aws

用户指南

AWS 带有 Amazon Q 的工具包

AWS 带有 Amazon Q 的工具包: 用户指南

Table of Contents

AWS 带有 Amazon Q 的工具包

这是 AWS Toolkit for Visual Studio with Amazon Q 的用户指南。如果您要找AWS Toolkit for VS Code，请参阅《 AWS Toolkit for Visual Studio Code用户指南》。

搭载 Amazon Q 的 Visual Studio 的 Visual Studio AWS 工具包是什么

带有 Amazon Q 的 Visual Studio AWS 工具包是 Visual Studio IDE 的扩展，它使您可以更轻松地开发、调试和部署使用亚马逊 Web Services 的.NET 应用程序。Visual Studio 2022 及更高版本支持带有 Amazon Q 的 AWS 工具包。有关如何下载和安装该套件的详细信息，请参阅本用户指南中的安装和设置主题。

> **Note**
>
> Toolkit for Visual Studio 也曾发布过适用于 Visual Studio 2008、2010、2012、2013、2015、2017 和 2019 的版本。但是，这些版本已不再受支持。有关更多信息，请参阅本用户指南中的安装和设置主题。

带有 Amazon Q 的 AWS 工具包包含以下功能，可增强您的开发体验。

AWS 探险家

可以在 IDE 的 “查看” 菜单中访问 AWS 资源管理器工具窗口，并允许您与 Visual Studio 中的 AWS 服务进行交互。有关支持的 AWS 服务和功能的列表，请参阅本用户指南中的使用 AWS 服务主题。

Amazon Q

在 Visual Studio 中与 Amazon Q Developer 交谈，询问有关在软件开发方面进行构建的问题， AWS 并寻求软件开发方面的帮助。Amazon Q 可以解释编码概念和代码片段，生成代码和单元测试，并通过调试或重构来改进代码。

要为 Toolkit for Visual Studio 安装和设置 Amazon Q，请参阅本用户指南中的开始使用主题。要了解有关与 Amazon Q 开发者合作的更多信息，请参阅 Amazon Q 开发者用户指南中 IDEs主题中的 Amazon Q 开发人员。有关 Amazon Q 计划和定价的详细信息，请参阅 Amazon Q 定价指南。

相关信息

要打开议题或查看当前未解决的问题，请访问 https://github.com/aws/aws-toolkit-visual-studio/issues。

要了解有关 Visual Studio 的更多信息，请访问https://visualstudio.microsoft.com/vs/。

Amazon Q

什么是 Amazon Q？

自 2024 年 4 月 30 日起，Amazon CodeWhisperer 已正式成为 Amazon Q 开发者版的一部分，其行内代码建议和安全扫描等功能也保留并转移到后者。

要了解有关在 AWS Toolkit for Visual Studio 中使用 Amazon Q 开发者版的更多信息，请参阅《Amazon Q 开发者版用户指南》中的在 IDE 中使用 Amazon Q 开发者版主题。有关 Amazon Q 计划和定价的详细信息，请参阅 Amazon Q 定价指南。

下载 Toolkit for Visual Studio

您可以通过 IDE 中的 Visual Studio Marketplace 下载、安装和设置 Toolkit for Visual Studio。有关详细说明，请参阅本用户指南入门主题中的安装 AWS Toolkit for Visual Studio 部分。

从 Visual Studio Marketplace 下载 Toolkit

在网络浏览器中导航到 AWS Visual Studio 下载站点，下载 Toolkit for Visual Studio 安装文件。

其他来自 AWS 的 IDE 工具包

除了 Toolkit for Visual Studio 之外，AWS 还提供了适用于 VS Code 和 JetBrains 的 IDE 工具包。

AWS Toolkit for Visual Studio Code 链接

- 点击此链接，从 VS Code Marketplace 下载 AWS Toolkit for Visual Studio Code。
- 要了解有关 AWS Toolkit for Visual Studio Code 的更多信息，请参阅 AWS Toolkit for Visual Studio Code 用户指南。

AWS Toolkit for JetBrains 链接

- 点击此链接，从 JetBrains Marketplace 下载 AWS Toolkit for JetBrains。
- 要了解有关 AWS Toolkit for JetBrains 的更多信息，请参阅 AWS Toolkit for JetBrains 用户指南。

入门

借助 AWS Toolkit for Visual Studio，您可以直接通过 Visual Studio 集成式开发环境（IDE）使用 AWS 服务和资源。

为了帮助您入门，以下主题介绍了如何安装、设置和配置 AWS Toolkit for Visual Studio。

主题

安装和设置 AWS Toolkit for Visual Studio

以下主题介绍如何下载、安装、设置和卸载 AWS Toolkit for Visual Studio。

主题

先决条件

以下是设置支持的 AWS Toolkit for Visual Studio版本的先决条件。

- Visual Studio 19 或更高版本
- Windows 10 或更高版本的 Windows
- 对 Windows 和 Visual Studio 的管理员访问权限
- 有效 AWS 的 IAM 证书

> **Note**
>
> 不支持的版本适用于 Visual Studio 2008、2010、2012、2013、2015 和 2017。 AWS Toolkit for Visual Studio 要下载不带支持的版本，请导航到 AWS Toolkit for Visual Studio 登录页面，然后从下载链接列表中选择所需的版本。
> 要了解有关 IAM 凭证的更多信息或注册账户，请访问 AWS 控制台门户。

正在安装 AWS Toolkit for Visual Studio

要安装 AWS Toolkit for Visual Studio，请从以下步骤中找到您的 Visual Studio 版本并完成必要的步骤。 AWS Toolkit for Visual Studio 可在AWS Toolkit for Visual Studio登录页面上找到所有版本的下载链接。

> **Note**
>
> 如果您在安装时遇到问题 AWS Toolkit for Visual Studio，请参阅本指南中的安装问题疑难解答主题。

AWS Toolkit for Visual Studio 为 Visual Studio 2022 安装

要从 Visual Studio 安装 AWS Toolkit for Visual Studio 2022，请完成以下步骤：

1. 在主菜单中导航到扩展，然后选择管理扩展。
2. 在搜索框中搜索 AWS。
3. 选择 Visual Studio 2022 相关版本的下载按钮，然后按照安装提示进行操作。

   > **Note**
   >
   > 您可能需要手动关闭并重新启动 Visual Studio 才能完成安装过程。

4. 下载和安装完成后，您可以从 “查看” 菜单中选择 “AWS 资源管理器” AWS Toolkit for Visual Studio 来打开。

AWS Toolkit for Visual Studio 为 Visual Studio 2019 安装

要从 Visual Studio 安装 AWS Toolkit for Visual Studio 2019，请完成以下步骤：

1. 在主菜单中导航到扩展，然后选择管理扩展。
2. 在搜索框中搜索 AWS。
3. 选择 Visual Studio 2017 和 2019 的下载按钮，然后按照提示进行操作。

> **Note**
>
> 您可能需要手动关闭并重新启动 Visual Studio 才能完成安装过程。

4. 下载和安装完成后，您可以从 “查看” 菜单中选择 “AWS 资源管理器” AWS Toolkit for Visual Studio 来打开。

正在卸载 AWS Toolkit for Visual Studio

要卸载 AWS Toolkit for Visual Studio，请从以下步骤中找到您的 Visual Studio 版本并完成必要的步骤。

正在卸载 Visual Studio 2022 版 AWS Toolkit for Visual Studio

要从 Visual Studio 卸载 AWS Toolkit for Visual Studio 2022，请完成以下步骤：

1. 在主菜单中导航到扩展，然后选择管理扩展。
2. 在管理扩展导航菜单中，展开已安装标题。
3. 找到 AWS Toolkit for Visual Studio 2022 扩展并选择卸载按钮。

> **Note**
>
> 如果在导航菜单的 “已安装” 部分中看 AWS Toolkit for Visual Studio 不到，则可能需要重新启动 Visual Studio。

4. 按照屏幕提示完成卸载过程。

正在卸载 Visual Studio 2019 版 AWS Toolkit for Visual Studio

要从 Visual Studio 中卸载 AWS Toolkit for Visual Studio 2019，请完成以下步骤：

1. 在主菜单中导航到工具，然后选择管理扩展。
2. 在管理扩展导航菜单中，展开已安装标题。

3. 找到 AWS Toolkit for Visual Studio 2019 扩展并选择卸载按钮。
4. 按照屏幕提示完成卸载过程。

正在卸载 Visual Studio 2017 版 AWS Toolkit for Visual Studio

要在 Visual Studio 中卸载 AWS Toolkit for Visual Studio 2017，请完成以下步骤：

1. 在主菜单中，导航到工具，然后选择扩展和更新。
2. 在扩展和更新导航菜单中，展开已安装标题。
3. 找到 AWS Toolkit for Visual Studio 2017 扩展并选择卸载按钮。
4. 按照屏幕提示完成卸载过程。

正在卸载 Visual Studio 2013 或 2015 版 AWS Toolkit for Visual Studio

要卸载 AWS Toolkit for Visual Studio 2013 或 2015，请完成以下步骤：

1. 在 Windows 控制面板中，打开程序和功能。

> **Note**
>
> 您可以通过 Windows 命令提示符或 Windows 运行对话框运行 `appwiz.cpl`，立即打开“程序和功能”。

2. 从已安装程序列表中，打开 AWS Tools for Windows 的上下文菜单（右键单击）。
3. 选择卸载，然后按照提示完成卸载过程。

> **Note**
>
> 在卸载过程中，您的 Samples 目录不会被删除。如果您修改了示例，则此目录会保留。必须手动移除此目录。

正在连接到 AWS

以下各节介绍如何通过 Amazon Q 开始使用 Visual Studio for Visual Studio AWS 工具包。安装扩展程序后首次启动 Visual Studio 时，编辑器窗口中会显示 “入门”。在开始使用选项卡中，您可以完成以下操作。

- 启用或禁用 Amazon Q 和 AWS 工具包。
- 添加新凭证并使用新凭证进行身份验证。
- 使用现有凭证进行身份验证。
- 访问文档和教程以帮助您开始使用 Amazon Q 和 AWS Toolkit。

先决条件

要开始使用 Amazon Q 和 AWS Toolkit，您需要使用 AWS 证书进行身份验证。如果您之前设置过 AWS 账户并通过其他 AWS 工具或服务（例如 AWS Command Line Interface）进行身份验证，则 AWS Toolkit 会自动检测您的凭据。如果您是新用户 AWS 或尚未创建帐户，则可以从注册门户AWS注册一个 AWS 帐户。有关设置新 AWS 帐户的详细信息，请参阅《AWS 设置用户指南》中的概述主题。

AWS 从工具包连接到

要通过 AWS Toolkit 连接到您的 AWS 帐户，请随时通过完成以下操作打开 “入门” 选项卡。

在 Visual Studio 中打开开始使用选项卡

1. 在 Visual Studio 中，从主菜单展开扩展程序，然后展开 AWS 工具包子菜单。
2. 选择开始使用。
3. 开始使用选项卡将在 Visual Studio 编辑器窗口中打开。

在开始使用选项卡中，有两个主要部分：

- 功能：在本节中，您可以启用或禁用 Amazon Q 和 AWS 工具包等功能。
- 文档和教程：您已启用功能的参考汇总。

> **Note**
>
> 只有启用一项或多项功能后，才能看到“文档和教程”部分。

Amazon Q 开发者版

在开始使用选项卡的“Amazon Q”部分，您可以启用或禁用 Amazon Q、添加新连接或切换到其他 AWS 连接。必须先启用 Amazon Q，然后才能查看或访问这些操作。要启用 Amazon Q，请单击启用按钮。

禁用 Amazon Q 后，所有 Amazon Q 特性和功能都将从 Visual Studio 中完全删除。启用 Amazon Q 会自动在开始使用选项卡中打开为 Amazon Q 设置身份验证。要继续操作，您必须使用 AWS IAM Identity Center 凭据进行身份验证才能访问专业级别，或者使用 AWS 建筑商 ID 进行身份验证才能访问免费套餐。有关每种套餐选项的详细信息，请参阅《Amazon Q 开发者版用户指南》中的了解 Amazon Q 开发者版的服务套餐主题。

要继续，请完成下列过程之一。

使用 IAM Identity Center 进行专业套餐访问认证

> **Note**
>
> 专业套餐访问认证所需的配置文件名称、起始 URL、配置文件区域或 SSO 区域字段通常由您公司或组织的管理员提供。有关 IAM Identity Center 凭证的详细信息，请参阅《AWS IAM Identity Center 用户指南》中的什么是 IAM Identity Center？主题。

1. 在 “入门：带有 Amazon Q 的 AWS 工具包” 屏幕中，选择 Amazon Q 图块中的登录按钮，导航到 Amazon Q 的设置身份验证屏幕。
2. 在为 Amazon Q 设置身份验证屏幕上，导航到专业套餐部分，填写必需字段并选择连接按钮。
3. 确认您要在默认 Web 浏览器中打开 AWS 授权请求门户。
4. 完成 AWS 授权请求门户网站要求的步骤，当可以安全关闭浏览器并返回 Visual Studio 时，你会收到通知
5. 完成该过程后，在开始使用选项卡中，Amazon Q 会更新显示，告知您已通过 IAM Identity Center 成功连接。

使用 AWS 建造者 ID 进行免费等级认证

> **Note**
>
> 有关 AWS 生成器 ID 的更多详细信息，请参阅《登录用户指南》中的 “使用 AWS 生成器 ID AWS 登录” 主题。

1. 在 “入门：带有 Amazon Q 的 AWS 工具包” 屏幕中，选择 Amazon Q 图块中的登录按钮，导航到 Amazon Q 的设置身份验证屏幕。
2. 在为 Amazon Q 设置身份验证屏幕上，导航至免费套餐部分，然后选择注册或登录按钮。

3. 确认您要在默认 Web 浏览器中打开 AWS 授权请求门户。
4. 完成 AWS 授权请求门户网站要求的步骤，当可以安全关闭浏览器并返回 Visual Studio 时，你会收到通知。
5. 在“入门”选项卡中，Amazon Q 会更新，在流程完成后显示您已与 AWS 建筑商 ID 相关联。

使用 IAM 身份中心或 AWS 生成器 ID 凭证进行身份验证后，您可以在 Visual Studio 中访问 Amazon Q。此外，您还可以在开始使用选项卡中执行以下操作：

- 注销：断开您当前的凭证与所有 Amazon Q 功能的连接。Amazon Q 仍处于启用状态，但大多数功能无法使用。
- 禁用 Amazon Q：完全禁用 Visual Studio 中的所有 Amazon Q 功能。

AWS 工具包

在“ AWS 工具包入门”选项卡的“ AWS 工具包”部分，您可以启用或禁用 AWS Toolkit、添加新连接或切换到其他 AWS 连接。必须先启用 AWS Toolkit，然后才能查看或访问这些操作中的任何一个。要启用该 AWS 工具包，请单击“启用”按钮。

启用 AWS Toolkit 后， AWS Toolkit 的安装身份验证会自动加载到“ AWS 工具包入门”选项卡中。要继续，您必须使用您的 AWS IAM Identity Center 凭证或 IAM 用户角色凭证进行身份验证。

> **Note**
>
> 有关 IAM Identity Center 凭证的详细信息，请参阅《AWS IAM Identity Center 用户指南》中的什么是 IAM Identity Center？主题。有关 IAM 用户角色证书的详细信息，请参阅AWS SDKs 和工具参考指南中的AWS 访问密钥：长期证书主题。

通过 IAM Identity Center 进行身份验证和连接

1. 在“入门：带有 Amazon Q 的 AWS Toolkit”屏幕中，选择 AWS Toolkit 磁贴中的登录按钮，导航到 T AWS oolkit 的设置身份验证屏幕。
2. 在为 AWS Toolkit 设置身份验证屏幕中，从配置文件类型下拉菜单中选择 IAM 身份中心（单点登录的继任者）。
3. 在从现有配置文件中选择或添加新配置文件下拉菜单中，选择一个现有配置文件，或选择添加新配置文件以添加新的配置文件信息。

> **Note**
>
> 如果选择现有配置文件，请转到步骤 7。

4. 在配置文件名称字段中，输入您要用于身份验证的关联 IAM Identity Center 账户的 **`profile name`**。
5. 在起始 URL 文本字段中，输入附加到您的 IAM Identity Center 凭证的 **`Start URL`**。
6. 从配置文件区域（默认为 us-east-1）下拉菜单中，选择由您正在用于身份验证的 IAM Identity Center 用户配置文件定义的配置文件区域。
7. 在 SSO 区域（默认为 us-east-1）下拉菜单中，选择由您的 IAM Identity Center 凭证定义的 SSO 区域。
8. 选择连接按钮，在您的默认 Web 浏览器中打开 AWS 授权请求站点。
9. 按照默认 Web 浏览器中的提示操作，当授权流程完成时您会收到通知，届时您即可安全地关闭浏览器并返回 Visual Studio。
10. 完成该过程后，在开始使用选项卡中， AWS Toolkit 部分会更新显示，告知您已通过 IAM Identity Center 成功连接。

使用 IAM 用户角色凭证进行身份验证和连接

1. 在 “入门：带有 Amazon Q 的 AWS Toolkit” 屏幕中，选择 AWS Toolkit 磁贴中的登录按钮，导航到 T AWS oolkit 的设置身份验证屏幕。
2. 在 AWS Toolkit 的设置身份验证屏幕中，从配置文件类型下拉菜单中选择 IAM 用户角色。
3. 在从现有配置文件中选择或添加新配置文件下拉菜单中，选择 **`Add new profile`**。

> **Note**
>
> 如果您要从列表中选择现有的配置文件名称，请跳至步骤 8。

4. 在配置文件名称文本字段中，输入新配置文件的名称。
5. 在访问密钥 ID 文本字段中，为用于身份验证的配置文件输入 **`Access Key ID`**。
6. 在密钥文本字段中，为用于身份验证的配置文件输入 **`Secret Key`**。
7. 从存储位置（默认为“共享的凭证文件”）下拉菜单中，指定是使用共享的凭证文件还是 .NET 加密存储来存储您的凭证。

8. 从配置文件区域（默认为 us-east-1）下拉菜单中，选择附加到用于身份验证的配置文件的分区和配置文件区域。
9. 选择 Connect 按钮，将此配置文件添加到您的 AWS 存储位置 and/or 进行身份验证 AWS。
10. 完成该过程后，在开始使用选项卡中， AWS Toolkit 部分会更新显示，告知您已通过 IAM 用户角色凭证成功连接。

使用 IAM 身份中心或 IAM 用户角色证书进行身份验证后，您可以在 Visual Studio 的 Toolkit for Visual Studio 中访问 AWS 资源管理器。此外，您还可以从开始使用选项卡中注销并禁用 AWS Toolkit for Visual Studio with Amazon Q。

文档和教程

根据您的 AWS 服务和功能偏好，文档和教程部分会自动更新文档和教程建议。这些参考资料只有在您启用了至少一项功能后才会显示。

排查 AWS Toolkit for Visual Studio 安装问题

已知以下信息可以解决设置 AWS Toolkit for Visual Studio 时的常见安装问题。

如果您在安装 AWS Toolkit for Visual Studio 时遇到错误，或者不清楚安装是否已完成，请查看以下各部分中的信息。

Visual Studio 管理员权限

AWS Toolkit for Visual Studio 扩展需要管理员权限才能确保可访问所有 AWS 服务和功能。

如果您拥有本地管理员权限，则您的管理员权限可能无法直接扩展到 Visual Studio 实例。

要使用管理员权限在本地启动 Visual Studio，请执行以下操作：

1. 在 Windows 中，找到 Visual Studio 应用程序启动器（图标）。
2. 右键单击 Visual Studio 图标，以打开上下文菜单。
3. 从上下文菜单中选择以管理员身份运行。

要使用管理员权限远程启动 Visual Studio，请执行以下操作：

1. 在 Windows 中，找到您用于连接到远程 Visual Studio 实例的应用程序的应用程序启动器。
2. 右键单击应用程序，以打开上下文菜单。

3. 从上下文菜单中选择以管理员身份运行。

> **Note**
>
> 无论您是在本地启动程序还是远程连接到程序，Windows 都可能提示您确认管理凭证。

获取安装日志

如果您已经完成了上述管理员权限部分中的步骤，并且确认您正在使用管理员权限运行或连接到 Visual Studio，那么获取安装日志文件可帮助您诊断其他问题。

要通过 `.vsix` 文件手动安装 AWS Toolkit for Visual Studio 并生成安装日志文件，请完成以下步骤。

1. 在 AWS Toolkit for Visual Studio 登录页面上，点击下载链接，保存要安装的 AWS Toolkit for Visual Studio 版本的 `.vsix` 文件。
2. 从 Visual Studio 主菜单中，展开工具标题，展开命令行子菜单，然后选择 Visual Studio 开发人员命令提示。
3. 在 Visual Studio 开发人员命令提示中输入以下格式的 `vsixinstaller` 命令：

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

4. 将 `[file path to log file]` 替换为用于创建安装日志的文件名和目录的完整文件路径。使用您指定的文件路径和文件名的 `vsixinstaller` 命令示例类似于以下内容：

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

5. 将 `[file path to Toolkit installation file]` 替换为 `AWSToolkitPackage.vsix` 所在目录的完整文件路径。

   包含 Toolkit 安装文件完整文件路径的 `vsixinstaller` 命令示例应类似于以下内容：

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

6. 检查以确保文件名和路径正确，然后运行 `vsixinstaller` 命令。

   完整 `vsixinstaller` 命令的示例类似于以下内容：

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

安装其他 Visual Studio 扩展

如果您已获得安装日志文件，但仍无法确定安装过程失败的原因，请检查是否能够安装其他 Visual Studio 扩展。安装其他 Visual Studio 扩展可以进一步了解安装问题。如果您无法安装任何 Visual Studio 扩展，则可能需要改为排查 Visual Studio 的问题，而不是 AWS Toolkit for Visual Studio 的问题。

联系支持人员

如果您已阅读本指南的所有部分，但仍需要其他资源或支持，则可以前往 AWS Toolkit for Visual Studio Github Issues 站点查看过去的问题或提交新问题。

为了帮助加快问题的解决，请执行以下操作：

- 查看过去和当前的问题，看看其他人是否遇到过类似的情况。
- 详细记录您为解决问题而采取的每个步骤。
- 保存您在安装 AWS Toolkit for Visual Studio 扩展或其他扩展时获得的所有日志文件。
- 将您的 AWS Toolkit for Visual Studio 安装日志文件附在新问题中。

配置文件和窗口绑定

Toolkit for Visual Studio 的配置文件和窗口绑定

在使用 Toolkit for Visual Studio 的发布工具、向导和其他功能时，请注意以下几点：

- 一次将 AWS 各区服务浏览器窗口绑定到一个配置文件和区域。从 AWS 各区服务浏览器打开的窗口将默认使用该绑定的配置文件和区域。
- 打开新窗口后，您可以使用该 AWS 各区服务浏览器实例切换到其他配置文件或区域。
- Toolkit for Visual Studio 发布工具和功能会自动默认使用在 AWS 各区服务浏览器中设置的配置文件和区域。
- 如果在发布工具、向导或功能中指定了新的配置文件或区域，则之后创建的所有资源将继续使用新的配置文件和区域设置。

- 如果您打开了多个 Visual Studio 实例，则每个实例都可绑定到不同的配置文件和区域。
- AWS 各区服务浏览器会保存上次指定的配置文件和区域，最后关闭的 Visual Studio 实例的值将保留。

身份验证和访问

您无需进行身份验证即可开始使用 AWS 适用于 Visual Studio 的 AWS Toolkit for Amazon Q。但是，大多数 AWS 资源都是通过 AWS 账户管理的。要使用所有带有 Amazon Q 服务和功能的 Visual Studio Toolkit for Visual Studio AWS 工具包，您至少需要两种类型的账户身份验证：

1. 对您的 AWS 账户进行AWS Identity and Access Management (IAM) AWS IAM Identity Center身份验证或身份验证。大多数 AWS 服务和资源都是通过 IAM 和 IAM 身份中心管理的。
2. 对于某些其他 AWS 服务，AWS 生成器 ID 是可选的。

以下主题包含其他详细信息以及每种凭证类型和身份验证方法的设置说明。

主题

AWS 中的 IAM 身份中心证书 AWS Toolkit for Visual Studio

AWS IAM Identity Center 是管理 AWS 账户身份验证的推荐最佳实践。

有关如何为软件开发套件 (SDKs) 和设置 IAM 身份中心的详细说明 AWS Toolkit for Visual Studio，请参阅AWS SDKs 和工具参考指南的 IAM 身份中心身份验证部分。

通过 IAM 身份中心进行身份验证 AWS Toolkit for Visual Studio

要 AWS Toolkit for Visual Studio 通过向您的`credentials`或`config`文件中添加 IAM 身份中心配置文件来通过 IAM 身份中心进行身份验证，请完成以下步骤。

1. 在首选的文本编辑器中，打开存储在`<hone-directory>\.aws\credentials`文件中的 AWS 凭据信息。

2. 在 `credentials file` 的 `[default]` 部分下，为 IAM Identity Center 命名配置文件添加模板。以下是示例模板：

> **Important**
>
> 在 `credential` 文件中创建条目时，请勿使用 profile 一词，因为这会与 `credential` 文件命名约定发生冲突。
> 仅当配置 `config` 文件中的命名配置文件时，才包含前缀词 `profile_`。

```
[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`**：指向贵组织 IAM Identity Center 用户门户的 URL。
- **`sso_region`**：包含您的 IAM 身份中心门户主机的 AWS 区域。这可能与稍后在默认`region`参数中指定的 AWS 区域不同。
- **`sso_account_id`**：包含您要向此 IAM 身份中心用户授予权限的 IAM 角色的 AWS 账户 ID。
- **`sso_role_name`**：使用此配置文件通过 IAM Identity Center 获取凭证时，定义用户权限的 IAM 角色的名称。
- **`region`**：此 IAM 身份中心用户登录的默认 AWS 区域。

> **Note**
>
> 您还可以通过运行`aws configure sso`命令将启用 IAM Identity Center AWS CLI 的配置文件添加到您的配置文件中。运行此命令后，您需要为 IAM 身份中心起始 URL (`sso_start_url`) 和托管 IAM 身份中心目录的 AWS 区域 (`region`) 提供值。
> 有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的 “将 AWS CLI 配置为使用 AWS 单点登录”。

使用 IAM Identity Center 登录

使用 IAM Identity Center 配置文件登录时，默认浏览器将启动到 `credential file` 中指定的 `sso_start_url`。您必须先验证自己的 IAM 身份中心登录信息，然后才能访问中的 AWS 资源 AWS Toolkit for Visual Studio。如果凭证过期，则必须重复连接过程以获取新的临时凭证。

AWS IAM 证书

AWS IAM 凭证通过本地存储的访问密钥对您的 AWS 账户进行身份验证。

以下各节介绍如何设置 IAM 证书，以便通过您的 AWS 账户进行身份验证 AWS Toolkit for Visual Studio。

> **Important**
>
> 在设置 IAM 凭证以使用您的 AWS 账户进行身份验证之前，请注意：
>
> - 如果您已经通过其他 AWS 服务（例如 AWS CLI）设置了 IAM 证书，则会 AWS Toolkit for Visual Studio 自动检测这些证书。
> - AWS 建议使用 AWS IAM Identity Center 身份验证。有关 AWS IAM 最佳实践的更多信息，请参阅 Identity and A ccess Managem AWS ent 用户指南的 “IAM 中的安全最佳实践” 部分。
> - 为了避免安全风险，在开发专用软件或处理真实数据时，请勿使用 IAM 用户进行身份验证，相反，请使用与身份提供商（例如）的联合 AWS IAM Identity Center。有关更多信息，请参阅《AWS IAM Identity Center User Guide》中的 What is IAM Identity Center?。

创建 IAM 用户

在将设置为使用 AWS 账户 AWS Toolkit for Visual Studio 进行身份验证之前，您需要先完成和工具参考指南中使用长期证书进行身份验证主题中的步骤 1：创建您的 IAM 用户AWS SDKs 和步骤 2：获取访问密钥。

> **Note**
>
> 步骤 3：更新共享凭证是可选步骤。
> 如果您完成了步骤 3，则会 AWS Toolkit for Visual Studio 自动从中检测您的凭证`credentials file`。

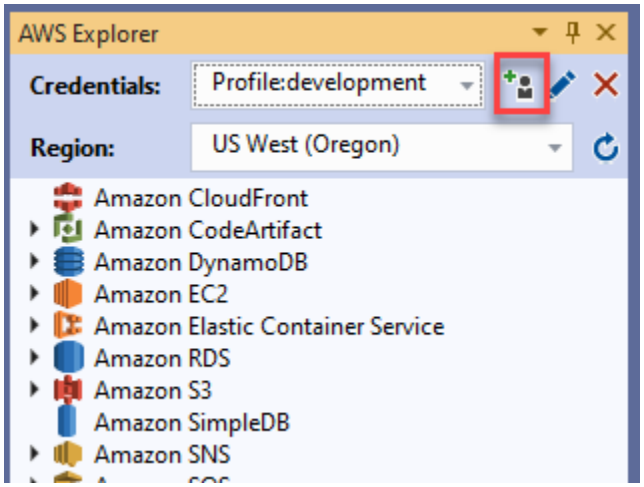
如果您尚未完成步骤 3，则将 AWS Toolkit for Visual Studio 引导您完成创建过程，`credentials file`如下面的 “从中创建证书文件 AWS Toolkit for Visual Studio” 部分所述。

创建凭证文件

要向 AWS Toolkit for Visual Studio添加用户或在其中创建 `credentials file`，请执行以下操作：

> **Note**
>
> 在 Toolkit 中添加新的用户配置文件时：
>
> - 如果 `credentials file` 已经存在，则新的用户信息将添加到现有文件中。
> - 如果 `credentials file` 不存在，则会创建一个新文件。

1. 从 AWS 资源管理器中选择 “新建账户资料” 图标以打开 “新建账户资料” 对话框。



2. 在新建账户配置文件对话框中填写必填字段，然后选择确定按钮创建 IAM 用户。

在 Toolkit 中编辑 IAM 用户凭证

要在 Toolkit 中编辑 IAM 用户凭证，请完成以下步骤：

1. 从 AWS 资源管理器的证书下拉列表中，选择要编辑的 IAM 用户证书。
2. 选择编辑配置文件图标以打开编辑配置文件对话框。

3. 在编辑配置文件对话框中完成更新，然后选择确定按钮保存更改。

要从 Toolkit 中删除 IAM 用户凭证，请完成以下步骤：

1. 从 AWS Explorer 的 “证书” 下拉列表中，选择要删除的 IAM 用户证书。
2. 选择删除配置文件图标以打开删除配置文件提示。
3. 确认您要删除该配置文件，以将其从 `Credentials file` 中移除。

> **Important**
>
> 无法在 AWS Toolkit for Visual Studio中通过编辑配置文件对话框编辑支持高级访问功能 [例如 IAM Identity Center 或多重身份验证（MFA）] 的配置文件。要更改这些类型的配置文件，必须使用文本编辑器编辑 `credentials file`。

使用文本编辑器编辑 IAM 用户凭证

除了使用管理 IAM 用户外 AWS Toolkit for Visual Studio，您还可以使用首选`credential files`的文本编辑器进行编辑。在 Windows 中，`credential file` 的默认位置是 `C:\Users\`*`USERNAME`*`\.aws\credentials`。

有关位置和结构的更多详细信息`credential files`，请参阅和工具参考指南的共享配置AWS SDKs和凭据文件部分。

通过 AWS Command Line Interface (AWS CLI) 创建 IAM 用户

AWS CLI 是另一个工具，您可以使用命令在中创建 IAM 用户`aws configure`。`credentials file`

有关从中创建 IAM 用户的 AWS CLI 详细信息，请参阅AWS CLI 用户指南中的配置 AWS CLI主题。

Toolkit for Visual Studio 支持以下配置属性：

```
aws_access_key_id
aws_secret_access_key
aws_session_token
credential_process
credential_source
```

```
external_id
mfa_serial
role_arn
role_session_name
source_profile
sso_account_id
sso_region
sso_role_name
sso_start_url
```

AWS 生成器 ID

AWS Builder ID 是一种额外的 AWS 身份验证方法，可能需要使用某些服务或功能，例如通过 Amazon 克隆第三方存储库 CodeCatalyst。

有关 AWS 生成器 ID 身份验证方法的详细信息，请参阅《登录用户指南》中的 “使用 AWS 生成器 ID AWS 登录” 主题。

有关 CodeCatalyst 从中克隆存储库的更多信息 AWS Toolkit for Visual Studio，请参阅本用户指南中的使用Amazon CodeCatalyst 主题。

Toolkit for Visual Studio 中的多重身份验证（MFA）

多重身份验证 (MFA) 为您的账户提供了额外的安全保障。 AWS MFA 要求用户在访问网站或服务时提供登录凭证和来自支持的 AWS MFA 机制的唯一身份验证。 AWS

AWS 支持一系列虚拟设备和硬件设备进行 MFA 身份验证。以下是通过智能手机应用程序启用的虚拟 MFA 设备示例。要了解有关 MFA 设备选项的更多信息，请参阅《IAM 用户指南》中的在 AWS中使用多重身份验证（MFA）。

步骤 1：创建 IAM 角色，以向 IAM 用户委派访问权限

以下过程介绍如何设置向 IAM 用户分配权限的角色委派。有关角色委派的详细信息，请参阅《AWS Identity and Access Management 用户指南》中的创建向 IAM 用户委派权限的角色主题。

1. 前往 https://console.aws.amazon.com/iam 上的 IAM 控制台。
2. 在导航栏中选择角色，然后选择创建角色。
3. 在创建角色页面中，选择另一个 AWS 账户。
4. 输入所需的账户 ID 并选中需要 MFA 复选框。

> **Note**
>
> 要查找您的 12 位账号（ID），请在控制台顶部的导航栏上，选择支持，然后选择支持中心。

5. 选择下一步: 权限。
6. 将现有策略附加到角色或为其创建新策略。您在此页面上选择的策略决定了 IAM 用户可以通过 Toolkit 访问哪些 AWS 服务。
7. 附加策略后，选择下一步：标签，设置向角色添加 IAM 标签的选项。然后，选择下一步：审核以继续。
8. 在审核页面中，输入所需的角色名称（例如 toolkit-role）。您也可以添加可选的角色描述。
9. 选择创建角色。
10. 当显示确认消息（例如“角色 toolkit-role 已创建”）时，请在消息中选择该角色的名称。
11. 在摘要页面中，选择复制图标以复制角色 ARN 并将其粘贴到一个文件中。（在配置代入角色的 IAM 用户时，您需要此 ARN。）

步骤 2：创建代入角色权限的 IAM 用户

此步骤将创建一个没有权限的 IAM 用户，以便可以添加内联策略。

1. 前往 https://console.aws.amazon.com/iam 上的 IAM 控制台。
2. 在导航栏中，选择用户，然后选择添加用户。
3. 在添加用户页面中，输入所需的用户名（例如 toolkit-user），然后选中编程访问复选框。
4. 选择下一步：权限、下一步：标签和下一步：审核，进行翻页。您不必在此阶段添加权限，因为用户将代入通过角色委派的权限。
5. 在审核页面中，您会被告知此用户没有权限。选择创建用户。
6. 在成功页面中，选择下载.csv 以下载包含访问密钥 ID 和秘密访问密钥的文件。（在凭证文件中定义用户的配置文件时，您将需要这两个信息。）
7. 选择关闭。

步骤 3：添加允许 IAM 用户代入角色的策略

以下过程将创建一个内联策略，允许用户代入角色（以及该角色的权限）。

1. 在 IAM 控制台的用户页面中，选择您刚刚创建的 IAM 用户（例如 toolkit-user）。
2. 在摘要页面的权限选项卡上，选择添加内联策略。
3. 在创建策略页面中，选择选择服务，在查找服务中输入 STS，然后从结果中选择 STS。
4. 在 “操作” 中，开始输入术语AssumeRole。当AssumeRole复选框出现时，将其选中。
5. 在资源部分，确保选中特定，然后单击添加 ARN 以限制访问。
6. 对于添加 ARN 对话框中的为角色指定 ARN，添加您在步骤 1 中创建的角色的 ARN。

   添加该角色的 ARN 后，与该角色关联的可信账户和角色名称将显示在账户和带路径的角色名称中。
7. 选择添加。
8. 返回创建策略页面，选择指定请求条件（可选），选中需要 MFA 复选框，然后选择关闭进行确认。
9. 选择查看策略
10. 在查看策略页面中，为策略输入名称，然后选择创建策略。

    权限选项卡中将显示直接附加到 IAM 用户的新内联策略。

步骤 4：为 IAM 用户管理虚拟 MFA 设备

1. 将虚拟 MFA 应用程序下载并安装到智能手机。

   有关支持的应用程序列表，请参阅多重身份验证资源页面。
2. 在 IAM 控制台中，从导航栏中选择用户，然后选择代入角色的用户（在本例中为 toolkit-user）。
3. 在摘要页面中，选择安全凭证选项卡，然后为已分配的 MFA 设备选择管理。
4. 在管理 MFA 设备窗格中，选择虚拟 MFA 设备，然后选择继续。
5. 在设置虚拟 MFA 设备窗格中，选择显示 QR 码，然后使用安装在智能手机上的虚拟 MFA 应用程序扫描二维码。
6. 扫描二维码后，虚拟 MFA 应用程序会生成一次性 MFA 代码。在 MFA 代码 1 和 MFA 代码 2 中连续输入两个 MFA 代码。
7. 选择分配 MFA。
8. 返回用户的安全凭证选项卡，复制新分配的 MFA 设备的 ARN。

   ARN 包含您的 12 位账户 ID，其格式类似于以下内容：`arn:aws:iam::123456789012:mfa/toolkit-user`。在下个步骤中定义 MFA 配置文件时，您需要用到此 ARN。

步骤 5：创建配置文件以允许 MFA

以下过程创建了在通过 Visual Studio 的 Toolkit for Visual Studi AWS o 访问服务时允许 MFA 的配置文件。

您创建的配置文件包括您在前面的步骤中复制和存储的三条信息：

- IAM 用户的访问密钥（访问密钥 ID 和秘密访问密钥）
- 向 IAM 用户委派权限的角色的 ARN
- 分配给 IAM 用户的虚拟 MFA 设备的 ARN

在包含您的 AWS 凭据的 AWS 共享凭证文件或 SDK 商店中，添加以下条目：

```
[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
```

提供的示例中定义了两个配置文件：

- `[toolkit-user]` 配置文件包含您在步骤 2 中创建 IAM 用户时生成和保存的访问密钥和秘密访问密钥。
- `[mfa]` 配置文件定义了如何支持多因素身份验证。该配置文件有三个条目：

  ◦ `source_profile`：指定配置文件，将使用其凭证代入此配置文件中的 `role_arn` 设置所指定的角色。在本例中，该条目为 `toolkit-user` 配置文件。

  ◦ `role_arn`：指定 IAM 角色的 Amazon 资源名称（ARN），您将通过此角色执行使用此配置文件请求的操作。在本例中，该条目您在步骤 1 中创建的角色的 ARN。

  ◦`mfa_serial`：指定用户在代入角色时必须使用的 MFA 设备的标识或序列号。在本例中，该条目是您在步骤 3 中设置的虚拟设备的 ARN。

设置外部凭证

如果 AWS不直接支持您生成或查找凭证的方法，则可以在共享的凭证文件中添加一个包含 `credential_process` 设置的配置文件。此设置指定运行的外部命令，以生成或检索要使用的身份验证凭证。例如，您可以在 `config` 文件中包含类似于以下内容的条目：

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

有关使用外部凭证和相关安全风险的更多信息，请参阅《AWS Command Line Interface 用户指南》中的使用外部进程获取凭证。

更新防火墙和网关以允许访问

如果您使用 Web 内容筛选解决方案筛选对特定 AWS 域或 URL 终端节点的访问，则必须允许列出以下终端节点才能访问通过和 Amazon Q 提供的所有服务和功能。有关如何使用 AWS Toolkit for Visual Studio Amazon Q 对 Toolkit 的防火墙和代理设置进行故障排除的详细步骤，请参阅本用户指南故障排除主题中的防火墙和代理设置部分。 AWS 有关为 Amazon Q 配置公司代理的详细信息，请参阅《Amazon Q 开发者版用户指南》中的在 Amazon Q 中配置公司代理主题。

AWS Toolkit for Visual Studio 端点

以下是需要允许列出的 AWS Toolkit for Visual Studio 特定端点和参考文献的列表。

端点

```
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 插件端点

以下是需要纳入允许列表的特定于 Amazon Q 插件的端点和引用的列表。

```
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 开发者版端点

以下是需要纳入允许列表的特定于 Amazon Q 开发者版的端点和引用的列表。

```
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 代码转换端点

以下是需要纳入允许列表的特定于 Amazon Q 代码转换的端点和引用的列表。

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

身份验证端点

以下是需要纳入允许列表的身份验证端点和引用的列表。

```
[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
```

身份端点

以下列表包含特定于身份的端点，例如 AWS IAM Identity Center 和 AWS 生成器 ID。

AWS IAM Identity Center

有关 IAM Identity Center 所需端点的详细信息，请参阅《AWS IAM Identity Center用户指南》中的启用 IAM Identity Center 主题。

企业 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
```

AWS 生成器 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)
```

遥测

以下是需要纳入允许列表的特定于遥测的端点。

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

引用

以下是端点引用的列表。

```
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
```

与 AWS 服务配合使用

以下主题介绍如何开始从 AWS Toolkit for Visual Studio with Amazon Q 使用 AWS 服务。

主题

亚马逊 CodeCatalyst 为带亚马逊 Q 的 Visual Studio 的 Visual Studio AWS Toolkit

什么是亚马逊 CodeCatalyst？

Amazon CodeCatalyst 是一个面向软件开发团队的基于云的协作空间。使用带有 Amazon Q 的 Visual Studio AWS 工具包，您可以直接从 Visual Studio 的 Toolkit for Amazon Q 中查看和管理 CodeCatalyst 资源。有关更多信息 CodeCatalyst，请参阅亚马逊 CodeCatalyst用户指南。 AWS

以下主题介绍如何将 Visual Studio AWS 工具包与 Amazon Q 连接起来， CodeCatalyst 以及如何 CodeCatalyst 通过 Visual Studio AWS 工具包与 Amazon Q 配合使用。

主题

亚马逊入门 CodeCatalyst 和 Visual Studio 的 AWS Toolkit for Amazon Q

要开始使用带有 Amazon CodeCatalyst Q 的 Visual Studio AWS 工具包与亚马逊合作，请完成以下操作。

主题

使用 Amazon Q 安装适用于 Visual Studio 的 AWS Toolkit

在将 Visual Studio AWS 工具包与 Amazon Q 集成到您的 CodeCatalyst账户之前，请确保使用的是最新版本的 Visual Studio AWS 工具包和 Amazon Q。有关如何使用亚马逊 Q 安装和设置最新版本的 Visual Studio AWS 工具包的详细信息，请参阅本用户指南的 “使用 Ama zon Q 为 Visual Studio 设置 AWS 工具包” 部分。

创建 CodeCatalyst 账号和 AWS 建筑商 ID

除了使用 Amazon Q 安装最新版本的 Visual Studio Visual Studio AWS 工具包外，您还必须拥有有效的 AWS 生成器 ID 和 CodeCatalyst 账户才能通过 Amazon Q 与 Visual Studio 的 AWS Toolkit for Visual Studio 连接。如果您没有有效的 AWS 生成器 ID 或 CodeCatalyst 账户，请参阅CodeCatalyst用户指南中的设置方式 CodeCatalyst部分。

> **Note**
>
> AWS 生成器 ID 与您的 AWS 凭证不同。有关如何使用 AWS 生成器 ID 注册和进行身份验证的说明，请参阅本用户指南中的身份验证和访问权限： AWS 生成器 ID 主题。

有关 AWS Builder 的详细信息 IDs，请参阅《AWS 通用参考用户指南》中的 “AWS 生成器 ID” 主题。

将 Visual Studio 的 AWS Toolkit 与 Amazon Q 连接起来 CodeCatalyst

要将 Visual Studio 的 AWS Toolkit for Visual Studio 与 Amazon Q 关联到您的 CodeCatalyst 账户，请完成以下步骤。

1. 在 Visual Studio 的 Git 菜单项中，选择克隆存储库...。
2. 从 “浏览存储库” 部分，选择 Amazon CodeCatalyst 作为提供商。
3. 在 “连接” 部分中，选择 Connect w AWS ith Builder ID，在首选的 Web 浏览器中打开 CodeCatalyst 控制台。
4. 在浏览器中，在提供的字段中输入您的 AWS 生成器 ID，然后按照说明继续操作。
5. 出现提示时，选择 “允许” 以确认 Visual Studio 的 AWS Toolkit for Visual Studio 和 Amazon Q 与您的 CodeCatalyst 账户之间的连接。连接过程完成后， CodeCatalyst 会显示一条确认消息，表明可以安全地关闭浏览器。

在 Amazon Q 上使用 Visual Studio AWS 工具包中的亚马逊 CodeCatalyst 资源

以下各节概述了带有 Amazon Q 的 Visual Studio 的 AWS Toolkit for Visual Studio 的可用亚马逊 CodeCatalyst 资源管理功能。

主题

- 克隆存储库

克隆存储库

CodeCatalyst 是一项基于云的服务，需要您连接到云端才能处理 CodeCatalyst 项目。要在本地处理项目，您可以将 CodeCatalyst 存储库克隆到本地计算机，并在下次连接到云端时与您的 CodeCatalyst 项目同步。

要将存储库克隆到本地计算机，请完成以下步骤。

1. 在 Visual Studio 的 Git 菜单项中，选择克隆存储库...。
2. 从 “浏览存储库” 部分，选择 Amazon CodeCatalyst 作为提供商。

> **Note**
>
> 如果 “连接” 部分显示一条Not Connected消息，请先完成本用户指南的 “身份验证和访问权限： AWS 生成器 ID” 部分中的步骤，然后再继续。

3. 选择要从中克隆存储库的空间和项目。
4. 在存储库部分，选择要克隆的存储库。
5. 在路径部分，选择要将存储库克隆到的文件夹。

> **Note**
>
> 此文件夹最初必须为空才能成功克隆。

6. 选择克隆，以开始克隆存储库。
7. 克隆存储库后，Visual Studio 将加载您克隆的解决方案

> **Note**
>
> 如果 Visual Studio 未在克隆的存储库中打开解决方案，则可以从源控制菜单的 Git 全局设置中，通过打开 Git 存储库时自动加载解决方案设置调整 Visual Studio 选项。

问题排查

以下是使用亚马逊 Q 的 Visual Studio For Visual Studio AWS 工具包解决与亚马逊 CodeCatalyst 合作时已知问题的疑难解答主题。

主题

凭据

如果您在尝试从中克隆基于 git 的存储库时遇到要求提供凭据的对话框 CodeCatalyst，则您的AWS CodeCommit 凭据助手可能会被全局配置，从而造成干扰。 CodeCatalyst有关 AWS CodeCommit 凭证助手的更多信息，请参阅《AWS CodeCommit用户指南》的 “使用 AWS CLI 凭据帮助器在 Windows 上设置与 Windows AWS CodeCommit 存储库的 HTTPS 连接的步骤” 部分。

要将AWS CodeCommit 凭据助手限制为只能处理 CodeCommit URLs，请完成以下步骤。

1. 打开以下位置的全局 Git 配置文件：`%userprofile%\.gitconfig`
2. 在文件中找到以下部分：

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

3. 将该部分更改为以下内容：

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

4. 保存更改，然后完成克隆存储库的步骤。

Visual Studio 的 Amazon CloudWatch Logs 集成

通过 AWS Toolkit for Visual Studio with Amazon Q 中的 Amazon CloudWatch Logs 集成，您无需离开 IDE 即可监控、存储和访问 CloudWatch Logs 资源。要了解有关设置 CloudWatch 服务以及如何使用 CloudWatch Logs 功能的更多信息，请从以下主题中进行选择。

主题

- 为 Visual Studio 设置 CloudWatch Logs 集成
- 使用 Visual Studio 中的 CloudWatch 日志

为 Visual Studio 设置 CloudWatch Logs 集成

在使用 Amazon CloudWatch Logs 与 AWS Toolkit with Amazon Q 集成之前，您需要一个 AWS 账户。您可以从 AWS 登录站点创建一个新 AWS 账户。可以使用有效 AWS 凭证访问 AWS Toolkit with Amazon Q 中提供的大多数 CloudWatch Logs 功能。如果某项功能需要额外配置，则这些要求包含在使用 CloudWatch Logs 指南的相关部分中。

有关设置 CloudWatch Logs 的更多信息和选项，请参阅 Amazon CloudWatch Logs 指南的开始设置部分。

使用 Visual Studio 中的 CloudWatch 日志

Amazon CloudWatch Logs 集成允许您使用 Amazon Q 监控、存储和访问 CloudWatch 来自 Visual Studio 的 AWS Toolkit for Visual Studio 中的 CloudWatch 日志。无需离开 IDE 即可访问日志功能，从而简化 CloudWatch 日志开发过程并减少对工作流程的干扰，从而提高效率。以下主题介绍如何使用 Lo CloudWatch gs 集成的基本特性和功能。

主题

CloudWatch 日志组

`log group` 是一组具有相同保留期、监控和访问控制设置的 `log streams`。对可属于一个日志组的日志流数没有限制。

查看日志组

该`View Log Groups`功能在日志组资源管理器中显示 CloudWatch日志组列表。

要访问查看日志组功能并打开 CloudWatch 日志组资源管理器，请完成以下步骤。

1. 在 AWS 资源管理器中，展开 Amazon CloudWatch。
2. 双击 “日志组” 或打开快捷菜单（右键单击），然后选择 “查看”，打开 “CloudWatch 日志组浏览器”。

> **Note**
>
> CloudWatch 日志组资源管理器将在与解决方案资源管理器相同的窗口位置打开。

筛选日志组

您的个人账户可以包含成千上万个不同的日志组。要简化对特定组的搜索，请使用下述 `filtering` 功能。

1. 在CloudWatch 日志组资源管理器中，将光标置于窗口顶部的搜索栏中。
2. 开始键入与要查找的日志组相关的前缀。
3. CloudWatch 日志组资源管理器会自动更新，以显示与您在上一步中指定的搜索词匹配的结果。

删除日志组

要删除特定日志组，请参阅以下步骤。

1. 在CloudWatch 日志组资源管理器中，右键单击要删除的日志组。
2. 在提示时，确认您希望删除当前选中的日志组。
3. 选择 “是” 按钮会删除选定的日志组，然后刷新 “CloudWatch 日志组浏览器”。

刷新日志组

要刷新 “日志组资源管理器” 中显示的当前CloudWatch 日志组列表，请选择工具栏上的 “刷新” 图标按钮。

复制日志组 ARN

要复制特定日志组的 ARN，请完成以下步骤。

1. 在CloudWatch 日志组资源管理器中，右键单击要从中复制 ARN 的日志组。
2. 从菜单中选择复制 ARN 选项。
3. 现在，ARN 已复制到本地剪贴板并可以粘贴了。

CloudWatch 日志流

日志流是共享同一来源的一系列日志事件。

> **Note**
>
> 查看日志流时，请注意以下属性：
>
> - 默认情况下，日志流按事件时间戳由近及远排序。

- 通过切换列标题中的尖角符号，可以按升序或降序对与日志流关联的列进行排序。
- 经过筛选的条目只能按日志流名称排序。

查看日志流

1. 在CloudWatch 日志组资源管理器中，双击日志组，或者右键单击日志组，然后从快捷菜单中选择查看日志流。
2. 将在文档窗口中打开一个新选项卡，其中包含与日志组关联的日志流列表。

筛选日志流

1. 在文档窗口的日志流选项卡中，将光标置于搜索栏中。
2. 开始键入与要查找的日志流相关的前缀。
3. 随着您键入，当前显示会自动更新，以根据您的输入筛选日志流。

刷新日志流

要刷新文档窗口中显示的当前日志流列表，请选择工具栏中搜索栏旁的刷新图标按钮。

复制日志流 ARN

要复制特定日志流的 ARN，请完成以下步骤。

1. 在文档窗口的日志流选项卡中，右键单击要从中复制 ARN 的日志流。
2. 从菜单中选择复制 ARN 选项。
3. 现在，ARN 已复制到本地剪贴板并可以粘贴了。

下载日志流

导出日志流功能可将选定的日志流下载并存储在本地，以供自定义工具和软件访问并进行其他处理。

1. 在文档窗口的日志流选项卡中，右键单击要下载的日志流。
2. 选择导出日志流，以打开导出到文本文件对话框。
3. 选择用于存储文件的本地位置，并在提供的文本字段中指定名称。
4. 选择确定以确认下载。下载状态显示在 Visual Studio 任务状态中心中

CloudWatch 记录事件

日志事件是所监视的应用程序或资源所记录的活动记录 CloudWatch。

日志事件操作

日志事件以表格形式显示。默认情况下，事件按从旧到新的顺序排序。

以下操作与 Visual Studio 中的日志事件相关联：

- 自动换行模式：您可以通过单击事件来切换自动换行。
- 文本换行按钮：此按钮位于 `document window` **`toolbar`** 中，可为所有条目开启和关闭文本换行。
- 将消息复制到剪贴板：选择要复制的消息，然后右键单击所选内容并选择复制（键盘快捷键 `Ctrl + C`）。

查看日志事件

1. 在文档窗口中，选择包含日志流列表的选项卡。
2. 双击日志流，或右键单击日志流并从菜单中选择查看日志流。
3. 将在文档窗口中打开一个新的日志事件选项卡，其中包含与所选日志流关联的日志事件表。

筛选日志事件

您可以通过三种方式筛选日志事件：按内容、按时间范围或同时按两者。要同时按内容和时间范围筛选日志事件，请先按内容或时间范围筛选消息，然后使用另一种方法对结果进行筛选。

要按内容筛选日志事件，请执行以下操作：

1. 在文档窗口的日志事件选项卡中，将光标置于窗口顶部的搜索栏中。
2. 开始键入与要搜索的日志事件相关的字词或短语。
3. 随着您键入，当前显示会自动开始筛选日志事件。

> **Note**
>
> 筛选条件模式区分大小写。将确切的字词和短语以及非字母数字字符用双引号括起来（`*""*`），可以改善搜索结果。有关筛选模式的更多详细信息，请参阅 Amazon CloudWatch 指南中的筛选条件和模式语法主题。

要查看在特定时间范围内生成的日志事件，请执行以下操作：

1. 在文档窗口的日志事件选项卡中，选择工具栏上的日历图标按钮。
2. 使用提供的字段，指定要搜索的时间范围。
3. 随着您指定日期和时间限制，筛选结果会自动更新。

> **Note**
>
> “清除筛选器” 选项可清除您当前选择的所有 date-and-time筛选条件。

刷新日志事件

要刷新日志事件选项卡中显示的当前日志事件列表，请选择工具栏上的刷新图标按钮。

对 CloudWatch 日志的额外访问权限

您可以直接从 Visual Studio 中的 AWS 工具包访问与其他 AWS 服务和资源关联的 CloudWatch 日志。

Lambda

要查看与 Lambda 函数关联的日志流，请执行以下操作：

> **Note**
>
> 您的 Lambda 执行角色必须具有相应的权限才能将日志发送到日志。 CloudWatch有关 CloudWatch 日志所需的 Lambda 权限的更多信息，请参阅 https://docs.aws.amazon.com/lambda/latest/dg/monitoring-cloudwatchlogs.html#monitoring-cloudwatchlogs-prereqs

1. 在 AWS 工具包资源管理器中，展开 Lambda。
2. 右键单击要查看的函数，然后选择查看日志，以在文档窗口中打开关联的日志流。

要使用 Lambda 集成 `function view` 查看日志流，请执行以下操作：

1. 在 AWS 工具包资源管理器中，展开 Lambda。
2. 右键单击要查看的函数，然后选择查看函数，以在文档窗口中打开函数视图。
3. 在 `function view` 中，切换到日志选项卡，将显示与所选 Lambda 函数关联的日志流。

ECS

要查看与 ECS 任务容器关联的日志资源，请完成以下步骤。

> **Note**
>
> 为了让 Amazon ECS 服务向其发送日志 CloudWatch，给定 Amazon ECS 任务的每个容器都必须满足所需的配置。有关所需设置和配置的更多信息，请参阅使用 AWS 日志日志驱动程序指南。

1. 在 AWS 工具包资源管理器中，展开 Amazon ECS。
2. 选择要查看的 Amazon ECS 集群，以在文档窗口中打开新的 ECS 集群选项卡。
3. 在 ECS 集群选项卡左侧的导航菜单中，选择任务以列出与该集群关联的所有任务。
4. 在任务显示屏中，选择任务并选择位于左下角的查看日志链接。

   > **Note**
   >
   > 此显示列出了集群中包含的所有任务，但只有符合所需日志配置的任务显示 `View Logs` 链接。
   >
   > - 如果任务仅与单个容器关联，则查看日志链接会打开该容器的日志流。
   > - 如果一个任务与多个容器关联，则查看日志链接会打开查看 ECS 任务的 CloudWatch 日志对话框，使用容器:下拉菜单选择要查看其日志的容器，然后选择确定。

5. 将在文档窗口中打开一个新选项卡，显示与您的容器选择相关的日志流。

管理 Amazon EC2 实例

AWS Explorer 提供亚马逊系统映像 (AMI) 和亚马逊弹性计算云 (Amazon EC2) 实例的详细视图。从这些视图中，您可从 AMI 启动 Amazon EC2 实例，连接到此实例，停止或终止此实例，所有这些操作都是在 Visual Studio 开发环境中执行的。您可以使用实例视图 AMIs 从您的实例进行创建。有关更多信息，请参阅从 Amazon EC2 实例创建 AMI。

Amazon 系统映像和 Amazon EC2 实例视图

AWS 在 Explorer 中，您可以显示亚马逊系统映像 (AMI) 和 Amazon EC2 实例的视图。在 AWS 资源管理器中，展开 Amazon EC2 节点。

要显示 AMIs 视图，请在第一个子节点上打开上下文（右键单击）菜单 AMIs，然后选择 “查看”。

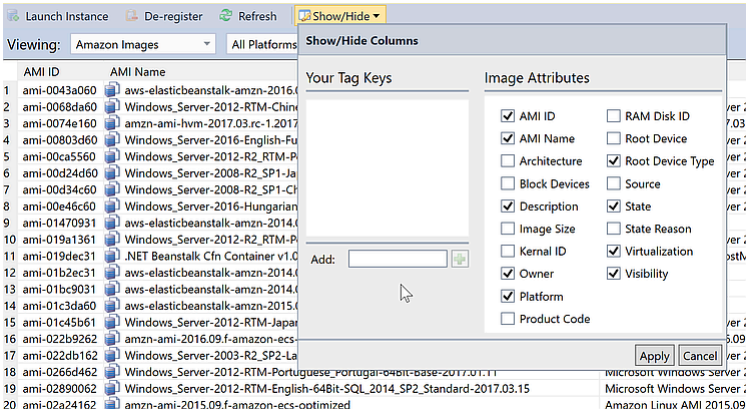
要显示 Amazon EC2 实例视图，请在 Instances (实例) 节点上，打开上下文（右键单击）菜单，然后选择 View (查看)。

您也可通过双击相应的节点来显示任一视图。

- 视图的范围限定在 AWS Explorer 中指定的区域（例如，美国西部（加利福尼亚北部）区域）。
- 您可通过单击并拖动来重新整理列。要对列中的值进行排序，请单击列标题。
- 可使用 Viewing (查看) 中的下拉列表和筛选器框来配置视图。初始视图显示 AMIs AWS 资源管理器中指定的帐户所拥有的任何平台类型（Windows 或 Linux）。

显示/隐藏列

您还可选择位于视图顶部的 Show/Hide (显示/隐藏) 下拉列表来配置显示的列。如果您关闭并重新打开视图，您选择的列将保留。



AMI 和实例视图的 Show/Hide Columns (显示/隐藏列) UI

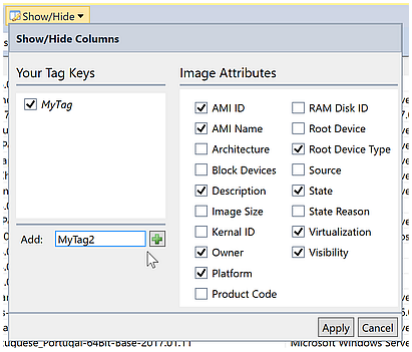
标记 AMIs、实例和卷

您还可使用显示/隐藏下拉列表为您拥有的 AMI、Amazon EC2 实例或卷添加标签。标签是名称/值对，使您能够将元数据附加到您的 AMIs、实例和卷。标签名称的作用域既限于您的账户，也适用于您的 AMIs和实例。例如，如果您 AMIs和您的实例使用相同的标签名称，则不会发生冲突。标签名称不区分大小写。

有关标签的更多信息，请参阅《Amazon EC2 用户指南（适用于 Linux 实例）》中的使用标签。

添加标签

1. 在 Add (添加) 框中，键入标签的名称。选择带加号 (+) 的绿色按钮，然后选择 Apply (应用)。



向 AMI 或 Amazon EC2 实例添加标签

新标签以斜体形式显示，这表示此标签未与任何值关联。

在列表视图中，标签名称显示为新列。如果标签已与至少一个值关联，则标签将在 AWS 管理控制台中可见。

2. 要为标签添加值，请双击标签对应的列中的单元格，然后键入值。要删除标签值，请双击单元格并删除文本。

如果在 Show/Hide (显示/隐藏) 下拉列表中清除标签，则对应的列将从视图中消失。该标签以及与实例或卷关联 AMIs的所有标签值都将保留。

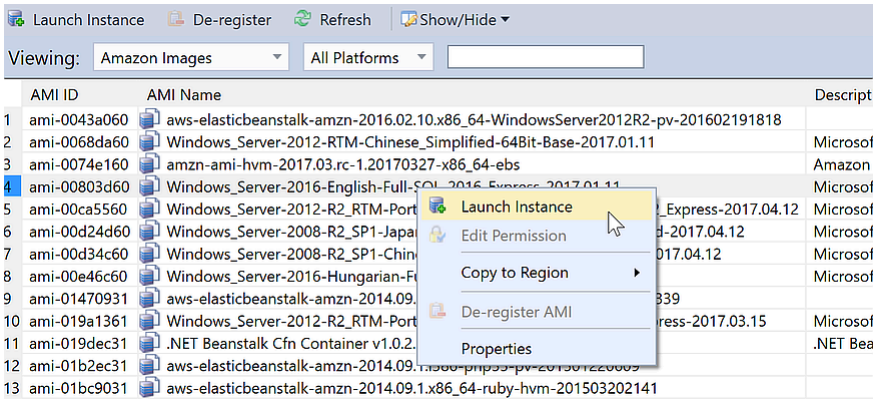
> **Note**
>
> 如果您清除 “显示/隐藏” 下拉列表中没有关联值的标签， AWS Toolkit 将完全删除该标签。它将不再显示在列表视图或 Show/Hide (显示/隐藏) 下拉列表中。要再次使用标签，请使用 Show/Hide (显示/隐藏) 对话框重新创建它。

启动 Amazon EC2 实例

AWS Explorer 提供了启动 Amazon EC2 实例所需的所有功能。在本节中，我们将选择并配置 Amazon 系统映像 (AMI)，然后将它作为 Amazon EC2 实例启动。

启动 Windows Server Amazon EC2 实例

1. 在 AMIs 视图顶部的左侧下拉列表中，选择 Amazon Images。在右侧的下拉列表中，选择 Windows。在筛选器框中，为 Elastic Block 存储键入 ebs。刷新视图可能需要一点时间。
2. 选择列表中的 AMI，打开上下文（右键单击）菜单，然后选择 Launch Instance (启动实例)。



AMI 列表

3. 在 Launch New Amazon EC2 Instance (启动新 Amazon EC2 实例) 对话框中，为您的应用程序配置 AMI。

   实例类型

   选择要启动的 EC2 实例的类型。您可以在 EC2 Pricing (EC2 定价) 页面上找到实例类型和定价信息的列表。

   名称

   为您的实例键入名称。此名称不能超过 256 个字符。

   密钥对

   密钥对用于获取您用于通过远程桌面协议 (RDP) 登录到 EC2 实例的 Windows 密码。选择您有权访问私有密钥的密钥对，或选择用于创建密钥对的选项。如果您在 Toolkit 中创建密钥对，则 Toolkit 可为您存储私有密钥。

   存储在工具包中的密钥对将被加密。您可以在 `%LOCALAPPDATA%\AWSToolkit\keypairs`（通常为 `C:\Users\<user>\AppData\Local\AWSToolkit\keypairs`）位置找到它们。您可以将加密密钥对导出到 `.pem` 文件。

   a. 在 Visual Studio 中，选择视图，然后单击 AWS 各区服务浏览器。

   b. 单击 Amazon EC2，选择 Key Pairs (密钥对)。

   c. 将列出密钥对，工具包中的密钥 created/managed 对将标记为 “存储在” AWSToolkit。

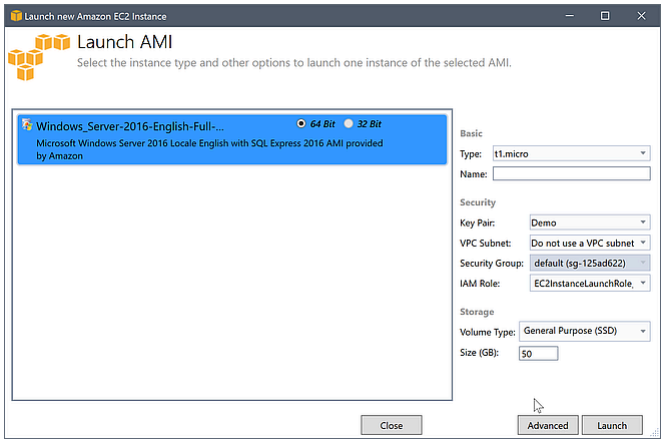
   d. 右键单击您创建的密钥对，然后选择 Export Private Key (导出私有密钥)。私有密钥将取消加密并保存在您指定的位置。

安全组

安全组控制 EC2 实例将接受的网络流量的类型。选择将允许端口 3389（RDP 使用的端口）上的传入流量的安全组，以便能连接到 EC2 实例。有关如何使用工具包创建安全组的信息，请参阅通过 AWS Explorer 管理安全组。

实例配置文件

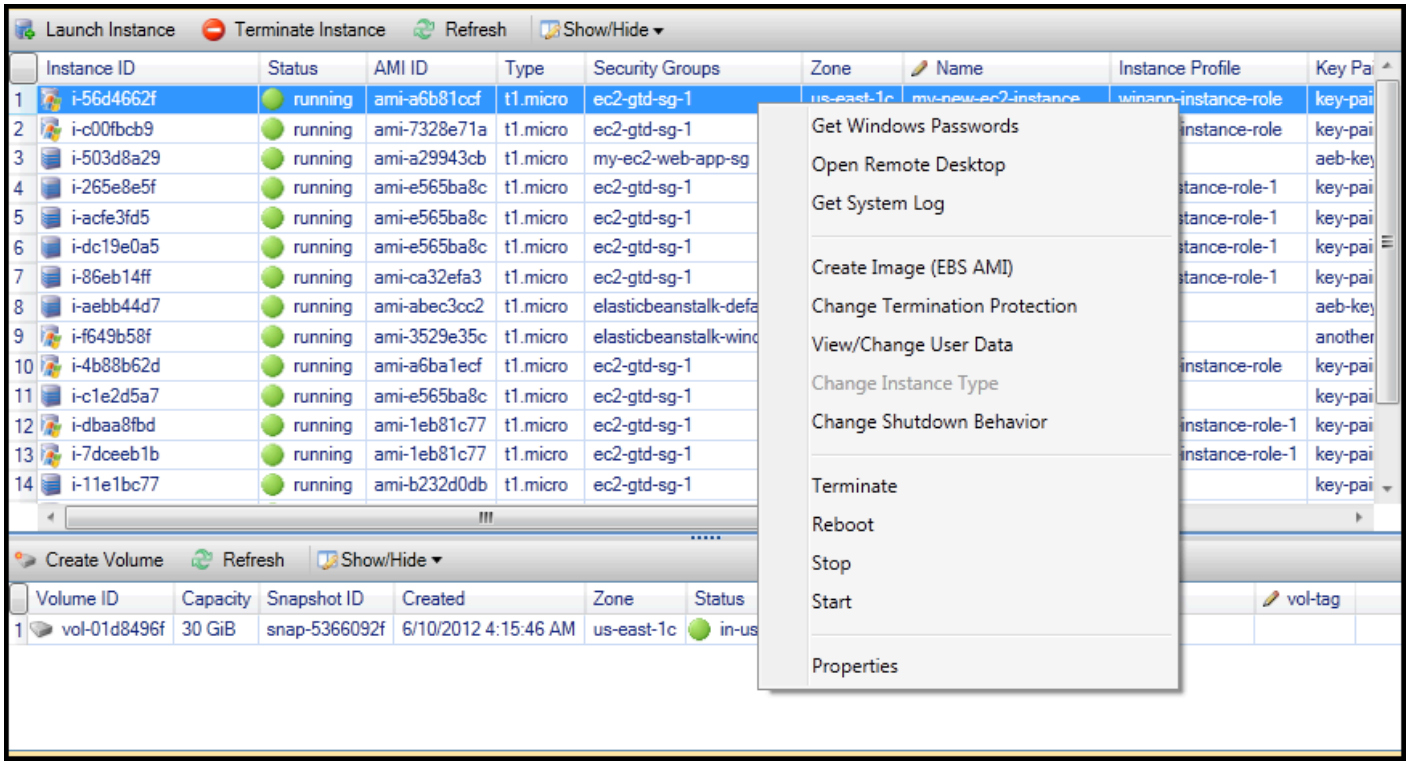
实例配置文件是 IAM 角色的逻辑容器。当您选择实例配置文件时，会将对应的 IAM 角色与 EC2 实例关联。IAM 角色是使用指定对 Amazon Web Services 和账户资源的访问权限的策略配置的。当 EC2 实例与 IAM 角色关联时，在实例上运行的应用程序软件将使用 IAM 角色指定的权限运行。这使应用程序软件无需自行指定任何 AWS 凭据即可运行，从而使软件更加安全。有关 IAM 角色的更多信息，请转到 IAM 用户指南。



EC2 Launch AMI (启动 AMI) 对话框

4. 选择启动。

在 AWS Explorer 中，在 Amazon EC2 的 “实例” 子节点上，打开上下文（右键单击）菜单，然后选择 “查看”。该 AWS 工具包显示与活跃账户关联的 Amazon EC2 实例列表。您可能需要选择 Refresh (刷新) 才能查看您的新实例。当实例第一次显示时，它可能处于挂起状态，但在几分钟后，它将过渡到运行状态。



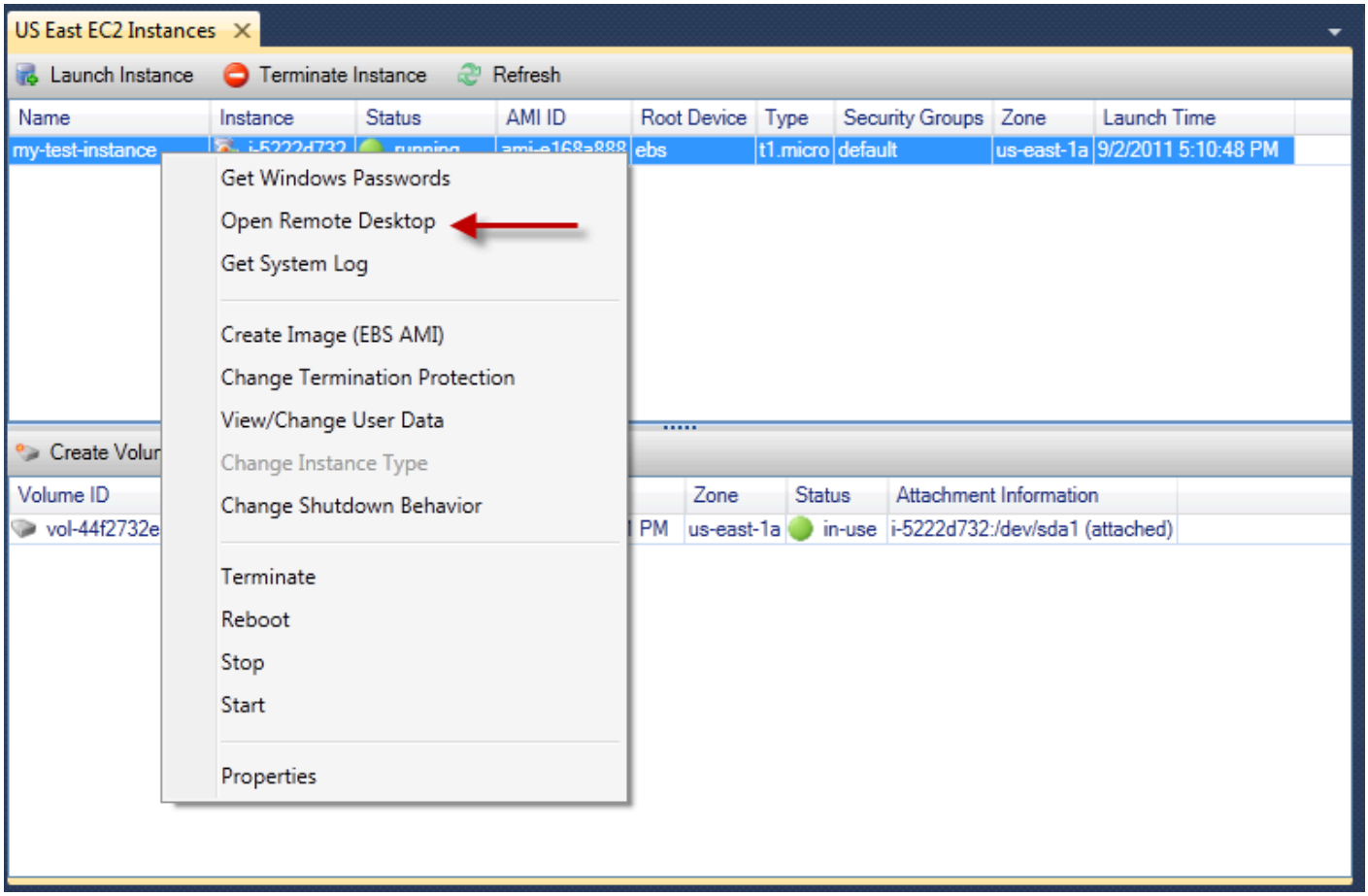
连接到 Amazon EC2 实例

您可使用 Windows 远程桌面连接到 Windows Server 实例。对于身份验证， AWS Toolkit 允许您检索实例的管理员密码，也可以直接使用与实例关联的存储密钥 pair。在以下过程中，我们将使用存储的密钥对。

使用 Windows 远程桌面连接到 Windows Server 实例

1. 在 EC2 实例列表中，右键单击要连接到的 Windows Server 实例。从上下文菜单中，选择 Open Remote Desktop (打开远程桌面)。

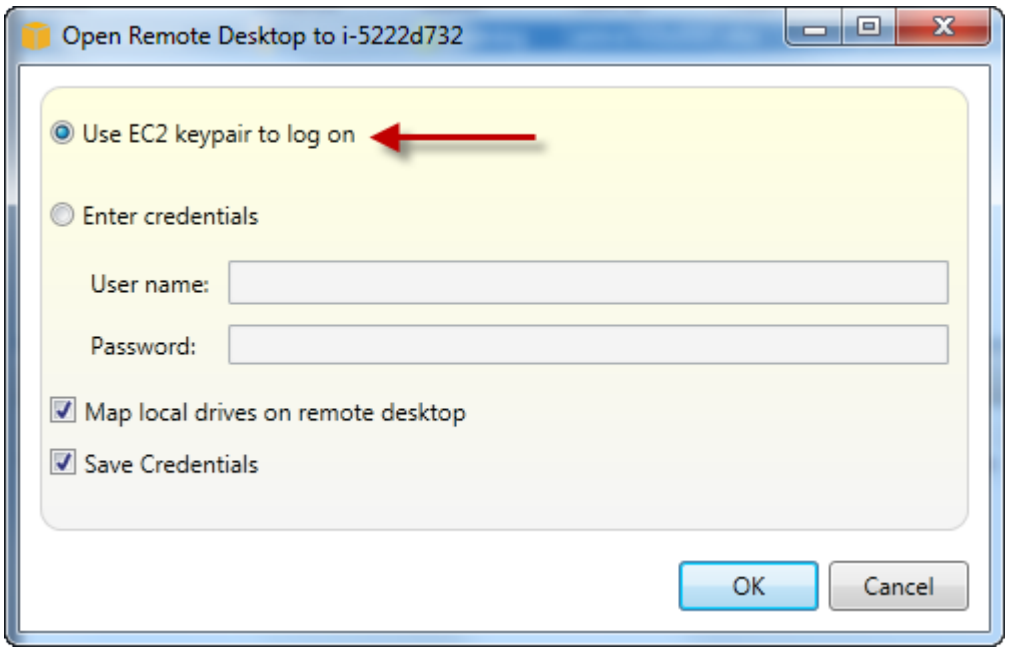
   如果您要使用管理员密码进行身份验证，请选择 Get Windows Passwords (获取 Windows 密码)。



EC2 实例上下文菜单

2. 在 Open Remote Desktop (打开远程桌面) 对话框中，选择 Use EC2 keypair to log on (使用 EC2 密钥对进行登录)，然后选择 OK (确定)。

   如果您未在 AWS Toolkit 中存储密钥对，请指定包含私钥的 PEM 文件。

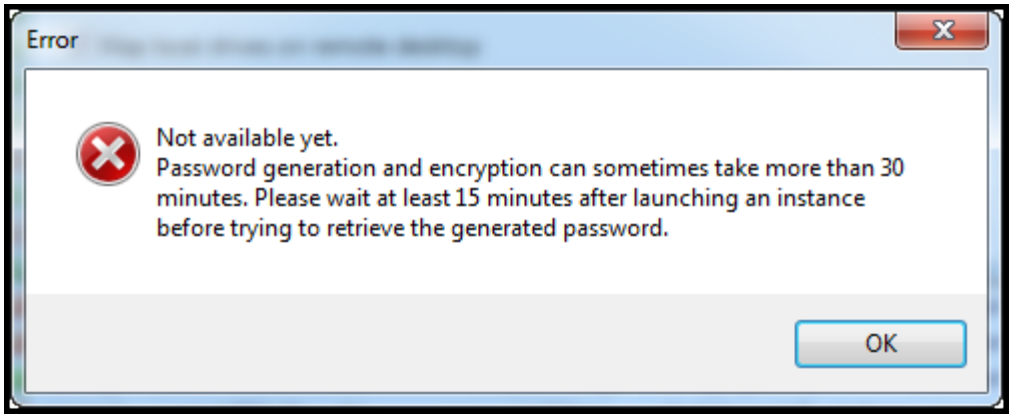


Open Remote Desktop (打开远程桌面) 对话框

3. Remote Desktop (远程桌面) 窗口将打开。由于已使用密钥对进行身份验证，因此您无需登录。您将以管理员身份在 Amazon EC2 实例上运行。

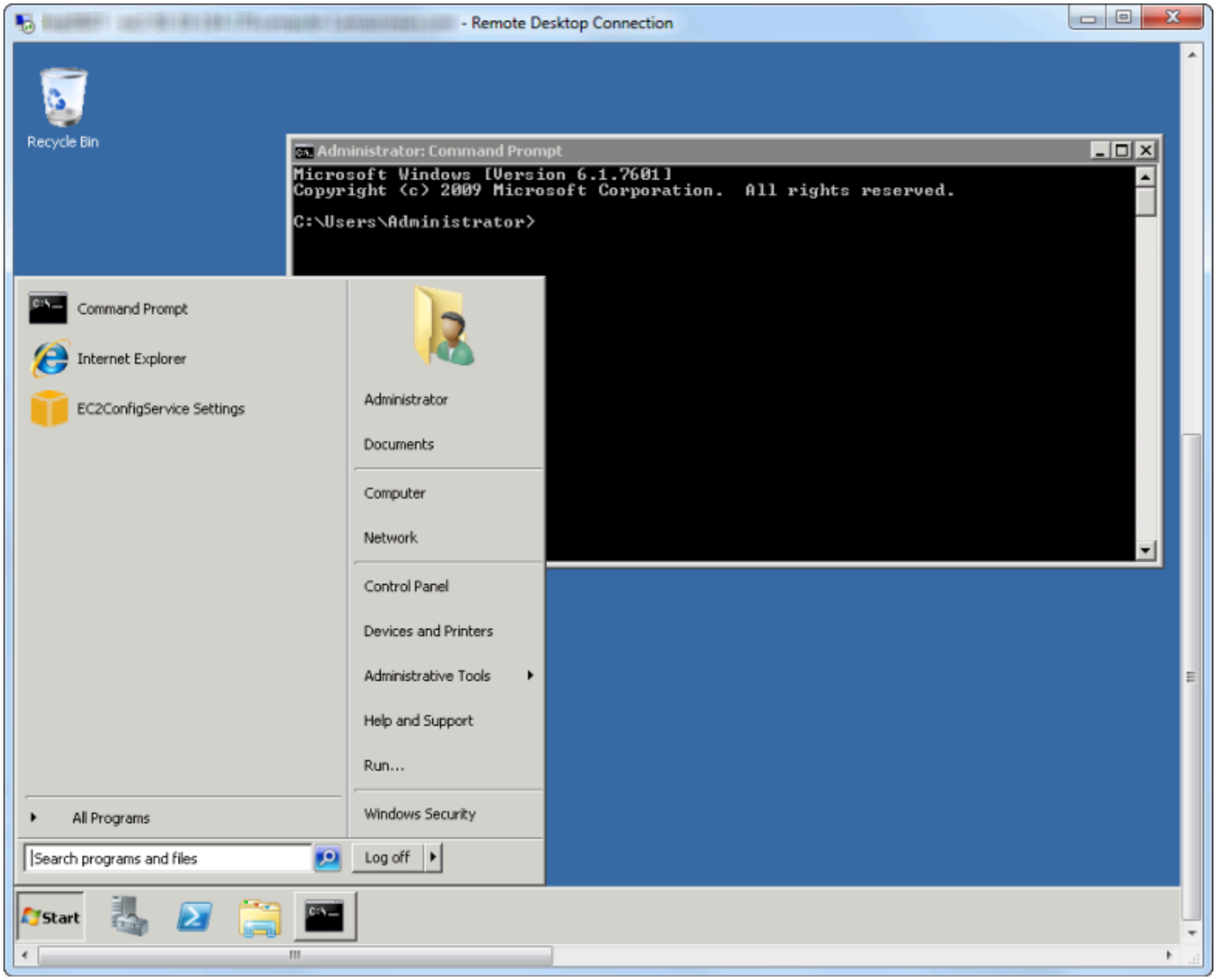
   如果 EC2 实例仅最近启动，您可能因以下两个原因无法连接：

   - 远程桌面服务可能尚未启动并运行。请等待几分钟，然后重试。
   - 密码信息可能尚未传输到实例。在此情况下，您将看到与下面类似的消息框。



密码尚不可用

以下屏幕截图显示以管理员身份通过远程桌面连接的用户。



远程桌面

结束 Amazon EC2 实例

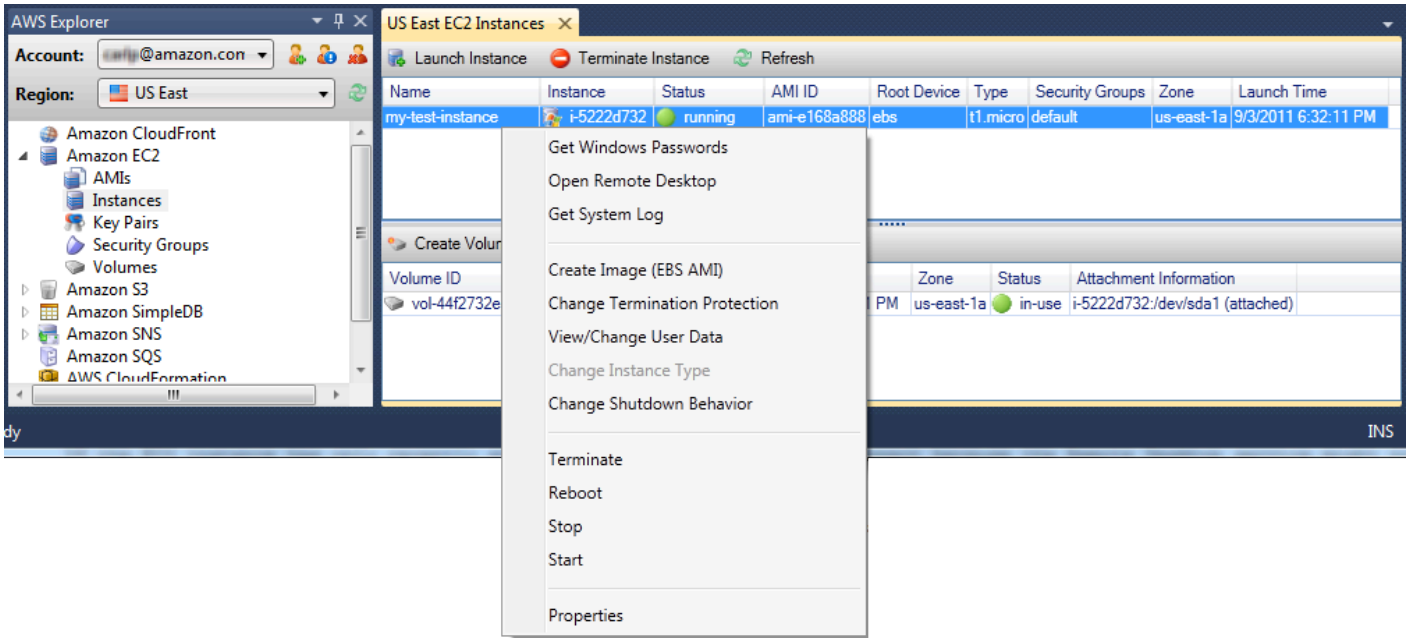
使用该 AWS 工具包，您可以从 Visual Studio 中停止或终止正在运行的 Amazon EC2 实例。要停止实例，EC2 实例必须使用 Amazon EBS 卷。如果 EC2 实例未使用 Amazon EBS 卷，您的唯一选择是终止实例。

如果您停止实例，EBS 卷上存储的数据将保留。如果您终止实例，实例的本地存储设备上存储的所有数据将丢失。在停止或终止的情况下，将停止向您收取 EC2 实例的费用。但是，如果您停止实例，将继续向您收取实例停止后保留 EBS 存储的费用。

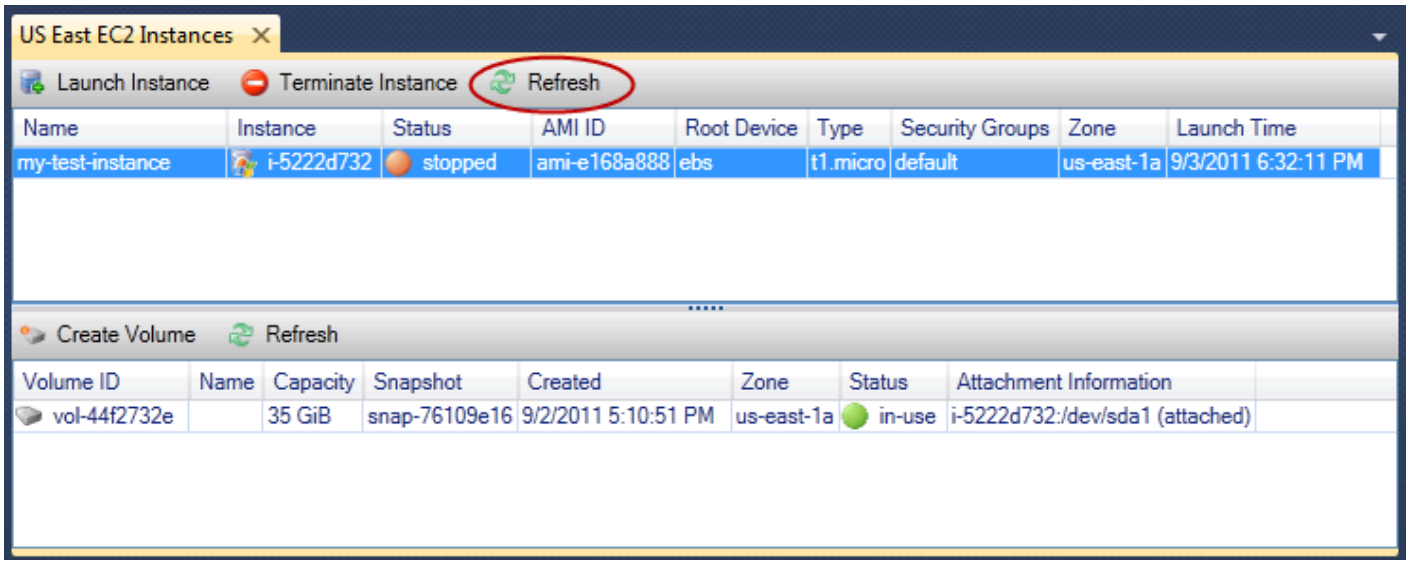
另一种结束实例的可能方式是，使用远程桌面连接到实例，然后从 Windows 开始菜单中，使用关机。您可将实例配置为在此方案中停止或终止。

停止 Amazon EC2 实例

1. 在 AWS 资源管理器中，展开 Amazon EC2 节点，打开实例的上下文（右键单击）菜单，然后选择查看。在 Instances (实例) 列表中，右键单击要停止的实例，然后从上下文菜单中选择 Stop (停止)。选择 Yes (是) 确认您要停止实例。

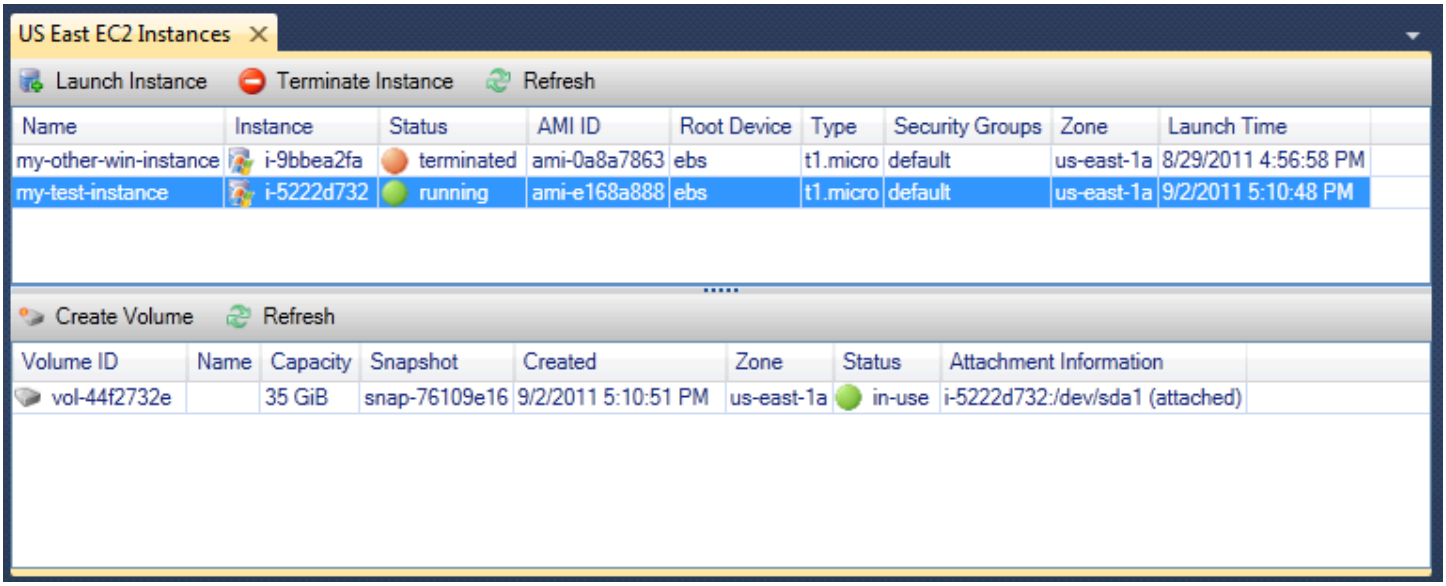


2. 在 Instances (实例) 列表的顶部，选择 Refresh (刷新) 以查看 Amazon EC2 实例状态的更改。由于我们已停止而不是终止实例，因此与实例关联的 EBS 卷仍处于活动状态。



终止的实例仍可见

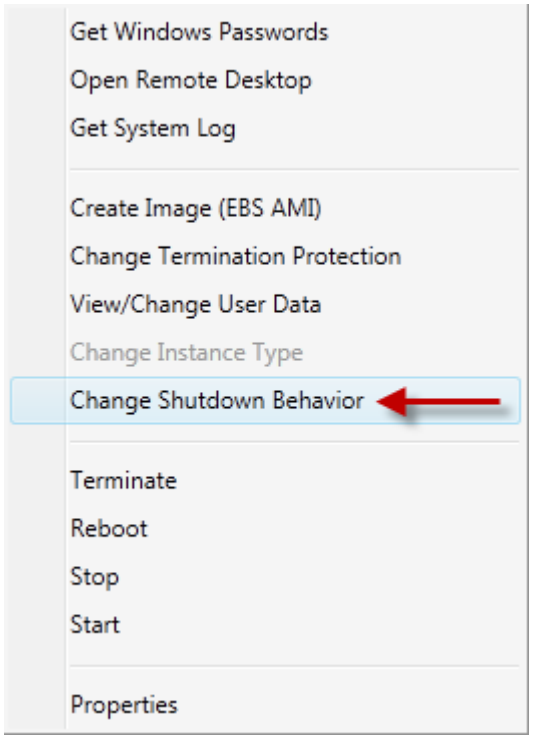
如果您终止一个实例，此实例将与正在运行的或已停止的实例一起继续显示在 Instance (实例) 列表中。最终， AWS 回收这些实例，它们就会从列表中消失。不会向您收取处于已终止状态的实例的费用。



指定 EC2 实例在关闭时的行为

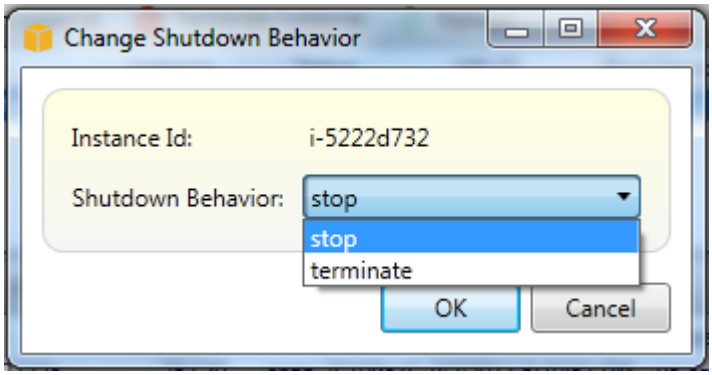
该 AWS 工具包允许您指定如果从 “开始” 菜单中选择 “关闭”，Amazon EC2 实例将停止还是终止。

1. 在 Instances (实例) 列表中，右键单击 Amazon EC2 实例，然后选择 Change shutdown behavior (更改关机行为)。



Change Shutdown Behavior (更改关机行为) 菜单项

2. 在 Change Shutdown Behavior (更改关机行为) 对话框中，从 Shutdown Behavior (关机行为) 下拉列表中，选择 Stop (停止) 或 Terminate (终止)。



管理 Amazon ECS 实例

AWS Explorer 提供了亚马逊弹性容器服务 (Amazon ECS) 集群和容器存储库的详细视图。您可以在 Visual Studio 开发环境中创建、删除和管理集群和容器详细信息。

修改服务属性

您可以在集群视图中查看服务详细信息、服务事件和服务属性。

1. 在 AWS 资源管理器中，打开要管理的群集的上下文（右键单击）菜单，然后选择查看。
2. 在 ECS集群视图中，单击左侧的 Services (服务)，然后单击详细信息视图中的 Details (详细信息) 选项卡。您可以单击 Events (事件) 以查看事件消息，也可以单击 Deployments (部署) 以查看部署状态。
3. 单击编辑。您可以更改所需任务计数以及最小和最大的正常运行状态百分比。
4. 单击 Save (保存) 以接受更改，或单击 Cancel(取消) 以恢复为现有值。

停止任务

您可以查看任务的当前状态并停止集群视图中的一个或多个任务。

停止任务

1. 在 AWS Explorer 中，打开包含要停止的任务的群集的上下文（右键单击）菜单，然后选择 “查看”。
2. 在 ECS 集群视图中，单击左侧的 Tasks (任务)。
3. 确保将 Desired Task Status (所需任务状态) 设置为 `Running`。选择要停止的各个任务，然后单击 Stop (停止) 或单击 Stop All (全部停止) 以选择和停止所有正在运行的任务。
4. 在 Stop Tasks (停止任务) 对话框中选择Yes (是)。

删除服务

您可以从群集视图中删除集群中的服务。

删除集群服务

1. 在 AWS Explorer 中，打开包含要删除服务的集群的上下文（右键单击）菜单，然后选择 “查看”。
2. 在 ECS 集群视图中，单击左侧的 Services (服务)，然后单击 Delete (删除)。
3. 在 Delete Cluster (删除集群) 对话框中，如果您的集群中有负载均衡器和目标组，您可以选择将它们与集群一同删除。删除该服务后，将不会再使用它们。
4. 在 Delete Cluster (删除集群) 对话框中，选择 OK (确定)。集群被删除后，也将从 AWS 各区服务浏览器中移除。

删除集群

您可以从 E AWS xplorer 中删除 Amazon 弹性容器服务集群。

删除集群

1. 在 AWS 资源管理器中，在 Amazon ECS 的集群节点下打开要删除的集群的上下文（右键单击）菜单，然后选择删除。
2. 在 Delete Cluster (删除集群) 对话框中，选择 OK (确定)。集群被删除后，也将从 AWS 各区服务浏览器中移除。

创建存储库

您可以通过 E AWS xplorer 创建 Amazon 弹性容器注册表存储库。

创建存储库

1. 在 AWS 资源管理器中，打开 Amazon ECS 下存储库节点的上下文（右键单击）菜单，然后选择“创建存储库”。
2. 在 Create Repository (创建存储库) 对话框中，为存储库命名，然后选择 OK (确定)。

删除存储库

您可以从 AWS 资源管理器中删除 Amazon 弹性容器注册表存储库。

删除存储库

1. 在 AWS 资源管理器中，打开 Amazon ECS 下存储库节点的上下文（右键单击）菜单，然后选择删除存储库。
2. 在 Delete Repository (删除存储库) 对话框中，您可以选择删除存储库（即使其中包含映像）。否则，它只有为空时才会被删除。单击 Yes (是)。

通过 AWS 资源管理器管理安全组

Toolkit for Visual Studio 使您可以创建和配置安全组，以便与 Amazon Elastic Compute Cloud（Amazon EC2）实例和 CloudFormation配合使用。当您启动 Amazon EC2 实例或向其部署应用程序时 CloudFormation，您需要指定一个与这些 Amazon EC2 实例关联的安全组。（部署以 CloudFormation 创建 Amazon EC2 实例。）

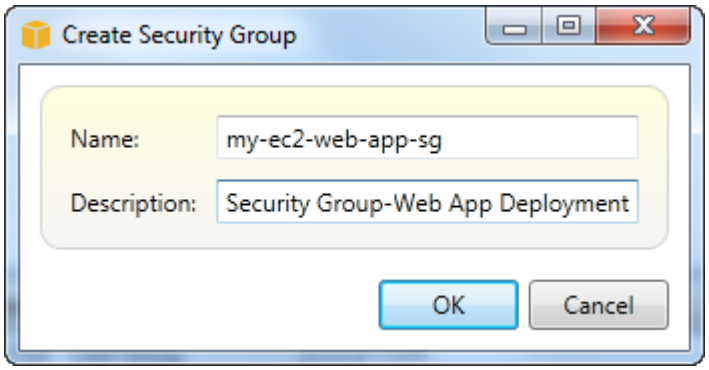
安全组的行为类似于传入网络流量的防火墙。安全组指定 Amazon EC2 实例上允许的网络流量的类型。它还指定仅接受来自特定 IP 地址或指定用户或其他安全组的传入流量。

正在创建安全组

在此部分中，我们将创建一个安全组。创建安全组后，安全组将未配置任何权限。权限配置是通过其他操作来处理的。

创建安全组

1. 在 AWS 资源管理器中，在 Amazon EC2 节点下，打开安全组节点上的上下文（右键单击）菜单，然后选择查看。
2. 在 EC2 Security Groups (EC2 安全组) 选项卡上，选择 Create Security Group (创建安全组)。
3. 在 Create Security Group (创建安全组) 对话框中，键入该安全组的名称和描述，然后选择 OK (确定)。

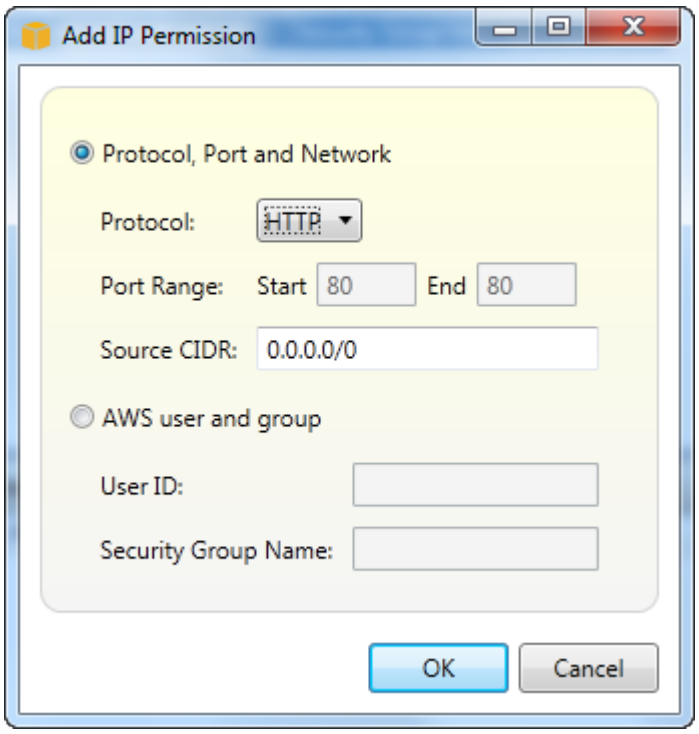


向安全组添加权限

在此部分中，我们将向安全组添加权限以通过 HTTP 和 HTTPS 协议允许 Web 流量。我们还将允许其他计算机通过使用 Windows 远程桌面协议 (RDP) 进行连接。

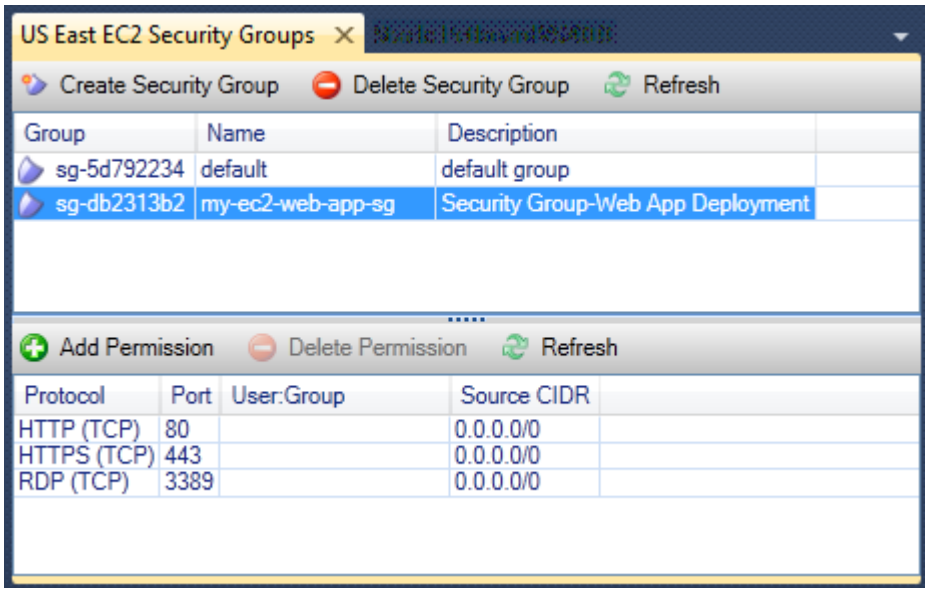
向安全组添加权限

1. 在 EC2 Security Groups (EC2 安全组) 选项卡上，选择安全组，然后选择 Add Permission (添加权限) 按钮。
2. 在 Add IP Permission (添加 IP 权限) 对话框中，选择 Protocol, Port and Network (协议、端口和网络) 单选按钮，然后从 Protocol (协议) 下拉列表中，选择 HTTP。端口范围将自动调整为端口 80（HTTP 的默认端口）。Source CIDR (源 CIDR) 字段默认为 0.0.0.0/0，这指定将接受来自任何外部 IP 地址的 HTTP 网络流量。选择确定。



为此安全组打开端口 80 (HTTP)

3. 对 HTTPS 和 RDP 重复此过程。您的安全组权限现在看起来应与下面内容类似。



您还可通过指定用户 ID 和安全组名称在安全组中设置权限。在此情况下，此安全组中的 Amazon EC2 实例将接受来自指定安全组中的 Amazon EC2 实例的所有传入网络流量。您还必须指定用户 ID 以消除安全组名称的歧义；安全组名称不要求在所有名称中都是唯一的。 AWS有关安全组的更多信息，请转到 EC2 文档。

从 Amazon EC2 实例创建 AMI

您可以使用 AWS Toolkit for Visual Studio创建亚马逊机器映像（AMI）。有关更多详细信息 AMIs，请参阅《适用于 Windows 的亚马逊弹性计算云用户指南》中的亚马逊系统映像 (AMI) 主题。

要从现有 Amazon EC2 实例创建 AMI，请完成以下过程。

从现有 Amazon EC2 实例创建 AMI

1. 在 AWS Toolkit Explorer 中，展开 Amazon EC2，然后选择实例以查看现有实例的列表。
2. 右键单击要用作 AMI 基础的实例，然后选择创建映像（ABS AMI）以打开创建映像对话窗口。
3. 在创建映像对话窗口中，将映像的名称和描述添加到提供的字段中，然后选择确定按钮继续。
4. 当映像创建完成后，Visual Studio 中会打开映像已创建确认窗口，请选择确定按钮以继续。

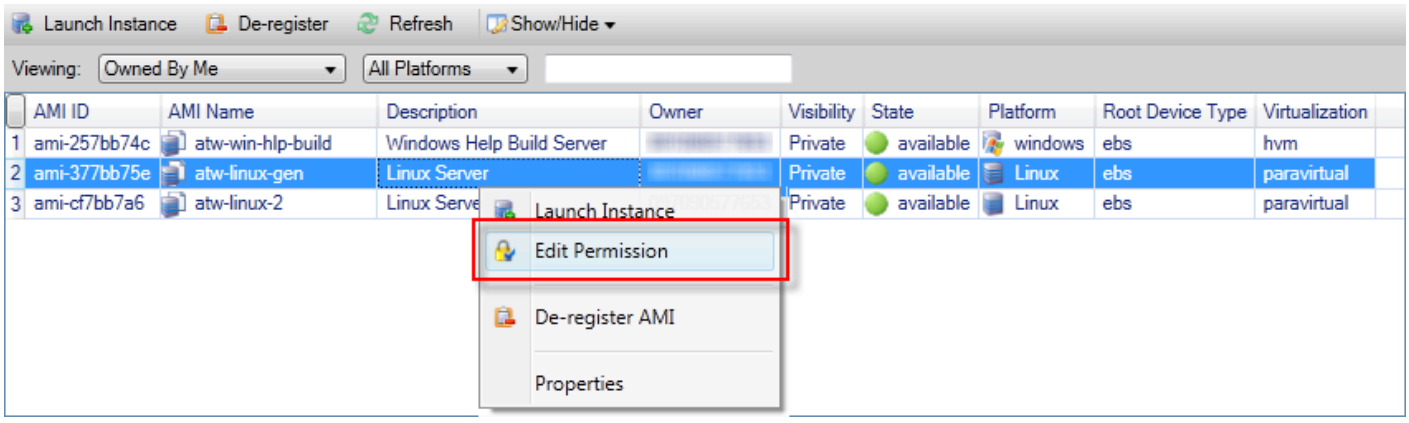
要使用 AWS 工具包查看您的新 AMI，请展开 Amazon EC2，然后双击AMIs在 Visual Studio 编辑器 payne 中打开一个显示现有 AMIs列表的窗口。如果您未在列表中看到您的新 AMI，请选择位于 AMI 窗口顶部的刷新按钮。

在 Amazon 系统映像上设置启动许可

您可以从 AWS Explorer 中的AMIs视图中设置亚马逊系统映像 (AMIs) 的启动权限。您可以使用 “设置 AMI 权限” 对话框从中复制权限 AMIs。

设置 AMI 上的权限

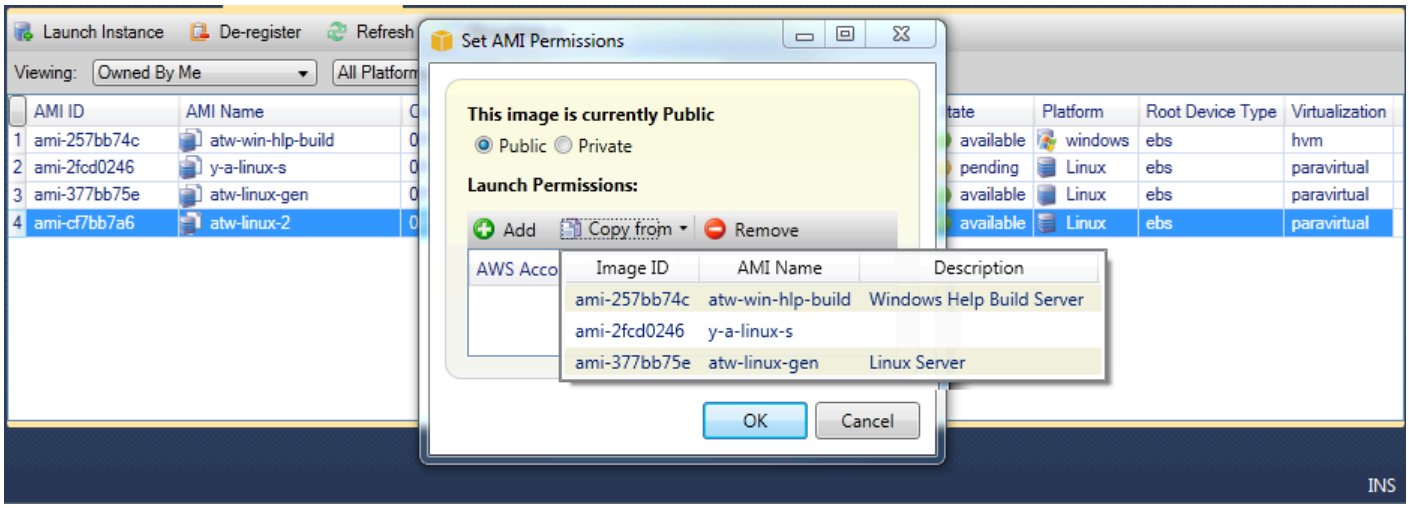
1. 在 AWS 资源管理器的AMIs视图中，打开 AMI 的上下文（右键单击）菜单，然后选择编辑权限。



2. Set AMI Permissions (设置 AMI 权限) 对话框中有三个选项：
   - 要授予启动权限，请选择 “添加”，然后键入要向其授予启动权限的 AWS 用户的账号。

- 要移除启动权限，请选择要移除启动权限的 AWS 用户的账号，然后选择 “移除”。
- 要将许可从一个 AMI 复制到另一个 AMI，请从列表中选择一个 AMI，然后选择 Copy from (复制自)。具有您选择的 AMI 上的启动许可的用户将获得当前 AMI 上的启动许可。您可以使用 Copy-fro AMIs m 列表中的其他人重复此过程， AMIs 将多个权限复制到目标 AMI。

  Copy-f rom 列表仅包含在 Explorer 中显示AMIs视图时处于活动状态的账户所 AMIs 拥有的 AWS 内容。因此， AMIs 如果活跃账户没有其他所有权， AMIs 则复制自列表可能不会显示任何内容。



Copy AMI permissions (复制 AMI 权限) 对话框

Amazon Virtual Private Cloud (VPC)

通过 Amazon Virtual Private Cloud（Amazon VPC），您可以将 Amazon Web Services 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。有关更多信息，请转到 Amazon VPC 用户指南。

利用 Toolkit for Visual Studio，开发人员可以访问 VPC 功能，该功能类似于由 AWS 管理控制台公开的功能，但位于 Visual Studio 开发环境中。 AWS Explorer 的 Amazon VPC 节点包括以下区域的子节点。

- VPCs
- 子网
- 弹性 IPs
- 互联网网关
- Network ACLs

- 路由表
- 安全组

创建用于部署的公共 VPC AWS Elastic Beanstalk

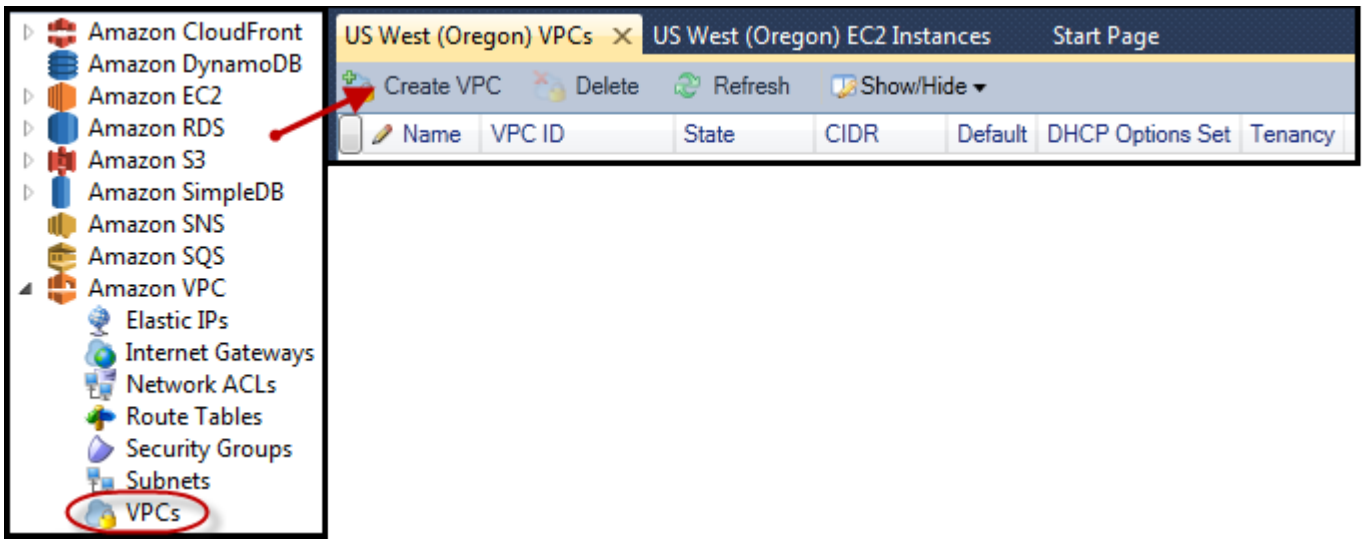
本部分介绍如何创建同时包含公有子网和私有子网的 Amazon VPC。公有子网包含一个 Amazon EC2 实例，该实例将执行网络地址转换（NAT）以支持私有子网中的实例与公共 Internet 通信。两个子网必须位于同一可用区 (AZ)。

这是在 VPC 中部署 AWS Elastic Beanstalk 环境所需的最低 VPC 配置。在这种情况下，托管您的应用程序的 Amazon EC2 实例位于私有子网中；将传入流量路由到您的应用程序的 Elastic Load Balancing 负载均衡器位于公有子网内。

有关网络地址转换（NAT）的更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的 NAT 实例。有关如何将部署配置为使用 VPC 的示例，请参阅部署到 Elastic Beanstalk。

创建公有-私有子网 VPC

1. 在 AWS 资源管理器的 Amazon VPC 节点中，打开VPCs子节点，然后选择创建 VPC。

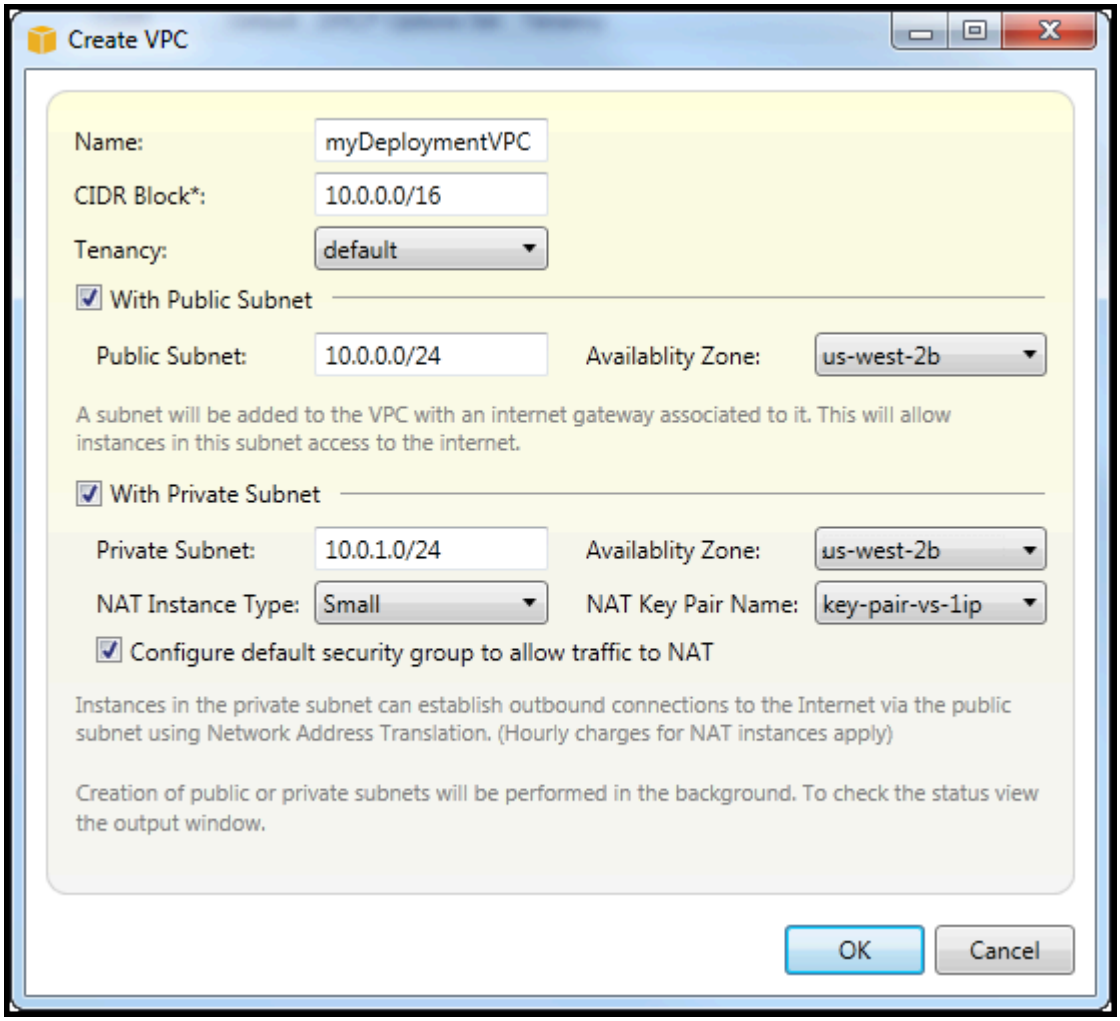


2. 按下面所示配置 VPC：
   - 键入 VPC 的名称。
   - 选中 With Public Subnet (使用公有子网) 和 With Private Subnet (使用私有子网) 复选框。
   - 从每个子网的 Availability Zone (可用区) 下拉列表框中，选择一个可用区。确保对两个子网使用相同的 AZ。

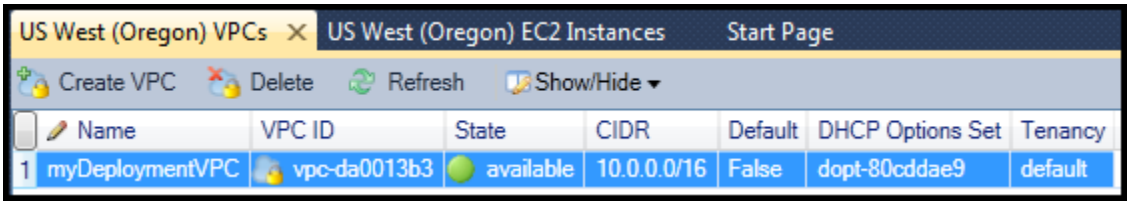
- 对于 NAT Key Pair Name (NAT 密钥对名称) 中的私有子网，提供一个键前缀。此密钥对用于执行从私有子网到公共 Internet 的网络地址转换的 Amazon EC2 实例。
- 选中 Configure default security group to allow traffic to NAT (配置默认安全组以允许到 NAT 的流量) 复选框。

键入 VPC 的名称。选中 With Public Subnet (使用公有子网) 和 With Private Subnet (使用私有子网) 复选框。从每个子网的 Availability Zone (可用区) 下拉列表框中，选择一个可用区。确保对两个子网使用相同的 AZ。对于 NAT Key Pair Name (NAT 密钥对名称) 中的私有子网，提供一个键前缀。此密钥对用于执行从私有子网到公共 Internet 的网络地址转换的 Amazon EC2 实例。选中 Configure default security group to allow traffic to NAT (配置默认安全组以允许到 NAT 的流量) 复选框。

选择确定。

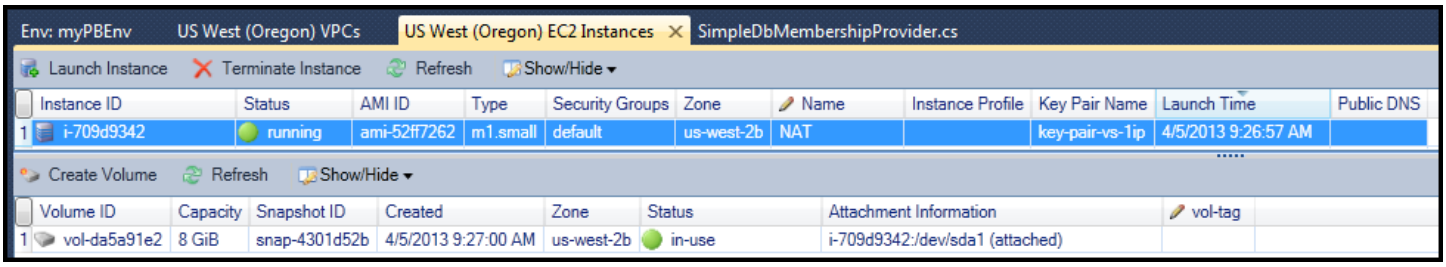


您可以在 AWS 资源管理器的VPCs选项卡中查看新的 VPC。



启动 NAT 实例可能需要几分钟。当它可用时，您可以通过在 AWS Explorer 中展开 Amazon EC2 节点，然后打开实例子节点来查看它。

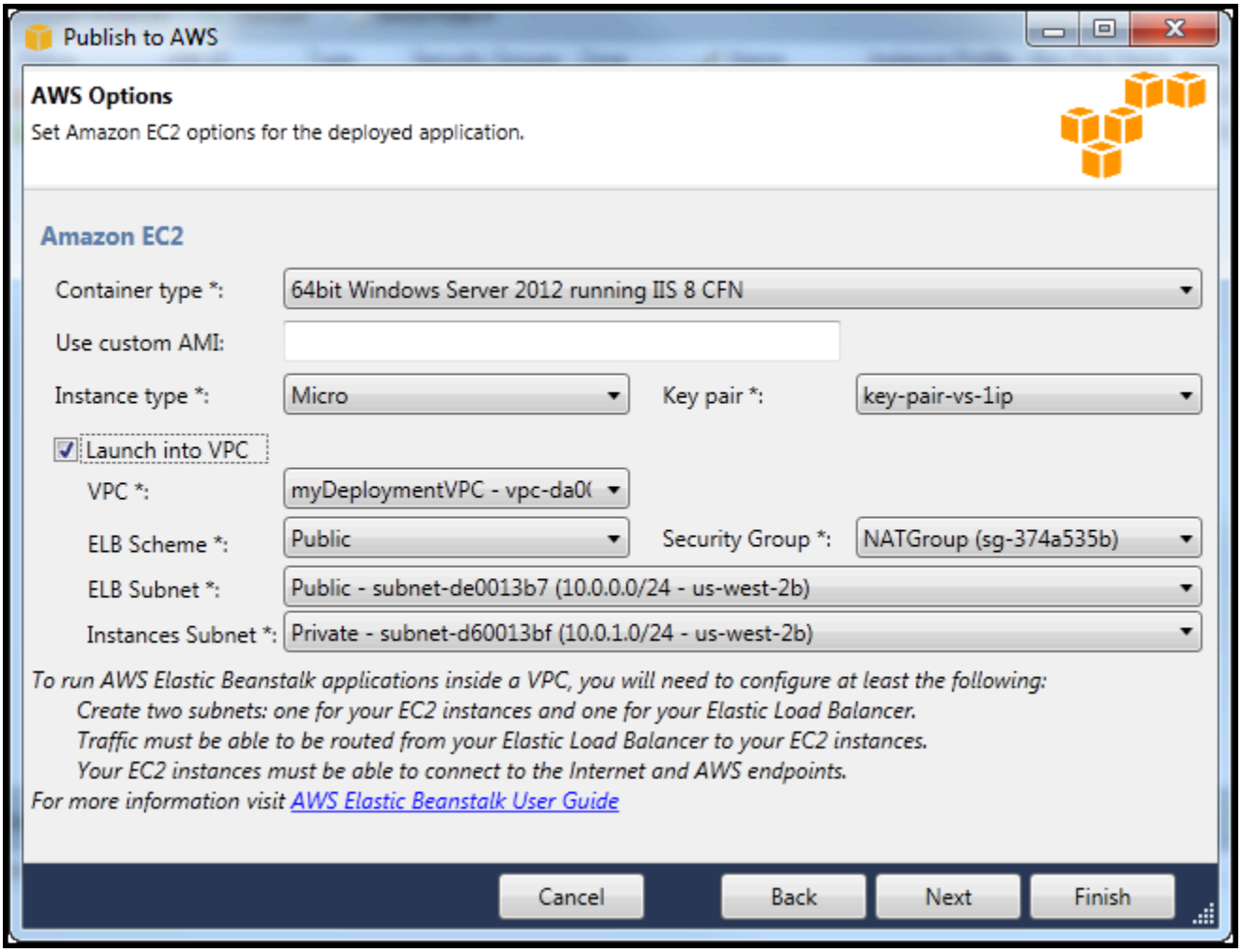
系统会自动为 NAT 实例创建一个 Amazon Elastic Block Store（Amazon EBS）卷。有关 Amazon EBS 的更多信息，请参阅《适用于 Linux 实例的 Amazon EC2 用户指南》中的 Amazon Elastic Block Store（Amazon EBS）主题。



如果您将应用程序部署到 AWS Elastic Beanstalk 环境并选择在 VPC 中启动该环境，则工具包将使用您的 VPC 的配置信息填充发布到 Amazon Web Services对话框。

Toolkit 仅使用在 Toolkit 中创建 VPCs 的信息填充对话框，而不是使用工具包 VPCs 创建的信息。AWS 管理控制台这是因为，当 Toolkit 创建 VPC 时，它将对 VPC 的组件进行标记，以便能够访问这些组件的信息。

来自部署向导的以下屏幕截图显示了一个对话框的示例，该对话框是使用来自在 Toolkit 中创建的 VPC 的值填充的。



删除 VPC

要删除 VPC，您必须先终止其中的所有 Amazon EC2 实例。

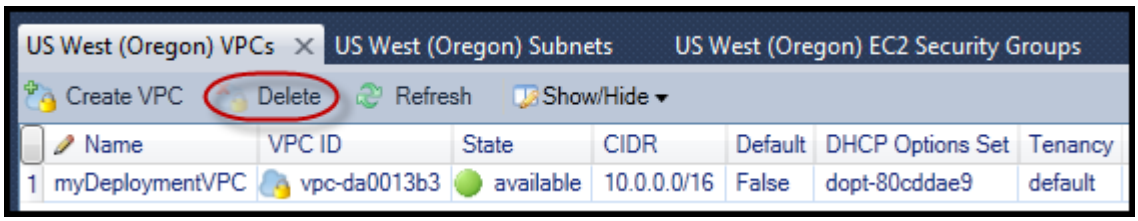
1. 如果您已将应用程序部署到 VPC 中的某个 AWS Elastic Beanstalk 环境，请删除该环境。这将终止托管您的应用程序的所有 Amazon EC2 实例以及 Elastic Load Balancing 负载均衡器。

   如果您尝试直接终止托管您的应用程序的实例而不删除该环境，Auto Scaling 服务将自动创建新实例来替换删除的实例。有关更多信息，请访问 Auto Scaling 开发人员指南。

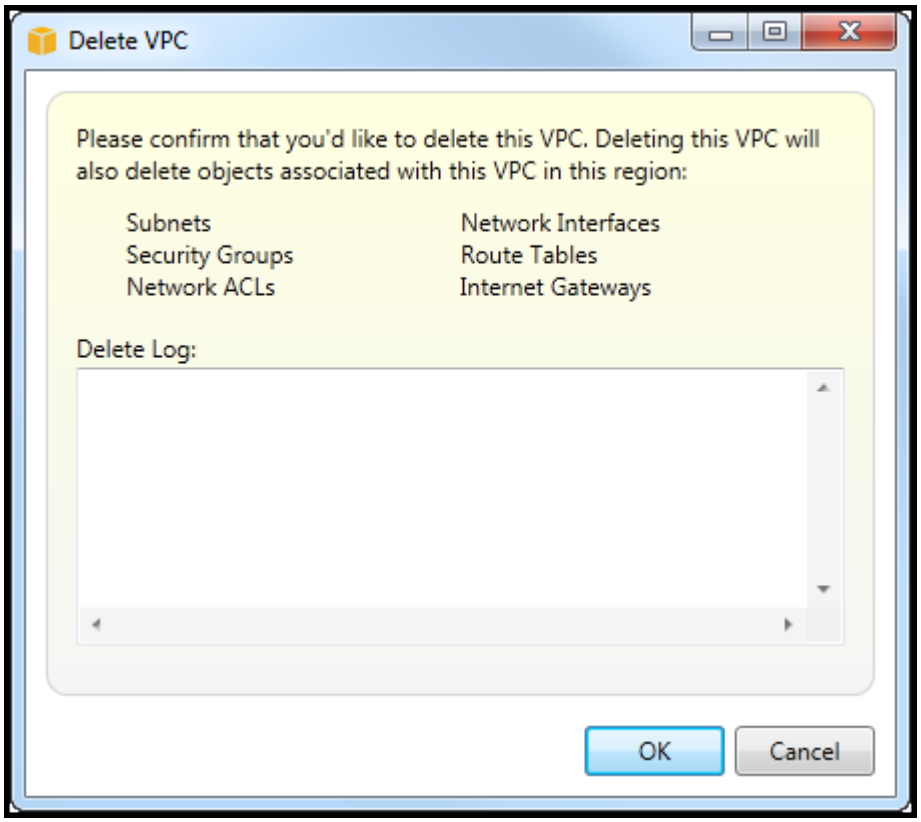
2. 删除 VPC 的 NAT 实例。

   您无需删除与 NAT 实例关联的 Amazon EBS 卷即可删除 VPC。但是，如果您不删除卷，则会继续为其付费，即使您删除了 NAT 实例和 VPC 也是如此。

3. 在 VPC 选项卡上，选择 Delete (删除) 链接以删除 VPC。



4. 在 Delete VPC (删除 VPC) 对话框中，选择 OK (确定)。



使用适用于 Visual Studio 的 CloudFormation 模板编辑器

Toolkit for Visual Studio 包含适用于 Visual Studio 的 CloudFormation 模板编辑器和 CloudFormation 模板项目。支持的功能包括：

- 使用随附的 CloudFormation 模板项目类型创建新模板（空模板或从现有堆栈或示例模板复制的模板）。
- 利用自动 JSON 验证、自动完成、代码折叠和语法突出显示来编辑模板。
- 模板中字段值的内部函数和资源参考参数的自动建议。
- 用于从 Visual Studio 中对您的模板执行常见操作的菜单项。

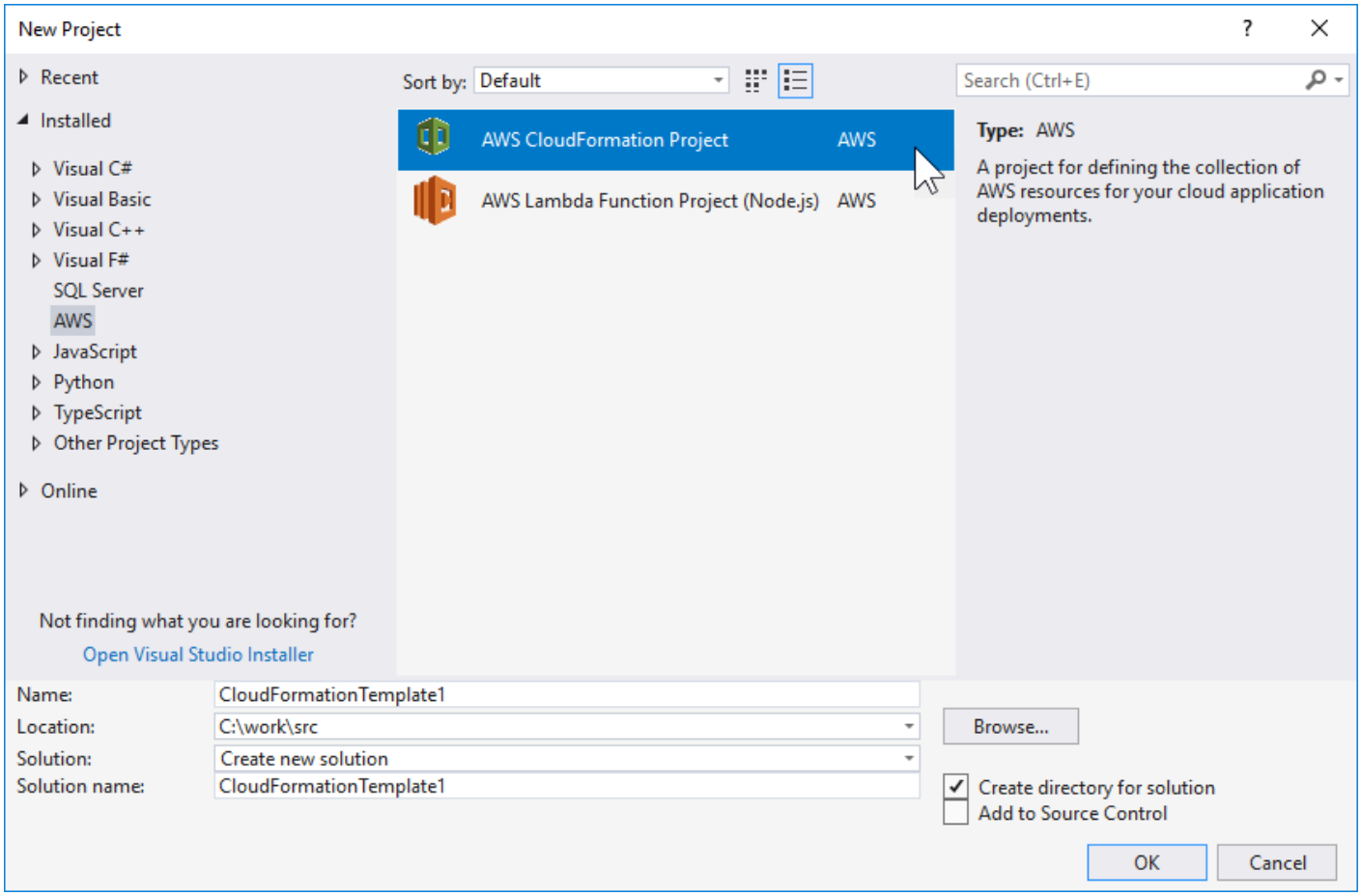
主题

在 Visual Studio 中创建 CloudFormation 模板项目

创建模板项目

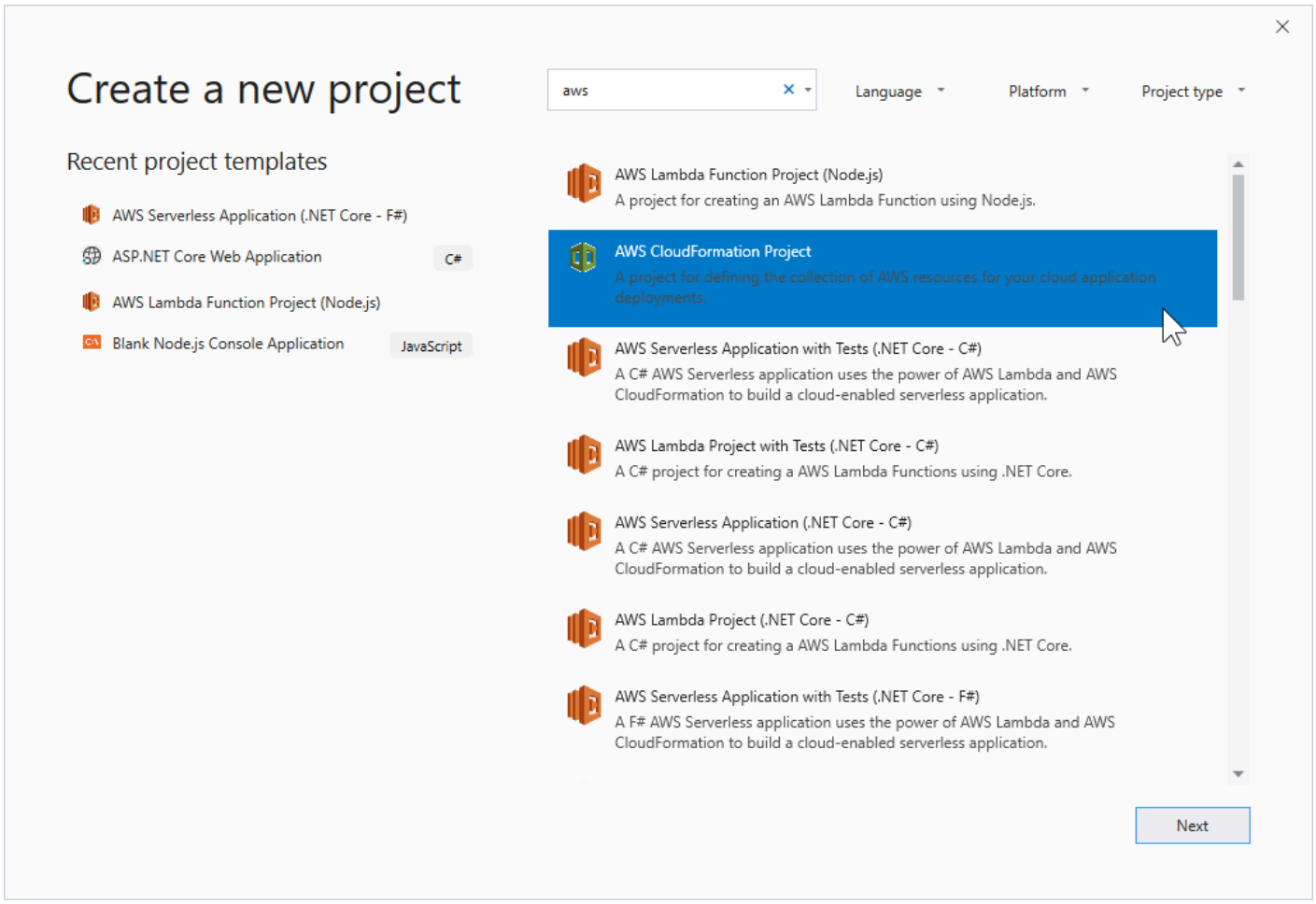
1. 在 Visual Studio 中，依次选择 File (文件)、New (新建) 和 Project (项目)。
2. 对于 Visual Studio 2017：

   在新建项目对话框中，展开已安装，然后选择 AWS。



   对于 Visual Studio 2019：

在新建项目对话框中，确保将语言、平台和项目类型下拉框都设置为“全部...”，然后在搜索字段中键入 aws。

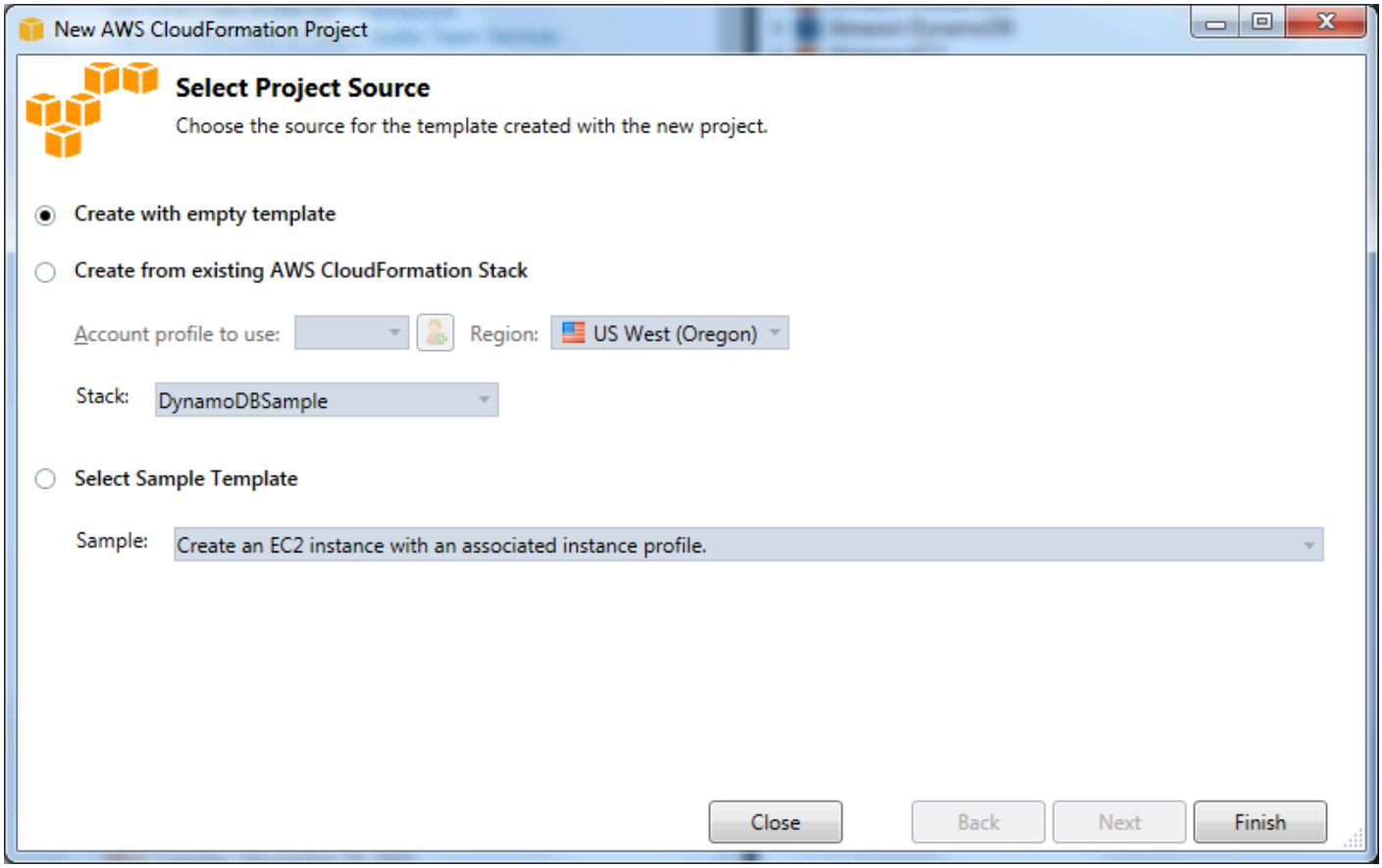


3. 选择AWS CloudFormation 项目模板。
4. 对于 Visual Studio 2017：

   为您的模板项目输入所需的名称、位置等，然后单击确定。

   对于 Visual Studio 2019：

   单击下一步。在下一个对话框中，为您的模板项目输入所需的名称、位置等，然后单击创建。
5. 在 Select Project Source (选择项目源) 页面上，选择您将创建的模板的源：
   - Create with empty template (使用空模板创建) 生成新的空 CloudFormation 模板。
   - 从现有 AWS |CFN| 堆栈创建从您账户中的现有堆栈生成模板。 AWS （ 该堆栈不需要具有状态 `CREATE_COMPLETE`。 ）
   - Select sample template (选择示例模板) 从 CloudFormation 示例模板之一生成模板。

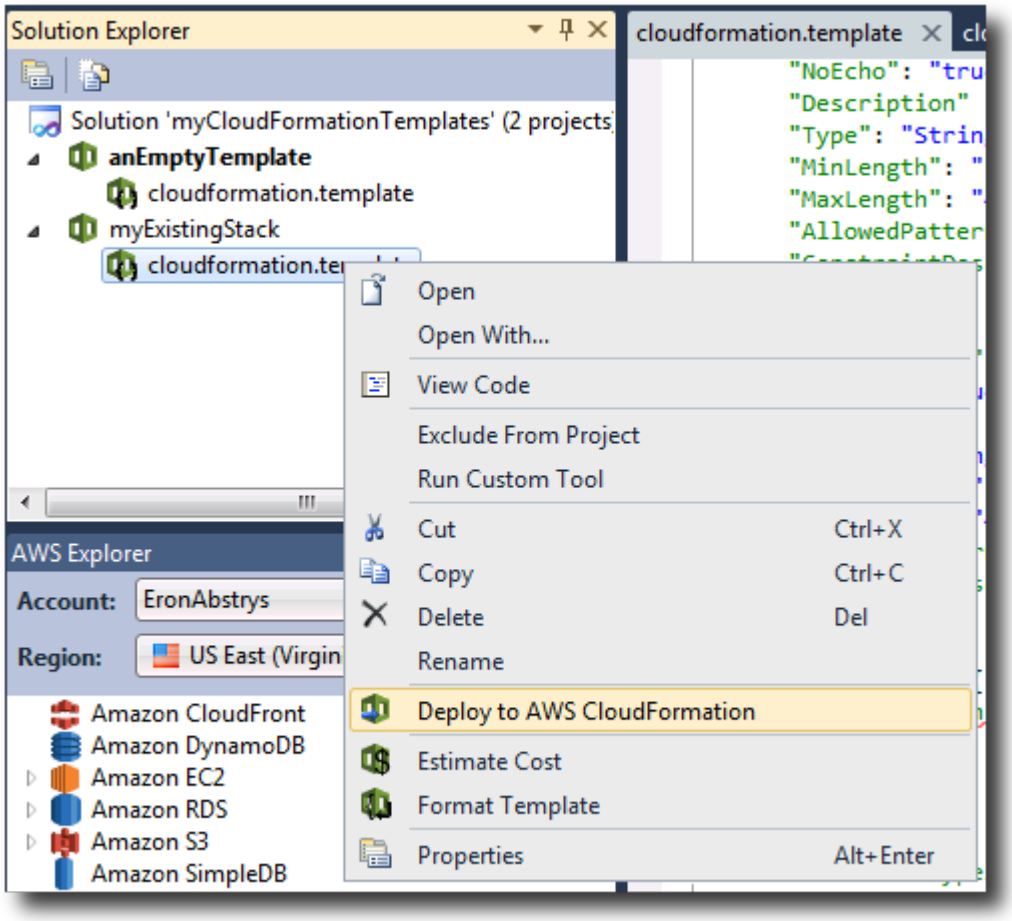


6. 要完成 CloudFormation 模板项目的创建，请选择 “完成”。

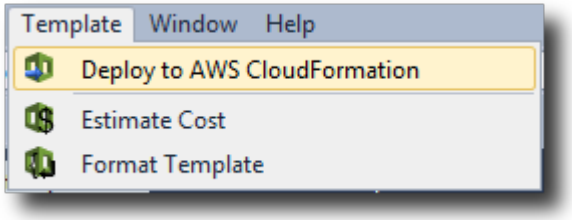
在 Visual Studio 中部署 CloudFormation 模板

部署 CFN 模板

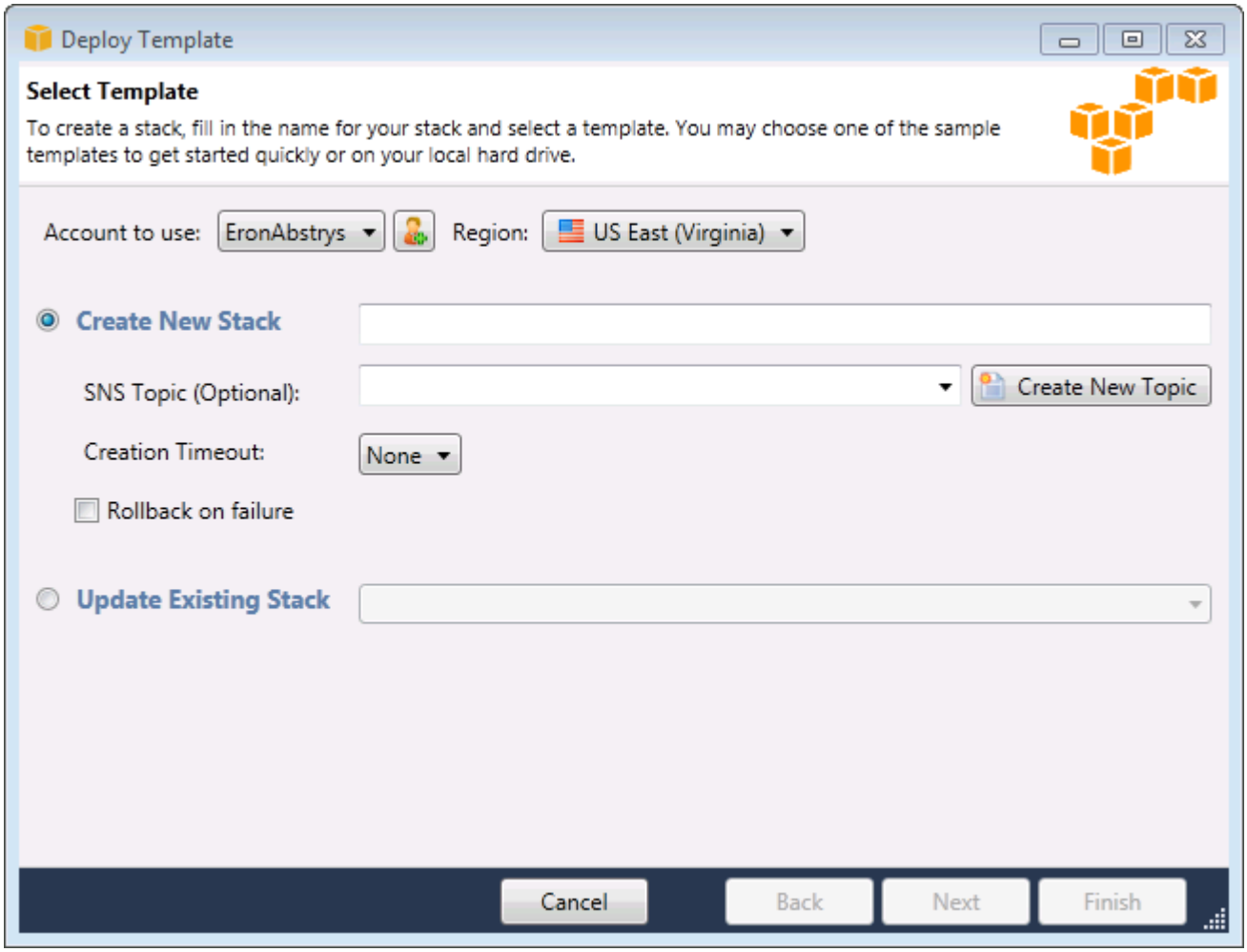
1. 在解决方案浏览器中，打开要部署的模板的上下文（右键单击）菜单，然后选择部署到 AWS CloudFormation。



或者，要部署您当前正在编辑的模板，请从模板菜单中，选择部署到 AWS CloudFormation。



2. 在部署模板页面上，选择 AWS 账户 用于启动堆栈的以及启动堆栈的区域。

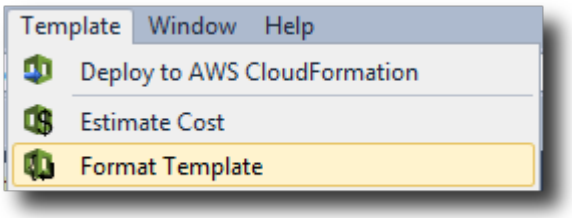


3. 选择 Create New Stack (创建新堆栈) 并为您的堆栈键入名称。
4. 选择以下任一选项（或不选择任何选项）：
   - 要接收有关堆栈的进度的通知，请从 SNS Topic (SNS 主题) 下拉列表中，选择 SNS 主题。您还可以通过选择 Create New Topic (创建新主题) 并在框中键入电子邮件地址来创建 SNS 主题。
   - 使用 Creati on Timeout（创建超时）指定在堆栈被宣布失败（除非清除 “失败时回滚” 选项，否则 CloudFormation 应允许创建堆栈多长时间）。
   - 如果您希望堆栈在失败时回滚（即，自行删除），请使用 Rollback on failure (失败时回滚)。如果您出于调试目的希望堆栈保持活动状态，请将此选项保持清除状态，即使堆栈未能完成启动。
5. 选择 Finish (完成) 以启动堆栈。

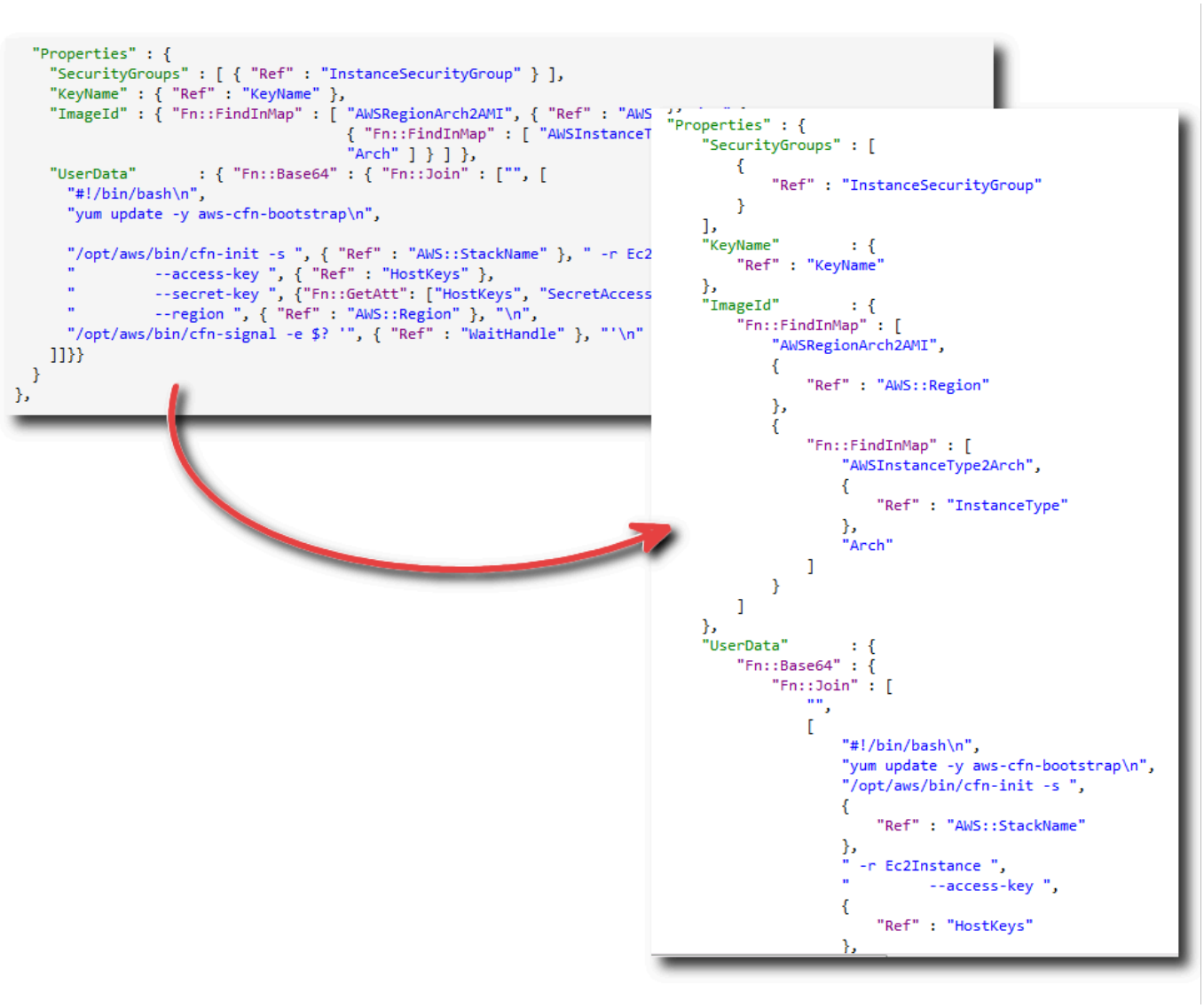
在视觉工作室中格式化 CloudFormation 模板

- 在解决方案资源管理器中，打开模板的上下文（右键单击）菜单，然后选择 Format Template (格式模板)。

或者，要对您当前编辑的模板进行格式设置，请从 Template (模板) 菜单中，选择 Format Template (格式模板)。



您的 JSON 代码将进行格式设置，以便清晰地呈现其结构。



使用 Exp AWS lorer 中的 Amazon S3

利用 Amazon Simple Storage Service（Amazon S3），您可以通过连接到 Internet 来存储和检索数据。您在 Amazon S3 上存储的所有数据与您的账户关联，而且默认情况下，只能由您访问。利用 Toolkit for Visual Studio，您可以将数据存储在 Amazon S3 上并查看、管理、检索和分布数据。

Amazon S3 使用了桶的概念，您可以将此视为类似于文件系统或逻辑驱动器。存储桶可包含文件夹（类似于目录）和对象（类似于文件）。在此部分中，我们将在介绍由 Toolkit for Visual Studio 公开的 Amazon S3 功能时使用这些概念。

> **Note**
>
> 要使用此工具，您的 IAM policy 必须为 `s3:GetBucketAcl`、`s3:GetBucket` 和 `s3:ListBucket` 操作授予权限。有关更多信息，请参阅 AWS IAM 策略概述。

创建 Amazon S3 存储桶

桶是 Amazon S3 中最基本的存储单位。

创建 S3 存储桶

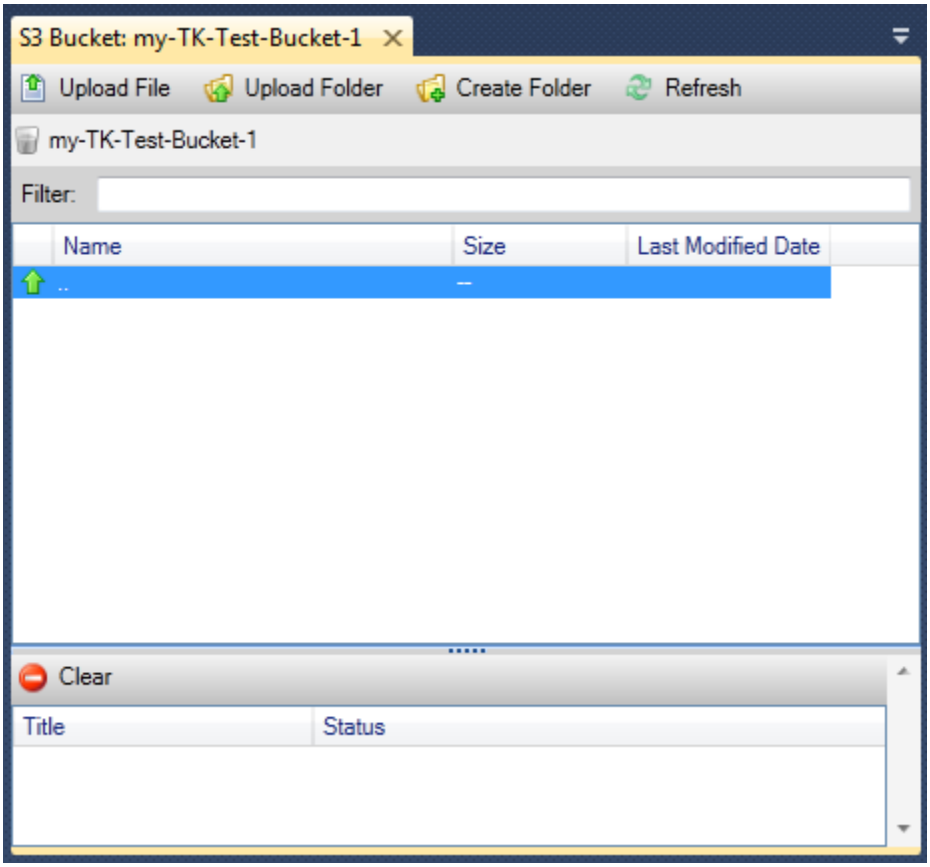
1. 在 AWS 资源管理器中，打开 Amazon S3 节点的上下文（右键单击）菜单，然后选择创建存储桶。
2. 在 Create Bucket (创建存储桶) 对话框中，为存储桶键入名称。桶名称在 AWS中必须是唯一的。有关其他约束的信息，请转到 Amazon S3 文档。
3. 选择确定。

通过资源管理器管理 Amazon S3 存储桶 AWS

在 AWS Explorer 中，当您打开 Amazon S3 存储桶的上下文（右键单击）菜单时，可以使用以下操作。

浏览

显示包含在存储桶中的对象的视图。从此处，您可以创建文件夹或者从您的本地计算机上传文件或整个目录和文件夹。下方窗格将显示有关上传过程的状态消息。要清除这些消息，请选择 Clear (清除) 图标。您也可以通过在 AWS Explorer 中双击存储桶名称来访问此存储桶视图。



属性

显示您可在其中执行以下操作的对话框：

- 将 Amazon S3 权限范围设置为：
  - 作为存储桶拥有者的您。
  - 已在 AWS上进行身份验证的所有用户。
  - 具有 Internet 访问权限的每个人。
- 开启对存储桶的日志记录。
- 使用 Amazon Simple Notification Service（Amazon SNS）设置通知，这样，在您使用低冗余存储（RRS）时，如果发生数据丢失，您将会接到通知。RRS 是一个 Amazon S3 存储选项，与标准存储相比，它提供的持久性较差，但成本较低。有关更多信息，请参阅 S3 FAQs。
- 使用存储桶中的数据创建静态网站。

Policy

使您能够为存储桶设置 AWS Identity and Access Management (IAM) 策略。有关更多信息，请转到 IAM 文档及 IAM 和 S3 的使用案例。

创建预签名 URL

使您能够生成一个限时的 URL，您可分配该 URL 以提供对存储桶内容的访问权限。有关更多信息，请参阅如何创建预签名 URL。

查看分段上传

使您可以查看分段上传。Amazon S3 支持将大型对象上传分解为多个部分以使上传过程更高效。有关更多信息，请转到 S3 文档中的分段上传的讨论。

删除

使您能够删除存储桶。您只能删除空桶。

将文件和文件夹上传到 Amazon S3

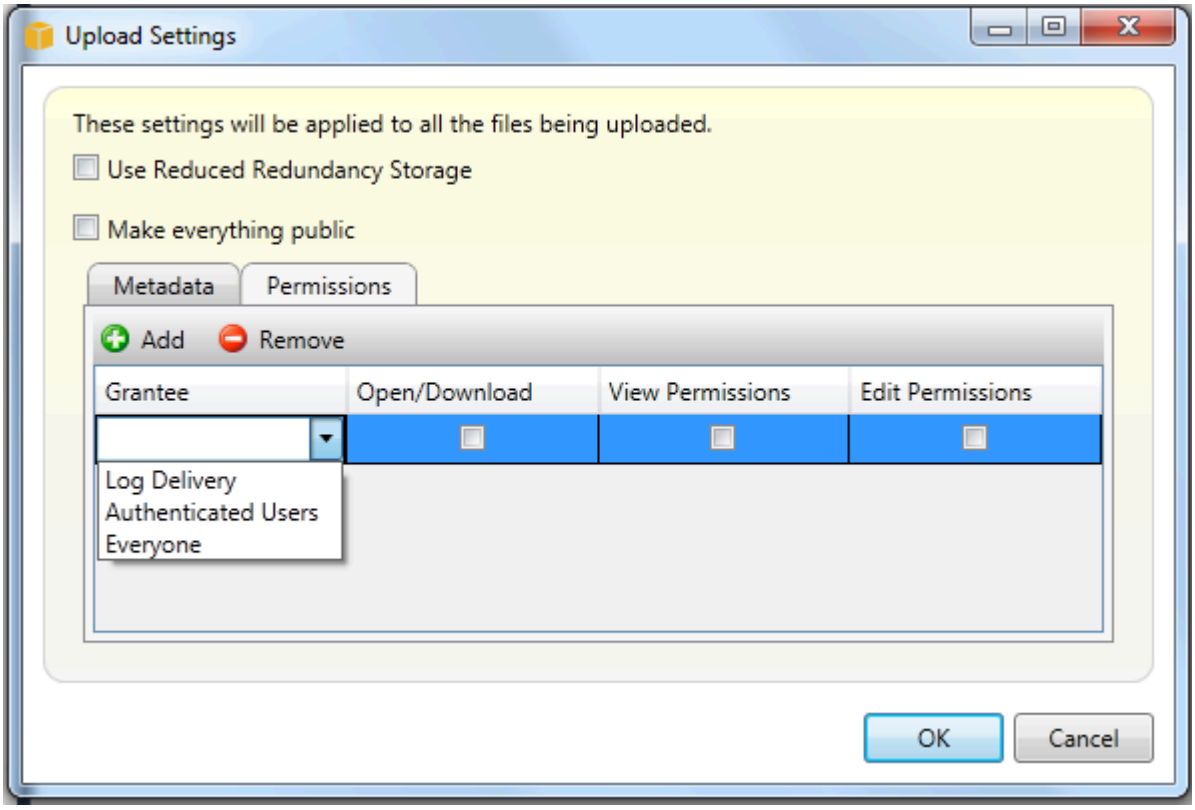
您可以使用 AWS Explorer 将文件或整个文件夹从本地计算机传输到任何存储桶。

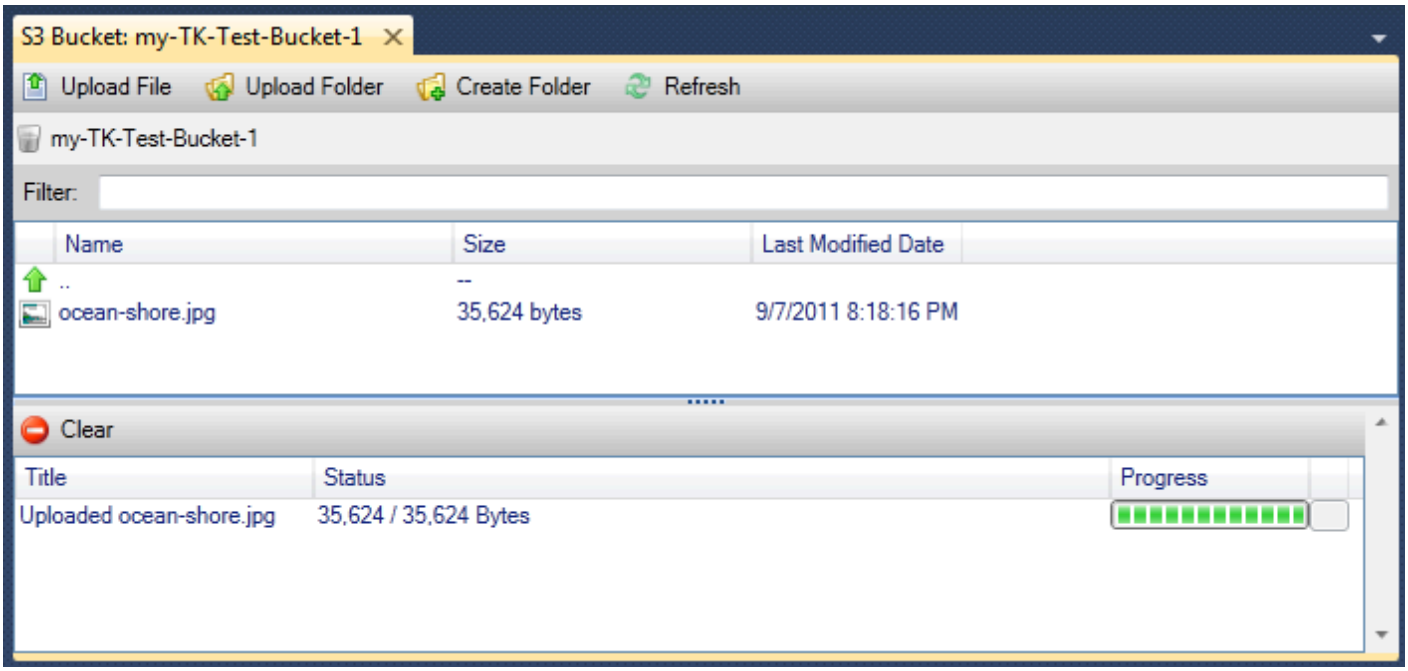
> **Note**
>
> 如果您上传的文件或文件夹与已存在于 Amazon S3 桶中的文件或文件夹具有相同的名称，则您上传的文件将覆盖现有文件而不进行提示。

将文件上传到 S3

1. 在 AWS 资源管理器中，展开 Amazon S3 节点，双击存储桶或打开存储桶的上下文（右键单击）菜单并选择浏览。
2. 在您的存储桶的 Browse (浏览) 视图中，选择 Upload File (上传文件) 或 Upload Folder (上传文件夹)。
3. 在 File-Open (文件-打开) 对话框中，导航到要上传的文件，选择这些文件，然后选择 Open (打开)。如果您要上传文件夹，请导航到并选择该文件夹，然后选择 Open (打开)。

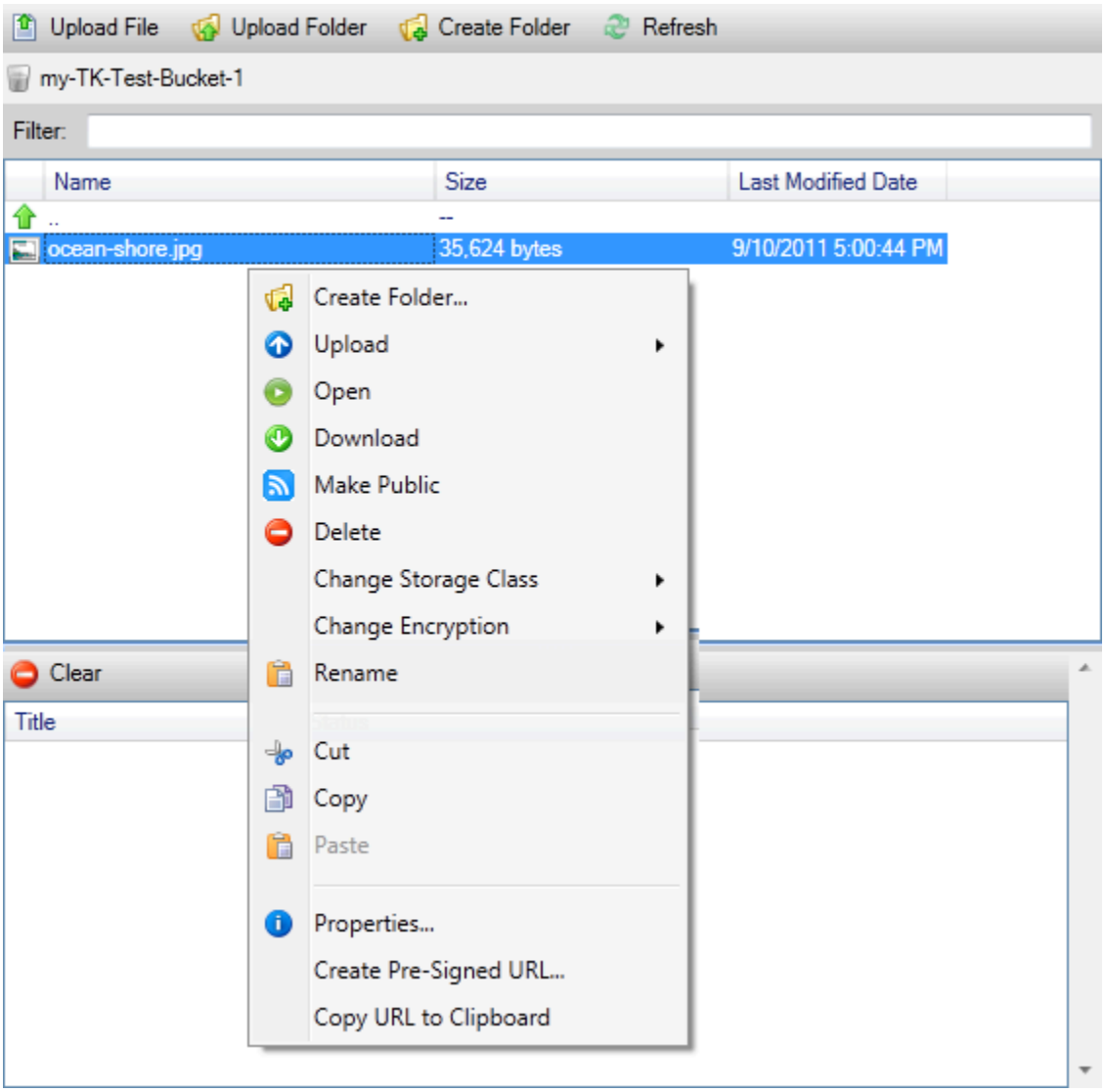
   利用 Upload Settings (上传设置) 对话框，您能够在要上传的文件或文件夹上设置元数据和权限。选中 Make everything public (将所有项公开化) 复选框等同于将 Open/Download (打开/下载) 权限设置为 Everyone (所有人)。您可以为上传的文件选择使用 Reduced Redundancy Storage (减小冗余存储) 的选项。





Visual Studio AWS 工具包中的亚马逊 S3 文件操作

如果您选择 Amazon S3 视图中的一个文件并打开相应的上下文（右键单击）菜单，则可对该文件执行各种操作。



Create Folder

使您能够在当前存储桶中创建文件夹。（等同于选择 Create Folder (创建文件夹) 链接。）

上传

使您能够上传文件或文件夹。（等同于选择 Upload File (上传文件) 或 Upload Folder (上传文件夹) 链接。）

Open

尝试在您的默认浏览器中打开所选文件。根据文件的类型和您的默认浏览器的功能，可能不会显示该文件。您的浏览器可能只是下载该文件。

下载

打开 Folder-Tree (文件夹-树) 对话框以使您能够下载所选文件。

公开化

将所选文件的权限设置为 Open/Download (打开/下载) 和 Everyone (所有人)。（等同于在上传设置对话框中选中公开所有内容复选框。）

删除

删除所选文件或文件夹。您也可以通过选择文件或文件夹并按 `Delete` 来删除它们。

更改存储类

将存储类设置为“标准”或“低冗余存储 (RRS)”。要查看当前存储类设置，请选择 Properties (属性)。

更改加密

使您能够对文件设置服务器端加密。要查看当前加密设置，请选择 Properties (属性)。

重命名

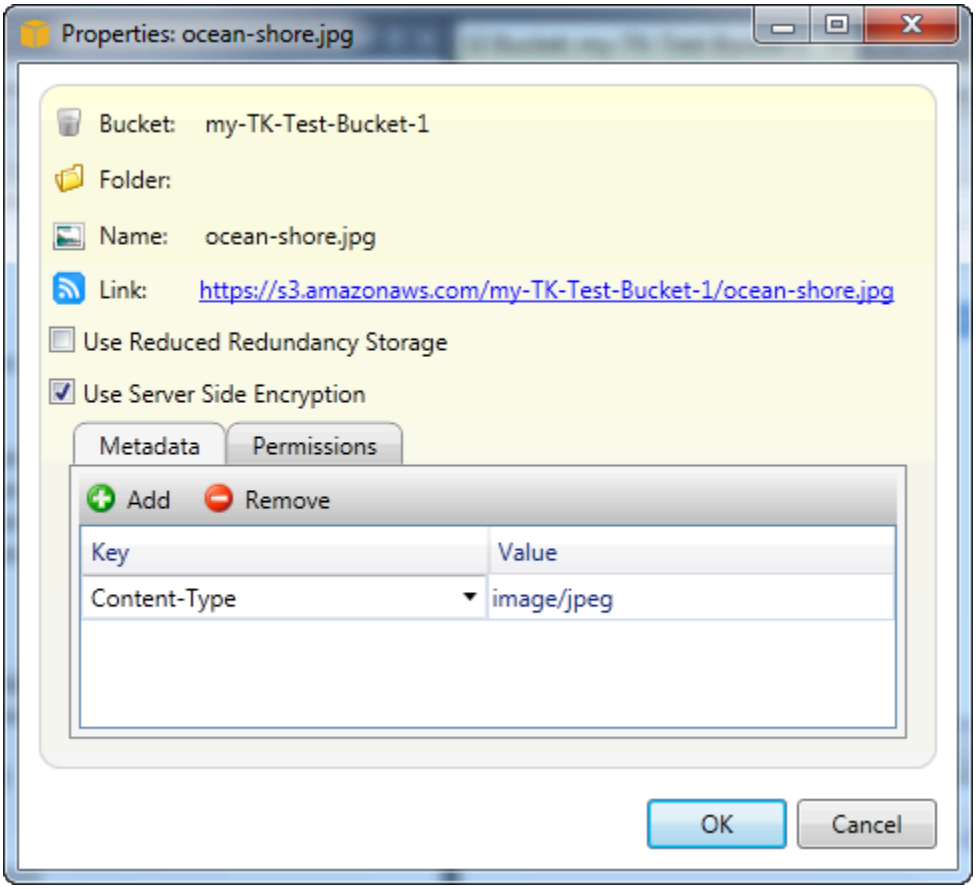
使您能够重命名文件。您无法重命名文件夹。

剪切 | 复制 | 粘贴

使您能够在文件夹或存储桶之间剪切、复制和粘贴文件或文件夹。

属性

显示一个对话框，使您可以在其中设置该文件的元数据和权限，以及在“低冗余存储 (RRS)”和“标准”之间切换文件的存储并为该文件设置服务器端加密。此对话框还显示一个指向该文件的 https 链接。如果选择此链接，Toolkit for Visual Studio 将在您的默认浏览器中打开该文件。如果您将该文件的权限设置为 Open/Download (打开/下载) 和 Everyone (所有人)，则其他人员均将能够通过此链接访问该文件。我们建议您创建和分发预签名 URLs链接，而不是分发此链接。



创建预签名 URL

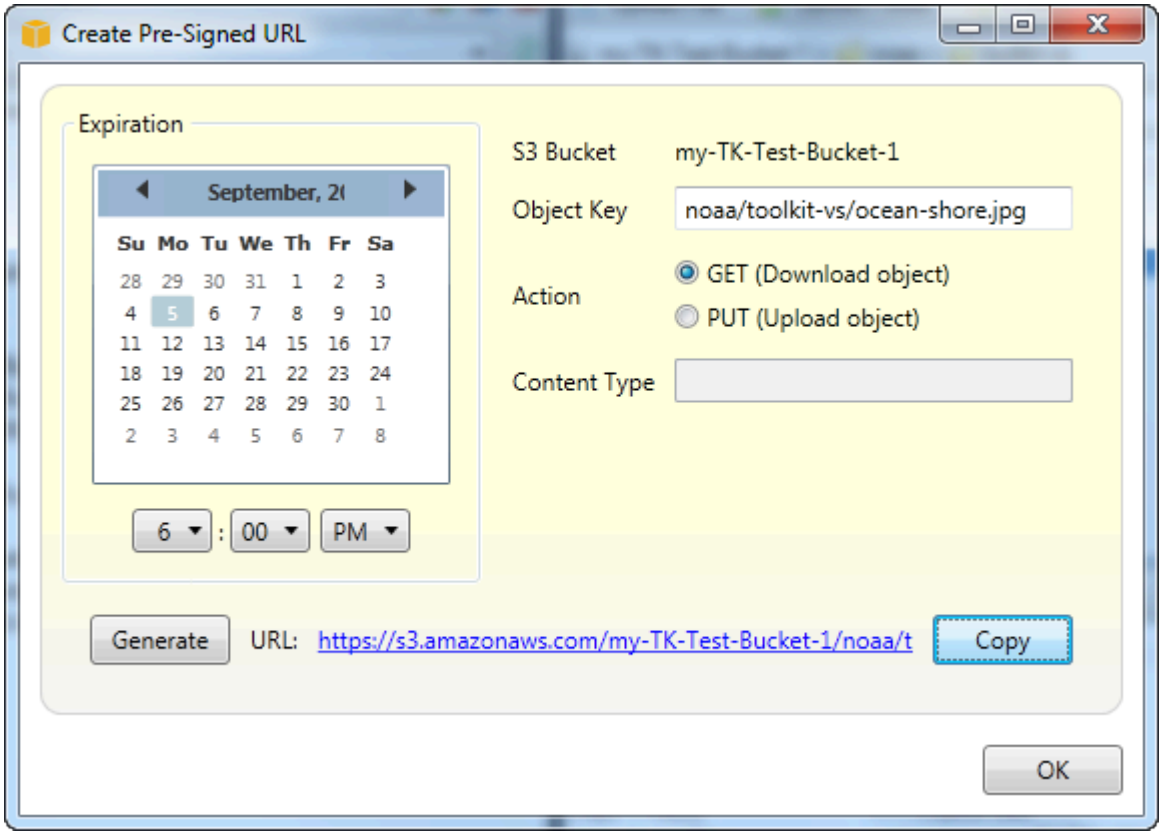
使您能够创建一个限时的预签名 URL，您可分发该 URL 以使其他人员能够访问已存储在 Amazon S3 上的内容。

如何创建预签名 URL

您可以为存储桶或存储桶中的文件创建预签名 URL。然后，其他人员可以使用此 URL 访问存储桶或文件。当您在创建此 URL 时指定的时间段过后，此 URL 将会过期。

创建预签名 URL

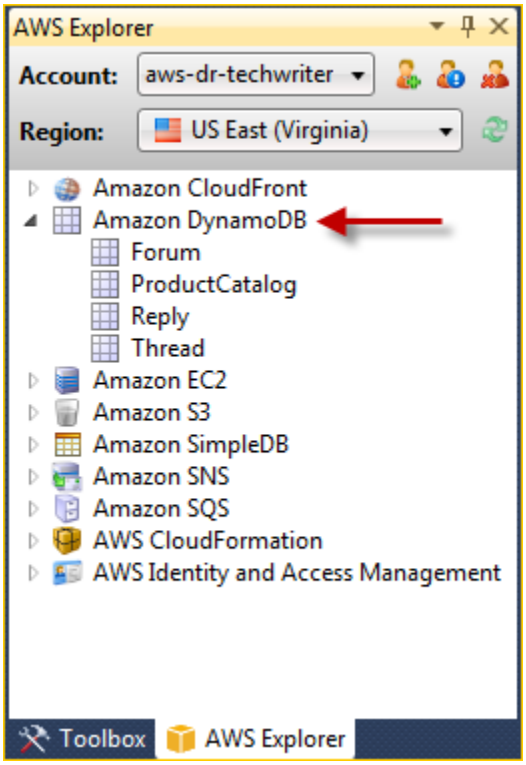
1. 在 Create Pre-Signed URL (创建预签名 URL) 对话框中，设置此 URL 的到期日期和时间。默认设置为从当前时间开始的一个小时。
2. 选择 Generate（生成）按钮。
3. 要将此 URL 复制到剪贴板，请选择 Copy（复制）。



使用 Explorer 中的 DynamoD AWS B

Amazon DynamoDB 是一项快速、高度可扩展、高度可用且经济实惠的非关系数据库服务。DynamoDB 消除了传统上对数据存储可扩展性的限制，同时保留了低延迟性和可预测的性能。Toolkit for Visual Studio 提供了在开发上下文中使用 DynamoDB 的功能。有关 DynamoDB 的更多信息，请参阅 Amazon Web Services 网站上的 DynamoDB。

在 Visual Studio 的 Toolkit for Visual Studio 中， AWS Explorer 会显示与活动表关联的所有 DynamoDB 表。 AWS 账户



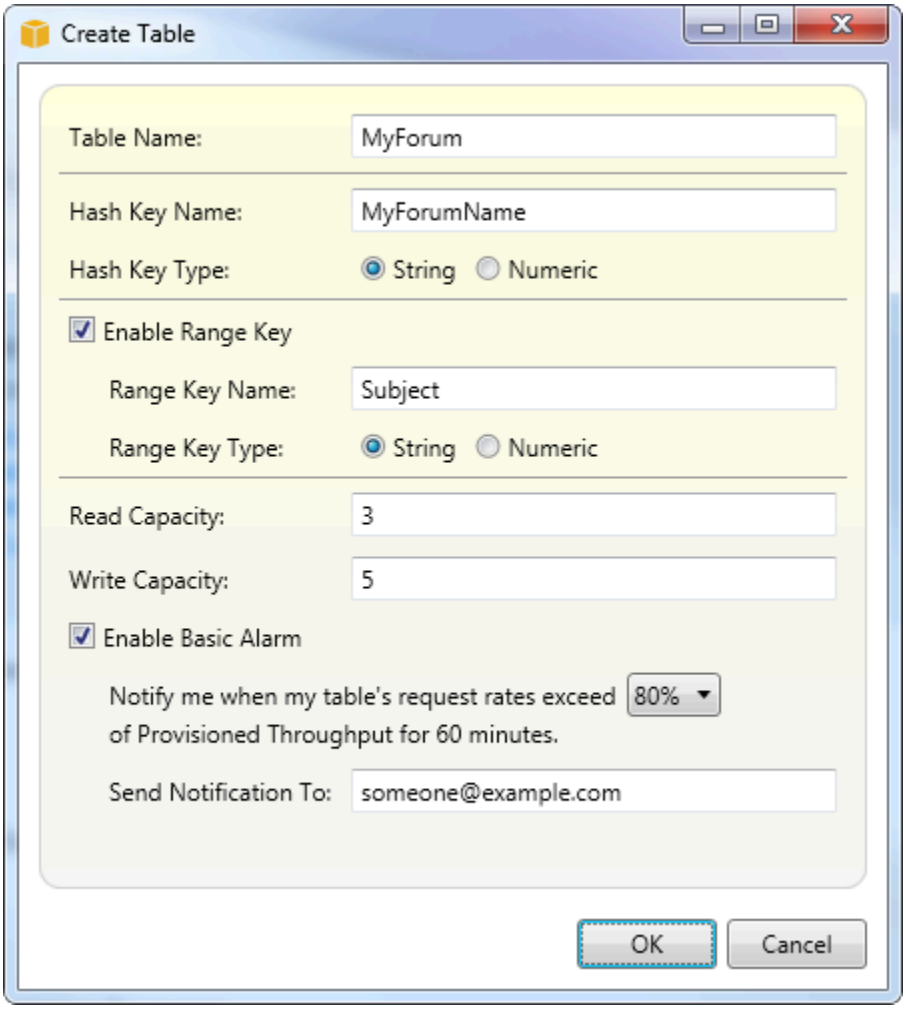
创建 DynamoDB 表

您可以使用 Toolkit for Visual Studio 创建 DynamoDB 表。

在 AWS 资源管理器中创建表

1. 在 AWS 资源管理器中，打开 Amazon DynamoD B 的上下文（右键单击）菜单，然后选择 “创建表”。
2. 在 Create Table (创建表) 向导的 Table Name (表名称) 中，键入表的名称。
3. 在哈希键名称字段中，键入主哈希键属性，然后从哈希键类型按钮中选择哈希键类型。DynamoDB 使用主键属性构建无序哈希索引，并使用范围主键属性构建可选的有序范围索引。有关主哈希键属性的更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的主键部分。
4. （可选）选择 Enable Range Key (启用范围键)。在 Range Key Name (范围键名称) 字段中，键入范围键属性，然后从 Range Key Type (范围键类型) 按钮中，选择范围键类型。
5. 在 Read Capacity (读取容量) 字段中，键入读取容量单位的数量。在 Write Capacity (写入容量) 字段中，键入写入容量单位的数量。您必须至少指定 3 个读取容量单位和 5 个写入容量单位。有关读取和写入容量单位的更多信息，请转到 DynamoDB 中预配置的吞吐量。
6. （可选）选择 Enable Basic Alarm (启用基本警报) 以在表的请求速率过快时提醒您。选择每 60 分钟的预配置吞吐量的百分比，必须超过此百分比才会发送提醒。在 Send Notifications To (将通知发送到) 中，键入电子邮件地址。

7. 单击 OK (确定) 以创建表。

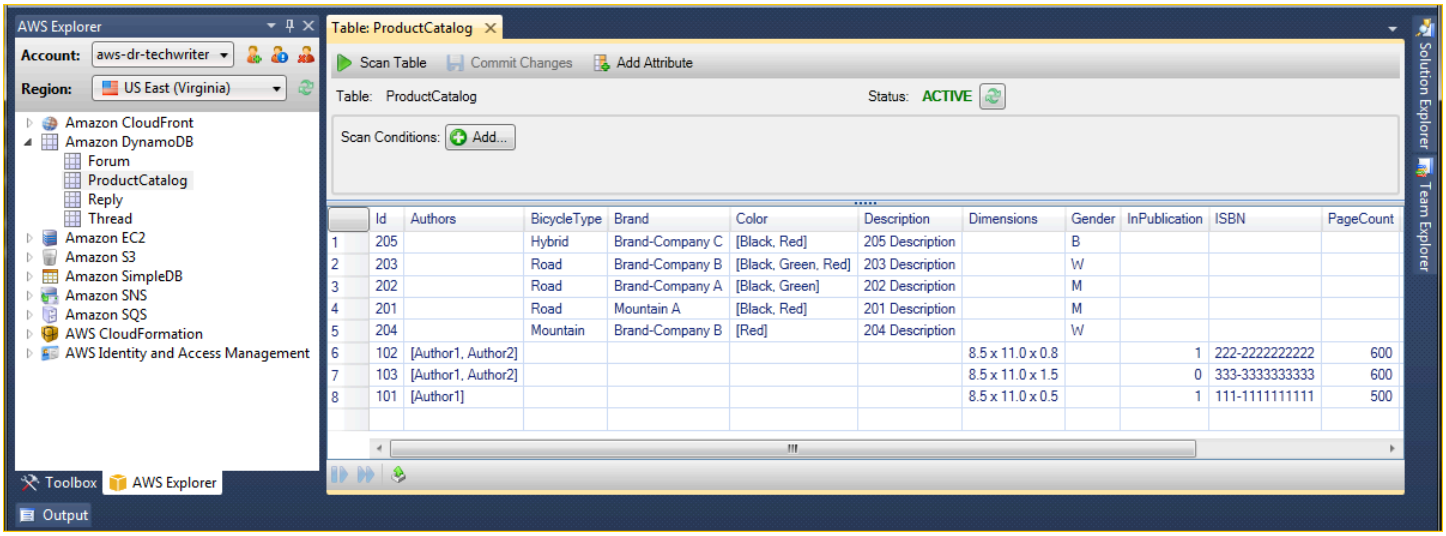


有关 DynamoDB 表的更多信息，请参阅数据模型概念 – 表、项目和属性。

以网格形式查看 DynamoDB 表

要打开其中一个 DynamoDB 表的网格视图，请在资源管理器 AWS 中双击与该表对应的子节点。从网格视图中，您可以查看存储在表中的项目、属性和值。每个行对应于表中的一个项目。表列与属性对应。表的每个单元格保存与该项目的该属性关联的值。

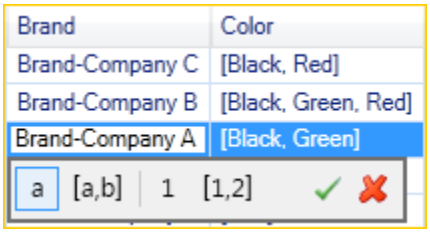
属性可以包含字符串或数字形式的值。某些属性包含由一系列 字符串或数字组成的值。系列值显示为用方括号括起的逗号分隔列表。



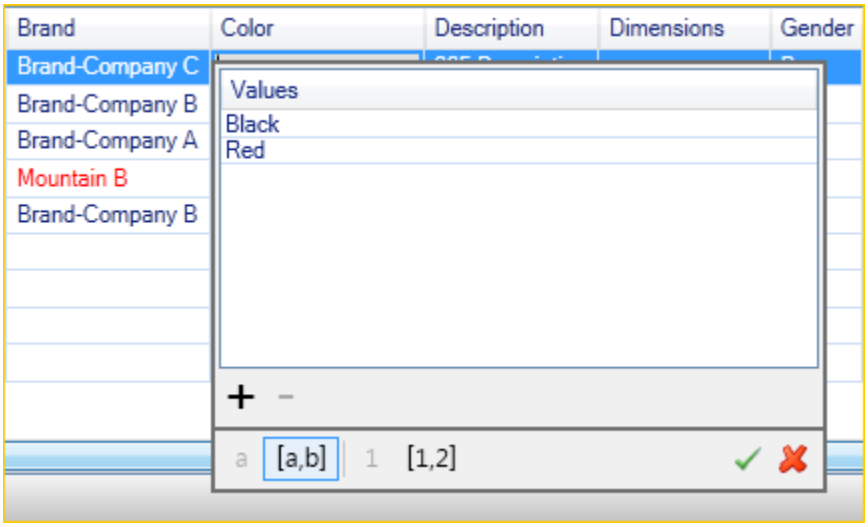
	Id	Authors	BicycleType	Brand	Color	Description	Dimensions	Gender	InPublication	ISBN	PageCount
1	205		Hybrid	Brand-Company C	[Black, Red]	205 Description		B			
2	203		Road	Brand-Company B	[Black, Green, Red]	203 Description		W			
3	202		Road	Brand-Company A	[Black, Green]	202 Description		M			
4	201		Road	Mountain A	[Black, Red]	201 Description		M			
5	204		Mountain	Brand-Company B	[Red]	204 Description		W			
6	102	[Author1, Author2]					8.5 x 11.0 x 0.8		1	222-2222222222	600
7	103	[Author1, Author2]					8.5 x 11.0 x 1.5		0	333-3333333333	600
8	101	[Author1]					8.5 x 11.0 x 0.5		1	111-1111111111	500

编辑和添加属性和值

通过双击单元格，您可以编辑项目对应属性的值。对于系列值属性，您还可以在该系列中添加或删除单个值。



除了更改属性的值之外，您还可以更改属性的值的格式（存在一些限制）。例如，任何数字值均可转换为字符串值。如果您有一个字符串值，内容为数字（如 125），那么单元格编辑器可让您将值的格式从字符串转换为数字。您还可以将单一值转换为系列值。但是，您通常无法将系列值转换为单一值；有一个例外情况，即当系列值实际上只包含一个元素时。

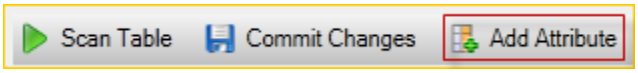


在编辑属性值后，请选择绿色的勾号以确认更改。如果要放弃更改，请选中红色 X。

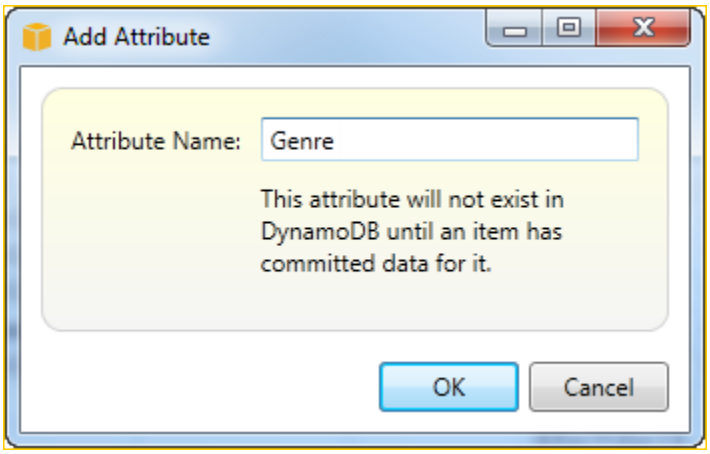
在您确认更改后，属性值将以红色显示。这表示属性已更新，但新值尚未写回到 DynamoDB 数据库。要将更改写回到 DynamoDB，请选择提交更改。要放弃更改，请选择 Scan Table (扫描表)，当 Toolkit 询问您是否要在扫描之前提交更改时，选择 No (否)。

添加属性

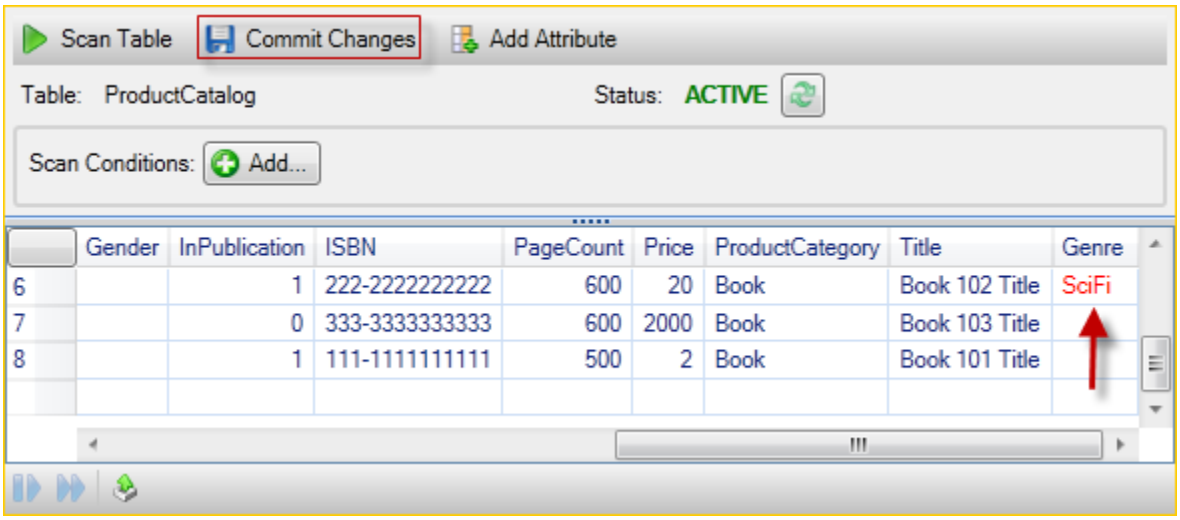
从网格视图中，您还可以将属性添加到表。要添加新属性，请选择 Add Attribute (添加属性)。



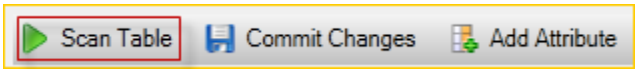
在 Add Attribute (添加属性) 对话框中，键入属性的名称，然后选择 OK (确定)。



要使新属性成为表的一部分，您必须至少为一个项目向新属性添加值，然后选择 Commit Changes (提交更改) 按钮。要放弃新属性，只需关闭表的网格视图，同时不选择 Commit Changes (提交更改) 即可。



扫描 DynamoDB 表

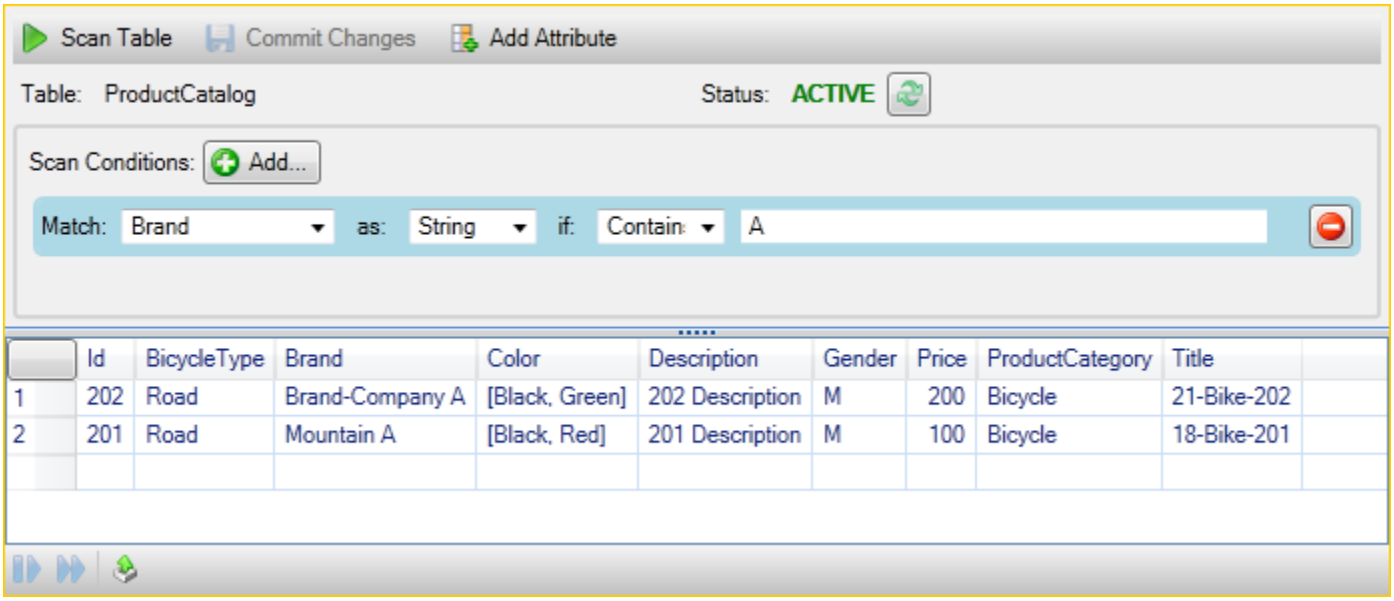


您可以从 Toolkit 对 DynamoDB 表执行扫描。在一次扫描中，您将定义一组条件，扫描将返回表中符合您的条件的所有项目。扫描是代价高昂的操作，应谨慎使用以避免干扰表中优先级更高的生产流量。有关使用扫描操作的更多信息，请参阅《Amazon DynamoDB 开发人员指南》。

使用资源管理器对 DynamoDB 表执行扫描 AWS

1. 在网格视图中，选择 scan conditions: add (扫描条件: 添加) 按钮。
2. 在扫描子句编辑器中，选择要与之匹配的属性，应如何解释属性的值（字符串、数字、系列值）、应如何匹配属性（例如“开头为”或“包含”）以及属性应匹配的文本值。
3. 根据需要为您的搜索添加更多扫描子句。扫描只会返回符合所有扫描子句中的条件的项目。当与字符串值匹配时，扫描将执行区分大小写的比较。
4. 在网格视图顶部的按钮栏上，选择 Scan Table (扫描表)。

要删除扫描子句，请选择每个子句右侧带白线的红色按钮。



要返回到包含所有项目的表的视图，请删除所有扫描子句并再次选择 Scan Table (扫描表)。

为扫描结果分页

视图底部有三个按钮。

前两个蓝色按钮为扫描结果提供分页。第一个按钮将显示另外一页的结果。第二个按钮将显示另外十页的结果。在此上下文中，一个页面等于 1 MB 的内容。

将扫描结果导出到 CSV

第三个按钮将结果从当前扫描导出到 CSV 文件。

将 AWS CodeCommit 与 Visual Studio Team Explorer 配合使用

您可以使用 AWS Identity and Access Management（IAM）用户账户创建 Git 凭证，并使用该凭证在 Team Explorer 中创建和克隆存储库。

AWS CodeCommit 的凭证类型

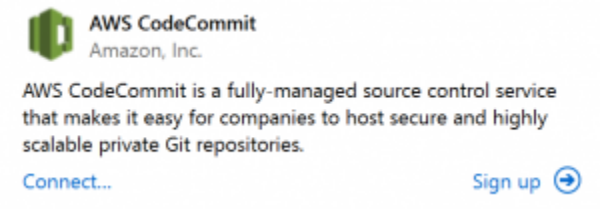
大多数 AWS Toolkit for Visual Studio 用户了解如何设置包含自己的访问密钥和秘密密钥的 AWS 凭证配置文件。这些凭证配置文件用于在 Toolkit for Visual Studio 中实现对服务 API 的调用，例如，在 AWS 各区服务浏览器中列出 Amazon S3 桶或者启动 Amazon EC2 实例。AWS CodeCommit 与 Team Explorer 的集成也使用这些凭证配置文件。但是，要使用 Git 本身，您需要额外的凭证，特别是用于 HTTPS 连接的 Git 凭证。有关这些凭证（用户名和密码）的信息，请参阅《AWS CodeCommit User Guide》中的 Setup for HTTPS Users Using Git Credentials。

您只能为 IAM 用户账户创建用于 AWS CodeCommit 的 Git 凭证。您不能为根账户创建这些凭证。您最多可以为服务创建两组这样的凭证，虽然您可以将一组凭证标记为不活动，但仍会计入您的两组凭证的限制中。请注意，您可以随时删除和重新创建凭证。当您在 Visual Studio 中使用 AWS CodeCommit 时，您的传统 AWS 凭证用于处理服务本身，例如，在您创建和列出存储库时。使用托管在 AWS CodeCommit 中的实际 Git 存储库时，您使用 Git 凭证。

作为对 AWS CodeCommit 支持的一部分，Toolkit for Visual Studio 会自动为您创建和管理这些 Git 凭证并将其关联到您的 AWS 凭证配置文件。您无需担心手头必须有一组合适的凭证在 Team Explorer 中执行 Git 操作。一旦您使用 AWS 凭证配置文件连接到 Team Explorer，关联的 Git 凭证将在您使用 Git remote 时自动使用。

将 连接到 AWS CodeCommit

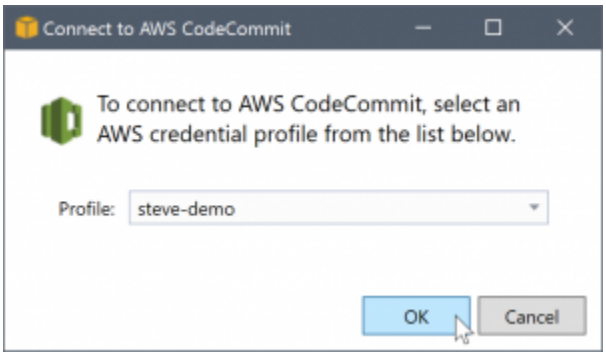
当您在 Visual Studio 2015 或更高版本中打开 Team Explorer 窗口时，您将在“Manage Connections”的“Hosted Service Providers”中看到 AWS CodeCommit 条目。



选择注册将在浏览器窗口中打开 Amazon Web Services 主页。选择连接时将发生的情况，取决于 Toolkit for Visual Studio 是否能够找到含有 AWS 访问密钥和秘密密钥的凭证配置文件以使其可以代表您调用 AWS。您可能使用在 IDE 中显示的新“开始使用”页面设置了凭证配置文件，这会使 Toolkit for Visual Studio 找不到任何存储在本地的凭证。或者，您可能已使用过 Toolkit for Visual Studio、AWS Tools for Windows PowerShell 或 AWS CLI，在这种情况下，您已经有了可供 Toolkit for Visual Studio 使用的 AWS 凭证配置文件。

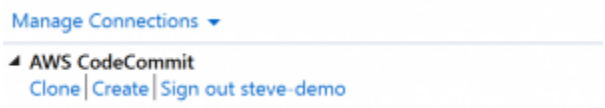
在您选择连接时，Toolkit for Visual Studio 启动流程来查找在连接中使用的凭证配置文件。如果 Toolkit for Visual Studio 找不到凭证配置文件，将打开一个对话框，邀请您输入您的 AWS 账户的访问密钥和秘密密钥。强烈建议您使用 IAM 用户账户而非使用您的根凭证。此外，如前文所述，只能为 IAM 用户创建您最终需要的 Git 凭证。提供访问密钥和秘密密钥并创建凭证配置文件之后，Team Explorer 和 AWS CodeCommit 之间的连接已可供使用。

如果 Toolkit for Visual Studio 找到多个 AWS 凭证配置文件，系统将提示您选择要在 Team Explorer 中使用的账户。



如果您只有一个凭证配置文件，Toolkit for Visual Studio 将绕过配置文件选择对话框，您将立即连接：

在 Team Explorer 与 AWS CodeCommit 之间通过凭证配置文件建立连接时，将关闭邀请对话框并显示连接面板。



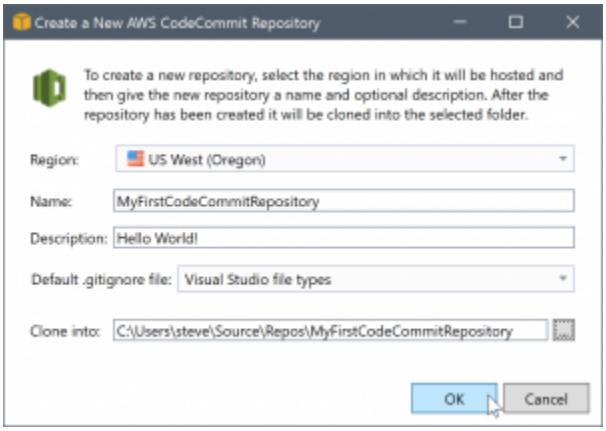
由于您没有本地克隆的存储库，面板只显示您可以执行的操作：Clone (克隆)、Create (创建) 和 Sign out (注销)。与其他提供程序一样，Team Explorer 中的 AWS CodeCommit 在任意时间只能绑定到一

个 AWS 凭证配置文件。要切换账户，您可以使用 Sign out (注销) 删除连接，以便您使用不同账户启动新的连接。

现在，您已建立了连接，您可以通过单击 Create (创建) 链接创建存储库。

创建存储库

当您单击创建链接时，将打开创建新 AWS CodeCommit 存储库对话框。



AWS CodeCommit 存储库按区域排列，因此您可以在 Region (区域) 中选择在哪个区域中托管存储库。列表中有 AWS CodeCommit 支持的所有区域。您为新存储库提供名称 (必需) 和说明 (可选)。

对话框的默认行为是使用存储库名称 (您输入的名称，同时更新文件夹位置) 作为新存储库文件夹位置的后缀。要使用不同的文件夹名称，请在完成输入存储库名称后编辑 Clone into (克隆到) 文件夹路径。

您还可以选择自动为存储库创建初始 `.gitignore` 文件。AWS Toolkit for Visual Studio 为 Visual Studio 文件类型提供了内置的默认值。您还可以选择没有文件，或者使用您希望在各个存储库中重用的自定义现有文件。只需在列表中选择 Use custom (使用自定义) 并导航到要使用的自定义文件。

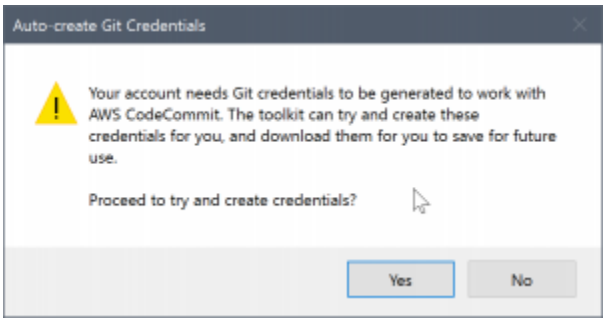
在您有了存储库的名称和位置之后，您可以单击 OK (确定) 并开始创建存储库。Toolkit for Visual Studio 请求服务创建存储库，然后本地克隆新的存储库，添加 .gitignore 文件的初始提交（如果您使用该文件）。此时您开始使用 Git remote，因此 Toolkit for Visual Studio 现在需要访问之前所述的 Git 凭证。

设置 Git 凭证

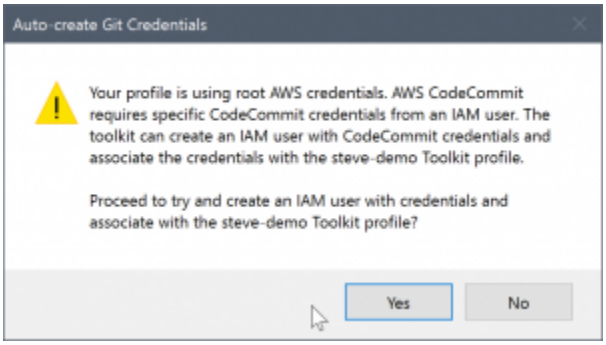
到目前为止，您已使用 AWS 访问密钥和秘密密钥请求服务创建您的存储库。现在，您需要使用 Git 本身执行实际克隆操作，而 Git 无法使用 AWS 访问密钥和秘密密钥。因此您需要向 Git 提供用户名和密码凭证以在 HTTPS 连接上使用 remote。

如设置 Git 凭证中所述，您将使用的 Git 凭证必须与 IAM 用户关联。您不能为根凭证生成它们。设置 AWS 凭证配置文件时，务必在其中包含 IAM 用户访问密钥和秘密密钥，而不是根密钥。Toolkit for Visual Studio 可以尝试为您设置用于 AWS CodeCommit 的 Git 凭证，并将该凭证与您之前用来在 Team Explorer 中进行连接的 AWS 凭证配置文件关联起来。

当您在创建新 AWS CodeCommit 存储库对话框中选择确定并成功创建了存储库时，Toolkit for Visual Studio 会检查在 Team Explorer 中连接的 AWS 凭证配置文件，以确定用于 AWS CodeCommit 的 Git 凭证是否存在并与配置文件本地关联。如果两个条件均满足，则 Toolkit for Visual Studio 指示 Team Explorer 开始在新存储库上的克隆操作。如果 Git 凭证在本地不可用，则 Toolkit for Visual Studio 会检查在 Team Explorer 的连接中使用的账户凭证的类型。如果该凭证用于 IAM 用户 (就像我们建议的那样)，则将显示以下消息。

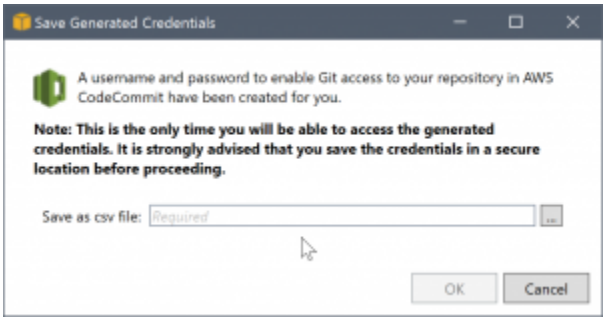


如果凭证是根凭证，将会改为显示以下消息。



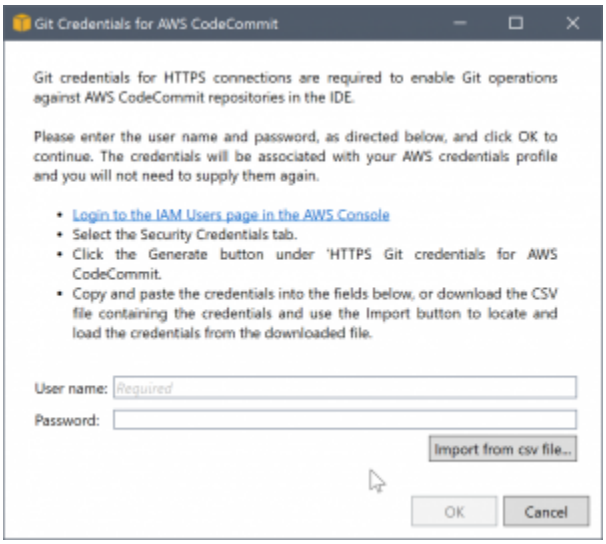
在这两种情况下，Toolkit for Visual Studio 都会提出尝试完成操作来为您创建必需的 Git 凭证。在第一种情况下，所有需要创建的是 IAM 用户的一组 Git 凭证。在使用根账户的情况下，Toolkit for Visual Studio 首先尝试创建一个 IAM 用户，然后继续为该新用户创建 Git 凭证。如果 Toolkit for Visual Studio 需要创建新用户，则会将 AWS CodeCommit 高级用户托管策略应用到该新用户账户。此策略只允许访问 AWS CodeCommit 并允许使用 AWS CodeCommit 执行除了存储库删除之外的所有操作。

在您创建凭证时，您只能查看一次凭证。因此，Toolkit for Visual Studio 提示您将新创建的凭证另存为 `.csv` 文件，然后再继续。



这是我们强烈建议的操作，并确保将它们保存到安全位置！

在有些情况下，Toolkit for Visual Studio 可能无法自动创建凭证。例如，您创建的用于 AWS CodeCommit 的 Git 凭证组数量已达上限（两个），或者您可能没有足够的编程权限让 Toolkit for Visual Studio 为您完成工作（如果您以 IAM 用户身份登录）。在这些情况下，您可以登录 AWS 管理控制台管理凭证或从管理员处获取这些凭证。然后，您可以在 Toolkit for Visual Studio 显示的用于 AWS CodeCommit 的 Git 凭证对话框中输入它们。

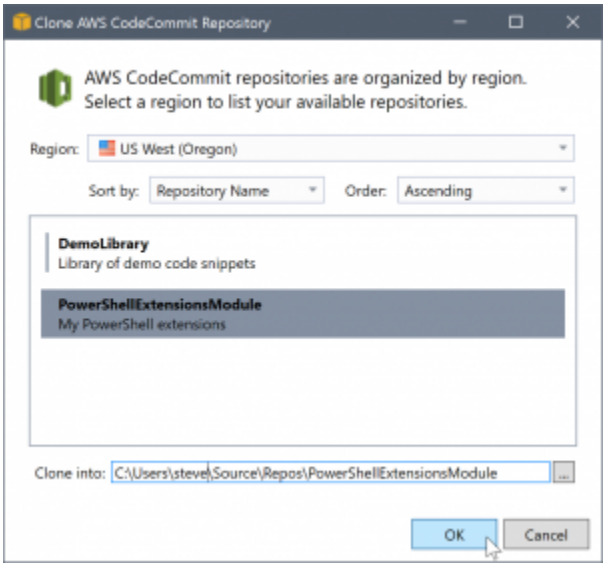


现在 Git 的凭证已可用，新存储库的克隆操作将继续 (请查看 Team Explorer 中操作的进度指示)。如果您选择应用默认 `.gitignore` 文件，则会使用注释“Initial Commit”将其提交到存储库。

这是在 Team Explorer 中设置凭证和创建存储库所需的全部操作。一旦准备好所需的凭证，以后您在创建新存储库时就只会看到创建新 AWS CodeCommit 存储库对话框本身。

克隆存储库

要克隆现有存储库，请返回到 Team Explorer 中 AWS CodeCommit 的连接面板。单击克隆链接以打开克隆 AWS CodeCommit 存储库对话框，然后选择要克隆的存储库以及您要在磁盘上放置该存储库的位置。



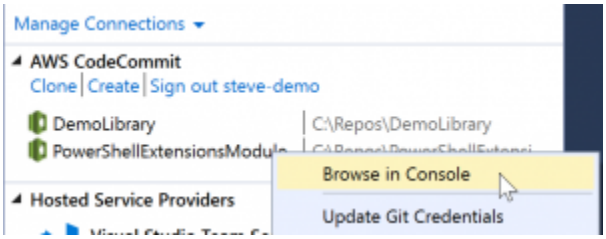
选择区域后，Toolkit for Visual Studio 将查询服务以发现该区域中可用的存储库，并将其显示在对话框的中央列表部分。还将显示每个存储库的名称和可选说明。您可以重新排序列表，将其按存储库名称或上次修改日期，以升序或降序排序。

在您选择存储库之后，您可以选择要克隆到的位置。这会默认为在 Team Explorer 中其他插件使用的相同存储库位置，不过您可以浏览或输入任何其他位置。默认情况下，存储库名称作为后缀添加到选定的路径。但是，如果您希望使用特定路径，只需在选择文件夹之后编辑文本框。单击 OK (确定) 时文本框中的任何文本将成为文件夹，您可在其中查找克隆的存储库。

选择了存储库和文件夹位置之后，您可以单击 OK (确定) 以继续克隆操作。就像创建存储库一样，您可以在 Team Explorer 中查看报告的克隆操作进度。

使用存储库

当您克隆或创建存储库时，请注意在 Team Explorer 中连接面板的操作链接下列出的本地存储库的连接。这些条目为您提供了一种简便的方法访问存储库以浏览内容。只需右键单击存储库并选择 Browse in Console (在控制台中浏览)。



您也可以使用 Update Git Credentials (更新 Git 凭证) 更新与凭证配置文件关联的已存储 Git 凭证。如果您已轮换凭证，这非常有用。该命令将打开用于 AWS CodeCommit 的 Git 凭证对话框，您可以在其中输入或导入新凭证。

存储库上的 Git 操作均按您的预期工作。您可以发出本地提交，并在准备好共享时，使用 Team Explorer 中的“Sync”选项。由于 Git 凭证已存储在本地并与我们的已连接 AWS 凭证配置文件关联，系统不会提示我们再次为 AWS CodeCommit remote 操作提供凭证。

在 Visual Studio 中使用 CodeArtifact

AWS CodeArtifact 是一项完全托管的构件存储库服务，可让组织轻松安全地存储和共享用于应用程序开发的软件包。您可以将 CodeArtifact 与常用的构建工具和程序包管理器配合使用，例如 NuGet 和.NET Core CLI 以及 Visual Studio。您还可以将 CodeArtifact 配置为从外部公共存储库（例如 NuGet.org）提取程序包。

在 CodeArtifact 中，您的程序包存储在存储库中，而后者存储在域中。AWS Toolkit for Visual Studio 简化了 Visual Studio 使用 CodeArtifact 存储库时的配置，从而可以轻松地在 Visual Studio 中直接使用 CodeArtifact 和 Nuget.org 中的程序包。

将您的 CodeArtifact 存储库添加为 NuGet 程序包来源

要使用来自 CodeArtifact 的程序包，您需要在 Visual Studio 的 NuGet Package Manager 中将存储库作为可打包来源添加

将您的存储库添加为程序包来源

1. 在 AWS 各区服务浏览器中，导航到 AWS CodeArtifact 节点中的存储库。
2. 打开要添加的存储库的上下文（右键单击）菜单，然后选择复制 NuGet 来源端点。
3. 在工具 > 选项菜单中，导航到 NuGet Package Manager 节点下的程序包来源。
4. 在程序包来源中，选择加号（+），编辑名称，然后将之前复制的 NuGet 来源端点 URL 粘贴到来源字段中。
5. 选中新添加的程序包来源旁的复选框以启用该来源。

> **Note**
>
> 我们建议在 CodeArtifact 中添加指向 NuGet.org 的外部连接，并在 Visual Studio 中禁用 nuget.org 程序包来源。使用外部连接时，从 NuGet.org 提取的所有依赖项都存储在 CodeArtifact 中。如果 NuGet.org 因任何原因出现故障，您需要的程序包仍然可用。有

关外部连接的更多信息，请参阅《AWS CodeArtifact User Guide》中的 Add an external connection。

6. 选择确定以关闭菜单。

有关将 CodeArtifact 与 Visual Studio 配合使用的更多信息，请参阅《AWS CodeArtifact User Guide》中的 Use CodeArtifact with Visual Studio。

从 AWS 各区服务浏览器使用 Amazon RDS

Amazon Relational Database Service（Amazon RDS）服务使您可以在云中预置和管理 SQL 关系数据库系统。Amazon RDS 支持以下三种类型的数据库系统：

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

有关更多信息，请参阅 Amazon RDS 用户指南。

此处讨论的很多功能也可通过适用于 Amazon RDS 的 AWS 管理控制台获得。

主题

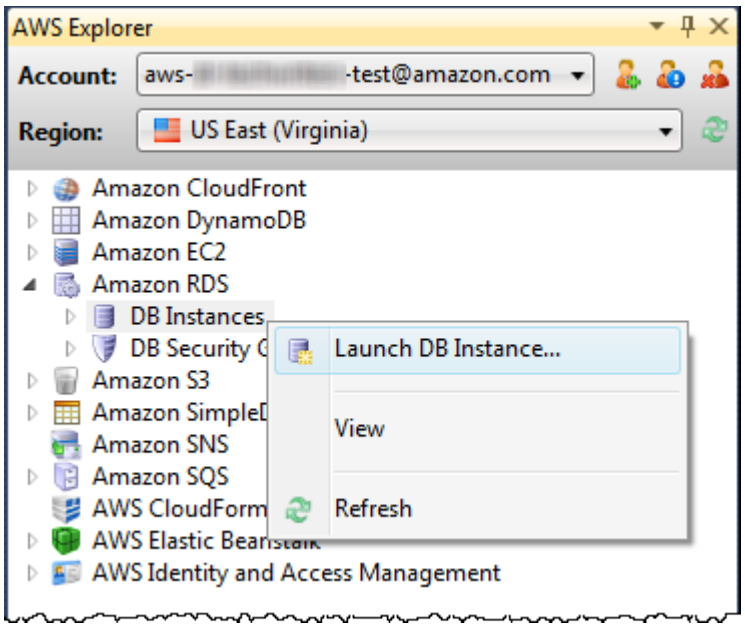
- 启动 Amazon RDS 数据库实例
- 在 RDS 实例中创建 Microsoft SQL Server 数据库
- Amazon RDS 安全组

启动 Amazon RDS 数据库实例

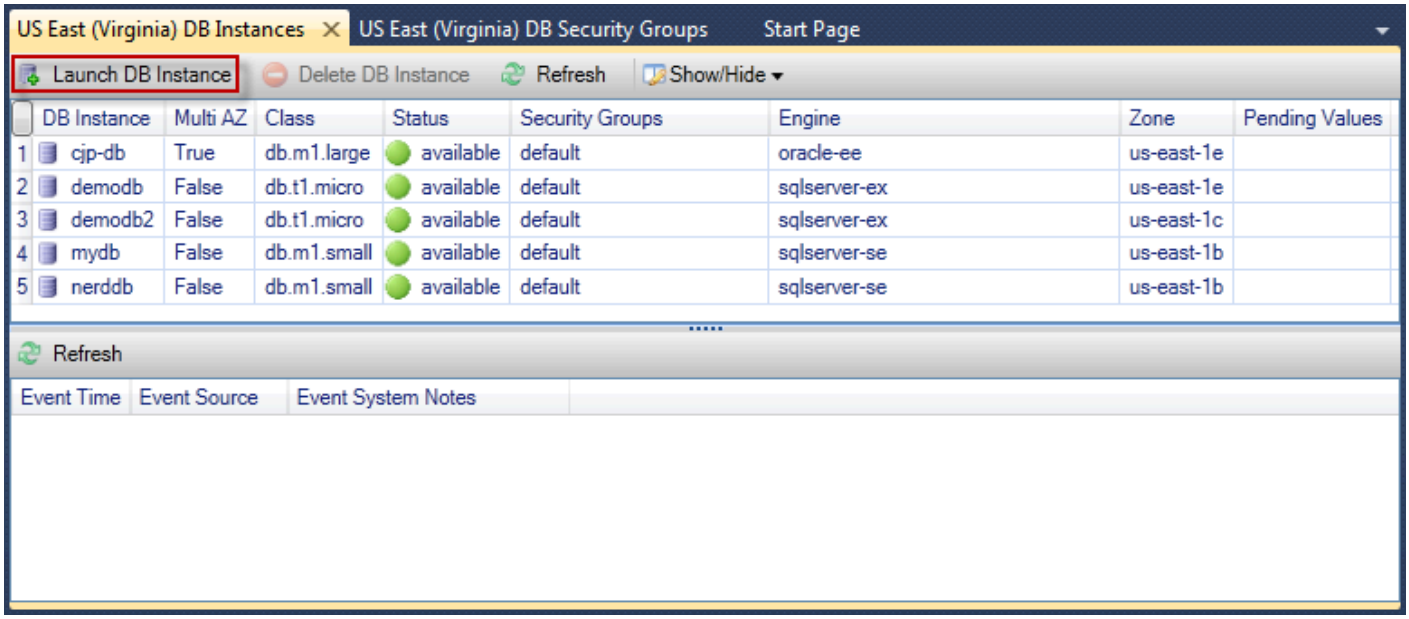
使用 AWS Explorer，您可以启动 Amazon RDS 支持的任何数据库引擎的实例。以下演练显示启动 Microsoft SQL Server Standard Edition 的实例的用户体验，但此用户体验对于所有支持的引擎均类似。

启动 Amazon RDS 实例

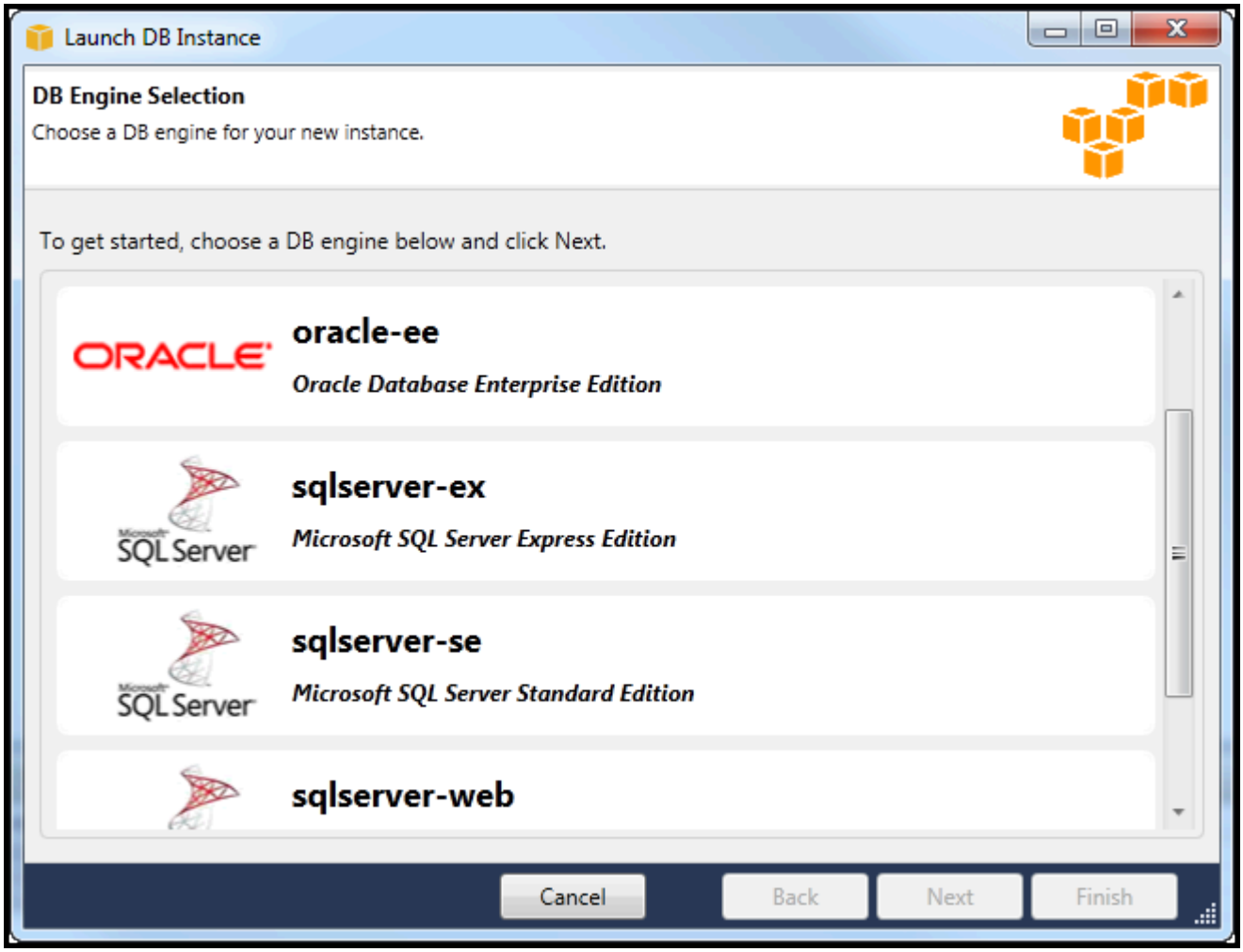
1. 在 AWS 资源管理器中，打开 Amazon RDS 节点的上下文（右键单击）菜单，然后选择 “启动数据库实例”。



或者，在 DB Instances (数据库实例) 选项卡上，选择 Launch DB Instance (启动数据库实例)。



2. 在 DB Engine Selection (数据库引擎选择) 对话框中，选择要启动的数据库引擎的类型。在此演练中，选择 Microsoft SQL Server Standard Edition (sqlserver-se)，然后选择 Next (下一步)。



3. 在 DB Engine Instance Options (数据库引擎实例选项) 对话框中，选择配置选项。

   在 DB Engine Instance Options and Class (数据库引擎实例选项和类) 部分中，您可以指定以下设置。

   许可模式

引擎类型	许可证
Microsoft SQL Server	附带许可
MySql	general-public-license
Oracle	bring-your-own-license

许可证模型因数据库引擎的类型而异。引擎类型许可微软 SQL Server 许可证包含甲骨文 MySql general-public-license bring-your-own-license

DB Instance Version (数据库实例版本)

选择您要使用的数据库引擎的版本。如果仅支持一个版本，则将为您选择该版本。

数据库实例类

选择数据库引擎的实例类。实例类的定价有所差异。有关更多信息，请参阅 Amazon RDS 定价。

Perform a multi AZ deployment (执行多可用区部署)

选择此选项可创建多可用区部署，以增强数据持久性和可用性。Amazon RDS 可在不同的可用区中预置和维护数据库的备用副本，以确保发生计划内或计划外停机时自动执行失效转移。有关多可用区部署的定价信息，请参阅 Amazon RDS 详细信息页面的定价部分。Microsoft SQL Server 不支持此选项。

Upgrade minor versions automatically (自动升级次要版本)

选择此选项可 AWS 自动为您在 RDS 实例上执行次要版本更新。

在 RDS Database Instance (RDS 数据库实例) 部分中，您可以指定以下设置。

分配的存储空间

Engine	最小 (GB)	最大 (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

最小和最大的分配存储取决于数据库引擎的类型。引擎最小值 (GB) 最大值 (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

为数据库实例指定名称。此名称不区分大小写。它将在 AWS 资源管理器中以小写形式显示。

Master User Name (主用户名

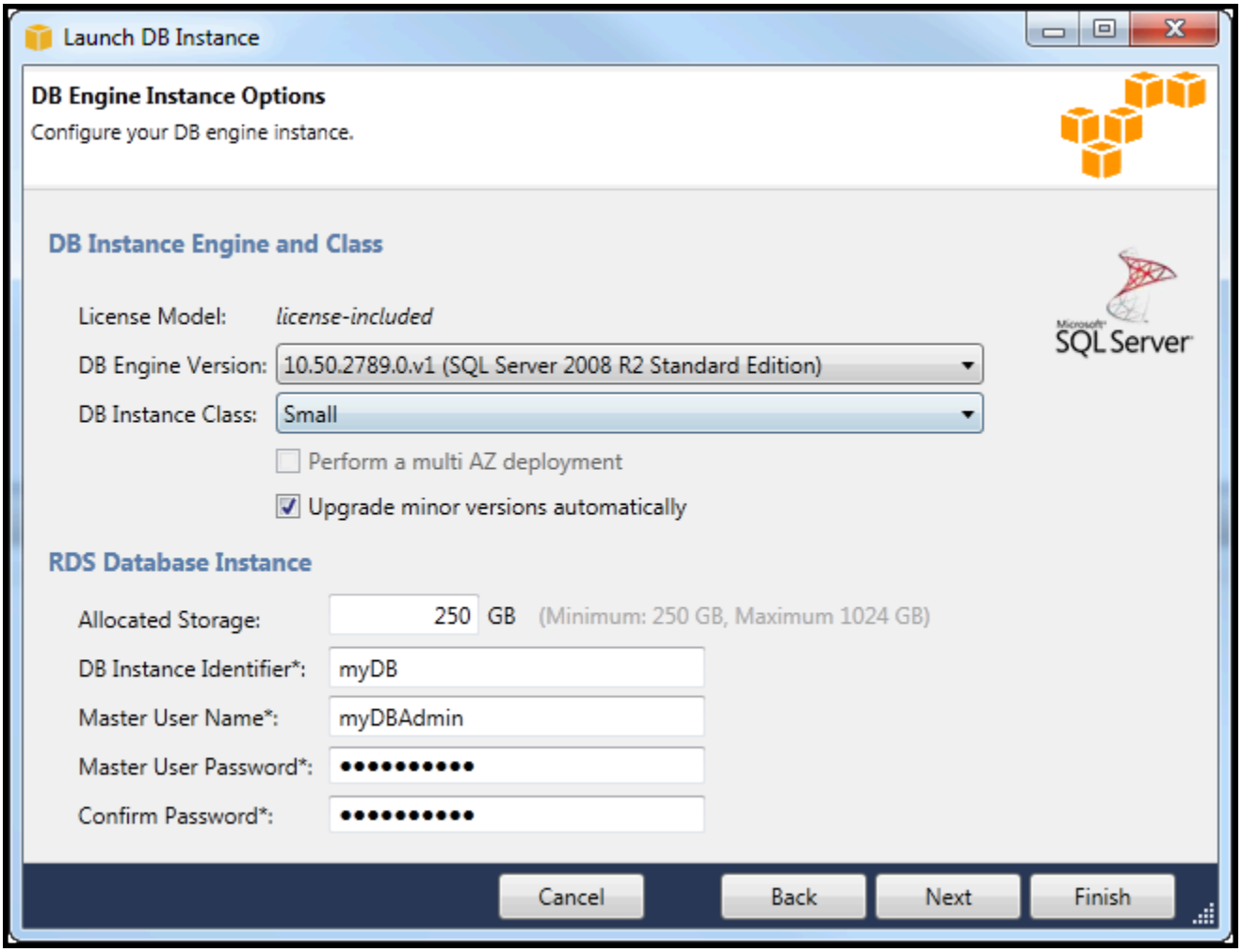
键入数据库实例的管理员的姓名。

主用户密码

键入数据库实例的管理员的密码。

确认密码

再次键入密码以验证其是否正确。



1. 在 Additional Options (其他选项) 对话框中，您可以指定以下设置。

   Database Port

   这是实例将用于在网络上进行通信的 TCP 端口。如果您的计算机通过防火墙访问 Internet，请将此值设置为您的防火墙允许流量通过的端口。

   可用区

   如果您希望在您区域的某个特定可用区中启动实例，请使用此选项。您指定的数据库实例可能不会在某个给定区域的所有可用区中可用。

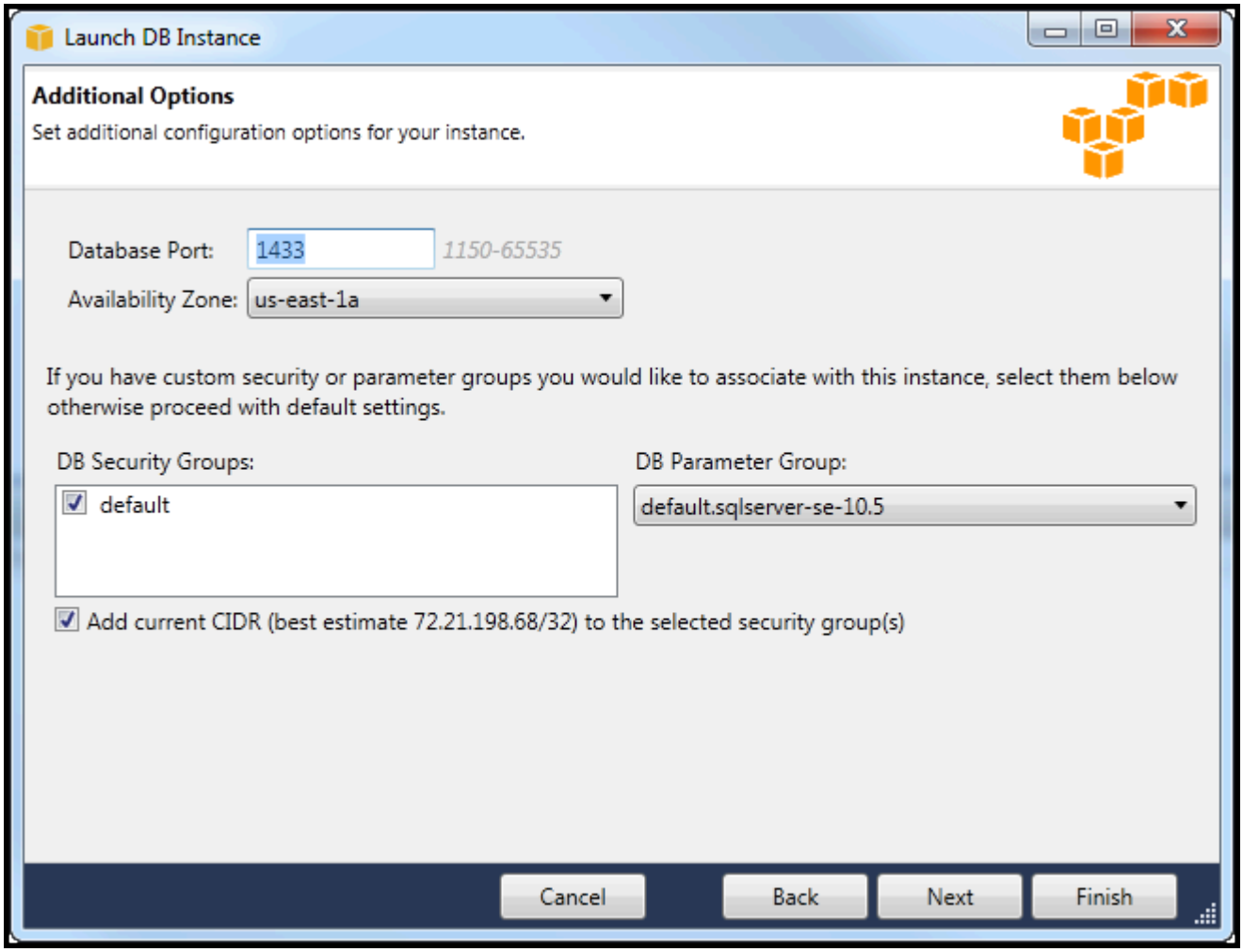
   RDS Security Group (RDS 安全组)

   选择要与您的实例关联的 RDS 安全组。RDS 安全组指定允许访问您的实例的 IP 地址、Amazon EC2 实例和 AWS 账户 实例。有关 RDS 安全组的更多信息，请参阅 Amazon RDS 安全组。Toolkit for Visual Studio 尝试确定您的当前 IP 地址并提供用于将此地址添加到与实例关联的安全组的选项。但是，如果您的计算机通过防火墙访问 Internet，则 Toolkit 为您的计算机生成的 IP 地址可能不准确。若要确定要使用的 IP 地址，请咨询您的系统管理员。

   数据库参数组

   （可选）从此下拉列表中，选择要与您的实例关联的数据库参数组。利用数据库参数组，您可以更改实例的默认配置。有关更多信息，请转到 Amazon Relational Database Service 用户指南和本文章。

   您在此对话框上指定设置之后，选择 Next (下一步)。

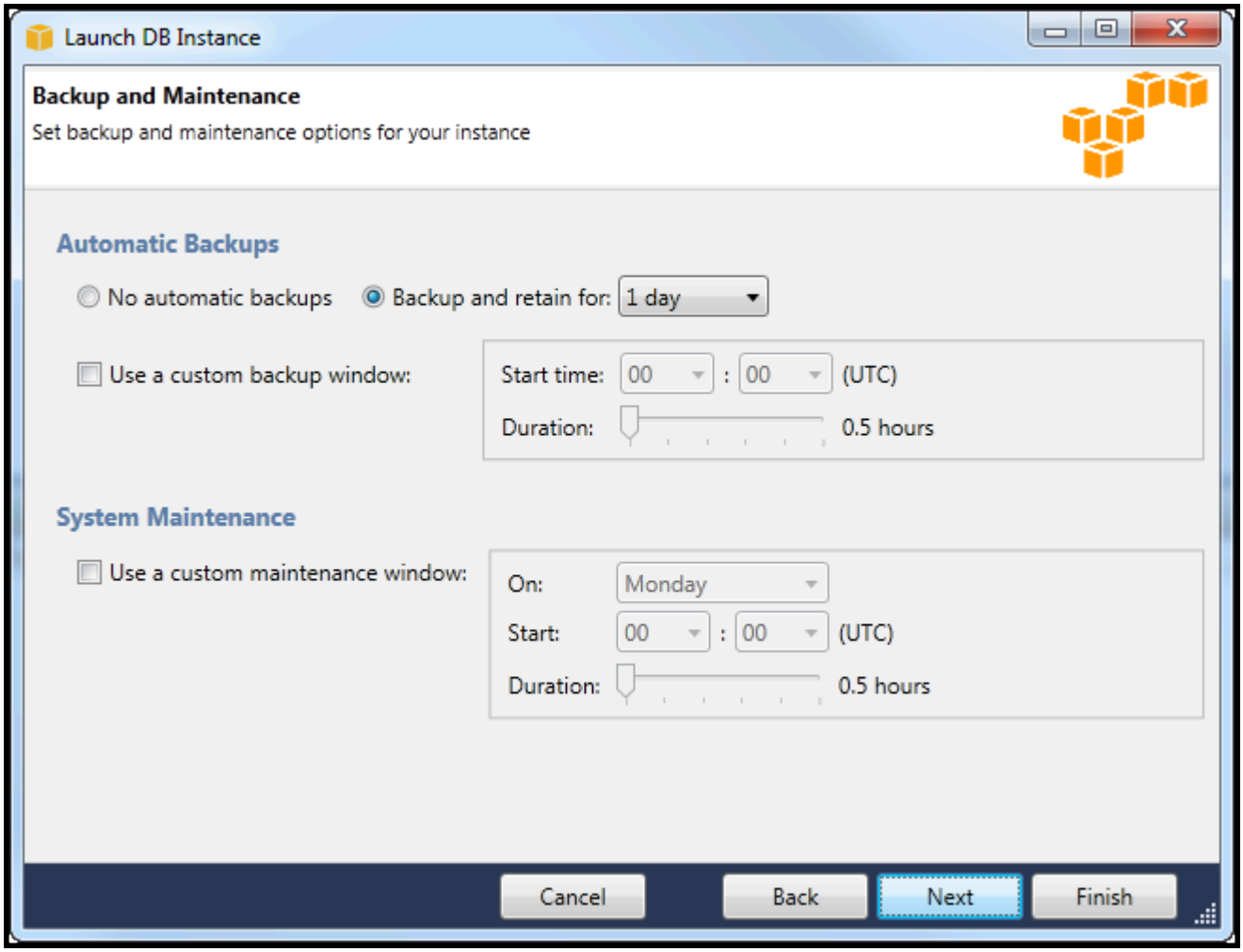


2. 利用备份和维护对话框，您可以指定 Amazon RDS 是否应对实例进行备份，如果备份，还可指定备份应保留多长时间。您还可以指定备份应发生的时间范围。

   此对话框还使您能够指定是否希望 Amazon RDS 在实例上执行系统维护。维护包括常规补丁和次要版本升级。

   您为系统维护指定的时间范围不得与为备份指定的时间范围重叠。

   选择下一步。



3. 利用向导中的最后一个对话框，您可以检查实例的设置。如果需要修改设置，请使用 Back (返回) 按钮。如果所有设置均正确，请选择 Launch (启动)。

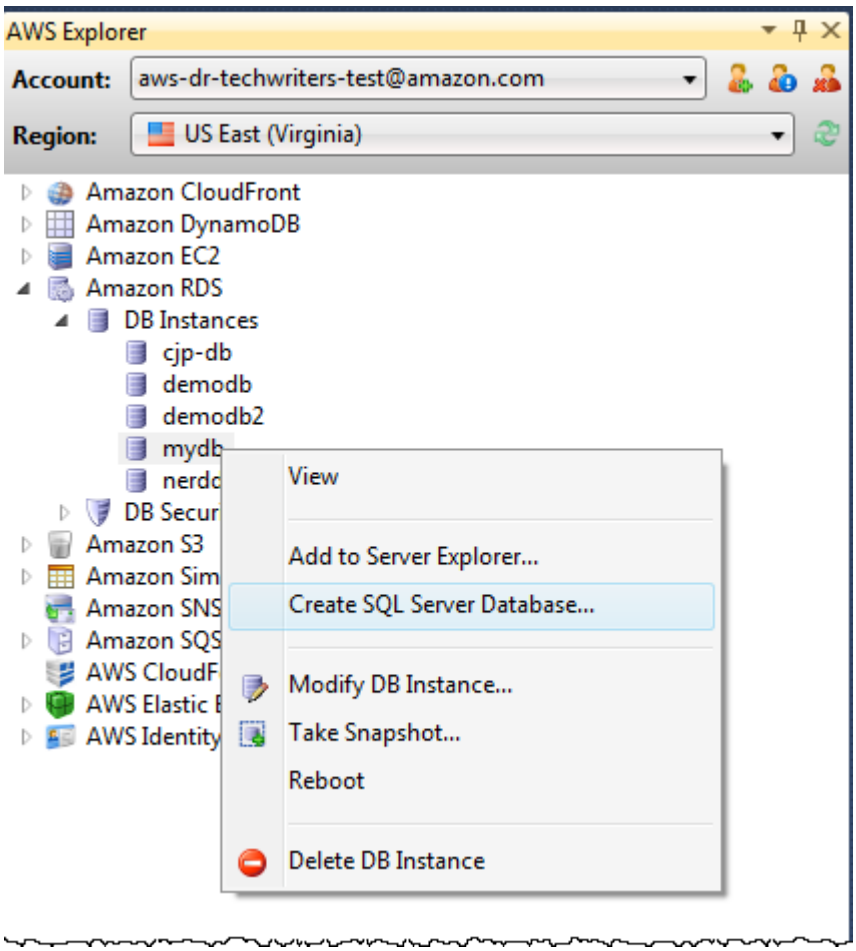
在 RDS 实例中创建 Microsoft SQL Server 数据库

Microsoft SQL Server 的设计方式是，在启动 Amazon RDS 实例后，您需要在 RDS 实例中创建 SQL Server 数据库。

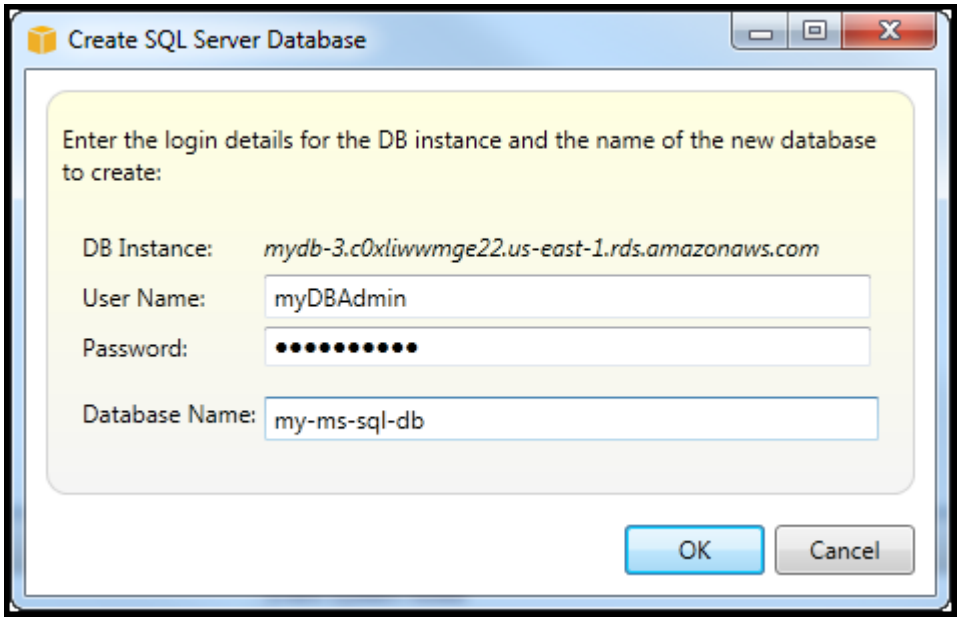
有关如何创建 Amazon RDS 实例的信息，请参阅启动 Amazon RDS 数据库实例。

创建 Microsoft SQL Server 数据库

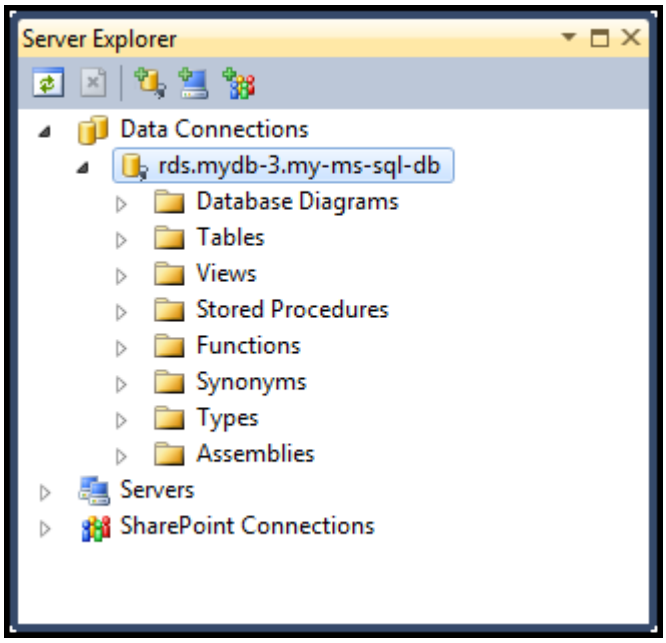
1. 在 AWS 资源管理器中，打开与你的 Microsoft SQL Server 的 RDS 实例对应的节点的上下文（右键单击）菜单，然后选择创建 SQL Server 数据库。



2. 在 Create SQL Server Database (创建 SQL Server 数据库) 对话框中，键入您在创建 RDS 实例时指定的密码，键入 Microsoft SQL Server 数据库的名称，然后选择 OK (确定)。



3. Toolkit for Visual Studio 创建 Microsoft SQL Server 数据库并将其添加到 Visual Studio 服务器浏览器。



Amazon RDS 安全组

Amazon RDS 安全组使您能够管理对 Amazon RDS 实例的网络访问。利用安全组，您可以使用 CIDR 表示法指定一系列 IP 地址，只有来自这些地址的网络流量才能被您的 Amazon RDS 实例接受。

尽管 Amazon RDS 安全组与 Amazon EC2 安全组的工作方式相似，但仍有一些差别。您可将 EC2 安全组添加到 RDS 安全组。作为 EC2 安全组成员的任何 EC2 实例随后都能访问作为 RDS 安全组成员的 RDS 实例。

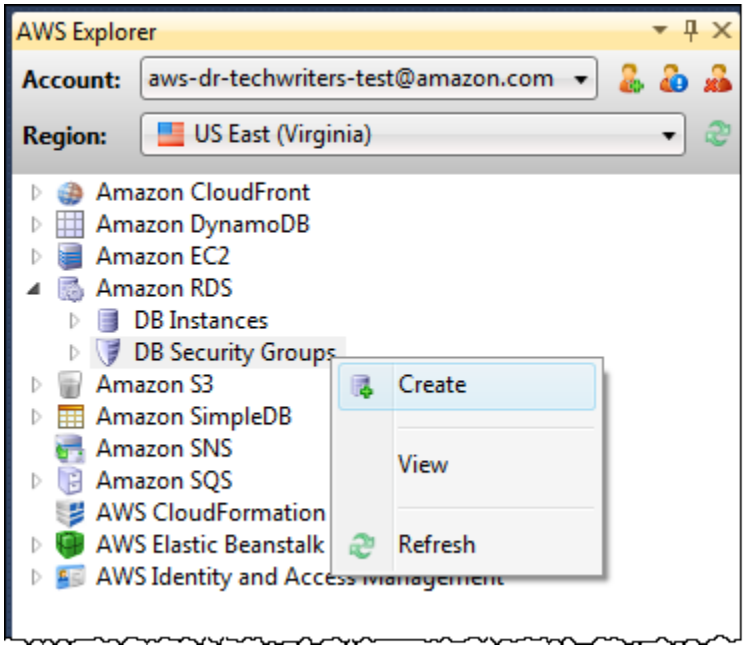
有关 Amazon RDS 安全组的更多信息，请参阅 RDS 安全组。有关 Amazon EC2 安全组的更多信息，请参阅 EC2 用户指南。

创建 Amazon RDS 安全组

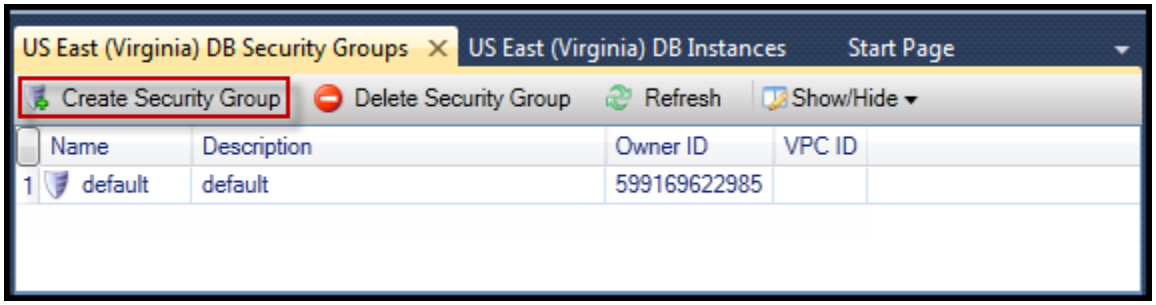
您可以使用 Toolkit for Visual Studio 创建 RDS 安全组。如果您使用 AWS 工具包启动 RDS 实例，则该向导将允许您指定用于实例的 RDS 安全组。您可在启动该向导前使用以下步骤创建该安全组。

创建 Amazon RDS 安全组

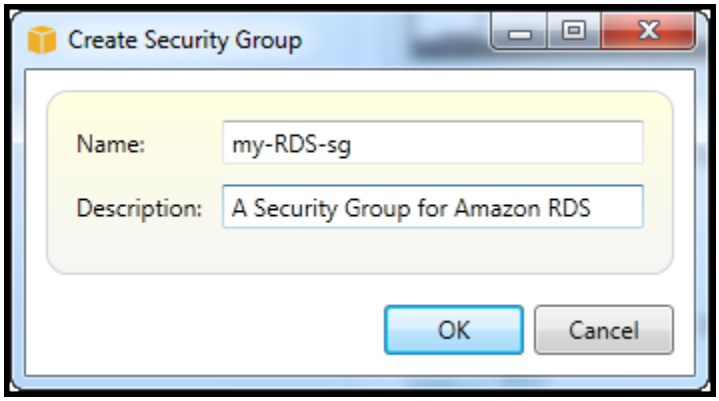
1. 在 AWS 资源管理器中，展开 Amazon RDS 节点，打开数据库安全组子节点的上下文（右键单击）菜单，然后选择创建。



或者，在 Security Groups (安全组) 选项卡上，选择 Create Security Group (创建安全组)。如果此选项卡未显示，则打开 DB Security Groups (数据库安全组) 子节点的上下文（右键单击）菜单，然后选择 View (查看)。



2. 在 Create Security Group (创建安全组) 对话框中，键入该安全组的名称和描述，然后选择 OK (确定)。



设置 Amazon RDS 安全组的访问权限

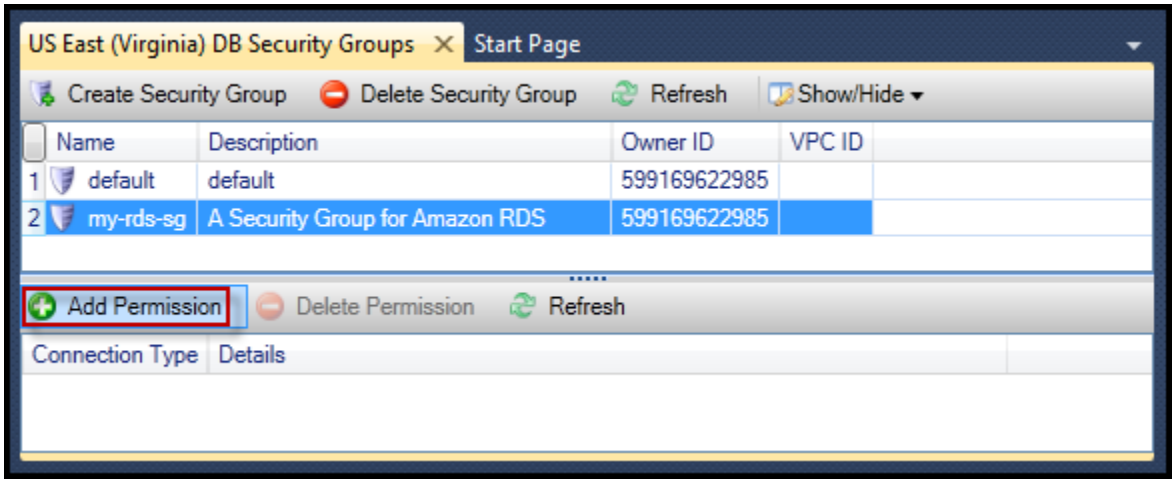
默认情况下，新的 Amazon RDS 安全组不提供网络访问权限。要启用对使用该安全组的 Amazon RDS 实例的访问，请使用以下步骤设置其访问权限。

设置 Amazon RDS 安全组的访问权限

1. 在 Security Groups (安全组) 选项卡上，从列表视图中选择该安全组。如果您的安全组未显示在列表中，请选择 Refresh (刷新)。如果您的安全组仍未出现在列表中，请确认您查看的 AWS 区域列表是否正确。 AWS 工具包中的 “安全组” 选项卡是特定于区域的。

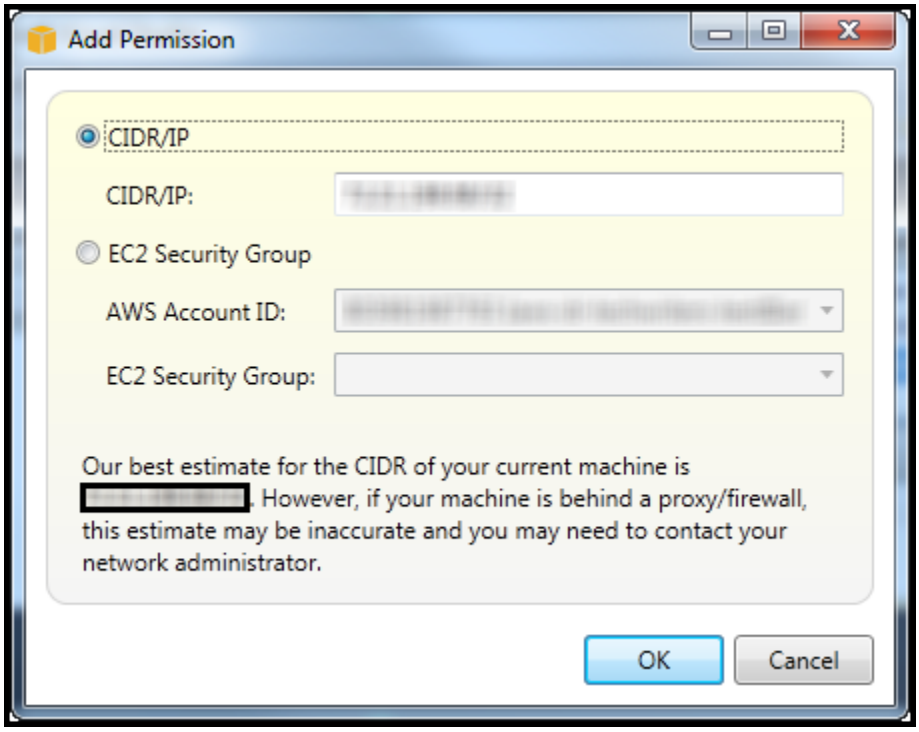
   如果未显示任何安全组选项卡，请在 AWS 资源管理器中打开数据库安全组子节点的上下文（右键单击）菜单，然后选择查看。

2. 选择 Add Permission。



安全组选项卡上的添加权限按钮

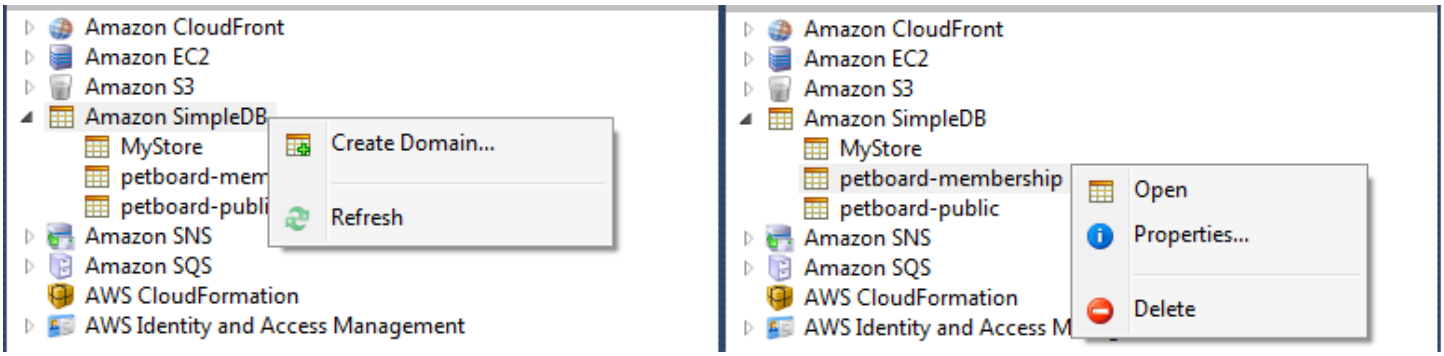
3. 在 Add Permission (添加权限) 对话框中，您可以使用 CIDR 表示法指定可访问您的 RDS 实例的 IP 地址，或者指定可访问您的 RDS 实例的 EC2 安全组。选择 EC2 安全组时，您可以为与 AWS 账户 拥有访问权限关联的所有 EC2 实例指定访问权限，也可以从下拉列表中选择 EC2 安全组。



该 AWS 工具包会尝试确定您的 IP 地址，并使用相应的 CIDR 规范自动填充对话框。但是，如果您的计算机是通过防火墙访问 Internet 的，则由 Toolkit 确定的 CIDR 可能不准确。

使用 Explorer 中的亚马逊 Simp AWS leDB

AWS Explorer 显示与活跃账户关联的所有亚马逊 SimpleDB 域名。 AWS 在 AWS 资源管理器中，您可以创建或删除亚马逊 SimpleDB 域名。

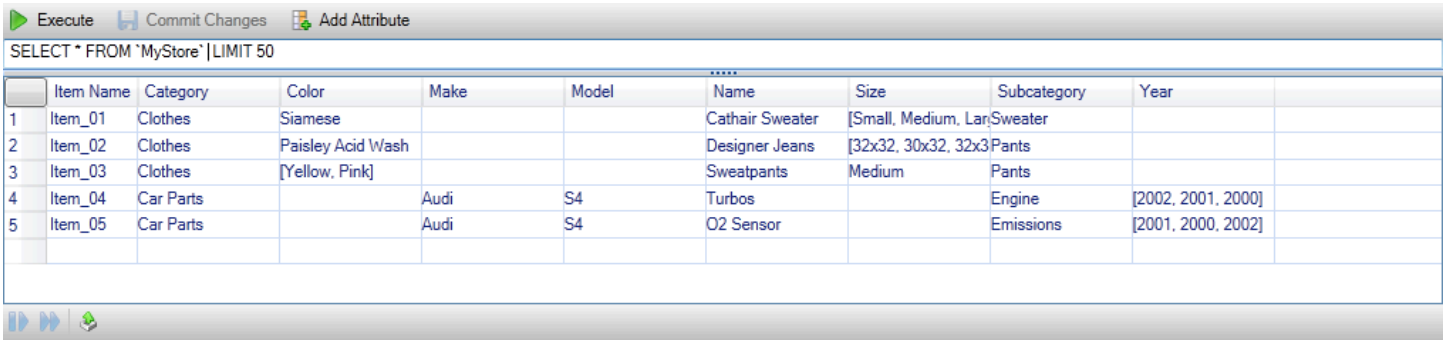


Create, delete, or open Amazon SimpleDB domains associated with your account

执行查询并编辑结果

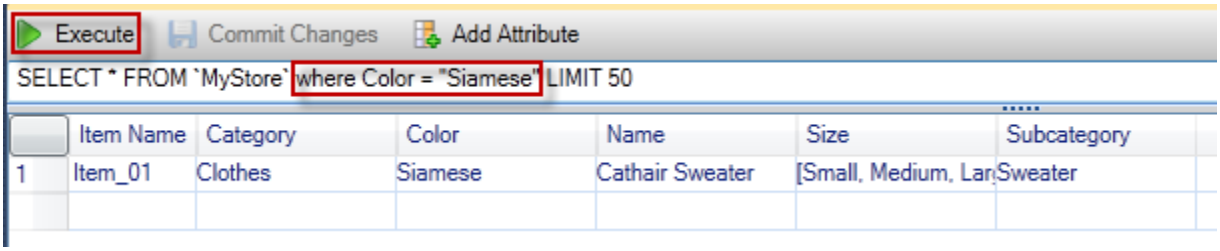
AWS Explorer 还可以显示 Amazon SimpleDB 域的网格视图，您可以从中查看该域中的项目、属性和值。您可以执行查询，以便仅显示该域的一部分项目。通过双击单元格，您可以编辑该项目对应属性的值。您还可以向域添加新属性。

此处显示的域来自于 适用于 .NET 的 AWS SDK包含的 Amazon SimpleDB 示例。



Amazon SimpleDB grid view

要执行查询，请在网格视图顶部的文本框中编辑查询，然后选择 Execute (执行)。将筛选此视图以仅显示与查询匹配的项目。

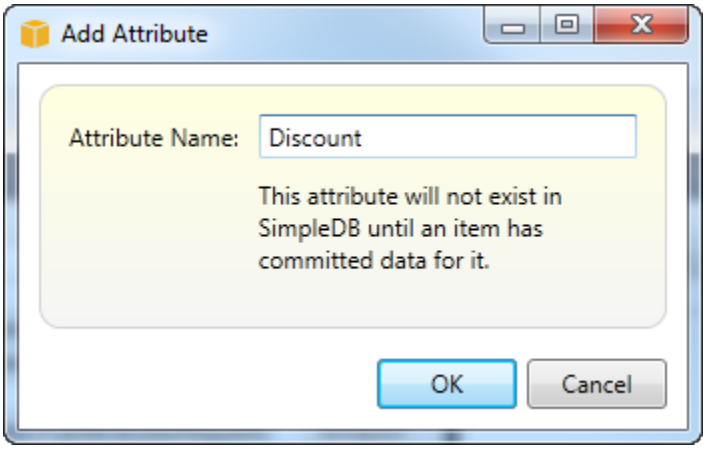


Execute query from AWS Explorer

要编辑与某个属性关联的值，请双击对应的单元格，编辑值，然后选择 Commit Changes (提交更改)。

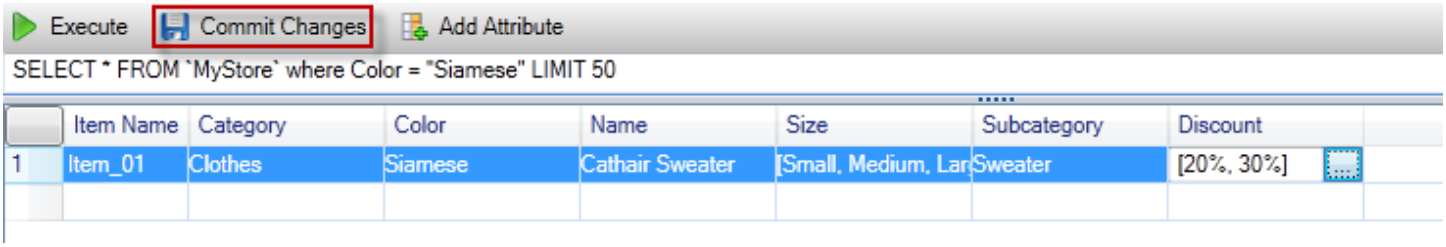
添加属性

要添加属性，请在视图顶部选择 Add Attribute (添加属性)。



添加属性 dialog box

要使属性成为域的一部分，您必须将属性的一个值添加到至少一个项目，然后选择 Commit Changes (提交更改)。



Commit changes for a new attribute

为查询结果分页

视图底部有三个按钮。

Paginate and export buttons

前两个按钮为查询结果提供分页。要额外显示一页结果，请选择第一个按钮。要额外显示十页结果，请选择第二个按钮。在此上下文中，一个页面等于 100 行或等于 LIMIT 值指定的结果数（如果该值包含在查询中）。

导出到 CSV

最后一个按钮将当前结果导出到 CSV 文件。

使用 Explorer 中的 Amazon SQS AWS

Amazon Simple Queue Service（Amazon SQS）是一项灵活的队列服务，用于实现软件应用程序中的不同执行进程之间的消息传递。Amazon SQS 队列位于 AWS 基础设施中，但是传递消息的进程可以位于本地、Amazon EC2 实例上，也可以位于这些实例的某种组合上。Amazon SQS 非常适合用于协调跨多台计算机的工作分配。

利用 Toolkit for Visual Studio，您可以查看与活动账户关联的 Amazon SQS 队列、创建并删除队列以及通过队列发送消息。（“活动账户”是指 AWS 各区服务浏览器中的选定账户。）

有关 Amazon SQS 的更多信息，请转至文档中的 SQS 简介。 AWS

创建队列

您可以通过资源管理器创建 Amazon SQS 队列。 AWS 队列的 ARN 和 URL 将基于活动账户的账号和您在创建队列时指定的队列名称。

创建队列

1. 在 AWS 资源管理器中，打开 Amazon SQS 节点的上下文（右键单击）菜单，然后选择 “创建队列”。
2. 在 Create Queue (创建队列) 对话框中，指定队列名称、默认可见性超时和默认传递延迟。默认可见性超时和默认传递延迟以秒为单位指定。默认可见性超时是在某个给定进程已获得消息后该消息将对潜在接收进程不可见的时间量。默认传递延迟是从发送消息到消息首次对潜在接收进程可见的时间量。
3. 选择确定。新队列将显示为 Amazon SQS 节点下的一个子节点。

删除队列

您可以从 AWS 资源管理器中删除现有队列。如果删除了某个队列，与该队列关联的所有消息都不再可用。

删除队列

1. 在 AWS 资源管理器中，打开要删除的队列的上下文（右键单击）菜单，然后选择 “删除”。

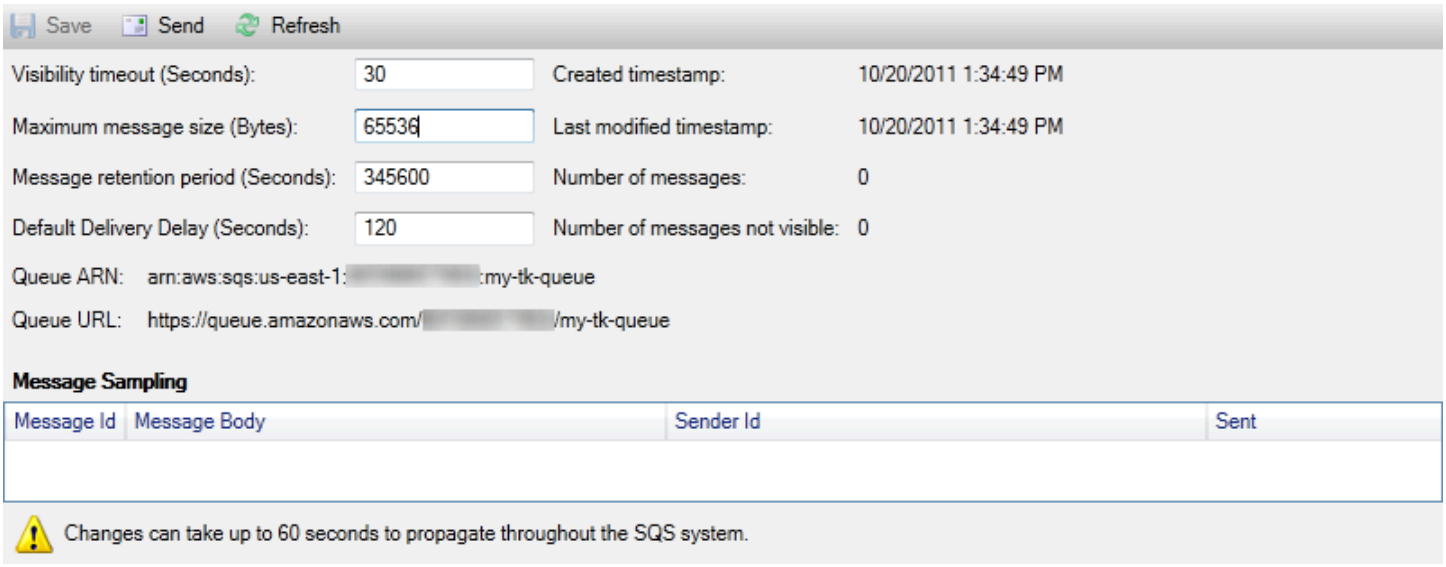
管理队列属性

您可以查看和编辑在 E AWS xplorer 中显示的任何队列的属性。还可以从此属性视图向队列发送消息。

管理队列属性

- 在 AWS 资源管理器中，打开要管理其属性的队列的上下文（右键单击）菜单，然后选择 “查看队列”。

  在队列属性视图中，您可以编辑可见性超时、最大消息大小、消息保留期和默认传递延迟。可在发送消息时覆盖默认传递延迟。在以下屏幕截图中，模糊化的文字是队列 ARN 和 URL 的账号组成部分。



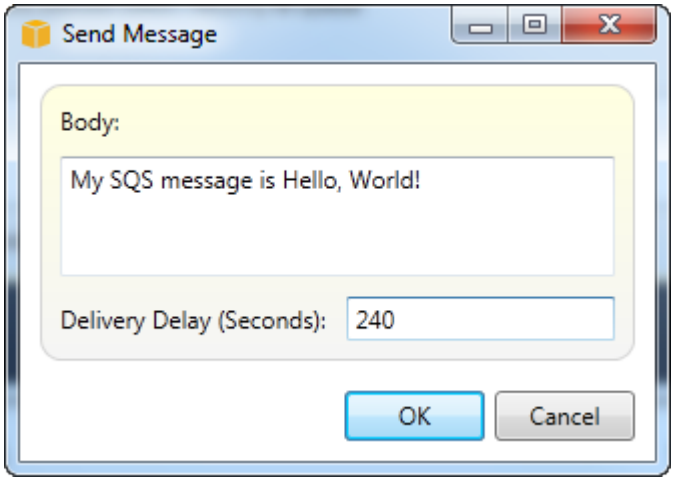
SQS queue properties view

向队列发送消息

在队列属性视图中，您可以向队列发送消息。

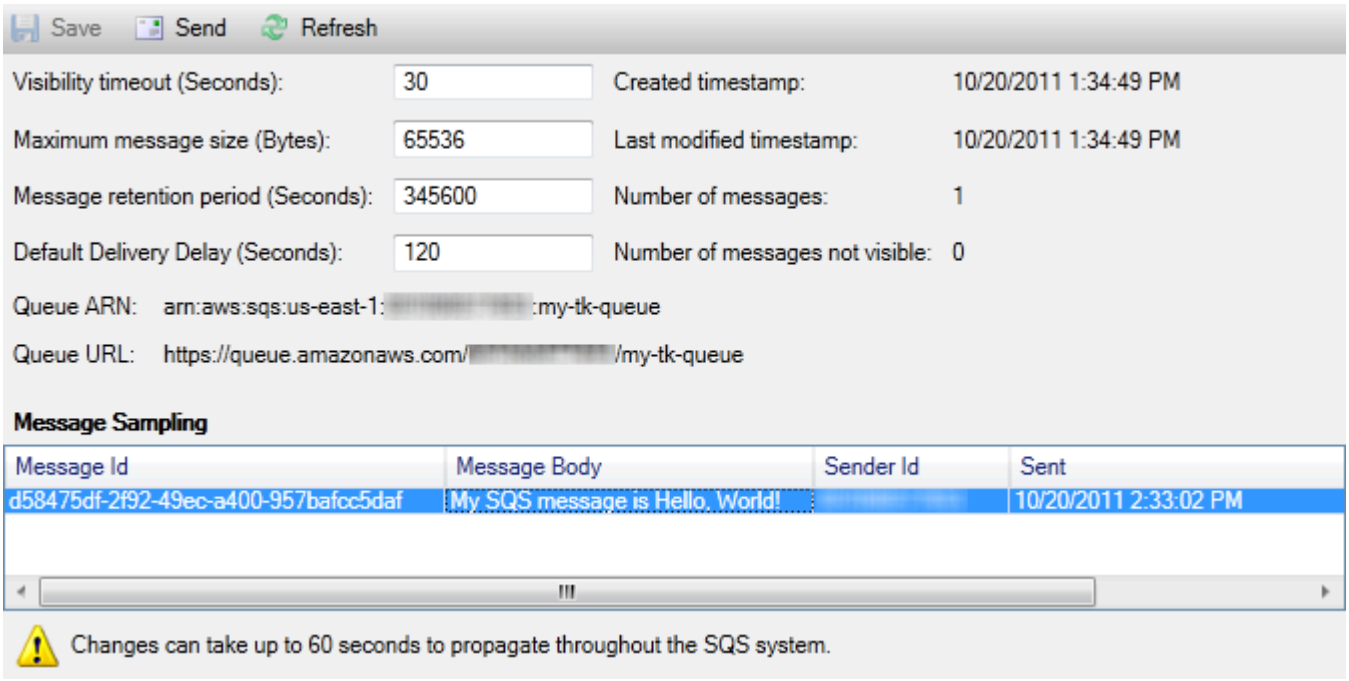
发送邮件

1. 在队列属性视图的顶部，选择 Send (发送) 按钮。
2. 键入消息。（可选）输入将覆盖队列的默认传递延迟的传递延迟。在以下示例中，我们已使用值 240 秒覆盖默认延迟。选择确定。



发送消息 dialog box

3. 等待约 240 秒（4 分钟）。消息将显示在队列属性视图的 Message Sampling (消息采样) 部分中。



SQS properties view with sent message

队列属性视图中的时间戳是您选择 Send (发送) 按钮的时间。此时间不包含延迟。因此，消息显示在队列中并对接收方可用的时间可能晚于此时间戳。此时间戳以计算机的本地时间显示。

身份和访问管理

AWS Identity and Access Management (IAM) 使您能够更安全地管理对 AWS 账户 和资源的访问权限。使用 IAM，您可以在主用户（根）中创建多个用户 AWS 账户。这些用户可以有自己的凭证：密码、访问密钥 ID 和秘密密钥。但是，所有 IAM 用户都共享单个账号。

您可以通过将 IAM policy 附加到用户来管理每个 IAM 用户的资源访问级别。例如，您可以将策略附加到一个 IAM 用户，该用户提供了对您的账户中的 Amazon S3 服务和相关资源的访问权限，但未提供对任何其他服务或资源的访问权限。

如要实现更有效的访问管理，您可以创建 IAM 组，即若干用户的集合。当您将一个策略附加到组时，它将影响作为该组成员的所有用户。

除了在用户和组级别管理权限之外，IAM 还支持 IAM 角色的概念。与用户和组相似，您可以将策略附加到 IAM 角色。随后，您可以将该 IAM 角色与 Amazon EC2 实例关联。在 EC2 实例上运行的应用程序可以 AWS 使用 IAM 角色提供的权限进行访问。有关将 IAM 角色与 Toolkit 一起使用的更多信息，请参阅创建 IAM 角色。有关 IAM 的更多信息，请参阅《IAM 用户指南》。

创建和配置 IAM 用户

IAM 用户允许您向其他人授予访问您的权限 AWS 账户。由于您能够将策略附加到 IAM 用户，因此您可以精确地限制 IAM 用户可访问的资源以及他们可对这些资源执行的操作。

作为最佳实践，所有访问的用户都 AWS 账户 应以 IAM 用户身份进行访问，即使是账户的所有者也是如此。这确保了在一个 IAM 用户的凭证泄露时只需停用该用户的凭证。不需要停用或更改账户的根凭证。

在 Toolkit for Visual Studio 中，您可以将 IAM policy 附加到用户或将用户分配到组，由此向 IAM 用户分配权限。分配到某个组的 IAM 用户将从附加到该组的策略派生其权限。有关更多信息，请参阅创建 IAM 组和将 IAM 用户添加到 IAM 组。

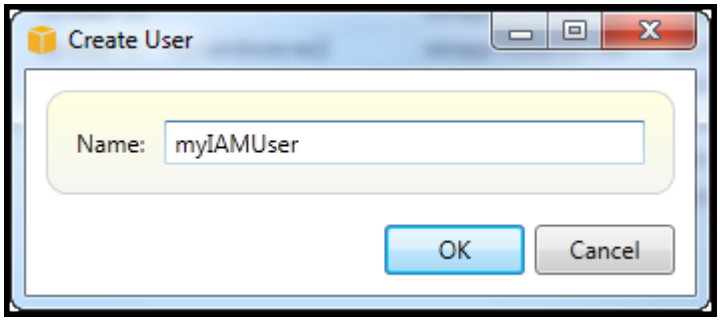
您还可以从 Visual Studio 的 Toolkit for Visual Studio 中为 IAM 用户生成 AWS 证书（访问密钥 ID 和密钥）。有关更多信息，请参阅为 IAM 用户生成凭证

适用于 Visual Studio 的 Toolkit 支持指定用于通过 AWS Explorer 访问服务的 IAM 用户证书。由于 IAM 用户通常无法完全访问所有 Amazon Web Services，因此 AWS Explorer 中的某些功能可能不可用。如果您在活跃账户是 IAM 用户时使用 AWS Explorer 更改资源，然后将活跃账户切换到根账户，则在您刷新 AWS Explorer 中的视图之前，更改可能不可见。要刷新该视图，请选择刷新 () 按钮。

有关如何从中配置 IAM 用户的信息 AWS 管理控制台，请转到 IAM 用户指南中的使用用户和群组。

创建 IAM 用户

1. 在 AWS 资源管理器中，展开AWS Identity and Access Management节点，打开 “用户” 的上下文（右键单击）菜单，然后选择 “创建用户”。
2. 在创建用户对话框中，为 IAM 用户键入名称并选择确定。这是 IAM 友好名称。有关 IAM 用户的名称限制信息，请参阅《IAM 用户指南》。



Create an IAM user

新用户将作为子节点显示在 AWS Identity and Access Management 节点的用户下。

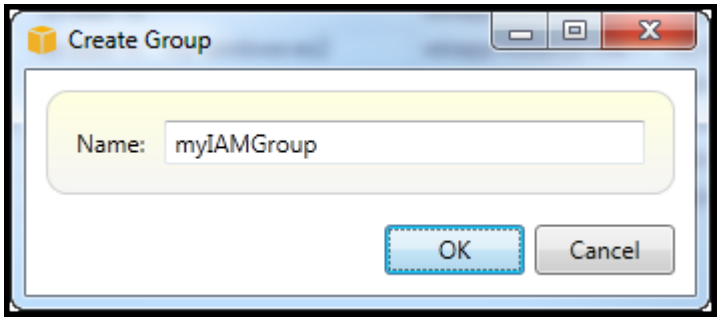
有关如何创建策略并将其附加到用户的信息，请参阅创建 IAM 策略。

创建 IAM 组

组提供了一个将 IAM policy 应用于用户集合的方法。有关如何管理 IAM 用户和组的信息，请参阅《IAM 用户指南》中的使用用户和组。

创建 IAM 组

1. 在 AWS Explorer 中，在 “身份和访问管理” 下，打开群组的上下文（右键单击）菜单，然后选择 “创建群组”。
2. 在创建组对话框中，为 IAM 组键入名称并选择确定。



Create IAM group

新 IAM 组将显示在身份和访问管理的组子节点下。

有关创建策略并将其附加到 IAM 组的信息，请参阅创建 IAM policy。

将 IAM 用户添加到 IAM 组

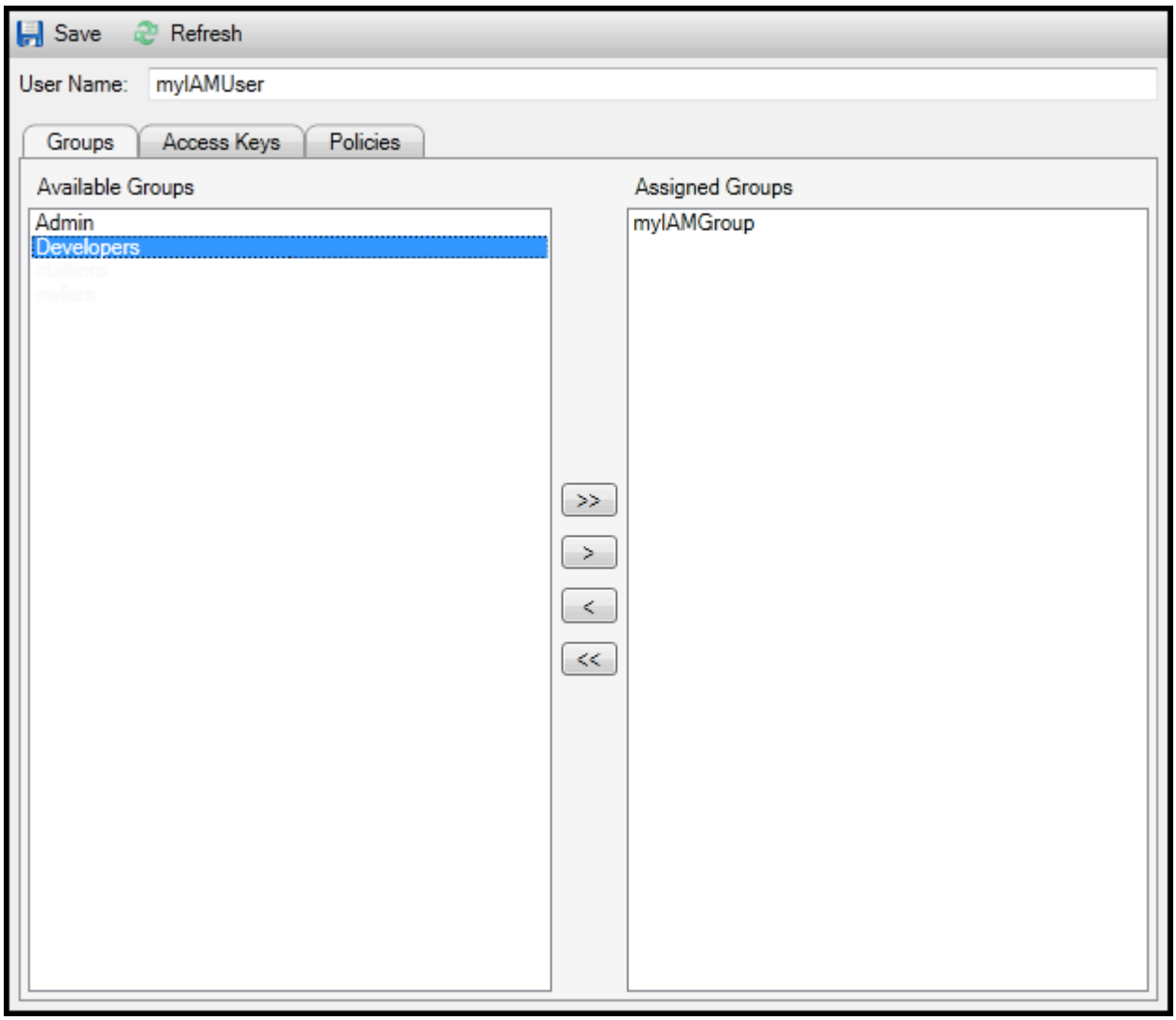
作为 IAM 组成员的 IAM 用户将从附加到该组的策略派生访问权限。IAM 组的用途是简化对一系列 IAM 用户的权限的管理。

有关附加到某个 IAM 组的策略如何与附加到作为该 IAM 组成员的 IAM 用户的策略交互的信息，请参阅《IAM 用户指南》中的“管理 IAM policy”。

在 AWS Explorer 中，您可以从 “用户” 子节点（而不是群组）子节点将 IAM 用户添加到 IAM 群组。

将 IAM 用户添加到 IAM 组

1. 在 AWS Explorer 中，在 “身份和访问管理” 下，打开 “用户” 的上下文（右键单击）菜单，然后选择 “编辑”。



Assign an IAM user to a IAM group

2. 组选项卡的左侧窗格显示了可用 IAM 组。右侧窗格显示了已包含指定 IAM 用户的组。

   要将 IAM 用户添加到某个组，请在左侧窗格中，选择该 IAM 组，然后选择 > 按钮。

   要从某个组中移除 IAM 用户，请在右侧窗格中，选择该 IAM 组，然后选择 < 按钮。

   要将 IAM 用户添加到所有 IAM 组，请选择 >> 按钮。同样，要从所有组中移除 IAM 用户，请选择 << 按钮。

   要选择多个组，请按顺序选择它们。您不需要按住 Ctrl 键。要从您的选项中清除某个组，只需再次选择它。

3. 完成向 IAM 组分配 IAM 用户后，选择保存。

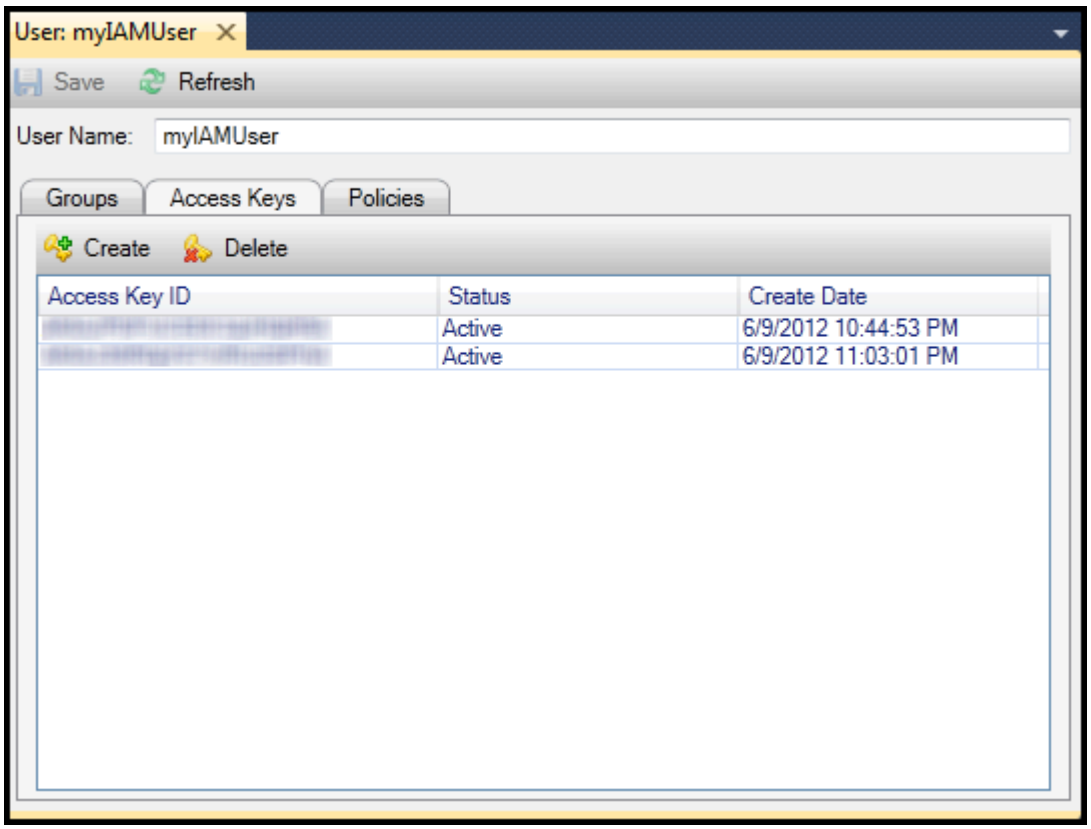
为 IAM 用户生成凭证

利用 Toolkit for Visual Studio，您可以生成用于对 AWS进行 API 调用的访问密钥 ID 和秘密密钥。这些密钥也可指定用于通过 Toolkit 访问 Amazon Web Services。有关如何指定与 Toolkit 配合使用的凭证的更多信息，请参阅凭证。有关如何安全处理凭证的更多信息，请参阅管理 AWS 访问密钥的最佳实践。

Toolkit 无法用于为 IAM 用户生成密码。

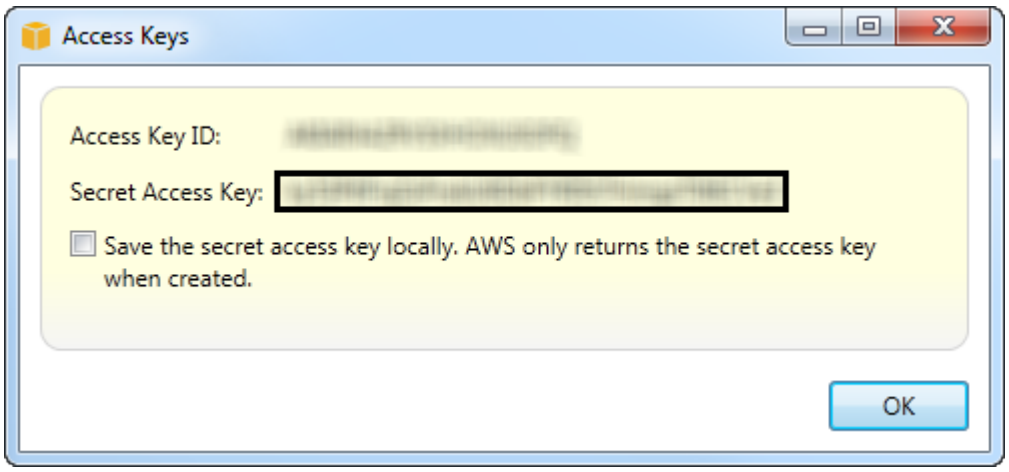
为 IAM 用户生成凭证

1. 在 AWS 资源管理器中，打开 IAM 用户的上下文（右键单击）菜单，然后选择编辑。



2. 要生成凭证，请在 Access Keys (访问密钥) 选项卡上，选择 Create (创建)。

   您只能为每个 IAM 用户生成两组凭证。如果您已经有两组凭证并需要再创建一组凭证，则必须删除现有的凭证组之一。

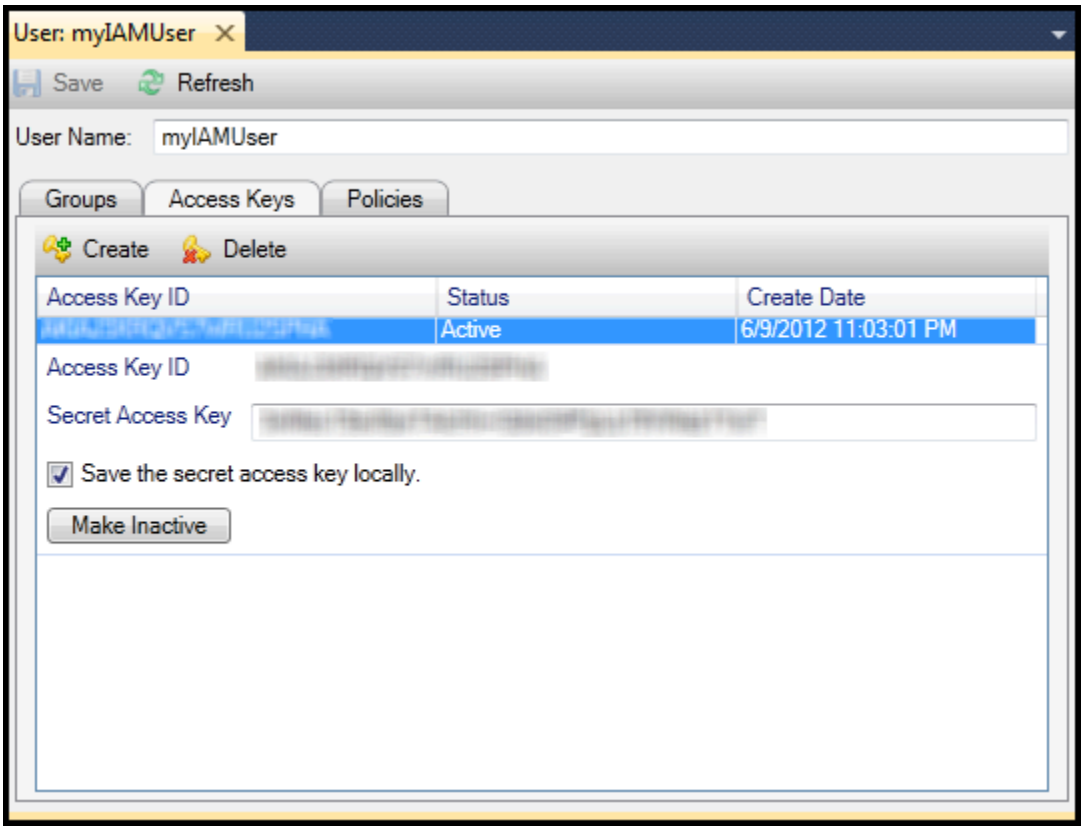


reate credentials for IAM user

如果您希望 Toolkit 将私有访问密钥的加密副本保存到本地驱动器，请选择 “将私有访问密钥保存在本地”。 AWS 仅在创建时返回私有访问密钥。您还可以从该对话框中复制秘密访问密钥并将其保存在安全的位置。

3. 选择确定。

生成凭证后，您可以从 Access Keys (访问密钥) 选项卡查看它们。如果您选择了让 Toolkit 本地保存私有密钥的选项，该密钥将显示在此处。



Create credentials for IAM user

如果您自行保存了私有密钥并且希望 Toolkit 保存它，请在 Secret Access Key (秘密访问密钥) 框中，键入秘密访问密钥，然后选择 Save the secret access key locally (在本地保存秘密访问密钥)。

要停用凭证，请选择 Make Inactive (转为非活动)。（如果您怀疑凭证已泄露，则可以这样做。如果您得到凭证安全的保证，则可以重新激活凭证。）

创建 IAM 角色

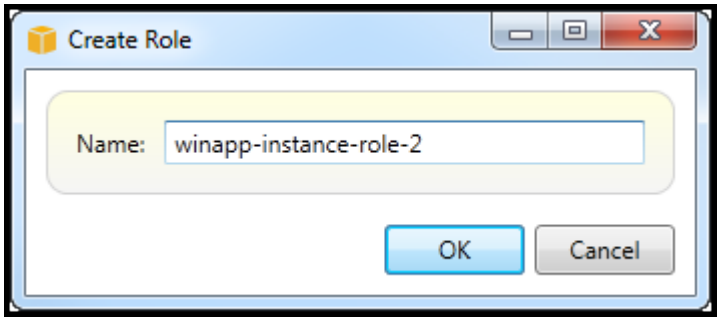
Toolkit for Visual Studio 支持创建和配置 IAM 角色。和使用用户和组一样，您可以将策略附加到 IAM 角色。随后，您可以将该 IAM 角色与 Amazon EC2 实例关联。与 EC2 实例的关联通过实例配置文件来处理，后者是角色的逻辑容器。在 EC2 实例上运行的应用程序将自动获得由与 IAM 角色关联的策略指定的访问级别。即使应用程序未指定其他 AWS 凭据，也是如此。

例如，您可以创建角色并将仅限于访问 Amazon S3 的策略附加到该角色。将此角色与 EC2 实例关联，随后您可以在该实例上运行一个应用程序，该应用程序将能够访问 Amazon S3，但无法访问任何其他服务或资源。这种方法的优点是，您无需担心在 EC2 实例上安全地传输和存储 AWS 证书。

有关 IAM 角色的更多信息，请参阅《IAM 用户指南》中的“使用 IAM 角色”。有关 AWS 使用与 Amazon EC2 实例关联的 IAM 角色进行访问的程序的示例，请参阅 Java、.NET、PHP 和 Ruby 的 AWS 开发者指南（使用 IAM 设置证书、创建 IAM 角色和使用 IAM 策略）。

创建 IAM 角色

1. 在 AWS Explorer 中，在 “身份和访问管理” 下，打开 “角色” 的上下文（右键单击）菜单，然后选择 “创建角色”。
2. 在创建角色对话框中，为 IAM 角色键入名称并选择确定。



Create IAM role

新 IAM 角色将显示在身份和访问管理中的角色下方。

有关如何创建策略并将其附加到角色的信息，请参阅创建 IAM 策略。

创建 IAM 策略

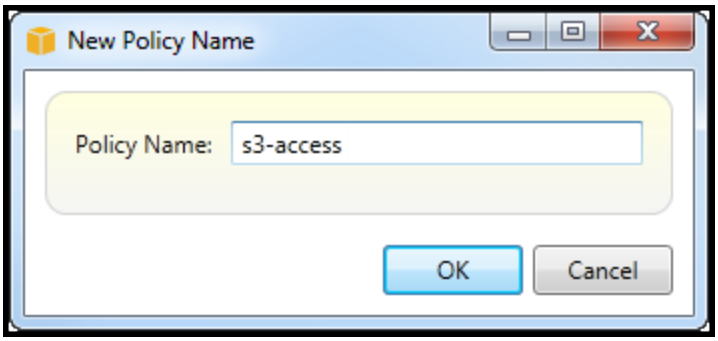
策略是 IAM 的基础。策略可与 IAM 实体（如用户、组或角色）关联。策略指定为用户、组或角色启用的访问级别。

创建 IAM 策略

在 AWS Explorer 中，展开节AWS Identity and Access Management点，然后展开要将策略附加到的实体类型（群组、角色或用户）的节点。例如，打开某个 IAM 角色的上下文菜单并选择编辑。

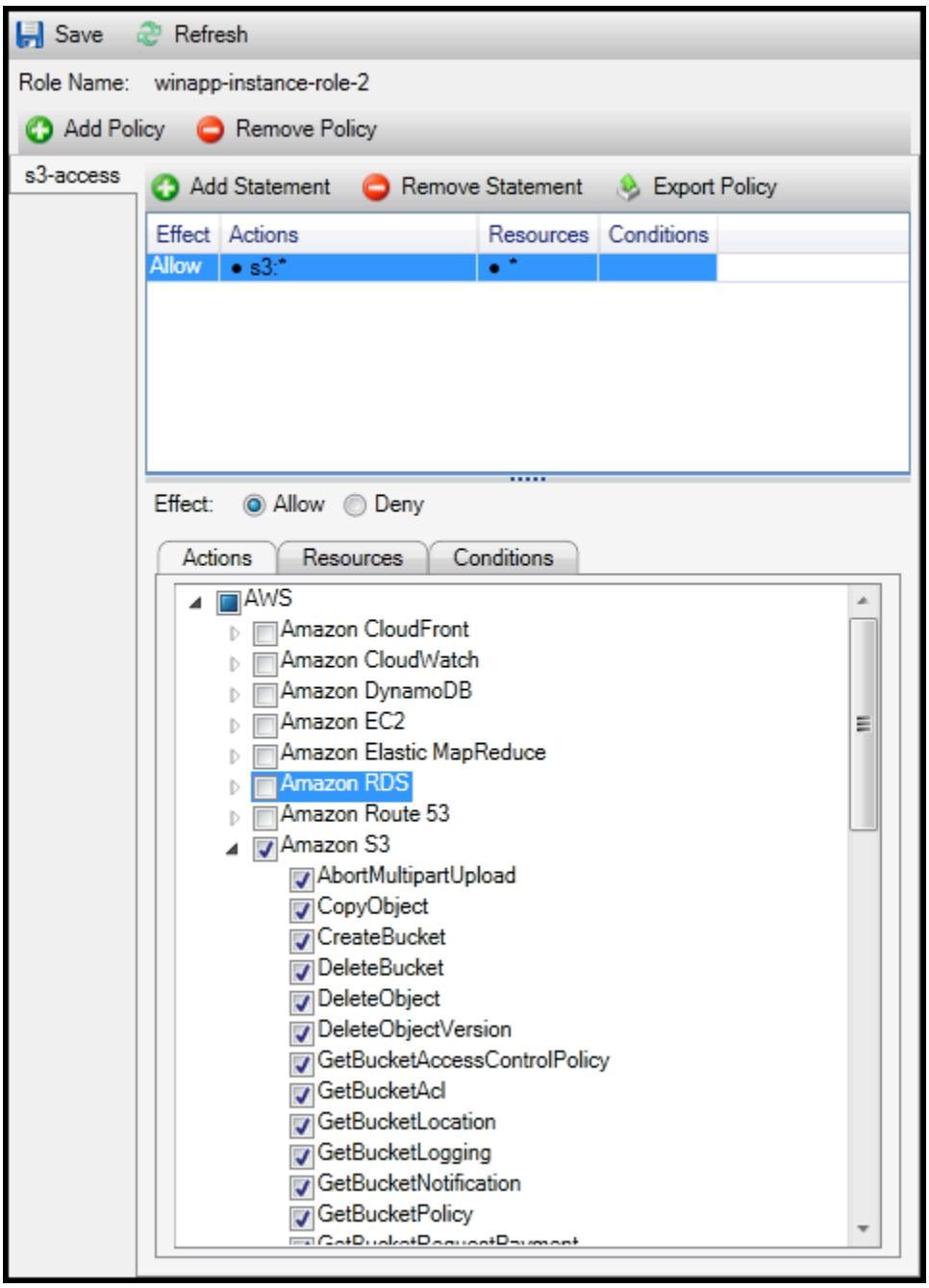
与该角色关联的选项卡将出现在 AWS 资源管理器中。选择 Add Policy (添加策略) 链接。

在 New Policy Name (新策略名称) 对话框中，键入策略的名称（例如，s3-access）。



New Policy Name dialog box

在策略编辑器中，添加策略语句以指定向角色提供的访问权限级别（在本示例中为 winapp-instance-role -2）。在本示例中，策略提供了对 Amazon S3 的完全访问权限，但未提供对任何其他资源的访问权限。



Specify IAM policy

若要获得更精确的访问控制，您可以在策略编辑器中展开子节点以允许或拒绝与 Amazon Web Services 关联的操作。

当您编辑完策略时，请选择 Save (保存) 链接。

AWS Lambda

使用开发和部署基于.NET 内核的 C# Lambda 函数。 AWS Toolkit for Visual Studio AWS Lambda 是一项计算服务，允许您在不预置或管理服务器的情况下运行代码。Visual Studio 工具包包括 Visual Studio 的 AWS Lambda .NET 核心项目模板。

有关的更多信息 AWS Lambda，请参阅 AWS Lambda 开发人员指南。

有关 .NET Core 的更多信息，请参阅 Microsoft .NET Core 指南。有关适用于 Windows、macOS 和 Linux 平台的 .NET Core 先决条件和安装说明，请参阅 .NET Core 下载。

以下主题介绍如何 AWS Lambda 使用适用于 Visual Studio 的 Toolkit。

主题

基础 AWS Lambda 项目

您可以使用 AWS Toolkit for Visual Studio中提供的 Microsoft .NET Core 项目模板来创建 Lambda 函数。

创建 Visual Studio .NET Core Lambda 项目

您可以使用 Lambda Visual Studio 模板和蓝图来帮助加速项目初始化。Lambda 蓝图包含预先编写好的函数，它们简化了创建灵活项目基础的过程。

> **Note**
>
> Lambda 服务对不同程序包类型有数据限制。有关数据限制的详细信息，请参阅《AWS Lambda 用户指南》中的 Lambda 配额主题。

在 Visual Studio 中创建 Lambda 项目

1. 在 Visual Studio 中，展开文件菜单，展开新建，然后选择项目。

2. 在新建项目对话框中，将语言、平台和项目类型下拉框都设置为“全部”，然后在搜索字段中键入 aws lambda。选择 AWS Lambda 项目（.NET Core - C＃）模板。
3. 在名称字段中输入 **AWSLambdaSample**，指定所需的文件位置，然后选择创建以继续。
4. 在选择蓝图页面上，选择空函数蓝图，然后选择完成以创建 Visual Studio 项目。

复查项目文件

有两个项目文件需要复查：aws-lambda-tools-defaults.json 和 Function.cs。

以下示例显示 aws-lambda-tools-defaults.json 文件，该文件作为项目的一部分自动创建。您可以使用该文件中的字段设置构建选项。

> **Note**
>
> Visual Studio 中的项目模板包含多个不同的字段，请注意以下几点：
>
> - function-handler：指定运行 Lambda 函数时运行的方法
> - 在 function-handler 字段中指定一个值会在发布向导中预先填充该值。
> - 如果您重命名了函数、类或程序集，则也需要更新 aws-lambda-tools-defaults.json 文件中的相应字段。

```
{
  "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,
```

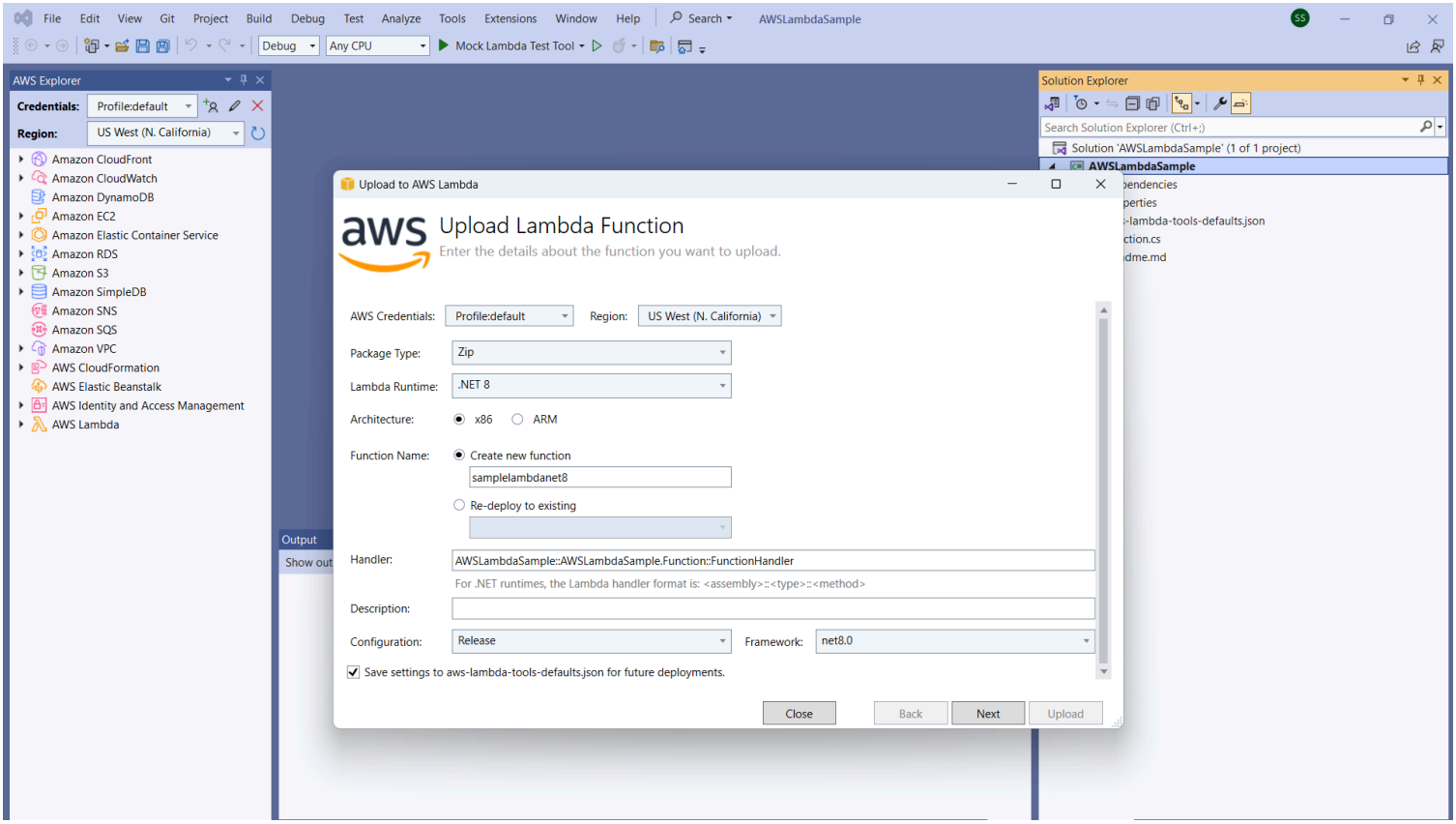
```
  "function-handler": "AWSLambdaSample::AWSLambdaSample.Function::FunctionHandler"
}
```

检查 Function.cs 文件。Function.cs 定义了要作为 Lambda 函数公开的 c# 函数。这里的 FunctionHandler 是在运行 Lambda 函数时运行的 Lambda 功能。在此项目中，定义了一个函数：FunctionHandler，它在输入文本上调用 ToUpper()。

您的项目现在可发布到 Lambda。

发布到 Lambda

以下过程和图像演示了如何使用 AWS Toolkit for Visual Studio将您的函数上传到 Lambda。



将函数发布到 Lambda

1. 展开 “视图” 并选择 “ AWS 资源管理器”，即可导航到AWS 资源管理器。
2. 在解决方案资源管理器中，打开（右键单击）要发布的项目的快捷菜单，然后选择发布到 AWS Lambda 以打开上传 Lambd a 函数窗口。
3. 在上传 Lambda 函数窗口中，填写以下字段：

a. 程序包类型：选择 **Zip**。作为构建过程的结果，将创建一个 ZIP 文件并将其上传到 Lambda。或者，也可以选择程序包类型 **Image**。教程：创建 Docker 映像的基本 Lambda 项目描述了如何使用程序包类型 **Image** 进行发布。

b. Lambda 运行时：从下拉菜单中选择 Lambda 运行时。

c. 架构：通过单选按钮选择您偏好的架构。

d. 函数名称：通过单选按钮选择创建新函数，然后输入 Lambda 实例的显示名称。 AWS 浏览器和 AWS 管理控制台 显示器都会引用此名称。

e. 处理程序：使用此字段指定函数处理程序。例如：**AWSLambdaSample::AWSLambdaSample.Function::FunctionHandler**。

f. （可选）描述：输入描述性文本，该文本将在 AWS 管理控制台中与您的实例一同显示。

g. 配置：从下拉菜单中选择所需的配置。

h. 框架：从下拉菜单中选择所需的框架。

i. 保存设置：选中此框将您当前的设置保存到 `aws-lambda-tools-defaults.json` 中，作为未来部署的默认值。

j. 选择下一步进入高级函数详细信息窗口。

4. 在高级函数详细信息窗口中，填写以下字段：

a. 角色名称：选择与您的账户关联的角色。该角色为函数中的代码发出的任何 AWS 服务调用提供临时证书。如果您没有角色，请在下拉选择器中滚动找到 “基于 AWS 托管策略的新角色”，然后选择AWSLambdaBasicExecutionRole。此角色拥有最小访问权限。

> **Note**
>
> 您的账户必须拥有运行 IAM ListPolicies 操作的权限，否则角色名称列表将为空，您将无法继续。

b. （可选）如果您的 Lambda 函数访问 Amazon VPC 上的资源，请选择子网和安全组。

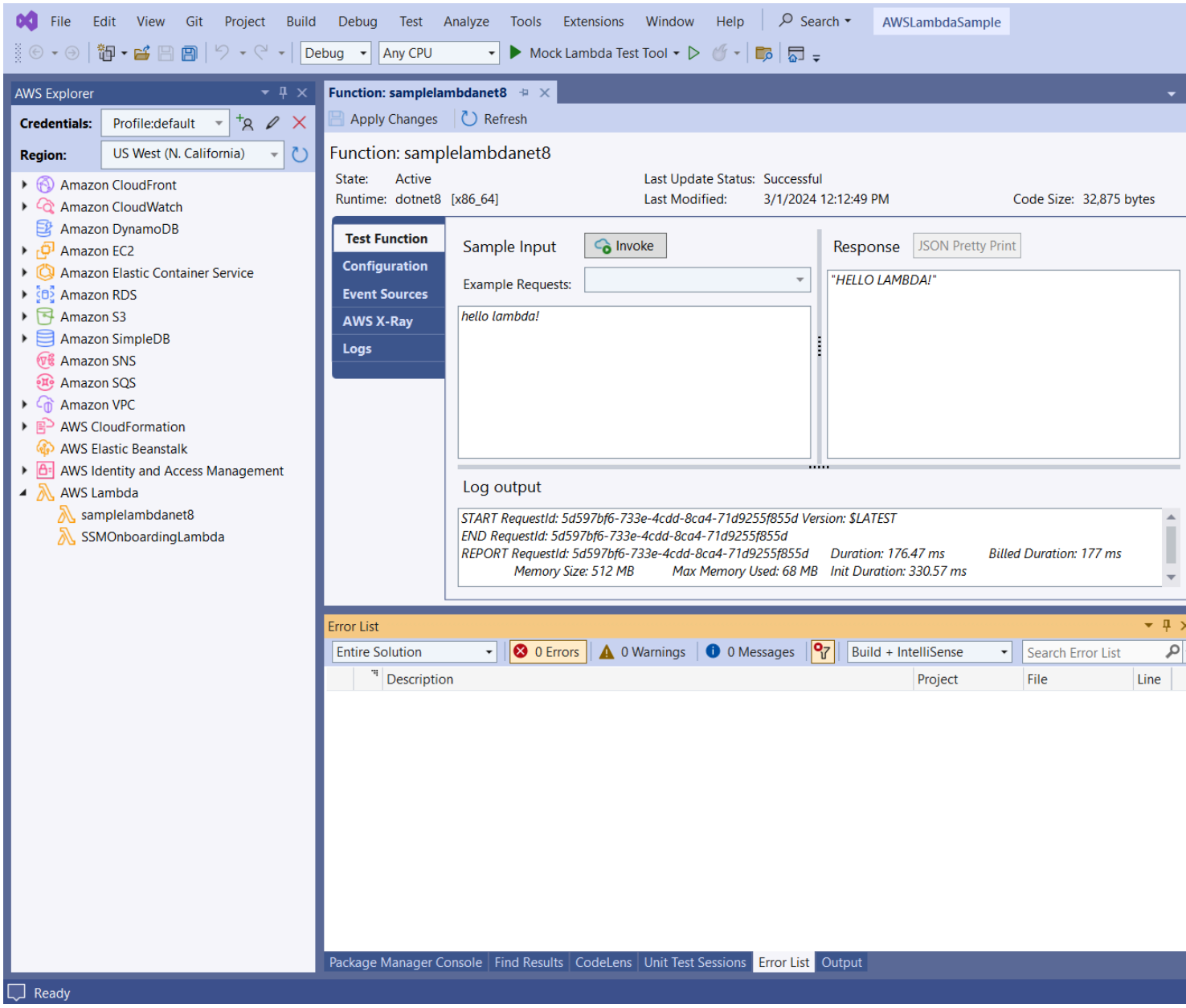
c. （可选）设置您的 Lambda 函数所需的任何环境变量。这些密钥会被默认服务密钥自动加密，此项服务是免费的。或者，您可以指定需要付费的 AWS KMS 密钥。KMS 是一项托管服务，可使用它创建和控制用于对数据进行加密的加密密钥。如果您有 AWS KMS 密钥，则可以从列表中将其选中。

5. 选择上传以打开上传函数窗口并开始上传过程。

> **Note**
>
> 函数上传到时，将显示 “上传函数” 页面 AWS。要使向导在上传后保持打开状态以便查看报告，请在上传完成之前取消选中表单底部的在成功完成时自动关闭向导。
> 函数上传后，您的 Lambda 函数就会上线。函数：视图页面将打开并显示您的新 Lambda 函数的配置。

6. 在测试函数选项卡上，在文本输入字段输入 `hello lambda!`，然后选择调用以手动调用您的 Lambda 函数。您的文本会出现在响应选项卡中，并已转换为大写字母。

> **Note**
>
> 您可以随时通过双击 AWS Explorer 内 AWS Lambda 节点下已部署的实例来重新打开函数：视图。



7. （可选）要确认您已成功发布您的 Lambda 函数，请登录 AWS 管理控制台 并选择 Lambda。控制台会显示您发布的所有 Lambda 函数，包括您刚刚创建的函数。

清理

如果您不打算继续使用此示例进行开发，请删除您部署的函数，这样就不会为账户中未使用的资源付费。

> **Note**
>
> Lambda 会自动为您监控 Lambda 函数，并通过亚马逊报告指标。 CloudWatch要监控您的函数并对其进行故障排除，请参阅 AWS Lambda 开发者指南中的使用 Amazon 进行故障排除和监控 AWS Lambda 函数 CloudWatch的主题。

删除函数

1. 在 AWS Explorer 中，展开 AWS Lambda 节点。
2. 右键单击已部署的实例，然后选择删除。

创建 Docker 镜像的基本 AWS Lambda 项目

你可以使用 Visual Studio 的 Toolkit for Visual Studio 将你的 AWS Lambda 函数部署为 Docker 镜像。使用 Docker 可以更好地控制运行时。例如，您可以选择诸如 .NET 8.0 这样的自定义运行时。您可以像部署任何其他容器映像一样部署 Docker 映像。本教程与教程：基本 Lambda 项目非常相似，但有两个区别：

- 项目中包含一个 Dockerfile。
- 选择了备用发布配置。

有关 Lambda 容器映像的信息，请参阅《AWS Lambda 开发人员指南》中的 Lambda 部署包。

有关使用 Lambda 的更多信息 AWS Toolkit for Visual Studio，请参阅本用户指南 AWS Toolkit for Visual Studio主题中的使用 AWS Lambda 模板。

创建 Visual Studio .NET Core Lambda 项目

您可以使用 Lambda Visual Studio 模板和蓝图来帮助加快项目初始化的速度。Lambda 蓝图包含预先编写好的函数，它们简化了创建灵活项目基础的过程。

创建 Visual Studio .NET Core Lambda 项目

1. 在 Visual Studio 中，展开文件菜单，展开新建，然后选择项目。
2. 在新建项目对话框中，将语言、平台和项目类型下拉框都设置为“全部”，然后在搜索字段中键入 **`aws lambda`**。选择 AWS Lambda 项目（.NET Core - C# ）模板。
3. 在项目名称字段中输入 **`AWSLambdaDocker`**，指定您的文件位置，然后选择创建。

4. 在“选择蓝图”页面上，选择 .NET 8（容器映像）蓝图，然后选择“完成”创建 Visual Studio 项目。您可以现在复查项目的结构和代码。

审查项目文件

以下各节分析了 .NET 8（容器映像）蓝图创建的三个项目文件：

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

1. Dockerfile

一个 `Dockerfile` 执行三个主要操作：

- `FROM`：确定构建此映像所依据的基础映像。此基础映像包含 .NET 运行时系统、Lambda 运行时系统以及为 Lambda .NET 进程提供入口点的 shell 脚本。
- `WORKDIR`：将映像的内部工作目录确定为 `/var/task`。
- `COPY`：将构建过程生成的文件从其本地位置复制到映像的工作目录中。

您可以指定以下可选 `Dockerfile` 操作：

- `ENTRYPOINT`：基础映像已经包含一个 `ENTRYPOINT`，即启动映像时执行的启动过程。如要指定自己的入口点，可以覆盖该基本入口点。
- `CMD`: 指示您要执行 AWS 哪个自定义代码。这要求自定义方法有一个完全限定名称。此行可以直接包含在 Dockerfile 中，也可以在发布过程中指定。

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

以下是 .NET 8（容器映像）蓝图创建的 Dockerfile 的示例。

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

WORKDIR /var/task
```

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

2。 aws-lambda-tools-defaults.json

`aws-lambda-tools-defaults.json` 文件用于为 Toolkit for Visual Studio 部署向导和 .NET Core CLI 指定默认值。以下列表描述了可在 `aws-lambda-tools-defaults.json` 文件中设置的字段。

- `profile`: 设置您的 AWS 个人资料。
- `region`：设置您的资源存储 AWS 区域。
- `configuration`：设置用于发布函数的配置。
- `package-type`：将部署包类型设置为容器映像或 .zip 文件存档。
- `function-memory-size`：设置函数的内存分配大小（以 MB 为单位）。
- `function-timeout`：超时是指一个 Lambda 函数可以运行的最长时间（以秒为单位）。您能够以 1 秒为增量来调整此值，最大值为 15 分钟。
- `docker-host-build-output-dir`：设置构建过程的输出目录，该目录与 `Dockerfile` 中的指令相关。
- `image-command`：您的方法的完全限定名称，即您希望 Lambda 函数运行的代码。语法如下：`{Assembly}::{Namespace}.{ClassName}::{MethodName}`。有关更多信息，请参阅处理程序签名。在此处设置 `image-command` 后，稍后会在 Visual Studio 的“发布”向导中预填充此值。

以下是 aws-lambda-tools-defaults .NET 8（容器镜像）蓝图创建的.json 示例。

```
{
  "Information": [
```

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

3. Function.cs

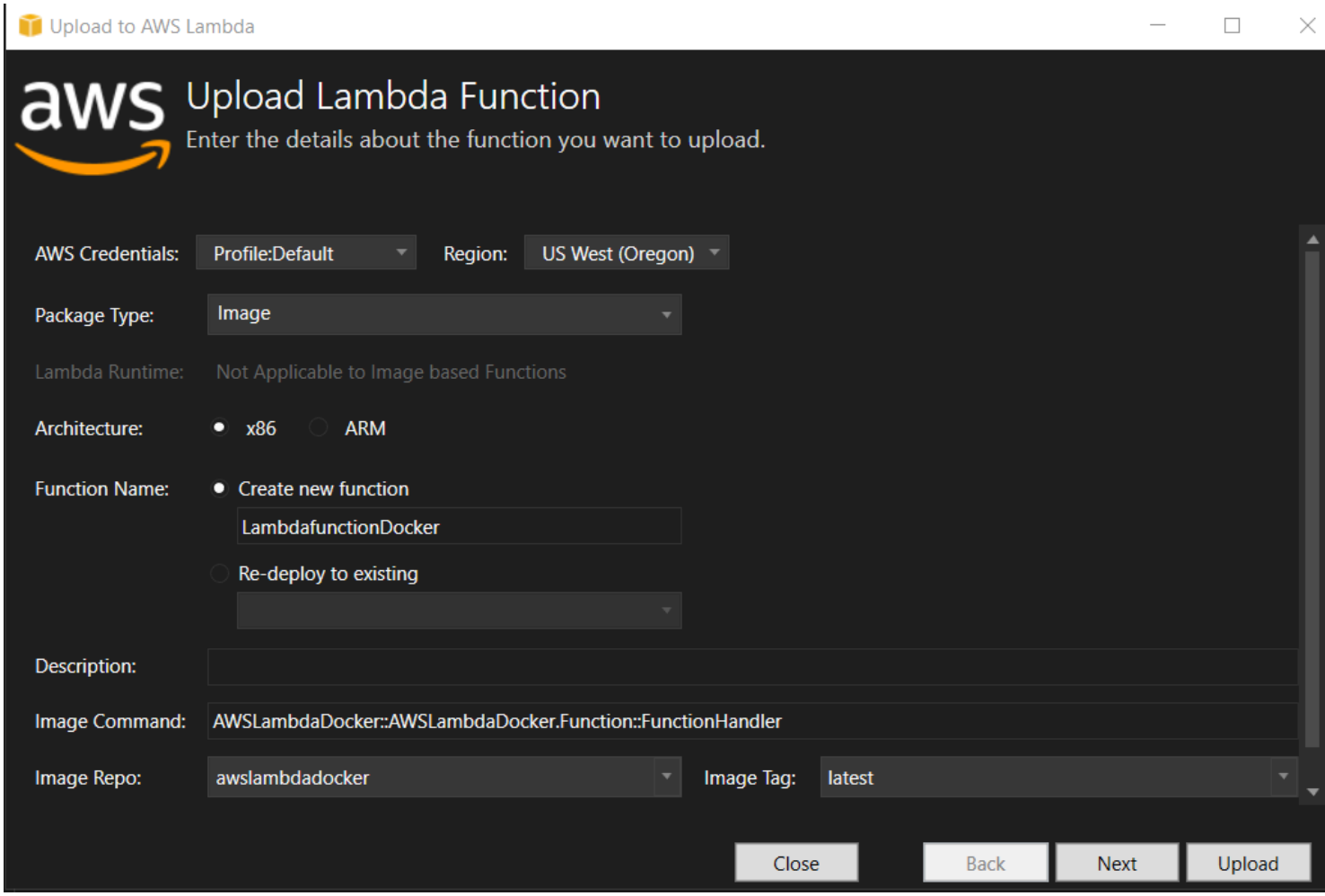
`Function.cs` 文件定义要作为 Lambda 函数公开的 c# 函数。`FunctionHandler` 是在运行 Lambda 函数时运行的 Lambda 功能。在这个项目中，`FunctionHandler` 对输入的文本调用 `ToUpper()`。

发布到 Lambda

构建过程中生成的 Docker 映像上传到 Amazon Elastic Container Registry（Amazon ECR）。Amazon ECR 一个完全托管式 Docker 容器映像库，您可以使用该映像库存储、管理和部署 Docker 容器映像。Amazon ECR 托管映像，然后 Lambda 会引用该映像，以便在调用时提供编程的 Lambda 功能。

将函数发布到 Lambda

1. 在解决方案浏览器中，（右键单击）打开项目的上下文菜单，然后选择发布到 AWS Lambda 以打开上传 Lambda 函数窗口。
2. 在上传 Lambda 函数页面上，执行以下操作：



a. 对于包类型，**`Image`** 已被自动选为包类型，因为发布向导在项目中检测到了 `Dockerfile`。

b. 对于函数名称，为 Lambda 实例输入显示名称。此名称是在 Visual Studio 的 AWS 各区服务浏览器中和 AWS 管理控制台中显示的引用名称。

c. 对于描述，输入要在 AWS 管理控制台中与您的实例一起显示的文本。

d. 对于映像命令，输入希望 Lambda 函数运行的方法的完全限定路径：**`AWSLambdaDocker::AWSLambdaDocker.Function::FunctionHandler`**

> **Note**
>
> 此处输入的任何方法名称都将覆盖 Dockerfile 中的任何 CMD 指令。只有在 `Dockerfile` 包含用于指示如何启动 Lambda 函数的 `CMD` 时，输入映像命令才是可选的。

e. 对于映像存储库，输入新的或现有 Amazon Elastic Container Registry 的名称。构建过程创建的 Docker 映像将上传到此映像库。要发布的 Lambda 定义将引用该 Amazon ECR 映像。

f. 对于映像标签，输入一个 Docker 标签以与存储库中的映像相关联。

g. 选择下一步。

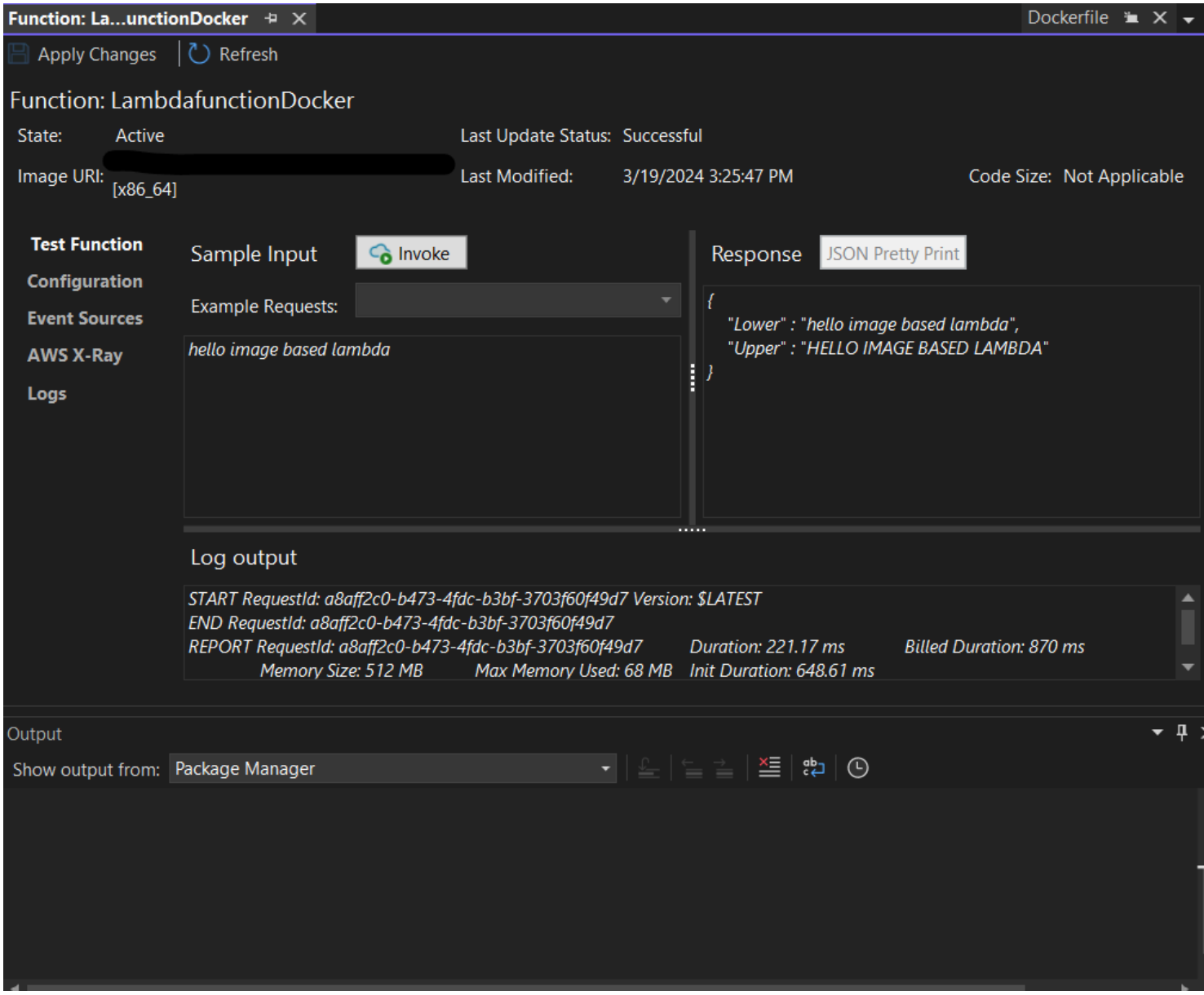
3. 在高级函数详细信息页面的角色名称中，选择与您的账户关联的角色。该角色用于为函数中的代码所发起的任何 Amazon Web Services 调用提供临时凭证。如果您没有角色，请选择 “基于 AWS 托管策略新建角色”，然后选择AWSLambdaBasicExecutionRole。

> **Note**
>
> 您的账户必须拥有运行 IAM ListPolicies 操作的权限，否则角色名称列表将为空。

4. 选择上传以启动上传和发布过程。

> **Note**
>
> 上传函数时，将显示正在上传函数页面。然后，发布过程根据配置参数构建映像，必要时创建 Amazon ECR 存储库，将映像上传到存储库，然后创建引用包含该映像的存储库的 Lambda。
>
> 上传函数后，函数页面将打开并显示新 Lambda 函数的配置。

5. 要手动调用 Lambda 函数，请在测试函数选项卡上，在请求的自由文本输入字段输入 `hello image based lambda`，然后选择调用。您的文本将转换为大写并显示在响应中。



6. 要查看存储库，请在 AWS 各区服务浏览器中的 Amazon Elastic Container Service 下选择存储库。

   您可以随时通过双击 AWS Explorer 内 AWS Lambda 节点下已部署的实例来重新打开函数：视图。

> **Note**
>
> 如果你的 AWS 资源管理器窗口未打开，你可以通过 “视图”-> “AWS 资源管理器” 将其停靠

7. 请注意配置选项卡上其他特定于映像的配置选项。此选项卡提供了一种覆盖可能已在 Dockerfile 中指定的 ENTRYPOINT、CMD、和 WORKDIR 的方法。描述是您在上传/发布期间输入的描述（如果有）。

清理

如果您不打算继续使用此示例进行开发，请记得删除已部署的函数和 ECR 映像，这样就不会为账户中未使用的资源付费。

- 右键单击位于 AWS 各区服务浏览器中 AWS Lambda 节点下的已部署实例，即可删除函数。
- 可以在 AWS 各区服务浏览器中的 Amazon Elastic Container Service -> 存储库下删除存储库。

后续步骤

有关创建和测试 Lambda 映像的信息，请参阅使用 Lambda 容器映像。

有关容器映像部署、权限和覆盖配置设置的信息，请参阅配置函数。

教程：使用以下方法构建和测试无服务器应用程序 AWS Lambda

您可以使用模板构建无服务器 Lambda 应用程序。 AWS Toolkit for Visual Studio Lambda 项目模板包括一个用于AWS 无服务器应用程序的模板，即AWS 无服务器应用程序模型 (S AM) 的 AWS Toolkit for Visual Studio 实现。AWS 使用此项目类型，您可以开发一组 AWS Lambda 函数，并使用任何必要的 AWS 资源将它们作为整个应用程序进行部署， AWS CloudFormation 用于协调部署。

有关设置的先决条件和信息 AWS Toolkit for Visual Studio，请参阅使用 Visual Stu AWS dio AWS 工具包中的 Lambda 模板。

主题

- 创建新的 AWS 无服务器应用程序项目
- 查看无服务器应用程序文件
- 部署无服务器应用程序
- 测试无服务器应用程序

创建新的 AWS 无服务器应用程序项目

AWS 无服务器应用程序项目使用无服务器模板创建 Lambda 函数。 CloudFormation CloudFormation 模板使您能够定义其他资源，例如数据库、添加 IAM 角色和一次部署多个函数。这与 AWS Lambda 项目不同，后者侧重于开发和部署单个 Lambda 函数。

以下过程介绍了如何创建新的 AWS 无服务器应用程序项目。

1. 在 Visual Studio 中，展开文件菜单，展开新建，然后选择项目。
2. 在新建项目对话框中，确保将语言、平台和项目类型下拉框都设置为“全部...”，然后在搜索字段中键入 **`aws lambda`**。
3. 选择带测试的AWS 无服务器应用程序（.NET Core - C#）模板。

> **Note**
>
> 带测试的AWS 无服务器应用程序（.NET Core - C＃）模板可能不会填充到结果的顶部。

4. 单击下一步打开配置您的新项目对话框。
5. 在配置您的新项目对话框中，为名称输入 **`ServerlessPowertools`**，然后根据您的偏好填写其余字段。选择创建按钮进入选择蓝图对话框。
6. 在选择蓝图对话框中，选择 Powertools for AWS Lambda 蓝图，然后选择完成以创建 Visual Studio 项目。

查看无服务器应用程序文件

以下各节详细介绍为您的项目创建的三个无服务器应用程序文件：

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

1. serverless.template

`serverless.template`文件是用于声明您的无服务器函数和其他 AWS 资源的 AWS CloudFormation 模板。此项目附带的该文件中包含单个 Lambda 函数的声明，该函数将作为一项 HTTP `*Get*` 操作通过 Amazon API Gateway 公开。您可以编辑此模板以自定义现有函数或添加应用程序所需的更多函数和其他资源。

以下是 serverless.template 文件的示例：

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

请注意，许多 `...AWS:: Serverless::Function...` 声明字段都与 Lambda 项目部署的字段类似。Powertools 日志、指标和跟踪是通过以下环境变量配置的：

- POWERTOOLS 服务名称= ServerlessGreeting
- POWERTOOLS_LOG_LEVEL=Info
- POWERTOOLS_LOGGER_C PascalCase
- POWERTOOLS_TRACER_CAPTURE_RESPONSE=true
- POWERTOOLS_TRACER_CAPTURE_ERROR=true
- POWERTOOLS_METRICS_NAMESPAC ServerlessGreeting

有关环境变量的定义和其他详细信息，请参阅 Powertools 供 AWS Lambda 参考网站。

2. Functions.cs

`Functions.cs` 是一个包含 C# 方法的类文件，该方法映射到模板文件中声明的单个函数。该 Lambda 函数响应来自 API Gateway 的 `HTTP Get` 方法。以下是 `Functions.cs` 文件的示例：

```
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提供了 Visual Studio 内部 AWS 部署向导的默认值以及添加到.NET Core CLI 中的 AWS Lambda 命令。以下是本项目附带的 aws-lambda-tools-defaults.json 文件的示例：

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

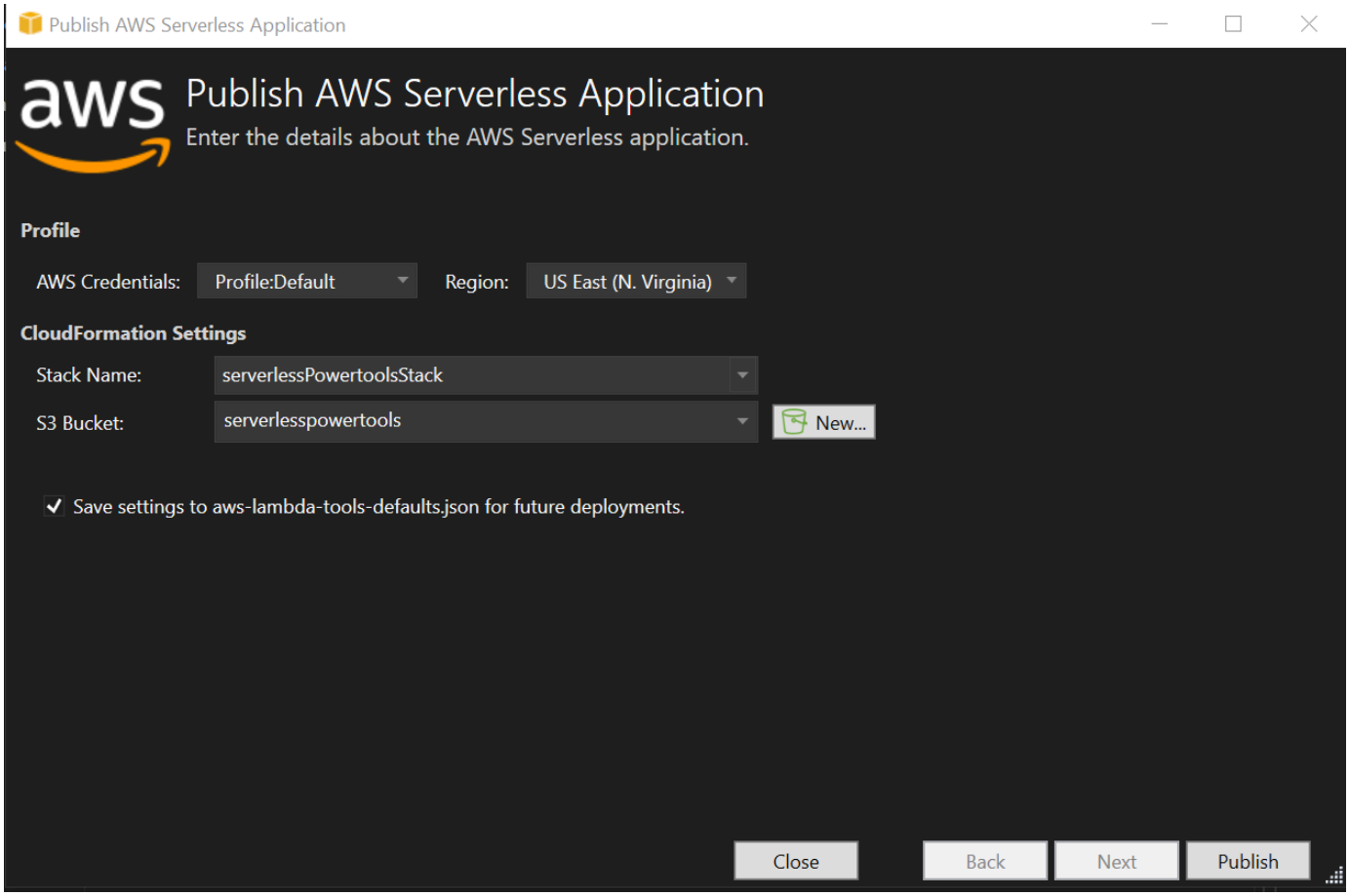
部署无服务器应用程序

要部署无服务器应用程序，请完成以下步骤：

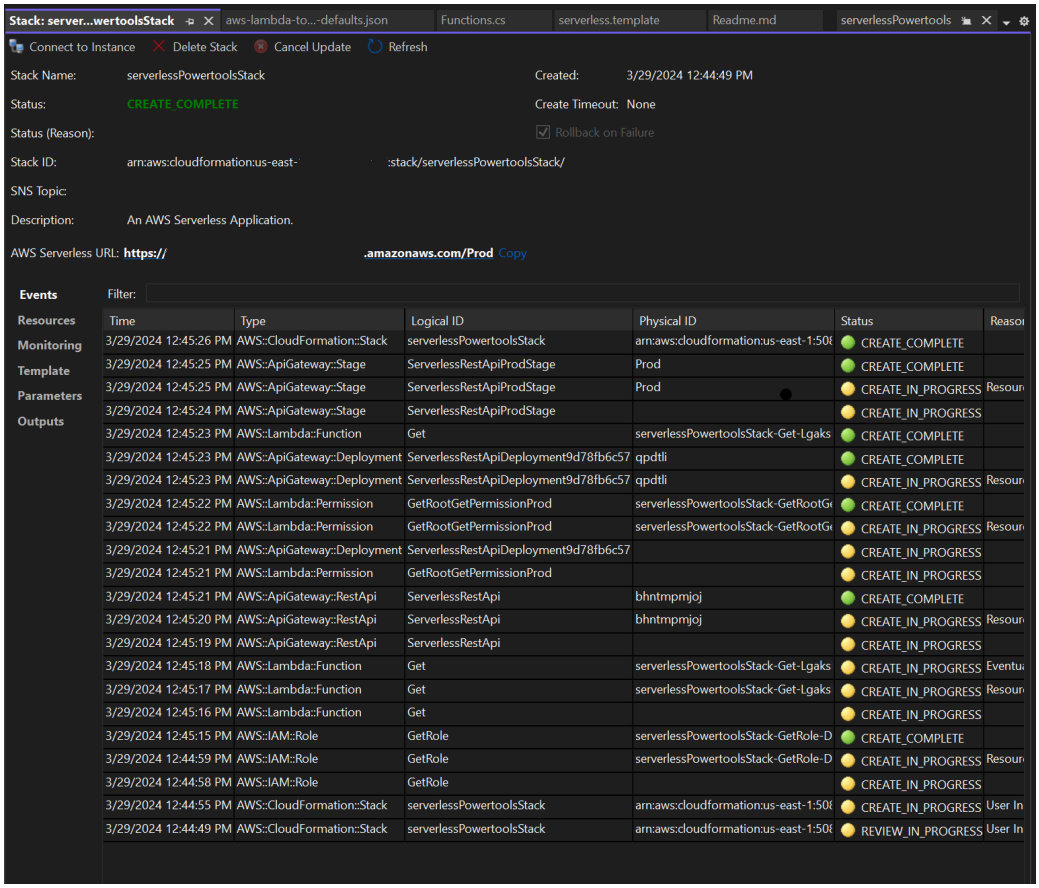
1. 在解决方案资源管理器中，打开项目的快捷菜单（右键单击），然后选择 “发布到 AWS Lambda” 以打开 “发布 AWS 无服务器应用程序” 对话框。

2. 在 “发布 AWS 无服务器应用程序” 对话框中，在 CloudFormation 堆栈名称字段中输入堆栈容器的名称。
3. 在 S3 存储桶字段中，选择您的应用程序捆绑包将上传到的 Amazon S3 存储桶，或者选择新建...按钮，输入新 Amazon S3 存储桶的名称。然后选择发布以进行发布，来部署您的应用程序。

> **Note**
>
> 您的 CloudFormation 堆栈和 Amazon S3 存储桶必须位于同一 AWS 区域。您的项目的其余设置在 `serverless.template` 文件中定义。



4. 在发布过程中，堆栈视图窗口会打开，当部署完成后，状态字段将显示：CREATE_COMPLETE。



测试无服务器应用程序

堆栈创建完成后，您可以使用 AWS 无服务器 URL 查看您的应用程序。如果您在完成本教程时未添加任何其他函数或参数，则访问您的 AWS 无服务器 URL 会在您的网络浏览器中显示以下短语：`Hello Powertools for AWS Lambda (.NET)`。

教程：创建 Amazon Rekognition Lambda 应用程序

本教程向您说明如何创建 Lambda 应用程序，该应用程序使用 Amazon Rekognition 标记包含检测到的标签的 Amazon S3 对象。

有关设置的先决条件和信息 AWS Toolkit for Visual Studio，请参阅使用 Visual Stu AWS dio AWS 工具包中的 Lambda 模板。

创建 Visual Studio .NET Core Lambda Image Rekognition 项目

以下过程描述如何从 AWS Toolkit for Visual Studio创建 Amazon Rekognition Lambda 应用程序。

> **Note**
>
> 创建完成后，您的应用程序会生成一个包含两个项目的解决方案：一个是包含要部署到 Lambda 的 Lambda 函数代码的源项目，另一个是使用 xUnit 来本地测试函数的测试项目。有时 Visual Studio 无法找到你的项目的所有 NuGet 参考资料。这是因为蓝图需要必须从中 NuGet检索的依赖关系。创建新项目时，Visual Studio 仅从中提取本地引用，而不会从 NuGet 中提取远程引用。修复 NuGet 错误：右键单击您的参考文献，然后选择 “还原包”。

1. 在 Visual Studio 中，展开文件菜单，展开新建，然后选择项目。
2. 在新建项目对话框中，确保将语言、平台和项目类型下拉框都设置为“全部...”，然后在搜索字段中键入 **`aws lambda`**。
3. 选择带测试的AWS Lambda （ .NET Core – C # ）模板。
4. 单击下一步打开配置您的新项目对话框。
5. 在 “配置您的新项目” 对话框中，输入 ImageRekognition “” 作为名称，然后根据自己的喜好填写其余字段。选择创建按钮进入选择蓝图对话框。
6. 在选择蓝图对话框上，选择删除映像标签蓝图，然后选择完成以创建 Visual Studio 项目。

> **Note**
>
> 此蓝图提供了用于侦听 Amazon S3 事件的代码，并使用 Amazon Rekognition 检测标签并将其作为标记添加到 S3 对象。

审查项目文件

以下各节分析这些项目文件：

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

1. Function.cs

在 `Function.cs` 文件中，第一段代码是位于文件顶部的程序集属性。默认情况下，Lambda 只接受 `System.IO.Stream` 类型的输入参数和返回类型。您必须注册一个序列化程序才能使用类型化类作为

输入参数和返回类型。该程序集属性注册了 Lambda JSON 序列化程序，它使用 Newtonsoft.Json 将流转换为类型化类。您可以在程序集或方法级别设置串行器。

以下是程序集属性的一个示例：

```
// 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
```

该类有两个构造函数。第一个是 Lambda 调用您的函数时使用的默认构造函数。此构造函数创建 Amazon S3 和 Amazon Rekognition 服务客户端。构造函数还会从您在部署函数时分配给该函数的 IAM 角色检索这些客户端的 AWS 证书。客户端的 AWS 区域设置为运行 Lambda 函数的区域。在此蓝图中，您只想在 Amazon Rekognition 服务对标签具有最低置信度时，才将标签添加到 Amazon S3 对象。此构造函数将检查环境变量 MinConfidence 以确定可接受的置信度级别。您可以在部署 Lambda 函数时设置该环境变量。

以下是 Function.cs 中第一个类构造函数的示例：

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

以下示例演示如何利用第二个构造函数进行测试。测试项目会配置它自己的 S3 和 Rekognition 客户端并将其传入：

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

以下是 Function.cs 文件内的 FunctionHandler 方法示例。

```
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 是 Lambda 构建实例后调用的方法。请注意，输入参数的类型是 S3Event，而不是 Stream。您可以执行此操作，因为您已注册 Lambda JSON 串行器。S3Event 包含在 Amazon S3 中触发的事件的所有信息。该函数将遍历组成事件的所有 S3 对象并让 Rekognition 检测标签。在检测标签后，标签将作为标记添加到 S3 对象。

> **Note**
>
> 代码中包含对 Console.WriteLine() 的调用。当该函数在 Lambda 中运行时，所有调用都Console.WriteLine()将重定向到 Ama CloudWatch zon 日志。

2。 aws-lambda-tools-defaults.json

`aws-lambda-tools-defaults.json` 文件包含由蓝图设置的默认值，用于预填充部署向导中的某些字段。它也有助于设置与 .NET Core CLI 集成的命令行选项。

要访问 .NET Core CLI 集成，请导航到该函数的项目目录并键入 **`dotnet lambda help`**。

> **Note**
>
> 函数处理程序指明了 Lambda 在响应函数调用时要调用的方法。此字段的格式为：`<assembly-name>::<full-type-name>::<method-name>`。类型名称必须包含命名空间。

部署函数

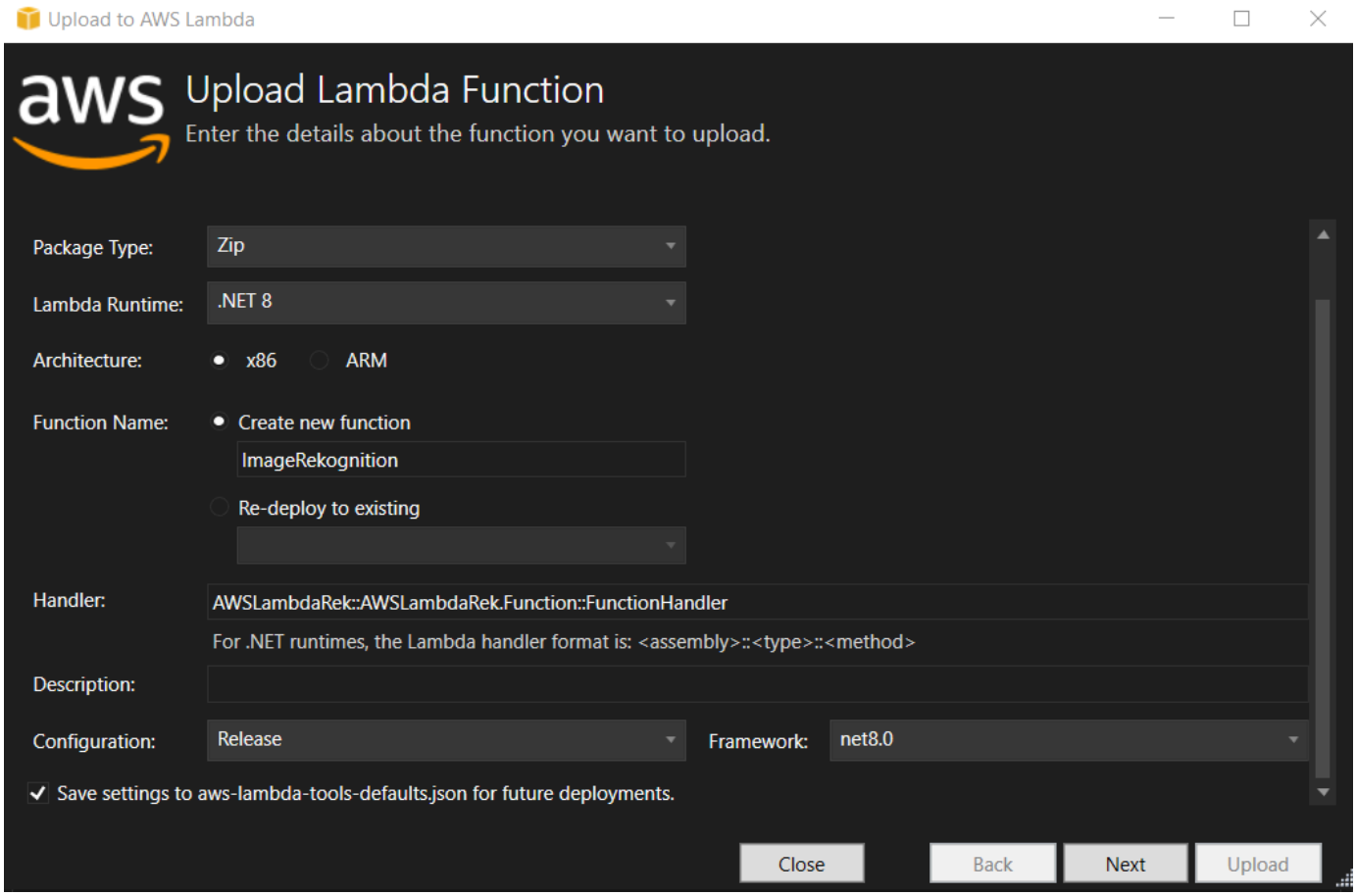
以下过程介绍如何部署您的 Lambda 函数。

1. 在解决方案资源管理器中，右键单击 Lambda 项目，然后选择 “发布到 Lambda” 以打开 “ AWS 上传到” 窗口。 AWS Lambda

   > **Note**
   >
   > 预设值是从 `aws-lambda-tools-defaults.json` 文件中检索的。

2. 从上传到 AWS Lambda 窗口中，在函数名称字段中输入名称，然后选择下一步按钮进入高级函数详细信息窗口。

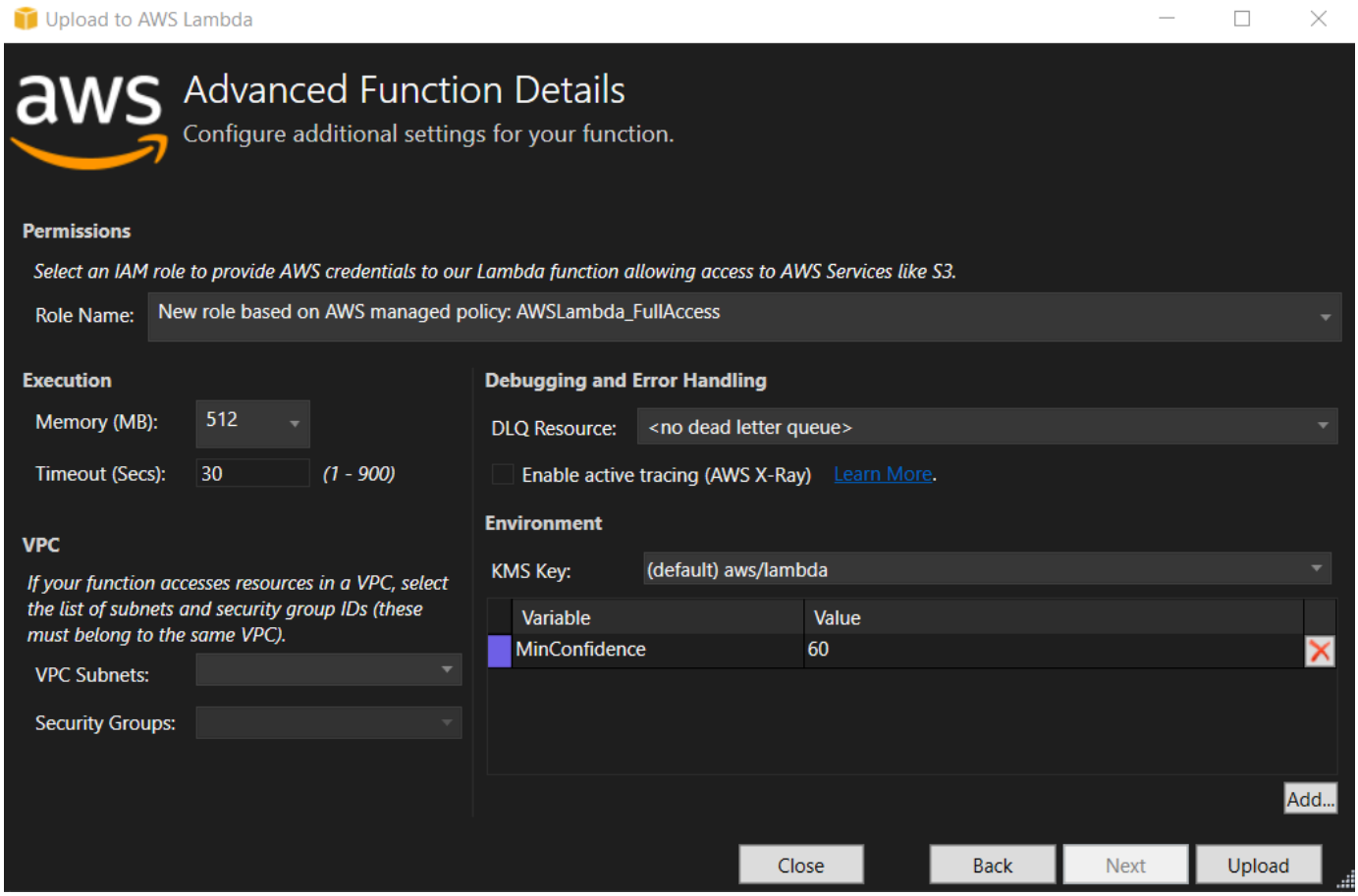
   > **Note**
   >
   > 该示例使用函数名称 **`ImageRekognition`**。



3. 从高级函数详细信息窗口，选择一个 IAM 角色，该角色授予您的代码访问 Amazon S3 和 Amazon Rekognition 资源的权限。

> **Note**
>
> 如果您正在按照此示例进行操作，请选择 `AWSLambda_FullAccess` 角色。

4. 将环境变量 `MinConfidence` 设置为 60，然后选择上传以启动部署过程。当函数视图显示在 AWS Explorer 中时，发布过程完成。



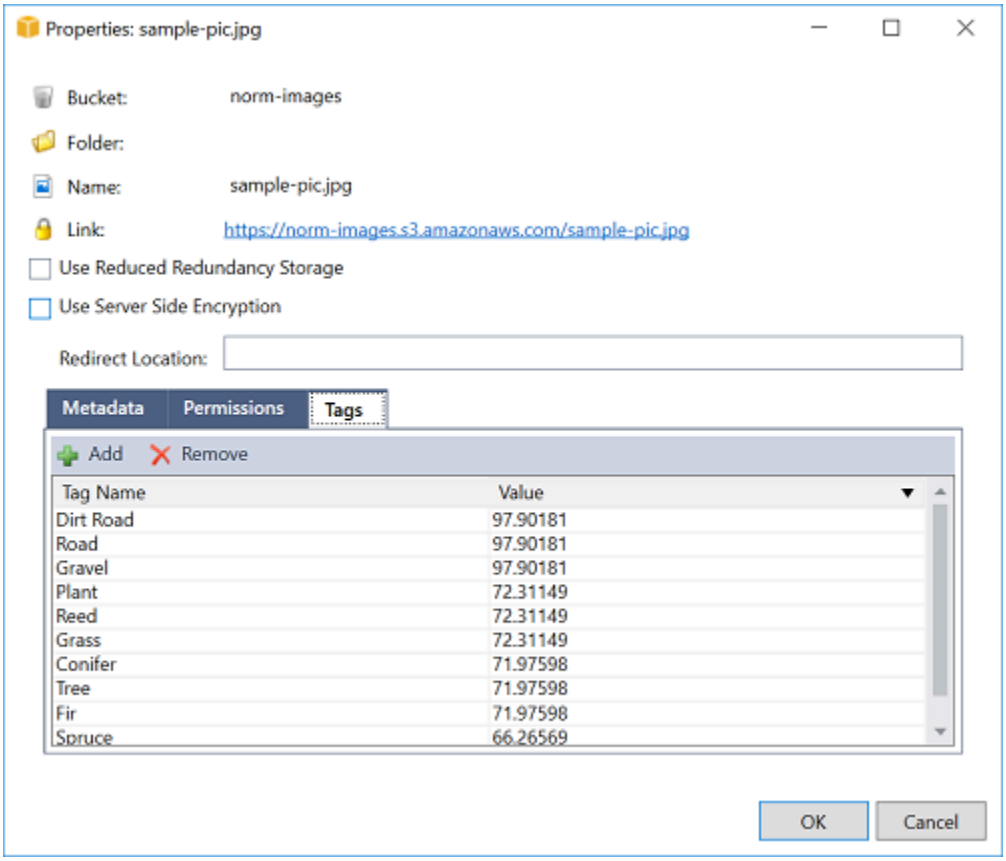
5. 成功部署后，通过导航到事件源选项卡，配置 Amazon S3 将其事件发送到您的新函数。
6. 在事件源选项卡中，选择添加按钮，然后选择要连接您的 Lambda 函数的 Amazon S3 存储桶。

> **Note**
>
> 存储桶必须与您的 Lambda 函数位于同一 AWS 区域。

测试函数。

现在已部署该函数，并将 S3 桶配置为函数的事件源，请从 AWS 各区服务浏览器中为您选定的桶打开 S3 桶浏览器。然后上传一些图像。

上传完成后，您可以通过在函数视图中查看日志来确认您的函数已运行。或者，右键单击存储桶浏览器中的图像，然后选择 Properties (属性)。在 Tags (标签) 选项卡上，您可以查看应用到您的对象的标签。



教程：使用 Amazon 日志框架和 AWS Lambda 创建应用程序日志

您可以使用 Amazon CloudWatch Logs 来监控、存储和访问应用程序的日志。要将日志数据导入 CloudWatch 日志，请使用 S AWS DK 或安装 Log CloudWatch s 代理来监控某些日志文件夹。CloudWatch 日志与几个流行的.NET 日志框架集成，从而简化了工作流程。

要开始使用 CloudWatch 日志和.NET 日志框架，请将相应的 NuGet 包和 CloudWatch 日志输出源添加到您的应用程序中，然后像往常一样使用您的日志库。这使您的应用程序能够使用.NET 框架记录消息，将其发送到 Lo CloudWatch gs，在日志控制台中显示应用程序的 CloudWatch 日志消息。您还可以根据应用程序的日志消息，在 CloudWatch 日志控制台中设置指标和警报。

受支持的 .NET 日志记录框架包括：

- NLog: 要查看，请参阅 nuget.org 软件包。 NLog
- Log4net：若要查看，请访问 nuget.org Log4net 包。
- ASP.NET Core logging Framework：若要查看，请访问 nuget.org ASP.NET Core 日志记录框架包。

以下是一个文件示例，该NLog.config文件通过将AWS.Logger.NLog NuGet 软件包和 AWS 目标添加到中来启用日志和控制台作为日志消息的输出NLog.config。 CloudWatch

```
<?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>
```

日志插件都建立在之上，通过类似于 SDK 的流程对您的 AWS 凭据进行身份验证。 适用于 .NET 的 AWS SDK 以下示例详细说明了日志插件凭据访问 CloudWatch 日志所需的权限：

> **Note**
>
> . AWS NET 日志插件是一个开源项目。有关更多信息、示例和说明，请参阅 Lo AWS gging .NET GitHub 存储库中的示例和说明主题。

JSON

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

```
        "arn:aws:logs:*:*:*"
      ]
    }
  ]
}
```

部署到 AWS

Toolkit for Visual Studio 支持将应用程序部署到 AWS Elastic Beanstalk 容器或 CloudFormation 堆栈。

> **Note**
>
> 如果您使用的是 Visual Studio Express Edition：
>
> - 您可以使用 Docker CLI 将应用程序部署到 Amazon ECS 容器。
> - 可使用 AWS 管理控制台将应用程序部署到 Elastic Beanstalk 容器。
>
> 对于 Elastic Beanstalk 部署，您必须先创建一个 Web 部署包。有关更多信息，请参阅如何：在 Visual Studio 中创建 Web 部署包。对于 Amazon ECS 部署，您必须拥有一个 Docker 映像。有关更多信息，请参阅适用于 Docker 的 Visual Studio 工具。

主题

在 Visual Studio 中使用“发布到 AWS”

发布到 AWS 是一种交互式部署体验，可协助您将 .NET 应用程序发布到 AWS 部署目标，该体验支持以 .NET Core 3.1 及更高版本为目标的应用程序。使用“发布到 AWS”可以直接从 IDE 中提供以下部署功能，从而将工作流保持在 Visual Studio 内：

- 只需单击一下即可部署应用程序。
- 基于应用程序提供部署建议。
- 根据部署目标环境（部署目标），自动创建相关且符合要求的 Dockerfile。
- 根据部署目标的要求优化了用于构建和打包应用程序的设置。

> **Note**
>
> 有关发布 .NET Framework 应用程序的其他信息，请参阅在 Elastic Beanstalk 上创建和部署 .NET 应用程序指南
> 您也可以从 .NET CLI 访问“发布到 AWS”。有关更多信息，请参阅在 AWS 上部署 .NET 应用程序指南。

主题

先决条件

要成功将 .NET 应用程序发布到 AWS 服务，请在本地设备上安装以下内容：

- .NET Core 3.1 或更高版本（包括 .NET5 和 .NET6）：有关这些产品的更多信息和下载信息，请访问 Microsoft 下载网站。
- Node.js 14.x 或更高版本：需要 Node.js 才能运行 AWS Cloud Development Kit (AWS CDK)。要下载 Node.js 或获取有关 Node.js 的更多信息，请访问 Node.js 下载网站。

  > **Note**
  >
  > “发布到 AWS”利用 AWS CDK 将应用程序及其所有部署基础设施作为单个项目进行部署。有关 AWS CDK 的更多信息，请参阅云开发工具包指南。

- （可选）在部署到基于容器的服务（例如 Amazon ECS）时使用 Docker。如需了解更多信息并下载 Docker，请访问 Docker 下载网站。

支持的应用程序类型

在发布到新的或现有的目标之前，请先在 Visual Studio 中创建或打开以下项目类型之一：

- ASP.NET Core 应用程序
- .NET 控制台应用程序

- Blazor WebAssembly 应用程序

将应用程序发布到 AWS 目标

发布到新目标时，“发布到 AWS”将通过提出建议和使用常用设置来指导您完成整个过程。如果您需要发布到之前设置的目标，则您的首选项已被存储，您可以调整首选项或者立即进行一键部署。

> **Note**
>
> 工具包与 .NET CLI 服务器集成：
> 发布操作会在本地主机上启动一个 .NET 服务器进程来执行发布过程。

发布到新目标

以下内容介绍在发布到新目标时如何配置“发布到AWS”部署首选项。

1. 从 AWS Explorer，展开凭证下拉菜单，然后选择与您的部署所需的区域和 AWS 服务相对应的 AWS 配置文件。
2. 展开区域下拉菜单，然后选择包含部署所需 AWS 服务的 AWS 区域。
3. 在 Visual Studio 解决方案浏览器窗格中，打开项目名称的上下文菜单（右键单击），然后选择发布到 AWS 这将打开发布到 AWS。
4. 在发布到 AWS 中，选择发布到新目标以配置新部署。

   > **Note**
   >
   > 要修改默认部署凭证，请在发布到 AWS 中，选择或单击凭证部分旁的编辑链接。
   > 要绕过目标配置过程，请选择发布到现有目标，然后从先前的部署目标列表中选择您的首选配置。

5. 在发布目标窗格中，选择一项用于管理应用程序部署的 AWS 服务。
6. 如果您对配置满意，请选择发布开始部署过程。

   > **Note**
   >
   > 启动部署后，发布到 AWS 会显示以下状态更新：
   >
   > - 在部署过程中，发布到 AWS 显示有关部署进度的信息。

- 部署过程结束后，发布到 AWS 会指示部署是成功还是失败。
- 成功部署后，资源面板将提供有关已创建资源的其他信息。此信息将因应用程序类型和部署配置而异。

发布到现有目标

以下内容介绍如何将 .NET 应用程序重新发布到现有 AWS 目标。

1. 从 AWS Explorer，展开凭证下拉菜单，然后选择与您的部署所需的区域和 AWS 服务相对应的 AWS 配置文件。
2. 展开区域下拉菜单，然后选择包含部署所需 AWS 服务的 AWS 区域。
3. 在 Visual Studio 解决方案浏览器窗格中，右键单击项目名称，然后选择发布到 AWS 以打开发布到 AWS。
4. 在发布到 AWS 中，选择发布到现有目标，然后从现有目标列表中选择您的部署环境。

> **Note**
>
> 如果您最近向 AWS Cloud 发布了任何应用程序，则这些应用程序将显示在“发布到 AWS”中。

5. 选择要将应用程序部署到的发布目标，然后单击发布，开始部署过程。

使用.NET 核心 CLI 部署 AWS Lambda 项目

AWS Toolkit for Visual Studio 包括 Visual Studio 的 AWS Lambda .NET 核心项目模板。您可以使用 .NET Core 命令行界面（CLI）部署 Visual Studio 中构建的 Lambda 函数。

主题

先决条件

在使用.NET Core CLI 部署 Lambda 函数之前，您必须满足以下先决条件：

- 确保安装了 Visual Studio 2015 更新 3。
- 安装了 适用于 Windows 的 .NET Core。
- 将 .NET Core CLI 设置为可与 Lambda 配合使用。有关更多信息，请参阅《AWS Lambda 开发人员指南》中的 .NET Core CLI。
- 安装了 Toolkit for Visual Studio。有关更多信息，请参阅 正在安装 AWS Toolkit for Visual Studio。

相关主题

使用 .NET Core CLI 部署 Lambda 函数时，以下相关主题可能会有所帮助：

- 有关 Lambda 函数的更多信息，请参阅什么是 Lambda AWS ？ 在《AWS Lambda 开发人员指南》中。
- 有关在 Visual Studio 中创建 Lambda 函数的信息，请参阅 AWS Lambda。
- 有关 Microsoft .NET Core 的更多信息，请参阅 Microsoft 在线文档中的 .NET Core。

列出可通过 .NET Core CLI 使用的 Lambda 命令

要列出可通过 .NET Core CLI 使用的 Lambda 命令，请执行以下操作。

1. 打开命令提示符窗口并导航到包含 Visual Studio .NET Core Lambda 项目的文件夹。
2. 输入 `dotnet lambda --help`。

```
C:\Lambda\AWSLambda1\AWSLambda1>dotnet lambda --help AWS Lambda Tools for .NET Core
 functions
    Project Home: https://github.com/aws/aws-lambda-dotnet
    .
    Commands to deploy and manage Lambda functions:
    .
            deploy-function         Deploy the project to Lambda
            invoke-function         Invoke the function in Lambda with an optional
 input
            list-functions          List all of your Lambda functions
            delete-function         Delete a Lambda function
```

```
            get-function-config     Get the current runtime configuration for a Lambda
 function
            update-function-config  Update the runtime configuration for a Lambda
 function
    .
    Commands to deploy and manage AWS serverless applications using AWS CloudFormation:
    .
            deploy-serverless       Deploy an AWS serverless application
            list-serverless         List all of your AWS serverless applications
            delete-serverless       Delete an AWS serverless application
    .
    Other Commands:
    .
            package                 Package a Lambda project into a .zip file ready for
 deployment
    .
    To get help on individual commands, run the following:

            dotnet lambda help <command>
```

从 .NET Core CLI 发布 .NET Core Lambda 项目

以下说明假设你已在 Visual Studio 中创建了一个 AWS Lambda .NET 核心函数。

1. 打开命令提示符窗口并导航到包含 Visual Studio .NET Core Lambda 项目的文件夹。
2. 输入 `dotnet lambda deploy-function`。
3. 当系统提示时，输入要部署的函数的名称。它可以是新名称或现有函数的名称。
4. 出现提示时，输入 AWS 区域（您的 Lambda 函数将部署到的区域）。
5. 当系统提示时，选择或创建 Lambda 将在执行函数时代入的 IAM 角色。

成功完成后，将显示消息 New Lambda function created (新 Lambda 函数已创建)。

```
C:\Lambda\AWSLambda1\AWSLambda1>dotnet lambda deploy-function
Executing publish command
... invoking 'dotnet publish', working folder 'C:\Lambda\AWSLambda1\AWSLambda1\bin
\Release\netcoreapp1.0\publish'
... publish: Publishing AWSLambda1 for .NETCoreApp,Version=v1.0
... publish: Project AWSLambda1 (.NETCoreApp,Version=v1.0) will be compiled because
 expected outputs are missing
... publish: Compiling AWSLambda1 for .NETCoreApp,Version=v1.0
```

```
... publish: Compilation succeeded.
... publish:     0 Warning(s)
... publish:     0 Error(s)
... publish: Time elapsed 00:00:01.2479713
... publish:
... publish: publish: Published to C:\Lambda\AWSLambda1\AWSLambda1\bin\Release
\netcoreapp1.0\publish
... publish: Published 1/1 projects successfully
Zipping publish folder C:\Lambda\AWSLambda1\AWSLambda1\bin\Release
\netcoreapp1.0\publish to C:\Lambda\AWSLambda1\AWSLamb
da1\bin\Release\netcoreapp1.0\AWSLambda1.zip
Enter Function Name: (AWS Lambda function name)
DotNetCoreLambdaTest
Enter AWS Region: (The region to connect to AWS services)
us-west-2
Creating new Lambda function
Select IAM Role that Lambda will assume when executing function:
    1) lambda_exec_LambdaCoreFunction
    2) *** Create new IAM Role ***
1
New Lambda function created
```

如果您部署现有函数，则部署函数仅要求提供 AWS 区域。

```
C:\Lambda\AWSLambda1\AWSLambda1>dotnet lambda deploy-function
Executing publish command
Deleted previous publish folder
... invoking 'dotnet publish', working folder 'C:\Lambda\AWSLambda1\AWSLambda1\bin
\Release\netcoreapp1.0\publish'
... publish: Publishing AWSLambda1 for .NETCoreApp,Version=v1.0
... publish: Project AWSLambda1 (.NETCoreApp,Version=v1.0) was previously compiled.
 Skipping compilation.
... publish: publish: Published to C:\Lambda\AWSLambda1\AWSLambda1\bin\Release
\netcoreapp1.0\publish
... publish: Published 1/1 projects successfully
Zipping publish folder C:\Lambda\AWSLambda1\AWSLambda1\bin\Release
\netcoreapp1.0\publish to C:\Lambda\AWSLambda1\AWSLamb
da1\bin\Release\netcoreapp1.0\AWSLambda1.zip
Enter Function Name: (AWS Lambda function name)
DotNetCoreLambdaTest
Enter AWS Region: (The region to connect to AWS services)
us-west-2
Updating code for existing function
```

在部署 Lambda 函数后，便可使用该函数。有关更多信息，请参阅如何使用 AWS Lambda 的示例。

Lambda 会自动为您监控 Lambda 函数，并通过亚马逊报告指标。 CloudWatch要监控您的 Lambda 函数并对其进行故障排除，请参阅使用亚马逊对 Lambda AWS 函数进行故障排除和监控。 CloudWatch

使用适用 AWS Elastic Beanstalk 于 Visual Studio 的 AWS Toolkit 和 Amazon Q 部署到 Visual Studio 中

AWS Elastic Beanstalk 是一项服务，可简化为应用程序配置 AWS 资源的过程。Elastic Beanstalk 提供了部署应用程序所需 AWS 的所有基础架构。此基础设施包括：

- 一些 Amazon EC2 实例，可托管应用程序的可执行文件和内容。
- 一个自动扩缩组，可维持适当数量的 Amazon EC2 实例来支持您的应用程序。
- 一个 Elastic Load Balancing 负载均衡器，可将传入流量路由至具有最大带宽的 Amazon EC2 实例。

本用户指南主题介绍如何在 Amazon Q 上使用工具包中的 AWS Elastic Beanstalk 向导。有关 Elastic Beanstalk 的详细信息，请参阅开发者指南。AWS Elastic Beanstalk以下主题部分描述了 Amazon Q 工具包的 Elastic Beanstalk AWS 向导。

主题

- 将传统的 ASP.NET 应用程序部署到 Elastic Beanstalk
- 将 ASP.NET Core 应用程序部署到 Elastic Beanstalk（旧版）
- 如何为您的应用程序指定 AWS 安全证书
- 如何将应用程序重新发布到 Elastic Beanstalk 环境（旧版）
- 自定义 Elastic Beanstalk 应用程序部署
- 自定义 ASP.NET Core Elastic Beanstalk 部署
- 对 .NET 和 Elastic Beanstalk 的多应用程序支持

将传统的 ASP.NET 应用程序部署到 Elastic Beanstalk

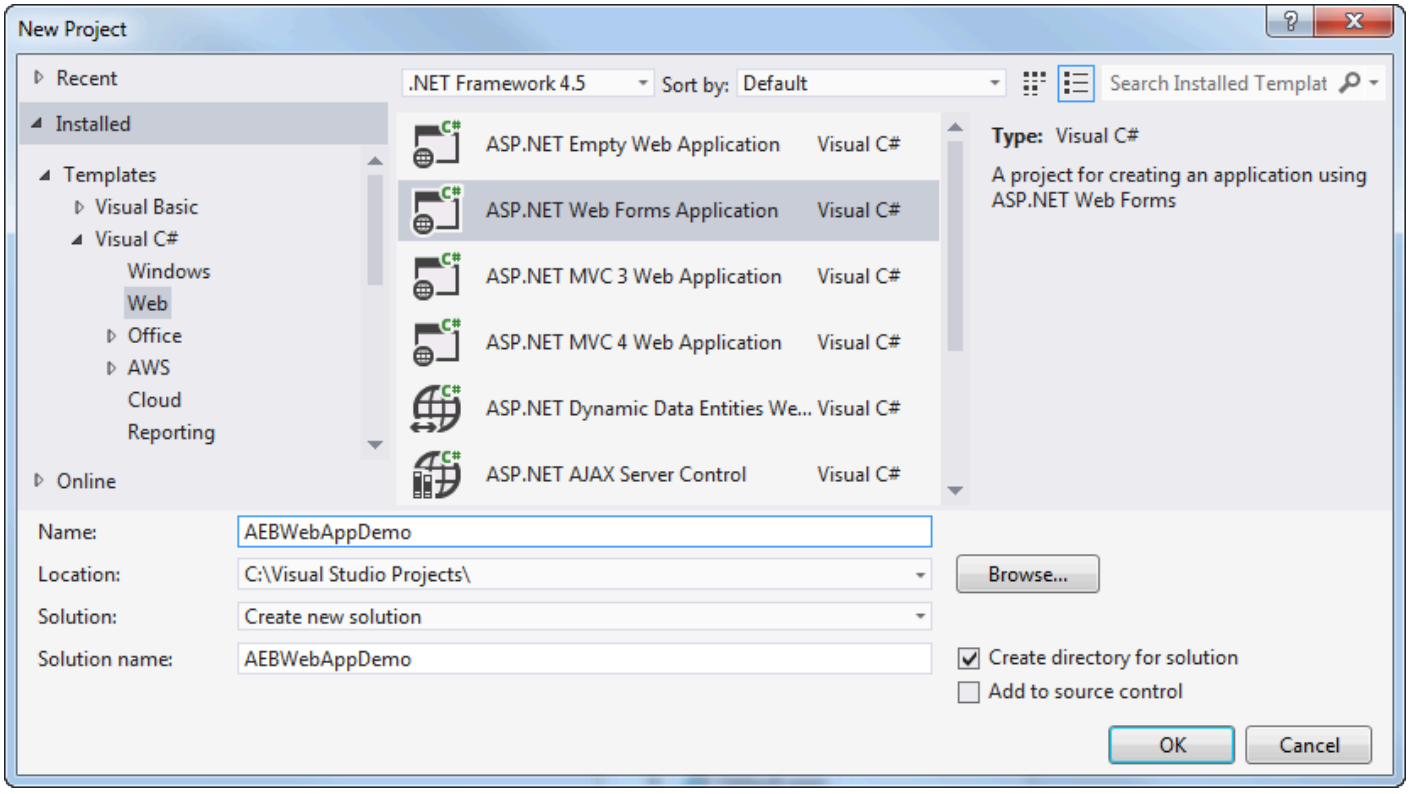
本部分介绍如何使用发布到 Elastic Beanstalk 向导（作为 Toolkit for Visual Studio 的一部分提供）通过 Elastic Beanstalk 部署应用程序。要进行练习，您可使用 Visual Studio 中内置的 Web 应用程序初学者项目的实例，也可使用您自己的项目。

> **Note**
>
> 此向导还支持部署 ASP.NET 内核应用程序。有关 ASP.NET Core 的信息，请参阅《AWS .NET deployment tool》指南和更新的部署到 AWS 目录。

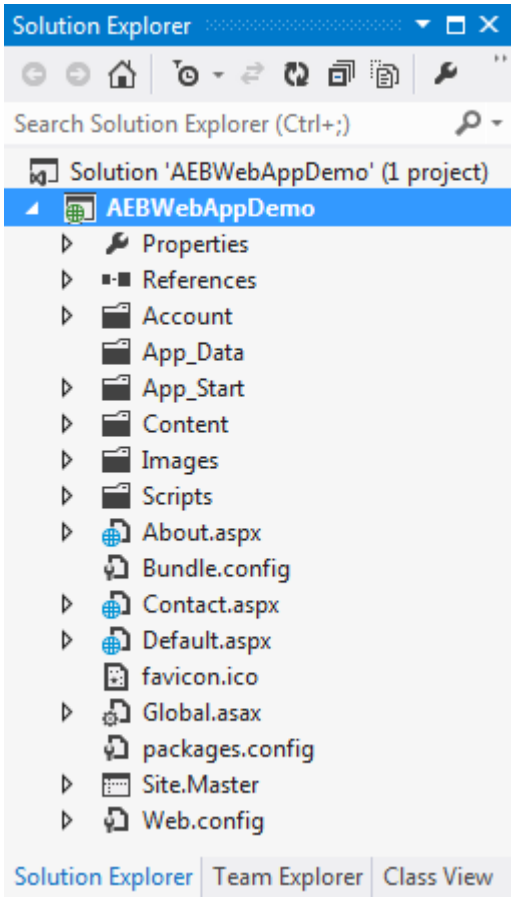
> **Note**
>
> 在使用发布到 Elastic Beanstalk 向导之前，必须先下载并安装 Web Deploy。此向导依赖 Web Deploy 将 Web 应用程序和网站部署到 Internet Information Services (IIS) Web 服务器。

创建示例 Web 应用程序初学者项目

1. 在 Visual Studio 中，从 File (文件) 菜单中，选择 New (新建)，然后选择 Project (项目)。
2. 在 New Project (新建项目) 对话框的导航窗格中，依次展开 Installed (已安装)、Templates (模板) 和 Visual C#，然后选择 Web。
3. 在 Web 项目模板的列表中，选择其说明中包含 `Web` 和 `Application` 字样的任何模板。在本示例中，请选择 ASP.NET Web Forms Application (ASP.NET Web 表单应用程序)。

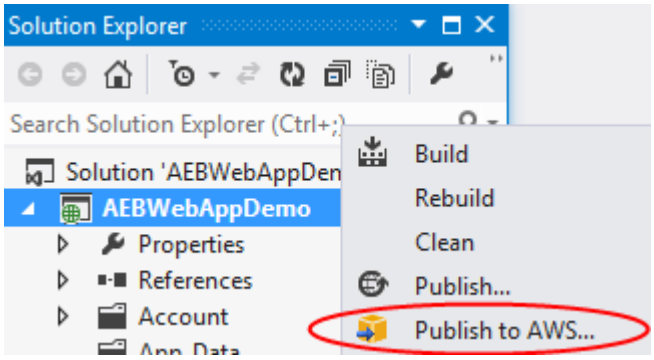


4. 在 Name (名称) 框中，键入 AEBWebAppDemo。
5. 在 Location (位置) 框中，键入您的开发计算机上的解决方案文件夹的路径或选择 Browse (浏览)，然后浏览并选择解决方案文件夹，再选择 Select Folder (选择文件夹)。
6. 确认选中了 Create directory for solution (为解决方案创建目录) 框。在 Solution (解决方案) 下拉列表中，确认选择了 Create new solution (创建新解决方案)，然后选择 OK (确定)。Visual Studio 将基于 ASP.NET Web 表单应用程序项目模板创建解决方案和项目。随后，Visual Studio 将显示解决方案资源管理器，其中将显示新的解决方案和项目。

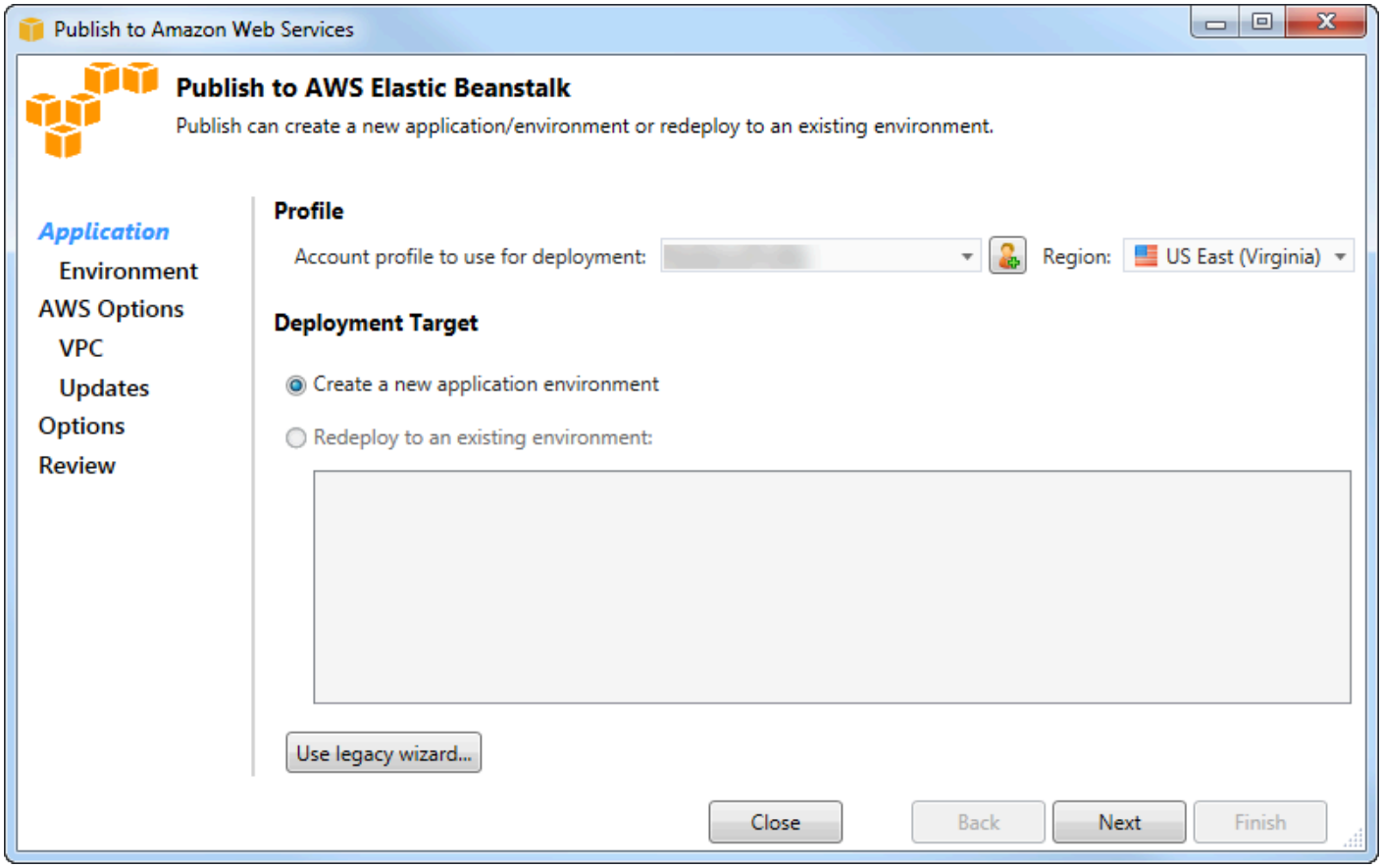


使用 Publish to Elastic Beanstalk 向导部署应用程序

1. 在解决方案资源管理器中，打开您在上一节中创建的AEBWebAppDemo项目的项目文件夹的上下文（右键单击）菜单，或者打开您自己的应用程序的项目文件夹的快捷菜单，然后选择发布到 E AWS lastic Beanstalk。



随即显示 Publish to Elastic Beanstalk (发布到 Elastic Beanstalk) 向导。



2. 在配置文件中，从用于部署的账户配置文件下拉列表中，选择要用于部署的 AWS 账户配置文件。

   或者，如果您有要使用的 AWS 帐户，但尚未为其创建 AWS 账户资料，则可以选择带有加号 (+) 的按钮来添加 AWS 账户资料。

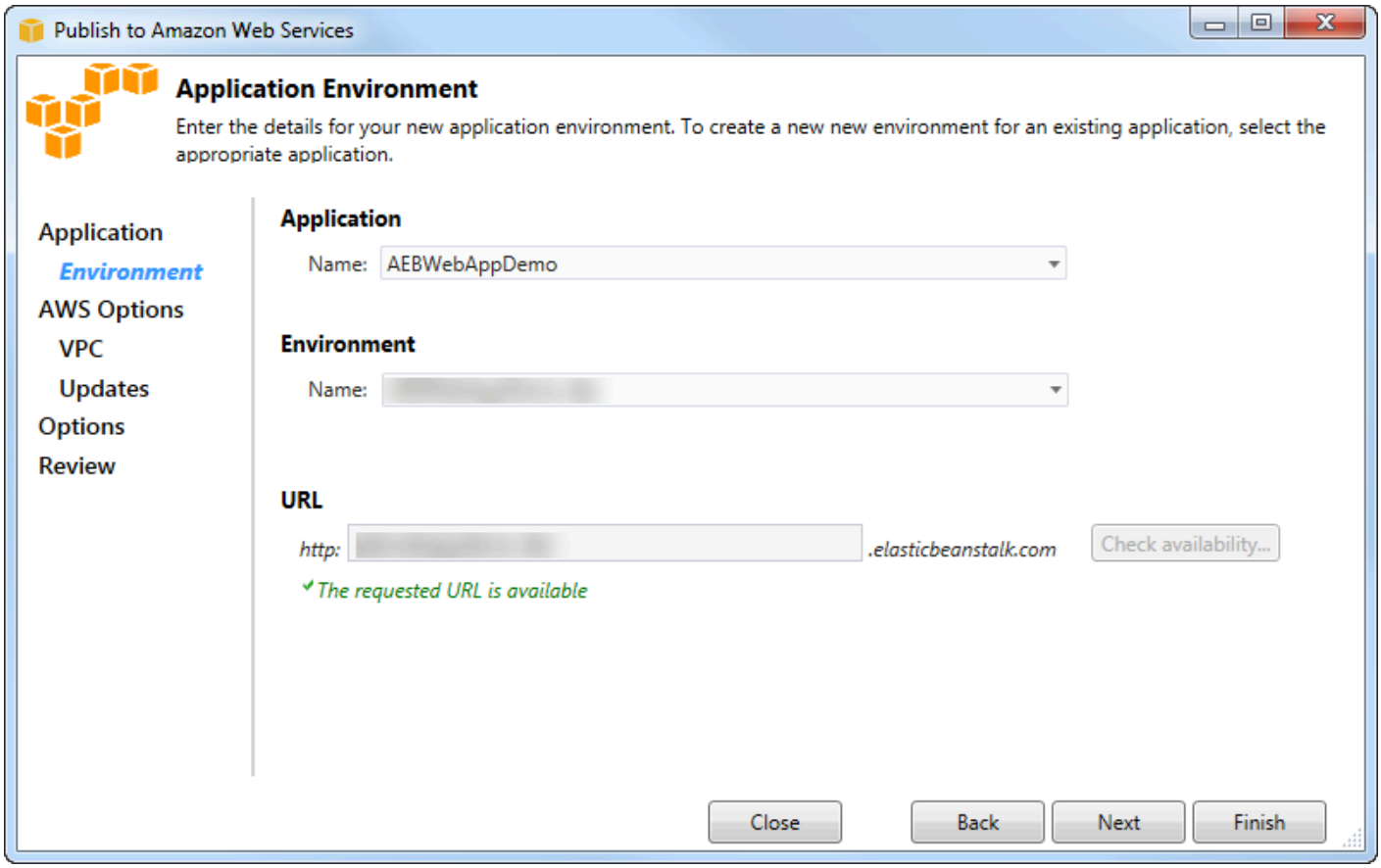
3. 从区域下拉列表中，选择希望 Elastic Beanstalk 将应用程序部署到的区域。

4. 在 Deployment Target (部署目标) 中，您可选择 Create a new application environment (创建新应用程序环境) 执行应用程序的初始部署或选择 Redeploy to an existing environment (重新部署到现有环境) 重新部署之前已部署的应用程序。（之前的部署可能是使用向导或已弃用的独立部署工具执行

的。）如果您选择 Redeploy to an existing environment (重新部署到现有环境)，则当向导从当前正在运行的之前的部署中检索信息时可能会出现延迟。

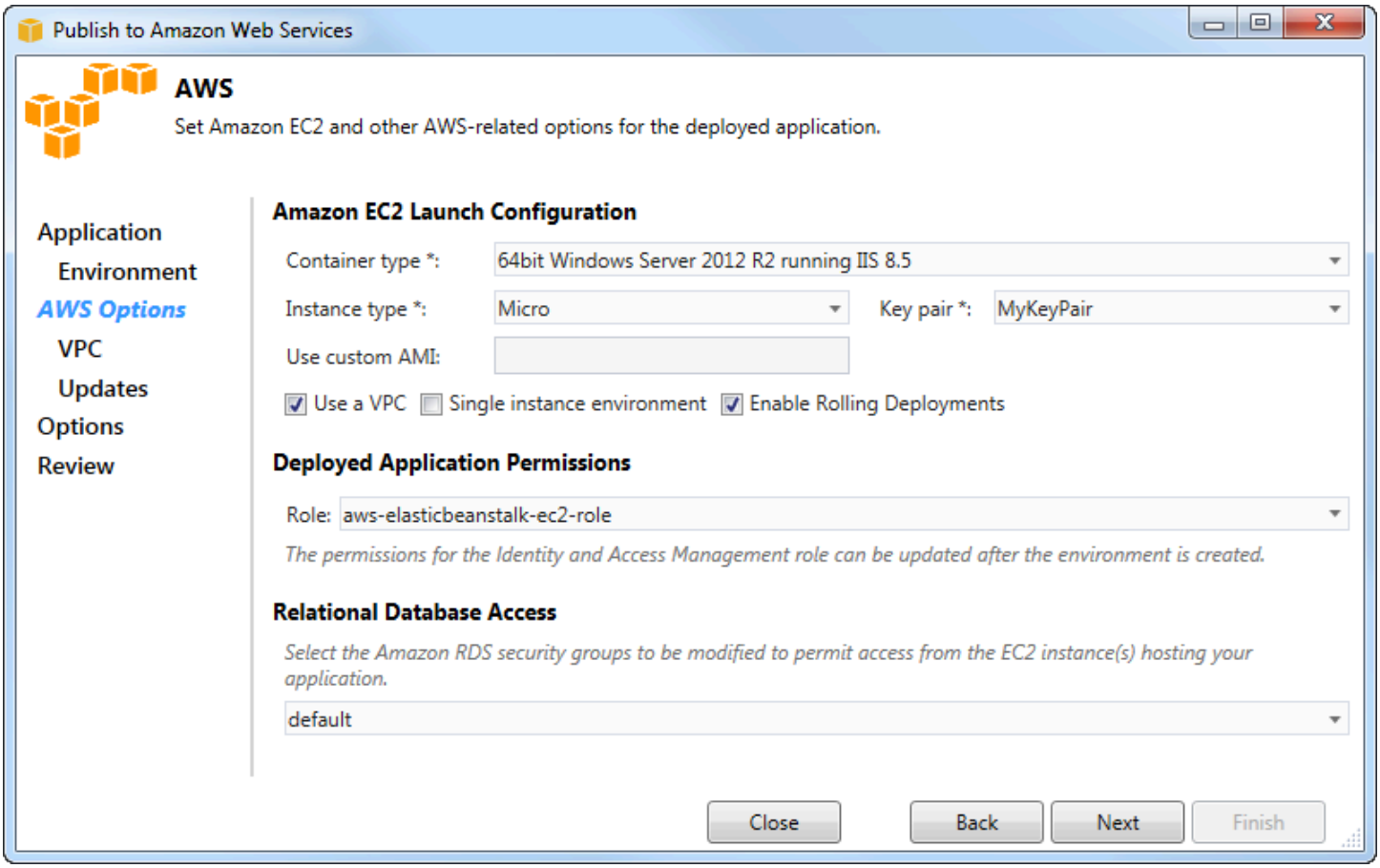
> **Note**
>
> 如果您选择 Redeploy to an existing environment (重新部署到现有环境)，再选择列表中的环境，然后选择 Next (下一步)，则向导会将您定向至 Application Options (应用程序选项) 页面。如果您执行此过程，请向前跳至此部分中后面描述如何使用 Application Options (应用程序选项) 页面的说明。

5. 选择下一步。



6. 在 Application Environment (应用程序环境) 页面上的 Application (应用程序) 区域中，Name (名称) 下拉列表将为应用程序建议默认名称。您可通过选择此下拉列表中的其他名称来更改默认名称。
7. 在环境区域的名称下拉列表中，为您的 Elastic Beanstalk 环境键入一个名称。在此上下文中，术语环境是指 Elastic Beanstalk 为您的应用程序预置的基础设施。此下拉列表中可能已建议默认名称。如果未建议默认名称，您可键入一个名称或从下拉列表中选择一个名称（如果提供了任何其他名称）。环境名称的长度不得超过 23 个字符。

8. 在 URL 区域中，此框会建议将作为您的 Web 应用程序 URL 的默认子域 `.elasticbeanstalk.com`。您可键入新的子域名来更改默认子域。
9. 选择 Check availability (检查可用性) 以确保您的 Web 应用程序 URL 未在使用中。
10. 如果您的 Web 应用程序 URL 可以使用，请选择 Next (下一步)。



1. 在 AWS 选项页面上的 Amazon EC2 启动配置中，从容器类型下拉列表中，选择将用于您的应用程序的亚马逊机器映像（AMI）类型。
2. 在实例类型下拉列表中，指定要使用的 Amazon EC2 实例类型。在本示例中，我们建议您使用 Micro (微型)。这将最大程度降低相关的实例运行成本。有关 Amazon EC2 成本的更多信息，请转至 EC2 定价页面。
3. 在密钥对下拉列表中，选择一个 Amazon EC2 实例密钥对，用于登录您的应用程序将使用的实例。
4. （可选）在 Use custom AMI (使用自定义 AMI) 框中，您可指定将覆盖 Container type (容器类型) 下拉列表中指定的 AMI 的自定义 AMI。有关如何创建自定义 AMI 的更多信息，请转到 Elasti AWS c Beanstalk 开发人员指南 AMIs中的使用自定义，然后从 Amazon EC 2 实例创建 AMI。
5. （可选）如果您要在 VPC 中启动实例，请选中 Use a VPC (使用 VPC) 框。

6. （可选）如果您要启动单个 Amazon EC2 实例，然后将应用程序部署到该实例，请选中单实例环境选项框。

   如果您选中此框，Elastic Beanstalk 仍将创建自动扩缩组，但不会配置改组。如果您希望稍后配置自动扩缩组，则可使用 AWS 管理控制台。

7. （可选）如果您希望控制将应用程序部署到实例时的条件，请选中 Enable Rolling Deployments (启用滚动部署) 框。只能在未选中 Single instance environment (单个实例环境) 框时选中此框。

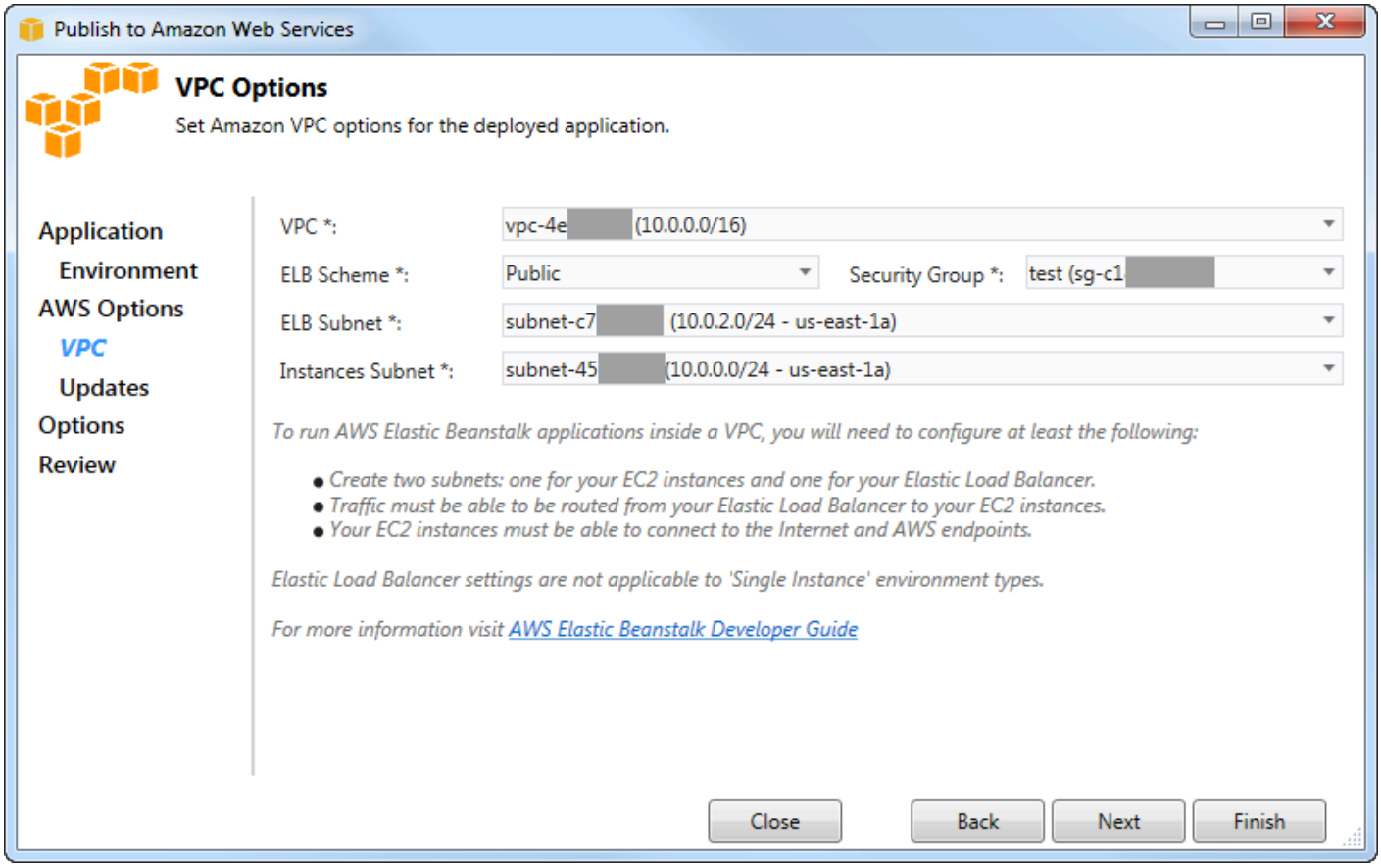
8. 如果您的应用程序使用诸如 Amazon S3 和 DynamoDB 之类的 AWS 服务，则提供证书的最佳方式是使用 IAM 角色。在已部署应用程序权限区域中，您可选择现有 IAM 角色或创建一个供向导用来启动环境的角色。使用的应用程序在向 AWS 服务发出请求时 适用于 .NET 的 AWS SDK 将自动使用此 IAM 角色提供的证书。

9. 如果您的应用程序访问 Amazon RDS 数据库，请在关系数据库访问权限区域的下拉列表中，选中向导将更新的任何 Amazon RDS 安全组旁的框，以便您的 Amazon EC2 实例可访问该数据库。

10选择下一步。

   - 如果您已选择 Use a VPC (使用 VPC)，则将显示 VPC Options (VPC 选项) 页面。
   - 如果您已选择 Enable Rolling Deployments (启用滚动部署)，但未选择 Use a VPC (使用 VPC)，则将显示 Rolling Deployments (滚动部署) 页面。向前跳至此部分中后面描述如何使用 Rolling Deployments (滚动部署) 页面的说明。
   - 如果您未选择 Use a VPC (使用 VPC) 或 Enable Rolling Deployments (启用滚动部署)，则将显示 Application Options (应用程序选项) 页面。向前跳至此部分中后面描述如何使用 Application Options (应用程序选项) 页面的说明。

11如果您已选择 Use a VPC (使用 VPC)，请在 VPC Options (VPC 选项) 页面上指定信息以在 VPC 中启动应用程序。



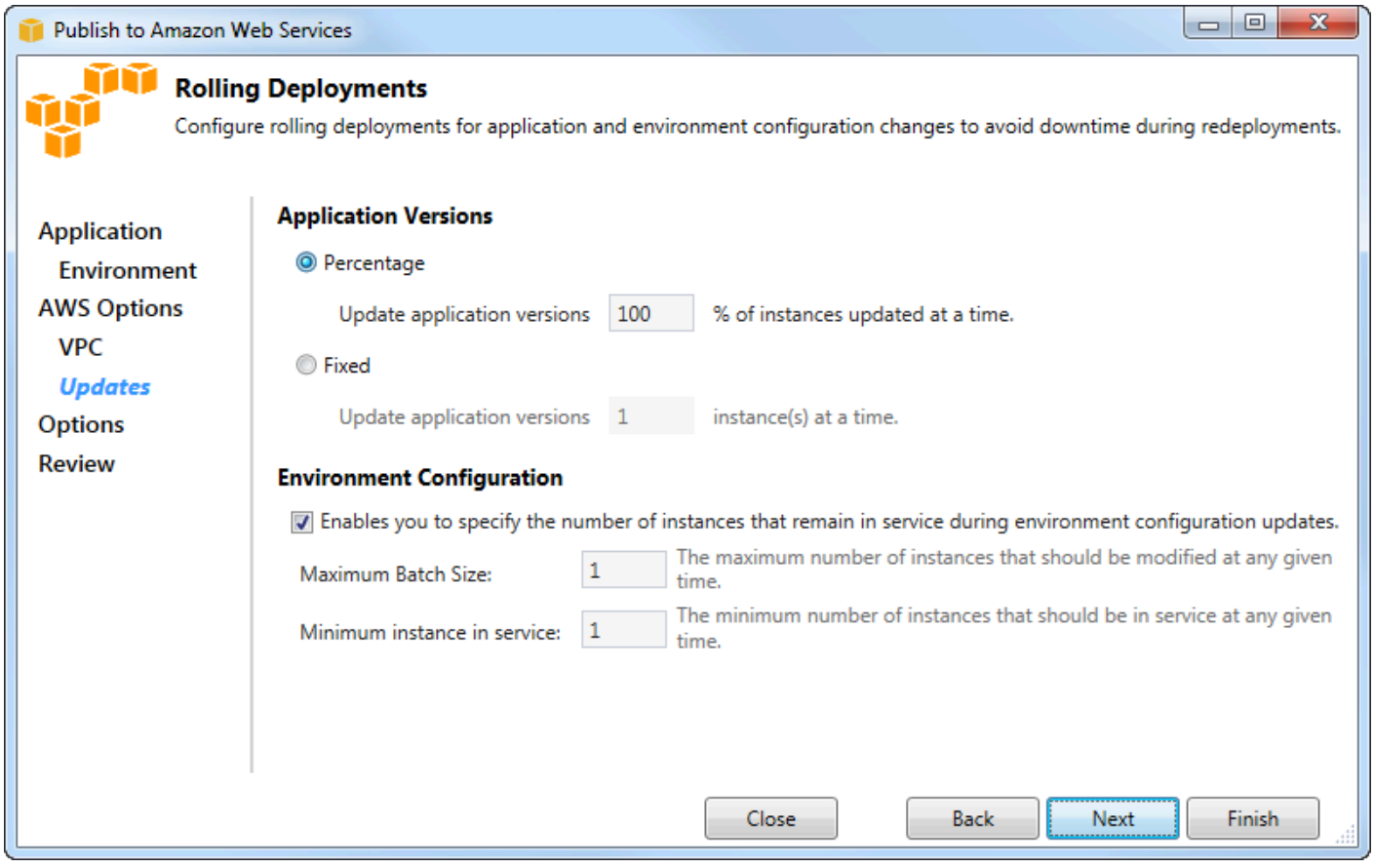
必须已创建 VPC。如果您在 Toolkit for Visual Studio 中创建了 VPC，则 Toolkit for Visual Studio 将为您填充此页面。如果您在 AWS 管理控制台中创建了 VPC，请在此页面中键入有关您的 VPC 的信息。

针对 VPC 的部署的主要注意事项

- 您的 VPC 需要至少一个公有子网和一个私有子网。
- 在 ELB Subnet (ELB 子网) 下拉列表中，指定公有子网。Toolkit for Visual Studio 将应用程序的 Elastic Load Balancing 负载均衡器部署到公有子网。公有子网与具有指向 Internet 网关的入口的路由表关联。您可识别 Internet 网关，因为它具有以 `igw-` 开头的 ID（例如，`igw-83cddaex`）。您使用 Toolkit for Visual Studio 创建的公有子网包含将其标识为公有的标签值。
- 在 Instances Subnet (实例子网) 下拉列表中，指定私有子网。Toolkit for Visual Studio 会将应用程序的 Amazon EC2 实例部署到私有子网。
- 应用程序的 Amazon EC2 实例将通过公有子网中执行网络地址转换（NAT）的 Amazon EC2 实例实现从私有子网到 Internet 的通信。要启用此通信，您需要允许流量从私有子网流至 NAT 实例的 VPC 安全组。在 Security Group (安全组) 下拉列表中，指定此 VPC 安全组。

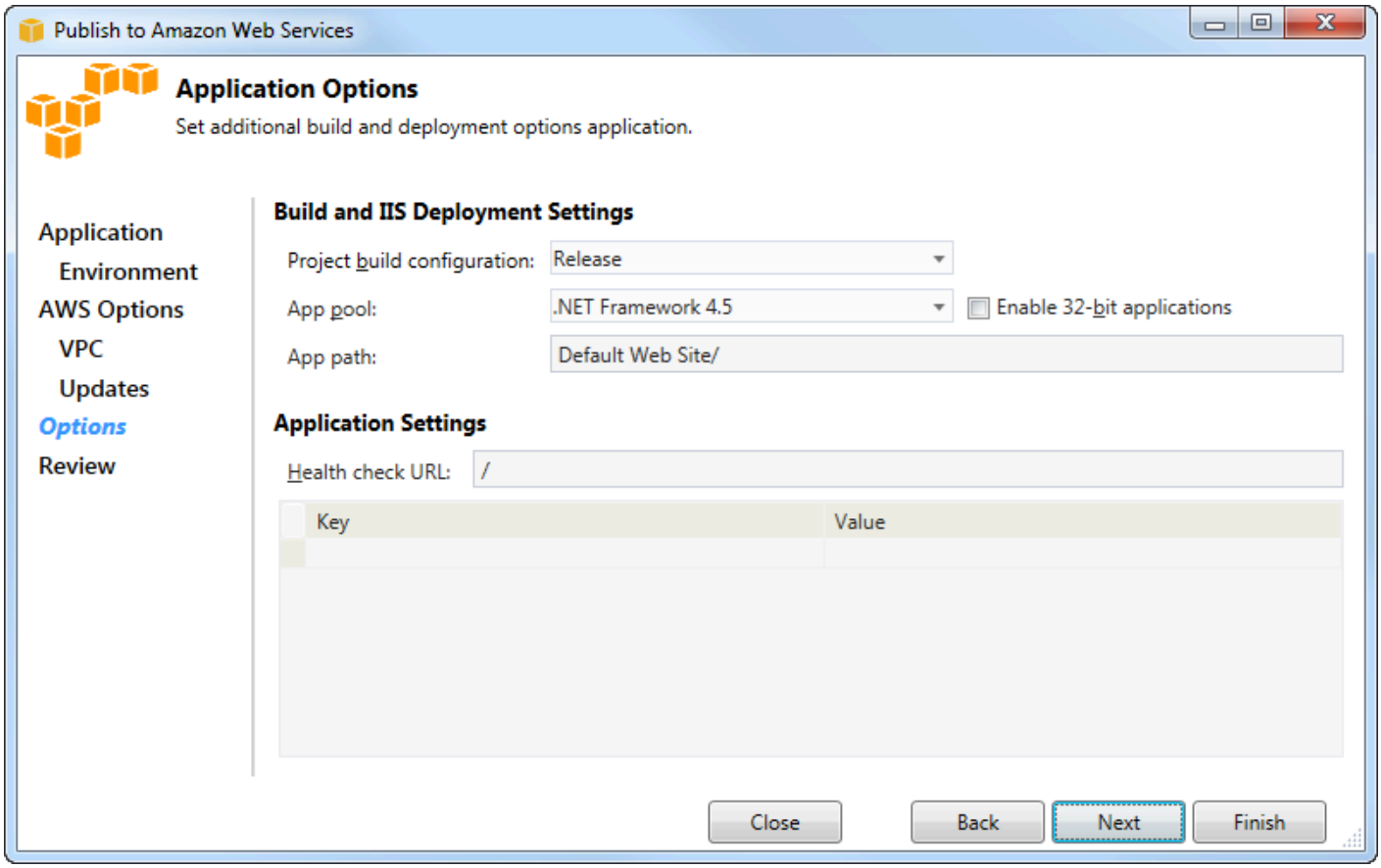
有关如何将 Elastic Beanstalk 应用程序部署到 VPC 的更多信息，请参阅《AWS Elastic Beanstalk 开发人员指南》。

1. 在填充 VPC Options (VPC 选项) 页面上的所有信息后，请选择 Next (下一步)。
   - 如果您已选择 Enable Rolling Deployments (启用滚动部署)，则将显示 Rolling Deployments (滚动部署) 页面。
   - 如果您未选择 Enable Rolling Deployments (启用滚动部署)，则将显示 Application Options (应用程序选项) 页面。向前跳至此部分中后面描述如何使用 Application Options (应用程序选项) 页面的说明。
2. 如果您已选择 Enable Rolling Deployments (启用滚动部署)，请在 Rolling Deployments (滚动部署) 页面上指定信息以配置新版本的应用程序部署到负载均衡环境中的实例的方式。例如，如果您的环境中有 4 个实例，并且您需要更改实例类型，则可将环境配置为一次更改 2 个实例。这可帮助确保您的应用程序在执行更改时仍处于运行状态。



3. 在 Application Versions (应用程序版本) 区域中，选择用于控制一次部署的实例的百分比或数量的选项。指定所需百分比或数量。

4. （可选）在 Environment Configuration (环境配置) 区域中，如果要指定在部署期间保持运行的实例数量，请选中此框。如果选中此框，请指定一次应修改的实例的最大数目和/或一次应保持运行的实例的最小数目。
5. 选择下一步。
6. 在 Application Options (应用程序选项) 页面上，指定有关版本、Internet Information Services (IIS) 和应用程序设置的信息。



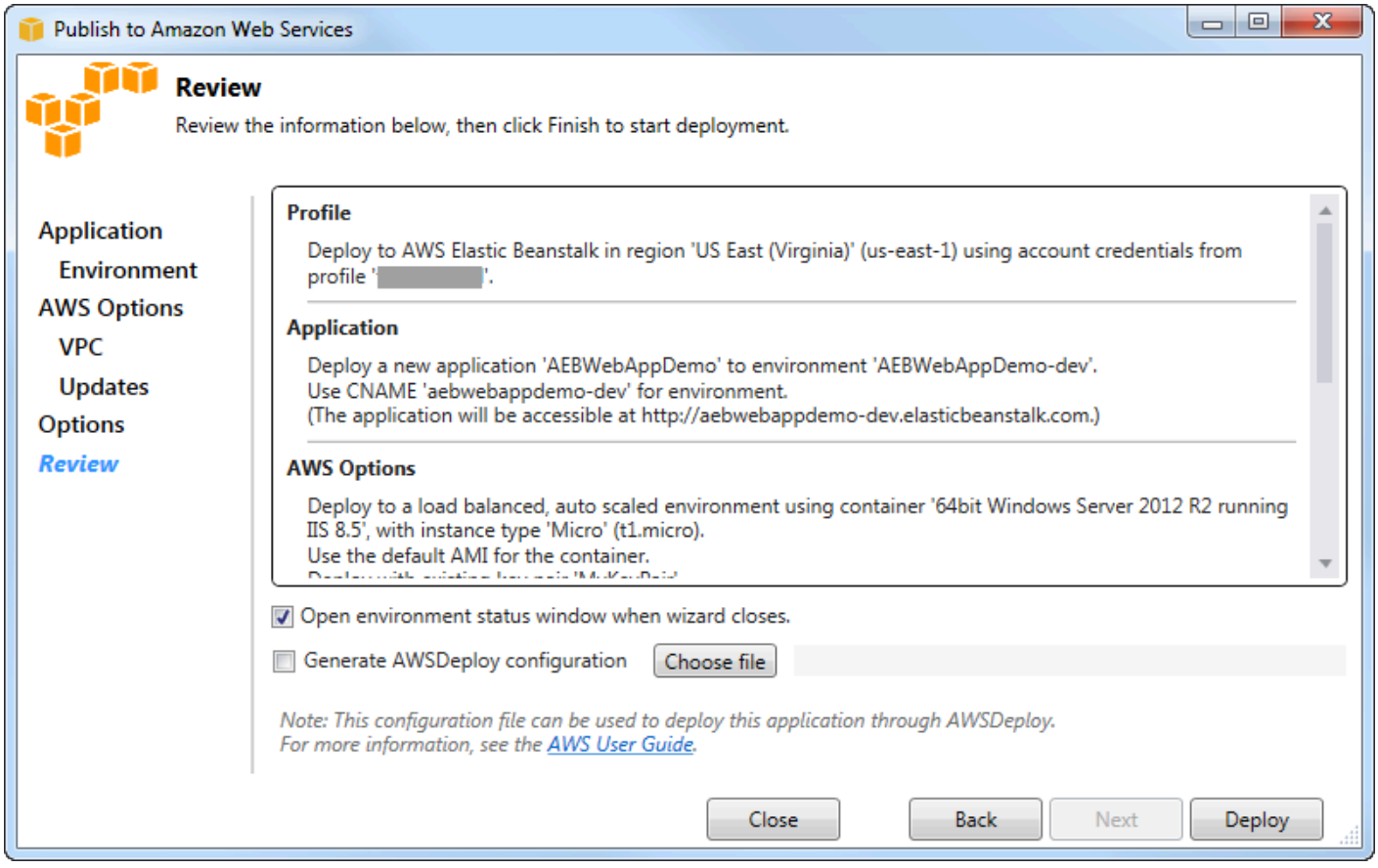
7. 在 Build and IIS Deployment Settings (生成和 IIS 部署设置) 区域的 Project build configuration (项目生成配置) 下拉列表中，选择目标版本配置。如果向导可以找到它，则 Release (发布) 将显示，否则此框中将显示有效配置。
8. 在 App pool (应用程序池) 下拉列表中，选择您的应用程序所需的 .NET Framework 版本。应已显示正确的 .NET Framework 版本。
9. 如果您的应用程序是 32 位的，请选中 Enable 32-bit applications (启用 32 位应用程序) 框。
10. 在 App path (应用程序路径) 框中，指定 IIS 将用来部署应用程序的路径。默认情况下，指定 Default Web Site/，它通常会转换为路径 `c:\inetpub\wwwroot`。如果您指定了 Default Web Site/ 之外的路径，向导将在 Default Web Site/ 路径中放置指向您指定的路径的重定向。

11.在应用程序设置区域的运行状况检查 URL 框中，键入要检查的 Elastic Beanstalk URL，以确定您的 Web 应用程序是否仍响应。此 URL 相对于根服务器 URL。默认情况下，已指定根服务器 URL。例如，如果完整 URL 为 `example.com/site-is-up.html`，则键入 `/site-is-up.html`。

12.在 Key (键) 和 Value (值) 的区域中，可指定要添加到应用程序的 `Web.config` 文件的任何密钥和值对。

> **Note**
>
> 尽管不建议这样做，但您可以使用 Key 和 Value 区域来指定应用程序运行时应使用的 AWS 凭据。首选方式是在 AWS 选项页面上的 Identity and Access Management 角色下拉列表中指定 IAM 角色。但是，如果您必须使用 AWS 证书而不是 IAM 角色来运行应用程序，请在密钥行中选择AWSAccess密钥。在 Value (值) 行中，键入访问密钥。对 AWSSecretKey 重复这些步骤。

13.选择下一步。



14.在 Review (查看) 页面上，查看您配置的选项，然后选中 Open environment status window when wizard closes (在向导关闭时打开环境状态窗口) 框。

15.如果一切正常，请选择 Deploy (部署)。

> **Note**
>
> 部署应用程序时，将向活动账户收取应用程序使用的 AWS 资源所产生的费用。

有关部署的信息将显示在 Visual Studio 状态栏和 Output (输出) 窗口中。该过程可能需要几分钟。部署完成后，Output (输出) 窗口中将显示确认消息。

16要删除部署，请在 AWS 资源管理器中展开 El asti c Beanstalk 节点，打开部署子节点的上下文（右键单击）菜单，然后选择删除。此删除过程可能需要几分钟。

将 ASP.NET Core 应用程序部署到 Elastic Beanstalk（旧版）

> **Important**
>
> 本文档涉及旧版服务和功能。有关更新的指南和内容，请参阅 AWS .NET deployment tool 指南和更新的部署到 AWS 目录。

AWS Elastic Beanstalk 是一项服务，可简化为应用程序配置 AWS 资源的过程。 AWS Elastic Beanstalk 提供了部署应用程序所需的所有 AWS 基础架构。

Visual Studio 工具包支持 AWS 使用 Elastic Beanstalk 部署 ASP.NET Core 应用程序。ASP.NET 内核是对 ASP.NET 的重新设计，具有模块化的架构，它最大程度地降低了依赖项开销并简化了应用程序以便在云中运行。

AWS Elastic Beanstalk 可以轻松地将各种不同语言的应用程序部署到 AWS。Elastic Beanstalk 支持传统 ASP.NET 应用程序和 ASP.NET Core 应用程序。本主题描述如何部署 ASP.NET 内核应用程序。

使用部署向导

将 ASP.NET Core 应用程序部署到 Elastic Beanstalk 的最简单方法是使用 Toolkit for Visual Studio。

如果您之前用过此工具包部署传统 ASP。NET 应用程序，您将发现 ASP.NET 内核的体验与之非常相似。在以下步骤中，我们将演练部署体验。

如果您以前从未使用过该工具包，则安装该工具包后需要做的第一件事就是向该工具包注册您的 AWS 凭据。有关操作方法的详细信息，请参阅 Visual Studio 的 “如何为应用程序指定 AWS 安全证书” 文档。

要部署 ASP.NET Core Web 应用程序，请在解决方案资源管理器中右键单击该项目，然后选择发布到 AWS...。

在发布到 AWS Elastic Beanstalk 部署向导的第一页上，选择创建新的 Elastic Beanstalk 应用程序。Elastic Beanstalk 应用程序是 Elastic Beanstalk 组件的逻辑集合，包括环境、版本和环境配置。此部署向导将生成一个应用程序，此应用程序将包含应用程序版本和环境的集合。这些环境包含运行应用程序版本的实际 AWS 资源。每次部署应用程序时，都会创建一个新的应用程序版本，并且此向导会将环境指向此版本。您可在 Elastic Beanstalk 组件中了解有关这些概念的更多信息。

接下来，为应用程序及其第一个环境设置名称。每个环境均关联一个唯一的别名，可使用此别名在部署完成后访问应用程序。

下一页 “AWS 选项” 允许您配置要使用的 AWS 资源类型。在此示例中，将保留默认值（Key pair (密钥对) 部分除外）。利用密钥对，可以检索 Windows 管理员密码，以便您能登录到计算机。如果您尚未创建密钥对，可能需要选择 Create new key pair (创建新密钥对)。

Permissions

权限页面用于向运行您的应用程序的 EC2 实例分配 AWS 证书。如果您的应用程序使用访问其他 AWS 服务， 适用于 .NET 的 AWS SDK 这一点很重要。如果您未使用应用程序中的任何其他服务，则可将保留此页面的默认值。

应用程序选项

Application Options (应用程序选项) 页面上的详细信息不同于部署传统 ASP.NET 应用程序时指定的详细信息。在此处，可指定用于打包应用程序的版本配置和框架，还可指定应用程序的 IIS 资源路径。

完成 Application Options (应用程序选项) 页面后，单击 Next (下一步) 查看设置，然后单击 Deploy (部署) 开始部署过程。

检查环境状态

将应用程序打包并上传到后 AWS，您可以在 Visual Studio 的资源管理器中打开环境状态视图，查看 AWS Elastic Beanstalk 环境的状态。

事件将在环境联机时显示在状态栏中。一切完成后，环境状态将变为正常状态。您可单击 URL 来查看站点。在此处，您还可从环境或远程桌面，将日志提取到作为 Elastic Beanstalk 环境一部分的 Amazon EC2 实例。

任何应用程序的首次部署都将比随后的重新部署花费更长的时间，因为它会创建新的 AWS 资源。在开发期间对应用程序执行迭代时，可再次通过向导快速重新部署，或通过在右键单击项目时选择 Republish (重新发布) 选项来快速重新部署。

重新发布操作将使用通过部署向导进行的上次运行中的设置来打包您的应用程序，并将应用程序包上传到现有 Elastic Beanstalk 环境。

如何为您的应用程序指定 AWS 安全证书

你在 “发布到 Elastic Beanstalk” 向导中指定的 AWS 账户 AWS 是该向导部署到 Elastic Beanstalk 时将使用的账户。

尽管不建议这样做，但您可能还需要指定应用程序在部署后用于访问 AWS 服务的 AWS 账户证书。首选方法是指定一个 IAM 角色。在发布到 Elastic Beanstalk 向导中，您可以通过AWS 选项页面上的 Identity and Access Management 角色下拉列表执行此操作。在旧版的发布到 Amazon Web Services 向导中，您可以通过 AWS 选项页面上的 IAM 角色下拉列表执行此操作。

如果您必须使用 AWS 账户证书而不是 IAM 角色，则可以通过以下方式之一为您的应用程序指定 AWS 账户证书：

- 在项目文件`appSettings`元素中引用与 AWS 账户凭据对应的`Web.config`个人资料。（要创建配置文件，请参阅配置 AWS 凭据。）以下示例指定了配置文件名称为 `myProfile` 的凭证。

```
<appSettings>
  <!-- AWS CREDENTIALS -->
  <add key="AWSProfileName" value="myProfile"/>
</appSettings>
```

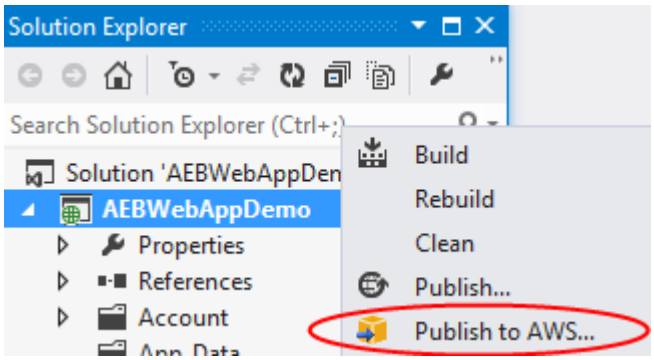
- 如果您使用的是发布到 Elastic Beanstalk 向导，请在 “应用程序选项” 页面的 “键和值” 区域的 “密钥” 行中选择。AWS AccessKey在 Value (值) 行中，键入访问密钥。对重复这些步骤AWS SecretKey。
- 如果您使用旧版发布到 Amazon Web Services 向导，则在应用程序选项页面上的应用程序凭证区域，选择使用这些凭证，然后在访问密钥和密钥框中分别键入访问密钥和秘密访问密钥。

如何将应用程序重新发布到 Elastic Beanstalk 环境（旧版）

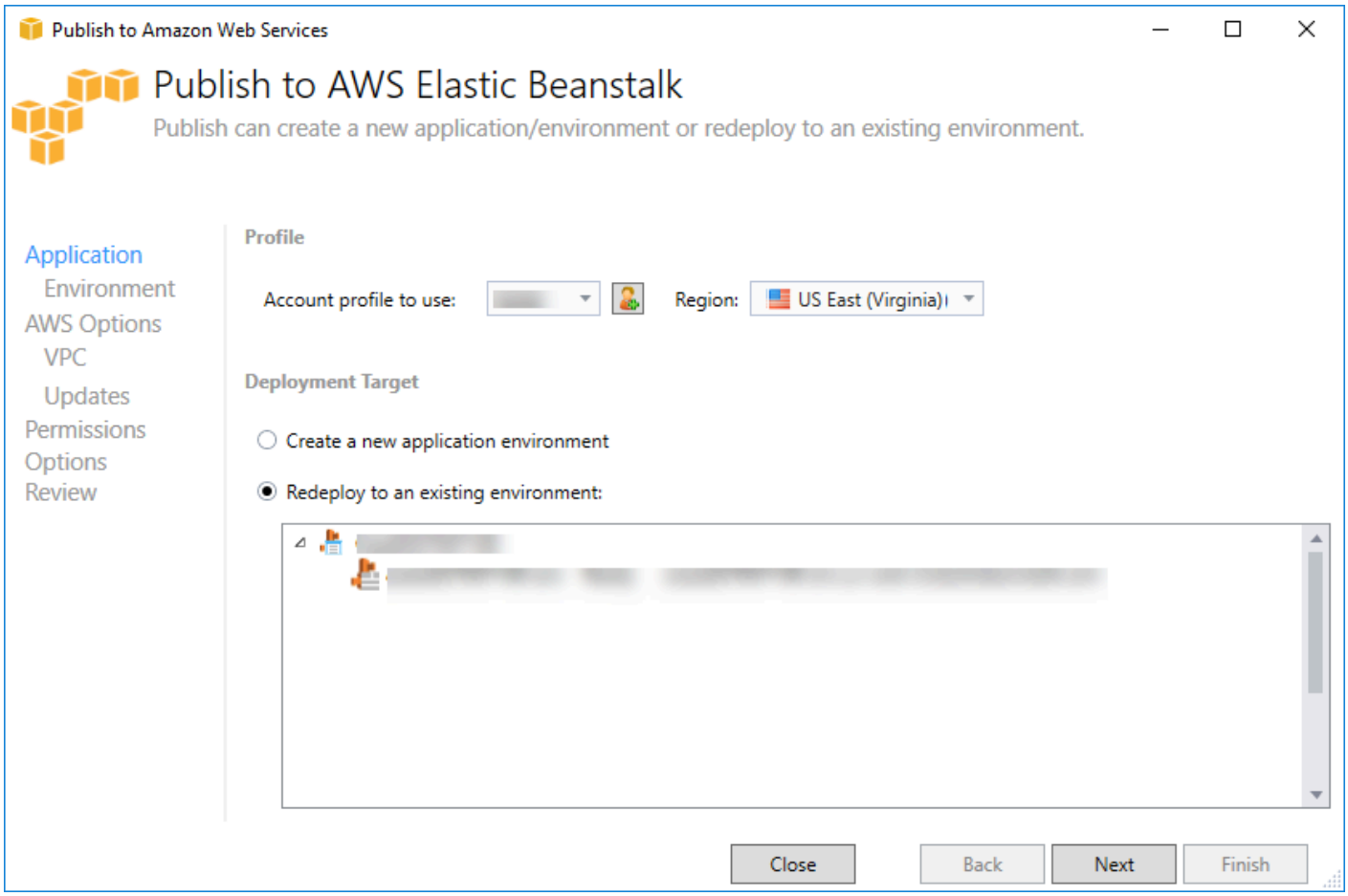
> **Important**
>
> 本文档涉及旧版服务和功能。有关更新的指南和内容，请参阅 AWS .NET 部署工具指南。

您可以进行不连续的更改，然后将新版本重新发布到已启动的 Elastic Beanstalk 环境，从而在应用程序上进行迭代。

1. 在 “解决方案资源管理器” 中，打开您在上一节中发布的AEBWebAppDemo项目的项目文件夹的上下文（右键单击）菜单，然后选择 “发布到” AWS Elastic Beanstalk。

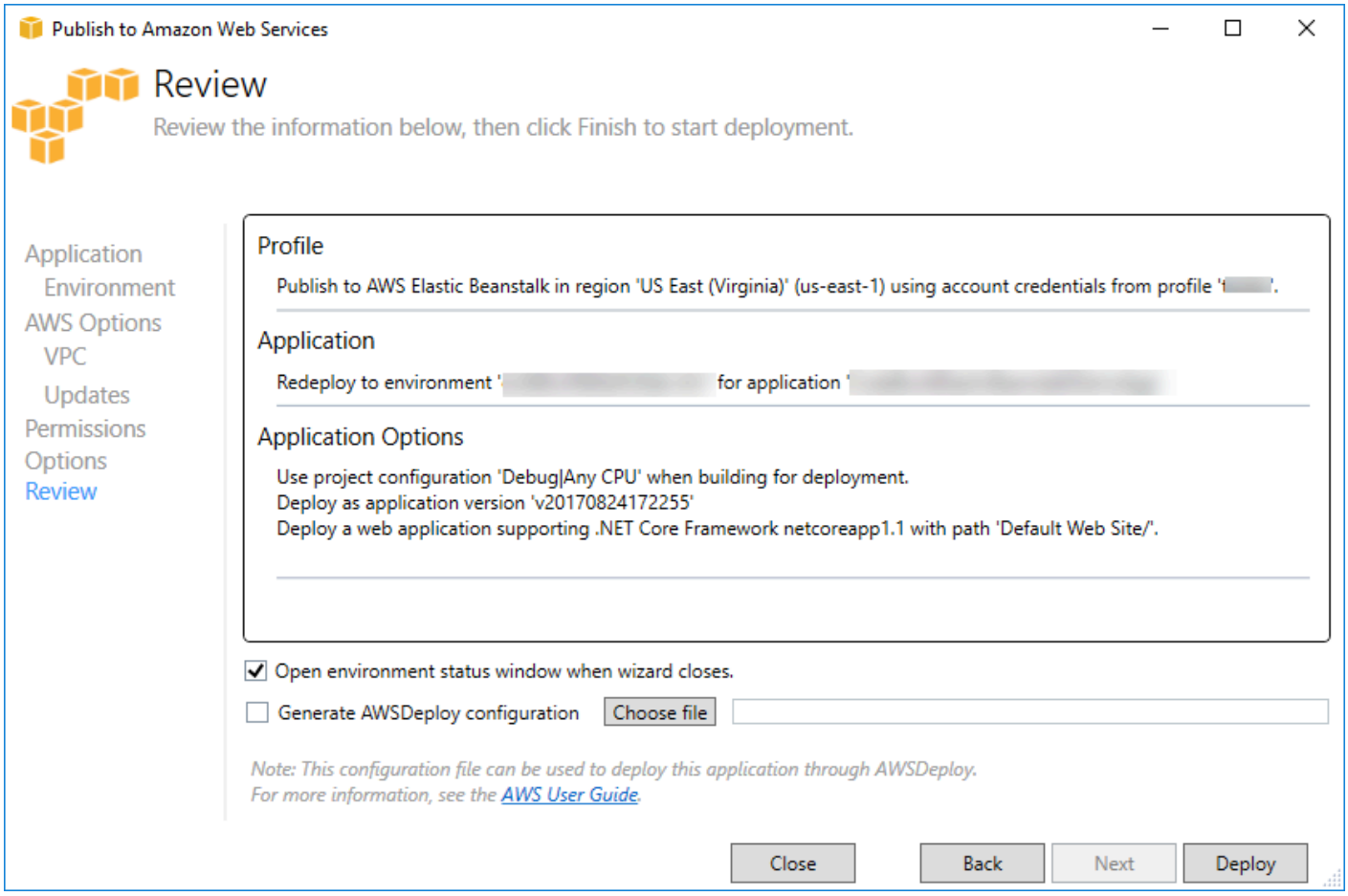


随即显示 Publish to Elastic Beanstalk (发布到 Elastic Beanstalk) 向导。



2. 选择 Redeploy to an existing environment (重新部署到现有环境)，然后选择您之前所发布到的环境。单击下一步。

Review (查看) 向导随即出现。



3. 单击 Deploy (部署)。该应用程序将重新部署到相同的环境。

如果您的应用程序正处于启动或终止过程，则无法重新发布。

自定义 Elastic Beanstalk 应用程序部署

本主题介绍 Elastic Beanstalk 的 Microsoft Windows 容器的部署清单如何支持自定义应用程序部署。

对于想利用 Elastic Beanstalk 的功能来创建和管理 AWS 资源，同时希望完全控制应用程序的部署方式的高级用户，自定义应用程序部署是一个强大的功能。对于自定义应用程序部署，您将为 Elastic Beanstalk 执行的三个不同操作创建 Windows PowerShell 脚本。安装操作在启动部署时使用，重新启动操作在从工具包或 Web 控制台调用 `RestartAppServer` API 时使用，卸载操作在新部署出现时被任何之前的部署调用。

例如，当您的文档团队编写了一个他们希望包含在部署中的静态网站情况下，您可能希望部署一个 ASP.NET 应用程序。您可以按如下方式编写部署清单来执行该操作：

```
{
```

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

为每个操作列出的脚本必须位于与部署清单文件相关的应用程序包中。在本示例中，应用程序包还将包含一个 documentation.zip 文件，该文件包含由您的文档团队创建的静态网站。

`install.ps1` 脚本将提取该 zip 文件并设置 IIS 路径。

```
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}
```

由于您的应用程序在 IIS 中运行，重新启动操作将调用 IIS 重置。

```
iisreset /timeout:1
```

若要卸载脚本，则务必清除在安装阶段使用的所有设置和文件。这样，在新版本的安装阶段，您可以避免与以前的部署之间的所有冲突。在本示例中，您需要删除静态网站的 IIS 应用程序并删除网站文件。

```
powershell.exe -Command {Remove-WebApplication -Name documentation}
Remove-Item -Recurse -Force 'c:\inetpub\wwwroot\documentation'
```

由于这些脚本文件以及 documentation.zip 文件包含在您的应用程序包中，该部署将创建 ASP.NET 应用程序，然后部署文档站点。

在本示例中，我们将选择一个部署简单静态网站的简单示例，但通过自定义应用程序部署，您可以部署任何类型的应用程序并让 Elastic Beanstalk 管理其 AWS 资源。

自定义 ASP.NET Core Elastic Beanstalk 部署

本主题介绍部署的工作原理，以及在利用 Elastic Beanstalk 和 Toolkit for Visual Studio 创建 ASP.NET Core 应用程序时如何自定义部署。

在 Toolkit for Visual Studio 中完成部署向导后，Toolkit 将对该应用程序打包并将其发送到 Elastic Beanstalk。创建应用程序包的第一步是借助新的 dotnet CLI 为应用程序做好使用 publish 命令进行发布的准备。框架和配置将从向导中的设置向下传递到 publish 命令。因此，如果您为 `configuration` 选择了发布，为 `framework` 选择了 netcoreapp1.0，则 Toolkit 将执行以下命令：

```
dotnet publish --configuration Release --framework netcoreapp1.0
```

当 publish 命令完成后，Toolkit 会将新的部署清单写入到发布文件夹。此部署清单是一个名为 aws-windows-deployment-manifest.json 的 JSON 文件，Elastic Beanstalk Windows 容器（版本 1.2 或更高版本）将读取该文件以确定如何部署应用程序。例如，对于要在 IIS 的根处部署的 ASP.NET 内核应用程序，Toolkit 将生成一个清单文件，如下所示：

```
{
  "manifestVersion": 1,
  "deployments": {

    "aspNetCoreWeb": [
      {
```

```
        "name": "app",
        "parameters": {
          "appBundle": ".",
          "iisPath": "/",
          "iisWebSite": "Default Web Site"
        }
      }
    ]
  }
}
```

appBundle 属性指示了应用程序位与清单文件相关的位置。此属性可指向目录或 ZIP 存档。iisPath 和 iisWebSite 属性指示了 IIS 中要托管应用程序的位置。

自定义清单

如果某个清单文件在发布文件夹中尚不存在，则 Toolkit 仅写入该清单文件。如果该文件已存在，Toolkit 将更新该清单的 appBundle 部分下列出的第一个应用程序中的 iisPath、iisWebSite 和 aspNetCoreWeb 属性。这使您可以将 aws-windows-deployment-manifest.json 添加到您的项目并自定义该清单。要对 Visual Studio 中的 ASP.NET 内核 Web 应用程序执行此操作，请将新的 JSON 文件添加到项目的根并将其命名为 aws-windows-deployment-manifest.json。

该清单必须命名为 aws-windows-deployment-manifest.json 且必须位于项目的根处。Elastic Beanstalk 容器将在根中寻找该清单，如果找到，则会调用部署工具。如果该文件不存在，Elastic Beanstalk 容器将回退到较早的部署工具，该工具假定存档为 msdeploy 存档。

要确保 dotnet CLI publish 命令包含该清单，请更新 project.json 文件以将该清单文件包含在 include 中 publishOptions 下的 include 部分。

```
{
   "publishOptions": {
     "include": [
       "wwwroot",
       "Views",
       "Areas/**/Views",
       "appsettings.json",
       "web.config",
       "aws-windows-deployment-manifest.json"
     ]
   }
 }
```

既然您已声明该清单以便让它包含在应用程序包中，您可以进一步配置要部署应用程序的方式。除了部署向导所支持的部署之外，您还可以自定义部署。AWS 已为 aws-windows-deployment-manifest.json 文件定义了 JSON 架构，并且在您安装 Toolkit for Visual Studio 后，该安装程序为架构注册了 URL。

当您打开 `windows-deployment-manifest.json` 时，您将看到在“Schema (架构)”下拉框中选择的架构 URL。您可以导航到该 URL 以获取可在该清单中设置的内容的完整说明。在已选择该架构的情况下，Visual Studio 将在您编辑该清单时提供 IntelliSense。

您可以执行的一项自定义是配置应用程序将在其下运行的 IIS 应用程序池。以下示例显示了如何定义 IIS 应用程序池（“customPool”），该池每 60 分钟再循环一次流程，并使用 `"appPool": "customPool"` 将流程分配到应用程序。

```
{
  "manifestVersion": 1,
  "iisConfig": {
    "appPools": [
      {
        "name": "customPool",
        "recycling": {
          "regularTimeInterval": 60
        }
      }
    ]
  },
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "app",
        "parameters": {
          "appPool": "customPool"
        }
      }
    ]
  }
}
```

此外，该清单还可以声明 Windows PowerShell 脚本在安装、重启和卸载操作之前和之后运行。例如，以下清单运行 Windows PowerShell 脚本 `PostInstallSetup.ps1` 以在 ASP.NET 内核应用程序部署到 IIS 之后完成进一步设置工作。在添加类似这样的脚本时，请确保将它们添加到 `project.json` 文件中的 publishOptions 下的 include 部分，正如对 `aws-windows-deployment-manifest.json` 文件的处理方式一样。如果没有这样做，这些脚本将不会作为 dotnet CLI publish 命令的一部分包含。

```
{
  "manifestVersion": 1,
  "deployments": {
    "aspNetCoreWeb": [
      {
        "name": "app",
        "scripts": {
          "postInstall": {
            "file": "SetupScripts/PostInstallSetup.ps1"
          }
        }
      }
    ]
  }
}
```

ebextensions 怎么样？

Elastic Beanstalk .ebextensions 配置文件像在所有其他 Elastic Beanstalk 容器中一样受支持。要在 ASP.NET 内核应用程序中包含 ebextensions，请将 `.ebextensions` 目录添加到 `include` 文件中的 `publishOptions` 下的 `project.json` 部分。有关 ebextensions 的更多信息，请查阅 Elastic Beanstalk 开发人员指南。

对 .NET 和 Elastic Beanstalk 的多应用程序支持

通过使用部署清单，您能够将多个应用程序部署到同一 Elastic Beanstalk 环境。

部署清单支持 ASP.NET 内核 Web 应用程序以及用于传统 ASP.NET 应用程序的 msdeploy 存档。设想以下情形：您为前端编写了一个使用 ASP.NET 内核的很棒的新应用程序并为扩展 API 编写了一个 Web API 项目。您还有一个使用传统 ASP.NET 编写的管理应用程序。

工具包的部署向导侧重于部署单一项目。要利用多应用程序部署，您必须手动构造应用程序包。要开始，请写入清单。在本示例中，您将在解决方案的根部写入清单。

清单中的部署部分包含两个子级：要部署的 ASP.NET 内核 Web 应用程序的数组以及要部署的 msdeploy 存档的数组。对于每个应用程序，您应设置应用程序的位相对于清单的 IIS 路径和位置。

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

在写入清单后，您将使用 Windows PowerShell 创建应用程序包并更新现有 Elastic Beanstalk 环境以运行此包。写入脚本时将假定它在包含您的 Visual Studio 解决方案的文件夹中运行。

您在脚本中需要执行的一个操作是设置在其中创建应用程序包的工作区文件夹。

```
$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
}
```

创建此文件夹后，是时候为前端做好准备了。与使用部署向导时一样，应使用 dotnet CLI 来发布应用程序。

```
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
```

请注意，子文件夹“frontend”用于输出文件夹 (与您在清单中设置的文件夹匹配)。现在您需要对 Web API 项目执行相同的操作。

```
Write-Host 'Publish the ASP.NET Core extensibility API'
$publishExtAPIFolder = [System.IO.Path]::Combine($publishWorkspace, "ext-api")
dotnet publish .\src\AmazingExtensibleAPI\project.json -o $publishExtAPIFolder -c
 Release -f netcoreapp1.0
```

管理员站点是传统 ASP.NET 应用程序，因此您无法使用 dotnet CLI。对于管理应用程序，您应使用在生成目标包中传递的 msbuild 来创建 msdeploy 存档。默认情况下，包目标将在 `obj\Release\Package` 文件夹下创建 msdeploy 存档，因此您需要将此存档部署到发布工作区。

```
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
```

要告知 Elastic Beanstalk 环境如何处理所有这些应用程序，请将此清单从您的解决方案复制到发布工作区，然后压缩文件夹。

```
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)
```

现在您已拥有应用程序包，您可转到 Web 控制台并将存档上传到 Elastic Beanstalk 环境。或者，您也可以继续使用 AWS PowerShell cmdlet，用应用程序包更新 Elastic Beanstalk 环境。确保您使用

`Set-AWSCredentials` 和 `Set-DefaultAWSRegion` cmdlet 将当前配置文件和区域设置为包含您的 Elastic Beanstalk 环境的配置文件和区域。

```
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
```

现在，通过在 Toolkit 或 Web 控制台中使用 Elastic Beanstalk 环境状态页面来检查更新状态。完成后，您将能够导航到部署于在部署清单中设置的 IIS 路径的所有应用程序。

部署到 Amazon EC2 Container Service

> **Important**
>
> 新的发布到 AWS 功能旨在简化将 .NET 应用程序发布到 AWS 的方式。在您选择将容器发布到 AWS 后，系统可能会询问您是否要切换到此发布体验。有关更多信息，请参阅 在 Visual Studio 中使用“发布到 AWS”。

Amazon Elastic Container Service 是一种高度可扩展、高性能的容器管理服务，它支持 Docker 容器并可让您在 Amazon EC2 实例的托管集群上轻松运行应用程序。

要在 Amazon Elastic Container Service 上部署应用程序，您开发应用程序组件时必须要使其可以在 Docker 容器中运行。Docker 容器是一个软件开发的标准化单位，包含您的软件应用程序需要运行的一切：代码、运行时、系统工具、系统库等。

Toolkit for Visual Studio 提供了一个向导，可简化通过 Amazon ECS 进行的应用程序发布。此向导将在以下部分中介绍。

有关 Amazon ECS 的更多信息，请参阅 Elastic Container Service 文档。它概述了 Docker 的基础知识和集群创建。

主题

- 为您的 ASP.NET 酷睿 2 应用程序指定 AWS 凭据
- 将 ASP.NET Core 2.0 应用程序部署到 Amazon ECS（Fargate）（旧版）
- 将 ASP.NET Core 2.0 应用程序部署到 Amazon ECS（EC2）

为您的 ASP.NET 酷睿 2 应用程序指定 AWS 凭据

当您将应用程序部署到 Docker 容器时，有两种类型的凭证在发挥作用：部署凭证和实例凭证。

发布容器到向 AWS 导使用部署凭证在 Amazon ECS 中创建环境。这包括任务、服务、IAM 角色、Docker 容器存储库等，如果您选择的话，还包括负载均衡器。

实例（包括您的应用程序）使用实例证书来访问不同的 AWS 服务。例如，如果您的 ASP.NET Core 2.0 应用程序读取和写入到 Amazon S3 对象，则该应用程序需要适当的权限。您可以根据环境使用不同的方法提供不同凭证。例如，您的 ASP.NET 内核e 2 应用程序可能面向开发和生产环境。您可以使用本地 Docker 实例和凭证进行开发，而在生产中使用定义的角色。

指定部署凭证

您在 “ AWS 将容器发布到 AWS” 向导中指定的 AWS 账户就是该向导用于部署到 Amazon ECS 的账户。账户资料必须具有访问亚马逊弹性计算云、亚马逊弹性容器服务和 AWS Identity and Access Management。

如果您注意到下拉列表中缺少某些选项，可能是因为您缺乏权限。例如，如果您为应用程序创建了一个集群但在将容器发布到 AWS 向导的集群页面中没有看到它。如果出现这种情况，请添加所缺的权限并重试该向导。

指定开发实例凭证

对于非生产环境，您可以在 appsettings.<environment>.json 文件中配置凭证。例如，要在 Visual Studio 2017 的 appsettings.Development.json 文件中配置凭证，请执行以下操作：

1. 添加 AWSSDK .Extensions。 NETCore.将 NuGet 软件包安装到您的项目中。
2. 将 AWS 设置添加到 appSettings.developmen 以下配置将设置 `Profile` 和 `Region`。

```
{
```

```
    "AWS": {
        "Profile": "local-test-profile",
        "Region": "us-west-2"
    }
}
```

指定生产实例凭证

对于生产实例，建议使用 IAM 角色来控制应用程序（和服务）可以访问的内容。例如，要在 AWS 管理控制台中将 Amazon ECS 的 IAM 角色配置为具有对 Amazon Simple Storage Service 和 Amazon DynamoDB 的权限的服务主体，请执行以下操作：

1. 登录 AWS 管理控制台 并打开 IAM 控制台，网址为https://console.aws.amazon.com/iam/。
2. 在 IAM 控制台的导航窗格中，选择角色，然后选择创建角色。
3. 选择 AWS 服务角色类型，然后选择 EC2 Container Service。
4. 选择 EC2 Container Service Task (EC2 Container Service 任务) 使用案例。用例由服务定义以包含服务要求的信任策略。然后选择下一步：权限。
5. 选择 AmazonS3 FullAccess 和AmazonDynamoDBFull访问权限策略。选中每个策略旁边的复选框，然后选择 Next: Review (下一步: 审核)。
6. 对于Role name (角色名称)，键入有助于识别此角色的作用的角色名称或角色名称后缀。角色名称在您的 AWS 账户内必须是唯一的。它们不按大小写区分。例如，您无法同时创建名为 PRODROLE 和 prodrole 的角色。由于多个单位可能引用该角色，角色创建完毕后无法编辑角色名称。
7. (可选) 对于 Role description，键入新角色的描述。
8. 检查角色，然后选择创建角色。

您可以使用此角色作为将容器发布到 AWS 向导中 ECS 任务定义页面上的任务角色。

有关更多信息，请参阅使用基于服务的角色。

将 ASP.NET Core 2.0 应用程序部署到 Amazon ECS（Fargate）（旧版）

> **Important**
>
> 本文档涉及旧版服务和功能。有关更新的指南和内容，请参阅 AWS .NET deployment tool 指南和更新的部署到 AWS 目录。

本部分介绍如何借助将容器发布到 AWS 向导（作为 Toolkit for Visual Studio 的一部分提供）来使用 Fargate 启动类型通过 Amazon ECS 部署针对 Linux 的容器化 ASP.NET Core 2.0 应用程序。由于 Web 应用程序要持续运行，因此将作为一项服务部署。

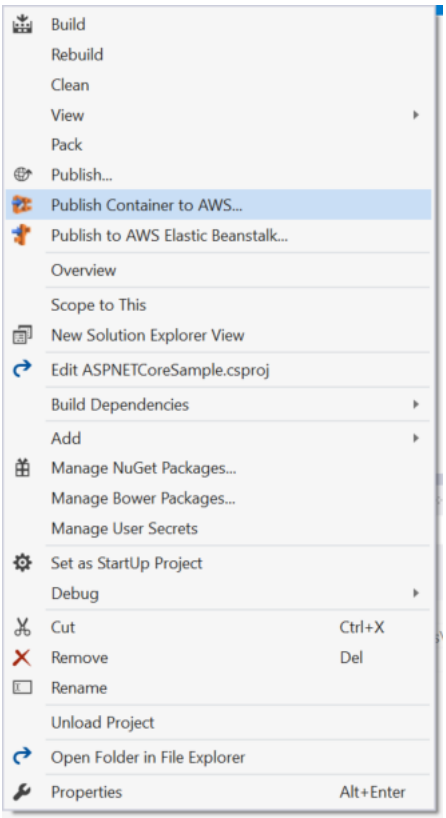
在您发布容器之前

在使用将容器发布到 AWS 向导部署 ASP.NET Core 2.0 应用程序之前，请执行以下操作：

- 指定 AWS 凭证并设置以使用 Amazon ECS。
- 安装 Docker。有几个不同的安装选项，包括 Docker for Windows。
- 在 Visual Studio 中创建（或打开）一个针对 Linux 的 ASP.NET Core 2.0 容器化应用程序的项目。

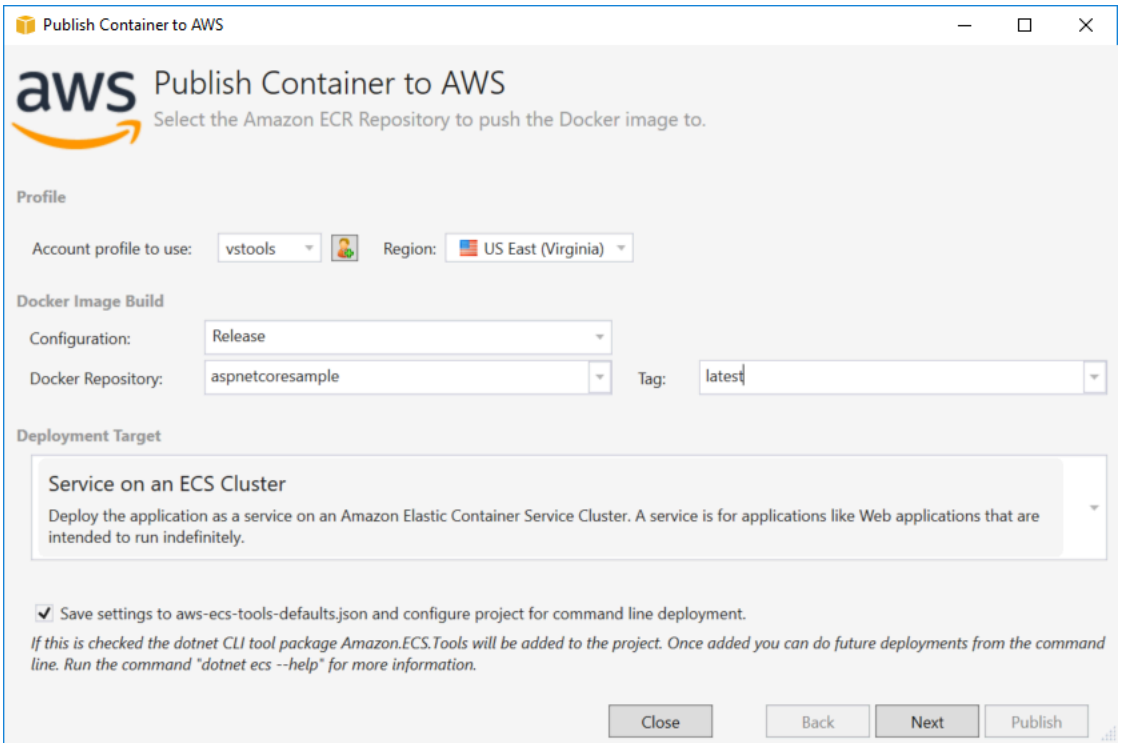
访问 “将容器发布到 AWS ” 向导

要部署针对 Linux 的 ASP.NET Core 2.0 容器化应用程序，请右键单击解决方案浏览器中的项目并选择将容器发布到 AWS。



您还可以在 Visual Studio 的“构建”菜单中选择将容器发布到 AWS。

将容器发布到 AWS 向导



Account profile to use (要使用的账户配置文件) - 选择要使用的账户配置文件。

Region (区域) - 选择部署区域。配置文件和区域用于设置您的部署环境资源并选择默认的 Docker 注册表。

Configuration (配置) - 选择 Docker 映像构建配置。

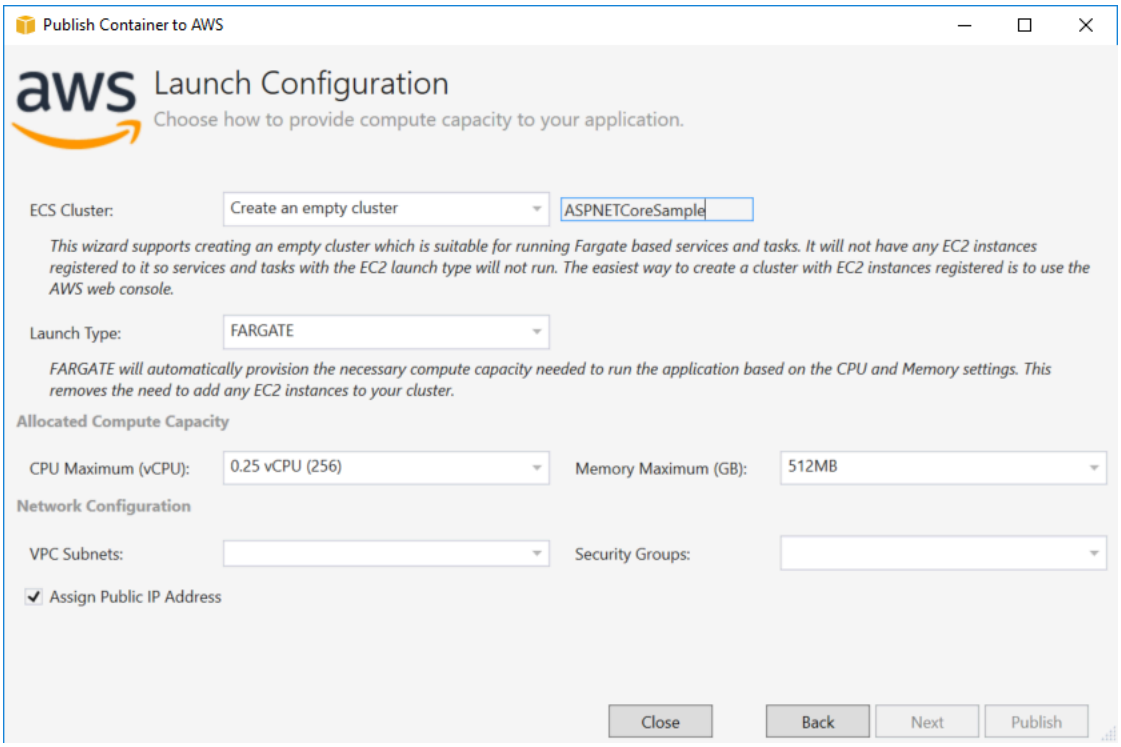
Docker Repository (Docker 存储库) - 选择现有 Docker 存储库，或键入新存储库的名称即可创建新存储库。这是构建容器要推送到的存储库。

Tag (标签) - 选择现有标签或键入新标签的名称。标签可以跟踪重要详细信息，如 Docker 容器的版本、选项或其他唯一配置元素。

Deployment Target (部署目标) – 选择 Service on an ECS Cluster (ECS 集群上的服务)。当您的应用程序（如 ASP.NET Web 应用程序）计划长时间运行时，请使用此部署选项。

将设置保存到 **`aws-docker-tools-defaults.json`** 并为命令行部署配置项目 - 如果您需要从命令行灵活部署，请选中此选项。使用您的项目目录中的 `dotnet ecs deploy` 以部署和 `dotnet ecs publish` 容器。

“Launch Configuration (启动配置)”页面



ECS Cluster (ECS 集群) – 选择将运行 Docker 映像的集群。如果您选择创建空集群，请为您的新集群命名。

Launch Type (启动类型) - 选择 FARGATE。

CPU Maximum (vCPU) (CPU 最大容量(vCPU)) - 选择您的应用程序所需的最大计算容量。要查看 CPU 和内存值的允许范围，请参阅任务大小。

Memory Maximum (GB) (CPU 最大容量(GB)) – 选择您的应用程序可用的最大内存容量。

VPC Subnets (VPC 子网) – 选择单个 VPC 中的一个或多个子网。如果您选择多个子网，则您的任务将分配到这几个子网中。这可以提高可用性。有关更多信息，请参阅默认 VPC 和默认子网。

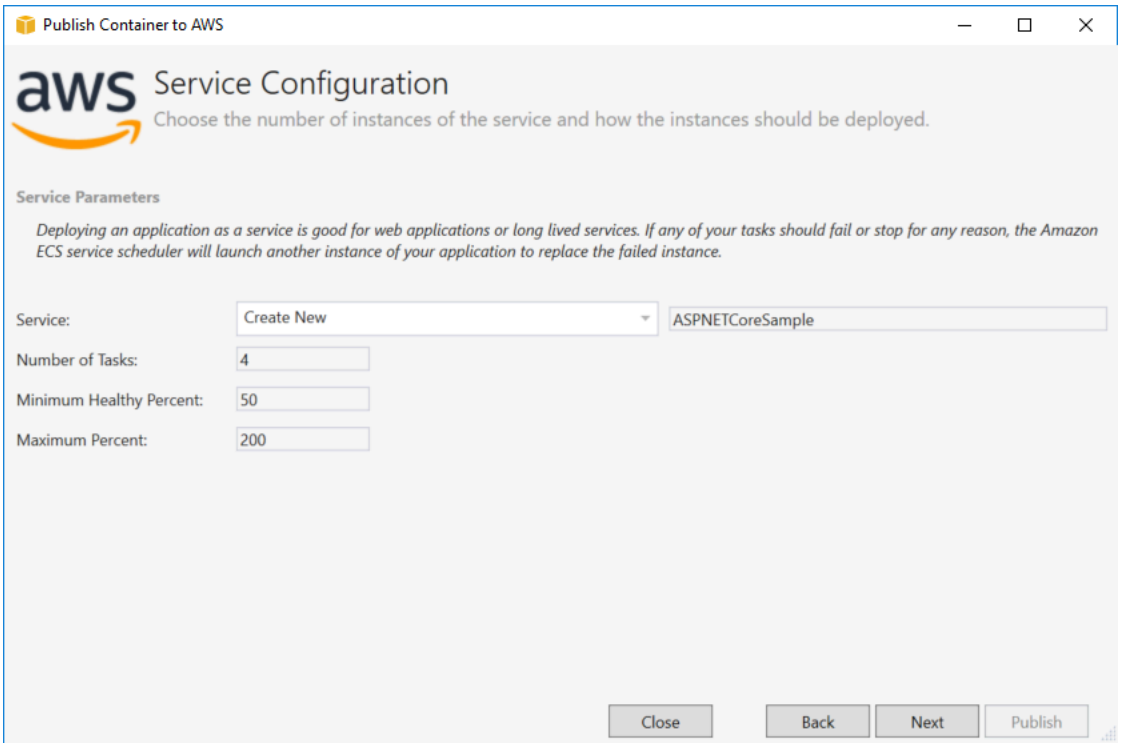
Security Groups (安全组) - 选择一个安全组。

安全组可作为关联 Amazon EC2 实例的防火墙，在实例级别控制入站和出站流量。

默认安全组配置为允许来自分配给同一安全组的实例的入站 IPv4 流量和所有出站流量。您需要允许出站，以便服务可以访问容器存储库。

Assign Public IP Address (分配公有 IP 地址) – 选中此复选框以便从 Internet 访问任务。

“Service Configuration (服务配置)”页面



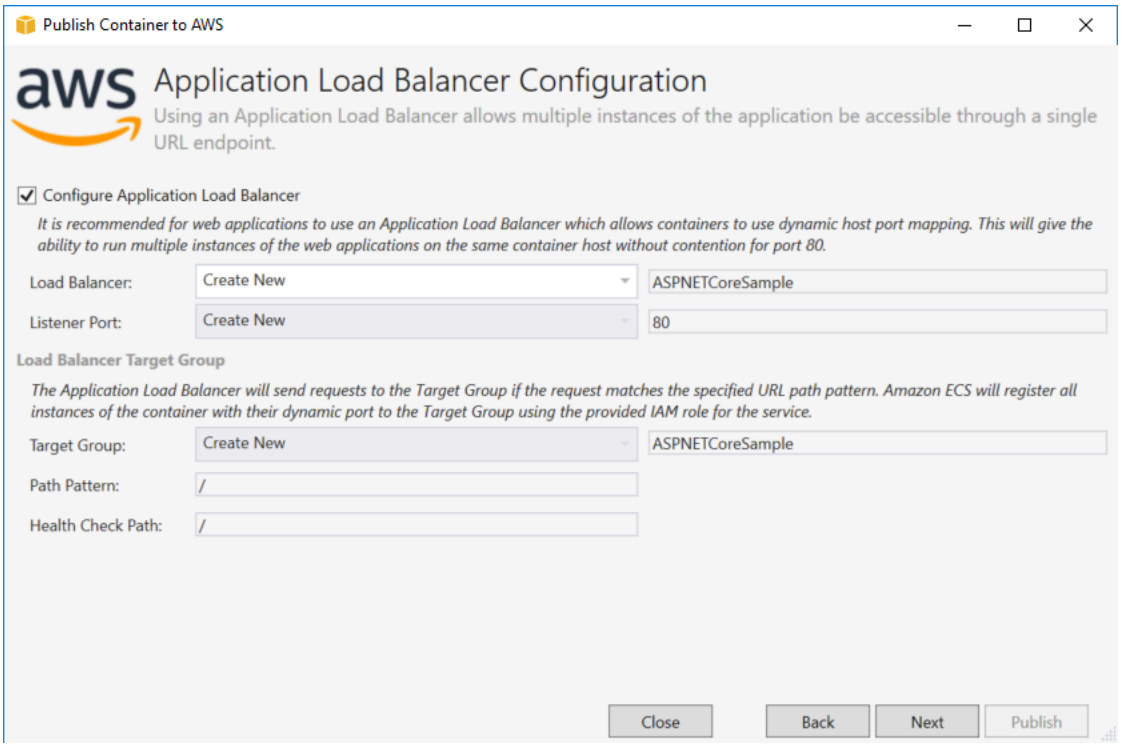
Service (服务) - 从下拉框中选择一项服务，将您的容器部署到该现有服务。或者选择 Create New (新建) 新建一项服务。一个集群中的服务名称必须唯一，但是您可以为一个区域或多个区域中多个集群中的服务提供相似的名称。

Number of Tasks (任务数) - 要在您的集群中部署并保持运行的任务数量。每个任务都是您的容器的一个实例。

Minimum Healthy Percent (最小正常运行状况百分比) - 在部署期间必须处于 RUNNING 状态的任务百分比（四舍五入到最近的整数）。

Maximum Percent (最大百分比) - 在部署期间允许处于 RUNNING 或 PENDING 状态的任务百分比（向下舍入到最近的整数）。

“Application Load Balancer (应用程序负载均衡器)”页面



Configure Application Load Balancer (配置应用程序负载均衡器) - 选中此项可配置应用程序负载均衡器。

Load Balancer (负载均衡器) - 选择一个现有负载均衡器，或者选择 Create New (新建) 并键入新负载均衡器的名称。

Listener Port (侦听器端口) - 选择一个现有侦听器端口，或者选择 Create New (新建) 并键入一个端口号。默认端口 80 适用于大多数 Web 应用程序。

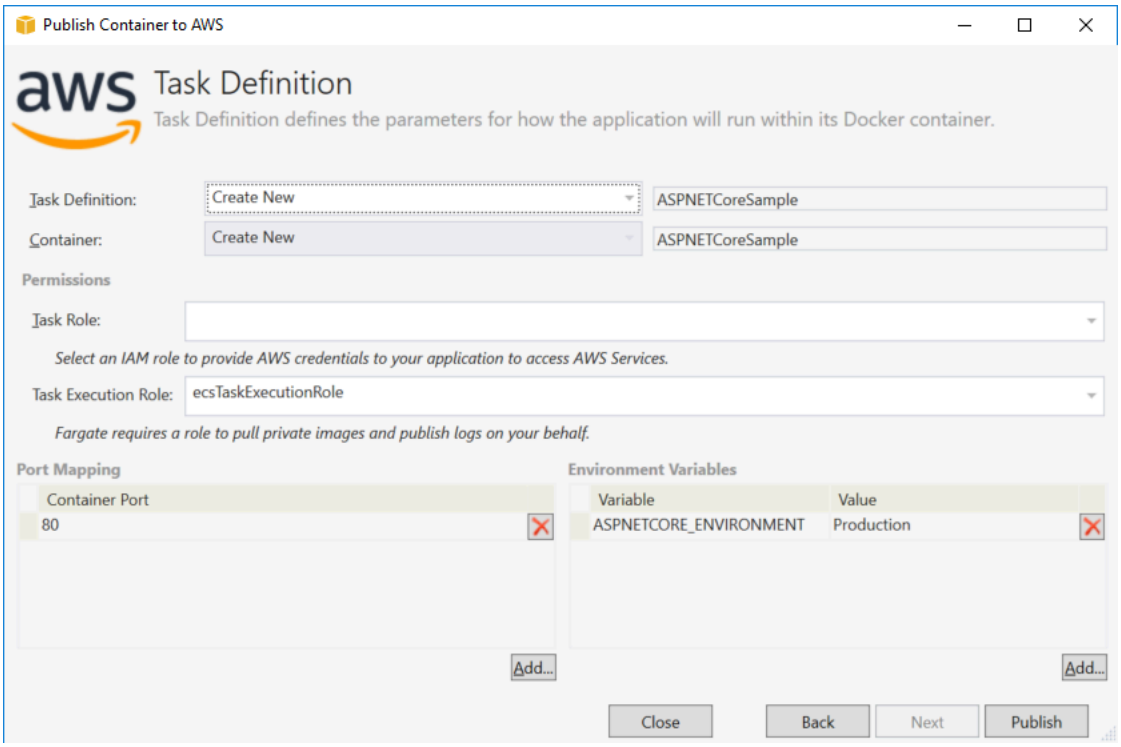
目标组 – 选择 Amazon ECS 要将服务任务注册到的目标组。

Path Pattern (路径模式) - 负载均衡器将使用基于路径的路由。接受默认 / 或提供一个不同模式。路径模式区分大小写，长度最多为 128 个字符，并且可包含一组选定字符。

Health Check Path (运行状况检查路径) - 进行运行状况检查的目标上的目的地的 Ping 路径。默认为 /。输入不同的路径（如果需要）。如果您输入的路径无效，则运行状况检查将失败，并将视为运行状况不佳。

如果您要部署多个服务，且每个服务都将部署到不同的路径或位置，您需要自定义检查路径。

“Task Definition (任务定义)”页面



Task Definition (任务定义) - 选择一个现有任务定义，或者选择 Create New (新建) 并键入新任务定义的名称。

Container (容器) - 选择一个现有容器，或者选择 Create New (新建) 并键入新容器的名称。

任务角色-选择一个 IAM 角色，该角色具有您的应用程序访问 AWS 服务所需的证书。凭证就是通过这种方法传递给您的应用程序的。请参阅如何为应用程序指定 AWS 安全凭证。

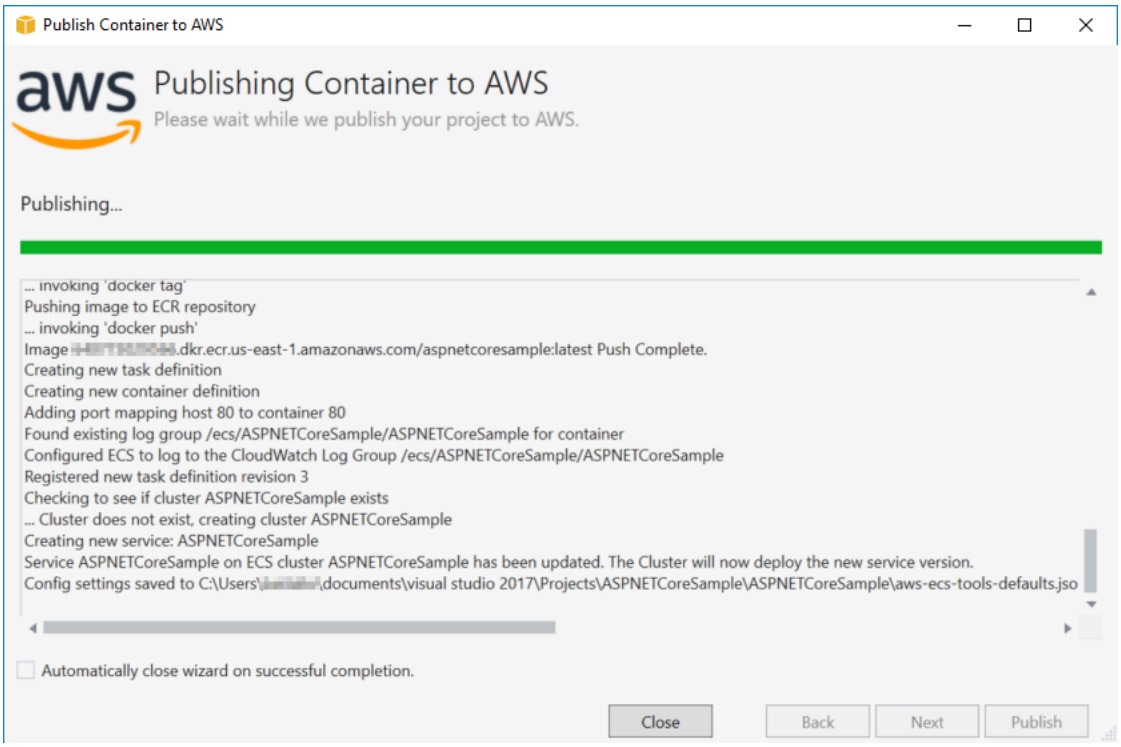
任务执行角色-选择有权提取私有镜像和发布日志的角色。 AWS Fargate 将代表你使用它。

Port Mapping (端口映射) - 选择绑定到自动分配的主机端口的容器上的端口号。

Environment Variables (环境变量) - 添加、修改或删除容器的环境变量。您可以进行修改以满足部署要求。

如果您对配置满意，请单击 Publish (发布) 开始部署流程。

将容器发布到 AWS



在部署过程中会显示事件。成功完成后向导会自动关闭。您可以通过取消选中页面底部的复选框来覆盖该功能。

您可以在 AWS 资源管理器中找到新实例的 URL。展开 Amazon ECS 和集群，然后单击您的集群。

将 ASP.NET Core 2.0 应用程序部署到 Amazon ECS（EC2）

本部分介绍如何借助将容器发布到 AWS 向导（作为 Toolkit for Visual Studio 的一部分提供）来使用 EC2 启动类型通过 Amazon ECS 部署针对 Linux 的容器化 ASP.NET Core 2.0 应用程序。由于 Web 应用程序要持续运行，因此该应用程序将作为一项服务部署。

在您发布容器之前

在使用将容器发布到 AWS 部署 ASP.NET Core 2.0 应用程序之前，请执行以下操作：

- 指定 AWS 凭证并设置以使用 Amazon ECS。
- 安装 Docker。有几个不同的安装选项，包括 Docker for Windows。
- 根据 Web 应用程序的需求创建 Amazon ECS 集群。只需几个步骤即可完成。
- 在 Visual Studio 中创建（或打开）一个针对 Linux 的 ASP.NET Core 2.0 容器化应用程序的项目。

访问 “将容器发布到 AWS ” 向导

要部署针对 Linux 的 ASP.NET Core 2.0 容器化应用程序，请右键单击解决方案浏览器中的项目并选择将容器发布到 AWS。

您还可以在 Visual Studio 的“构建”菜单中选择将容器发布到 AWS。

将容器发布到 AWS 向导

Account profile to use (要使用的账户配置文件) - 选择要使用的账户配置文件。

Region (区域) - 选择一个部署区域。配置文件和区域用于设置您的部署环境资源并选择默认的 Docker 注册表。

Configuration (配置) - 选择 Docker 映像构建配置。

Docker Repository (Docker 存储库) - 选择现有 Docker 存储库，或键入新存储库的名称即可创建新存储库。这是构建的容器映像要推送到的存储库。

Tag (标签) - 选择现有标签或键入新标签的名称。标签可以跟踪重要详细信息，如 Docker 容器的版本、选项或其他唯一配置元素。

Deployment (部署) – 选择 Service on an ECS Cluster (ECS 集群上的服务)。当您的应用程序（如 ASP.NET 内核 2.0 Web 应用程序）计划长时间运行时，请使用此部署选项。

将设置保存到 **`aws-docker-tools-defaults.json`** 并为命令行部署配置项目 - 如果您需要从命令行灵活部署，请选中此选项。使用您的项目目录中的 `dotnet ecs deploy` 以部署和 `dotnet ecs publish` 容器。

“Launch Configuration (启动配置)”页面

ECS Cluster (ECS 集群) – 选择将运行 Docker 映像的集群。您可以使用 AWS 管理控制台创建 ECS 集群。

Launch Type (启动类型) - 选择 EC2。要使用 Fargate 启动类型，请参阅将 ASP.NET 内核 2.0 应用程序部署到 Amazon ECS (Fargate)。

“Service Configuration (服务配置)”页面

Service (服务) - 从下拉框中选择一项服务，将您的容器部署到该现有服务。或者选择 Create New (新建) 新建一项服务。一个集群中的服务名称必须唯一，但是您可以为一个区域或多个区域中多个集群中的服务提供相似的名称。

Number of Tasks (任务数) - 要在您的集群中部署并保持运行的任务数量。每个任务都是您的容器的一个实例。

Minimum Healthy Percent (最小正常运行状况百分比) - 在部署期间必须处于 RUNNING 状态的任务百分比（四舍五入到最近的整数）。

Maximum Percent (最大百分比) - 在部署期间允许处于 RUNNING 或 PENDING 状态的任务百分比（向下舍入到最近的整数）。

Placement Templates (放置模板) – 选择任务放置模板。

如果您在集群中启动任务，Amazon ECS 必须根据任务定义中指定的要求（例如 CPU 和内存）确定将任务放置在何处。同样，如果您缩减任务计数，Amazon ECS 必须确定终止哪些任务。

放置模板用于控制任务如何在集群中启动：

- AZ Balanced Spread (AZ 均衡分散) - 在各个可用区以及每个可用区中的各个容器实例中分配任务。
- 可用区平衡 BinPack -在可用内存最少的可用区域和容器实例之间分配任务。
- BinPack -根据最少可用的 CPU 或内存量来分配任务。
- One Task Per Host (每个主机一项任务) - 在每个容器实例中最多可放置服务的一个任务。

有关更多信息，请参阅 Amazon ECS 任务放置。

“Application Load Balancer (应用程序负载均衡器)”页面

Configure Application Load Balancer (配置应用程序负载均衡器) - 选中此项可配置应用程序负载均衡器。

Select IAM role for service (为服务选择 IAM 角色) – 选择一个现有角色，或者选择 Create New (新建) 即可创建一个新角色。

Load Balancer (负载均衡器) - 选择一个现有负载均衡器，或者选择 Create New (新建) 并键入新负载均衡器的名称。

Listener Port (侦听器端口) - 选择一个现有侦听器端口，或者选择 Create New (新建) 并键入一个端口号。默认端口 80 适用于大多数 Web 应用程序。

Target Group (目标组) - 默认情况下，负载均衡器使用您为目标组指定的端口和协议将请求发送到已注册目标。在将每个目标注册到目标组时，可以覆盖此端口。

Path Pattern (路径模式) - 负载均衡器将使用基于路径的路由。接受默认 / 或提供一个不同模式。路径模式区分大小写，长度最多为 128 个字符，并且可包含一组选定字符。

Health Check Path (运行状况检查路径) - 进行运行状况检查的目标上的目的地的 Ping 路径。默认为 /，该设置适用于 Web 应用程序。输入不同的路径（如果需要）。如果您输入的路径无效，则运行状况检查将失败，并将视为运行状况不佳。

如果您要部署多个服务，且每个服务都将部署到不同的路径或位置，您可能需要自定义检查路径。

“ECS Task Definition (ECS 任务定义)”页面

Task Definition (任务定义) - 选择一个现有任务定义，或者选择 Create New (新建) 并键入新任务定义的名称。

Container (容器) - 选择一个现有容器，或者选择 Create New (新建) 并键入新容器的名称。

Memory (MiB) (内存(MiB)) - 提供软限制和/或硬限制的值。

要为容器预留的内存量的软限制（以 MiB 为单位）。Docker 尝试将容器内存控制在软限制以下。如果容器需要消耗更多内存，则上限为内存参数指定的硬限制（如果适用）或者容器实例中的全部可用内存，以较低者为准。

要提供给容器的内存的硬限制（以 MiB 为单位）。如果容器尝试使用超出此处指定的内存，该容器将被终止。

任务角色-为 IAM 角色选择一个任务角色，该角色允许容器代表您调用其关联策略中指定的权限。AWS APIs 凭证就是通过这种方法传递给您的应用程序的。了解如何为您的应用程序指定 AWS 安全证书。

Port Mapping (端口映射) – 添加、修改或删除容器的端口映射。如果负载均衡器为开启状态，主机端口将默认设置为 0，端口分配采用动态方式。

Environment Variables (环境变量) - 添加、修改或删除容器的环境变量。

如果您对配置满意，请单击 Publish (发布) 开始部署流程。

将容器发布到 AWS

在部署过程中会显示事件。成功完成后向导会自动关闭。您可以通过取消选中页面底部的复选框来覆盖该功能。

您可以在 AWS 资源管理器中找到新实例的 URL。展开 Amazon ECS 和集群，然后单击您的集群。

疑难解答 AWS Toolkit for Visual Studio

以下各节包含有关 AWS Toolkit for Visual Studio 和使用该工具包中的 AWS 服务的一般疑难解答信息。

> **Note**
>
> 安装和 set-up-specific疑难解答信息可在本用户指南中的安装问题疑难解答主题中找到。

主题

问题排查最佳实践

以下是推荐使用的 AWS Toolkit for Visual Studio 问题排查最佳实践。

- 修复 Visual Studio 并重新启动系统
- 在发送报告之前，尝试重现问题或错误。
- 详细记录重现过程中的每个步骤、设置和错误消息。
- 收集 AWS 工具包日志。有关如何查找 AWS Toolkit 日志的详细说明，请参阅本指南主题中的 “如何找到您的 AWS 日志” 过程。
- 查看未解决的请求、已知的解决方案，或者在 AWS Toolkit for Visual Studio GitHub 存储库的 “AWS Toolkit for Visual Studio 问题” 部分报告未解决的问题。

修复 Visual Studio 并重新启动系统

1. 关闭所有正在运行的 Visual Studio 实例。
2. 从 Windows“开始”菜单中启动 Visual Studio 安装程序。
3. 对受影响的 Visual Studio 安装运行“修复”。这允许 Visual Studio 重新构建其已安装扩展的索引。
4. 在重新启动 Visual Studio 之前重新启动 Windows。

如何找到你的 AWS Toolkit 日志

1. 在 Visual Studio 主菜单中，展开扩展。
2. 选择AWS 工具包以展开 Tool AWS kit 菜单，然后选择查看 Toolkit 日志。
3. 当 AWS Toolkit 日志文件夹在您的操作系统中打开时，按日期对文件进行排序，然后找到任何包含与当前问题相关的信息的日志文件。

查看和筛选 Amazon Q 安全扫描

要在 Visual Studio 中查看 Amazon Q 安全扫描，请通过在 Visual Studio 主菜单中展开查看标题并选择错误列表，来打开 Visual Studio 错误列表。

默认情况下，Visual Studio 错误列表会显示您的代码库的所有警告和错误。要从 Visual Studio 错误列表中筛选出 Amazon Q 安全扫描结果，请完成以下流程以创建筛选器。

> **Note**
>
> 只有当运行了安全扫描并且检测出问题后，才会显示 Amazon Q 安全扫描结果。Amazon Q 安全扫描结果会以警告形式显示在 Visual Studio 中。要从错误列表查看 Amazon Q 安全扫描结果，必须选择错误列表标题中的警告选项。

1. 从 Visual Studio 主菜单中展开查看标题，选择错误列表以打开错误列表窗格。
2. 从错误列表窗格中，右键单击标题行，以打开上下文菜单。
3. 在上下文菜单中，展开显示列，然后在展开的菜单中选择工具。
4. 工具列已添加到您的错误列表中。
5. 从工具列标题中，选择筛选图标，选择 Amazon Q 以筛选 Amazon Q 安全扫描结果。

AWS 工具包安装不正确

问题：

在启动 Visual Studio 后的一分钟内，输出窗格和信息栏中将分别显示以下消息： AWS Toolkit for Visual Studio

```
Some Toolkit components could not be initialized. Some functionality may
not work during this IDE session.
```

```
The AWS Toolkit is not properly installed.
```

解决方案：

更新或安装扩展程序可能会导致 Visual Studio 的某些内部缓存文件失效 out-of-sync。以下过程介绍如何让 Visual Studio 在下次启动时重新生成这些文件。

> **Note**
>
> 此解决方案可能会影响您的 Visual Studio 自定义设置。完成此过程后， AWS Toolkit 扩展应列为已安装且不再报告错误消息。如果您在完成以下步骤后仍然遇到此问题，请参阅 AWS Toolkit for Visual Studio GitHub 存储库中的问题 #452 以获取更多信息。

1. 安装 Visual Studio 2022 的最新版本。

   > **Note**
   >
   > 所需的最低版本为 17.11.5。

2. 关闭所有正在运行的 Visual Studio 实例。
3. 在 Windows 系统中，以管理员身份打开开发者命令提示符。
4. 在开发者命令提示符下，运行以下命令：`devenv /updateconfiguration / resetExtensions`，然后等待命令完成。
5. 命令完成后，重新启动 Visual Studio。
6. 在 Visual Studio 中，该 AWS 扩展现在列为已安装，不再报告此问题顶部列出的错误消息。

防火墙和代理设置

排查防火墙和代理设置故障

安全扫描软件可能会删除您从 AWS Toolkit 语言服务器下载的文件或完全阻止下载，从而干扰您的下载能力。

要检查您的防火墙和代理设置，请使用与 Visual https://aws-toolkit-language-servers.amazonaws.com/codewhisperer/Studio 实例安装在同一系统上的互联网浏览器导航到 0/manifest.json。如果您遇到错误或页面无法加载，则表明可能有防火墙或代理过滤器在阻止您访问 `aws-toolkit-language-servers.amazonaws.com`。

自定义证书

AWS Toolkit for Visual Studio 使用在 Node.js 运行时上运行的语言服务器。有关如何检查您的网络是否使用自定义证书的详细信息，请参阅《AWS Command Line Interface用户指南（版本 1）》中的AWS CLI中的配置和凭证文件设置。

要配置代理设置和定义证书，必须配置 HTTPS_PROXY 环境变量并为 NODE_OPTIONS 和 NODE_EXTRA_CA_CERTS 键创建 Windows 环境变量。

要配置 HTTPS_PROXY 环境变量，请完成以下步骤：

1. 从 Visual Studio 主菜单中选择工具，然后选择选项。
2. 从选项菜单中展开 AWS Toolkit，然后选择代理。
3. 在代理菜单中，定义您的主机和端口。

> **Note**
>
> 有关HTTPS_PROXY从中配置的信息 AWS CLI，请参阅《AWS Command Line Interface用户指南》中该 AWS CLI主题的 “使用 HTTP 代理”。

为以下键创建 Windows 环境变量。

- `NODE_OPTIONS = --use-openssl-ca`
- `NODE_EXTRA_CA_CERTS = Path/To/Corporate/Certs`

> **Note**
>
> 有关提取企业根证书的更多信息，请参阅 learn.microsoft.com 上的使用私有密钥导出证书一文。有关 Windows 环境变量键的详细信息，请参阅 nodejs.org 上的 Node.js v23.3.0 文档。

允许列表和额外步骤

除了干扰 AWS Toolkit 语言服务器外，防火墙设置还可以阻止 Amazon Q 上传到 Amazon S3 并调用服务 API。为了尽量避免这些错误，建议允许在端口 443（HTTPS）上对以下端点进行出站互联网访问：

- 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

有关端点的详细列表，请参阅本用户指南中的更新防火墙和网关以允许访问主题。有关为 Amazon Q 配置公司代理的详细信息，请参阅《Amazon Q 开发者版用户指南》中的在 Amazon Q 中配置公司代理主题。如果您仍然遇到防火墙和代理问题，请收集您的 AWS Toolkit 日志，并通过 AWS Toolkit for Visual Studio GitHub 存储库的 “AWS Toolkit for Visual Studio 问题” 部分与 AWS Toolkit for Visual Studio 团队联系。有关收集 AWS Toolkit 日志的详细信息，请查看本用户指南主题的 “故障排除最佳实践” 部分中的信息。

的安全性 AWS Toolkit for Visual Studio

云安全性一直是 Amazon Web Services（AWS）的重中之重。作为 AWS 客户，您将从专为满足大多数安全敏感型企业的要求而打造的数据中心和网络架构中受益。安全是双方共同承担 AWS 的责任。责任共担模式将其描述为云的安全性和云中的安全性。

云安全 — AWS 负责保护运行 AWS 云中提供的所有服务的基础架构，并为您提供可以安全使用的服务。我们的安全责任是重中之重 AWS，作为AWS 合规计划的一部分，第三方审计师定期测试和验证我们安全的有效性。

云端安全 — 您的责任由您使用的 AWS 服务以及其他因素决定，包括数据的敏感性、组织的要求以及适用的法律和法规。

本 AWS 产品或服务通过其支持的特定 Amazon Web Services (AWS) 服务遵循分担责任模式。有关 AWS 服务安全信息，请参阅AWS 服务安全文档页面和合规计划合 AWS 规工作范围内的AWS 服务。

主题

- 中的数据保护 AWS Toolkit for Visual Studio
- 身份和访问管理
- 此 AWS 产品或服务的合规性验证
- 本 AWS 产品或服务的弹性
- 本 AWS 产品或服务的基础设施安全
- 中的配置和漏洞分析 AWS Toolkit for Visual Studio

中的数据保护 AWS Toolkit for Visual Studio

AWS 分担责任模型分 AWS AWS Cloud您负责维护对托管在此基础结构上的内容的控制。您还负责您所使用的 AWS 服务 的安全配置和管理任务。有关数据隐私的更多信息，请参阅数据隐私常见问题。有关欧洲数据保护的信息，请参阅 AWS Security Blog 上的 AWS Shared Responsibility Model and GDPR 博客文章。

出于数据保护目的，我们建议您保护 AWS 账户 凭证并使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 设置个人用户。这样，每个用户只获得履行其工作职责所需的权限。还建议您通过以下方式保护数据：

- 对每个账户使用多重身份验证（MFA）。
- 用于 SSL/TLS 与 AWS 资源通信。我们要求使用 TLS 1.2，建议使用 TLS 1.3。

- 使用设置 API 和用户活动日志 AWS CloudTrail。有关使用 CloudTrail 跟踪捕获 AWS 活动的信息，请参阅《AWS CloudTrail 用户指南》中的使用跟 CloudTrail 踪。
- 使用 AWS 加密解决方案以及其中的所有默认安全控件 AWS 服务。
- 使用高级托管安全服务（例如 Amazon Macie），它有助于发现和保护存储在 Amazon S3 中的敏感数据。
- 如果您在 AWS 通过命令行界面或 API 进行访问时需要经过 FIPS 140-3 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅《美国联邦信息处理标准（FIPS）第 140-3 版》https://aws.amazon.com/compliance/fips/。

强烈建议您切勿将机密信息或敏感信息（如您客户的电子邮件地址）放入标签或自由格式文本字段（如名称字段）。这包括您使用控制台、API 或 Amazon Q 或其他 AWS 服务 方式使用 AWS Toolkit 时 AWS SDKs。 AWS CLI在用于名称的标签或自由格式文本字段中输入的任何数据都可能会用于计费或诊断日志。如果您向外部服务器提供 URL，强烈建议您不要在网址中包含凭证信息来验证对该服务器的请求。

身份和访问管理

AWS Identity and Access Management (IAM) AWS 服务 可帮助管理员安全地控制对 AWS 资源的访问权限。IAM 管理员控制谁可以进行身份验证（登录）和授权（有权限）使用 AWS 资源。您可以使用 IAM AWS 服务 ，无需支付额外费用。

主题

- 受众
- 使用身份进行身份验证
- 使用策略管理访问
- 如何 AWS 服务 使用 IAM
- 对 AWS 身份和访问进行故障排除

受众

您的使用方式 AWS Identity and Access Management (IAM) 会有所不同，具体取决于您所做的工作 AWS。

服务用户-如果您 AWS 服务 曾经完成工作，则您的管理员会为您提供所需的凭证和权限。当你使用更多 AWS 功能来完成工作时，你可能需要额外的权限。了解如何管理访问权限有助于您向管理员请求适

合的权限。如果您无法访问中的功能 AWS，请参阅对 AWS 身份和访问进行故障排除或 AWS 服务 您正在使用的用户指南。

服务管理员-如果您负责公司的 AWS 资源，则可能拥有完全访问权限 AWS。您的工作是确定您的服务用户应访问哪些 AWS 功能和资源。然后，您必须向 IAM 管理员提交请求以更改服务用户的权限。请查看该页面上的信息以了解 IAM 的基本概念。要详细了解您的公司如何使用 IAM AWS，请参阅 AWS 服务 您正在使用的用户指南。

IAM 管理员：如果您是 IAM 管理员，您可能希望了解如何编写策略以管理对 AWS的访问权限的详细信息。要查看您可以在 IAM 中使用的 AWS 基于身份的策略示例，请参阅 AWS 服务 您正在使用的用户指南。

使用身份进行身份验证

身份验证是您 AWS 使用身份凭证登录的方式。您必须以 IAM 用户身份进行身份验证 AWS 账户根用户，或者通过担任 IAM 角色进行身份验证。

您可以使用来自身份源的证书 AWS IAM Identity Center（例如（IAM Identity Center）、单点登录身份验证或 Google/Facebook 证书，以联合身份登录。有关登录的更多信息，请参阅《AWS 登录 用户指南》中的如何登录您的 AWS 账户。

对于编程访问，AWS 提供 SDK 和 CLI 来对请求进行加密签名。有关更多信息，请参阅《IAM 用户指南》中的适用于 API 请求的AWS 签名版本 4。

AWS 账户 root 用户

创建时 AWS 账户，首先会有一个名为 AWS 账户 root 用户的登录身份，该身份可以完全访问所有资源 AWS 服务 和资源。我们强烈建议不要使用根用户进行日常任务。有关需要根用户凭证的任务，请参阅《IAM 用户指南》中的需要根用户凭证的任务。

联合身份

作为最佳实践，要求人类用户使用与身份提供商的联合身份验证才能 AWS 服务 使用临时证书进行访问。

联合身份是指来自您的企业目录、Web 身份提供商的用户 Directory Service ，或者 AWS 服务 使用来自身份源的凭据进行访问的用户。联合身份代入可提供临时凭证的角色。

要集中管理访问权限，建议使用。 AWS IAM Identity Center有关更多信息，请参阅《AWS IAM Identity Center 用户指南》中的什么是 IAM Identity Center？。

IAM 用户和群组

IAM 用户是对某个人员或应用程序具有特定权限的一个身份。建议使用临时凭证，而非具有长期凭证的 IAM 用户。有关更多信息，请参阅 IAM 用户指南中的要求人类用户使用身份提供商的联合身份验证才能 AWS 使用临时证书进行访问。

IAM 组指定一组 IAM 用户，便于更轻松地对大量用户进行权限管理。有关更多信息，请参阅《IAM 用户指南》中的 IAM 用户使用案例。

IAM 角色

IAM 角色是具有特定权限的身份，可提供临时凭证。您可以通过从用户切换到 IAM 角色（控制台）或调用 AWS CLI 或 AWS API 操作来代入角色。有关更多信息，请参阅《IAM 用户指南》中的担任角色的方法。

IAM 角色对于联合用户访问、临时 IAM 用户权限、跨账户访问、跨服务访问以及在 Amazon EC2 上运行的应用程序非常有用。有关更多信息，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

使用策略管理访问

您可以 AWS 通过创建策略并将其附加到 AWS 身份或资源来控制中的访问权限。策略定义了与身份或资源关联时的权限。 AWS 在委托人提出请求时评估这些政策。大多数策略都以 JSON 文档的 AWS 形式存储在中。有关 JSON 策略文档的更多信息，请参阅《IAM 用户指南》中的 JSON 策略概述。

管理员使用策略，通过定义哪个主体可以在什么条件下对哪些资源执行哪些操作来指定谁有权访问什么。

默认情况下，用户和角色没有权限。IAM 管理员创建 IAM 策略并将其添加到角色中，然后用户可以担任这些角色。IAM 策略定义权限，与执行操作所用的方法无关。

基于身份的策略

基于身份的策略是您附加到身份（用户、组或角色）的 JSON 权限策略文档。这些策略控制身份可以执行什么操作、对哪些资源执行以及在什么条件下执行。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的使用客户管理型策略定义自定义 IAM 权限。

基于身份的策略可以是内联策略（直接嵌入到单个身份中）或托管策略（附加到多个身份的独立策略）。要了解如何在托管策略和内联策略之间进行选择，请参阅《IAM 用户指南》中的在托管策略与内联策略之间进行选择。

基于资源的策略

基于资源的策略是附加到资源的 JSON 策略文档。示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。您必须在基于资源的策略中指定主体。

基于资源的策略是位于该服务中的内联策略。您不能在基于资源的策略中使用 IAM 中的 AWS 托管策略。

访问控制列表 (ACLs)

访问控制列表 (ACLs) 控制哪些委托人（账户成员、用户或角色）有权访问资源。 ACLs 与基于资源的策略类似，尽管它们不使用 JSON 策略文档格式。

Amazon S3 和 Amazon VPC 就是支持的服务示例 ACLs。 AWS WAF要了解更多信息 ACLs，请参阅《亚马逊简单存储服务开发者指南》中的访问控制列表 (ACL) 概述。

其他策略类型

AWS 支持其他策略类型，这些策略类型可以设置更常见的策略类型授予的最大权限：

- 权限边界 – 设置基于身份的策略可以授予 IAM 实体的最大权限。有关更多信息，请参阅《 IAM 用户指南》中的 IAM 实体的权限边界。
- 服务控制策略 (SCPs)-在中指定组织或组织单位的最大权限 AWS Organizations。有关更多信息，请参阅《AWS Organizations 用户指南》中的服务控制策略。
- 资源控制策略 (RCPs)-设置账户中资源的最大可用权限。有关更多信息，请参阅《AWS Organizations 用户指南》中的资源控制策略 (RCPs)。
- 会话策略 – 在为角色或联合用户创建临时会话时，作为参数传递的高级策略。有关更多信息，请参阅《IAM 用户指南》中的会话策略。

多个策略类型

当多个类型的策略应用于一个请求时，生成的权限更加复杂和难以理解。要了解在涉及多种策略类型时如何 AWS 确定是否允许请求，请参阅 IAM 用户指南中的策略评估逻辑。

如何 AWS 服务 使用 IAM

要全面了解如何 AWS 服务 使用大多数 IAM 功能，请参阅 IAM 用户指南中的与 IAM 配合使用的AWS 服务。

要了解如何在 IAM 中 AWS 服务 使用特定的，请参阅相关服务的用户指南的安全部分。

对 AWS 身份和访问进行故障排除

使用以下信息来帮助您诊断和修复在使用 AWS 和 IAM 时可能遇到的常见问题。

主题

我无权在以下位置执行操作 AWS

如果您收到错误提示，指明您无权执行某个操作，则必须更新策略以允许执行该操作。

当 mateojackson IAM 用户尝试使用控制台查看有关虚构 *my-example-widget* 资源的详细信息，但不拥有虚构 awes:*GetWidget* 权限时，会发生以下示例错误。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 awes:GetWidget on resource: my-example-widget
```

在此情况下，必须更新 mateojackson 用户的策略，以允许使用 awes:*GetWidget* 操作访问 *my-example-widget* 资源。

如果您需要帮助，请联系您的 AWS 管理员。您的管理员是提供登录凭证的人。

我无权执行 iam：PassRole

如果您收到一个错误，表明您无权执行 iam:PassRole 操作，则必须更新策略以允许您将角色传递给。 AWS

有些 AWS 服务 允许您将现有角色传递给该服务，而不是创建新的服务角色或服务相关角色。为此，您必须具有将角色传递到服务的权限。

当名为 marymajor 的 IAM 用户尝试使用控制台在 AWS中执行操作时，会发生以下示例错误。但是，服务必须具有服务角色所授予的权限才可执行此操作。Mary 不具有将角色传递到服务的权限。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在这种情况下，必须更新 Mary 的策略以允许她执行 `iam:PassRole` 操作。

如果您需要帮助，请联系您的 AWS 管理员。您的管理员是提供登录凭证的人。

我想允许我以外的人 AWS 账户 访问我的 AWS 资源

您可以创建一个角色，以便其他账户中的用户或您组织外的人员可以使用该角色来访问您的资源。您可以指定谁值得信赖，可以代入角色。对于支持基于资源的策略或访问控制列表 (ACLs) 的服务，您可以使用这些策略向人们授予访问您的资源的权限。

要了解更多信息，请参阅以下内容：

- 要了解是否 AWS 支持这些功能，请参阅如何 AWS 服务 使用 IAM。
- 要了解如何提供对您拥有的资源的访问权限 AWS 账户 ，请参阅 IAM 用户指南中的向您拥有 AWS 账户 的另一个 IAM 用户提供访问权限。
- 要了解如何向第三方提供对您的资源的访问权限 AWS 账户，请参阅 IAM 用户指南中的向第三方提供访问权限。 AWS 账户
- 要了解如何通过身份联合验证提供访问权限，请参阅《IAM 用户指南》中的为经过外部身份验证的用户（身份联合验证）提供访问权限。
- 要了解使用角色和基于资源的策略进行跨账户访问之间的差别，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

此 AWS 产品或服务的合规性验证

要了解是否属于特定合规计划的范围，请参阅AWS 服务 “按合规计划划分的范围” ”，然后选择您感兴趣的合规计划。 AWS 服务 有关一般信息，请参阅AWS 合规计划AWS。

您可以使用下载第三方审计报告 AWS Artifact。有关更多信息，请参阅中的 “下载报告” 中的 “ AWS Artifact。

您在使用 AWS 服务 时的合规责任取决于您的数据的敏感性、贵公司的合规目标以及适用的法律和法规。有关您在使用时的合规责任的更多信息 AWS 服务，请参阅AWS 安全文档。

本 AWS 产品或服务通过其支持的特定 Amazon Web Services (AWS) 服务遵循分担责任模式。有关 AWS 服务安全信息，请参阅AWS 服务安全文档页面和合规计划合 AWS 规工作范围内的AWS 服务。

本 AWS 产品或服务的弹性

AWS 全球基础设施是围绕 AWS 区域 可用区构建的。

AWS 区域 提供多个物理隔离和隔离的可用区，这些可用区通过低延迟、高吞吐量和高度冗余的网络连接。

利用可用区，您可以设计和操作在可用区之间无中断地自动实现失效转移的应用程序和数据库。与传统的单个或多个数据中心基础设施相比，可用区具有更高的可用性、容错能力和可扩展性。

有关 AWS 区域和可用区的更多信息，请参阅AWS 全球基础设施。

本 AWS 产品或服务通过其支持的特定 Amazon Web Services (AWS) 服务遵循分担责任模式。有关 AWS 服务安全信息，请参阅AWS 服务安全文档页面和合规计划合 AWS 规工作范围内的AWS 服务。

本 AWS 产品或服务的基础设施安全

本 AWS 产品或服务使用托管服务，因此受到 AWS 全球网络安全的保护。有关 AWS 安全服务以及如何 AWS 保护基础设施的信息，请参阅AWS 云安全。要使用基础设施安全的最佳实践来设计您的 AWS 环境，请参阅 S AWS ecurity Pillar Well-Architected Fram ework 中的基础设施保护。

您可以使用 AWS 已发布的 API 调用通过网络访问此 AWS 产品或服务。客户端必须支持以下内容：

- 传输层安全性协议（TLS）。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 具有完全向前保密（PFS）的密码套件，例如 DHE（临时 Diffie-Hellman）或 ECDHE（临时椭圆曲线 Diffie-Hellman）。大多数现代系统（如 Java 7 及更高版本）都支持这些模式。

此外，必须使用访问密钥 ID 和与 IAM 主体关联的秘密访问密钥来对请求进行签名。或者，您可以使用 AWS Security Token Service（AWS STS）生成临时安全凭证来对请求进行签名。

本 AWS 产品或服务通过其支持的特定 Amazon Web Services (AWS) 服务遵循分担责任模式。有关 AWS 服务安全信息，请参阅AWS 服务安全文档页面和合规计划合 AWS 规工作范围内的AWS 服务。

中的配置和漏洞分析 AWS Toolkit for Visual Studio

随着新功能或修补程序的开发，Toolkit for Visual Studio 会发布到 Visual Studio Marketplace。这些更新有时会包含安全更新，因此请务必更新带有 Amazon Q 的 AWS Toolkit。

验证是否已为扩展启用自动更新

1. 通过选择工具、扩展和更新（Visual Studio 2017）或扩展、管理扩展（Visual Studio 2019）打开扩展管理器。
2. 选择更改扩展和更新设置（Visual Studio 2017）或更改扩展的设置（Visual Studio 2019）。

3. 调整环境设置。

如果选择对扩展禁用自动更新，请务必按适合您环境的间隔检查 AWS Toolkit with Amazon Q 更新。

AWS Toolkit for Visual Studio 用户指南的文档历史记录

文档历史记录

下表描述了 AWS Toolkit for Visual Studio 用户指南的最新重要更改。如需获取对此文档的更新的通知，您可以订阅 RSS 源。

变更	说明	日期
更新“开始使用”相关内容	更新“开始使用”和“连接到 AWS”内容，以反映用户界面中所做的更改。	2025 年 4 月 24 日
更新防火墙和网关以允许访问	列表包含为访问具有扩展功能的 AWS Toolkit for Visual Studio with Amazon Q 中的所有服务和功能而必须纳入允许列表的端点和资源。	2025 年 3 月 20 日
排查防火墙和代理设置故障	增加了新的故障排除主题，以解决有关 AWS Toolkit for Visual Studio 和 Amazon Q 的防火墙和代理设置问题。	2024 年 12 月 15 日
排查安装问题更新	更新安装问题相关内容以考虑来自 Microsoft 的一项更新。	2024 年 11 月 20 日
更新“开始使用”相关内容	更新“开始使用”和“连接到 AWS”内容，以反映用户界面中所做的更改。	2024 年 10 月 24 日
更新“连接到 AWS”	更新“连接到 AWS”相关内容。	2024 年 9 月 26 日
Amazon EC2 AMI 内容更新	已对内容进行更新，以记录 Amazon EC2 AMI 流程和过程更改。	2024 年 9 月 13 日

无法初始化 AWS Toolkit 组件	增加了一个故障排除主题以解决 AWS Toolkit for Visual Studio 组件不能初始化的问题。	2024 年 9 月 13 日
查看和筛选 Amazon Q 安全扫描	增加了一个故障排除主题以帮助查看和筛选 Amazon Q 安全扫描。	2024 年 7 月 31 日
Amazon Q for AWS Toolkit for Visual Studio	Amazon Q 现在已集成到 AWS Toolkit for Visual Studio 中。	2024 年 6 月 30 日
内容更新和维护	更新内容以反映对用户界面和 AWS 风格指南的更改。	2024 年 3 月 6 日
内容更新和维护	更新内容以反映对用户界面和 AWS 风格指南的更改。	2024 年 3 月 6 日
内容更新和维护	更新内容以反映对用户界面和 AWS 风格指南的更改。	2024 年 3 月 6 日
内容更新和维护	更新内容以反映对用户界面和 AWS 风格指南的更改。	2024 年 3 月 6 日
内容更新和维护	更新内容以反映对用户界面和 AWS 风格指南的更改。	2024 年 3 月 6 日
设置和身份验证的更新	设置和身份验证主题已更新，以改善安全性和 Toolkit 入门体验。要查看更改，请参阅入门和身份验证和访问主题目录。	2023 年 6 月 22 日
身份验证和访问	“提供 AWS 凭证”现在名为“身份验证和访问”。重构目录和子主题以满足 AWS 风格和安全要求。	2023 年 5 月 4 日

设置部分和主题的更新	本用户指南中的设置 AWS Toolkit for Visual Studio 部分和主题已更新，以改善 AWS Toolkit for Visual Studio 入门体验。	2023 年 1 月 30 日
设置部分和主题的更新	本用户指南中的设置 AWS Toolkit for Visual Studio 部分和主题已更新，以改善 AWS Toolkit for Visual Studio 入门体验。	2023 年 1 月 30 日
添加了 2022 AWS Toolkit for Visual Studio 的信息	已在 AWS Toolkit for Visual Studio 中增加对 Visual Studio 2022 的支持。	2022 年 12 月 20 日
“发布到 AWS”指南的更新	文档更新反映了对 GA 发布服务所做的更改。	2022 年 7 月 6 日
标题更新和位置调整	为了更好地反映内容，对标题进行了细微更改。本指南现在位于“发布到 AWS”指南中。	2022 年 7 月 6 日
部署到 AWS：标题和内容更新	指南的此部分之前的标题为“使用 AWS Toolkit 进行部署”，现在更新了此部分的目录（TOC），并使用标题“部署到 AWS”。以下指南已完成弃用，无法再访问：部署到 Elastic Beanstalk（旧版）和部署到 AWS CloudFormation（旧版）。有关部署到 Elastic Beanstalk 和 CloudFormation 的更新内容可从本指南更新的目录中找到。	2022 年 7 月 6 日

部署 ASP.NET Core 2.0 应用程序（Fargate）”现已属于旧版指南	本文档涉及旧版服务和功能。有关更新的指南和内容，请参阅 AWS .NET deployment tool 指南和更新的部署到 AWS 目录。	2022 年 7 月 6 日
部署 ASP.NET 应用程序”现已属于旧版指南	本文档涉及旧版服务和功能。有关更新的指南和内容，请参阅 AWS .NET deployment tool 指南和更新的部署到 AWS 目录。	2022 年 7 月 6 日
部署 ASP.NET 应用程序”现已属于旧版指南	本文档涉及旧版服务和功能。有关更新的指南和内容，请参阅 AWS .NET deployment tool 指南和更新的部署到 AWS 目录。	2022 年 7 月 6 日
新指南主题：在 Visual Studio 中使用 CloudWatch Logs	为 Visual Studio 中的 Amazon CloudWatch Logs 集成指南创建了新的概述主题。	2022 年 6 月 29 日
新指南主题：为 Visual Studio 设置 CloudWatch Logs 集成	为 Visual Studio 中的 Amazon CloudWatch Logs 集成指南创建了新的设置部分。	2022 年 6 月 29 日
Visual Studio 的 CloudWatch Logs 集成	为 Visual Studio 中的 Amazon CloudWatch Logs 集成创建了新指南，包括的指南主题有：为 Visual Studio 设置 CloudWatch Logs 和在 Visual Studio 中使用 CloudWatch Logs。	2022 年 6 月 29 日
Publish to AWS (发布到 CloudWatch)	“发布到 AWS”不再处于预览状态。更新反映了用户界面的更改和发布建议的改进。	2022 年 6 月 1 日

全新“发布到 AWS”可供预览	增强的部署体验，可指导您选择哪项 AWS 服务适合您的应用程序。	2021 年 10 月 21 日
针对 AWS 凭证的 SSO 和 MFA 支持	更新以记录对 AWS 单点登录（IAM Identity Center）和 AWS 凭证中的多因素身份验证的新支持。	2021 年 4 月 21 日
创建 Docker 映像的基本 AWS Lambda 项目	添加了对 Lambda 容器映像的支持。	2020 年 12 月 1 日
安全性内容	增加了安全性内容。	2020 年 2 月 6 日
提供 AWS 凭证	更新了有关在共享的 AWS 凭证文件中创建凭证配置文件的信息。	2019 年 6 月 20 日
在 AWS Toolkit for Visual Studio 中使用 AWS Lambda 项目	已将对 Visual Studio 2019 的支持添加到 AWS Toolkit for Visual Studio 中。	2019 年 3 月 28 日
教程：创建 Amazon Rekognition Lambda 应用程序	已将对 Visual Studio 2019 的支持添加到 AWS Toolkit for Visual Studio 中。	2019 年 3 月 28 日
教程：使用 AWS Lambda 构建和测试无服务器应用程序	已将对 Visual Studio 2019 的支持添加到 AWS Toolkit for Visual Studio 中。	2019 年 3 月 28 日
设置 AWS Toolkit for Visual Studio	已在 AWS Toolkit for Visual Studio 中增加对 Visual Studio 2019 的支持。	2019 年 3 月 28 日
部署 ASP.NET Core 2.0 应用程序（Fargate）	已将对 Visual Studio 2019 的支持添加到 AWS Toolkit for Visual Studio 中。	2019 年 3 月 28 日

部署 ASP.NET Core 2.0 应用程序（EC2）	已将对 Visual Studio 2019 的支持添加到 AWS Toolkit for Visual Studio 中。	2019 年 3 月 28 日
在 Visual Studio 中创建 AWS CloudFormation 模板项目	已将对 Visual Studio 2019 的支持添加到 AWS Toolkit for Visual Studio 中。	2019 年 3 月 28 日
Container Service 的详细视图	添加了有关 AWS 各区服务浏览器提供的 Amazon Elastic Container Service 集群和容器存储库的详细视图的信息。	2018 年 2 月 16 日
部署到 Amazon EC2 Container Service	添加了有关部署到 Amazon EC2 Container Service 的信息。	2018 年 2 月 16 日
使用 Fargate 部署 Container Service	添加了有关如何使用 Fargate 启动类型通过 Amazon ECS 部署针对 Linux 的容器化 ASP.NET Core 2.0 应用程序的信息。	2018 年 2 月 16 日
使用 EC2 部署 Container Service	添加了有关如何使用 EC2 启动类型通过 Amazon ECS 部署针对 Linux 的容器化 ASP.NET Core 2.0 应用程序的信息。	2018 年 2 月 16 日
用于部署到 Amazon EC2 Container Service 的凭证	添加了有关如何在部署到 Amazon EC2 Container Service 时指定凭证的信息。	2018 年 2 月 16 日

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。