗する原因を特定できない場合は、他の Visual Studio 拡張機能をインストールできるかどうかを確認してください。さまざまな Visual Studio 拡張機能をインストールすると、インストールの問題についてさらに詳しく知ることができます。Visual Studio 拡張機能をインストールできない場合は、代わりに Visual Studio を使用して問題のトラブルシューティングを行う必要がある場合があります。AWS Toolkit for Visual Studio

Support へのお問い合わせ

このガイドに含まれるセクションをすべて読んでいて、追加のリソースやサポートが必要な場合は、AWS Toolkit for Visual StudioGithub Issuesサイトから過去のIssueを確認したり、新しいIssueをオープンしたりできます。

問題の解決を早めるために:

- 過去と現在の問題をチェックして、他の人が同様の状況に遭遇していないか確認してください。
- 問題に対処するために実行した各ステップを詳細にメモしておいてください。
- AWS Toolkit for Visual Studioまたは他の拡張機能をインストールして取得したログファイルを保存します。
- AWS Toolkit for Visual Studioインストールログファイルを新しい号に添付してください。

プロファイルとウィンドウバインディング

Toolkit for Visual Studio のプロファイルとウィンドウバインディング

Toolkit for Visual Studio パブリッシングツール、ウィザード、およびその他の機能を使用するときは、次の点に注意してください。

- AWS Explorer ウィンドウは一度に 1 つのプロファイルおよびリージョンにバインドされます。AWSExplorer から開いたウィンドウでは、デフォルトのバインドされているプロファイルとリージョンが使用されます。
- 新しいウィンドウが開いたら、AWSそのエクスプローラのインスタンスを使用して別のプロファイルまたはリージョンに切り替えることができます。
- Toolkit for Visual Studio パブリッシングツールと機能は、AWSエクスプローラーで設定されたプロファイルとリージョンに自動的にデフォルト設定されます。
- 公開ツール、ウィザード、または機能で新しいプロファイルまたはリージョンを指定した場合、その後に作成されたすべてのリソースは、引き続き新しいプロファイルとリージョンの設定を使用します。
- Visual Studio の複数のインスタンスを開いている場合は、各インスタンスを異なるプロファイルとリージョンにバインドできます。
- AWSExplorer では、最後に閉じた Visual Studio インスタンスの値が保持されます。

認証とアクセス

AWS Toolkit for Visual Studio の使用を開始するために AWS 、 で認証する必要はありません。ただし、ほとんどの AWS リソースは AWS アカウントを通じて管理されます。 AWS Toolkit for Visual Studio のすべてのサービスと機能にアクセスするには、少なくとも 2 種類のアカウント認証が必要です。

1. AWS アカウントの AWS Identity and Access Management (IAM) またはAWS IAM Identity Center認証のいずれか。ほとんどの AWS サービスとリソースは、IAM および IAM Identity Center を通じて管理されます。
2. AWS Builder ID は、他の特定の AWS サービスではオプションです。

以下のトピックには、各認証情報の種類と認証方法の詳細と設定手順が記載されています。

トピック

AWS の IAM Identity Center 認証情報 AWS Toolkit for Visual Studio

AWS IAM Identity Center は、 AWS アカウント認証を管理するための推奨されるベストプラクティスです。

ソフトウェア開発キット (SDKs「IAM Identity Center 認証AWS SDKs」セクションを参照してください。 AWS Toolkit for Visual Studio

からの IAM Identity Center による認証 AWS Toolkit for Visual Studio

IAM Identity Center プロファイルを `credentials`または `config` ファイルに追加 AWS Toolkit for Visual Studio して から IAM Identity Center で認証するには、次の手順を実行します。

1. 任意のテキストエディタから、 `<hone-directory>\.aws\credentials` ファイルに保存されている AWS 認証情報を開きます。

2. `credentials file` セクションの `[default]` から、名前付きの IAM アイデンティティセンタープロファイルのテンプレートを追加します。以下はテンプレートの例です。

> **Important**
>
> `credential` ファイルにエントリを作成するときは、`credential` ファイルの命名規則と矛盾するプロファイルという単語を使用しないでください。
> `config` ファイル内に名前付きプロファイルを設定する場合にのみ `profile_` プレフィックスを含めます。

```
[sso-user-1]
sso_start_url = https://example.com/start
sso_region = us-east-2
sso_account_id = 123456789011
sso_role_name = readOnly
region = us-west-2
```

- **`sso_start_url`**: 組織の IAM アイデンティティセンターユーザーポータルを指す URL。
- **`sso_region`**: IAM Identity Center ポータルホストを含む AWS リージョン。これは、デフォルトの `region`パラメータで後述する AWS リージョンとは異なる場合があります。
- **`sso_account_id`**: この IAM Identity Center ユーザーに付与するアクセス許可を持つ IAM ロールを含む AWS アカウント ID。
- **`sso_role_name`**: IAM アイデンティティセンターを経由して認証情報を得るために、このプロファイルを使用するときのユーザーのアクセス許可を定義する IAM ロールの名前。
- **`region`**: この IAM Identity Center ユーザーがサインインするデフォルトの AWS リージョン。

> **Note**
>
> `aws configure sso` コマンド AWS CLI を実行して、IAM Identity Center 対応プロファイルを に追加することもできます。このコマンドを実行したら、IAM Identity Center ディレクトリをホストする IAM Identity Center 開始 URL (`sso_start_url`) と AWS リージョン (`region`) の値を指定します。

詳細については、「 ユーザーガイド」の AWS 「シングルサインオンを使用するように AWS CLI を設定するAWS Command Line Interface 」を参照してください。

IAM アイデンティティセンターでサインインする

IAM アイデンティティセンタープロファイルでサインインする場合は、`credential file` で指定された `sso_start_url` でデフォルトのブラウザが起動されます。の AWS リソースにアクセスするには、IAM Identity Center のログインを確認する必要があります AWS Toolkit for Visual Studio。認証情報の有効期限が切れた場合は、新しい一時的な認証情報を取得するために接続プロセスを繰り返す必要があります。

AWS IAM 認証情報

AWS IAM 認証情報は、ローカルに保存されたアクセスキーを介して AWS アカウントで認証されます。

以下のセクションでは、 から AWS アカウントで認証するための IAM 認証情報を設定する方法について説明します AWS Toolkit for Visual Studio。

Important

AWS アカウントで認証するように IAM 認証情報を設定する前に、次の点に注意してください。

- 別の AWS サービス (など AWS AWS CLI) を介して IAM 認証情報を既に設定している場合、 はそれらの認証情報 AWS Toolkit for Visual Studio を自動的に検出します。
- AWS は AWS IAM Identity Center 認証の使用を推奨します。 AWS IAM のベストプラクティスの詳細については、「 Identity and Access Management ユーザーガイド」の「IAM のセキュリティのベストプラクティス」セクションを参照してください。 AWS
- セキュリティリスクを避けるため、専用ソフトウェアの開発や実際のデータを扱うときは、IAM ユーザーを認証に使用しないでください。代わりに、 などの ID プロバイダーとのフェデレーションを使用してください AWS IAM Identity Center。詳細については、「AWS IAM Identity Center ユーザーガイド」の「IAM アイデンティティセンターとは?」を参照してください。

「IAM ユーザーの作成」

AWS アカウントで認証 AWS Toolkit for Visual Studio するように を設定する前に、「 SDK およびツールリファレンスガイド」の「長期認証情報を使用した認証」トピックの「ステップ 1: IAM ユーザーの作成」と「ステップ 2: アクセスキーの取得」を完了する必要があります。 AWS SDKs

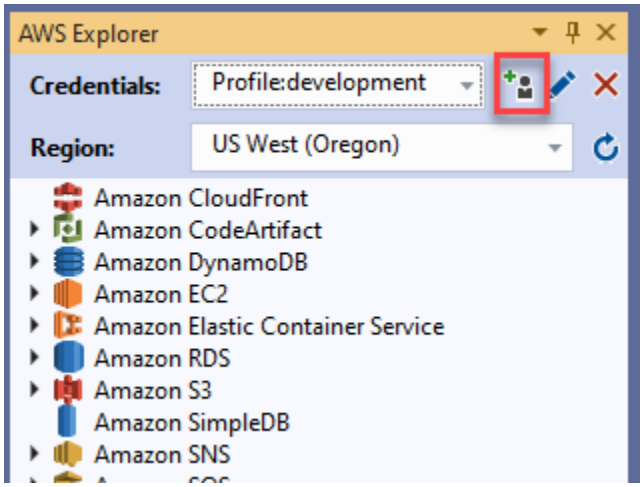
> **Note**
>
> 「ステップ 3: 共有認証情報ファイルの更新」はオプションです。
> ステップ 3 を完了すると、 は から認証情報 AWS Toolkit for Visual Studio を自動的に検出しますcredentials file。
> ステップ 3 を完了していない場合、 は、以下の セクションから認証情報ファイルを作成するで説明credentials fileされているように、 を作成するプロセスを AWS Toolkit for Visual Studio 順を追って説明します。 AWS Toolkit for Visual Studio

認証情報ファイルの作成

AWS Toolkit for Visual Studioでユーザーを追加する、あるいは credentials file を作成するには、以下の手順を実行します。

> **Note**
>
> ツールキットで新しいユーザープロファイルを追加する場合:
>
> - credentials file が既に存在する場合は、新しいユーザー情報が既存のファイルに追加されます。
> - credentials file が存在しない場合は、新しいファイルが作成されます。

1. AWS Explorer から新しいアカウントプロファイルアイコンを選択し、新しいアカウントプロファイルダイアログを開きます。



2. [新しいアカウントプロファイル] ダイアログの必須フィールドに入力し、[OK] ボタンを選択して IAM ユーザーを作成します。

ツールキットで IAM ユーザー認証情報を編集する

ツールキットで IAM ユーザー認証情報を編集するには、以下のステップを実行します。

1. AWS Explorer の認証情報ドロップダウンから、編集する IAM ユーザー認証情報を選択します。
2. [プロファイルを編集] アイコンをクリックして、[プロファイルを編集] ダイアログを開きます。
3. [プロファイルを編集] ダイアログで更新を完了し、[OK] ボタンを選択して変更を保存します。

ツールキットで IAM ユーザー認証情報を削除するには、以下のステップを実行します。

1. AWS Explorer の認証情報ドロップダウンから、削除する IAM ユーザー認証情報を選択します。
2. [プロファイルを削除] アイコンをクリックして、[プロファイルを削除] プロンプトを開きます。
3. `Credentials file` からプロファイルを削除することを確認します。

> **Important**
>
> [プロファイルを編集] ダイアログで IAM アイデンティティセンターや多要素認証 (MFA) など、高度なアクセス機能をサポートするプロファイルは、 AWS Toolkit for Visual Studioから編集できません。これらのタイプのプロファイルを変更するには、テキストエディタを使用して `credentials file` を編集する必要があります。

テキストエディタで IAM ユーザー認証情報を編集する

を使用して IAM ユーザーを管理するだけでなく AWS Toolkit for Visual Studio、任意のテキストエディタcredential filesから編集することもできます。Windows の場合、credential file のデフォルトの場所は C:\Users*USERNAME*\.aws\credentials です。

credential files の場所と構造の詳細については、「AWS SDK とツールのリファレンスガイド」の「Shared config and credentials files」セクションを参照してください。

AWS Command Line Interface（ AWS CLI) からの IAM ユーザーの作成

AWS CLI は、 credentials fileコマンドを使用して で IAM ユーザーを作成するために使用できる別のツールですaws configure。

から IAM ユーザーを作成する方法の詳細については、「 ユーザーガイド」の「トピックの設定 AWS CLIAWS CLI AWS CLI 」を参照してください。

Toolkit for Visual Studio は、次の構成プロパティをサポートしています。

```
aws_access_key_id
aws_secret_access_key
aws_session_token
credential_process
credential_source
external_id
mfa_serial
role_arn
role_session_name
source_profile
sso_account_id
sso_region
sso_role_name
sso_start_url
```

AWS ビルダー ID

AWS Builder ID は、Amazon でサードパーティーのリポジトリのクローンを作成するなど、特定のサービスや機能を使用するために必要な追加の AWS 認証方法です CodeCatalyst。

AWS Builder ID 認証方法の詳細については、「 サインインユーザーガイド」の「Builder ID AWS でAWS サインイン」トピックを参照してください。

CodeCatalyst から のリポジトリのクローンを作成する方法の詳細については AWS Toolkit for Visual Studio、このユーザーガイドの「Amazon の使用 CodeCatalyst」トピックを参照してください。

Toolkit for Visual Studio での多要素認証 (MFA)

多要素認証 (MFA) は、 AWS アカウントの追加のセキュリティです。MFA では、ウェブサイト AWS またはサービスにアクセスするときに、 AWS サポートされている MFA メカニズムからのサインイン認証情報と一意の認証を提供する必要があります。

AWS は、MFA 認証用の仮想デバイスとハードウェアデバイスの両方をサポートします。以下にスマートフォンアプリケーションを通じて有効化された仮想 MFA デバイスの例を示します。MFAデバイスオプションの詳細については、IAM ユーザーガイドの「AWSでの多要素認証 (MFA) の使用」を参照してください。

ステップ 1: IAM ユーザーにアクセス権を委任するための IAM ロールの作成

以下の手順では、IAM ユーザーにアクセス許可を割り当てるロールの委任を設定する方法について説明します。ロールの委任の詳細については、「AWS Identity and Access Management ユーザーガイド」の「IAM ユーザーにアクセス許可を委任するロールの作成」トピックを参照してください。

1. https://console.aws.amazon.com/iam/home の IAM コンソールに移動します。
2. ナビゲーションバーで [Roles] (ロール) を選択した後、[Create Role] (ロールの作成) を選択します。
3. [Create role] (ロールの作成) ページで、[Another AWS account] (もう 1 つの AWS アカウント) を選択します。
4. 必要な [Account ID] (アカウント ID) を入力し、[Require MFA] (MFA が必要な) チェックボックスをオンにします。

   > **Note**
   >
   > 12 桁のアカウント ID 番号を確認するには、コンソールのナビゲーションバーで [Support]、[Support Center] の順に選択します。

5. [次のステップ: アクセス許可] を選択します。
6. 既存のポリシーをロールにアタッチするか、新しいポリシーを作成します。このページで選択するポリシーによって、IAM ユーザーが Toolkit でアクセスできる AWS サービスが決まります。

7. ポリシーをアタッチしたら、ロールに IAM タグを追加するオプションのために、[Next: Tags] (次へ:タグ) を選択します。[Next: Review] (次へ: 確認) を選択して続行します。
8. [Review] (確認) ページで、必須の [Role name] (ロール名) (例えば、toolkit-role) を入力します。オプションの [Role description] (ロールの説明) 値を追加することもできます。
9. [ロールを作成] を選択します。
10. 確認メッセージ (「ロール toolkit-role が作成されました」など) が表示されたら、メッセージ内のロールの名前を選択します。
11. [Summary] (概要) ページで、[Copy] (コピー) アイコンを選択して [Role ARN] (ロールの ARN) にコピーし、ファイルに貼り付けます。この ARN は、ロールを引き受けるように IAM ユーザーを設定するときに必要です。

ステップ 2: ロールのアクセス許可を引き受ける IAM ユーザーの作成

このステップでは、インラインポリシーを追加できるように、アクセス許可のない IAM ユーザーを作成します。

1. https://console.aws.amazon.com/iam/home の IAM コンソールに移動します。
2. ナビゲーションバーで、 [Users] (ユーザー)、[Add user] (ユーザーを追加する) の順に選択します。
3. [Add user] (ユーザーの追加) ページで必要な [User name] (ユーザー名) (例えば、toolkit-user) を入力し、[Programmatic access] (プログラムによるアクセス) チェックボックスをオンにします。
4. [Next: Permissions] (次へ: アクセス許可)、[Next: Tags] (次へ: タグ)、[Next: Review] (次へ: 確認) の順に選択して、次ページに移動します。ユーザーがロールの権限を引き受けるため、この段階では権限を追加しません。
5. [Review] (確認) ページでは、[This user has no permissions] (このユーザーにはアクセス許可がありません) という旨が通知されます。[ユーザーの作成] を選択します。
6. [Success] (成功) ページで、[Download .csv] (.csv をダウンロード) を選択して、アクセスキー ID およびシークレットアクセスキーを含むファイルをダウンロードします。(認証情報ファイルのプロファイルを定義する場合は、両方とも必要です。)
7. [閉じる] を選びます。

ステップ 3: IAM ユーザーがロールを引き受けることを許可するポリシーを追加する

次の手順では、ユーザーがロール (およびそのロールのアクセス許可) を引き受けることを許可するインラインポリシーを作成します。

1. IAM コンソールの [Users] (ユーザー) ページで、作成した IAM ユーザー (例えば、toolkit-user) を選択します。
2. [Summary] (概要) ページの[Permissions] (アクセス許可) タブで [Add inline policy] (インラインポリシーの追加 を選択します。
3. [Create policy] (ポリシーの作成) ページで、[Choose a service] (サービスを選択) を選択し、[Find a service] (サービスを探す) で [STS] を入力し、結果から [STS] を選択します。
4. アクション で、 という用語の入力を開始しますAssumeRole。チェックボックスが表示されAssumeRoleたら、マークします。
5. [Resource] (リソース) セクションで、[Specific] (特定) が選択されていることを確認し、アクセスを制限するため [Add ARN] (ARN の追加) を選択します。
6. [Add ARN(s)] (ARN を追加する) ダイアログボックスの [Specify ARN for role] (ロールの ARN を指定します) にステップ 1 で作成したロールの ARN を追加します。

   ロールの ARN を追加すると、そのロールに関連付けられた信頼済みアカウントおよびロール名が [Account] (アカウント) および [Role name with path] (パス付きのロール名) に表示されます。
7. [追加] を選択します。
8. [Create policy] (ポリシーの作成) ページに戻り、[Specify request conditions (optional)] (リクエスト条件の指定 (オプション)) を選択し、[MFA required] (MFA が必要) チェックボックスをオンにしてから、確認するため [Close] (閉じる) を選択します。
9. [Review poilicy] (ポリシーの確認) を選択します。
10. [Review poilicy] (ポリシーの確認) ページで、ポリシーの [Name] (名前) を入力してから [Create policy] (ポリシーの作成) を選択します。

    [Permissions] (アクセス許可) タブには、IAM ユーザーに直接アタッチされた新しいインラインポリシーが表示されます。

ステップ 4: IAM ユーザーの仮想 MFA デバイスの管理

1. 仮想 MFA アプリケーションをスマートフォンにダウンロードしてインストールします。

サポートされているアプリケーションのリストについては、「多要素認証」リソースページを参照してください。

2. IAM コンソールで、ナビゲーションバーから、ユーザーを選択し、ロールを引き受けるユーザーを選択します (今回は toolkit-user)。
3. [Summary] (概要) ページで [Security credentials] (セキュリティ認証情報) タブを選択し、[Assigned MFA device] (割り当てられた MFA デバイス) に対して [Manage] (管理) を選択します。
4. [Manage MFA device] (MFA デバイスの管理) ペーンで、[Virtual MFA device] (仮想 MFA デバイス)、[Continue] (続行) の順に選択します。
5. [Set up virtual MFA device] (仮想 MFA デバイスの設定) ペインで [Show QR code] (QR コードを表示する) を選択してからスマートフォンにインストールした仮想 MFA アプリケーションを使用してコードをスキャンします。
6. QR コードをスキャンすると、仮想 MFA アプリケーションはワンタイムの MFA コードを生成します。[MFA code 1] (MFA コード 1) および [MFA code 2] (MFA コード 2) に 2 つの連続した MFAコードを入力します。
7. MFA の割り当てを選択します。
8. ユーザーの [Security credentials] (セキュリティ認証情報) タブに戻り、新しく[Assigned MFA device] (割り当てられた MFA デバイス) のARNをコピーします。

   ARN には 12 桁のアカウント ID が含まれ、形式は次の `arn:aws:iam::123456789012:mfa/toolkit-user` のようになります。この ARN は、次のステップで MFA プロファイルを定義するときに必要になります。

ステップ 5: MFA を許可するプロファイルの作成

次の手順では、Toolkit for Visual Studio から AWS サービスにアクセスするときに MFA を許可するプロファイルを作成します。

作成したプロファイルには、前の手順でコピーして保存した 3 つの情報が含まれています。

- IAM ユーザーのアクセスキー (アクセスキー ID およびシークレットアクセスキー)
- IAM ユーザーにアクセス許可を委任しているロールの ARN
- IAM ユーザーに割り当てられている仮想 MFA デバイスの ARN

認証情報を含む AWS 共有 AWS 認証情報ファイルまたは SDK ストアで、次のエントリを追加します。

```
[toolkit-user]
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

[mfa]
source_profile = toolkit-user
role_arn = arn:aws:iam::111111111111:role/toolkit-role
mfa_serial = arn:aws:iam::111111111111:mfa/toolkit-user
```

この例では、次の 2 つのプロファイルが定義されています：

- [toolkit-user] プロファイルには、ステップ 2 で IAM ユーザーを作成したときに生成され、保存されたアクセスキーおよびシークレットアクセスキーが含まれます。
- [mfa] プロファイルは、多要素認証のサポート方法を定義します。3 つのエントリがあります：

  ◦ source_profile: このプロファイル内のこの role_arn 設定で指定されたロールを継承するために使用される認証情報のプロファイルを指定します。この場合は、toolkit-user プロファイルです。

  ◦role_arn: このプロファイルを使用してリクエストされた操作の実行に使用する IAM ロールの Amazon リソースネーム (ARN) を指定します。この場合、ステップ 1 で作成したロールの ARN です。

  ◦mfa_serial: ロールを引き受けるときに使用する必要がある MFA デバイスの ID もしくはシリアル番号を指定します。この場合、ステップ 3 で設定した仮想デバイスの ARN です。

外部認証情報の設定

AWSで直接サポートされていない認証情報を生成または参照する方法がある場合、credential_process 設定を含むプロファイルを共有認証ファイルに追加することができます。この設定は、コマンドに使用する認証情報を生成もしくは取得するために実行する外部のコマンドを指定します。例えば、config ファイルに次のように似たエントリを含めることができます。

```
[profile developer]
credential_process = /opt/bin/awscreds-custom --username helen
```

外部認証情報の使用および関連するセキュリティリスクの詳細については、「AWS Command Line Interface ユーザーガイド」の「外部プロセスを使用した認証情報の調達」を参照してください。

AWS サービスの使用

以下のトピックでは、 AWS Toolkit for Visual Studio から AWS サービスの使用を開始する方法について説明します。

トピック

CodeCatalystAWSビジュアルスタジオ用ツールキット用アマゾン

Amazon CodeCatalyst とは

Amazon CodeCatalyst は、ソフトウェア開発チームのためのクラウドベースのコラボレーションスペースです。Visual Studio AWS 用ツールキットを使用すると、Visual Studio CodeCatalyst

AWS 用ツールキットから直接リソースを表示および管理できます。の詳細については CodeCatalyst、Amazon CodeCatalyst ユーザーガイドを参照してください。

以下のトピックでは、Visual Studio AWS CodeCatalyst 用ツールキットを接続する方法と、Visual Studio CodeCatalyst AWS 用ツールキットを介して操作する方法について説明します。

トピック

- Amazon CodeCatalyst 入門ガイドと Visual Studio AWS 用ツールキット
- Visual Studio AWS 用ツールキットからの Amazon CodeCatalyst リソースを使った作業
- トラブルシューティング

Amazon CodeCatalyst 入門ガイドと Visual Studio AWS 用ツールキット

Visual Studio CodeCatalyst AWS 用ツールキットから Amazon での作業を開始するには、以下を完了してください。

トピック

- Visual Studio AWS 用ツールキットのインストール
- CodeCatalystAWSアカウントとビルダー ID の作成
- Visual Studio AWS 用ツールキットをとの接続 CodeCatalyst

Visual Studio AWS 用ツールキットのインストール

Visual Studio AWS CodeCatalyst 用ツールキットをアカウントと統合する前に、Visual Studio AWS 用ツールキットの最新バージョンを使用していることを確認してください。最新バージョンの AWS Toolkit for Visual Studio をインストールしてセットアップする方法の詳細については、このユーザーガイドの「Visual Studio AWS 用ツールキットのセットアップ」セクションを参照してください。

CodeCatalystAWSアカウントとビルダー ID の作成

Visual Studio AWS 用ツールキットの最新バージョンをインストールすることに加えて、Visual Studio AWS 用ツールキットに接続するには、AWSアクティブなビルダー ID CodeCatalyst とアカウントが必要です。有効な AWS Builder ID CodeCatalyst またはアカウントをお持ちでない場合は、CodeCatalystユーザーガイドの「Setup with CodeCatalyst」セクションを参照してください。

> **Note**
>
> AWSビルダー ID AWS は認証情報とは異なります。AWSBuilder ID を使用してサインアップして認証する方法については、このユーザーガイドの「認証とアクセス:AWSBuilder ID」トピックを参照してください。
> AWSビルダー ID の詳細については、『AWS一般リファレンスユーザーガイド』の「AWSビルダー ID」トピックを参照してください。

Visual Studio AWS 用ツールキットをとの接続 CodeCatalyst

Visual Studio AWS CodeCatalyst 用ツールキットをアカウントに接続するには、次の手順を実行してください。

1. Visual Studio の Git メニュー項目から、「リポジトリをクローン...」を選択します。 。
2. 「リポジトリを参照」セクションから、CodeCatalystプロバイダーとして Amazon を選択します。
3. 「接続」セクションから「AWSビルダー ID で接続」を選択し、お好みの Web CodeCatalyst ブラウザーでコンソールを開きます。
4. ブラウザから、AWS表示されたフィールドにビルダー ID を入力し、指示に従って続行します。
5. メッセージが表示されたら、[許可] を選択して AWS Toolkit for Visual Studio CodeCatalyst とアカウント間の接続を確認します。接続プロセスが完了すると、CodeCatalystブラウザを閉じても問題ないことを示す確認メッセージが表示されます。

Visual Studio AWS 用ツールキットからの Amazon CodeCatalyst リソースを使った作業

以下のセクションでは、Visual Studio AWS 用ツールキットで使用できる Amazon CodeCatalyst リソース管理機能の概要を説明します。

トピック

- リポジトリを複製する

リポジトリを複製する

CodeCatalystはクラウドベースのサービスで、CodeCatalystプロジェクトに取り組むにはクラウドに接続する必要があります。プロジェクトをローカルで作業するには、CodeCatalystリポジトリをローカルマシンに複製し、CodeCatalyst次にクラウドに接続したときにプロジェクトと同期できます。

リポジトリをローカルマシンに複製するには、次の手順を実行します。

1. Visual Studio の Git メニュー項目から、「リポジトリをクローン...」を選択します。 。
2. 「リポジトリを参照」セクションから、CodeCatalystプロバイダーとして Amazon を選択します。

> **Note**
>
> **Not Connected**接続セクションにメッセージが表示される場合は、先に進む前に、このユーザーガイドの「認証とアクセス:AWSビルダーID」セクションの手順を完了してください。

3. リポジトリのクローンを作成するスペースとプロジェクトを選択します。
4. 「リポジトリ」セクションから、クローンを作成するリポジトリを選択します。
5. パスセクションから、リポジトリのクローンを作成するフォルダーを選択します。

> **Note**
>
> クローンを正常に作成するには、このフォルダを最初は空にする必要があります。

6. [クローン] を選択して、リポジトリのクローンを開始します。
7. リポジトリがクローンされると、Visual Studio はクローンされたソリューションをロードします。

> **Note**
>
> Visual Studio がクローンされたリポジトリでソリューションを開かない場合、Visual Studio のオプションは、[ソース管理] メニューの [Git グローバル設定] にある [Git リポジトリを開くときにソリューションを自動的に読み込む] 設定から調整できます。

トラブルシューティング

以下は、Visual Studio CodeCatalyst AWS 用ツールキットから Amazon を使用する際の既知の問題に対処するためのトラブルシューティングトピックです。

トピック

- 認証情報

認証情報

Git ベースのリポジトリをクローンしようとしたときに認証情報を求めるダイアログが表示される場合CodeCatalyst、AWSCodeCommitCredential ヘルパーがグローバルに設定されているために干渉が発生する可能性があります。CodeCatalystAWSCodeCommit認証情報ヘルパーの詳細については、ユーザーガイドの「AWSCLI 認証情報ヘルパーを使用して Windows AWS CodeCommit 上のリポジトリへの HTTPS 接続の設定手順」セクションを参照してください。AWSCodeCommit

AWSCodeCommit認証情報ヘルパーが CodeCommit URL のみを処理するように制限するには、次の手順を実行してください。

1. グローバルgit設定ファイルを次の場所で開きます。`%userprofile%\.gitconfig`
2. ファイル内の次のセクションを探してください。

```
[credential]
    helper = !aws codecommit credential-helper $@
    UseHttpPath = true
```

3. そのセクションを次のように変更します。

```
[credential "https://git-codecommit.*.amazonaws.com"]
    helper = !aws codecommit credential-helper $@
    UseHttpPath = true
```

4. 変更を保存し、手順を実行してリポジトリを複製します。

アマゾン CloudWatch Visual Studio のログ統合

アマゾン CloudWatch Logs 統合はAWSToolkit for Visual Studio は、監視、保存、アクセスする機能を提供します CloudWatch IDE から離れることなくリソースをログに記録します。の設定の詳細は、CloudWatch サービスと使用方法 CloudWatch ログ機能については、次のトピックから選択します。

トピック

- セットアップ CloudWatch ログ統合 for Visual Studio
- の使用 CloudWatch Visual Studio でのログ

セットアップ CloudWatch ログ統合 for Visual Studio

Amazonを使う前に CloudWatch Toolkit for Visual Studio との統合をログに記録するにはAWSアカウント. 新しいものを作成しますAWSアカウントからAWSサインインサイト。ほとんどの CloudWatch Toolkit for Visual Studio から利用できるログ機能は、アクティブの状態でアクセスできます。AWS認証情報。特定の機能に追加の設定が必要な場合、その要件はの使用 CloudWatch ログガイド

設定に関する追加情報とオプションについて CloudWatch ログ、を参照してください準備作業アマゾンのセクション CloudWatch ログガイド。

の使用 CloudWatch Visual Studio でのログ

アマゾン CloudWatch ログの統合により、モニタリング、保存、アクセスが可能になります CloudWatch ログログログのAWSToolkit for Visual Studio へのアクセス権がある CloudWatch IDEから離れることなくログ機能を使用することで、 CloudWatch 開発プロセスを記録し、ワークフローの中断を減らします。次のトピックでは、の基本的な特徴と機能の操作方法について説明します。CloudWatch ログの統合

トピック

- CloudWatch ロググループ
- CloudWatch ログストリーム
- CloudWatch ログイベント
- への追加アクセス CloudWatchログ

CloudWatch ロググループ

あるlog groupのグループですかlog streams保持、監視、アクセス制御について同じ設定を共有します。1 つのロググループに属することができるログストリーミングの数に制限はありません。

ロググループのロググループを表示する

-View Log Groups機能を使用すると、ロググループのリストが表示されます。 CloudWatchロググループエクスプローラー。

ロググループの表示機能にアクセスし、 CloudWatch ロググループエクスプローラーでは、以下の手順を完了します。

1. 開始元AWSエクスプローラー、展開アマゾン CloudWatch。
2. ダブルクリックロググループまたはコンテキストメニュー (右クリック) を開き、表示をクリックして、CloudWatch ロググループエクスプローラー。

> **Note**
>
> - CloudWatch ロググループエクスプローラーは、ソリューションエクスプローラーと同じウィンドウの場所で開きます。

ロググループのフィルタリング

個人アカウントには、何千もの異なるロググループを含めることができます。特定のグループの検索を簡略化するには、filtering機能の詳細は以下のとおりです。

1. 開始元CloudWatch ロググループエクスプローラー、ウィンドウの上部にある検索バーにカーソルを置きます。
2. 探しているロググループに関連する接頭辞の入力を開始します。
3. CloudWatch ロググループエクスプローラーが自動的に更新され、前の手順で指定した検索語に一致する結果が表示されます。

ロググループを削除する

特定のロググループを削除するには、以下の手順を参照してください。

1. 開始元CloudWatch ロググループエクスプローラーで、削除するロググループを右クリックします。
2. プロンプトが表示されたら、現在選択されているロググループを削除することを確認します。
3. を選ぶはいボタンをクリックすると、選択したロググループが削除され、CloudWatch ロググループエクスプローラー。

ロググループを更新する

に表示されているロググループの現在のリストを更新するにはCloudWatch ロググループエクスプローラーで、[更新] アイコンボタンはツールバー。

ロググループ ARN のコピー

特定のロググループの ARN をコピーするには、以下に説明する手順を実行します。

1. 開始元CloudWatch ロググループエクスプローラーで、ARN のコピー元のロググループを右クリックします。
2. [ARN をコピー[] をクリックします。
3. これで ARN がローカルクリップボードにコピーされ、貼り付ける準備が整いました。

CloudWatch ログストリーム

ログストリームは、同じソースを共有する一連のログイベントです。

> **Note**
>
> ログストリームを表示するときは、以下のプロパティに注意してください。
>
> - デフォルトでは、ログストリームは最新のイベントタイムスタンプでソートされます。
> - ログストリームに関連付けられている列と列に関連付けられている列は、キャレット列見出しにあります。
> - フィルタされたエントリは以下でのみソートできます[ログストリーム名]。

ログストリームの表示

1. 開始元CloudWatch ロググループエクスプローラーロググループをダブルクリックするか、ロググループを右クリックしてログストリームコンテキストメニューから、

2. 新しいタブが開きます資料ウィンドウには、ロググループに関連付けられているログストリームのリストが含まれています。

ログストリームのフィルタリング

1. 開始元ログストリームタブの資料ウィンドウで、検索バーにカーソルを置きます。
2. 探しているログストリームに関連する接頭辞の入力を開始します。
3. 入力すると、現在の表示が自動的に更新され、入力によってログストリームがフィルタリングされます。

ログストリームの更新

に表示されているログストリームの現在のリストを更新するには資料ウィンドウで、[更新] アイコンボタン、ツールバーをクリックして、検索バー。

ログストリーム ARN

特定のログストリームの ARN をコピーするには、以下に説明する手順を実行します。

1. 開始元ログストリームタブの資料ウィンドウで、ARN のコピー元のログストリームを右クリックします。
2. [ARN をコピー[] をクリックします。
3. これで ARN がローカルクリップボードにコピーされ、貼り付ける準備が整いました。

ログストリームのダウンロード

-ログストリーム機能は、選択したログストリームをローカルにダウンロードして保存し、カスタムツールやソフトウェアからアクセスして追加の処理を行うことができます。

1. 開始元ログストリームタブの資料ウィンドウで、ダウンロードするログストリームを右クリックします。
2. 選択ログストリームをクリックして、テキストファイルにエクスポートするダイアログダイアログを開きます。
3. ファイルをローカルに保存する場所を選択し、表示されたテキストフィールドに名前を指定します。
4. 選択してダウンロードを確定しますOK。ダウンロードのステータスは、Visual Studio タスクステータスセンター

CloudWatch ログイベント

ログイベントは、によって監視されているアプリケーションまたはリソースによって記録されたアクティビティのレコードです。 CloudWatch。

ログイベントアクション

ログイベントはテーブルとして表示されます。デフォルトでは、イベントは最も古いイベントから最新のイベントにソートされます。

Visual Studio のログイベントには、次のアクションが関連付けられています。

- 折り返しテキストモード: イベントをクリックすると、折り返しテキストを切り替えることができます。
- [テキストの折り返し] ボタン:`document window` **`toolbar`**を選択すると、このボタンはすべてのエントリに対してテキストの折り返しのオンとオフを切り替えます。
- メッセージをクリップボードにコピーする:コピーするメッセージを選択し、選択したメッセージを右クリックしてコピー(ECS)]`Ctrl + C`).

ログイベントのログイベントを表示する

1. 開始元資料ウィンドウで、ログストリームのリストを含むタブを選択します。
2. ログストリームをダブルクリックするか、ログストリームを右クリックしてログストリームメニューをクリックします。
3. 新規ログイベントタブが開き、資料ウィンドウ。選択したログストリームに関連するログイベントのテーブルが表示されます。

ログイベントのフィルタリング

ログイベントをフィルターするには、コンテンツ、時間範囲、またはその両方の 3 つの方法があります。ログイベントをコンテンツと時間範囲の両方でフィルタリングするには、まずコンテンツまたは時間範囲でメッセージをフィルタリングし、その結果を別の方法でフィルタリングします。

ログイベントをコンテンツでフィルタリングするには、次の手順を実行します。

1. 開始元ログイベントタブの資料ウィンドウで、ウィンドウの上部にある検索バーにカーソルを置きます。
2. 検索しているログイベントに関連する用語または語句の入力を開始します。

3. 入力すると、現在の表示が自動的にログイベントのフィルタリングを開始します。

> **Note**
>
> フィルターパターンでは大文字と小文字が区別されます。英数字以外の文字を含む正確なフレーズとフレーズとフレーズとフレーズとフレーズとフレーズとフレーズとフレーズとフレーズとフレーズと語句とフレーズとフレーズとフレーズとフレーズとフレーズとフレーズとフレーズとフレーズとフレーズ フィルタパターンの詳細については、フィルターとパターンの構文アマゾンのトピック CloudWatch ガイドガイド

特定の時間範囲に生成されたログイベントを表示するには、次の手順を実行します。

1. 開始元ログイベントタブの資料ウィンドウで、カレンダーアイコンボタン、ツールバー。
2. 提供されたフィールドを使用して、検索する時間範囲を指定します。
3. フィルターされた結果は、日付と時刻の制約を指定すると自動的に更新されます。

> **Note**
>
> -CalSオプションはあなたの現在のものをすべてクリアします date-and-time フィルタ選択をフィルタします。

ログログイベントを更新する

に表示されているログイベントの現在のリストを更新するにはログイベントタブで、[更新] アイコンボタン、ツールバー。

への追加アクセス CloudWatchログ

アクセスすることができます CloudWatch 他に関連付けられているログAWSサービスとリソースを直接AWSビジュアルスタジオのツールキット。

Lambda

Lambda 関数に関連付けられているログストリームを表示するには:

> **Note**
>
> Lambda 実行ロールには、ログを送信するための適切なアクセス権限が必要です CloudWatchログ。以下のために必要な Lambda アクセス権限の詳細については CloudWatch ログ、を参照してくださいhttps://docs.aws.amazon.com/lambda/latest/dg/monitoring-cloudwatchlogs.html#monitoring-cloudwatchlogs-prereqs

1. 開始元AWSツールキットエクスプローラー、展開Lambda。
2. 表示する関数を右クリックし、[] を選択しますログを表示するをクリックして、関連するログストリームを資料Windows。

Lambda インテグレーションを使用してログストリームを表示するには`function view`:

1. 開始元AWSツールキットエクスプローラー、展開Lambda。
2. 表示する関数を右クリックし、[] を選択します表示機能をクリックしてファンクションビューを資料Windows。
3. 開始元`function view`をクリックして、ログタブでは、選択した Lambda 関数に関連付けられているログストリームが表示されます。

ECS

ECS タスクコンテナに関連付けられているログリソースを表示するには、次の手順を実行します。

> **Note**
>
> Amazon ECS サービスがログをに送信するには CloudWatchでは、特定の Amazon ECS タスクの各コンテナは、必要な設定を満たす必要があります。必要なセットアップと構成の詳細については、ガイドを参照してくださいの使用AWSログログログドライバー。

1. 開始元AWSツールキットエクスプローラー、展開Amazon ECS。
2. 新しいクラスターを開くために表示する Amazon ECS クラスターを選択します。ECS クラスタータブの資料Windows。
3. の左側にあるナビゲーションメニューから、ECS クラスタータブで、タスクをクリックして、クラスターに関連付けられているすべてのタスクをリスト表示します。

4. 開始元タスク表示し、タスクを選択し、ログを表示するリンク、左下隅にあります。

> **Note**
>
> この画面には、クラスタに含まれるすべてのタスクが一覧表示されます。`View Logs`リンクは、必要なログ設定を満たす各タスクに対してのみ表示されます。
>
> - タスクが単一のコンテナにのみ関連付けられている場合、ログを表示するリンクはそのコンテナのログストリームを開きます。
> - タスクが複数のコンテナに関連付けられている場合、ログを表示するリンクを開くと表示 CloudWatch ECS タスクタスクダイアログで、コンテナ:ドロップダウンメニューでログを表示するコンテナを選択し、OK。

5. 新しいタブが開きます資料コンテナの選択に関連するログストリームを表示するウィンドウ。

Amazon EC2 インスタンスの管理

AWSExplorer では、Amazon Machine Image (AMI) と Amazon Elastic Compute Cloud (Amazon EC2) インスタンスの詳しいビューが用意されています。これらのビューで、AMI から Amazon EC2 インスタンスを起動、そのインスタンスに接続、インスタンスを停止または終了できます。すべて Visual Studio 開発環境から可能です。インスタンスビューを使用して、使用するインスタンスから AMI を作成できます。詳細については、「Amazon EC2 インスタンスから AMI を作成する」を参照してください。

Amazon マシンイメージビューと Amazon EC2 インスタンスビュー

送信元AWSExplorer では、Amazon Machine Image (AMI) と Amazon EC2 インスタンスのビューを表示できます。EclipseAWSエクスプローラ、Amazon EC2ノード。

AMI のビューを表示するには、最初の、[AMIs (AMI)] サブノードでコンテキスト (右クリック) メニューを開き、[View (新規)] を選択します。

Amazon EC2 インスタンスビューを表示するには、[Instances (インスタンス)] ノードで、コンテキスト (右クリック) メニューを開き、[View (ビュー)] を選択します。

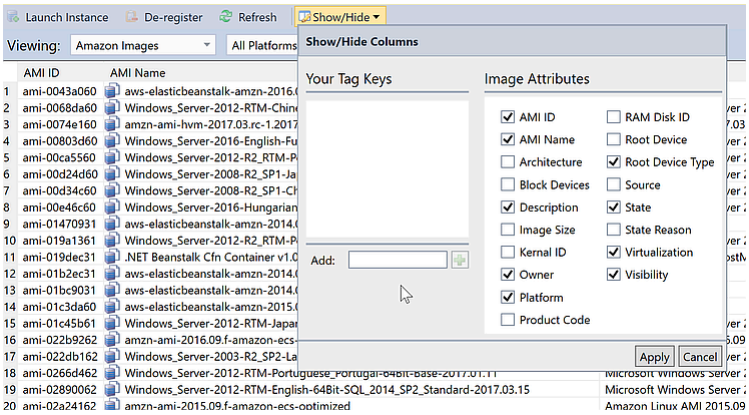
いずれのビューも、該当するノードをダブルクリックして、表示することもできます。

- ビューのスコープは、で指定したリージョンに限定されます。AWSExplorer (たとえば、米国西部 (北カリフォルニア) リージョン)

- クリックしてドラッグすると列をソートすることができます。列で値をソートするには、列見出しをクリックします。
- [Viewing (表示する)] のフィルタボックスとドロップダウンリストを使用してビューを設定できます。初期ビューには、で指定されたアカウントが所有している任意のプラットフォームタイプ (Windows または Linux) の AMI が表示されます。AWSExplorer。

列の表示/非表示

[Show/Hide (表示/非表示)] ドロップダウンをビューの上部で選択して、表示する列の種類を設定することもできます。ビューを閉じて、再度開いた場合でも、表示する列の選択は保持されます。



AMI とインスタンスビューの [Show/Hide Columns (列の表示/非表示)] UI

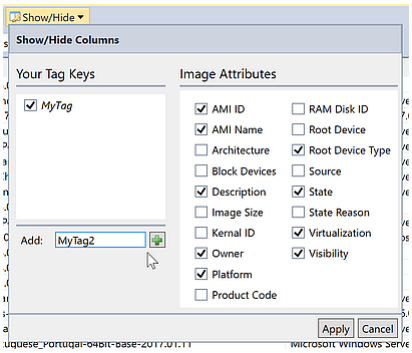
AMI、インスタンス、ボリュームのタグ付け

また、 を使用することもできます[Show/HideAMI、Amazon EC2 インスタンス、自分のボリュームにタグを追加するためのドロップダウンリストが表示されます。タグとは名前と値のペアで、インスタンス、ボリューム、AMI にメタデータをアタッチできます。タグ名の限定範囲は、お客様のアカウントと、AMI とインスタンスごとの両方があります。たとえば、AMI とインスタンスに同じタグ名を使用した場合、競合にはなりません。タグ名では大文字と小文字が区別されません。

タグの詳細については、「」を参照してください。タグの使用のLinux インスタンス用 Amazon EC2 ユーザーガイド。

タグを追加するには

1. [Add (追加)] ボックスにタグの名前を入力します。緑のプラス記号 (+) ボタンを選択し、[Apply (適用)] を選択します。



AMI または Amazon EC2 インスタンスにタグを追加する

新しいタグはイタリック体で表示されます。そのタグに関連付けられている値がないことを示します。

リストビューでは、新しい列としてタグ名が表示されます。少なくとも 1 つの値がタグに関連付けられているとき、タグはAWS Management Console。

2. タグの値を追加するには、該当するタグの列でセルをダブルクリックし、値を入力します。タグの値を削除するには、セルをダブルクリックして、値のテキストを削除します。

   [Show/Hide (表示/非表示)] ドロップダウンリストでタグをクリアすると、ビューから対応する列が消えます。タグは、AMI、インスタンス、またはボリュームに関連付けられた任意のタグ値とともに保持されます。

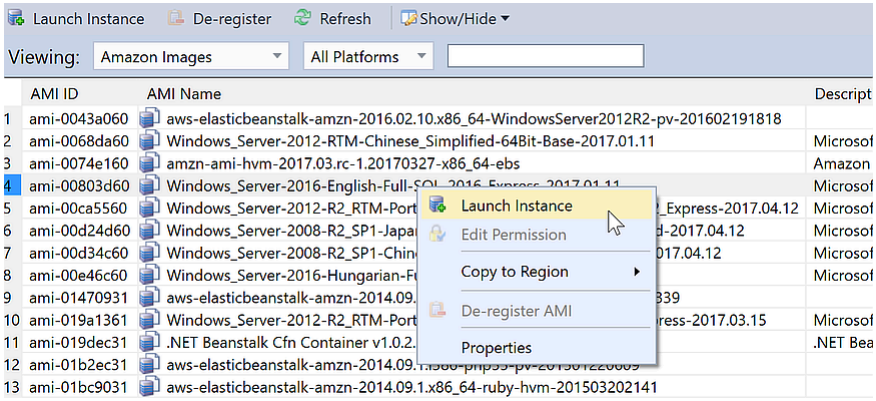
> **Note**
>
> でタグをクリアすると[Show/Hide値が関連付けられていないドロップダウンリストでは、AWSToolkit はタグを完全に削除します。このリストビューや [Show/Hide (表示/非表示)] ドロップダウンリストに表示されなくなります。再度そのタグを使用するには、[Show/Hide (表示/非表示)] ダイアログボックスを使用して再作成します。

Amazon EC2 インスタンスを起動する

AWSExplorer では、Amazon EC2 インスタンスを起動するために必要なすべての機能を提供します。このセクションでは、Amazon Machine Image (AMI) を選択し、設定を行い、Amazon EC2 インスタンスとして起動します。

Windows Server の Amazon EC2 インスタンスを起動するには

1. AMI ビューの上部にある、左側のドロップダウンリストから [Amazon Images] を選択します。右側のドロップダウンリストで、[Windows] を選択します。フィルタボックスに、Elastic Block Storage に対する ebs を入力してください。ビューが更新されるまでに数分かかることがあります。
2. リストで AMI を選択し、コンテキスト (右クリック) メニューを開き、[Launch Instance (起動インスタンス)] を選択します。



AMI リスト

3. [Launch New Amazon EC2 Instance (新しい Amazon EC2 インスタンスを起動する)] ダイアログボックスで、アプリケーション用に AMI を設定します。

インスタンスタイプ

起動する EC2 インスタンスのタイプを選択します。インスタンスのタイプと料金については、EC2 料金表に関するページにアクセスしてください。

[Name] (名前)

インスタンスの名前を入力します。256 文字を超える名前は使用できません。

キーペア

キーペアは Windows パスワードを取得するために使用されます。これは Remote Desktop Protocol (RDP) を使用して EC2 インスタンスにログインするために使用します。プライベートキーにアクセスするためのキーペアを選択します。またはキーペアを作成するオプションを選択します。Toolkit でキーペアを作成した場合、Toolkit でプライベートキーを保存できます。

ツールキットに保存されているキーペアは暗号化されます。それらは%LOCALAPPDATA%\AWSToolkit\keypairs(通常: C:\Users\<user>\AppData\Local\AWSToolkit\keypairs). 暗号化されたkey pair を.pemファイルを開きます。

a. Visual Studio で、表示[] をクリックし、AWSエクスプローラ。

b. [Amazon EC2] をクリックし、[Key Pairs (キーペア)] を選択します。

c. キーペアが一覧表示され、Toolkit で作成/管理されているものは、[Stored in AWSToolkit (AWSToolkit で保存)] とマークされます。

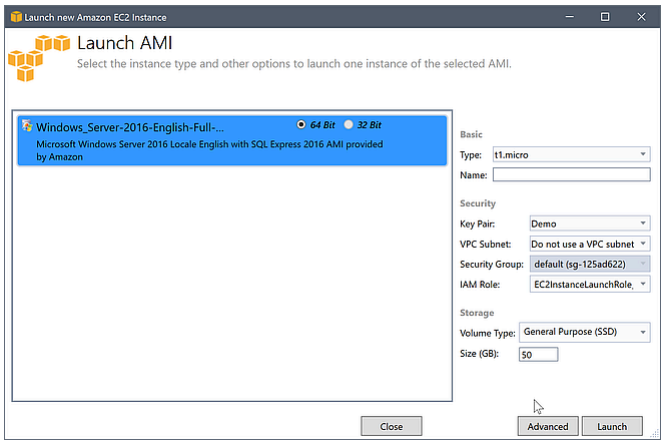
d. 作成したキーペアを右クリックし、[Export Private Key (プライベートキーのエクスポート)] を選択します。プライベートキーが復号され、指定した場所に保存されます。

セキュリティグループ

セキュリティグループを使用して、EC2 インスタンスに許可するネットワークトラフィックのタイプを制御します。セキュリティグループではポート 3389 上の受信トラフィックを有効にして、EC2 インスタンスへの RDP 接続を許可する必要があります。Toolkit を使用してセキュリティグループを作成する方法については、を参照してください。からのセキュリティグループの管理AWSエクスプローラ。

インスタンスプロファイル

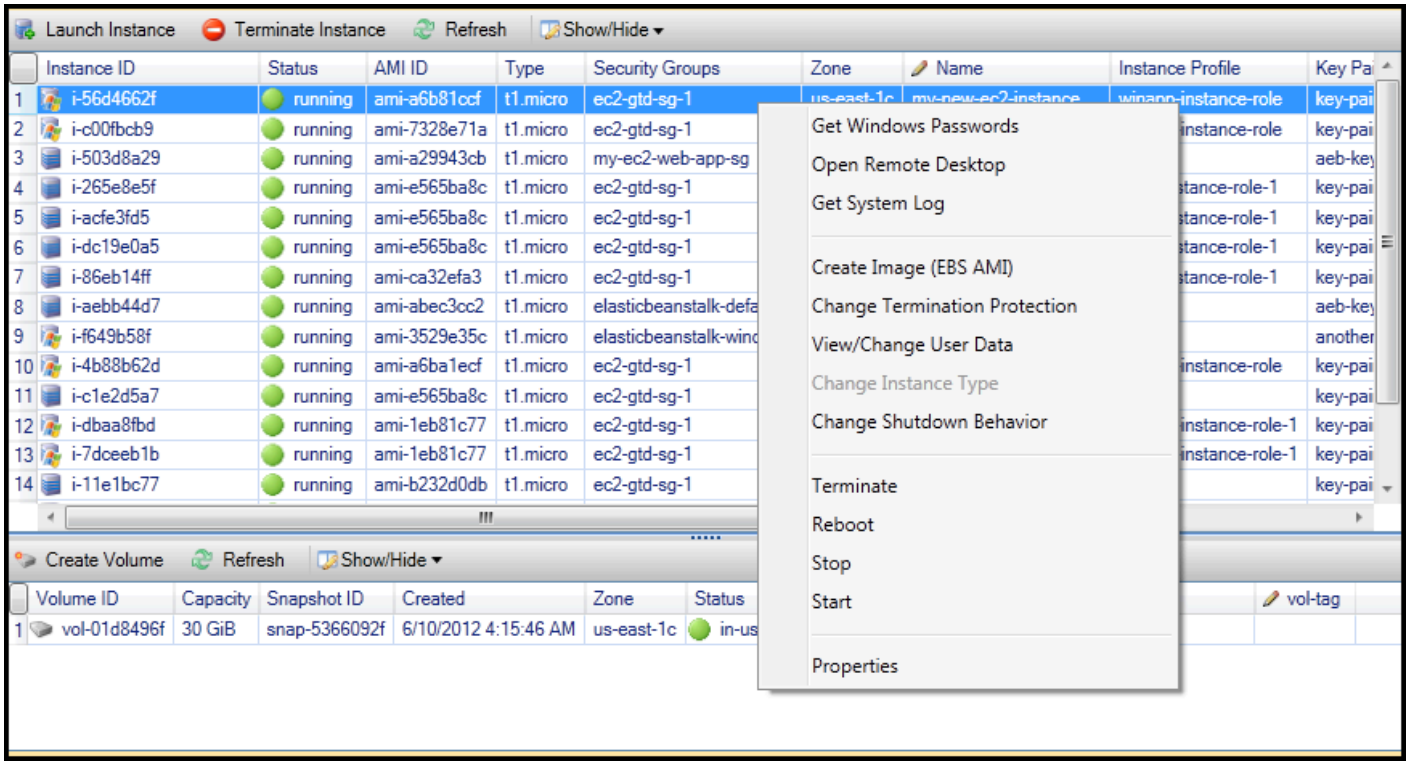
インスタンスプロファイルは IAM ロールの論理コンテナです。インスタンスプロファイルを選択するとき、対応する IAM ロールを EC2 インスタンスに関連付けます。IAM ロールには、Amazon Web Services アカウントリソースへのアクセスを指定するポリシーを設定します。EC2 インスタンスを IAM ロールに関連付けると、インスタンスで実行されるアプリケーションソフトウェアには、IAM ロールに指定したアクセス許可が付与されます。これにより、アプリケーションソフトウェアはそれ自体の AWS 認証情報を指定しなくても実行可能になり、より安全になります。IAM ロールの詳細については、「IAM ユーザーガイド」を参照してください。



EC2 の [Launch AMI (AMI の起動)] ダイアログボックス

4. [Launch] (起動) を選択します。

EclipseAWSエクスプローラー、インスタンスのサブノードAmazon EC2[] を選択してから、コンテキスト (右クリック) メニューを開き、[] を選択します。表示。-AWSToolkit では、アクティブなアカウントに関連付けられている Amazon EC2 インスタンスのリストが表示されます。新しいインスタンスを表示するには、[Refresh (更新)] を選択する必要があります。最初にインスタンスが表示されたときは、インスタンスが保留状態であることがありますが、しばらくすると、実行状態に移行します。



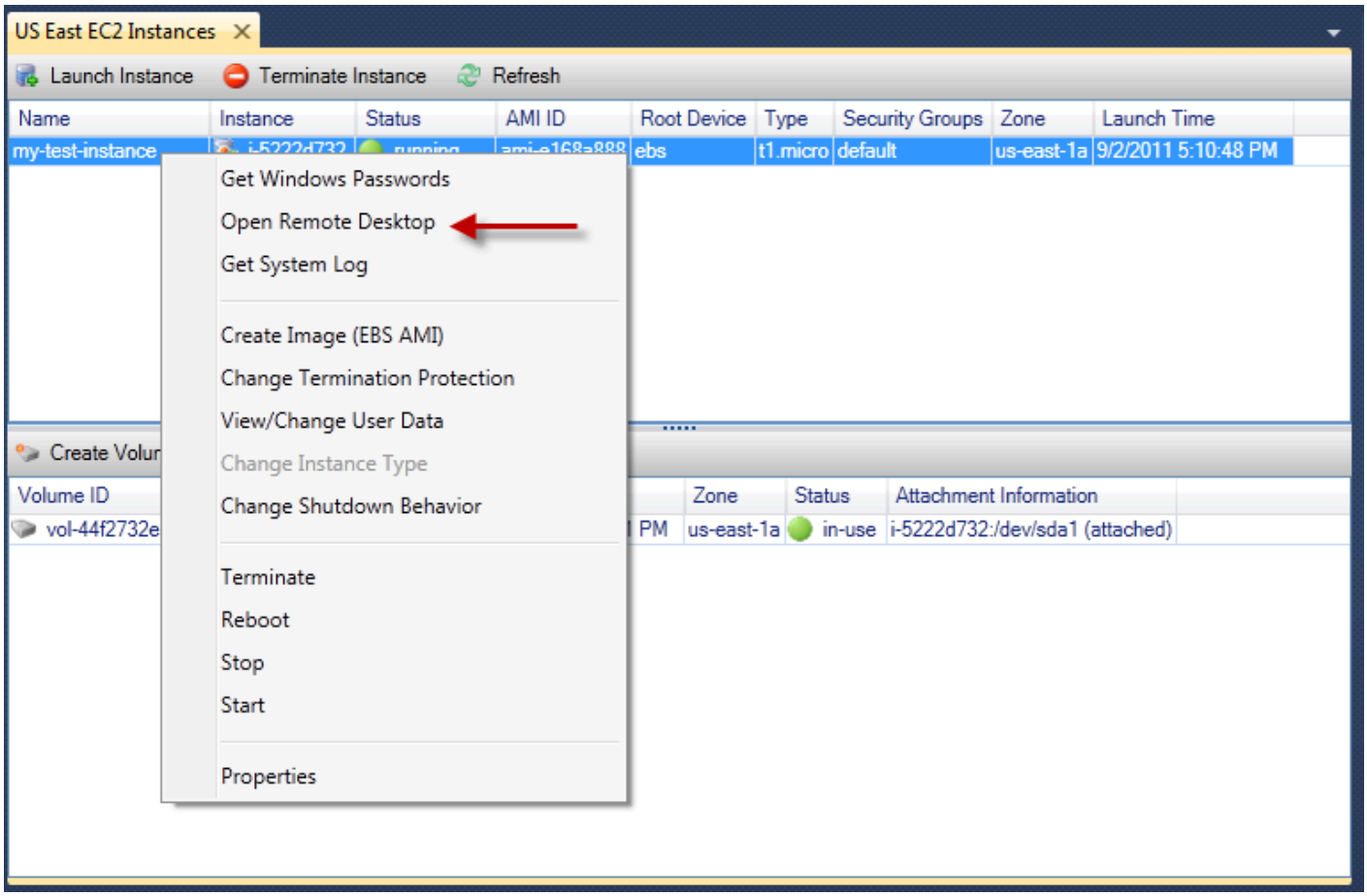
Amazon EC2 インスタンスへの接続

Windows リモートデスクトップを使用して、Windows Server インスタンスに接続します。認証の場合、AWSToolkit では、インスタンスの管理者パスワードを取得できます。またはインスタンスに関連付けられた格納key pair を単に使用できます。次の手順では、保存されたキーペアを使用します。

Windows リモートデスクトップを使用して、Windows Server インスタンスに接続するには

1. EC2 インスタンスのリストで、接続する対象の Windows Server インスタンスを右クリックします。コンテキストメニューから [Open Remote Desktop (リモートデスクトップを開く)] を選択します。

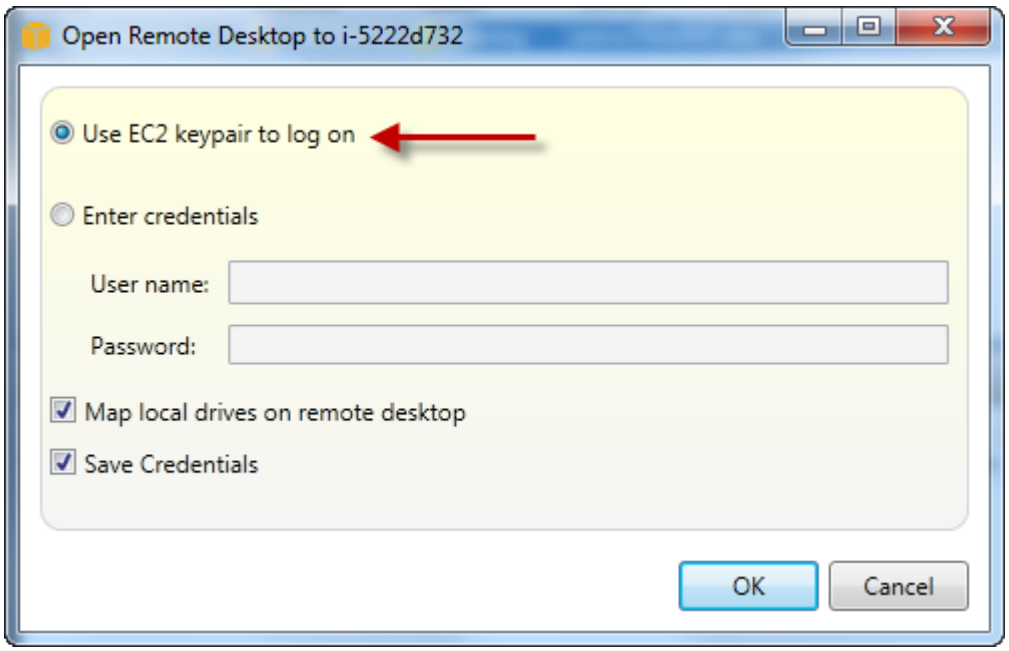
管理者パスワードを使用して認証する場合は、[Get Windows Passwords (Windows のパスワードを取得)] を選択します。



EC2 インスタンスのコンテキストメニュー

2. [Open Remoe Desktop (リモートデスクトップを開く)] ダイアログボックスで、[Use EC2 keypair to log on (EC2 キーペアを使用してログオンする)] を選択し、[OK] を選択します。

   key pair をキーペアに保存していない場合は、AWSToolkit で、秘密キーを含む PEM ファイルを指定します。

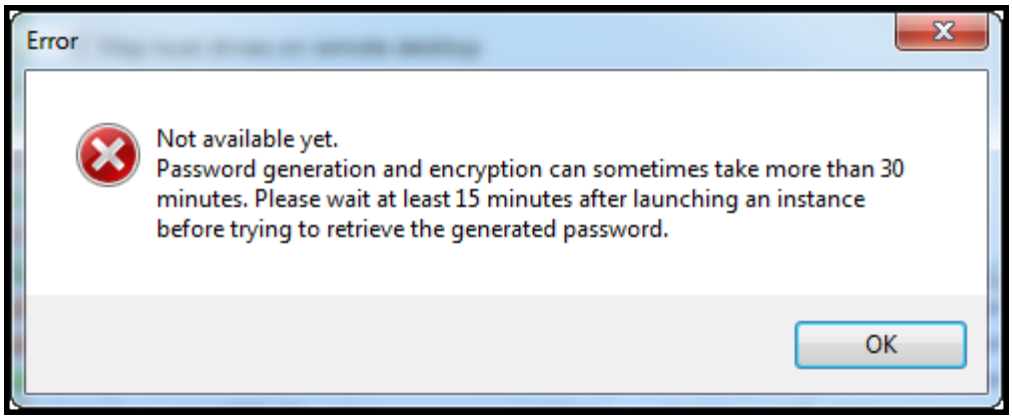


[Open Remote Desktop (リモートデスクトップを開く)] ダイアログボックス

3. リモートデスクトップウィンドウが開きます。キーペアで認証されているため、サインインする必要はありません。Amazon EC2 インスタンスでユーザーは管理者となります。

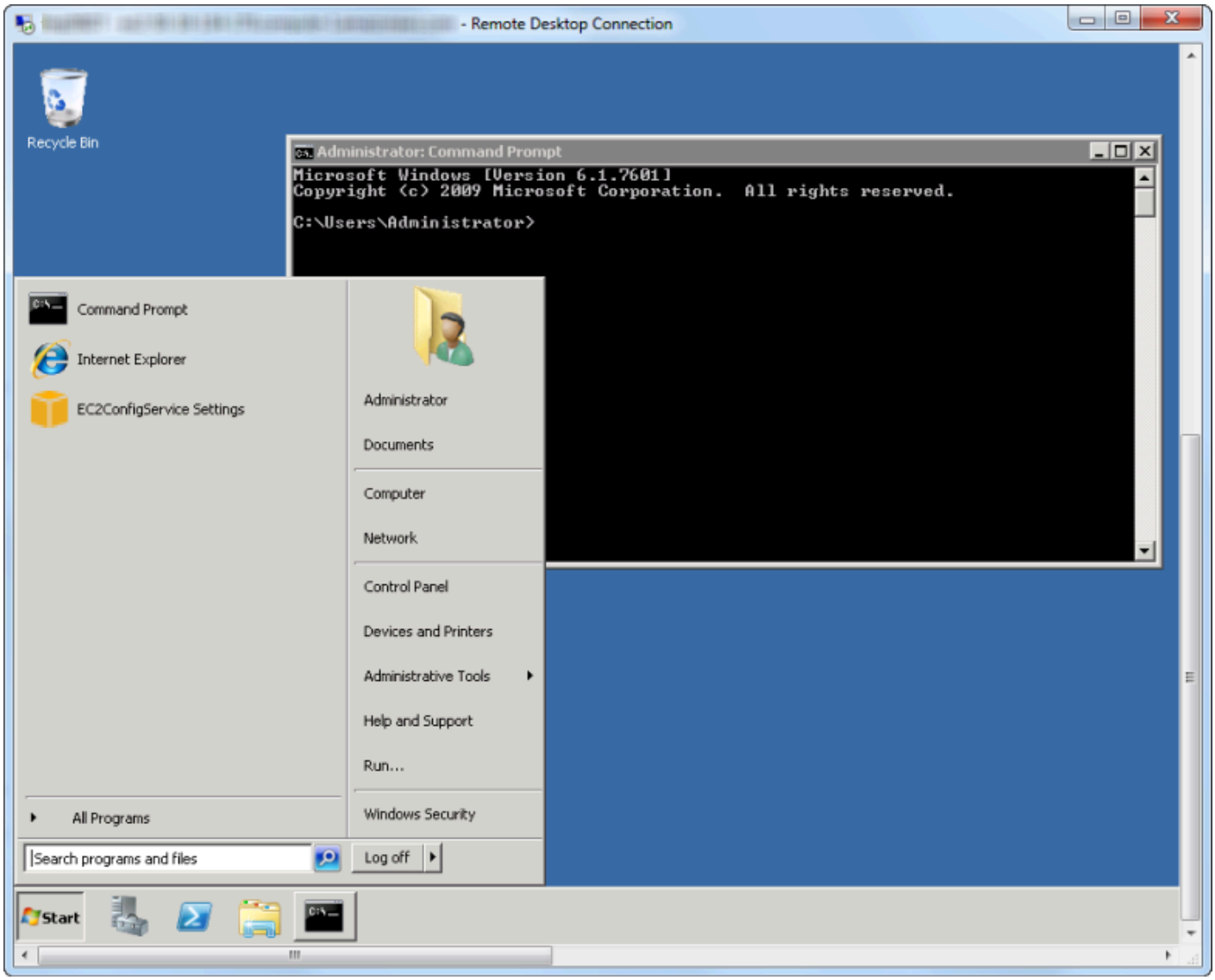
   EC2 インスタンスが最近開始された場合のみ、接続できない可能性があります。次の 2 つの理由が考えられます。

   - リモートデスクトップサービスがまだ稼働していません。数分後にもう一度お試しください。
   - パスワード情報がまだインスタンスに転送されていません。この場合、次のようなメッセージボックスが表示されます。



パスワードはまだ使用できません

次のスクリーンショットでは、ユーザーがリモートデスクトップを介して管理者として接続されています。



リモートデスクトップ

Amazon EC2 インスタンスを削除する

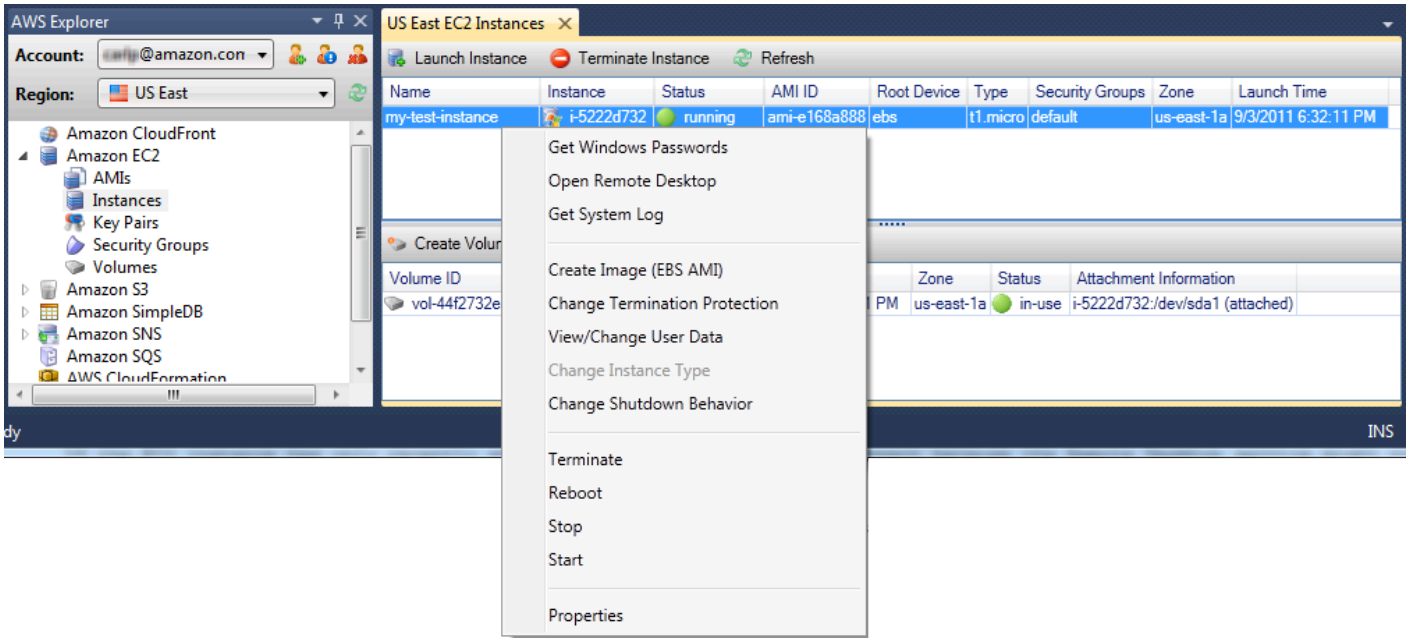
の使用AWSToolkit、Visual Studio から実行中の Amazon EC2 インスタンスを停止または終了できます。インスタンスを停止するには、EC2 インスタンスが Amazon EBS ボリュームを使用している必要があります。EC2 インスタンスが Amazon EBS ボリュームを使用していない場合は、インスタンスを削除するオプションのみが表示されます。

インスタンスを停止した場合でも、EBS ボリュームに保存されているデータは保持されます。インスタンスを終了した場合は、インスタンスのローカルストレージデバイスに保存されているデータはすべて失われます。いずれの場合も、停止または終了すると、EC2 インスタンスへの課金は継続されません。ただし、インスタンスを停止した場合は、インスタンスが停止した後も保持される EBS ストレージに対しては、引き続きお客様への課金が発生することになります。

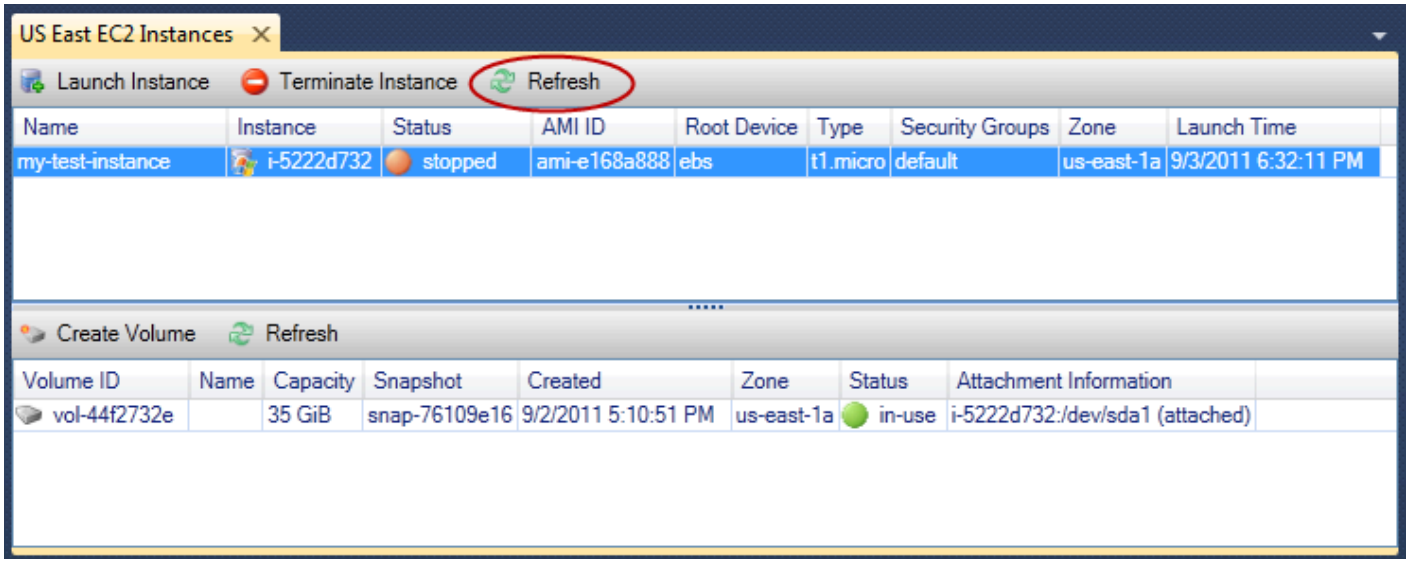
インスタンスを削除するもう 1 つの可能な方法は、インスタンスに接続しているリモートデスクトップを使用することです。Windows の [スタート] メニューから [シャットダウン] を使用します。このシナリオではインスタンスの停止または終了のいずれにも設定できます。

Amazon EC2 インスタンスを停止するには

1. EclipseAWSエクスプローラ、Amazon EC2ノードで、[] のコンテキスト (右クリック) メニューを開きます。インスタンス[] を選択してから、表示。[Instances (インスタンス)] リストで、停止するインスタンスを右クリックして、コンテキストメニューから [Stop (停止)] を選択します。[Yes (はい)] をクリックしてインスタンスの停止を確定します。

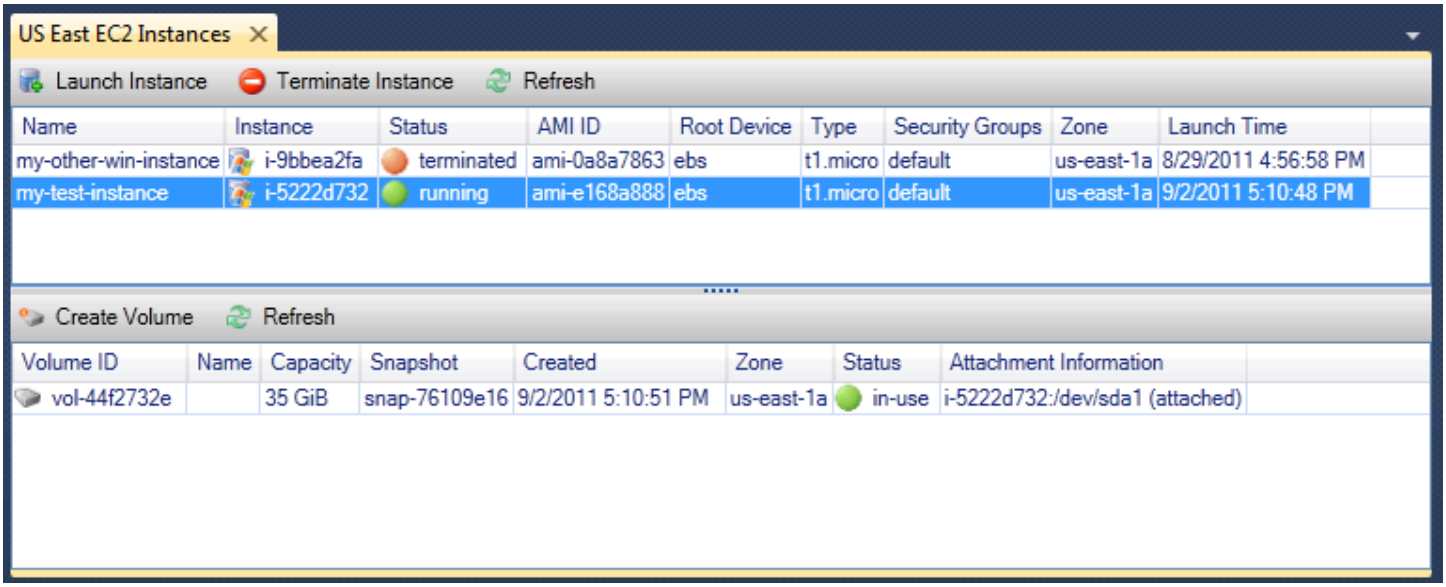


2. [Instances (インスタンス)] リストの上部で、Amazon EC2 インスタンスのステータスの変更を確認するには、[Refresh (更新)] を選択します。インスタンスを終了ではなく停止したため、インスタンスと関連付けられた EBS ボリュームはアクティブのままです。



終了したインスタンスを引き続き表示

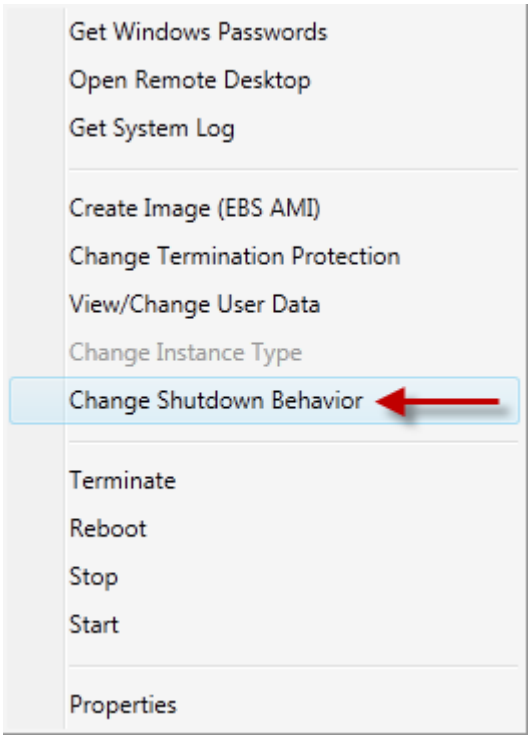
インスタンスを終了した場合、実行中や停止したインスタンスとともに [Instance (インスタンス)] リストに引き続き表示されます。最終的に、AWSこれらのインスタンスを再利用すると、リストから消去されます。終了状態のインスタンスに対して課金されることはありません。



シャットダウン時の EC2 インスタンスの動作を指定するには

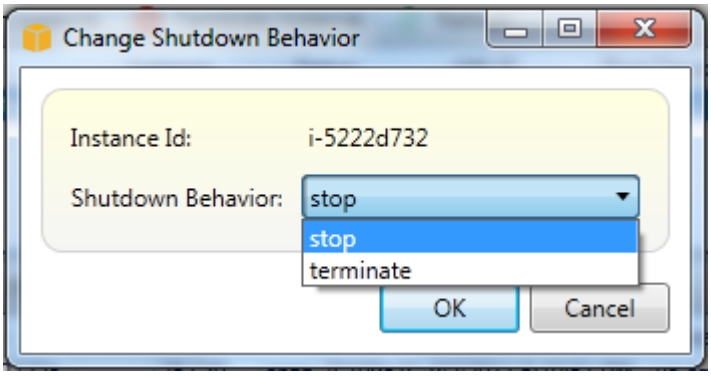
-AWSToolkit では、次の場合に Amazon EC2 インスタンスを停止するか終了するかどうかを指定できます。シャットダウンから選択される。を起動メニュー。

1. [Instances (インスタンス)] リストで、Amazon EC2 インスタンスを右クリックし、[Change shutdown behavior (シャットダウン動作の変更)] を選択します。



[Change Shutdown Behavior (シャットダウン動作の変更)] メニューアイテム

2. [Change Shutdown Behavior (シャットダウン動作の変更)] ダイアログボックスで、[Shutdown Behavior (シャットダウン動作)] ドロップダウンリストから [Stop (停止)] または [Terminate (終了)] を選択します。



Amazon ECS インスタンスの管理

AWSExplorer には、Amazon Elastic Container Service (Amazon ECS) クラスターとコンテナリポジトリの詳細なビューが用意されています。Visual Studio 開発環境内からクラスターとコンテナの詳細を作成、削除、管理できます。

サービスのプロパティの変更

サービスの詳細、イベント、プロパティはクラスタービューで表示できます。

1. EclipseAWSExplorer で管理するクラスターのコンテキスト (右クリック) メニューを開いて、[] を選択します。表示。
2. [ECS Cluster (ECS クラスター)] ビューで、左側の [Services (サービス)] をクリックしてから、詳細ビューの [Details (詳細)] タブをクリックします。[Events (イベント)] をクリックしてイベントメッセージを表示できます。また、[Deployments (デプロイ)] をクリックしてデプロイステータスを表示できます。
3. [Edit (編集)] をクリックします。目的のタスク数と、最小および最大ヘルス率を変更できます。
4. 変更を受け入れるには、[Save (保存)] をクリックします。既存の値に戻すには、[Cancel (キャンセル)] をクリックします。

タスクの停止

クラスタービューで、タスクの現在のステータスを表示し、1 つ以上のタスクを停止できます。

タスクを停止するには

1. EclipseAWSExplorer で、停止するタスクがあるクラスターのコンテキスト (右クリック) メニューを開いて、[表示。
2. [ECS Cluster (ECS クラスター)] ビューで、左側の [Tasks (タスク)] をクリックします。
3. [Desired Task Status (必要なタスクのステータス)] が `Running` に設定されていることを確認します。停止する個々のタスクを選択し、[Stop (停止)] をクリックします。または、[Stop All (すべてを停止)] をクリックし、実行中のすべてのタスクを選択して停止します。
4. [Stop Tasks (タスクの停止)] ダイアログボックスで、[Yes (はい)] を選択します。

サービスの削除

クラスタービューで、クラスターからサービスを削除できます。

クラスターサービスを削除するには

1. EclipseAWSExplorer で、削除するサービスがあるクラスターのコンテキスト (右クリック) メニューを開いて、[] を選択します。表示。
2. [ECS Cluster (ECS クラスター)] ビューで、左側の [Services (サービス)] をクリックしてから、[Delete (削除)] をクリックします。
3. [Delete Cluster (クラスターの削除)] ダイアログボックスで、クラスターにロードバランサーとターゲットグループがある場合にクラスターからそれらを削除するかどうかを選択できます。それらのバランサーとグループはサービスが削除されると使用されなくなります。
4. [Delete Cluster (クラスターの削除)] ダイアログボックスで、[OK] を選択します。クラスターは削除されると、AWSExplorer。

クラスターの削除

[] から Amazon Elastic Container Service クラスターを削除できます。AWSExplorer。

クラスターを削除するには

1. EclipseAWSExplorer で、削除するクラスターのコンテキスト (右クリック) メニューを開きます。クラスターのノードAmazon ECS[] を選択してから、削除。
2. [Delete Cluster (クラスターの削除)] ダイアログボックスで、[OK] を選択します。クラスターは削除されると、AWSExplorer。

リポジトリの作成

Amazon Elastic コンテナレジストリリポジトリは、AWSExplorer。

リポジトリを作成するには

1. EclipseAWSExplorer で、[] のコンテキスト (右クリック) メニューを開きます。リポジトリノードの下ではありますAmazon ECS[] を選択してから、リポジトリの作成。
2. [Create Repository (リポジトリの作成)] ダイアログボックスでリポジトリ名を指定し、[OK] を選択します。

リポジトリの削除

[] から Amazon Elastic Container レジストリリポジトリを削除できます。AWSExplorer。

リポジトリを削除するには

1. EclipseAWSExplorer で、[] のコンテキスト (右クリック) メニューを開きます。リポジトリノードの下ではありますAmazon ECS[] を選択してから、リポジトリの削除。
2. [Delete Repository (リポジトリの削除)] ダイアログボックスで、リポジトリを削除できます。この場所では、リポジトリはイメージを含んでいても削除されます。それ以外の場所では、リポジトリは空の場合にのみ削除されます。[Yes (はい)] をクリックします。

AWS Explorer からのセキュリティグループの管理

Toolkit for Visual Studio を使用すると、Amazon Elastic Compute Cloud (Amazon EC2) インスタンスで使用するセキュリティグループを作成および設定できます。AWS CloudFormation。Amazon EC2 インスタンスを起動する場合、またはアプリケーションをにデプロイする場合にはAWS CloudFormationでは、Amazon EC2 インスタンスに関連付けるセキュリティグループを指定します。(へのデプロイAWS CloudFormationAmazon EC2 インスタンスを作成します。)

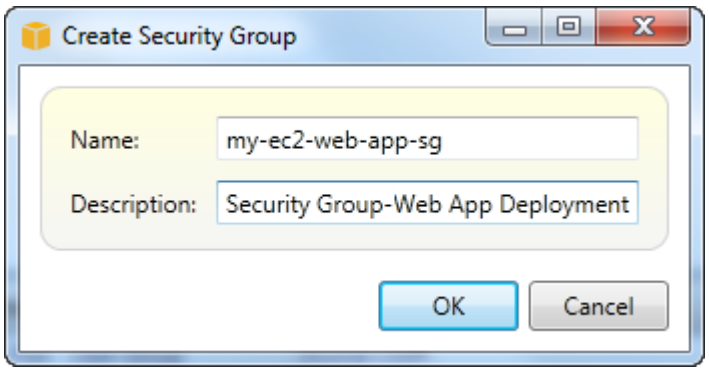
セキュリティグループは受信ネットワークトラフィックに対するファイアウォールのような役割を果たします。セキュリティグループでは、Amazon EC2 インスタンスで許可する受信ネットワークトラフィックのタイプを指定します。また、特定の IP アドレスまたは指定したユーザーまたは他のセキュリティグループからのみの受信トラフィックを許可するように指定することもできます。

セキュリティグループを作成する

このセクションでは、セキュリティグループを作成します。セキュリティグループが作成されたときは、セキュリティグループでいずれのアクセス許可も設定されていません。アクセス許可の設定はこの後のオペレーションで扱います。

セキュリティグループを作成するには

1. EclipseAWSエクスプローラー、Amazon EC2ノードで、[] のコンテキスト (右クリック) メニューを開きます。セキュリティグループノードを選択し、表示。
2. [EC2 Security Groups (EC2 セキュリティグループ)] タブで、[Create Security Group (セキュリティグループの作成)] をクリックします。
3. [Create Security Group (セキュリティグループの作成)] ダイアログボックスで、セキュリティグループ名と説明を入力し、[OK] を選択します。

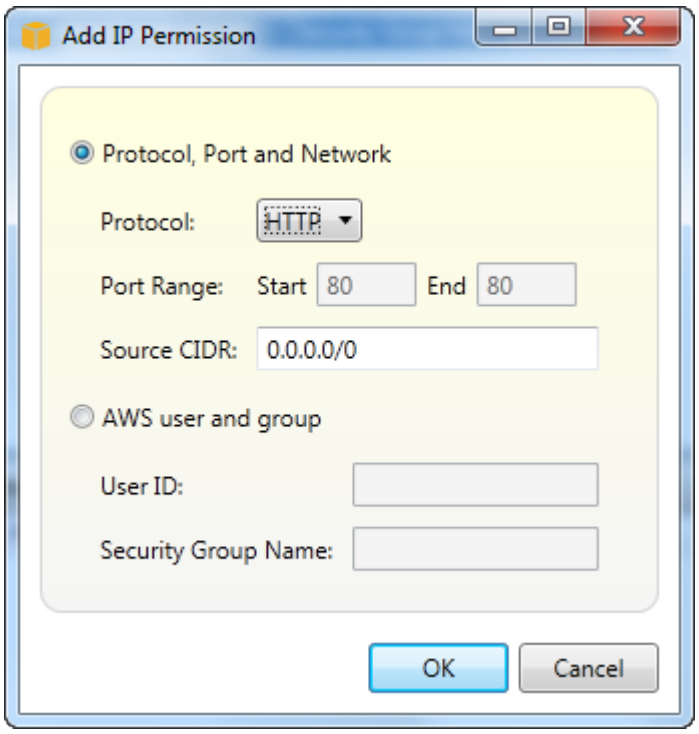


セキュリティグループにアクセス許可を追加する

このセクションでは、セキュリティグループにアクセス許可を追加して、HTTP および HTTPS プロトコルを使用したウェブトラフィックを許可します。また、他のコンピュータが Windows リモートデスクトッププロトコル (RDP) を使用して接続することを許可します。

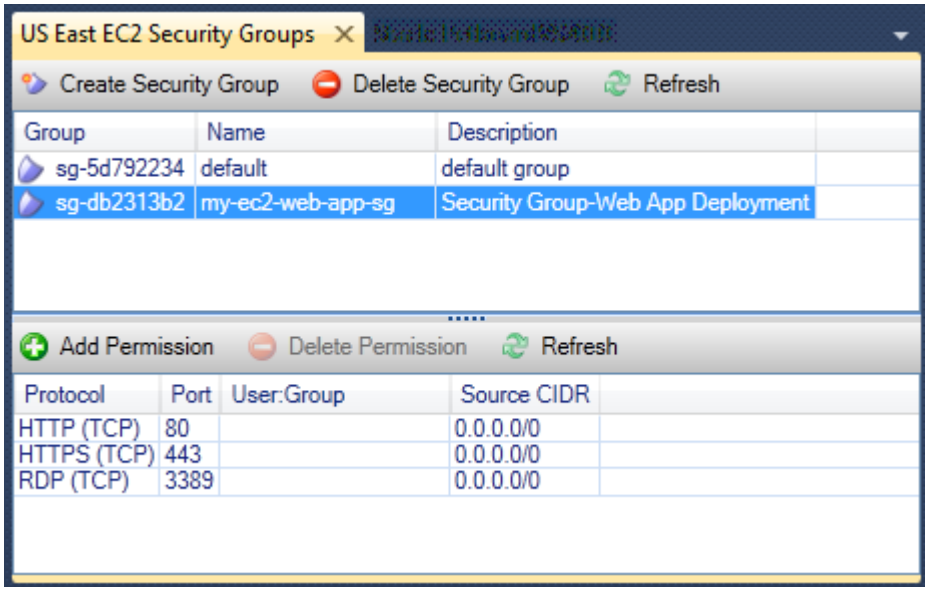
セキュリティグループにアクセス許可を追加するには

1. [EC2 Security Groups (EC2 セキュリティグループ)] タブで、セキュリティグループを選択し、[Add Permission (アクセス許可の追加)] ボタンを選択します。
2. [Add IP Permission (アクセス許可の追加)] ダイアログボックスで、[Protocol, Port and Network (プロトコル、ポート、ネットワーク)] ラジオボタンを選択し、[Protocol (プロトコル)] ドロップダウンリストで、[HTTP] を選択します。ポート範囲は HTTP のデフォルトポートであるポート 80 に自動的に調整されます。[Source CIDR (ソース CIDR)] フィールドのデフォルトは 0.0.0.0/0 となります。これは、任意の外部 IP アドレスからの HTTP ネットワークトラフィックが許可されることを指定します。[OK] をクリックします。



このセキュリティグループ用にポート 80 (HTTP) を開く

3. このプロセスを HTTPS および RDP に繰り返します。セキュリティグループのアクセス許可は以下のようになります。



また、ユーザー ID とセキュリティグループ名を指定することで、セキュリティグループにアクセス許可を設定することもできます。この場合、このセキュリティグループの Amazon EC2 インスタンスは、指定したセキュリティグループの Amazon EC2 インスタンスからのすべての受信ネッ

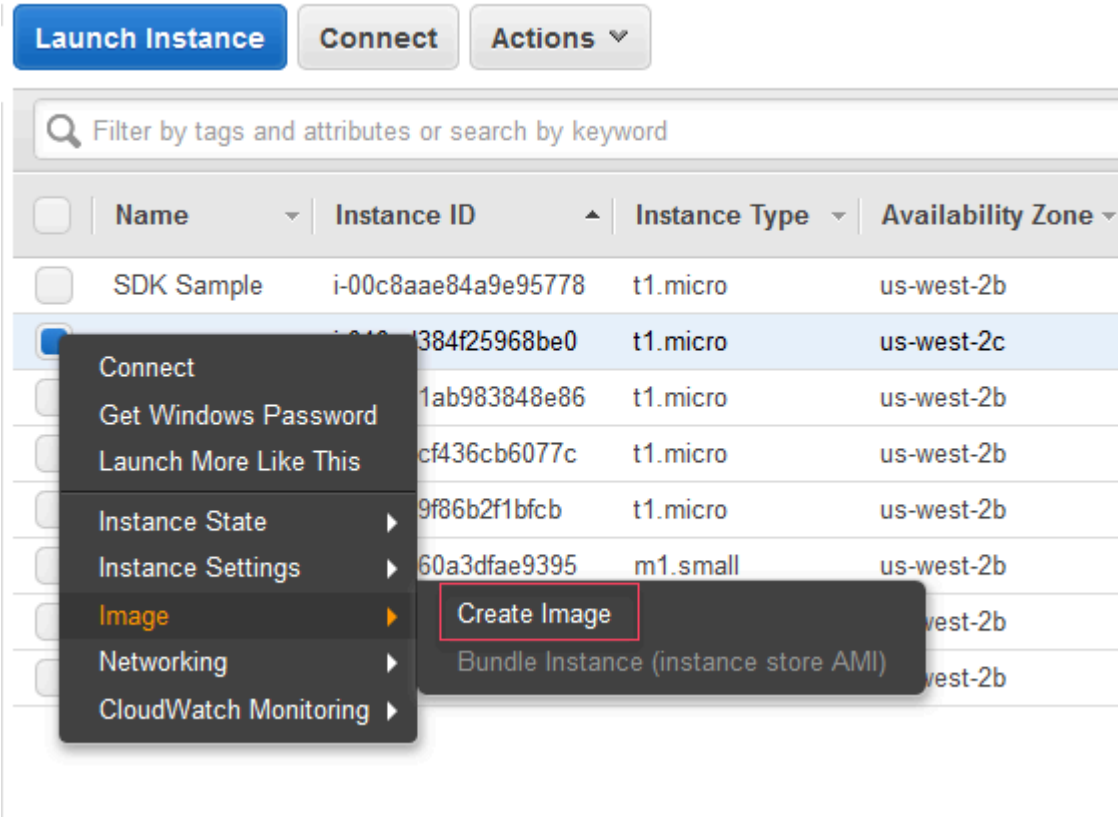
トワークトラフィックを許可します。また、セキュリティグループ名を特定するためにユーザー ID も指定しなければなりません。セキュリティグループ名はすべての間で一意であるとは限りません。AWS。セキュリティグループの詳細については、EC2 のドキュメントを参照してください。

Amazon EC2 インスタンスからの AMI の作成

[Amazon EC2 Instances (Amazon EC2 インスタンス)] ビューで、実行中または停止したインスタンスのいずれかを使用して、Amazon Machine Image (AMI) を作成することができます。AMI の詳細については、Windows インスタンス用 Amazon Elastic Compute Cloud ユーザーガイドの「Amazon マシンイメージ (AMI)」トピックを参照してください。

インスタンスから AMI を作成するには

1. AMI の基本として使用するインスタンスを右クリックして、コンテキストメニューから [Create Image (イメージの作成)] を選択します。

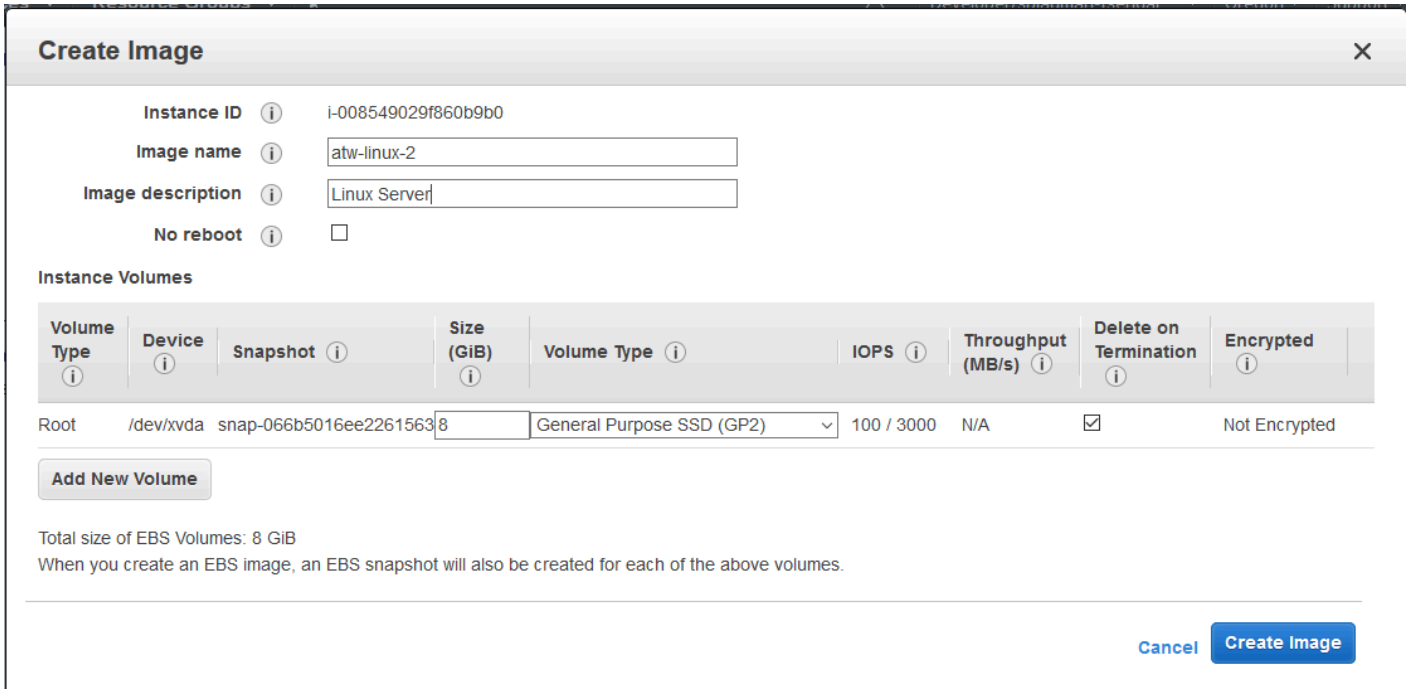


[Create Image (イメージの作成)] のコンテキストメニュー

2. [Create Image (イメージの作成)] ダイアログボックスで、一意の名前と説明を入力して、[Create Image (イメージの作成)] を選択します。デフォルトでは、Amazon EC2 はインスタンスをシャットダウンし、アタッチされていたすべてのボリュームのスナップショットを作成し、AMI を作

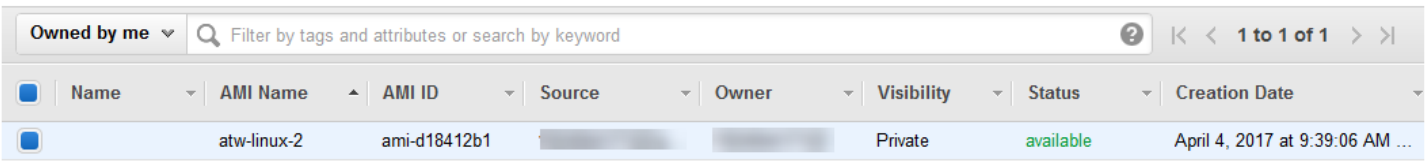
成して登録し、インスタンスを再起動します。インスタンスをシャットダウンしない場合、[No reboot] (再起動しない) を選択します。

> **Warning**
>
> [再起動しない] を選択した場合は、作成されたイメージのファイルシステムの整合性は保証されません。



[Create Image (イメージの作成)] ダイアログボックス

AMI が作成されるまでには数分かかる場合があります。作成されると、AWS Explorer の [AMIs] ビューに表示されます。このビューを表示するには、AWS Explorer の [Amazon EC2 | AMIs] ノードをダブルクリックします。AMI を表示するには、[Viewing (表示中)] ドロップダウンリストから [Owned By Me (自己所有)] を選択します。AMI を表示するには、[Refresh (更新)] を選択する必要がある場合があります。最初に AMI が表示されたときは、保留状態であることがありますが、しばらくすると、利用可能な状態になります。



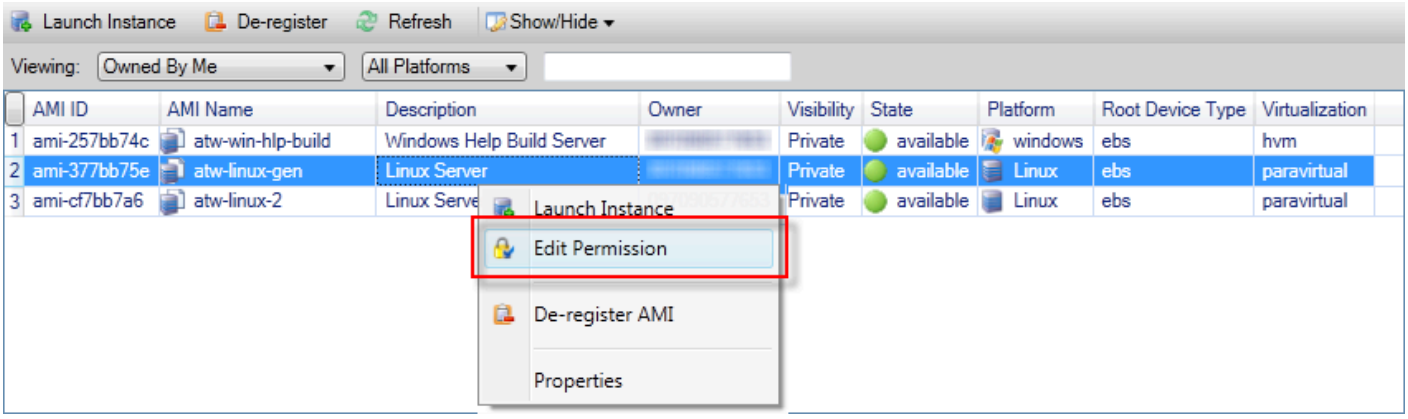
作成された AMI のリスト

Amazon マシンイメージに起動許可を設定する

Amazon マシンイメージ (AMI) に起動許可を設定するには、AMIの表示AWSExplorer。[Set AMI Permissions (AMI アクセス許可の設定)] ダイアログボックスを使用して、AMI から許可をコピーします。

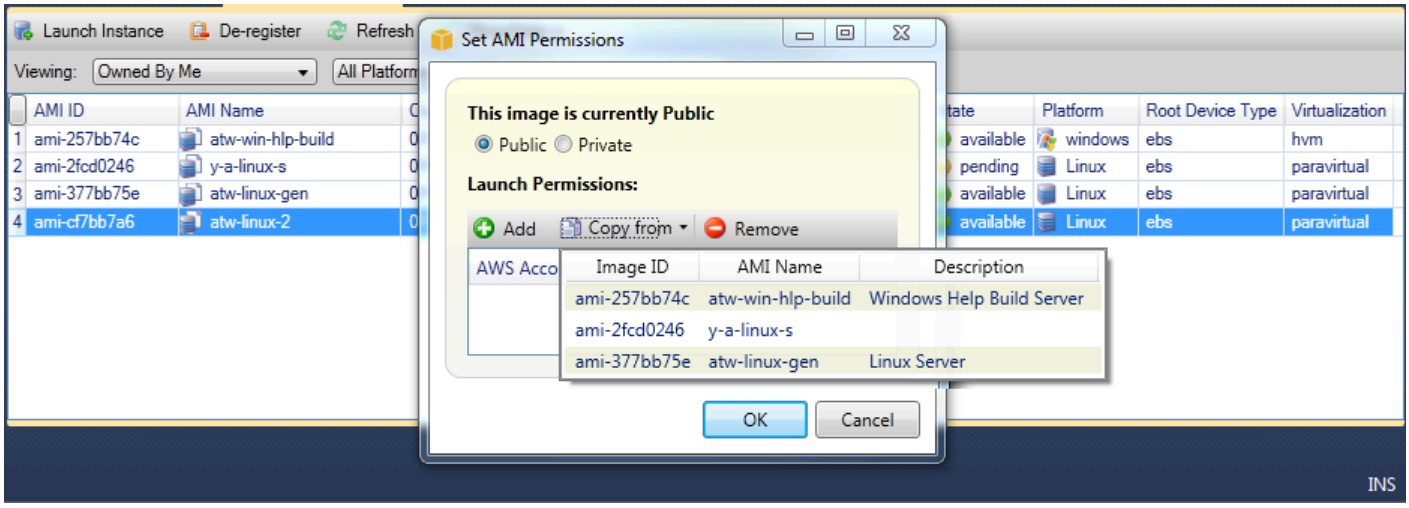
AMI に許可を設定するには

1. 左AMIの表示AWSExplorer で、AMI のコンテキスト (右クリック) メニューを開き、[アクセス許可の編集。



2. [Set AMI Permissions (AMI アクセス許可の設定)] ダイアログボックスに、選択できる 3 つのオプションがあります。
   - 起動権限を付与するには、を追加します。をクリックし、のアカウント番号を入力します。AWS起動権限を与えているユーザー。
   - 起動許可を削除するには、AWS起動許可を削除する [ユーザー] を選択し、を削除します。。
   - 1 つの AMI から別の AMI に許可をコピーするには、リストから AMI を選択し、[Copy from (コピー)] を選択します。AMI の起動許可を持っているユーザーを選択すると、現在の AMI の起動許可が与えられます。このプロセスを [Copy-from (コピー)] リストの他の AMI で繰り返すことができ、これにより複数の AMI からターゲットの AMI へ許可をコピーできます。

     -からの COPYリストには、AMIビューはから表示されましたAWSExplorer。その結果、アクティブアカウントによって所有されている他の AMI がない場合には、[Copy-from (コピー)] リストに AMI が表示されない場合があります。



[Copy AMI permissions (AMI アクセス許可のコピー)] ダイアログボックス

Amazon Virtual Private Cloud (VPC)

Amazon Virtual Private Cloud (Amazon VPC) を使用すると、定義した仮想ネットワーク内でAmazon Web Services リソースを起動できます。仮想ネットワークは、お客様自身のデータセンターで運用されていた従来のネットワークに似ていますが、のスケーラブルなインフラストラクチャを使用できるというメリットがあります。AWS。詳細については、Amazon VPC ユーザーガイドをご覧ください。

Toolkit for Visual Studio により、開発者は VPC 機能にアクセスできます。AWS Management Consoleただし、Visual Studio の開発環境から。-Amazon VPCのノードAWSExplorer には、次の領域のサブノードが含まれています。

- VPC
- Subnetshttps://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Subnets.html
- Elastic IP
- インターネットゲートウェイ
- ネットワーク ACL
- ルートテーブル
- セキュリティグループ

でのデプロイのためのパブリック/プライベート VPC の作成AWS Elastic Beanstalk

このセクションでは、パブリックサブネットとプライベートサブネットの両方が含まれる Amazon VPC を作成する方法を説明します。パブリックサブネットは、ネットワークアドレス変換 (NAT) を実行する Amazon EC2 インスタンスを使用して、プライベートサブネットのインスタンスがパブリックインターネットと通信できるようにします。2 つのサブネットは同じアベイラビリティーゾーン (AZ) に存在している必要があります。

これは、VPC に AWS Elastic Beanstalk 環境をデプロイするために必要な最小限の VPC 設定です。このシナリオでは、アプリケーションをホストする Amazon EC2 インスタンスがプライベートサブネット内に存在し、受信トラフィックをアプリケーションにルーティングする Elastic Load Balancing ロードバランサーはパブリックサブネット内にあります。

ネットワークアドレス変換 (NAT) の詳細については、「」を参照してください。NAT インスタンスのAmazon Virtual Private Cloud ユーザーガイド。VPC を使用するためにデプロイを構成する方法の例については、「Elastic Beanstalk へのデプロイ」を参照してください。

パブリック/プライベートサブネット VPC を作成するには

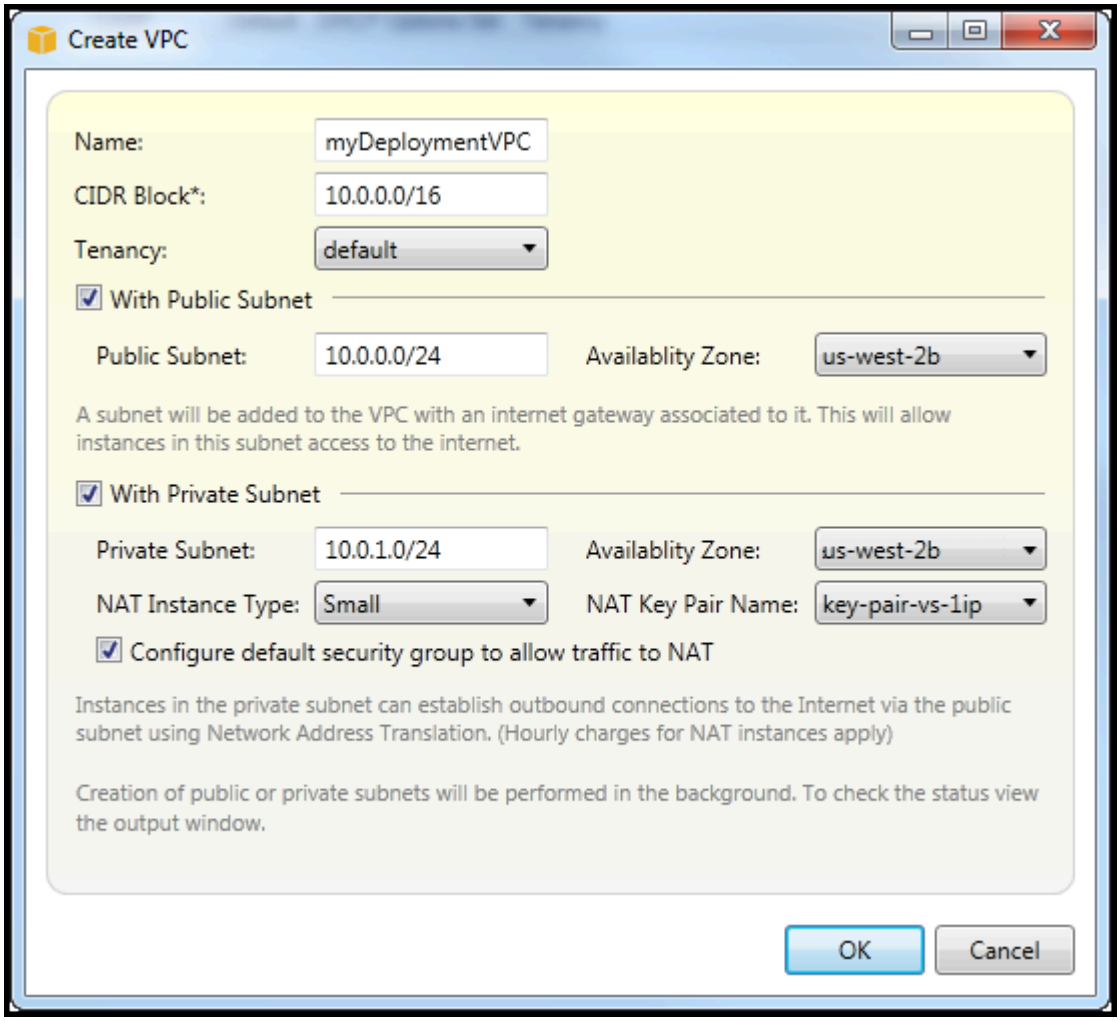
1. 左Amazon VPCノード内AWSエクスプローラ、VPCサブノードを選択し、VPC の作成。

2. VPC を次のように設定します。
   - 使用する VPC の名前を入力します。
   - [With Public Subnet (パブリックサブネットを使用)] と [With Private Subnet (プライベートサブネットの使用)] チェックボックスをオンにします。

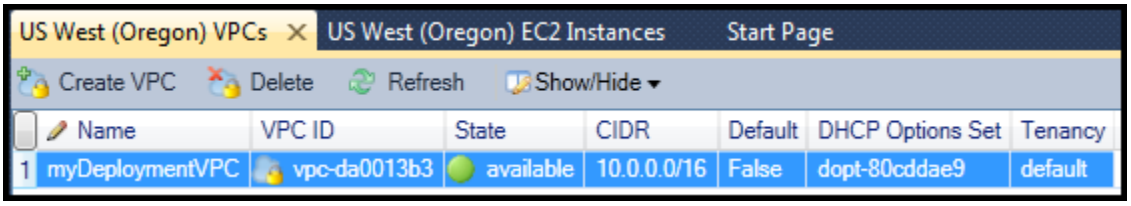
- 各サブネットの [Availability Zone (アベイラビリティーゾーン)] ドロップダウンリストから、アベイラビリティーゾーンを選択します。両方のサブネットに必ず同じ AZ を使用してください。
- プライベートサブネットでは、[NAT Key Pair Name (NAT キーペア名)] にキーペアを入力します。このkey pair はAmazon EC2 インスタンス用で、このインスタンスがプライベートサブネットからパブリックインターネットへのネットワークアドレス変換を実行します。
- [Configure default security group to allow traffic to NAT (NAT へのトラフィックを許可するようにデフォルトセキュリティグループを設定する)] チェックボックスをオンにします。

使用する VPC の名前を入力します。[With Public Subnet (パブリックサブネットを使用)] と [With Private Subnet (プライベートサブネットの使用)] チェックボックスをオンにします。各サブネットの [Availability Zone (アベイラビリティーゾーン)] ドロップダウンリストから、アベイラビリティーゾーンを選択します。両方のサブネットに必ず同じ AZ を使用してください。プライベートサブネットでは、[NAT Key Pair Name (NAT キーペア名)] にキーペアを入力します。このkey pair はAmazon EC2 インスタンス用で、このインスタンスがプライベートサブネットからパブリックインターネットへのネットワークアドレス変換を実行します。[Configure default security group to allow traffic to NAT (NAT へのトラフィックを許可するようにデフォルトセキュリティグループを設定する)] チェックボックスをオンにします。

[OK] をクリックします。

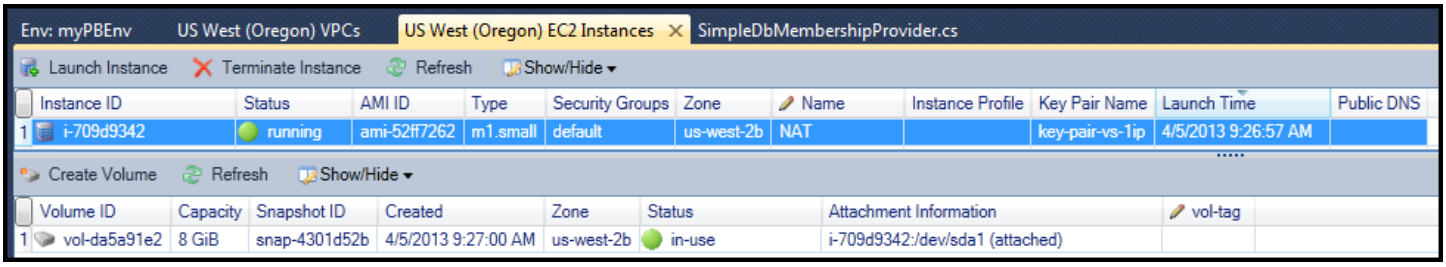


新しい VPC は、VPCタブのタブAWSExplorer。



NAT インスタンスが起動するまでに数分かかることがあります。使用可能な場合は、Amazon EC2ノード内AWSエクスプローラーを開き、インスタンスサブノード。

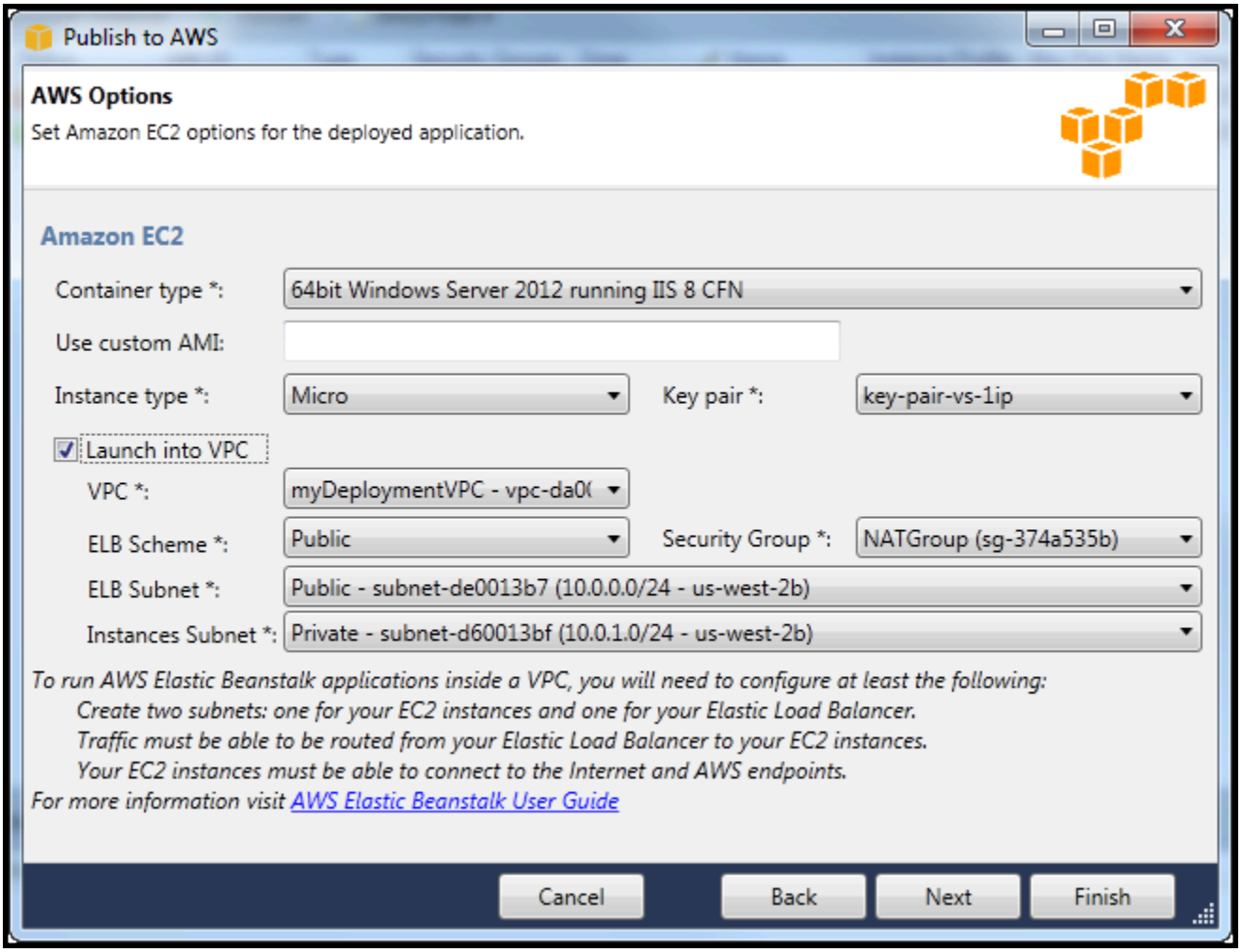
AnAWS Elastic Beanstalk(Amazon EBS) ボリュームは、NAT インスタンスに対して自動的に作成されます。Elastic Beanstalk の詳細については、を参照してください。[AWS Elastic Beanstalk(EBS)]のLinux インスタンス用 Amazon EC2 ユーザーガイド。



例えばへのアプリケーションのデプロイAWS Elastic Beanstalk環境VPC で環境を起動するように選択すると、Toolkit はへの発行Amazon Web ServicesVPC の設定情報を含むダイアログボックス。

Toolkit によりダイアログボックスに入力される情報は、Toolkit で作成された VPC からの情報だけで、を使用して作成された VPC のものは含まれません。AWS Management Console。これは、Toolkit が VPC を作成したとき、情報にアクセスできるように VPC のコンポーネントにタグ付けしているからです。

次のデプロイウィザードのスクリーンショットは、Toolkit で作成した VPC の値が入ったダイアログボックスの例を示します。



VPC を削除するには

VPC を削除するには、VPC のすべての Amazon EC2 インスタンスをまず終了する必要があります。

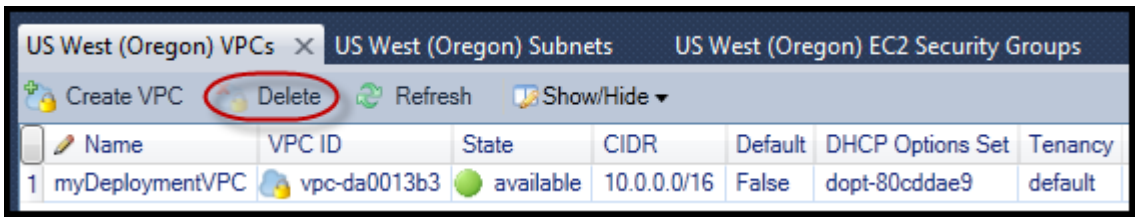
1. VPC 内の AWS Elastic Beanstalk 環境にアプリケーションをデプロイした場合は、環境を削除します。これにより、Elastic Load Balancing ロードバランサーとともにユーザーのアプリケーションをホストするすべての Amazon EC2 インスタンスが終了します。

   環境を削除しないで、アプリケーションのホスティングしているインスタンスを直接終了しようとすると、Auto Scaling サービスは、削除されたインスタンスを置き換える新しいインスタンスを自動的に作成します。詳細については、Auto Scaling 開発者ガイド を参照してください。

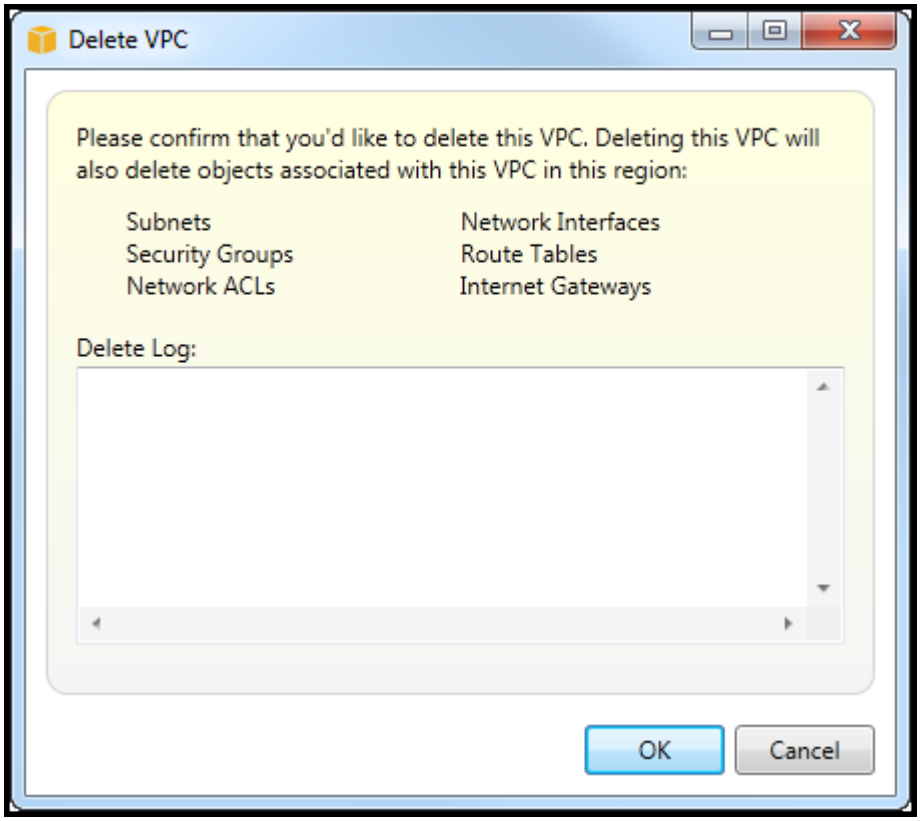
2. VPC 向け NAT インスタンスを削除します。

   VPC を削除するために、NAT インスタンスに関連付けられている Amazon EBS ボリュームを削除する必要はありません。ただし、このボリュームを削除しない場合は、NAT インスタンスおよび VPC を削除した場合でも、引き続きお客様への課金が発生することになります。

3. [VPC] タブで、[Delete (削除)] リンクを選択して VPC を削除します。



4. [Delete VPC (VPC の削除)] ダイアログボックスで、[OK] を選択します。



Visual Studio の AWS CloudFormation テンプレートエディタの使用

Toolkit for Visual Studio には、Visual Studio 用の AWS CloudFormation テンプレートエディタと AWS CloudFormation テンプレートプロジェクトが含まれています。以下の機能がサポートされています。

- 指定されたテンプレートプロジェクトタイプを使用して、新しい AWS CloudFormation テンプレート (空か、既存のスタックまたはサンプルテンプレートからコピー) を作成します。
- 自動 JSON 検証、オートコンプリート、コードの折りたたみ、構文の強調表示を使った、テンプレートの編集。
- テンプレートでの、組み込み関数の自動候補およびフィールド値のリソース参照パラメータの自動候補。
- Visual Studio からテンプレートの一般的なアクションを実行するためのメニュー項目。

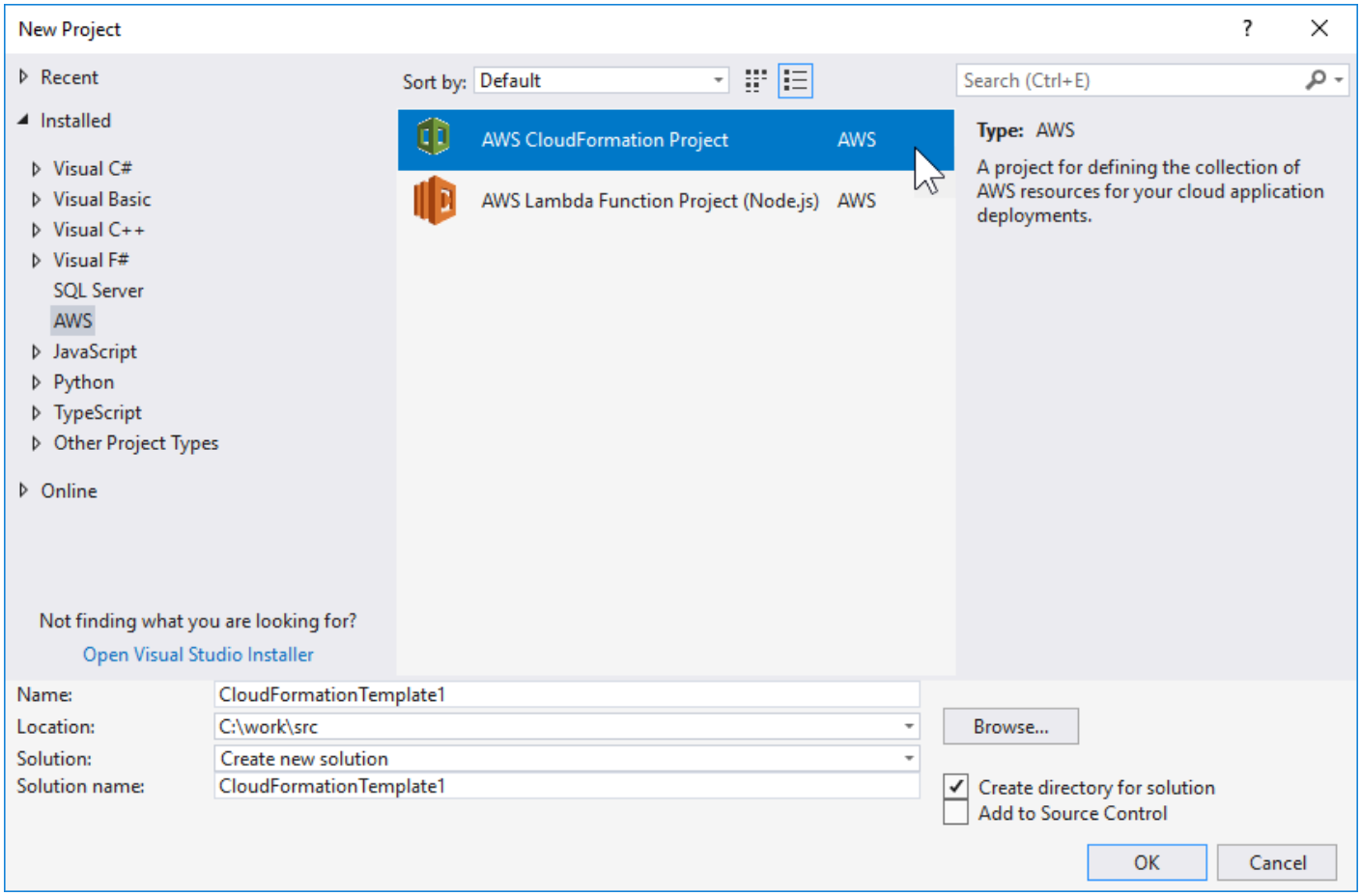
トピック

Visual Studio で AWS CloudFormation テンプレートプロジェクトを作成する

テンプレートプロジェクトを作成するには

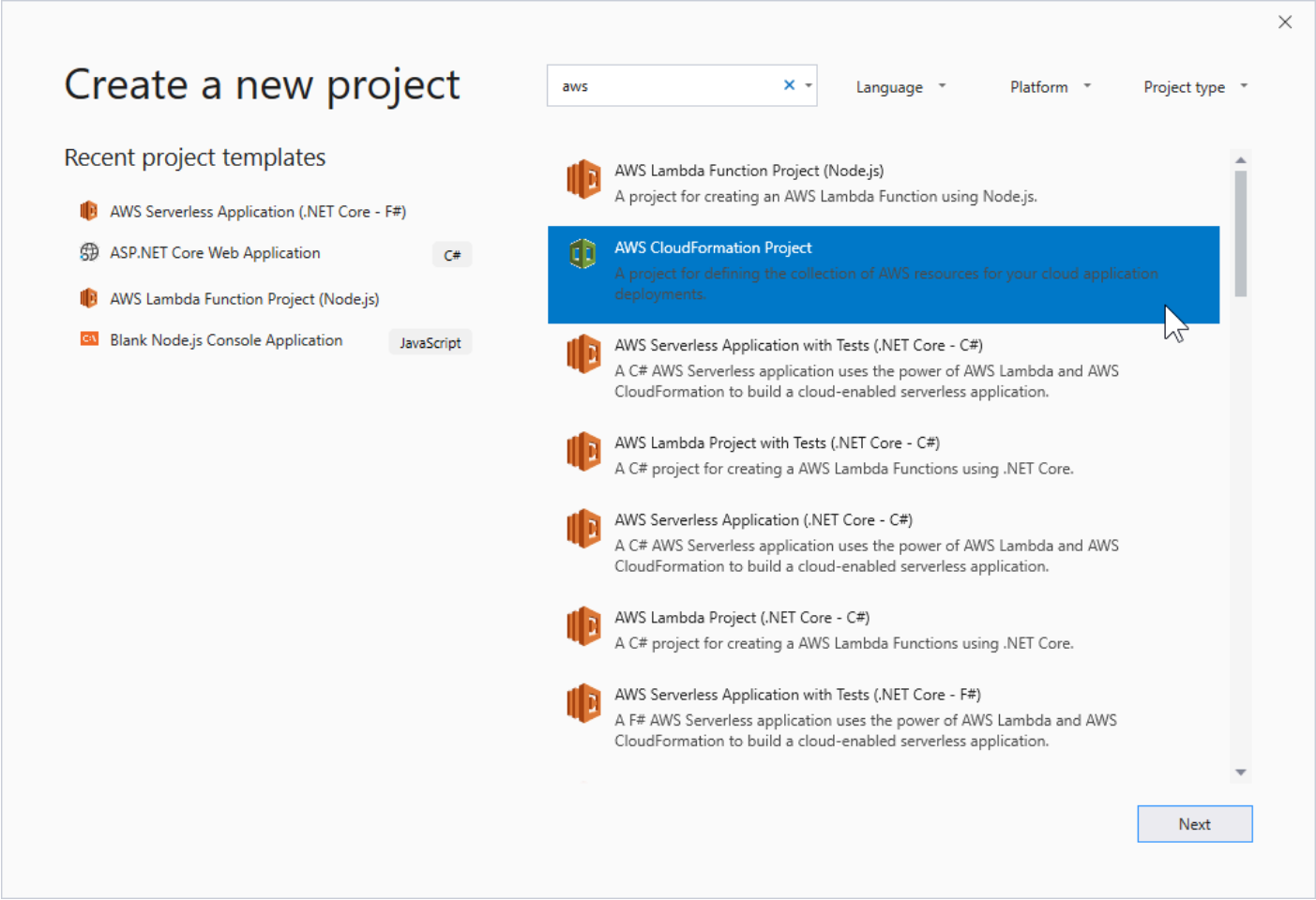
1. Visual Studio で [File (ファイル)]、[New (新規)] の順に選択し、[Project (プロジェクト)] を選択します。
2. Visual Studio 2017 の場合:

   左新しいプロジェクトダイアログボックス、展開インストール済みを選択し、AWS。



   Visual Studio 2019 の場合:

左新しいプロジェクト] ダイアログボックスで、言語,プラットフォーム, およびプロジェクトタイプドロップダウンボックスが [すべて...] に設定され、次のように入力します。awsの検索フィールド。



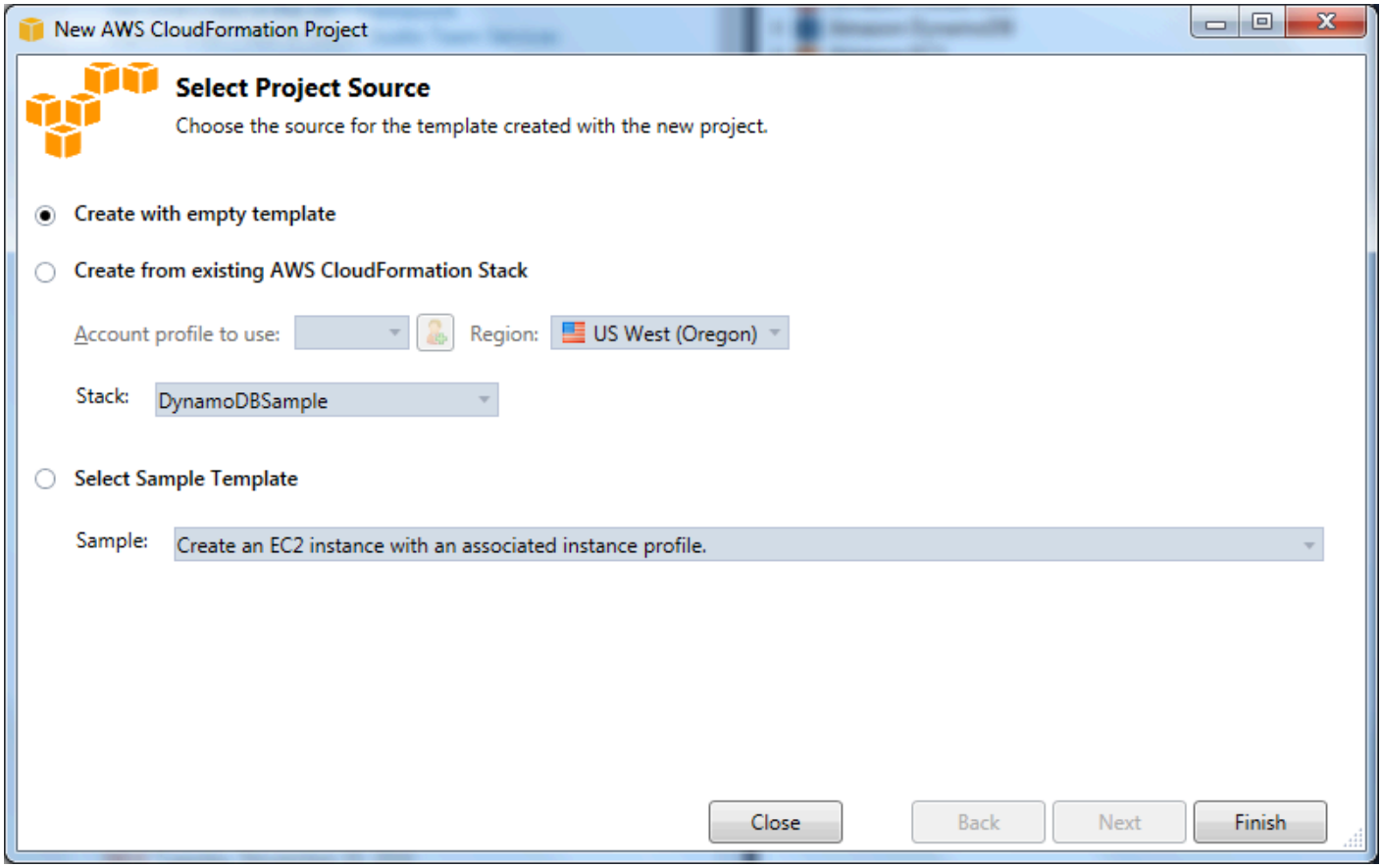
3. を選択します。AWSCloudFormation プロジェクトテンプレートテンプレート。
4. Visual Studio 2017 の場合:

   テンプレートプロジェクトの目的の [名前]、[場所] などを入力し、[OK] をクリックします。

   Visual Studio 2019 の場合:

   [Next] (次へ) をクリックします。次のダイアログで、テンプレートプロジェクトの目的の [名前]、[場所] などを入力し、[作成] をクリックします。
5. [Select Project Source (プロジェクトソースの選択)] ページで、作成するテンプレートのソースを選択します。
   - [Create with empty template (空のテンプレートを使用して作成)] では、新しい空の AWS CloudFormation テンプレートが生成されます。

- 既存から作成AWS|CFN| スタックでは、の既存のスタックからテンプレートが生成されます。AWSアカウント. (スタックのステータスは CREATE_COMPLETE である必要はありません。)
- [Select sample template (サンプルテンプレートを選択)] では、AWS CloudFormation サンプルテンプレートの 1 つからテンプレートを生成します。

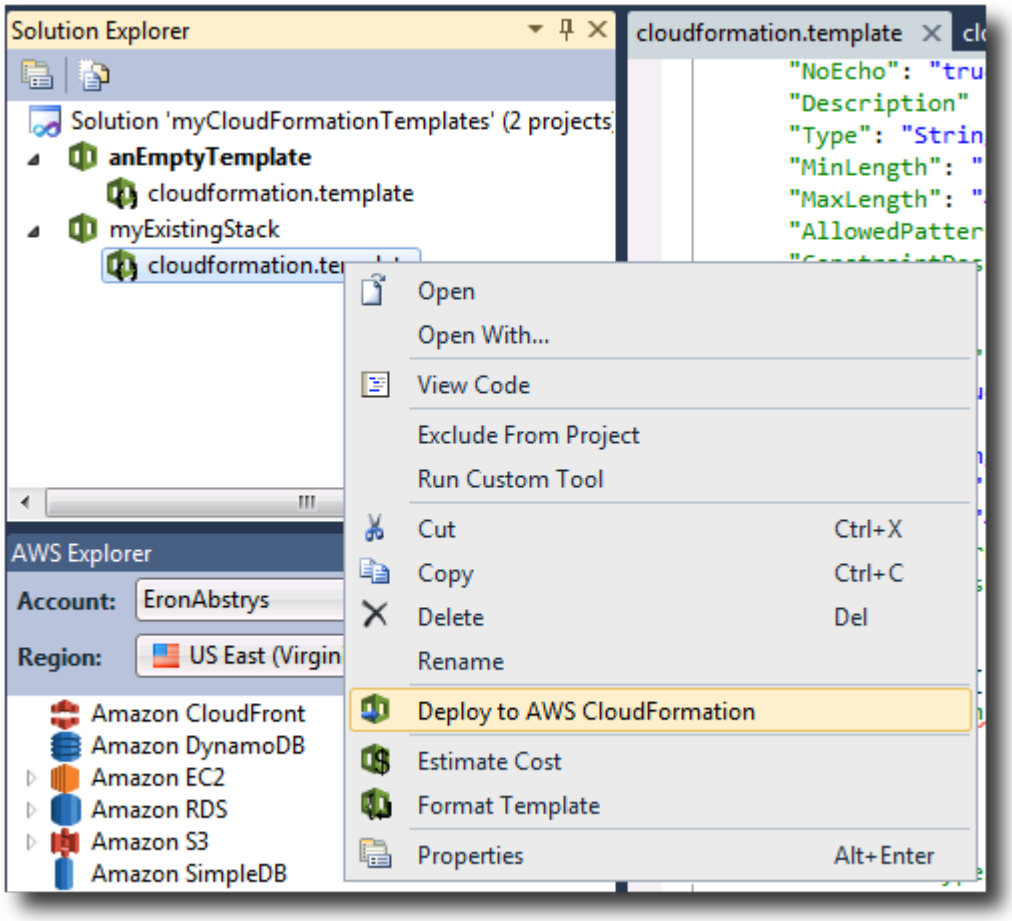


6. AWS CloudFormation テンプレートプロジェクトの作成を完了するには、[Finish (完了)] を選択します。

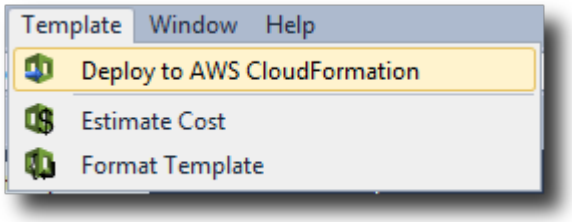
Visual Studio での AWS CloudFormation テンプレートのデプロイ

CFN テンプレートをデプロイするには

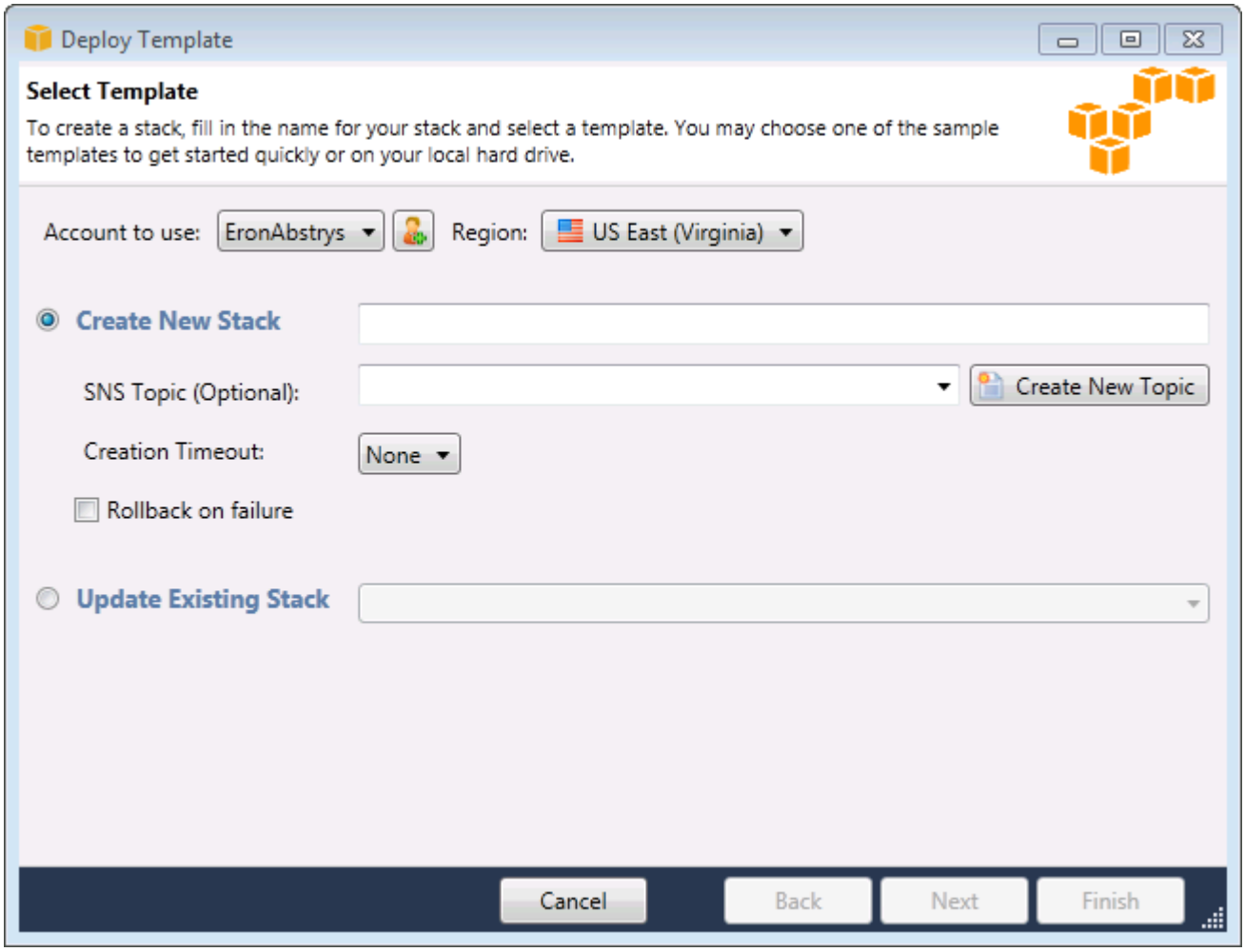
1. Solution Explorer でデプロイしたいテンプレートのコンテキスト (右クリック) メニューを開いて、へのデプロイAWS CloudFormation。



また、現在編集中のテンプレートをデプロイするには、テンプレートメニューで、[] を選択します。へのデプロイAWS CloudFormation。



2. リポジトリの []テンプレートのデプロイ[] ページで、AWS アカウントスタック起動に使用し、起動するリージョンを指定します。

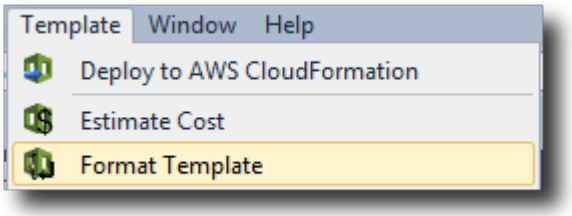


3. [Create New Stack (新しいスタックの作成)] を選択してスタックの名前を入力します。
4. 以下のオプションのいずれかを選択します (いずれも選択しない場合もあります)。
   - スタックの進行状況についての通知を受信するには、[SNS Topic (SNS トピック)] ドロップダウンリストで SNS トピックを選択します。[Create New Topic (新しいトピックの作成)] を選択し、ボックスに E メールアドレスを入力することで、SNS トピックを作成することもできます。
   - [Creation Timeout (タイムアウトの作成)] を使用して、スタックの作成を失敗と AWS CloudFormation 判断するまでの時間 (および、[Rollback on failure (失敗時のロールバック)] オプションの選択が外れていない場合はロールバックされるまでの時間) を指定できます。
   - 失敗した際にスタックをロールバック (つまり削除) する場合は、[Rollback on failure (失敗時のロールバック)] を使用します。デバッグ目的で、起動に失敗したスタックをアクティブのままにする場合は、このチェックボックスをオフのままにします。
5. [Finish (完了)] をクリックして、スタックを起動します。

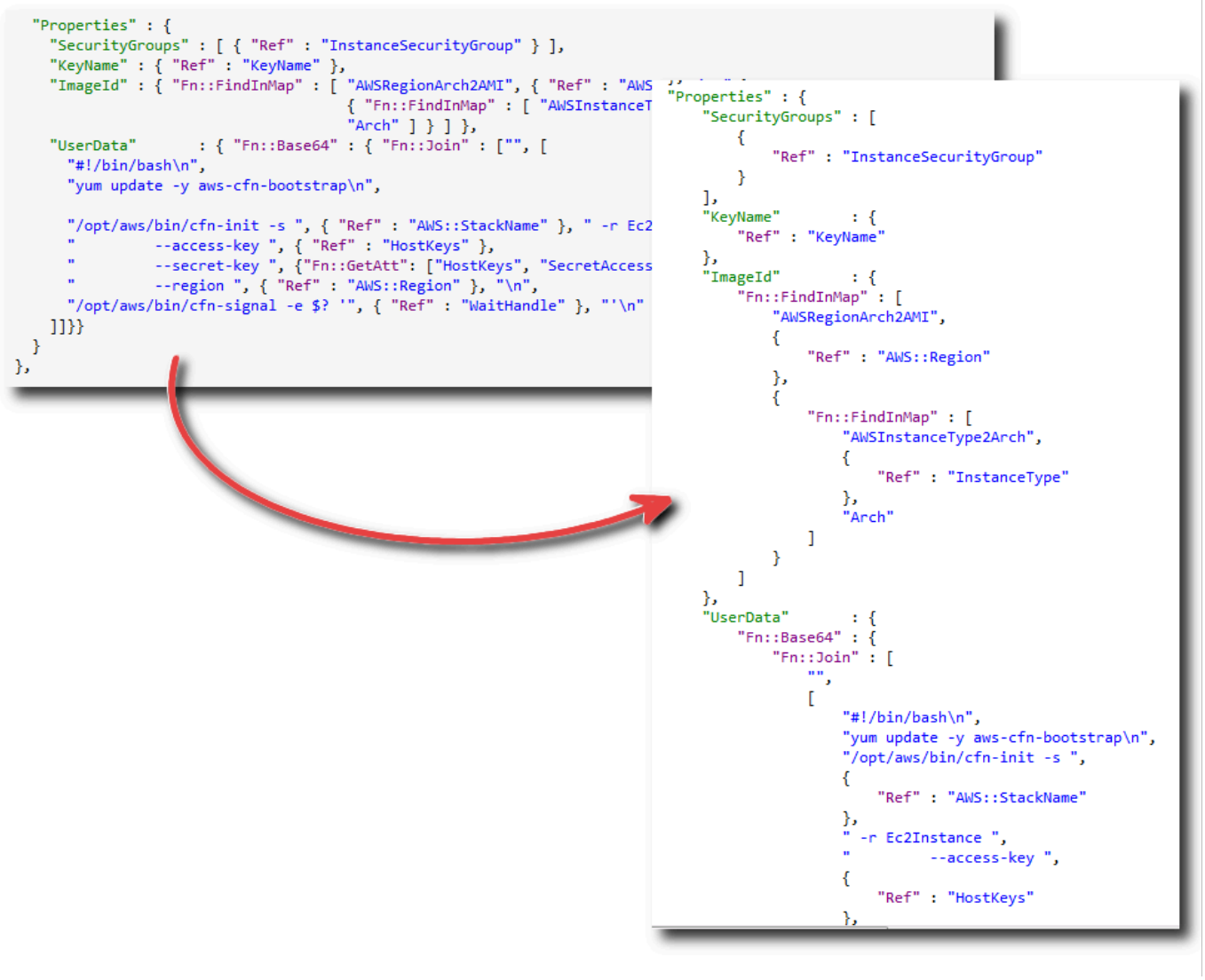
Visual Studio での AWS CloudFormation テンプレートのフォーマット

- Solution Explorer でテンプレートのコンテキスト (右クリック) メニューを開いて、[Format Template (フォーマットテンプレート)] を選択します。

  また、現在編集中のテンプレートをフォーマットするには、編集中の [Template (テンプレート)] メニューで、[Format Template (フォーマットテンプレート)] を選択します。

  ユーザーの JSON コードがフォーマットされ、構造が明確に表示されます。



Amazon S3 の使用AWSエクスプローラ

Amazon Simple Storage Service (Amazon S3) を使用すると、インターネット接続を使って、データの保存と取得を行うことができます。Amazon S3 に保存したすべてのデータは、デフォルトではアカウントに関連付けられ、自分のみがアクセスできます。Toolkit for Visual Studio を使用すると、Amazon S3 にデータを保存して、そのデータを表示、管理、取得、配布することができます。

Amazon S3 では、バケットの概念を使用しており、これはファイルシステムまたは論理ドライブに類似したものと考えることができます。バケットにはフォルダとオブジェクトを含めることができます。フォルダはディレクトリに類似しています。オブジェクトはファイルに類似しています。このセクションでは、これらの概念を使用して、Toolkit for Visual Studio によって公開されている Amazon S3 の機能を紹介します。

> **Note**
>
> このツールを使用するには、IAM ポリシーでs3:GetBucketAcl,s3:GetBucket, およびs3:ListBucketアクション。詳細については、「」を参照してください。の概要AWSIAM ポリシー。

Amazon S3 バケットを作成する

バケットは Amazon S3 で最も基本的なストレージの単位です。

S3 バケットを作成するには

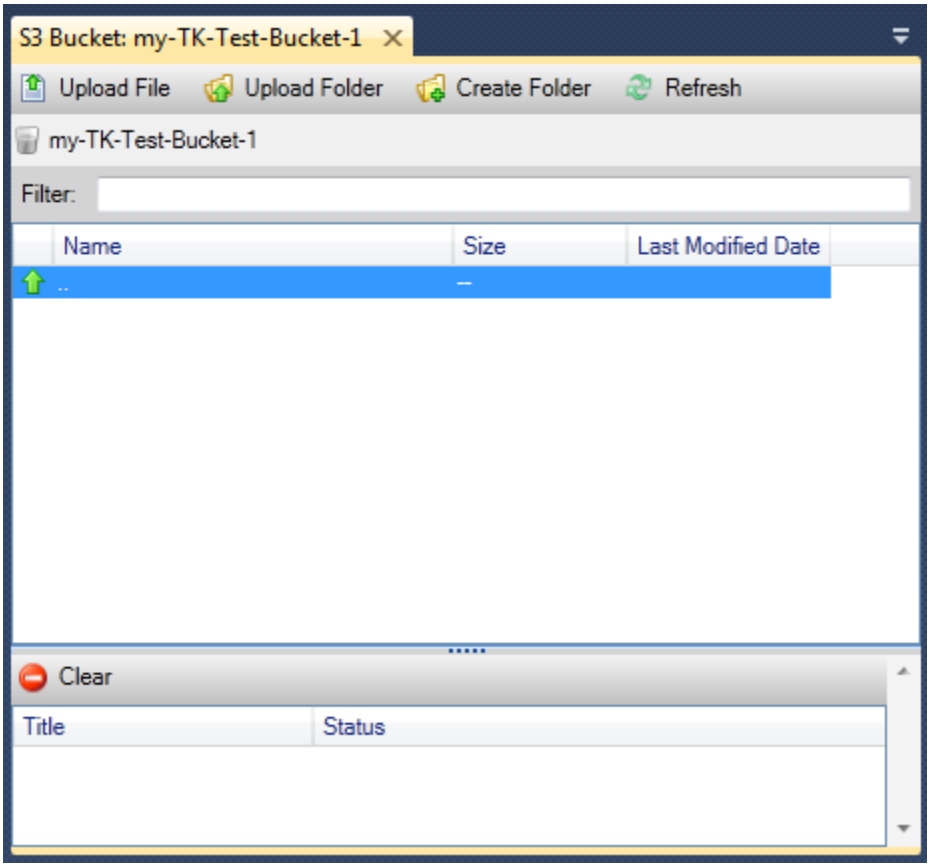
1. EclipseAWSExplorer で、のコンテキスト (右クリック) メニューを開きます。Amazon S3ノードを選択し、バケットを作成する。
2. [Create Bucket (バケットの作成)] ダイアログで、バケットの名前を入力します。バケット名は、で一意である必要があります。AWS。その他の制約については、Amazon S3 ドキュメントを参照してください。
3. [OK] をクリックします。

Amazon S3 バケットの管理AWSエクスプローラ

EclipseAWSExplorer では、Amazon S3 バケットでコンテキスト (右クリック) メニューを開くと、次の操作を実行できます。

参照

バケットに含まれるオブジェクトのビューを表示します。ここでは、フォルダを作成したり、ローカルコンピュータからファイルまたはディレクトリ全体とフォルダをアップロードすることができます。下のペインには、アップロードプロセスについてのステータスメッセージが表示されます。これらのメッセージをクリアするには、[Clear (クリア)] アイコンを選択します。バケットのこのビューには、でバケット名をダブルクリックしてアクセスすることもできます。AWSExplorer。



プロパティ

ダイアログボックスが表示されます。ここでは、次のことを実行できます。

- Amazon S3 のアクセス権限を次の範囲に設定します。
  - バケット所有者。
  - で認証されたすべてのユーザーAWS。
  - インターネットにアクセスできるすべてのユーザー。
- バケットのログ記録を有効にします。
- Amazon Simple Notification Service (Amazon SNS) を使用して通知をセットアップして、低冗長化ストレージ (RRS) を使用している場合に、データ損失が発生したときに通知されるようにします。RRS は、標準のストレージよりも耐久性とコストが低い、Amazon S3 ストレージオプションです。詳細については、「S3 のよくある質問」を参照してください。
- バケットのデータを使用して静的ウェブサイトを作成します。

ポリシー

バケットの AWS Identity and Access Management (IAM) ポリシーを設定できます。詳細については、IAM ドキュメントを参照して、IAM および S3 のユースケースを使用してください。

署名付き URL の作成

バケットのコンテンツへのアクセスを提供するために配布できる、時間制限のある URL を生成します。詳細については、「署名付き URL の作成方法」を参照してください。

マルチパートアップロードの表示

マルチパートアップロードを表示します。Amazon S3 は、大きなオブジェクトを一部に分割してアップロードして、アップロードのプロセスを効率的にすることができます。詳細については、S3 ドキュメントのマルチパートアップロードの説明を参照してください。

削除

バケットを削除します。空のバケットのみを削除できます。

ファイルとフォルダを Amazon S3 にアップロードする

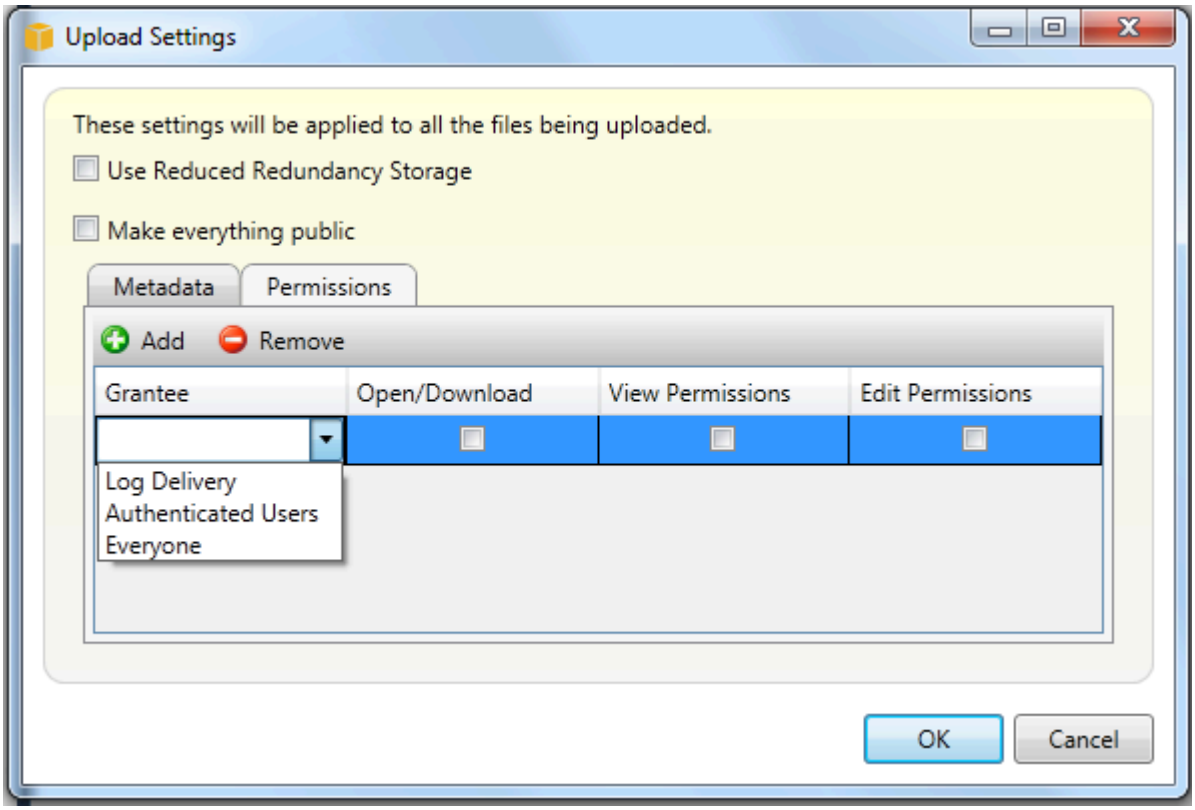
次を使用できます。AWSExplorer を使用して、ローカルコンピュータのファイルまたはフォルダ全体を、いずれかのバケットに転送します。

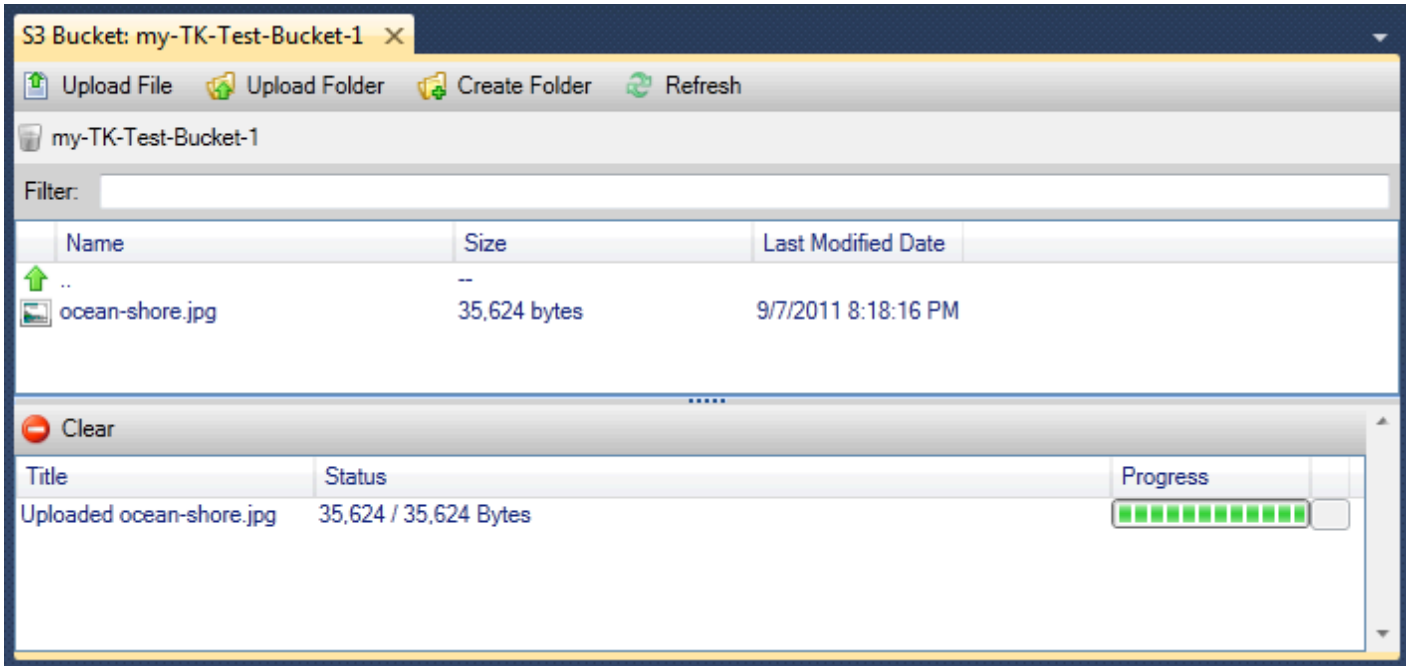
> **Note**
>
> Amazon S3 バケットにすでに存在するファイルまたはフォルダと同じ名前を持つファイルまたはフォルダをアップロードすると、アップロードされたファイルは既存のファイルを警告なしに上書きします。

S3 にファイルをアップロードするには、以下を行います。

1. EclipseAWSエクスプローラ、Amazon S3ノードを開き、バケットをダブルクリックするか、またはバケットのコンテキスト (右クリック) メニューを開き、ブラウズ。
2. バケットの [Browse (参照)] ビューで、[Upload File (ファイルのアップロード)] または [Upload Folder (フォルダのアップロード)] を選択します。
3. [File-Open] ダイアログボックスで、アップロードするファイルに移動して、それを選択し、[Open (開く)] を選択します。フォルダをアップロードする場合は、そのフォルダに移動して選択し、[Open (開く)] を選択します。

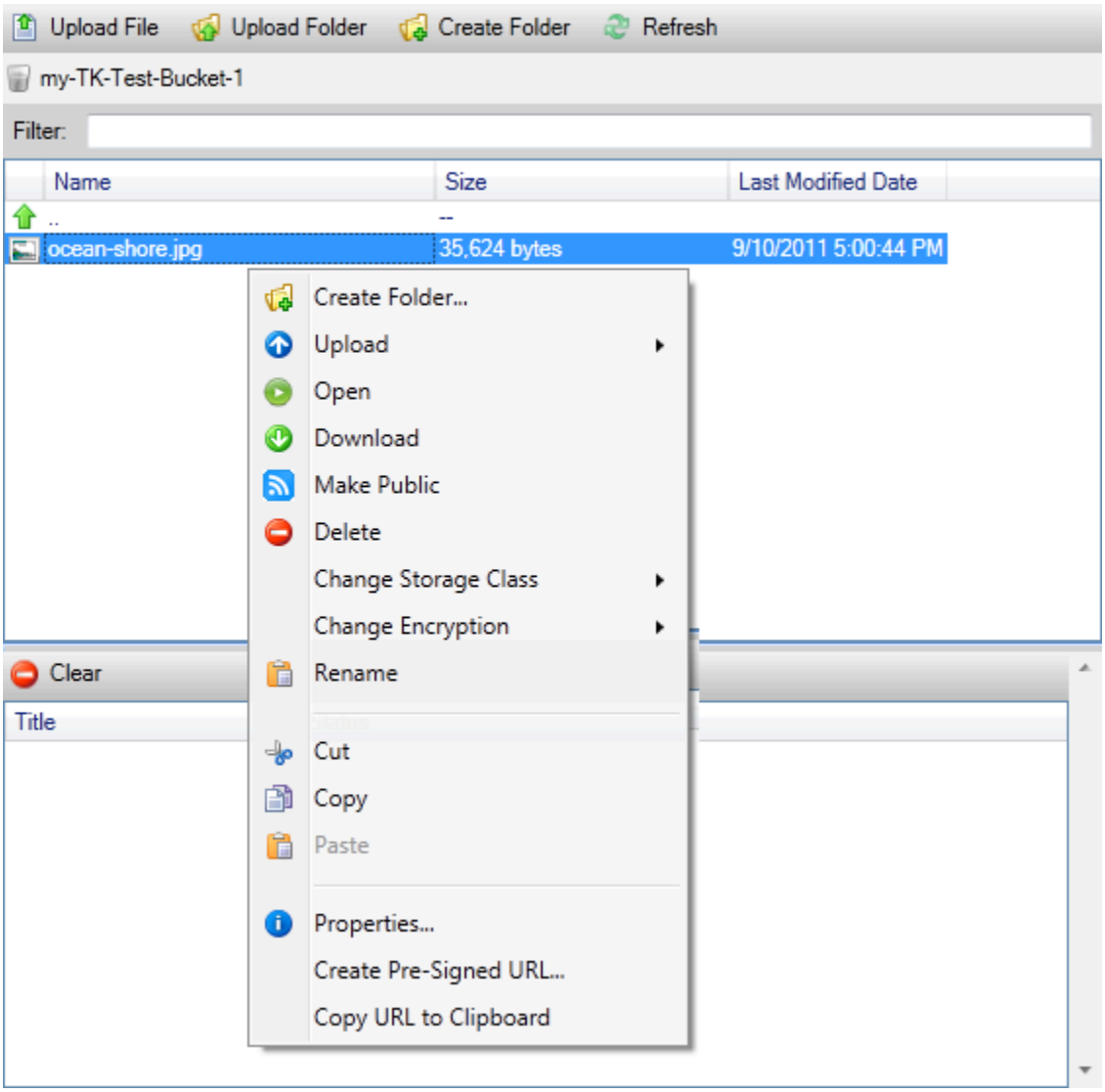
[Upload Settings (アップロード設定)] ダイアログボックスを使用すると、アップロードするファイルやフォルダに、メタデータとアクセス権限を設定できます。[Make everything public (すべてを公開する)] チェックボックスを選択すると、[Everyone (全員)] に [Open/Download (開く/ダウンロード)] のアクセス権限を設定することと同じになります。アップロードされたファイルに低冗長化ストレージを使用するオプションを選択することができます。





Amazon S3 ファイルオペレーションAWSToolkit for Visual Studio

Amazon S3 ビューでファイルを選択して、コンテキスト (右クリック) メニューを開くと、ファイルにさまざまな操作を実行できます。



フォルダを作成する

現在のバケット内にフォルダを作成することができます。([Create Folder (フォルダの作成)] のリンクを選択した場合と同じです。)

アップロード

ファイルまたはフォルダをアップロードすることができます。([Upload File (ファイルのアップロード)] または [Upload Folder (フォルダのアップロード)] のリンクを選択した場合と同じです。)

Open (開く)

選択したファイルをデフォルトのブラウザで開きます。ファイルの種類とデフォルトのブラウザの機能によっては、ファイルが表示されない場合があります。ブラウザによってダウンロードが行われる場合もあります。

ダウンロード

[Folder-Tree] ダイアログボックスが開き、選択したファイルをダウンロードできます。

公開する

選択したファイルで、のアクセス権限を次の範囲に設定します。開く/ダウンロードそして全員。([Upload Settings] のダイアログボックスで [Make everything public] のチェックボックスを選択するのと同じです。)

削除

選択されたファイルまたはフォルダを削除します。ファイルまたはフォルダを選択し、`Delete` を押して削除することもできます。

ストレージクラスの変更

ストレージクラスをスタンダードまたは低冗長化ストレージ (RRS) に設定します。現在のストレージクラスの設定を表示するには、[Properties (プロパティ)] を選択します。

暗号化の変更

ファイルのサーバー側の暗号化を設定できます。現在の暗号化の設定を表示するには、[Properties (プロパティ)] を選択します。

名前の変更

ファイルの名前を変更することができます。フォルダ名を変更することはできません。

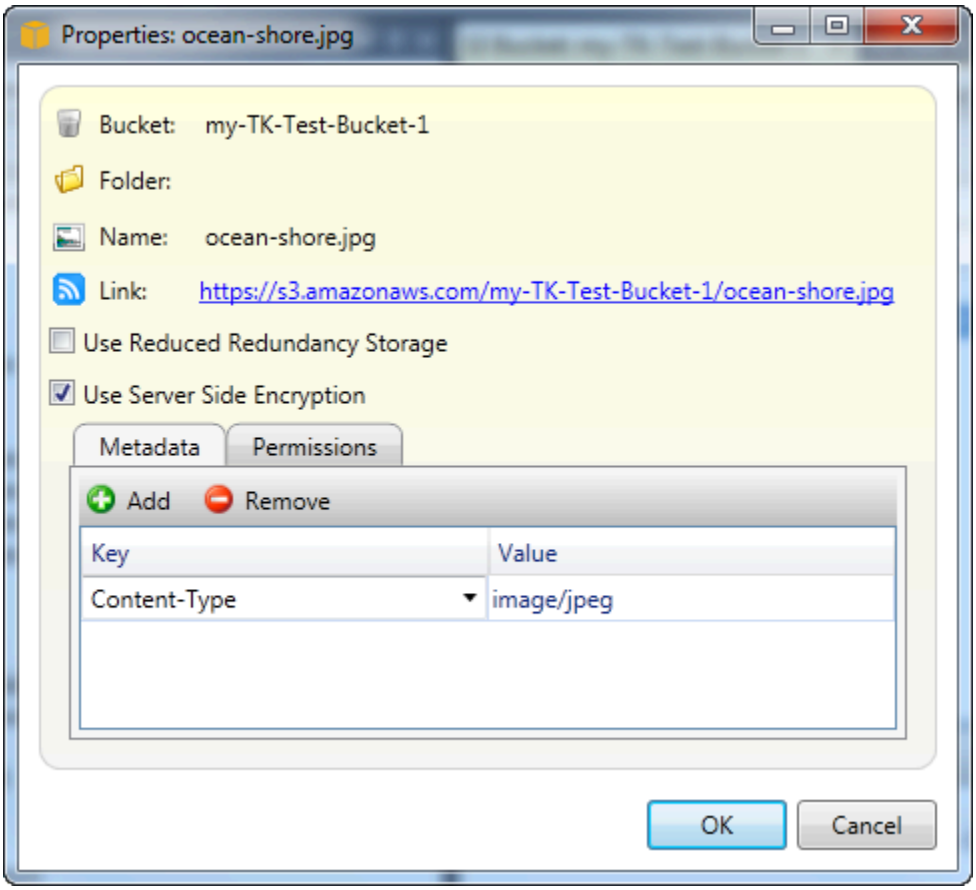
切り取り | コピー | 貼り付け

フォルダ間またはバケット間で、ファイルまたはフォルダの切り取り、コピー、貼り付けを行うことができます。

プロパティ

表示されるダイアログボックスを使って、ファイルにメタデータとアクセス権限を設定できます。ファイルのストレージを低冗長化ストレージ (RRS) またはスタンダードに切り替えることができます。ファイルにサーバー側の暗号化を設定できます。このダイアログボックスには、ファイルへの https リンクも表示されます。このリンクを選択すると、Toolkit for Visual Studio) はデフォルトのブ

ラウザでファイルを開きます。ファイルに対するパーミッションが [] に設定されている場合開く/ダウンロードそして全員をクリックすると、このリンクを使って他のユーザーがファイルにアクセスできます。このリンクを配布するのではなく、署名付き URL を作成して配布することをお勧めします。



署名付き URL の作成

Amazon S3 に保存されているコンテンツに他のユーザーがアクセスできるように配布する、時間制限のある署名付き URL を作成します。

署名付き URL の作成方法

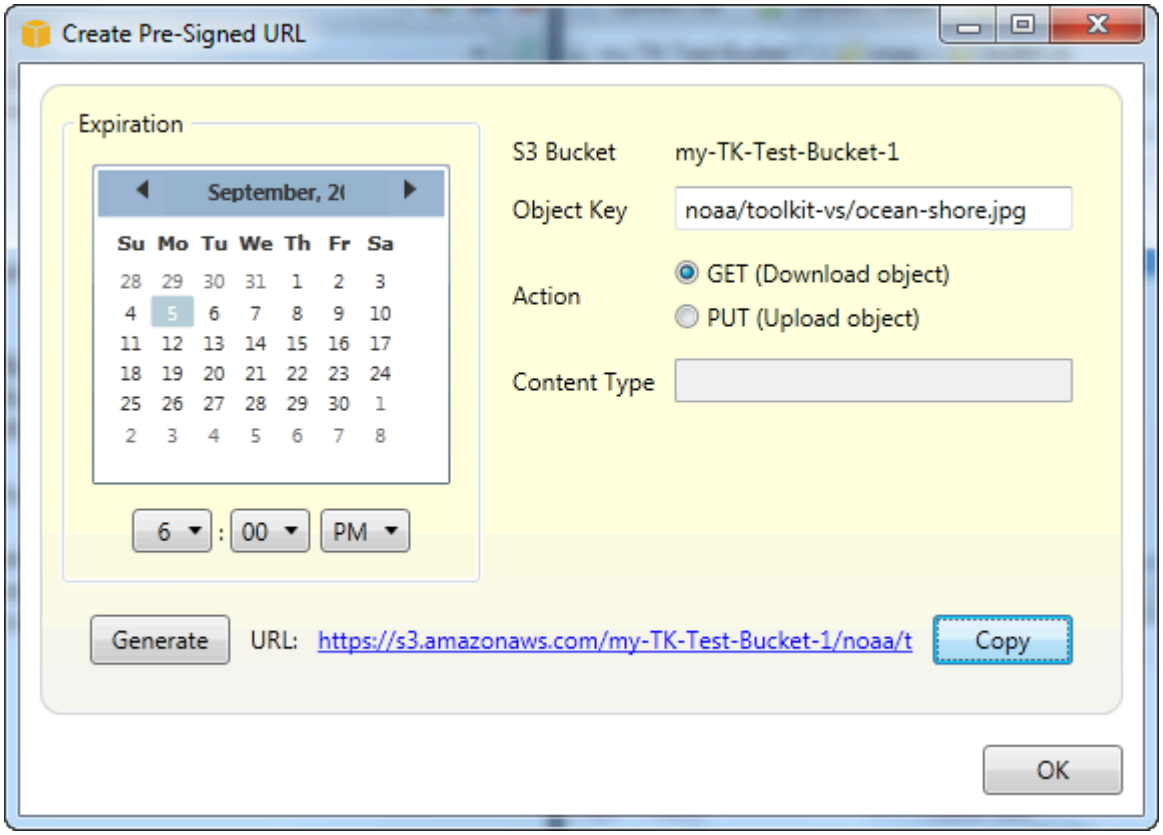
バケットまたはバケット内のファイルに署名付き URL を作成できます。他のユーザーはこの URL を使用して、バケットまたはファイルにアクセスできます。この URL は、URL の作成時に指定した有効期限になると、失効します。

署名付き URL を作成するには

1. [Create Pre-Signed URL (署名付き URL の作成)] ダイアログボックスで、URL の有効期限の日時を設定します。デフォルト設定では、現在の時間から 1 時間後に設定されます。

2. [Generate (生成)] ボタンを選択します。

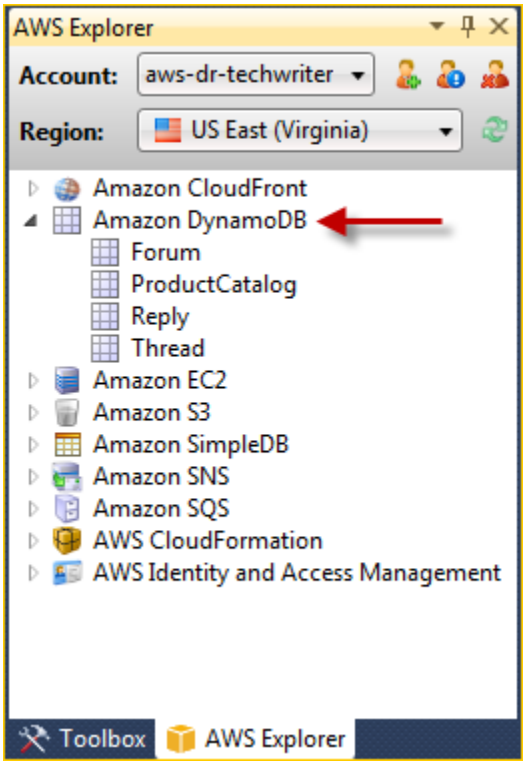
3. URL をクリップボードにコピーするには、[Copy (コピー)] を選択します。



DynamoDB from を使用するAWSエクスプローラ

Amazon DynamoDB は、拡張性と可用性に優れた、費用効果の高い、高速な非リレーショナルデータベースサービスです。DynamoDB により、データストレージに対して低いレイテンシーと予測可能なパフォーマンスを維持しながら、従来の拡張性の限界を排除できます。Toolkit for Visual Studio には、開発の場面での DynamoDB の操作を行う機能が用意されています。DynamoDB の詳細については、「」を参照してください。DynamoDBAmazon Web Services ウェブサイトで。

Toolkit for Visual Studio では、AWSExplorer では、アクティブに関連付けられているすべての DynamoDB テーブルが表示されます。AWS アカウント。



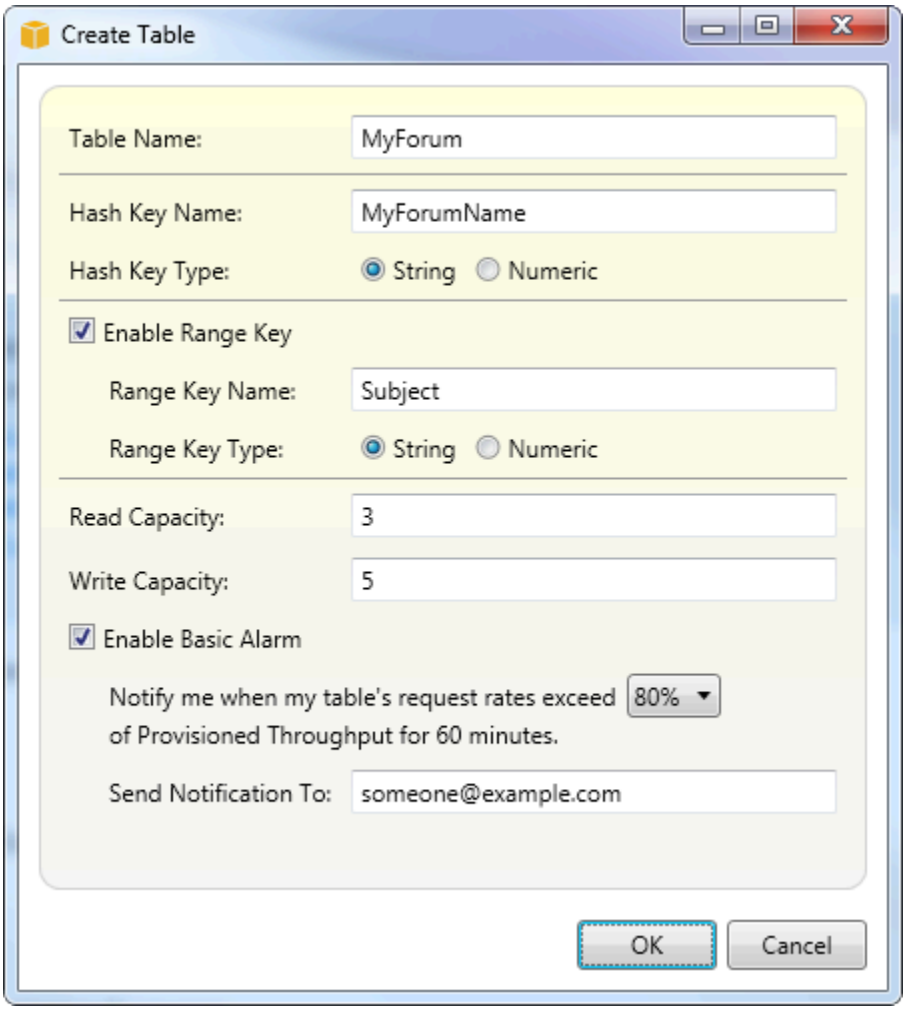
DynamoDB テーブルの作成

Toolkit for Visual Studio を使用して DynamoDB テーブルを作成できます。

でテーブルを作成するにはAWSエクスプローラ

1. EclipseAWSExplorer で、[] のコンテキスト (右クリック) メニューを開きます。Amazon DynamoDB[] を選択してから、[Create table。
2. [Create Table (テーブルの作成)] ウィザードで、[Table Name (テーブル名)] に、テーブルの名前を入力します。
3. 左ハッシュキー名フィールドに、プライマリハッシュキー属性を入力し、ハッシュキータイプ[] ボタンで、ハッシュキーのタイプを選択します。DynamoDB によって、プライマリキー属性を使用するハッシュインデックス (ソートなし) が生成され、必要に応じて、レンジプライマリキー属性を使用するレンジインデックス (ソートあり) が生成されます。プライマリハッシュキー属性の詳細については、を参照してください。プライマリキー「」セクションAmazon DynamoDB 開発者ガイド。
4. (オプション) [Enable Range Key (レンジキーを有効にする)] を選択します。[Range Key Name (レンジキー名)] フィールドにレンジキー属性を入力し、[Range Key Type (レンジキータイプ)] ボタンでレンジキータイプを選択します。

5. [Read Capacity (読み込みキャパシティー)] フィールドで、読み込みキャパシティーユニットの数を入力します。[Write Capacity (書き込みキャパシティー)] フィールドで、書き込みキャパシティーユニットの数を入力します。読み込みキャパシティーユニット数として 3 以上、書き込みキャパシティーユニット数として 5 以上の値を指定する必要があります。読み込みおよび書き込みキャパシティーユニットに関する情報は、「プロビジョニングされたスループット」を参照してください。
6. (オプション) [Enable Basic Alarm (基本アラームを有効にする)] をクリックすると、テーブルのリクエスト率が高すぎるときに警告を発します。プロビジョニングされたスループットの割合 (60 分ごと) のしきい値を選択します。この値を超えるとアラートが送信されます。[Send Notifications To (通知の送信先)] に E メールアドレスを入力します。
7. [OK] をクリックすると、テーブルが作成されます。

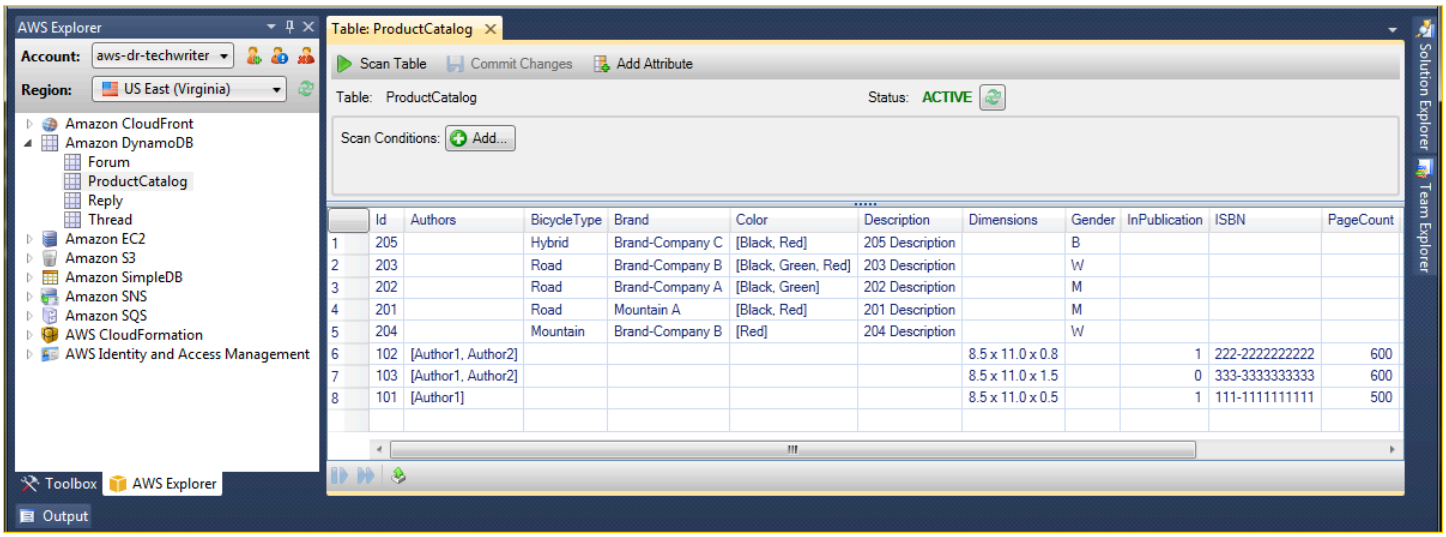


DynamoDB テーブルの詳細については、「」を参照してください。データモデルのコンセプト-テーブル、項目、属性。

DynamoDB テーブルをグリッドとして表示

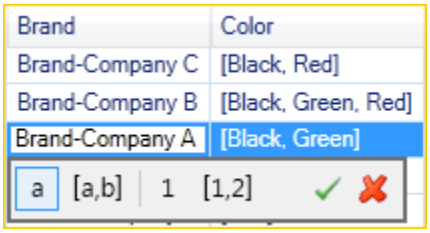
いずれかの DynamoDB テーブルのグリッドビューを開くには、[] をクリックします。AWSExplorer で、テーブルに対応するサブノードをダブルクリックします。グリッドビューで、テーブルに格納されている項目、属性、値を表示できます。各行は、テーブル内の項目に対応しています。各列は、テーブル内の属性に対応しています。グリッドビューの各セルには、その項目のその属性に関連付けられている値が表示されます。

属性の値は、文字列または数字です。文字列または数字のセットになる場合もあります。セット値は、角かっこで囲まれたカンマ区切りのリストとして表示されます。

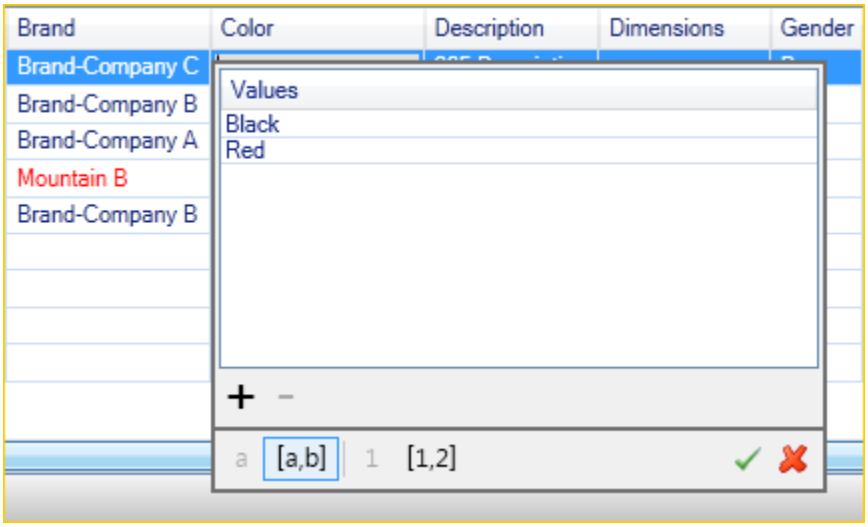


属性と値を編集および追加する

セルをダブルクリックすることで、項目の対応する属性の値を編集できます。セット値の属性の場合は、セットの個々の値を追加したり削除したりできます。



属性の値を変更するだけでなく、一部制限はありますが、属性の値の形式も変更できます。たとえば、任意の数値を文字列値に変換できます。文字列値があり、その内容が「125」などの数字である場合は、セル工ディタにより値の形式を文字列から数字に変換できます。また、単一の値をセット値に変換できます。ただし一般的に、セット値から 1 つの値に変換することはできません。その例外として、セット値の値が実際は 1 つしかない場合は 1 つの値に変換できます。

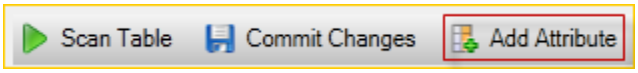


属性値の編集後、変更を確定するには、緑のチェックマークを選択します。変更を破棄する場合は、赤の [X] をクリックします。

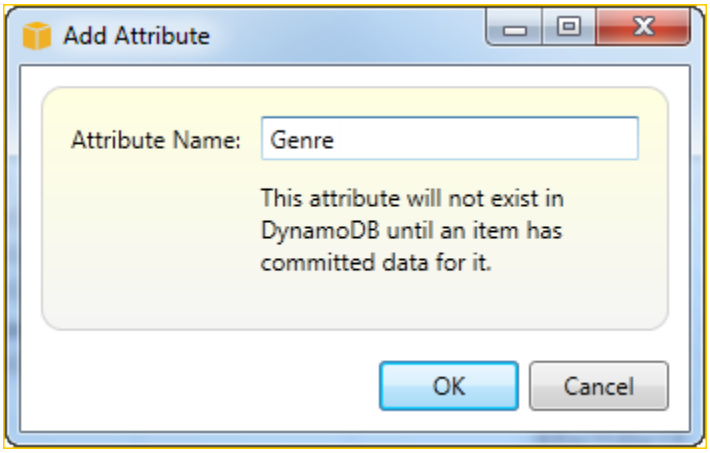
変更の確定後、属性値は赤で表示されます。これは、属性は更新されたが新しい値が DynamoDB データベースにまだ書き戻されていないことを示します。DynamoDB に変更を書き戻すには、[変更をコミットする。変更を破棄するには、[Scan Table (テーブルスキャン)] をクリックし、スキャン前に変更をコミットするかどうか Toolkit によって尋ねられたら、[No] をクリックします。

属性の追加

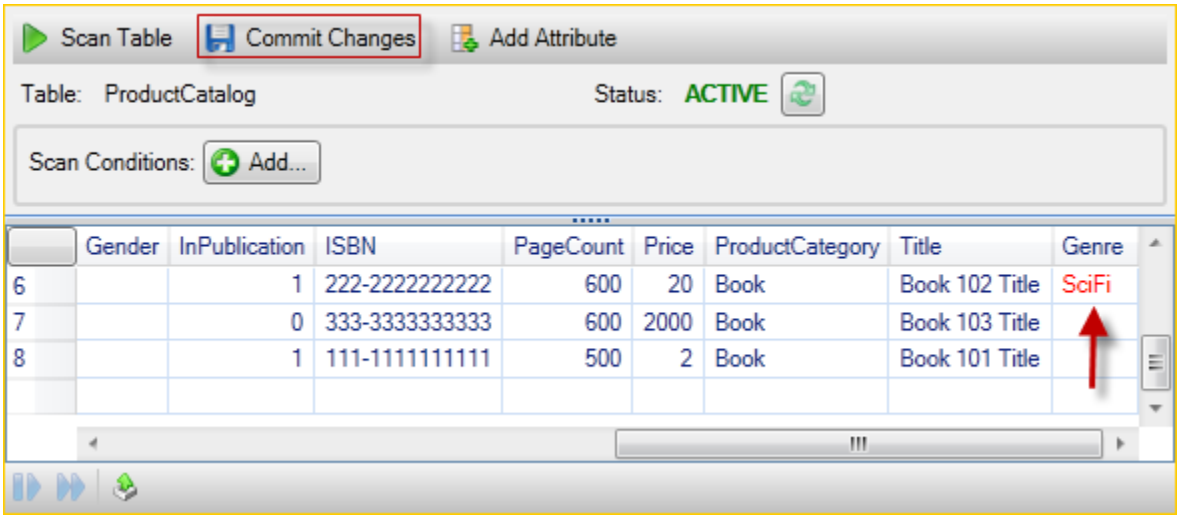
グリッドビューで、テーブルに属性を追加することもできます。新しい属性を追加するには、[Add Attribute (属性を追加)] を選択します。



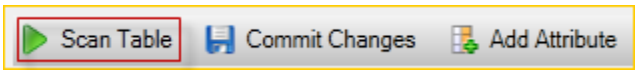
[Add Attribute (属性を追加)] ダイアログボックスに使用する属性の名前を入力して、[OK] をクリックします。



新しい属性をテーブルの一部にするには、少なくとも 1 つの項目に値を追加し、[Commit Changes (変更をコミット)] ボタンをクリックします。新しい属性を破棄するには、[Commit Changes (変更をコミット)] を選択せずにテーブルのグリッドビューを閉じるだけです。



DynamoDB テーブルのスキャン

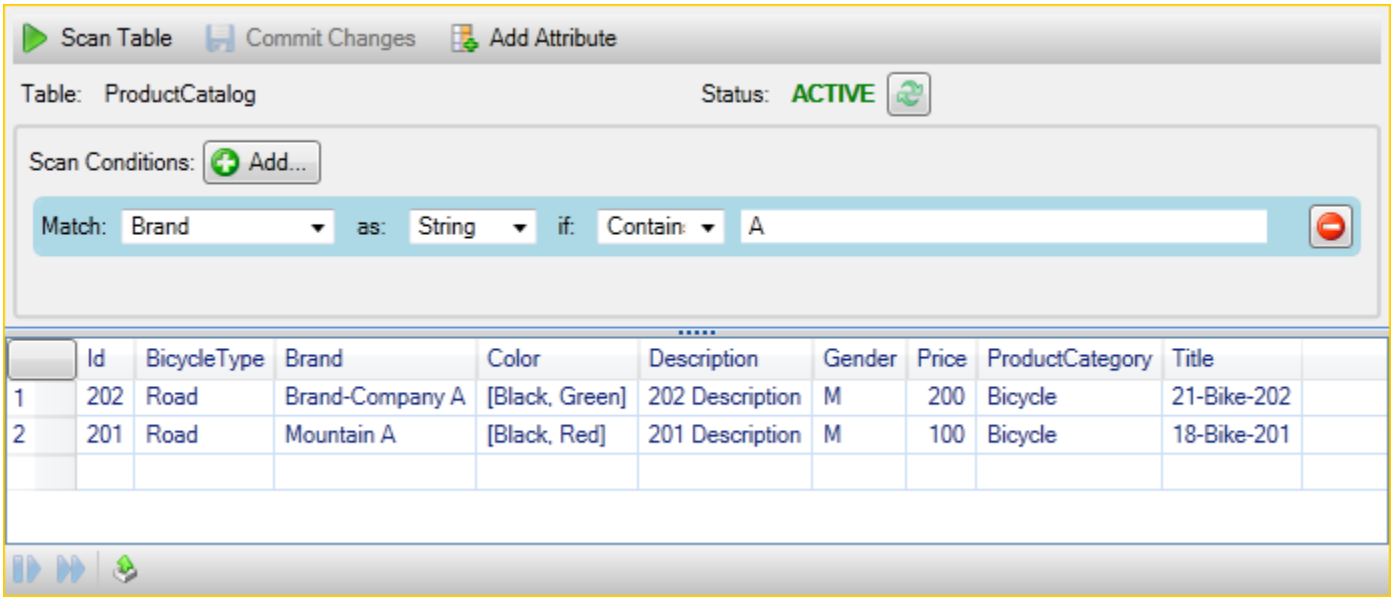


Toolkit から、DynamoDB テーブルのスキャンを実行できます。スキャンでは、定義した一連の条件に一致したテーブルの項目がすべて返されます。スキャンはワークロードの高いオペレーションであるため、テーブルに対する優先度の高い本稼働トラフィックが中断されないように、注意して使用する必要があります。スキャンオペレーションの使用の詳細については、「」を参照してください。Amazon DynamoDB 開発者ガイド。

から DynamoDB テーブルのスキャンを実行するにはAWSエクスプローラ

1. グリッドビューで、[Scan Conditions: Add (スキャン条件: 追加)] ボタンをクリックします。
2. スキャン句の工ディタで、一致条件の対象属性、属性値の解釈方法 (文字列、数値、セット値)、一致の方法 ([Begins With]、[Contains] など)、一致すべきリテラル値を選択します。
3. 検索での必要に応じてスキャン句を追加します。スキャンによってすべてのスキャン句の条件に一致する項目のみが返されます。スキャンでは、文字列値に対する一致を調べるときに大文字と小文字が区別されます。
4. グリッドビューの上部にあるボタンバーで、[Scan Table (テーブルスキャン)] を選択します。

スキャン句を削除するには、各フレーズの右側にある赤色のボタン (-) を選択します。



すべての項目が表示されたテーブルのビューに戻るには、すべてのスキャン句を削除し、[Scan Table(テーブルスキャン)] を再度選択します。

スキャン結果をページ分割する

ビューの下部には 3 つのボタンがあります。

最初の 2 つの青色のボタンをクリックすると、スキャン結果がページ分割されます。最初のボタンでは結果の次のページが表示されます。2 番目のボタンでは、結果の 10 ページ先が表示されます。この場合、1 ページは 1 MB に相当します。

スキャン結果を CSV にエクスポートする

3 番目のボタンをクリックすると、現在のスキャンの結果が CSV ファイルにエクスポートされます。

を使用するAWS CodeCommitVisual Studio Team Explorer

次を使用できます。AWS Identity and Access Management(IAM) ユーザーアカウントを使って、Git 認証情報を作成し、それを使ってリポジトリの作成とクローンの作成を行います。

AWS CodeCommit の認証情報の種類

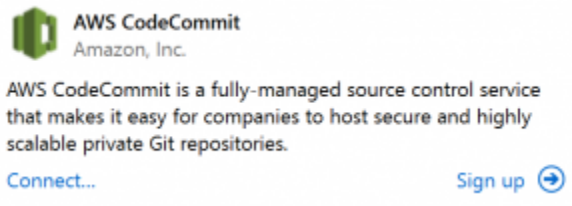
MostAWS Toolkit for Visual Studioユーザーは設定を認識しているAWSアクセスキーとシークレットキーを含む認証情報プロファイル。これらの認証情報プロファイルは、Toolkit for Visual Studio で使用され、のサービスの API に対する呼び出し (たとえば、で Amazon S3 バケットをリストするための) を可能にします。AWSExplorer または Amazon EC2 インスタンスの起動 AWS CodeCommit と Team Explorer の統合も、これらの認証情報プロファイルを使用します。ただし、Git 自体を使用するには、追加の認証情報が必要となります。具体的には、HTTPS 接続のための Git 認証情報が必要となります。これらの認証情報 (ユーザー名とパスワード) については、Git 認証情報を使用する HTTPS ユーザー用のセットアップのAWS CodeCommitユーザーガイド。

の Git 認証情報を作成できます。AWS CodeCommitIAM ユーザーアカウントに対してのみ。ルートアカウントに作成することはできません。サービスに対してこれらの認証情報のセットを 2 つまで作成できます。認証情報のセットを非アクティブとしてマークすることができますが、非アクティブなセットは引き続き 2 セットの制限にカウントされます。認証情報はいつでも削除と再作成を行うことができます。使用する場合AWS CodeCommitVisual Studio 内から、従来のAWS認証情報はサービス自体を操作するために使用されます。たとえば、リポジトリの作成や一覧表示を行う場合などです。AWS CodeCommit でホストされている実際の Git リポジトリを使用する場合は、Git の認証情報を使用します。

のサポートの一部としてAWS CodeCommitの場合、Toolkit for Visual Studio によって、これらの Git 認証情報は自動的に作成されて管理され、AWS認証情報プロファイル。Team Explorer 内で Git 操作を実行するために適切な認証情報のセットが設定されているかどうかを心配する必要はありません。でチームエクスプローラーに接続したらAWS認証情報プロファイルを使用すると、Git remote を使用する場合は常に関連付けられている Git 認証情報が自動的に使用されます。

AWS CodeCommit に接続する

Visual Studio 2015 以降で Team Explorer ウィンドウを開くと、[Manage Connections] の [Hosted Service Providers] セクションに AWS CodeCommit エントリが表示されます。

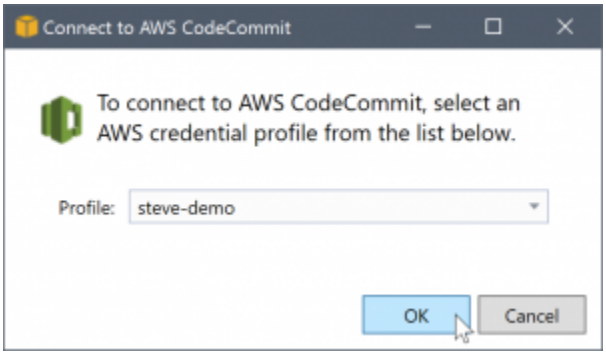


選ぶサインアップは、ブラウザウィンドウでAmazon Web Services ホームページを開きます。選択した場合はどうなるか接続Toolkit for Visual Studio で認証情報プロファイルが見つかるかどうかによって異なります。AWSアクセスキーと秘密キーを使用して、AWSはユーザーに代わって行われま

す。Toolkit for Visual Studio でローカルに保存された認証情報が見つからない場合には、IDE に表示される新しい [ご利用開始にあたって] のページを使って認証情報プロファイルが設定されている場合もあります。あるいは、Toolkit for Visual Studio を使用していた可能性があります。AWS Tools for Windows PowerShell、またはAWS CLIそしてすでに持っているAWSToolkit for Visual Studio で使用できる認証情報プロファイル。

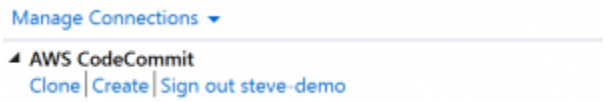
を選択する場合接続の場合、Toolkit for Visual Studio は、接続で使用する認証情報プロファイルの検索を始めます。Toolkit for Visual Studio で認証情報プロファイルが見つからない場合は、ダイアログボックスが表示され、そこでアクセスキーとシークレットキーを入力できます。AWS アカウント。ルート認証情報ではなく、IAM ユーザーアカウントを使用することをお勧めします。また、前に説明したように、最終的に必要となる Git 認証情報は IAM ユーザーにのみ作成できます。アクセスキーとシークレットキーが用意され、認証情報プロファイルが作成されると、Team Explorer と AWS CodeCommit の間の接続を使用できるようになります。

Toolkit for Visual Studio で複数の Visual Studio が検出された場合AWS認証情報プロファイルの場合、Team Explorer で使用するアカウントを選択するように求められます。



認証情報プロファイルが 1 つのみの場合は、Toolkit for Visual Studio ではプロファイルの選択ダイアログボックスは表示されずに、直ちに接続されます。

認証情報プロファイルを使って Team Explorer と AWS CodeCommit 間で接続が確立されると、ダイアログボックスは閉じられて、接続パネルが表示されます。

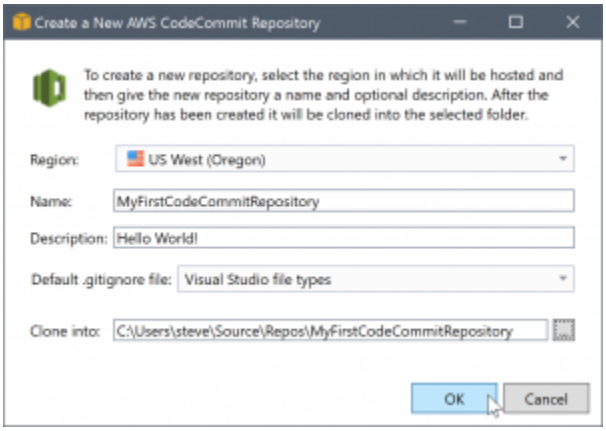


ローカルにクローンが作成されたリポジトリがないため、パネルには実行可能な操作のみが表示されます。クローン,作成, およびサインアウト。他のプロバイダーと同様に、AWS CodeCommitTeam Explorer では、1 つのみにバインドできます。AWS任意の時点で認証情報プロファイル。アカウントを切り替えるには、[Sign out (サインアウト)] を使って接続を削除し、別のアカウントを使って新しい接続を開始します。

接続を確立できたら、[Create] リンクをクリックして、リポジトリを作成することができます。

リポジトリの作成

クリックすると作成link、新しいを作成するAWS CodeCommitリポジトリダイアログボックスが開きます。



AWS CodeCommit リポジトリはリージョンごとに整理されており、リポジトリをホストするリージョンを [Region] で選択できます。リストには、AWS CodeCommit をサポートしているすべてのリージョンが表示されます。新しいリポジトリの名前 (必須) と説明 (オプション) を入力します。

ダイアログボックスのデフォルトの動作では、新しいリポジトリのフォルダの場所にリポジトリの名前のサフィックスを追加します (名前を入力すると、フォルダの場所も更新されます)。別のフォルダ名を使用するには、リポジトリの名前の入力の終了後に [Clone into] のフォルダのパスを編集します。

リポジトリに `.gitignore` の初期ファイルを自動的に作成することを選ぶこともできます。AWS Toolkit for Visual Studio は、Visual Studio のファイルの種類の組み込みのデフォルトを提供します。ファイルを使用しないか、または既存のカスタムのファイルの使用を選択して複数のリポジトリにわたって再利用することもできます。それには、リストで [Use custom (カスタムを使用)] を選択し、使用するカスタムファイルに移動します。

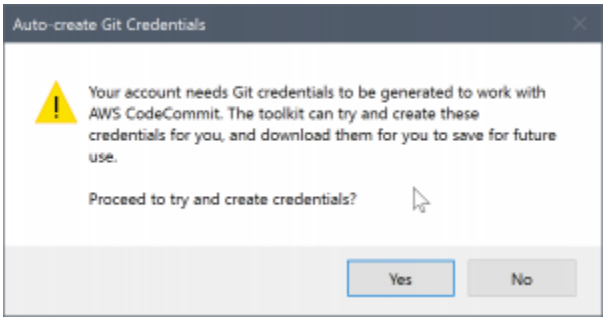
リポジトリ名と場所を取得したら、[] をクリックする準備ができました。OKリポジトリの作成を開始します。Toolkit for Visual Studio は、リポジトリの作成をサービスにリクエストし、新しいリポジトリのクローンをローカルに作成します。.gitignore ファイルを使用している場合には、初期のコミットを追加します。この時点で、Git remote を使用できるようになります。では、Toolkit for Visual Studio では、前に説明した Git 認証情報へのアクセスが必要になります。

Git 認証情報を設定する

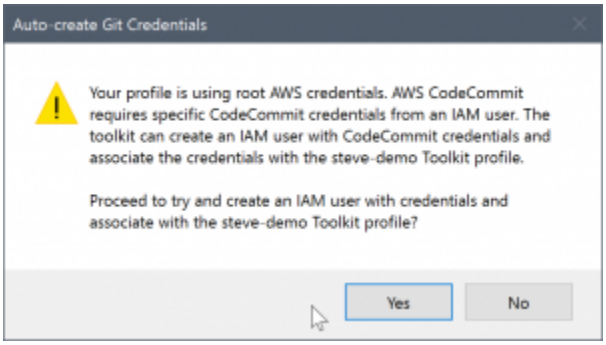
この時点まで使ってたなAWSアクセスキーとシークレットキーを使用して、リポジトリの作成をサービスにリクエストします。次に Git 自体を使って実際のクローンの操作を行う必要がありますが、Git は理解しませんAWSアクセスキーと秘密キー。代わりに、Git にユーザー名とパスワードの認証情報を指定する必要があります。Git はリモートでこれを HTTPS 接続に使用します。

で述べたようにGit 認証情報を設定するでは、使用する Git 認証情報は IAM ユーザーに関連付けられている必要があります。ルート認証情報に生成することはできません。はいつでもセットアップする必要があります。AWS認証情報プロファイル。ルートキーではなく、IAM ユーザーアクセスキーとシークレットキーが含まれます。Toolkit for Visual Studio では、の Git 認証情報をセットアップすることができます。AWS CodeCommitあなたのために、それらをAWS以前にチームエクスプローラーで接続するために使用した資格情報プロファイル。

を選択する場合OKの新しいを作成するAWS CodeCommitリポジトリダイアログボックスを開き、リポジトリを正常に作成すると、Toolkit for Visual Studio はAWSチームエクスプローラーで接続され、Git の認証情報かどうかを判断する資格情報プロファイルAWS CodeCommit存在し、プロファイルにローカルに関連付けられています。確認された場合、Toolkit for Visual Studio は Team Explorer に新しいリポジトリでクローン操作を開始するように指示します。Git 認証情報がローカルで使用できない場合、Toolkit for Visual Studio は Team Explorer の接続で使用されたアカウントの認証情報の種類を確認します。認証情報が推奨されたように IAM ユーザーのものである場合、次のメッセージが表示されます。

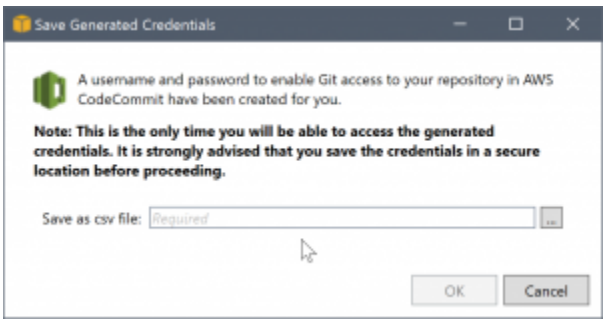


認証情報がルート認証情報の場合には、次のメッセージが表示されます。



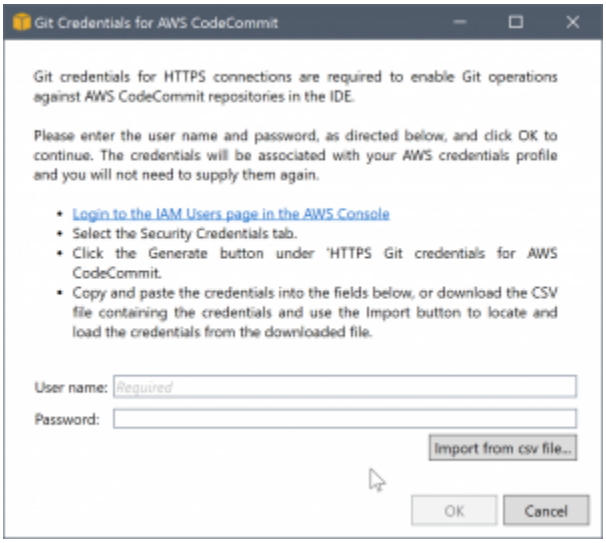
どちらの場合も、Toolkit for Visual Studio は必要な Git 認証情報を作成しようとします。最初のシナリオでは、作成する必要があるのは、IAM ユーザー用の Git 認証情報です。ルートアカウントが使用されている場合は、Toolkit for Visual Studio はまず IAM ユーザーの作成を試みて、次に新しいユーザーの Git 認証情報の作成に進みます。Toolkit for Visual Studio で新しいユーザーを作成する必要がある場合は、AWS CodeCommitユーザー管理ポリシーをその新しいユーザーアカウントに適用します。このポリシーでは、アクセスを AWS CodeCommit のみに許可します。リポジトリの削除以外のすべての操作を AWS CodeCommit を使って実行できるようにします。

認証情報を作成する際には、それを 1 回のみ表示できます。そのため、Toolkit for Visual Studio では、新しく作成した認証情報を.csv続行する前に、ファイルを入力します。



必ず安全な場所に保存しておくことを、強くお勧めします。

Toolkit for Visual Studio で認証情報を自動的に作成できない場合があります。たとえば、の Git 認証情報セットの最大数をすでに作成している可能性があります。AWS CodeCommit(2)、または、Toolkit for Visual Studio がプログラムによって認証情報を作成するために十分な権限がない場合 (IAM ユーザーとしてサインインしている場合) もあります。このような場合は、AWS Management Consoleをクリックして、認証情報を管理するか、または管理者から認証情報を取得します。その後、Git 認証情報AWS CodeCommitダイアログボックスが開きます。このダイアログボックスには、Toolkit for Visual Studio が表示されます。

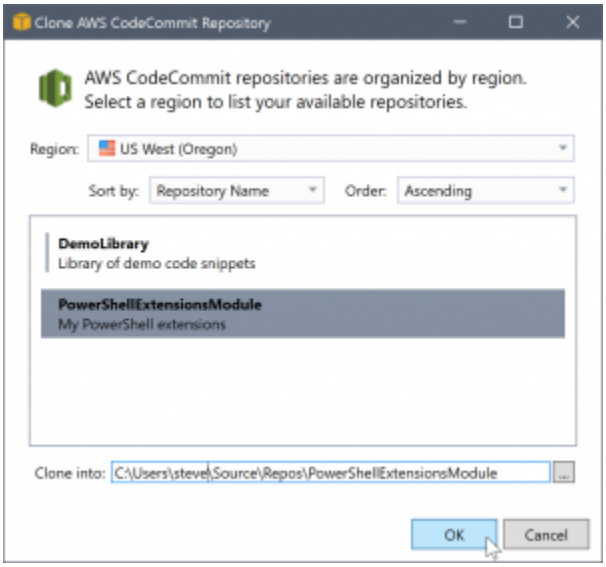


Git 認証情報が利用可能になったので、新しいリポジトリのクローン操作が進行します (Team Explorer 内の操作の進行状態の表示を確認してください)。デフォルトの `.gitignore` ファイルの適用を選択した場合、「Initial Commit」のコメントでリポジトリにコミットされます。

以上で Team Explorer 内で認証情報を設定し、リポジトリを作成できました。必要な認証情報が設定されたので、今後新しいリポジトリを作成する場合には、新しいを作成するAWS CodeCommitリポジトリダイアログボックス自体。

リポジトリのクローンを作成する

既存のリポジトリのクローンを作成するには、Team Explorer の AWS CodeCommit の接続パネルに戻ります。[] をクリックします。クローンリンクを開くにはクローンAWS CodeCommitリポジトリダイアログボックスで、クローンを作成するリポジトリと、配置するディスク上の場所を選択します。



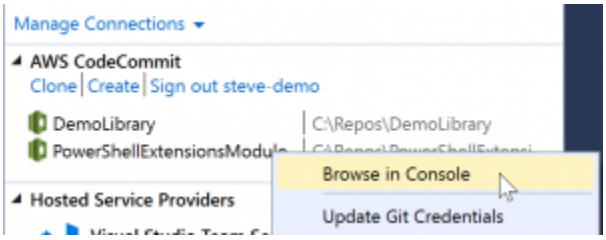
リージョンを選択すると、Toolkit for Visual Studio はサービスのクエリを行って、リージョンで使用できるリポジトリを検出し、それをダイアログボックスの中央のリストに表示します。各リポジトリの名前と説明 (オプション) も表示されます。リポジトリ名または最終更新日で、昇順または降順に、リストの順序をソートすることができます。

リポジトリを選択すると、クローンを作成する場所を選択できます。Team Explorer の他のプラグインで使用されたものと同じ場所がデフォルトとなりますが、他の場所を参照するか、または入力することもできます。デフォルトでは、リポジトリ名が、選択されたパスにサフィックスとして追加されます。ただし、特定のパスを指定する場合には、フォルダを選択した後にテキストボックスを編集します。[OK] をクリックしたときにボックスに含まれるテキストを使ったフォルダに、リポジトリのクローンが作成されます。

リポジトリとフォルダの場所を選択したら、OKをクリックして、クローンオペレーションを続行します。リポジトリの作成と同様に、クローン操作の進行状況も Team Explorer に表示されます。

リポジトリを操作する

リポジトリのクローンまたはリポジトリを作成する場合、接続のためのローカルリポジトリは、Team Explorer の操作のリンクの下の接続パネルに一覧表示されます。これらのエントリを使うと、リポジトリにアクセスしてコンテンツを参照するのに便利です。リポジトリを右クリックして、[Browse in Console] を選択します。



また、Git 認証情報を更新するをクリックして、認証情報プロファイルに関連付けられている保存された Git 認証情報を更新します。これは、認証情報をローテーションする場合に役立ちます。コマンドを実行すると、Git 認証情報AWS CodeCommitダイアログボックスでは、新しい認証情報を入力またはインポートすることができます。

リポジトリ上での Git 操作は、期待どおりに機能します。ローカルコミットを行うことができ、共有する準備ができたら、Team Explorer で [Sync] オプションを使用します。Git の認証情報はすでにローカルに保存され、接続されているものに関連付けられているからです。AWS認証情報プロファイル。AWS CodeCommitリモートの。

Visual Studio で CodeArtifact を使用する

AWS CodeArtifactは、フルマネージドのアーティファクトリポジトリサービスであり、組織がアプリケーション開発に使用するソフトウェアパッケージを安全に保存および共有できるようにします。CodeArtifact は、NuGet、.NET Core CLI、Visual Studio などの一般的なビルドツールやパッケージマネージャーで使用できます。また、次のような外部のパブリックリポジトリからパッケージをプルするように CodeArtifact を設定することもできます。Nuget.org。

CodeArtifact では、パッケージはリポジトリに格納され、リポジトリはドメイン内に保存されます。-AWS Toolkit for Visual StudioCodeArtifact リポジトリを使用して Visual Studio の構成を簡素化し、CodeArtifact と NuGet.org の両方から Visual Studio のパッケージを簡単に利用できます。

CodeArtifact リポジトリを NuGet パッケージソースとして追加します。

CodeArtifact からパッケージを使用するには、リポジトリをパッケージソースとしてNuGet パッケージマネージャVisual Studio で

リポジトリをパッケージソースとして追加するには

1. EclipseAWSExplorer で、リポジトリに移動します。AWS CodeArtifactノード。
2. 追加するリポジトリのコンテキスト (右クリック) メニューを開き、NuGet ソースエンドポイントをコピーする。

3. に移動します。パッケージソースその下にNuGet パッケージマネージャ内のノードツール > オプションメニュー。
4. Eclipseパッケージソースで、プラス記号 () を選択します。+) をクリックし、名前を編集し、以前にコピーした NuGet ソースエンドポイントの URL を送信元フィールド。
5. 新しく追加したパッケージソースの横にあるチェックボックスを選択して有効にします。

> **Note**
>
> 外部接続をに追加することをお勧めしますNuget.orgCodeArtifact にアクセスして無効にするnuget.orgVisual Studio で、パッケージソースを指定します。外部接続を使用する場合、すべての依存関係が取得されますNuget.orgCodeArtifact に格納されます。もしNuget.org何らかの理由で停止しても、必要なパッケージはまだ利用可能になります。外部接続の詳細については、「」を参照してください。外部接続を追加するのAWS CodeArtifactユーザーガイド。

6. 選択OKをクリックしてメニューを閉じます。

Visual Studio で CodeArtifact を使用する方法については、「」を参照してください。Visual Studio で CodeArtifact を使用するのAWS CodeArtifactユーザーガイド。

Amazon RDSAWSエクスプローラ

Amazon Relational Database Service (Amazon RDS) は、クラウドでの SQL リレーショナルデータベースシステムのプロビジョニングと管理を行うサービスです。Amazon RDS は、3 つの種類のデータベースシステムをサポートします。

- MySQL Community Edition
- Oracle Database Enterprise Edition
- Microsoft SQL Server (Express Edition、Standard Edition、または Web Edition)

詳細は、『Amazon RDS ユーザーガイド』をご覧ください。

ここで説明されている機能の多くは、AWSマネジメントコンソールAmazon RDS

トピック

- Amazon RDS データベースインスタンスを起動する

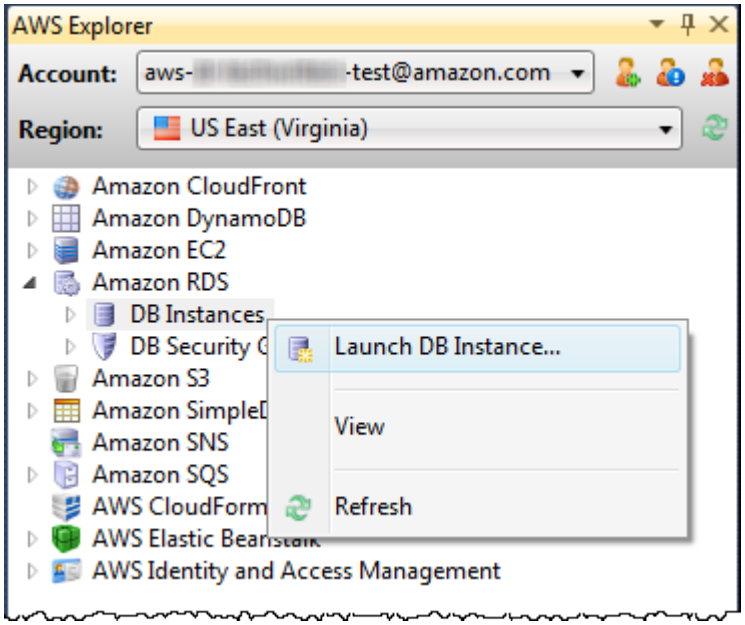
- RDS インスタンスでの Microsoft SQL Server データベースの作成
- Amazon RDS セキュリティグループ

Amazon RDS データベースインスタンスを起動する

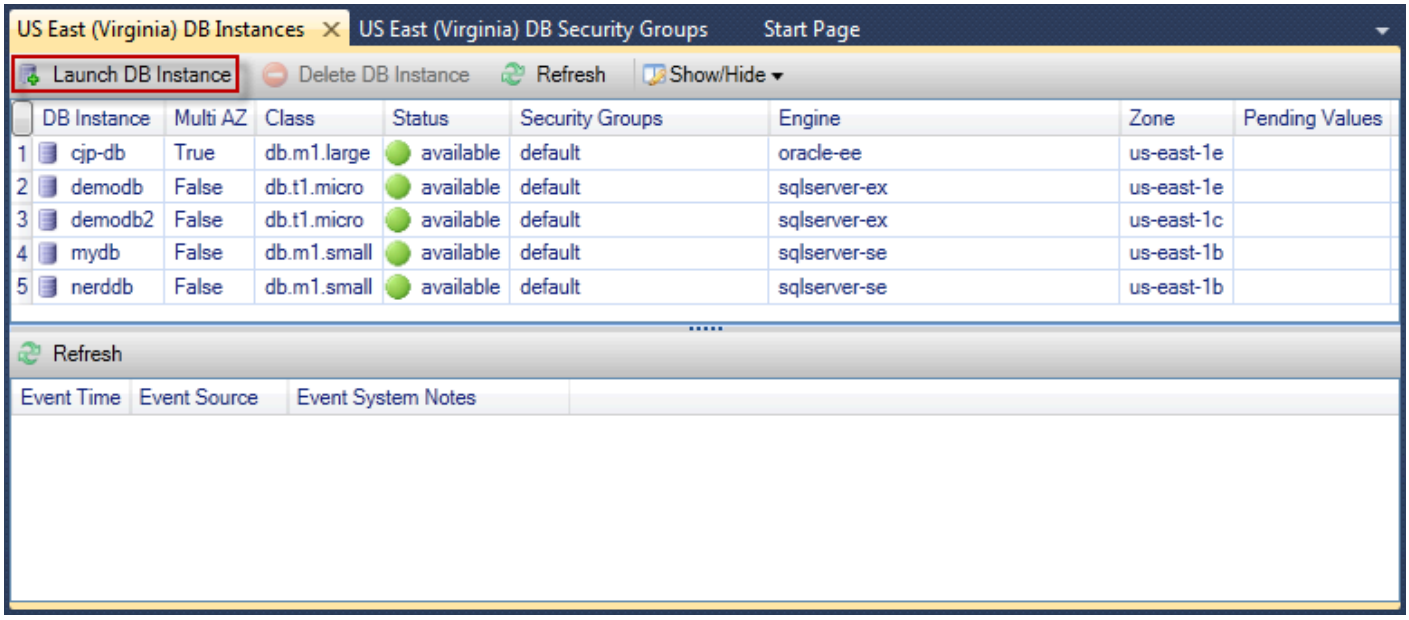
とAWSExplorer では、Amazon RDS がサポートする任意のデータベースエンジンのインスタンスを起動できます。次のウォークスルーでは、Microsoft SQL Server Standard Edition のインスタンスを起動するためのユーザーエクスペリエンスについて記していますが、ユーザーエクスペリエンスはサポートするすべてのエンジンで類似しています。

Amazon RDS インスタンスを起動するには

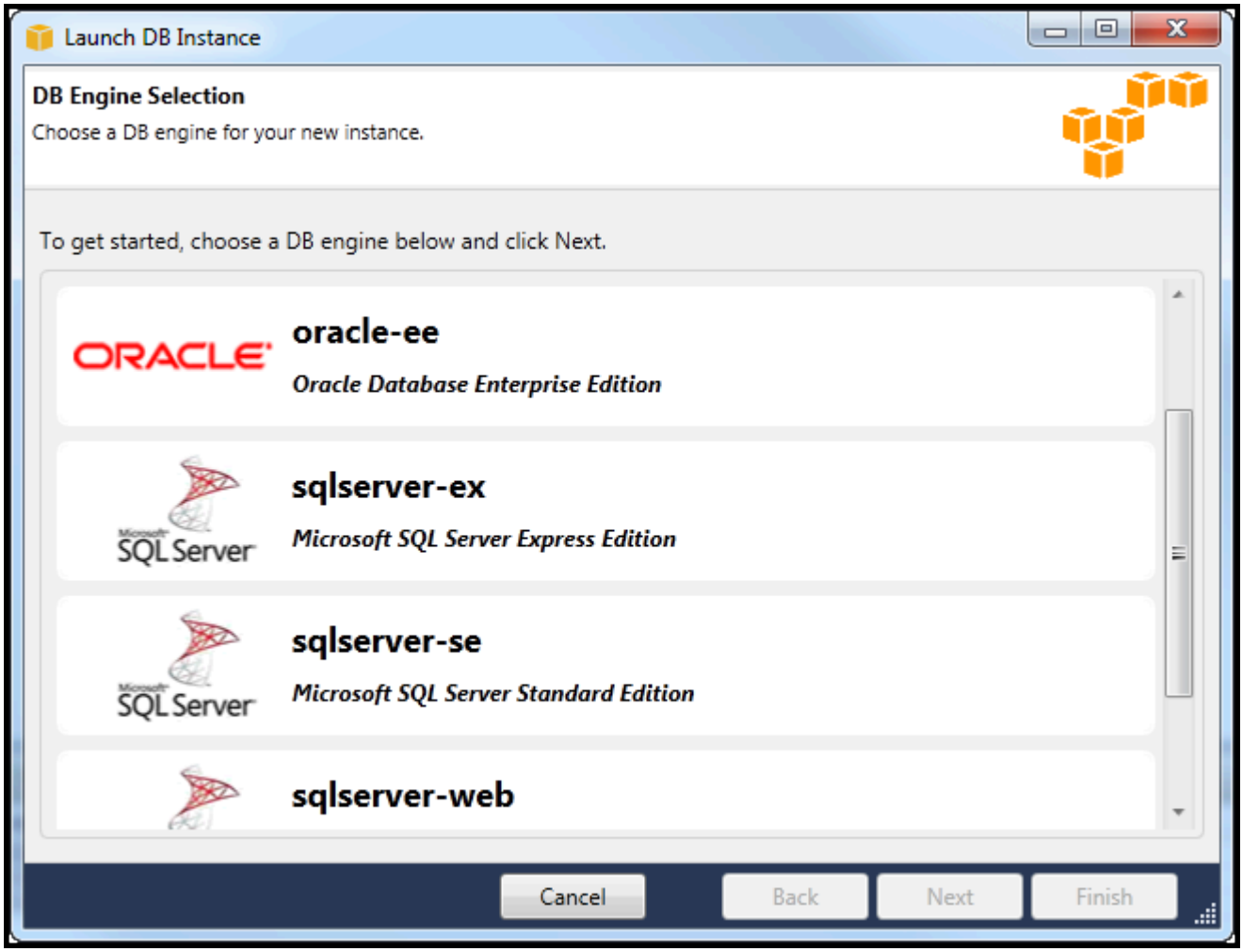
1. EclipseAWSExplorer で、のコンテキスト (右クリック) メニューを開きます。Amazon RDSノードを選択し、DB インスタンスの起動。



   または、[DB Instances (DB インスタンス)] タブで、[Launch DB Instance (DB インスタンスの起動)] を選択します。



2. [DB Engine Selection (DB エンジンの選択)] ダイアログボックスで、起動するデータベースエンジンのタイプを選択します。このウォークスルーでは、Microsoft SQL Server Standard Edition (sqlserver-se) を選択し、[Next (次へ)] を選択します。



3. [DB Engine Instance Options (DB エンジンインスタンスオプション)] ダイアログボックスで、設定オプションを選択します。

   [DB Engine Instance Options and Class (DB エンジンインスタンスオプションとクラス)] セクションで、次の設定を指定できます。

   ライセンスモデル

エンジンのタイプ	ライセンス
Microsoft SQL Server	license-included
MySql	general-public-license
Oracle	自分のライセンス使用

データベースエンジンによってライセンスモデルは異なります。エンジンのタイプ、ライセンス、Microsoft SQL Server、license-included、MySql、general-public-license、Oracle、bring-your-own-license

DB インスタンスのバージョン

使用するデータベースエンジンのバージョンを選択します。サポートするバージョンが 1 つのみの場合は、そのバージョンが選択されています。

DB インスタンスクラス

データベースエンジンのインスタンスクラスを選択します。インスタンスクラスの料金はさまざまです。詳細については、Amazon RDS 料金表を参照してください。

マルチ AZ 配置の実行

このオプションを選択し、データの冗長性と可用性を強化するマルチ AZ 配置を作成します。Amazon RDS は、計画されたシステム停止または予期しない停止の際の自動フェイルオーバーのために、異なるアベイラビリティーゾーンにあるデータベースのスタンバイコピーのプロビジョニングと維持を行います。マルチ AZ 配置の料金の詳細については、Amazon RDS の詳細ページの料金表セクションを参照してください。このオプションは、Microsoft SQL Server ではサポートされていません。

マイナーバージョンの自動アップグレード

このオプションを選択し、AWSRDS インスタンスのマイナーバージョンアップを自動的に実行します。

[RDS Database Instance (RDS データーベースインスタンス)] セクションで、以下の設定を指定します。

ストレージ割り当て

エンジン	最小 (GB)	最大 (GB)
MySQL	5	1024
Oracle Enterprise Edition	10	1024

エンジン	最小 (GB)	最大 (GB)
Microsoft SQL Server Express Edition	30	1024
Microsoft SQL Server Standard Edition	250	1024
Microsoft SQL Server Web Edition	30	1024

割り当てられたストレージの最小値および最大値はデータベースエンジンのタイプによって異なります。Engine Minimum (GB)、Maximum (GB)、MySQL、5、1024、Oracle Enterprise Edition、10、1024、Microsoft SQL Server Express Edition、30、1024、Microsoft SQL Server Standard Edition、250、1024、Microsoft SQL Server Web Edition、30、1024

DB Instance Identifier

データベースインスタンスの名前を指定します。この値は大文字と小文字が区別されません。小文字で表示されますAWSExplorer。

マスターユーザー名

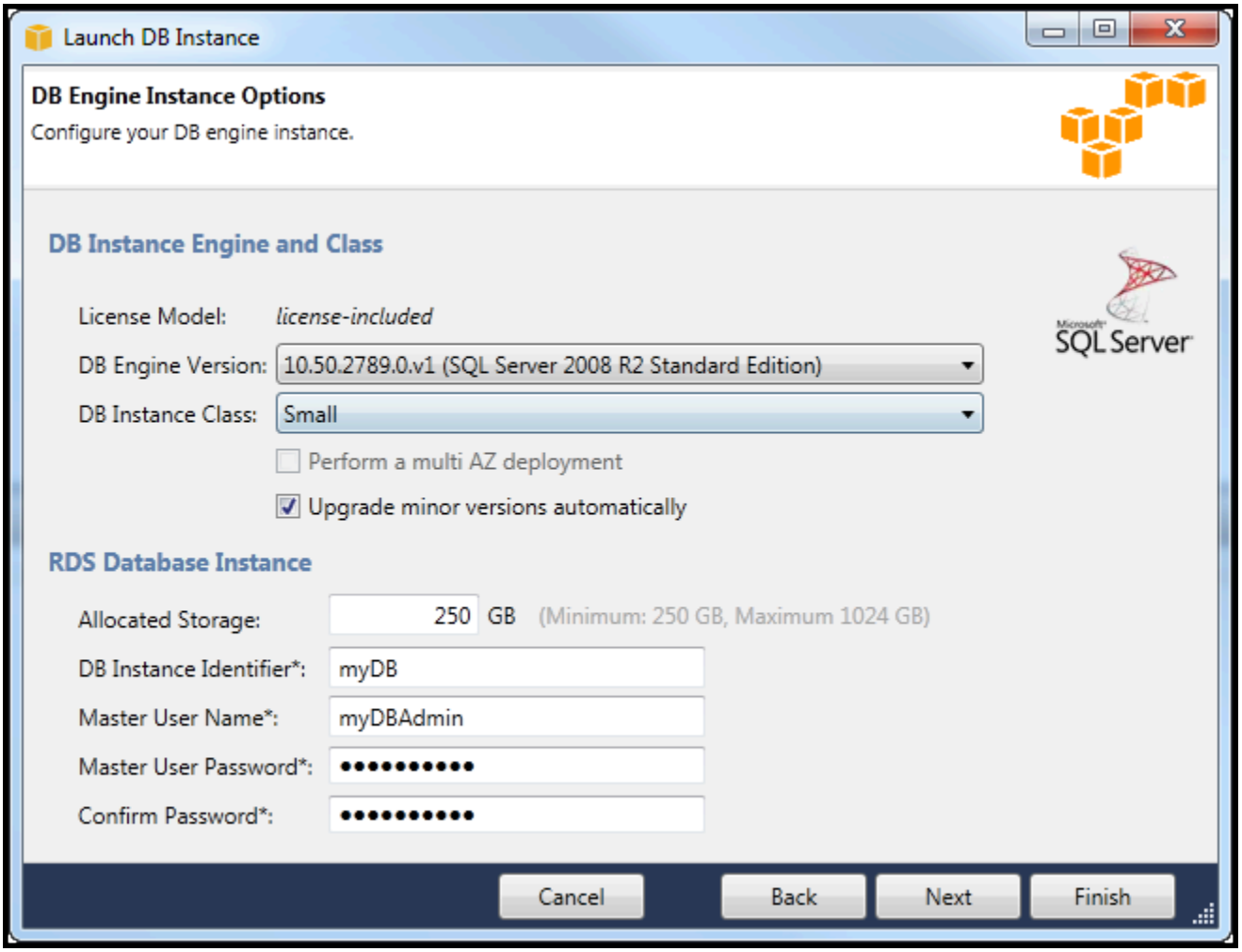
データベースインスタンスの管理者名を入力します。

マスターユーザのパスワード

データベースインスタンスの管理者パスワードを入力します。

[Confirm Password]

パスワードをもう一度入力して誤りがないことを確認します。



1. [Additional Options (追加オプション)] ダイアログボックスで、以下の設定を指定します。

   Database Port

   これは、ネットワークで通信を行う際にインスタンスが使用する TCP ポートです。コンピュータがファイアウォールを介してインターネットにアクセスしている場合は、ファイアウォールが許可するポートを設定します。

   アベイラビリティーゾーン

   リージョン内の特定のアベイラビリティーゾーンでインスタンスを起動する場合は、このオプションを使用します。指定したデータベースインスタンスがすべてのアベイラビリティーゾーンで使用できない場合があります。

   RDS Security Group

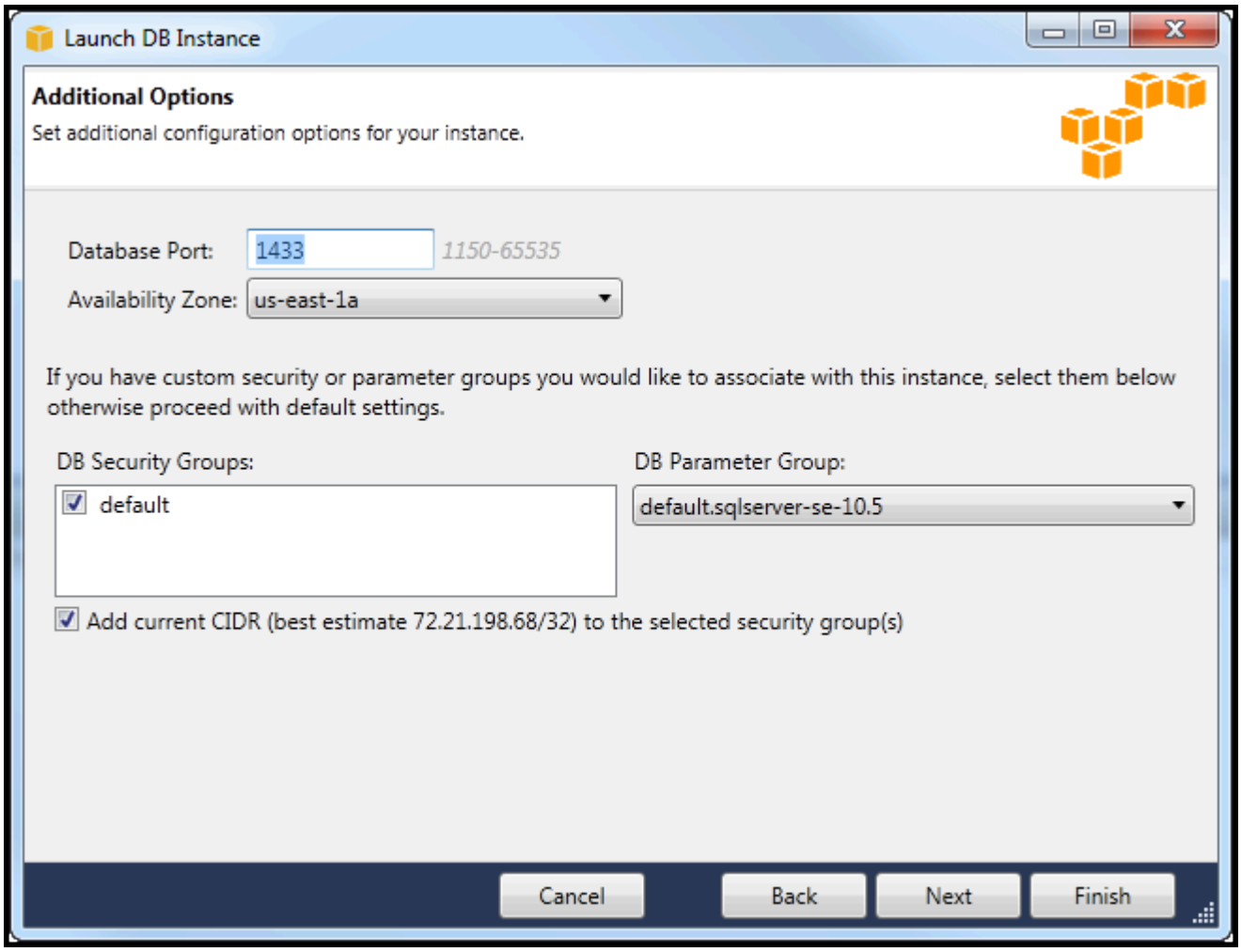
   インスタンスに関連付ける RDS セキュリティグループを 1 つ以上選択します。RDS セキュリティグループにより IP アドレス、Amazon EC2 インスタンス、およびAWS アカウントインス

タンスにアクセスすることが許可されています。RDS セキュリティグループの詳細については、Amazon RDS セキュリティグループを参照してください。Toolkit for Visual Studio はユーザーの現在の IP アドレスを特定しようと試み、このアドレスをインスタンスに関連付けられたセキュリティグループに追加するかを確認します。ただし、コンピュータがファイアウォール経由でインターネットにアクセスしている場合、Toolkit により生成されたコンピュータの IP アドレスが正しくない可能性があります。使用する IP アドレスを決定するには、システム管理者にお問い合わせください。

DB パラメータグループ

(オプション) このドロップダウンリストで、インスタンスに関連付ける DB パラメータグループを選択します。DB パラメータグループを使用することで、インスタンスのデフォルト設定を変更できます。詳細については、Amazon Relational Database Service ユーザーガイドおよびこちらの記事を参照してください。

このダイアログボックスで設定を指定している場合は、[Next (次へ)] を選択します。

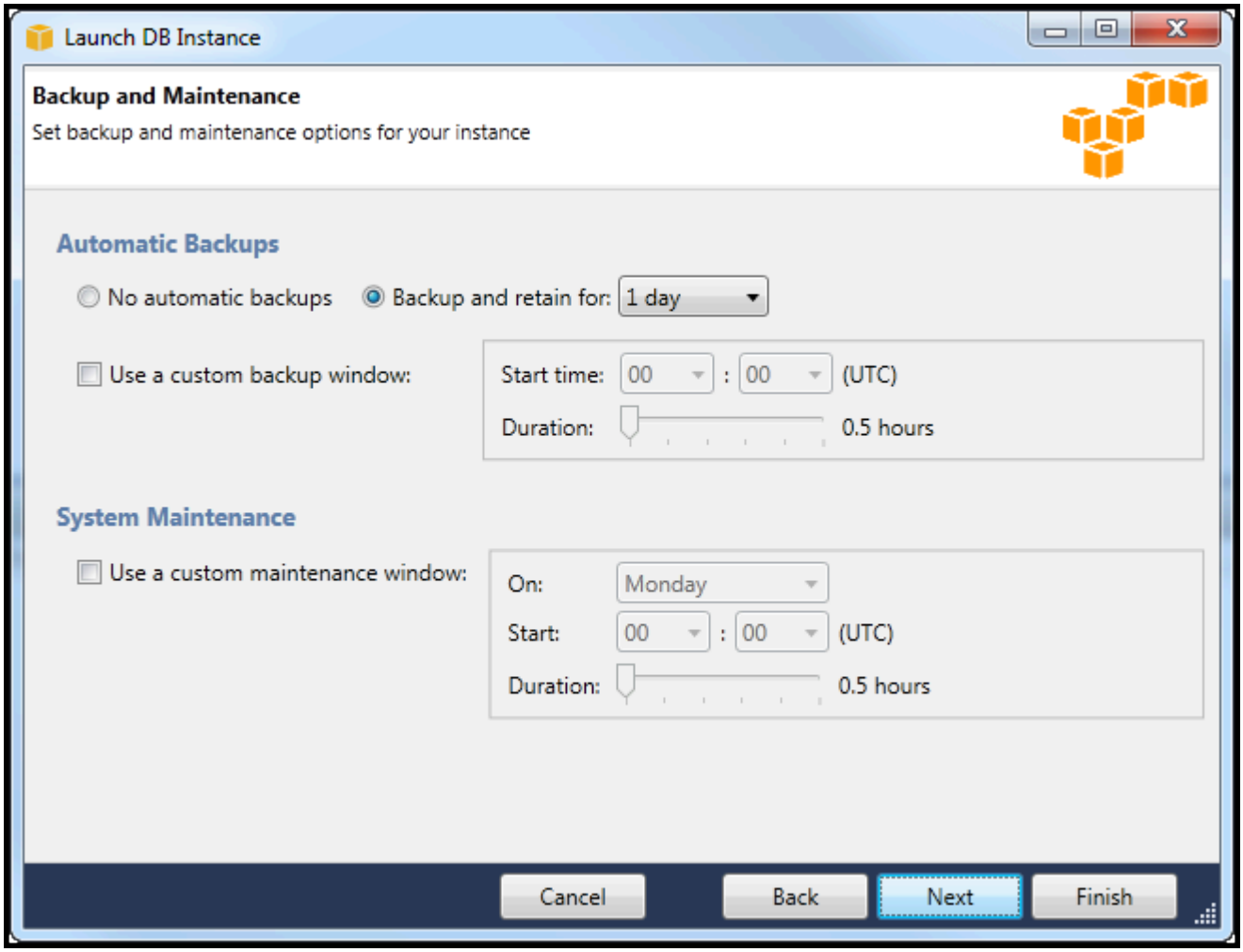


2. -Backup とメンテナンスダイアログボックスでは、Amazon RDS がインスタンスをバックアップするかどうか、そしてもしバックアップするのであれば、その保持期間を設定できます。バックアップを行う時間帯も指定できます。

   このダイアログボックスでは、Amazon RDS でインスタンスのシステムメンテナンスを行うかも指定できます。メンテナンスには、定期的なパッチとマイナーバージョンアップグレードが含まれます。

   システムメンテナンスに指定した時間帯とバックアップに指定した時間帯は重複できません。

   [Next] (次へ) をクリックします。



3. ウィザードの最後のダイアログボックスでは、インスタンスの設定を確認することができます。設定を変更する必要がある場合は、[Back (戻る)] ボタンをクリックします。すべての設定が正しい場合は、[Launch (起動)] を選択します。

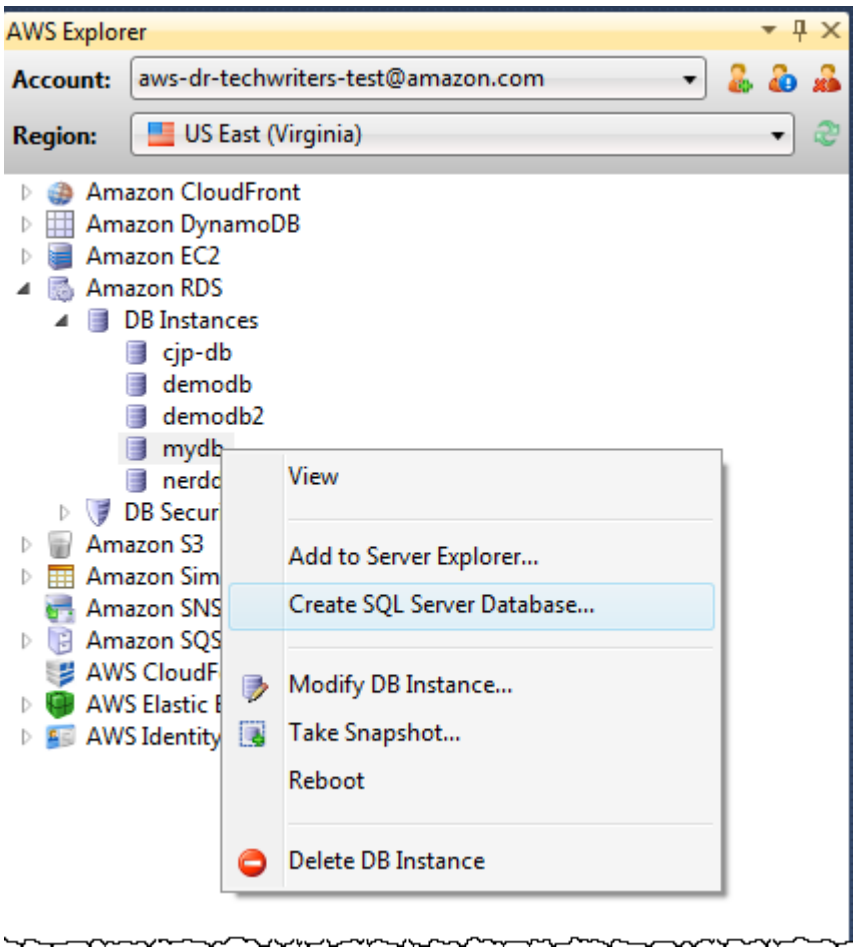
RDS インスタンスでの Microsoft SQL Server データベースの作成

Microsoft SQL Server は、Amazon RDS インスタンスの起動後に、RDS インスタンスに SQL Server データベースを作成する必要があるように設計されています。

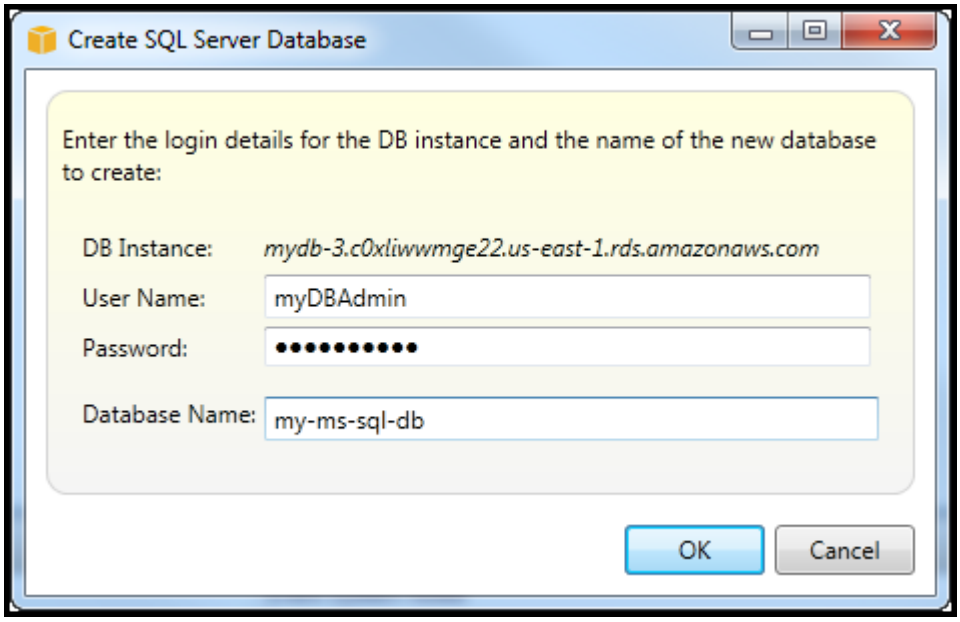
Amazon RDS インスタンスの作成方法については、「」を参照してください。Amazon RDS データベースインスタンスを起動する。

Microsoft SQL Server データベースを作成するには

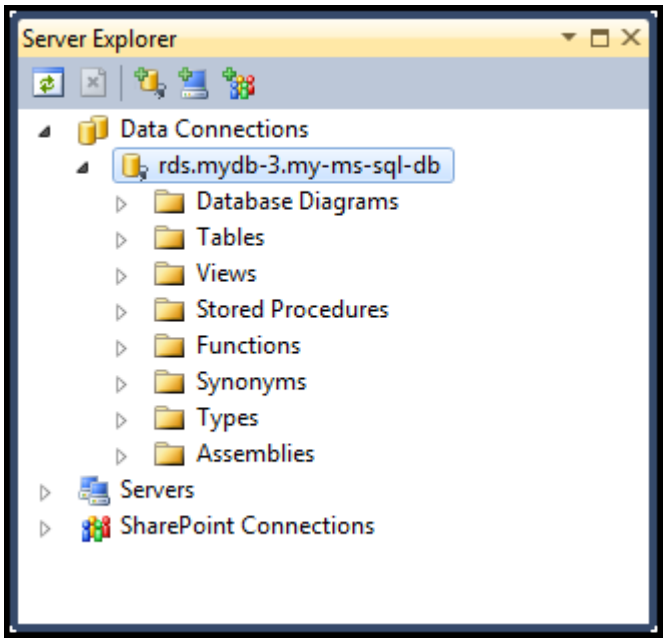
1. EclipseAWSExplorer で、Microsoft SQL Server の RDS インスタンスに対応するノードのコンテキスト (右クリック) メニューを開き、SQL Server データベースの作成。



2. [Create SQL Server Database (SQL サーバーデーターベースの作成)] ダイアログボックスで、RDS インスタンスの作成時に指定したパスワードを入力し、Microsoft SQL Server データベースの名前を入力して、[OK] を選択します。



3. Toolkit for Visual Studio は、Microsoft SQL Server データベースを作成して、Visual Studio の Server Explorer に追加します。



Amazon RDS セキュリティグループ

Amazon RDS セキュリティグループを使用すると、Amazon RDS インスタンスへのネットワークアクセスを管理できます。セキュリティグループを使用して、CIDR 表記を使って IP アドレスのセットを指定すると、Amazon RDS インスタンスは、これらのアドレスから送信されるネットワークトラフィックのみを認識します。

Amazon RDS セキュリティグループは、Amazon EC2 セキュリティグループと類似動作をしますが、両者は異なります。RDS セキュリティグループに EC2 セキュリティグループを追加することができます。すると、EC2 セキュリティグループのメンバーである EC2 インスタンスは、RDS セキュリティグループのメンバーである RDS インスタンスにアクセスできます。

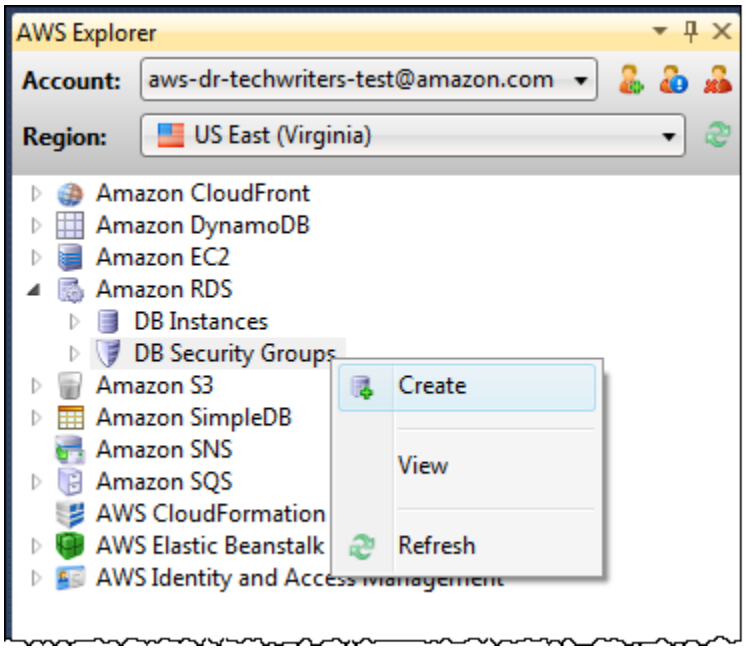
Amazon RDS セキュリティグループの詳細については、以下を参照してください。RDS セキュリティグループ。Amazon EC2 セキュリティグループの詳細については、以下を参照してください。EC2 ユーザーガイド。

Amazon RDS セキュリティグループを作成する

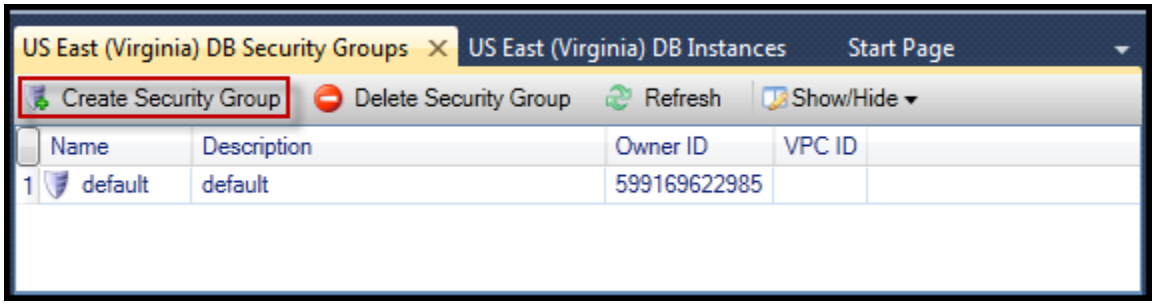
Toolkit for Visual Studio を使用して、RDS セキュリティグループを作成できます。♪AWSToolkit RDS インスタンスを起動するには、ウィザードを使って、インスタンスで使用する RDS セキュリティグループを指定できます。ウィザードを開始する前に、次の手順を使用してそのセキュリティグループを作成することができます。

Amazon RDS セキュリティグループを作成するには

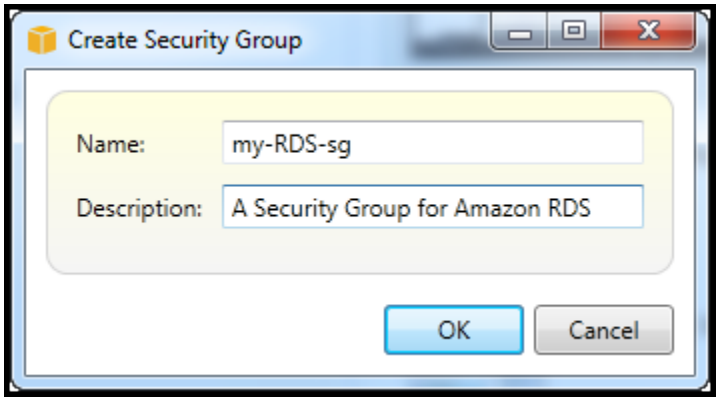
1. EclipseAWSエクスプローラ、Amazon RDSノードを使用して、[] のコンテキスト (右クリック) メニューを開きます。DB セキュリティグループサブノードを選択し、作成。



   または、[Security Groups (セキュリティグループ)] タブで、[Create Security Group (セキュリティグループの作成)] をクリックします。このタブが表示されていない場合は、[DB Security Groups (DB セキュリティグループ)] サブノードのコンテキスト (右クリック) メニューを開き、[View (表示)] を選択します。



2. [Create Security Group (セキュリティグループの作成)] ダイアログボックスで、セキュリティグループ名と説明を入力し、[OK] を選択します。



Amazon RDS セキュリティグループのアクセス許可の設定

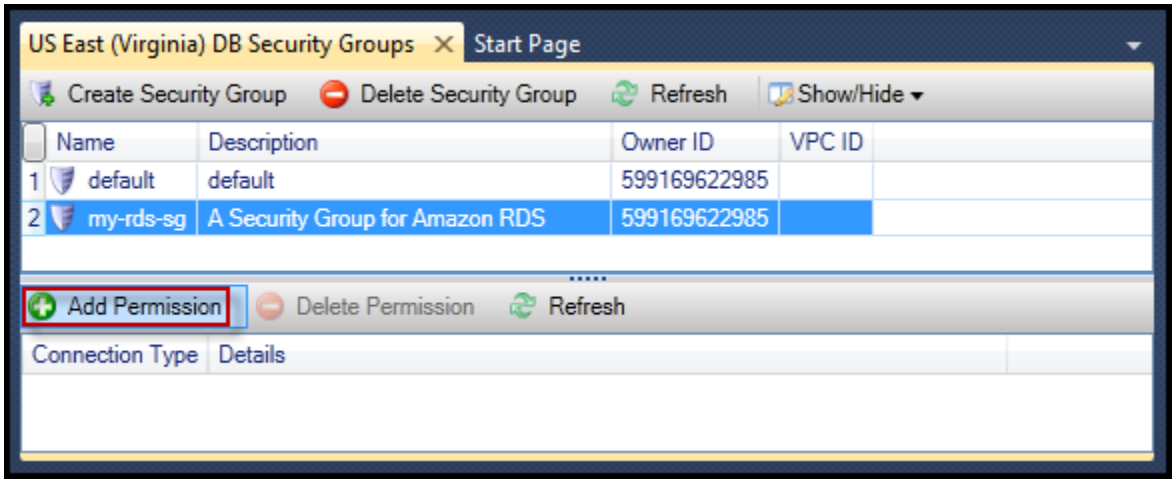
デフォルトでは、新しい Amazon RDS セキュリティグループはネットワークアクセスを提供しません。セキュリティグループを使用する Amazon RDS インスタンスへのアクセスを有効にするには、次の手順を使用してアクセス許可を設定します。

Amazon RDS セキュリティグループへのアクセスを設定するには

1. [Security Groups (セキュリティグループ)] タブで、リストビューからセキュリティグループを選択します。セキュリティグループがリストに表示されない場合は、[Refresh (更新)] を選択します。それでもセキュリティグループがリストに表示されない場合は、正しいリストが表示されていることを確認します。AWS地域。セキュリティグループ[] のタブAWSToolkit はリージョンに固有のものです。

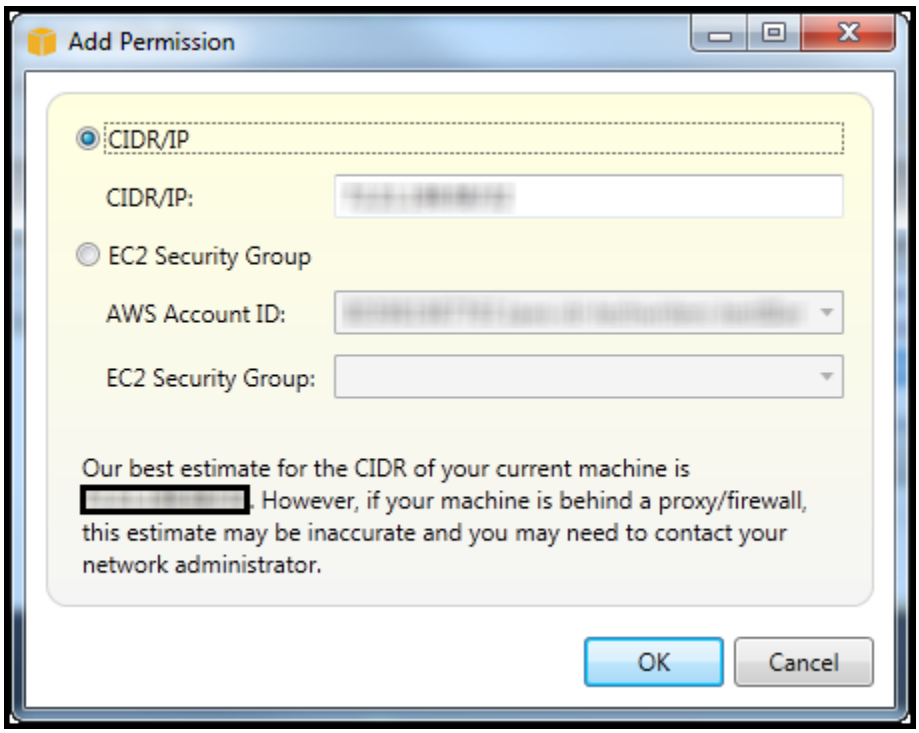
   そうでない場合セキュリティグループタブが表示されます。AWSExplorer を使用して、[] のコンテキスト (右クリック) メニューを開きます。DB セキュリティグループサブノードを選択し、表示。

2. [Add Permission] を選択します。



[Security Groups] タブの [Add Permissions] ボタン

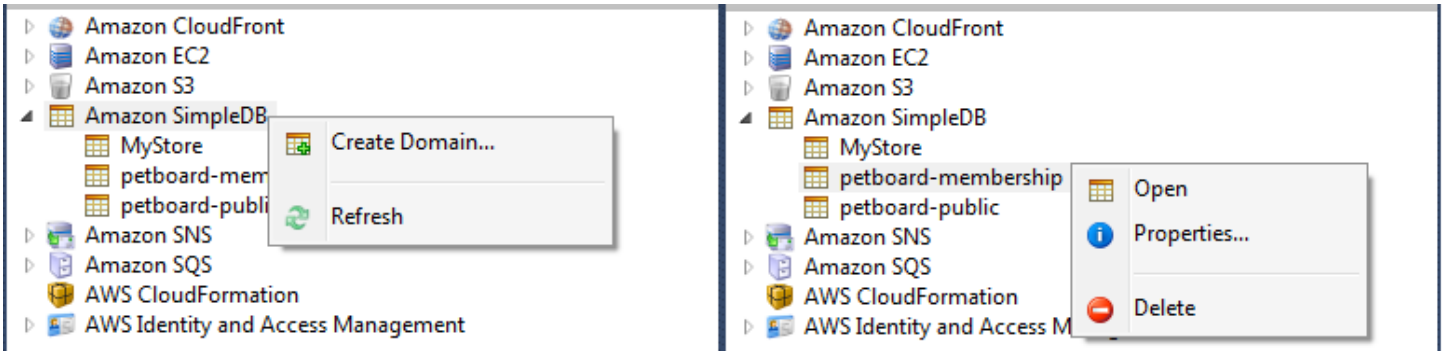
3. [Add Permission (アクセス許可の追加)] ダイアログボックスで、CIDR 表記を使用して、RDS インスタンスにアクセスできる IP アドレスを指定するか、または RDS インスタンスにアクセスできる EC2 セキュリティグループを指定することができます。選択した場合EC2 セキュリティグループでは、に関連するすべての EC2 インスタンスに対するアクセスを指定できます。AWS アカウントアクセス権を持っている、または、ドロップダウンリストから EC2 セキュリティグループを選択することができます。



-AWSToolkit は、IP アドレスを判別して、適切な CIDR 仕様を使ってダイアログボックスを自動入力しようとします。コンピュータがファイアウォール経由でインターネットにアクセスしている場合は、ツールキットにより検出された CIDR が正しくない可能性があります。

Amazon SimpleDB を使用してAWSエクスプローラ

AWSExplorer では、アクティブに関連付けられているすべての Amazon SimpleDB のドメインが表示されます。AWSアカウント. 送信元AWSExplorer で、Amazon SimpleDB のドメインを作成または削除することができます。

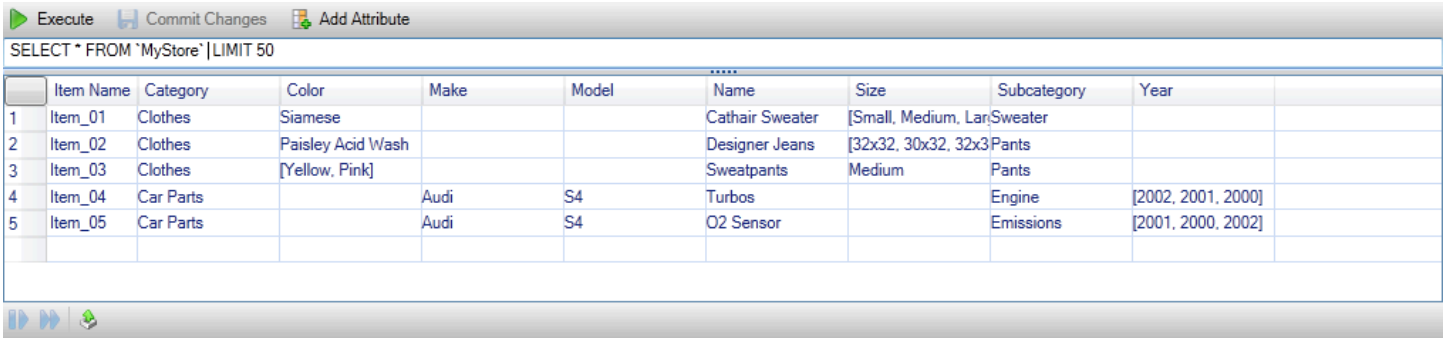


Create, delete, or open Amazon SimpleDB domains associated with your account

クエリの実行と結果の編集

AWSExplorer では、Amazon SimpleDB ドメインのグリッドビューを表示して、ドメインの項目、属性、値を表示することもできます。クエリを実行して、ドメインの項目のサブセットのみを表示することができます。セルをダブルクリックすると、その項目に対応する属性の値を編集できます。ドメインに新しい属性を追加することもできます。

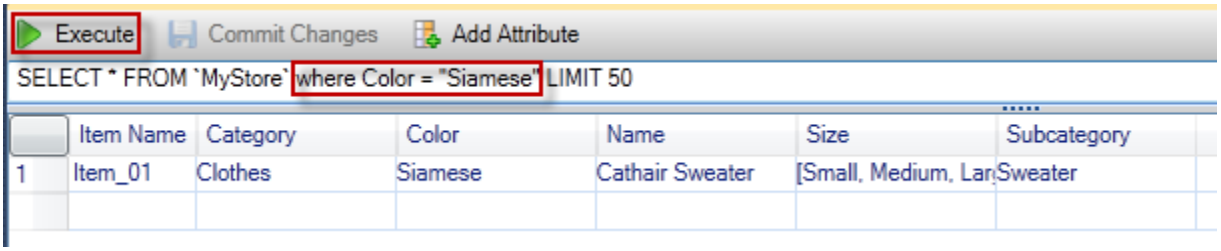
ここに表示されるドメインは、に含まれるAmazon SimpleDB のサンプルのものです。AWS SDK for .NET。



	Item Name	Category	Color	Make	Model	Name	Size	Subcategory	Year
1	Item_01	Clothes	Siamese			Cathair Sweater	[Small, Medium, Lar	Sweater	
2	Item_02	Clothes	Paisley Acid Wash			Designer Jeans	[32x32, 30x32, 32x3	Pants	
3	Item_03	Clothes	[Yellow, Pink]			Sweatpants	Medium	Pants	
4	Item_04	Car Parts		Audi	S4	Turbos		Engine	[2002, 2001, 2000]
5	Item_05	Car Parts		Audi	S4	O2 Sensor		Emissions	[2001, 2000, 2002]

Amazon SimpleDB grid view

クエリを実行するには、グリッドビューの上部にあるテキストボックスでクエリを編集し、[Execute (実行)] を選択します。このビューは、クエリに一致する項目のみを表示するようにフィルタリングされます。

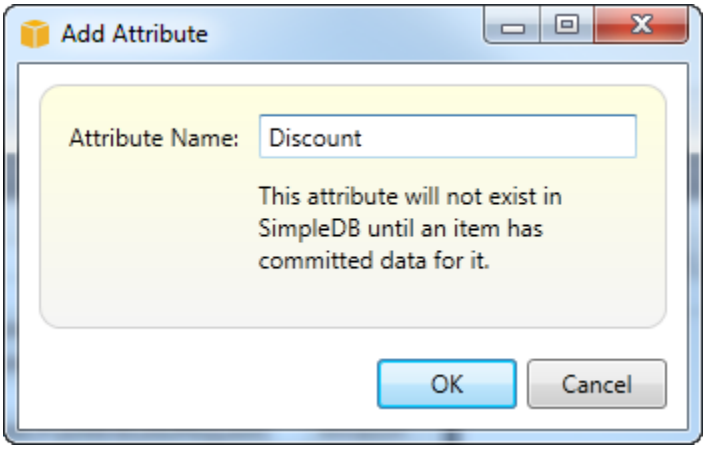


Execute query from AWS Explorer

属性に関連付けられている値を編集するには、対応するセルをダブルクリックし、値を編集して、[Commit Changes (変更をコミット)] を選択します。

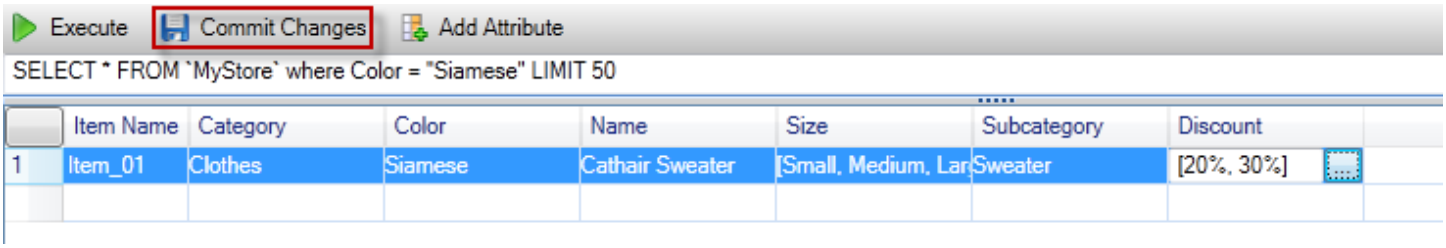
属性の追加

属性を追加するには、ビューの上部で、[Add Attribute (属性の追加)] を選択します。



属性の追加 dialog box

ドメインの属性部分を作成するには、そのための値を少なくとも 1 つの項目に追加して、[Commit Changes (変更をコミット)] ボタンを選択する必要があります。



Commit changes for a new attribute

クエリ結果をページ分割する

ビューの下部には 3 つのボタンがあります。

Paginate and export buttons

最初の 2 つのボタンを使うと、クエリ結果がページ分割されます。最初のボタンでは、結果の次のページが表示されます。2 番目のボタンでは、結果の 10 ページ先が表示されます。この場合では、1 ページは 100 行または LIMIT 値によって指定された行数 (クエリに含まれている場合) となります。

CSV へエクスポート

最後のボタンを使うと、現在の結果が CSV ファイルにエクスポートされます。

Amazon SQS の使用AWSエクスプローラ

Amazon Simple Queue Service (Amazon SQS) は柔軟なキューサービスであり、ソフトウェアアプリケーションで実行されている異なるプロセス間でメッセージの受け渡しを行うことができます。Amazon SQS キューは、AWSインフラストラクチャですが、メッセージを受け渡すプロセスはローカル、Amazon EC2 インスタンス、またはそれらを組み合わせて配置できます。Amazon SQS は複数のコンピュータ間での分散作業を調整するために最適です。

Toolkit for Visual Studio を使用すると、アクティブなアカウントに関連付けられている Amazon SQS キューの表示、キューの作成と削除、キューを使ったメッセージの送信を行うことができます。(アクティブなアカウントとは、[] で選択されたアカウントを意味します。AWSExplorer。)

Amazon SQS の詳細については、「」を参照してください。SQS の紹介のAWSドキュメント内) を参照してください。

キューの作成

Amazon SQS キューは以下から作成できます。AWSExplorer。キューの ARN と URL は、アクティブなアカウントのアカウント番号と、作成時に指定したキューの名前に基づきます。

キューを作成するには

1. EclipseAWSExplorer で、[] のコンテキスト (右クリック) メニューを開きますAmazon SQSノードを選択し、キューの作成。

2. [Create Queue (キューの作成)] ダイアログボックスで、キュー名、デフォルトの可視性タイムアウト、およびデフォルト配信遅延を指定します。デフォルトの可視性タイムアウトとデフォルトの配信遅延は秒単位で指定します。デフォルトの可視性タイムアウトは、あるプロセスがメッセージを取得した後に、潜在的な受信プロセスに対してそのメッセージが不可視になる時間です。デフォルトの配信遅延は、メッセージが送信されてから、潜在的な受信プロセスに対して可視となるまでの時間です。
3. [OK] をクリックします。新しいキューは、[Amazon SQS] ノードの下のサブノードとして表示されます。

キューの削除

から既存のキューを削除することができますAWSExplorer。キューを削除すると、キューに関連付けられているメッセージは利用できなくなります。

キューを削除するには

1. EclipseAWSExplorer で、削除するキューのコンテキスト (右クリック) メニューを開き、[] を選択します。削除。

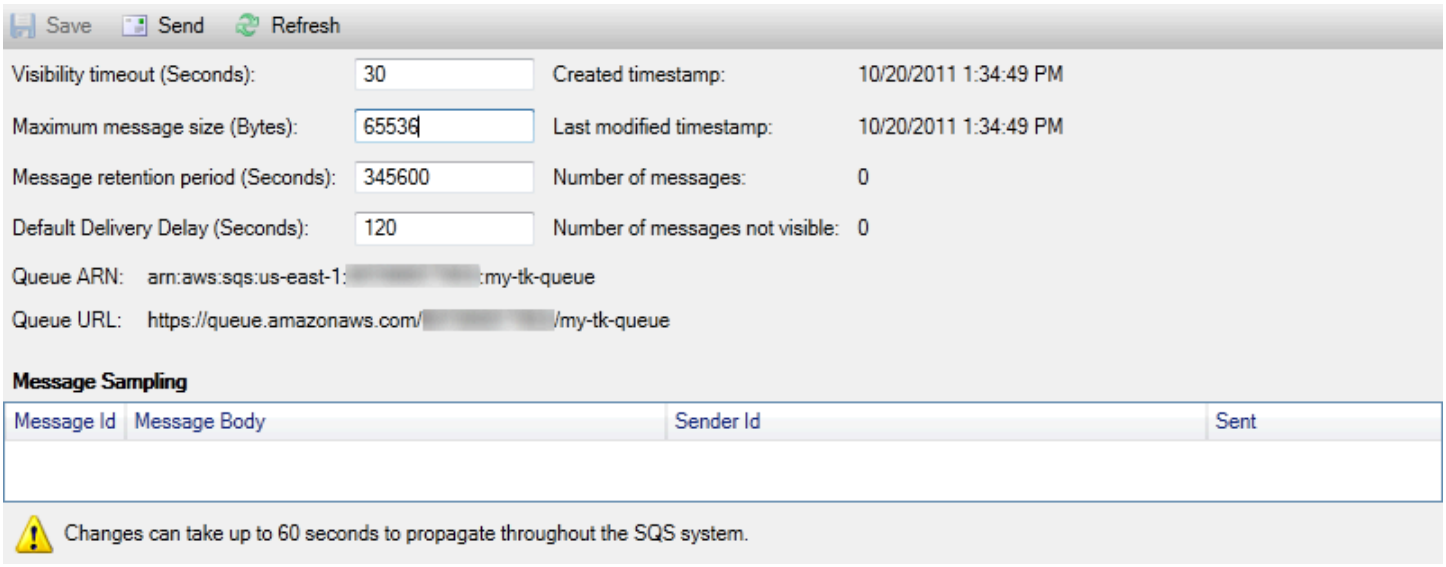
キューのプロパティの管理

に表示されているすべてのキューのプロパティを表示および編集できますAWSExplorer。このプロパティビューから、キューにメッセージを送信することもできます。

キューのプロパティを管理するには

- EclipseAWSExplorer で、プロパティを管理するキューのコンテキスト (右クリック) メニューを開き、[] を選択します。キューの表示。

  キューのプロパティビューから、可視性タイムアウト、最大メッセージサイズ、メッセージの保持期間、デフォルトの配信遅延を編集できます。デフォルトの配信遅延は、メッセージの送信時に上書きできます。次のスクリーンショットでは、キューの ARN と URL に含まれるアカウント番号は非表示にしています。



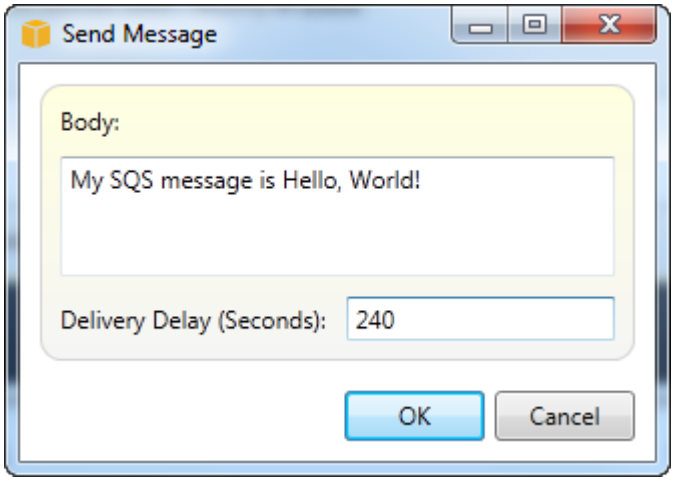
SQS queue properties view

キューへのメッセージ送信

キューのプロパティビューから、キューにメッセージを送信できます。

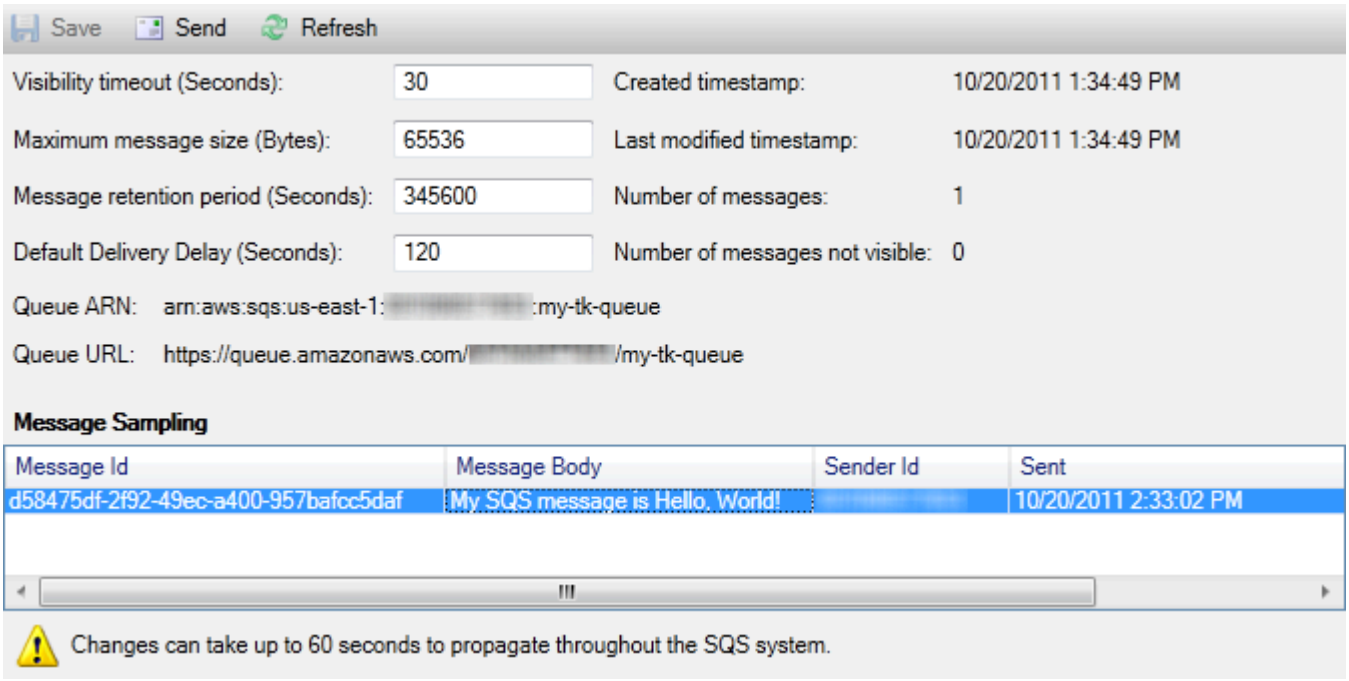
メッセージを送信するには

1. キューのプロパティビューの上部で、[Send (送信)] ボタンを選択します。
2. メッセージを入力します。(オプション) 配信遅延を入力すると、キューのデフォルトの配信遅延を上書きします。次の例では、遅延の値を 240 秒で上書きします。[OK] をクリックします。



メッセージの送信 dialog box

3. 約 240 秒 (4 分) 待ちます。メッセージが、キューのプロパティビューの [Message Sampling (メッセージの送信)] セクションに表示されます。



SQS properties view with sent message

キューのプロパティビューのタイムスタンプは、[Send (送信)] ボタンを選択した時間です。これには遅延は含まれません。したがって、メッセージがキューに表示されてレシーバーに利用可能になる時間は、このタイムスタンプよりも後になる場合があります。タイムスタンプは、コンピュータの現地時間で表示されます。

Identity and Access Management

AWS Identity and Access Management(IAM) を利用すると、へのアクセスを、より安全に管理することができますAWS アカウントとリソース。IAM を使用すると、プライマリに複数のユーザーを作成できます (ルート)AWS アカウント。これらのユーザーは自分の認証情報 (パスワード、アクセスキー ID、およびシークレットキー) を持つことができますが、すべての IAM ユーザーは 1 つのアカウント番号を共有します。

ユーザーに IAM ポリシーをアタッチして、各 IAM ユーザーのリソースアクセスのレベルを管理することができます。たとえば、Amazon S3 サービスおよびアカウントの関連リソースへのユーザーアクセスを付与するポリシーを IAM ユーザーにアタッチすることができますが、他のサービスまたはリソースへのアクセスはできません。

より効率的なアクセス管理を行うためには、ユーザーの集合である、IAM グループを作成できます。グループにポリシーをアタッチすると、そのグループのメンバーであるすべてのユーザーに反映されます。

IAM は、ユーザーおよびグループレベルでのアクセス許可の管理に加えて、IAM ロールの概念もサポートしています。ユーザーおよびグループと同様に、IAM ロールにポリシーをアタッチすることができます。その後、IAM ロールを Amazon EC2 インスタンスに関連付けることができます。EC2 インスタンス上で実行されるアプリケーションは、アクセスすることができますAWSIAM ロールが提供するアクセス権限を使用します。ツールキットでの IAM ロールの使用の詳細については、「IAM ロールの作成」を参照してください。IAM の詳細については、を参照してくださいIAM ユーザーガイド。

IAM ユーザーを作成して設定する

IAM ユーザーを使うと、他のユーザーにAWS アカウント。IAM ユーザーにポリシーをアタッチすることができるため、IAM ユーザーがアクセスできるリソースおよびそれらのリソースに実行できる操作を正確に制限することができます。

ベストプラクティスとして、にアクセスするすべてのユーザーがAWS アカウントIAM ユーザーとしてアクセスする必要があります。アカウントの所有者についても同様です。これにより、IAM ユーザーのうちの 1 人の認証情報が漏洩した場合でも、それらの認証情報のみを非アクティブ化することができます。アカウントのルート認証情報を非アクティブ化または変更する必要はありません。

Toolkit for Visual Studio では、ユーザーにアクセス許可を割り当てるには、ユーザーに IAM ポリシーをアタッチするか、ユーザーをグループに割り当てます。グループに割り当てた IAM ユーザーは、グループにアタッチされているポリシーからアクセス許可を引き継ぎます。詳細については、「IAM グループを作成する」と「IAM グループに IAM ユーザーを追加する」を参照してください。

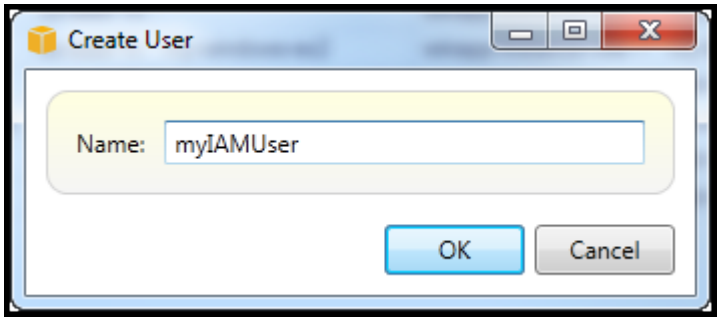
Toolkit for Visual Studio から、生成することもできますAWSIAM ユーザーの認証情報 (アクセスキー ID とシークレットキー) 詳細については、「IAM ユーザーの認証情報を生成する」を参照してください。

Toolkit for Visual Studio では、を使用してサービスにアクセスするための IAM ユーザー認証情報の指定がサポートされています。AWSExplorer IAM ユーザーは通常、すべてのAmazon Web Services へのフルアクセスができないため、AWSExplorer は使用できない可能性があります。!AWSアクティブアカウントが IAM ユーザーであるときに Explorer を使用してリソースを変更して、次にアクティブアカウントをルートアカウントに切り替える場合には、で表示を更新するまで変更が表示されない場合があります。AWSExplorer 表示を更新するには、[Refresh (更新)] ボタンを選択します。

から IAM ユーザーを設定する方法については、を参照してくださいAWS Management Console[]、[]に移動しますユーザーおよびグループの使用IAM ユーザーガイド

IAM ユーザーを作成するには

1. EclipseAWSエクスプローラ、AWS Identity and Access Management[] ノードで、[] のコンテキスト (右クリック) メニューを開きますユーザー[] を選択して [] を選択しますユーザーの作成。
2. 左ユーザーの作成ダイアログボックスで、IAM ユーザーの名前を入力して [] を選択します。OK。これがIAMフレンドリネーム。IAM ユーザー名の制約については、「IAM ユーザーガイド。



Create an IAM user

新しいユーザーは [] の下にサブノードとして表示されます。ユーザーでAWS Identity and Access Managementノード。

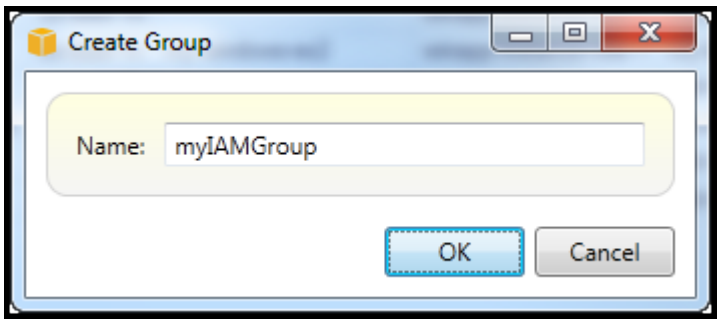
ポリシーを作成してユーザーにアタッチする方法については、「IAM ポリシーを作成する」を参照してください。

IAM グループを作成する

グループは、IAM ポリシーをユーザーの集合に適用する方法を提供します。IAM ユーザーおよびグループを管理する方法の詳細については、「」を参照してください。ユーザーおよびグループの使用IAM ユーザーガイド

IAM グループを作成するには

1. EclipseAWSExplorerIdentity and Access Management[] で、[] のコンテキスト (右クリック) メニューを開きますGroupsを選択しグループの作成。
2. 左グループの作成ダイアログボックスで、IAM グループの名前を入力して [] を選択します。OK。



Create IAM group

新しい IAM グループがGroupsのサブノードIdentity and Access Management。

ポリシーを作成して IAM グループにアタッチする方法については、「」を参照してください。IAM ポリシーの作成。

IAM グループに IAM ユーザーを追加する

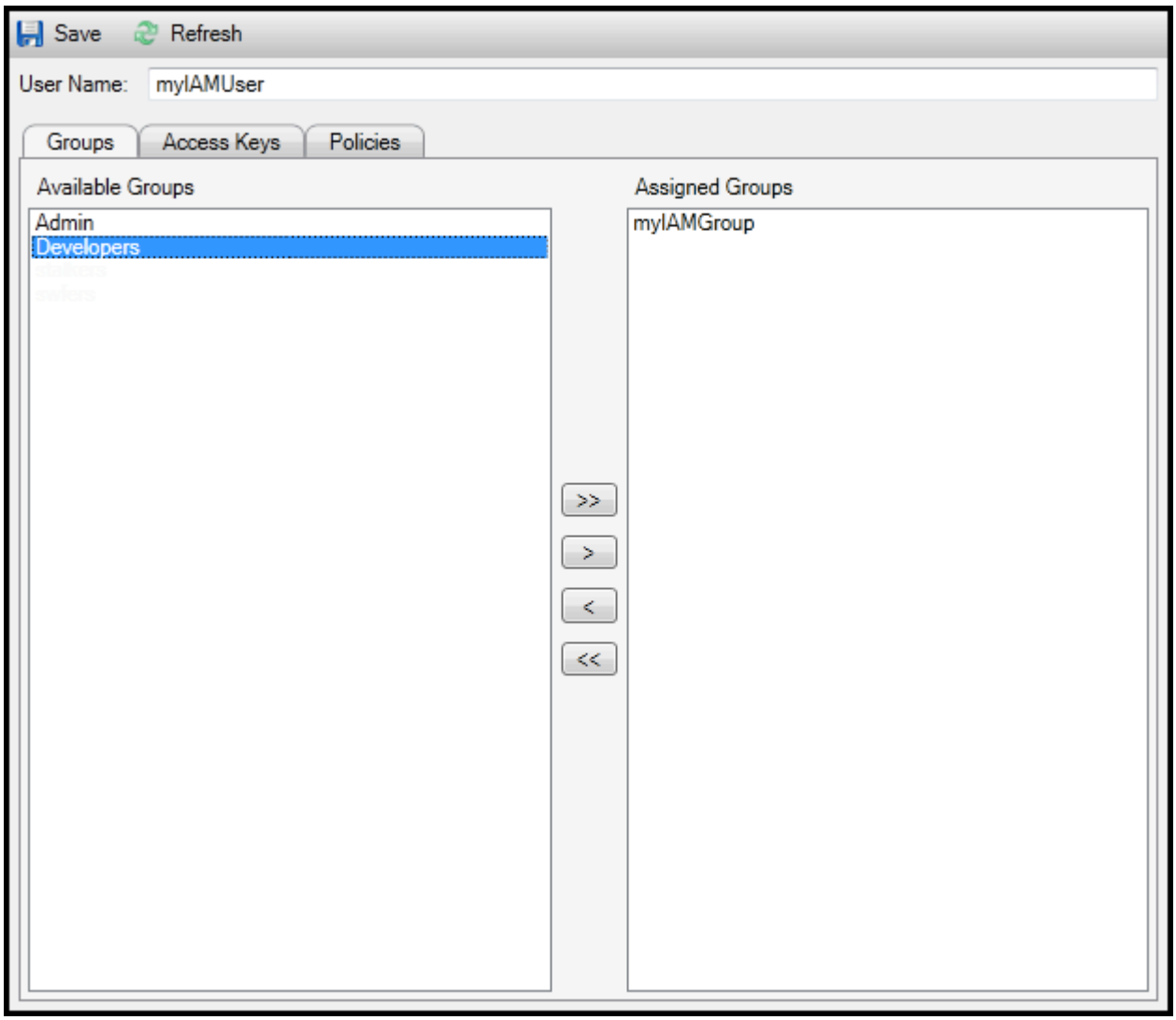
IAM グループのメンバーである IAM ユーザーは、グループにアタッチされているポリシーからアクセス許可を引き継ぎます。IAM グループの目的は、IAM ユーザーの集合全体のアクセス許可の管理を容易にすることです。

IAM グループにアタッチされたポリシーが、その IAM グループのメンバーである IAM ユーザーにアタッチされたポリシーと連携する方法の詳細については、「」を参照してください。IAM ユーザーガイドの IAM ポリシーの管理。

EclipseAWSエクスプローラーでは、IAM ユーザーを IAM グループに追加するにはユーザーサブノードではなくGroupsサブノード。

IAM グループに IAM ユーザーを追加するには

1. EclipseAWSExplorerIdentity and Access Management[] で、[] のコンテキスト (右クリック) メニューを開きますユーザーを選択し編集。



Assign an IAM user to a IAM group

2. [] の左ペインGroupsタブには、使用可能な IAM グループが表示されます。右側のペインには、指定された IAM ユーザーがすでにメンバーであるグループが表示されます。

   IAM ユーザーをグループに追加するには、左側のペインで、IAM グループを選択し、>[] ボタン。

   グループから IAM ユーザーを削除するには、右側のペインで、IAM グループを選択し、<[] ボタン。

   IAM ユーザーをすべての IAM グループに追加するには、>>[] ボタン。同様に、IAM ユーザーをすべてのグループから削除するには、[<<[] ボタン。

   複数のグループを選択するには、それらを順に選択します。コントロールキーを押しながら選択する必要はありません。グループを選択からクリアするには、再度選択します。

3. IAM ユーザーの IAM グループへの割り当てが完了したら、[] を選択します。保存。

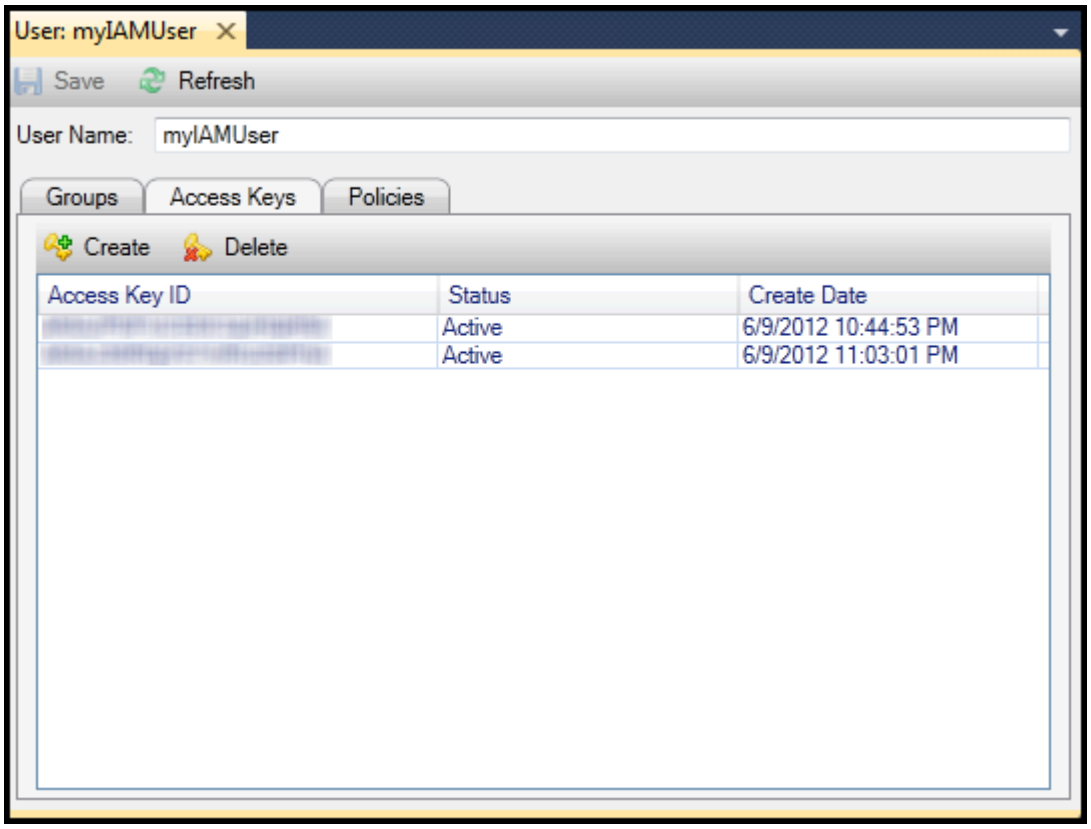
IAM ユーザーの認証情報を生成する

Toolkit for Visual Studio を使用すると、への API 呼び出しを実行するために使用する、アクセスキー ID とシークレットキーを生成することができます。AWS。これらのキーは、ツールキットを使って Amazon Web Services にアクセスするために指定することもできます。Toolkit 用に認証情報を指定する方法については、「認証情報」を参照してください。認証情報の安全な処理の詳細については、「」を参照してください。管理のベストプラクティスAWSアクセスキー。

ツールキットは、IAM ユーザーのパスワードを生成するために使用することはできません。

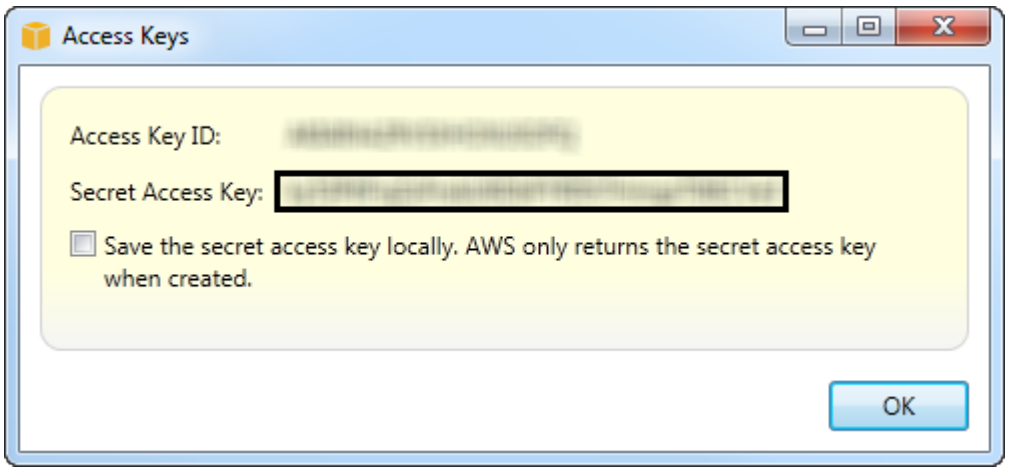
IAM ユーザーの認証情報を生成するには

1. EclipseAWSExplorer で IAM ユーザーのコンテキスト (右クリック) メニューを開き、[編集。



2. 認証情報を生成するには、[Access Keys (アクセスキー)] タブで、[Create (作成)] を選択します。

   IAM ユーザーごとに 2 セットのみの認証情報を生成できます。2 セットの認証情報がすでにあり、追加のセットを作成する必要がある場合は、既存のセットのいずれかを削除する必要があります。

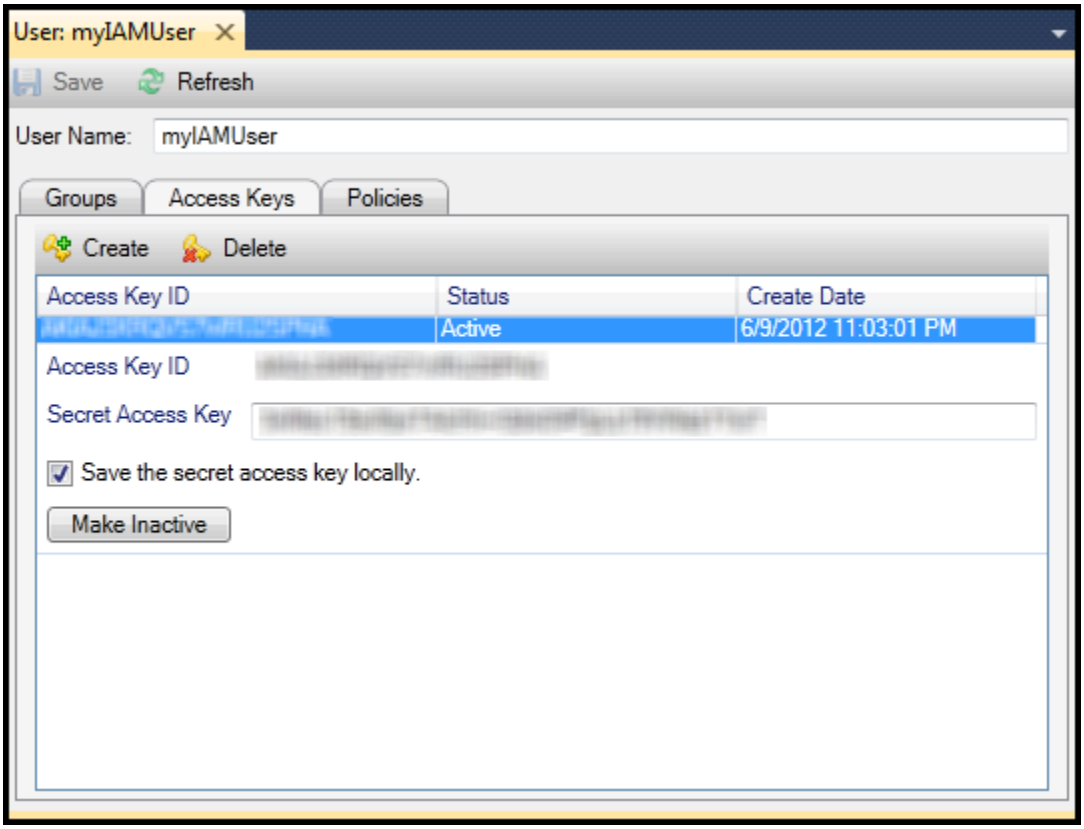


reate credentials for IAM user

ツールキットでシークレットアクセスキーの暗号化済みコピーをローカルドライブに保存する場合は、シークレットアクセスキーをローカルに保存します。AWS作成されたときにシークレットアクセスキーのみを返します。ダイアログボックスからシークレットアクセスキーをコピーして、それを安全な場所に保存することもできます。

3. [OK] をクリックします。

認証情報を生成した後、それを [Access Keys (アクセスキー)] タブから表示できます。ツールキットでシークレットキーをローカルに保存するオプションを選択した場合は、ここに表示されます。



Create credentials for IAM user

自分でシークレットキーを保存した場合で、ツールキットでもそれを保存する場合は、[Secret Access Key (シークレットアクセスキー)] ボックスにシークレットアクセスキーを入力し、[Save the secret access key locally (シークレットアクセスキーをローカルに保存)] を選択します。

認証情報を無効化するには、[Make Inactive (無効化)] を選択します。(認証情報の漏洩が疑われる場合は、これを実行します。) 認証情報が安全であることが確認できたら、再アクティブ化することができます。

IAM ロールを作成します。

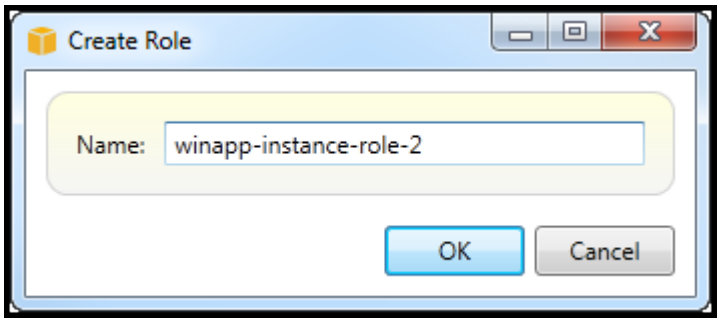
Toolkit for Visual Studio では、IAM ロールの作成および設定をサポートしています。ユーザーおよびグループと同様に、IAM ロールにポリシーをアタッチすることができます。その後、IAM ロールを Amazon EC2 インスタンスに関連付けることができます。EC2 インスタンスとの関連付けは、インスタンスプロファイルを介して処理されます。これはロールの論理コンテナです。EC2 インスタンスで実行されるアプリケーションは、IAM ロールに関連付けられたポリシーによって指定されるアクセスのレベルを自動的に付与されます。これは、アプリケーションが他を指定していない場合にもあてはまります。AWS認証情報。

たとえば、ロールを作成して、Amazon S3 のみへのアクセスに制限するポリシーをそのロールにアタッチできます。このロールを EC2 インスタンスに関連付けた後、そのインスタンスでアプリケーションを実行すると、アプリケーションは Amazon S3 にアクセスできますが、他のサービスまたはリソースにはアクセスできません。この方法の利点は、安全に転送して保存することを心配する必要がないことです。AWSEC2 インスタンスの認証情報。

IAM ロールの詳細については、「」を参照してください。IAM ユーザーガイドの IAM ロールの使用。アクセスするプログラムの例AWSAmazon EC2 インスタンスに関連付けた IAM ロールを使用して、AWS開発者ガイドJava,.NET,PHP、Ruby (IAM を使用して認証情報を設定する,IAM ロールの作成, およびIAM ポリシーの使用).

IAM ロールを作成するには

1. EclipseAWSExplorerIdentity and Access Management[] で、[] のコンテキスト (右クリック) メニューを開きますロール[] を選択して [] を選択しますロールの作成。
2. 左ロールの作成ダイアログボックスで、IAM ロールの名前を入力して [] を選択します。OK。



Create IAM role

新しい IAM ロールは、ロールにIdentity and Access Management。

ポリシーを作成してロールにアタッチする方法については、「IAM ポリシーを作成する」を参照してください。

IAM ポリシーの作成

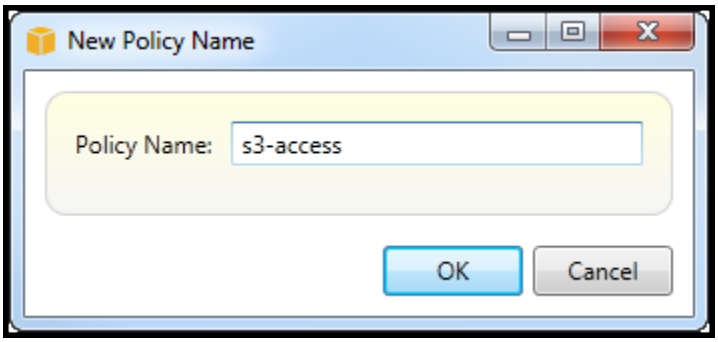
IAM にとってポリシーが重要です。ポリシーを IAM に関連付けることができますエンティティユーザー、グループ、ロールなどがあります。ポリシーによって、ユーザー、グループ、またはロールに対して有効にする、アクセスのレベルを指定できます。

IAM ポリシーを作成するには

EclipseAWSエクスプローラ、AWS Identity and Access Managementノードを選択し、エンティティのタイプ () のノードを展開します。Groups,ロール, またはユーザー) にポリシーを添付します。たとえば、IAM ロールのコンテキストメニューを開き、[] を選択します。編集。

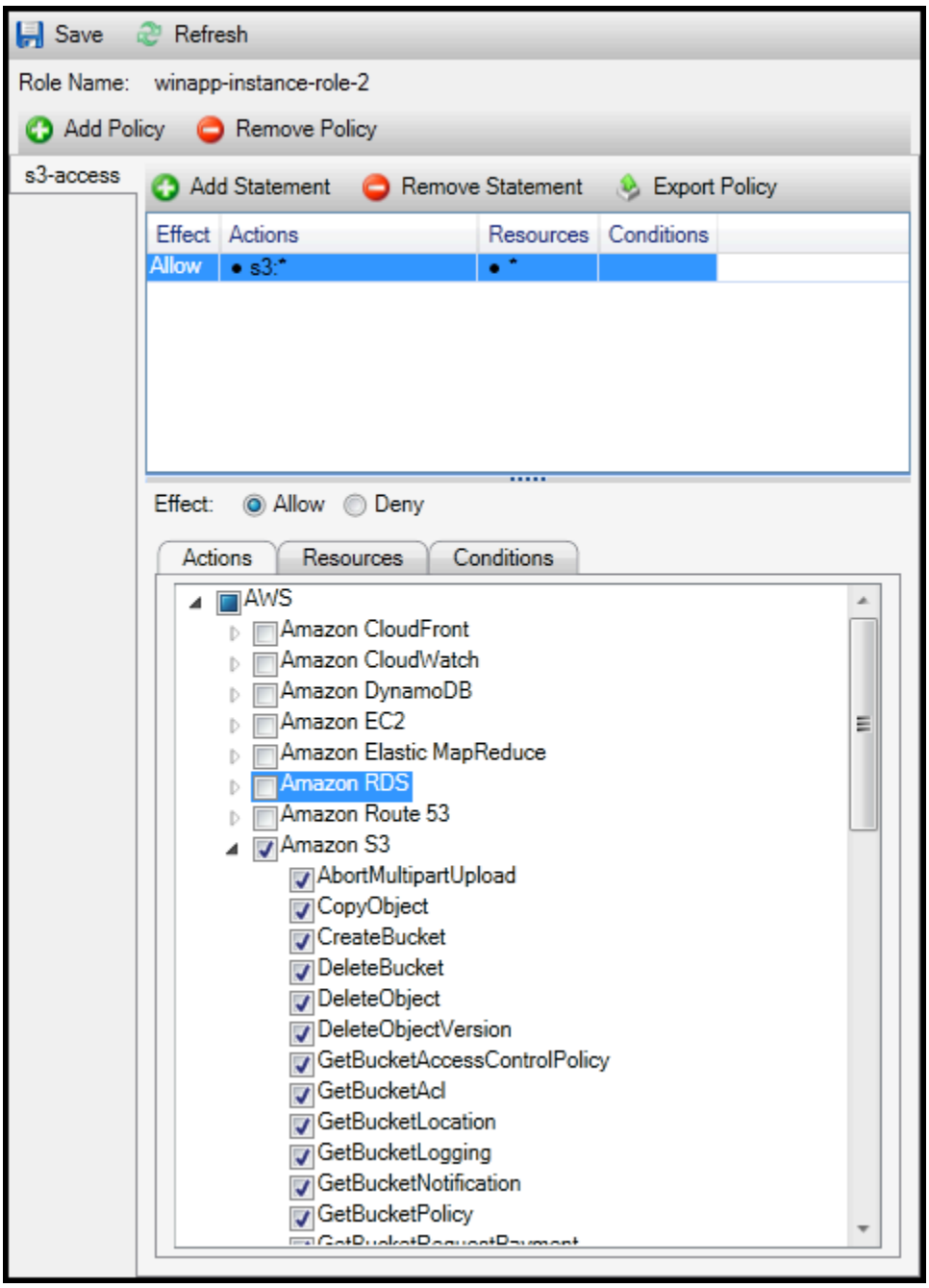
ロールに関連付けられたタブが [AWSExplorer [Add Policy (ポリシーの追加)] リンクを選択します。

[New Policy Name (新しいポリシー名)] ダイアログボックスで、ポリシーの名前 (たとえば s3-access) を入力します。



New Policy Name dialog box

ポリシーエディタで、ポリシーステートメントを追加して、ロールに提供するアクセスレベルを指定します (この例では、ポリシーに関連付けられている winapp-instance-role-2 です)。この例では、ポリシーは Amazon S3 へのフルアクセスを提供しますが、他のリソースへのアクセスはできません。



Specify IAM policy

より正確なアクセス制御には、ポリシーエディタでサブノードを展開して、Amazon Web Services に関連付けられたアクションを許可または拒否します。

ポリシーを編集したら、[Save (保存)] リンクを選択します。

AWS Lambda

を使用して、.NET Core ベースの C# Lambda 関数を開発してデプロイします。 AWS Toolkit for Visual Studio AWS Lambda は、サーバーのプロビジョニングや管理を行わずにコードを実行できるコンピューティングサービスです。Visual Studio Toolkit for Visual Studio 用の AWS Lambda .NET Core プロジェクトテンプレートが含まれています。

詳細については AWS Lambda、『AWS Lambda 開発者ガイド』を参照してください。

.NET Core の詳細については、『Microsoft .NET Core ガイド』を参照してください。Windows、macOS、Linux の各プラットフォームでの、.NET Core の前提条件とインストール方法については、「.NET Core のダウンロード」を参照してください。

以下のトピックでは、Toolkit for Visual Studio AWS Lambda を使用して作業する方法について説明します。

トピック

- 基本 AWS Lambda プロジェクト
- 基本 AWS Lambda Docker イメージの作成プロジェクト
- チュートリアル:でサーバーレスアプリケーションをビルドしてテストする AWS Lambda
- チュートリアル: Amazon Rekognition Lambda アプリケーションの作成
- チュートリアル:Amazon AWS Lambda ロギングフレームワークを使用したアプリケーションログの作成

基本 AWS Lambda プロジェクト

の Microsoft .NET Core プロジェクトテンプレートを使用して Lambda 関数を作成できます。 AWS Toolkit for Visual Studio

Visual Studio .NET Core Lambda プロジェクトを作成する

Lambda-Visual Studio のテンプレートとブループリントを使用すると、プロジェクトの初期化をスピードアップできます。Lambda ブループリントには、柔軟なプロジェクト基盤を簡単に作成できるようにあらかじめ記述された関数が含まれています。

> **Note**
>
> Lambda サービスには、さまざまなパッケージタイプにデータ制限があります。データ制限の詳細については、『Lambda ユーザーガイド』の「Lambda クォータ」トピックを参照してください。AWS

Visual Studio で Lambda プロジェクトを作成するには

1. Visual Studio から [ファイル] メニューを展開し、[新規] を展開して [プロジェクト] を選択します。
2. 「新規プロジェクト」ダイアログボックスで、「言語」、「プラットフォーム」、および「プロジェクトタイプ」ドロップダウンボックスを「すべて」に設定し、「検索」フィールドに入力します。aws lambdaAWS Lambda プロジェクト (.NET コア-C#) テンプレートを選択します。
3. [名前] フィールドに**`AWSLambdaSample`**、目的のファイルの場所を入力して指定し、[作成] を選択して続行します。
4. 「ブループリントの選択」 ページから「Empty Function」ブループリントを選択し、「完了」を選択して Visual Studio プロジェクトを作成します。

プロジェクトファイルを確認する

見直すべきプロジェクトファイルとして、`aws-lambda-tools-defaults.json` と `Function.cs` の 2 つがあります。

次の例は、`aws-lambda-tools-defaults.json`プロジェクトの一部として自動的に作成されるファイルを示しています。このファイルのフィールドを使用してビルド· オプションを設定できます。

> **Note**
>
> Visual Studio のプロジェクトテンプレートにはさまざまなフィールドが含まれています。次の点に注意してください。
>
> - 関数ハンドラー:Lambda 関数の実行時に実行されるメソッドを指定します
> - 関数ハンドラーフィールドに値を指定すると、その値がパブリッシュウィザードに事前入力されます。

- 関数、クラス、またはアセンブリの名前を変更する場合は、ファイル内の対応するフィールドも更新する必要があります。aws-lambda-tools-defaults.json

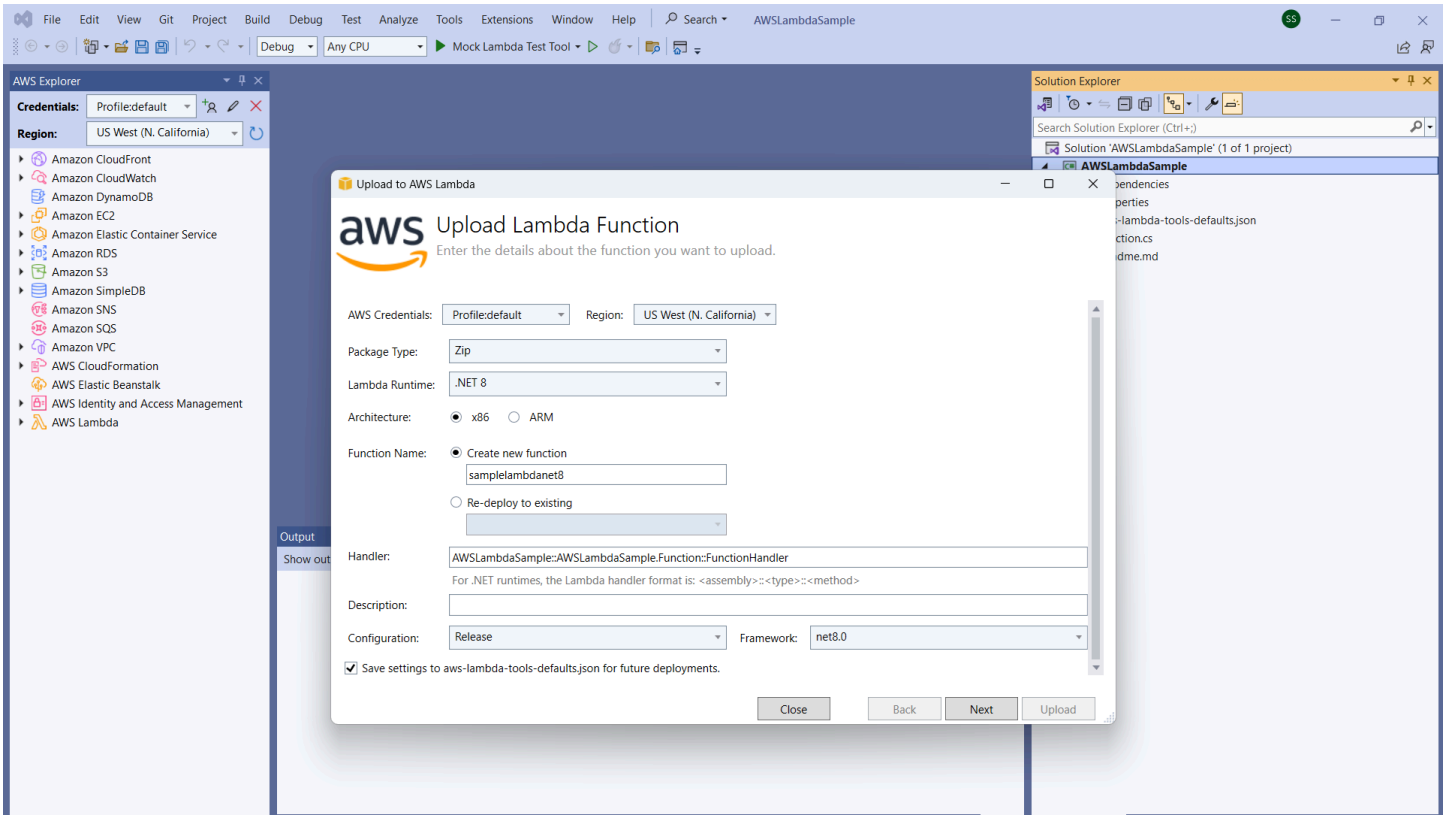
```
{
  "Information": [
    "This file provides default values for the deployment wizard inside Visual Studio
 and the AWS Lambda commands added to the .NET Core CLI.",
    "To learn more about the Lambda commands with the .NET Core CLI execute the
 following command at the command line in the project root directory.",
    "dotnet lambda help",
    "All the command line options for the Lambda command can be specified in this
 file."
  ],
  "profile": "default",
  "region": "us-west-2",
  "configuration": "Release",
  "function-architecture": "x86_64",
  "function-runtime": "dotnet8",
  "function-memory-size": 512,
  "function-timeout": 30,
  "function-handler": "AWSLambdaSample::AWSLambdaSample.Function::FunctionHandler"
}
```

Function.cs ファイルを調べます。Function.cs は、c# 関数を Lambda 関数として公開できるように定義します。このFunctionHandlerは、Lambda 関数の実行時に実行される Lambda 関数です。このプロジェクトでは、FunctionHandler という 1 つの関数が定義され、入力テキストに ToUpper() を呼び出します。

これでプロジェクトを Lambda に発行できるようになりました。

Lambda へのパブリッシング

以下の手順と画像は、を使用して Lambda に関数をアップロードする方法を示しています。 AWS Toolkit for Visual Studio



関数を Lambda にパブリッシュする

1. [View] を展開して [AWS Explorer] を選択しAWS 、エクスプローラーに移動します。
2. ソリューションエクスプローラーで、公開するプロジェクトのコンテキストメニューを開き (右クリック)、[Lambda に公開] を選択して [AWS Lambda 関数のアップロード] ウィンドウを開きます。
3. [Lambda 関数のアップロード] ウィンドウから、以下のフィールドを入力します。

    a. Package タイプ:**Zip**選択してください。ZIP ファイルはビルドプロセスの結果として作成され、Lambda にアップロードされます。または、Package タイプを選択することもできます**Image**。「チュートリアル:基本的な Lambda プロジェクト Docker イメージの作成」では、Package タイプを使用して公開する方法について説明しています。 **Image**

    b. Lambda ランタイム:ドロップダウンメニューから Lambda ランタイムを選択します。

    c. アーキテクチャ:好みのアーキテクチャの放射状を選択します。

    d. 関数名:[新しい関数の作成] のラジアルを選択し、Lambda インスタンスの表示名を入力します。 AWS この名前はエクスプローラーとディスプレイの両方で参照されます。 AWS Management Console

e. ハンドラー:このフィールドを使用して関数ハンドラーを指定します。例: **`AWSLambdaSample::AWSLambdaSample.Function::FunctionHandler`**。

f. (オプション) 説明: AWS Management Consoleインスタンスに表示する説明文を内部から入力します。

g. 設定:ドロップダウンメニューから希望の設定を選択します。

h. フレームワーク:ドロップダウンメニューから好みのフレームワークを選択します。

i. 設定を保存:現在の設定をfuture `aws-lambda-tools-defaults.json` のデフォルトとして保存するには、このボックスを選択します。

j. [次へ] を選択して [拡張機能の詳細] ウィンドウに進みます。

4. 「拡張機能の詳細」ウィンドウで、以下のフィールドに入力します。

a. ロール名:アカウントに関連付けられているロールを選択します。このロールは、 AWS 関数内のコードによって行われるすべてのサービスコールの一時的な認証情報を提供します。ロールがない場合は、スクロールしてドロップダウンセレクターの「 AWS 管理ポリシーに基づく新規ロール」を探し、を選択しますAWSLambdaBasicExecutionRole。このロールには最小限のアクセス権限しかありません。

> **Note**
>
> アカウントには IAM ListPolicies アクションを実行する権限が必要です。そうしないと、Role Name リストが空になり、続行できなくなります。

b. (オプション) Lambda 関数が Amazon VPC 上のリソースにアクセスする場合は、サブネットとセキュリティグループを選択します。

c. (オプション) Lambda 関数に必要な環境変数を設定します。キーは、無料のデフォルトのサービスキーによって自動的に暗号化されます。別の方法として、 AWS KMS 有料のキーを指定することもできます。KMS は、データの暗号化に使用される暗号化キーの作成と管理を行うために使用できる、マネージド型サービスです。 AWS KMS キーがある場合は、リストから選択できます。

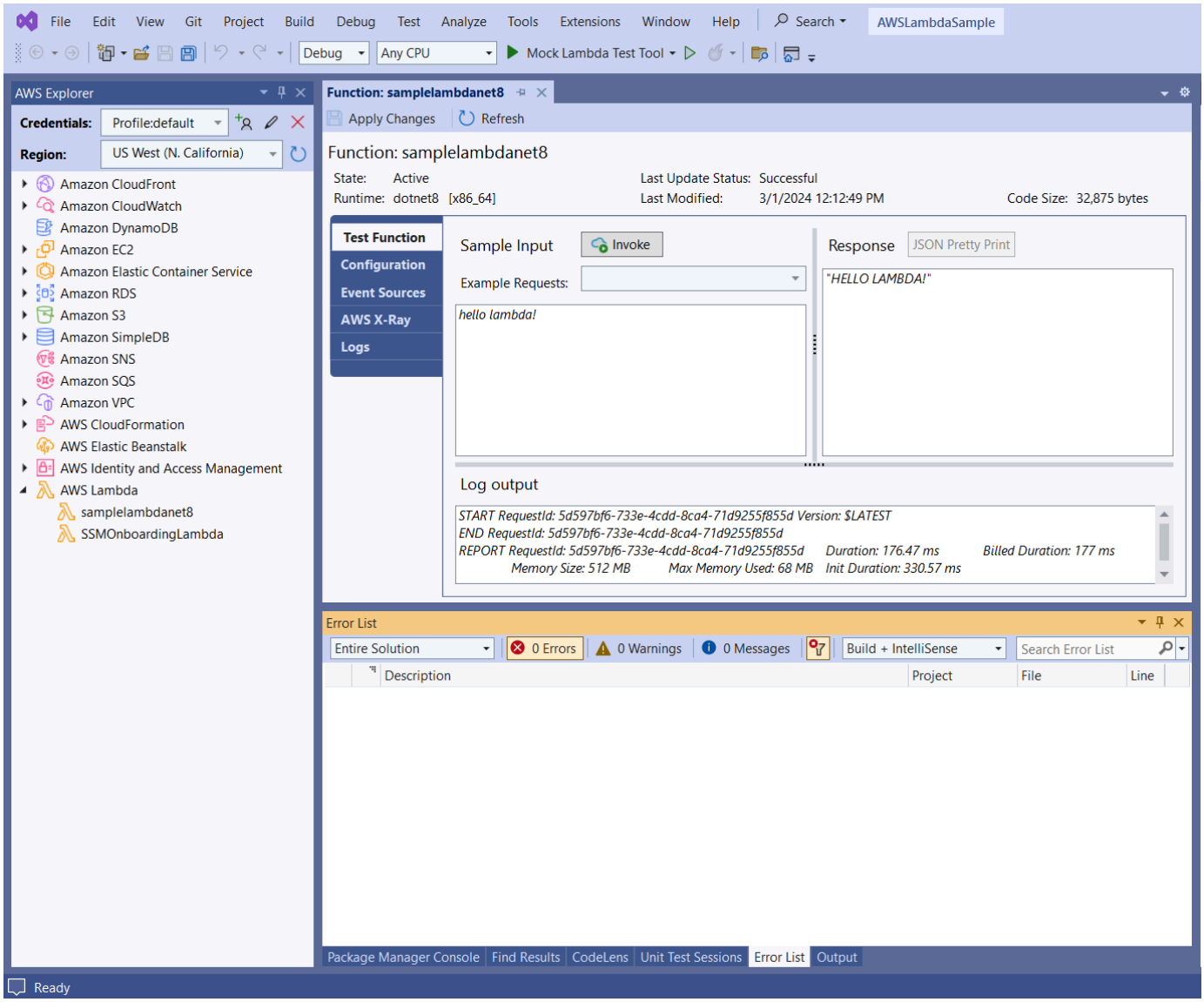
5. 「アップロード」を選択して「アップロード機能」ウィンドウを開き、アップロードプロセスを開始します。

> **Note**
>
> 関数がにアップロードされている間、「アップロード関数」ページが表示されます。AWSアップロード後にレポートを表示するためにウィザードを開いたままにするには、

アップロードが完了する前に、フォームの最後にある [Automatically close wizard on successful completion] (正常完了時にウィザードを自動で閉じる) のチェックボックスをオフにします。
関数がアップロードされると、Lambda 関数はライブ状態になります。[Function:] (関数:) ビューページが開き、新しい Lambda 関数の設定が表示されます。

6. [テスト関数] `hello lambda!` タブからテキスト入力フィールドに入力し、[呼び出し] を選択して Lambda 関数を手動で呼び出します。テキストは大文字に変換されてレスポンスタブに表示されます。

> **Note**
>
> AWS Explorer で AWS Lambda ノードの下にあ デプロイ済みインスタンスをダブルクリックすると、いつでも [Function:] (関数:) ビューを再び開くことができます。



7. (オプション) Lambda 関数が正常に公開されたことを確認するには、にログインして [Lambda] を選択します。 AWS Management Console コンソールには、先ほど作成した関数を含め、発行されたすべての Lambda 関数が表示されます。

クリーンアップ

この例で開発を続行しない場合は、デプロイした関数を削除して、アカウント内の未使用のリソースに対して請求されないようにします。

> **Note**
>
> Lambda は、お客様に代わって Lambda 関数を自動的にモニタリングし、Amazon を通じてメトリクスを報告します。 CloudWatch関数をモニタリングしてトラブルシューティングするには、開発者ガイドの「Amazon での AWS Lambda 関数のトラブルシューティングとモニタリング」 CloudWatch トピックを参照してください。 AWS Lambda

関数を削除するには

1. AWS エクスプローラーからノードを展開します。AWS Lambda
2. デプロイしたインスタンスを右クリックし、[Delete] を選択します。

基本 AWS Lambda Docker イメージの作成プロジェクト

Toolkit for Visual Studio を使用して、 AWS Lambda 関数を Docker イメージとしてデプロイできます。Docker を使用すると、ランタイムをより細かく制御できます。例えば、.NET 8.0 などのカスタムランタイムを選択できます。Docker イメージは、他のコンテナイメージと同じ方法でデプロイします。このチュートリアルは、「チュートリアル: 基本的な Lambda プロジェクト」に忠実に従っていますが、2 つには以下の違いがあります。

- Dockerfile がプロジェクトに含まれています。
- 別の発行設定が選択されます。

Lambda コンテナイメージに関する詳細については、AWS Lambda デベロッパーガイドの「Lambda デプロイパッケージ」を参照してください。

Lambda の操作の詳細については AWS Toolkit for Visual Studio、このユーザーガイドの「 トピックでの AWS Lambda テンプレートの使用 AWS Toolkit for Visual Studio」を参照してください。

Visual Studio .NET Core Lambda プロジェクトを作成する

Lambda Visual Studio テンプレートとブループリントを使用して、プロジェクトの初期化を高速化できます。Lambda ブループリントには、柔軟なプロジェクト基盤の作成を簡素化する、事前に作成された関数が含まれています。

Visual Studio .NET Core Lambdaプロジェクトを作成するには

1. Visual Studio からファイルメニューを展開し、新規 を展開し、プロジェクト を選択します。
2. 「新規プロジェクト」ダイアログボックスで、「言語」、「プラットフォーム」、および「プロジェクトタイプ」のドロップダウンボックスを「すべて」に設定し、「検索**aws lambda**」フィールドに入力します。AWS Lambda プロジェクト (.NET Core - C#) テンプレートを選択します。
3. プロジェクト名 フィールドに「」と入力し**AWSLambdaDocker**、ファイルの場所 を指定し、「 の作成」を選択します。
4. 設計図の選択ページで、.NET 8 (コンテナイメージ) 設計図を選択し、完了を選択して Visual Studio プロジェクトを作成します。これでプロジェクトの構造とコードを確認できるようになりました。

プロジェクトファイルの確認

以下のセクションでは、.NET 8 (コンテナイメージ) ブループリントによって作成された 3 つのプロジェクトファイルについて説明します。

1. `Dockerfile`
2. `aws-lambda-tools-defaults.json`
3. `Function.cs`

1. Dockerfile

`Dockerfile` は 3 つの主要なアクションを実行します。

- `FROM`: このイメージに使用するベースイメージを確立します。このベースイメージは、.NET ランタイム、Lambda ランタイム、および Lambda .NET プロセスのエントリポイントを提供するシェルスクリプトを提供します。
- `WORKDIR`: イメージの内部作業ディレクトリを として確立します`/var/task`。
- `COPY`: ビルドプロセスから生成されたファイルをローカルの場所からイメージの作業ディレクトリにコピーします。

以下は、指定できるオプションの`Dockerfile`アクションです。

- ENTRYPOINT: ベースイメージには既に が含まれています。これはENTRYPOINT、イメージの起動時に実行される起動プロセスです。独自のエントリを指定する場合は、その基本エントリポイントを上書きします。
- CMD: 実行する AWS カスタムコードを指示します。カスタムメソッドには完全修飾名が必要です。この行は Dockerfile に直接含める必要があるか、発行プロセス中に指定することができます。

```
# Example of alternative way to specify the Lambda target method rather than during
 the publish process.
CMD [ "AWSLambdaDocker::AWSLambdaDocker.Function::FunctionHandler"]
```

以下は、.NET 8 (コンテナイメージ) ブループリントによって作成された Dockerfile の例です。

```
FROM public.ecr.aws/lambda/dotnet:8

WORKDIR /var/task

# This COPY command copies the .NET Lambda project's build artifacts from the host
 machine into the image.
# The source of the COPY should match where the .NET Lambda project publishes its build
 artifacts. If the Lambda function is being built
# with the AWS .NET Lambda Tooling, the `--docker-host-build-output-dir` switch
 controls where the .NET Lambda project
# will be built. The .NET Lambda project templates default to having `--docker-host-
build-output-dir`
# set in the aws-lambda-tools-defaults.json file to "bin/Release/lambda-publish".
#
# Alternatively Docker multi-stage build could be used to build the .NET Lambda project
 inside the image.
# For more information on this approach checkout the project's README.md file.
COPY "bin/Release/lambda-publish"  .
```

2 aws-lambda-tools-defaults..json

aws-lambda-tools-defaults.json ファイルは、Toolkit for Visual Studio デプロイウィザードと .NET Core CLI のデフォルト値を指定するために使用されます。次のリストでは、 aws-lambda-tools-defaults.json ファイルで設定できるフィールドについて説明します。

- profile: AWS プロファイルを設定します。
- region: リソースが保存される AWS リージョンを設定します。

- configuration: 関数の発行に使用される設定を設定します。
- package-type: デプロイパッケージタイプをコンテナイメージまたは .zip ファイルアーカイブに設定します。
- function-memory-size: 関数のメモリ割り当てを MB 単位で設定します。
- function-timeout: タイムアウトは、Lambda 関数が実行できる秒単位の最大時間です。これを 1 秒単位で、最大値 15 分まで調整できます。
- docker-host-build-output-dir: ビルドプロセスの出力ディレクトリを設定し、 の手順と関連付けますDockerfile。
- image-command: は、Lambda 関数を実行するコードであるメソッドの完全修飾名です。構文は次のとおりです：{Assembly}::{Namespace}.{ClassName}::{MethodName}詳細については、「ハンドラー署名」を参照してください。ここで image-command を設定すると、後ほど Visual Studio の発行ウィザードにこの値が事前入力されます。

以下は、 aws-lambda-tools-defaults.NET 8 (コンテナイメージ) ブループリントによって作成された .json の例です。

```
{
  "Information": [
    "This file provides default values for the deployment wizard inside Visual Studio
 and the AWS Lambda commands added to the .NET Core CLI.",
    "To learn more about the Lambda commands with the .NET Core CLI execute the
 following command at the command line in the project root directory.",
    "dotnet lambda help",
    "All the command line options for the Lambda command can be specified in this
 file."
  ],
  "profile": "default",
  "region": "us-west-2",
  "configuration": "Release",
  "package-type": "image",
  "function-memory-size": 512,
  "function-timeout": 30,
  "image-command": "AWSLambdaDocker::AWSLambdaDocker.Function::FunctionHandler",
  "docker-host-build-output-dir": "./bin/Release/lambda-publish"
}
```

3. Function.cs

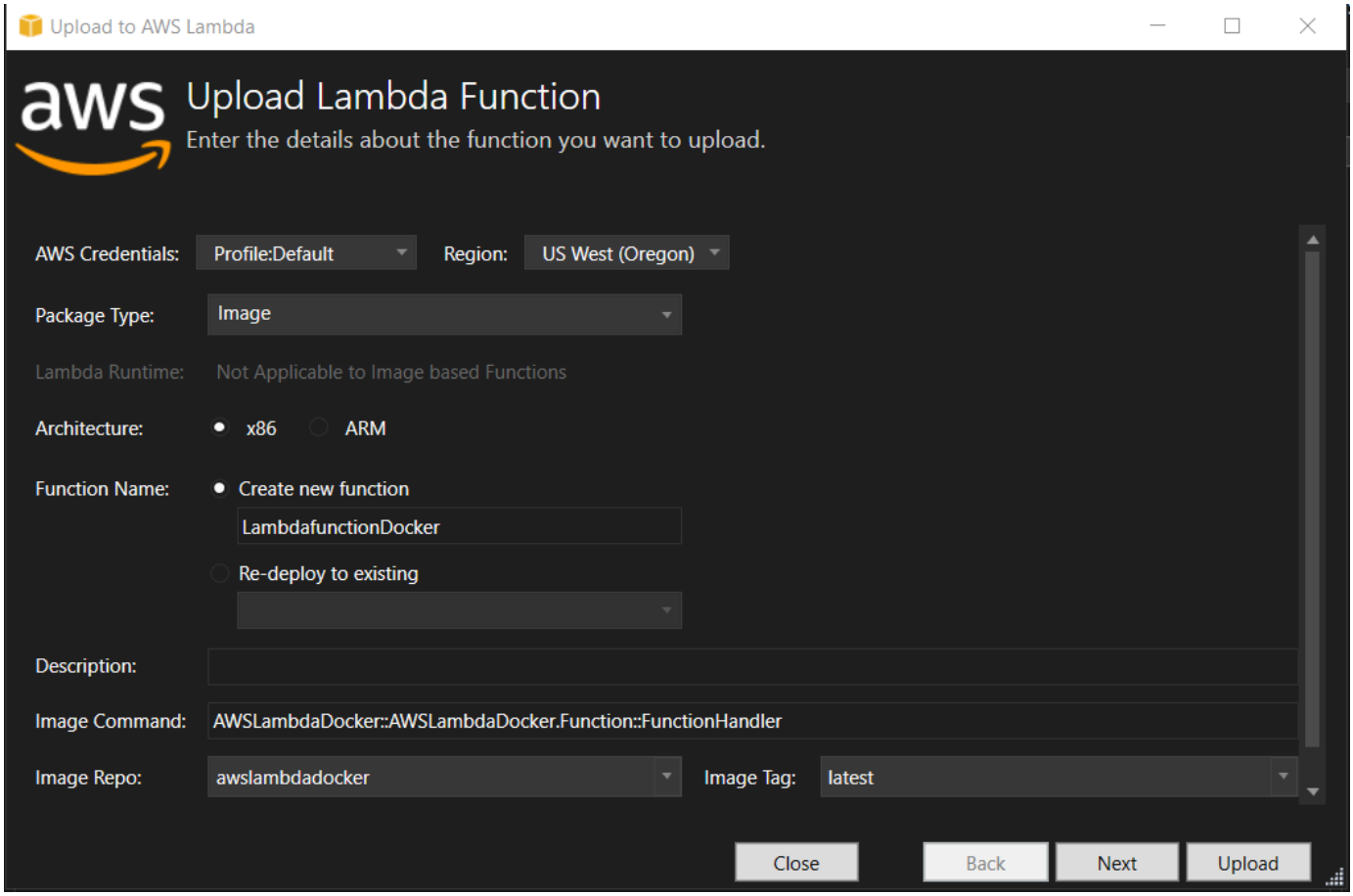
`Function.cs` ファイルは、Lambda 関数として公開される c# 関数を定義します。-`FunctionHandler`は、Lambda 関数の実行時に実行される Lambda 関数です。このプロジェクトでは、 は入力テキスト`ToUpper()`で をFunctionHandler呼び出します。

Lambda に発行する

ビルドプロセスによって生成された Docker イメージは、Amazon Elastic Container Registry (Amazon ECR) にアップロードされます。Amazon ECR は、デベロッパーが Docker コンテナイメージを簡単に保存、管理、デプロイできる完全マネージド型の Docker コンテナレジストリです。Amazon ECR はイメージをホストし、Lambda は呼び出されたときにプログラムされた Lambda 機能を提供するために参照します。

関数を Lambdaに発行するには

1. Solution Explorer から、プロジェクトのコンテキストメニューを開き (右クリック)、パブリッシュ先 AWS Lambdaを選択して Lambda 関数のアップロードウィンドウを開きます。
2. Lambda 関数のアップロードページで、次の操作を行います。



a. [Package Type] (パッケージタイプ) については、プロジェクト内で発行ウィザードが `Dockerfile` を検出したので、**`Image`** が パッケージタイプとして自動的に選択されます。

b. [Function Name] (関数名) には、Lambda インスタンスの表示名を入力します。この名前は、Visual Studio の AWS Explorer および AWS Management Consoleの両方に表示される参照名です。

c. [Description] (説明) には、 AWS Management Console内のインスタンスとともに表示するテキストを入力します。

d. [Image Command] (イメージコマンド) では、Lambda 関数で実行するメソッドへの完全修飾パスを入力します：**`AWSLambdaDocker::AWSLambdaDocker.Function::FunctionHandler`**

> **Note**
>
> ここに入力されたメソッド名は、Dockerfile 内のすべての CMD 命令を上書きします。[Image Command] (イメージコマンド) を入力するのが、オプションであるのは、`Dockerfile`が`CMD`Lambda 関数の起動方法を指示するを含む場合においてだけです。

e. [Image Repo] (イメージ Repo) では、新規もしくは既存の Amazon Elastic コンテナレジストリの名前を入力します。ビルドプロセスが作成する Docker イメージは、このレジストリにアップロードされます。公開されている Lambda 定義は、その Amazon ECR イメージを参照します。

f. [Image Tag] (イメージタグ)で、リポジトリ内のイメージに関連付ける Docker タグを入力します。

g. [次へ] をクリックします。

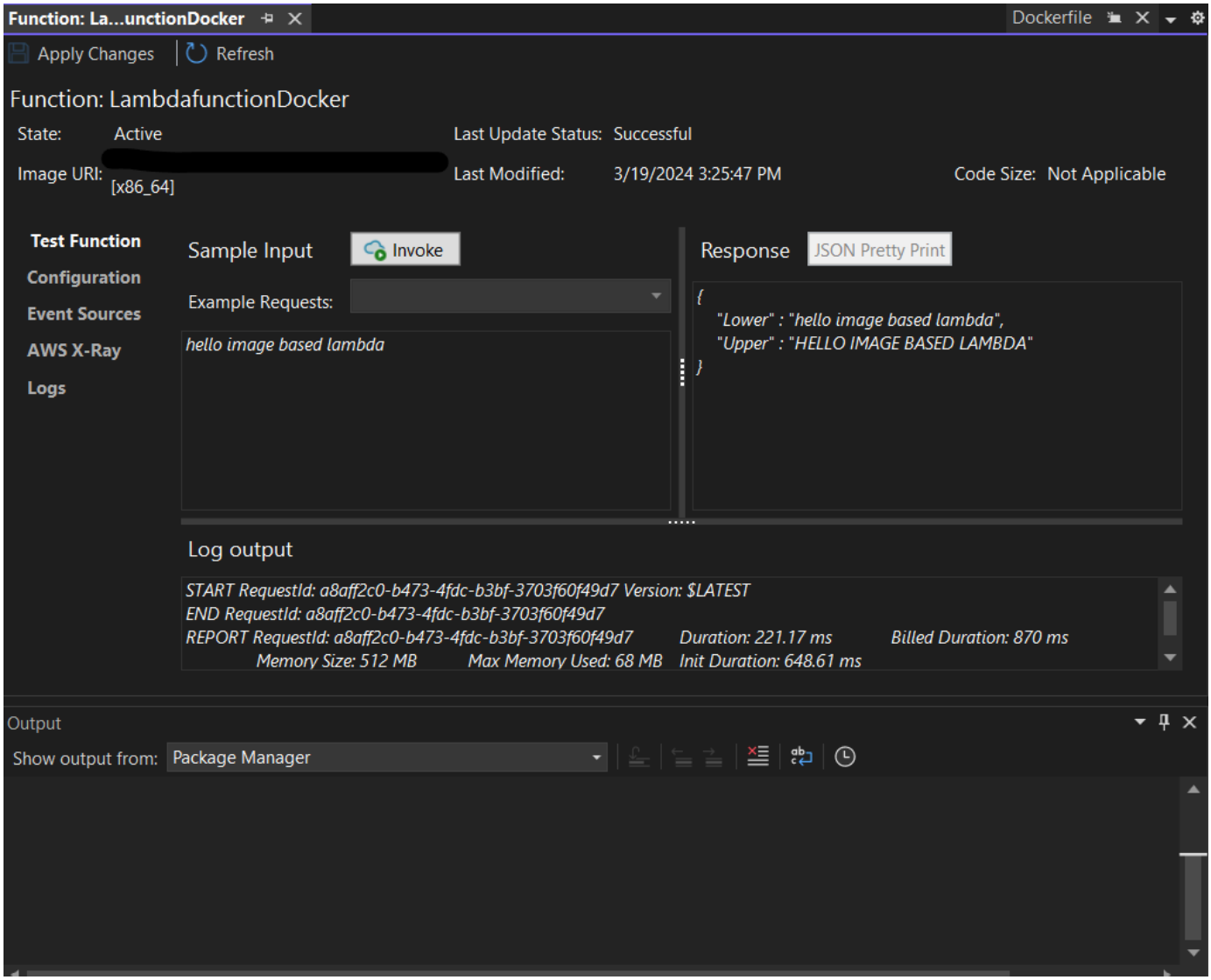
3. [Advanced Function Details] (アドバンスト関数の詳細) ページの [Role Name] (ロール名) で、アカウントに関連付けられているロールを選択します。このロールは、関数内のコードによって行われる Amazon Web Services コールに一時的な認証情報を提供するために使用されます。ロールがない場合は、 AWS 管理ポリシーに基づいて新しいロールを選択し、 を選択しますAWSLambdaBasicExecutionRole。

> **Note**
>
> アカウントには IAM ListPolicies アクションを実行するアクセス許可が必要です。許可がない場合、ロール名リストは空になります。

4. アップロードを選択して、アップロードと公開のプロセスを開始します。

> **Note**
>
> -関数がアップロード中にアップロード機能ページが表示されます。発行プロセスは、設定パラメータに基づいてイメージをビルドし、必要に応じて Amazon ECR リポジトリを作成し、必要ならばイメージをリポジトリにアップロードし、そのイメージを使用してそのリポジトリを参照する Lambda を作成します。
> 関数がアップロードされると、関数 ページが開き、新しい Lambda 関数の設定が表示されます。

5. Lambda 関数を手動で呼び出すには、[Test Function] (関数のテスト) タブで、リクエストの自由文入力フィールドに `hello image based lambda` を入力してから [Invoke] (呼び出し) を選択します。大文字に変換されたテキストは、[Response] (レスポンス) に表示されます。



6. リポジトリを参照するには、AWS Explorer で [Amazon Elastic Container Service] の下にある [Repositories] (リポジトリ) を選択します。

   AWS Explorer で、AWS Lambda ノードの下にあるデプロイ済みインスタンスをダブルクリックすると、いつでも [Function:] (関数:) ビューを再び開くことができます。

> **Note**
>
> AWS Explorer ウィンドウが開いていない場合は、ビュー -> AWS Explorer でドッキングできます。

7. [Configuration] (設定) タブに、イメージ固有の追加設定オプションがあります。このタブには、Dockerfile 内で指定された可能性がある `ENTRYPOINT`、`CMD`、および `WORKDIR` に上書きする方法が用意されています。説明アップロード/公開中に入力した説明 (ある場合) です。

クリーンアップ

この例で開発を続行しない場合は、デプロイされた関数および ECR イメージを削除して、アカウント内の未使用のリソースに対して請求されないようにすることを忘れないでください。

- 関数を削除するには、AWS Explorer で AWS Lambda ノードの下にあるデプロイ済みインスタンスを右クリックします。
- リポジトリは、AWS Explorer で [Amazon Elastic Container Service] - [Repositories] (リポジトリ) の下で削除できます。

次のステップ

Lambda イメージの作成およびテストについては、「Lambda でのコンテナイメージの使用」を参照してください。

コンテナイメージのデプロイ、アクセス許可、および構成設定の上書きの詳細については、「関数の設定」を参照してください。

チュートリアル:でサーバーレスアプリケーションをビルドしてテストする AWS Lambda

テンプレートを使用してサーバーレス Lambda アプリケーションを構築できます。 AWS Toolkit for Visual Studio Lambda プロジェクトのテンプレートには、AWS サーバーレスアプリケーション用のテンプレートが含まれています。これは、AWS サーバーレスアプリケーションモデル (SAM) AWS Toolkit for Visual Studio の実装です。AWS このプロジェクトタイプを使用すると、 AWS Lambda 一連の関数を開発し、 AWS 必要なリソースとともにアプリケーション全体としてデプロイできます。これにより、デプロイのオーケストレーションが可能になります AWS CloudFormation 。

の設定に関する前提条件と情報については AWS Toolkit for Visual Studio、「Toolkit for Visual Studio の AWS Lambda テンプレートの使用」を参照してください。 AWS

トピック

- 新しい AWS サーバーレスアプリケーションプロジェクトを作成します
- サーバーレスアプリケーションファイルの確認

- サーバーレスアプリケーションのデプロイ
- サーバーレスアプリケーションのテスト

新しい AWS サーバーレスアプリケーションプロジェクトを作成します

AWS サーバーレスアプリケーションプロジェクトは、サーバーレステンプレートを使用して Lambda 関数を作成します。 AWS CloudFormation AWS CloudFormation テンプレートを使用すると、データベースなどの追加リソースの定義、IAM ロールの追加、複数の関数のデプロイを一度に行うことができます。これは、単一の AWS Lambda 関数の開発とデプロイに重点を置く Lambda プロジェクトとは異なります。

以下の手順では、新しいサーバーレスアプリケーションプロジェクトを作成する方法について説明します。 AWS

1. Visual Studio から [ファイル] メニューを展開し、[新規] を展開して [プロジェクト] を選択します。
2. 「新規プロジェクト」ダイアログで、「言語」、「プラットフォーム」、「プロジェクトタイプ」の各ドロップダウンボックスが「すべて...」に設定されていることを確認し、「検索」**aws lambda** フィールドに入力します。
3. AWS Serverless Application with Tests (.NET Core - C#) テンプレートを選択します。

   > **Note**
   >
   > 結果の上部に [AWS テスト付きサーバーレスアプリケーション (.NET Core-C#)] テンプレートが表示されない場合があります。

4. [次へ] をクリックして [新規プロジェクトの設定] ダイアログを開きます。
5. 「新規プロジェクトの設定」ダイアログで「名前」を入力し**ServerlessPowertools**、残りのフィールドを好みに合わせて入力します。[作成] ボタンを選択して [ブループリントの選択] ダイアログに進みます。
6. [ブループリントの選択] AWS Lambdaダイアログからブループリント用の Powertools を選択し、次に [完了] を選択して Visual Studio プロジェクトを作成します。

サーバーレスアプリケーションファイルの確認

以下のセクションでは、プロジェクト用に作成された 3 つのサーバーレスアプリケーションファイルについて詳しく説明します。

1. serverless.template
2. Functions.cs
3. aws-lambda-tools-defaults.json

1. サーバーレス. テンプレート

`serverless.template`ファイルは、 AWS CloudFormation サーバーレス関数やその他のリソースを宣言するためのテンプレートです。 AWS このプロジェクトに含まれるファイルには、Amazon API Gateway HTTP *Get* を通じてオペレーションとして公開される単一の Lambda 関数の宣言が含まれています。このテンプレートを編集して既存の関数をカスタマイズしたり、アプリケーションに必要な関数やその他のリソースを追加したりできます。

次は、`serverless.template` ファイルの例です。

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Transform": "AWS::Serverless-2016-10-31",
  "Description": "An AWS Serverless Application.",
  "Resources": {
    "Get": {
      "Type": "AWS::Serverless::Function",
      "Properties": {
         "Architectures": [
            "x86_64"
            ],
         "Handler": "ServerlessPowertools::ServerlessPowertools.Functions::Get",
         "Runtime": "dotnet8",
         "CodeUri": "",
         "MemorySize": 512,
         "Timeout": 30,
         "Role": null,
         "Policies": [
            "AWSLambdaBasicExecutionRole"
            ],
         "Environment": {
            "Variables": {
               "POWERTOOLS_SERVICE_NAME": "ServerlessGreeting",
               "POWERTOOLS_LOG_LEVEL": "Info",
               "POWERTOOLS_LOGGER_CASE": "PascalCase",
               "POWERTOOLS_TRACER_CAPTURE_RESPONSE": true,
               "POWERTOOLS_TRACER_CAPTURE_ERROR": true,
```

```
                "POWERTOOLS_METRICS_NAMESPACE": "ServerlessGreeting"
                }
            },
         "Events": {
            "RootGet": {
               "Type": "Api",
               "Properties": {
                  "Path": "/",
                  "Method": "GET"
                  }
               }
            }
         }
      }
   },
  "Outputs": {
    "ApiURL": {
      "Description": "API endpoint URL for Prod environment",
      "Value": {
        "Fn::Sub": "https://${ServerlessRestApi}.execute-api.
${AWS::Region}.amazonaws.com/Prod/"
      }
    }
  }
}
```

...AWS:: Serverless::Function...宣言フィールドの多くは Lambda プロジェクトデプロイのフィールドと似ていることに注意してください。Powertools のロギング、メトリックス、トレーシングは、以下の環境変数を使用して設定されます。

- POWERTOOLS_SERVICE_NAME= ServerlessGreeting
- POWERTOOLS _LOG_LEVEL=INFO
- パワーツール_ロガー_ケース= PascalCase
- PowerTools_TRACER_CAPTURE_RESPONSE=TRUE
- PowerTools_TRACER_CAPTURE_ERROR=TRUE
- POWERTOOLS_METRICS_NAMESPACE= ServerlessGreeting

環境変数の定義やその他の詳細については、Powertools for Reference の Web サイトを参照してください。 AWS Lambda

2. Functions.cs

Functions.csは、テンプレートファイルで宣言された 1 つの関数にマップされた C# メソッドを含むクラスファイルです。Lambda 関数は API Gateway HTTP Get からのメソッドに応答します。ファイルの例を以下に示します。Functions.cs

```
public class Functions
{
    [Logging(LogEvent = true, CorrelationIdPath = CorrelationIdPaths.ApiGatewayRest)]
    [Metrics(CaptureColdStart = true)]
    [Tracing(CaptureMode = TracingCaptureMode.ResponseAndError)]
    public APIGatewayProxyResponse Get(APIGatewayProxyRequest request, ILambdaContext
 context)
    {
        Logger.LogInformation("Get Request");

        var greeting = GetGreeting();

        var response = new APIGatewayProxyResponse
        {
            StatusCode = (int)HttpStatusCode.OK,
            Body = greeting,
            Headers = new Dictionary (string, string) { { "Content-Type", "text/
plain" } }
        };

        return response;
    }

    [Tracing(SegmentName = "GetGreeting Method")]
    private static string GetGreeting()
    {
        Metrics.AddMetric("GetGreeting_Invocations", 1, MetricUnit.Count);

        return "Hello Powertools for AWS Lambda (.NET)";
    }
}
```

3. aws-lambda-tools-defaults.json

`aws-lambda-tools-defaults.json`Visual Studio AWS 内のデプロイウィザードのデフォルト値と、.NET Core CLI AWS Lambda に追加されるコマンドについて説明します。`aws-lambda-tools-defaults.json`このプロジェクトに含まれるファイルの例を以下に示します。

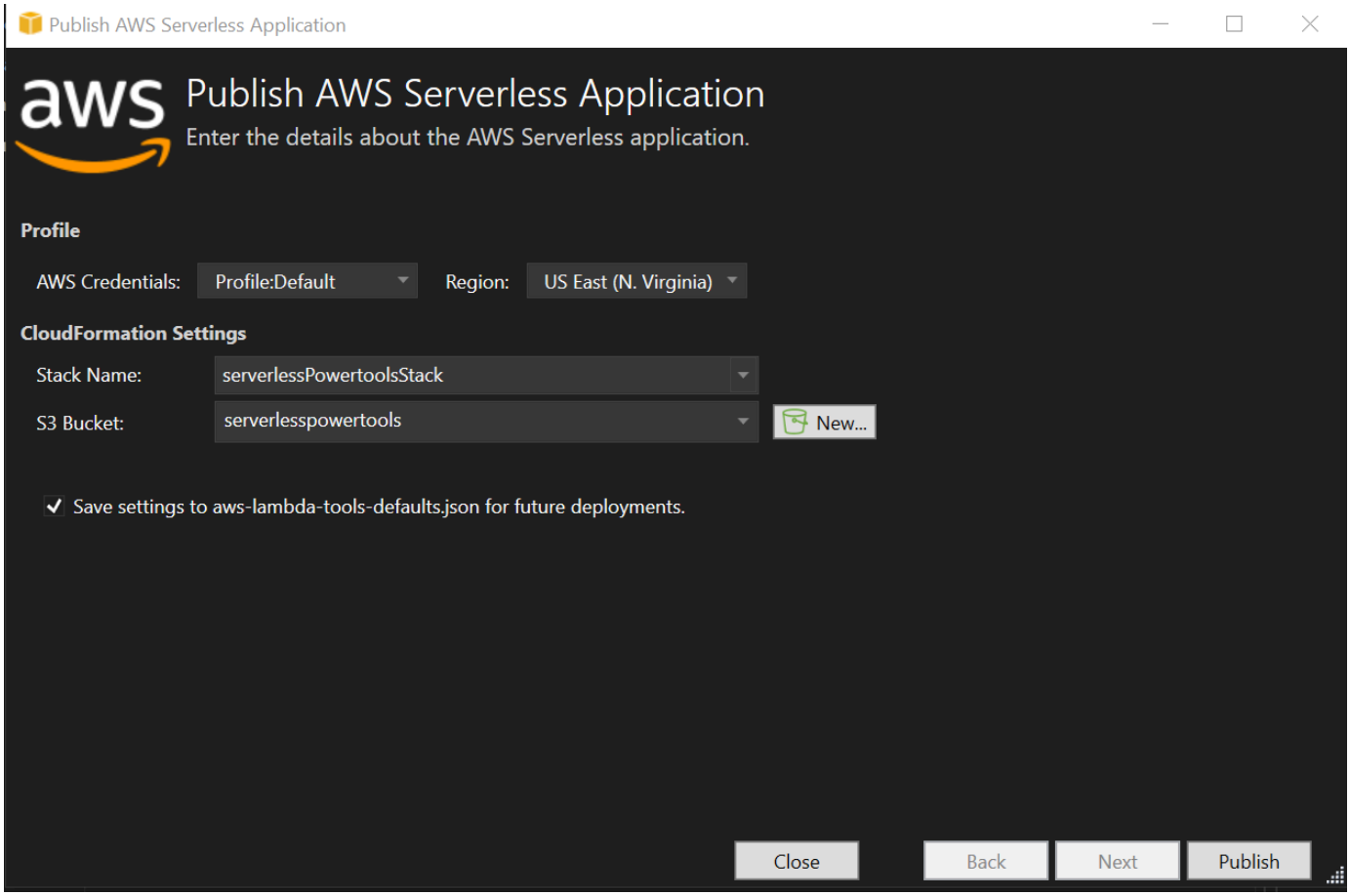
```
{
  "profile": "Default",
  "region": "us-east-1",
  "configuration": "Release",
  "s3-prefix": "ServerlessPowertools/",
  "template": "serverless.template",
  "template-parameters": ""
}
```

サーバーレスアプリケーションのデプロイ

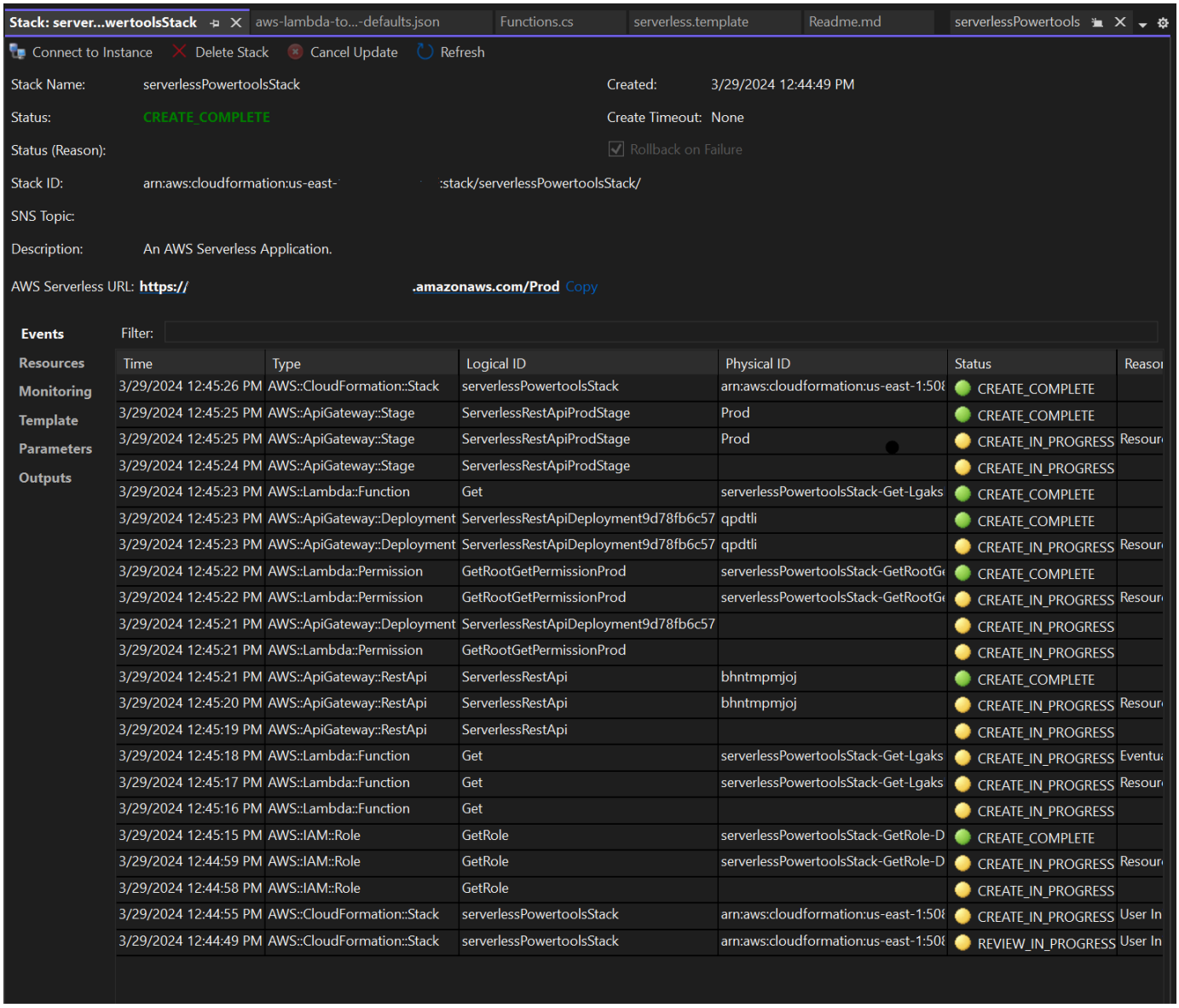
サーバーレスアプリケーションをデプロイするには、以下の手順を実行します。

1. ソリューションエクスプローラーから、プロジェクトのコンテキストメニューを開き (右クリック)、[AWS Lambda に公開] を選択して [AWS サーバーレスアプリケーションの公開] ダイアログを開きます。
2. 「 AWS サーバーレスアプリケーションの公開」ダイアログから、「スタック名」 AWS CloudFormation フィールドにスタックコンテナの名前を入力します。
3. S3 Bucket フィールドで、アプリケーションバンドルのアップロード先となる Amazon S3 バケットを選択するか、[New...] を選択します。 ボタンをクリックし、新しい Amazon S3 バケットの名前を入力します。次に [Publish to Publish] を選択してアプリケーションをデプロイします。

> **Note**
>
> AWS CloudFormation スタックと Amazon S3 AWS バケットは同じリージョンに存在する必要があります。プロジェクトの残りの設定はファイルで定義されます。`serverless.template`



4. 公開プロセス中にスタックビューウィンドウが開き、デプロイが完了すると Status フィールドが表示されます。CREATE_COMPLETE



サーバーレスアプリケーションのテスト

スタックの作成が完了すると、AWS サーバーレス URL を使用してアプリケーションを表示できます。関数やパラメーターを追加せずにこのチュートリアルを完了した場合、 AWS サーバーレス URL にアクセスすると、Web ブラウザーに次のフレーズが表示されます。`Hello Powertools for AWS Lambda (.NET)`

チュートリアル: Amazon Rekognition Lambda アプリケーションの作成

このチュートリアルでは、Amazon Rekognition を使用して、検出ラベルのタグを Amazon S3 オブジェクトに付ける Lambdaアプリケーションを作成する方法を示します。

の設定に関する前提条件と情報については AWS Toolkit for Visual Studio、「Toolkit for Visual Studio の AWS Lambda テンプレートの使用」を参照してください。 AWS

Visual Studio の NET Core Lambda Image Rekognition プロジェクトを作成します

以下の手順では、から Amazon Rekognition Lambda アプリケーションを作成する方法について説明します。 AWS Toolkit for Visual Studio

> **Note**
>
> 作成時、アプリケーションには 2 つのプロジェクトからなるソリューションが作成されます。1 つは Lambda にデプロイする Lambda 関数コードを含むソースプロジェクト、もう 1 つは関数をローカルでテストするための xUnit を使用するテストプロジェクトです。
> Visual Studio NuGet がプロジェクトのすべての参照を見つけられない場合があります。これは、ブループリントには依存関係が必要であり、そこから取得する必要があるためです。NuGet新しいプロジェクトが作成されると、Visual Studio はローカル参照のみを取得し、リモート参照は取得しません。 NuGet NuGet エラーを修正するには、参照を右クリックして [パッケージを復元] を選択します。

1. Visual Studio から [ファイル] メニューを展開し、[新規] を展開して [プロジェクト] を選択します。
2. 「新規プロジェクト」ダイアログで、「言語」、「プラットフォーム」、「プロジェクトタイプ」の各ドロップダウンボックスが「すべて...」に設定されていることを確認し、「検索」**aws lambda** フィールドに入力します。
3. AWS Lambda テスト付き (.NET Core-C#) テンプレートを選択します。
4. [次へ] をクリックして [新規プロジェクトの設定] ダイアログを開きます。
5. 「新規プロジェクトの設定」ダイアログで、「名前 ImageRekognition」に「」と入力し、残りのフィールドを好みに合わせて入力します。[作成] ボタンを選択して [ブループリントの選択] ダイアログに進みます。
6. [ブループリントの選択] ダイアログから [イメージラベルの検出] ブループリントを選択し、[完了] を選択して Visual Studio プロジェクトを作成します。

> **Note**
>
> このブループリントには Amazon S3 イベントをリッスンするためのコードがあり、Amazon Rekognition を使用してラベルを検出し、それらを Amazon S3 オブジェクトにタグとして追加できます。

プロジェクトファイルのレビュー

以下のセクションでは、これらのプロジェクトファイルを調べます。

1. `Function.cs`
2. `aws-lambda-tools-defaults.json`

1. Function.cs

`Function.cs`ファイル内のコードの最初の部分は、ファイルの上部にあるアセンブリ属性です。デフォルトでは、Lambda は入力パラメータと戻り値の型のみを受け入れます。`System.IO.Stream`入力パラメータと戻り値の型に型付きクラスを使用するには、シリアライザーを登録する必要があります。アセンブリ属性は、`Newtonsoft.Json`ストリームを型付きクラスに変換するために使用する Lambda JSON シリアライザーを登録します。シリアライザーをアセンブリまたはメソッドレベルで設定できます。

以下はアセンブリ属性の例です。

```
// Assembly attribute to enable the Lambda function's JSON input to be converted into
 a .NET class.
[assembly:
 LambdaSerializer(typeof(Amazon.Lambda.Serialization.SystemTextJson.DefaultLambdaJsonSerializer
```

このクラスには、2 つのコンストラクタがあります。1 つ目は、デフォルトコンストラクタで、Lambda がユーザーの関数を呼び出すときに使用されます。このコンストラクターは Amazon S3 サービスクライアントと Amazon Rekognition サービスクライアントを作成します。また、コンストラクタは、デプロイ時に関数に割り当てた IAM AWS ロールからこれらのクライアントの認証情報を取得します。 AWS クライアントのリージョンは、Lambda 関数が実行されているリージョンに設定されます。この設計図では、Amazon Rekognition サービスがラベルについて最低限の信頼度を持っている場合にのみ、Amazon S3 オブジェクトにタグを追加する必要があります。このコンスト

ラクタは環境変数 MinConfidence をチェックして、許容できる信頼レベルを決定します。この環境変数は、Lambda 関数をデプロイするときに、設定できます。

以下は、のファーストクラスコンストラクターの例です。Function.cs

```
public Function()
{
    this.S3Client = new AmazonS3Client();
    this.RekognitionClient = new AmazonRekognitionClient();

    var environmentMinConfidence =
 System.Environment.GetEnvironmentVariable(MIN_CONFIDENCE_ENVIRONMENT_VARIABLE_NAME);
    if(!string.IsNullOrWhiteSpace(environmentMinConfidence))
    {
        float value;
        if(float.TryParse(environmentMinConfidence, out value))
        {
            this.MinConfidence = value;
            Console.WriteLine($"Setting minimum confidence to {this.MinConfidence}");
        }
        else
        {
            Console.WriteLine($"Failed to parse value {environmentMinConfidence} for
 minimum confidence. Reverting back to default of {this.MinConfidence}");
        }
    }
    else
    {
        Console.WriteLine($"Using default minimum confidence of {this.MinConfidence}");
    }
}
```

次の例は、2 番目のコンストラクターをテストに利用する方法を示しています。テストプロジェクトは独自の S3 クライアントと Rekognition クライアントを設定し、それらを渡します。

```
public Function(IAmazonS3 s3Client, IAmazonRekognition rekognitionClient, float
 minConfidence)
{
    this.S3Client = s3Client;
    this.RekognitionClient = rekognitionClient;
    this.MinConfidence = minConfidence;
}
```

FunctionHandlerファイル内のメソッドの例を以下に示します。Function.cs

```
public async Task FunctionHandler(S3Event input, ILambdaContext context)
{
    foreach(var record in input.Records)
    {
        if(!SupportedImageTypes.Contains(Path.GetExtension(record.S3.Object.Key)))
        {
            Console.WriteLine($"Object {record.S3.Bucket.Name}:{record.S3.Object.Key}
 is not a supported image type");
            continue;
        }

        Console.WriteLine($"Looking for labels in image {record.S3.Bucket.Name}:
{record.S3.Object.Key}");
        var detectResponses = await this.RekognitionClient.DetectLabelsAsync(new
 DetectLabelsRequest
        {
            MinConfidence = MinConfidence,
            Image = new Image
            {
                S3Object = new Amazon.Rekognition.Model.S3Object
                {
                    Bucket = record.S3.Bucket.Name,
                    Name = record.S3.Object.Key
                }
            }
        });

        var tags = new List();
        foreach(var label in detectResponses.Labels)
        {
            if(tags.Count < 10)
            {
                Console.WriteLine($"\tFound Label {label.Name} with confidence
 {label.Confidence}");
                tags.Add(new Tag { Key = label.Name, Value =
 label.Confidence.ToString() });
            }
            else
            {
                Console.WriteLine($"\tSkipped label {label.Name} with confidence
 {label.Confidence} because maximum number of tags reached");
            }
```

```
        }

        await this.S3Client.PutObjectTaggingAsync(new PutObjectTaggingRequest
        {
            BucketName = record.S3.Bucket.Name,
            Key = record.S3.Object.Key,
            Tagging = new Tagging
            {
                TagSet = tags
            }
        });
    }
    return;
}
```

FunctionHandler は、インスタンスの作成後に Lambda が呼び出すメソッドです。入力パラメータは S3Event 型で Stream ではないことに注意してください。これができるのは、登録された Lambda JSON シリアライザーのためです。S3Event には Amazon S 3 でトリガーされたイベントに関するすべての情報が含まれています。関数は、イベントの一部であるすべての S3 オブジェクトをループし、Rekognition に対しラベルの検出を指示します。ラベルが検出された後、それらは S3 オブジェクトにタグとして追加されます。

> **Note**
>
> コードにはへの呼び出しが含まれていますConsole.WriteLine()。関数が Lambda で実行されている場合、すべての呼び出しは Amazon Logs Console.WriteLine() にリダイレクトされます。 CloudWatch

2. aws-lambda-tools-defaults.json

aws-lambda-tools-defaults.jsonこのファイルには、ブループリントがデプロイウィザードの一部のフィールドに事前入力するように設定したデフォルト値が含まれています。また、.NET Core CLI と統合するためのコマンドラインオプションを設定する場合にも役立ちます。

.NET Core CLI 統合にアクセスするには、関数のプロジェクトディレクトリに移動して入力します**dotnet lambda help**。

> **Note**
>
> 関数ハンドラーは、呼び出された関数に応答して Lambda が呼び出すメソッドを示します。このフィールドの形式は:. `<assembly-name>::<full-type-name>::<method-name>` 名前空間はタイプ名に含める必要があります。

関数をデプロイする

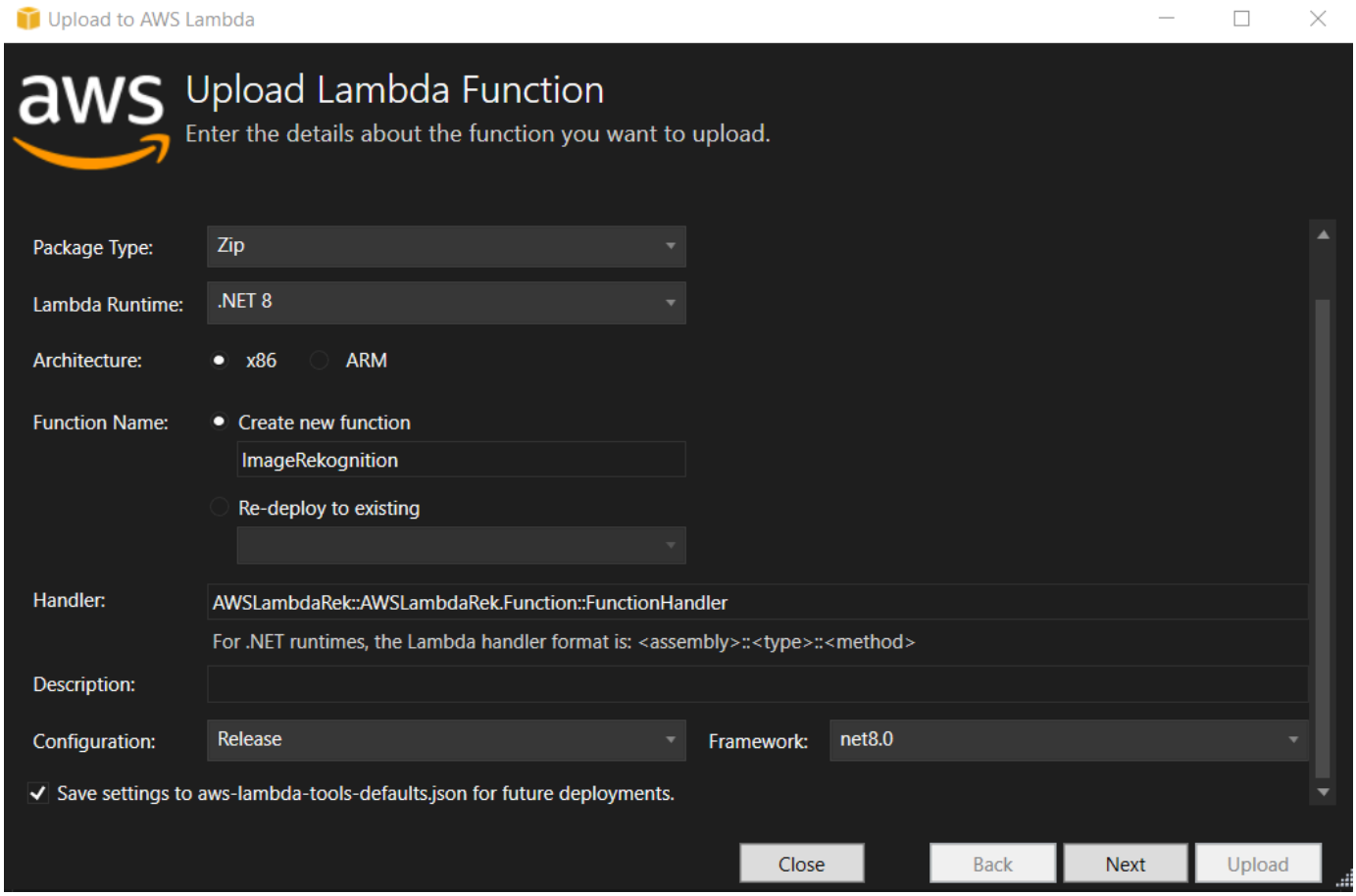
以下の手順では、Lambda 関数をデプロイする方法について説明します。

1. ソリューションエクスプローラーで Lambda プロジェクトを右クリックし、[AWS Lambda に公開] を選択して [アップロード先] ウィンドウを開きます。 AWS Lambda

   > **Note**
   >
   > プリセット値はファイルから取得されます。`aws-lambda-tools-defaults.json`

2. 「アップロード先 AWS Lambda」ウィンドウの「関数名」フィールドに名前を入力し、「次へ」ボタンを選択して「拡張機能の詳細」ウィンドウに進みます。

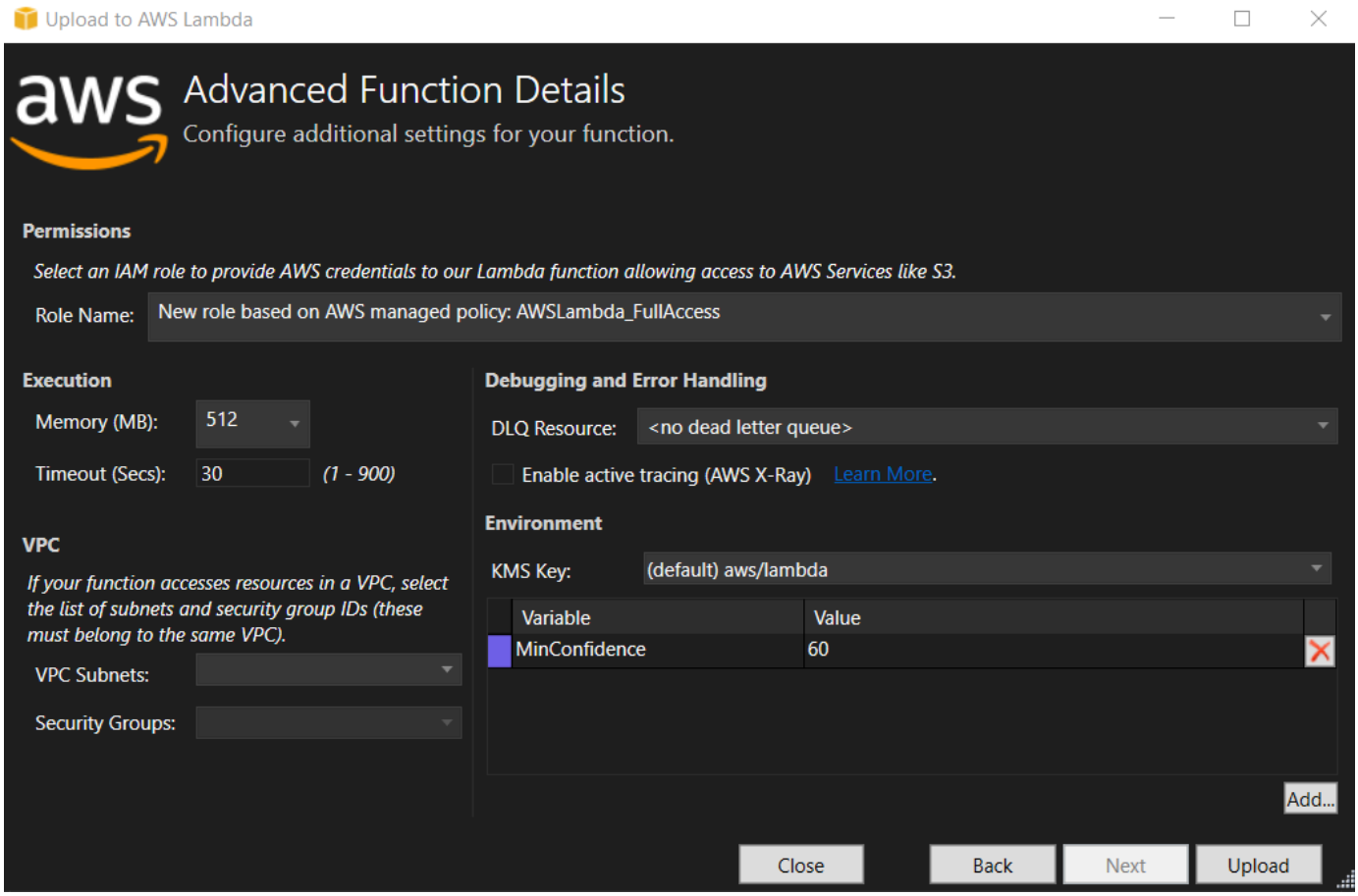
   > **Note**
   >
   > この例では、関数名を使用しています**`ImageRekognition`**。



3. 「拡張機能の詳細」ウィンドウから、コードに Amazon S3 および Amazon Rekognition リソースにアクセスする権限を付与する IAM ロールを選択します。

> **Note**
>
> この例に従っている場合は、ロールを選択してください。`AWSLambda_FullAccess`

4. 環境変数を 60 `MinConfidence` に設定し、[Upload] を選択してデプロイプロセスを開始します。AWS Explorer に Function ビューが表示されたら、公開プロセスは完了です。



5. デプロイが成功したら、[Event Sources] タブに移動して、新しい関数にイベントを送信するように Amazon S3 を設定します。
6. [イベントソース] タブから [追加] ボタンを選択し、Lambda 関数に接続する Amazon S3 バケットを選択します。

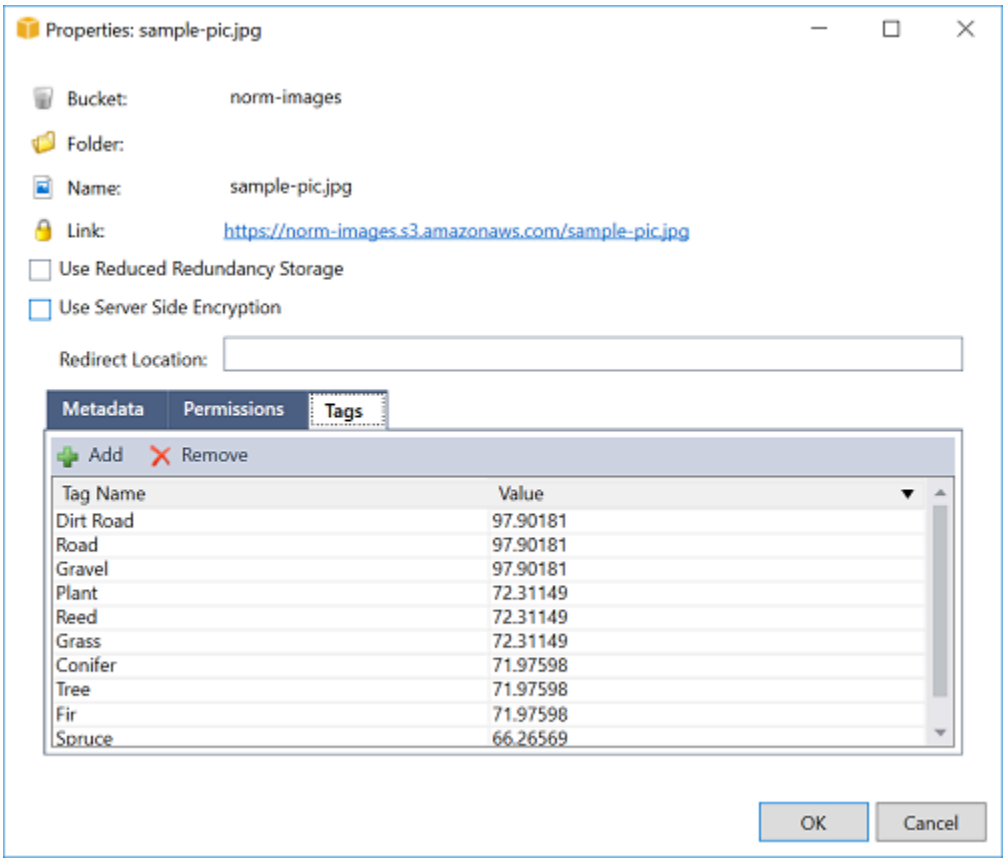
> **Note**
>
> バケットは Lambda AWS 関数と同じリージョンにある必要があります。

関数をテストする

これで、関数がデプロイされ、イベントのソースとして S3 バケットが関数に対して設定されたので、S3 バケットブラウザを AWS Explorer から、選択したバケットについて開きます。次に、いくつかのイメージをアップロードします。

アップロードが完了すると、使用する関数のビューからログを調べることで、関数が実行されたことを確認できます。または、バケットブラウザ内のイメージを右クリックし、[Properties (プロパ

ティ)] を選択します。[Tags (タグ)] タブで、使用するオブジェクトに適用されたタグを表示できます。



チュートリアル:Amazon AWS Lambda ロギングフレームワークを使用したアプリケーションログの作成

Amazon CloudWatch Logs を使用して、アプリケーションのログを監視、保存、アクセスできます。ログデータを CloudWatch Logs に取り込むには、 AWS SDK を使用するか、 CloudWatch Logs エージェントをインストールして特定のログフォルダを監視します。 CloudWatch Logs はいくつかの一般的な.NET ロギングフレームワークと統合されているため、ワークフローが簡素化されます。

CloudWatch Logs と .NET ロギングフレームワークを使い始めるには、 NuGet 適切なパッケージと CloudWatch Logs 出力ソースをアプリケーションに追加し、ロギングライブラリを通常どおりに使用してください。これにより、アプリケーションは.NET フレームワークを使用してメッセージをログに記録し、 CloudWatch Logs に送信し、 CloudWatch アプリケーションのログメッセージをログコンソールに表示できるようになります。アプリケーションのログメッセージに基づいて、CloudWatch ログコンソールからメトリクスとアラームを設定することもできます。

サポートされている .NET ロギングフレームワークには以下が含まれます。

- NLog: 表示するには、nuget.org NLog パッケージを参照してください。
- Log4Net: 表示するには、nuget.org Log4Net パッケージを参照してください。
- ASP.NET Core ロギングフレームワーク:表示するには、nuget.org ASP.NET Core ロギングフレームワークパッケージを参照してください。

以下は、AWS.Logger.NLog NuGet パッケージとターゲットをに追加することで、NLog.config CloudWatch ログとコンソールの両方をログメッセージの出力として有効にするファイルの例です。AWS NLog.config

```
<?xml version="1.0" encoding="utf-8" ?>
<nlog xmlns="http://www.nlog-project.org/schemas/NLog.xsd"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      throwExceptions="true">
  <targets>
    <target name="aws" type="AWSTarget" logGroup="NLog.ConfigExample" region="us-
east-1"/>
    <target name="logfile" xsi:type="Console" layout="${callsite} ${message}" />
  </targets>
  <rules>
    <logger name="*" minlevel="Info" writeTo="logfile,aws" />
  </rules>
</nlog>
```

ロギングプラグインはすべてをベースに構築されており、 AWS SDK for .NET SDK AWS と同様のプロセスで認証情報を認証します。次の例は、ロギングプラグインの認証情報が CloudWatch Logs にアクセスするために必要な権限の詳細を示しています。

> **Note**
>
> AWS .NET ロギングプラグインはオープンソースプロジェクトです。追加情報、サンプル、手順については、AWS Logging .NET GitHub リポジトリのサンプルと手順のトピックを参照してください。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups"
      ],
      "Resource": [
        "arn:aws:logs:*:*:*"
      ]
    }
  ]
}
```

AWS へのデプロイ

Toolkit for Visual Studio は、AWS Elastic Beanstalk コンテナまたは AWS CloudFormation スタックへのアプリケーションのデプロイをサポートします。

> **Note**
>
> Visual Studio Express Edition を使用している場合:
>
> - Docker CLI を使用して、アプリケーションを Amazon ECS コンテナにデプロイできます。
> - AWS マネジメントコンソールを使用して、アプリケーションを Elastic Beanstalk コンテナにデプロイできます。
>
> Elastic Beanstalk デプロイでは、まずウェブデプロイパッケージを作成する必要があります。詳細については、「Visual Studio でウェブデプロイパッケージを作成する方法」を参照してください。Amazon ECS デプロイには、Docker イメージが必要です。詳細については、「Visual Studio Tools for Docker」を参照してください。

トピック

への発行AWSVisual で、

への発行AWSは、.NET アプリケーションをAWSデプロイターゲット、.NET Core 3.1 以降をターゲットとするアプリケーションをサポートします。への発行AWS次の配備機能を IDE から直接使用できるようにすることで、ワークフローを Visual Studio 内に保持します。

- ワンクリックでアプリケーションをデプロイする機能。
- アプリケーションに基づいた展開の推奨事項。
- デプロイ先の環境 (デプロイターゲット) に関連して必要な、Dockerfile の自動作成。

- デプロイターゲットの要求に応じて、アプリケーションを構築およびパッケージ化するための最適化された設定。

> **Note**
>
> .NET Framework アプリケーションの公開に関する追加情報については、ガイドを参照してください。Elastic Beanstalk での .NET アプリケーションの作成とデプロイ への発行にアクセスすることもできます。AWS.NET CLI から実行します。詳細については、「」を参照してください。.NET アプリケーションをAWSガイド。

トピック

前提条件

.NET アプリケーションを正常に公開するにはAWSサービスで、ローカルデバイスに以下をインストールします。

- .NET コア 3.1+ (.NET5 と.NET6 を含む): これらの製品の追加情報およびダウンロード情報については、マイクロソフトのダウンロードサイト。
- Node.js 14.x 以降のバージョン: 実行するには Node.js が必要ですAWS Cloud Development Kit (AWS CDK)。Node.js に関する詳細情報をダウンロードまたは入手するには、Node.js ダウンロードサイト。

  > **Note**
  >
  > への発行AWS活用するAWS CDKアプリケーションとそのすべてのデプロイインフラストラクチャを 1 つのプロジェクトとしてデプロイします。の詳細AWS CDK「」を参照してください。Cloud Development Kitガイド。

- (オプション) Docker は、Amazon ECS などのコンテナベースのサービスにデプロイするときに使用されます。Docker の詳細および Docker のダウンロードについては、「」を参照してください。Docker のダウンロードサイト。

サポートされるアプリケーションタイプ

新しいターゲットまたは既存のターゲットにパブリッシュする前に、Visual Studio で次のいずれかのプロジェクトタイプを作成するか開くことから始めます。

- ASP.NET Core Applications
- .NET コンソールアプリケーション
- Blazor WebAssembly 応用

へのアプリケーションの公開AWSターゲット

新しいターゲットにパブリッシュする場合、パブリッシュ先AWS[] は、推奨を行い、共通の設定を使用して、プロセスを案内します。以前に設定したターゲットにパブリッシュする必要がある場合、プリファレンスは保存され、調整できるか、ワンクリックデプロイですぐに使用できます。

新しいターゲットへの発行

以下に、への発行の設定方法について説明します。AWS新しいターゲットに公開するときのデプロイメント設定。

1. 「」からのAWSエクスプローラ[] で、[認証情報[] ドロップダウンメニューから [] を選択し、AWS地域に対応するプロファイルとAWS展開に必要なサービス。
2. を拡張します。リージョン[] ドロップダウンメニューから [] を選択し、AWSを含むリージョンAWS展開に必要なサービス。
3. Visual Studio から、ソリューションExplorerペインで、プロジェクト名のコンテキストメニューを開き、への発行AWS。これが開きます。への発行AWS。
4. 送信元への発行AWS、選択新しいターゲットへの発行新しいデプロイを設定します。

   > **Note**
   >
   > デフォルトのデプロイメント認証情報を変更するには、編集の隣にあるリンク認証情報セクションで、への発行AWS。
   > ターゲット設定プロセスをバイパスするには、既存のターゲットに公開を選択し、以前のデプロイターゲットのリストから希望する設定を選択します。

5. 「」からのターゲットの公開ペインで、AWSサービスを使用して、アプリケーションのデプロイを管理します。

6. 設定内容が適切であることを確認できたら、[] を選択します。公開デプロイプロセスを開始します。

> **Note**
>
> デプロイを開始した後、への発行AWSには、次のステータス更新が表示されます。
>
> - デプロイプロセス中、への発行AWSデプロイの進捗に関する情報が表示されます。
> - デプロイプロセスに従って、への発行AWSデプロイが成功したか失敗したかを示します。
> - 展開が成功すると、リソースパネルには、作成されたリソースに関する追加情報が表示されます。この情報は、アプリケーションのタイプとデプロイメント構成のタイプに応じて変わります。

既存のターゲットへの発行

次に、.NET アプリケーションを既存のAWSターゲット。

1. 「」からのAWSエクスプローラ[] で、[認証情報[] ドロップダウンメニューから [] を選択し、AWS地域に対応するプロファイルとAWS展開に必要なサービス。
2. を拡張します。リージョン[] ドロップダウンメニューから [] を選択し、AWSを含むリージョンAWS展開に必要なサービス。
3. Visual Studio から、ソリューションExplorerペインで、プロジェクトの名前を右クリックして [] を選択し、への発行AWSを開くにはへの発行AWS。
4. 送信元への発行AWS、選択既存のターゲットに公開既存のターゲットのリストからデプロイ環境を選択します。

> **Note**
>
> 最近、アプリケーションをAWSクラウド、それらのアプリケーションは [公開先] に表示されますAWS。

5. アプリケーションのデプロイ先の公開ターゲットを選択し、[] をクリックします。公開デプロイプロセスを開始します。

へのデプロイAWS Lambda.NET Core CLI を使用したプロジェクト

AWS Toolkit for Visual Studio には Visual Studio 用の AWS Lambda .NET Core プロジェクトテンプレートが含まれています。.NET コアコマンドラインインターフェイス (CLI) を使用して、Visual Studio に組み込まれた Lambda 関数をデプロイできます。

トピック

前提条件

.NET Core CLI を使用して Lambda 関数をデプロイする前に、次の前提条件を満たす必要があります。

- Visual Studio 2015 Update 3 がインストールされていることを確認してください。
- .NET Core for Widnows をインストールします。
- .NET Core CLI をセットアップして Lambda を操作します。詳細については、AWS Lambda デベロッパーガイドの「 NET Core CLI」を参照してください。
- Toolkit for Visual Studio をインストールします。詳細については、「をインストールする AWS Toolkit for Visual Studio」を参照してください。

関連トピック

.NET Core CLI を使用して Lambda 関数をデプロイする場合、次の関連トピックが役立ちます。

- Lambda 関数 の詳細については、AWS Lambda Lambda デベロッパーガイドの「AWS Lambda とは？」を参照してください。
- Visual Studio で Lambda 関数を作成する方法については、「AWS Lambda」を参照してください。
- Microsoft .NET Core の詳細については、Microsoft オンラインドキュメントの「.NET Core」を参照してください。

.NET Core CLI を介して使用可能な Lambda コマンドを一覧表示する

.NET Core CLI を介して使用できる Lambda コマンドを一覧表示するには、次の手順を実行します。

1. コマンドプロンプトウインドウを開いて、Visual Studio の .NET Core Lambda プロジェクトが含まれるフォルダに移動します。
2. `dotnet lambda --help` と入力します。

```
C:\Lambda\AWSLambda1\AWSLambda1>dotnet lambda --help
    AWS Lambda Tools for .NET Core functions
    Project Home: https://github.com/aws/aws-lambda-dotnet
    .
    Commands to deploy and manage Lambda functions:
    .
            deploy-function         Deploy the project to Lambda
            invoke-function         Invoke the function in Lambda with an optional
 input
            list-functions          List all of your Lambda functions
            delete-function         Delete a Lambda function
            get-function-config     Get the current runtime configuration for a Lambda
 function
            update-function-config  Update the runtime configuration for a Lambda
 function
    .
    Commands to deploy and manage AWS serverless applications using AWS CloudFormation:
    .
            deploy-serverless       Deploy an AWS serverless application
            list-serverless         List all of your AWS serverless applications
            delete-serverless       Delete an AWS serverless application
    .
    Other Commands:
    .
            package                 Package a Lambda project into a .zip file ready for
 deployment
    .
    To get help on individual commands, run the following:

            dotnet lambda help <command>
```

.NET Core CLI から .NET Core Lambda プロジェクトを発行する

次の手順では、Visual Studio で AWS Lambda .NET Core 関数を作成したと想定しています。

1. コマンドプロンプトウィンドウを開いて、Visual Studio の .NET Core Lambda プロジェクトが含まれるフォルダに移動します。
2. `dotnet lambda deploy-function` と入力します。
3. プロンプトされたら、デプロイする関数の名前を入力します。新しい名前または既存の関数の名前を入力できます。
4. プロンプトが表示されたら、AWS リージョン (Lambda 関数のデプロイ先になるリージョン) を入力します。
5. プロンプトがされたら、関数が実行されるときに Lambda が継承する IAM ロールを選択または作成します。

正常に完了すると、[New Lambda function created (新しい Lambda 関数が作成されました)] のメッセージが表示されます。

```
C:\Lambda\AWSLambda1\AWSLambda1>dotnet lambda deploy-function
Executing publish command
... invoking 'dotnet publish', working folder 'C:\Lambda\AWSLambda1\AWSLambda1\bin
\Release\netcoreapp1.0\publish'
... publish: Publishing AWSLambda1 for .NETCoreApp,Version=v1.0
... publish: Project AWSLambda1 (.NETCoreApp,Version=v1.0) will be compiled because
 expected outputs are missing
... publish: Compiling AWSLambda1 for .NETCoreApp,Version=v1.0
... publish: Compilation succeeded.
... publish:     0 Warning(s)
... publish:     0 Error(s)
... publish: Time elapsed 00:00:01.2479713
... publish:
... publish: publish: Published to C:\Lambda\AWSLambda1\AWSLambda1\bin\Release
\netcoreapp1.0\publish
... publish: Published 1/1 projects successfully
Zipping publish folder C:\Lambda\AWSLambda1\AWSLambda1\bin\Release
\netcoreapp1.0\publish to C:\Lambda\AWSLambda1\AWSLamb
da1\bin\Release\netcoreapp1.0\AWSLambda1.zip
Enter Function Name: (AWS Lambda function name)
DotNetCoreLambdaTest
Enter AWS Region: (The region to connect to AWS services)
us-west-2
```

```
Creating new Lambda function
Select IAM Role that Lambda will assume when executing function:
    1) lambda_exec_LambdaCoreFunction
    2) *** Create new IAM Role ***
1
New Lambda function created
```

既存の関数をデプロイする場合、デプロイ機能は AWS リージョンのみを求めます。

```
C:\Lambda\AWSLambda1\AWSLambda1>dotnet lambda deploy-function
Executing publish command
Deleted previous publish folder
... invoking 'dotnet publish', working folder 'C:\Lambda\AWSLambda1\AWSLambda1\bin
\Release\netcoreapp1.0\publish'
... publish: Publishing AWSLambda1 for .NETCoreApp,Version=v1.0
... publish: Project AWSLambda1 (.NETCoreApp,Version=v1.0) was previously compiled.
 Skipping compilation.
... publish: publish: Published to C:\Lambda\AWSLambda1\AWSLambda1\bin\Release
\netcoreapp1.0\publish
... publish: Published 1/1 projects successfully
Zipping publish folder C:\Lambda\AWSLambda1\AWSLambda1\bin\Release
\netcoreapp1.0\publish to C:\Lambda\AWSLambda1\AWSLamb
da1\bin\Release\netcoreapp1.0\AWSLambda1.zip
Enter Function Name: (AWS Lambda function name)
DotNetCoreLambdaTest
Enter AWS Region: (The region to connect to AWS services)
us-west-2
Updating code for existing function
```

デプロイされた Lambda 関数は、すぐに使用できる状態になっています。詳細については、「AWS Lambda の使用例」を参照してください。

Lambda は、ユーザーに代わって Lambda 関数を自動的にモニタリングし、Amazon からメトリクスを報告します CloudWatch。Lambda 関数をモニタリングおよびトラブルシューティングする方法については、「」を参照してください。トラブルシューティングとモニタリングAWSLambda 関数 with Amazon CloudWatch。

Elastic Beanstalk へのデプロイ

AWS Elastic Beanstalk は、アプリケーションの AWS リソースのプロビジョニングプロセスを簡素化するサービスです。アプリケーションのデプロイに必要な AWS インフラストラクチャは、Elastic Beanstalk がすべて提供します。このインフラストラクチャには次のものが含まれます。

- 実行可能ファイルとアプリケーションのコンテンツをホストする Amazon EC2 インスタンス。
- Amazon EC2 インスタンスの数を適切に維持することでアプリケーションをサポートする Auto Scaling グループ。
- 受信トラフィックを帯域幅が最も広い Amazon EC2 インスタンスにルーティングする Elastic Load Balancing ロードバランサー。

Toolkit for Visual Studio には、Amazon ECS によるアプリケーションの発行を簡素化するウィザードが用意されています。このウィザードについては、次のセクションで説明します。

Elastic Beanstalkの詳細については、Elastic Beanstalk ドキュメントを参照してください。

トピック

.Traditional ASP NET アプリケーションを Elastic Beanstalk にデプロイします。

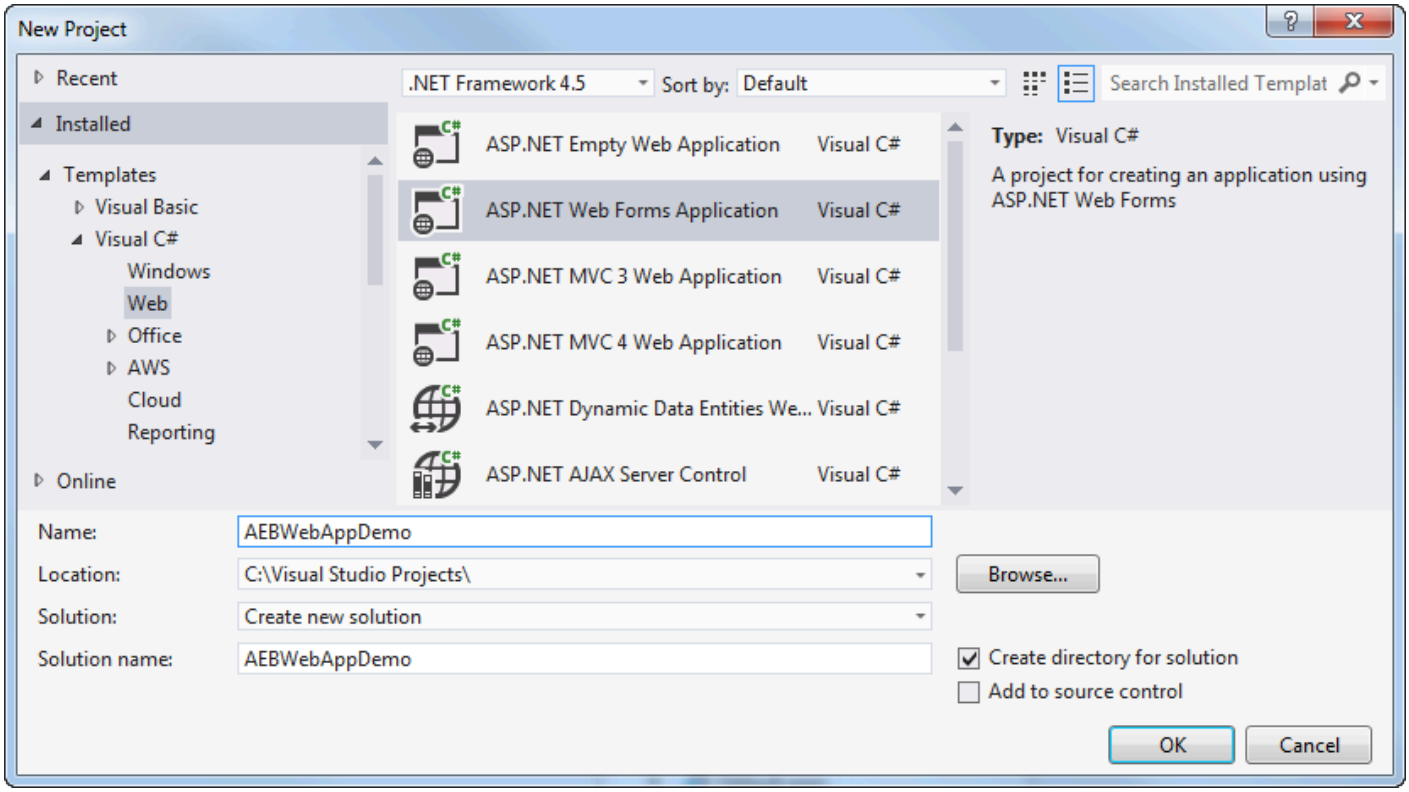
このセクションでは、 Elastic Beanstalk を介してアプリケーションをデプロできるように、Toolkit for Visual Studio の一部として提供される [Publish to Elastic Beanstalk] (Elastic Beanstalk への発行) ウィザードの使用方法を記述します。演習を行う場合は、ウェブアプリケーションのスタータープロジェクトのインスタンスを使うことができます。これは Visual Studio に組み込まれています。または独自のプロジェクトを使用することもできます。

> **Note**
>
> ウィザードでは ASP.NET Core アプリケーションのデプロイもサポートしています。ASP.NET Core の詳細については、AWS.NET デプロイツールガイドと更新された「AWSデプロイの目次」を参照してください。

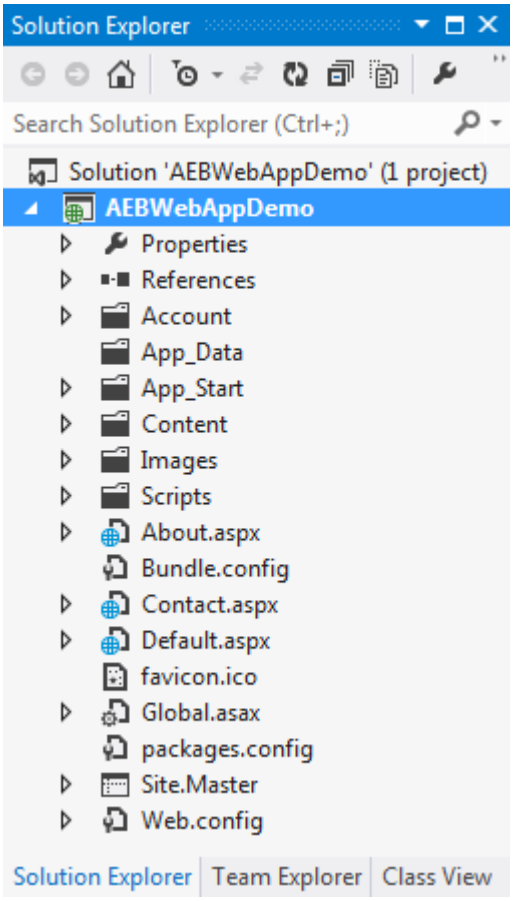
> **Note**
>
> [Publish to Elastic Beanstalk] (Elastic Beanstalk への発行) ウィザードを使用するには、Web Deploy をダウンロードしてインストールする必要があります。ウィザードでは、Web Deploy を用いて Internet Information Services (IIS) ウェブサーバーにウェブアプリケーションやウェブサイトをデプロイします。

サンプルウェブアプリケーションのスタータープロジェクトを作成するには

1. Visual Studio の [File (ファイル)] メニューで [New (新規)] を選択し、[Project (プロジェクト)] を選択します。
2. [New Project (新規プロジェクト)] ダイアログボックスのナビゲーションペインで、[Installed (インストール済み)]、[Templates (テンプレート)]、[Visual C#] の順に展開し、[Web] を選択します。
3. ウェブプロジェクトテンプレートのリストの中から、`Web` と `Application` が説明に含まれるテンプレートのいずれかを選択します。この例では、[ASP.NET Web Forms Application (ASP.NET Web Forms アプリケーション)] を選択します。

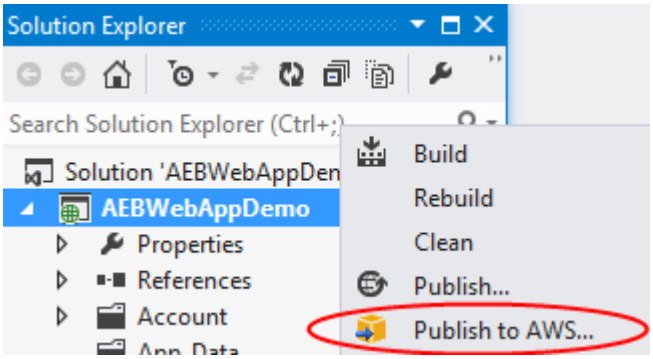


4. [Name (名前)] ボックスに、「AEBWebAppDemo」と入力します。
5. [Location] ボックスで、開発マシンのソリューションフォルダへのパスを入力するか、[Browse] を選択し、ソリューションフォルダを参照して選択し、[Select Folder] を選択します。
6. [Create directory for solution (ソリューションのディレクトリの作成)] が選択されていることを確認します。[Solution] (ソリューション) ドロップダウンリストで、[Create new solution] (新しいソリューションの作成) が選択されていることを確認し、[OK] を選択します。Visual Studio を用いて、ASP.NET Web Forms Application プロジェクトテンプレートをベースにしたソリューションやプロジェクトを作成できます。新しいソリューションやプロジェクトが表示される Solution Explorer が、Visual Studio の画面に表示されます。

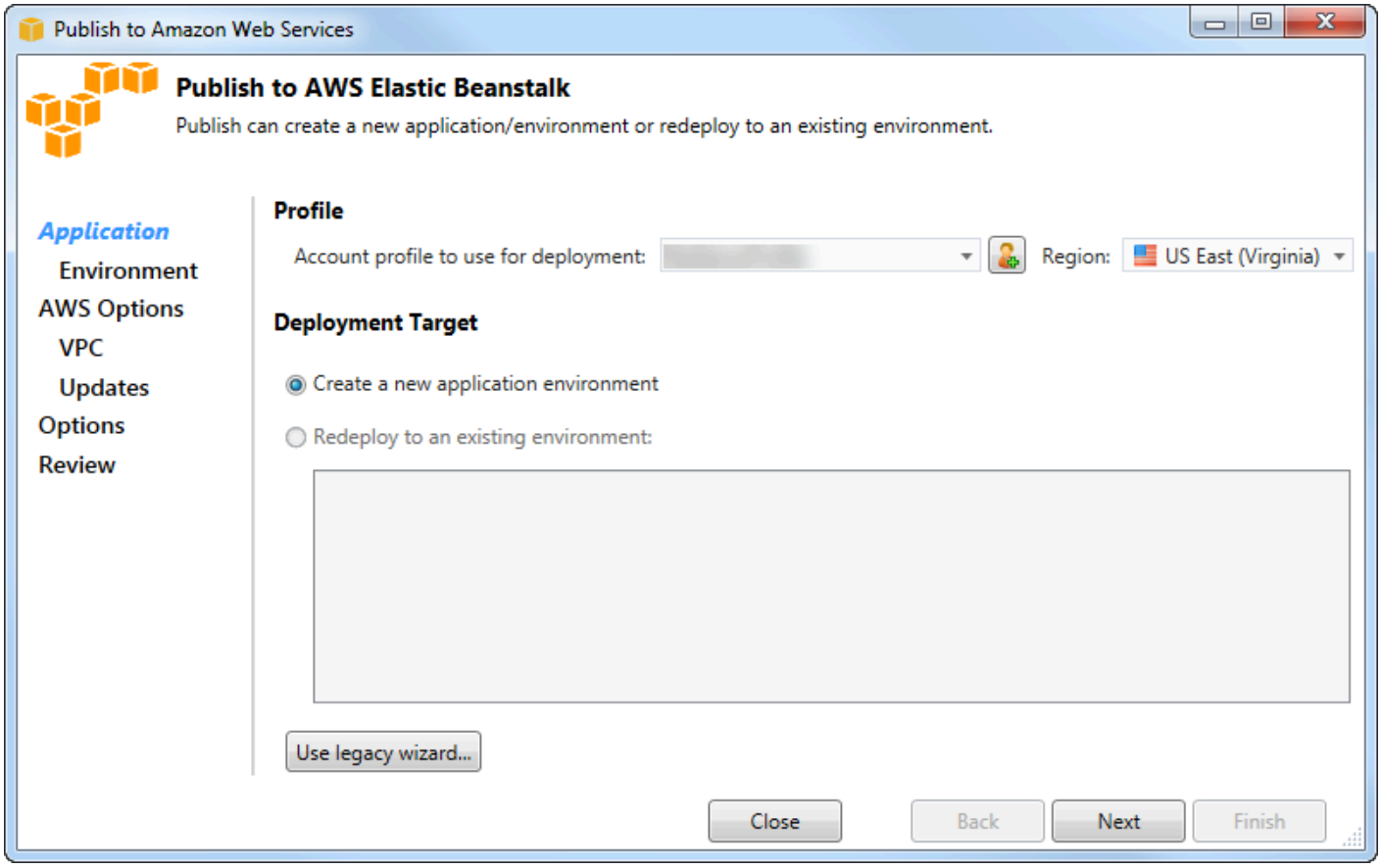


[Publish to Elastic Beanstalk (Elastic Beanstalk への発行)] ウィザードを使用してアプリケーションをデプロイするには

1. Solution Explorer で、前のセクションで作成したプロジェクトの AEBWebAppDemo プロジェクトフォルダのコンテキストメニュー (右クリック) するか、またはユーザーが作成したアプリケーションのプロジェクトフォルダのコンテキストメニューを開き、[Publish toAWS Elastic Beanstalk] (Elastic Beanstalk への発行) を選択します。



   [Publish to Elastic Beanstalk] (Elastic Beanstalk への発行) ウィザードが表示されます。



2. [Profile] (プロファイル) の [Account profile to use for deployment] (デプロイに使用するアカウントプロファイル) ドロップダウンリストで、デプロイに使用する AWS アカウントプロファイルを選択します。

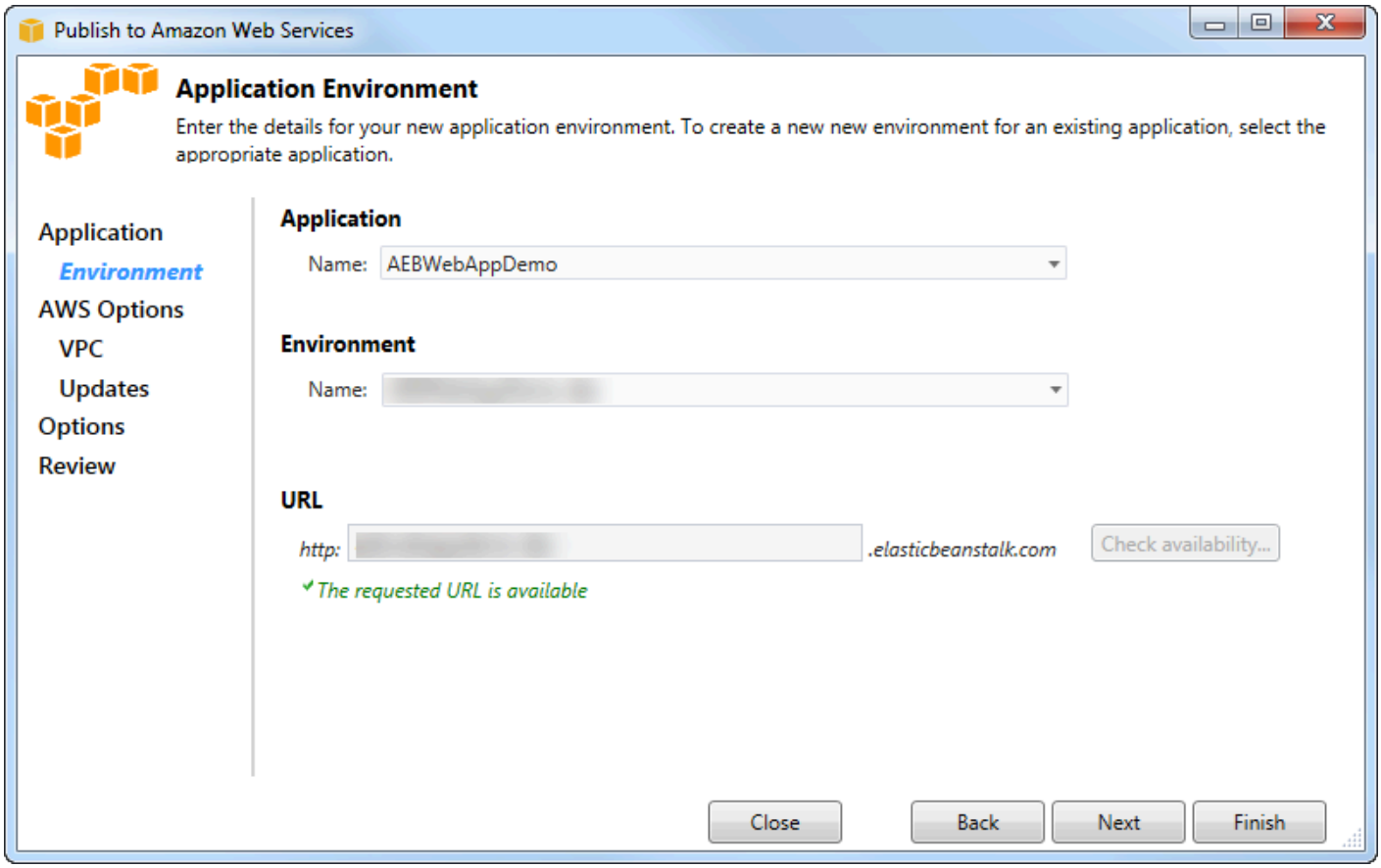
   使用したい AWS アカウントはあるけれども AWS アカウントプロファイルをまだ作成していない場合、オプションとして、プラス記号 (+) のボタンを選択して AWS アカウントのプロファイルを追加できます。

3. [Region] (リージョン) ドロップダウンリストから、Elastic Beanstalkをアプリケーションにデプロイしたいリージョンを選択します。

4. [Deployment Target] (デプロイ先) では、[Create a new application environment] (新しいアプリケーション環境を作成) を選択してアプリケーションの初期デプロイを行うことも、[Redeploy to an existing environment] (既存の環境に再デプロイ) を選択してデプロイ済みアプリケーションの再デプロイを行うかこともできます。(前回のデプロイをウィザードまたは [Standalone Deployment Tool] (スタンドアロンデプロイツール) のどちらで実行していてもかまいません。) [Redeploy to an existing environment] を選択した場合は、現在稼働中の前回のデプロイに関する情報をウィザードが取得するのに時間を要することがあります。

> **Note**
>
> [Redeploy to an existing environment (既存の環境に再デプロイする)] を選択した場合は、リストの中から環境を選択し、[Next (次へ)] を選択すると、[Application Options (アプリケーションオプション)] ページを直接表示できます。この方法でページを表示した場合は、本セクションで口述する [Application Options (アプリケーションオプション)] ページの使用方法について説明している箇所までスキップしてください。

5. [Next] (次へ) を選択します。

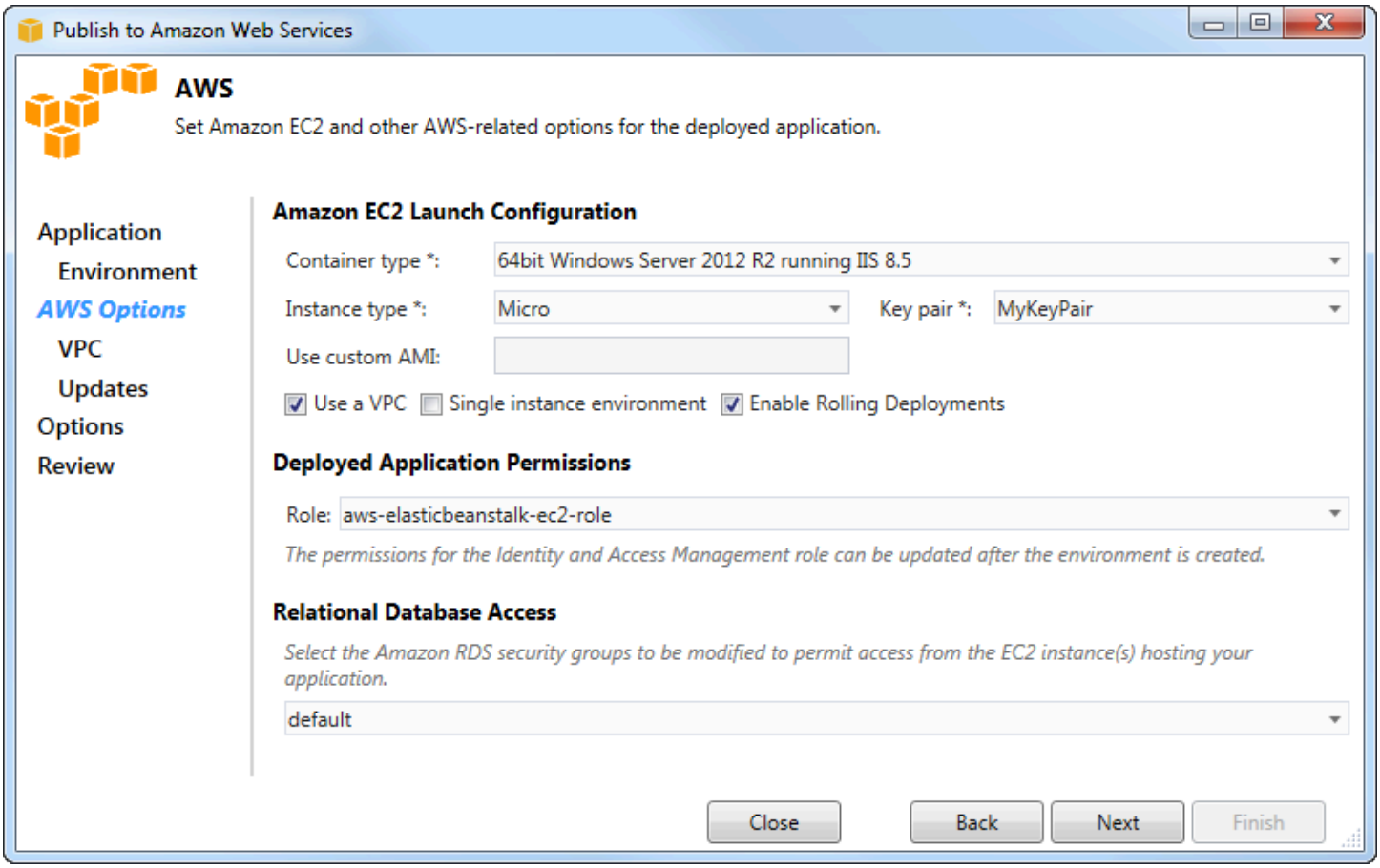


6. [Application Environment] (アプリケーション環境) ページの [Application] (アプリケーション) エリアにある [Name] (名前) ドロップダウンリストに、推奨されるデフォルトのアプリケーション名が表示されます。ドロップダウンリストの別の名前を選択することで、デフォルト名を変更できます。

7. [Environment] (環境) エリアの [Name] (名前) ドロップダウンリストで、Elastic Beanstalk 環境の名前を入力します。このコンテキストでは、環境という用語は、アプリケーションのインフラストラクチャ Elastic Beanstalk プロビジョニングを意味します。推奨されるデフォルト名がドロップダウンリストに既に表示されている場合があります。推奨されるデフォルト名が表示されてい

ない場合は自身で入力するか、他に選択候補がある場合は、ドロップダウンリストからいずれか 1 つを選択します。環境の名前は 23 文字より長くすることはできません。

8. [URL] エリアでは、ウェブアプリケーションの URL として使用される、デフォルトの `.elasticbeanstalk.com` のサブドメイン候補が表示されます。新しいサブドメイン名を入力することで、デフォルトのサブドメインを変更できます。
9. [Check availability] を選択して、ウェブアプリケーションの URL が使用されていないことを確認します。
10. ウェブアプリケーションの URL が使用可能な場合は、[Next] を選択します。



1. [AWS Options] (オプション) ページの [Amazon EC2 Launch Configuration] (Amazon EC2 起動設定) にある [Container type] (コンテナタイプ) ドロップダウンリストから、アプリケーションで使用する Amazon Machine Image (AMI) を選択します。
2. [Instance type] (インスタンスタイプ) ドロップダウンリストで、使用する Amazon EC2 インスタンスタイプを指定します。この例では、[Micro (マイクロ)] の使用をお勧めします。これにより、

インスタンスの実行に関連するコストが最小化されます。Amazon EC2 の料金の詳細については、「EC2 料金」のページを参照してください。

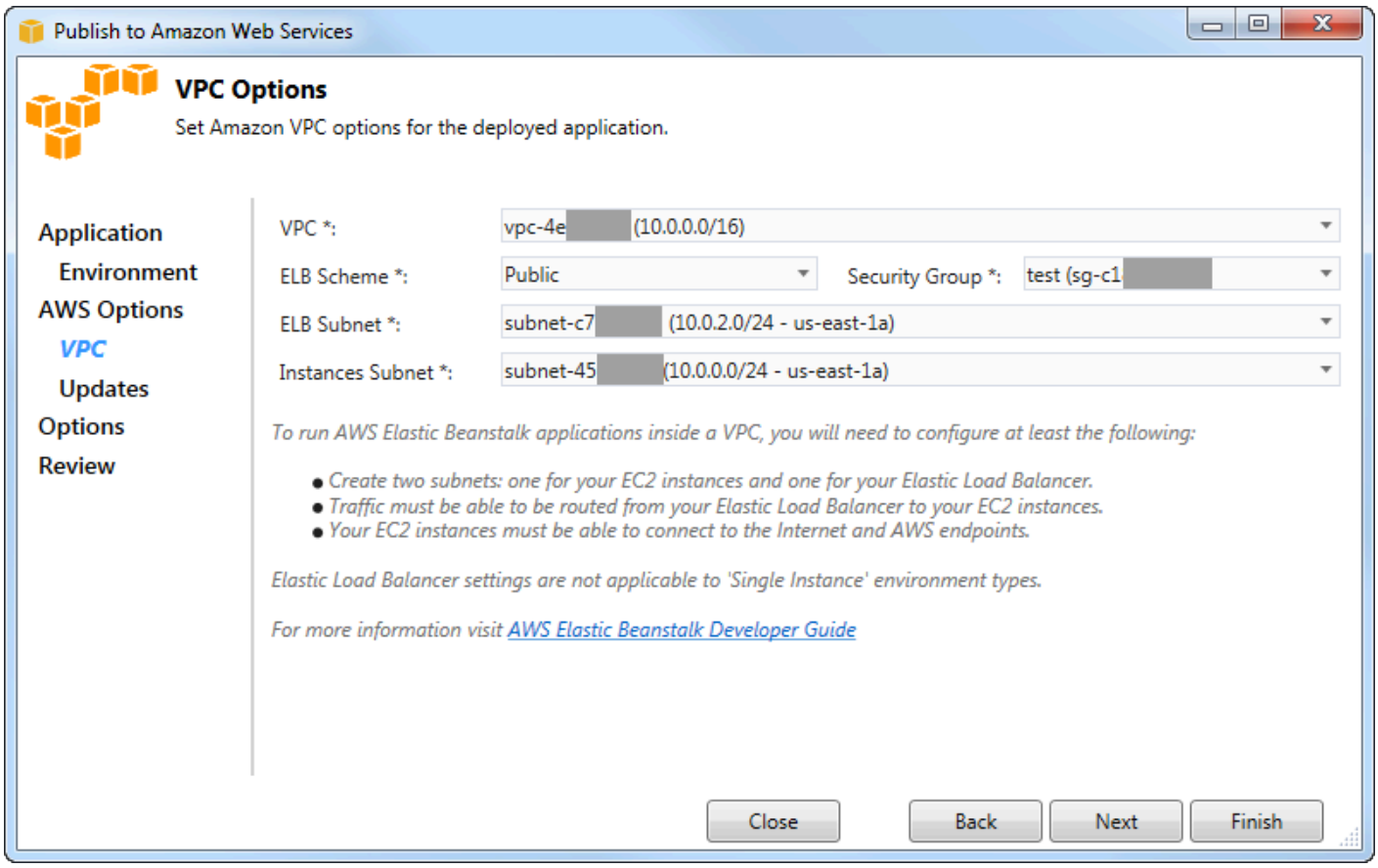
3. [Key pair] (キーペア) ドロップダウンリストで、アプリケーションで使用するインスタンスにサインインする際の Amazon EC2 インスタンスキーペアを選択します。
4. オプションとして、[Use custom AMI] (カスタム AMI を使用) ボックスで、[Container type] (カスタムタイプ) ドロップダウンリストで指定した AMI を上書きするカスタム AMI を指定できます。カスタム AMI の作成方法の詳細については、AWS Elastic Beanstalk デベロッパーガイドの「カスタム AMI の使用」と「Amazon EC2 インスタンスからの AMI の作成」を参照してください。
5. VPC 内でインスタンスを起動する場合は、[Use a VPC] ボックスを選択します。
6. オプションとして、Amazon EC2 インスタンスを 1 つだけ起動してアプリケーションをそのインスタンスにデプロイしたい場合、[Single instance environment] (単一インスタンス環境) ボックスを選択します。

   このボックスを選択する場合、 Elastic BeanstalkによりAuto Scaling グループが作成されますが、設定はされません。Auto Scaling グループを後で設定したい場合、AWS Management Console を使用できます。
7. オプションとして、アプリケーションをデプロイするインスタンスを条件で制御する場合は、[Enable Rolling Deployments] (ローリングデプロイを有効にする) ボックスを選択します。このボックスは [Single instance environment (単一のインスタンス環境)] ボックスを選択していない場合のみ選択できます。
8. アプリケーションで Amazon S3 および DynamoDB などの AWS のサービスを使用している場合、認証情報を提供する最適な方法は、IAM ロールを使用することです。[Deployed Application Permissions] (デプロイ済みアプリケーションアクセス許可) エリアでは、ウィザードが環境を起動する際に使用するロールとして既存の IAM ロールを選択するか、または新しくロールを作成することができます。AWS SDK for .NET を使用するアプリケーションは、AWS のサービスにリクエストを送る際に、ロールが提供する認証情報を自動的に使用します。
9. アプリケーションから Amazon RDS データベースにアクセスしている場合は、[Relational Database Access] エリアのドロップダウンリストで、ウィザードが更新する Amazon RDS セキュリティグループの横のボックスを選択することで、Amazon EC2 インスタンスからそのデータベースにアクセスできます。
10. [Next] (次へ) を選択します。
    - [Use a VPC] を選択した場合は、[VPC Options] ページが表示されます。
    - [Enable Rolling Deployments] (ローリングデプロイを有効にする) を選択し、かつ [Use a VPC] (VPC を使用) を選択しなかった場合は、[Rolling Deployments] ページが表示されます。本セク

ションで後述している [Rolling Deployments (ローリングデプロイ)] ページの使用方法について説明している箇所までスキップしてください。

- [Use a VPC] (VPC を使用) および [Enable Rolling Deployments] (ローリングデプロイを有効にする) を選択しなかった場合は、[Application Options] (アプリケーションオプション) ページが表示されます。本セクションで後述されている [Application Options (アプリケーションオプション)] ページの使用方法について説明している箇所までスキップしてください。

11[Use a VPC] を選択した場合は、[VPC Options] ページの情報を指定して VPC でアプリケーションを起動します。



VPC は既に作成されている必要があります。Toolkit for Visual Studio で VPC を作成した場合、Toolkit for Visual Studio によってこのページが自動的に設定されます。AWS マネジメントコンソールで VPC を作成した場合、VPC に関する情報をこのページに入力します。

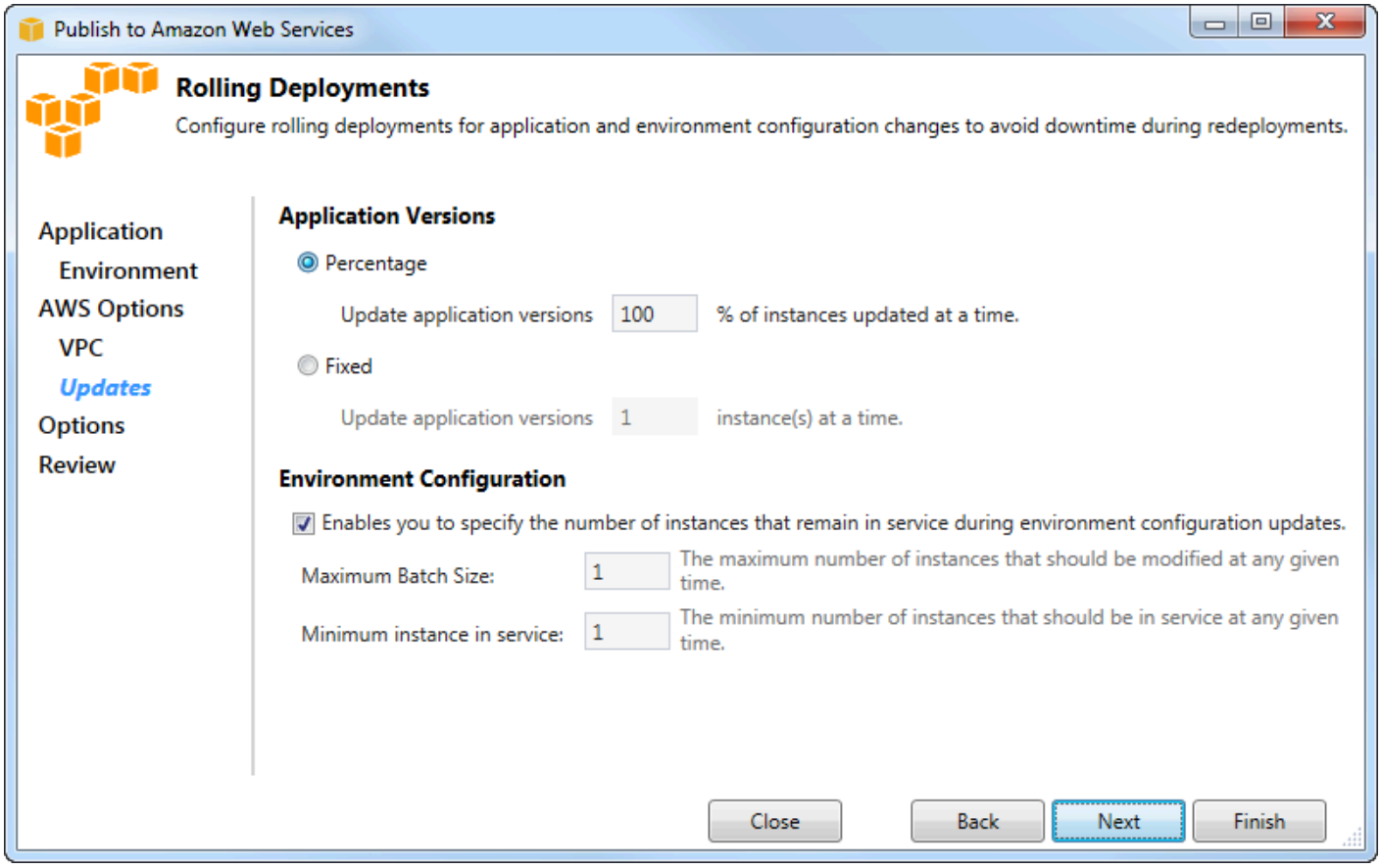
VPC へのデプロイに際して特に考慮すべき事項

- VPC には、少なくとも 1 つのパブリックサブネットと 1 つのプライベートサブネットが必要です。

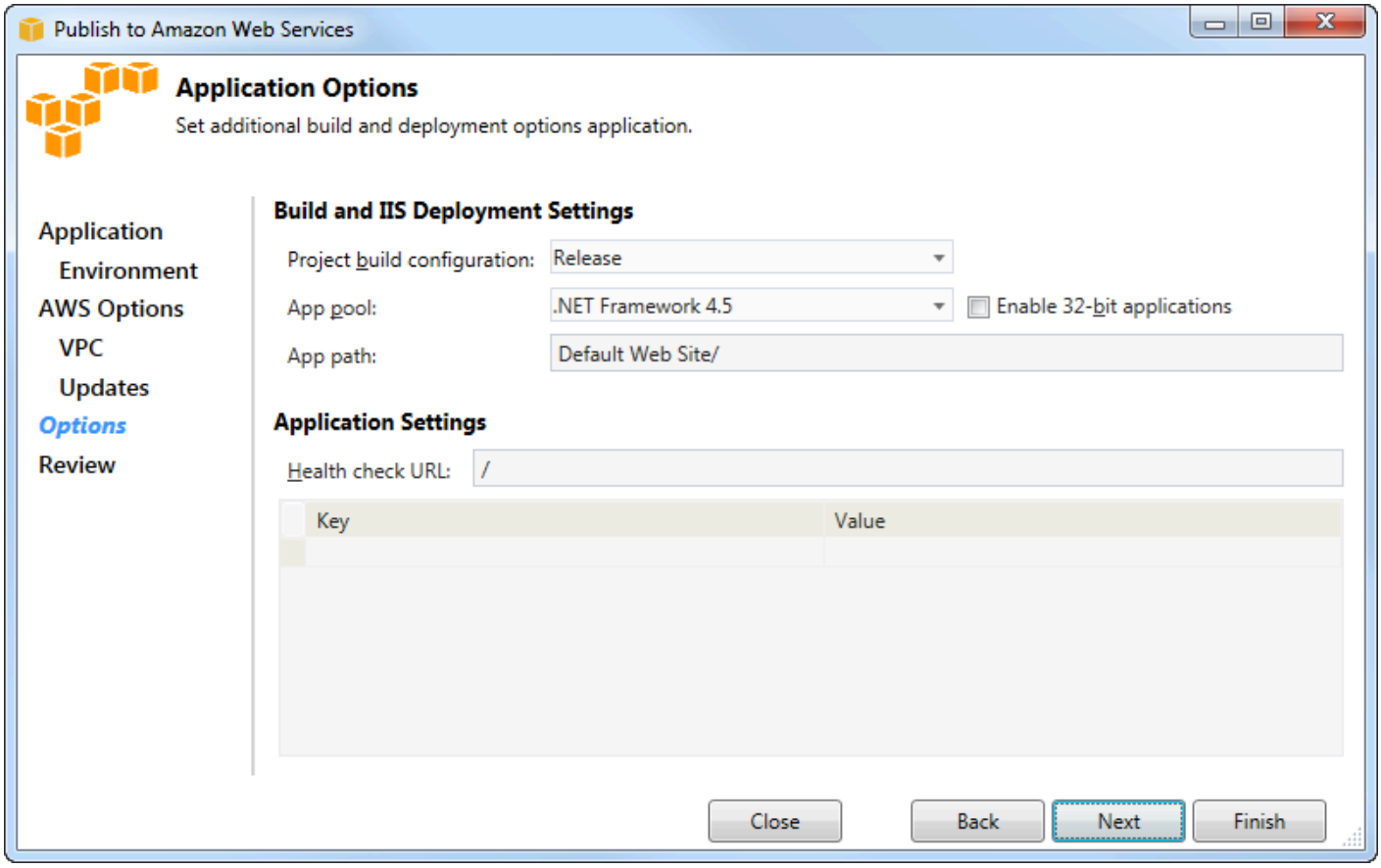
- [ELB Subnet] (ELB サブネット) ドロップダウンリストでパブリックサブネットを設定します。Toolkit for Visual Studio は、アプリケーションの Elastic Load Balancing ロードバランサーをパブリックサブネットに向けてデプロイします。パブリックサブネットは、インターネットゲートウェイを指すエントリが含まれているルーティングテーブルに関連付けられています。インターネットゲートウェイは `igw-` で始まる ID を持つため (例 :`igw-83cddaex`)、インターネットゲートウェイを認識することができます。Toolkit for Visual Studio を使用して作成したパブリックサブネットには、パブリックとして識別するタグ値が付与されています。
- [Instances Subnet] (インスタンスサブネット) ドロップダウンリストでプライベートサブネットを設定します。Toolkit for Visual Studio はアプリケーションのAmazon EC2 Eインスタンスをプライベートサブネットに向けてデプロイします。
- アプリケーションの Amazon EC2 インスタンスは、ネットワークアドレス変換 (NAT) を行うパブリックサブネットの Amazon EC2 インスタンスを介してプライベートサブネットからインターネットに通信します。通信を可能にするには、プライベートサブネットから NAT インスタンスにトラフィックが流れるようにするための VPC セキュリティグループが必要です。VPC セキュリティグループを [Security Group] ドロップダウンリストから指定します。

Elastic Beanstalkアプリケーションを VPC にデプロイする方法の詳細については、AWS Elastic Beanstalk デベロッパーガイドを参照してください。

1. VPC Options ページですべての情報を入力してから、[Next] を選択します。
   - [Enable Rolling Deployments] を選択した場合は、[Rolling Deployments] ページが表示されます。
   - [Enable Rolling Deployments] (ローリングデプロイを有効にする) を選択しなかった場合は、[Application Options] (アプリケーションオプション) ページが表示されます。本セクションで後述されている [Application Options (アプリケーションオプション)] ページの使用方法について説明している箇所までスキップしてください。
2. [Enable Rolling Deployments] (ローリングデプロイを有効にする) を選択した場合は、[Rolling Deployments] (ローリングデプロイ) ページの情報を設定して、新しいバージョンのアプリケーションをロードバランス環境のインスタンスにデプロイする際の方法を構成します。例えば、環境内に 4 つのインスタンスがあってインスタンスタイプを変更する場合、一度に 2 つのインスタンスを変更するように環境を設定できます。これにより、アプリケーションを停止することなく変更できます。



3. [Application Versions] (アプリケーションのバージョン) エリアで、一度にデプロイするインスタンスを全体の割合にするか実際の数にするかを制御するオプションを選択します。目的のパーセンテージまたは最大数を指定します。
4. オプションとして、デプロイ中にサービスを実行し続けるインスタンス数を指定する場合は、[Environment Configuration] (環境の設定) エリアのボックスを選択します。このボックスを選択した場合は、一度に変更するインスタンスの最大数、サービスを実行し続ける最小のインスタンス数、またはその両方を設定します。
5. [Next] (次へ) を選択します。
6. [Application Options] ページで、構築、Internet Information Services (IIS)、およびアプリケーション設定に関する情報を指定します。



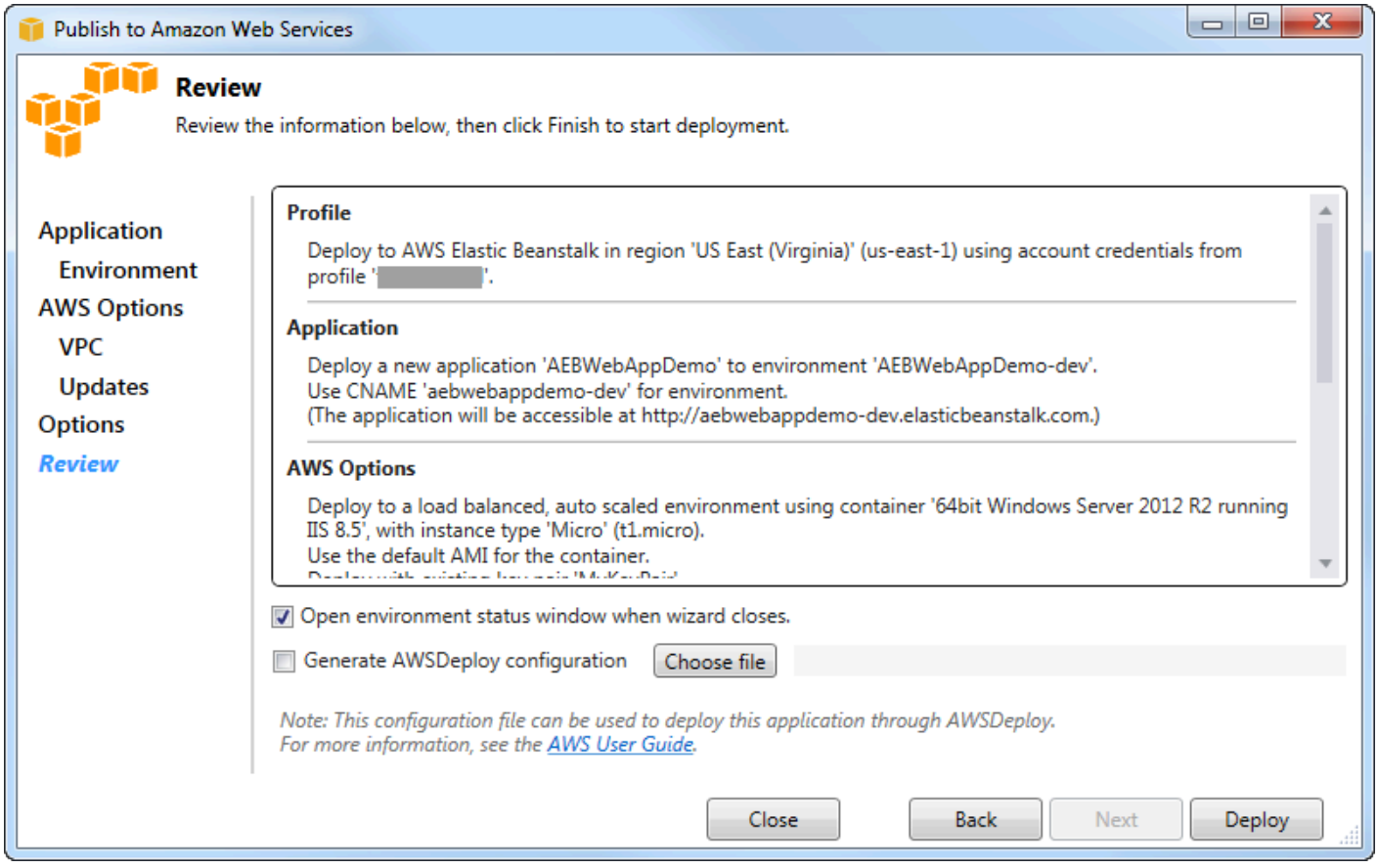
7. [Build and IIS Deployment Settings] (ビルドと IIS デプロイの設定) エリアの [Project build configuration] (プロジェクトビルド設定) ドロップダウンリストで、ターゲットビルド設定を選択します。ウィザードが発見できた場合は [Release (リリース)] が、発見できなかった場合はアクティブ設定がこのボックスに表示されます。
8. [App pool] (アプリケーションプール) ドロップダウンリストで、アプリケーションが必要とする .NET Framework のバージョンを選択します。適切な .NET Framework のバージョンが表示されています。
9. アプリケーションが 32 ビットの場合は、[Enable 32-bit applications] ボックスを選択します。
10. [App path] (アプリケーションパス) ボックスで、アプリケーションのデプロイに使用する IIS パスを指定します。デフォルトでは、[Default Web Site/ (デフォルトウェブサイト/)] が指定され、通常は `c:\inetpub\wwwroot` に変換されます。[Default Web Site (デフォルトウェブサイト)] 以外のパスを指定した場合は、ウィザードは指定したパスを指すリダイレクトを [Default Web Site/ (デフォルトウェブサイト/)] に配置します。
11. [Application Settings] (アプリケーションの設定) エリアの [Health check URL] (ヘルスチェック URL) ボックスに、ウェブアプリケーションが応答可能かを判別する際に確認する Elastic Beanstalk の URL を入力します。この URL はルートサーバー URL への相対 URL です。デフォ

ルトではルートサーバーの URL が指定されます。例えば、完全な URL が `example.com/site-is-up.html` である場合は `/site-is-up.html` と入力します。

12[Key] と [Value] には、アプリケーションの `Web.config` ファイルに追加したい任意のキーと値のペアを指定できます。

> **Note**
>
> 推奨はされませんが、[Key] (キー) と [Value] (値) のエリアを使用してアプリケーションの実行に必要な AWS 認証情報を設定できます。推奨されるのは、[AWS Options] (オプション) ページの [Identity and Access Management Role] ドロップダウンリストで IAM ロールを指定する方法です。ただし、アプリケーションの実行に IAM ロールではなくAWS認証情報が必要な場合、[Key] (キー) 列でを選択しますAWSAccessKey。[Value (値)] 行にアクセスキーを入力します。についても、AWSSecretKey上記の手順を繰り返します。

13[Next] (次へ) を選択します。



14[Review] ページで、選択したオプションを確認し、[Open environment status window when wizard closes] ボックスを選択します。

15.すべてが正しい場合は、[Deploy (デプロイ)] を選択します。

> **Note**
>
> アプリケーションをデプロイすると、アプリケーションが使用する AWS リソースの料金が発生し、それはアクティブなアカウントに請求されます。

Visual Studio ステータスバーおよび [Output] (出力) ウィンドウに、デプロイに関する情報が表示されます。これには数分間かかる場合があります。デプロイが完了すると、確認メッセージが [Output (出力)] ウィンドウに表示されます。

16. デプロイを削除するには、AWS Explorer で [Elastic Beanstalk] ノードを展開し、デプロイ対象のサブノードのコンテキスト (右クリック) メニューを開いて [Delete] (削除) を選択します。削除には数分かかる場合があります。

Elastic Beanstalk を使用した ASP.NET Core アプリケーションのデプロイ (レガシー)

> **Important**
>
> このドキュメントでは、従来のサービスと機能について説明しています。更新されたガイドとコンテンツについては、AWS.NET デプロイツールガイドと更新された Deploying toAWS の目次を参照してください。

AWS Elastic Beanstalk は、アプリケーションの AWS リソースのプロビジョニングプロセスを簡素化するサービスです。 AWS Elastic Beanstalk では、アプリケーションのデプロイに必要なすべての AWS インフラストラクチャを提供します。

AWS Toolkit for Visual Studioは、 Elastic Beanstalkを使用して、ASP.NET Core アプリケーションをデプロイするのをサポートします。ASP.NET Core は、ASP.NET をモジュラー型アーキテクチャで再設計したもので、依存関係のオーバーヘッドを最小限に抑え、アプリケーションのクラウド実行を合理化します。

AWS Elastic Beanstalk ではアプリケーションをさまざまな言語で AWS にデプロイすることが簡単になります。EElastic Beanstalk は従来の ASP.NET アプリケーションおよび ASP.NET Core アプリケーションの両方をサポートしています。このトピックでは、ASP.NET Core アプリケーションのデプロイについて説明します。

デプロイウィザードを使用する

ASP.NET Core アプリケーションを Elastic Beanstalk にデプロイする最も簡単な方法は、Toolkit for VToolkit for VisualToolkit for Visual Studio を使用します。

以前にツールキットを使用して、従来の ASP.NET アプリケーションをデプロイしたことがある場合、ASP.NET Core の場合もよく似ていることがわかると思います。以下の手順では、デプロイをウォークスルーで体験します。

ツールキットを使用したことがない場合、ツールキットのインストール後、最初に実行する必要があるのは、ツールキットに AWS 認証情報を登録することです。詳細な方法については、「Visual Studio 用ドキュメントの「アプリケーションに AWS セキュリティの認証情報を指定する方法」を参照してください。

ASP.NET Core ウェブアプリケーションをデプロイするには、Solution Explorer でプロジェクトを右クリックし、[Publish to AWS...] (... への発行) を選択します。

[Publish to AWS Elastic Beanstalk] (AWS Elastic Beanstalk への発行) デプロイウィザードの最初のページで、新しい Elastic Beanstalk アプリケーションの作成を選択します。Elastic Beanstalk アプリケーションは、Elastic Beanstalk コンポーネントの論理コレクションで、環境、バージョン、環境設定などがあります。デプロイウィザードでアプリケーションが生成されます。このアプリケーションには、さらにアプリケーションバージョンと環境のコレクションが含まれます。環境には、アプリケーションバージョンを実行する、実際の AWS リソースが含まれています。アプリケーションをデプロイするたびに、新しいアプリケーションバージョンが作成され、ウィザードでそのバージョンが環境に指定されます。これらの概念の詳細については、「Elastic Beanstalk コンポーネント」を参照してください。

次に、アプリケーションとその最初の環境の名前を設定します。各環境には一意の CNAME が関連付けられていて、デプロイが完了したときに、アプリケーションにアクセスするためにこの名前を使用できます。

次のページ [AWS Options] (オプション) では、使用する AWS リソースのタイプを設定できます。この例では、[Key pair (キーペア)] セクションを除いて、デフォルト値のままにします。キーペアにより、Windows 管理者パスワードを取得できます。そのため、該当マシンにログオンできます。キーペアをまだ作成していない場合は、[Create new key pair (新しいキーペアの作成)] を選択する必要があります。

許可

[Permissions] (アクセス許可) ページを使用して、アプリケーションを実行している EC2 インスタンスに AWS 認証情報を割り当てます。これはユーザーのアプリケーションが AWS SDK for .NET を使用して他の AWS のサービスにアクセスする場合に重要です。アプリケーションから他のサービスを使用していない場合、このページの設定をデフォルトのまま利用できます。

アプリケーションオプション

[Application Options (アプリケーションオプション)] ページの詳細は、従来の ASP.NET アプリケーションのデプロイ時に指定するものとは異なります。ここでは、アプリケーションをパッケージするために使用するビルド設定とフレームワークを指定し、アプリケーション用の IIS リソースパスも指定します。

[Application Options] ページを完了した後、[Next (次へ)] をクリックして設定を確認し、[Deploy (デプロイ)] をクリックして、デプロイプロセスを開始します。

環境ステータスをチェックする

アプリケーションは、パッケージ化され、AWS にアップロードされた後は、Visual Studio の AWS Explorer から環境ステータスビューを開いて、Elastic Beanstalk 環境のステータスを確認できます。

環境がオンラインになると、ステータスバーにイベントが表示されます。すべてが完了すると、環境のステータスが正常な状態に移行します。URL をクリックして、サイトを表示することができます。ここから、環境またはリモートデスクトップから、Elastic Beanstalk 環境の一部である Amazon EC2 インスタンスにログを取り込むこともできます。

アプリケーションの最初のデプロイでは、新しい AWS リソースが作成されるので、以降の再デプロイよりも少し時間がかかります。開発中にアプリケーションを繰り返し更新するときは、ウィザードを最初からもう一度実行するか、または、プロジェクトを右クリックし、[Republish (再発行)] オプションを選択することで、簡単に再デプロイできます。

再公開処理では、デプロイウィザードで以前に一通り実行した設定を使用してアプリケーションをパッケージし、アプリケーションバンドルを既存のElastic Beanstalk環境にアップロードします。

アプリケーションに AWS セキュリティの認証情報を指定する方法

-AWSで指定するアカウントElastic Beanstalk への発行ウィザードの操作はAWSウィザードのアカウントを使用して Elastic Beanstalk にデプロイします。

推奨はされませんが、アプリケーションをデプロイした後で AWS のサービスにアクセスするために使用する、AWS アカウントの認証情報も指定する必要がある場合があります。推奨される方法は、IAM ロールを指定することです。[Publish to Elastic Beanstalk] (Elastic Beanstalk への発行) ウィザードでは、同じことを [AWS Options] (オプション) ページの [Identity and Access Management Role] ドロップダウンリストからできます。レガシーの [Publish to Amazon Web Services] (アマゾン ウェブ サービスへの発行) ウィザードでは、[AWS Options] (オプション) ページの [IAM Role] (IAM ロール) ドロップダウンリストがこれに相当します。

IAM ロールではなく、AWS アカウントの認証情報を使用する必要がある場合、アプリケーション用の AWS アカウントの認証情報を、次のいずれかの方法で指定できます。

- プロジェクトの `Web.config` ファイルの `appSettings` 要素内で AWS アカウントの認証情報に対応するプロファイルを参照します。(プロファイルを作成するには、「AWS 認証情報の設定」を参照してください。) 次の例では、プロファイル名が `myProfile` である認証情報を指定しています。

```
<appSettings>
  <!-- AWS CREDENTIALS -->
  <add key="AWSProfileName" value="myProfile"/>
</appSettings>
```

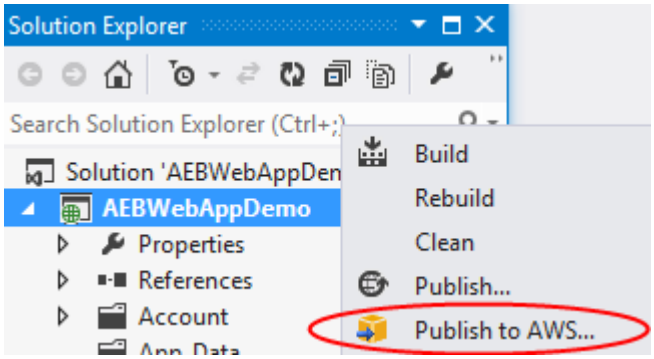
- 使用している IDElastic Beanstalk への発行ウィザード、アプリケーションオプション[] ページで次の操作を行いますキー[] 行のキーそしてValue[] エリアで、AWSAccessKey。[Value (値)] 行にアクセスキーを入力します。次の操作を繰り返します。AWSSecretKey。
- レガシーの [Publish to Amazon Web Services] (アマゾン ウェブ サービスへの発行) ウィザードでは、[Application Options] (アプリケーションオプション) ページの [Application Credentials] (アプリケーション認証情報) エリアで、[Use these credentials] (使用する認証情報) を選択し、[Access Key] (アクセスキー) と [Secret Key] (シークレットキー) のボックスにアクセスキーとシークレットアクセスキーを入力します。

Elastic Beanstalk 環境にアプリケーションを再発行する方法 (レガシー)

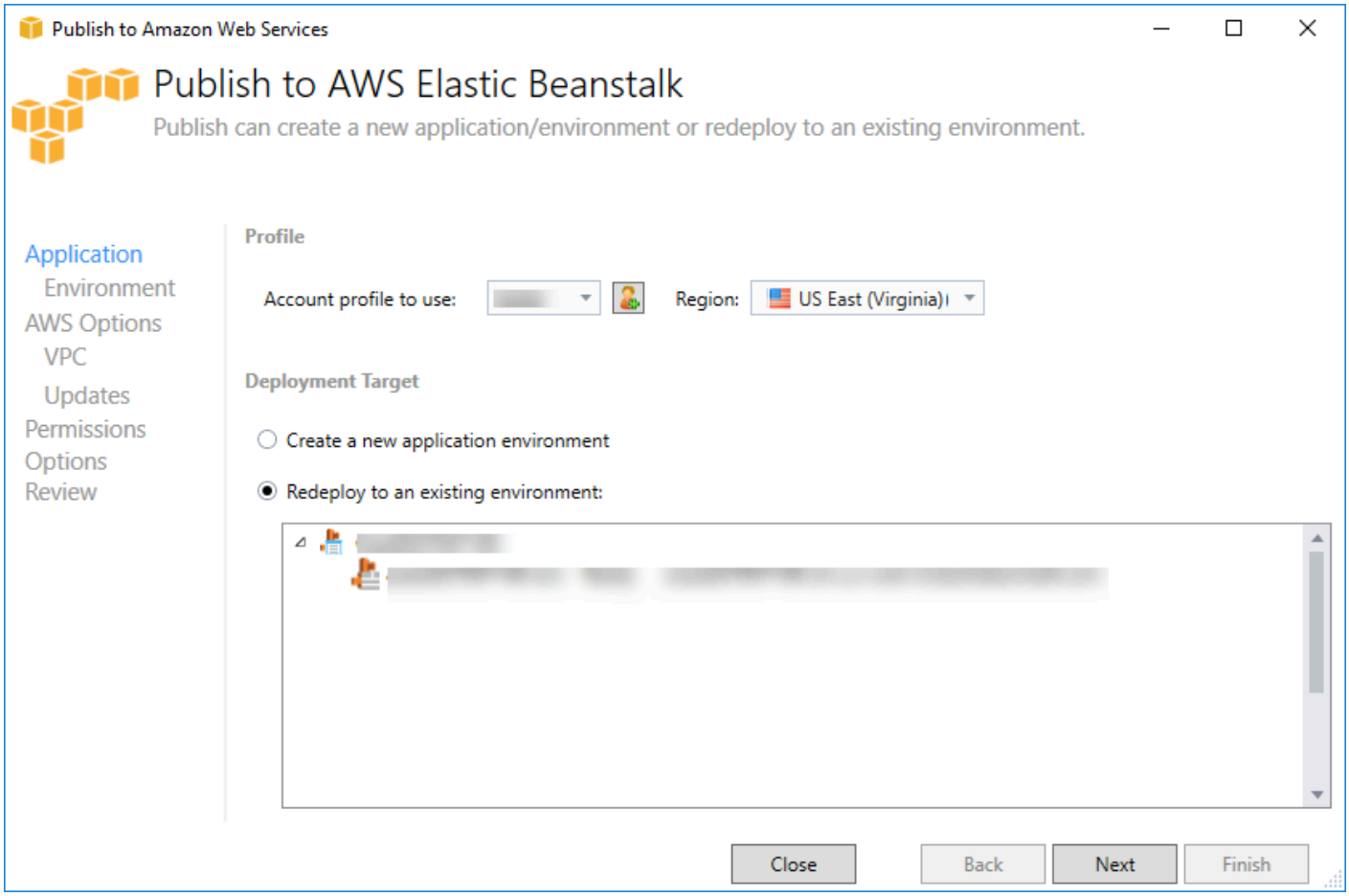
> **Important**
>
> このドキュメントでは、従来のサービスと機能について説明しています。更新されたガイドとコンテンツについては、AWS.NET デプロイツールガイドと更新された Deploying toAWS の目次を参照してください。

個別の変更を行って既に起動しているElastic Beanstalk環境に新しいバージョンを再発行することで、アプリケーションで反復処理することができます。

1. Solution Explorer で、前のセクションで発行したプロジェクトの AEBWebAppDemo プロジェクトフォルダのコンテキスト (右クリック) メニューを開き、[Publish to] (への発行) を選択します AWS Elastic Beanstalk。

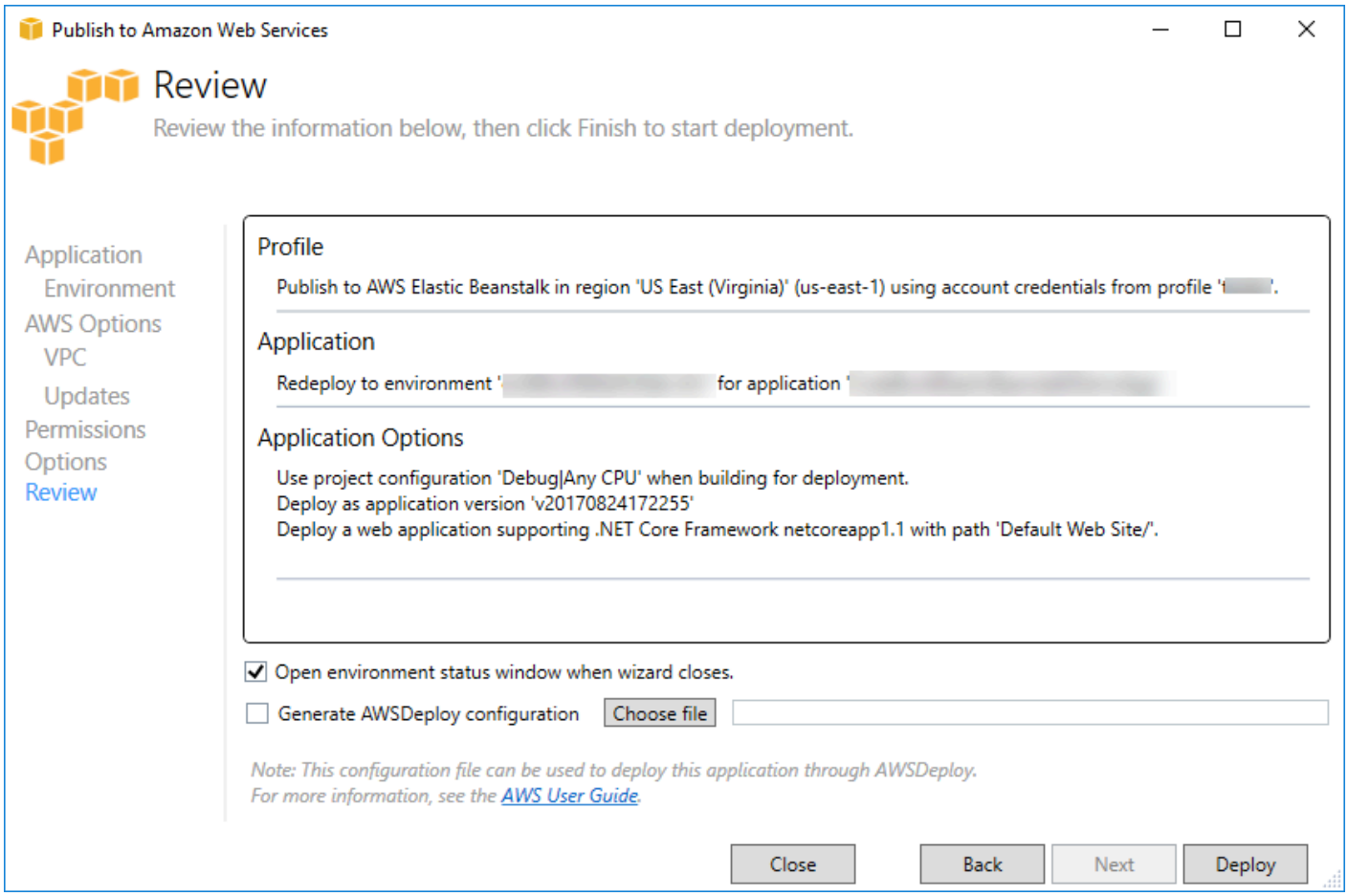


[Publish to Elastic Beanstalk (Elastic Beanstalk への発行] ウィザードが表示されます。



2. [Redeploy to an existing environment (既存の環境への再デプロイ)] を選択してから、先ほど公開した環境を選択します。[Next] (次へ) をクリックします。

   [Review (レビュー)] ウィザードが表示されます。



3. [Deploy (デプロイ)] をクリックします。アプリケーションは同じ環境に再デプロイされます。

アプリケーションが起動中または終了中の場合は再発行することはできません。

Elastic Beanstalk アプリケーションのカスタムデプロイ

このトピックでは、Elastic Beanstalk の Microsoft Windows コンテナ用のデプロイマニフェストがアプリケーションのカスタムデプロイをサポートする方法について説明します。

アプリケーションのカスタムデプロイは、Elastic Beanstalk を活用して、作成と管理を行う、上級ユーザーのための強力な機能です。AWSリソースですが、アプリケーションのデプロイ方法を完全に制御する必要があります。アプリケーションのカスタムデプロイでは、Elastic Beanstalk が実行する 3 つの異なるアクションの Windows PowerShell スクリプトを作成します。デプロイが開始されるときには、インストールアクションが使用され、ツールキットまたはウェブコンソールから

RestartAppServer API が呼び出されるときには、再起動が使用され、新しいデプロイが実行されるときには、以前のデプロイでアンインストールが使用されます。

たとえば、デプロイを行う ASP.NET アプリケーションがあり、ドキュメントチームがデプロイに含むための静的ウェブサイトを作成したとします。次のようなデプロイマニフェストを作成して、これを行うことができます。

```
{
  "manifestVersion": 1,
  "deployments": {

    "msDeploy": [
      {
        "name": "app",
        "parameters": {
          "appBundle": "CoolApp.zip",
          "iisPath": "/"
        }
      }
    ],
    "custom": [
      {
        "name": "PowerShellDocs",
        "scripts": {
          "install": {
            "file": "install.ps1"
          },
          "restart": {
            "file": "restart.ps1"
          },
          "uninstall": {
            "file": "uninstall.ps1"
          }
        }
      }
    ]
  }
}
```

各アクションにリストされているスクリプトは、デプロイマニフェストファイルと関連するアプリケーションバンドルに置かれる必要があります。この例では、アプリケーションバンドルには

documentation.zip ファイルも含まれ、このファイルにドキュメントチームによって作成された静的ウェブサイトが含まれています。

`install.ps1` スクリプトは、zip ファイルを抽出して、IIS パスをセットアップします。

```
Add-Type -assembly "system.io.compression.filesystem"
[io.compression.zipfile]::ExtractToDirectory('./documentation.zip', 'c:\inetpub\wwwroot
\documentation')

powershell.exe -Command {New-WebApplication -Name documentation -PhysicalPath  c:
\inetpub\wwwroot\documentation -Force}
```

アプリケーションは IIS で実行されているので、再起動アクションは IIS のリセットを呼び出します。

```
iisreset /timeout:1
```

アンインストールスクリプトでは、インストールステージで使用された、すべての設定とファイルをクリーンアップすることが重要です。それによって、新しいバージョンのインストールフェーズで、以前のデプロイとの競合を回避できます。この例では、静的ウェブサイトの IIS アプリケーションとウェブサイトのファイルを削除する必要があります。

```
powershell.exe -Command {Remove-WebApplication -Name documentation}
Remove-Item -Recurse -Force 'c:\inetpub\wwwroot\documentation'
```

アプリケーションバンドルに含まれているこれらのスクリプトファイルと documentation.zip ファイルを使って、デプロイによって、ASP.NET アプリケーションが作成され、ドキュメントのサイトがデプロイされます。

この例では、単純な静的ウェブサイトをデプロイする簡単な例を選択しましたが、アプリケーションのカスタムデプロイを使うと、あらゆる種類のアプリケーションをデプロイして、Elastic BeanstalkがAWSそれのためのリソース。

ASP.NET Core Elastic Beanstalk デプロイのカスタム

このトピックでは、デプロイのしくみと、Elastic Beanstalk と Toolkit for Visual Studio を使って ASP.NET Core アプリケーションを作成するときにカスタムデプロイで行うことについて説明します。

Toolkit for Visual Studio のデプロイウィザードを完了すると、ツールキットはアプリケーションをバンドルして Elastic Beanstalk に送信します。アプリケーションバンドルを作成する最初のステップは、新しい dotnet CLI を使い、publish コマンドを使用して、アプリケーションの発行の準備をすることです。フレームワークと設定は、ウィザードの設定から publish コマンドに継承されます。 に Releaseconfiguration を選び、 に netcoreapp1.0framework を選んだ場合は、ツールキットは次のコマンドを実行します。

```
dotnet publish --configuration Release --framework netcoreapp1.0
```

publish コマンドが完了すると、ツールキットは新しいデプロイマニフェストを発行フォルダに書き込みます。デプロイマニフェストは、名前の JSON ファイルです。AWS windows-deployment-manifest.jsElastic Beanstalk Windows コンテナ (バージョン 1.2 以降) はこれを読み込み、アプリケーションのデプロイ方法を決定します。たとえば、IIS のルートにデプロイする ASP.NET Core アプリケーションでは、ツールキットは次のようなマニフェストファイルを生成します。

```
{
  "manifestVersion": 1,
  "deployments": {

    "aspNetCoreWeb": [
      {
        "name": "app",
        "parameters": {
          "appBundle": ".",
          "iisPath": "/",
          "iisWebSite": "Default Web Site"
        }
      }
    ]
  }
}
```

appBundle プロパティは、アプリケーションのバイナリの場所をマニフェストファイルを基準として指定します。このプロパティは、ディレクトリまたは ZIP アーカイブを指すことができます。iisPath プロパティと iisWebSite プロパティは、IIS でアプリケーションをホストする場所を示します。

マニフェストをカスタマイズする

ツールキットは、マニフェストファイルが発行フォルダに存在しない場合のみ、マニフェストファイルを書き込みます。ファイルが存在する場合は、ツールキットは最初のアプリケーションの、マニ

フェストの aspNetCoreWeb セクションの下に一覧表示されている、appBundle、iisPath および iisWebSite プロパティを更新します。これにより、aws-windows-deployment-manifest.json をプロジェクトに追加して、マニフェストをカスタマイズできます。これを Visual Studio で ASP.NET Core ウェブアプリケーションに行うには、新しい JSON ファイルをプロジェクトのルートに追加して、その名前を aws-windows-deployment-manifest.json とします。

マニフェストは aws-windows-deployment-manifest.json という名前である必要があり、プロジェクトのルートに存在する必要があります。Elastic Beanstalk コンテナはルートでマニフェストを検索します。検出された場合には、デプロイツールを呼び出します。ファイルが存在しない場合は、Elastic Beanstalk コンテナは以前のデプロイツールにフォールバックします。これは、アーカイブがmsデプロイアーカイブ

dotnet CLI の publish コマンドにマニフェストが含まれるようにするため、project.json ファイルを更新して、マニフェストファイルが include の include セクション publishOptions に含まれるようにします。

```
{
   "publishOptions": {
     "include": [
       "wwwroot",
       "Views",
       "Areas/**/Views",
       "appsettings.json",
       "web.config",
       "aws-windows-deployment-manifest.json"
     ]
   }
 }
```

マニフェストの宣言が完了し、マニフェストはアプリケーションバンドルに含まれているため、アプリケーションのデプロイの方法をさらに設定することができます。デプロイウィザードよりも詳細に、デプロイをカスタマイズすることができます。AWSは、の JSON スキーマを定義しています。AWS windows-deployment-manifest.json ファイル、Toolkit for Visual Studio をインストールするとき、セットアップはスキーマの URL を登録します。

windows-deployment-manifest.json を開くと、スキーマのドロップダウンボックスで選択した、スキーマの URL があります。URL に移動すると、マニフェストで設定できる機能の完全な説明を取得できます。Visual Studio は、選択されたスキーマを使って、マニフェストの編集時に IntelliSense を提供します。

実行できる 1 つのカスタマイズは、アプリケーションが実行される IIS アプリケーションプールの設定です。以下の例では、IIS アプリケーションプール ("customPool") の定義方法を示しています。このアプリケーションプールはプロセスを 60 分ごとにリサイクルし、`"appPool": "customPool"` を使ってアプリケーションに割り当てます。

```
{
  "manifestVersion": 1,
  "iisConfig": {
    "appPools": [
      {
        "name": "customPool",
        "recycling": {
          "regularTimeInterval": 60
        }
      }
    ]
  },
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "app",
        "parameters": {
          "appPool": "customPool"
        }
      }
    ]
  }
}
```

さらにマニフェストは、インストール、再起動、アンインストールのアクションの前後に実行する、Windows PowerShell スクリプトを宣言できます。たとえば、次のマニフェストは、Windows PowerShell スクリプト `PostInstallSetup.ps1` を実行して、ASP.NET Core アプリケーションが IIS にデプロイされた後にさらにセットアップを行います。このようなスクリプトを追加する場合、スクリプトが `project.json` ファイルの publishOptions の下の include セクションに追加されるようにします。これは `aws-windows-deployment-manifest.json` ファイルの場合と同様です。このようにしない場合は、スクリプトは dotnet CLI の publish コマンドの一部として含まれません。

```
{
  "manifestVersion": 1,
  "deployments": {
    "aspNetCoreWeb": [
```

```
      {
        "name": "app",
        "scripts": {
          "postInstall": {
            "file": "SetupScripts/PostInstallSetup.ps1"
          }
        }
      }
    ]
  }
}
```

.ebextensions について

Elastic Beanstalk.ebextensions設定ファイルは、他の Elastic Beanstalk コンテナのすべてでサポートされています。.ebextensions を ASP.NET Core アプリケーションに含めるには、`.ebextensions` ディレクトリを、`project.json` ファイルの `publishOptions` の下の `include` セクションに追加します。.ebextensions の詳細については、Elastic Beanstalk 開発者ガイドを参照してください。

.NET と Elastic Beanstalk 用の複数アプリケーションのSupport

デプロイマニフェストを使用すると、複数のアプリケーションを同じ Elastic Beanstalk 環境にデプロイすることができます。

デプロイマニフェストは、ASP.NET Core ウェブアプリケーションおよび従来の ASP.NET アプリケーションの msdeploy アーカイブをサポートします。たとえば、フロントエンドに ASP.NET Core を使用したすばらしい新たなアプリケーションを作成し、拡張 API 用のウェブ API プロジェクトがあるとします。また、従来の ASP.NET を使用して作成した管理者アプリがあるとします。

ツールキットのデプロイウィザードは 1 つのプロジェクトのデプロイに重点を置いています。複数のアプリケーションのデプロイを利用するには、アプリケーションバンドルを手動で作成する必要があります。まず、マニフェストを作成します。この例では、ソリューションのルートにマニフェストを作成します。

マニフェストのデプロイメントのセクションには 2 つの子があります。デプロイを行う一連の ASP.NET Core ウェブアプリケーションと、デプロイを行う一連の msdeploy アーカイブです。各アプリケーションで、IIS パスとアプリケーションのビットのマニフェストへの相対位置を設定します。

```
{
```

```
  "manifestVersion": 1,
  "deployments": {

    "aspNetCoreWeb": [
      {
        "name": "frontend",
        "parameters": {
          "appBundle": "./frontend",
          "iisPath": "/frontend"
        }
      },
      {
        "name": "ext-api",
        "parameters": {
          "appBundle": "./ext-api",
          "iisPath": "/ext-api"
        }
      }
    ],
    "msDeploy": [
      {
        "name": "admin",
        "parameters": {
          "appBundle": "AmazingAdmin.zip",
          "iisPath": "/admin"
        }
      }
    ]
  }
}
```

作成したマニフェストを使って、Windows PowerShell を使用して、アプリケーションバンドルを作成し、それを実行する既存の Elastic Beanstalk 環境を更新します。スクリプトは、Visual Studio ソリューションが含まれるフォルダから実行されることを前提として作成されます。

スクリプトで最初に必要なことは、アプリケーションバンドルを作成する、ワークスペースフォルダをセットアップすることです。

```
$publishFolder = "c:\temp\publish"

$publishWorkspace = [System.IO.Path]::Combine($publishFolder, "workspace")
$appBundle = [System.IO.Path]::Combine($publishFolder, "app-bundle.zip")
```

```
If (Test-Path $publishWorkspace){
  Remove-Item $publishWorkspace -Confirm:$false -Force
}
If (Test-Path $appBundle){
  Remove-Item $appBundle -Confirm:$false -Force
}
```

フォルダを作成したら、次にフロントエンドの準備を行います。デプロイウィザードと同様に、dotnet CLI を使用してアプリケーションを発行します。

```
Write-Host 'Publish the ASP.NET Core frontend'
$publishFrontendFolder = [System.IO.Path]::Combine($publishWorkspace, "frontend")
dotnet publish .\src\AmazingFrontend\project.json -o $publishFrontendFolder -c Release
 -f netcoreapp1.0
```

サブフォルダ "frontend" は出力フォルダに使用されていることに注意し、マニフェストで設定するフォルダを一致させます。次に、同様の作業をウェブ API プロジェクトに対して行う必要があります。

```
Write-Host 'Publish the ASP.NET Core extensibility API'
$publishExtAPIFolder = [System.IO.Path]::Combine($publishWorkspace, "ext-api")
dotnet publish .\src\AmazingExtensibleAPI\project.json -o $publishExtAPIFolder -c
 Release -f netcoreapp1.0
```

管理者サイトは、従来の ASP.NET アプリケーションであるため、dotnet CLI を使用することはできません。管理者アプリケーションには、msbuild を使用して、ビルドターゲットパッケージを渡して msdeploy アーカイブを作成します。デフォルトでは、パッケージターゲットは msdeploy アーカイブを obj\Release\Package フォルダの下に作成するため、アーカイブを発行ワークスペースにコピーする必要があります。

```
Write-Host 'Create msdeploy archive for admin site'
msbuild .\src\AmazingAdmin\AmazingAdmin.csproj /t:package /p:Configuration=Release
Copy-Item .\src\AmazingAdmin\obj\Release\Package\AmazingAdmin.zip $publishWorkspace
```

Elastic Beanstalk 環境に対してこれらすべてのアプリケーションに関する指示を加えるため、マニフェストをソリューションから発行ワークスペースへコピーし、フォルダを圧縮します。

```
Write-Host 'Copy deployment manifest'
Copy-Item .\aws-windows-deployment-manifest.json $publishWorkspace
```

```
Write-Host 'Zipping up publish workspace to create app bundle'
Add-Type -assembly "system.io.compression.filesystem"
[io.compression.zipfile]::CreateFromDirectory( $publishWorkspace, $appBundle)
```

アプリケーションバンドルの作成が完了したので、ウェブコンソールに移動して、アーカイブを Elastic Beanstalk 環境にアップロードします。あるいは、を引き続き使用することもできます。AWSPowerShell コマンドレットを使用して Elastic Beanstalk 環境を更新します。Elastic Beanstalk 環境を含むプロファイルとリージョンに、現在のプロファイルとリージョンが設定されていることを確認します。これには、`Set-AWSCredentials`そして`Set-DefaultAWSRegion`コマンドレット。

```
Write-Host 'Write application bundle to S3'
# Determine S3 bucket to store application bundle
$s3Bucket = New-EBStorageLocation
Write-S3Object -BucketName $s3Bucket -File $appBundle


$applicationName = "ASPNETCoreOnAWS"
$environmentName = "ASPNETCoreOnAWS-dev"
$versionLabel = [System.DateTime]::Now.Ticks.ToString()

Write-Host 'Update Beanstalk environment for new application bundle'
New-EBApplicationVersion -ApplicationName $applicationName -VersionLabel $versionLabel
 -SourceBundle_S3Bucket $s3Bucket -SourceBundle_S3Key app-bundle.zip
Update-EBEnvironment -ApplicationName $applicationName -EnvironmentName
 $environmentName -VersionLabel $versionLabel
```

ここで、ツールキットまたはウェブコンソールのどちらかの Elastic Beanstalk 環境のステータスページを使用して、更新のステータスを確認します。完了すると、デプロイマニフェストで設定した IIS パスを使って、デプロイされた各アプリケーションに移動できるようになります。

Amazon EC2 Container Service へのデプロイ

> **Important**
>
> 新への発行AWSこの機能は、.NET アプリケーションを公開する方法を簡素化するように設計されています。AWS。選択した後に、この公開エクスペリエンスに切り替えるかどうか求められることがあります[コンテナの公開]AWS。詳細については、「への発行AWSVisual で、」を参照してください。

Amazon Elastic Container Service は非常にスケーラブルかつ高性能なコンテナ管理サービスで、Docker コンテナに対応しており、Amazon EC2 インスタンスのマネージド型クラスターでアプリケーションを簡単に実行できるようになります。

Amazon Elastic Container Service にアプリケーションをデプロイするには、アプリケーションコンポーネントを開発して Docker コンテナで実行する必要があります。Docker コンテナは標準化されたソフトウェア開発用のユニットであり、コード、ランタイム、システムツール、システムライブラリなど、ソフトウェアアプリケーションの実行に必要なものがすべて含まれています。

Toolkit for Visual Studio には、Amazon ECS によるアプリケーションの公開を簡素化するウィザードが用意されています。このウィザードについては、次のセクションで説明します。

Amazon ECS の詳細については、を参照してください。Elastic コンテナサービスのドキュメント。このドキュメントには、Docker の基本とクラスターの作成の概要が示されています。

トピック

- を指定するAWSASP.NET コア 2 アプリケーションの認証情報
- Amazon ECS への ASP.NET Core 2.0 アプリケーションのデプロイ (Fargate) (レガシー)
- Amazon ECS への ASP.NET Core 2.0 アプリケーションのデプロイ (EC2)

を指定するAWSASP.NET コア 2 アプリケーションの認証情報

アプリケーションを Docker コンテナにデプロイするときに必要になる認証情報には、デプロイ認証情報とインスタンス認証情報の 2 タイプがあります。

デプロイ認証情報は、Publish Container によって次のように使用されますAWSAmazon ECS で環境を作成するためのウィザードです。これには、タスク、サービス、IAM ロール、Docker コンテナリポジトリのほか、選択した場合はロードバランサーも含まれます。

インスタンス認証情報は、インスタンス (アプリケーションを含む) によってさまざまな AWS サービスにアクセスするために使用されます。たとえば、ASP.NET Core 2.0 アプリケーションが Amazon S3 オブジェクトに対して読み書きを行う場合は、適切なアクセス権限が必要になります。そのために、環境に基づいて異なる方法で異なる認証情報を提供できます。たとえば、ASP.NET Core 2 アプリケーションは開発環境と本番稼働用環境を対象とする場合があります。ローカルの Docker インスタンスおよび認証情報を開発環境に使用し、定義したロールを本番稼働用環境に使用できます。

デプロイ認証情報の指定

-AWSで指定したアカウントコンテナの公開先AWSウィザードはAWSウィザードが Amazon ECS へのデプロイに使用するアカウント。アカウントプロファイルには、Amazon Elastic Compute Cloud、Amazon Elastic コンテナサービス、およびAWS Identity and Access Management。

ドロップダウンリストにオプションが見つからない場合は、それらのアクセス権限が不足している可能性があります。たとえば、アプリケーション用のクラスターを作成したが、に表示されない場合です。コンテナの公開先AWSウィザードの [クラスタ] ページ この場合、不足しているアクセス権限を追加してウィザードを再試行してください。

開発用のインスタンス認証情報の指定

本番稼働用でない環境では、appsettings.<environment>.json ファイルに認証情報を設定できます。たとえば、Visual Studio 2017 の appsettings.Development.json ファイルで認証情報を設定するには、以下の手順に従います。

1. プロジェクトに AWSSDK.Extensions.NETCore.Setup NuGet パッケージを追加します。
2. を追加します。AWSAppSettings.Development.jsonへの設定。以下に示しているのは、`Profile` と `Region` の設定です。

```
{
    "AWS": {
        "Profile": "local-test-profile",
        "Region": "us-west-2"
    }
}
```

本番稼働用のインスタンス認証情報の指定

本番稼働用インスタンスでは、アプリケーション (およびサービス) のアクセスコントロールに IAM ロールを使用することをお勧めします。たとえば、Amazon ECS をサービスプリンシパルとし、Amazon シンプルストレージサービスと Amazon DynamoDB に対するアクセス権限を持つ IAM ロールをAWS Management Console:

1. AWS Management Console にサインインして、https://console.aws.amazon.com/iam/ で IAM コンソールを開きます。
2. IAM コンソールのナビゲーションペインで、[Roles]、[Create role] の順に選択します。

3. [AWSサービスロールタイプを選択し、次に [] を選択します。EC2 Container Service　。
4. [EC2 Container Service Task (EC2 コンテナサービスタスク)] ユースケースを選択します。ユースケースは、サービスに必要な信頼ポリシーを含めるように定義されています。続いて、[次へ] を選択します。アクセス許可.
5. [AmazonS3FullAccess] および [AmazonDynamoDBFullAccess] 権限ポリシーを選択します。各ポリシーの横にあるチェックボックスをオンにし、次に [] を選択します。次へ: 確認,
6. [Role name (ロール名)] に、このロールの目的を識別するのに役立つロール名またはロール名サフィックスを入力します。ロール名は AWS アカウント内で一意でなければなりません。大文字と小文字は区別されません。たとえば、`PRODROLE` と `prodrole` というロール名を両方作成することはできません。多くのエンティティによりロールが参照されるため、作成後にロール名を変更することはできません。
7. (オプション) [Role description] に、新しいロールの説明を入力します。
8. ロール情報を確認し、[Create role (ロールの作成)] を選択します。

このロールは、タスクロールでECS タスク定義のページコンテナの公開先AWSウィザード。

詳細については、「サービスベースのロールの使用」を参照してください。

Amazon ECS への ASP.NET Core 2.0 アプリケーションのデプロイ (Fargate) (レガシー)

> **Important**
>
> このドキュメントでは、従来のサービスと機能について説明しています。更新されたガイドとコンテンツについては、AWS.NET デプロイツールガイドと更新された Deploying toAWS の目次を参照してください。

このセクションでは、 Toolkit for Visual Studioの一部として提供される [Publish Container to AWS] (AWS にコンテナを発行) ウィザードにより、Fargate 起動タイプを使用してAmazon ECS経由で Linux 向けコンテナ化 ASP.NET Core 2.0 アプリケーションをデプロイする方法について説明します。ウェブアプリケーションは継続的に実行されるため、サービスとしてデプロイされます。

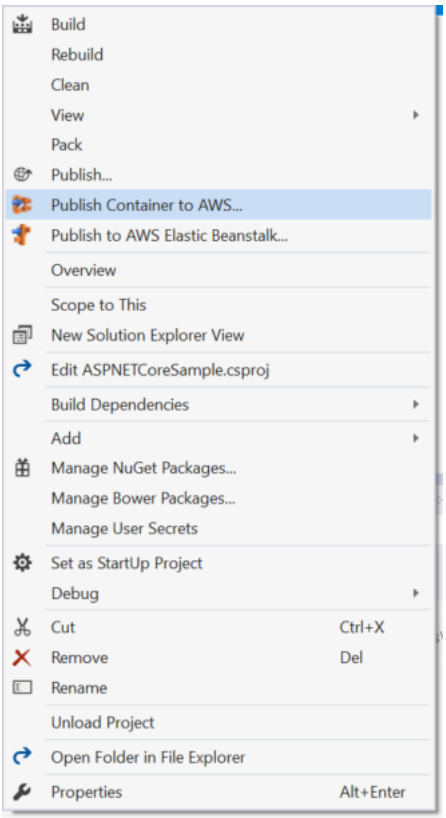
コンテナを発行する前に

[Publish Container to AWS] (AWS にコンテナを発行) ウィザードを使用して ASP.NET Core 2.0 アプリケーションをデプロイする前に

- AWS 認証情報を指定し、Amazon ECS をセットアップします。
- Docker をインストールします。Docker for Windows などさまざまなインストールオプションがあります。
- Visual Studio で、Linux をターゲットとする ASP.NET Core 2.0 コンテナ化アプリケーションのプロジェクトを作成します (または開きます) 。

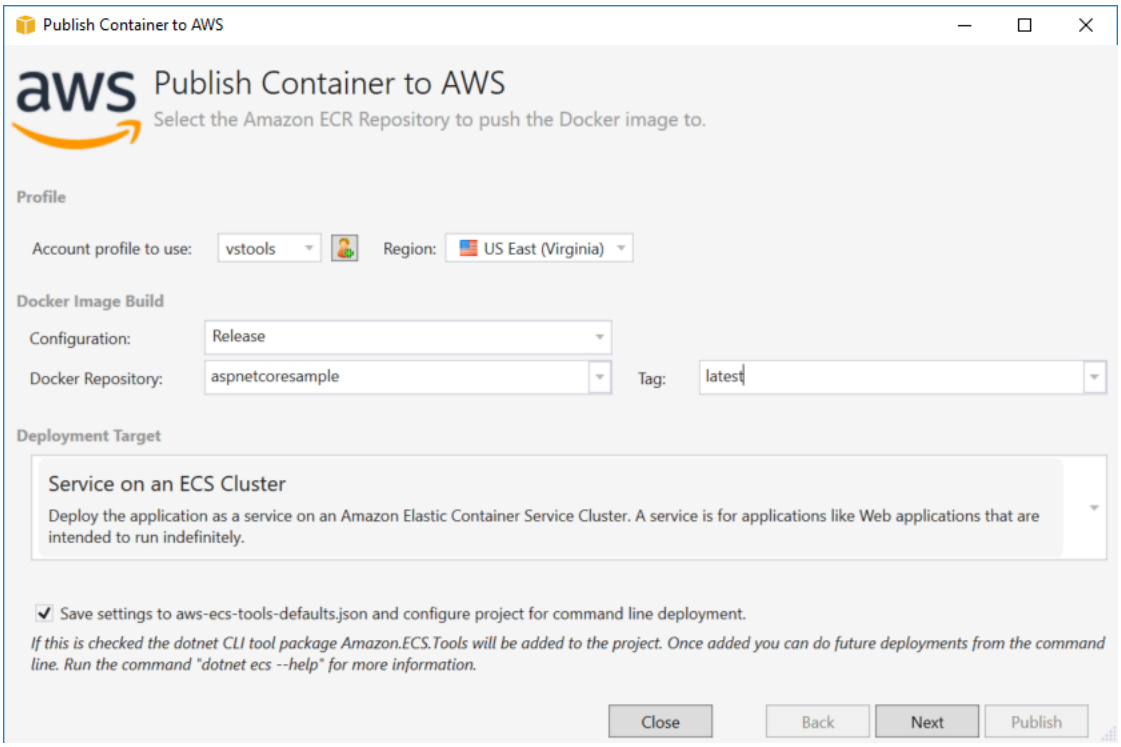
[Publish Container to AWS] (AWS にコンテナを発行) ウィザードへのアクセス

Linux 用 ASP.NET Core 2.0 コンテナ化アプリケーションをデプロイするには、Solution Explorer でプロジェクトを右クリックし、[Publish Container to AWS] (AWS にコンテナを発行) を選択します。



Visual Studio で [Build] (ビルド) メニューの [Publish Container to AWS] (AWS にコンテナを発行) を選択することもできます。

[Publish Container to AWS] (AWS にコンテナを発行) ウィザード



Account profile to use (使用するアカウントのプロファイル) - 使用するアカウントプロファイルを選択します。

Region (リージョン) - デプロイリージョンを選択します。プロファイルとリージョンは、デプロイ環境リソースのセットアップとデフォルト Docker レジストリの選択に使用されます。

Configuration (設定) - Docker イメージビルド設定を選択します。

Docker Repository (Docker リポジトリ) - 既存の Docker リポジトリを選択するか、新しいリポジトリの名前を入力します (新しいリポジトリが作成されます)。これは、ビルドコンテナがプッシュされるリポジトリです。

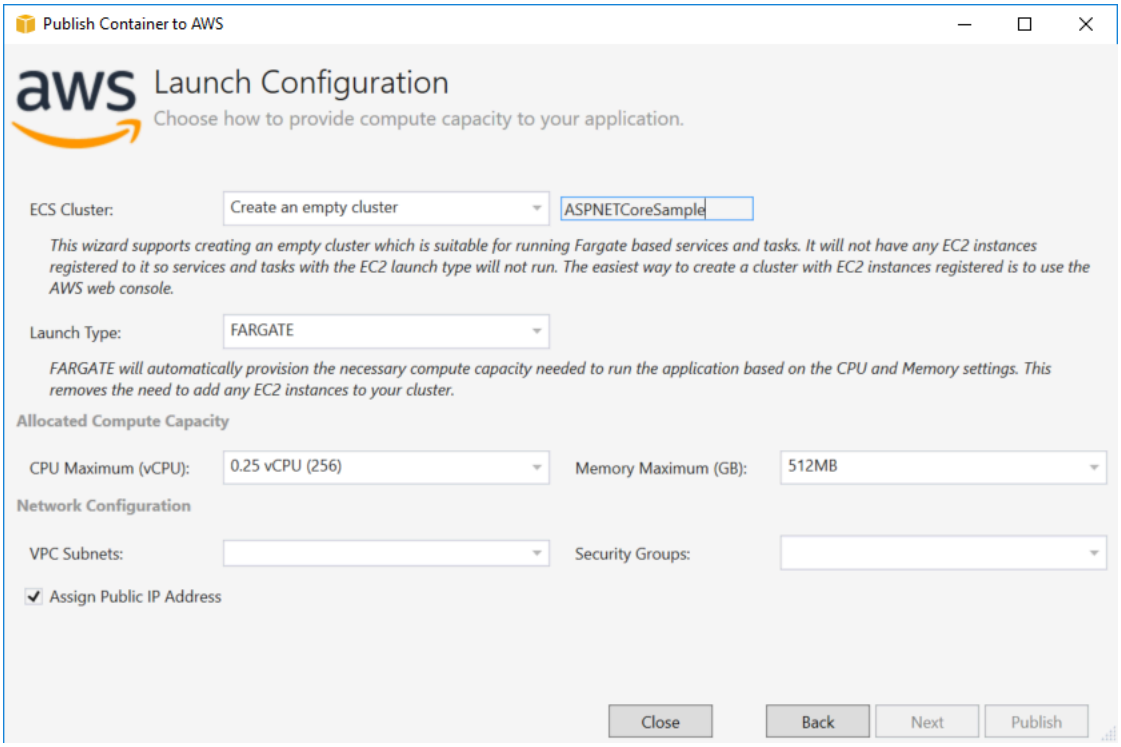
Tag (タグ) - 既存のタグを選択するか、新しいタグの名前を入力します。タグを使用して、Docker コンテナに固有の設定要素 (バージョン、オプションなど) のような重要な詳細を追跡できます。

Deployment Target (デプロイのターゲット) - [Service on an ECS Cluster (ECS クラスターのサービス)] を選択します。このデプロイオプションは、実行時間の長いアプリケーション (ASP.NET ウェブアプリケーションなど) に使用します。

Save settings to **`aws-docker-tools-defaults.json`** and configure project for command line deployment (`aws-docker-tools-defaults.json` に設定を保存して、コマンドラインデプロイ

のプロジェクトを設定する) - コマンドラインから柔軟にデプロイしたいて場合には、このチェックボックスをオンにします。`dotnet ecs deploy` を使用してプロジェクトディレクトリからコンテナをデプロイし、`dotnet ecs publish` を使用してコンテナを発行します。

[Launch Configuration (起動設定)] ページ



ECS Cluster (ECS クラスター) - Docker イメージを実行するクラスターを選択します。空のクラスターを作成する場合は、新しいクラスターの名前を入力します。

Launch Type (起動タイプ) - [FARGATE] を選択します。

CPU Maximum (vCPU) - アプリケーションに必要なコンピューティング性能の最大数を選択します。CPU とメモリの値の許容範囲を確認するには、「タスクサイズ」を参照してください。

Memory Maximum (GB) (最大メモリ (GB)) - アプリケーションで使用できるメモリの最大量を選択します。

VPC Subnets (VPC サブネット) - 1 つの VPC の 1 つ以上のサブネットを選択します。複数のサブネットを選択すると、タスクはそれらのサブネット間で分散されます。これにより、可用性が向上します。詳細については、「デフォルトの VPC とサブネット」を参照してください。

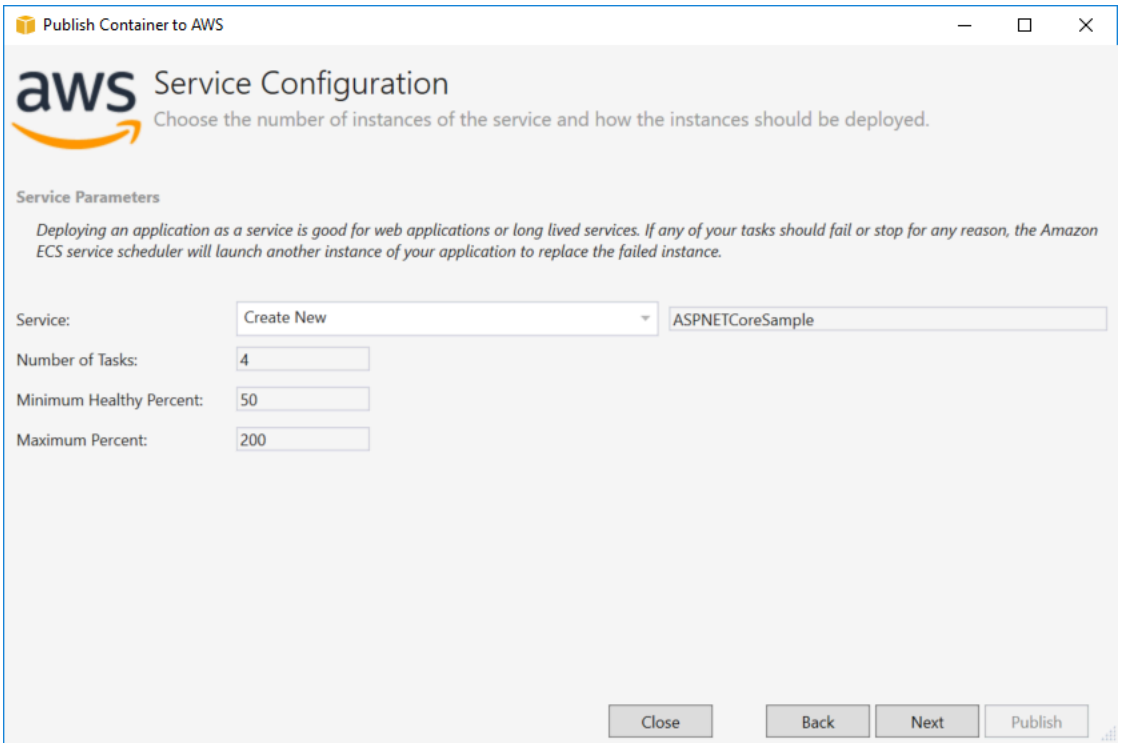
Security Groups (セキュリティグループ) - セキュリティグループを選択します。

セキュリティグループは、関連付けられた Amazon EC2 インスタンスのファイアウォールとして動作し、インバウンドトラフィックとアウトバウンドトラフィックの両方をインスタンスレベルでコントロールします。

デフォルトセキュリティグループは、同じセキュリティグループに割り当てられたインスタンスからのインバウンドトラフィックとすべてのアウトバウンド IPv4 トラフィックを許可するように設定されています。サービスがコンテナリポジトリに到達できるように、アウトバウンドトラフィックを許可している必要があります。

Assign Public IP Address (パブリック IP アドレスの割り当て) - タスクをインターネットからアクセス可能にするには、このチェックボックスをオンにします。

[Service Configuration (サービス設定)] ページ



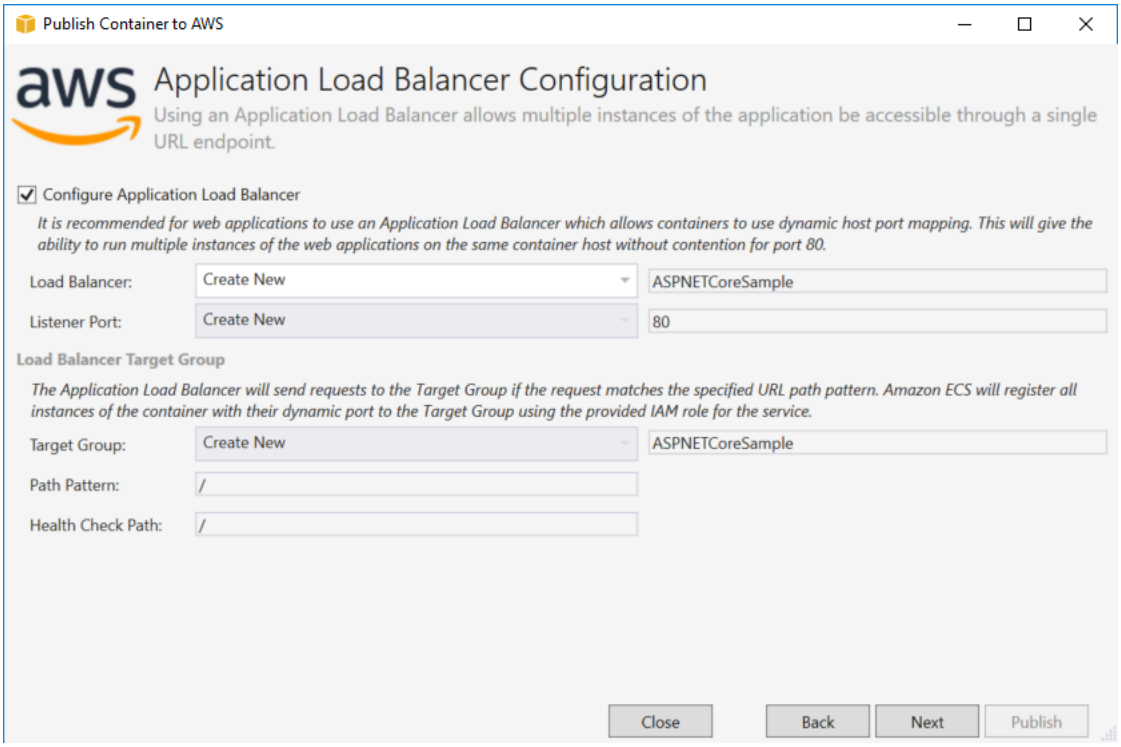
Service (サービス) - 既存のサービス内にコンテナをデプロイするには、ドロップダウンリストでいずれかのサービスを選択します。または、新しいサービスを作成するには、[Create New (新規作成)] を選択します。サービス名は同じクラスター内で一意になるようにしてください。ただし、リージョン内のクラスター間や複数のリージョンにまたがるクラスター間では、同様の名前のサービスがあっても構いません。

Number of Tasks (タスクの数) - クラスターにデプロイして実行状態に保つタスクの数。各タスクはコンテナの 1 つのインスタンスです。

Minimum Healthy Percent (最小ヘルス率) - デプロイ中に RUNNING 状態に保つ必要のあるタスクの最小割合 (最も近い整数に切り上げ)。

Maximum Percent (最大率) - デプロイ中に RUNNING または PENDING 状態に保つことのできるタスクの最大割合 (最も近い整数に切り下げ)。

[Application Load Balancer (アプリケーションロードバランサー)] ページ



Configure Application Load Balancer (アプリケーションロードバランサーの設定) - Application Load Balancer を設定するには、このチェックボックスをオンにします。

Load Balancer (ロードバランサー) - 既存のロードバランサーを選択するか、[Create New (新規作成)] を選択して新しいロードバランサーの名前を入力します。

Listener Port (リスナーポート) - 既存のリスナーポートを選択するか、[Create New (新規作成)] を選択してポート番号を入力します。デフォルトのポート 80 はほとんどのウェブアプリケーションに適しています。

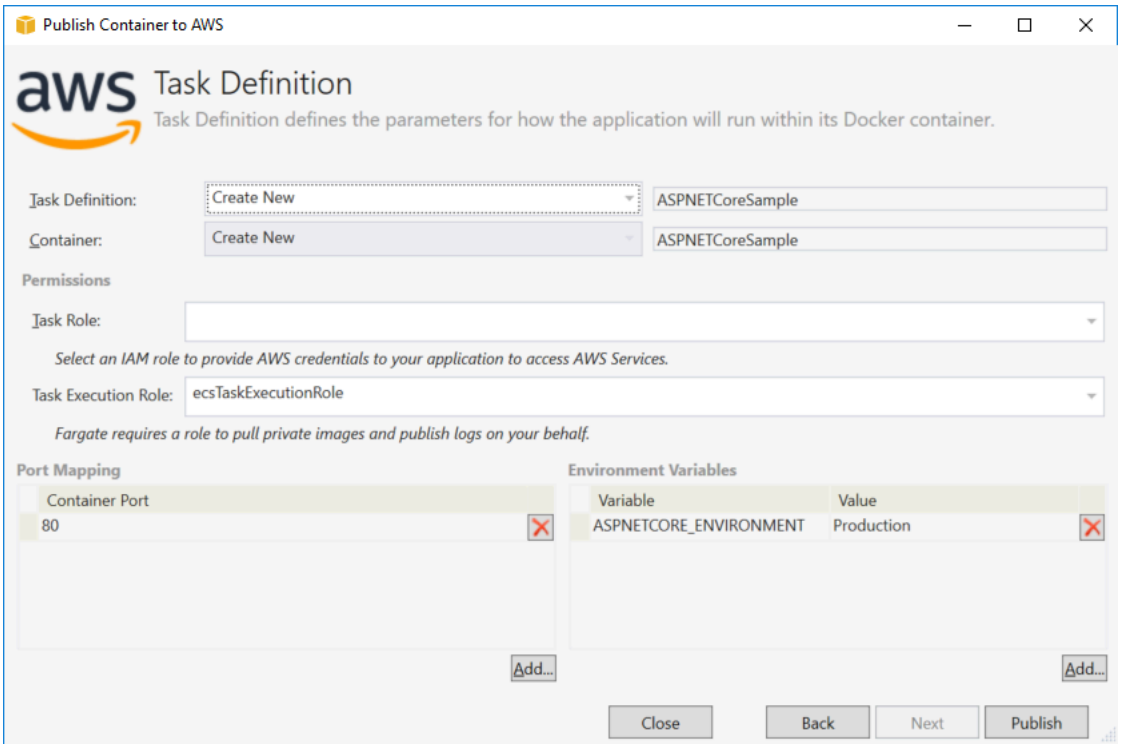
[Target Group] (ターゲットグループ) - Amazon ECS がサービスに対するタスクを登録するターゲットグループを選択します。

Path Pattern (パスパターン) - ロードバランサーがパスベースのルーティングを使用するようになります。デフォルトの / を受け入れるか、別のパターンを指定します。パスパターンでは大文字と小文字が区別され、最大 128 文字までの長さで、特定の文字セットを含めることができます。

Health Check Path (ヘルスチェックパス) - ヘルスチェックのターゲットの ping 先のパス。デフォルトでは、/ です。必要に応じて別のパスを入力します。入力したパスが無効な場合、ヘルスチェックは失敗し、異常とみなされます。

複数のサービスをデプロイし、各サービスが異なるパスまたは場所にデプロイされる場合は、カスタムチェックパスが必要になります。

[Task Definition (タスク定義)] ページ



Task Definition (タスク定義) - 既存のタスク定義を選択するか、[Create New (新規作成)] を選択して新しいタスク定義の名前を入力します。

Container (コンテナ) - 既存のコンテナを選択するか、[Create New (新規作成)] を選択して新しいコンテナの名前を入力します。

Task Role (タスクロール) - アプリケーションが AWS サービスにアクセスするための認証情報がある IAM ロールを選択します。これは、アプリケーションに認証情報が渡される方法になります。「アプリケーション用に AWS セキュリティの認証情報を指定する方法」を参照してください。

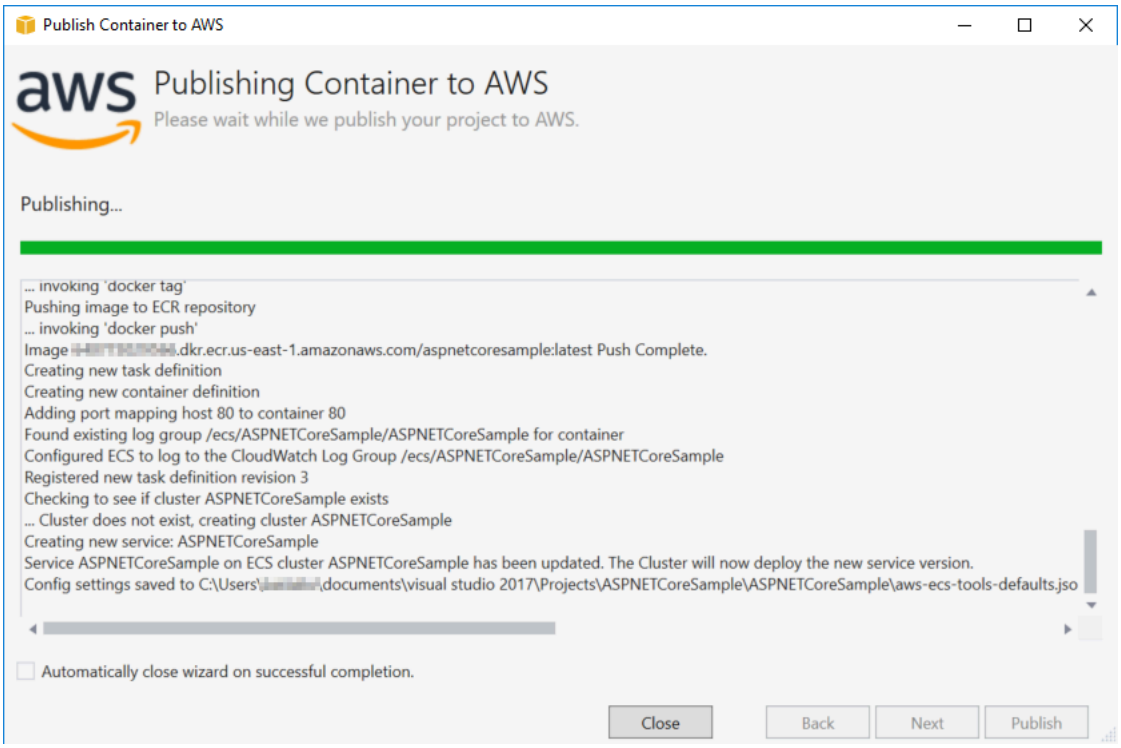
Task Execution Role (タスク実行ロール) - プライベートイメージをプルしてログを公開するアクセス許可を持つロールを選択します。AWSFargate がユーザーに代わってそのロールを使用します。

Port Mapping (ポートマッピング) - 自動的に割り当てられたホストポートにバインドされるコンテナポートの番号を選択します。

Environment Variables (環境変数) - コンテナの環境変数を追加、変更、または削除します。環境変数はデプロイに合わせて変更できます。

設定に問題がなければ、[Publish (発行)] をクリックしてデプロイプロセスを開始します。

AWS へのコンテナの発行



デプロイ中にイベントが表示されます。ウィザードは正常終了時に自動的に閉じられます。この動作を無効にするには、ページの下部にあるチェックボックスをオフにします。

新しいインスタンスの URL は AWS Explorer で見つけることができます。[Amazon ECS and Clusters (Amazon ECS およびクラスター)] を展開し、クラスターをクリックします。

Amazon ECS への ASP.NET Core 2.0 アプリケーションのデプロイ (EC2)

このセクションでは、の使用方法について説明します。コンテナのパブリッシュ先AWSToolkit for Visual Studio の一部として提供されるウィザード。EC2 起動タイプを使用して Amazon ECS を通じて Linux をターゲットとする ASP.NET Core 2.0 アプリケーションをデプロイします。ウェブアプリケーションは継続的に実行されるため、サービスとしてデプロイされます。

コンテナを公開する前に

を使用する前にコンテナのパブリッシュ先AWSASP.NET Core 2.0 アプリケーションをデプロイするには:

- を指定します。AWS資格情報そしてAmazon ECS をセットアップする。
- Docker をインストールします。Docker for Windows などさまざまなインストールオプションがあります。
- ウェブアプリケーションのニーズに基づいて、Amazon ECS クラスターを作成します。このプロセスはほんの数ステップで完了します。
- Visual Studio で、Linux 向け ASP.NET Core 2.0 コンテナ化アプリケーションのプロジェクトを作成します (または開きます)。

へのコンテナの公開AWSウィザード

Linux 向け ASP.NET Core 2.0 コンテナ化アプリケーションをデプロイするには、Solution Explorer でプロジェクトを右クリックし、コンテナのパブリッシュ先AWS。

また、[] を選択できます。コンテナのパブリッシュ先AWSVisual Studio の [Build] メニューで。

コンテナのパブリッシュ先AWSウィザード

Account profile to use (使用するアカウントのプロファイル) - 使用するアカウントプロファイルを選択します。

Region (リージョン) - デプロイリージョンを選択します。プロファイルとリージョンは、デプロイ環境リソースのセットアップとデフォルト Docker レジストリの選択に使用されます。

Configuration (設定) - Docker イメージビルド設定を選択します。

Docker Repository (Docker リポジトリ) - 既存の Docker リポジトリを選択するか、新しいリポジトリの名前を入力します (新しいリポジトリが作成されます)。これは、ビルドコンテナイメージがプッシュされるリポジトリです。

Tag (タグ) - 既存のタグを選択するか、新しいタグの名前を入力します。タグを使用して、Docker コンテナに固有の設定要素 (バージョン、オプションなど) のような重要な詳細を追跡できます。

Deployment (デプロイ) - [Service on an ECS Cluster (ECS Cluster のサービス)] を選択します。このデプロイオプションは、実行時間の長いアプリケーション (ASP.NET Core 2.0 ウェブアプリケーションなど) に使用します。

Save settings to**`aws-docker-tools-defaults.json`**コマンドライン展開用にプロジェクトを構成する-コマンドラインから柔軟にデプロイする場合は、このチェックボックスをオンにし

ます。`dotnet ecs deploy` を使用してプロジェクトディレクトリからコンテナをデプロイし、`dotnet ecs publish` を使用してコンテナを公開します。

[Launch Configuration (起動設定)] ページ

ECS Cluster (ECS クラスター) - Docker イメージを実行するクラスターを選択します。以下の操作を実行できます。ECS クラスターを作成するを使用してAWSマネジメントコンソール。

Launch Type (起動タイプ) - [EC2] を選択します。Fargate 起動タイプを使用するには、Amazon ECS への ASP.NET Core 2.0 アプリケーションのデプロイ (Fargate) を参照してください。

[Service Configuration (サービス設定)] ページ

Service (サービス) - 既存のサービス内にコンテナをデプロイするには、ドロップダウンリストでいずれかのサービスを選択します。または、新しいサービスを作成するには、[Create New (新規作成)] を選択します。サービス名は同じクラスター内で一意になるようにしてください。ただし、リージョン内のクラスター間や複数のリージョンにまたがるクラスター間では、同様の名前のサービスがあっても構いません。

Number of Tasks (タスクの数) - クラスターにデプロイして実行状態に保つタスクの数。各タスクはコンテナの 1 つのインスタンスです。

Minimum Healthy Percent (最小ヘルス率) - デプロイ中に `RUNNING` 状態に保つ必要のあるタスクの最小割合 (最も近い整数に切り上げ)。

Maximum Percent (最大率) - デプロイ中に `RUNNING` または `PENDING` 状態に保つことのできるタスクの最大割合 (最も近い整数に切り下げ)。

Placement Templates (配置テンプレート) - タスク配置テンプレートを選択します。

クラスター内にタスクを起動するとき、Amazon ECS では、タスク定義で指定されている CPU やメモリなどの要件に基づいてタスクを配置する場所を決定する必要があります。同様に、タスク数を減らすときも、Amazon ECS でどのタスクを終了させるか決定する必要があります。

配置テンプレートは、クラスター内にタスクを起動する方法をコントロールします。

- [AZ Balanced Spread] - アベイラビリティーゾーン間およびアベイラビリティーゾーン内のコンテナインスタンス間でタスクを分散します。
- [AZ Balanced BinPack] - 利用可能な最小メモリでアベイラビリティーゾーン間およびコンテナインスタンス間でタスクを分散します。

- [BinPack] - CPU またはメモリの最小利用可能量に基づいてタスクを配置します。
- [One Task Per Instance] - 各コンテナインスタンスのサービスから最大 1 タスクを配置します。

詳細については、Amazon ECS のタスク配置を参照してください。

[Application Load Balancer (アプリケーションロードバランサー)] ページ

Configure Application Load Balancer (アプリケーションロードバランサーの設定) - Application Load Balancer を設定するには、このチェックボックスをオンにします。

Select IAM role for service (サービス用の IAM ロールの選択) - 既存のロールを選択するか、[Create New (新規作成)] を選択して新しいロールを作成します。

Load Balancer (ロードバランサー) - 既存のロードバランサーを選択するか、[Create New (新規作成)] を選択して新しいロードバランサーの名前を入力します。

Listener Port (リスナーポート) - 既存のリスナーポートを選択するか、[Create New (新規作成)] を選択してポート番号を入力します。デフォルトのポート 80 はほとんどのウェブアプリケーションに適しています。

Target Group (ターゲットグループ) - デフォルトでは、ロードバランサーはターゲットグループ用に指定したポートとプロトコルを使用して登録済みターゲットにリクエストを送ります。ターゲットグループに各ターゲットを登録するときに、このポートを上書きできます。

Path Pattern (パスパターン) - ロードバランサーがパスベースのルーティングを使用するようになります。デフォルトの / を受け入れるか、別のパターンを指定します。パスパターンでは大文字と小文字が区別され、最大 128 文字までの長さで、特定の文字セットを含めることができます。

Health Check Path (ヘルスチェックパス) - ヘルスチェックのターゲットの ping 先のパス。デフォルトでは / であり、ウェブアプリケーションに適しています。必要に応じて別のパスを入力します。入力したパスが無効な場合、ヘルスチェックは失敗し、異常とみなされます。

複数のサービスをデプロイし、各サービスが異なるパスまたは場所にデプロイされる場合は、カスタムチェックパスが必要になることがあります。

[ECS Task Definition (ECS タスク定義)] ページ

Task Definition (タスク定義) - 既存のタスク定義を選択するか、[Create New (新規作成)] を選択して新しいタスク定義の名前を入力します。

Container (コンテナ) - 既存のコンテナを選択するか、[Create New (新規作成)] を選択して新しいコンテナの名前を入力します。

Memory (MiB) (メモリ(MiB)) - ソフト制限とハード制限のいずれか、または両方の値を入力します。

コンテナ用に予約するメモリのソフト制限 (MiB 単位)。Docker により、コンテナメモリはソフト制限を超えないように保たれます。コンテナは、メモリパラメータで指定したハード制限 (該当する場合) に達するか、コンテナインスタンス上の使用可能なメモリの上限に達するか、いずれか早く達する方まで、メモリを使用できます。

コンテナに適用されるメモリのハード制限 (MiB 単位)。コンテナは、ここで指定したメモリを超えようとすると、強制終了されます。

タスクロール-IAM ロールのタスクロールを選択します。これにより、コンテナへのアクセス権限がAWS関連するポリシーに指定されている API をユーザーに代わって指定する API。これは、アプリケーションに認証情報が渡される方法になります。「」を参照してください。指定方法AWSアプリケーションのセキュリティ認証情報。

Port Mapping (ポートマッピング) - コンテナのポートマッピングを追加、変更、または削除します。ロードバランサーがオンの場合、ホストポートはデフォルトで 0 になり、ポート割り当ては動的になります。

Environment Variables (環境変数) - コンテナの環境変数を追加、変更、または削除します。

設定に問題がなければ、[Publish (公開)] をクリックしてデプロイプロセスを開始します。

コンテナのへの公開AWS

デプロイ中にイベントが表示されます。ウィザードは正常終了時に自動的に閉じられます。この動作を無効にするには、ページの下部にあるチェックボックスをオフにします。

新しいインスタンスの URL は、AWSExplorer。[Amazon ECS and Clusters (Amazon ECS およびクラスター)] を展開し、クラスターをクリックします。

のトラブルシューティング AWS Toolkit for Visual Studio

以下のセクションでは、 AWS Toolkit for Visual Studio および ツールキットの AWS サービスの使用に関する一般的なトラブルシューティング情報について説明します。

> **Note**
>
> インストールと set-up-specific トラブルシューティングに関する情報は、このユーザーガイドの「インストールに関する問題のトラブルシューティング」トピックで入手できます。

トピック

トラブルシューティングのベストプラクティス

以下は、 AWS Toolkit for Visual Studio 問題のトラブルシューティング時に推奨されるベストプラクティスです。

- レポートを送信する前に、問題またはエラーの再現を試してください。
- 再現プロセス中に、各ステップ、設定、エラーメッセージを詳細にメモしてください。
- AWS Toolkit Logs を収集します。Toolkit AWS ログの検索方法の詳細については、このガイドトピックにあるAWS 「ログの検索方法」の手順を参照してください。
- 未解決のリクエストや既知の解決策がないか確認するか、 AWS Toolkit for Visual Studio GitHub リポジトリの「問題AWS Toolkit for Visual Studio」セクションで未解決の問題を報告します。

AWS Toolkit ログを見つける方法

1. Visual Studio のメインメニューから [Extensions] を展開します。
2. AWS ツールキットを選択して AWS ツールキットメニューを展開し、ツールキットログの表示を選択します。
3. オペレーティングシステムで AWS Toolkit Logs フォルダが開いたら、ファイルを日付別にソートし、現在の問題に関連する情報を含むログファイルを見つけます。

Amazon CodeWhisperer サインインとサインアウトは無効になっています

サインインメニュー項目とサインアウトメニュー項目の両方が無効になっている CodeWhisperer サービスで問題が発生した場合は、次の手順を実行して問題をトラブルシューティングします。

1. Windows File Explorer から、 にある AWS Toolkit キャッシュフォルダに移動します`%LOCALAPPDATA%/aws/toolkits/language-servers/CodeWhisperer`。
2. キャッシュフォルダの内容をクリアします。
3. 現在のソリューションを閉じて再度開きます。

のセキュリティ AWS Toolkit for Visual Studio

クラウドセキュリティは Amazon Web Services (AWS) の最優先事項です。 AWS のお客様は、セキュリティを非常に重視する組織の要件を満たせるように構築されたデータセンターとネットワークアーキテクチャーから利点を得ます。セキュリティは、 AWS とユーザーの間で共有される責任です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティとして説明しています。

クラウドのセキュリティ — AWS クラウドで提供されるすべてのサービスを実行するインフラストラクチャ AWS を保護し、安全に使用できるサービスを提供します。当社のセキュリティ責任は で最優先事項であり AWS、当社のセキュリティの有効性は、AWS コンプライアンスプログラム の一環として、サードパーティーの監査者によって定期的にテストおよび検証されています。

クラウドにおけるセキュリティ — お客様の責任は、使用している AWS サービス、およびデータの機密性、組織の要件、適用可能な法律や規制などのその他の要因によって決まります。

この AWS 製品またはサービスは、サポートする特定の Amazon Web Services (AWS) サービスを通じて責任共有モデルに従います。 AWS サービスセキュリティ情報については、AWS 「サービスセキュリティドキュメント」ページとAWS 、 AWS コンプライアンスプログラム によるコンプライアンスの取り組みの対象となるサービスを参照してください。

トピック

- でのデータ保護 AWS Toolkit for Visual Studio
- Identity and Access Management
- この AWS 製品またはサービスのコンプライアンス検証
- この AWS 製品またはサービスの耐障害性
- この AWS 製品またはサービスのインフラストラクチャセキュリティ
- での設定と脆弱性の分析 AWS Toolkit for Visual Studio

でのデータ保護 AWS Toolkit for Visual Studio

AWS Toolkit for Visual Studio でのデータ保護には、 責任 AWS 共有モデル責任が適用されます。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。お客様は、このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任があります。また、使用する AWS のサービス のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、

「データプライバシーのよくある質問」を参照してください。欧州でのデータ保護の詳細については、AWS セキュリティブログに投稿された記事「AWS 責任共有モデルおよび GDPR」を参照してください。

データ保護の目的で、 認証情報を保護し AWS アカウント 、 AWS IAM Identity Center または AWS Identity and Access Management (IAM) を使用して個々のユーザーを設定することをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 は必須であり TLS 1.3 がお勧めです。
- で API とユーザーアクティビティのログ記録を設定します AWS CloudTrail。
- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度なマネージドセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API AWS を介して にアクセスするときに FIPS 140-2 検証済みの暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-2」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報は、タグ、または名前フィールドなどの自由形式のテキストフィールドに配置しないことを強くお勧めします。これは、コンソール、API、または AWS CLI SDK を使用して Toolkit for Visual Studio または他の AWS のサービス を使用する場合も同様です。 AWS SDKs 名前に使用する自由記述のテキストフィールドやタグに入力したデータは、課金や診断ログに使用される場合があります。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

Identity and Access Management

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰を認証 (サインイン) し、誰に AWS リソースの使用を承認する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで AWS のサービス 使用できる です。

トピック

- 対象者

- アイデンティティを使用した認証
- ポリシーを使用したアクセスの管理
- IAM の AWS のサービス 仕組み
- AWS ID とアクセスのトラブルシューティング

対象者

AWS Identity and Access Management (IAM) の使用方法は、 で行う作業によって異なります AWS。

サービスユーザー – AWS のサービス を使用してジョブを実行する場合、管理者から必要な認証情報とアクセス許可が与えられます。さらに多くの AWS 機能を使用して作業を行う場合は、追加のアクセス許可が必要になることがあります。アクセスの管理方法を理解しておくと、管理者に適切な許可をリクエストするうえで役立ちます。の機能にアクセスできない場合は AWS、AWS ID とアクセスのトラブルシューティング「」または AWS のサービス 使用している のユーザーガイドを参照してください。

サービス管理者 – 社内の AWS リソースを担当している場合は、通常、 へのフルアクセスがあります AWS。サービスユーザーがどの AWS 機能やリソースにアクセスするかを決めるのは管理者の仕事です。その後、IAM 管理者にリクエストを送信して、サービスユーザーの権限を変更する必要があります。このページの情報を点検して、IAM の基本概念を理解してください。会社で IAM を で使用する方法の詳細については AWS、使用している の AWS のサービス ユーザーガイドを参照してください。

IAM 管理者 - 管理者は、 AWSへのアクセスを管理するポリシーの書き込み方法の詳細について確認する場合があります。IAM で使用できる AWS アイデンティティベースのポリシーの例を表示するには、 AWS のサービス 使用している のユーザーガイドを参照してください。

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。として、IAM ユーザーとして AWS アカウントのルートユーザー、または IAM ロールを引き受けて認証 (にサインイン AWS) される必要があります。

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center （ IAM Identity Center) ユーザー、会社のシングルサインオン認証、Google または Facebook の認証情報は、フェデレーティッド ID の例です。フェデレーティッドアイデンティティとしてサインインする場合、IAM ロールを使用して、前もって管理者により ID

フェデレーションが設定されています。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーのタイプに応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、「 ユーザーガイド」の「 へのサインイン AWS アカウント方法AWS サインイン 」を参照してください。

AWS プログラムで にアクセスする場合、 は Software Development Kit (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、認証情報を使用してリクエストに暗号で署名します。 AWS ツールを使用しない場合は、リクエストに自分で署名する必要があります。推奨される方法を使用してリクエストを自分で署名する方法の詳細については、IAM ユーザーガイドの API AWS リクエストの署名を参照してください。

使用する認証方法を問わず、追加セキュリティ情報の提供をリクエストされる場合もあります。例えば、 AWS では、多要素認証 (MFA) を使用してアカウントのセキュリティを向上させることをお勧めします。詳細については、『AWS IAM Identity Center ユーザーガイド』の「Multi-factor authentication」(多要素認証) および『IAM ユーザーガイド』の「AWSにおける多要素認証 (MFA) の使用」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウント内のすべての およびリソースへの AWS のサービス 完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、『IAM ユーザーガイド』の「ルートユーザー認証情報が必要なタスク」を参照してください。

フェデレーティッドアイデンティティ

ベストプラクティスとして、管理者アクセスを必要とするユーザーを含む人間のユーザーに、一時的な認証情報を使用して にアクセスするための ID プロバイダーとのフェデレーションの使用を要求 AWS のサービス します。

フェデレーティッド ID は、エンタープライズユーザーディレクトリ、ウェブ ID プロバイダー、AWS Directory Service、アイデンティティセンターディレクトリのユーザー、または ID ソースを通じて提供された認証情報 AWS のサービス を使用して にアクセスするユーザーです。フェデレー

ティッド ID が にアクセスすると AWS アカウント、ロールを引き受け、ロールは一時的な認証情報を提供します。

アクセスを一元管理する場合は、 AWS IAM Identity Centerを使用することをお勧めします。IAM Identity Center でユーザーとグループを作成することも、独自の ID ソース内のユーザーとグループのセットに接続して同期して、すべての AWS アカウント とアプリケーションで使用することもできます。IAM Identity Center の詳細については、『AWS IAM Identity Center ユーザーガイド』の「What is IAM Identity Center?」(IAM Identity Center とは) を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を保有する IAM ユーザーを作成する代わりに、一時認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、IAM ユーザーガイドの「長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの集団を指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、IAMAdmins という名前のグループを設定して、そのグループに IAM リソースを管理する権限を与えることができます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時的な認証情報が提供されます。詳細については、『IAM ユーザーガイド』の「IAM ユーザー (ロールではなく) の作成が適している場合」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。これは IAM ユーザーに似ていますが、特定のユーザーには関連付けられていません。ロール を切り替える AWS Management Console ことで、 で IAM ロールを一時的に引き受けることができます。ロールを引き受けるには、 または AWS API AWS CLI オペレーションを呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「IAM ロールの使用」を参照してください。

IAM ロールと一時的な認証情報は、次の状況で役立ちます:

- フェデレーションユーザーアクセス – フェデレーティッドアイデンティティに権限を割り当てるには、ロールを作成してそのロールの権限を定義します。フェデレーティッドアイデンティティが認証されると、そのアイデンティティはロールに関連付けられ、ロールで定義されている権限が付与されます。フェデレーションの詳細については、『IAM ユーザーガイド』の「サードパーティーアイデンティティプロバイダー向けロールの作成」を参照してください。IAM アイデンティティセンターを使用する場合、権限セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM Identity Center は、権限セットを IAM のロールに関連付けます。権限セットの詳細については、『AWS IAM Identity Center ユーザーガイド』の「権限セット」を参照してください。
- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで引き受けることができます。
- クロスアカウントアクセス - IAM ロールを使用して、自分のアカウントのリソースにアクセスすることを、別のアカウントの人物 (信頼済みプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、(ロールをプロキシとして使用する代わりに) リソースに直接ポリシーをアタッチできます。クロスアカウントアクセスにおけるロールとリソースベースのポリシーの違いについては、『IAM ユーザーガイド』の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。
- クロスサービスアクセス — 一部の は、他の の機能 AWS のサービス を使用します AWS のサービス。例えば、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でアプリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスでは、呼び出し元プリンシパルの権限、サービスロール、またはサービスにリンクされたロールを使用してこれを行う場合があります。
  - 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用して でアクションを実行する場合 AWS、ユーザーはプリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストリクエストリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。
  - サービスロール - サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細

については、IAM ユーザーガイドの「 AWS のサービスにアクセス許可を委任するロールの作成」を参照してください。

- サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールの権限を表示できますが、編集することはできません。

- Amazon EC2 で実行されているアプリケーション – IAM ロールを使用して、EC2 インスタンスで実行され、 AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2 インスタンス内でのアクセスキーの保存に推奨されます。 AWS ロールを EC2 インスタンスに割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルにはロールが含まれ、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、『IAM ユーザーガイド』の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用して権限を付与する」を参照してください。

IAM ロールと IAM ユーザーのどちらを使用するかについては、『IAM ユーザーガイド』の「(IAM ユーザーではなく) IAM ロールをいつ作成したら良いのか?」を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、 AWS ID またはリソースにアタッチします。ポリシーは、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) AWS がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーは JSON ドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

デフォルトでは、ユーザーやロールに権限はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き継ぐことができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの権限を定義します。例えば、`iam:GetRole` アクションを許可するポリシーがあるとします。そのポリシーを持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS API からロール情報を取得できます。

アイデンティティベースのポリシー

アイデンティティベースポリシーは、IAM ユーザー、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 権限ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、IAM ユーザーガイドの「IAM ポリシーの作成」を参照してください。

アイデンティティベースポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 内の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。マネージドポリシーまたはインラインポリシーのいずれかを選択する方法については、『IAM ユーザーガイド』の「マネージドポリシーとインラインポリシーの比較」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーでは、IAM の AWS マネージドポリシーを使用できません。

アクセスコントロールリスト (ACL)

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための権限を持つかをコントロールします。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

Amazon S3、 AWS WAF、および Amazon VPC は、ACLs。ACL の詳細については、『Amazon Simple Storage Service デベロッパーガイド』の「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS は、一般的ではない追加のポリシータイプをサポートします。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- アクセス許可の境界 - アクセス許可の境界は、アイデンティティベースのポリシーによって IAM エンティティ (IAM ユーザーまたはロール) に付与できる権限の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。許可の境界の詳細については、「IAM ユーザーガイド」の「IAM エンティティの許可の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPs は、 の組織または組織単位 (OU) に対する最大アクセス許可を指定する JSON ポリシーです AWS Organizations。 AWS Organizations は、 AWS アカウント ビジネスが所有する複数の をグループ化して一元管理するためのサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCP) を一部またはすべてのアカウントに適用できます。SCP は、各 を含むメンバーアカウントのエンティティのアクセス許可を制限します AWS アカウントのルートユーザー。Organizations と SCP の詳細については、『AWS Organizations ユーザーガイド』の「SCP の仕組み」を参照してください。
- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「IAM ユーザーガイド」の「セッションポリシー」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、IAM ユーザーガイドの「ポリシー評価ロジック」を参照してください。

IAM の AWS のサービス 仕組み

ほとんどの IAM 機能との AWS のサービス 連携方法の概要については、IAM ユーザーガイドのAWS「IAM と連携する のサービス」を参照してください。

IAM AWS のサービス で特定の を使用する方法については、関連する サービスのユーザーガイドのセキュリティセクションを参照してください。

AWS ID とアクセスのトラブルシューティング

次の情報は、 と IAM の使用時に発生する可能性がある一般的な問題の診断 AWS と修正に役立ちます。

トピック

- でアクションを実行する権限がない AWS
- iam を実行する権限がありません。PassRole
- 自分の 以外のユーザーに自分の AWS リソース AWS アカウント へのアクセスを許可したい

でアクションを実行する権限がない AWS

「I am not authorized to perform an action in Amazon Bedrock」というエラーが表示された場合、そのアクションを実行できるようにポリシーを更新する必要があります。

次のエラー例は、mateojackson IAM ユーザーがコンソールを使用して、ある *my-example-widget* リソースに関する詳細情報を表示しようとしたことを想定して、その際に必要なawes:*GetWidget* アクセス許可を持っていない場合に発生するものです。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 awes:GetWidget on resource: my-example-widget
```

この場合、awes:*GetWidget* アクションを使用して *my-example-widget*リソースへのアクセスを許可するように、mateojackson ユーザーのポリシーを更新する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

iam を実行する権限がありません。PassRole

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して AWSにロールを渡すことができるようにする必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

以下の例のエラーは、marymajor という IAM ユーザーがコンソールを使用して AWSでアクションを実行しようする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。Mary には、ロールをサービスに渡す権限がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

自分の 以外のユーザーに自分の AWS リソース AWS アカウント へのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセスコントロールリスト (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- がこれらの機能 AWS をサポートしているかどうかを確認するには、「」を参照してくださいIAM の AWS のサービス 仕組み。
- 所有 AWS アカウント している のリソースへのアクセスを提供する方法については、IAM ユーザーガイドの「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティー に提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。

- ID フェデレーションを介してアクセスを提供する方法については、『IAM ユーザーガイド』の「外部で認証されたユーザー (ID フェデレーション) へのアクセス権限」を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いの詳細については、「IAM ユーザーガイド」の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

この AWS 製品またはサービスのコンプライアンス検証

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、コンプライアンスプログラムAWS のサービス による対象範囲内のコンプライアンスプログラムを参照し、関心のあるコンプライアンスプログラムを選択します。一般的な情報については、AWS 「コンプライアンスプログラム」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「 でのレポートのダウンロード AWS Artifact」の」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS のサービス は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境 AWS を にデプロイする手順について説明します。
- アマゾン ウェブ サービスにおける HIPAA セキュリティとコンプライアンスのアーキテクチャ — このホワイトペーパーでは、企業が AWS を使用して HIPAA 対象アプリケーションを作成する方法について説明します。

> **Note**
>
> すべて AWS のサービス HIPAA の対象となるわけではありません。詳細については、「HIPAA 対応サービスのリファレンス」を参照してください。

- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- AWS カスタマーコンプライアンスガイド — コンプライアンスの観点から責任共有モデルを理解します。このガイドでは、ガイダンスを保護し AWS のサービス 、複数のフレームワーク (米国国立標準技術研究所 (NIST)、Payment Card Industry Security Standards Council (PCI)、国際標準化

機構 (ISO) を含む) のセキュリティコントロールにマッピングするためのベストプラクティスをまとめています。

- 「 デベロッパーガイド」の「ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が社内プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – これにより AWS のサービス 、 内のセキュリティ状態を包括的に確認できます AWS。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールのリストについては、「Security Hub のコントロールリファレンス」を参照してください。
- Amazon GuardDuty – これにより AWS アカウント、疑わしいアクティビティや悪意のあるアクティビティがないか環境を監視することで、、、ワークロード、コンテナ、データに対する潜在的な脅威 AWS のサービス を検出します。 GuardDuty は、特定のコンプライアンスフレームワークで義務付けられている侵入検知要件を満たすことで、PCI DSS などのさまざまなコンプライアンス要件への対応に役立ちます。
- AWS Audit Manager – これにより AWS のサービス 、 AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

この AWS 製品またはサービスは、サポートする特定の Amazon Web Services (AWS) サービスを通じて責任共有モデルに従います。 AWS サービスセキュリティ情報については、AWS 「サービスセキュリティドキュメント」ページとAWS 、 AWS コンプライアンスプログラム によるコンプライアンスの取り組みの対象となるサービスを参照してください。

この AWS 製品またはサービスの耐障害性

AWS グローバルインフラストラクチャは、 AWS リージョン およびアベイラビリティーゾーンを中心に構築されています。

AWS リージョン は、低レイテンシー、高スループット、および高度に冗長なネットワークで接続された、物理的に分離および隔離された複数のアベイラビリティーゾーンを提供します。

アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性が高く、フォールトトレラントで、スケーラブルです。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS 「 グローバルインフラストラクチャ」を参照してください。

この AWS 製品またはサービスは、サポートする特定の Amazon Web Services (AWS) サービスを通じて責任共有モデルに従います。 AWS サービスセキュリティ情報については、AWS 「サービスセキュリティドキュメント」ページとAWS 、 AWS コンプライアンスプログラム によるコンプライアンスの取り組みの対象となるサービスを参照してください。

この AWS 製品またはサービスのインフラストラクチャセキュリティ

この AWS 製品またはサービスはマネージドサービスを使用するため、 グローバルネットワークセキュリティによって保護されています AWS 。 AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「 Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

が AWS 公開した API コールを使用して、ネットワーク経由でこの AWS 製品またはサービスにアクセスします。クライアントは以下をサポートする必要があります:

- Transport Layer Security (TLS)。TLS 1.2 は必須で TLS 1.3 がお勧めです。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。

また、リクエストには、アクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

この AWS 製品またはサービスは、サポートする特定の Amazon Web Services (AWS) サービスを通じて責任共有モデルに従います。 AWS サービスセキュリティ情報については、AWS 「サービスセキュリティドキュメント」ページとAWS 、 AWS コンプライアンスプログラム によるコンプライアンスの取り組みの対象となるサービスを参照してください。

での設定と脆弱性の分析 AWS Toolkit for Visual Studio

新しい機能や修正が開発されると、Toolkit for Visual Studio が Visual Studio Marketplace にリリースされます。これらの更新にはセキュリティ更新が含まれることがあるため、 Toolkit for Visual Studio を最新の状態に保つことが重要です。

拡張機能の自動更新が有効になっていることを確認するには

1. [Tools] (ツール) を選択し、Visual Studio 2017 の場合には [Extensions and Updates (Visual Studio 2017)] (拡張機能と更新プログラム)、Visual Studio 2019 の場合には [Extensions] (拡張機能)、[Manage Extensions] (拡張機能の管理) の順に選択します。
2. [Change your Extensions and Updates settings] (拡張機能と更新設定の変更) (Visual Studio 2017) または [Change your settings for Extensions] (拡張機能と設定の変更) (Visual Studio 2019) を選択します。
3. 環境の設定を調整します。

拡張機能の自動更新を無効にする場合は、Toolkit for Visual Studio の更新が環境に適した間隔で実行されていることを必ず確認してください。

AWS Toolkit for Visual Studio ユーザーガイドの文書履歴

ドキュメントの最終更新日: 2021 年 4 月 21 日

ドキュメント履歴

次の表は、 AWS Toolkit for Visual Studio ユーザーガイドの最近の重要な変更点をまとめたものです。このドキュメントの更新に関する通知については、RSS フィードを購読してください。

変更	説明	日付
コンテンツの更新とメンテナンス	UI AWS とスタイルガイドラインの変更に伴うコンテンツの更新。	2024 年 3 月 6 日
コンテンツの更新とメンテナンス	UI AWS とスタイルガイドラインの変更に伴うコンテンツの更新。	2024 年 3 月 6 日
コンテンツの更新とメンテナンス	UI AWS とスタイルガイドラインの変更に伴うコンテンツの更新。	2024 年 3 月 6 日
コンテンツの更新とメンテナンス	UI AWS とスタイルガイドラインの変更に伴うコンテンツの更新。	2024 年 3 月 6 日
コンテンツの更新とメンテナンス	UI AWS とスタイルガイドラインの変更に伴うコンテンツの更新。	2024 年 3 月 6 日
セットアップと認証を更新	セキュリティとツールキットのオンボーディングエクスペリエンスを向上させるため、「セットアップと認証」のトピックが更新されました。変更点については、「開始方法」および「認証とアクセ	2023 年 6 月 22 日

	ス」トピックの目次を参照してください。	
認証とアクセス	AWS 認証情報の提供が「認証とアクセス」になりました。AWS スタイルとセキュリティの要件を満たすように目次とサブトピックをリファクタリングします。	2023 年 5 月 4 日
一般的なトラブルシューティングに関する新しいトピック	「トラブルシューティング」トピックには、および関連サービスの一般的なトラブルシューティング情報が含まれています。 AWS Toolkit for Visual Studio	2023 年 4 月 30 日
「セットアップ」セクションとトピックを更新	AWS Toolkit for Visual Studio のオンボーディングエクスペリエンスを向上させるため、このユーザーガイドの「AWS Toolkit for Visual Studioのセットアップ」のセクションとトピックが更新されました。	2023 年 1 月 30 日
「セットアップ」セクションとトピックを更新	AWS Toolkit for Visual Studio のオンボーディングエクスペリエンスを向上させるため、このユーザーガイドの「AWS Toolkit for Visual Studioのセットアップ」のセクションとトピックが更新されました。	2023 年 1 月 30 日
2022 AWS Toolkit for Visual Studio 年の情報を追加しました。	ビジュアルスタジオ 2022 Support がに追加されました AWS Toolkit for Visual Studio。	2022 年 12 月 20 日

「公開」 AWS ガイドの更新	一般提供の開始に向けてサービスに加えられた変更を反映するために、ドキュメントを更新しました。	2022 年 7 月 6 日
タイトルの更新と再配置	内容をより正確に反映するため、タイトルを若干変更しました。ガイドは「パブリッシング先」 AWS ガイドに配置されました。	2022 年 7 月 6 日
配信先 AWS:タイトルとコンテンツの更新	ガイドセクションの正式タイトルは「 AWS ツールキットを使用したデプロイ」でしたが、目次 (TOC) が更新され、「デプロイ先」というタイトルになりました。 AWS Elastic Beanstalk へのデプロイ (レガシー) とデプロイ (レガシー) のガイドは廃止され、アクセスできなくなりました。 AWS CloudFormation Elastic Beanstalk と Cloudformation へのデプロイに関する更新されたコンテンツは、このガイドの更新された目次から参照できます。	2022 年 7 月 6 日
「ASP.NET Core 2.0 アプリケーションのデプロイ (Fargate)」はレガシーガイドになりました	このドキュメントでは、従来のサービスと機能について言及しています。更新されたガイドとコンテンツについては、「AWS .NET Deployment tool」ガイドと更新された目次の「AWSへのデプロイ」を参照してください。	2022 年 7 月 6 日

「ASP.NET アプリケーションのデプロイ」はレガシーガイドになりました	このドキュメントでは、従来のサービスと機能について言及しています。更新されたガイドとコンテンツについては、「AWS .NET Deployment tool」ガイドと、目次にある更新された「AWSへのデプロイ」を参照してください。	2022 年 7 月 6 日
「ASP.NET アプリケーションのデプロイ」はレガシーガイドになりました	このドキュメントでは、従来のサービスと機能について言及しています。更新されたガイドとコンテンツについては、「AWS .NET Deployment tool」ガイドと、目次にある更新された「AWSへのデプロイ」を参照してください。	2022 年 7 月 6 日
新しいガイドトピック:Visual Studio での CloudWatch ログの操作	Visual Studio ガイドの Amazon CloudWatch ログ統合の概要トピックを新たに作成しました。	2022 年 1 月 29 日
新しいガイドトピック:Visual Studio CloudWatch のログ統合のセットアップ	Visual Studio ガイドの Amazon CloudWatch Logs インテグレーションの新しいセットアップセクションを作成しました。	2022 年 1 月 29 日

CloudWatch Visual Studio のログ統合	Visual Studio に Amazon CloudWatch Logs を統合するための新しいガイドを作成しました。ガイドのトピック「Visual Studio CloudWatch のログのセットアップ」と「Visual Studio CloudWatch でのログの操作」が含まれています。	2022 年 1 月 29 日
にパブリッシュします。 AWS	AWS 公開先はプレビュー版ではなくなりました。UI の変更ならびに発行の提案に関する改善を反映するために更新しました。	2022 年 6 月 1 日
AWS 新規公開先はプレビュー可能です	AWS どのサービスがアプリケーションに適しているかについてのガイダンスを提供する、強化された導入エクスペリエンス。	2021 年 10 月 21 日
認証情報の SSO と MFA のサポート AWS	AWS 認証情報におけるシングル・サインオン (IAM Identity Center) と多要素認証の新規サポートを文書化するために更新されました。 AWS	2021 年 4 月 21 日
基本プロジェクト AWS Lambda :Docker イメージの作成	Lambda コンテナイメージのサポートが追加されました。	2020 年 12 月 1 日
セキュリティコンテンツ	セキュリティコンテンツを追加しました。	2020 年 2 月 6 日

認証情報の提供 AWS	共有 AWS 認証情報ファイルでの認証情報プロファイルの作成に関する情報が更新されました。	2019年6月20日
AWS Toolkit for Visual Studio の AWS Lambda プロジェクトの使用	Visual Studio 2019 Support が AWS VisToolkit for Visual Studio に追加されました。	2019 年 3 月 28 日
チュートリアル: Amazon Rekognition Lambda アプリケーションの作成	Visual Studio 2019 Support が AWS VisToolkit for Visual Studio に追加されました。	2019 年 3 月 28 日
チュートリアル:Lambda AWS によるサーバーレスアプリケーションの構築とテスト	Visual Studio 2019 Support が AWS VisToolkit for Visual Studio に追加されました。	2019 年 3 月 28 日
のセットアップ AWS Toolkit for Visual Studio	ビジュアルスタジオ 2019 Support がに追加されました AWS Toolkit for Visual Studio。	2019 年 3 月 28 日
ASP.NET Core 2.0 アプリケーションのデプロイ (Fargate)	Visual Studio 2019 Support が AWS VisToolkit for Visual Studio に追加されました。	2019 年 3 月 28 日
ASP.NET Core 2.0 アプリケーションのデプロイ (EC2)	Visual Studio 2019 Support が AWS VisToolkit for Visual Studio に追加されました。	2019 年 3 月 28 日
Visual Studio AWS CloudFormation でのテンプレートプロジェクトの作成	Visual Studio 2019 Support が AWS VisToolkit for Visual Studio に追加されました。	2019 年 3 月 28 日

Container Service の詳細ビュー	AWS Explorer によって提供される Amazon Elastic Container Service クラスターとコンテナリポジトリの詳細ビューに関する情報を追加しました。	2018 年 2 月 16 日
Amazon EC2 Container Service へのデプロイ	Amazon EC2 Container Service へのデプロイに関する情報を追加しました。	2018 年 2 月 16 日
Fargate の使用による Container Service のデプロイ	Fargate 起動タイプを使用して Amazon ECS を通じて Linux をターゲットとするコンテナ化された ASP.NET Core 2.0 アプリケーションをデプロイする方法についての情報を追加しました。	2018 年 2 月 16 日
EC2 の使用による Container Service のデプロイ	EC2 起動タイプを使用して Amazon ECS を通じて Linux をターゲットとするコンテナ化された ASP.NET Core 2.0 アプリケーションをデプロイする方法についての情報を追加しました。	2018 年 2 月 16 日
Amazon EC2 Container Service にデプロイするための認証情報	Amazon EC2 Container Service にデプロイするときに認証情報を指定する方法についての情報を追加しました。	2018 年 2 月 16 日

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。