User Guide

Amazon Toolkit with Amazon Q

Amazon Toolkit with Amazon Q: User Guide

Table of Contents

Amazon Toolkit with Amazon Q 1
What is the Amazon Toolkit for Visual Studio with Amazon Q 1
Amazon Explorer 1
Amazon Q 1
Related Information 2
Amazon Q 3
What is Amazon Q 3
Download the Toolkit 4
Downloading the Toolkit from the Visual Studio Marketplace 4
Additional IDE Toolkits from Amazon 4
Getting Started 5
Installation and set up 5
Prerequisites 5
Installing the Amazon Toolkit 6
Uninstalling the Amazon Toolkit 7
Connecting to Amazon 9
Prerequisites 9
Connecting to Amazon from the Toolkit 10
Amazon Q Developer 10
Amazon Toolkit 1
Documentation and Tutorials 14
Troubleshooting installation issues 15
Administrator permissions for Visual Studio 15
Obtaining an installation log 16
Installing different Visual Studio extensions 17
Contacting support 17
Profiles and Window Binding 17
Profiles and Window Binding for the Toolkit for Visual Studio 17
Authentication and access 19
IAM Identity Center 19
Authenticating with IAM Identity Center from the Amazon Toolkit for Visual Studio 20
IAM credentials 21
Creating an IAM user 22
Creating a credentials file 22

Editing IAM user credentials from the toolkit ... 23
Editing IAM user credentials from a text editor ... 24
Creating IAM users from the Amazon Command Line Interface (Amazon CLI) ... 24
Amazon Builder ID ... 25
Multi-factor authentication (MFA) ... 25
Step 1: Creating an IAM role to delegate access to IAM users ... 25
Step 2: Creating an IAM user that assumes the role's permissions ... 26
Step 3: Adding a policy to allow the IAM user to assume the role ... 26
Step 4: Managing a virtual MFA device for the IAM user ... 27
Step 5: Creating profiles to allow MFA ... 28
External credentials ... 29
Updating firewalls and gateways ... 29
Amazon Toolkit for Visual Studio Endpoints ... 29
Amazon Q plugin endpoints ... 30
Amazon Q Developer endpoints ... 30
Amazon Q Code Transform Endpoints ... 31
Authentication endpoints ... 31
Identity Endpoints ... 31
Telemetry ... 32
References ... 32
Working with Amazon Services ... 34
Amazon CodeCatalyst ... 34
What is Amazon CodeCatalyst? ... 34
Getting Started with CodeCatalyst ... 35
Working with CodeCatalyst ... 36
Troubleshooting ... 38
CloudWatch Logs integration ... 39
Setting up CloudWatch Logs ... 39
Working with CloudWatch Logs ... 39
Managing Amazon EC2 Instances ... 46
The Amazon Machine Images and Amazon EC2 Instances Views ... 46
Launching an Amazon EC2 Instance ... 48
Connecting to an Amazon EC2 Instance ... 51
Ending an Amazon EC2 Instance ... 54
Managing Amazon ECS Instances ... 57
Modifying service properties ... 58

Stopping a task ... 58
Deleting a service ... 58
Deleting a cluster ... 59
Creating a repository ... 59
Deleting a repository ... 59
Managing Security Groups from Amazon Explorer ... 60
Creating a Security Group ... 60
Adding Permissions to Security Groups ... 61
Creating an AMI from an Amazon EC2 Instance ... 62
Setting Launch Permissions on an Amazon Machine Image ... 63
Amazon Virtual Private Cloud (VPC) ... 64
Creating a Public-Private VPC for Deployment with Amazon Elastic Beanstalk ... 65
Using the Amazon CloudFormation Template Editor for Visual Studio ... 69
Creating an Amazon CloudFormation Template Project in Visual Studio ... 70
Deploying a Amazon CloudFormation Template in Visual Studio ... 73
Formatting a Amazon CloudFormation Template in Visual Studio ... 76
Using Amazon S3 from Amazon Explorer ... 77
Creating an Amazon S3 Bucket ... 78
Managing Amazon S3 Buckets from Amazon Explorer ... 78
Uploading Files and Folders to Amazon S3 ... 80
Amazon S3 File Operations from Amazon Toolkit for Visual Studio ... 82
Using DynamoDB from Amazon Explorer ... 85
Creating an DynamoDB Table ... 86
Viewing an DynamoDB Table as a Grid ... 87
Editing and Adding Attributes and Values ... 88
Scanning an DynamoDB Table ... 90
Using Amazon CodeCommit with Visual Studio Team Explorer ... 91
Credential Types for Amazon CodeCommit ... 91
Connecting to Amazon CodeCommit ... 92
Creating a Repository ... 94
Setting up Git Credentials ... 94
Cloning a Repository ... 97
Working with Repositories ... 98
Using CodeArtifact in Visual Studio ... 98
Add your CodeArtifact repository as a NuGet package source ... 99
Amazon RDS from Amazon Explorer ... 99

Launch an Amazon RDS Database Instance 100
Create a Microsoft SQL Server Database in an RDS Instance 108
Amazon RDS Security Groups 109
Using Amazon SimpleDB from Amazon Explorer 113
Using Amazon SQS from Amazon Explorer 115
Creating a Queue 115
Deleting a Queue 116
Managing Queue Properties 116
Sending a Message to a Queue 117
Identity and Access Management 118
Create and Configure an IAM User 119
Create an IAM Group 120
Add an IAM User to an IAM Group 121
Generate Credentials for an IAM User 123
Create an IAM Role 125
Create an IAM Policy 126
Amazon Lambda 129
Basic Amazon Lambda Project 129
Basic Amazon Lambda Project Creating Docker Image 135
Tutorial: Build and Test a Serverless Application with Amazon Lambda 143
Tutorial: Creating an Amazon Rekognition Lambda Application 149
Tutorial: Using Amazon Logging Frameworks with Amazon Lambda to Create Application Logs 157
Deploying to Amazon 160
Publish to Amazon 160
Prerequisites 161
Supported application types 162
Publishing applications to Amazon targets 162
Amazon Lambda 164
Prerequisites 164
Related topics 165
Listing the Lambda Commands Available through the .NET Core CLI 165
Publishing a .NET Core Lambda Project from the .NET Core CLI 166
Deploying to Amazon Elastic Beanstalk 168
Deploy an ASP.NET App (Traditional) 169
Deploy an ASP.NET App (.NET Core) (Legacy) 181

Specify Amazon Credentials ... 183
Republish to Elastic Beanstalk (Legacy) ... 184
Custom Deployments (Traditional) ... 186
Custom Deployments (.NET Core) ... 188
Multiple Application Support ... 192
Deploying to Amazon EC2 Container Service ... 195
Specify Amazon Credentials ... 196
Deploy an ASP.NET Core 2.0 App (Fargate) (Legacy) ... 198
Deploy an ASP.NET Core 2.0 App (EC2) ... 206
Troubleshooting ... 211
Troubleshooting best practices ... 211
Viewing and filtering Amazon Q security scans ... 212
The Amazon Toolkit isn't properly installed ... 213
Firewall and proxy settings ... 214
Troubleshooting firewall and proxy settings ... 214
Custom certificates ... 214
Allow listing and additional steps ... 215
Security ... 217
Data Protection ... 217
Identity and Access Management ... 218
Audience ... 219
Authenticating with identities ... 219
Managing access using policies ... 222
How Amazon Web Services services work with IAM ... 225
Troubleshooting Amazon identity and access ... 225
Compliance Validation ... 227
Resilience ... 228
Infrastructure Security ... 229
Configuration and Vulnerability Analysis ... 229
Document history ... 231
Document history ... 231

Amazon Toolkit with Amazon Q

This is the user guide for the **Amazon Toolkit for Visual Studio with Amazon Q**. If you are looking for the **Amazon Toolkit for VS Code**, see the *User Guide for the Amazon Toolkit for Visual Studio Code*.

What is the Amazon Toolkit for Visual Studio with Amazon Q

The Amazon Toolkit for Visual Studio with Amazon Q is an extension for the Visual Studio IDE that makes it easier for you to develop, debug, and deploy .NET applications that use Amazon Web Services. The Amazon Toolkit with Amazon Q is supported for Visual Studio versions 2019 and later. For details about how to download and install the kit, see the Installation and set up topic in this User Guide.

> **Note**
>
> The Toolkit for Visual Studio was also released for Visual Studio 2008, 2010, 2012, 2013, 2015, and 2017 versions. However, those versions are no longer supported. For more information, see the Installation and set up topic in this User Guide.

The Amazon Toolkit with Amazon Q contains the following features to enhance your development experience.

Amazon Explorer

The Amazon Explorer tool window is accessible in the IDE's **View** menu and enables you to interact with Amazon services in Visual Studio. For a list of supported Amazon services and features, see the Working with Amazon Services topic in this User Guide.

Amazon Q

Chat with Amazon Q Developer in Visual Studio to ask questions about building at Amazon and for assistance with software development. Amazon Q can explain coding concepts and code snippets, generate code and unit tests, and improve code through debugging or refactoring.

To install and set up Amazon Q for the Toolkit for Visual Studio, see the Getting started topic in this User Guide. To learn more about working with Amazon Q Developer, see the Amazon Q

Developer in IDEs topic in the *Amazon Q Developer* User Guide. For detailed information about plans and pricing for Amazon Q, see the Amazon Q pricing guide.

Related Information

To open an issue or view currently open issues, visit https://github.com/aws/aws-toolkit-visual-studio/issues.

To learn more about Visual Studio, visit https://visualstudio.microsoft.com/vs/.

Amazon Q

What is Amazon Q

As of April 30th 2024, Amazon CodeWhisperer is now part of Amazon Q Developer, this includes inline code suggestions and security scans.

To learn more about working with Amazon Q Developer in the Amazon Toolkit for Visual Studio, see the Amazon Q Developer in IDEs topic in the *Amazon Q Developer* User Guide. For detailed information about plans and pricing for Amazon Q, see the Amazon Q pricing guide.

Downloading the Toolkit for Visual Studio

You can download, install, and set up the Toolkit for Visual Studio through the Visual Studio Marketplace in your IDE. For detailed instructions, see the Installing the Amazon Toolkit for Visual Studio section in the *Getting started* topic of this User Guide.

Downloading the Toolkit from the Visual Studio Marketplace

Download the Toolkit for Visual Studio installation files by navigating to the **Amazon Visual Studio dowloads** site in your web browser.

Additional IDE Toolkits from Amazon

In addition to the Toolkit for Visual Studio, Amazon also offers IDE Toolkits for VS Code and JetBrains.

Amazon Toolkit for Visual Studio Code links

- Follow this link to Download the Amazon Toolkit for Visual Studio Code from the VS Code Marketplace.
- To learn more about the Amazon Toolkit for Visual Studio Code, see the Amazon Toolkit for Visual Studio Code User Guide.

Amazon Toolkit for JetBrains links

- Follow this link to Download the Amazon Toolkit for JetBrains from the JetBrains Marketplace.
- To learn more about the Amazon Toolkit for JetBrains, see the Amazon Toolkit for JetBrains User Guide.

Getting Started

The Amazon Toolkit for Visual Studio makes your Amazon services and resources available from the Visual Studio integrated development environment (IDE).

To assist you with getting started, the following topics describe how to install, set up, and configure the Amazon Toolkit for Visual Studio.

Topics

- Installing and setting up the Amazon Toolkit for Visual Studio
- Connecting to Amazon
- Troubleshooting installation issues for the Amazon Toolkit for Visual Studio
- Profiles and Window Binding

Installing and setting up the Amazon Toolkit for Visual Studio

The following topics describe how to download, install, set up, and uninstall the Amazon Toolkit for Visual Studio.

Topics

- Prerequisites
- Installing the Amazon Toolkit for Visual Studio
- Uninstalling the Amazon Toolkit for Visual Studio

Prerequisites

The following are prerequisites for setting up supported versions of the Amazon Toolkit for Visual Studio.

- Visual Studio 19 or a later release
- Windows 10 or a later Windows release
- Administrator access to Windows and Visual Studio
- Active Amazon IAM Credentials

Note

Unsupported versions of the Amazon Toolkit for Visual Studio are available for Visual Studio 2008, 2010, 2012, 2013, 2015, and 2017. To download an unsupported version, navigate to the Amazon Toolkit for Visual Studio landing page and choose the version you want from the list of download links.
To learn more about IAM credentials or sign up for an account, visit the Amazon Console gateway.

Installing the Amazon Toolkit for Visual Studio

To install the Amazon Toolkit for Visual Studio, find your version of Visual Studio from the following procedures and complete the necessary steps. Download links for all versions of the Amazon Toolkit for Visual Studio can be found at the Amazon Toolkit for Visual Studio landing page.

Note

If you encounter issues while installing the Amazon Toolkit for Visual Studio, see the Troubleshooting installation issues topic in this guide.

Installing the Amazon Toolkit for Visual Studio for Visual Studio 2022

To install Amazon Toolkit for Visual Studio 2022 from Visual Studio, complete the following steps:

1. From the **Main menu**, navigate to **Extensions** and choose **Manage Extensions**.
2. From the search box, search for *Amazon*.
3. Choose the **Download** button for the relevant version of **Visual Studio 2022** and follow the installation prompts.

   Note

   You may need to manually close and restart Visual Studio to complete the installation process.

4. When the download and installation are complete, you can open the Amazon Toolkit for Visual Studio by choosing **Amazon Explorer** from the **View** menu.

Installing the Amazon Toolkit for Visual Studio for Visual Studio 2019

To install Amazon Toolkit for Visual Studio 2019 from Visual Studio, complete the following steps:

1. From the **Main menu**, navigate to **Extensions** and choose **Manage Extensions**.
2. From the search box, search for *Amazon*.
3. Choose the **Download** button for **Visual Studio 2017 and 2019** and follow the prompts.

> **Note**
>
> You may need to manually close and restart Visual Studio to complete the installation process.

4. When the download and installation are complete, you can open the Amazon Toolkit for Visual Studio by choosing **Amazon Explorer** from the **View** menu.

Uninstalling the Amazon Toolkit for Visual Studio

To uninstall the Amazon Toolkit for Visual Studio, find your version of Visual Studio from the following procedures and complete the necessary steps.

Uninstalling the Amazon Toolkit for Visual Studio for Visual Studio 2022

To Uninstall Amazon Toolkit for Visual Studio 2022 from Visual Studio, complete the following steps:

1. From the **Main menu**, navigate to **Extensions** and choose **Manage Extensions**.
2. From the **Manage Extensions** navigation menu, expand the **Installed** heading.
3. Locate the **Amazon Toolkit for Visual Studio 2022** extension and choose the **Uninstall** button.

> **Note**
>
> If the Amazon Toolkit for Visual Studio isn't visible from the **Installed** section of the navigation menu, you may need to restart Visual Studio.

4. Follow the onscreen prompts to complete the uninstall process.

Uninstalling the Amazon Toolkit for Visual Studio for Visual Studio 2019

To uninstall Amazon Toolkit for Visual Studio 2019 from Visual Studio, complete the following steps:

1. From the **Main menu**, navigate to **Tools** and choose **Manage Extensions**.
2. From the **Manage Extensions** navigation menu, expand the **Installed** heading.
3. Locate the **Amazon Toolkit for Visual Studio 2019** extension and choose the **Uninstall** button.
4. Follow the onscreen prompts to complete the uninstall process.

Uninstalling the Amazon Toolkit for Visual Studio for Visual Studio 2017

To uninstall Amazon Toolkit for Visual Studio 2017 in Visual Studio, complete the following steps:

1. From the **Main** menu, navigate to **Tools** and choose **Extensions and Updates**.
2. From the **Extensions and Updates** navigation menu, expand the **Installed** heading.
3. Locate the **Amazon Toolkit for Visual Studio 2017** extension and choose the **Uninstall** button.
4. Follow the onscreen prompts to complete the uninstall process.

Uninstalling the Amazon Toolkit for Visual Studio for Visual Studio 2013 or 2015

To uninstall Amazon Toolkit for Visual Studio 2013 or 2015, complete the following steps:

1. From your Windows Control Panel, open **Programs and Features**.

> **Note**
>
> You can open Programs and Features immediately by running `appwiz.cpl` from a Windows command prompt or the Windows **Run** dialog.

2. From the list of installed programs, open the context menu for (right-click) **Amazon Tools for Windows**.
3. Choose **Uninstall** and follow the prompts to complete the uninstall process.

> **Note**
>
> Your **Samples** directory isn't deleted during the uninstall process. This directory is preserved in case you have modified samples. This directory must be manually removed.

Connecting to Amazon

The following sections describe how to get started with the Amazon Toolkit for Visual Studio with Amazon Q. The first time you launch Visual Studio after installing the extension, a **Getting Started** displays in the editor window. From the **Getting Started** tab you can complete the following actions.

- Enable or disable Amazon Q and the Amazon Toolkit.
- Add and authenticate with new credentials.
- Authenticate with existing credentials.
- Access documentation and tutorials to help you get started working with Amazon Q and the Amazon Toolkit.

Prerequisites

To get started working with Amazon Q and the Amazon Toolkit, you need to authenticate with Amazon credentials. If you've previously set up an Amazon account and authentication through another Amazon tool or service (such as the Amazon Command Line Interface), then the Amazon Toolkit automatically detects your credentials. If you're new to Amazon or haven't created an

account, then you can sign up for an Amazon account from the Amazon sign up portal. For detailed information on setting up a new Amazon account, see the Overview topic in the *Amazon Setup User Guide*.

Connecting to Amazon from the Toolkit

To connect to your Amazon accounts from the Amazon Toolkit, open the **Getting Started** tab at any time by completing the following.

Opening the Getting Started tab in Visual Studio

1. From Visual Studio, expand **Extensions** from the main menu and then expand the **Amazon Toolkit** sub-menu.
2. Choose **Getting started**.
3. The **Getting Started** tab opens in the Visual Studio editor window.

From the **Getting Started** tab, there are 2 main sections:

- **Features:** In this section you can enable or disable features like Amazon Q and the Amazon Toolkit.
- **Documentation and Tutorials:** A collection of references to your enabled features.

> **Note**
>
> The Documentation and Tutorials section is only visible when one or more features is enabled.

Amazon Q Developer

From the Amazon Q section in the **Getting Started** tab, you can enable or disable Amazon Q, add a new connection, or switch to a different Amazon connection. Before you can view or access any of these actions, Amazon Q must be enabled. To enable Amazon Q click the **Enable** button.

When Amazon Q is disabled, all Amazon Q features and functions are completely removed from Visual Studio. Enabling Amazon Q automatically opens the **Setup authentication for Amazon Q** in the **Getting Started** tab. To proceed, you must authenticate with your Amazon IAM Identity Center credentials to access the **Professional Tier** or your Amazon Builder ID to access the **Free**

Tier. For detailed information about each of the tier options, see the Understanding tiers of service for Amazon Q Developer topic in the *Amazon Q Developer* User Guide.

To proceed complete one of the following procedures.

Professional tier authentication with IAM Identity Center

> **Note**
>
> The **Profile Name**, **Start URL**, **Profile Region**, or **SSO Region** fields that are required to authenticate with the Professional tier are typically provided by an administrator at your company or organization. For detailed information about IAM Identity Center credentials, see the What is IAM Identity Center topic in the *Amazon IAM Identity Center* User Guide.

1. From the **Getting Started: Amazon Toolkit with Amazon Q** screen, choose the **Sign in** button in the Amazon Q tile to navigate to the **Setup authentication for Amazon Q** screen.
2. From the **Setup authentication for Amazon Q** screen, navigate to the **Professional tier** section, fill in the required fields and choose the **Connect** button.
3. Confirm that you want to open the Amazon Authorize request portal in your default web browser.
4. Complete the steps required by the Amazon Authorize request portal, you're notified when it's safe to close your browser and return to Visual Studio
5. In the **Getting Started** tab, Amazon Q updates to show you're connected with IAM Identity Center when the process is complete.

Free tier authentication with Amazon Builder ID

> **Note**
>
> For additional details about Amazon Builder ID, see the Sign in with Amazon Builder ID topic in the *Amazon Sign-in* User Guide.

1. From the **Getting Started: Amazon Toolkit with Amazon Q** screen, choose the **Sign in** button in the Amazon Q tile to navigate to the **Setup authentication for Amazon Q** screen.

2. From the **Setup authentication for Amazon Q** screen, navigate to the **Free Tier** section and choose the **Sign up or Sign in** button.
3. Confirm that you want to open the Amazon Authorize request portal in your default web browser.
4. Complete the steps required by the Amazon Authorize request portal, you're notified when it's safe to close your browser and return to Visual Studio.
5. In the **Getting Started** tab, Amazon Q updates to show you're connected with your Amazon Builder ID when the process is complete.

After you've authenticated with either your IAM Identity Center or Amazon Builder ID credentials, you can access Amazon Q in Visual Studio. Additionally, you're able to perform the following actions in the **Getting Started** tab:

- **Sign out**: disconnects your current credential connection from all Amazon Q functions. Amazon Q remains enabled, but most features don't work.
- **Disable Amazon Q**: Completely disables all Amazon Q features in Visual Studio.

Amazon Toolkit

From the Amazon Toolkit section in the **Getting Started with the Amazon Toolkit** tab, you can enable or disable the Amazon Toolkit, add a new connection, or switch to a different Amazon connection. Before you can view or access any of these actions, the Amazon Toolkit must be enabled. To enable the Amazon Toolkit, click the **Enable** button.

When the Amazon Toolkit is enabled, **Setup authentication for Amazon Toolkit** automatically loads in the **Getting Started with the Amazon Toolkit** tab. To proceed, you must authenticate with either your **Amazon IAM Identity Center** credentials or your **IAM User Role** credentials.

> **Note**
>
> For detailed information about IAM Identity Center credentials, see the What is IAM Identity Center topic in the *Amazon IAM Identity Center* User Guide. For detailed information about IAM User Role credentials, see the Amazon Access keys: Long-term credentials topic in the *Amazon SDKs and Tools* reference guide.

Authenticate and connect with IAM Identity Center

1. From the **Getting Started: Amazon Toolkit with Amazon Q** screen, choose the **Sign in** button in the Amazon Toolkit tile to navigate to the **Setup authentication for Amazon Toolkit** screen.
2. From the **Setup Authentication for Amazon Toolkit** screen, choose **IAM Identity Center (Successor to Single Sign-on)** from the **Profile Type** drop-down menu.
3. From the **Choose from an existing Profile or add new** drop-down menu, choose an existing profile or select **Add new profile** to add new profile information.

   > **Note**
   >
   > If you choose an existing profile, go to step 7.

4. In the **Profile Name** field, enter the **`profile name`** associated with the IAM Identity Center account that you want to authenticate with.
5. In the **Start URL** text field, enter the **`Start URL`** that's attached to your IAM Identity Center credentials.
6. From the **Profile Region (defaults to us-east-1)** drop-down menu, choose the **Profile Region** that's defined by the IAM Identity Center user profile you're authenticating with.
7. From the **SSO Region (defaults to us-east-1)** drop-down menu, choose the **SSO Region** that's defined by your IAM Identity Center credentials.
8. Choose the **Connect** button to open the **Amazon Authorize request** site in your default web browser.
9. Follow the prompts in your default web browser, you're notified when the authorization process is complete, it's safe to close your browser, and return to Visual Studio.
10. In the **Getting Started** tab, the Amazon Toolkit section updates to show you're connected with IAM Identity Center when the process is complete.

Authenticate and connect with IAM User Role credentials

1. From the **Getting Started: Amazon Toolkit with Amazon Q** screen, choose the **Sign in** button in the Amazon Toolkit tile to navigate to the **Setup authentication for Amazon Toolkit** screen.
2. From the **Setup authentication for Amazon Toolkit** screen, choose **IAM User Role** from the **Profile Type** drop-down menu.

3. In the **Choose from an existing Profile or add new** drop-down menu, choose **`Add new profile`**.

   > **Note**
   >
   > If you're choosing an existing profile name from the list, skip to **Step 8**.

4. In the **Profile Name** text field, enter a name for your new profile.
5. In the **Access Key ID** text field, enter the **`Access Key ID`** for the profile you want to authenticate with.
6. In the **Secret Key** text field, enter the **`Secret Key`** for the profile you want to authenticate with.
7. From the **Storage Location (defaults to Shared Credentials File)** drop-down menu, specify whether you want to store your credentials with a **Shared Credentials** file or with **.NET Encrypted Store**.
8. From the **Profile Region (defaults to us-east-1)** drop-down menus, choose the **Partition** and **Profile Region** that are attached to the profile you want to authenticate with.
9. Choose the **Connect** button to add this profile to your Amazon storage location and/or authenticate with Amazon.
10. In the **Getting Started** tab, the Amazon Toolkit section updates to show you're connected with your IAM User role credentials when the process is complete.

After you've authenticated with either your IAM Identity Center or IAM User Role credentials, you can access the Amazon Explorer in the Toolkit for Visual Studio. Additionally, you're able to **Sign out** and **Disable the Amazon Toolkit for Visual Studio with Amazon Q** from the **Getting Started** tab.

Documentation and Tutorials

The documentation and Tutorials section automatically updates with documentation and tutorial suggestions based on your Amazon service and feature preferences. These references are only visible when at least one feature has been enabled.

Troubleshooting installation issues for the Amazon Toolkit for Visual Studio

The following information is known to resolve common installation issues while setting up the Amazon Toolkit for Visual Studio.

If you encounter an error while installing the Amazon Toolkit for Visual Studio or it's unclear whether or not the installation was complete, review the information in each of the following sections.

Administrator permissions for Visual Studio

The Amazon Toolkit for Visual Studio extension requires administrator permissions to ensure that all Amazon services and features are accessible.

If you have local administrator permissions it's possible that your administrator permissions don't extend directly to your Visual Studio instance.

To launch Visual Studio with administrator permissions locally:

1. From Windows, locate the Visual Studio application launcher (icon).
2. Open the context menu for (right-click) the Visual Studio icon to open the context menu.
3. Select **Run as administrator** from the context menu.

To launch Visual Studio with administrator permissions remotely:

1. From Windows, locate the application launcher for the application that you are using to connect to your remote instance of Visual Studio.
2. Open the context menu for (right-click) the application to open the context menu.
3. Select **Run as administrator** from the context menu.

> **Note**
>
> Whether you are launching the program locally or connecting remotely, Windows may prompt you to confirm your administrative credentials.

Obtaining an installation log

If you have completed the steps in the previous *Administrator permissions* section located above and it's confirmed that you're running or connecting to Visual Studio with administrator permissions, then obtaining an installation log file can help diagnose other issues.

To manually install the Amazon Toolkit for Visual Studio from a `.vsix` file and generate an installation log file, complete the following steps.

1. From the Amazon Toolkit for Visual Studio landing page, follow the **Download** link and save the `.vsix` file of the Amazon Toolkit for Visual Studio version you want to install.
2. From the Visual Studio main menu, expand the **Tools** header, expand the **Command Line** sub menu, then choose **Visual Studio Developer Command Prompt**.
3. From the **Visual Studio Developer Command Prompt** enter the `vsixinstaller` command with the following format:

   ```
   vsixinstaller /logFile:[file path to log file] [file path to Toolkit installation file]
   ```
4. Replace`[file path to log file]` with the file name and full file path of the directory you want the installation log to be created in. An example of the `vsixinstaller` command with your specified file path and file name resembles the following:

   ```
   vsixinstaller /logFile:C:\Users\Documents\install-log.txt [file path to AWSToolkitPackage.vsix]
   ```
5. Replace `[file path to Toolkit installation file]` with the full file path of the directory where the `AWSToolkitPackage.vsix` is located.

   An example of the `vsixinstaller` command with the full file path to the Toolkit installation file should resemble the following:

   ```
   vsixinstaller /logFile:[file path to log file] C:\Users\Downloads\AWSToolkitPackage.vsix
   ```
6. Check to make sure your file name and paths are correct, then run the `vsixinstaller` command.

   An example of a complete `vsixinstaller` command resembles the following:

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

Installing different Visual Studio extensions

If you've obtained an installation log file and you're still unable to determine why the installation process is failing, check to see if you're able to install other Visual Studio extensions. Installing different Visual Studio extensions can provide additional insight to your installation issues.
In the event that you're unable to install any Visual Studio extensions, it may be necessary to troubleshoot issues with Visual Studio, instead of Amazon Toolkit for Visual Studio.

Contacting support

If you've reviewed all of the sections contained in this guide and require additional resources or support, you can view past issues or open a new issue from the Amazon Toolkit for Visual Studio Github Issues site.

To help expedite a solution to your issue:

- Check past and current issues to see if others have encountered a similar situation.
- Keep detailed notes of each step you've taken to address the issue.
- Save any log files you've obtained from installing the Amazon Toolkit for Visual Studio or other extensions.
- Attach your Amazon Toolkit for Visual Studio installation logfiles to the new issue.

Profiles and Window Binding

Profiles and Window Binding for the Toolkit for Visual Studio

When working with the publishing tools, wizards, and other features of the Toolkit for Visual Studio, take note of the following:

- The Amazon Explorer window is bound to a single profile and region at a time. Windows opened from the Amazon Explorer default to that bound profile and region.
- After a new window has been opened, you can use that instance of the Amazon Explorer to switch to a different profile or region.

- The Toolkit for Visual Studio publishing tools and features automatically default to the profile and region set in the Amazon Explorer.
- If a new profile or region is specified in a publishing tool, wizard, or feature: all resources created afterwards will continue to use the new profile and region settings.
- If you have multiple instances of Visual Studio open, each instance can be bound to a different profile and region.
- The Amazon Explorer saves the last profile and region that were specified and the very last Visual Studio instance closed will have its values persisted.

Authentication and access

You don't need to authenticate with Amazon to start working with the Amazon Toolkit for Visual Studio with Amazon Q. However, most Amazon resources are managed through an Amazon account. To access all of the Amazon Toolkit for Visual Studio with Amazon Q services and features, you'll need at least 2 types of account authentication:

1. Either **Amazon Identity and Access Management (IAM)** or **Amazon IAM Identity Center** authentication for your Amazon accounts. Most Amazon services and resources are manged through IAM and IAM Identity Center.
2. An **Amazon Builder ID** is either optional for certain other Amazon services.

The following topics contain additional details and set up instructions for each credential type and authentication method.

Topics

- Amazon IAM Identity Center credentials in Amazon Toolkit for Visual Studio
- Amazon IAM credentials
- Amazon Builder ID
- Multi-factor authentication (MFA) in Toolkit for Visual Studio
- Setting up external credentials
- Updating firewalls and gateways to allow access

Amazon IAM Identity Center credentials in Amazon Toolkit for Visual Studio

Amazon IAM Identity Center is the recommended best practice for managing your Amazon account authentication.

For detailed instructions on how to set up IAM Identity Center for Software Development Kits (SDKs) and the Amazon Toolkit for Visual Studio, see the IAM Identity Center authentication section of the *Amazon SDKs and Tools Reference Guide*.

Authenticating with IAM Identity Center from the Amazon Toolkit for Visual Studio

To authenticate with IAM Identity Center from the Amazon Toolkit for Visual Studio by adding an IAM Identity Center profile to your `credentials` or `config` file, complete the following steps.

1. From your preferred text editor, open the Amazon credentials information stored in the `<hone-directory>\.aws\credentials` file.
2. From the `credentials file` under the section `[default]`, add a template for a named IAM Identity Center profile. The following is an example template:

> **Important**
>
> Do not use the word *profile* when creating an entry in the `credential` file because creates a conflict with the `credential` file naming conventions.
> Include the prefix word `profile_` only when configuring a named profile in the `config` file.

```
[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`**: The URL that points to your organization's IAM Identity Center user portal.
- **`sso_region`**: The Amazon Region that contains your IAM Identity Center portal host. This can be different from the Amazon Region specified later in the default `region` parameter.
- **`sso_account_id`**: The Amazon account ID that contains the IAM role with the permission that you want to grant to this IAM Identity Center user.
- **`sso_role_name`**: The name of the IAM role that defines the user's permissions when using this profile to get credentials through IAM Identity Center.
- **`region`**: The default Amazon Region that this IAM Identity Center user signs into.

Note

You can also add an IAM Identity Center enabled profile to your Amazon CLI by running the `aws configure sso` command. After running this command, you provide values for the IAM Identity Center start URL (`sso_start_url`) and the Amazon Region (`region`) that hosts the IAM Identity Center directory.
For more information, see Configuring the Amazon CLI to use Amazon Single Sign-On in the *Amazon Command Line Interface User Guide*.

Signing in with IAM Identity Center

When signing in with an IAM Identity Center profile, the default browser is launched to the `sso_start_url` specified in your `credential file`. You must verify your IAM Identity Center login before you can access your Amazon resources in Amazon Toolkit for Visual Studio. If your credentials expire, you'll have to repeat the connection process to obtain new temporary credentials.

Amazon IAM credentials

Amazon IAM credentials authenticate with your Amazon account through locally stored access keys.

The following sections describe how to set up IAM credentials to authenticate with your Amazon account from the Amazon Toolkit for Visual Studio.

Important

Before setting up IAM credentials to authenticate with your Amazon account, note that:

- If you've already set IAM credentials through another Amazon service (such as the Amazon CLI), then the Amazon Toolkit for Visual Studio automatically detects those credentials.
- Amazon recommends using Amazon IAM Identity Center authentication. For additional information about Amazon IAM best practices, see the Security best practice in IAM section of the *Amazon Identity and Access Management User Guide*.
- To avoid security risks, don't use IAM users for authentication when developing purpose-built software or working with real data. Instead, use federation with an identity provider

such as Amazon IAM Identity Center. For more information see the What is IAM Identity Center? in the *Amazon IAM Identity Center User Guide*.

Creating an IAM user

Before you can set up the Amazon Toolkit for Visual Studio to authenticate with your Amazon account, you need to complete **Step 1: Create your IAM user** and **Step 2: Get your access keys** in the Authenticate using long-term credentials topic in the *Amazon SDKs and Tools Reference Guide*.

Note

Step 3: Update the shared credentials is optional.
If you complete Step 3, the Amazon Toolkit for Visual Studio automatically detects your credentials from the `credentials file`.
If you haven't completed Step 3, the Amazon Toolkit for Visual Studio walks you through the process of creating a `credentials file` as described in the Creating a credentials file from the Amazon Toolkit for Visual Studio section, located below.

Creating a credentials file

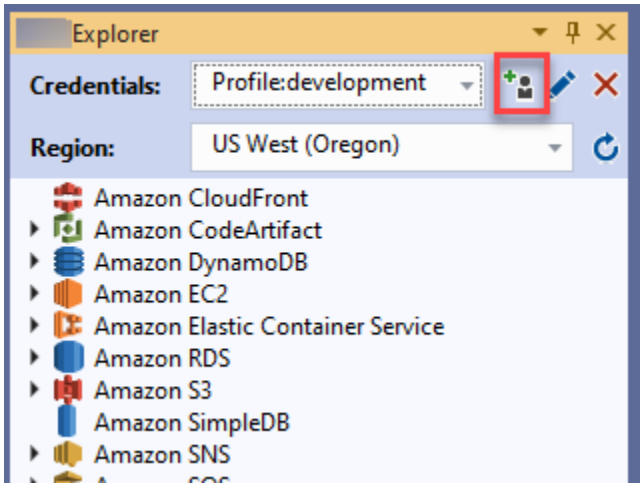
To add a user to or create a `credentials file` from the Amazon Toolkit for Visual Studio:

Note

When new user profile is added from the toolkit:

- If a `credentials file` already exists, the new user information is added to the existing file.
- If a `credentials file` doesn't exist a new file is created.

1. From the Amazon Explorer choose **New Account Profile** icon to open the **New Account Profile** dialog.



2. Complete the required fields in the **New Account Profile** dialog and choose the **OK** button to create the IAM user.

Editing IAM user credentials from the toolkit

To edit IAM user credentials from the toolkit, complete the following steps:

1. From the **Credentials** drop-down in the Amazon Explorer, choose the IAM user credential you want to edit.
2. Choose the **Edit Profile** icon to open the **Edit Profile** dialog.
3. From the **Edit Profile** dialog complete your updates and choose the **OK** button to save your changes.

To delete IAM user credentials from the toolkit, complete the following steps:

1. From the **Credentials** drop down in the Amazon Explorer, choose the IAM user credential you want to delete.
2. Choose the **Delete Profile** icon to open the **Delete Profile** prompt.
3. Confirm that you want to delete the profile to remove it from your `Credentials file`.

> **Important**
>
> Profiles that support advanced access features, such as IAM Identity Center or Multi-factor authentication (MFA) in the **Edit Profile** dialog, can't be edited from the Amazon Toolkit for

Visual Studio. To make changes to these types of profiles, you must edit the `credentials file`using a text editor.

Editing IAM user credentials from a text editor

In addition to managing IAM users with the Amazon Toolkit for Visual Studio, you can edit `credential files` from your preferred text editor. The default location of the `credential file` in Windows is `C:\Users\`*`USERNAME`*`\.aws\credentials`.

For more details on the location and structure of `credential files`, see the Shared config and credentials files section of the *Amazon SDKs and Tools Reference guide*.

Creating IAM users from the Amazon Command Line Interface (Amazon CLI)

The Amazon CLI is another tool you can use to create an IAM user in the `credentials file`, using the command `aws configure`.

For detailed information about creating IAM users from the Amazon CLI see the Configuring the Amazon CLI topics in the *Amazon CLI User Guide*.

The Toolkit for Visual Studio supports the following configuration properties:

```
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
```

Amazon Builder ID

Amazon Builder ID is an additional Amazon authentication method that may be required to use certain services or features, such as cloning a 3rd party repository with Amazon CodeCatalyst.

For detailed information about the Amazon Builder ID authentication method, see the Sign in with Amazon Builder ID topic in the *Amazon Sign-in* User Guide.

For additional information about cloning a repository for CodeCatalyst from Amazon Toolkit for Visual Studio, see the Working with Amazon CodeCatalyst topic in this User Guide.

Multi-factor authentication (MFA) in Toolkit for Visual Studio

Multi-factor authentication (MFA) is additional security for your Amazon accounts. MFA requires users to provide sign-in credentials and unique authentication from an Amazon supported MFA mechanism when accessing Amazon websites or services.

Amazon supports a range of both virtual and hardware devices for MFA authentication. The following is an example of a virtual MFA device enabled through a smartphone application. For more information on MFA device options, see Using multi-factor authentication (MFA) in Amazon in the *IAM User Guide*.

Step 1: Creating an IAM role to delegate access to IAM users

The following procedure describes how to set up role deligation for assigning permissions to an IAM user. For detailed information of role deligation, see the Creating a role to delegate permissions to an IAM user topic in the *Amazon Identity and Access Management User Guide*.

1. Go to the IAM console at https://console.aws.amazon.com/iam.
2. Choose **Roles** in the navigation bar, and then choose **Create Role**.
3. In the **Create role** page, choose **Another Amazon account**.
4. Enter your required **Account ID** and mark the **Require MFA** check box.

   > **Note**
   >
   > To find your 12-digit account number (ID), go to the navigation bar in the console, and then choose **Support**, **Support Center**.

5. Choose **Next: Permissions**.

6. Attach existing policies to your role or create a new policy for it. The policies that you choose on this page determine which Amazon services the IAM user can access with the Toolkit.
7. After attaching policies, choose **Next: Tags** for the option of adding IAM tags to your role. Then choose **Next: Review** to continue.
8. In the **Review** page, enter a required **Role name** (*toolkit-role*, for example). You can also add an optional **Role description**.
9. Choose **Create role**.
10. When the confirmation message displays ("The role **toolkit-role** has been created", for example), choose the name of the role in the message.
11. In the **Summary** page, choose the copy icon to copy the **Role ARN** and paste it into a file. (You need this ARN when configuring the IAM user to assume the role.).

Step 2: Creating an IAM user that assumes the role's permissions

This step creates an IAM user without permissions so that an in-line policy can be added.

1. Go to the IAM console at https://console.aws.amazon.com/iam.
2. Choose **Users** in the navigation bar and then choose **Add user**.
3. In the **Add user** page, enter a required **User name** (*toolkit-user*, for example) and mark the **Programmatic access** check box.
4. Choose **Next: Permissions**, **Next: Tags**, and **Next: Review** to move through the next pages. You're not adding permissions at this stage because the user is going to assume the role's permissions.
5. In the **Review** page, you're informed that **This user has no permissions**. Choose **Create user**.
6. In the **Success** page, choose **Download .csv** to download the file containing the access key ID and secret access key. (You need both when defining the user's profile in the credentials file.)
7. Choose **Close**.

Step 3: Adding a policy to allow the IAM user to assume the role

The following procedure creates an in-line policy that allows the user to assume the role (and that role's permissions).

1. In the **Users** page of the IAM console, choose the IAM user you've just created (*toolkit-user*, for example).

2. In the **Permissions** tab of the **Summary** page, choose **Add inline policy**.
3. In the **Create policy** page, choose **Choose a service**, enter **STS** in **Find a service**, and then choose **STS** from the results.
4. For **Actions**, start entering the term *AssumeRole*. Mark the **AssumeRole** check box when it appears.
5. In the **Resource section**, ensure **Specific** is selected, and click **Add ARN** to restrict access.
6. In the **Add ARN(s)** dialog box, for the **Specify ARN for role** add the ARN of the role you that you created in Step 1.

   After you add the role's ARN, the trusted account and role name associated with that role are displayed in **Account** and **Role name with path**.
7. Choose **Add**.
8. Back in the **Create policy** page, choose **Specify request conditions (optional)**, mark the **MFA required** check box, and then choose **close** to confirm..
9. Choose **Review policy**
10. In **Review policy** page, enter a **Name** for the policy, and then choose **Create policy**.

    The **Permissions** tab displays the new inline policy attached directly to IAM user.

Step 4: Managing a virtual MFA device for the IAM user

1. Download and install a virtual MFA application to your smartphone.

   For a list of supported applications, see the Multi-factor Authentication resource page.
2. In the IAM console, choose **Users** from the navigation bar and then choose the user that's assuming a role (*toolkit-user*, in this case).
3. In the **Summary** page, choose the **Security credentials** tab, and for **Assigned MFA device** choose **Manage**.
4. In the **Manage MFA device** pane, choose **Virtual MFA device**, and then choose **Continue**.
5. In the **Set up virtual MFA device** pane, choose **Show QR code** and then scan the code using the virtual MFA application that you installed on your smartphone.
6. After you scan the QR code, the virtual MFA application generates one-time MFA codes. Enter two consecutive MFA codes in **MFA code 1** and **MFA code 2**.
7. Choose **Assign MFA**.

8. Back in the **Security credentials** tab for the user, copy the ARN of the new **Assigned MFA device**.

   The ARN includes your 12-digit account ID and the format is similar to the following: `arn:aws:iam::123456789012:mfa/toolkit-user`. You need this ARN when defining the MFA profile in the next step.

Step 5: Creating profiles to allow MFA

The following procedure creates the profiles allowing MFA when accessing Amazon services from the Toolkit for Visual Studio.

The profiles that you create include three pieces of information that you've copied and stored during the previous steps:

- Access keys (access key ID and secret access key) for the IAM user
- ARN of the role that's delegating permissions to the IAM user
- ARN of the virtual MFA device that's assigned to the IAM user

In the Amazon shared credential file or SDK Store that contain your Amazon credentials, add the following entries:

```
[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
```

There are two profiles defined in the example provided:

- `[toolkit-user]` profile includes the access key and secret access key that were generated and saved when you created the IAM user in Step 2.
- `[mfa]` profile defines how multi-factor authentication is supported. There are three entries:

  ◦ `source_profile`: Specifies the profile whose credentials are used to assume the role specified by this `role_arn` setting in this profile. In this case, it's the `toolkit-user` profile.

- `role_arn`: Specifies the Amazon Resource Name (ARN) of the IAM role that you want to use to perform operations requested using this profile. In this case, it's the ARN for the role you created in Step 1.

- `mfa_serial`: Specifies the identification or serial number of the MFA device that the user must use when assuming a role. In this case, it's the ARN of the virtual device you set up in Step 3.

Setting up external credentials

If you have a method to generate or look up credentials that isn't directly supported by Amazon, you can add to the shared credentials file a profile that contains the `credential_process` setting. This setting specifies an external command that's run to generate or retrieve authentication credentials to use. For example, you might include an entry similar to the following in the `config` file:

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

For more information on using external credentials and the associated security risks, see Sourcing credentials with an external process in the *Amazon Command Line Interface User Guide*.

Updating firewalls and gateways to allow access

If you filter access to specific Amazon domains or URL endpoints by using a web-content filtering solution, the following endpoints must be allow listed in order to access all of the services and features available through the Amazon Toolkit for Visual Studio and Amazon Q. For detailed steps on how to troubleshoot firewall and proxy settings for the Amazon Toolkit with Amazon Q, see the Firewall and proxy settings section in the *Troubleshooting* topic in this User Guide.

Amazon Toolkit for Visual Studio Endpoints

The following are lists of Amazon Toolkit for Visual Studio specific endpoints and references that need to be allow listed.

Endpoints

```
https://idetoolkits-hostedfiles.amazonaws.com/*
https://idetoolkits.amazonwebservices.com/*
http://vstoolkit.amazonwebservices.com/*
https://aws-vs-toolkit.s3.amazonaws.com/*
https://raw.githubusercontent.com/aws/aws-toolkit-visual-studio/main/version.json
https://aws-toolkit-language-servers.amazonaws.com/*
```

Amazon Q plugin endpoints

The following is a list of Amazon Q plugin specific endpoints and references that need to be allow listed.

```
https://idetoolkits-hostedfiles.amazonaws.com/*    (Plugin for configs)
https://idetoolkits.amazonwebservices.com/*   (Plugin for endpoints)
https://aws-toolkit-language-servers.amazonaws.com/*  (Language Server Process)
https://client-telemetry.us-east-1.amazonaws.com/ (Telemetry)
https://cognito-identity.us-east-1.amazonaws.com    (Telemetry)
https://aws-language-servers.us-east-1.amazonaws.com (Language Server Process)
```

Amazon Q Developer endpoints

The following is a list of Amazon Q Developer specific endpoints and references that need to be allow listed.

```
https://codewhisperer.us-east-1.amazonaws.com (Inline,Chat, QSDA,...)
https://q.us-east-1.amazonaws.com (Inline,Chat, QSDA....)
https://desktop-release.codewhisperer.us-east-1.amazonaws.com/ (Download URL for CLI.)
https://specs.q.us-east-1.amazonaws.com (URL for auto-complete specs used by CLI)
* aws-language-servers.us-east-1.amazonaws.com (Local Workspace context)
```

Amazon Q Code Transform Endpoints

The following is a list of Amazon Q Code Transform specific endpoints and references that need to be allow listed.

```
https://docs.aws.amazon.com/amazonq/latest/qdeveloper-ug/security_iam_manage-access-
with-policies.html
```

Authentication endpoints

The following is a list of authentication endpoints and references that need to be allow listed.

```
[Directory ID or alias].awsapps.com
* oidc.[Region].amazonaws.com
*.sso.[Region].amazonaws.com
*.sso-portal.[Region].amazonaws.com
*.aws.dev
*.awsstatic.com
*.console.aws.a2z.com
*.sso.amazonaws.com
```

Identity Endpoints

The following lists contain endpoints that are specific to identity, such as Amazon IAM Identity Center and Amazon Builder ID.

Amazon IAM Identity Center

For details on required endpoints for IAM Identity Center, see the Enable IAM Identity Center topic in the *Amazon IAM Identity Center* User Guide.

Enterprise IAM Identity Center

```
https://[Center director id].awsapps.com/start (should be permitted to initiate auth)
https://us-east-1.signin.aws (for facilitating authentication, assuming IAM Identity
 Center is in IAD)
```

```
https://oidc.(us-east-1).amazonaws.com
https://log.sso-portal.eu-west-1.amazonaws.com
https://portal.sso.eu-west-1.amazonaws.com
```

Amazon Builder ID

```
https://view.awsapps.com/start (must be blocked to disable individual tier)
https://codewhisperer.us-east-1.amazonaws.com and q.us-east-1.amazonaws.com (should be
 permitted)
```

Telemetry

The following is a Telemetry specific endpoint that needs to be allow listed.

```
https://telemetry.aws-language-servers.us-east-1.amazonaws.com/
https://client-telemetry.us-east-1.amazonaws.com
```

References

The following is a list of endpoint references.

```
idetoolkits-hostedfiles.amazonaws.com
cognito-identity.us-east-1.amazonaws.com
amazonwebservices.gallery.vsassets.io
eu-west-1.prod.pr.analytics.console.aws.a2z.com
prod.pa.cdn.uis.awsstatic.com
portal.sso.eu-west-1.amazonaws.com
log.sso-portal.eu-west-1.amazonaws.com
prod.assets.shortbread.aws.dev
prod.tools.shortbread.aws.dev
prod.log.shortbread.aws.dev
a.b.cdn.console.awsstatic.com
assets.sso-portal.eu-west-1.amazonaws.com
oidc.eu-west-1.amazonaws.com
aws-toolkit-language-servers.amazonaws.com
```

```
aws-language-servers.us-east-1.amazonaws.com
idetoolkits.amazonwebservices.com
```

Working with Amazon Services

The following topics describes how to get started working with Amazon services from the Amazon Toolkit for Visual Studio with Amazon Q.

Topics

- Amazon CodeCatalyst for the Amazon Toolkit for Visual Studio with Amazon Q
- Amazon CloudWatch Logs integration for Visual Studio
- Managing Amazon EC2 Instances
- Managing Amazon ECS Instances
- Managing Security Groups from Amazon Explorer
- Creating an AMI from an Amazon EC2 Instance
- Setting Launch Permissions on an Amazon Machine Image
- Amazon Virtual Private Cloud (VPC)
- Using the Amazon CloudFormation Template Editor for Visual Studio
- Using Amazon S3 from Amazon Explorer
- Using DynamoDB from Amazon Explorer
- Using Amazon CodeCommit with Visual Studio Team Explorer
- Using CodeArtifact in Visual Studio
- Amazon RDS from Amazon Explorer
- Using Amazon SimpleDB from Amazon Explorer
- Using Amazon SQS from Amazon Explorer
- Identity and Access Management
- Amazon Lambda

Amazon CodeCatalyst for the Amazon Toolkit for Visual Studio with Amazon Q

What is Amazon CodeCatalyst?

Amazon CodeCatalyst is a cloud-based collaboration space for software development teams. Using the Amazon Toolkit for Visual Studio with Amazon Q, you can view and manage CodeCatalyst

resources directly from Amazon Toolkit for Visual Studio with Amazon Q. For more information about CodeCatalyst, see the Amazon CodeCatalyst User Guide.

The following topics describe how to connect the Amazon Toolkit for Visual Studio with Amazon Q with CodeCatalyst and how to work with CodeCatalyst through the Amazon Toolkit for Visual Studio with Amazon Q.

Topics

- Getting Started with Amazon CodeCatalyst and the Amazon Toolkit for Visual Studio with Amazon Q
- Working with Amazon CodeCatalyst resources from the Amazon Toolkit for Visual Studio with Amazon Q
- Troubleshooting

Getting Started with Amazon CodeCatalyst and the Amazon Toolkit for Visual Studio with Amazon Q

To get started working with Amazon CodeCatalyst from the Amazon Toolkit for Visual Studio with Amazon Q, complete the following.

Topics

- Installing the Amazon Toolkit for Visual Studio with Amazon Q
- Creating a CodeCatalyst account and Amazon Builder ID
- Connecting Amazon Toolkit for Visual Studio with Amazon Q with CodeCatalyst

Installing the Amazon Toolkit for Visual Studio with Amazon Q

Before you integrate the Amazon Toolkit for Visual Studio with Amazon Q with your CodeCatalyst accounts, make sure that you're using a current version of Amazon Toolkit for Visual Studio with Amazon Q. For details on how to install and set up the latest version of Amazon Toolkit for Visual Studio with Amazon Q, see the Setting up the Amazon Toolkit for Visual Studio with Amazon Q section of this User Guide.

Creating a CodeCatalyst account and Amazon Builder ID

In addition to installing the latest version of the Amazon Toolkit for Visual Studio with Amazon Q, you must have an active Amazon Builder ID and CodeCatalyst account to connect with Amazon

Toolkit for Visual Studio with Amazon Q. If you don't have an active Amazon Builder ID or CodeCatalyst account, see the Setting up with CodeCatalyst section in the *CodeCatalyst* User Guide.

> **Note**
>
> An Amazon Builder ID is different from your Amazon Credentials. For instructions on how to sign up and authenticate with an Amazon Builder ID, see the Authentication and access: Amazon Builder ID topic in this User Guide.
> For detailed information about Amazon Builder IDs, see the Amazon Builder ID topic in the *Amazon General Reference* User Guide.

Connecting Amazon Toolkit for Visual Studio with Amazon Q with CodeCatalyst

To connect Amazon Toolkit for Visual Studio with Amazon Q with your CodeCatalyst account, complete the following steps.

1. From the **Git** menu item in Visual Studio, choose **Clone Repository...**.
2. From the **Browse a Repository** section, select **Amazon CodeCatalyst** as the provider.
3. From the **Connection** section, choose **Connect with Amazon Builder ID** to open the CodeCatalyst console in your preferred web browser.
4. From your browser, enter your Amazon Builder ID into the provided field and follow the instructions to continue.
5. When prompted, choose **Allow** to confirm the connection between Amazon Toolkit for Visual Studio with Amazon Q and your CodeCatalyst account. When the connection process is complete, CodeCatalyst displays a confirmation indicating that it's safe to close your browser.

Working with Amazon CodeCatalyst resources from the Amazon Toolkit for Visual Studio with Amazon Q

The following sections provide an overview of the Amazon Amazon CodeCatalyst resource management features that are available for the Amazon Toolkit for Visual Studio with Amazon Q.

Topics

- Clone a repository

Clone a repository

CodeCatalyst is a cloud-based service that requires you to be connected to the cloud to work on CodeCatalyst projects. To work on a project locally, you can clone CodeCatalyst repositories to your local machine and sync with your CodeCatalyst project the next time that you connect to the cloud.

To clone a repository to your local machine, complete the following steps.

1. From the **Git** menu item in Visual Studio, choose **Clone Repository...**.
2. From the **Browse a Repository** section, select **Amazon CodeCatalyst** as the provider.

> **Note**
>
> If the **Connection** section displays a `Not Connected` message, complete the steps in the Authentication and access: Amazon Builder ID section of this User Guide before proceeding.

3. Choose the **Space** and **Project** that you want to clone a repository from.
4. From the **Repositories** section, choose the repository that you want to clone.
5. From the **Path** section, choose the folder you want to clone your repository to.

> **Note**
>
> This folder must initially be empty to clone successfully.

6. Select **Clone** to begin cloning the repository.
7. After the repository has been cloned, Visual Studio will load your cloned solution

> **Note**
>
> If Visual Studio does not open the solution in the cloned repository, your Visual Studio options can be adjusted from the **Automatically load the solution when opening a Git repository** setting, located in the **Git Global Settings**, of the **Source Control** menu.

Troubleshooting

The following are troubleshooting topics for addressing known issues when working with Amazon CodeCatalyst from the Amazon Toolkit for Visual Studio with Amazon Q.

Topics

- Credentials

Credentials

If you encounter a dialog asking for credentials when attempting to clone a git-based repository from CodeCatalyst, your **Amazon CodeCommit Credential helper** may be configured globally, causing interference with CodeCatalyst. For additional information about the Amazon CodeCommit credential helper, see the Set up steps for HTTPS connections to Amazon CodeCommit repositories on Windows with the Amazon CLI credential helper section of the *Amazon CodeCommit* User Guide.

To limit the **Amazon CodeCommit Credential helper** to handling only CodeCommit URLs, complete the following steps.

1. open the global git config file in: `%userprofile%\.gitconfig`
2. Locate the following section in your file:

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

3. Change that section to the following:

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

4. Save your changes, then complete the steps to clone your repository.

Amazon CloudWatch Logs integration for Visual Studio

The Amazon CloudWatch Logs integration from the Amazon Toolkit for Visual Studio with Amazon Q gives you the ability to monitor, store, and access CloudWatch Logs resources, without having to leave your IDE. To learn more about setting up the CloudWatch service and how to work with CloudWatch Logs features, choose from the following topics.

Topics

- Setting up CloudWatch Logs integration for Visual Studio
- Working with CloudWatch Logs in Visual Studio

Setting up CloudWatch Logs integration for Visual Studio

Before you can use the Amazon CloudWatch Logs integration with the Amazon Toolkit with Amazon Q, you need an Amazon account. You can create a new Amazon account from the Amazon sign in site. Most of the CloudWatch Logs features that are available from the Amazon Toolkit with Amazon Q are accessible with active Amazon credentials. If a particular feature requires additional configuration, the requirements are included in the relevant sections of the Working with CloudWatch Logs guide.

For additional information and options on setting up CloudWatch Logs, see the Getting set up section of the Amazon CloudWatch Logs guide.

Working with CloudWatch Logs in Visual Studio

Amazon CloudWatch Logs integration allows you to monitor, store, and access CloudWatch Logs from the Amazon Toolkit for Visual Studio with Amazon Q. Having access to CloudWatch Logs features—without the need to leave your IDE—improves efficiency by simplifying the CloudWatch Logs development process and reducing disruptions to your work flow. The following topics describe how to work with the basic features and functions of the CloudWatch Logs integration.

Topics

- CloudWatch Log Groups
- CloudWatch Log Streams
- CloudWatch Log Events
- Additional access to CloudWatch Logs

CloudWatch Log Groups

A `log group` is a group of `log streams` that share the same retention, monitoring, and access control settings. There is no limit on the number of log streams that can belong to one log group.

Viewing Log Groups

The `View Log Groups` feature displays a list of log groups in the CloudWatch Log Groups Explorer.

To access the View Log Groups feature and open the CloudWatch Log Groups Explorer, complete the following steps.

1. From the Amazon Explorer, expand **Amazon CloudWatch**.
2. Double-click **Log Groups** or open the context menu (right-click) and select **View**, to open the **CloudWatch Log Groups Explorer**.

> **Note**
>
> The CloudWatch Log Groups Explorer will open in the same window location as the Solutions Explorer.

Filtering Log Groups

Your individual account is able to contain thousands of different log groups. To simplify your search for specific groups, use the `filtering` feature described below.

1. From the **CloudWatch Log Groups Explorer**, set your cursor in the search bar located at the top of the window.
2. Start typing a prefix related to the log groups that you're looking for.
3. **CloudWatch Log Groups Explorer** is automatically updated to show results matching the search terms you specified in the previous step.

Delete Log Groups

To delete a specific log group, refer to the following procedure.

1. From the **CloudWatch Log Groups Explorer**, right-click the Log Group that you want to delete.

2. When prompted, confirm that you want to delete the currently selected Log Group.
3. Choosing the **Yes** button deletes the selected log group, then refreshes the **CloudWatch Log Groups Explorer**.

Refresh Log Groups

To refresh the current list of log groups displayed in the **CloudWatch Log Groups Explorer**, choose the **Refresh icon** button located in the **toolbar**.

Copy Log Group ARN

To copy the ARN of a specific log group, complete the steps described below.

1. From the **CloudWatch Log Groups Explorer**, right-click the Log Group you want to copy an ARN from.
2. Choose the **Copy ARN** option from the menu.
3. The ARN is now copied to your local clipboard and ready to paste.

CloudWatch Log Streams

A log stream is a sequence of log events that share the same source.

> **Note**
>
> When viewing log streams, be aware of the following properties:
>
> - By default the log streams are sorted by the most-recent event time stamp.
> - Columns associated with a log stream can be sorted in either ascending or descending order, by toggling the **caret** located in the column headers.
> - Filtered entries can only be sorted by **Log Stream Name**.

Viewing Log Streams

1. From the **CloudWatch Log Groups Explorer** double-click a Log Group, or right-click a log group and select **View Log Stream** from the context menu.
2. A new tab will open in the **document** window, which contains a list of log streams associated with your log group.

Filtering Log Streams

1. From the **Log Streams** tab, in the **document** window, set your cursor in the search bar.
2. Start typing a prefix related to the log stream that you're looking for.
3. As you type, the current display automatically updates to filter your Log Streams by your input.

Refresh Log Streams

To refresh the current list of log streams displayed in the **document** window, choose the **Refresh icon** button, located in the **toolbar**, next to the **search bar**.

Copy Log Streams ARN

To copy the ARN of a specific log stream, complete the steps described below.

1. From the **Log Streams** tab, in the **document** window, right-click the log stream you want to copy an ARN from.
2. Choose the **Copy ARN** option from the menu.
3. The ARN is now copied to your local clipboard and ready to paste.

Download Log Streams

The **Export Log Stream** feature downloads and stores the selected log stream locally, where it can be accessed by custom tools and software for additional processing.

1. From the **Log Streams** tab, in the **document** window, right-click the log stream you want to download.
2. Choose **Export Log Stream** to open the **Export to a text file** dialog.
3. Choose the location where you want to store the file locally and specify a name in the provided text field.
4. Confirm the download by selecting **OK**. The status of the download is displayed in the **Visual Studio Task Status Center**

CloudWatch Log Events

Log events are records of activity recorded by the application or resource being monitored by CloudWatch.

Log Event actions

Log events are displayed as a table. By default, the events are sorted from the oldest event to the most recent.

The following actions are associated with log events in Visual Studio:

- Wrapped-text mode: You can toggle wrapped-text by clicking an event.
- Text-wrap button: located in the `document window` **`toolbar`**, this button toggles text-wrap on and off, for all entries.
- Copy messages to your clipboard: select the messages you want to copy, then right-click the selection and choose **Copy** (keyboard shortcut `Ctrl + C`).

Viewing Log Events

1. From the **document** window, choose a tab that contains a list of log streams.
2. Double-click a log stream, or right-click a log stream and select **View Log Stream** from the menu.
3. A new **log event** tab will open in the **document** window, which contains a table of log events associated with your chosen log stream.

Filtering Log Events

There are three ways for you to filter log events: by content, time range, or both. To filter your log events by both content and time range, start by filtering your messages by either content or time range, then filter those results by the other method.

To filter your log events by content:

1. From the **log event** tab, in the **document** window, set your cursor in the search bar, located at the top of the window.
2. Start typing a term or phrase related to the log events that you're searching for.
3. As you type, the current display automatically begins to filter your log events.

> **Note**
>
> Filter patterns are case sensitive. You can improve search results by enclosing exact terms, and phrases, with non-alphanumeric characters in double quotation marks (*""*). For more detailed information about filter patterns, see the Filter and Pattern Syntax topic in the Amazon CloudWatch guide.

To view log events generated during a specific time range:

1. From the **log event** tab, in the **document** window, choose the **Calendar icon** button, located in the **toolbar**.
2. Using the provided fields, specify the time range that you want to search.
3. The filtered results update automatically as you specify the date and time constraints.

> **Note**
>
> The **Clear Filter** option clears all of your current date-and-time filter selections.

Refresh Log Events

To refresh the current list of log events displayed in the **log event** tab, choose the **Refresh icon** button, located in the **toolbar**.

Additional access to CloudWatch Logs

You can access CloudWatch Logs associated with other Amazon services and resources directly from the Amazon Toolkit in Visual Studio.

Lambda

To view log streams that are associated with a Lambda function:

> **Note**
>
> Your Lambda execution role must have appropriate permissions to send logs to CloudWatch Logs. For more information about the Lambda permissions required for

CloudWatch Logs, see the https://docs.amazonaws.cn/lambda/latest/dg/monitoring-cloudwatchlogs.html#monitoring-cloudwatchlogs-prereqs

1. From the Amazon Toolkit Explorer, expand **Lambda**.
2. right-click the function you want to view, then choose **View Logs** to open the associated log streams in the **document** window.

To view log streams using the Lambda integration `function view`:

1. From the Amazon Toolkit Explorer, expand **Lambda**.
2. right-click the function you want to view, then choose **View Function** to open the function view in the **document** window.
3. From the `function view`, switch to the **Logs** tab, the log streams associated with the chosen Lambda function are displayed.

ECS

To view log resources that are associated with an ECS Task Container, complete the following procedure.

Note

In order for the Amazon ECS service to send logs to CloudWatch, each container for a given Amazon ECS Task must meet the required configuration. For additional information about the required set up and configurations, please see the guide Using the Amazon Logs Log Driver.

1. From the Amazon Toolkit Explorer, expand **Amazon ECS**.
2. Choose the Amazon ECS Cluster that you want to view to open a new **ECS Cluster** tab, in the **document** window.
3. From the navigation menu, located on the left side of the **ECS Cluster** tab, choose **Tasks** to list all tasks associated with the cluster.
4. From the **Tasks** display, select a task and choose the **View Logs** link, located in the bottom-left corner.

> **Note**
>
> This display lists all tasks contained in the cluster, the `View Logs` link is only visible for each task that meets the required logs configuration.
>
> - If a Task is only associated with a single container, the **View Logs** link opens that container's log stream.
> - If a Task is associated with multiple containers, the **View Logs** link opens the **View CloudWatch Logs for ECS Task** dialog, use the **Container:** drop-down menu to choose the container you want to view Logs for, then choose **OK**.

5. A new tab opens in the **document** window displaying the log streams associated with your container selection.

Managing Amazon EC2 Instances

Amazon Explorer provides detailed views of Amazon Machine Images (AMI) and Amazon Elastic Compute Cloud (Amazon EC2) instances. From these views, you can launch an Amazon EC2 instance from an AMI, connect to that instance, and either stop or terminate the instance, all from inside the Visual Studio development environment. You can use the instances view to create AMIs from your instances. For more information, see Create an AMI from an Amazon EC2 Instance.

The Amazon Machine Images and Amazon EC2 Instances Views

From Amazon Explorer, you can display views of Amazon Machine Images (AMIs) and Amazon EC2 instances. In Amazon Explorer, expand the **Amazon EC2** node.

To display the AMIs view, on the first subnode, **AMIs**, open the context (right-click) menu and then choose **View**.

To display the Amazon EC2 instances view, on the **Instances** node, open the context (right-click) menu and then choose **View**.

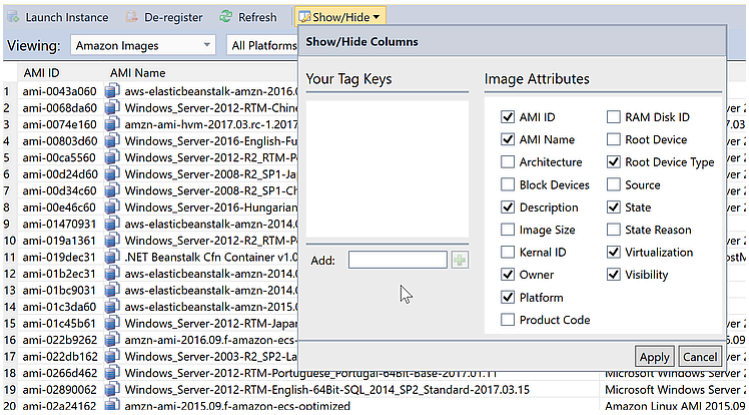
You can also display either view by double-clicking the appropriate node.

- The views are scoped to the region specified in Amazon Explorer (for example, the US West (N. California) region).

- You can rearrange columns by clicking and dragging. To sort the values in a column, click the column heading.
- You can use the drop-down lists and filter box in **Viewing** to configure views. The initial view displays AMIs of any platform type (Windows or Linux) that are owned by the account specified in Amazon Explorer.

Show/Hide Columns

You can also choose the **Show/Hide** drop-down at the top of the view to configure which columns are displayed. Your choice of columns will persist if you close the view and reopen it.



Show/Hide Columns UI for AMI and Instances views

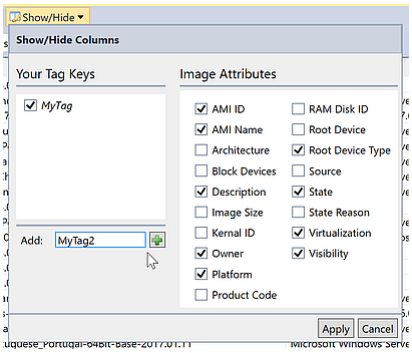
Tagging AMIs, Instances, and Volumes

You can also use the **Show/Hide** drop-down list to add tags for AMIs, Amazon EC2 instances, or volumes you own. Tags are name-value pairs that enable you to attach metadata to your AMIs, instances, and volumes. Tag names are scoped both to your account and also separately to your AMIs and instances. For example, there would be no conflict if you used the same tag name for your AMIs and your instances. Tag names are not case-sensitive.

For more information about tags, go to Using Tags in the *Amazon EC2 User Guide for Linux Instances*.

To add a tag

1. In the **Add** box, type a name for the tag. Choose the green button with the plus sign (+), and then choose **Apply**.



Add a tag to an AMI or Amazon EC2 instance

The new tag is displayed in italic, which indicates no values have yet been associated with that tag.

In the list view, the tag name appears as a new column. When at least one value has been associated with the tag, the tag will be visible in the Amazon Web Services Management Console.

2. To add a value for the tag, double-click a cell in the column for that tag, and type a value. To delete the tag value, double-click the cell and delete the text.

   If you clear the tag in the **Show/Hide** drop-down list, the corresponding column disappears from the view. The tag is preserved, along with any tag values associated with AMIs, instances, or volumes.

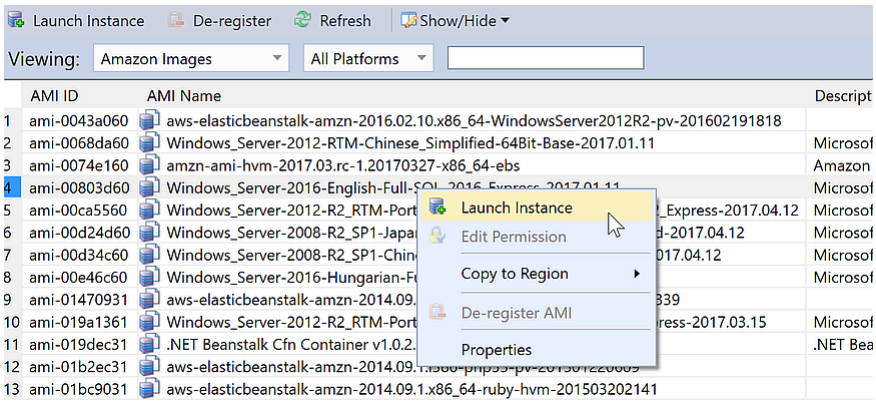
> **Note**
>
> If you clear a tag in the **Show/Hide** drop-down list that has no associated values, the Amazon Toolkit will delete the tag entirely. It will no longer appear in the list view or in the **Show/Hide** drop-down list. To use that tag again, use the **Show/Hide** dialog box to re-create it.

Launching an Amazon EC2 Instance

Amazon Explorer provides all of the functionality required to launch an Amazon EC2 instance. In this section, we'll select an Amazon Machine Image (AMI), configure it, and then start it as an Amazon EC2 instance.

To launch a Windows Server Amazon EC2 instance

1. At the top of the AMIs view, in the drop-down list on the left, choose **Amazon Images**. In the drop-down list on the right, choose **Windows**. In the filter box, type `ebs` for Elastic Block Storage. It may take a few moments for the view to be refreshed.
2. Choose an AMI in the list, open the context (right-click) menu, and then choose **Launch Instance**. .



AMI list

3. In the **Launch New Amazon EC2 Instance** dialog box, configure the AMI for your application.

Instance Type

Choose the type of the EC2 instance to launch. You can find a list of instance types and pricing information on the EC2 Pricing page.

Name

Type a name for your instance. This name cannot be more than 256 characters.

Key Pair

A key pair is used to obtain the Windows password that you use to log in to the EC2 instance using Remote Desktop Protocol (RDP). Choose a key pair for which you have access to the private key, or choose the option to create a key pair. If you create the key pair in the Toolkit, the Toolkit can store the private key for you.

Key pairs stored in the Toolkit are encrypted. you can find them at `%LOCALAPPDATA%\AWSToolkit\keypairs` (typically: `C:\Users\<user>\AppData\Local\AWSToolkit\keypairs`). You can export the encrypted key pair into a `.pem` file.

a. In Visual Studio, select **View** and click **Amazon Explorer**.

b. Click on **Amazon EC2** and select **Key Pairs**.

c. The key pairs will be listed, and those created/managed by the Toolkit marked as **Stored in AWSToolkit**.

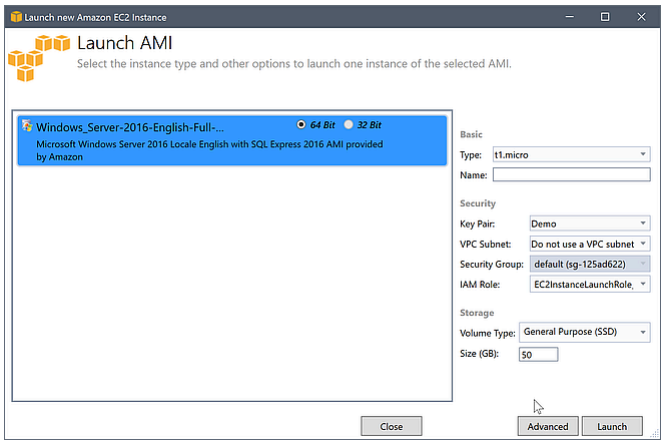
d. Right click on the key pair you created and select **Export Private Key**. The private key will be unencrypted and stored in the location you specify.

Security Group

The security group controls the type of network traffic the EC2 instance will accept. Choose a security group that will allow incoming traffic on port 3389, the port used by RDP, so that you can connect to the EC2 instance. For information about how to use the Toolkit to create security groups, see Managing Security Groups from Amazon Explorer.

Instance Profile

The instance profile is a logical container for an IAM role. When you choose an instance profile, you associate the corresponding IAM role with the EC2 instance. IAM roles are configured with policies that specify access to Amazon Web Services and account resources. When an EC2 instance is associated with an IAM role, application software that runs on the instance runs with the permissions specified by the IAM role. This enables the application software to run without having to specify any Amazon credentials of its own, which makes the software more secure. For more information about IAM roles, go to the IAM User Guide.

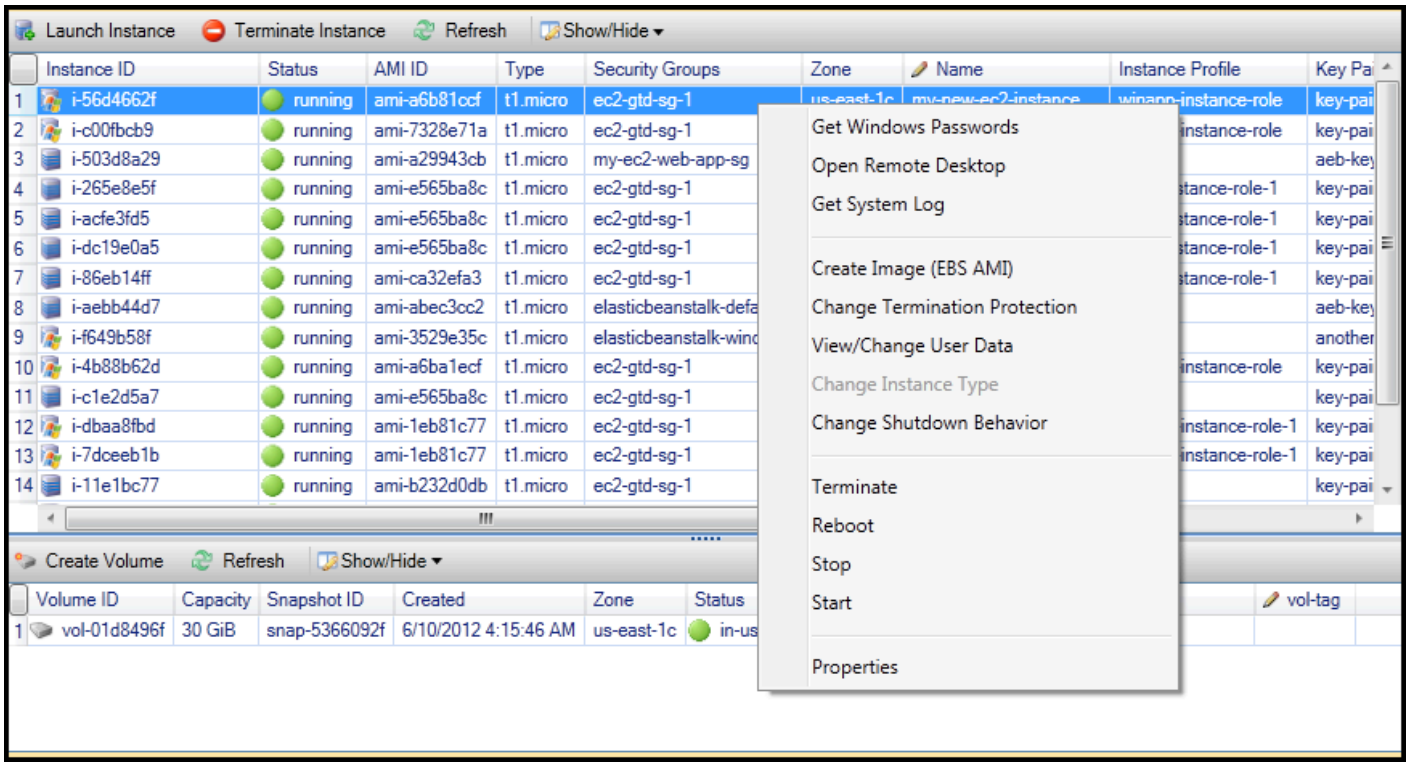


EC2 **Launch AMI** dialog box

4. Choose **Launch**.

In Amazon Explorer, on the **Instances** subnode of **Amazon EC2**, open the context (right-click) menu and then choose **View**. The Amazon Toolkit displays the list of Amazon EC2 instances associated with the active account. You may need to choose **Refresh** to see your new instance.

When the instance first appears, it may be in a pending state, but after a few moments, it transitions to a running state.



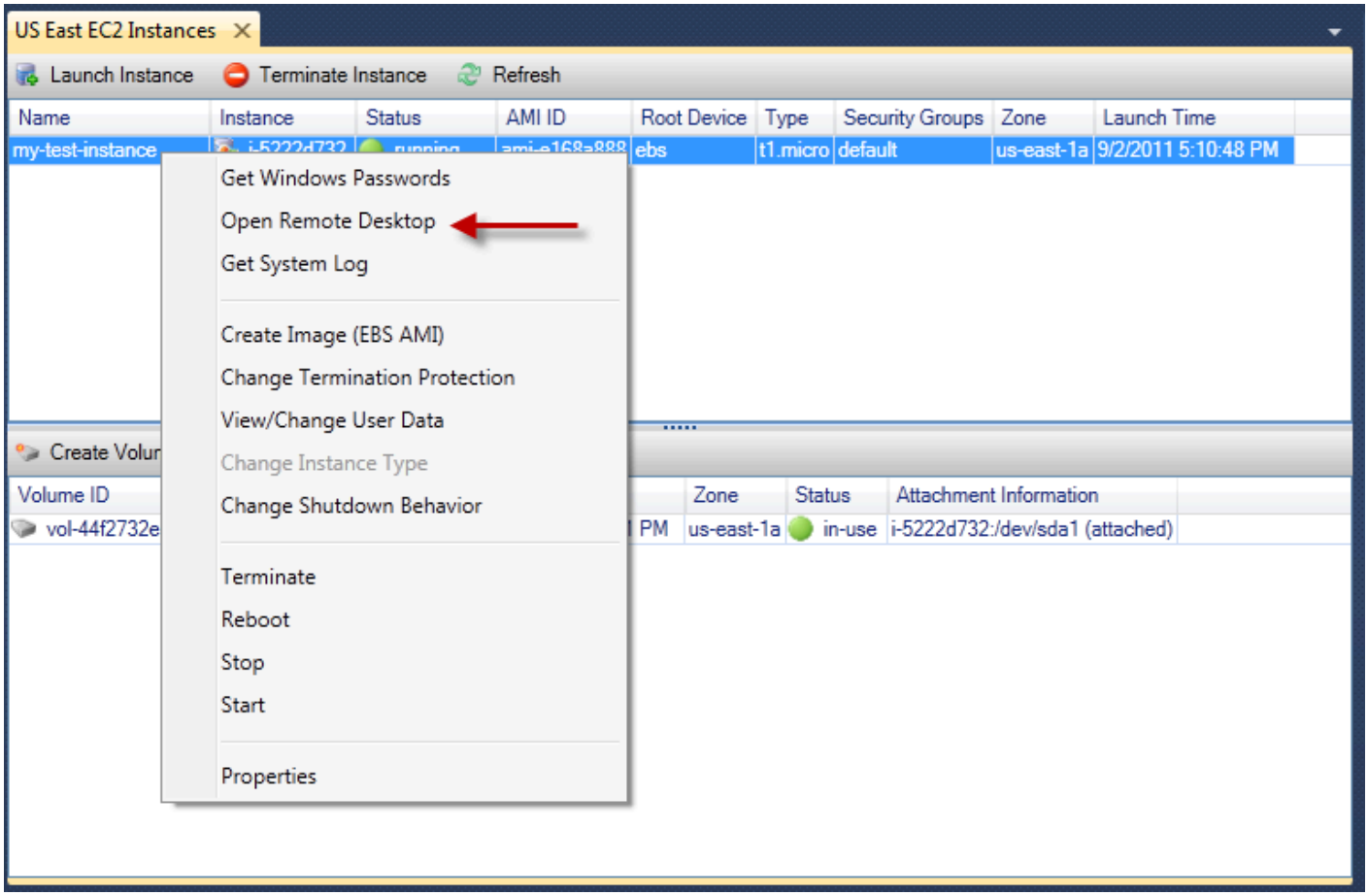
Connecting to an Amazon EC2 Instance

You can use Windows Remote Desktop to connect to a Windows Server instance. For authentication, the Amazon Toolkit enables you to retrieve the administrator password for the instance, or you can simply use the stored key pair associated with the instance. In the following procedure, we'll use the stored key pair.

To connect to a Windows Server instance using Windows Remote Desktop

1. In the EC2 instance list, right-click the Windows Server instance to which you want to connect. From the context menu, choose **Open Remote Desktop**.

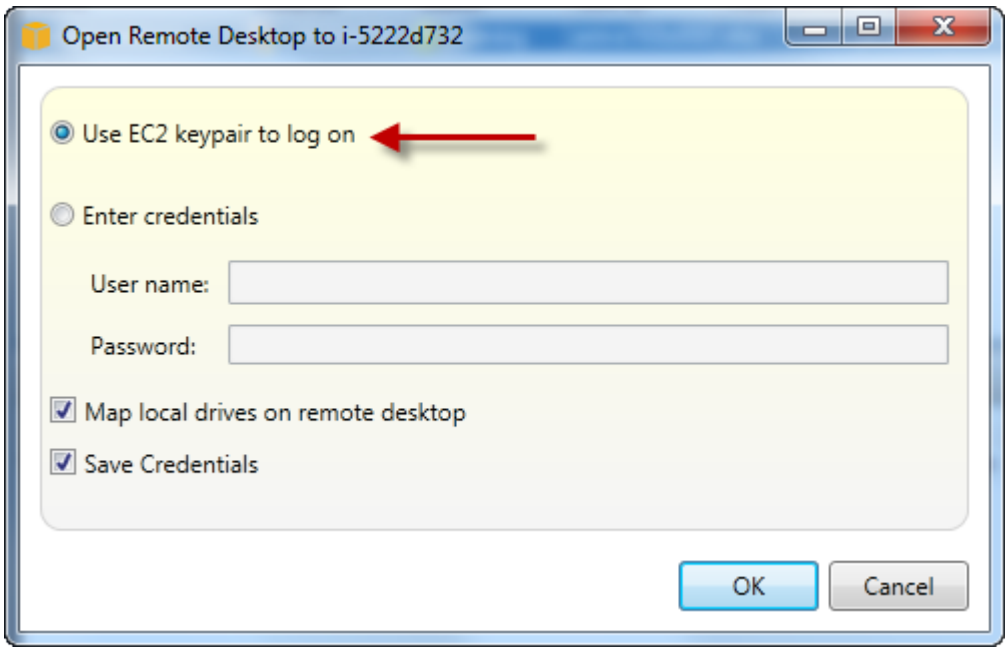
   If you want to authenticate using the administrator password, you would choose **Get Windows Passwords**.



EC2 Instance context menu

2. In the **Open Remote Desktop** dialog box, choose **Use EC2 keypair to log on**, and then choose **OK**.

   If you did not store a key pair with the Amazon Toolkit, specify the PEM file that contains the private key.

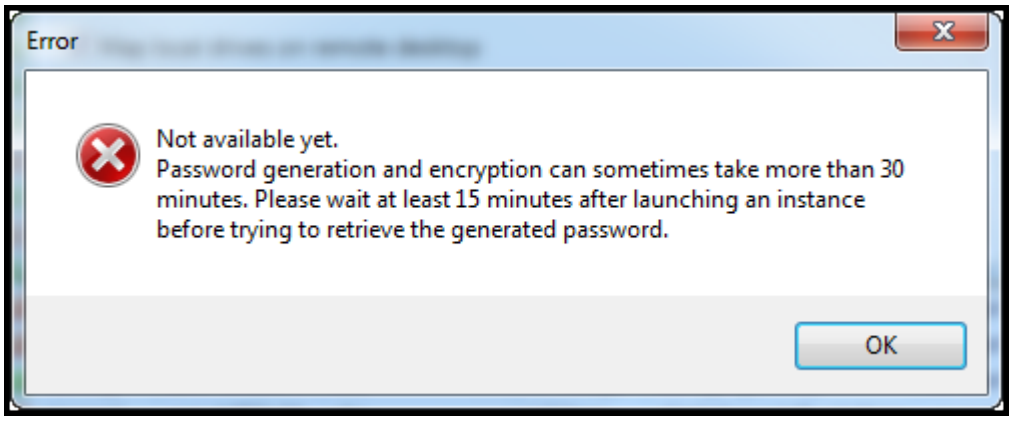


Open Remote Desktop dialog box

3. The **Remote Desktop** window will open. You do not need to sign in because authentication occurred with the key pair. You will be running as the administrator on the Amazon EC2 instance.

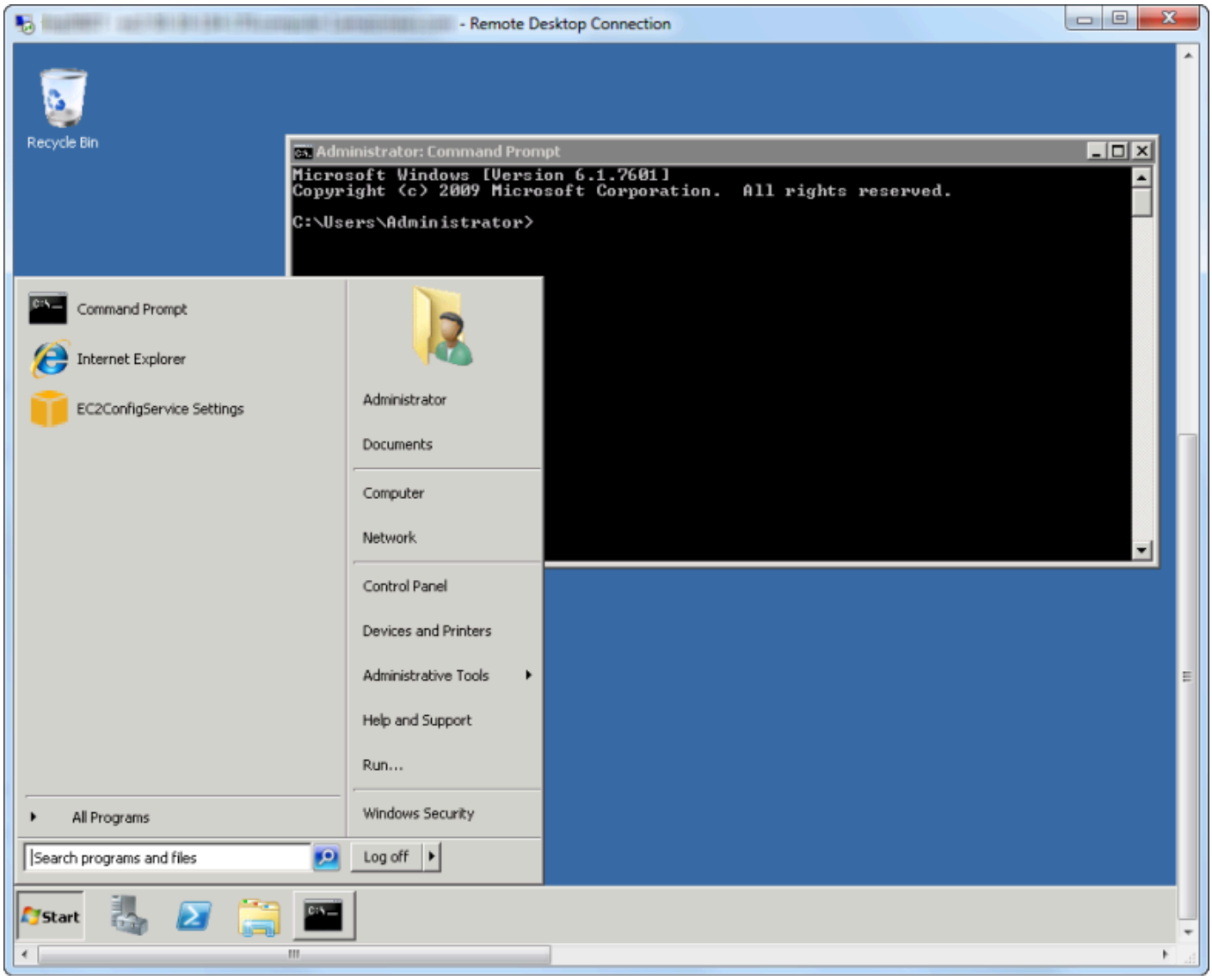
   If the EC2 instance has only recently started, you may not be able to connect for two possible reasons:

   - The Remote Desktop service might not yet be up and running. Wait a few minutes and try again.
   - Password information might not yet have been transferred to the instance. In this case, you will see a message box similar to the following.



Password not yet available

The following screenshot shows a user connected as administrator through Remote Desktop.



Remote Desktop

Ending an Amazon EC2 Instance

Using the Amazon Toolkit, you can stop or terminate a running Amazon EC2 instance from Visual Studio. To stop the instance, the EC2 instance must be using an Amazon EBS volume. If the EC2 instance is not using an Amazon EBS volume, then your only option is to terminate the instance.

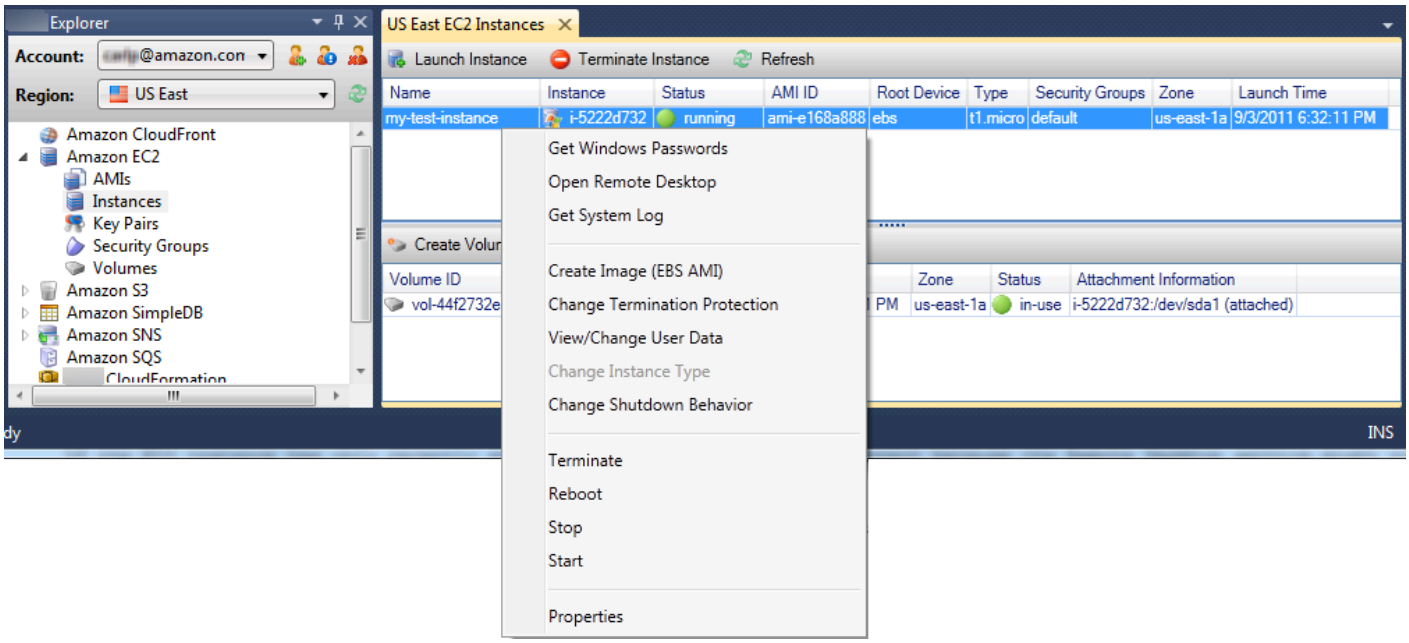
If you stop the instance, data stored on the EBS volume is retained. If you terminate the instance, all data stored on the local storage device of the instance will be lost. In either case, stop or

terminate, you will not continue to be charged for the EC2 instance. However, if you stop an instance, you will continue to be charged for the EBS storage that persists after the instance is stopped.

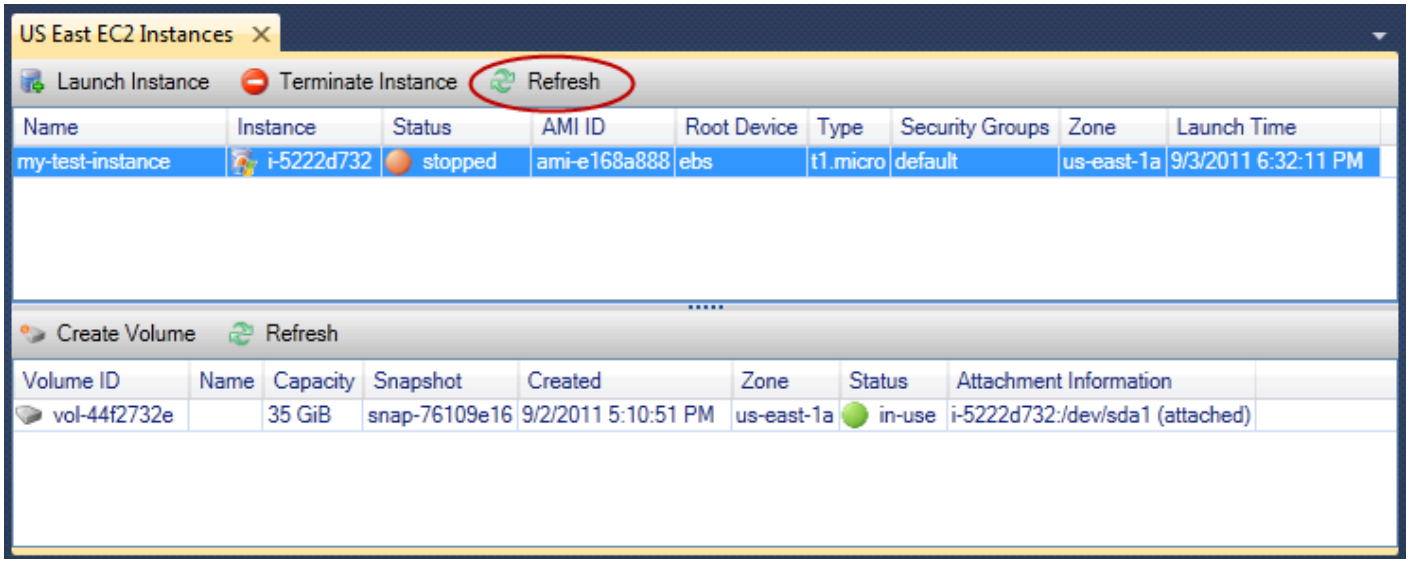
Another possible way to end an instance is to use Remote Desktop to connect to the instance, and then from the Windows **Start** menu, use **Shutdown**. You can configure the instance to either stop or terminate in this scenario.

To stop an Amazon EC2 instance

1. In Amazon Explorer, expand the **Amazon EC2** node, open the context (right-click) menu for **Instances**, and then choose **View**. In the **Instances** list, right-click the instance you want to stop and choose **Stop** from the context menu. Choose **Yes** to confirm you want to stop the instance.

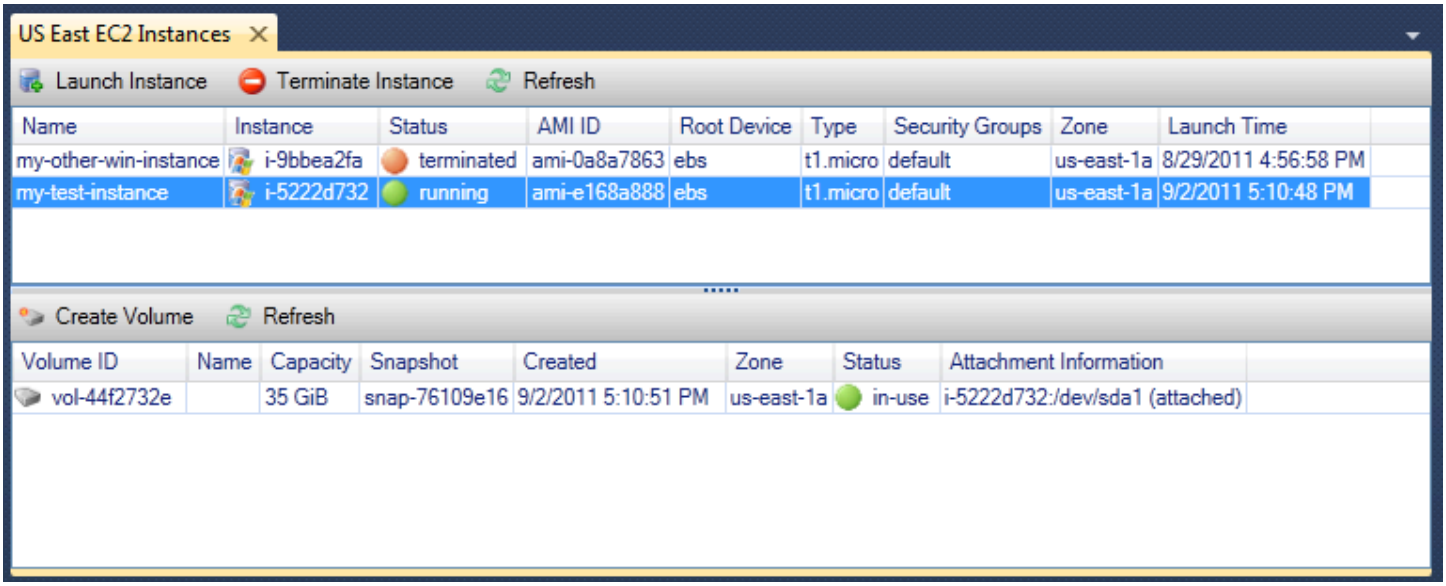


2. At the top of the **Instances** list, choose **Refresh** to see the change in the status of the Amazon EC2 instance. Because we stopped rather than terminated the instance, the EBS volume associated with the instance is still active.



Terminated Instances Remain Visible

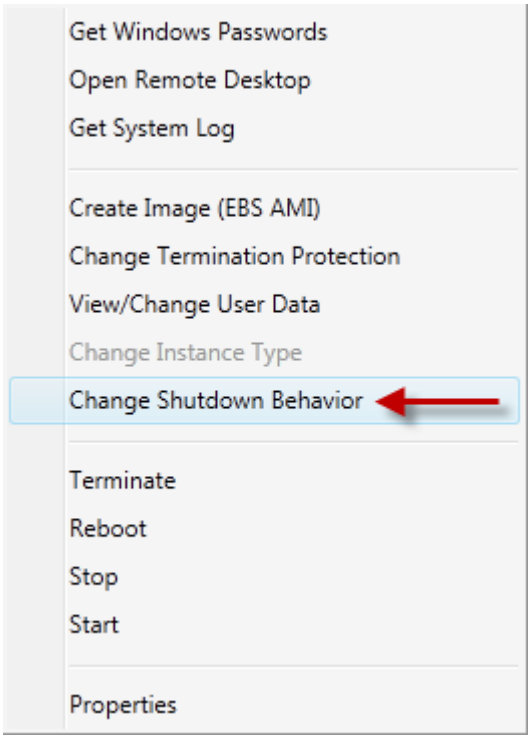
If you terminate an instance, it will continue to appear in the **Instance** list alongside running or stopped instances. Eventually, Amazon reclaims these instances and they disappear from the list. You are not charged for instances in a terminated state.



To specify the behavior of an EC2 instance at shutdown

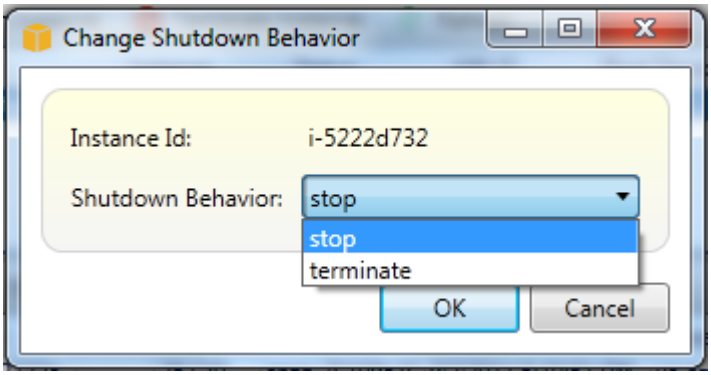
The Amazon Toolkit enables you to specify whether an Amazon EC2 instance will stop or terminate if **Shutdown** is selected from the **Start** menu.

1. In the **Instances** list, right-click an Amazon EC2 instance, and then choose **Change shutdown behavior**.



Change Shutdown Behavior menu item

2. In the **Change Shutdown Behavior** dialog box, from the **Shutdown Behavior** drop-down list, choose **Stop** or **Terminate**.



Managing Amazon ECS Instances

Amazon Explorer provides detailed views of Amazon Elastic Container Service (Amazon ECS) clusters and container repositories. You can create, delete and manage cluster and container details from within the Visual Studio development environment.

Modifying service properties

You can view service details, service events and service properties from the cluster view.

1. In Amazon Explorer, open the context (right-click) menu for the cluster to manage, and then choose **View**.
2. In the ECS Cluster view, click **Services** on the left, and then click the **Details** tab in the details view. You can click **Events** to see event messages and **Deployments** to deployment status.
3. Click **Edit**. You can change the desired task count and the minimum and maximum healthy percent.
4. Click **Save** to accept changes or **Cancel** to revert to existing values.

Stopping a task

You can see the current status of tasks and stop one or more tasks in the cluster view.

To stop a task

1. In Amazon Explorer, open the context (right-click) menu for the cluster with tasks you wish to stop, and then choose **View**.
2. In the ECS Cluster view, click **Tasks** on the left.
3. Make sure **Desired Task Status** is set to `Running`. Choose the individual tasks to stop and then click **Stop** or click **Stop All** to select and stop all running tasks.
4. In the **Stop Tasks** dialog box, choose **Yes**.

Deleting a service

You can delete services from a cluster from the cluster view.

To delete a cluster service

1. In Amazon Explorer, open the context (right-click) menu for the cluster with a service you want to delete, and then choose **View**.
2. In the ECS Cluster view, click **Services** on the left, and then click **Delete**.
3. In the **Delete Cluster** dialog box, if there is a load balancer and target group in your cluster, you can choose to delete them with the cluster. They will not be used when the service is deleted.

4. In the **Delete Cluster** dialog box, choose **OK**. When the cluster is deleted, it will be removed from the Amazon Explorer.

Deleting a cluster

You can delete an Amazon Elastic Container Service cluster from Amazon Explorer.

To delete a cluster

1. In Amazon Explorer, open the context (right-click) menu for the cluster you want to delete under the **Clusters** node of **Amazon ECS**, and then choose **Delete**.
2. In the **Delete Cluster** dialog box, choose **OK**. When the cluster is deleted, it will be removed from the Amazon Explorer.

Creating a repository

You can create an Amazon Elastic Container Registry repository from Amazon Explorer.

To create a repository

1. In Amazon Explorer, open the context (right-click) menu of the **Repositories** node under **Amazon ECS**, and then choose **Create Repository**.
2. In the **Create Repository** dialog box, provide a repository name and then choose **OK**.

Deleting a repository

You can delete an Amazon Elastic Container Registry repository from Amazon Explorer.

To delete a repository

1. In Amazon Explorer, open the context (right-click) menu of the **Repositories** node under **Amazon ECS**, and then choose **Delete Repository**.
2. In the **Delete Repository** dialog box, you can choose to delete the repository even if it contains images. Otherwise, it will only be deleted if it is empty. Click **Yes**.

Managing Security Groups from Amazon Explorer

The Toolkit for Visual Studio enables you to create and configure security groups to use with Amazon Elastic Compute Cloud (Amazon EC2) instances and Amazon CloudFormation. When you launch Amazon EC2 instances or deploy an application to Amazon CloudFormation, you specify a security group to associate with the Amazon EC2 instances. (Deployment to Amazon CloudFormation creates Amazon EC2 instances.)

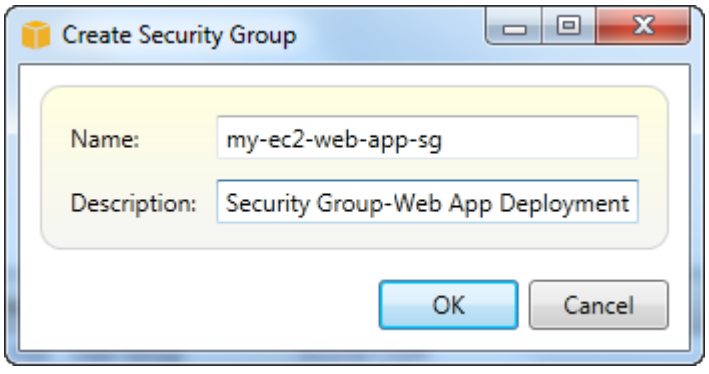
A security group acts like a firewall on incoming network traffic. The security group specifies which types of network traffic are allowed on an Amazon EC2 instance. It can also specify that incoming traffic will be accepted from certain IP addresses only or from specified users or other security groups only.

Creating a Security Group

In this section, we'll create a security group. After it has been created, the security group will not have any permissions configured. Configuring permissions is handled through an additional operation.

To create a security group

1. In Amazon Explorer, under the **Amazon EC2** node, open the context (right-click) menu on the **Security Groups** node, and then choose **View**.
2. On the **EC2 Security Groups** tab, choose **Create Security Group**.
3. In the **Create Security Group** dialog box, type a name and description for the security group, and then choose **OK**.

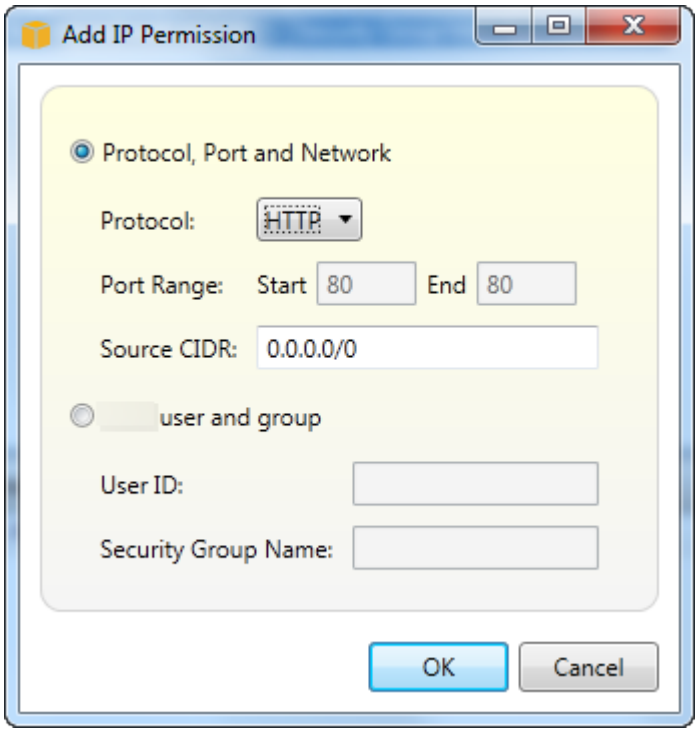


Adding Permissions to Security Groups

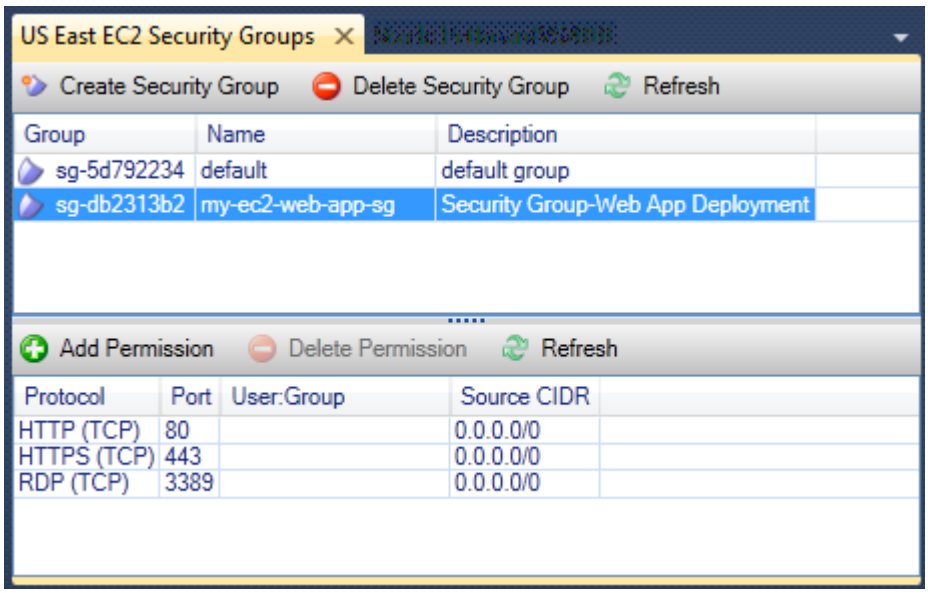
In this section, we'll add permissions to the security group to allow web traffic through the HTTP and HTTPS protocols. We'll also allow other computers to connect by using Windows Remote Desktop Protocol (RDP).

To add permissions to a security group

1. On the **EC2 Security Groups** tab, choose a security group and then choose the **Add Permission** button.
2. In the **Add IP Permission** dialog box, choose the **Protocol, Port and Network** radio button, and then from the **Protocol** drop-down list, choose **HTTP**. The port range automatically adjusts to port 80, the default port for HTTP. The **Source CIDR** field defaults to 0.0.0.0/0, which specifies that HTTP network traffic will be accepted from any external IP address. Choose **OK**.



   Open port 80 (HTTP) for this security group
3. Repeat this process for HTTPS and RDP. Your security groups permissions should now look like the following.



You can also set permissions in the security group by specifying a user ID and security group name. In this case, Amazon EC2 instances in this security group will accept all incoming network traffic from Amazon EC2 instances in the specified security group. You must also specify the user ID as a way to disambiguate the security group name; security group names are not required to be unique across all of Amazon. For more information about security groups, go to the EC2 documentation.

Creating an AMI from an Amazon EC2 Instance

You can create an Amazon Machine Image (AMI) with the Amazon Toolkit for Visual Studio. For more detailed information about AMIs, see the Amazon Machine Images (AMI) topic in the *Amazon Elastic Compute Cloud for Windows Instances* User Guide.

To create an AMI from an exiting Amazon EC2 instance, complete the following procedure.

Creating an AMI from an existing Amazon EC2 instance

1. From the Amazon Toolkit Explorer, expand **Amazon EC2** and choose **Instances** to view a list of your existing instances.
2. Right-click the instance that you want to use as the basis for your AMI and choose **Create Image (ABS AMI)** to open the **Create Image** dialog window.
3. From the **Create Image** dialog window, add a name and a description for your image into the provided fields, then choose the **OK** button to continue.

4. The **Image Created** confirmation window opens in Visual Studio when the image is created, choose the **OK** button to continue.

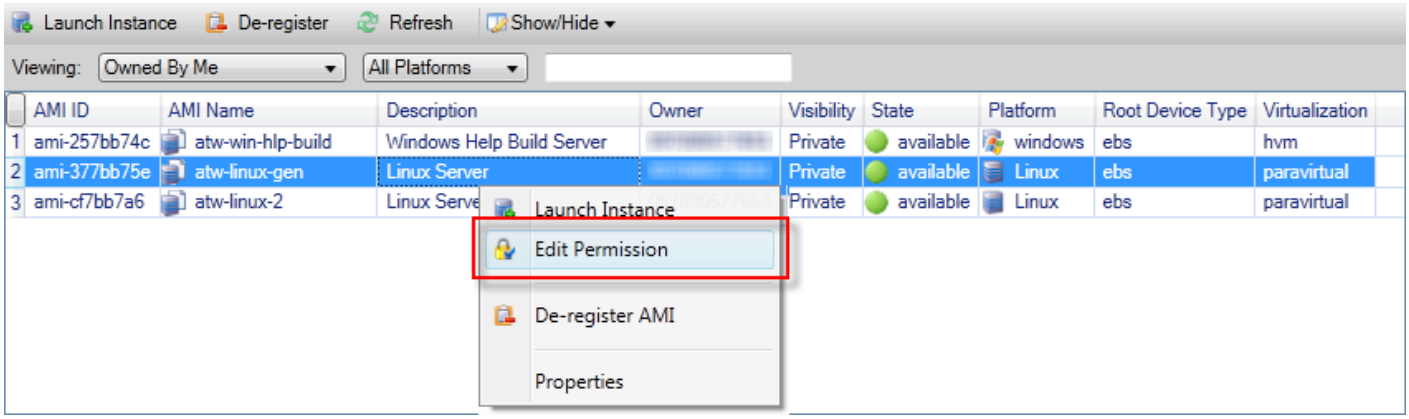
To view your new AMI with the Amazon Toolkit, expand **Amazon EC2** and double-click **AMIs** to open a window in the Visual Studio Editor payne that displays a list of your existing AMIs. If you don't see you new AMI in the list, choose the **Refresh** button located at the top of the AMI window.

Setting Launch Permissions on an Amazon Machine Image

You can set launch permissions on your Amazon Machine Images (AMIs) from the **AMIs** view in Amazon Explorer. You can use the **Set AMI Permissions** dialog box to copy permissions from AMIs.

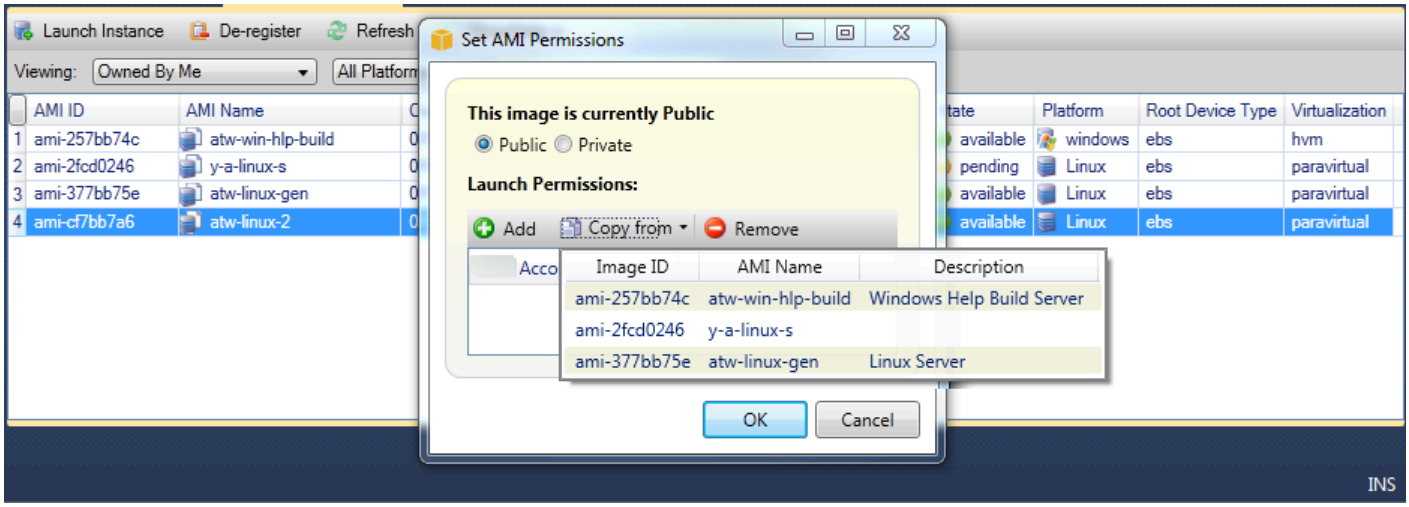
To set permissions on an AMI

1. In the **AMIs** view in Amazon Explorer, open the context (right-click) menu on an AMI, and then choose **Edit Permission**.



2. There are three options available in the **Set AMI Permissions** dialog box:
   - To give launch permission, choose **Add**, and type the account number for the Amazon user to whom you are giving launch permission.
   - To remove launch permission, choose the account number for the Amazon user from whom you are removing launch permission, and choose **Remove**.
   - To copy permissions from one AMI to another, choose an AMI from the list, and choose **Copy from**. The users who have launch permissions on the AMI you chose will be given launch permissions on the current AMI. You can repeat this process with other AMIs in the **Copy-from** list to copy permissions from multiple AMIs into the target AMI.

The **Copy-from** list contains only those AMIs owned by the account that was active when the **AMIs** view was displayed from Amazon Explorer. As a result, the **Copy-from** list might not display any AMIs if no other AMIs are owned by the active account.



Copy AMI permissions dialog box

Amazon Virtual Private Cloud (VPC)

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch Amazon Web Services resources into a virtual network you've defined. This virtual network resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of Amazon. For more information, go to the Amazon VPC User Guide.

The Toolkit for Visual Studio enables a developer to access VPC functionality similar to that exposed by the Amazon Web Services Management Console but from the Visual Studio development environment. The **Amazon VPC** node of Amazon Explorer includes subnodes for the following areas.

- VPCs
- Subnets
- Elastic IPs
- Internet Gateways
- Network ACLs
- Route Tables

- Security Groups

Creating a Public-Private VPC for Deployment with Amazon Elastic Beanstalk

This section describes how to create an Amazon VPC that contains both public and private subnets. The public subnet contains an Amazon EC2 instance that performs network address translation (NAT) to enable instances in the private subnet to communicate with the public internet. The two subnets must reside in the same Availability Zone (AZ).

This is the minimal VPC configuration required to deploy an Amazon Elastic Beanstalk environment in a VPC. In this scenario, the Amazon EC2 instances that host your application reside in the private subnet; the Elastic Load Balancing load balancer that routes incoming traffic to your application resides in the public subnet.

For more information about network address translation (NAT), go to NAT Instances in the *Amazon Virtual Private Cloud User Guide*. For an example of how to configure your deployment to use a VPC, see Deploying to Elastic Beanstalk.

To create a public-private subnet VPC

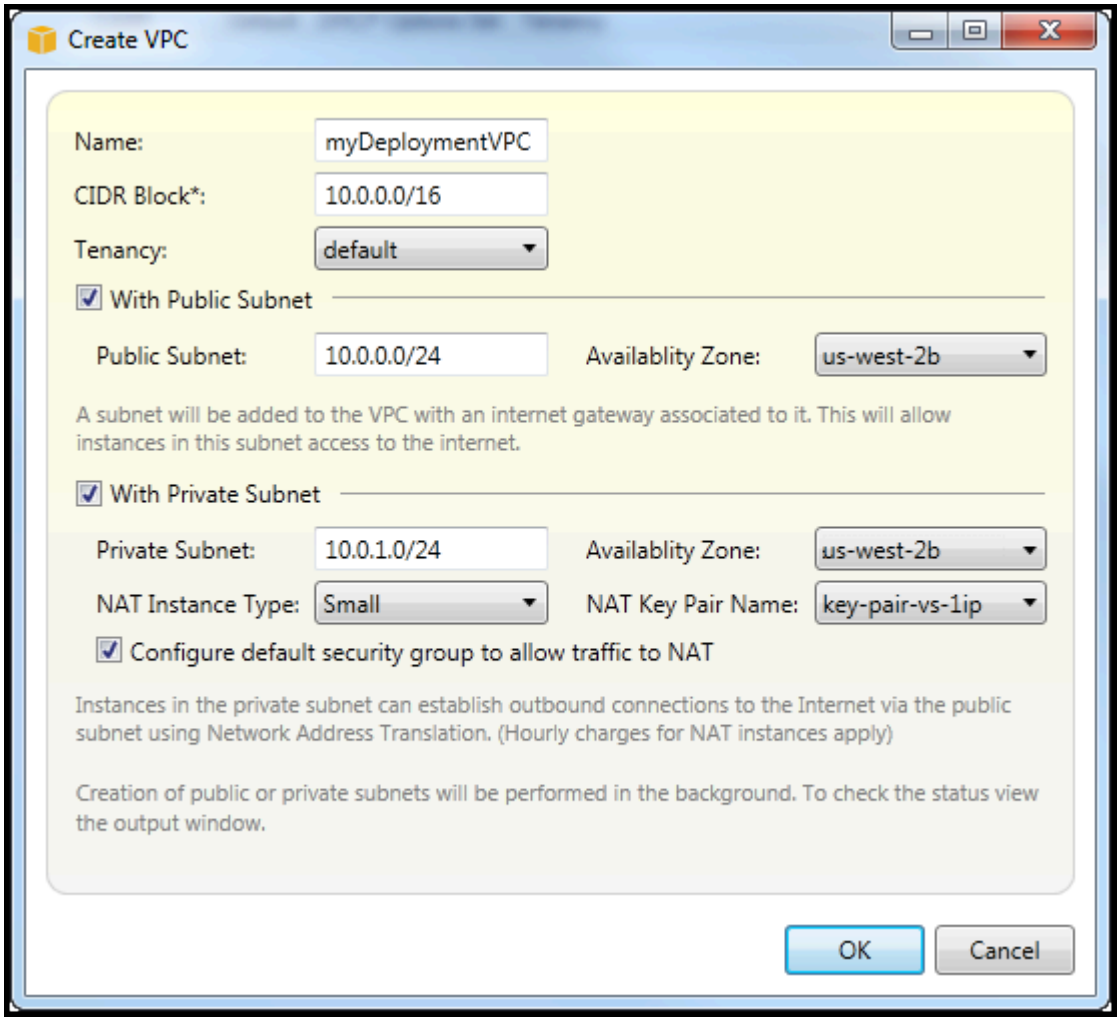
1. In the **Amazon VPC** node in Amazon Explorer, open the **VPCs** subnode, then choose **Create VPC**.

2. Configure the VPC as follows:
   - Type a name for your VPC.
   - Select the **With Public Subnet** and the **With Private Subnet** check boxes.

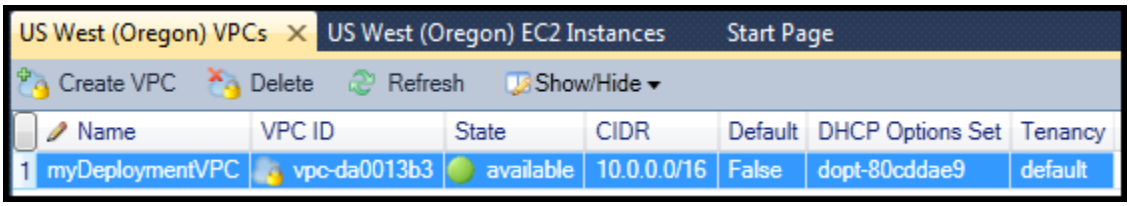
- From the **Availability Zone** drop-down list box for each subnet, choose an Availability Zone. Be sure to use the same AZ for both subnets.
- For the private subnet, in **NAT Key Pair Name**, provide a key pair. This key pair is used for the Amazon EC2 instance that performs network address translation from the private subnet to the public Internet.
- Select the **Configure default security group to allow traffic to NAT** check box.

Type a name for your VPC. Select the **With Public Subnet** and the **With Private Subnet** check boxes. From the **Availability Zone** drop-down list box for each subnet, choose an Availability Zone. Be sure to use the same AZ for both subnets. For the private subnet, in **NAT Key Pair Name**, provide a key pair. This key pair is used for the Amazon EC2 instance that performs network address translation from the private subnet to the public Internet. Select the **Configure default security group to allow traffic to NAT** check box.

Choose **OK**.

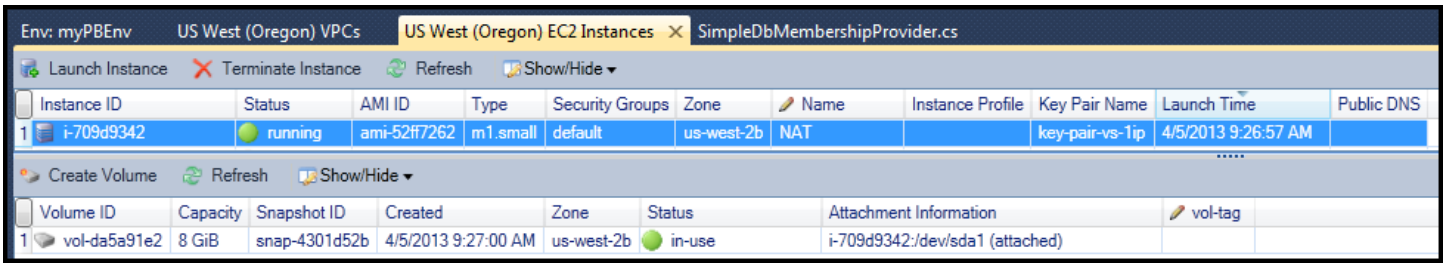


You can view the new VPC in the **VPCs** tab in Amazon Explorer.



The NAT instance might take a few minutes to launch. When it is available, you can view it by expanding the **Amazon EC2** node in Amazon Explorer and then opening the **Instances** subnode.

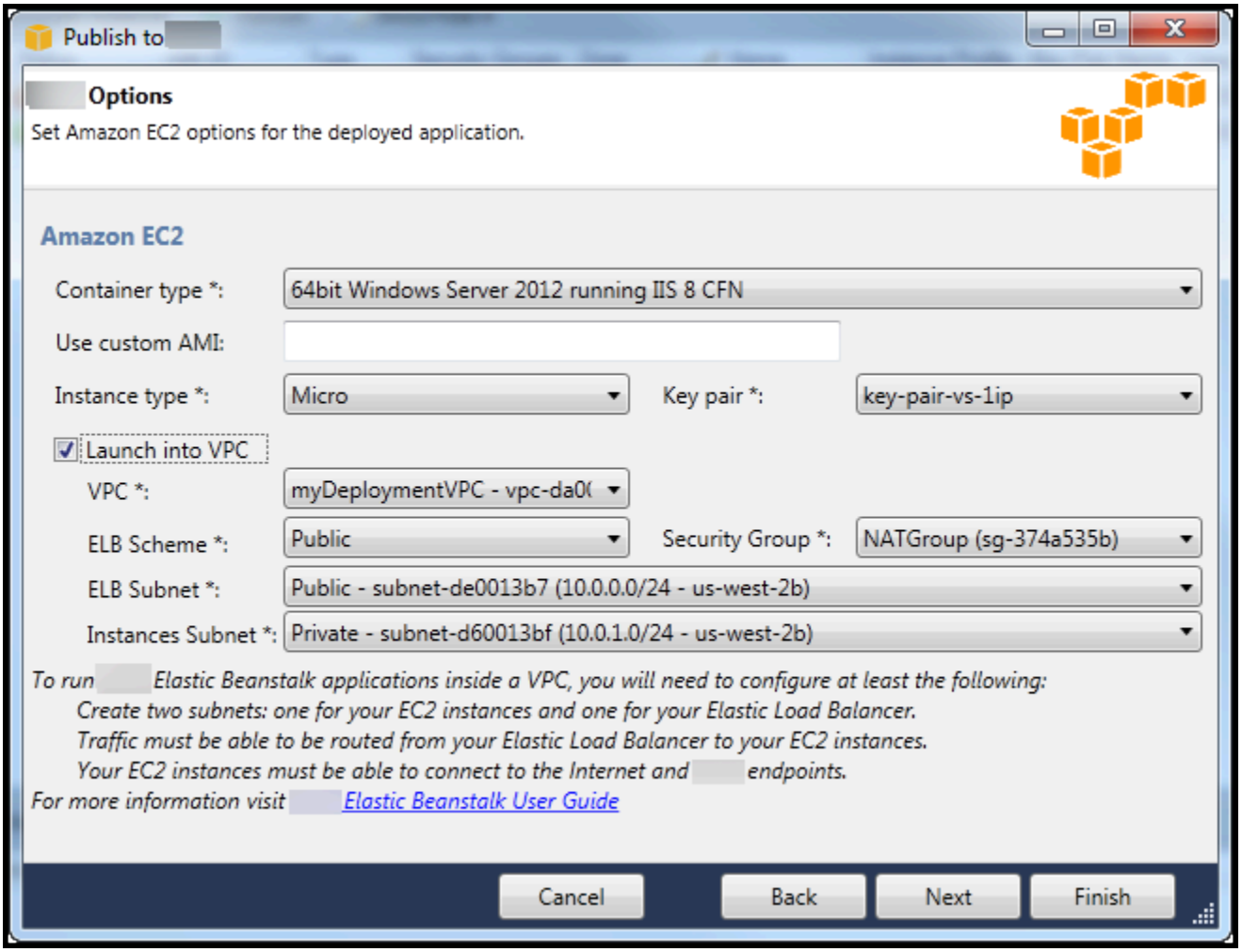
An Amazon Elastic Block Store (Amazon EBS) volume is created for the NAT instance automatically. For more information about Amazon EBS, see the Amazon Elastic Block Store (EBS) topic in the *Amazon EC2 User Guide for Linux Instances*.



If you deploy an application to an Amazon Elastic Beanstalk environment and choose to launch the environment in a VPC, the Toolkit will populate the **Publish to Amazon Web Services** dialog box with the configuration information for your VPC.

The Toolkit populates the dialog box with information only from VPCs that were created in the Toolkit, not from VPCs created using the Amazon Web Services Management Console. This is because when the Toolkit creates a VPC, it tags the components of the VPC so that it can access their information.

The following screenshot from the Deployment Wizard shows an example of a dialog box populated with values from a VPC created in the Toolkit.



To delete a VPC

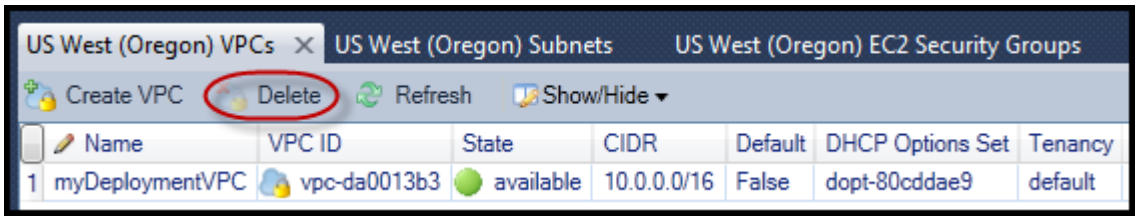
To delete the VPC, you must first terminate any Amazon EC2 instances in the VPC.

1. If you have deployed an application to an Amazon Elastic Beanstalk environment in the VPC, delete the environment. This will terminate any Amazon EC2 instances hosting your application along with the Elastic Load Balancing load balancer.

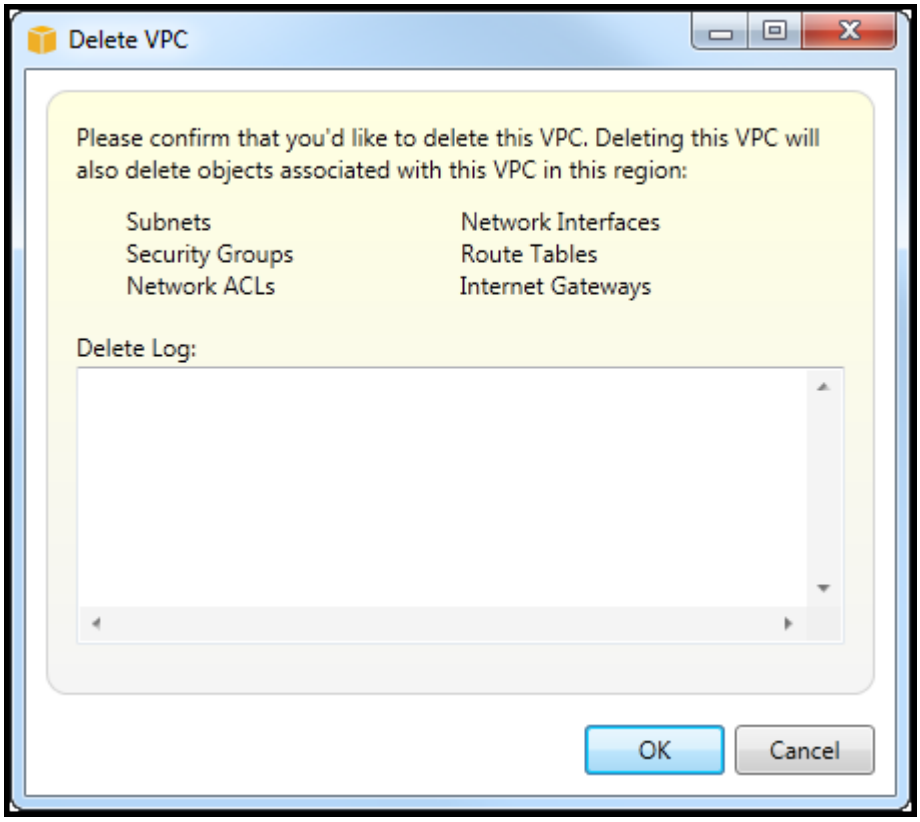
   If you attempt to directly terminate the instances hosting your application without deleting the environment, the Auto Scaling service will automatically create new instances to replace the deleted ones. For more information, go to the Auto Scaling Developer Guide.
2. Delete the NAT instance for the VPC.

   You do not need to delete the Amazon EBS volume associated with the NAT instance in order to delete the VPC. However, if you do not delete the volume, you will continue to be charged for it even if you delete the NAT instance and the VPC.

3. On the **VPC** tab, choose the **Delete** link to delete the VPC.



4. In the **Delete VPC** dialog box, choose **OK**.



Using the Amazon CloudFormation Template Editor for Visual Studio

The Toolkit for Visual Studio includes an Amazon CloudFormation template editor and Amazon CloudFormation template projects for Visual Studio. The supported features include:

- Creating new templates (either empty or copied from an existing stack or sample template) using the supplied Amazon CloudFormation template project type.
- Editing templates with automatic JSON validation, auto-completion, code folding, and syntax highlighting.

- Automatic suggestion of intrinsic functions and resource reference parameters for the field values in your template.
- Menu items to perform common actions for your template from Visual Studio.

Topics

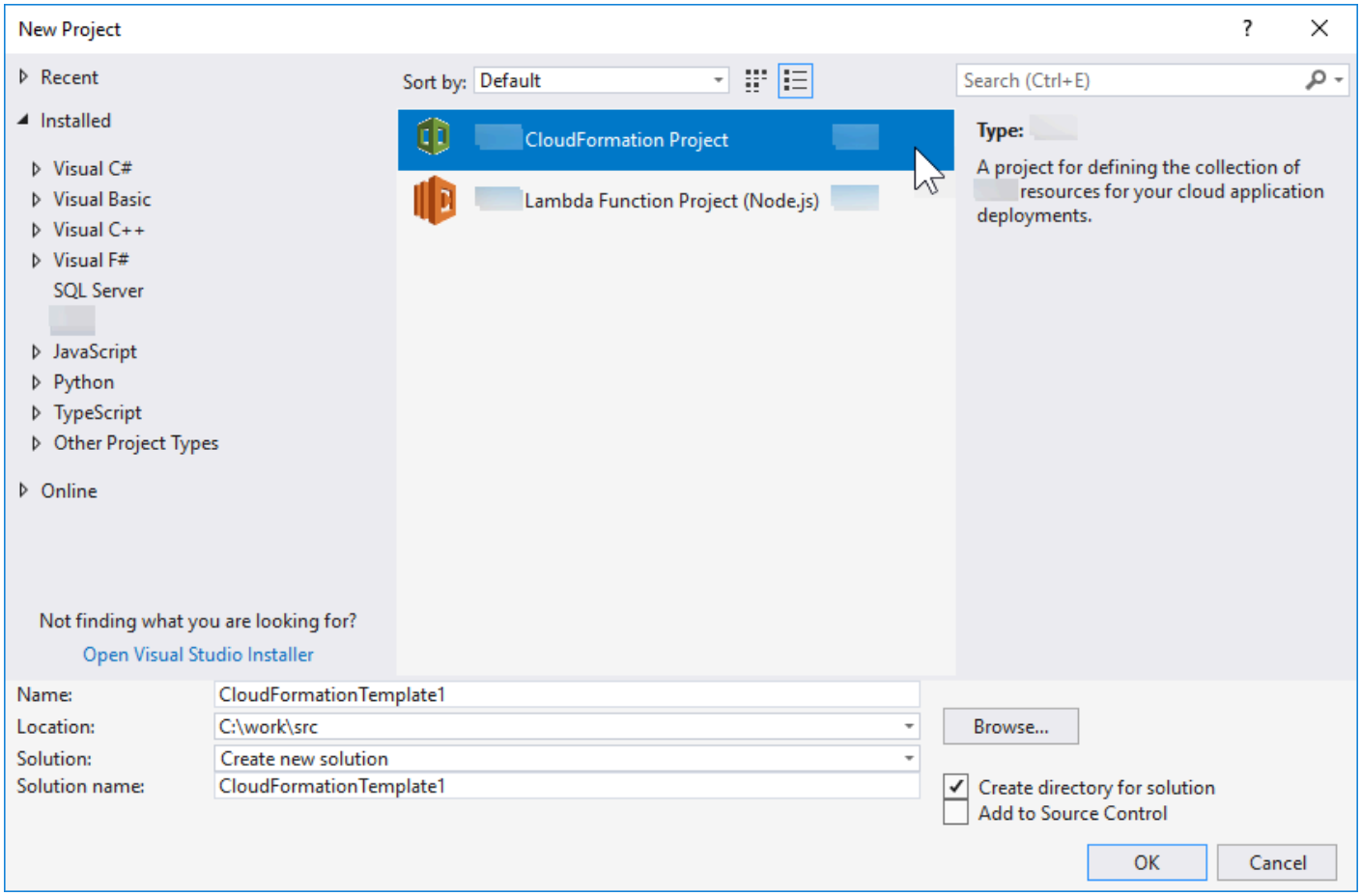
- Creating an Amazon CloudFormation Template Project in Visual Studio
- Deploying a Amazon CloudFormation Template in Visual Studio
- Formatting a Amazon CloudFormation Template in Visual Studio

Creating an Amazon CloudFormation Template Project in Visual Studio

To create a template project

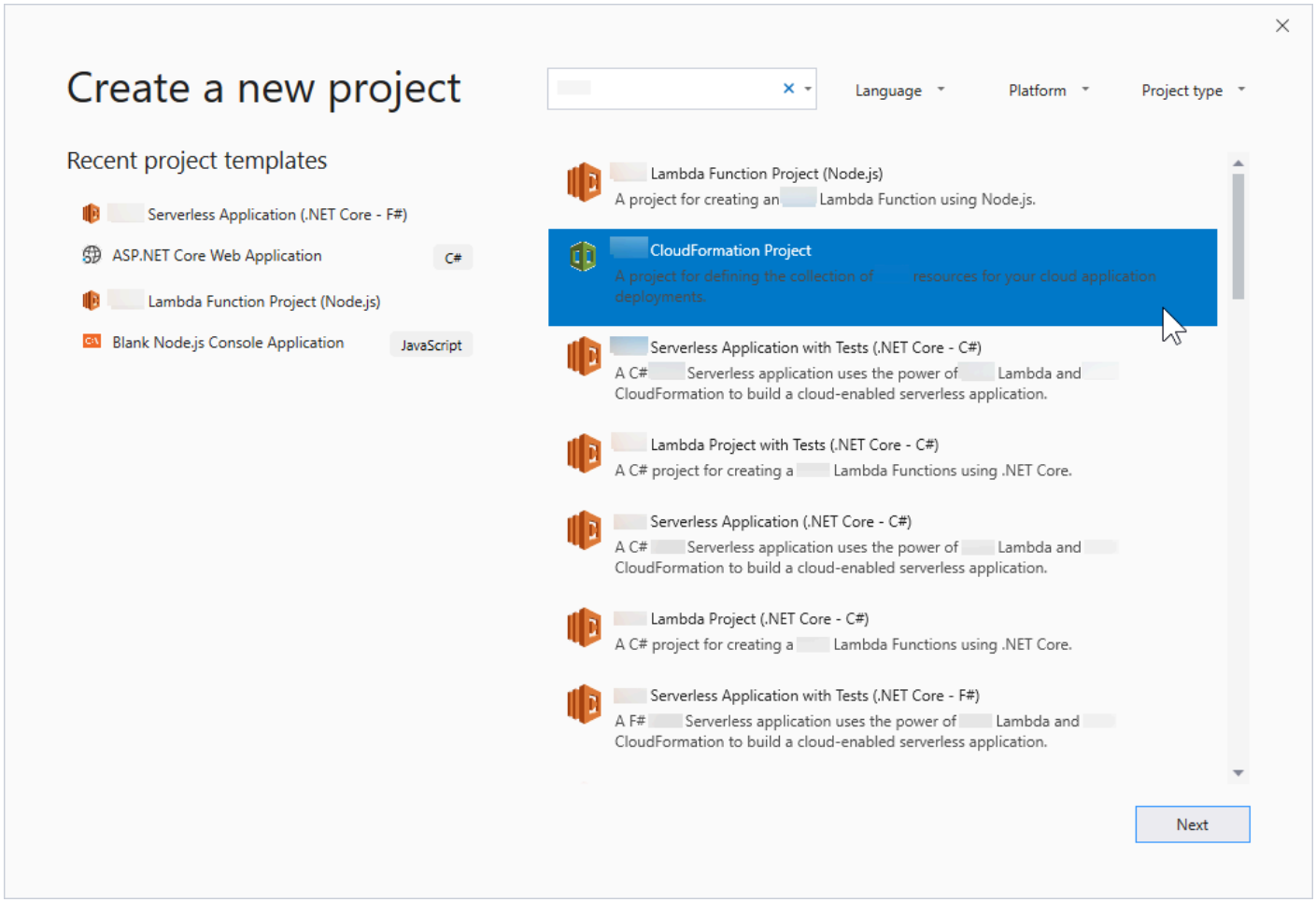
1. In Visual Studio, choose **File**, choose **New**, and then choose **Project**.
2. **For Visual Studio 2017**:

   In the **New Project** dialog box, expand **Installed** and select **Amazon**.



For Visual Studio 2019:

In the **New Project** dialog box, ensure that the **Language**, **Platform**, and **Project type** drop-down boxes are set to "All ..." and type **aws** in the **Search** field.

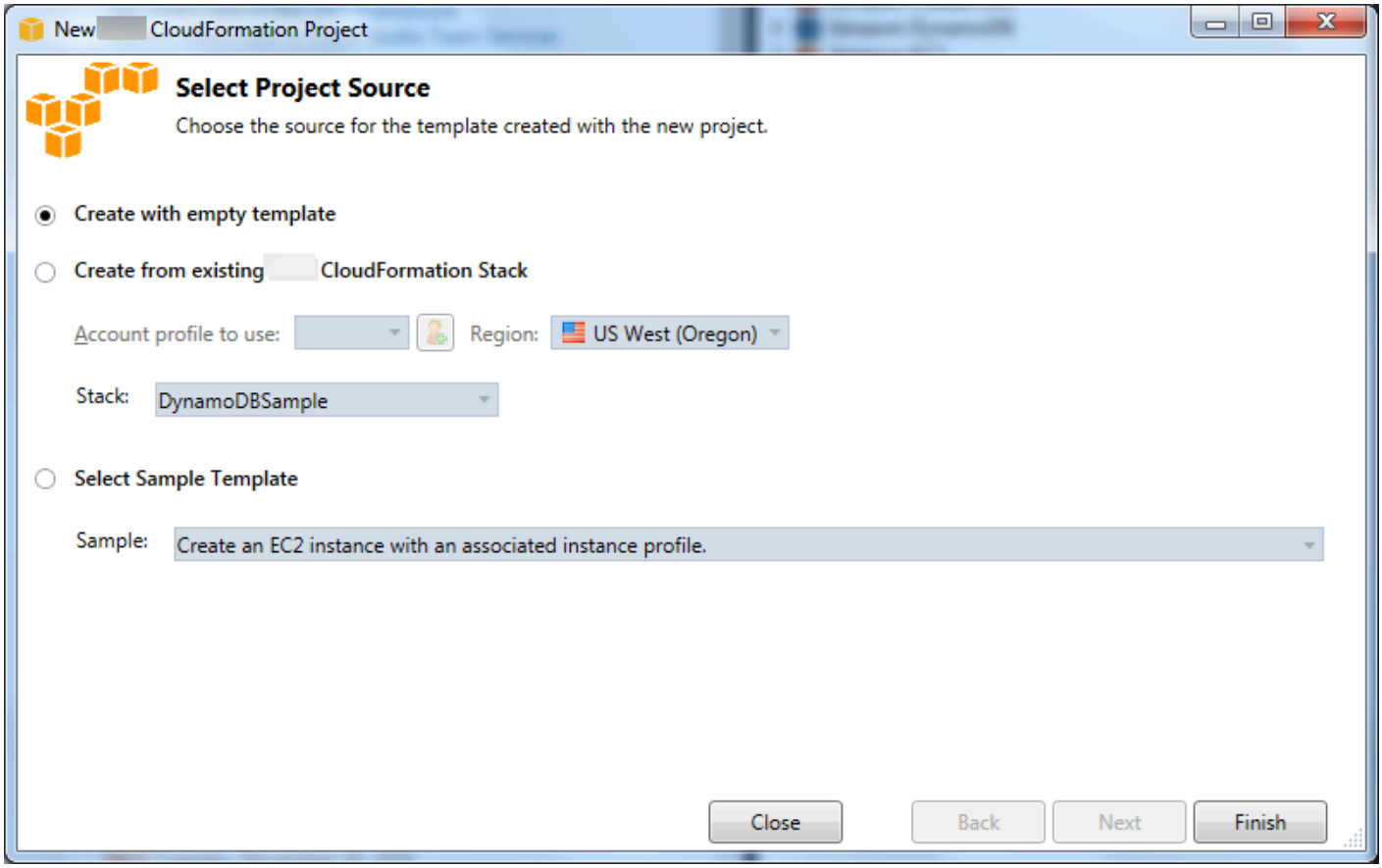


3. Select the **Amazon CloudFormation Project** template.
4. **For Visual Studio 2017**:

   Enter the desired **Name**, **Location**, etc., for your template project, and then click **OK**.

   For Visual Studio 2019:

   Click **Next**. In the next dialog, enter the desired **Name**, **Location**, etc., for your template project, and then click **Create**.
5. On the **Select Project Source** page, choose the source of the template you will create:
   - **Create with empty template** generates a new, empty Amazon CloudFormation template.
   - **Create from existing Amazon |CFN| stack** generates a template from an existing stack in your Amazon account. (The stack doesn't need to have a status of `CREATE_COMPLETE`.)
   - **Select sample template** generates a template from one of the Amazon CloudFormation sample templates.

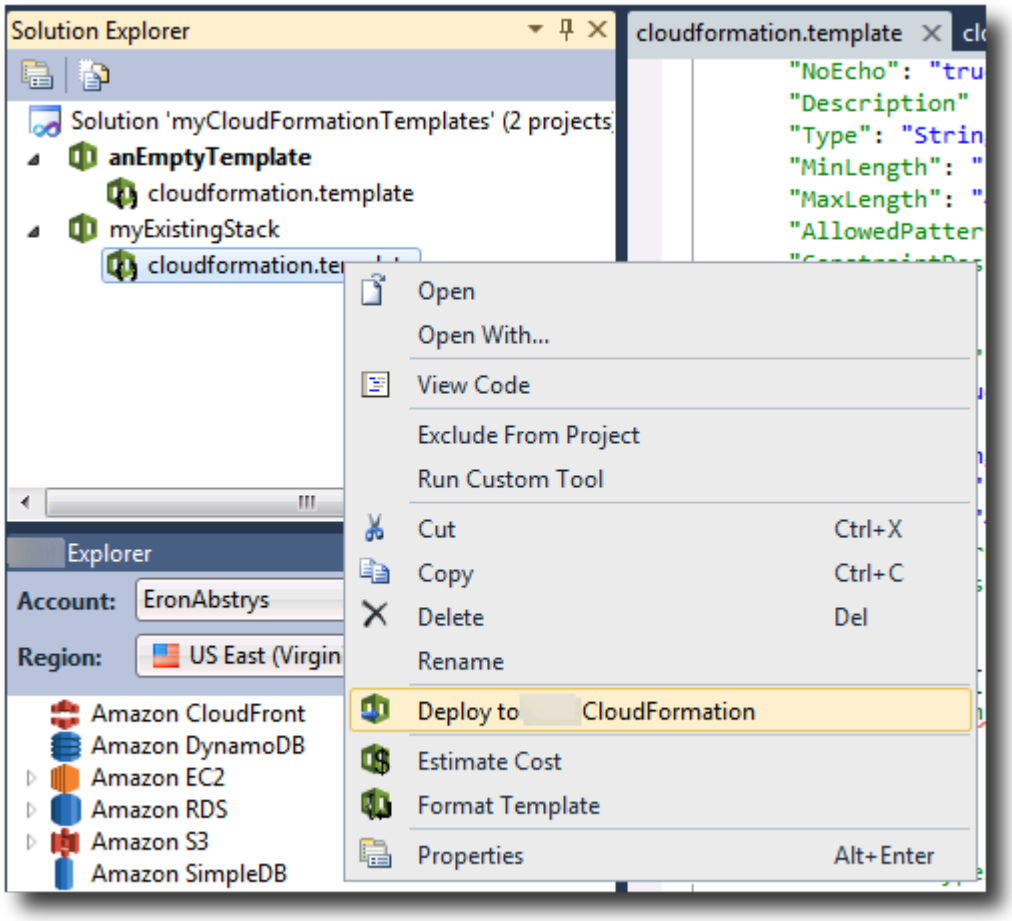


6. To complete the creation of your Amazon CloudFormation template project, choose **Finish**.

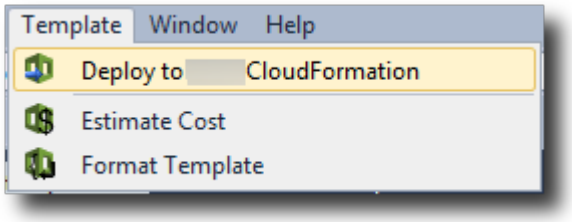
Deploying a Amazon CloudFormation Template in Visual Studio

To deploy an CFN template

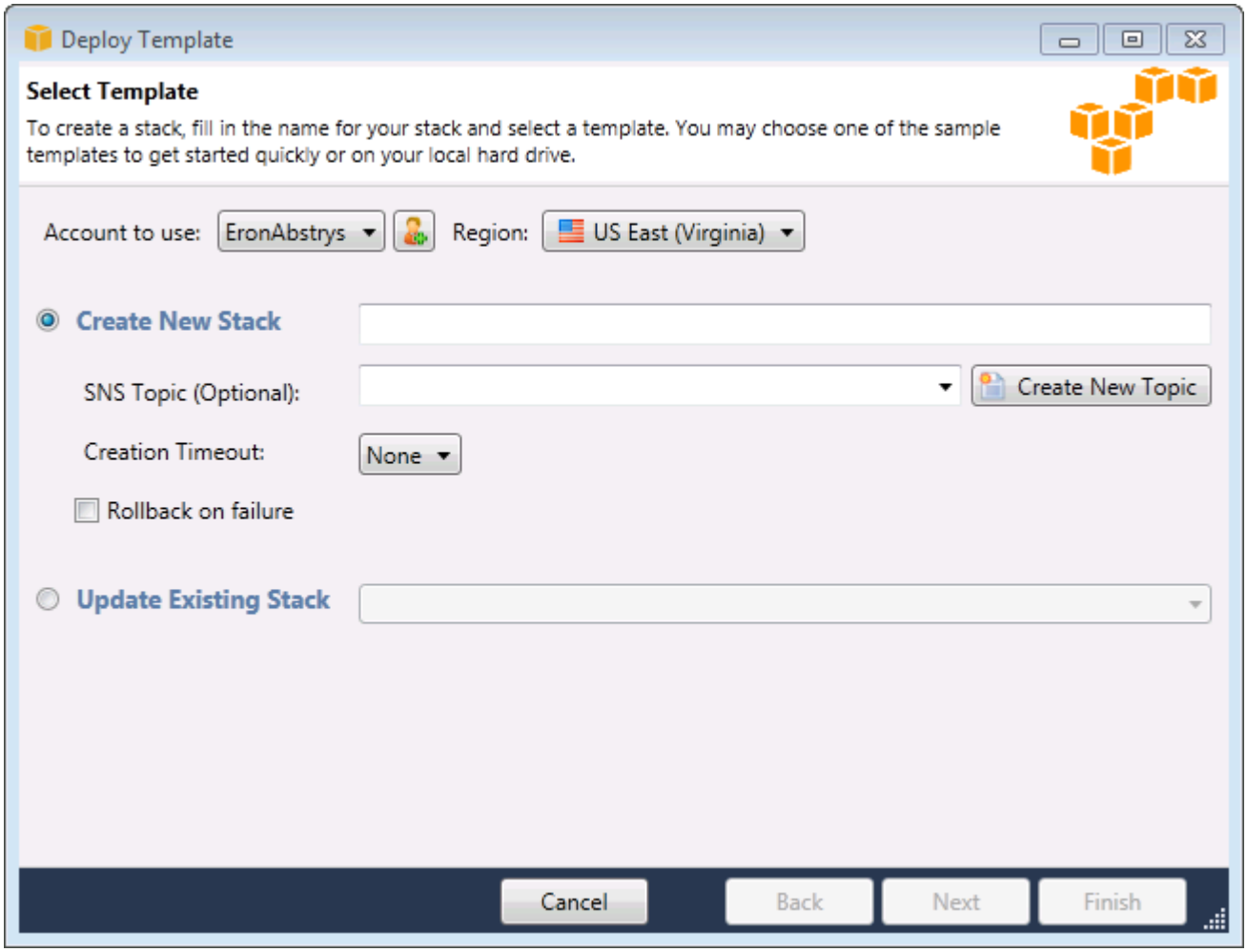
1. In Solution Explorer, open the context (right-click) menu for the template you want to deploy, and choose **Deploy to Amazon CloudFormation**.



Alternatively, to deploy the template you're currently editing, from the **Template** menu, choose **Deploy to Amazon CloudFormation** .



2. On the **Deploy Template** page, choose the Amazon Web Services account to use to launch the stack and the region where it will be launched.

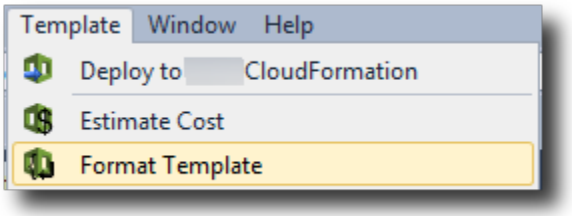


3. Choose **Create New Stack** and type a name for your stack.
4. Choose any (or none) of the following options:
   - To receive notifications about the stack's progress, from the **SNS Topic** drop-down list, choose an SNS topic. You can also create an SNS topic by choosing **Create New Topic** and typing an email address in the box.
   - Use **Creation Timeout** to specify how long Amazon CloudFormation should allow for the stack to be created before it is declared failed (and rolled back, unless the **Rollback on failure** option is cleared).
   - Use **Rollback on failure** if you want the stack to roll back (that is, delete itself) on failure. Leave this option cleared if you would like the stack to remain active for debugging purposes, even if it has failed to complete the launch.
5. Choose **Finish** to launch the stack.

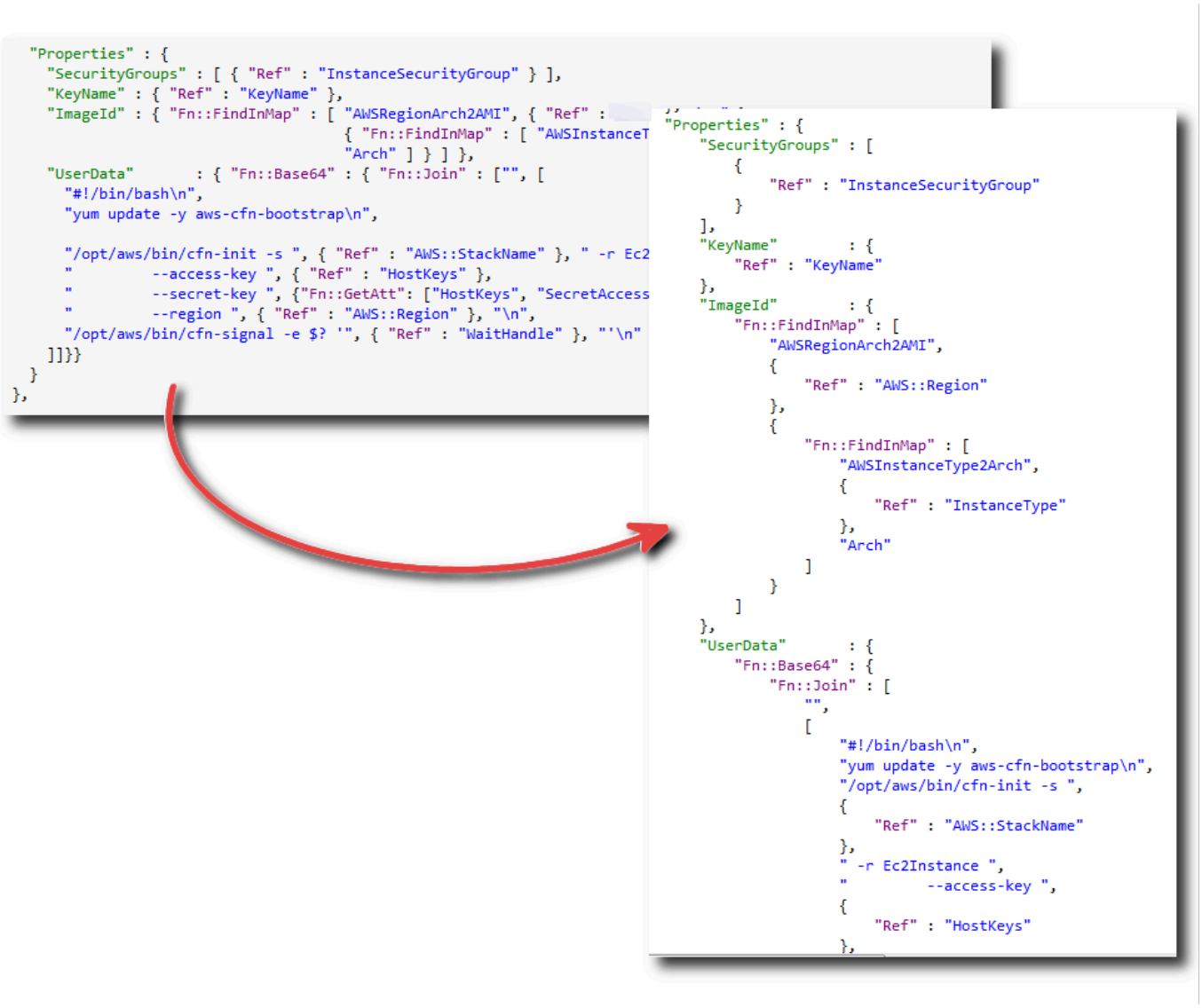
Formatting a Amazon CloudFormation Template in Visual Studio

- In Solution Explorer, open the context (right-click) menu for the template and choose **Format Template**.

  Alternatively, to format the template you're currently editing, from the **Template** menu, choose **Format Template**.

  Your JSON code will be formatted so that its structure is clearly presented.



Using Amazon S3 from Amazon Explorer

Amazon Simple Storage Service (Amazon S3) enables you to store and retrieve data from any connection to the Internet. All data you store on Amazon S3 is associated with your account and, by default, can only be accessed by you. The Toolkit for Visual Studio enables you to store data on Amazon S3 and to view, manage, retrieve, and distribute that data.

Amazon S3 uses the concept of buckets, which you can think of as being similar to file systems or logical drives. Buckets can contain folders, which are similar to directories, and objects, which are similar to files. In this section, we'll be using these concepts as we walk through the Amazon S3 functionality exposed by the Toolkit for Visual Studio.

Note

To use this tool, your IAM policy must grant permissions for the `s3:GetBucketAcl`, `s3:GetBucket`, and `s3:ListBucket` actions. For more information, see Overview of Amazon IAM Policies.

Creating an Amazon S3 Bucket

The bucket is most fundamental unit of storage in Amazon S3.

To create an S3 bucket

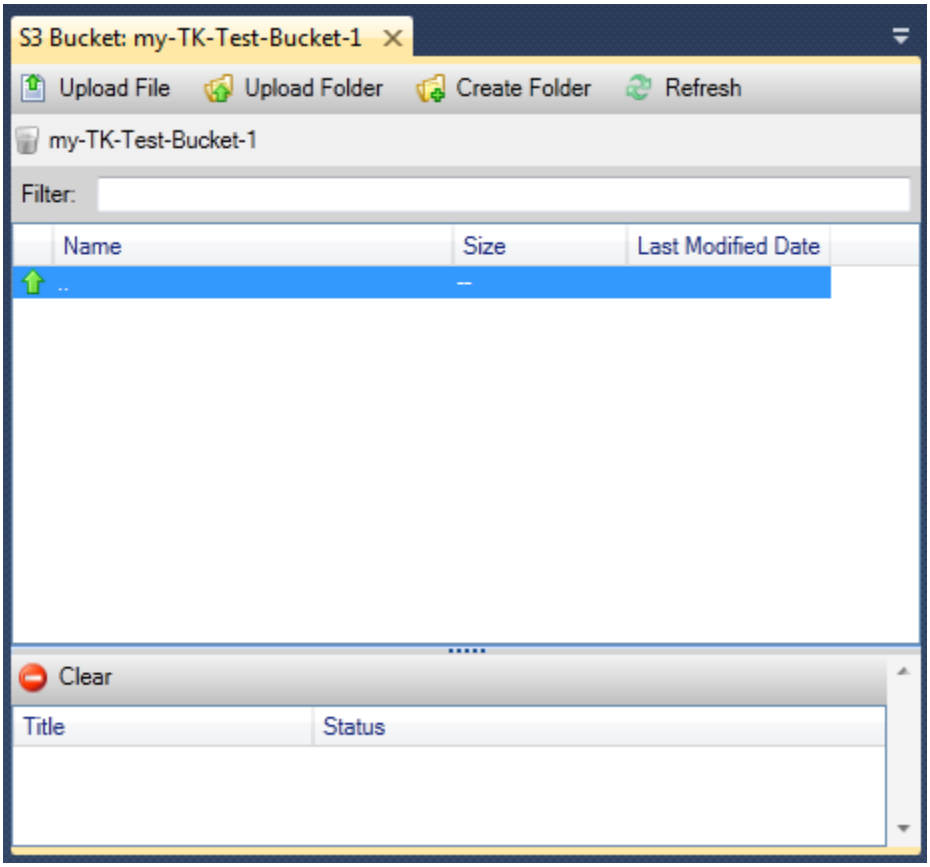
1. In Amazon Explorer, open the context (right-click) menu for the **Amazon S3** node, and then choose **Create Bucket**.
2. In the **Create Bucket** dialog box, type a name for the bucket. Bucket names must be unique across Amazon. For information about other constraints, go to the Amazon S3 documentation.
3. Choose **OK**.

Managing Amazon S3 Buckets from Amazon Explorer

In Amazon Explorer, the following operations are available when you open a context (right-click) menu for an Amazon S3 bucket.

Browse

Displays a view of the objects contained in the bucket. From here, you can create folders or upload files or entire directories and folders from your local computer. The lower pane displays status messages about the upload process. To clear these messages, choose the **Clear** icon. You can also access this view of the bucket by double-clicking the bucket name in Amazon Explorer.



Properties

Displays a dialog box where you can do the following:

- Set Amazon S3 permissions that scope to:
  - you as the bucket owner.
  - all users who have been authenticated on Amazon.
  - everyone with Internet access.
- Turn on logging for the bucket.
- Set up a notification using the Amazon Simple Notification Service (Amazon SNS) so that if you are using Reduced Redundancy Storage (RRS), you are notified if data loss occurs. RRS is an Amazon S3 storage option that provides less durability than standard storage, but at reduced cost. For more information, see S3 FAQs.
- Create a static website using the data in the bucket.

Policy

Enables you to set up Amazon Identity and Access Management (IAM) policies for your bucket. For more information, go to the IAM documentation and the use cases for IAM and S3.

Create Pre-Signed URL

Enables you to generate a time-limited URL you can distribute to provide access to the contents of the bucket. For more information, see How to Create a Pre-Signed URL.

View Multi-Part Uploads

Enables you to view multipart uploads. Amazon S3 supports breaking large object uploads into parts to make the upload process more efficient. For more information, go to the discussion of multipart uploads in the S3 documentation.

Delete

Enables you to delete the bucket. You can only delete empty buckets.

Uploading Files and Folders to Amazon S3

You can use Amazon Explorer to transfer files or entire folders from your local computer to any of your buckets.

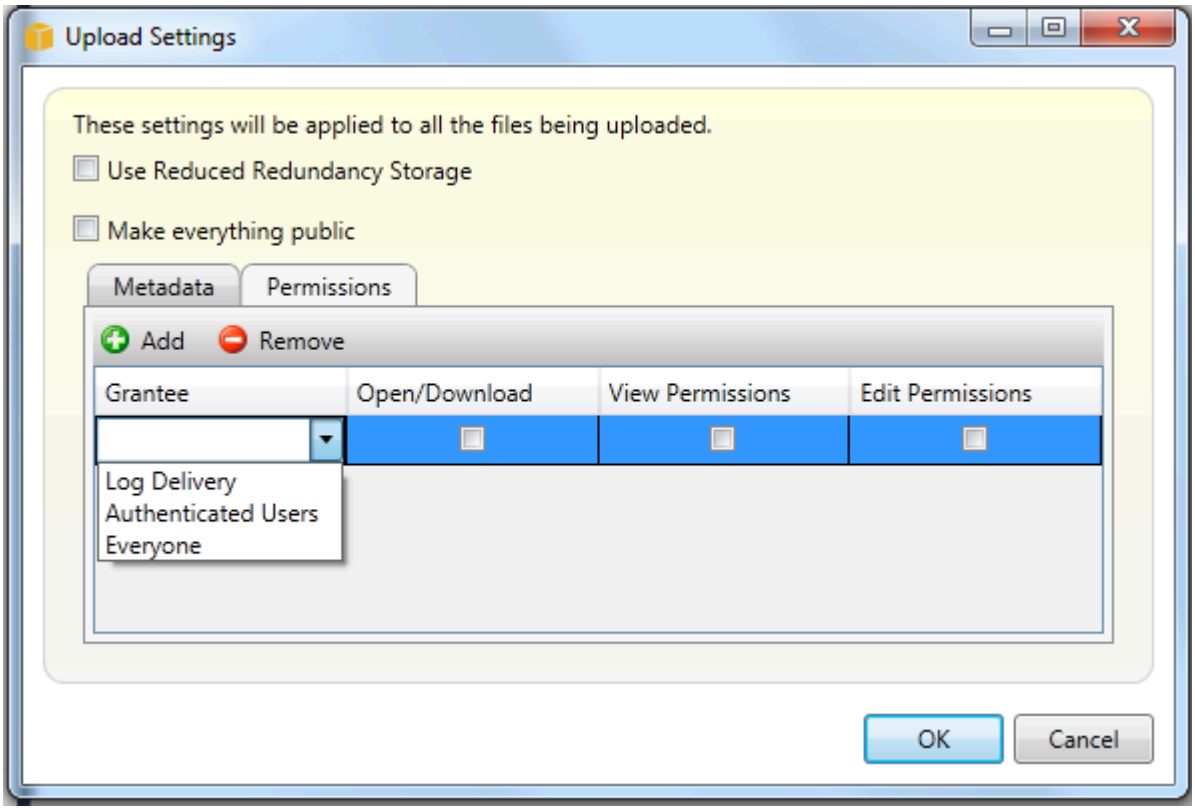
> **Note**
>
> If you upload files or folders that have the same name as files or folders that already exist in the Amazon S3 bucket, your uploaded files will overwrite the existing files without warning.

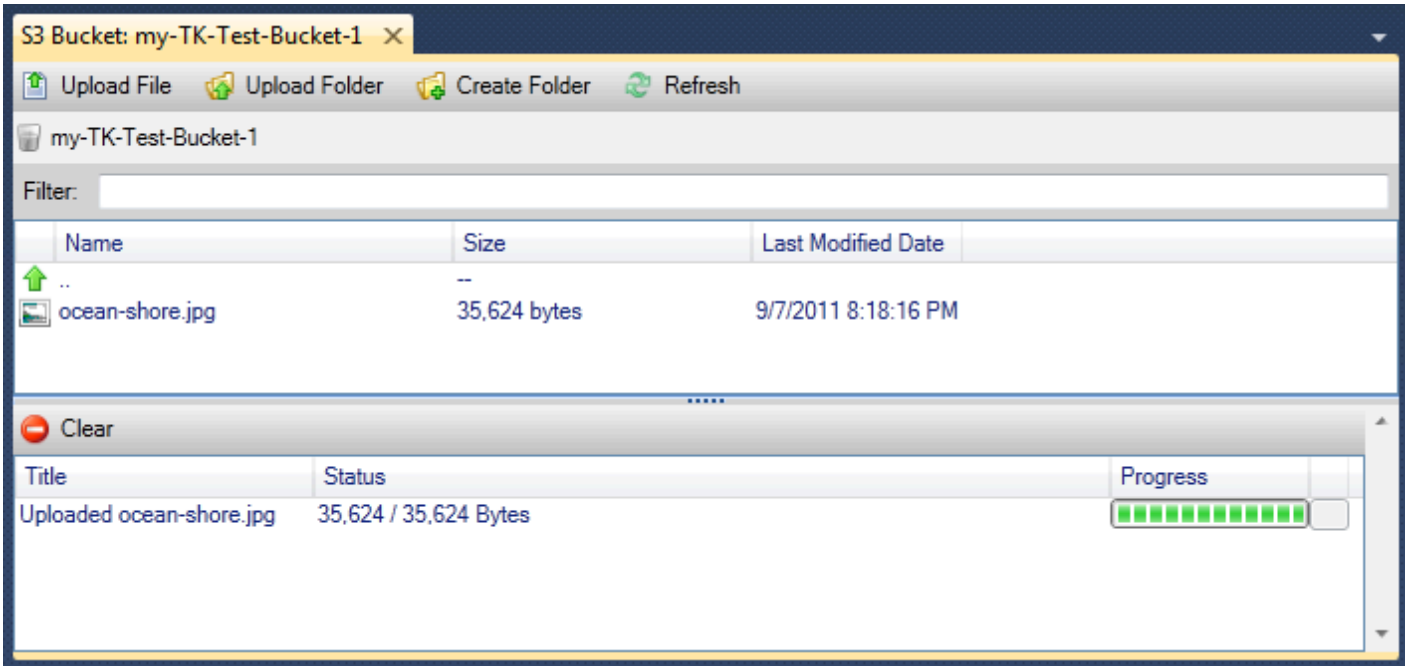
To upload a file to S3

1. In Amazon Explorer, expand the **Amazon S3** node, and double-click a bucket or open the context (right-click) menu for the bucket and choose **Browse**.
2. In the **Browse** view of your bucket, choose **Upload File** or **Upload Folder**.
3. In the **File-Open** dialog box, navigate to the files to upload, choose them, and then choose **Open**. If you are uploading a folder, navigate to and choose that folder, and then choose **Open**.

   The **Upload Settings** dialog box enables you to set metadata and permissions on the files or folder you are uploading. Selecting the **Make everything public** check box is equivalent to

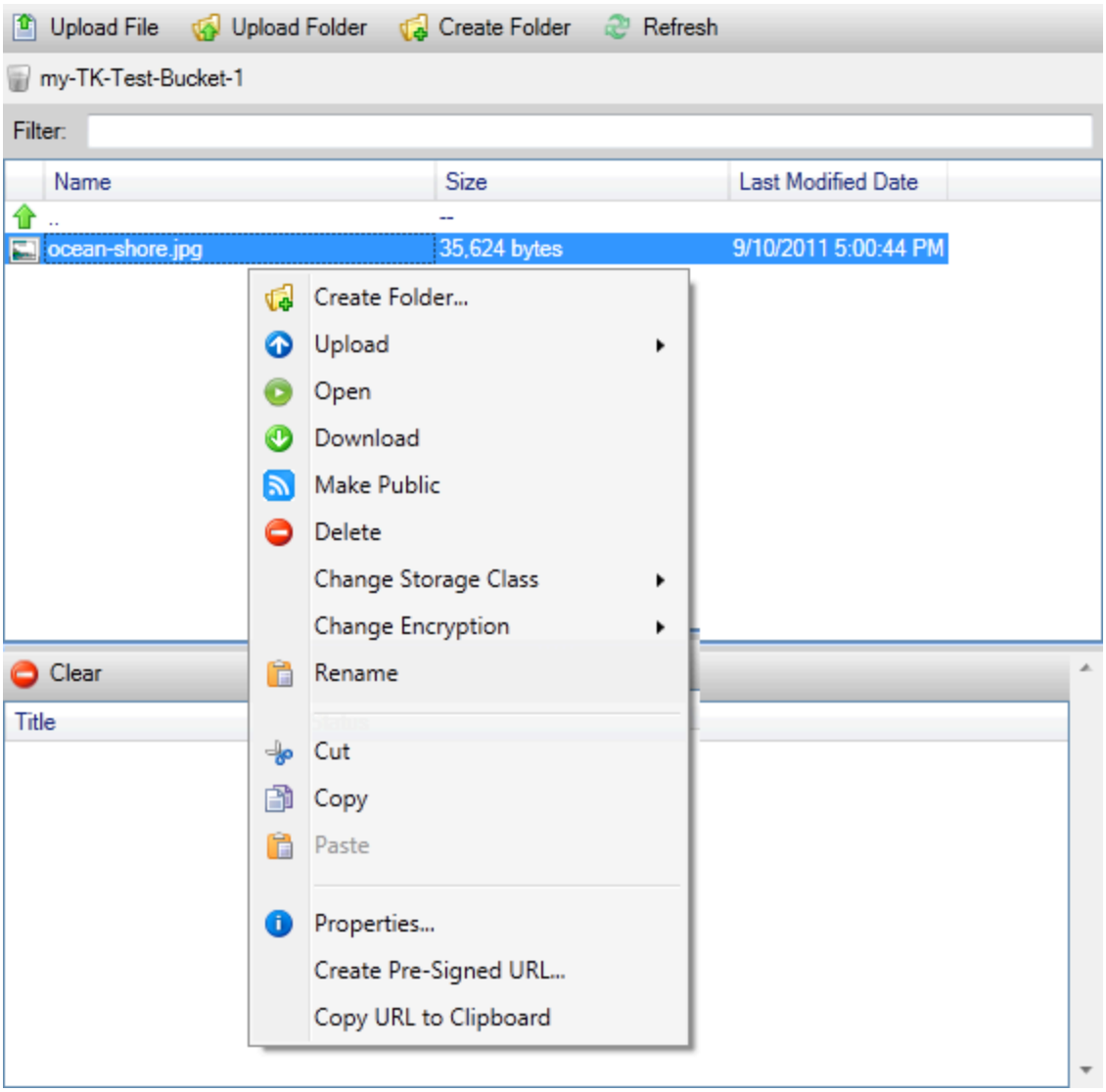
setting **Open/Download** permissions to **Everyone**. You can select the option to use Reduced Redundancy Storage for the uploaded files.





Amazon S3 File Operations from Amazon Toolkit for Visual Studio

If you choose a file in the Amazon S3 view and open the context (right-click) menu, you can perform various operations on the file.



Create Folder

Enables you to create a folder in the current bucket. (Equivalent to choosing the **Create Folder** link.)

Upload

Enables you to upload files or folders. (Equivalent to choosing the **Upload File** or **Upload Folder** links.)

Open

Attempts to open the selected file in your default browser. Depending on the type of file and your default browser's capabilities, the file might not be displayed. It might simply be downloaded by your browser instead.

Download

Opens a **Folder-Tree** dialog box to enable you to download the selected file.

Make Public

Sets permissions on the selected file to **Open/Download** and **Everyone**. (Equivalent to selecting the **Make everything public** check box on the **Upload Settings** dialog box.)

Delete

Deletes the selected files or folders. You can also delete files or folders by choosing them and pressing `Delete`.

Change Storage Class

Sets the storage class to either Standard or Reduced Redundancy Storage (RRS). To view the current storage class setting, choose **Properties**.

Change Encryption

Enables you to set server-side encryption on the file. To view the current encryption setting, choose **Properties**.

Rename

Enables you to rename a file. You cannot rename a folder.

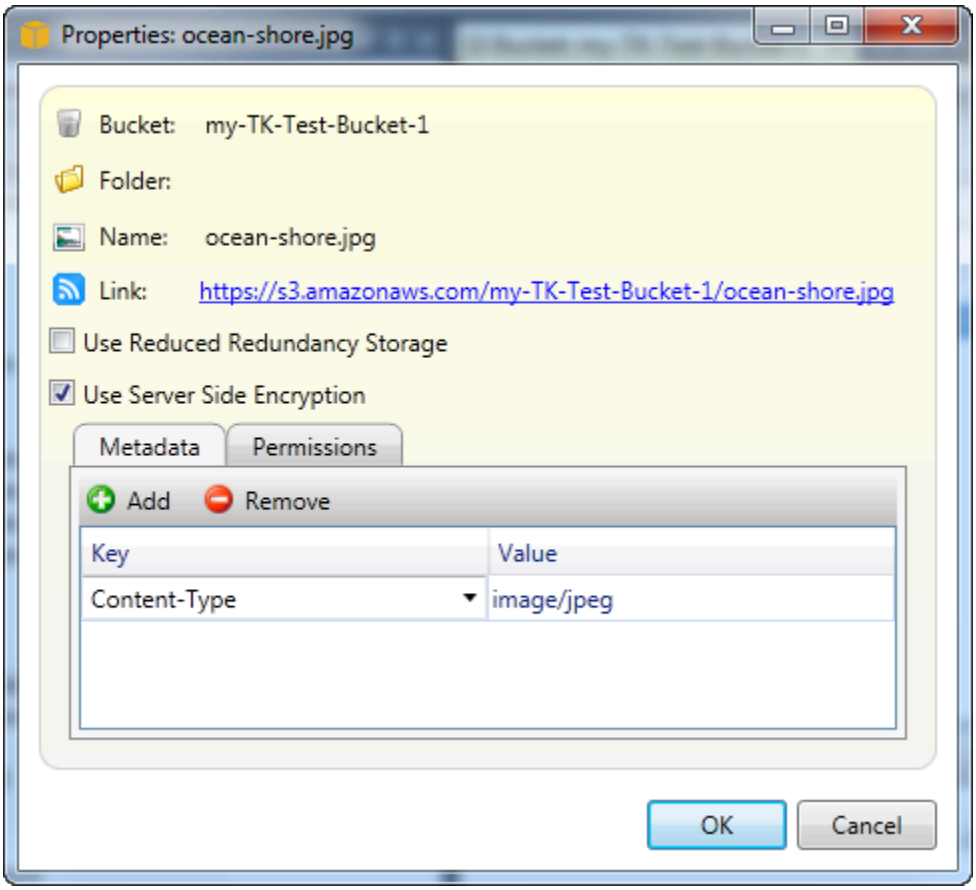
Cut | Copy | Paste

Enables you to cut, copy, and paste files or folders between folders or between buckets.

Properties

Displays a dialog box that enables you to set metadata and permissions for the file, as well as toggle storage for the file between Reduced Redundancy Storage (RRS) and Standard, and set server-side encryption for the file. This dialog box also displays an https link to the file. If

you choose this link, the Toolkit for Visual Studio opens the file in your default browser. If you have permissions on the file set to **Open/Download** and **Everyone**, other people will be able to access the file through this link. Rather than distributing this link, we recommend you create and distribute pre-signed URLs.



Create Pre-Signed URL

Enables you to create a time-limited pre-signed URL that you can distribute to enable other people to access the content you have stored on Amazon S3.

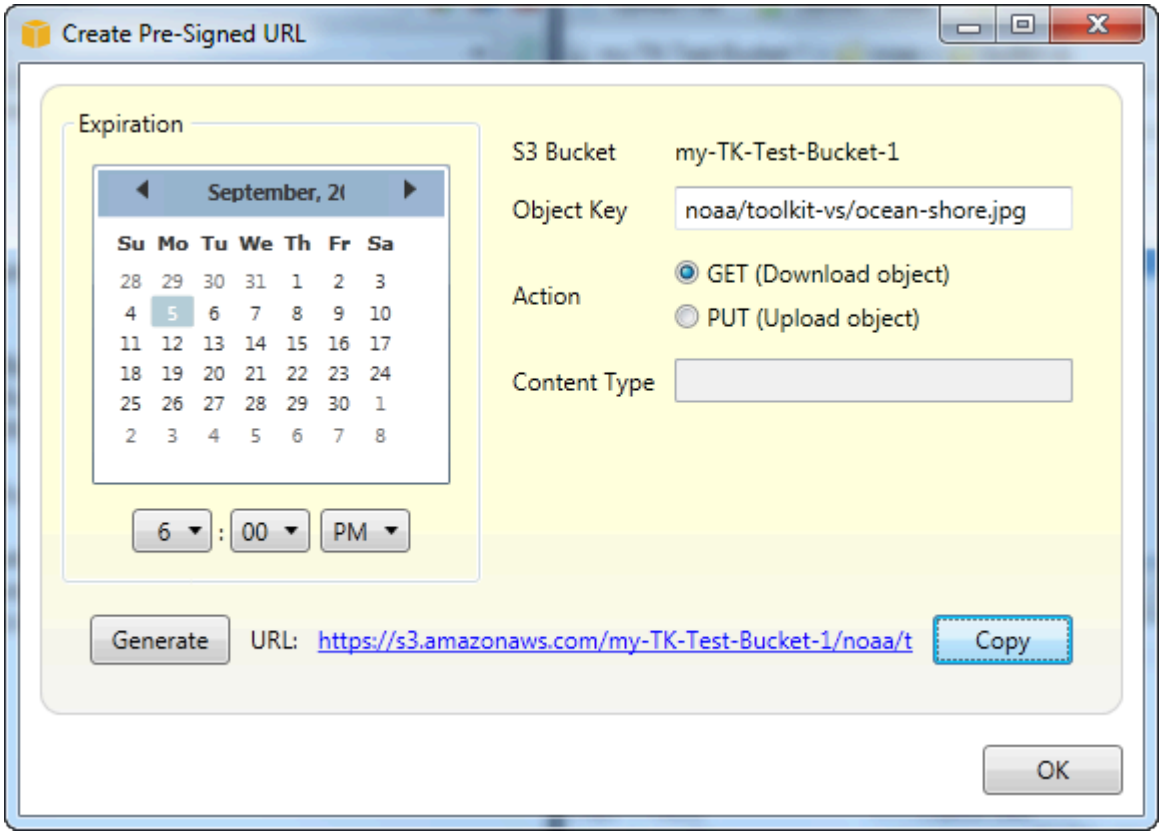
How to Create a Pre-Signed URL

You can create a pre-signed URL for a bucket or files in a bucket. Other people can then use this URL to access the bucket or file. The URL will expire after a period of time that you specify when you create the URL.

To create a pre-signed URL

1. In the **Create Pre-Signed URL** dialog box, set the expiration date and time for the URL. The default setting is one hour from the current time.

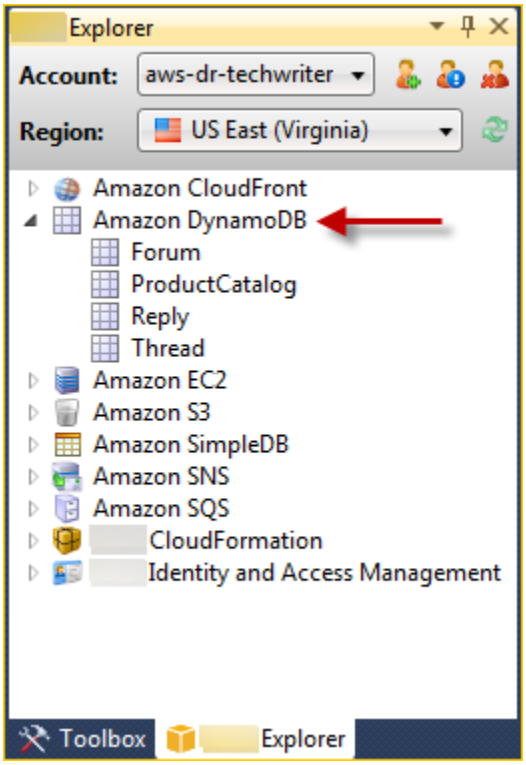
2. Choose the **Generate** button.
3. To copy the URL to the clipboard, choose **Copy**.



Using DynamoDB from Amazon Explorer

Amazon DynamoDB is a fast, highly scalable, highly available, cost-effective, non-relational database service. DynamoDB removes traditional scalability limitations on data storage while maintaining low latency and predictable performance. The Toolkit for Visual Studio provides functionality for working with DynamoDB in a development context. For more information about DynamoDB, see DynamoDB on the Amazon Web Services website.

In the Toolkit for Visual Studio, Amazon Explorer displays all of the DynamoDB tables associated with the active Amazon Web Services account.



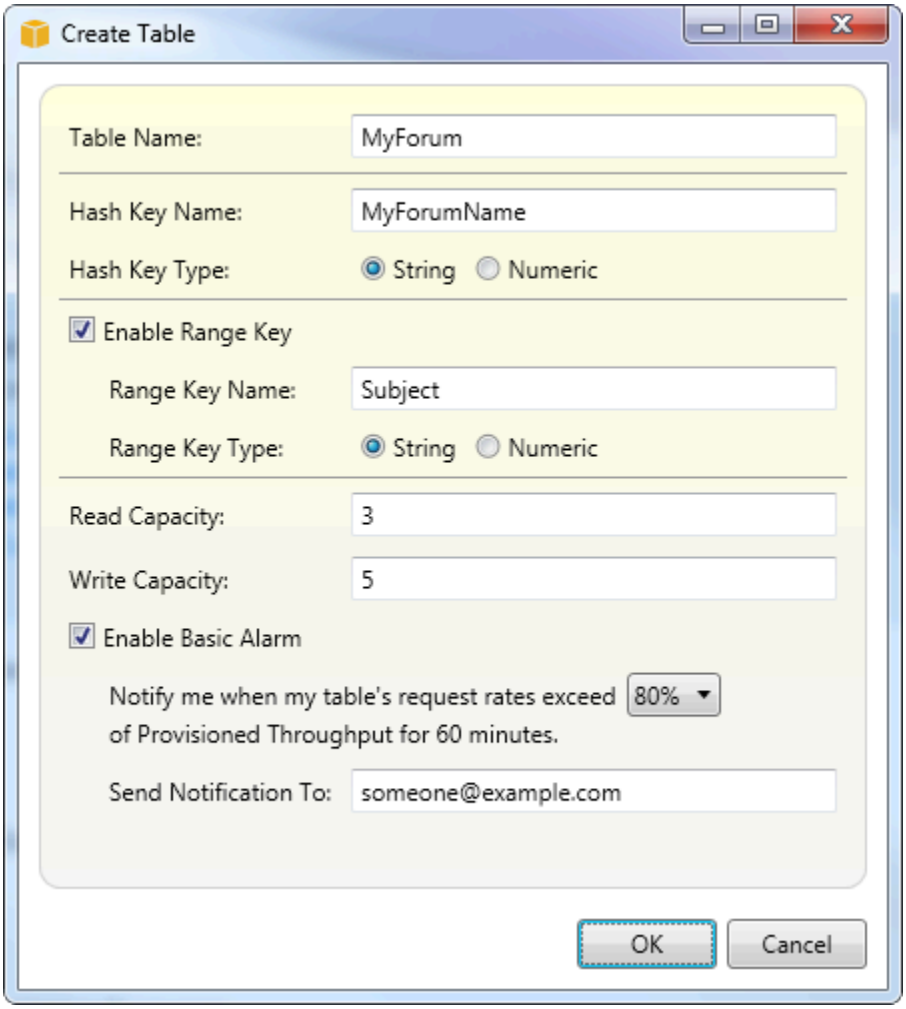
Creating an DynamoDB Table

You can use the Toolkit for Visual Studio to create a DynamoDB table.

To create a table in Amazon Explorer

1. In Amazon Explorer, open the context (right-click) menu for **Amazon DynamoDB**, and then choose **Create Table**.
2. In the **Create Table** wizard, in **Table Name**, type a name for the table.
3. In the **Hash Key Name** field, type a primary hash key attribute and from the **Hash Key Type** buttons, choose the hash key type. DynamoDB builds an unordered hash index using the primary key attribute and an optional sorted range index using the range primary key attribute. For more information about the primary hash key attribute, go to the Primary Key section in the *Amazon DynamoDB Developer Guide*.
4. (Optional) Select **Enable Range Key**. In the **Range Key Name** field, type a range key attribute, and then from the **Range Key Type** buttons, choose a range key type.
5. In the **Read Capacity** field, type the number of read capacity units. In the **Write Capacity** field, type the number of write capacity units. You must specify a minimum of three read capacity units and five write capacity units. For more information about read and write capacity units, go to Provisioned Throughput in DynamoDB.

6. (Optional) Select **Enable Basic Alarm** to alert you when your table's request rates are too high. Choose the percentage of provisioned throughput per 60 minutes that must be exceeded before the alert is sent. **In Send Notifications To**, type an email address.
7. Click **OK** to create the table.

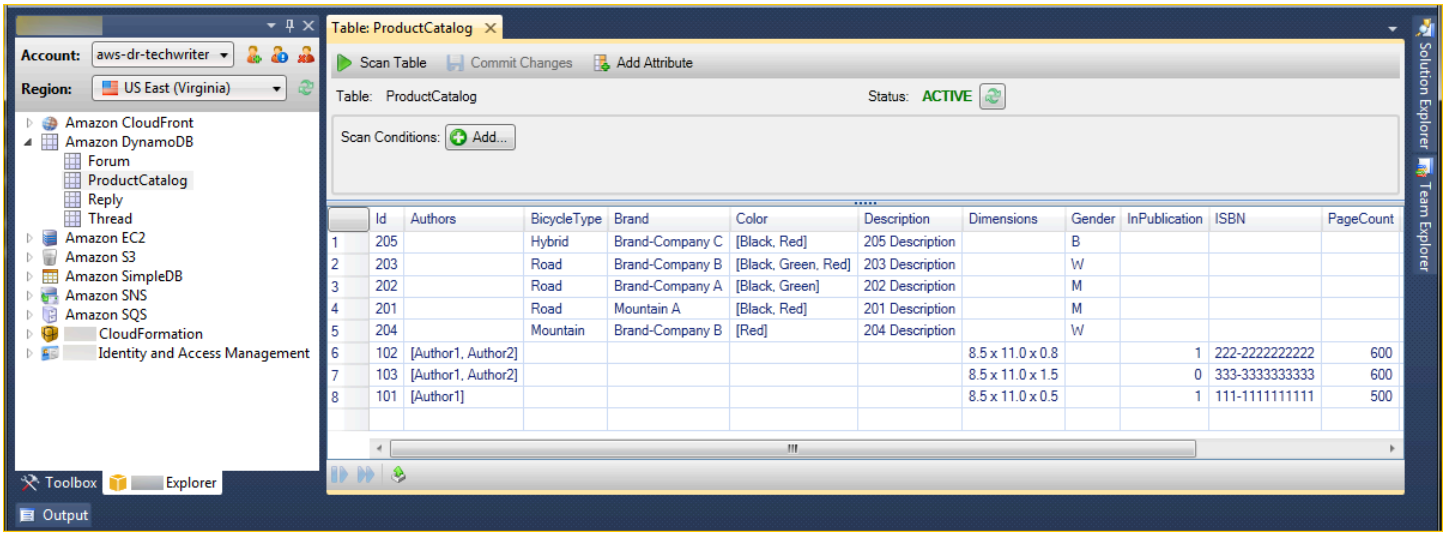


For more information about DynamoDB tables, go to Data Model Concepts - Tables, Items, and Attributes.

Viewing an DynamoDB Table as a Grid

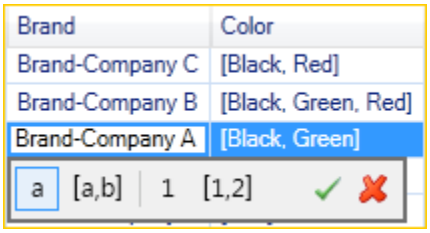
To open a grid view of one of your DynamoDB tables, in Amazon Explorer, double-click the subnode that corresponds to the table. From the grid view, you can view the items, attributes, and values stored in the table. Each row corresponds to an item in the table. The table columns correspond to attributes. Each cell of the table holds the values associated with that attribute for that item.

An attribute can have a value that is a string or a number. Some attributes have a value that consists of a *set* of strings or numbers. Set values are displayed as a comma-separated list enclosed by square brackets.

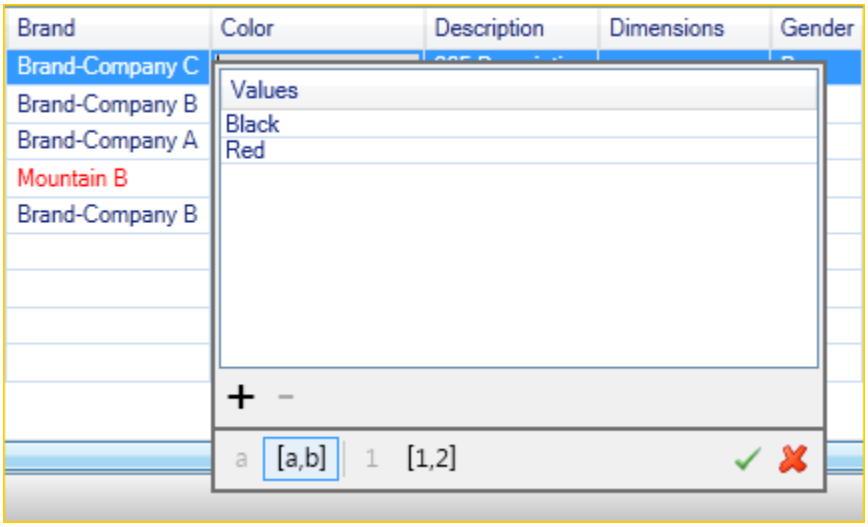


Editing and Adding Attributes and Values

By double-clicking a cell, you can edit the values for the item's corresponding attribute. For set-value attributes, you can also add or delete individual values from the set.



In addition to changing the value of an attribute, you can also, with some limitations, change the format of the value for an attribute. For example, any number value can be converted into a string value. If you have a string value, the content of which is a number, such as 125, the cell editor enables you to convert the format of the value from string to number. You can also convert a single-value to a set-value. However, you cannot generally convert from a set-value to a single-value; an exception is when the set-value has, in fact, only one element in the set.

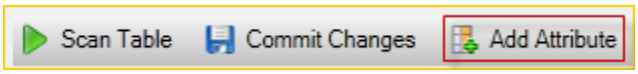


After editing the attribute value, choose the green check mark to confirm your changes. If you want to discard your changes, choose the red X.

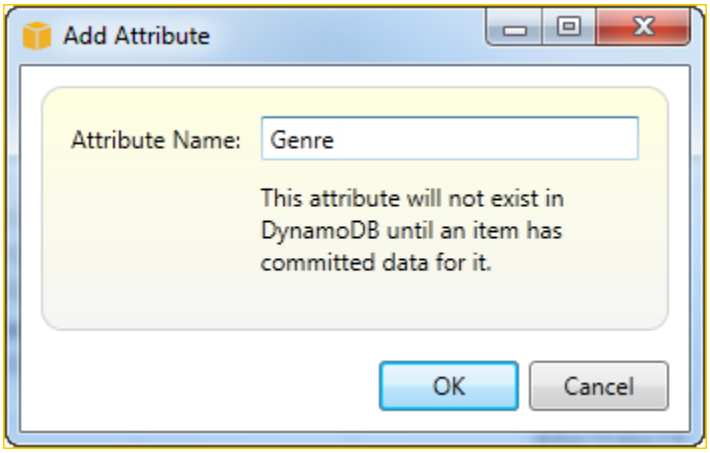
After you have confirmed your changes, the attribute value will be displayed in red. This indicates the attribute has been updated, but that the new value has not been written back to the DynamoDB database. To write your changes back to DynamoDB, choose **Commit Changes**. To discard your changes, choose **Scan Table** and when the Toolkit asks if you would like to commit your changes before the Scan, choose **No**.

Adding an Attribute

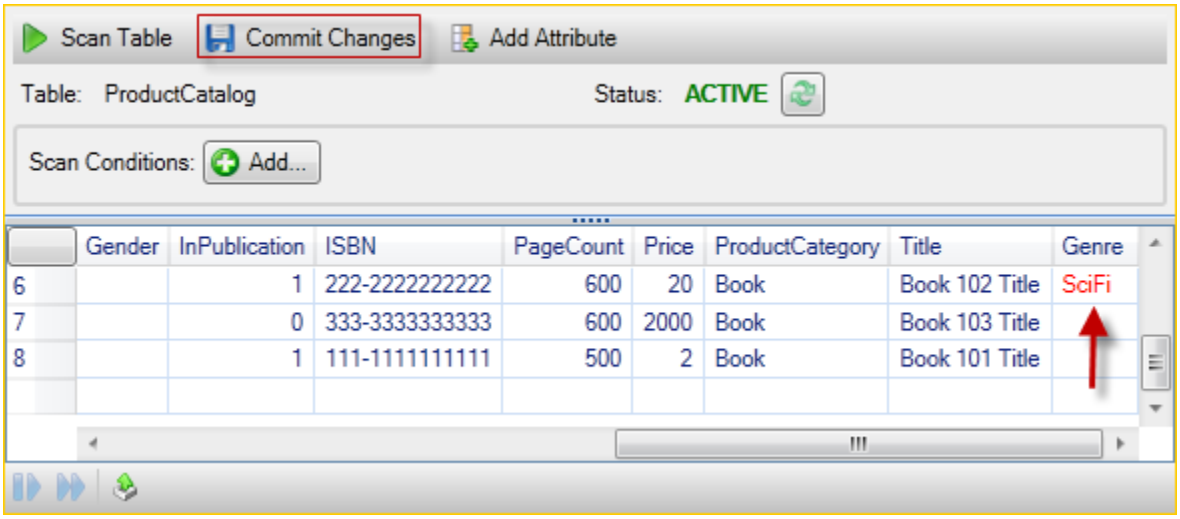
From the grid view, you can also add attributes to the table. To add a new attribute, choose **Add Attribute**.



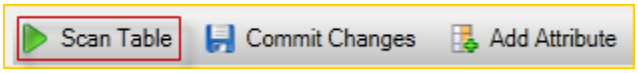
In the **Add Attribute** dialog box, type a name for your attribute, and then choose **OK**.



To make the new attribute become part of the table, you must add a value to it for at least one item and then choose the **Commit Changes** button. To discard the new attribute, just close the grid view of the table without choosing **Commit Changes**.



Scanning an DynamoDB Table

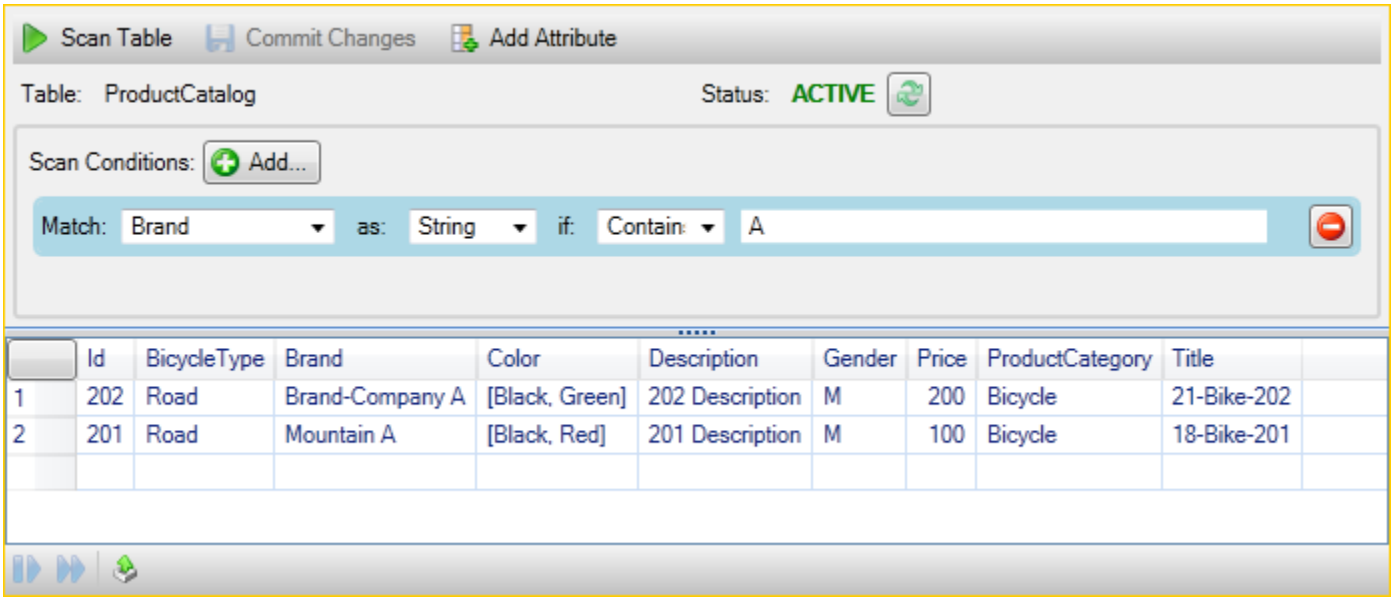


You can perform Scans on your DynamoDB tables from the Toolkit. In a Scan, you define a set of criteria and the Scan returns all items from the table that match your criteria. Scans are expensive operations and should be used with care to avoid disrupting higher priority production traffic on the table. For more information about using the Scan operation, go to the *Amazon DynamoDB Developer Guide*.

To perform a Scan on an DynamoDB table from Amazon Explorer

1. In the grid view, choose the **scan conditions: add** button.
2. In the Scan clause editor, choose the attribute to match against, how the value of the attribute should be interpreted (string, number, set value), how it should be matched (for example Begins With or Contains), and the literal value it should match.
3. Add more Scan clauses, as needed, for your search. The Scan will return only those items that match the criteria from all of your Scan clauses. The Scan will perform a case-sensitive comparison when matching against string values.
4. On the button bar at the top of the grid view, choose **Scan Table**.

To remove a Scan clause, choose the red button with the white line to the right of each clause.



To return to the view of the table that includes all items, remove all Scan clauses and choose **Scan Table** again.

Paginating Scan Results

At the bottom of the view are three buttons.

The first two blue buttons provide pagination for Scan results. The first button will display an additional page of results. The second button will display an additional ten pages of results. In this context, a page is equal to 1 MB of content.

Export Scan Result to CSV

The third button exports the results from the current Scan to a CSV file.

Using Amazon CodeCommit with Visual Studio Team Explorer

You can use Amazon Identity and Access Management (IAM) user accounts to create Git credentials and use them to create and clone repositories from within Team Explorer.

Credential Types for Amazon CodeCommit

Most Amazon Toolkit for Visual Studio users are aware of setting up Amazon credential profiles that contain their access and secret keys. These credential profiles are used in the Toolkit for

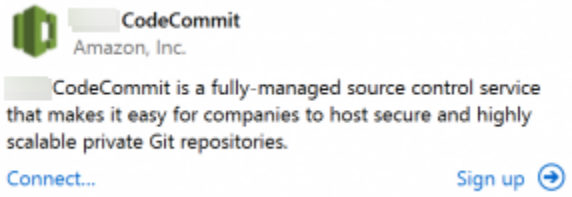
Visual Studio to enable the calls to service APIs, for example, to list Amazon S3 buckets in Amazon Explorer or to launch an Amazon EC2 instance. The integration of Amazon CodeCommit with Team Explorer also uses these credential profiles. However, to work with Git itself you need additional credentials, specifically, Git credentials for HTTPS connections. You can read about these credentials (a user name and password) at Setup for HTTPS Users Using Git Credentials in the *Amazon CodeCommit User Guide*.

You can create the Git credentials for Amazon CodeCommit only for IAM user accounts. You cannot create them for a root account. You can create up to two sets of these credentials for the service and, although you can mark a set of credentials as inactive, inactive sets still count toward your limit of two sets. Note that you can delete and recreate credentials at any time. When you use Amazon CodeCommit from within Visual Studio, your traditional Amazon credentials are used for working with the service itself, for example, when you're creating and listing repositories. When working with the actual Git repositories hosted in Amazon CodeCommit, you use the Git credentials.

As part of the support for Amazon CodeCommit, the Toolkit for Visual Studio automatically creates and manages these Git credentials for you and associates them with your Amazon credential profile. You don't need to be concerned about having the right set of credentials at hand to perform Git operations within Team Explorer. Once you connect to Team Explorer with your Amazon credential profile, the associated Git credentials are used automatically whenever you work with a Git remote.

Connecting to Amazon CodeCommit

When you open the Team Explorer window in Visual Studio 2015 or later, you'll see an Amazon CodeCommit entry in the Hosted Service Providers section of Manage Connections.

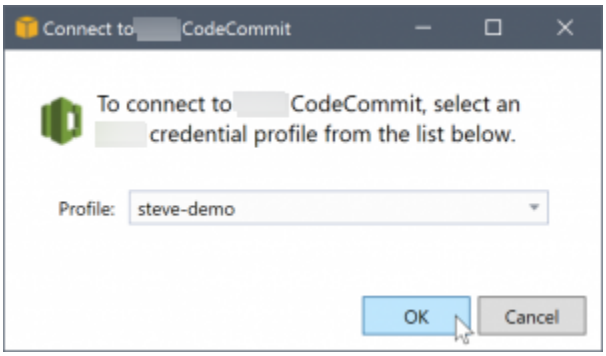


Choosing **Sign up** opens the Amazon Web Services home page in a browser window. What happens when you choose **Connect** depends on whether the Toolkit for Visual Studio can find a credential profile with Amazon access and secret keys to enable it to make calls to Amazon on your behalf. You might have set up a credential profile by using the new Getting Started page that displays in the IDE when the Toolkit for Visual Studio cannot find any locally stored credentials. Or you might

have been using the Toolkit for Visual Studio, the Amazon Tools for Windows PowerShell, or the Amazon CLI and already have Amazon credential profiles available for the Toolkit for Visual Studio to use.

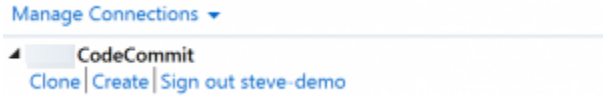
When you choose **Connect**, the Toolkit for Visual Studio starts the process to find a credential profile to use in the connection. If the Toolkit for Visual Studio can't find a credential profile, it opens a dialog box that invites you to enter the access and secret keys for your Amazon Web Services account. We strongly recommend that you use an IAM user account, and not your root credentials. In addition, as noted earlier, the Git credentials you eventually need can only be created for IAM users. Once the access and secret keys are provided and the credential profile is created, the connection between Team Explorer and Amazon CodeCommit is ready for use.

If the Toolkit for Visual Studio finds more than one Amazon credential profile, you're prompted to select the account you want to use within Team Explorer.



If you have only one credential profile, the Toolkit for Visual Studio bypasses the profile selection dialog box and you're connected immediately:

When a connection is established between Team Explorer and Amazon CodeCommit via your credential profiles, the invitation dialog box closes and the connection panel is displayed.

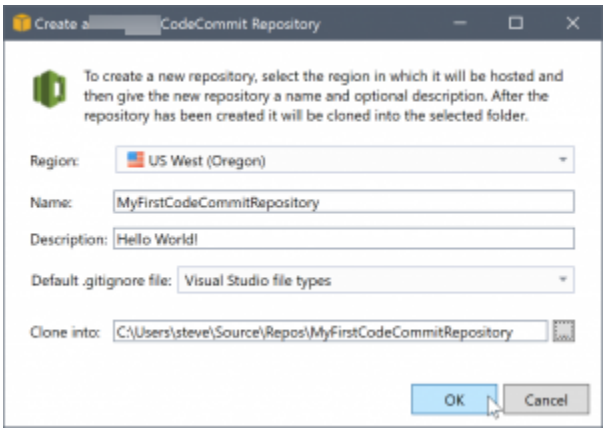


Because you have no repositories cloned locally, the panel shows just the operations you can perform: **Clone**, **Create**, and **Sign out**. Like other providers, Amazon CodeCommit in Team Explorer can be bound to only a single Amazon credential profile at any given time. To switch accounts, you use **Sign out** to remove the connection so you can start a new connection using a different account.

Now that you have established a connection, you can create a repository by clicking the **Create** link.

Creating a Repository

When you click the **Create** link, the **Create a New Amazon CodeCommit Repository** dialog box opens.



Amazon CodeCommit repositories are organized by region, so in **Region** you can select the region in which to host the repository. The list has all the regions in which Amazon CodeCommit is supported. You provide the Name (required) and Description (optional) for our new repository.

The default behavior of the dialog box is to suffix the folder location for the new repository with the repository name (as you enter the name, the folder location also updates). To use a different folder name, edit the **Clone into** folder path after you finish entering the repository name.

You can also choose to automatically create an initial `.gitignore` file for the repository. The Amazon Toolkit for Visual Studio provides a built-in default for Visual Studio file types. You can also choose to have no file or to use a custom existing file that you would like to reuse across repositories. Simply select **Use custom** in the list and navigate to the custom file to use.

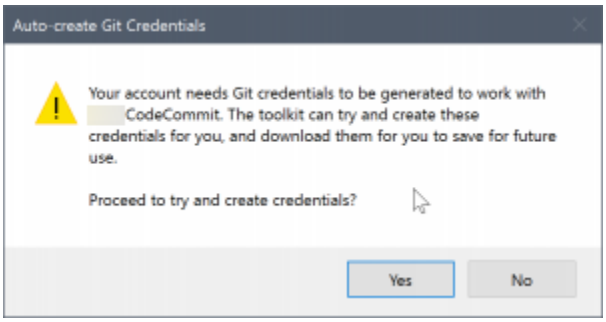
Once you have a repository name and location, you are ready to click **OK** and start creating the repository. The Toolkit for Visual Studio requests that the service create the repository and then clone the new repository locally, adding an initial commit for the .gitignore file, if you're using one. It's at this point that you start working with the Git remote, so the Toolkit for Visual Studio now needs access to the Git credentials described earlier.

Setting up Git Credentials

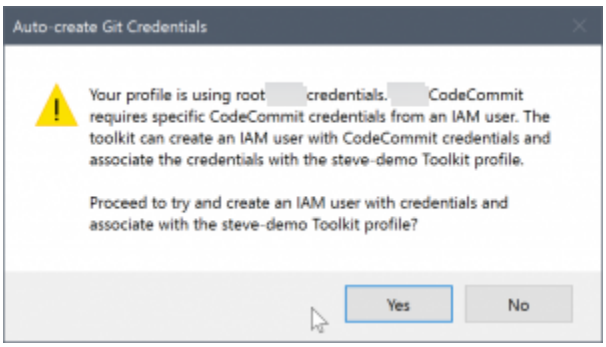
To this point you've been using Amazon access and secret keys to request that the service create your repository. Now you need to work with Git itself to do the actual clone operation, and Git doesn't understand Amazon access and secret keys. Instead, you need to supply the user name and password credentials to Git to use on an HTTPS connection with the remote.

As noted in Setting up Git credentials, the Git credentials you're going to use must be associated with an IAM user. You cannot generate them for root credentials. You should always set up your Amazon credential profiles to contain IAM user access and secret keys, and not root keys. The Toolkit for Visual Studio can attempt to set up Git credentials for Amazon CodeCommit for you, and associate them with the Amazon credential profile that you used to connect in Team Explorer earlier.

When you choose **OK** in the **Create a New Amazon CodeCommit Repository** dialog box and successfully create the repository, the Toolkit for Visual Studio checks the Amazon credential profile that is connected in Team Explorer to determine if Git credentials for Amazon CodeCommit exist and are associated locally with the profile. If so, the Toolkit for Visual Studio instructs Team Explorer to commence the clone operation on the new repository. If Git credentials are not available locally, the Toolkit for Visual Studio checks the type of account credentials that were used in the connection in Team Explorer. If the credentials are for an IAM user, as we recommend, the following message is shown.



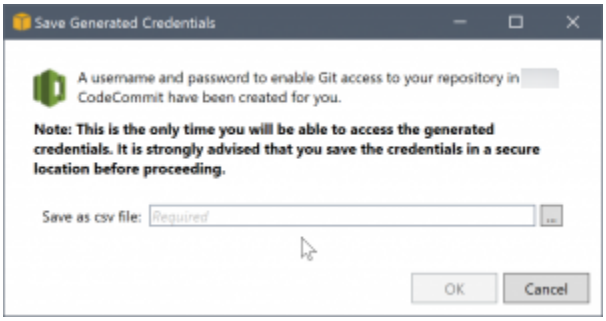
If the credentials are root credentials, the following message is shown instead.



In both cases, the Toolkit for Visual Studio offers to attempt to do the work to create the necessary Git credentials for you. In the first scenario, all it needs to create are a set of Git credentials for the IAM user. When a root account is in use, the Toolkit for Visual Studio first attempts to create an IAM user and then proceeds to create Git credentials for that new user. If the Toolkit for Visual Studio has to create a new user, it applies the Amazon CodeCommit Power User managed policy

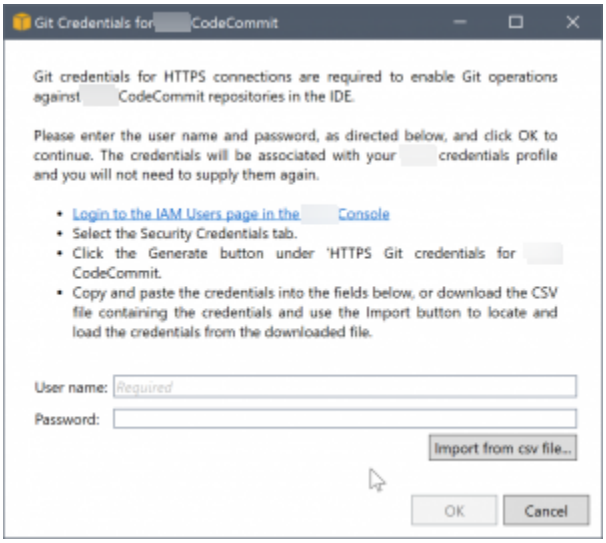
to that new user account. This policy allows access only to Amazon CodeCommit and enables all operations to be performed with Amazon CodeCommit except for repository deletion.

When you're creating credentials, you can only view them once. Therefore, the Toolkit for Visual Studio prompts you to save the newly created credentials as a `.csv` file before continuing.



This is something we also strongly recommend, and be sure to save them to a secure location!

There might be cases where the Toolkit for Visual Studio can't automatically create credentials. For example, you may already have created the maximum number of sets of Git credentials for Amazon CodeCommit (two), or you might not have sufficient programmatic rights for the Toolkit for Visual Studio to do the work for you (if you're signed in as an IAM user). In these cases, you can log into the Amazon Web Services Management Console to manage the credentials or obtain them from your administrator. You can then enter them in the **Git Credentials for Amazon CodeCommit** dialog box, which the Toolkit for Visual Studio displays.

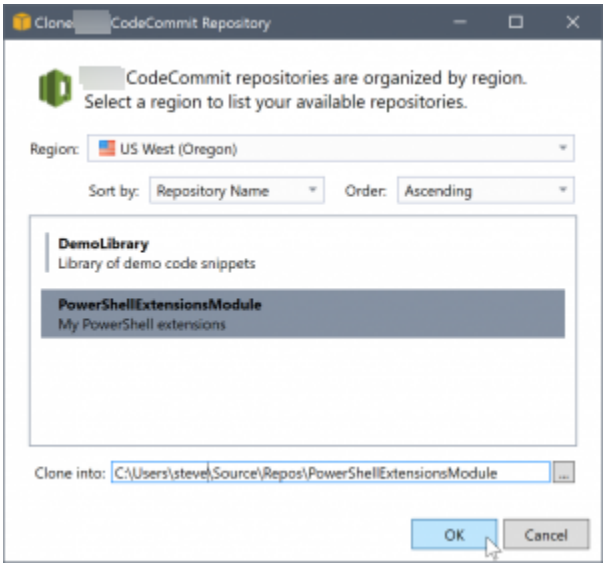


Now that the credentials for Git are available, the clone operation for the new repository proceeds (see progress indication for the operation inside Team Explorer). If you elected to have a default `.gitignore` file applied, it is committed to the repository with a comment of 'Initial Commit'.

That's all there is to setting up credentials and creating a repository within Team Explorer. Once the required credentials are in place, all you see when creating new repositories in the future is the **Create a New Amazon CodeCommit Repository** dialog box itself.

Cloning a Repository

To clone an existing repository, return to the connection panel for Amazon CodeCommit in Team Explorer. Click the **Clone** link to open the **Clone Amazon CodeCommit Repository** dialog box, and then select the repository to clone and the location on disk where you want to place it.



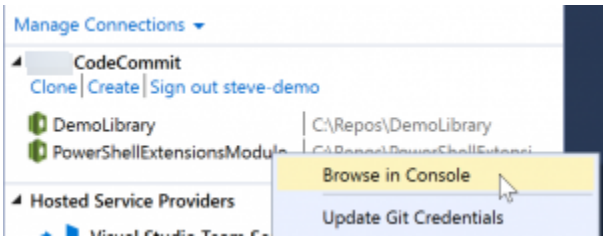
Once you choose the region, the Toolkit for Visual Studio queries the service to discover the repositories that are available in that region and displays them in the central list portion of the dialog box. The name and optional description of each repository are also displayed. You can reorder the list to sort it by either repository name or the last modified date, and to sort each in ascending or descending order.

Once you select the repository you can choose the location to clone to. This defaults to the same repository location used in other plugins to Team Explorer, but you can browse to or enter any other location. By default, the repository name is suffixed onto the selected path. However, if you want a specific path, simply edit the text box after you select the folder. Whatever text is in the box when you click **OK** will be the folder in which you will find the cloned repository.

Having selected the repository and a folder location, you then click **OK** to proceed with the clone operation. Just as with creating a repository, you can see the progress of the clone operation reported in Team Explorer.

Working with Repositories

When you clone or create repositories, notice that the local repositories for the connection are listed in the connection panel in Team Explorer under the operation links. These entries give you a convenient way to access the repository to browse content. Simply right-click the repository and choose **Browse in Console**.



You can also use **Update Git Credentials** to update the stored Git credentials associated with the credential profile. This is useful if you've rotated the credentials. The command opens the **Git Credentials for Amazon CodeCommit** dialog box where you can enter or import the new credentials.

Git operations on the repositories work as you'd expect. You can make local commits and, when you are ready to share, you use the Sync option in Team Explorer. Because the Git credentials are already stored locally and associated with our connected Amazon credential profile, we won't be prompted to supply them again for operations against the Amazon CodeCommit remote.

Using CodeArtifact in Visual Studio

Amazon CodeArtifact is a fully managed artifact repository service that makes it easy for organizations to securely store and share software packages used for application development. You can use CodeArtifact with popular build tools and package managers such as the NuGet and .NET Core CLIs and Visual Studio. You can also configure CodeArtifact to pull packages from an external, public repository such as NuGet.org.

In CodeArtifact, your packages are stored in repositories which are then stored within a domain. The Amazon Toolkit for Visual Studio simplifies the configuration of Visual Studio with your CodeArtifact repositories, making it easy to consume packages in Visual Studio from both CodeArtifact directly and NuGet.org.

Add your CodeArtifact repository as a NuGet package source

To consume packages from your CodeArtifact, you will need to add your repository as a packabe source in the **NuGet Package Manager** in Visual Studio

To add your repository as a package source

1. In Amazon Explorer, navigate to your repository in the **Amazon CodeArtifact** node.
2. Open the context (right-click) menu for the repository you want to add, and then choose **Copy NuGet Source Endpoint**.
3. Navigate to **Package Sources** underneath the **NuGet Package Manager** node in the **Tools > Options** menu.
4. In **Package Sources**, select the plus sign (**+**), edit the name, and paste the NuGet source endpoint URL that you copied earlier in the **Source** field.
5. Select the checkbox next to your newly added package source to enable it.

> **Note**
>
> We recommend adding an external connection to **NuGet.org** to your CodeArtifact and disabling the **nuget.org** package source in Visual Studio. When using an external connection, all of the dependencies pulled from **NuGet.org** are stored in CodeArtifact. If **NuGet.org** goes down for any reason, the packages you need will still be available. For more information about external connections, see Add an external connection in the *Amazon CodeArtifact User Guide*.

6. Choose **OK** to close the menu.

For more information about using CodeArtifact with Visual Studio, see Use CodeArtifact with Visual Studio in the *Amazon CodeArtifact User Guide*.

Amazon RDS from Amazon Explorer

Amazon Relational Database Service (Amazon RDS) is a service that enables you to provision and manage SQL relational database systems in the cloud. Amazon RDS supports three types of database systems:

- MySQL Community Edition

- Oracle Database Enterprise Edition
- Microsoft SQL Server (Express, Standard, or Web Editions)

For more information, see the Amazon RDS User Guide.

A lot of the functionality discussed here is also available through the Amazon Management Console for Amazon RDS.

Topics

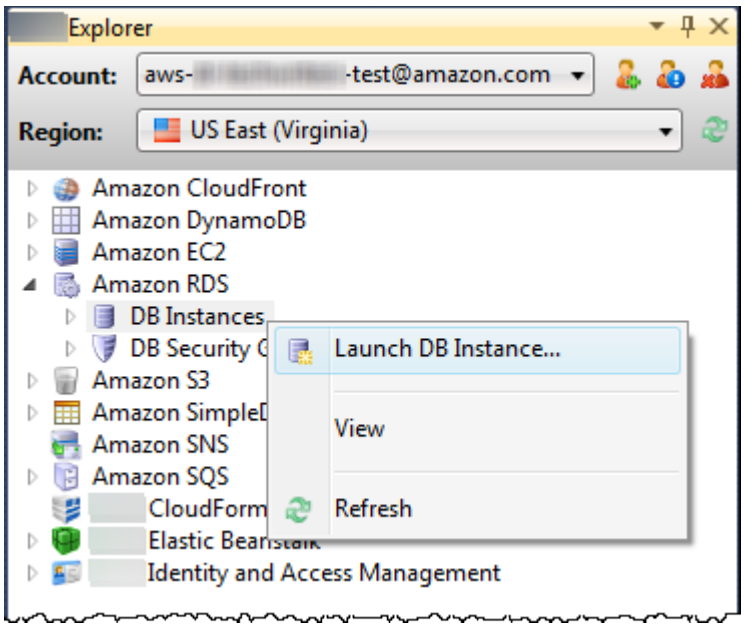
- Launch an Amazon RDS Database Instance
- Create a Microsoft SQL Server Database in an RDS Instance
- Amazon RDS Security Groups

Launch an Amazon RDS Database Instance

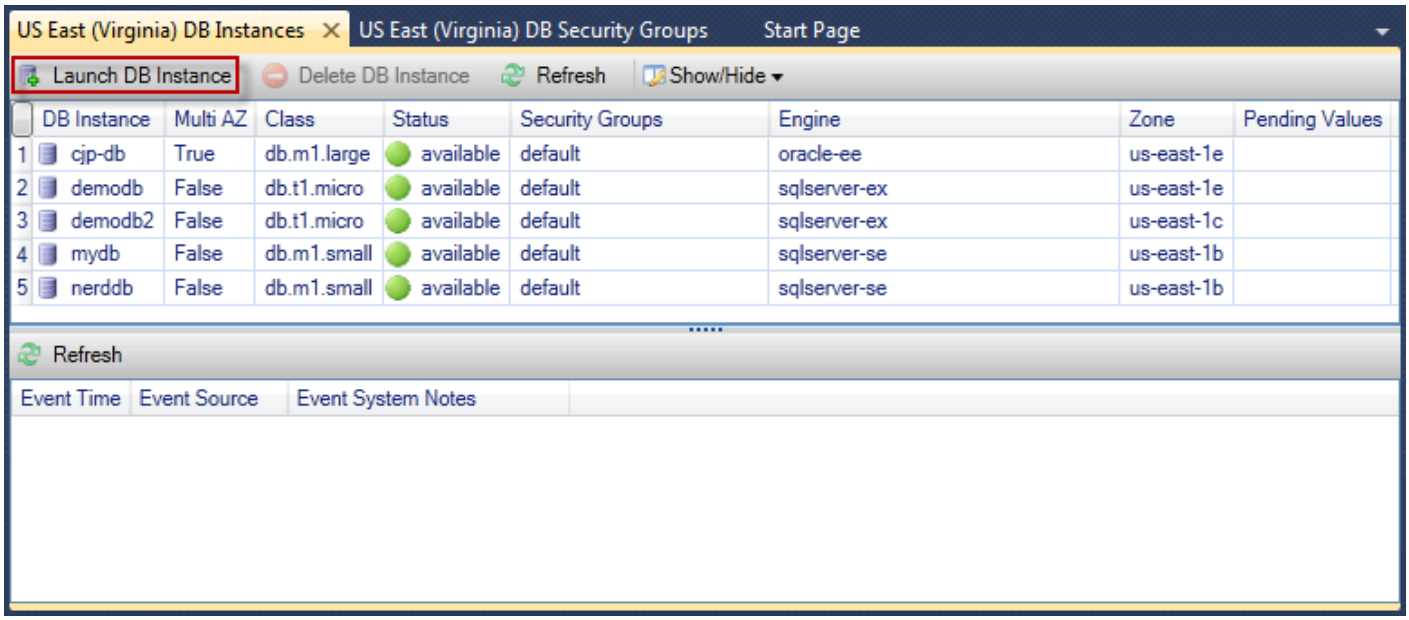
With Amazon Explorer, you can launch an instance of any of the database engines supported by Amazon RDS. The following walkthrough shows the user experience for launching an instance of Microsoft SQL Server Standard Edition, but the user experience is similar for all supported engines.

To launch an Amazon RDS instance

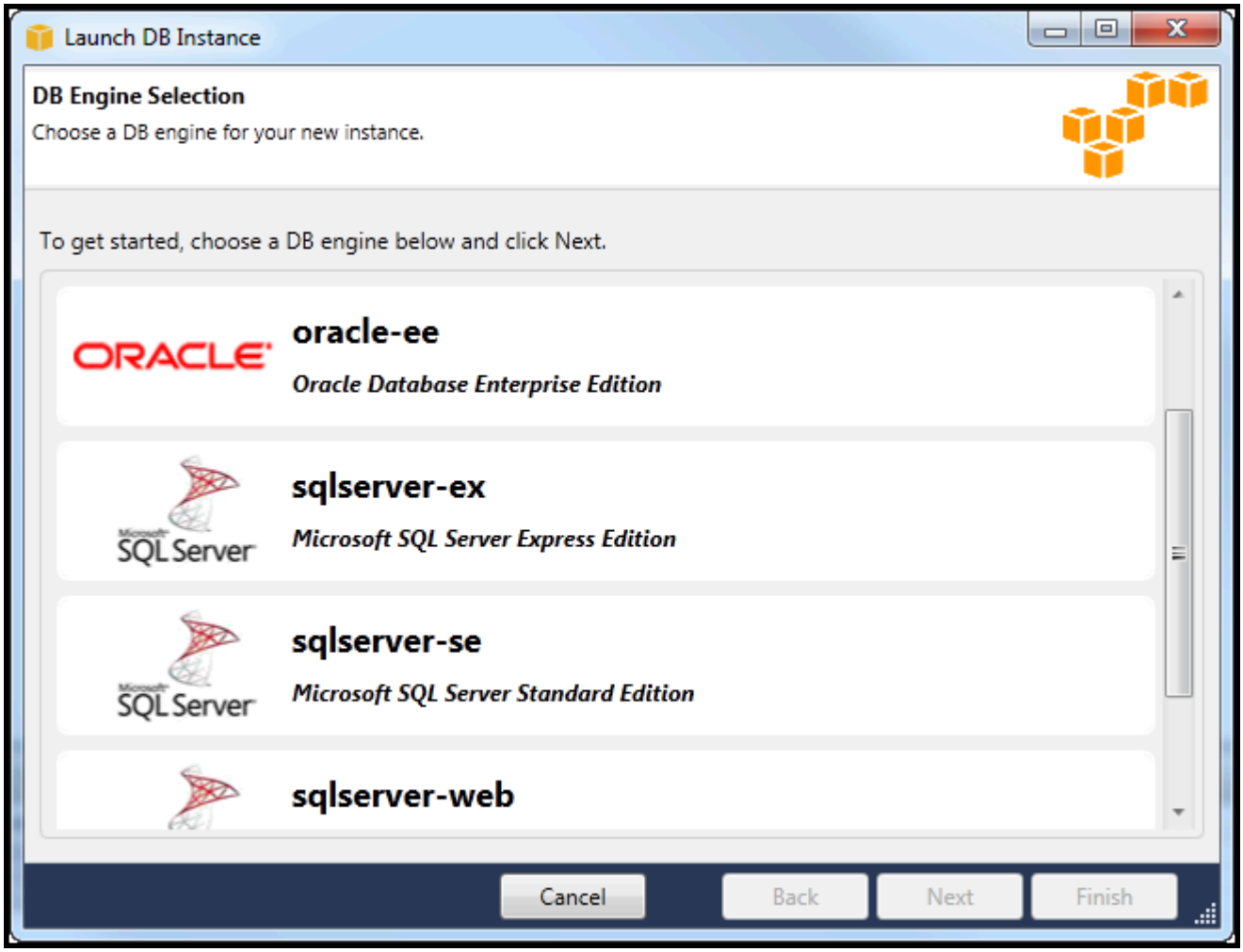
1. In Amazon Explorer, open the context (right-click) menu for the **Amazon RDS** node and choose **Launch DB Instance**.



Alternatively, on the **DB Instances** tab, choose **Launch DB Instance**.



2. In the **DB Engine Selection** dialog box, choose the type of database engine to launch. For this walkthrough, choose Microsoft SQL Server Standard Edition (sqlserver-se), and then choose **Next**.



3. In the **DB Engine Instance Options** dialog box, choose configuration options.

   In the **DB Engine Instance Options and Class** section, you can specify the following settings.

   License Model

Engine Type	License
Microsoft SQL Server	license-included
MySql	general-public-license
Oracle	bring-your-own-license

The license model varies, depending on the type of database engine. Engine Type License Microsoft SQL Server license-included MySql general-public-license Oracle bring-your-own-license

DB Instance Version

Choose the version of the database engine you would like to use. If only one version is supported, it is selected for you.

DB Instance Class

Choose the instance class for the database engine. Pricing for instance classes varies. For more information, see Amazon RDS Pricing.

Perform a multi AZ deployment

Select this option to create a multi-AZ deployment for enhanced data durability and availability. Amazon RDS provisions and maintains a standby copy of your database in a different Availability Zone for automatic failover in the event of a scheduled or unplanned outage. For information about pricing for multi-AZ deployments, see the pricing section of the Amazon RDS detail page. This option is not supported for Microsoft SQL Server.

Upgrade minor versions automatically

Select this option to have Amazon automatically perform minor version updates on your RDS instances for you.

In the **RDS Database Instance** section, you can specify the following settings.

Allocated Storage

Engine	Minimum (GB)	Maximum (GB)
MySQL	5	1024
Oracle Enterprise Edition	10	1024
Microsoft SQL Server Express Edition	30	1024

Engine	Minimum (GB)	Maximum (GB)
Microsoft SQL Server Standard Edition	250	1024
Microsoft SQL Server Web Edition	30	1024

The minimums and maximums for allocated storage depend on the type of database engine. 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

Specify a name for the database instance. This name is not case-sensitive. It will be displayed in lowercase form in Amazon Explorer.

Master User Name

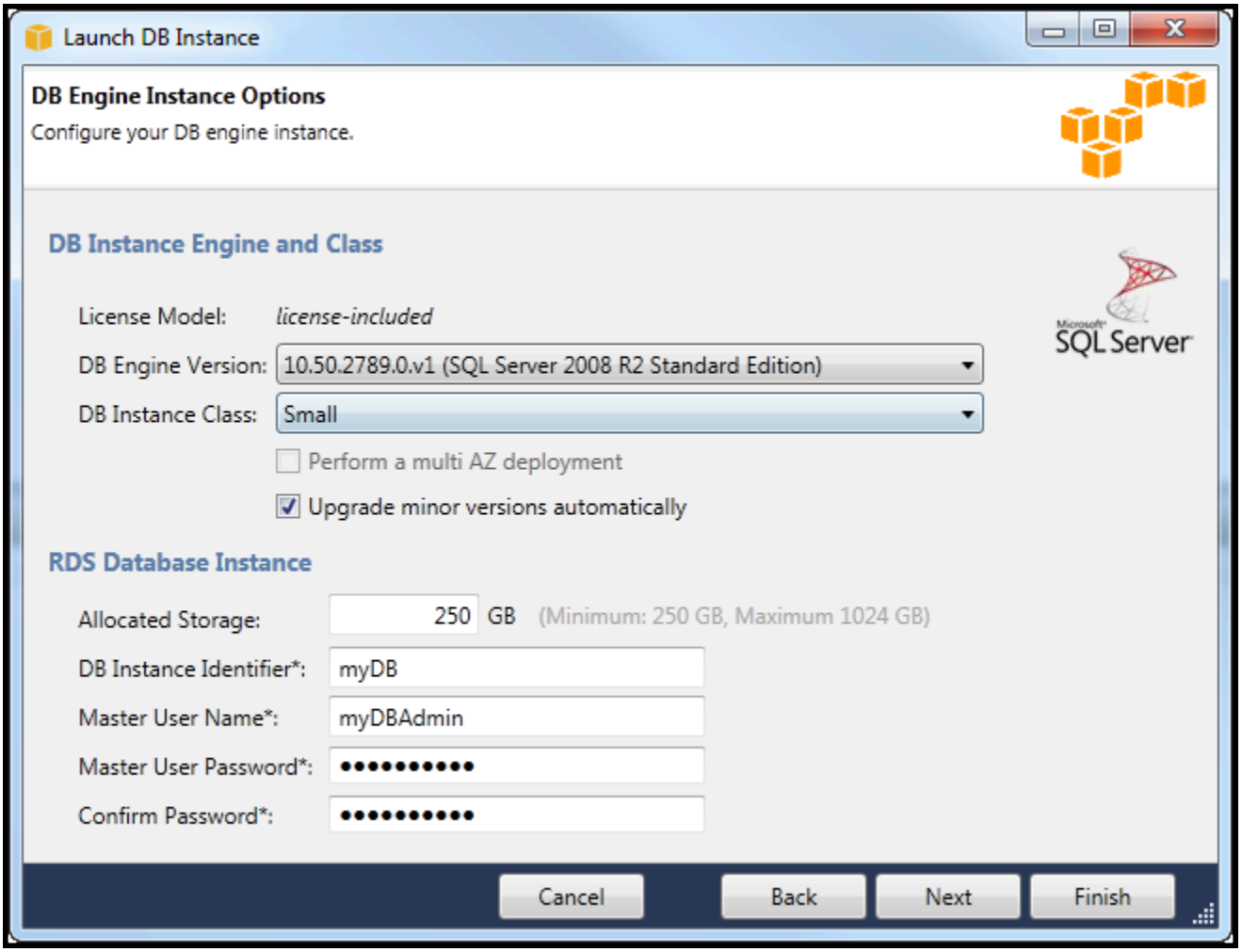
Type a name for the administrator of the database instance.

Master User Password

Type a password for the administrator of the database instance.

Confirm Password

Type the password again to verify it is correct.



1. In the **Additional Options** dialog box, you can specify the following settings.

   Database Port

   This is the TCP port the instance will use to communicate on the network. If your computer accesses the Internet through a firewall, set this value to a port through which your firewall allows traffic.

   Availability Zone

   Use this option if you want the instance to be launched in a particular Availability Zone in your region. The database instance you have specified might not be available in all Availability Zones in a given region.

   RDS Security Group

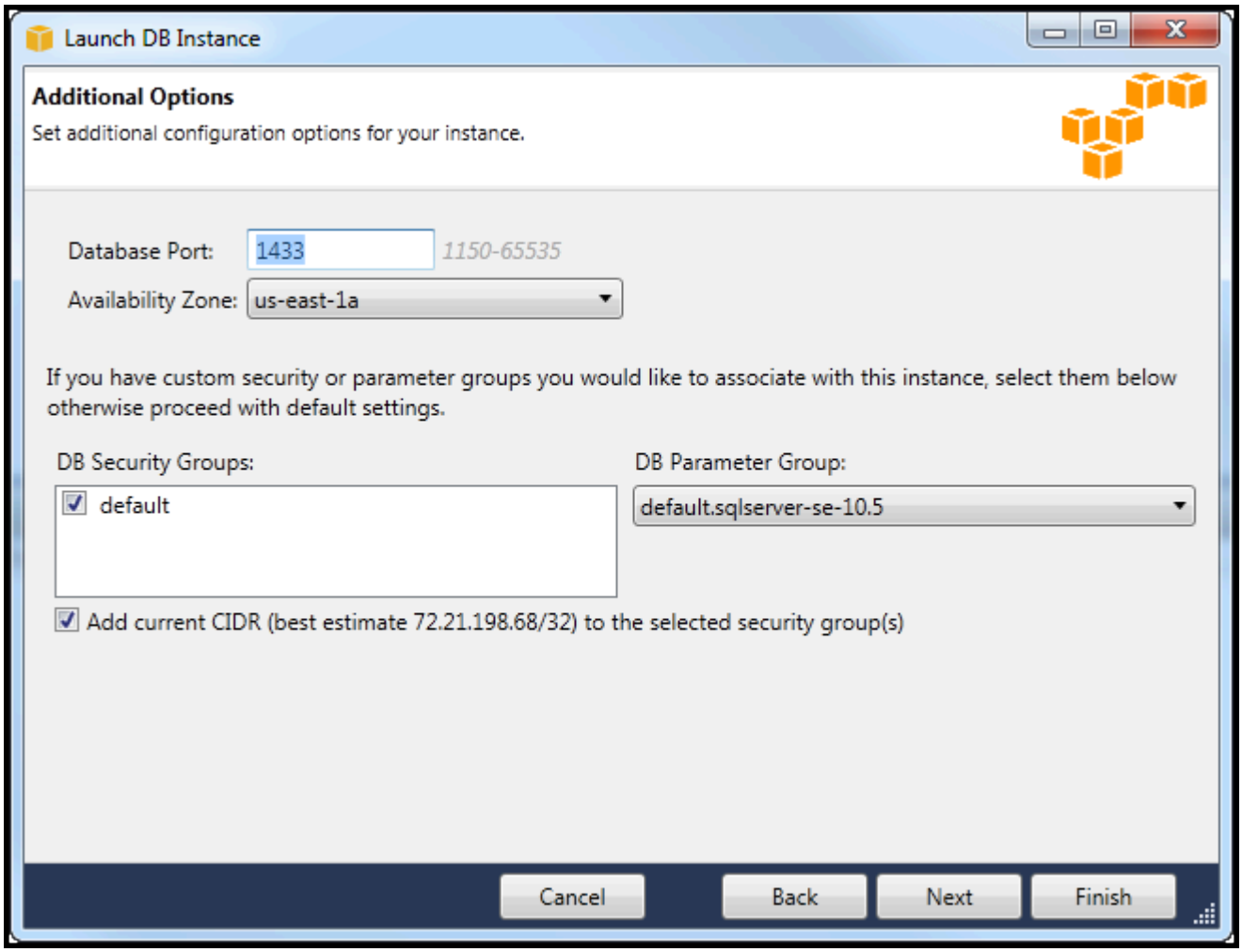
   Select an RDS security group (or groups) to associate with your instance. RDS security groups specify the IP address, Amazon EC2 instances, and Amazon Web Services accounts

that are allowed to access your instance. For more information about RDS security groups, see Amazon RDS Security Groups. The Toolkit for Visual Studio attempts to determine your current IP address and provides the option to add this address to the security groups associated with your instance. However, if your computer accesses the Internet through a firewall, the IP address the Toolkit generates for your computer may not be accurate. To determine which IP address to use, consult your system administrator.

DB Parameter Group

(Optional) From this drop-down list, choose a DB parameter group to associate with your instance. DB parameter groups enable you to change the default configuration for the instance. For more information, go to the Amazon Relational Database Service User Guide and this article.

When you have specified settings on this dialog box, choose **Next**.

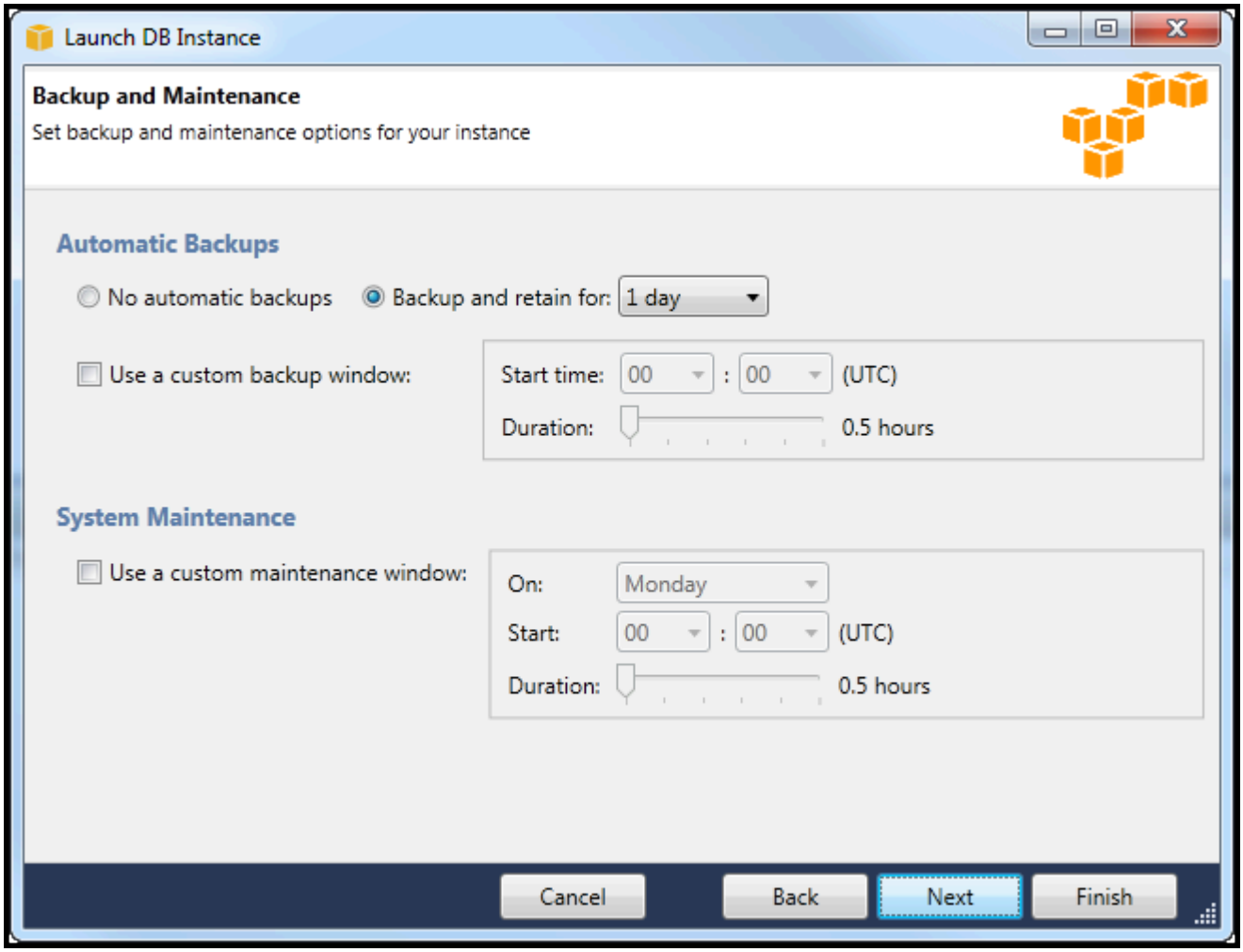


2. The **Backup and Maintenance** dialog box enables you to specify whether Amazon RDS should back up your instance and if so, for how long the backup should be retained. You can also specify a window of time during which the backups should occur.

   This dialog box also enables you to specify if you would like Amazon RDS to perform system maintenance on your instance. Maintenance includes routine patches and minor version upgrades.

   The window of time you specify for system maintenance cannot overlap with the window specified for backups.

   Choose **Next**.



3. The final dialog box in the wizard allows you to review the settings for your instance. If you need to modify settings, use the **Back** button. If all the settings are correct, choose **Launch**.

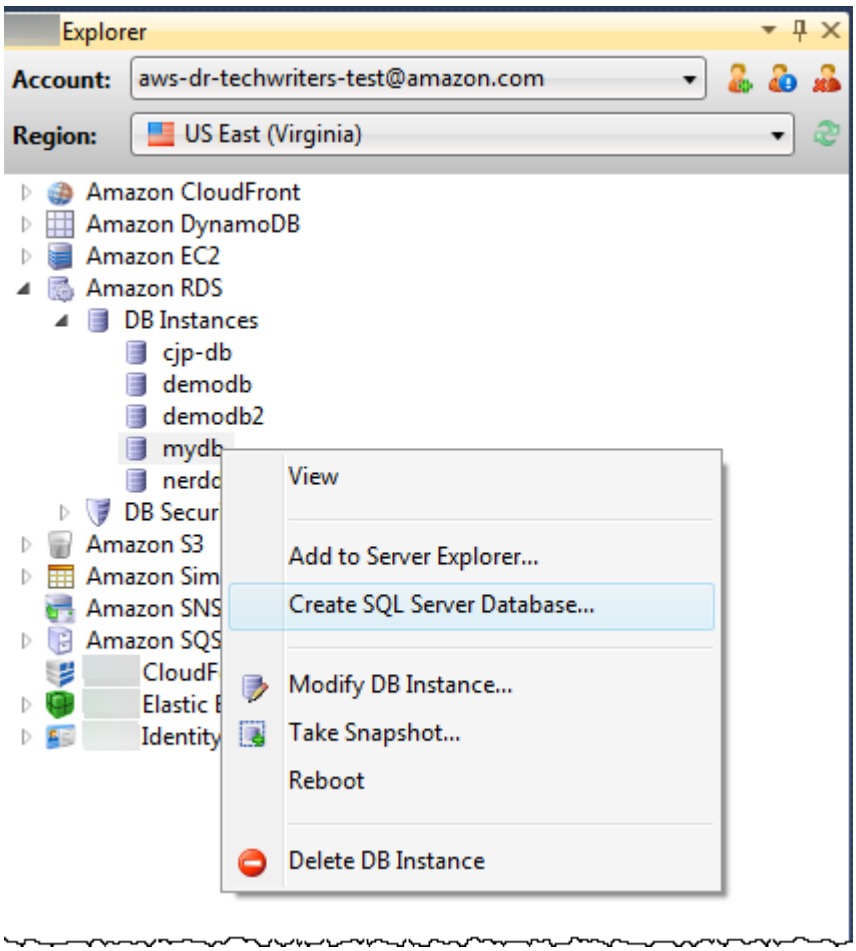
Create a Microsoft SQL Server Database in an RDS Instance

Microsoft SQL Server is designed in such a way that, after launching an Amazon RDS instance, you need to create an SQL Server database in the RDS instance.

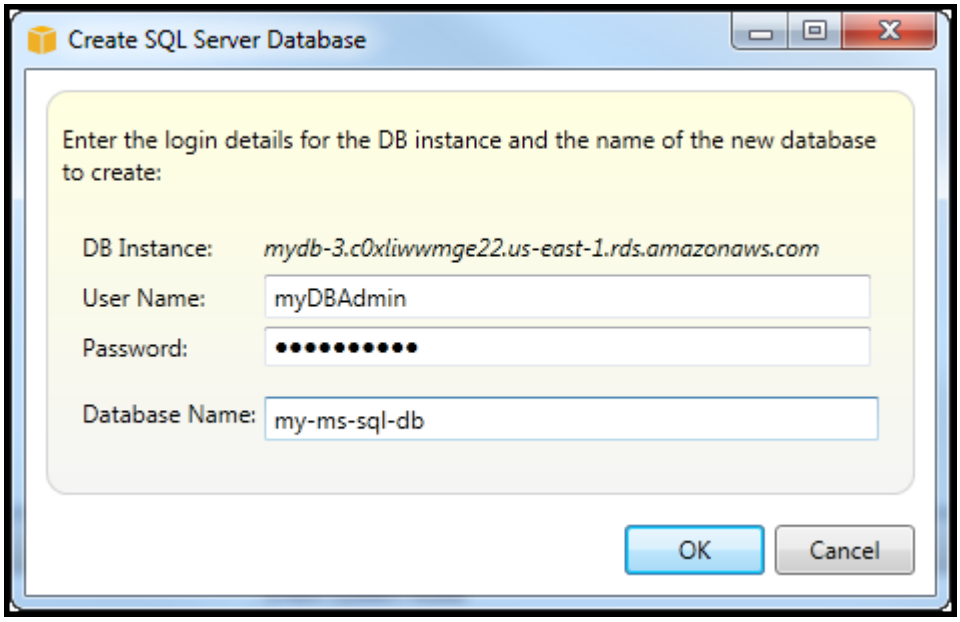
For information about how to create an Amazon RDS instance, see Launch an Amazon RDS Database Instance.

To create a Microsoft SQL Server database

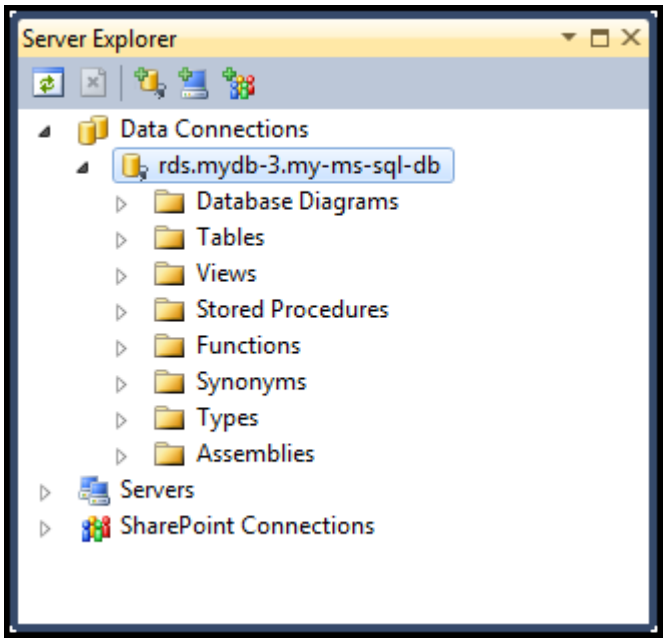
1. In Amazon Explorer, open the context (right-click) menu for the node that corresponds to your RDS instance for Microsoft SQL Server, and choose **Create SQL Server Database**.



2. In the **Create SQL Server Database** dialog box, type the password you specified when you created the RDS instance, type a name for the Microsoft SQL Server database, and then choose **OK**.



3. The Toolkit for Visual Studio creates the Microsoft SQL Server database and adds it to the Visual Studio Server Explorer.



Amazon RDS Security Groups

Amazon RDS security groups enable you to manage network access to your Amazon RDS instances. With security groups, you specify sets of IP addresses using CIDR notation, and only network traffic originating from these addresses is recognized by your Amazon RDS instance.

Although they function in a similar way, Amazon RDS security groups are different from Amazon EC2 security groups. It is possible to add an EC2 security group to your RDS security group. Any EC2 instances that are members of the EC2 security group are then able to access the RDS instances that are members of the RDS security group.

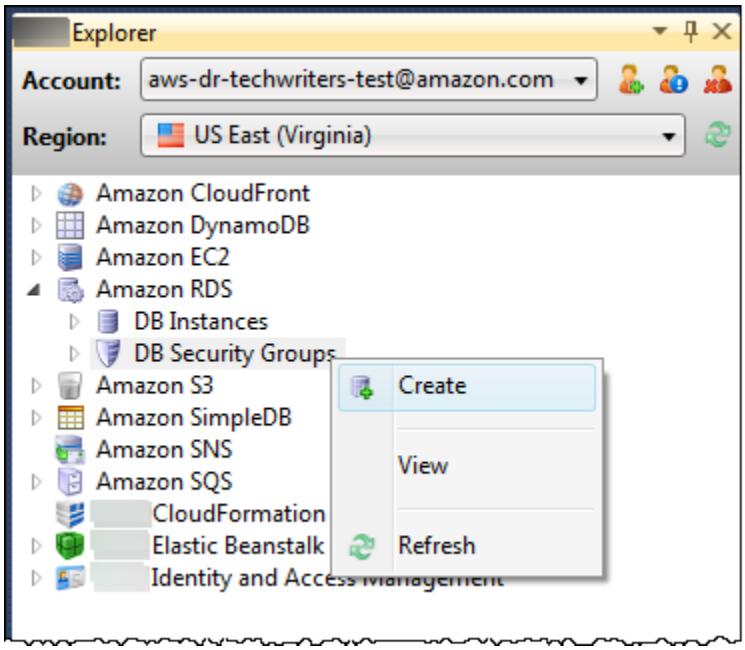
For more information about Amazon RDS security groups, go to the RDS Security Groups. For more information about Amazon EC2 security groups, go to the EC2 User Guide.

Create an Amazon RDS Security Group

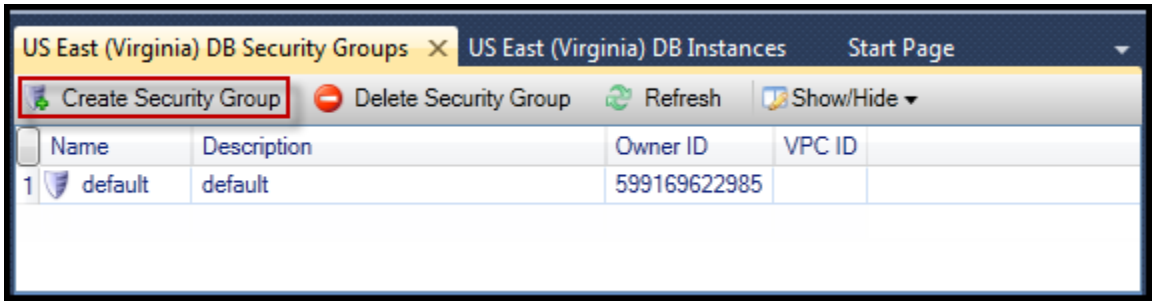
You can use the Toolkit for Visual Studio to create an RDS security group. If you use the Amazon Toolkit to launch an RDS instance, the wizard will allow you to specify an RDS security group to use with your instance. You can use the following procedure to create that security group before you start the wizard.

To create an Amazon RDS security group

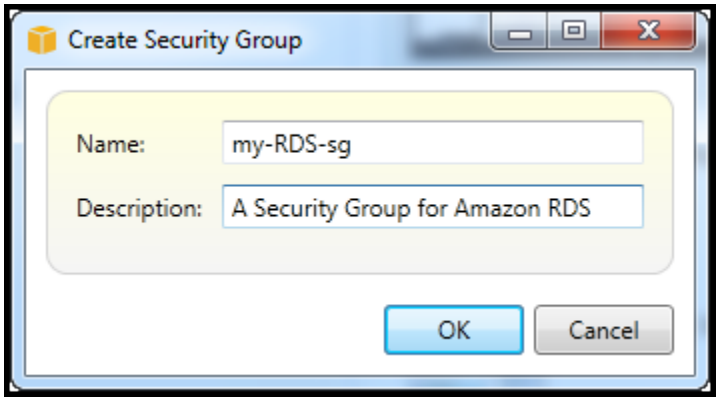
1. In Amazon Explorer, expand the **Amazon RDS** node, open the context (right-click) menu for the **DB Security Groups** subnode and choose **Create**.



   Alternatively, on the **Security Groups** tab, choose **Create Security Group**. If this tab isn't displayed, open the context (right-click) menu for the **DB Security Groups** subnode and choose **View**.



2. In the **Create Security Group** dialog box, type a name and description for the security group, and then choose **OK**.



Set Access Permissions for an Amazon RDS Security Group

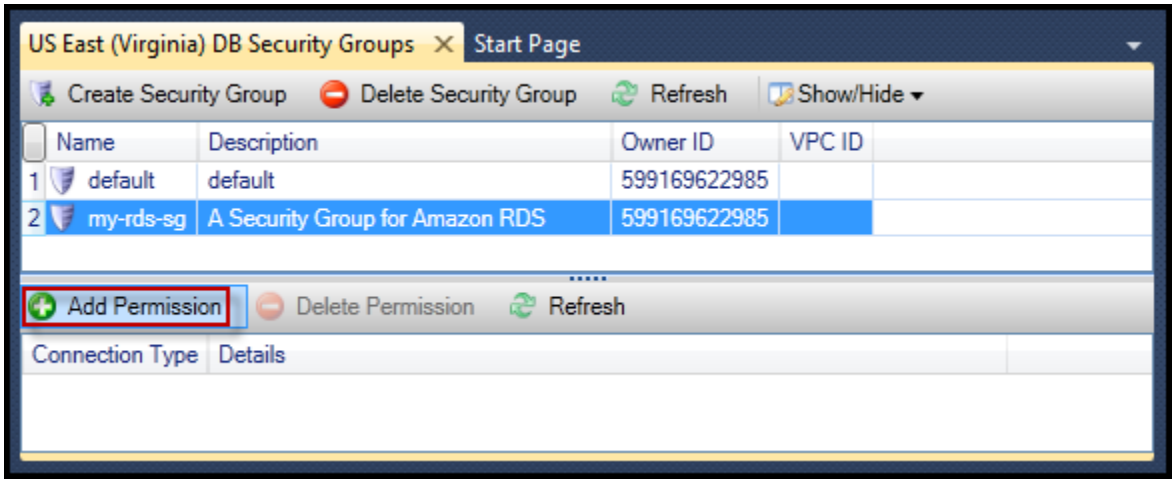
By default, a new Amazon RDS security group provides no network access. To enable access to Amazon RDS instances that use the security group, use the following procedure to set its access permissions.

To set access for an Amazon RDS security group

1. On the **Security Groups** tab, choose the security group from the list view. If your security group does not appear in the list, choose **Refresh**. If your security group still does not appear in the list, verify you are viewing the list for the correct Amazon region. **Security Group** tabs in the Amazon Toolkit are region-specific.

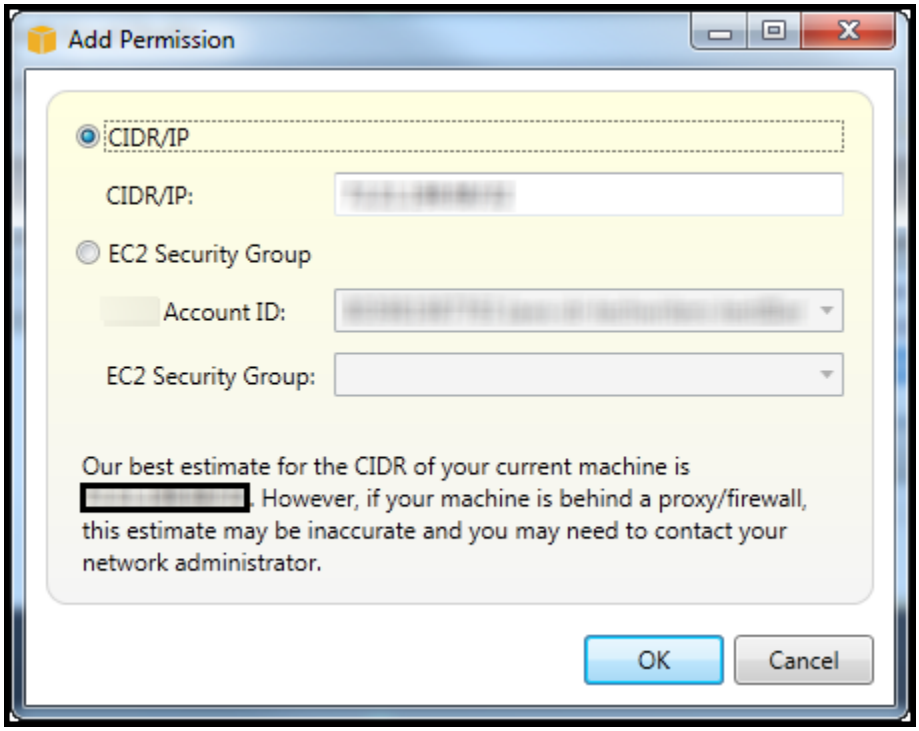
   If no **Security Group** tabs appear, in Amazon Explorer, open the context (right-click) menu for the **DB Security Groups** subnode and choose **View**.

2. Choose **Add Permission**.



Add Permissions button on the **Security Groups** tab

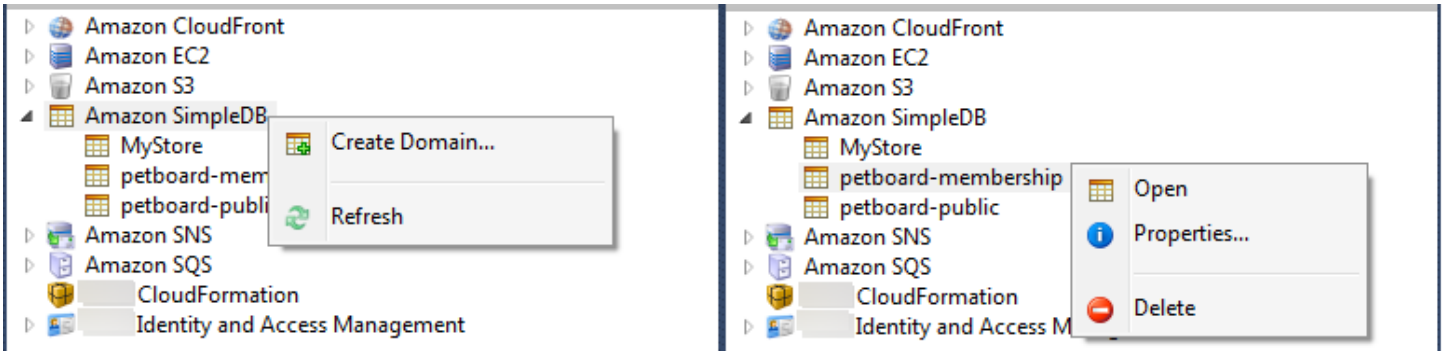
3. In the **Add Permission** dialog box, you can use CIDR notation to specify which IP addresses can access your RDS instance, or you can specify which EC2 security groups can access your RDS instance. When you choose **EC2 Security Group**, you can specify access for all EC2 instances associated with an Amazon Web Services account have access, or you can choose a EC2 security group from the drop-down list.



The Amazon Toolkit attempts to determine your IP address and auto-populate the dialog box with the appropriate CIDR specification. However, if your computer accesses the Internet through a firewall, the CIDR determined by the Toolkit may not be accurate.

Using Amazon SimpleDB from Amazon Explorer

Amazon Explorer displays all of the Amazon SimpleDB domains associated with the active Amazon account. From Amazon Explorer, you can create or delete Amazon SimpleDB domains.

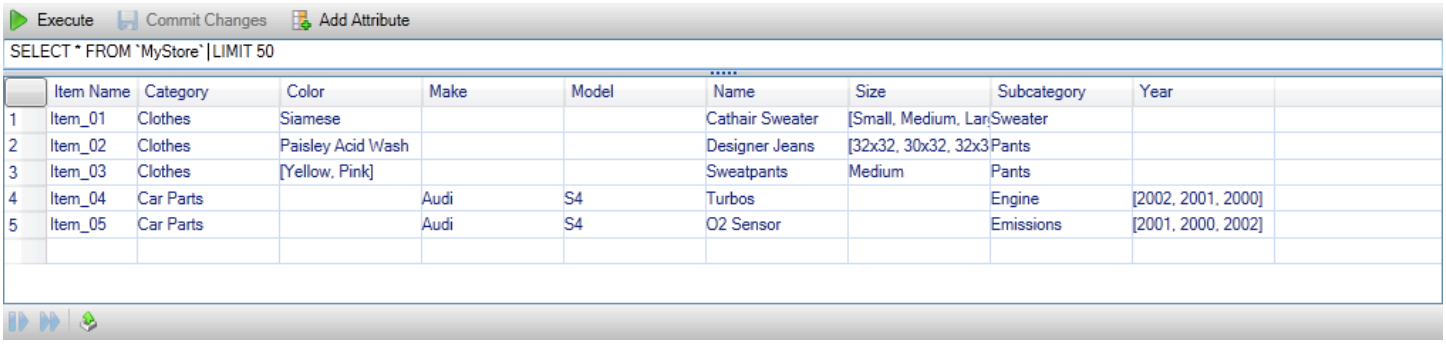


Create, delete, or open Amazon SimpleDB domains associated with your account

Executing Queries and Editing the Results

Amazon Explorer can also display a grid view of a Amazon SimpleDB domain from which you can view the items, attributes, and values in that domain. You can execute queries so that only a subset of the domain's items is displayed. By double-clicking a cell, you can edit the values for that item's corresponding attribute. You can also add new attributes to the domain.

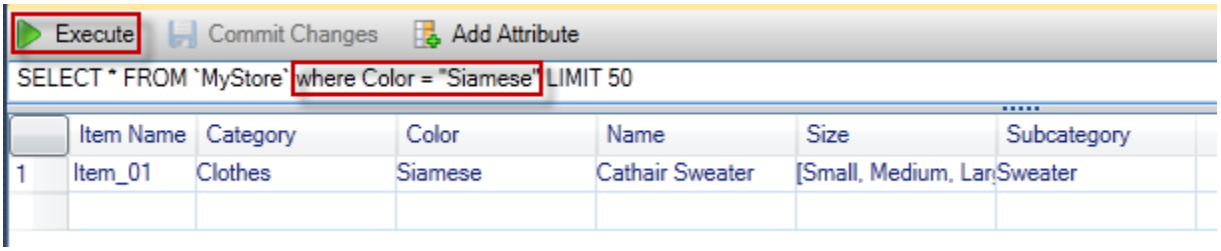
The domain displayed here is from the Amazon SimpleDB sample included with the Amazon SDK for .NET.



	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

To execute a query, edit the query in the text box at the top of the grid view, and then choose **Execute**. The view is filtered to show only the items that match the query.

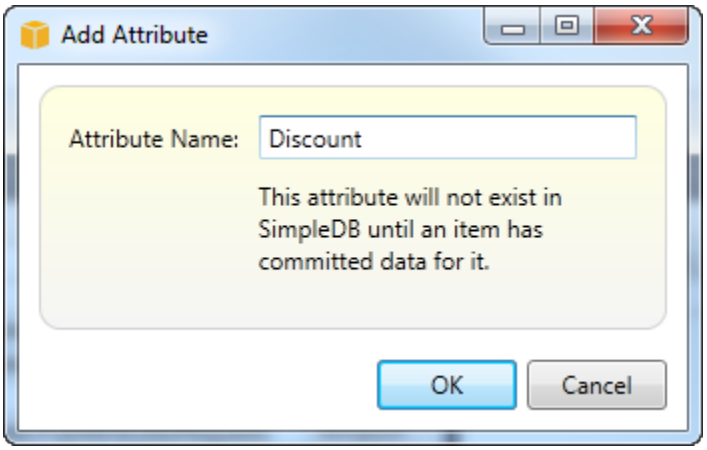


Execute query from Amazon Explorer

To edit the values associated with an attribute, double-click the corresponding cell, edit the values, and then choose **Commit Changes**.

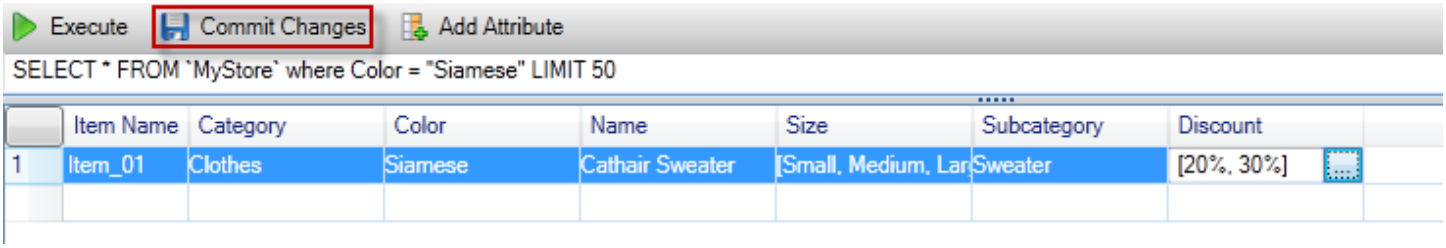
Adding an Attribute

To add an attribute, at the top of the view, choose **Add Attribute**.



Add Attribute dialog box

To make the attribute part of the domain, you must add a value for it to at least one item and then choose **Commit Changes**.



Commit changes for a new attribute

Paginating Query Results

There are three buttons at the bottom of the view.

Paginate and export buttons

The first two buttons provide pagination for query results. To display an additional page of results, choose the first button. To display an additional ten pages of results, choose the second button. In this context, a page is equal to 100 rows or the number of results specified by the LIMIT value, if it is included in the query.

Export to CSV

The last button exports the current results to a CSV file.

Using Amazon SQS from Amazon Explorer

Amazon Simple Queue Service (Amazon SQS) is a flexible queue service that enables message passing between different processes of execution in a software application. Amazon SQS queues are located in the Amazon infrastructure, but the processes that are passing messages can be located locally, on Amazon EC2 instances, or on some combination of these. Amazon SQS is ideal for coordinating the distribution of work across multiple computers.

The Toolkit for Visual Studio enables you to view Amazon SQS queues associated with the active account, create and delete queues, and send messages through queues. (By active account, we mean the account selected in Amazon Explorer.)

For more information about Amazon SQS, go to Introduction to SQS in the Amazon documentation.

Creating a Queue

You can create an Amazon SQS queue from Amazon Explorer. The ARN and URL for the queue will be based on the account number for the active account and the queue name you specify at creation.

To create a queue

1. In Amazon Explorer, open the context (right-click) menu for the **Amazon SQS** node, and then choose **Create Queue**.

2. In the **Create Queue** dialog box, specify the queue name, the default visibility timeout, and the default delivery delay. The default visibility timeout and the default delivery delay are specified in seconds. The default visibility timeout is the amount of time that a message will be invisible to potential receiving processes after a given process has acquired the message. The default delivery delay is the amount of time from the moment the message is sent to the moment it first becomes visible to potential receiving processes.
3. Choose **OK**. The new queue will appear as a subnode under the **Amazon SQS** node.

Deleting a Queue

You can delete existing queues from Amazon Explorer. If you delete a queue, any messages associated with the queue are no longer available.

To delete a queue

1. In Amazon Explorer, open the context (right-click) menus for the queue you want to delete, and then choose **Delete**.

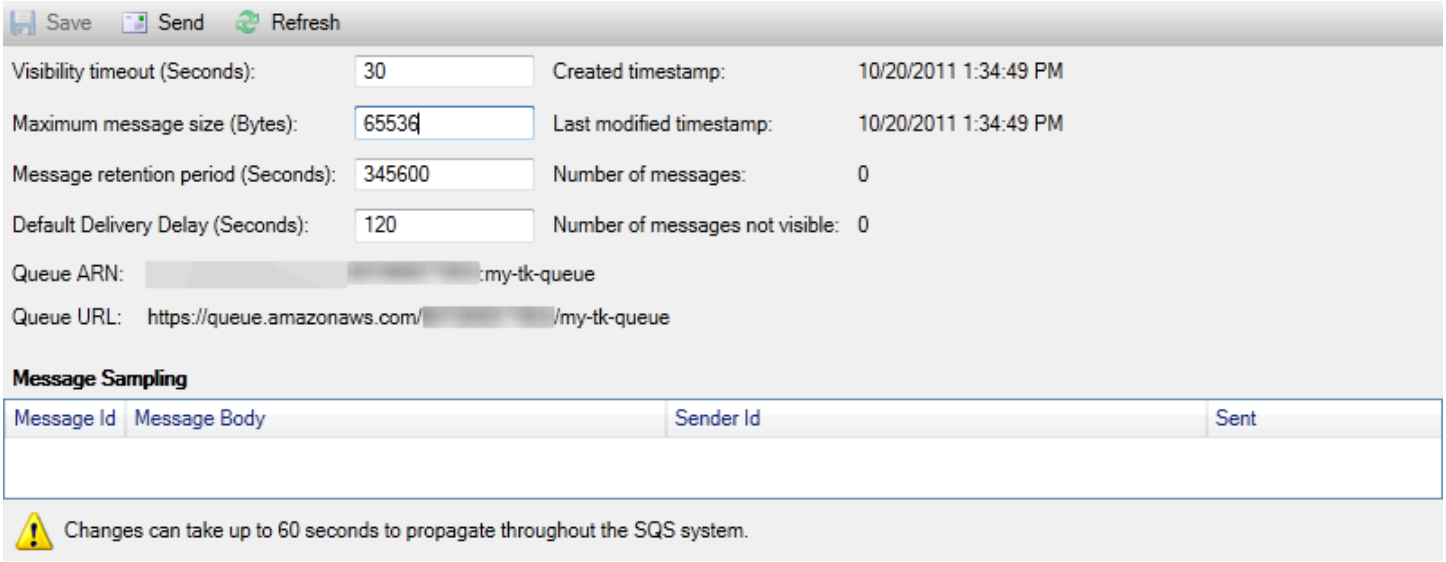
Managing Queue Properties

You can view and edit the properties for any of the queues displayed in Amazon Explorer. You can also send messages to the queue from this properties view.

To manage queue properties

- In Amazon Explorer, open the context (right-click) menu for the queue whose properties you want to manage, and then choose **View Queue**.

  From the queue properties view, you can edit the visibility timeout, the maximum message size, message retention period, and default delivery delay. The default delivery delay can be overridden when you send a message. In the following screenshot, the obscured text is the account number component of the queue ARN and URL.



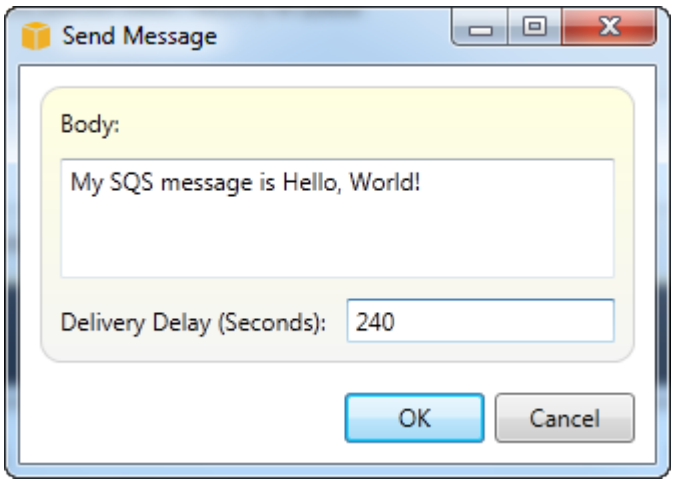
SQS queue properties view

Sending a Message to a Queue

From the queue properties view, you can send a message to the queue.

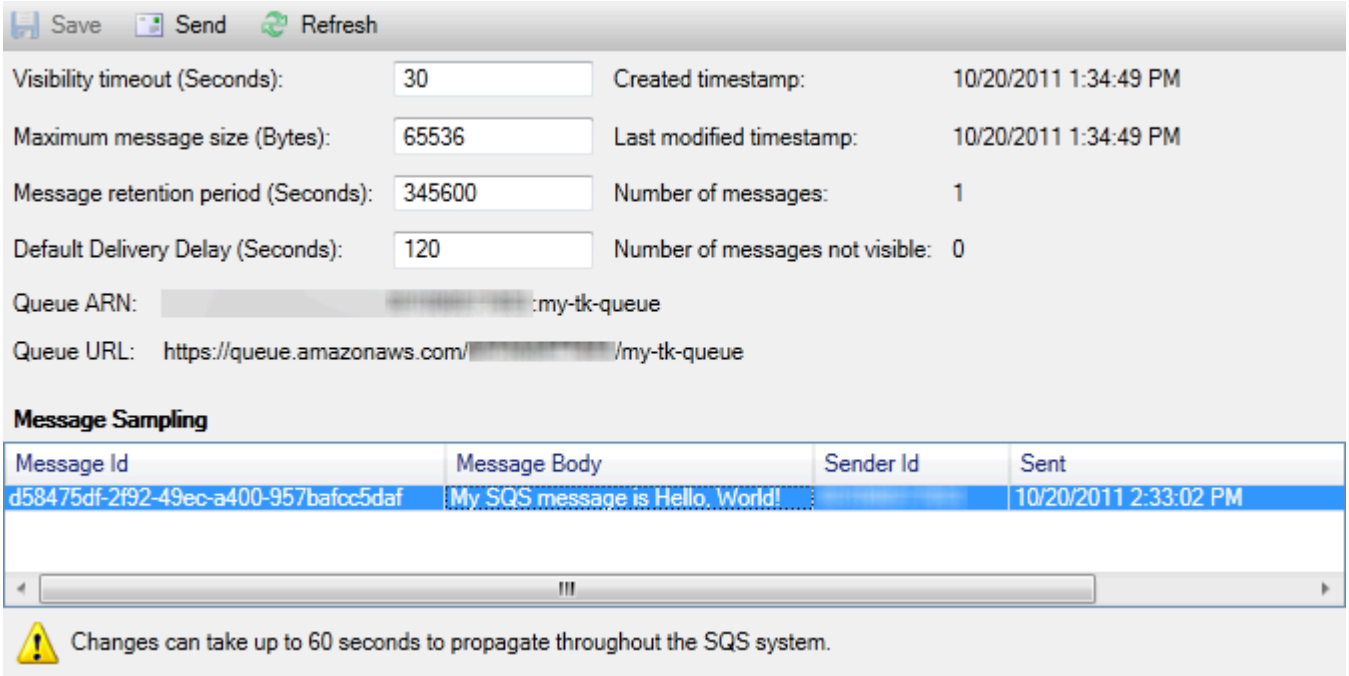
To send a message

1. At the top of the queue properties view, choose the **Send** button.
2. Type the message. (Optional) Enter a delivery delay that will override the default delivery delay for the queue. In the following example, we have overridden the delay with a value of 240 seconds. Choose **OK**.



Send Message dialog box

3. Wait for approximately 240 seconds (four minutes). The message will appear in the **Message Sampling** section of the of the queue properties view.



SQS properties view with sent message

The timestamp in the queue properties view is the time you chose the **Send** button. It does not include the delay. Therefore, the time that the message appears in the queue and is available to receivers might be later than this timestamp. The timestamp is displayed in your computer's local time.

Identity and Access Management

Amazon Identity and Access Management (IAM) enables you to more securely manage access to your Amazon Web Services accounts and resources. With IAM, you can create multiple users in your primary (*root*) Amazon Web Services account. These users can have their own credentials: password, access key ID, and secret key, but all IAM users share a single account number.

You can manage each IAM user's level of resource access by attaching IAM policies to the user. For example, you can attach a policy to an IAM user that gives the user access to the Amazon S3 service and related resources in your account, but which doesn't provide access to any other services or resources.

For more efficient access management, you can create IAM groups, which are collections of users. When you attach a policy to the group, it will affect all users who are members of that group.

In addition to managing permissions at the user and group level, IAM also supports the concept of IAM roles. Like users and groups, you can attach policies to IAM roles. You can then associate the IAM role with an Amazon EC2 instance. Applications that run on the EC2 instance are able to access Amazon using the permissions provided by the IAM role. For more information about using IAM roles with the Toolkit, see Create an IAM Role. For more information about IAM, go to the IAM User Guide.

Create and Configure an IAM User

IAM users enable you to grant others access to your Amazon Web Services account. Because you are able to attach policies to IAM users, you can precisely limit the resources an IAM user can access and the operations they can perform on those resources.

As a best practice, all users who access an Amazon Web Services account should do so as IAM users —even the owner of the account. This ensures that if the credentials for one of the IAM users are compromised, just those credentials can be deactivated. There is no need to deactivate or change the root credentials for the account.

From the Toolkit for Visual Studio, you can assign permissions to an IAM user either by attaching an IAM policy to the user or by assigning the user to a group. IAM users who are assigned to a group derive their permissions from the policies attached to the group. For more information, see Create an IAM Group and Add an IAM User to an IAM Group.

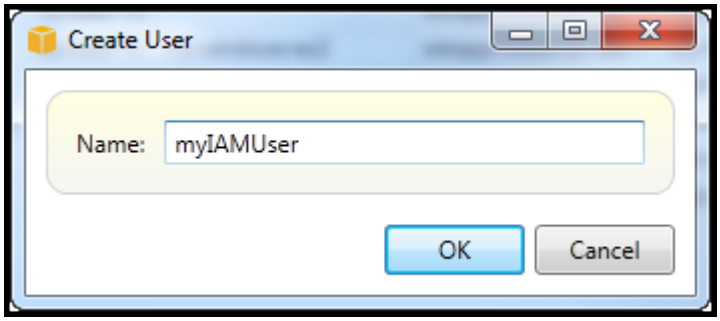
From the Toolkit for Visual Studio, you can also generate Amazon credentials (access key ID and secret key) for the IAM user. For more information, see Generate Credentials for an IAM User

The Toolkit for Visual Studio supports specifying IAM user credentials for accessing services through Amazon Explorer. Because IAM users typically do not have full access to all Amazon Web Services, some of the functionality in Amazon Explorer might not be available. If you use Amazon Explorer to change resources while the active account is an IAM user and then switch the active account to the root account, the changes might not be visible until you refresh the view in Amazon Explorer. To refresh the view, choose the refresh () button.

For information about how to configure IAM users from the Amazon Web Services Management Console, go to Working with Users and Groups in the IAM User Guide.

To create an IAM user

1. In Amazon Explorer, expand the **Amazon Identity and Access Management** node, open the context (right-click) menu for **Users** and then choose **Create User**.
2. In the **Create User** dialog box, type a name for the IAM user and choose **OK**. This is the IAM friendly name. For information about constraints on names for IAM users, go to the IAM User Guide.



Create an IAM user

The new user will appear as a subnode under **Users** under the **Amazon Identity and Access Management** node.

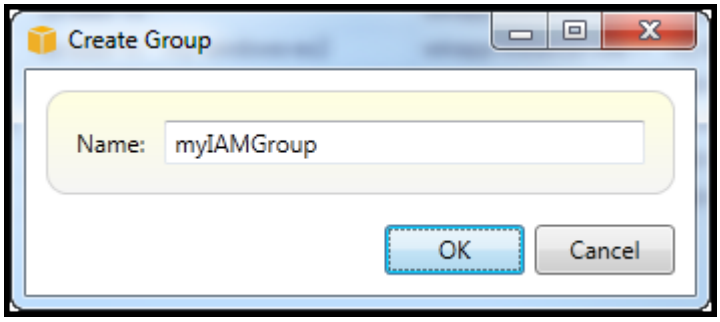
For information about how to create a policy and attach it to the user, see Create an IAM Policy.

Create an IAM Group

Groups provide a way of applying IAM policies to a collection of users. For information about how to manage IAM users and groups, go to Working with Users and Groups in the IAM User Guide.

To create an IAM group

1. In Amazon Explorer, under **Identity and Access Management**, open the context (right-click) menu for **Groups** and choose **Create Group**.
2. In the **Create Group** dialog box, type a name for the IAM group and choose **OK**.



Create IAM group

The new IAM group will appear under the **Groups** subnode of **Identity and Access Management**.

For information about to create a policy and attach it to the IAM group, see Create an IAM Policy.

Add an IAM User to an IAM Group

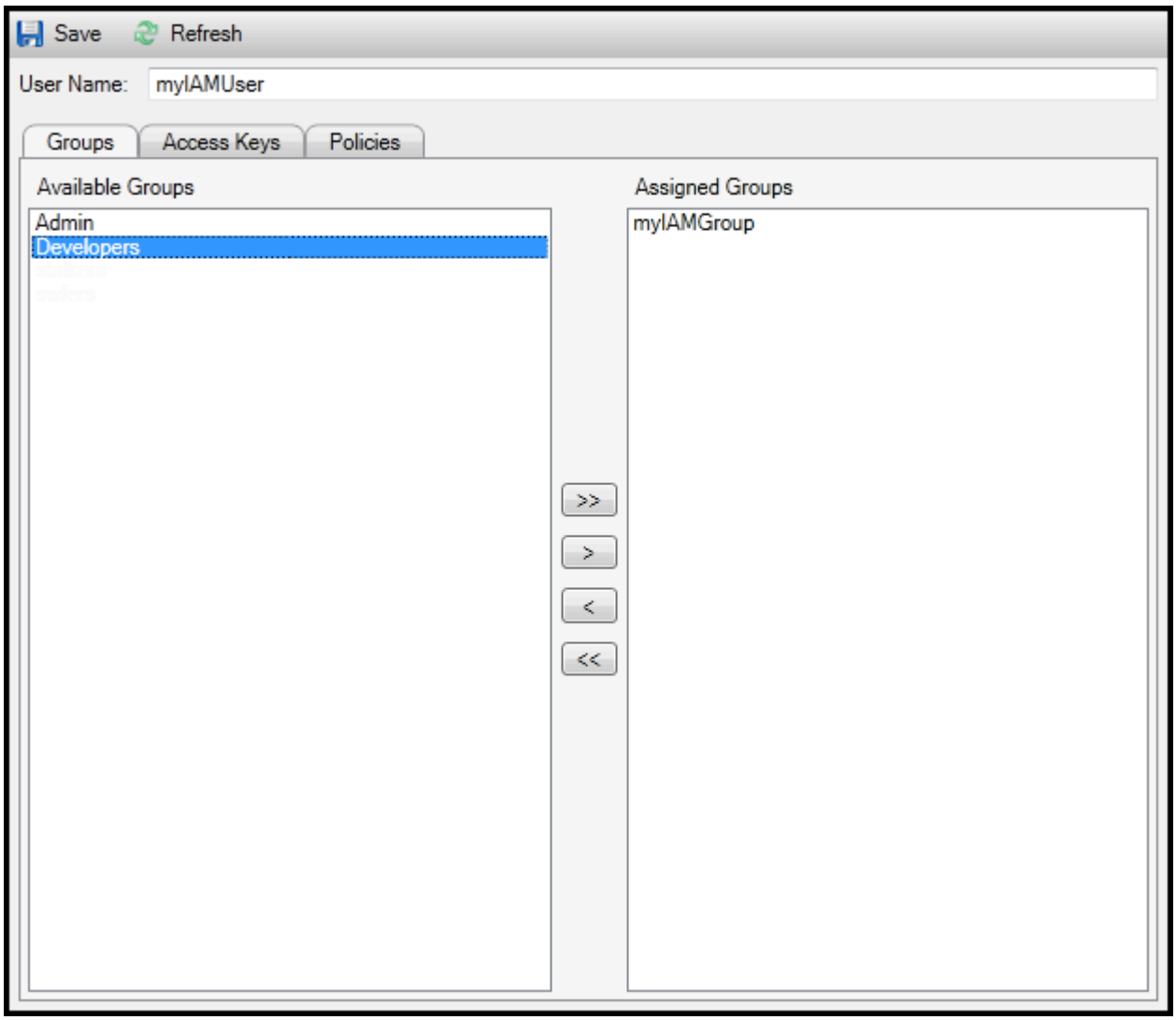
IAM users who are members of an IAM group derive access permissions from the policies attached to the group. The purpose of an IAM group is to make it easier to manage permissions across a collection of IAM users.

For information about how the policies attached to an IAM group interact with the policies attached to IAM users who are members of that IAM group, go to Managing IAM Policies in the IAM User Guide.

In Amazon Explorer, you add IAM users to IAM groups from the **Users** subnode, not the **Groups** subnode.

To add an IAM user to a IAM group

1. In Amazon Explorer, under **Identity and Access Management**, open the context (right-click) menu for **Users** and choose **Edit**.



Assign an IAM user to a IAM group

2. The left pane of the **Groups** tab displays the available IAM groups. The right pane displays the groups of which the specified IAM user is already a member.

   To add the IAM user to a group, in the left pane, choose the IAM group and then choose the **>** button.

   To remove the IAM user from a group, in the right pane, choose the IAM group and then choose the **<** button.

   To add the IAM user to all of the IAM groups, choose the **>>** button. Similarly, to remove the IAM user from all of the groups, choose the **<<** button.

   To choose multiple groups, choose them in sequence. You do not need to hold down the Control key. To clear a group from your selection, simply choose it a second time.

3. When you have finished assigning the IAM user to IAM groups, choose **Save**.

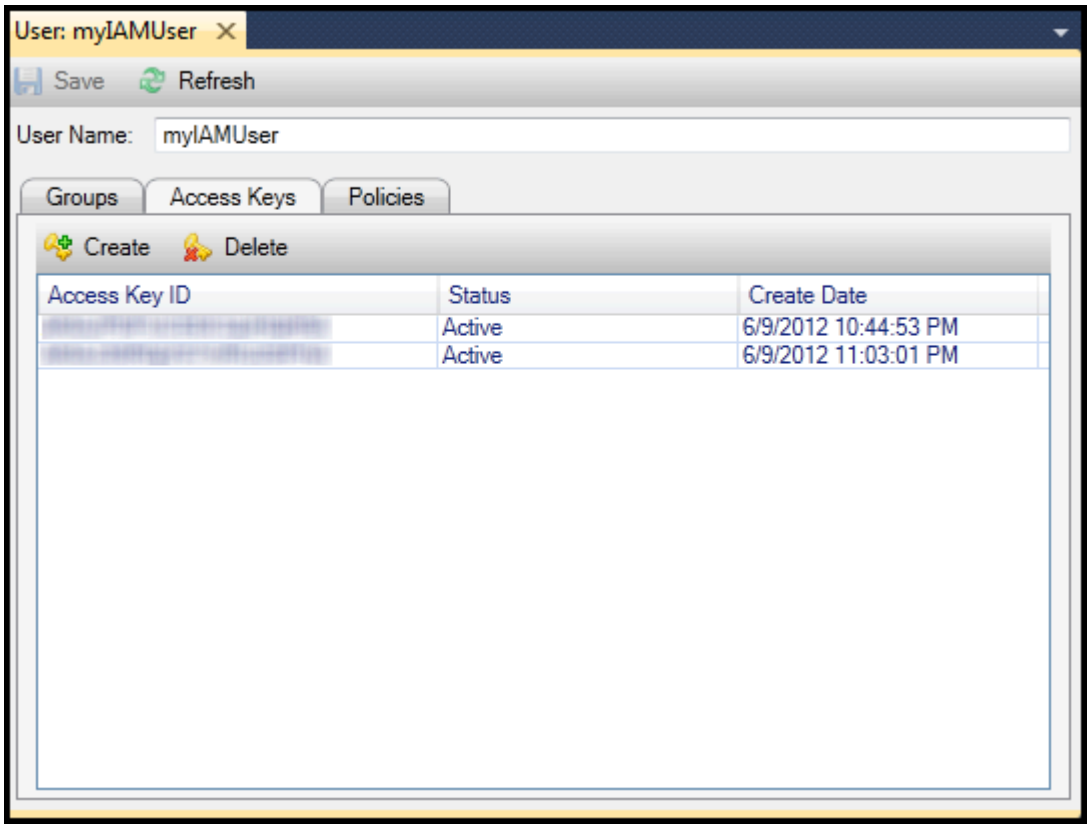
Generate Credentials for an IAM User

With Toolkit for Visual Studio, you can generate the access key ID and secret key used to make API calls to Amazon. These keys can also be specified to access Amazon Web Services through the Toolkit. For more information about how to specify credentials for use with the Toolkit, see creds. For more information about how to safely handle credentials, see Best Practices for Managing Amazon Access Keys.

The Toolkit cannot be used to generate a password for an IAM user.

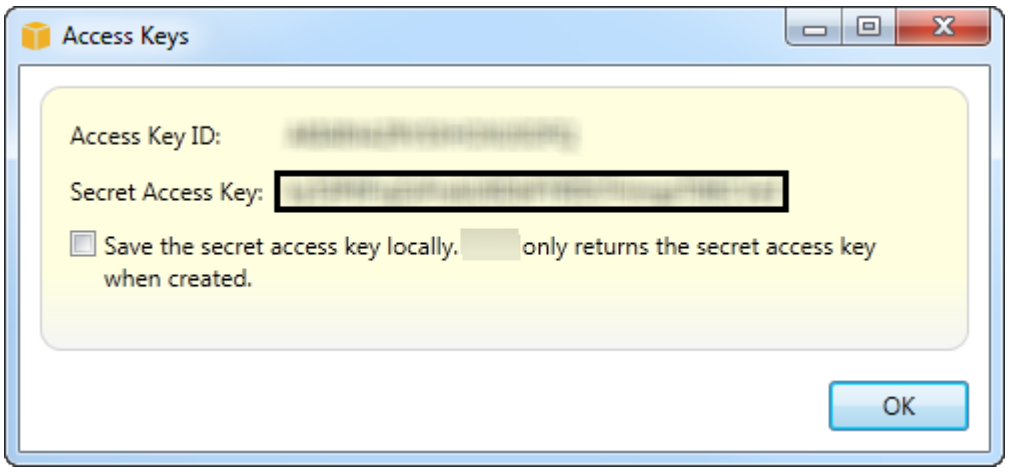
To generate credentials for an IAM user

1. In Amazon Explorer, open the context (right-click) menu for an IAM user and choose **Edit**.



2. To generate credentials, on the **Access Keys** tab, choose **Create**.

   You can generate only two sets of credentials per IAM user. If you already have two sets of credentials and need to create an additional set, you must delete one of the existing sets.

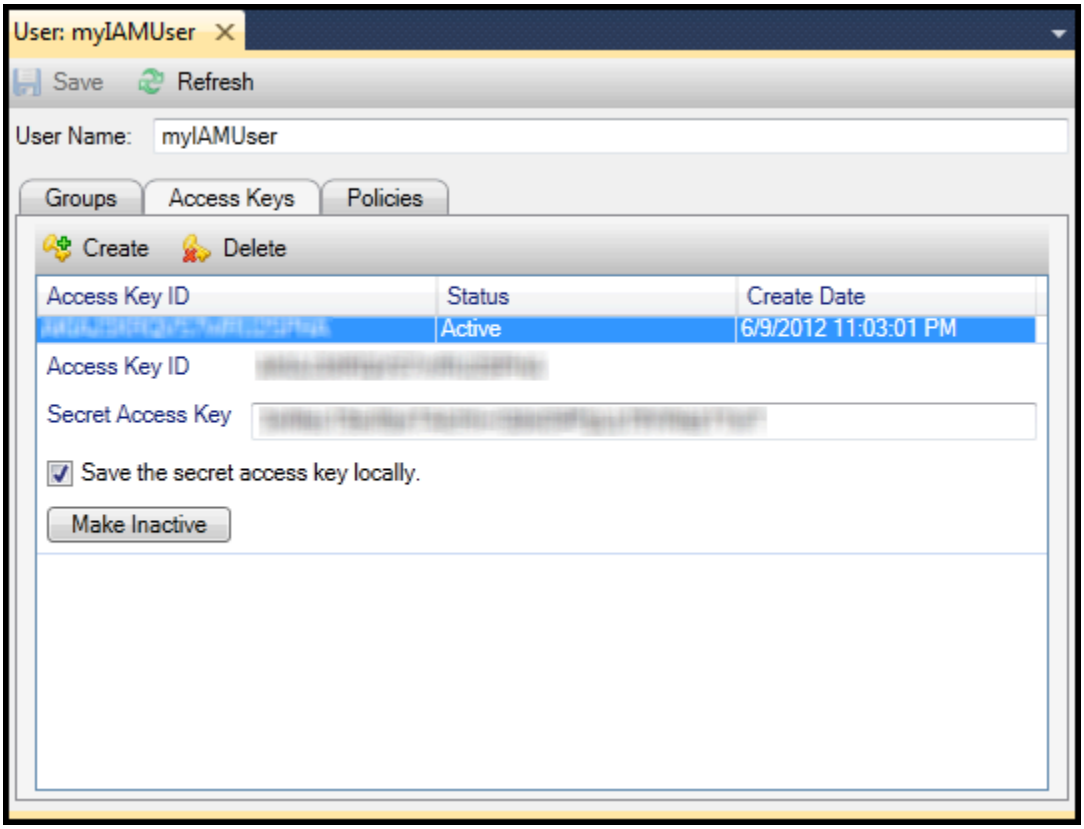


reate credentials for IAM user

If you want the Toolkit to save an encrypted copy of your secret access key to your local drive, select **Save the secret access key locally. Amazon only returns the secret access key when created**. You can also copy the secret access key from the dialog box and save it in a secure location.

3. Choose **OK**.

After you generate the credentials, you can view them from the **Access Keys** tab. If you selected the option to have the Toolkit save the secret key locally, it will be displayed here.



Create credentials for IAM user

If you saved the secret key yourself and would also like the Toolkit to save it, in the **Secret Access Key** box, type the secret access key, and then select **Save the secret access key locally**.

To deactivate the credentials, choose **Make Inactive**. (You might do this if you suspect the credentials have been compromised. You can reactivate the credentials if you receive an assurance they are secure.)

Create an IAM Role

The Toolkit for Visual Studio supports the creation and configuration of IAM roles. Just as with users and groups, you can attach policies to IAM roles. You can then associate the IAM role with an Amazon EC2 instance. The association with the EC2 instance is handled through an *instance profile*, which is a logical container for the role. Applications that run on the EC2 instance are automatically granted the level of access specified by the policy associated with the IAM role. This is true even when the application hasn't specified other Amazon credentials.

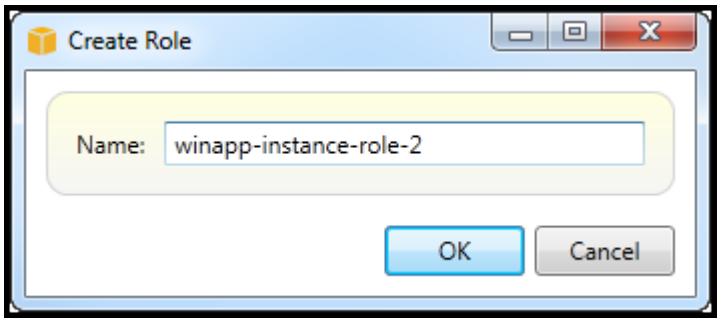
For example, you can create a role and attach a policy to that role that limits access to Amazon S3 only. After associating this role with an EC2 instance, you can then run an application on that instance and the application will have access to Amazon S3, but not to any other services or

resources. The advantage of this approach is that you don't need to be concerned with securely transferring and storing Amazon credentials on the EC2 instance.

For more information about IAM roles, go to Working with IAM Roles in the IAM User Guide. For examples of programs accessing Amazon using the IAM role associated with an Amazon EC2 instance, go to the Amazon developer guides for Java, .NET, PHP, and Ruby (Setting Credentials Using IAM, Creating an IAM Role, and Working with IAM Policies).

To create an IAM role

1. In Amazon Explorer, under **Identity and Access Management**, open the context (right-click) menu for **Roles** and then choose **Create Roles**.
2. In the **Create Role** dialog box, type a name for the IAM role and choose **OK**.



Create IAM role

The new IAM role will appears under **Roles** in **Identity and Access Management**.

For information about how to create a policy and attach it to the role, see Create an IAM Policy.

Create an IAM Policy

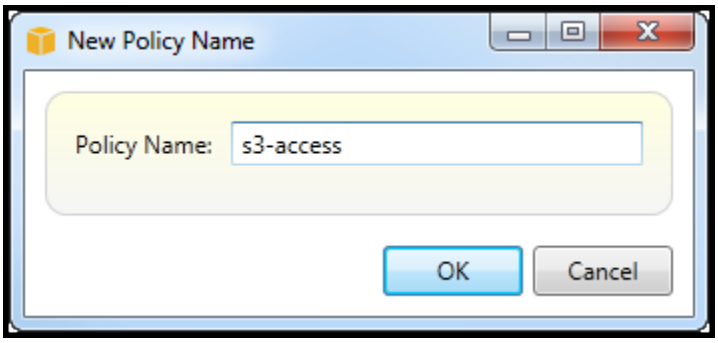
Policies are fundamental to IAM. Policies can be associated with IAM *entities* such as users, groups, or roles. Policies specify the level of access enabled for a user, group, or role.

To create an IAM policy

In Amazon Explorer, expand the **Amazon Identity and Access Management** node, then expand the node for the type of entity (**Groups**, **Roles**, or **Users**) to which you will attach the policy. For example, open a context menu for an IAM role and choose **Edit**.

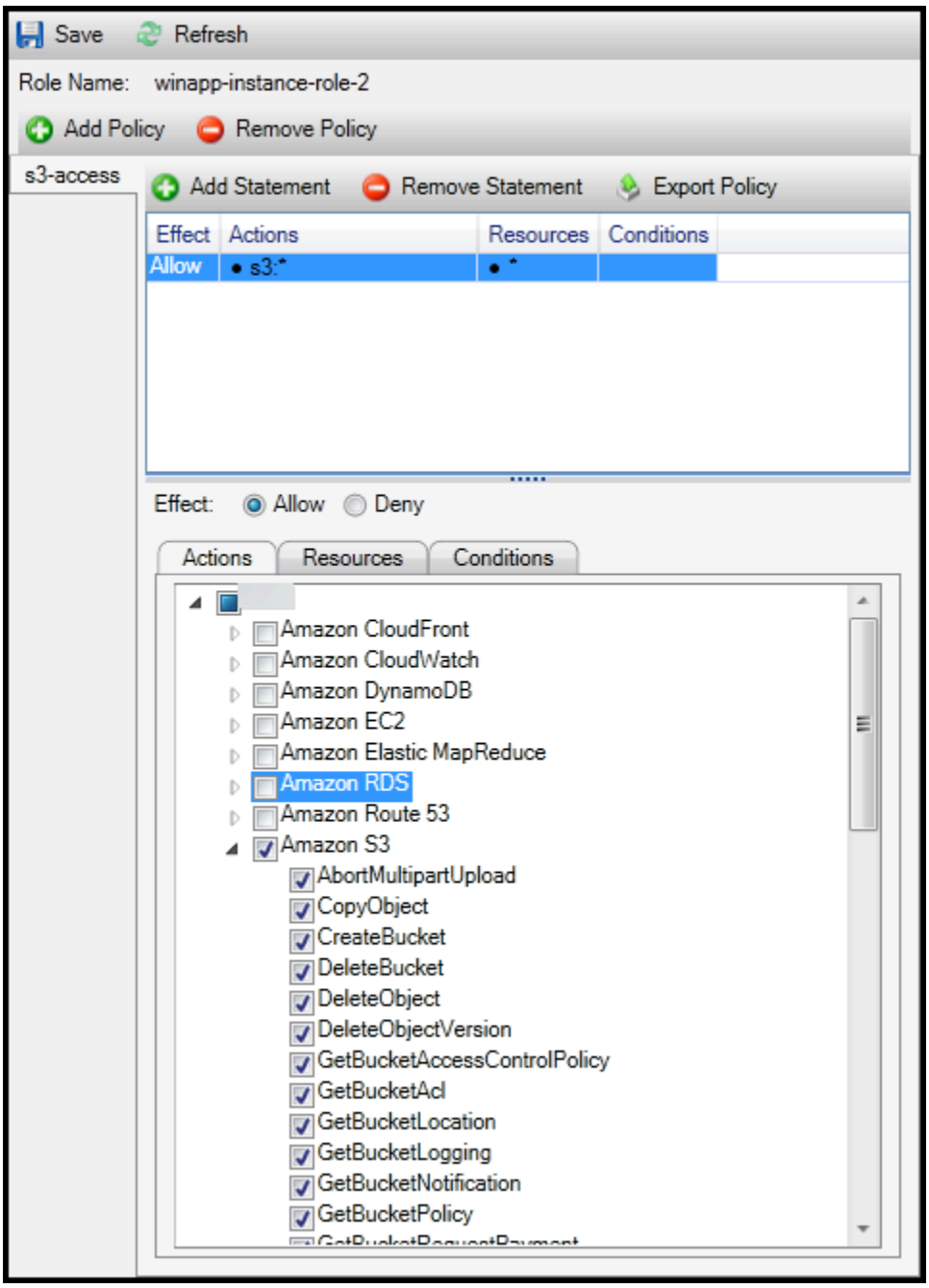
A tab associated with the role will appear in the Amazon Explorer. Choose the **Add Policy** link.

In the **New Policy Name** dialog box, type a name for the policy (for example, s3-access).



New Policy Name dialog box

In the policy editor, add policy statements to specify the level of access to provide to the role (in this example, winapp-instance-role-2 associated with the policy. In this example, a policy provides full access to Amazon S3, but no access to any other resources.



Specify IAM policy

For more precise access control, you can expand the subnodes in the policy editor to allow or disallow actions associated with Amazon Web Services.

When you have edited the policy, choose the **Save** link.

Amazon Lambda

Develop and deploy your .NET Core-based C# Lambda functions with the Amazon Toolkit for Visual Studio. Amazon Lambda is a compute service that lets you run code without provisioning or managing servers. The Toolkit for Visual Studio includes Amazon Lambda .NET Core project templates for Visual Studio.

For more information about Amazon Lambda, see the Amazon Lambda Developer Guide.

For more information about .NET Core, see the Microsoft .NET Core guide. For .NET Core prerequisites and installation instructions for Windows, macOS, and Linux platforms, see .NET Core Downloads.

The following topics describe how to work with Amazon Lambda using the Toolkit for Visual Studio.

Topics

- Basic Amazon Lambda Project
- Basic Amazon Lambda Project Creating Docker Image
- Tutorial: Build and Test a Serverless Application with Amazon Lambda
- Tutorial: Creating an Amazon Rekognition Lambda Application
- Tutorial: Using Amazon Logging Frameworks with Amazon Lambda to Create Application Logs

Basic Amazon Lambda Project

You can create a Lambda function using Microsoft .NET Core project templates, in the Amazon Toolkit for Visual Studio.

Create a Visual Studio .NET Core Lambda Project

You can use Lambda-Visual Studio templates and blueprints to help speed up your project initialization. Lambda blueprints contain pre-written functions that simplify the creation of a flexible project foundation.

> **Note**
>
> The Lambda service has data limits on different package types. For detailed information about data limits, see the Lambda quotas topic in the *Amazon Lambda* User Guide.

To create a Lambda project in Visual Studio

1. From Visual Studio expand the **File** menu, expand **New**, then choose **Project**.
2. From the **New Project** dialog box, set the **Language**, **Platform**, and **Project type** drop-down boxes to "All", then type **aws lambda** in the **Search** field. Choose the **Amazon Lambda Project (.NET Core - C#)** template.
3. In the **Name** field, enter **`AWSLambdaSample`**, specify your desired file **Location**, then choose **Create** to proceed.
4. From the **Select Blueprint** page, select the **Empty Function** blueprint, then choose **Finish** to create the Visual Studio project.

Review the Project Files

There are two project files to review: `aws-lambda-tools-defaults.json` and `Function.cs`.

The following example shows the `aws-lambda-tools-defaults.json` file, which is automatically created as part of your project. You can set build options by using the fields in this file.

> **Note**
>
> The project templates in Visual Studio contain many different fields, take note of the following:
>
> - **function-handler**: specifies the method that runs when the Lambda function runs
> - Specifying a value in the **function-handler** field pre-populates that value in the Publish wizard.
> - If you rename the function, class, or assembly then you also need to update the corresponding field in the `aws-lambda-tools-defaults.json` file.

```
{
  "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.",
```

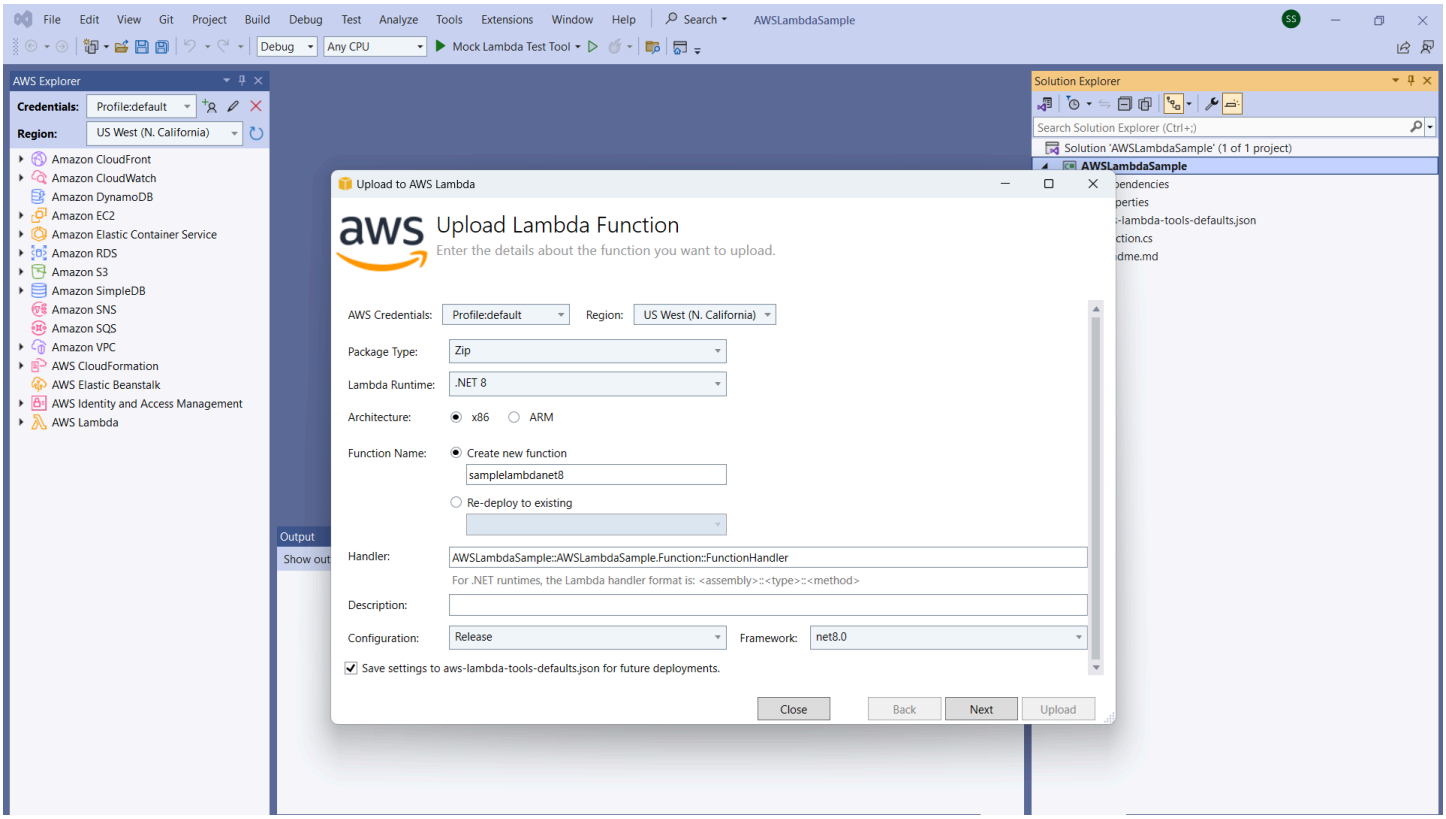
```
    "dotnet lambda help",
    "All the command line options for the Lambda command can be specified in this
 file."
  ],
  "profile": "default",
  "region": "us-west-2",
  "configuration": "Release",
  "function-architecture": "x86_64",
  "function-runtime": "dotnet8",
  "function-memory-size": 512,
  "function-timeout": 30,
  "function-handler": "AWSLambdaSample::AWSLambdaSample.Function::FunctionHandler"
}
```

Examine the `Function.cs` file. `Function.cs` defines the c# functions to expose as Lambda functions. This `FunctionHandler` is the Lambda functionality that runs when the Lambda function runs. In this project, there is one function defined: `FunctionHandler`, which calls `ToUpper()` on the input text.

Your project is now ready to publish to Lambda.

Publishing to Lambda

The following procedure and image demonstrate how to upload your function to Lambda using the Amazon Toolkit for Visual Studio.



Publishing your function to Lambda

1. Navigate to the Amazon Explorer by expanding **View** and choosing **Amazon Explorer**.
2. In the **Solution Explorer**, open the context menu for (right-click) the project you want to publish, then choose **Publish to Amazon Lambda** to open the **Upload Lambda Function** window.
3. From the **Upload Lambda Function** window, complete the following fields:

    a. **Package Type**: Choose **`Zip`**. A ZIP file will be created as a result of the build process and will be uploaded to Lambda. Alternatively, you can choose **Package Type `Image`**. The Tutorial: Basic Lambda Project Creating Docker Image describes how to publish using **Package Type `Image`**.

    b. **Lambda Runtime**: Choose your Lambda Runtime from the drop-down menu.

    c. **Architecture**: Select the radial for your preferred architecture.

    d. **Function Name**: Select the radial for **Create new function**, then enter a display name for your Lambda instance. This name is referenced by both the Amazon Explorer and Amazon Web Services Management Console displays.

e. **Handler**: Use this field to specify a function handler. For example: **`AWSLambdaSample::AWSLambdaSample.Function::FunctionHandler`**.

f. *(Optional)* **Description**: Enter descriptive text to display with your instance, from within the Amazon Web Services Management Console.

g. **Configuration**: Choose your preferred configuration from the drop-down menu.

h. **Framework**: Choose your preferred framework from the drop-down menu.

i. **Save settings**: Select this box to save your current settings to `aws-lambda-tools-defaults.json` as the default for future deploments.

j. Choose **Next** to proceed to the **Advanced Function Details** window.

4. In the **Advanced Function Details** window, complete the following fields:

a. **Role Name**: Choose a role associated with your account. The role provides temporary credentials for any Amazon service calls made by the code in the function. If you do not have a role, scroll to locate **New Role based on Amazon Managed Policy** in the drop-down selector, then choose **AWSLambdaBasicExecutionRole**. This role has minimal access permissions.

> **Note**
>
> Your account must have permission to run the IAM ListPolicies action, or the **Role Name** list will be empty and you will be unable to continue.

b. *(Optional)* If your Lambda function accesses resources on an Amazon VPC, select the subnets and security groups.

c. *(Optional)* Set any environment variables that your Lambda function needs. The keys are automatically encrypted by the default service key which is free. Alternatively, you can specify an Amazon KMS key, for which there is a charge. KMS is a managed service you can use to create and control the encryption keys used to encrypt your data. If you have an Amazon KMS key, you can select it from the list.

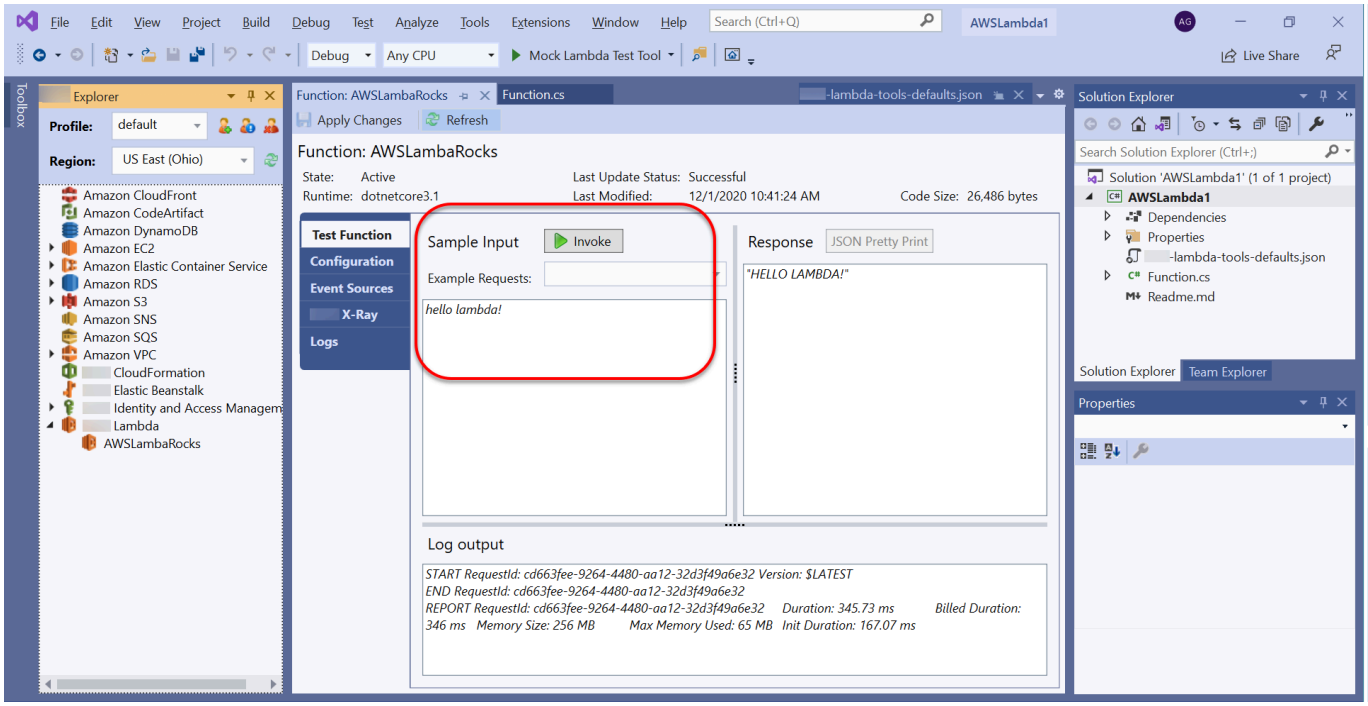
5. Choose **Upload** to open the **Uploading Function** window and begin the upload process.

> **Note**
>
> The **Uploading Function** page displays while the function is uploading to Amazon. To keep the wizard open after uploading so that you can view the report, clear

Automatically close wizard on successful completion at the bottom of the form before the upload completes.
After the function uploads, your Lambda function is live. The **Function:** view page opens and displays your new Lambda function's configuration.

6. From the **Test Function** tab, enter `hello lambda!` in the text-input field and then choose **Invoke** to manually invoke your Lambda function. Your text appears in the **Response** tab, converted to uppercase.

> **Note**
>
> You can reopen the **Function:** view at any time by double-clicking on your deployed instance located in the **Amazon Explorer** under the **Amazon Lambda** node.



7. *(Optional)* To confirm that you successfully published your Lambda function, log into the Amazon Web Services Management Console and then choose Lambda. The console displays all of your published Lambda functions, including the one you just created.

Clean-up

If you are not going to continue developing with this example, delete the function you deployed so that you do not get billed for unused resources in your account.

> **Note**
>
> Lambda automatically monitors Lambda functions for you, reporting metrics through Amazon CloudWatch. To monitor and troubleshoot your function, see the Troubleshooting and Monitoring Amazon Lambda Functions with Amazon CloudWatch topic in the Amazon Lambda Developer Guide.

To delete your function

1. From the **Amazon Explorer** expand the **Amazon Lambda** node.
2. Right click your deployed instance, then choose **Delete**.

Basic Amazon Lambda Project Creating Docker Image

You can use the Toolkit for Visual Studio to deploy your Amazon Lambda function as a Docker image. Using Docker, you have more control over your runtime. For example, you can choose custom runtimes like .NET 8.0. You deploy your Docker image in the same way as any other container image. This tutorial closely mimics Tutorial: Basic Lambda Project, with two differences:

- A Dockerfile is included in the project.
- An alternate publishing configuration is chosen.

For information about Lambda container images, see Lambda Deployment Packages in the *Amazon Lambda Developer Guide*.

For additional information about working with Lambda Amazon Toolkit for Visual Studio, see the Using the Amazon Lambda Templates in the Amazon Toolkit for Visual Studio topic in this User Guide.

Create a Visual Studio .NET Core Lambda Project

You can use Lambda Visual Studio templates and blueprints to help speed up your project initialization. Lambda blueprints contain pre-written functions that simplify the creation of a flexible project foundation.

To create a Visual Studio .NET Core Lambda project

1. From Visual Studio expand the **File** menu, expand **New**, then choose **Project**.
2. From the **New Project** dialog box, set the **Language**, **Platform**, and **Project type** drop-down boxes to "All", then type **`aws lambda`** in the **Search** field. Choose the **Amazon Lambda Project (.NET Core - C#)** template.
3. In the **Project Name** field, enter **`AWSLambdaDocker`**, specify your file **Location**, then choose **Create**.
4. On the **Select Blueprint** page, choose the **.NET 8 (Container Image)** blueprint, and then choose **Finish** to create the Visual Studio project. You can now review the project's structure and code.

Reviewing Project Files

The following sections examine the three project files created by the **.NET 8 (Container Image)** blueprint:

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

1. Dockerfile

A `Dockerfile` performs three primary actions:

- `FROM`: Establishes the base image to utilize for this image. This base image provides .NET Runtime, Lambda runtime, and a shell script that provides an entry point for the Lambda .NET process.
- `WORKDIR`: Establishes the image's internal work directory as `/var/task`.
- `COPY`: Will copy the files generated from the build process from their local location into the work directory of the image.

The following are optional `Dockerfile` actions that you can specify:

- `ENTRYPOINT`: The base image already includes an `ENTRYPOINT`, which is the start-up process executed when the image is started. If you wish to specify your own, then you are overriding that base entry point.
- `CMD`: Instructs Amazon which custom code you want executed. It expects a fully-qualified name to your custom method. This line either needs to be included directly in the Dockerfile or can be specified during the publish process.

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

The following is an example of a Dockerfile created by the .NET 8 (Container Image) blueprint.

```
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

The `aws-lambda-tools-defaults.json` file is used to specify default values for the Toolkit for Visual Studio deployment wizard and .NET Core CLI. The following list describes fields that you can set in your `aws-lambda-tools-defaults.json` file.

- `profile`: sets your Amazon profile.
- `region`: sets the Amazon region where your resources are stored.
- `configuration`: sets the configuration used to publish your function.
- `package-type`: sets the deployment package-type to a container image or .zip file archive.
- `function-memory-size`: sets the memory allocation for your function in MB.
- `function-timeout`: Timeout is the maximum amount of time in seconds that a Lambda function can run. You can adjust this in increments of 1 second up to a maximum value of 15 minutes.
- `docker-host-build-output-dir`: sets the output directory of the build process that correlates with the instructions in the `Dockerfile`.
- `image-command`: is a fully-qualified name to your method, the code you want the Lambda function to run. The syntax is: `{Assembly}::{Namespace}.{ClassName}::{MethodName}`. For more information, see Handler signatures. Setting `image-command` here pre-populates this value in Visual Studio's Publish wizard later on.

The following is an example of an aws-lambda-tools-defaults.json created by the .NET 8 (Container Image) blueprint.

```
{
  "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

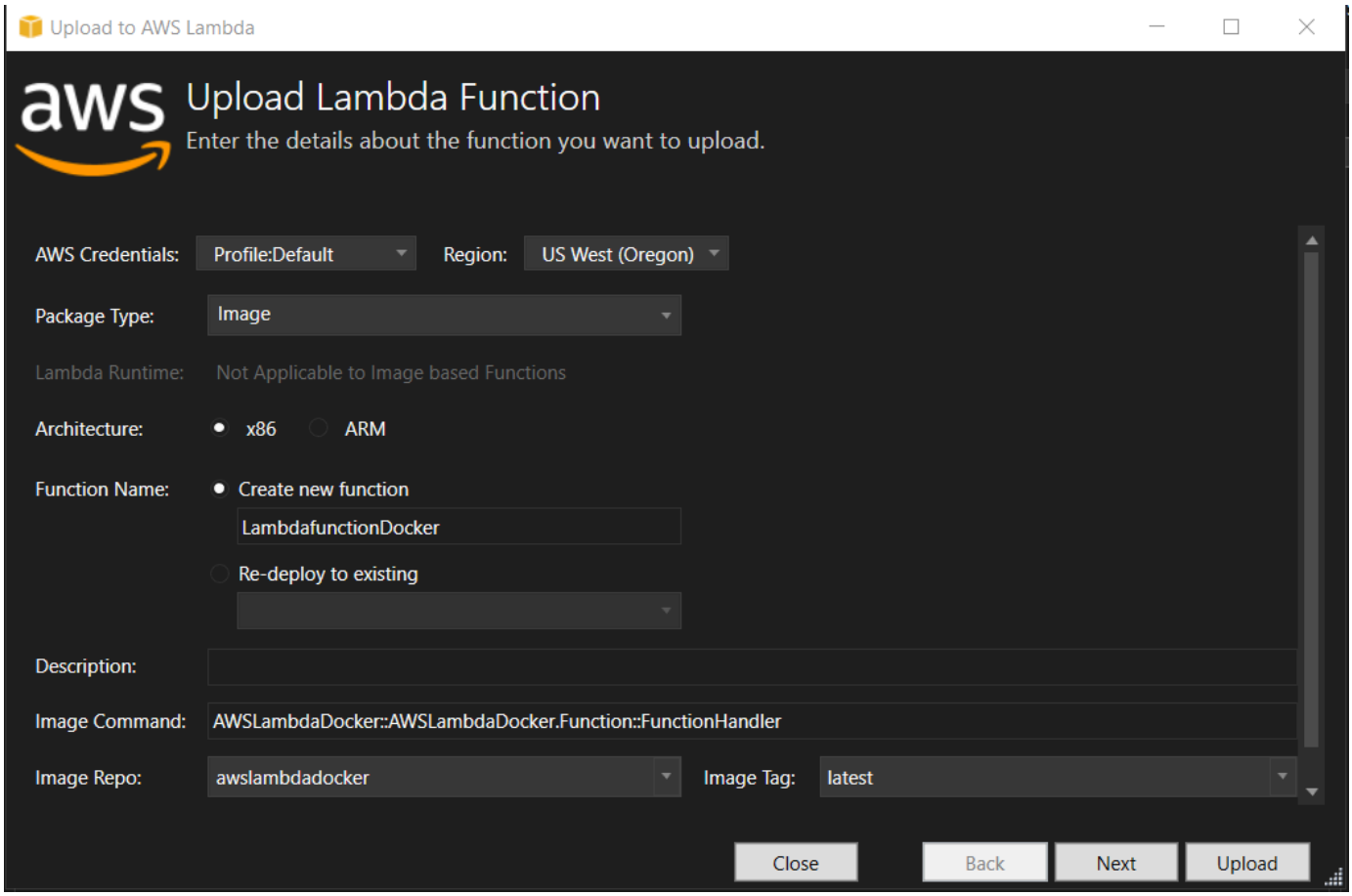
The `Function.cs` file defines the c# functions to be exposed as Lambda functions. The `FunctionHandler` is the Lambda functionality that runs when the Lambda function runs. In this project, `FunctionHandler` calls `ToUpper()` on the input text.

Publish to Lambda

Docker images that are generated by the build process are uploaded to Amazon Elastic Container Registry (Amazon ECR). Amazon ECR is a fully-managed Docker container registry that you use to store, manage, and deploy Docker container images. Amazon ECR hosts the image, which Lambda then references to provide the programmed Lambda functionality when invoked.

To publish your function to Lambda

1. From the **Solution Explorer**, open the context menu for (right-click) the project, then choose **Publish to Amazon Lambda** to open the **Upload Lambda Function** window.
2. From the **Upload Lambda Function** page, do the following:



a. For **Package Type**, **`Image`** has been automatically selected as your **Package Type** because the publish wizard detected a `Dockerfile` within your project.

b. For **Function Name**, enter a display name for your Lambda instance. This name is the reference name displayed in the both the Amazon Explorer in Visual Studio and the Amazon Web Services Management Console.

c. For **Description**, enter text to display with your instance in the Amazon Web Services Management Console.

d. For **Image Command**, enter a fully-qualified path to the method you want the Lambda function to run:
`AWSLambdaDocker::AWSLambdaDocker.Function::FunctionHandler`

> **Note**
>
> Any method name entered here will override any CMD instruction within the Dockerfile. Entering **Image Command** is optional only IF your `Dockerfile` includes a `CMD` to instruct how to launch the Lambda function.

e. For **Image Repo**, enter the name of a new or existing Amazon Elastic Container Registry. The Docker image the build process creates is uploaded to this registry. The Lambda definition that is being published will reference that Amazon ECR image.

f. For **Image Tag**, enter a Docker tag to associate with your image in the repository.

g. Choose **Next**.

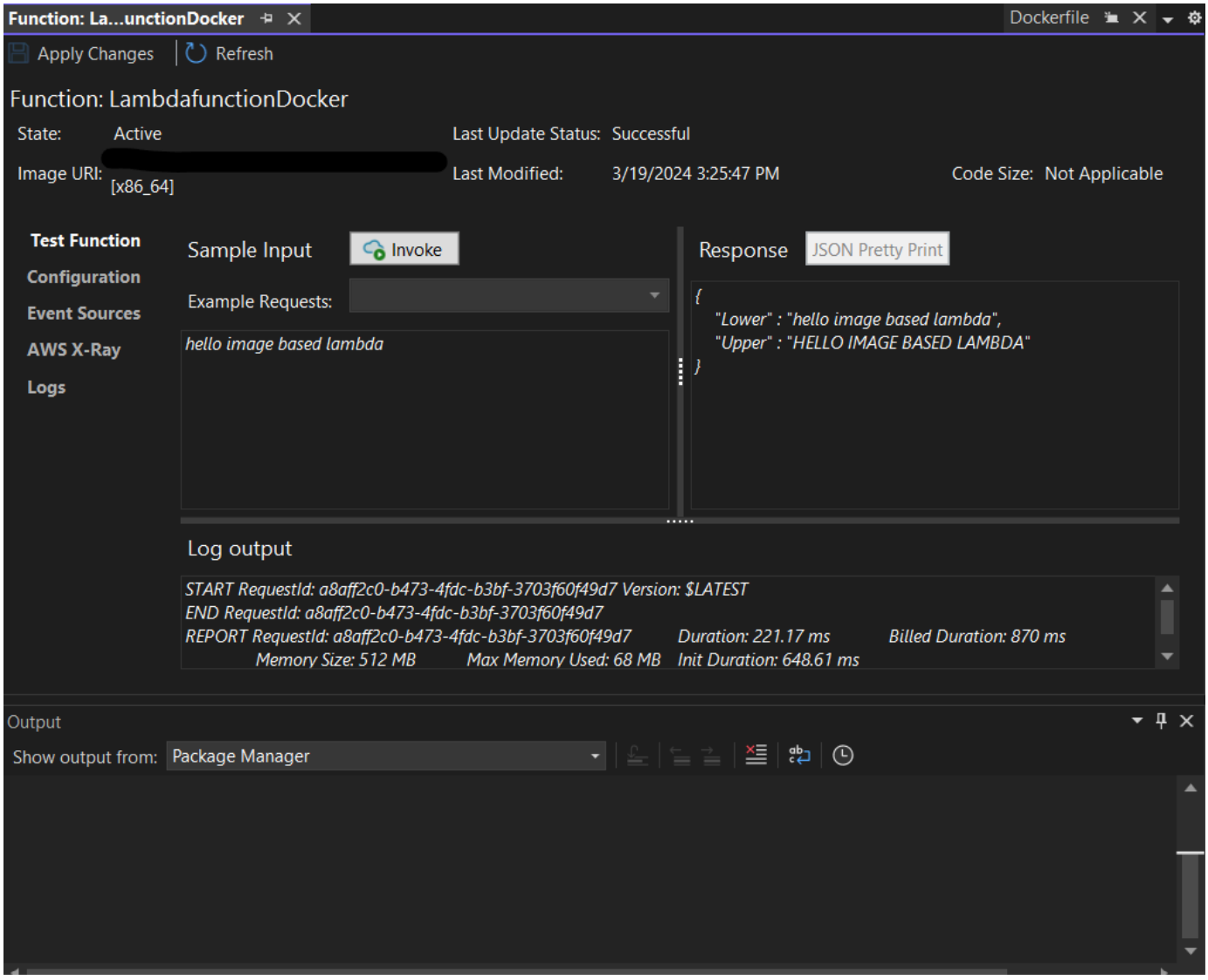
3. On the **Advanced Function Details** page, in **Role Name** choose a role associated with your account. The role is used to provide temporary credentials for any Amazon Web Services calls made by the code in the function. If you do not have a role, choose **New Role based on Amazon Managed Policy** and then choose **AWSLambdaBasicExecutionRole**.

> **Note**
>
> Your account must have permission to run the IAM ListPolicies action, or the **Role Name** list will be empty.

4. Choose **Upload** to start the uploading and publishing processes.

> **Note**
>
> The **Uploading Function** page displays while the function is uploading. The publish process then builds the image based on the configuration parameters, creates the Amazon ECR repository if necessary, uploads the image into the repository, and creates the Lambda referencing that repo with that image.
> After the function is uploaded, the **Function** page opens and displays your new Lambda function's configuration.

5. To manually invoke the Lambda function, on the **Test Function** tab, enter `hello image based lambda` into the request free-text input field and then choose **Invoke**. Your text, converted to uppercase, will appear in **Response**.



6. To view the repository, in the **Amazon Explorer**, under **Amazon Elastic Container Service**, choose **Repositories**.

   You can reopen the **Function:** view at any time by double-clicking on your deployed instance located in the **Amazon Explorer** under the **Amazon Lambda** node.

   > **Note**
   >
   > If your Amazon Explorer window is not open, you can dock it via **View** -> **Amazon Explorer**

7. Note additional image-specific configuration options on the **Configuration** tab. This tab provides a way to override the `ENTRYPOINT`, `CMD`, and `WORKDIR` that may have been specified within the Dockerfile. **Description** is the description you entered (if any) during upload/publish.

Clean-up

If you are not going to continue developing with this example, remember to delete the function and ECR image that was deployed so that you do not get billed for unused resources in your account.

- Functions can be deleted by right-clicking your deployed instance located in the **Amazon Explorer** under the **Amazon Lambda** node.
- Repositories can be deleted in the **Amazon Explorer** under the **Amazon Elastic Container Service** -> **Repositories**.

Next Steps

For information about creating and testing Lambda images, see Using Container Images with Lambda.

For information about container image deployment, permissions, and overriding configuration settings, see Configuring Functions.

Tutorial: Build and Test a Serverless Application with Amazon Lambda

You can build a serverless Lambda application by using an Amazon Toolkit for Visual Studio template. The Lambda project templates include one for an **Amazon Serverless Application**, which is the Amazon Toolkit for Visual Studio implementation of the Amazon Serverless Application Model (Amazon SAM). Using this project type you can develop a collection of Amazon Lambda functions and deploy them with any necessary Amazon resources as a whole application, using Amazon CloudFormation to orchestrate the deployment.

For prerequisites and information about setting up the Amazon Toolkit for Visual Studio, see Using the Amazon Lambda Templates in the Amazon Toolkit for Visual Studio.

Topics

- Create a New Amazon Serverless Application Project
- Reviewing the Serverless Application files
- Deploying the Serverless Application
- Test the Serverless Application

Create a New Amazon Serverless Application Project

Amazon Serverless Application projects create Lambda functions with a serverless Amazon CloudFormation template. Amazon CloudFormation templates enable you to define additional resources such as databases, add IAM roles, and deploy multiple functions at one time. This differs from Amazon Lambda projects, which focus on developing and deploying a single Lambda function.

The following procedure describes how to create a new Amazon Serverless Application Project.

1. From Visual Studio expand the **File** menu, expand **New**, then choose **Project**.
2. In the **New Project** dialog box, ensure that the **Language**, **Platform**, and **Project type** drop-down boxes are set to "All ..." and enter **`aws lambda`** in the **Search** field.
3. Select the **Amazon Serverless Application with Tests (.NET Core - C#)** template.

> **Note**
>
> It's possible that the **Amazon Serverless Application with Tests (.NET Core - C#)** template may not populate at the top of the results.

4. Click **Next** to open the **Configure your new project** dialog.
5. From the **Configure your new project** dialog, enter **`ServerlessPowertools`** for the **Name**, then complete the remaining fields to your preference. Choose the **Create** button to proceed to the **Select Blueprint** dialog.
6. From the **Select Blueprint** dialog choose the **Powertools for Amazon Lambda** blueprint, and then choose **Finish** to create the Visual Studio project.

Reviewing the Serverless Application files

The following sections provide a detailed look at three Serverless Application files created for your project:

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

1. serverless.template

A `serverless.template` file is an Amazon CloudFormation template for declaring your Serverless functions and other Amazon resources. The file included with this project contains a declaration for a single Lambda function that will be exposed through the Amazon API Gateway as an HTTP `*Get*` operation. You can edit this template to customize the existing function or add more functions and other resources that are required by your application.

The following is an example of a `serverless.template` file:

```
{
  "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/"
      }
    }
  }
}
```

Notice that many of the `...AWS:: Serverless::Function...` declaration fields are similar to the fields of a Lambda project deployment. Powertools Logging, Metrics and Tracing are configured through the following 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

For definitions and additional details about the environment variables, see the Powertools for Amazon Lambda references website.

2. Functions.cs

`Functions.cs` is a class file containing a C# method that's mapped to a single function declared in the template file. The Lambda function responds to `HTTP Get` methods from API Gateway. The following is an example of the `Functions.cs` file:

```
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` provides the default values for the Amazon deployment wizard inside Visual Studio and the Amazon Lambda commands added to the .NET Core CLI. The following is an example of the `aws-lambda-tools-defaults.json` file included with this project:

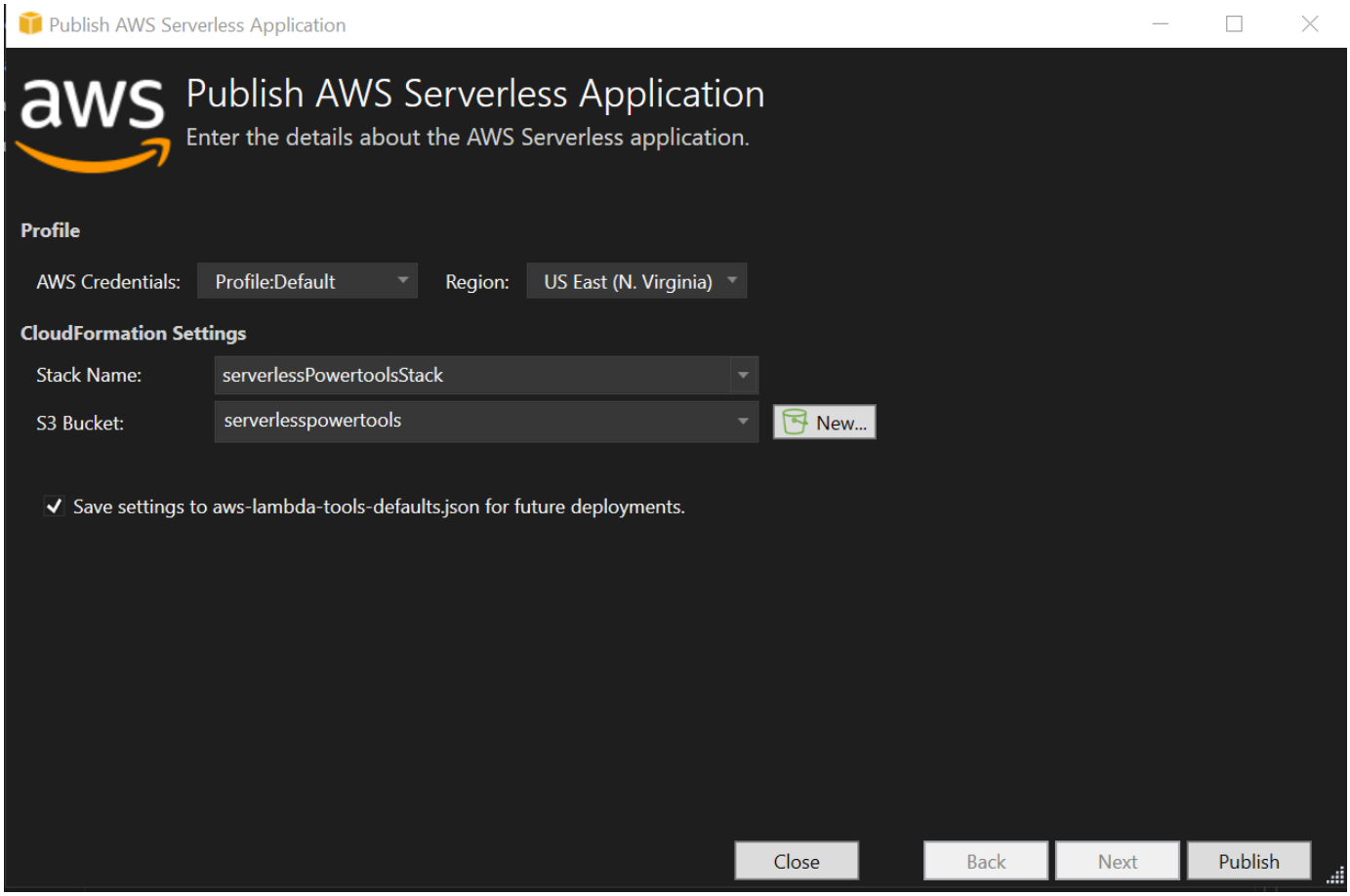
```
{
  "profile": "Default",
  "region": "us-east-1",
  "configuration": "Release",
  "s3-prefix": "ServerlessPowertools/",
  "template": "serverless.template",
  "template-parameters": ""
}
```

Deploying the Serverless Application

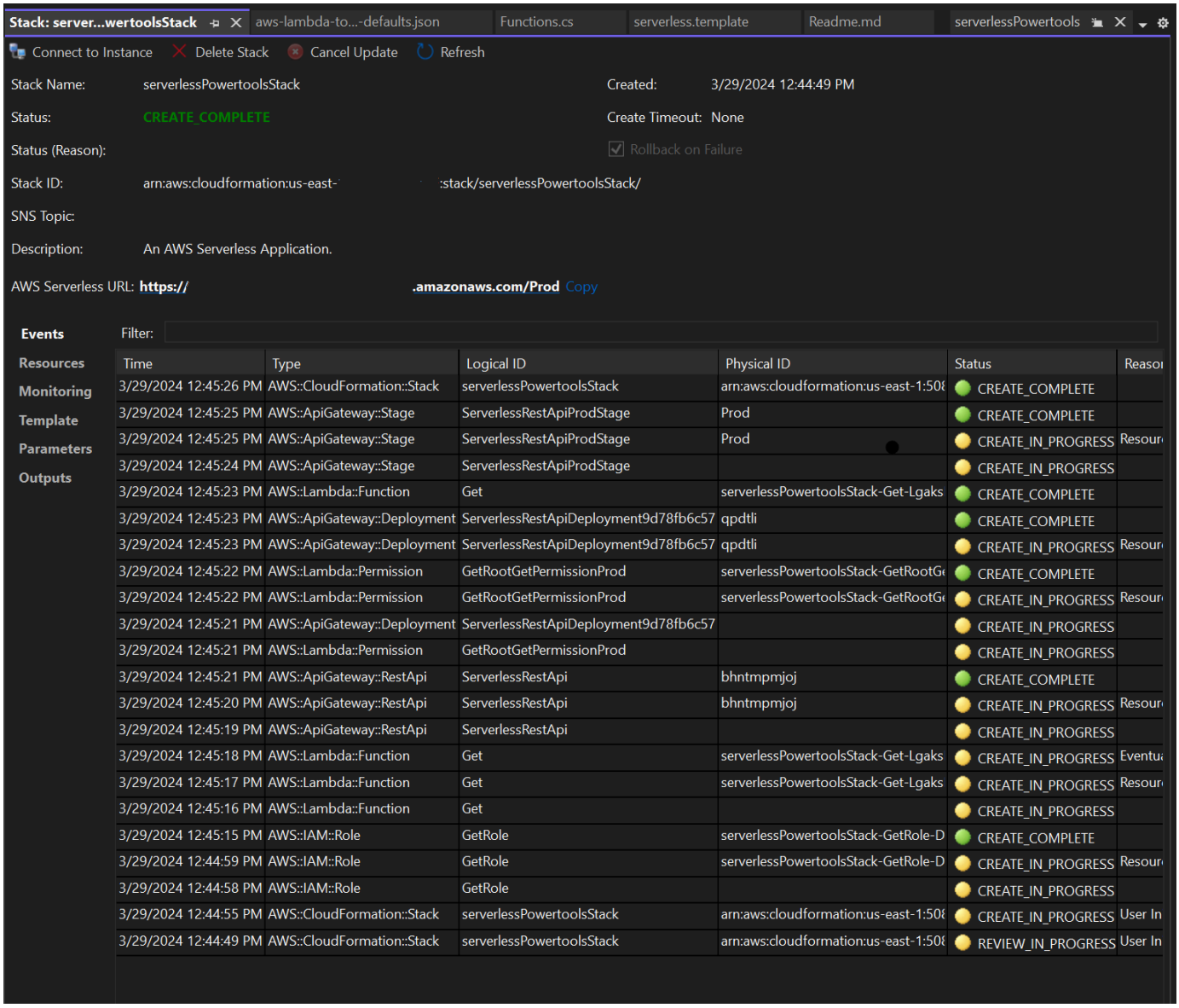
To deploy your serverless application complete the following steps

1. From the **Solution Explorer**, open the context menu for (right click) your project and choose **Publish to Amazon Lambda** to open the **Publish Amazon Serverless Application** dialog.
2. From the **Publish Amazon Serverless Application** dialog, enter a name for the Amazon CloudFormation stack container in the **Stack Name** field.
3. In the **S3 Bucket** field, choose an Amazon S3 bucket that your application bundle will upload to or choose the **New...** button and enter the name of a new Amazon S3 bucket. Then choose **Publish** to publish to deploy your application.

> **Note**
>
> Your Amazon CloudFormation stack and Amazon S3 Bucket must exist in the same Amazon region. The remaining settings for your project are defined in the `serverless.template` file.



4. The **Stack** view window opens during the publishing process, when deployment is complete the **Status** field displays: `CREATE_COMPLETE`.



Test the Serverless Application

When the stack creation is complete, you can view your application using the **Amazon Serverless URL**. If you've completed this tutorial without adding any additional functions or parameters, accessing your Amazon serverless URL displays the following phrase in your web browser: `Hello Powertools for AWS Lambda (.NET)`.

Tutorial: Creating an Amazon Rekognition Lambda Application

This tutorial shows you how to create an Lambda application that uses Amazon Rekognition to tag Amazon S3 objects with detected labels.

For prerequisites and information about setting up the Amazon Toolkit for Visual Studio, see Using the Amazon Lambda Templates in the Amazon Toolkit for Visual Studio.

Create a Visual Studio .NET Core Lambda Image Rekognition Project

The following procedure describes how to create an Amazon Rekognition Lambda application from the Amazon Toolkit for Visual Studio.

> **Note**
>
> Upon creation, your application has a solution with two projects: the source project that contains your Lambda function code to deploy to Lambda, and a test project using xUnit for testing your function locally.
> Sometimes Visual Studio can't find all NuGet references for your projects. This is because blueprints require dependencies that must be retrieved from NuGet. When new projects are created, Visual Studio only pulls in local references and not remote references from NuGet. To fix NuGet errors: right-click your references and choose **Restore Packages**.

1. From Visual Studio expand the **File** menu, expand **New**, then choose **Project**.
2. In the **New Project** dialog box, ensure that the **Language**, **Platform**, and **Project type** drop-down boxes are set to "All ..." and enter **`aws lambda`** in the **Search** field.
3. Select the **Amazon Lambda with Tests (.NET Core - C#)** template.
4. Click **Next** to open the **Configure your new project** dialog.
5. From the **Configure your new project** dialog, enter "ImageRekognition" for the **Name**, then complete the remaining fields to your preference. Choose the **Create** button to proceed to the **Select Blueprint** dialog.
6. From the **Select Blueprint** dialog, choose the **Detect Image Labels** blueprint, then choose **Finish** to create the Visual Studio project.

> **Note**
>
> This blueprint provides code for listening to Amazon S3 events and uses Amazon Rekognition to detect labels and add them to the S3 object as tags.

Reviewing Project Files

The following sections examine these project files:

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

1. Function.cs

Inside the `Function.cs` file, the first segment of code is the assembly attribute, located at the top of the file. By default, Lambda only accepts input parameters and return types of type `System.IO.Stream`. You must register a serializer to use typed classes for input parameters and return types. The assembly attribute registers the Lambda JSON serializer, which uses `Newtonsoft.Json` to convert streams to typed classes. You can set the serializer at the assembly or method level.

The following is an example of the assembly attribute:

```
// 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
```

The class has two constructors. The first is a default constructor that is used when Lambda invokes your function. This constructor creates the Amazon S3 and Amazon Rekognition service clients. The constructor also retrieves the Amazon credentials for these clients from the IAM role you assign to the function when you deploy it. The Amazon Region for the clients is set to the region your Lambda function is running in. In this blueprint, you only want to add tags to the Amazon S3 object if the Amazon Rekognition service has a minimum level of confidence about the label. This constructor checks the environment variable `MinConfidence` to determine the acceptable confidence level. You can set this environment variable when you deploy the Lambda function.

The following is an example of the first class constructor in `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}");
    }
}
```

The following example demonstrates how the second constructor can be utilized for testing. The test project configures its own S3 and Rekognition clients and passes them in:

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

The following is an example of the `FunctionHandler` method inside the `Function.cs` file.

```
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` is the method Lambda calls after it constructs the instance. Notice that the input parameter is of type `S3Event` and not a `Stream`. You can do this because of the registered Lambda JSON serializer. The `S3Event` contains all the information about the event triggered in Amazon S3. The function loops through all the S3 objects that were part of the event and tells Rekognition to detect labels. After the labels are detected, they are added as tags to the S3 object.

> **Note**
>
> The code contains calls to `Console.WriteLine()`. When the function is running in Lambda, all calls to `Console.WriteLine()` redirect to Amazon CloudWatch Logs.

2. aws-lambda-tools-defaults.json

The `aws-lambda-tools-defaults.json` file contains default values that the blueprint has set to prepopulate some of the fields in the deployment wizard. It's also helpful in setting command-line options for integration with the .NET Core CLI.

To access the .NET Core CLI integration, navigate to the function's project directory and type **`dotnet lambda help`**.

> **Note**
>
> The function handler indicates what method for Lambda to call in response to the invoked function. The format of this field is: `<assembly-name>::<full-type-name>::<method-name>`. The namespace must be included with the type name.

Deploy the Function

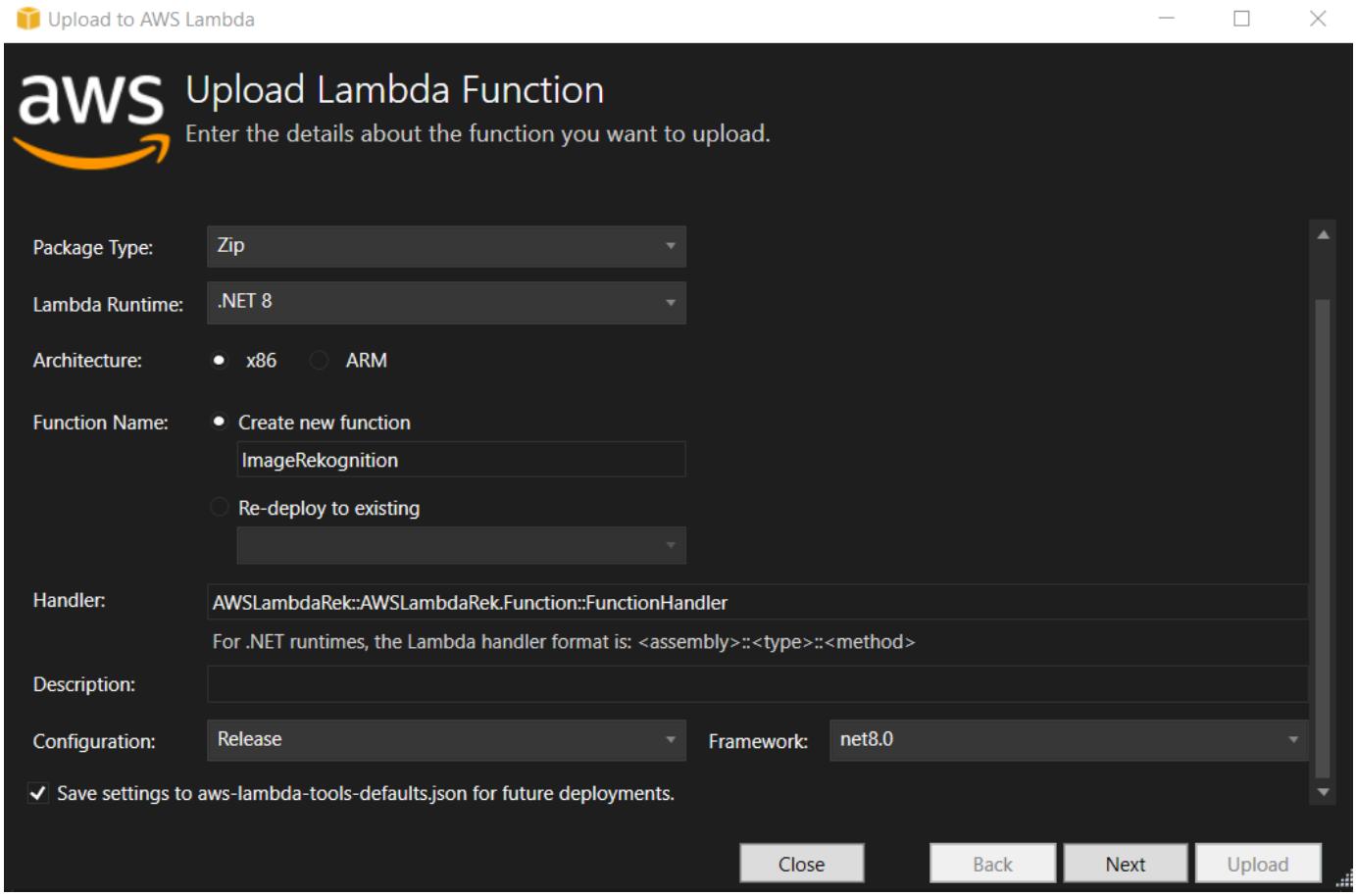
The following procedure describes how to deploy your Lambda function.

1. From the **Solution Explorer**, right-click the Lambda project and choose **Publish to Amazon Lambda** to open the **Upload to Amazon Lambda** window.

> **Note**
>
> The preset values are retrieved from the `aws-lambda-tools-defaults.json` file.

2. From the **Upload to Amazon Lambda** window, enter a name into the **Function Name** field, then choose the **Next** button to advance to the **Advanced Function Details** window.

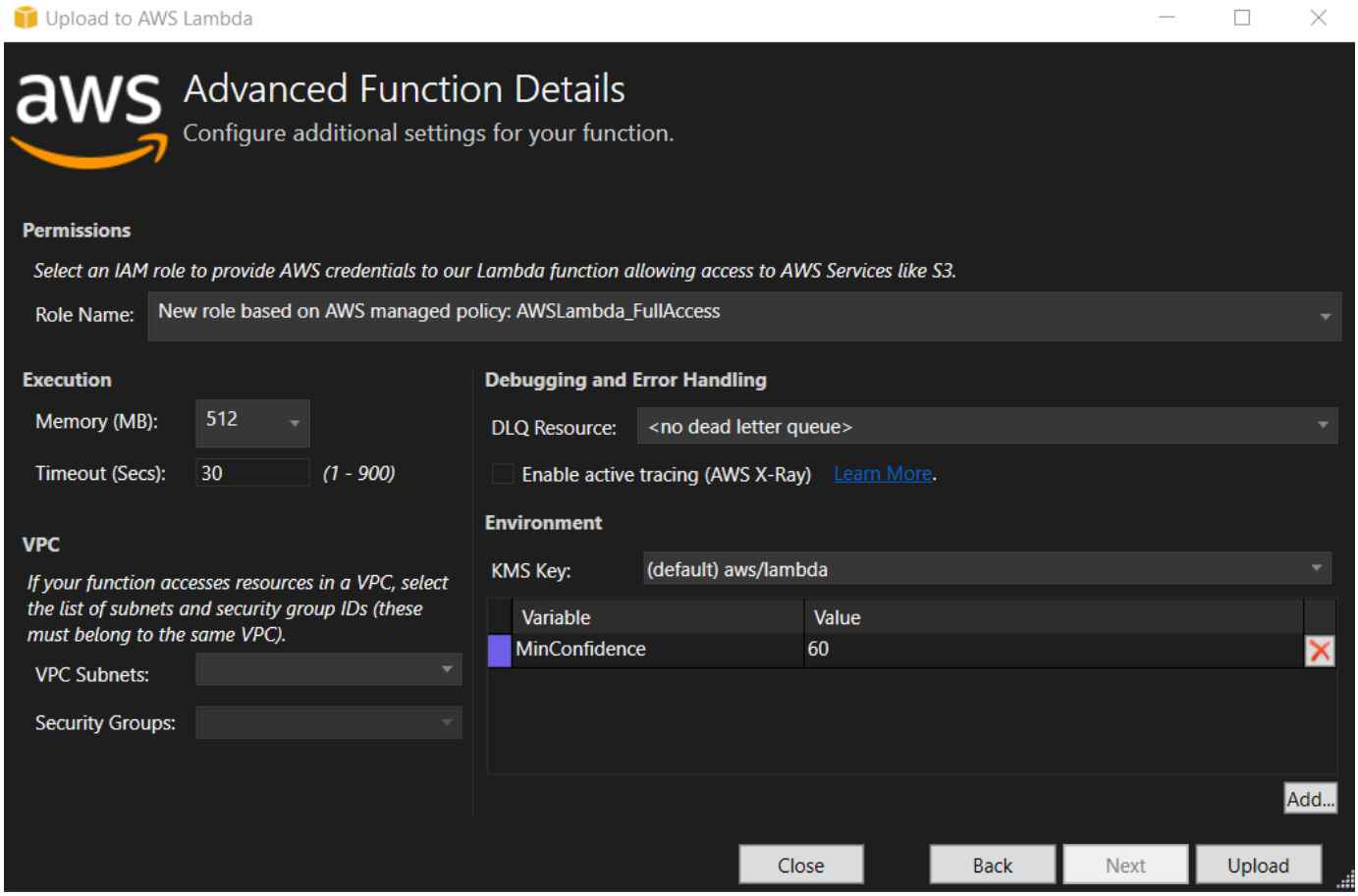
> **Note**
>
> This example, uses the **Function Name `ImageRekognition`**.



3. From the **Advanced Function Details** window, select an IAM role that gives permission for your code to access your Amazon S3 and Amazon Rekognition resources.

> **Note**
>
> If you're following along with this example, select the `AWSLambda_FullAccess` role.

4. Set the environment variable `MinConfidence` to 60, then choose **Upload** to launch the deployment process. The publishing process is complete when the **Function** view displays in the **Amazon Explorer**.



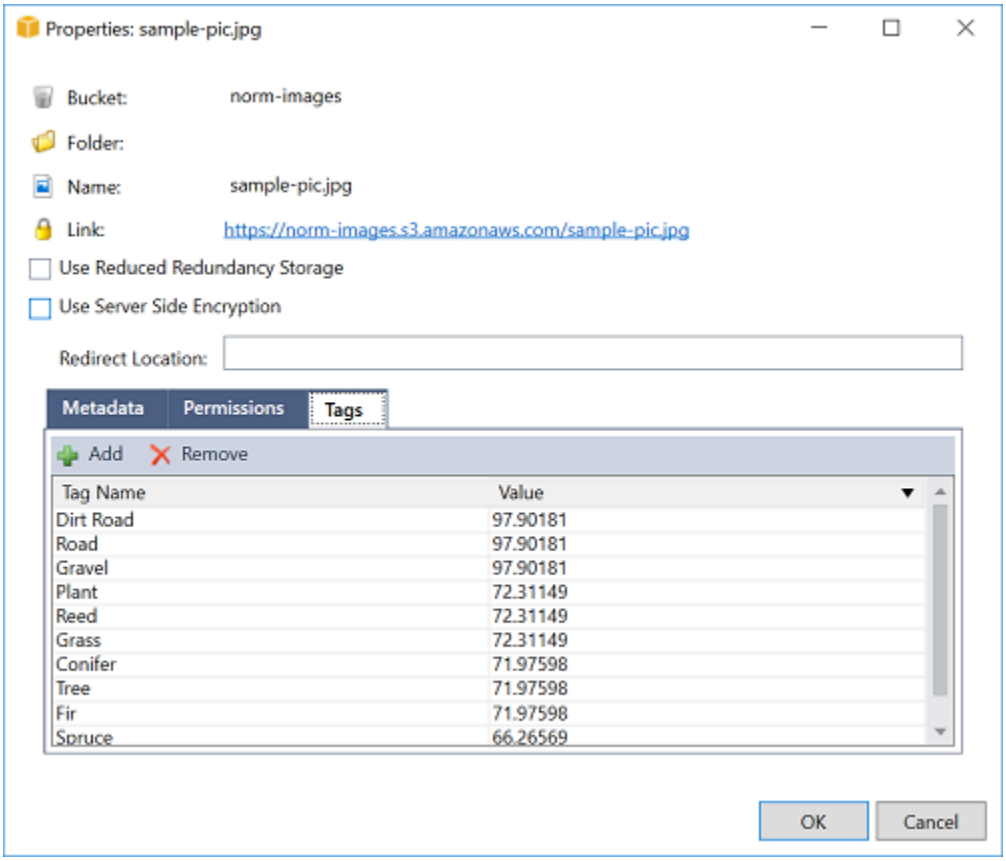
5. Following a successful deployment, configure Amazon S3 to send its events to your new function by navigating to the **Event Sources** tab.
6. From the **Event Sources** tab, choose the **Add** button, then select the Amazon S3 bucket to connect with your Lambda function.

> **Note**
>
> The bucket must be in the same Amazon region as your Lambda function.

Test the Function

Now that the function is deployed and an S3 bucket is configured as an event source for it, open the S3 bucket browser from the **Amazon Explorer** for the bucket you selected. Then upload some images.

When the upload is complete, you can confirm that your function ran by looking at the logs from your function view. Or, right-click the images in the bucket browser and choose **Properties**. On the **Tags** tab, you can view the tags that were applied to your object.



Tutorial: Using Amazon Logging Frameworks with Amazon Lambda to Create Application Logs

You can use Amazon CloudWatch Logs to monitor, store, and access your application's logs. To get log data into CloudWatch Logs, use an Amazon SDK or install the CloudWatch Logs agent to monitor certain log folders. CloudWatch Logs is integrated with several popular .NET logging frameworks, simplifying work flows.

To get started working with CloudWatch Logs and .NET logging frameworks, add the appropriate NuGet package and CloudWatch Logs output source to your application, then use your logging

library as you normally would. This enables your application to log messages with your .NET framework, sending them to CloudWatch Logs, displaying your application's log messages in the CloudWatch Logs console. You can also set up metrics and alarms from the CloudWatch Logs console, based on your application's log messages.

Supported .NET logging frameworks include:

- **NLog**: To view, see the nuget.org NLog package.
- **Log4net**: To view, see the nuget.org Log4net package.
- **ASP.NET Core logging Framework**: To view, see the nuget.org ASP.NET Core logging Framework package.

The following is an example of an `NLog.config` file that enables both CloudWatch Logs and the console as output for log messages by adding the `AWS.Logger.NLog` NuGet package, and Amazon target into `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>
```

The logging plugins are all built on top of the Amazon SDK for .NET and authenticate your Amazon credentials in a process similar to the SDK. The following example details permissions required by the logging plugin credentials to access CloudWatch Logs:

Note

The Amazon .NET logging plugins are an open source project. For additional information, samples, and instructions, see the samples and instructions topics in the Amazon Logging .NET GitHub repository.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups"
      ],
      "Resource": [
        "arn:aws:logs:*:*:*"
      ]
    }
  ]
}
```

Deploying to Amazon

The Toolkit for Visual Studio supports application deployment to Amazon Elastic Beanstalk containers or Amazon CloudFormation stacks.

Note

If you are using Visual Studio Express Edition:

- You can use the Docker CLI to deploy applications to Amazon ECS containers.
- You can use the Amazon Management Console to deploy applications to Elastic Beanstalk containers.

For Elastic Beanstalk deployments, you must first create a web deployment package. For more information, see How to: Create a Web Deployment Package in Visual Studio. For Amazon ECS deployment, you must have a Docker image. For more information, see Visual Studio Tools for Docker.

Topics

- Working with Publish to Amazon in Visual Studio
- Deploying an Amazon Lambda Project with the .NET Core CLI
- Deploying to Amazon Elastic Beanstalk in Visual Studio using Amazon Toolkit for Visual Studio with Amazon Q
- Deploying to Amazon EC2 Container Service

Working with Publish to Amazon in Visual Studio

Publish to Amazon is an interactive deployment experience that assists you with publishing your .NET applications to Amazon deployment targets, supporting applications targeting .NET Core 3.1 and later. Working with Publish to Amazon keeps your work flow inside of Visual Studio by making these deployment features available, directly from your IDE:

- The ability to deploy your application with a single click.
- Deployment recommendations based on your application.

- Automatic Dockerfile creation, as is relevant and required by your deployment destination's environment (deployment target).
- Optimized settings for building and packaging your applications, as required by your deployment target.

> **Note**
>
> For additional information about publishing .NET Framework applications, see the guide Creating and deploying .NET applications on Elastic Beanstalk
> You can also access Publish to Amazon from the .NET CLI. For more information, see the Deploy .NET applications on Amazon guide.

Topics

- Prerequisites
- Supported application types
- Publishing applications to Amazon targets

Prerequisites

To successfully publish .NET applications to an Amazon service, install the following to your local device:

- .NET Core 3.1+(which includes .NET5 and .NET6): For additional information about these products and download information, visit the Microsoft download site.
- Node.js 14.x or later version: Node.js is required to run Amazon Cloud Development Kit (Amazon CDK). To download or obtain more information about Node.js, visit the Node.js download site.

> **Note**
>
> Publish to Amazon utilizes Amazon CDK to deploy your application and all of its deployment infrastructure as a single project. For more information about Amazon CDK see the Cloud Development Kit guide.

- (Optional) Docker is used when deploying to a container-based service such as Amazon ECS. For more information and to download Docker, see the Docker download site.

Supported application types

Before publishing to a new or exiting target, begin by creating or opening one of the following project types in Visual Studio:

- ASP.NET Core application
- .NET Console application
- Blazor WebAssembly application

Publishing applications to Amazon targets

When publishing to a new target, Publish to Amazon will guide you through the process by making recommendations and using common settings. If you need to publish to a target that was set up previously, your preferences are stored and can be adjusted, or are immediately available for one-click deployment.

> **Note**
>
> **Toolkits integration with the .NET CLI Server:**
> Publishing launches a .NET server process on the localhost to perform the publication process.

Publish to a new target

The following describes how to configure your Publish to Amazon deployment preferences, when you're publishing to a new target.

1. From the **Amazon Explorer**, expand the **Credentials** drop-down menu, then choose the Amazon profile that corresponds with the region and Amazon services that are required for your deployment.
2. Expand the **Region** drop-down menu, then choose the Amazon region that contains the Amazon services that are necessary for your deployment.
3. From the Visual Studio **Solutions Explorer** pane, open the context menu for (right-click) the project's name, and choose **Publish to Amazon**. This will open **Publish to Amazon**.
4. From **Publish to Amazon**, choose **Publish to New Target** to configure a new deployment.

> **Note**
>
> To modify your default deployment credentials, choose or click the **Edit** link located next to the **Credentials** section, in **Publish to Amazon**.
> To bypass the target configuration process, choose **Publish to Existing Target**, then pick your preferred configuration from the list of your previous deployment targets.

5. From the **Publish Targets** pane, choose an Amazon service to manage your application deployment.
6. When you are satisfied with your configuration, choose **Publish** to start the deployment process.

> **Note**
>
> After initiating a deployment, **Publish to Amazon** displays the following status updates:
>
> - During the deployment process, **Publish to Amazon** displays information about the deployment's progress.
> - Following the deployment process, **Publish to Amazon** indicates if the deployment succeeded or failed.
> - After a successful deployment, the **Resources** panel offers additional information about the resource that was created. This information will vary depending on the type of application and deployment configuration.

Publish to an existing target

The following describes how to republish your .NET application to an existing Amazon target.

1. From the **Amazon Explorer**, expand the **Credentials** drop-down menu, then choose the Amazon profile that corresponds with the region and Amazon services that are required for your deployment.
2. Expand the **Region** drop-down menu, then choose the Amazon region that contains the Amazon services that are necessary for your deployment.

3. From the Visual Studio **Solutions Explorer** pane, right-click the project's name and choose **Publish to Amazon** to open **Publish to Amazon**.
4. From **Publish to Amazon**, choose **Publish to Existing Target** to select your deployment environment from a list of existing targets.

> **Note**
>
> If you have recently published any applications to the Amazon Cloud, those applications are displayed in Publish to Amazon.

5. Select the publishing target that you want to deploy your application to, then click **Publish** to start the deployment process.

Deploying an Amazon Lambda Project with the .NET Core CLI

The Amazon Toolkit for Visual Studio includes Amazon Lambda .NET Core project templates for Visual Studio. You can deploy Lambda functions built in Visual Studio using the .NET Core command line interface (CLI).

Topics

- Prerequisites
- Related topics
- Listing the Lambda Commands Available through the .NET Core CLI
- Publishing a .NET Core Lambda Project from the .NET Core CLI

Prerequisites

Before working with .NET Core CLI to deploy Lambda functions, you must meet the following prerequisites:

- Be sure Visual Studio 2015 Update 3 is installed.
- Install .NET Core for Windows.
- Set up the .NET Core CLI to work with Lambda. For more information, see .NET Core CLI in the *Amazon Lambda Developer Guide*.

- Install the Toolkit for Visual Studio. For more information, see Installing the Amazon Toolkit for Visual Studio.

Related topics

The following related topics can be helpful as you use the .NET Core CLI to deploy Lambda functions:

- For more information about Lambda functions, see What is Amazon Lambda? in the *Amazon Lambda Developer Guide*.
- For information about creating Lambda functions in Visual Studio, see Amazon Lambda.
- For more information about Microsoft .NET Core, see .NET Core in Microsoft's online documentation.

Listing the Lambda Commands Available through the .NET Core CLI

To list the Lambda commands that are available through the .NET Core CLI, do the following.

1. Open a command prompt window, and navigate to the folder containing a Visual Studio .NET Core Lambda project.
2. Enter `dotnet lambda --help`.

```
C:\Lambda\AWSLambda1\AWSLambda1>dotnet lambda --help
    Amazon 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 Amazon serverless applications using Amazon
CloudFormation:
   .
           deploy-serverless       Deploy an Amazon serverless application
           list-serverless         List all of your Amazon serverless applications
           delete-serverless       Delete an Amazon 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>
```

Publishing a .NET Core Lambda Project from the .NET Core CLI

The following instructions assume you've created an Amazon Lambda .NET Core function in Visual Studio.

1. Open a command prompt window, and navigate to the folder containing your Visual Studio .NET Core Lambda project.
2. Enter `dotnet lambda deploy-function`.
3. When prompted, enter the name of the function to deploy. It can be a new name or the name of an existing function.
4. When prompted, enter the Amazon Region (the Region to which your Lambda function will be deployed).
5. When prompted, select or create the IAM role that Lambda will assume when executing the function.

On successful completion, the message **New Lambda function created** is displayed.

```
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: (Amazon Lambda function name)
DotNetCoreLambdaTest
Enter Amazon Region: (The region to connect to Amazon 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
```

If you deploy an existing function, the deploy function asks only for the Amazon Region.

```
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: (Amazon Lambda function name)
DotNetCoreLambdaTest
Enter Amazon Region: (The region to connect to Amazon services)
us-west-2
```

```
Updating code for existing function
```

After your Lambda function is deployed, it's ready to use. For more information, see Examples of How to Use Amazon Lambda.

Lambda automatically monitors Lambda functions for you, reporting metrics through Amazon CloudWatch. To monitor and troubleshoot your Lambda function, see Troubleshooting and Monitoring Amazon Lambda Functions with Amazon CloudWatch.

Deploying to Amazon Elastic Beanstalk in Visual Studio using Amazon Toolkit for Visual Studio with Amazon Q

Amazon Elastic Beanstalk is a service that simplifies the process of provisioning Amazon resources for your application. Elastic Beanstalk provides all of the Amazon infrastructure required to deploy your application. This infrastructure includes:

- Amazon EC2 instances that host the executables and content for your application.
- An Auto Scaling group to maintain the appropriate number of Amazon EC2 instances to support your application.
- An Elastic Load Balancing load balancer that routes incoming traffic to the Amazon EC2 instance with the most bandwidth.

This user guide topic describes how to work with the Elastic Beanstalk wizard in the Amazon Toolkit with Amazon Q. For detailed information specific to Elastic Beanstalk, see the Amazon Elastic Beanstalk Developer Guide. The Elastic Beanstalk wizard for the Amazon Toolkit with Amazon Q is described in the following topic sections.

Topics

- Deploy a Traditional ASP.NET Application to Elastic Beanstalk
- Deploying an ASP.NET Core Application to Elastic Beanstalk (Legacy)
- How to Specify the Amazon Security Credentials for Your Application
- How to Republish Your Application to an Elastic Beanstalk Environment (Legacy)
- Custom Elastic Beanstalk Application Deployments
- Custom ASP.NET Core Elastic Beanstalk Deployments
- Multiple Application Support for .NET and Elastic Beanstalk

Deploy a Traditional ASP.NET Application to Elastic Beanstalk

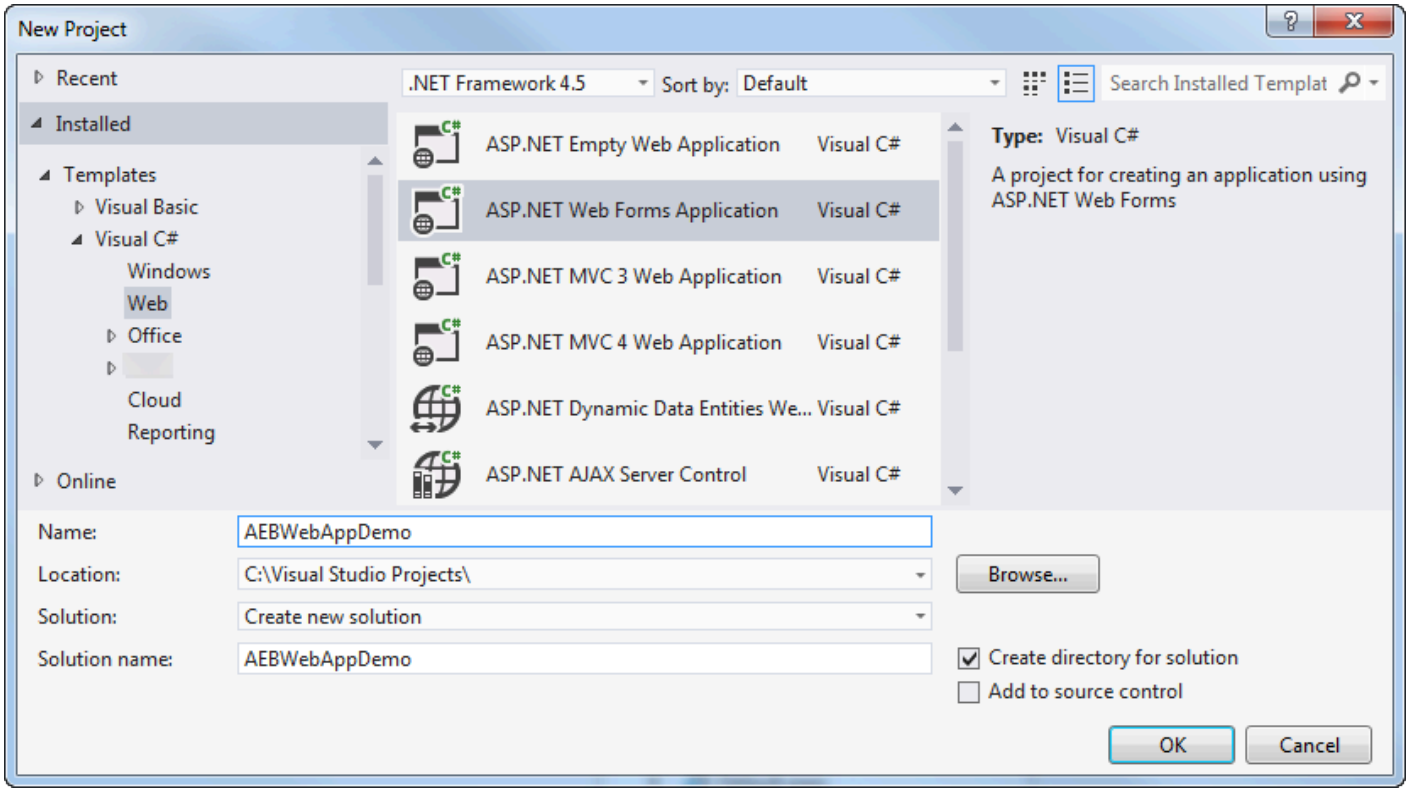
This section describes how to use the **Publish to Elastic Beanstalk** wizard, provided as part of the Toolkit for Visual Studio, to deploy an application through Elastic Beanstalk. To practice, you can use an instance of a web application starter project that is built in to Visual Studio or you can use your own project.

> **Note**
>
> The wizard also supports deploying ASP.NET Core applications. For information about ASP.NET Core, see the Amazon .NET deployment tool guide and the updated Deploying to Amazon table of contents.

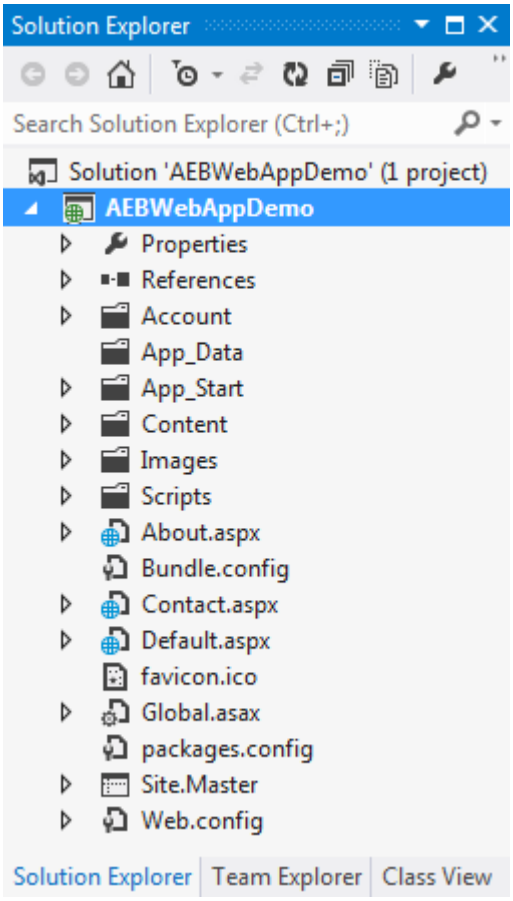
> **Note**
>
> Before you can use the **Publish to Elastic Beanstalk** wizard, you must download and install Web Deploy. The wizard relies on Web Deploy to deploy web applications and websites to Internet Information Services (IIS) web servers.

To create a sample web application starter project

1. In Visual Studio, from the **File** menu, choose **New**, and then choose **Project**.
2. In the navigation pane of the **New Project** dialog box, expand **Installed**, expand **Templates**, expand **Visual C#**, and then choose **Web**.
3. In the list of web project templates, choose any template containing the words `Web` and `Application` in its description. For this example, choose **ASP.NET Web Forms Application**.

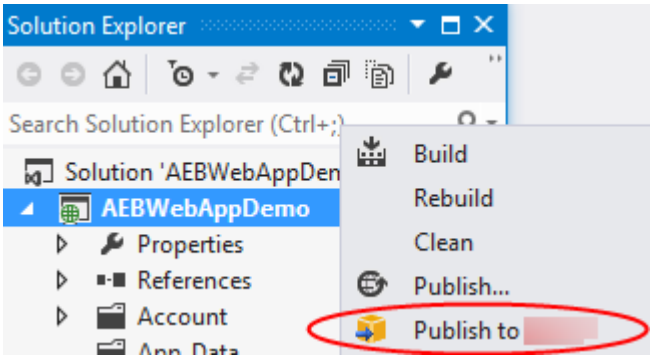


4. In the **Name** box, type `AEBWebAppDemo`.
5. In the **Location** box, type the path to a solution folder on your development machine or choose **Browse**, and then browse to and choose a solution folder, and choose **Select Folder**.
6. Confirm the **Create directory for solution** box is selected. In the **Solution** drop-down list, confirm **Create new solution** is selected, and then choose **OK**. Visual Studio will create a solution and project based on the ASP.NET Web Forms Application project template. Visual Studio will then display Solution Explorer where the new solution and project appear.

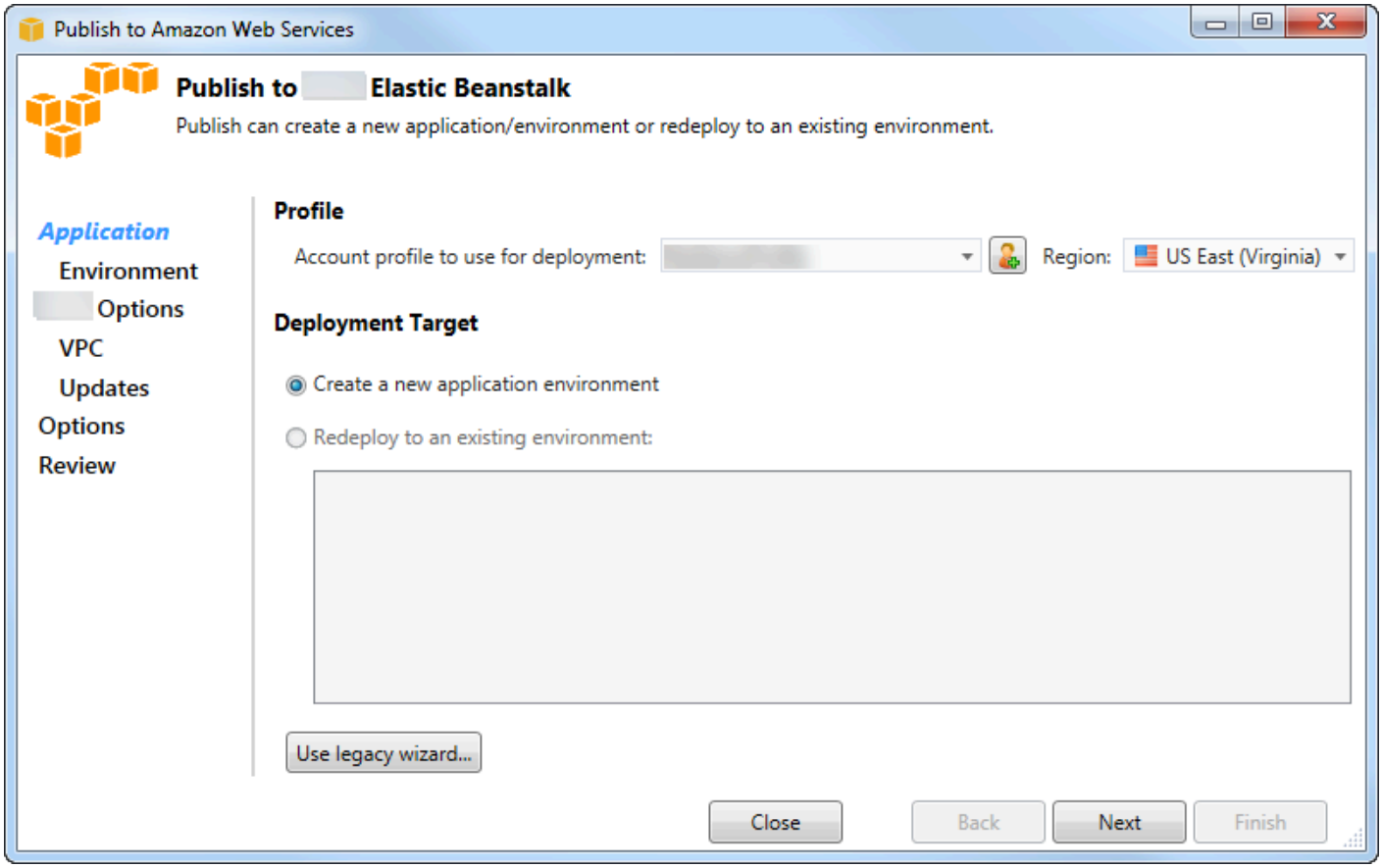


To deploy an application by using the Publish to Elastic Beanstalk wizard

1. In Solution Explorer, open the context (right-click) menu for the **AEBWebAppDemo** project folder for the project you created in the previous section, or open the context menu for the project folder for your own application, and choose **Publish to Amazon Elastic Beanstalk**.



   The **Publish to Elastic Beanstalk** wizard appears.



2. In **Profile**, from the **Account profile to use for deployment** drop-down list, choose the Amazon account profile you want to use for the deployment.

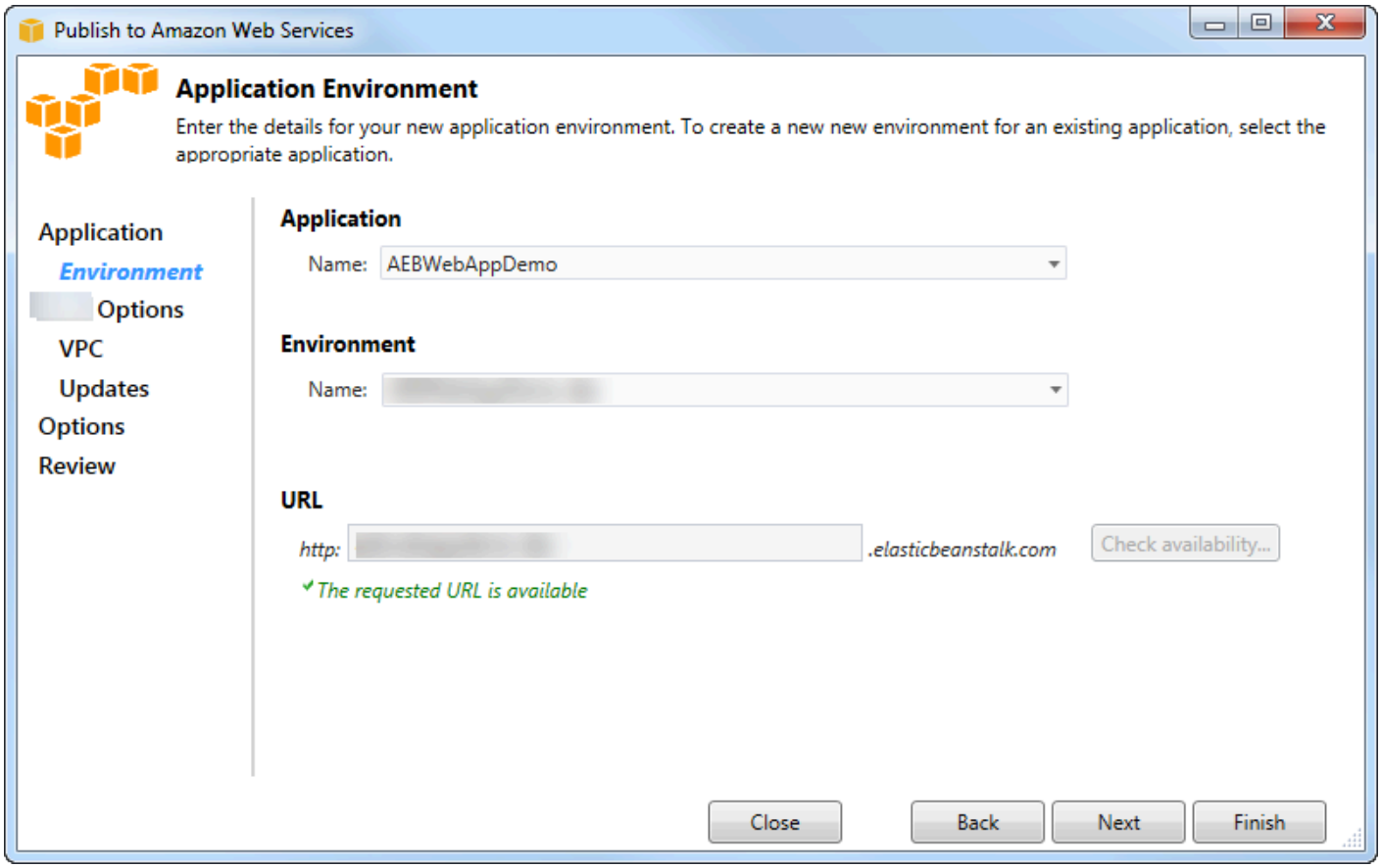
   Optionally, if you have an Amazon account you want to use, but you haven't yet created an Amazon account profile for it, you can choose the button with the plus symbol (+) to add an Amazon account profile.

3. From the **Region** drop-down list, choose the region to which you want Elastic Beanstalk to deploy the application.

4. In **Deployment Target**, you can choose either **Create a new application environment** to perform an initial deployment of an application or **Redeploy to an existing environment** to redeploy a previously deployed application. (The previous deployments may have been performed with either the wizard or the deprecated Standalone Deployment Tool.) If you choose **Redeploy to an existing environment**, there may be a delay while the wizard retrieves information from previous deployments that are currently running.

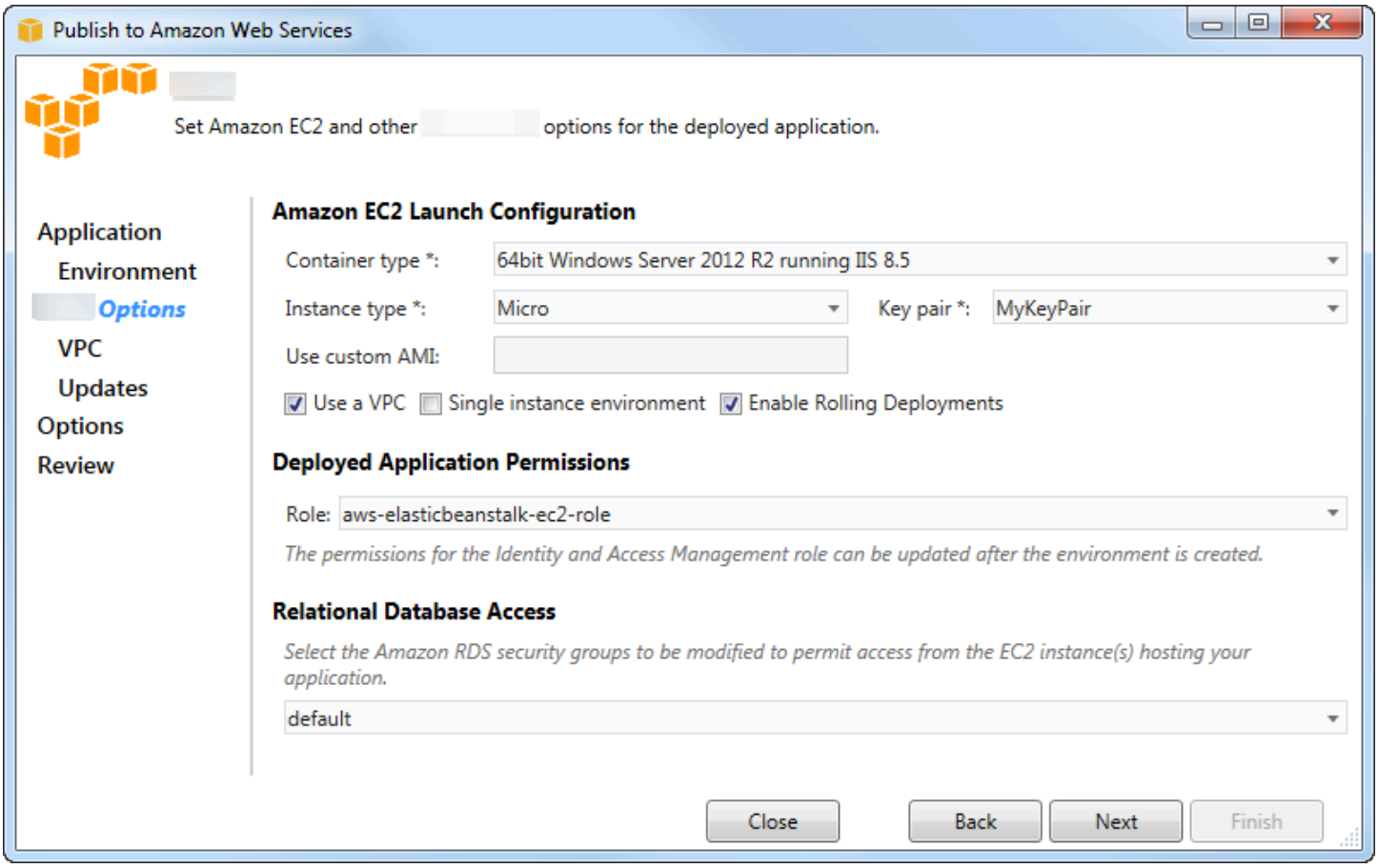
> **Note**
>
> If you choose **Redeploy to an existing environment**, choose an environment in the list, and then choose **Next**, the wizard will take you directly to the **Application Options** page. If you go this route, skip ahead to the instructions later in this section that describe how to use the **Application Options** page.

5. Choose **Next**.



6. On the **Application Environment** page, in the **Application** area, the **Name** drop-down list proposes a default name for the application. You can change the default name by choosing a different name from the drop-down list.
7. In the **Environment** area, in the **Name** drop-down list, type a name for your Elastic Beanstalk environment. In this context, the term *environment* refers to the infrastructure Elastic Beanstalk provisions for your application. A default name may already be proposed in this drop-down list. If a default name is not already proposed, you can type one or choose one from the drop-down list, if any additional names are available. The environment name cannot be longer than 23 characters.

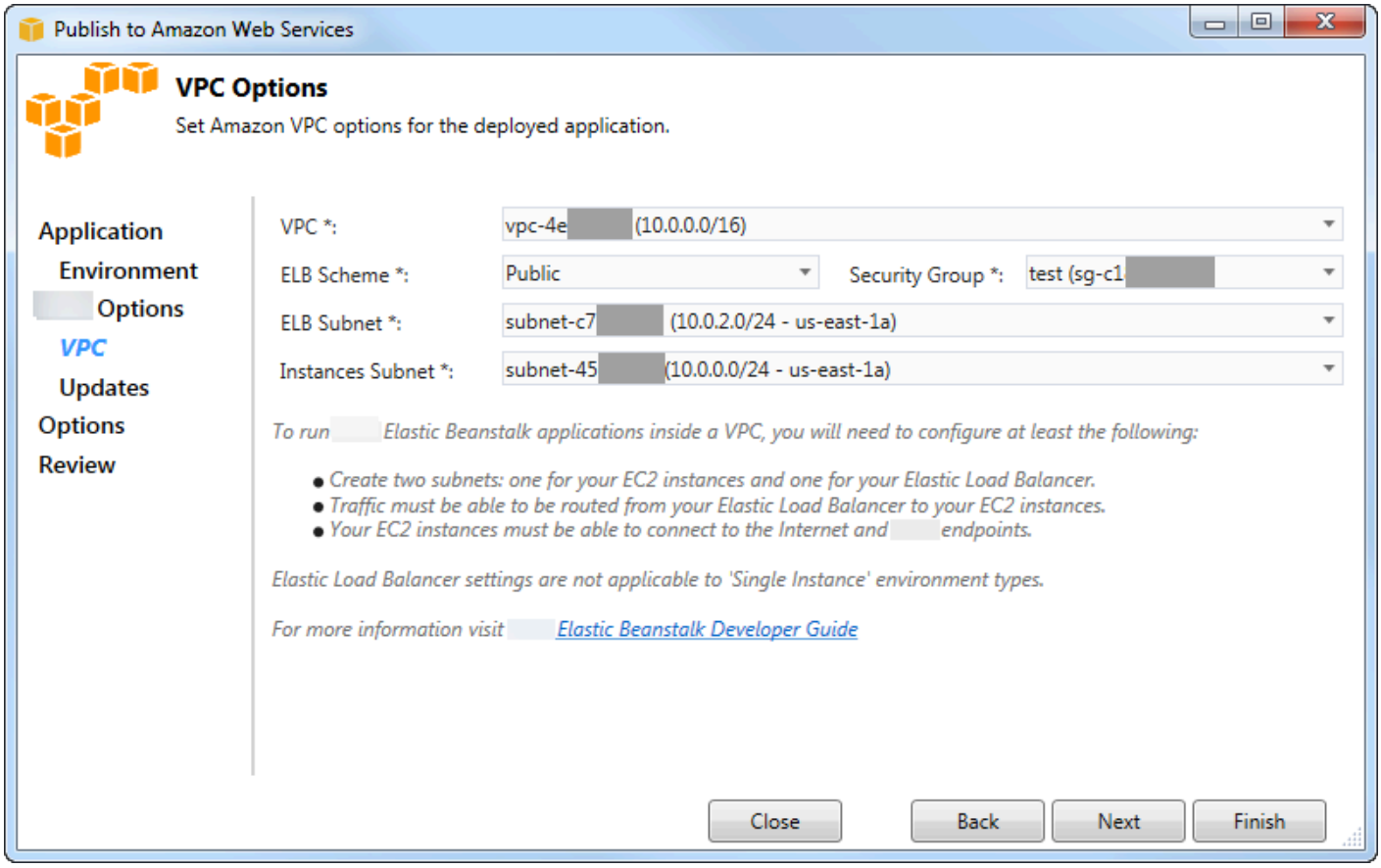
8. In the **URL** area, the box proposes a default subdomain of `.elasticbeanstalk.com` that will be the URL for your web application. You can change the default subdomain by typing a new subdomain name.
9. Choose **Check availability** to make sure the URL for your web application is not already in use.
10. If the URL for your web application is okay to use, choose **Next**.



1. On the **Amazon Options** page, in **Amazon EC2 Launch Configuration**, from the **Container type** drop-down list, choose an Amazon Machine Image (AMI) type that will be used for your application.
2. In the **Instance type** drop-down list, specify an Amazon EC2 instance type to use. For this example, we recommend you use **Micro**. This will minimize the cost associated with running the instance. For more information about Amazon EC2 costs, go to the EC2 Pricing page.
3. In the **Key pair** drop-down list, choose an Amazon EC2 instance key pair to use to sign in to the instances that will be used for your application.

4. Optionally, in the **Use custom AMI** box, you can specify a custom AMI that will override the AMI specified in the **Container type** drop-down list. For more information about how to create a custom AMI, go to Using Custom AMIs in the Amazon Elastic Beanstalk Developer Guide and Create an AMI from an Amazon EC2 Instance.
5. Optionally, if you want to launch your instances in a VPC, select the **Use a VPC** box.
6. Optionally, if you want to launch a single Amazon EC2 instance and then deploy your application to it, select the **Single instance environment** box.

   If you select this box, Elastic Beanstalk will still create an Auto Scaling group, but will not configure it. If you want to configure the Auto Scaling group later, you can use the Amazon Web Services Management Console.
7. Optionally, if you want to control the conditions under which your application is deployed to the instances, select the **Enable Rolling Deployments** box. You can select this box only if you have not selected the **Single instance environment** box.
8. If your application uses Amazon services such as Amazon S3 and DynamoDB, the best way to provide credentials is to use an IAM role. In the **Deployed Application Permissions** area, you can either choose an existing IAM role or create one the wizard will use to launch your environment. Applications using the Amazon SDK for .NET will automatically use the credentials provided by this IAM role when making a request to an Amazon service.
9. If your application accesses an Amazon RDS database, in the drop-down list in the **Relational Database Access** area, select the boxes next to any Amazon RDS security groups the wizard will update so that your Amazon EC2 instances can access that database.
10. Choose **Next**.
    - If you selected **Use a VPC**, the **VPC Options** page will appear.
    - If you selected **Enable Rolling Deployments**, but did not select **Use a VPC**, the **Rolling Deployments** page will appear. Skip ahead to the instructions later in this section that describe how to use the **Rolling Deployments** page.
    - If you did not select **Use a VPC** or **Enable Rolling Deployments**, the **Application Options** page will appear. Skip ahead to the instructions later in this section that describe how to use the **Application Options** page.
11. If you selected **Use a VPC**, specify information on the **VPC Options** page to launch your application into a VPC.



The VPC must have already been created. If you created the VPC in the Toolkit for Visual Studio, the Toolkit for Visual Studio will populate this page for you. If you created the VPC in the Amazon Management Console, type information about your VPC into this page.

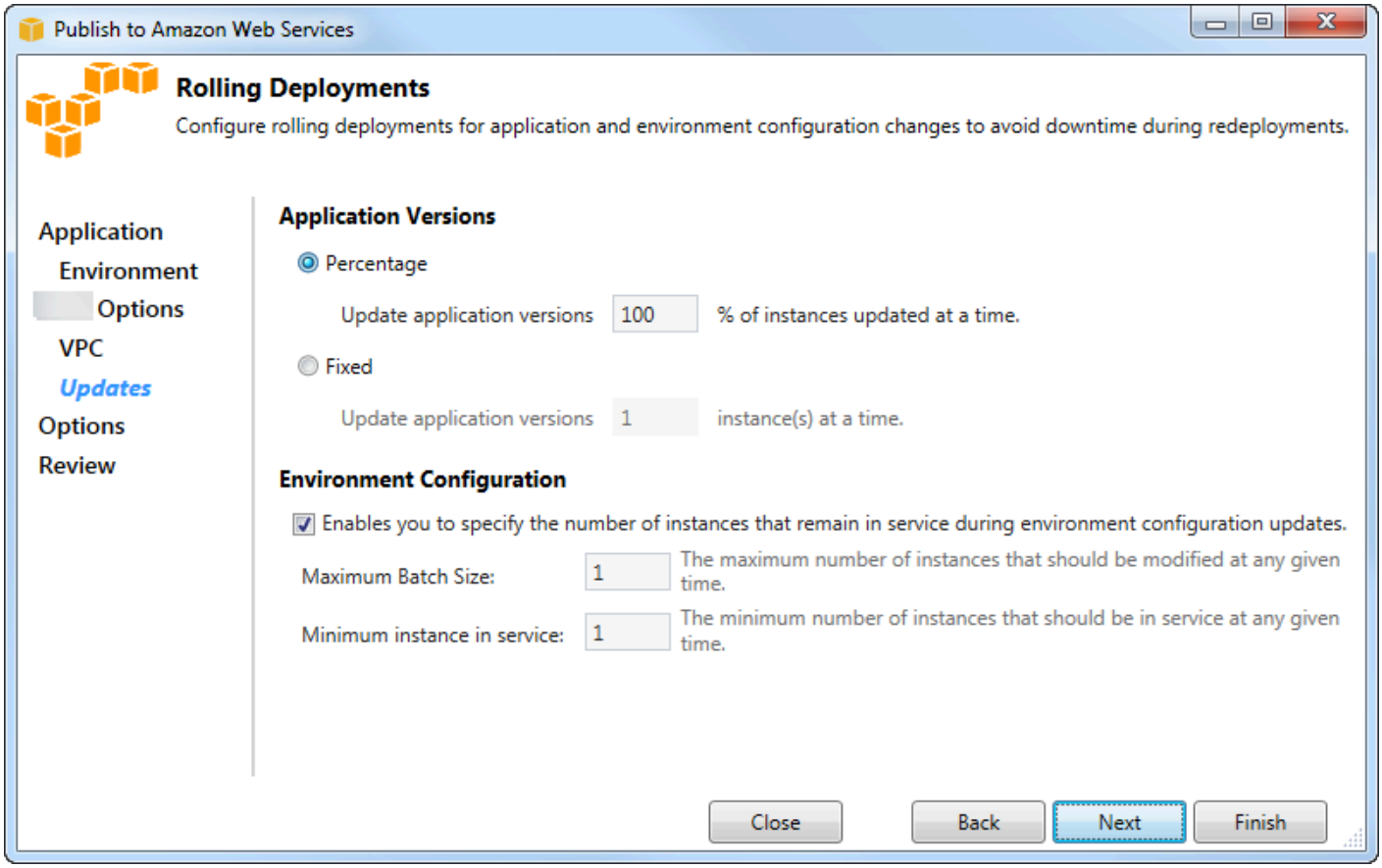
Key considerations for deployment to a VPC

- Your VPC needs at least one public and one private subnet.
- In the *ELB Subnet* drop-down list, specify the public subnet. The Toolkit for Visual Studio deploys the Elastic Load Balancing load balancer for your application to the public subnet. The public subnet is associated with a routing table that has an entry that points to an Internet gateway. You can recognize an Internet gateway because it has an ID that begins with `igw-` (for example, `igw-83cddaex`). Public subnets that you create by using the Toolkit for Visual Studio have tag values that identify them as public.
- In the *Instances Subnet* drop-down list, specify the private subnet. The Toolkit for Visual Studio deploys the Amazon EC2 instances for your application to the private subnet.

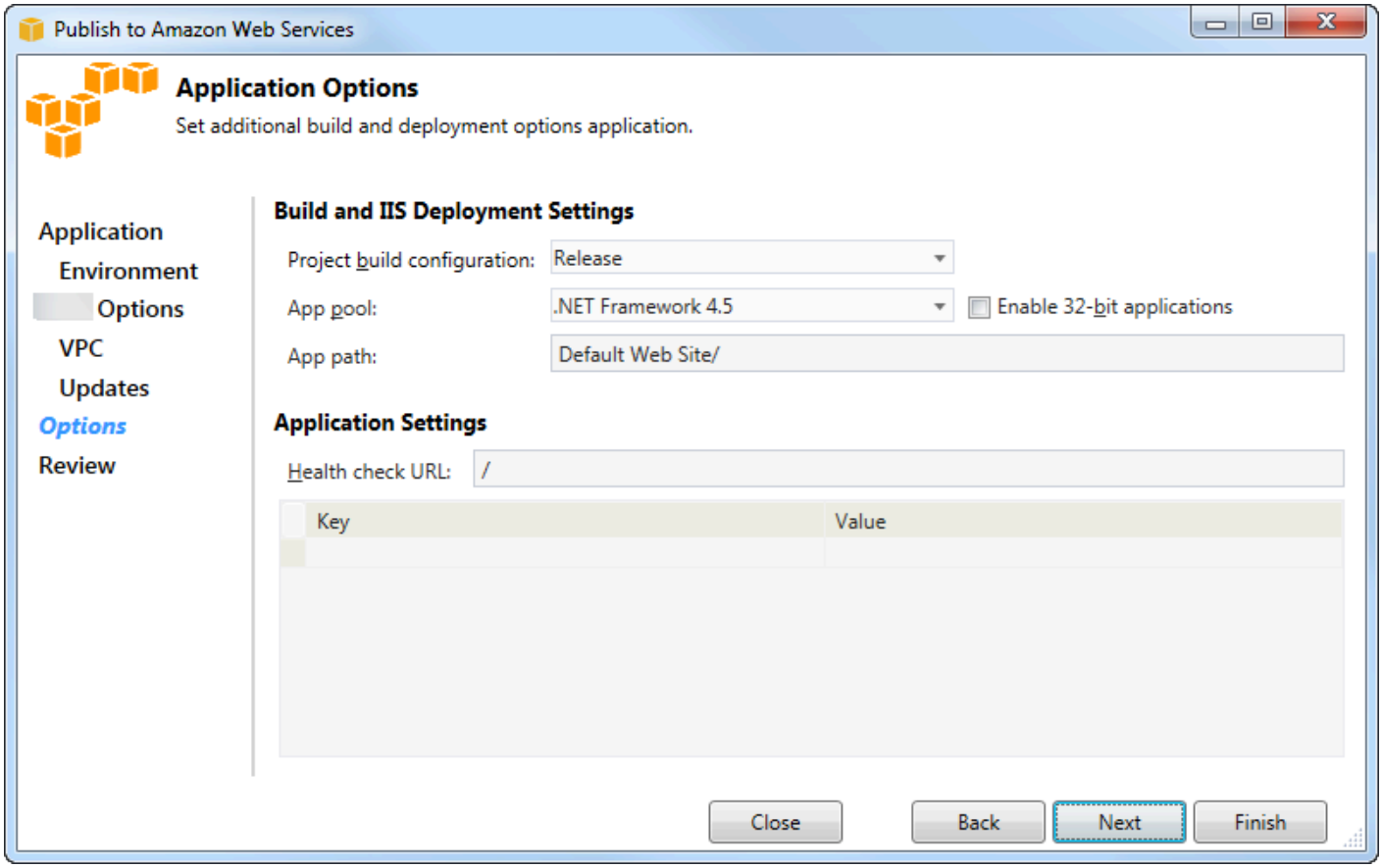
- The Amazon EC2 instances for your application communicate from the private subnet to the Internet through an Amazon EC2 instance in the public subnet that performs network address translation (NAT). To enable this communication, you will need a VPC security group that allows traffic to flow from the private subnet to the NAT instance. Specify this VPC security group in the *Security Group* drop-down list.

For more information about how to deploy an Elastic Beanstalk application to a VPC, go to the Amazon Elastic Beanstalk Developer Guide.

1. After you have filled in all of the information on the **VPC Options** page, choose **Next**.
   - If you selected **Enable Rolling Deployments**, the **Rolling Deployments** page will appear.
   - If you did not select **Enable Rolling Deployments**, the **Application Options** page will appear. Skip ahead to the instructions later in this section that describe how to use the **Application Options** page.
2. If you selected **Enable Rolling Deployments**, you specify information on the **Rolling Deployments** page to configure how new versions of your applications are deployed to the instances in a load-balanced environment. For example, if you have four instances in your environment and you want to change the instance type, you can configure the environment to change two instances at a time. This helps ensure your application is still running while changes are being made.



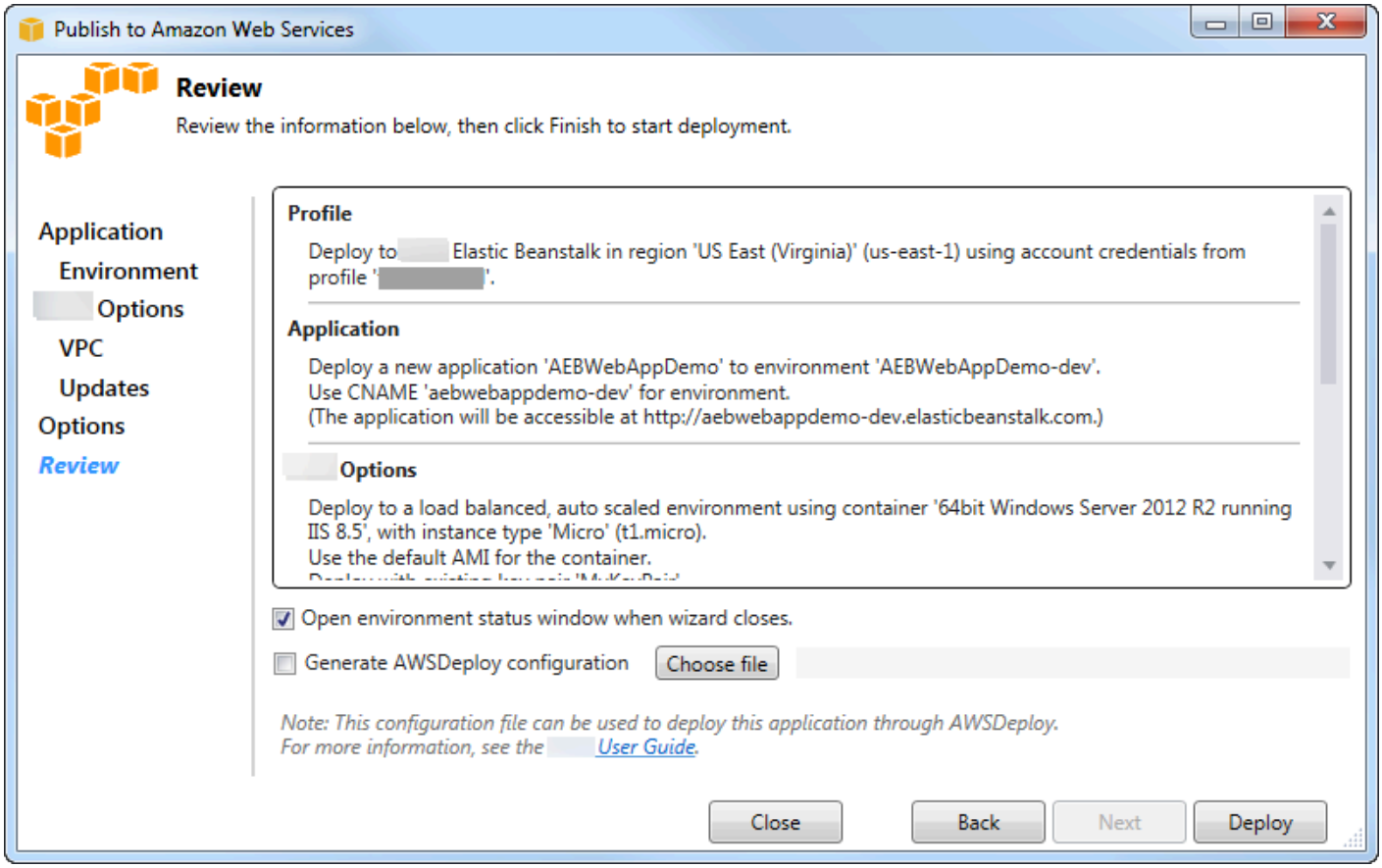
3. In the *Application Versions* area, choose an option to control deployments to either a percentage or number of instances at a time. Specify either the desired percentage or number.
4. Optionally, in the *Environment Configuration* area, select the box if you want to specify the number of instances that remain in service during deployments. If you select this box, specify the maximum number of instances that should be modified at a time, the minimum number of instances that should remain in service at a time, or both.
5. Choose *Next*.
6. On the **Application Options** page, you specify information about build, Internet Information Services (IIS), and application settings.



7. In the **Build and IIS Deployment Settings** area, in the **Project build configuration** drop-down list, choose the target build configuration. If the wizard can find it, **Release** appears otherwise, the active configuration is displayed in this box.
8. In the **App pool** drop-down list, choose the version of the .NET Framework required by your application. The correct .NET Framework version should already be displayed.
9. If your application is 32-bit, select the **Enable 32-bit applications** box.
10. In the **App path** box, specify the path IIS will use to deploy the application. By default, **Default Web Site/** is specified, which typically translates to the path `c:\inetpub\wwwroot`. If you specify a path other than **Default Web Site/**, the wizard will place a redirect in the **Default Web Site/** path that points to the path you specified.
11. In the **Application Settings** area, in the **Health check URL** box, type a URL for Elastic Beanstalk to check to determine if your web application is still responsive. This URL is relative to the root server URL. The root server URL is specified by default. For example, if the full URL is `example.com/site-is-up.html`, you would type `/site-is-up.html`.
12. In the area for **Key** and **Value**, you can specify any key and value pairs you want to add to your application's `Web.config` file.

> **Note**
>
> Although not recommended, you can use the area for **Key** and **Value**, to specify Amazon credentials under which your application should run. The preferred approach is to specify an IAM role in the **Identity and Access Management Role** drop-down list on the **Amazon Options** page. However, if you must use Amazon credentials instead of an IAM role to run your application, in the **Key** row, choose **AWSAccessKey**. In the **Value** row, type the access key. Repeat these steps for **AWSSecretKey**.

13. Choose **Next**.



14. On the **Review** page, review the options you configured, and select the **Open environment status window when wizard closes** box.

15. If everything looks correct, choose **Deploy**.

> **Note**
>
> When you deploy the application, the active account will incur charges for the Amazon resources used by the application.

Information about the deployment will appear in the Visual Studio status bar and the **Output** window. It may take several minutes. When the deployment is complete, a confirmation message will appear in the **Output** window.

16. To delete the deployment, in Amazon Explorer, expand the **Elastic Beanstalk** node, open the context (right-click) menu for the subnode for the deployment, and then choose **Delete**. The deletion process might take a few minutes.

Deploying an ASP.NET Core Application to Elastic Beanstalk (Legacy)

> **Important**
>
> This documentation refers to legacy services and features. For updated guides and content, see the Amazon .NET deployment tool guide and the updated Deploying to Amazon table of contents.

Amazon Elastic Beanstalk is a service that simplifies the process of provisioning Amazon resources for your application. Amazon Elastic Beanstalk provides all of the Amazon infrastructure required to deploy your application.

The Toolkit for Visual Studio supports deploying ASP.NET Core applications to Amazon using Elastic Beanstalk. ASP.NET Core is the redesign of ASP.NET with a modularized architecture that minimizes dependency overhead and streamlines your application to run in the cloud.

Amazon Elastic Beanstalk makes it easy to deploy applications in a variety of different languages to Amazon. Elastic Beanstalk supports both traditional ASP.NET applications and ASP.NET Core applications. This topic describes deploying ASP.NET Core applications.

Using the Deployment Wizard

The easiest way to deploy ASP.NET Core applications to Elastic Beanstalk is with the Toolkit for Visual Studio.

If you have used the toolkit before to deploy traditional ASP. NET applications, you'll find the experience for ASP.NET Core to be very similar. In the steps below, we'll walk through the deployment experience.

If you have never used the toolkit before, the first thing you'll need to do after installing the toolkit is register your Amazon credentials with the toolkit. See How to Specify the Amazon Security Credentials for Your Application for Visual Studio documentation for details on how to do so.

To deploy an ASP.NET Core web application, right-click the project in the Solution Explorer and select **Publish to Amazon...**.

On the first page of the Publish to Amazon Elastic Beanstalk deployment wizard, choose to create a new Elastic Beanstalk application. An Elastic Beanstalk application is a logical collection of Elastic Beanstalk components, including environments, versions, and environment configurations. The deployment wizard generates an application that in turn contains a collection of application versions and environments. The environments contain the actual Amazon resources that run an application version. Every time you deploy an application, a new application version is created and the wizard points the environment to that version. You can learn more about these concepts in Elastic Beanstalk Components..

Next, set names for the application and its first environment. Each environment has a unique CNAME associated with it that you can use to access the application when the deployment is complete.

The next page, **Amazon Options**, allows you to configure the type of Amazon resources to use. For this example, leave the default values, except for the **Key pair** section. Key pairs allow you retrieve the Windows administrator password so you can log on to the machine. If you haven't already created a key pair you might want to select **Create new key pair**.

Permissions

The **Permissions** page is used for assigning Amazon credentials to the EC2 instances running your application. This is important if your application uses the Amazon SDK for .NET to access other Amazon services. If you are not using any other services from your application then you can leave this page at its default.

Application Options

The details on the **Application Options** page are different from those specified when deploying traditional ASP.NET applications. Here, you specify the build configuration and framework used to package the application, and also specify the IIS resource path for the application.

After completing the **Application Options** page, click **Next** to review the settings, then click **Deploy** to begin the deployment process.

Checking Environment Status

After the application is packaged and uploaded to Amazon, you can check the status of the Elastic Beanstalk environment by opening the environment status view from the Amazon Explorer in Visual Studio.

Events are displayed in the status bar as the environment is coming online. Once everything is complete, the environment status will move to healthy state. You can click on the URL to view the site. From here, you can also pull the logs from the environment or remote desktop into the Amazon EC2 instances that are part of your Elastic Beanstalk environment.

The first deployment of any application will take a bit longer than subsequent re-deployments, as it creates new Amazon resources. As you iterate on your application during development, you can quickly re-deploy by going back through the wizard, or selecting the **Republish** option when you right click the project.

Republish packages your application using the settings from the previous run through the deployment wizard and uploads the application bundle to the existing Elastic Beanstalk environment.

How to Specify the Amazon Security Credentials for Your Application

The Amazon account you specify in the **Publish to Elastic Beanstalk** wizard is the Amazon account the wizard will use for deployment to Elastic Beanstalk.

Although not recommended, you may also need to specify Amazon account credentials that your application will use to access Amazon services after it has been deployed. The preferred approach is to specify an IAM role. In the **Publish to Elastic Beanstalk** wizard, you do this through the **Identity and Access Management Role** drop-down list on the **Amazon Options** page. In the legacy **Publish to Amazon Web Services** wizard, you do this through the **IAM Role** drop-down list on the **Amazon Options** page.

If you must use Amazon account credentials instead of an IAM role, you can specify the Amazon account credentials for your application in one of the following ways:

- Reference a profile corresponding to the Amazon account credentials in the `appSettings` element of the project's `Web.config` file. (To create a profile, see Configuring Amazon Credentials.) The following example specifies credentials whose profile name is `myProfile`.

```
<appSettings>
  <!-- Amazon CREDENTIALS -->
  <add key="AWSProfileName" value="myProfile"/>
</appSettings>
```

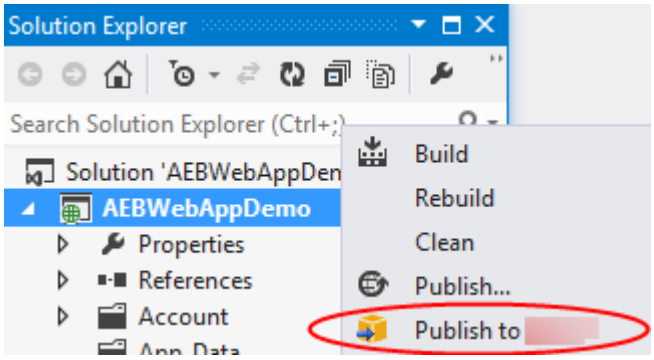
- If you're using the **Publish to Elastic Beanstalk** wizard, on the **Application Options** page, in the **Key** row of the **Key** and **Value** area, choose **AmazonAccessKey**. In the **Value** row, type the access key. Repeat these steps for **AmazonSecretKey**.
- If you're using the legacy **Publish to Amazon Web Services** wizard, on the **Application Options** page, in the **Application Credentials** area, choose **Use these credentials**, and then type the access key and secret access key into the **Access Key** and **Secret Key** boxes.

How to Republish Your Application to an Elastic Beanstalk Environment (Legacy)

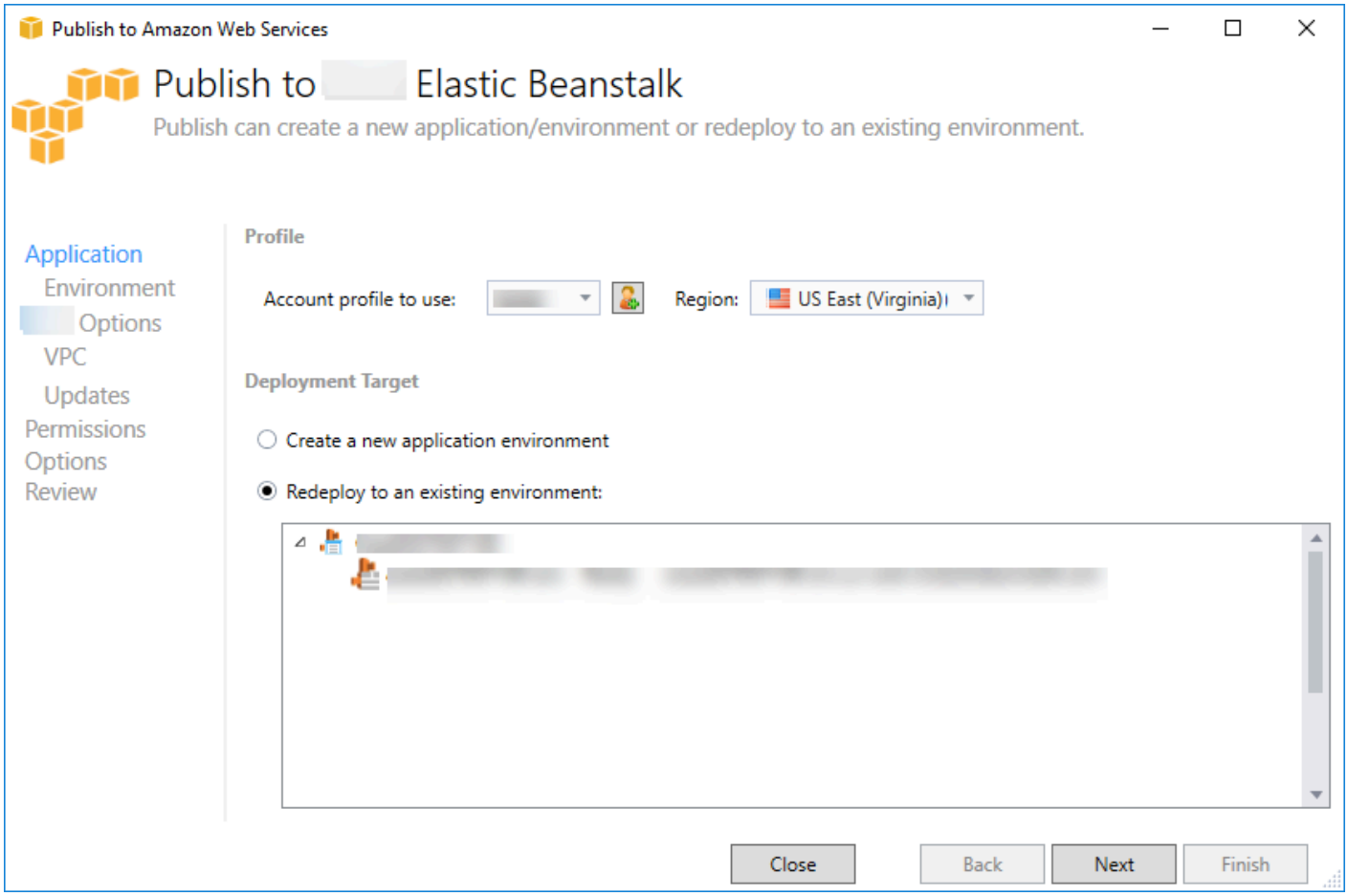
> **Important**
>
> This documentation refers to legacy services and features. For updated guides and content, see the Amazon .NET deployment tool guide.

You can iterate on your application by making discrete changes and then republishing a new version to your already launched Elastic Beanstalk environment.

1. In Solution Explorer, open the context (right-click) menu for the **AEBWebAppDemo** project folder for the project you published in the previous section, and choose **Publish to Amazon Elastic Beanstalk**.

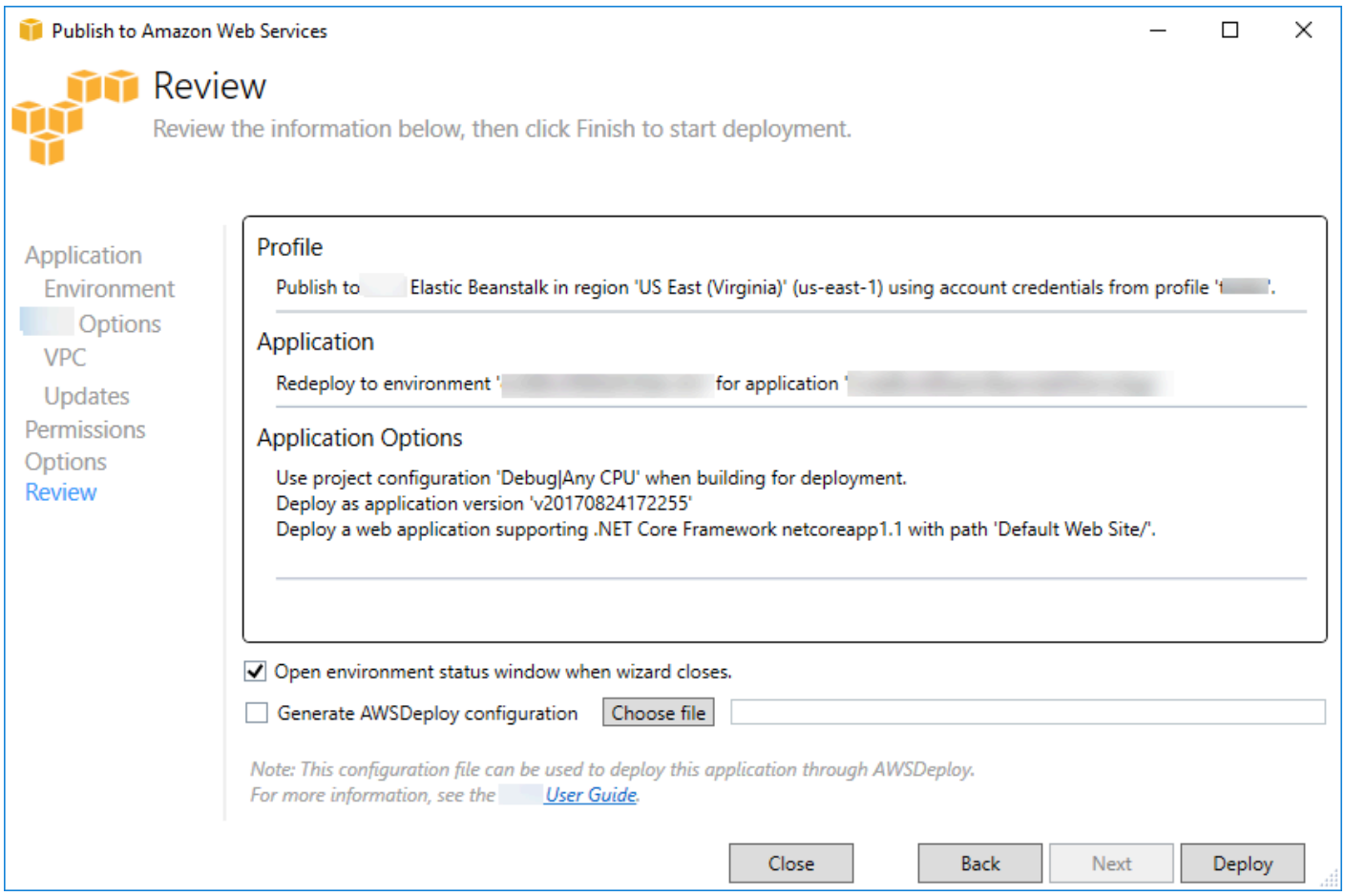


The **Publish to Elastic Beanstalk** wizard appears.



2. Select **Redeploy to an existing environment** and choose the environment you previously published to. Click **Next**.

   The **Review** wizard appears.



3. Click **Deploy**. The application will redeploy to the same environment.

You cannot republish if your application is in the process of launching or terminating.

Custom Elastic Beanstalk Application Deployments

This topic describes how the deployment manifest for Elastic Beanstalk's Microsoft Windows container supports custom application deployments.

Custom application deployments are a powerful feature for advanced users who want to leverage the power of Elastic Beanstalk to create and manage their Amazon resources, but want complete control on how their application is deployed. For a custom application deployment, you create Windows PowerShell scripts for the three different actions Elastic Beanstalk performs. The install action is used when a deployment is initiated, restart is used when the `RestartAppServer` API is called from either the toolkit or the web console, and uninstall which is invoked on any previous deployment whenever a new deployment occurs.

For example, you might have an ASP.NET application that you want to deploy while your documentation team has written a static website that they want included with the deployment. You can do that by writing your deployment manifest like this:

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

The scripts listed for each action must be in the application bundle relative to the deployment manifest file. For this example, the application bundle will also contain a documentation.zip file which contains a static website created by your documentation team.

The `install.ps1` script extracts the zip file and sets up the IIS Path.

```
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}
```

Since your application is running in IIS, the restart action will invoke an IIS reset.

```
iisreset /timeout:1
```

For uninstall scripts, it is important to clean up all settings and files used during the install stage. That way during the install phase for the new version, you can avoid any collision with previous deployments. For this example, you need to remove the IIS application for the static website and remove the website files.

```
powershell.exe -Command {Remove-WebApplication -Name documentation}
Remove-Item -Recurse -Force 'c:\inetpub\wwwroot\documentation'
```

With these script files and the documentation.zip file included in your application bundle, the deployment creates the ASP.NET application and then deploys the documentation site.

For this example, we choose a simple example that deploys a simple static website, but with custom application deployment you can deploy any type of application and let Elastic Beanstalk manage the Amazon resources for it.

Custom ASP.NET Core Elastic Beanstalk Deployments

This topic describes how deployment works and what you can do customize deployments when creating ASP.NET Core applications with Elastic Beanstalk and the Toolkit for Visual Studio.

After you complete the deployment wizard in the Toolkit for Visual Studio, the toolkit bundles the application and sends it to Elastic Beanstalk. Your first step in creating the application bundle is to use the new dotnet CLI to prepare the application for publishing by using the **publish** command. The framework and configuration are passed down from the settings in the wizard to the **publish** command. So if you selected **Release** for `configuration` and **netcoreapp1.0** for the `framework`, the toolkit will execute the following command:

```
dotnet publish --configuration Release --framework netcoreapp1.0
```

When the **publish** command finishes, the toolkit writes the new deployment manifest into the publishing folder. The deployment manifest is a JSON file named **aws-windows-deployment-manifest.json**, which the Elastic Beanstalk Windows container (version 1.2 or later) reads to determine how to deploy the application. For example, for an ASP.NET Core application you want to be deploy at the root of IIS, the toolkit generates a manifest file that looks like this:

```
{
  "manifestVersion": 1,
  "deployments": {

    "aspNetCoreWeb": [
      {
        "name": "app",
        "parameters": {
          "appBundle": ".",
          "iisPath": "/",
          "iisWebSite": "Default Web Site"
        }
      }
    ]
  }
}
```

The `appBundle` property indicates where the application bits are in relation to the manifest file. This property can point to either a directory or a ZIP archive. The `iisPath` and `iisWebSite` properties indicate where in IIS to host the application.

Customizing the Manifest

The toolkit only writes the manifest file if one doesn't already exist in the publishing folder. If the file does exist, the toolkit updates the `appBundle`, `iisPath` and `iisWebSite` properties in the first application listed under the `aspNetCoreWeb` section of the manifest. This allows you to add the **aws-windows-deployment-manifest.json** to your project and customize the manifest. To do this for an ASP.NET Core Web application in Visual Studio add a new JSON file to the root of the project and name it **aws-windows-deployment-manifest.json**.

The manifest must be named **aws-windows-deployment-manifest.json** and it must be at the root of the project. The Elastic Beanstalk container looks for the manifest in the root and if it finds it will invoke the deployment tooling. If the file doesn't exist, the Elastic Beanstalk container falls back to the older deployment tooling, which assumes the archive is an **msdeploy** archive.

To ensure the dotnet CLI `publish` command includes the manifest, update the `project.json` file to include the manifest file in the include section under `include` in `publishOptions`.

```
{
   "publishOptions": {
     "include": [
       "wwwroot",
       "Views",
       "Areas/**/Views",
       "appsettings.json",
       "web.config",
       "aws-windows-deployment-manifest.json"
     ]
   }
 }
```

Now that you've declared the manifest so that it's included in the app bundle, you can further configure how you want to deploy the application. You can customize deployment beyond what the deployment wizard supports. Amazon has defined a JSON schema for the **aws-windows-deployment-manifest.json file**, and when you installed the Toolkit for Visual Studio, the setup registered the URL for the schema.

When you open `windows-deployment-manifest.json`, you'll see the schema URL selected in the Schema drop down box. You can navigate to the URL to get a full description of what can be set in the manifest. With the schema selected, Visual Studio will provide IntelliSense while you're editing the manifest.

One customization you can do is to configure the IIS application pool under which the application will run. The following example shows how you can define an IIS Application pool ("customPool") that recycles the process every 60 minutes, and assigns it to the application using `"appPool": "customPool"`.

```
{
  "manifestVersion": 1,
  "iisConfig": {
    "appPools": [
      {
        "name": "customPool",
        "recycling": {
          "regularTimeInterval": 60
        }
```

```
      }
    ]
  },
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "app",
        "parameters": {
          "appPool": "customPool"
        }
      }
    ]
  }
}
```

Additionally, the manifest can declare Windows PowerShell scripts to run before and after the install, restart and uninstall actions. For example, the following manifest runs the Windows PowerShell script `PostInstallSetup.ps1` to do further setup work after the ASP.NET Core application is deployed to IIS. When adding scripts like this, make sure the scripts are added to the include section under publishOptions in the `project.json` file, just as you did with the `aws-windows-deployment-manifest.json` file. If you don't, the scripts won't be included as part of the dotnet CLI **publish** command.

```
{
  "manifestVersion": 1,
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "app",
        "scripts": {
          "postInstall": {
            "file": "SetupScripts/PostInstallSetup.ps1"
          }
        }
      }
    ]
  }
}
```

What about .ebextensions?

The Elastic Beanstalk **.ebextensions** configuration files are supported as with all the other Elastic Beanstalk containers. To include .ebextensions in an ASP.NET Core application, add the `.ebextensions` directory to the `include` section under `publishOptions` in the `project.json` file. For further information about .ebextensions checkout the Elastic Beanstalk Developer Guide.

Multiple Application Support for .NET and Elastic Beanstalk

Using the deployment manifest, you have the ability to deploy multiple applications to the same Elastic Beanstalk environment.

The deployment manifest supports ASP.NET Core web applications as well as msdeploy archives for traditional ASP.NET applications. Imagine a scenario where you have written a new amazing application using ASP.NET Core for the frontend and a Web API project for an extensions API. You also have an admin app that you wrote using traditional ASP.NET.

The toolkit's deployment wizard focuses on deploying a single project. To take advantage of multiple application deployment, you have to construct the application bundle by hand. To start, write the manifest. For this example, you will write the manifest at the root of your solution.

The deployment section in the manifest has two children: an array of ASP.NET Core web applications to deploy, and an array of msdeploy archives to deploy. For each application, you set the IIS path and the location of the application's bits relative to the manifest.

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

With the manifest written, you'll use Windows PowerShell to create the application bundle and update an existing Elastic Beanstalk environment to run it. The script is written assuming that it will be run from the folder containing your Visual Studio solution.

The first thing you need to do in the script is setup a workspace folder in which to create the application bundle.

```
$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
}
```

Once you've created the folder, it is time to get the frontend ready. As with the deployment wizard, use the dotnet CLI to publish the application.

```
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
```

Notice that the subfolder "frontend" was used for the output folder, matching the folder you set in the manifest. Now you need to do the same for the Web API project.

```
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
```

The admin site is a traditional ASP.NET application, so you can't use the dotnet CLI. For the admin application, you should use msbuild, passing in the build target package to create the msdeploy archive. By default the package target creates the msdeploy archive under the `obj\Release\Package` folder, so you will need to copy the archive to the publish workspace.

```
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
```

To tell the Elastic Beanstalk environment what to do with all these applications, copy the manifest from your solution to the publish workspace and then zip up the folder.

```
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)
```

Now that you have the application bundle, you could go to the web console and upload the archive to a Elastic Beanstalk environment. Alternatively, you can continue to use the Amazon PowerShell cmdlets to update the Elastic Beanstalk environment with the application bundle. Make sure you have set the current profile and region to the profile and region that contains your Elastic Beanstalk environment by using `Set-AWSCredentials` and `Set-DefaultAWSRegion` cmdlets.

```
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
```

Now, check the status of the update using either the Elastic Beanstalk environment status page in either the toolkit or the web console. Once complete you will be able to navigate to each of the applications you deployed at the IIS path set in the deployment manifest.

Deploying to Amazon EC2 Container Service

> **Important**
>
> The new **Publish to Amazon** feature is designed to simplify how you publish .NET applications to Amazon. You may be asked if you want to switch to this publishing experience after you choose **Publish Container to Amazon**. For more information, see Working with Publish to Amazon in Visual Studio.

Amazon Elastic Container Service is a highly scalable, high performance container management service that supports Docker containers and allows you to easily run applications on a managed cluster of Amazon EC2 instances.

To deploy applications on Amazon Elastic Container Service, your application components must be developed to run in a Docker container. A Docker container is a standardized unit of software development, containing everything that your software application needs to run: code, runtime, system tools, system libraries, etc.

The Toolkit for Visual Studio provides a wizard that simplifies publishing applications through Amazon ECS. This wizard is described in the following sections.

For more information about Amazon ECS, go to the Elastic Container Service documentation. It includes an overview of Docker basics and creating a cluster.

Topics

- Specify Amazon Credentials for Your ASP.NET Core 2 Application
- Deploying an ASP.NET Core 2.0 App to Amazon ECS (Fargate) (Legacy)
- Deploying an ASP.NET Core 2.0 App to Amazon ECS (EC2)

Specify Amazon Credentials for Your ASP.NET Core 2 Application

There are two types of credentials in play when you deploy your application to a Docker container: deployment credentials and instance credentials.

Deployment credentials are used by the Publish Container to Amazon wizard to create the environment in Amazon ECS. This includes things like tasks, services, IAM roles, a Docker container repository, and if you choose, a load balancer.

Instance credentials are used by the instance (including your application) to access different Amazon services. For example, if your an ASP.NET Core 2.0 application reads and writes to Amazon S3 objects, it will need appropriate permissions. You can provide different credentials using different methods based on the environment. For example, your ASP.NET Core 2 application might target *Development* and *Production* environments. You could use a local Docker instance and credentials for development and a defined role in production.

Specifying deployment credentials

The Amazon account you specify in the **Publish Container to Amazon** wizard is the Amazon account the wizard will use for deployment to Amazon ECS. The account profile must have permissions to Amazon Elastic Compute Cloud, Amazon Elastic Container Service, and Amazon Identity and Access Management.

If you notice options missing from drop-down lists, it may be because you lack permissions. For example, if you created a cluster for your application but do not see it on the **Publish Container to Amazon** wizard Cluster page. If this happens, add the missing permissions and try the wizard again.

Specifying development instance credentials

For non-production environments, you can configure your credentials in the appsettings.<environment>.json file. For example, to configure your credentials in the appsettings.Development.json file in Visual Studio 2017:

1. Add the AWSSDK.Extensions.NETCore.Setup NuGet package to your project.

2. Add Amazon settings to appsettings.Development.json. The configuration below sets `Profile` and `Region`.

```
{
    "AWS": {
        "Profile": "local-test-profile",
        "Region": "us-west-2"
    }
}
```

Specifying production instance credentials

For production instances, we recommend you use an IAM role to control what your application (and the service) can access. For example, to configure an IAM role with Amazon ECS as the service principal with permissions to Amazon Simple Storage Service and Amazon DynamoDB from the Amazon Web Services Management Console:

1. Sign in to the Amazon Web Services Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane of the IAM console, choose Roles, and then choose Create role.
3. Choose the **Amazon Service** role type, and then choose **EC2 Container Service**.
4. Choose the **EC2 Container Service Task** use case. Use cases are defined by the service to include the trust policy that the service requires. Then choose **Next: Permissions**.
5. Choose the **AmazonS3FullAccess** and **AmazonDynamoDBFullAccess** permissions policies. Check the box next to each policy, and then choose **Next: Review**,
6. For **Role name**, type a role name or role name suffix to help you identify the purpose of this role. Role names must be unique within your Amazon account. They are not distinguished by case. For example, you cannot create roles named both `PRODROLE` and `prodrole`. Because various entities might reference the role, you cannot edit the name of the role after it has been created.
7. (Optional) For **Role description**, type a description for the new role.
8. Review the role and then choose **Create role**.

You can use this role as the **task role** on the **ECS Task Definition** page of the **Publish Container to Amazon** wizard.

For more information, see Using Service-Based Roles.

Deploying an ASP.NET Core 2.0 App to Amazon ECS (Fargate) (Legacy)

> **Important**
>
> This documentation refers to legacy services and features. For updated guides and content, see the Amazon .NET deployment tool guide and the updated Deploying to Amazon table of contents.

This section describes how to use the **Publish Container to Amazon** wizard, provided as part of the Toolkit for Visual Studio, to deploy a containerized ASP.NET Core 2.0 application targeting Linux through Amazon ECS using the Fargate launch type. Because a web application is meant to run continuously, it will be deployed as a service.

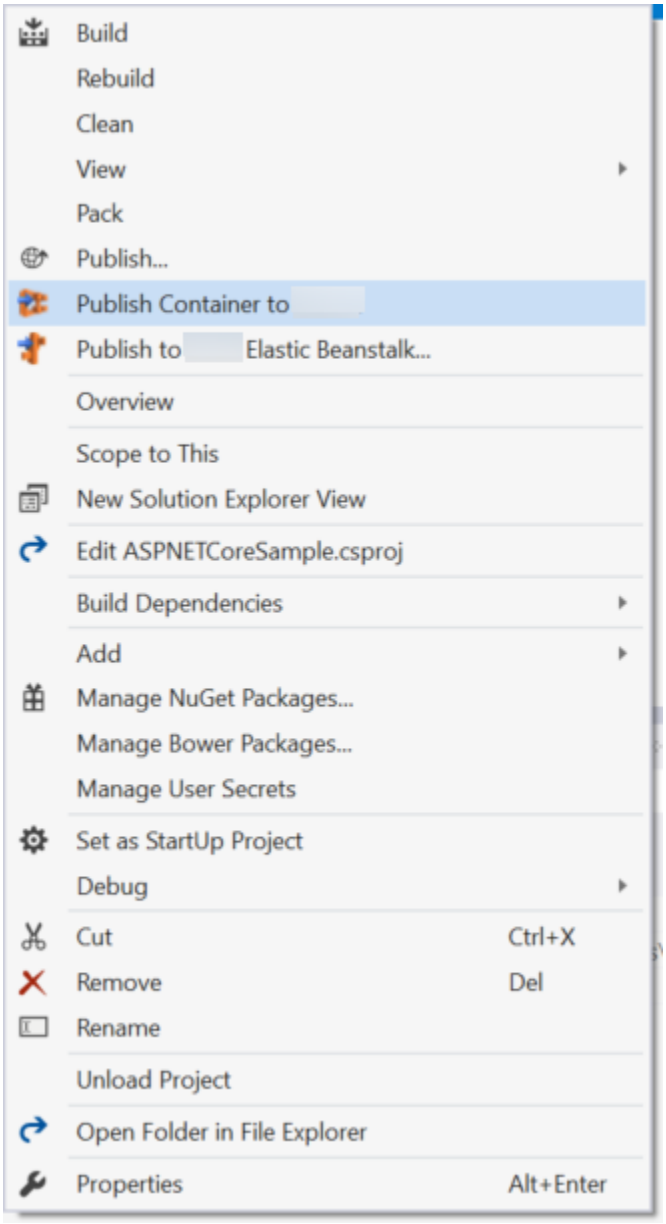
Before you publish your container

Before using the **Publish Container to Amazon** wizard to deploy your ASP.NET Core 2.0 application:

- Specify your Amazon credentials and get setup with Amazon ECS.
- Install Docker. You have a few different installation options including Docker for Windows.
- In Visual Studio, create (or open) a project for an ASP.NET Core 2.0 containerized app targeting Linux.

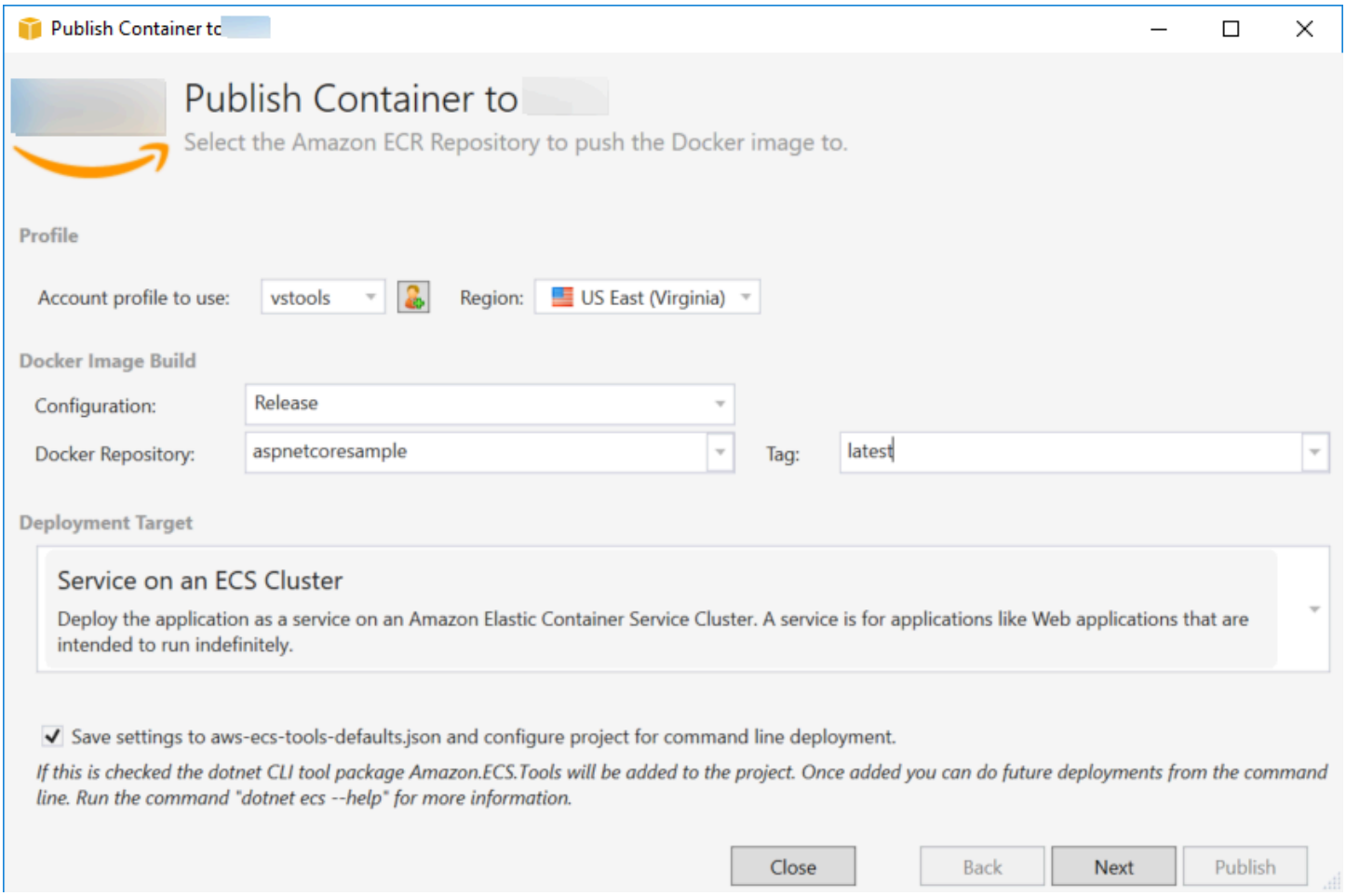
Accessing the Publish Container to Amazon wizard

To deploy an ASP.NET Core 2.0 containerized application targeting Linux, right-click the project in the Solution Explorer and select **Publish Container to Amazon**.



You can also select **Publish Container to Amazon** on the Visual Studio Build menu.

Publish Container to Amazon Wizard



Account profile to use - Select an account profile to use.

Region - Choose the deployment region. Profile and region are used to set up your deployment environment resources and to select the default Docker registry.

Configuration - Select the Docker image build configuration.

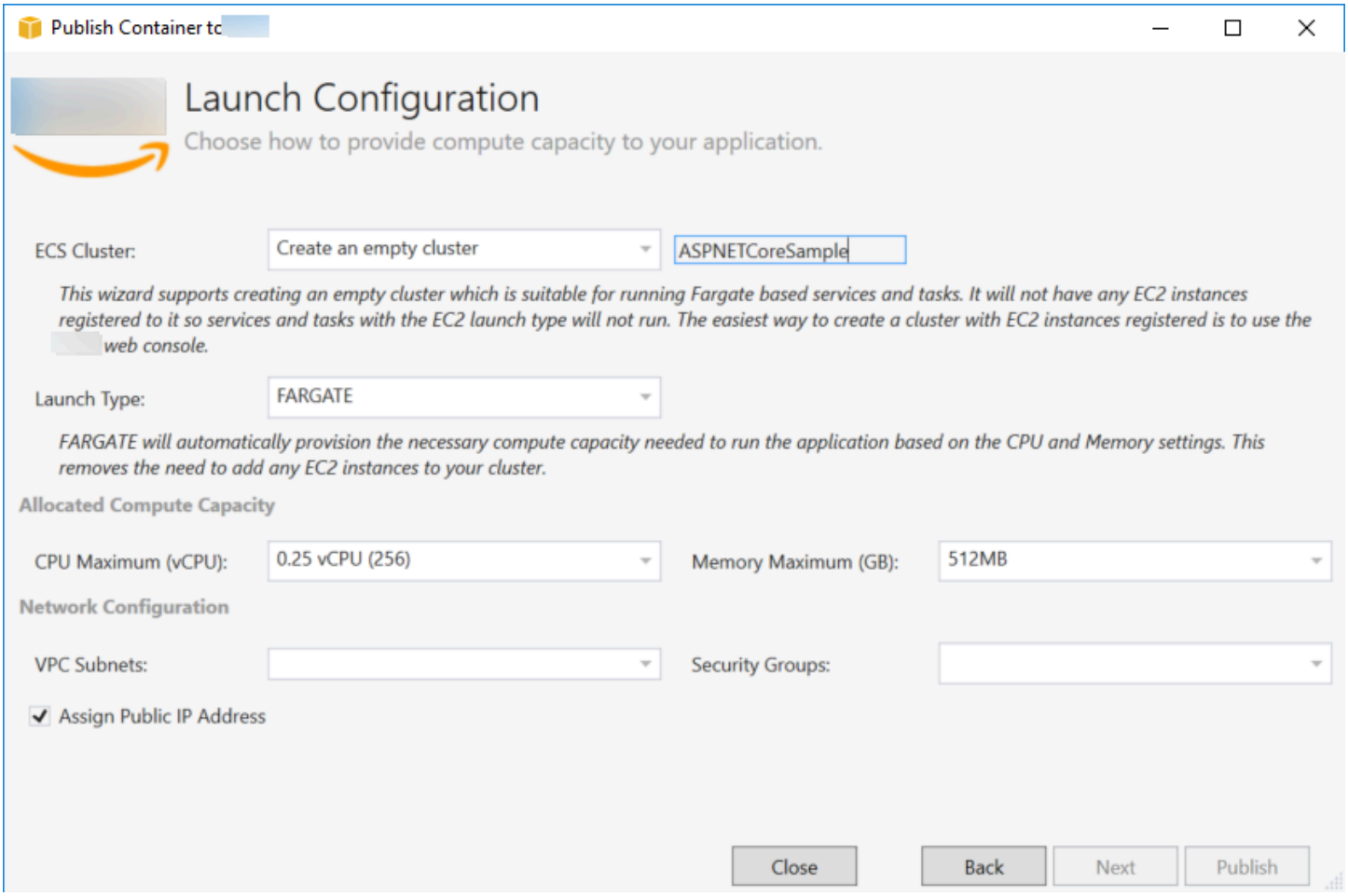
Docker Repository - Choose an existing Docker repository or type in the name of a new repository and it will be created. This is the repository the build container is pushed to.

Tag - Select an existing tag or type in the name of a new tag. Tags can track important details like version, options or other unique configuration elements of the Docker container.

Deployment Target - Select **Service on an ECS Cluster**. Use this deployment option when your application is meant to be long-running (like an ASP.NET web application).

Save settings to `aws-docker-tools-defaults.json` and configure project for command line deployment - Check this option if you want the flexibility of deploying from the command line. Use `dotnet ecs deploy` from your project directory to deploy and `dotnet ecs publish` the container.

Launch Configuration page



ECS Cluster - Pick the cluster that will run your Docker image. If you choose to create an empty cluster, provide a name for your new cluster.

Launch Type - Choose **FARGATE**.

CPU Maximum (vCPU) - Choose the maximum amount of compute capacity needed for your application. To see allowed ranges of CPU and Memory values, see task size.

Memory Maximum (GB) - Select the maximum amount of memory available to your application.

VPC Subnets - Choose one or more subnets under a single VPC. If you choose more than one subnet, your tasks will be distributed across them. This can improve availability. For more information, see default VPC and default subnets.

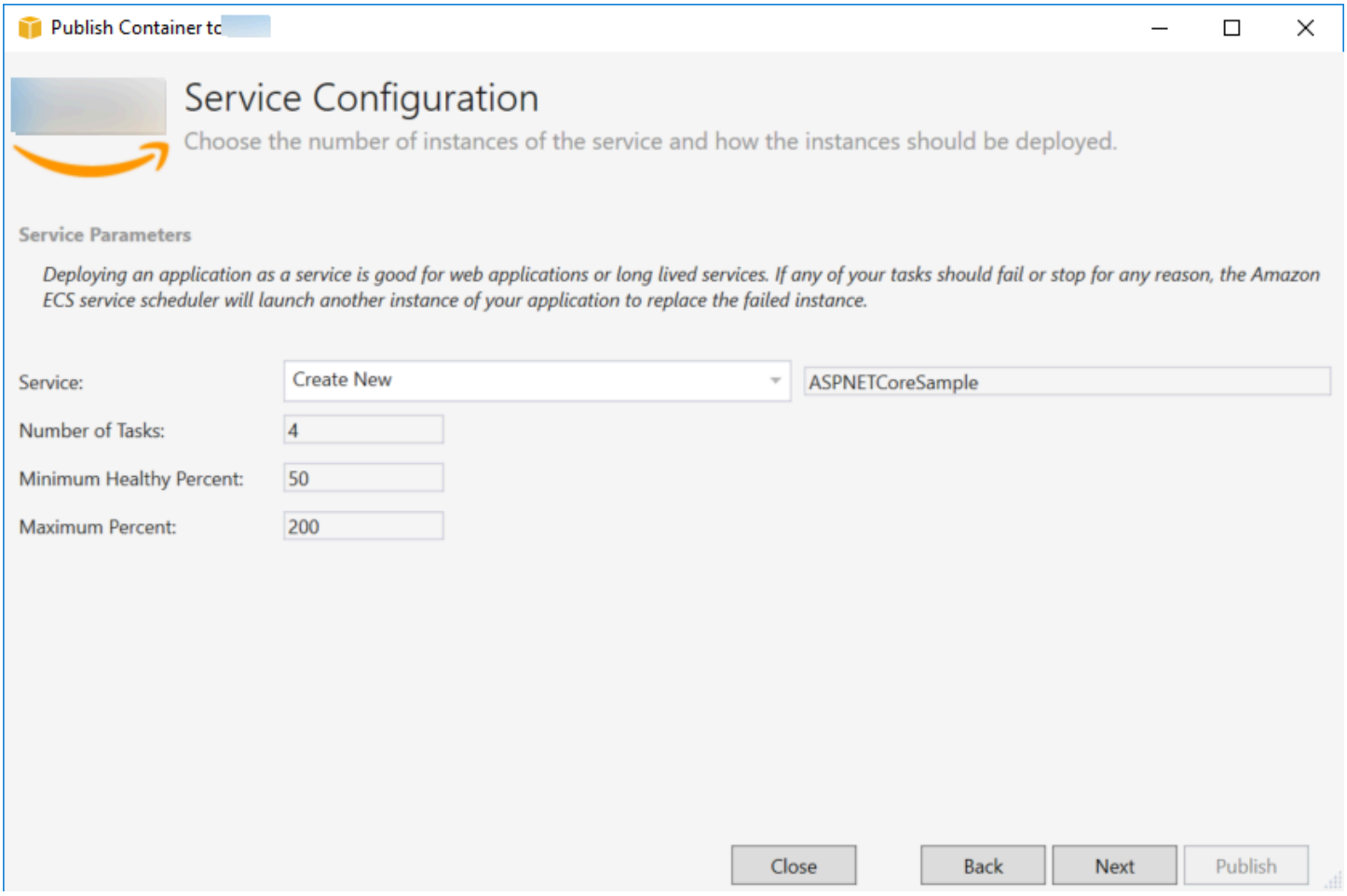
Security Groups - Choose a security group.

A security group acts as a firewall for associated Amazon EC2 instances, controlling both inbound and outbound traffic at the instance level.

Default security groups are configured to allow inbound traffic from instances assigned to the same security group and all outbound IPv4 traffic. You need outbound allowed so the service can reach the container repository.

Assign Public IP Address - Check this to make your task accessible from the internet.

Service Configuration page



Service - Select one of the services in the drop-down to deploy your container into an existing service. Or choose **Create New** to create a new service. Service names must be unique within a

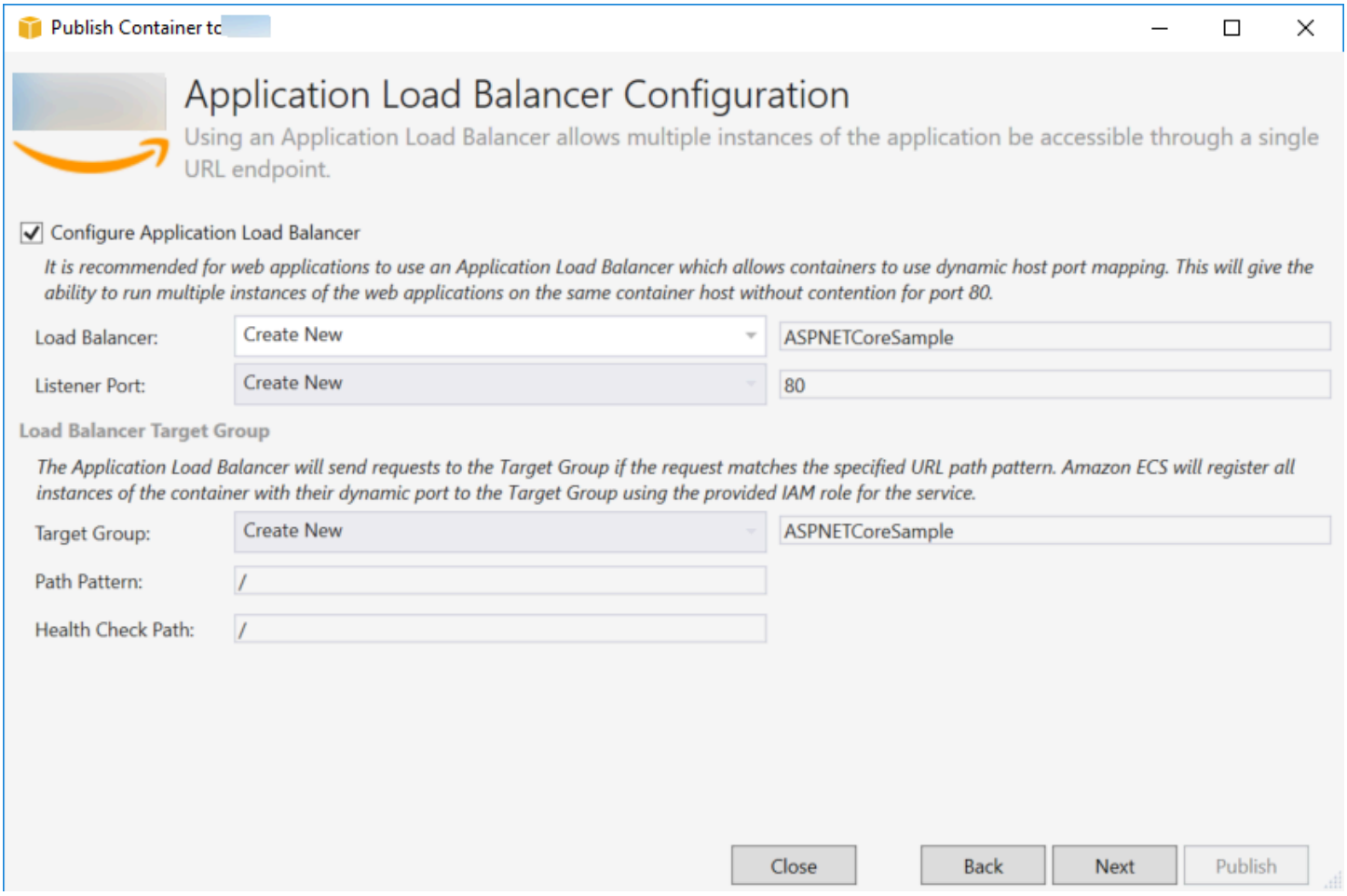
cluster, but you can have similarly named services in multiple clusters within a region or across multiple regions.

Number of Tasks - The number of tasks to deploy and keep running on your cluster. Each task is one instance of your container.

Minimum Healthy Percent - The percentage of tasks that must remain in `RUNNING` state during a deployment rounded up to the nearest integer.

Maximum Percent - The percentage of tasks that are allowed in the `RUNNING` or `PENDING` state during a deployment rounded down to the nearest integer.

Application Load Balancer page



Configure Application Load Balancer - Check to configure an application load balancer.

Load Balancer - Select an existing load balancer or choose **Create New** and type in the name for the new load balancer.

Listener Port - Select an existing listener port or choose **Create New** and type in a port number. The default, port 80, is appropriate for most web applications.

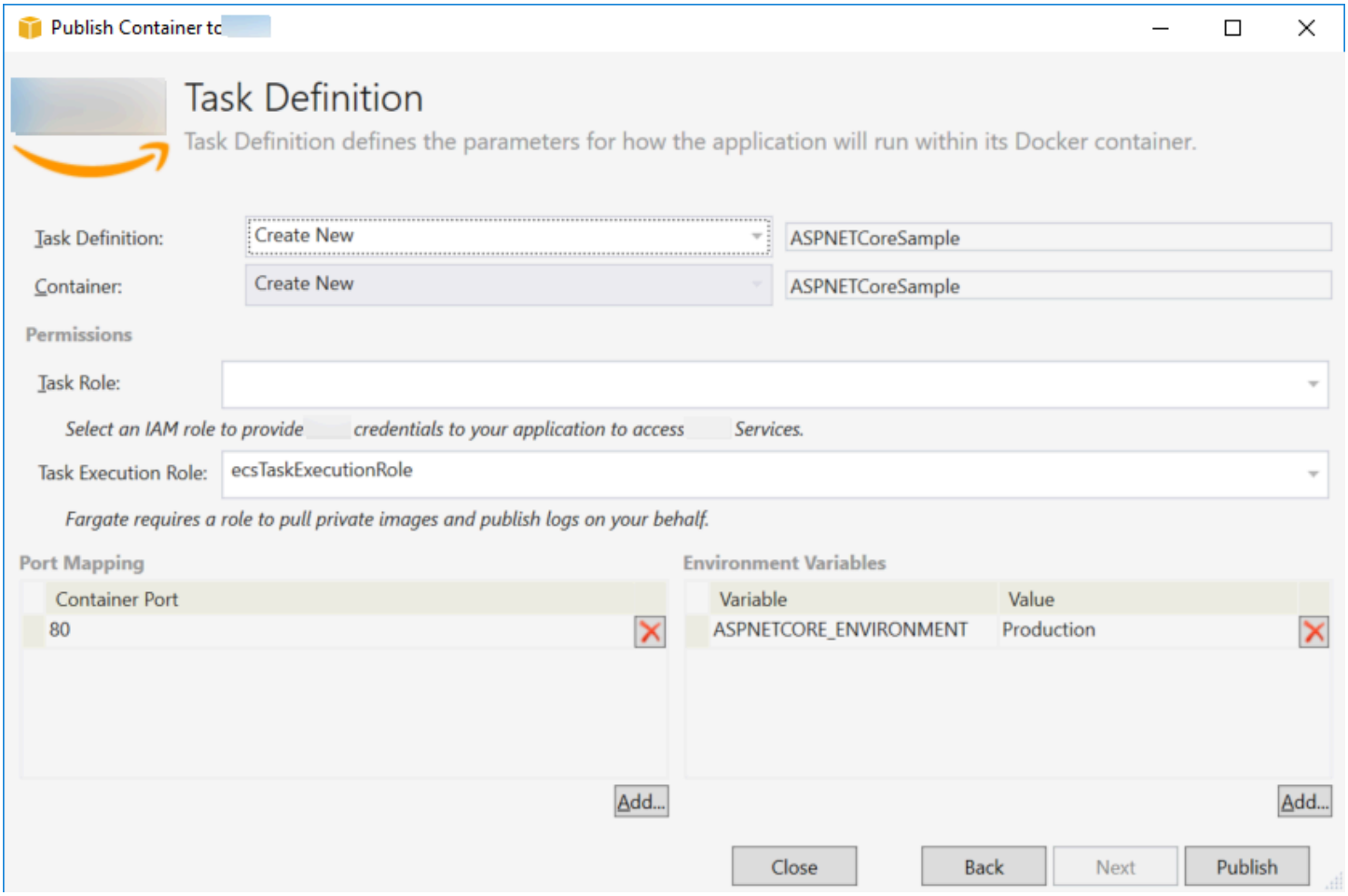
Target Group - Select the target group Amazon ECS will register the tasks to the service to.

Path Pattern - The load balancer will use path-based routing. Accept the default / or provide a different pattern. The path pattern is case-sensitive, can be up to 128 characters in length, and contains a select set of characters.

Health Check Path - The ping path that is the destination on the targets for health checks. By default, it is /. Enter a different path if needed. If the path you enter is invalid, the health check will fail and it will be considered unhealthy.

If you deploy multiple services, and each service will be deployed to a different path or location, you will need custom check paths.

Task Definition page



Task Definition - Select an existing task definition or choose **Create New** and type in the new task definition name.

Container - Select an existing container or choose **Create New** and type in the new container name.

Task Role - Select an IAM role that has the credentials your app needs to access Amazon Services. This is how credentials are passed in to your application. See how to specify Amazon security credentials for your application.

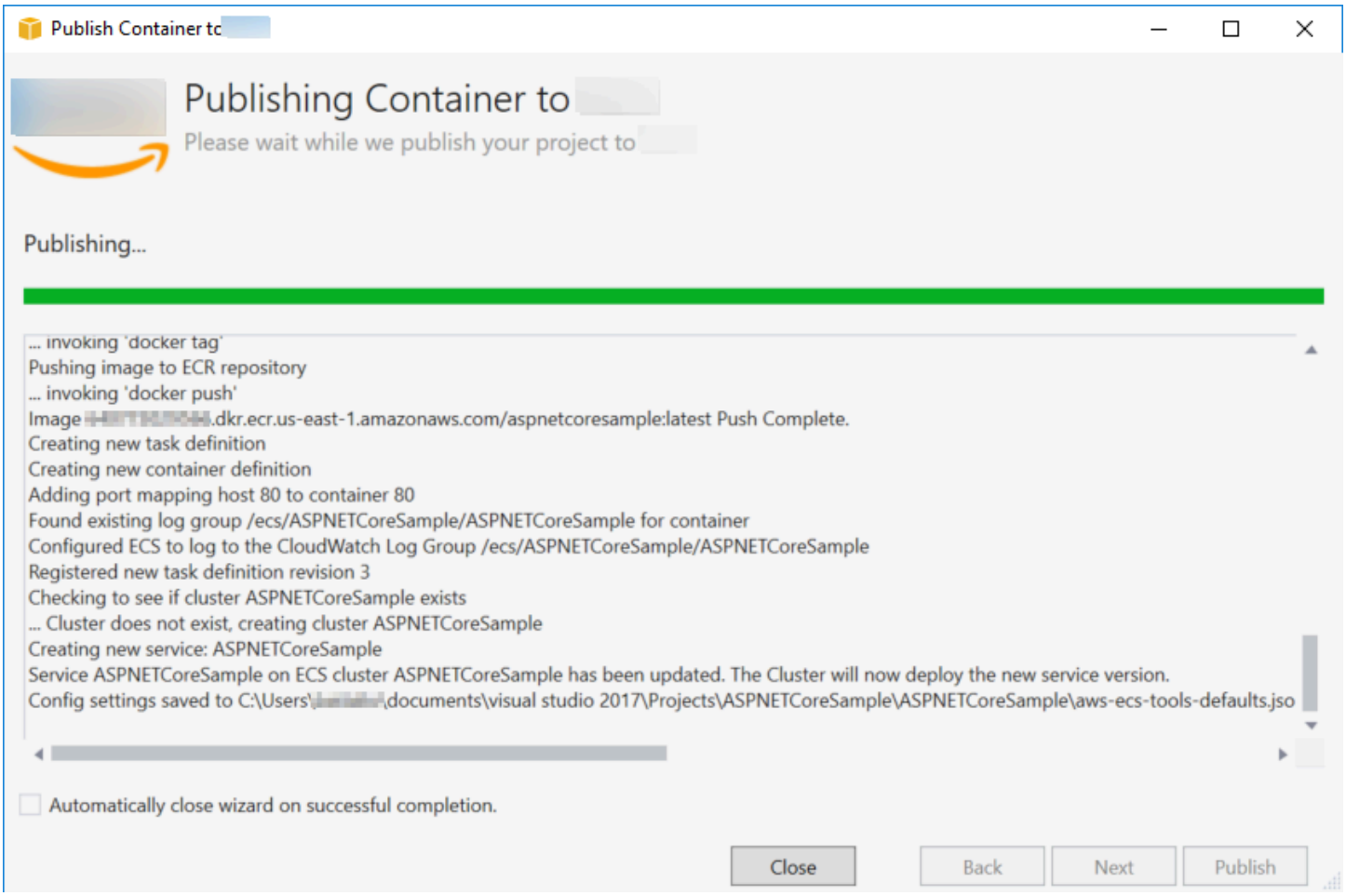
Task Execution Role - Select a role with permissions to pull private images and publish logs. Amazon Fargate will use it on your behalf.

Port Mapping - Choose the port number on the container that is bound to the automatically assigned host port.

Environment Variables - Add, modify, or delete environment variables for the container. You can modify it to suit your deployment.

When you are satisfied with the configuration, click **Publish** to begin the deployment process.

Publishing Container to Amazon



Events are displayed during deployment. The wizard is automatically closed on successful completion. You can override this by unchecking the box at the bottom of the page.

You can find the URL of your new instances in the Amazon Explorer. Expand Amazon ECS and Clusters, then click on your cluster.

Deploying an ASP.NET Core 2.0 App to Amazon ECS (EC2)

This section describes how to use the **Publish Container to Amazon** wizard, provided as part of the Toolkit for Visual Studio, to deploy a containerized ASP.NET Core 2.0 application targeting Linux through Amazon ECS using the EC2 launch type. Because a web application is meant run continuously, it will be deployed as a service.

Before you publish your container

Before using the **Publish Container to Amazon** to deploy your ASP.NET Core 2.0 application:

- Specify your Amazon credentials and get setup with Amazon ECS.
- Install Docker. You have a few different installation options including Docker for Windows.
- Create an Amazon ECS cluster based on the needs of your web application. It only takes a few steps.
- In Visual Studio, create (or open) a project for an ASP.NET Core 2.0 containerized app targeting Linux.

Accessing the Publish Container to Amazon wizard

To deploy an ASP.NET Core 2.0 containerized application targeting Linux, right-click the project in the Solution Explorer and select **Publish Container to Amazon**.

You can also select **Publish Container to Amazon** on the Visual Studio Build menu.

Publish Container to Amazon Wizard

Account profile to use - Select an account profile to use.

Region - Choose a deployment region. Profile and region are used to set up your deployment environment resources and select the default Docker registry.

Configuration - Select the Docker image build configuration.

Docker Repository - Choose an existing Docker repository or type in the name of a new repository and it will be created. This is the repository the built container image is pushed to.

Tag - Select an existing tag or type in the name of a new tag. Tags can track important details like version, options or other unique configuration elements of the Docker container.

Deployment - Select **Service on an ECS Cluster**. Use this deployment option when your application is meant to be long-running (like an ASP.NET Core 2.0 web application).

Save settings to `aws-docker-tools-defaults.json` and configure project for command line deployment - Check this option if you want the flexibility of deploying from the command line. Use `dotnet ecs deploy` from your project directory to deploy and `dotnet ecs publish` the container.

Launch Configuration page

ECS Cluster - Pick the cluster that will run your Docker image. You can create an ECS cluster using the Amazon Management Console.

Launch Type - Choose EC2. To use the Fargate launch type, see Deploying an ASP.NET Core 2.0 Application to Amazon ECS (Fargate).

Service Configuration page

Service - Select one of the services in the drop-down to deploy your container into an existing service. Or choose **Create New** to create a new service. Service names must be unique within a cluster, but you can have similarly named services in multiple clusters within a region or across multiple regions.

Number of Tasks - The number of tasks to deploy and keep running on your cluster. Each task is one instance of your container.

Minimum Healthy Percent - The percentage of tasks that must remain in `RUNNING` state during a deployment rounded up to the nearest integer.

Maximum Percent - The percentage of tasks that are allowed in the `RUNNING` or `PENDING` state during a deployment rounded down to the nearest integer.

Placement Templates - Select a task placement template.

When you launch a task into a cluster, Amazon ECS must determine where to place the task based on the requirements specified in the task definition, such as CPU and memory. Similarly, when you scale down the task count, Amazon ECS must determine which tasks to terminate.

The placement template controls how tasks are launched into a cluster:

- AZ Balanced Spread - distribute tasks across Availability Zones and across container instances in the Availability Zone.
- AZ Balanced BinPack - distribute tasks across Availability Zones and across container instances with the least available memory.
- BinPack - distribute tasks based on the least available amount of CPU or memory.
- One Task Per Host - place, at most, one task from the service on each container instance.

For more information, see Amazon ECS Task Placement.

Application Load Balancer page

Configure Application Load Balancer - Check to configure an application load balancer.

Select IAM role for service - Select an existing role or choose **Create New** and a new role will be created.

Load Balancer - Select an existing load balancer or choose **Create New** and type in the name for the new load balancer.

Listener Port - Select an existing listener port or choose **Create New** and type in a port number. The default, port `80`, is appropriate for most web applications.

Target Group - By default, the load balancer sends requests to registered targets using the port and protocol that you specified for the target group. You can override this port when you register each target with the target group.

Path Pattern - The load balancer will use path-based routing. Accept the default `/` or provide a different pattern. The path pattern is case-sensitive, can be up to 128 characters in length, and contains a select set of characters.

Health Check Path - The ping path that is the destination on the targets for health checks. By default, it is `/` and is appropriate for web applications. Enter a different path if needed. If the path you enter is invalid, the health check will fail and it will be considered unhealthy.

If you deploy multiple services, and each service will be deployed to a different path or location, you might need custom check paths.

ECS Task Definition page

Task Definition - Select an existing task definition or choose **Create New** and type in the new task definition name.

Container - Select an existing container or choose **Create New** and type in the new container name.

Memory (MiB) - Provide values for **Soft Limit** or **Hard Limit** or both.

The *soft limit* (in MiB) of memory to reserve for the container. Docker attempts to keep the container memory under the soft limit. The container can consume more memory, up to either the hard limit specified with the memory parameter (if applicable), or all of the available memory on the container instance, whichever comes first.

The *hard limit* (in MiB) of memory to present to the container. If your container attempts to exceed the memory specified here, the container is killed.

Task Role - Select a task role for an IAM role that allows the container permission to call the Amazon APIs that are specified in its associated policies on your behalf. This is how credentials are passed in to your application. See how to specify Amazon security credentials for your application.

Port Mapping - Add, modify or delete port mappings for the container. If a load balancer is on, the host port will be default to 0 and port assignment will be dynamic.

Environment Variables - Add, modify, or delete environment variables for the container.

When you are satisfied with the configuration, click **Publish** to begin the deployment process.

Publishing Container to Amazon

Events are displayed during deployment. The wizard is automatically closed on successful completion. You can override this by unchecking the box at the bottom of the page.

You can find the URL of your new instances in the Amazon Explorer. Expand Amazon ECS and Clusters, then click on your cluster.

Troubleshooting the Amazon Toolkit for Visual Studio

The following sections contain general troubleshooting information about the Amazon Toolkit for Visual Studio and working with Amazon services from the toolkit.

> **Note**
>
> Installation and set-up-specific troubleshooting information is available in the Troubleshooting installation issues topic, located in this User Guide.

Topics

- Troubleshooting best practices
- Viewing and filtering Amazon Q security scans
- The Amazon Toolkit isn't properly installed
- Firewall and proxy settings

Troubleshooting best practices

The following are recommended best practices when troubleshooting Amazon Toolkit for Visual Studio issues.

- Repair Visual Studio and restart your system
- Attempt to recreate your issue or error prior to sending a report.
- Take detailed notes of each step, setting, and error message during the recreation process.
- Collect Amazon Toolkit Logs. For a detailed description of how to locate your Amazon Toolkit logs, see the How to locate your Amazon logs procedure, located in this guide topic.
- Check for open requests, known solutions, or report your unresolved issue in the Amazon Toolkit for Visual Studio Issues section of the Amazon Toolkit for Visual Studio GitHub repository.

Repair Visual Studio and Restart your system

1. Close all running instances of Visual Studio.

2. From the Windows start menu, Launch **Visual Studio Installer**.
3. Run Repair on the affected installation(s) of Visual Studio. This allows Visual Studio to rebuild its index of installed extensions.
4. Restart Windows prior to re-launching Visual Studio.

How to locate your Amazon Toolkit logs

1. From the Visual Studio main menu, expand **Extensions**.
2. Choose the **Amazon Toolkit** to expand the Amazon Toolkit menu, then choose **View Toolkit Logs**.
3. When the Amazon Toolkit logs folder opens in your Operating System, sort the files by date and locate any log file that contains information relevant to your current issue.

Viewing and filtering Amazon Q security scans

To view your Amazon Q security scans in Visual Studio, open the Visual Studio **Error List** by expanding the **View** heading in the Visual Studio main menu and choosing **Error List**.

By default, the Visual Studio **Error List** displays all of the warnings and errors for your code base. To filter your Amazon Q security scan findings from the Visual Studio **Error List**, create a filter by completing the following procedure.

> **Note**
>
> Amazon Q security scan findings are only visible after security scan has run and detected issues.
> Amazon Q security scan findings appear as warnings in Visual Studio. In order to view Amazon Q security scan findings from your **Error List**, the **Warnings** option in the **Error List** heading must be selected.

1. From the Visual Studio main menu, expand the **View** heading and choose **Error List** to open the **Error List** pane.
2. From the **Error List** pane, right-click the header row to open the context menu.
3. From the context menu, expand **Show Columns**, then select **Tool** in the expanded menu.

4. The **Tool** column is added to your **Error List**.
5. From the **Tool** column header, select the **Filter** icon and choose Amazon Q to filter for Amazon Q security scan findings.

The Amazon Toolkit isn't properly installed

Issue:

Within one minute after starting Visual Studio, the Amazon Toolkit for Visual Studio the following messages appear in the output pane and info bar, respectively:

```
Some Toolkit components could not be initialized. Some functionality may
not work during this IDE session.

The Amazon Toolkit is not properly installed.
```

Solution:

It's possible that updating or installing an extension caused some of Visual Studio's internal cache files to go out-of-sync. The following procedure describes how to have these files rebuilt the next time that you launch Visual Studio.

> **Note**
>
> It's possible that this solution may impact your Visual Studio customizations. After completing this procedure, the Amazon Toolkit extension should be listed as installed and no longer report an error message. If you continue to experience this issue after completing the following steps, please see Issue #452 in the Amazon Toolkit for Visual Studio GitHub repository for additional information.

1. Install the latest version of Visual Studio 2022.

   > **Note**
   >
   > The minimum required version is 17.11.5.

2. Close all running instances of Visual Studio.

3. From Windows, open the **Developer Command Prompt** as an Administrator.
4. From the **Developer Command Prompt**, run the following command: `devenv /updateconfiguration /resetExtensions`, then wait for the command to finish.
5. After the command has finished, restart Visual Studio.
6. In Visual Studio the Amazon extension is now listed as installed and no longer reports the error messages listed at the top of this issue.

Firewall and proxy settings

Troubleshooting firewall and proxy settings

Security-scanning software can interfere with your ability to download files from Amazon Toolkit language servers by removing files from downloads or preventing downloads altogether.

To check your firewall and proxy settings, navigate to https://aws-toolkit-language-servers.amazonaws.com/codewhisperer/0/manifest.json from an internet browser installed on the same system as your instance of Visual Studio. If you encounter an error or the page is unable to load, then there may be a firewall or proxy filter preventing you from reaching `aws-toolkit-language-servers.amazonaws.com`.

Custom certificates

The Amazon Toolkit for Visual Studio utilizes a language server that runs on the Node.js runtime. For detailed information about how to check if your network uses a custom certificate, see the Configuration and credential file setting in the Amazon CLI topic in the *Amazon Command Line Interface* User Guide for Version 1.

To configure your proxy settings and define a certificate, you must configure your `HTTPS_PROXY` env variable and create Windows Environment Variables for the `NODE_OPTIONS` and `NODE_EXTRA_CA_CERTS` keys.

To configure your `HTTPS_PROXY` env variable, complete the following steps.

1. From the Visual Studio main menu, choose **Tools**, then choose **Options**.
2. From the **Options** menu, expand **Amazon Toolkit**, then choose **Proxy**.
3. From the **Proxy** menu, define your **Host** and **Port**.

Note

For information about configuring the HTTPS_PROXY from the Amazon CLI, see the Using an HTTP proxy for the Amazon CLI topic in the *Amazon Command Line Interface* User Guide.

Create Windows Environment Variables for the following keys.

- `NODE_OPTIONS = --use-openssl-ca`
- `NODE_EXTRA_CA_CERTS = Path/To/Corporate/Certs`

Note

For more information about extracting corporate root certificates, see the Export a certificate with its private key article at *learn.microsoft.com*. For detailed information about the Windows Environment Variable keys, see the Node.js v23.3.0 documentation at *nodejs.org*.

Allow listing and additional steps

In addition to interfering with Amazon Toolkit language servers, firewall settings can prevent Amazon Q from uploading to Amazon S3 and calling the service API. To minimize the potential of these errors, we recommend allowing outbound internet access on **port 443 (HTTPS)** for the following endpoints:

- `https://codewhisperer.us-east-1.amazonaws.com/`
- `https://amazonq-code-transformation-us-east-1-c6160f047e0.s3.amazonaws.com/`
- `https://aws-toolkit-language-servers.amazonaws.com/`
- `https://q.us-east-1.amazonaws.com`
- `https://client-telemetry.us-east-1.amazonaws.com`
- `https://cognito-identity.us-east-1.amazonaws.com`
- `https://oidc.us-east-1.amazonaws.com`

For a detailed list of endpoints, see the Updating firewalls and gateways to allow access topic in this User Guide. If you continue to encounter firewall and proxy issues, then collect your Amazon Toolkit Logs and reach out to the Amazon Toolkit for Visual Studio team through the Amazon Toolkit for Visual Studio issues section of the Amazon Toolkit for Visual Studio GitHub repository. For details on collecting your Amazon Toolkit Logs, review the information in the **Troubleshooting best practices** section of this User Guide topic.

Security for Amazon Toolkit for Visual Studio

Cloud security at Amazon Web Services (Amazon) is the highest priority. As an Amazon customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations. Security is a shared responsibility between Amazon and you. The Shared Responsibility Model describes this as Security of the Cloud and Security in the Cloud.

Security of the Cloud – Amazon is responsible for protecting the infrastructure that runs all of the services offered in the Amazon Cloud and providing you with services that you can use securely. Our security responsibility is the highest priority at Amazon, and the effectiveness of our security is regularly tested and verified by third-party auditors as part of the Amazon Compliance Programs.

Security in the Cloud – Your responsibility is determined by the Amazon service you are using, and other factors including the sensitivity of your data, your organization's requirements, and applicable laws and regulations.

This Amazon product or service follows the shared responsibility model through the specific Amazon Web Services (Amazon) services it supports. For Amazon service security information, see the Amazon service security documentation page and Amazon services that are in scope of Amazon compliance efforts by compliance program.

Topics

- Data Protection in Amazon Toolkit for Visual Studio
- Identity and Access Management
- Compliance Validation for this Amazon Product or Service
- Resilience for this Amazon Product or Service
- Infrastructure Security for this Amazon Product or Service
- Configuration and Vulnerability Analysis in Amazon Toolkit for Visual Studio

Data Protection in Amazon Toolkit for Visual Studio

The Amazon shared responsibility model applies to data protection in Amazon Toolkit for Visual Studio with Amazon Q. As described in this model, Amazon is responsible for protecting the global infrastructure that runs all of the Amazon Web Services Cloud. You are responsible for maintaining

control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the Amazon Web Services services that you use. For more information about data privacy, see the Data Privacy FAQ.

For data protection purposes, we recommend that you protect Amazon Web Services account credentials and set up individual users with Amazon IAM Identity Center or Amazon Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with Amazon resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with Amazon CloudTrail. For information about using CloudTrail trails to capture Amazon activities, see Working with CloudTrail trails in the *Amazon CloudTrail User Guide*.
- Use Amazon encryption solutions, along with all default security controls within Amazon Web Services services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-3 validated cryptographic modules when accessing Amazon through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-3.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with Amazon Toolkit with Amazon Q or other Amazon Web Services services using the console, API, Amazon CLI, or Amazon SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Identity and Access Management

Amazon Identity and Access Management (IAM) is an Amazon Web Services service that helps an administrator securely control access to Amazon resources. IAM administrators control who can

be *authenticated* (signed in) and *authorized* (have permissions) to use Amazon resources. IAM is an Amazon Web Services service that you can use with no additional charge.

Topics

- Audience
- Authenticating with identities
- Managing access using policies
- How Amazon Web Services services work with IAM
- Troubleshooting Amazon identity and access

Audience

How you use Amazon Identity and Access Management (IAM) differs, depending on the work that you do in Amazon.

Service user – If you use Amazon Web Services services to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Amazon features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in Amazon, see Troubleshooting Amazon identity and access or the user guide of the Amazon Web Services service you are using.

Service administrator – If you're in charge of Amazon resources at your company, you probably have full access to Amazon. It's your job to determine which Amazon features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with Amazon, see the user guide of the Amazon Web Services service you are using.

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Amazon. To view example Amazon identity-based policies that you can use in IAM, see the user guide of the Amazon Web Services service you are using.

Authenticating with identities

Authentication is how you sign in to Amazon using your identity credentials. You must be *authenticated* (signed in to Amazon) as the Amazon Web Services account root user, as an IAM user, or by assuming an IAM role.

If you access Amazon programmatically, Amazon provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use Amazon tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see Amazon Signature Version 4 for API requests in the *IAM User Guide*.

Regardless of the authentication method that you use, you might be required to provide additional security information. For example, Amazon recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Amazon Multi-factor authentication in IAM in the *IAM User Guide*.

Amazon Web Services account root user

When you create an Amazon Web Services account, you begin with one sign-in identity that has complete access to all Amazon Web Services services and resources in the account. This identity is called the Amazon Web Services account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the *IAM User Guide*.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access Amazon Web Services services by using temporary credentials.

A *federated identity* is a user from your enterprise user directory, a web identity provider, the Amazon Directory Service, or any user that accesses Amazon Web Services services by using credentials provided through an identity source. When federated identities access Amazon Web Services accounts, they assume roles, and the roles provide temporary credentials.

IAM users and groups

An *IAM user* is an identity within your Amazon Web Services account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the *IAM User Guide*.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see Use cases for IAM users in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your Amazon Web Services account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. To temporarily assume an IAM role in the Amazon Web Services Management Console, you can switch from a user to an IAM role (console). You can assume a role by calling an Amazon CLI or Amazon API operation or by using a custom URL. For more information about methods for using roles, see Methods to assume a role in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Create a role for a third-party identity provider (federation) in the *IAM User Guide*.
- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some Amazon Web Services services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.
- **Cross-service access** – Some Amazon Web Services services use features in other Amazon Web Services services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.

- **Forward access sessions (FAS)** – When you use an IAM user or role to perform actions in Amazon, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an Amazon Web Services service, combined with the requesting Amazon Web Services service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other Amazon Web Services services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.
- **Service role** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an Amazon Web Services service in the *IAM User Guide*.
- **Service-linked role** – A service-linked role is a type of service role that is linked to an Amazon Web Services service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your Amazon Web Services account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making Amazon CLI or Amazon API requests. This is preferable to storing access keys within the EC2 instance. To assign an Amazon role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Use an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

Managing access using policies

You control access in Amazon by creating policies and attaching them to Amazon identities or resources. A policy is an object in Amazon that, when associated with an identity or resource, defines their permissions. Amazon evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in Amazon as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use Amazon JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the Amazon Web Services Management Console, the Amazon CLI, or the Amazon API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your Amazon Web Services account. Managed policies include Amazon managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choose between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or Amazon Web Services services.

Resource-based policies are inline policies that are located in that service. You can't use Amazon managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, Amazon WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access control list (ACL) overview in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

Amazon supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of an entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in Amazon Organizations. Amazon Organizations is a service for grouping and centrally managing multiple Amazon Web Services accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each Amazon Web Services account root user. For more information about Organizations and SCPs, see Service control policies in the *Amazon Organizations User Guide*.
- **Resource control policies (RCPs)** – RCPs are JSON policies that you can use to set the maximum available permissions for resources in your accounts without updating the IAM policies attached to each resource that you own. The RCP limits permissions for resources in member accounts and can impact the effective permissions for identities, including the Amazon Web Services account root user, regardless of whether they belong to your organization. For more information about Organizations and RCPs, including a list of Amazon Web Services services that support RCPs, see Resource control policies (RCPs) in the *Amazon Organizations User Guide*.

- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how Amazon determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How Amazon Web Services services work with IAM

To get a high-level view of how Amazon Web Services services work with most IAM features, see Amazon services that work with IAM in the *IAM User Guide*.

To learn how to use a specific Amazon Web Services service with IAM, see the security section of the relevant service's User Guide.

Troubleshooting Amazon identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with Amazon and IAM.

Topics

- I am not authorized to perform an action in Amazon
- I am not authorized to perform iam:PassRole
- I want to allow people outside of my Amazon Web Services account to access my Amazon resources

I am not authorized to perform an action in Amazon

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional *`my-example-widget`* resource but doesn't have the fictional `awes:GetWidget` permissions.

```
User: arn:aws-cn:iam::123456789012:user/mateojackson is not authorized to perform:
 awes:GetWidget on resource: my-example-widget
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the *`my-example-widget`* resource by using the `awes:GetWidget` action.

If you need help, contact your Amazon administrator. Your administrator is the person who provided you with your sign-in credentials.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to Amazon.

Some Amazon Web Services services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in Amazon. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws-cn:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your Amazon administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my Amazon Web Services account to access my Amazon resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support

resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether Amazon supports these features, see How Amazon Web Services services work with IAM.
- To learn how to provide access to your resources across Amazon Web Services accounts that you own, see Providing access to an IAM user in another Amazon Web Services account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party Amazon Web Services accounts, see Providing access to Amazon Web Services accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.

Compliance Validation for this Amazon Product or Service

To learn whether an Amazon Web Services service is within the scope of specific compliance programs, see Amazon Web Services services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see Amazon Web Services Compliance Programs.

You can download third-party audit reports using Amazon Artifact. For more information, see Downloading Reports in Amazon Artifact.

Your compliance responsibility when using Amazon Web Services services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. Amazon provides the following resources to help with compliance:

- Security & Compliance – These solution implementation guides discuss architectural considerations and provide steps for deploying security and compliance features.
- Amazon Compliance Resources – This collection of workbooks and guides might apply to your industry and location.

- Evaluating Resources with Rules in the *Amazon Config Developer Guide* – The Amazon Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- Amazon Security Hub – This Amazon Web Services service provides a comprehensive view of your security state within Amazon. Security Hub uses security controls to evaluate your Amazon resources and to check your compliance against security industry standards and best practices. For a list of supported services and controls, see Security Hub controls reference.
- Amazon GuardDuty – This Amazon Web Services service detects potential threats to your Amazon Web Services accounts, workloads, containers, and data by monitoring your environment for suspicious and malicious activities. GuardDuty can help you address various compliance requirements, like PCI DSS, by meeting intrusion detection requirements mandated by certain compliance frameworks.

This Amazon product or service follows the shared responsibility model through the specific Amazon Web Services (Amazon) services it supports. For Amazon service security information, see the Amazon service security documentation page and Amazon services that are in scope of Amazon compliance efforts by compliance program.

Resilience for this Amazon Product or Service

The Amazon global infrastructure is built around Amazon Web Services Regions and Availability Zones.

Amazon Web Services Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking.

With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about Amazon Regions and Availability Zones, see Amazon Global Infrastructure.

This Amazon product or service follows the shared responsibility model through the specific Amazon Web Services (Amazon) services it supports. For Amazon service security information, see the Amazon service security documentation page and Amazon services that are in scope of Amazon compliance efforts by compliance program.

Infrastructure Security for this Amazon Product or Service

This Amazon product or service uses managed services, and therefore is protected by the Amazon global network security. For information about Amazon security services and how Amazon protects infrastructure, see Amazon Cloud Security. To design your Amazon environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar Amazon Well-Architected Framework*.

You use Amazon published API calls to access this Amazon Product or Service through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the Amazon Security Token Service (Amazon STS) to generate temporary security credentials to sign requests.

This Amazon product or service follows the shared responsibility model through the specific Amazon Web Services (Amazon) services it supports. For Amazon service security information, see the Amazon service security documentation page and Amazon services that are in scope of Amazon compliance efforts by compliance program.

Configuration and Vulnerability Analysis in Amazon Toolkit for Visual Studio

The Toolkit for Visual Studio is released to the Visual Studio Marketplace as new features or fixes are developed. These updates sometimes include security updates, so it's important to keep Amazon Toolkit with Amazon Q up to date.

To verify that automatic updates for extensions are enabled

1. Open the extensions manager by choosing **Tools**, **Extensions and Updates** (Visual Studio 2017), or **Extensions**, **Manage Extensions** (Visual Studio 2019).

2. Choose **Change your Extensions and Updates settings** (Visual Studio 2017), or **Change your settings for Extensions** (Visual Studio 2019).
3. Adjust the settings for your environment.

If you choose to disable automatic updates for extensions, be sure to check for updates to Amazon Toolkit with Amazon Q at intervals that are appropriate for your environment.

Document history of the Amazon Toolkit for Visual Studio User Guide

Document history

The following table describes the important recent changes of the Amazon Toolkit for Visual Studio User Guide. For notification about updates to this documentation, you can subscribe to an RSS feed.

Change	Description	Date
Updates to Getting Started Content	Updates made to Getting Started and Connecting to Amazon content to reflect changes made in the UI.	April 24, 2025
Updating firewalls and gateways to allow access	Lists of endpoints and resources that must be allow listed to access all service and features in the Amazon Toolkit for Visual Studio with Amazon Q for extensions.	March 20, 2025
Troubleshooting Firewall and proxy settings	Added new troubleshooting topic addressing firewall and proxy settings for the Amazon Toolkit for Visual Studio and Amazon Q.	December 15, 2024
Troubleshooting installation update	Updating installation issue content to account for an update from Microsoft.	November 20, 2024
Updates to Getting Started Content	Updates made to Getting Started and Connecting to	October 24, 2024

	Amazon content to reflect changes made in the UI.	
Updates to Connecting to Amazon	Updates made to Connecting to Amazon content.	September 26, 2024
Updates to the Amazon EC2 AMI content	Content updates have been made to document changes to the Amazon EC2 AMI process and procedures.	September 13, 2024
Amazon Toolkit components could not be initialized	Added troubleshooting topic to address issues with Amazon Toolkit for Visual Studio components not initializing.	September 13, 2024
Viewing and filtering Amazon Q security scans	Added a troubleshooting topic to assist with viewing and filtering Amazon Q security scans.	July 31, 2024
Amazon Q for Amazon Toolkit for Visual Studio	Amazon Q is now available for the Amazon Toolkit for Visual Studio.	June 30, 2024
Content updates and maintenance	Updating content for changes to the UI and Amazon style guidelines.	March 6, 2024
Content updates and maintenance	Updating content for changes to the UI and Amazon style guidelines.	March 6, 2024
Content updates and maintenance	Updating content for changes to the UI and Amazon style guidelines.	March 6, 2024

Content updates and maintenance	Updating content for changes to the UI and Amazon style guidelines.	March 6, 2024
Content updates and maintenance	Updating content for changes to the UI and Amazon style guidelines.	March 6, 2024
Updates to set up and authentication	The set up and authentication topics have been updated to improve security and the toolkit onboarding experienc e. See the Getting Started and Authentication and access topic TOCs to view changes.	June 22, 2023
Authentication and access	Providing Amazon credentia ls is now Authentication and access. Refactoring TOC and subtopics to meet Amazon style and secuirty requireme nts.	May 4, 2023
Updates to the Setting up sections and topics	The Setting up the Amazon Toolkit for Visual Studio sections and topics in this User Guide have been updated to improve the on boarding experience for the Amazon Toolkit for Visual Studio.	January 30, 2023

Updates to the Setting up sections and topics	The Setting up the Amazon Toolkit for Visual Studio sections and topics in this User Guide have been updated to improve the on boarding experience for the Amazon Toolkit for Visual Studio.	January 30, 2023
Added 2022 Amazon Toolkit for Visual Studio information	Support for Visual Studio 2022 was added to the Amazon Toolkit for Visual Studio.	December 20, 2022
Updates to Publish to Amazon guide	Documentation updates to reflect changes made to service for GA launch.	July 6, 2022
Title updates and relocation	Minor title changes were made to better reflect content. Guide is now located in the Publishing to Amazon guide.	July 6, 2022

Deploying to Amazon: title and content updates	The guide section formally titled: Deployment Using the Amazon Toolkit, has an updated table of contents (TOC) and is now titled: Deploying to Amazon. The following guides have completed deprecation and are no longer accessible: Deploying to Elastic Beanstalk (Legacy) and Deploying to Amazon CloudFormation (Legacy). Updated content regarding deployment to Elastic Beanstalk and Cloudformation can be found from the updated TOC in this guide.	July 6, 2022
Deploying an ASP.NET Core 2.0 App (Fargate) is now a legacy guide	This documentation refers to legacy services and features. For updated guides and content, see the Amazon .NET Deployment tool guide and the updated Deploying to Amazon table of contents.	July 6, 2022
Deploy an ASP.NET App is now a legacy guide	This documentation refers to legacy services and features. For updated guides and content, see the Amazon .NET deployment tool guide and the updated Deploying to Amazon table of contents.	July 6, 2022

Deploy an ASP.NET App is now a legacy guide	This documentation refers to legacy services and features. For updated guides and content, see the Amazon .NET deployment tool guide and the updated Deploying to Amazon table of contents.	July 6, 2022
New guide topic: Working with CloudWatch Logs in Visual Studio	Created new overview topic for the Amazon CloudWatch Logs integration in Visual Studio guide.	June 29, 2022
New guide topic: Setting up CloudWatch Logs integration for Visual Studio	Created new set-up section for the Amazon CloudWatch Logs integration in Visual Studio guide.	June 29, 2022
CloudWatch Logs integration for Visual Studio	Created new guide for Amazon CloudWatch Logs integration in Visual Studio, including guide topics: Setting up CloudWatch Logs for Visual Studio and Working with CloudWatch Logs in Visual Studio.	June 29, 2022
Publish to Amazon	Publish to Amazon is no longer in preview. Updates to reflect changes to UI and improvements to the publishing suggestions.	June 1, 2022

New Publish to Amazon available for preview	Enhanced deployment experience that provides guidance on which Amazon service is right for your application.	October 21, 2021
SSO and MFA support for Amazon credentials	Updated to document new support for Amazon Single Sign-On (IAM Identity Center) and multi-factor authentic ation in Amazon credentials.	April 21, 2021
Basic Amazon Lambda Project Creating Docker Image	Added support for Lambda container images.	December 1, 2020
Security Content	Added security content.	February 6, 2020
Providing Amazon credentials	Updated with information about creating credential profiles in the shared Amazon credentials file.	June 20, 2019
Using the Amazon Lambda Project in the Amazon Toolkit for Visual Studio	Support for Visual Studio 2019 was added to the Amazon Toolkit for Visual Studio.	March 28, 2019
Tutorial: Creating an Amazon Rekognition Lambda Applicati on	Support for Visual Studio 2019 was added to the Amazon Toolkit for Visual Studio.	March 28, 2019
Tutorial: Build and Test a Serverless Application with Amazon Lambda	Support for Visual Studio 2019 was added to the Amazon Toolkit for Visual Studio.	March 28, 2019

Setting Up the Amazon Toolkit for Visual Studio	Support for Visual Studio 2019 was added to the Amazon Toolkit for Visual Studio.	March 28, 2019
Deploying an ASP.NET Core 2.0 App (Fargate)	Support for Visual Studio 2019 was added to the Amazon Toolkit for Visual Studio.	March 28, 2019
Deploying an ASP.NET Core 2.0 App (EC2)	Support for Visual Studio 2019 was added to the Amazon Toolkit for Visual Studio.	March 28, 2019
Creating an Amazon CloudFormation Template Project in Visual Studio	Support for Visual Studio 2019 was added to the Amazon Toolkit for Visual Studio.	March 28, 2019
Detailed Views of Container Service	Added information about the detailed views of Amazon Elastic Container Service clusters and container repositories that are provided by Amazon Explorer.	February 16, 2018
Deploying to Amazon EC2 Container Service	Added information about deploying to Amazon EC2 container service.	February 16, 2018
Deploying Container Service using Fargate	Added information about how to deploy a containerized ASP.NET Core 2.0 applicati on targeting Linux through Amazon ECS using the Fargate launch type.	February 16, 2018

Deploying Container Service using EC2	Added information about how to deploy a containerized ASP.NET Core 2.0 applicati on targeting Linux through Amazon ECS using the EC2 launch type.	February 16, 2018
Credentials for Deploying to Amazon EC2 Container Service	Added information about how to specify credentials when deploying to Amazon EC2 container service.	February 16, 2018