



用户指南

# AWS Toolkit for VS Code



# AWS Toolkit for VS Code: 用户指南

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商标和商业外观不得用于任何非 Amazon 的商品或服务，也不得以任何可能引起客户混淆、贬低或诋毁 Amazon 的方式使用。所有非 Amazon 拥有的其他商标均为各自所有者的财产，这些所有者可能附属于 Amazon、与 Amazon 有关联或由 Amazon 赞助，也可能不是如此。

# Table of Contents

AWS Toolkit for Visual Studio Code .....	1
什么是 AWS Toolkit for Visual Studio Code .....	1
相关信息 .....	1
亚马逊 Q 开发者和亚马逊 CodeWhisperer .....	2
下载 Toolkit .....	3
从 VS Code Marketplace 下载 Toolkit for VS Code .....	3
其他来自 AWS 的 IDE 工具包 .....	3
开始使用 .....	4
安装 Toolkit for VS Code .....	4
先决条件 .....	4
下载并安装 AWS Toolkit for Visual Studio Code .....	4
可选的先决条件 .....	5
正在连接到 AWS .....	6
注册获取 AWS 账户 .....	6
先决条件 .....	6
打开登录面板 .....	6
正在连接到 AWS 来自工具包 .....	7
Amazon 身份验证 CodeCatalyst .....	8
更改 AWS 区域 .....	8
向 AWS 各区服务浏览器添加区域 .....	9
从 AWS Explorer 中隐藏区域 .....	9
配置工具链 .....	9
为 .NET Core 配置工具链 .....	9
为以下各项配置工具链 Node.js .....	10
为 Python 配置工具链 .....	10
为 Java 配置工具链 .....	10
为 Go 配置工具链 .....	11
使用工具链 .....	11
身份验证和访问 .....	12
IAM Identity Center .....	12
IAM 凭证 .....	12
创建 IAM 用户 .....	13
从中创建共享凭证文件 AWS Toolkit for Visual Studio Code .....	13
添加额外的凭证配置文件 .....	14

AWS 构建者 ID .....	15
使用外部凭证流程 .....	15
更新防火墙和网关 .....	15
AWS Toolkit for Visual Studio Code 端点 .....	15
Amazon Q 插件端点 .....	16
Amazon Q 开发者版端点 .....	17
Amazon Q 代码转换端点 .....	17
身份验证端点 .....	17
身份端点 .....	17
遥测 .....	18
引用 .....	18
与 AWS .....	20
实验性功能 .....	21
AWS Explorer .....	21
AWS 文档 .....	22
入门 AWS 文档 .....	23
在 VS Code 中查看文档、自动完成和验证 .....	23
Amazon CodeCatalyst .....	24
什么是亚马逊 CodeCatalyst ? .....	24
CodeCatalyst 入门 .....	24
使用 CodeCatalyst 资源 .....	25
处理开发环境 .....	28
问题排查 .....	30
Amazon API Gateway .....	31
AWS App Runner .....	31
先决条件 .....	32
定价 .....	34
创建 App Runner 服务 .....	35
管理 App Runner 服务 .....	37
AWS Application Builder .....	40
使用 Ap AWS p 开发器 .....	40
AWS 基础架构编辑器 .....	43
使用 AWS 基础架构编辑器 .....	43
AWS CDK .....	45
AWS CDK 应用程序 .....	45
CloudFormation 堆栈 .....	47

删除 CloudFormation 堆栈 .....	47
创建 CloudFormation 模板 .....	48
Amazon CloudWatch Logs .....	49
查看 CloudWatch 日志组和日志流 .....	50
处理 CloudWatch 日志事件 .....	51
搜索日志组 .....	52
CloudWatch Logs Live Tail .....	54
Amazon DocumentDB .....	56
使用 Amazon DocumentDB .....	56
Amazon EC2 .....	61
与 Amazon 合作 EC2 .....	61
Amazon EC2 问题排查 .....	69
Amazon ECR .....	71
使用 Amazon ECR .....	71
创建 App Runner 服务 .....	81
Amazon ECS .....	82
IntelliSense 用于任务定义文件 .....	82
Amazon ECS Exec .....	83
Amazon EventBridge .....	85
使用 Amazon EventBridge 架构 .....	85
AWS IAM 访问分析器 .....	88
使用 AWS IAM 访问分析器 .....	88
AWS IoT .....	92
AWS IoT 先决条件 .....	92
AWS IoT 事物 .....	92
AWS IoT 证书 .....	94
AWS IoT 策略 .....	96
AWS Lambda 函数 .....	99
使用 AWS Lambda 函数 .....	99
AWS Lambda console 到 IDE .....	105
AWS Lambda LocalStack 支持 .....	106
Lambda 远程调试 .....	111
Amazon Redshift .....	120
使用 Amazon Redshift .....	120
Amazon S3 .....	124
使用 Amazon S3 资源 .....	125

使用 Amazon S3 对象 .....	126
Amazon SageMaker 融通式合作开发工作室 .....	129
AWS Serverless Application .....	130
开始使用 .....	130
使用 Serverless Land .....	137
直接通过代码运行和调试 Lambda 函数 .....	139
运行和调试本地 Amazon API Gateway 资源 .....	143
调试无服务器应用程序的配置选项 .....	146
问题排查 .....	152
AWS Systems Manager .....	154
假设和先决条件 .....	154
Systems Manager 自动化文档的 IAM 权限 .....	155
创建新的 Systems Manager 自动化文档 .....	155
打开现有的 Systems Manager 自动化文档 .....	156
编辑 Systems Manager 自动化文档 .....	157
发布 Systems Manager 自动化文档 .....	157
删除 Systems Manager 自动化文档 .....	158
删除 Systems Manager 自动化文档 .....	158
问题排查 .....	159
AWS Step Functions .....	159
使用 Step Functions .....	160
使用 Workflow Studio .....	163
Threat Composer .....	167
使用 Threat Composer .....	167
资源 .....	168
用于访问资源的 IAM 权限 .....	169
添加现有资源并与之交互 .....	169
创建和编辑资源 .....	171
问题排查 .....	173
问题排查最佳实践 .....	173
配置文件 ... 在配置文件中找不到 .....	174
SAM json 架构：无法更改 template.yaml 文件中的架构 .....	175
安全性 .....	176
数据保护 .....	176
文档历史记录 .....	178
.....	clxxxiv

# AWS Toolkit for Visual Studio Code

这是 AWS Toolkit for VS Code 的用户指南。如果您正在查找 AWS Toolkit for Visual Studio，请参阅 [AWS Toolkit for Visual Studio 的用户指南](#)。

## 什么是 AWS Toolkit for Visual Studio Code

Toolkit for VS Code 是 Visual Studio Code ( VS Code ) 编辑器的开源扩展。此扩展使开发人员能够更轻松地在本地开发、调试使用 Amazon Web Services ( AWS ) 的无服务器应用程序并对其进行部署。

### 主题

- [AWS Toolkit for Visual Studio Code 入门](#)
- [使用 AWS 服务和工具](#)

## 相关信息

使用以下资源访问工具包的源代码或查看当前公开的问题。

- [源代码](#)
- [问题跟踪](#)

要了解有关 Visual Studio Code 编辑器的更多信息，请访问 <https://code.visualstudio.com/>。

# 亚马逊 Q 开发者和亚马逊 CodeWhisperer

截至 2024 年 4 月 30 日，亚马逊现已将 CodeWhisperer 作为 Amazon Q Developer 的一部分，其中包括内联代码建议和 Amazon Q 开发者安全扫描。[从 VS Code Marketplace 下载 Amazon Q 开发者版 IDE 扩展程序](#)即可开始使用。

有关 Amazon Q 开发者版服务的详细信息，请参阅《[Amazon Q 开发者版](#)用户指南》。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 VS Code

您可以在 IDE 中通过 VS Code Marketplace 下载、安装和设置 AWS Toolkit for Visual Studio Code。有关详细说明，请参阅本用户指南的“入门”主题中的 [下载并安装](#) 部分。

## 从 VS Code Marketplace 下载 Toolkit for VS Code

或者，您也可以通过从 Web 浏览器导航至 [VS Code Marketplace](#) 来下载 AWS Toolkit for Visual Studio Code 安装文件。

## 其他来自 AWS 的 IDE 工具包

除 AWS Toolkit for Visual Studio Code 之外，AWS 还提供适用于 JetBrains 和 Visual Studio 的 IDE 工具包。

### AWS Toolkit for JetBrains 链接

- 点击此链接以从 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。

### Toolkit for Visual Studio 链接

- 点击此链接以从 Visual Studio Marketplace [下载 Toolkit for Visual Studio](#)。
- 要了解有关 Toolkit for Visual Studio 的更多信息，请参阅 [Toolkit for Visual Studio](#) 用户指南。

# AWS Toolkit for Visual Studio Code 入门

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

为了帮助您入门，以下主题将介绍如何设置、安装和配置 AWS Toolkit for Visual Studio Code。

## 主题

- [安装 AWS Toolkit for Visual Studio Code](#)
- [正在连接到 AWS](#)
- [更改 AWS 区域](#)
- [配置工具链](#)

## 安装 AWS Toolkit for Visual Studio Code

### 先决条件

要从 VS Code 开始使用 AWS Toolkit for Visual Studio Code，必须满足以下先决条件。要详细了解如何从 AWS Toolkit for Visual Studio Code 访问所有可用的 AWS 服务和资源，请参阅本指南的[the section called “可选的先决条件”](#)部分。

- VS Code 要求采用 Windows、macOS 或 Linux 操作系统。
- AWS Toolkit for Visual Studio Code 要求采用 VS Code 版本 1.73.0 或更高版本。

如需详细了解 VS Code 或如何下载最新版本的 VS Code，请访问 [VS Code 下载](#)网站。

### 下载并安装 AWS Toolkit for Visual Studio Code

您可以在 IDE 中通过 VS Code Marketplace 下载、安装和设置 AWS Toolkit for Visual Studio Code。或者，您也可以通过从 Web 浏览器导航至 [VS Code Marketplace](#) 来下载 AWS Toolkit for Visual Studio Code 安装文件。

从 VS Code IDE Marketplace 安装 AWS Toolkit for Visual Studio Code

1. 使用以下链接在 VS Code IDE 中打开 AWS Toolkit for Visual Studio Code 扩展：[打开 VS Code Marketplace](#)。

**Note**

如果 VS Code 尚未在您的计算机上运行，则此操作可能需要一些时间等待 VS Code 加载。

2. 从 VS Code Marketplace 中的 AWS Toolkit for Visual Studio Code 扩展中，选择安装以开始安装流程。
3. 出现提示时，选择重新启动 VS Code 以完成安装流程。

## 可选的先决条件

您必须具备以下条件，才能使用 AWS Toolkit for Visual Studio Code 的某些功能：

- Amazon Web Services ( AWS ) 账户：AWS 账户并非使用 AWS Toolkit for Visual Studio Code 的必要条件，但如果没有账户，功能将会受到严重限制。要获取 AWS 帐户，请转至 [AWS 主页](#)。选择创建 AWS 账户或完成注册（如果您之前访问过该网站）。
  - 代码开发：您要使用的语言的相关开发工具包。您可以从以下链接下载，或使用您最喜爱的软件包管理器：
    - .NET 开发工具包：<https://dotnet.microsoft.com/download>
    - Node.js 开发工具包：<https://nodejs.org/en/download>
    - Python SDK：<https://www.python.org/downloads>
    - Java 开发工具包：<https://aws.amazon.com/corretto/>
    - Go SDK：<https://golang.org/doc/install>
  - AWS SAM CLI：这是一个 AWS CLI 工具，可帮助您在本地开发、测试和分析无服务器应用程序。安装工具包时不需要此工具。但是，我们建议您安装它（以及下面介绍的 Docker），因为任何 AWS Serverless Application Model ( AWS SAM ) 功能都需要它，例如[创建新的无服务器应用程序（本地）](#)。
- 有关更多信息，请参阅 [AWS Serverless Application Model 开发人员指南](#) 中的 [安装 AWS SAM CLI](#)。
- Docker：AWS SAM CLI 需要此开源软件容器平台。有关更多信息和下载说明，请参阅 [Docker](#)。
  - 程序包管理器：借助程序包管理器，您可以下载和共享应用程序代码。
    - .NET：[NuGet](#)
    - Node.js：[npm](#)
    - Python：[pip](#)

- Java : [Gradle](#) 或 [Maven](#)

## 正在连接到 AWS

大多数 Amazon Web Services (AWS) 资源都是通过 AWS 账户管理的。无需 AWS 账户即可使用 AWS Toolkit for Visual Studio Code，但是，如果没有连接，Toolkit 的功能将受到限制。

如果您之前设置过 AWS 账户并通过其他 AWS 服务（例如 AWS Command Line Interface）进行身份验证，则 AWS Toolkit for Visual Studio Code 会自动检测您的凭据。

## 注册获取 AWS 账户

要开始使用 AWS，你需要一个 AWS 账户。有关创建的信息 AWS 账户，请参阅《AWS 账户管理 参考指南》AWS 账户中的[入门](#)指南。

## 先决条件

如果您是新用户 AWS 或尚未创建账户，则需要通过两个主要步骤将其 AWS Toolkit for Visual Studio Code 与您的 AWS 账户进行关联：

1. 设置身份验证：有 3 种主要方法可以从中使用您的 AWS 账户进行身份验证 AWS Toolkit for Visual Studio Code。要了解上述每种方法的更多信息，请参阅本《用户指南》中的[身份验证和访问权限](#)主题。
2. 使用@@ 工具包进行 AWS 身份验证：完成本用户指南以下各节中的步骤，即可通过工具包与您的 AWS 账户建立联系。

## 打开登录面板

请完成以下过程之一以打开 AWS Toolkit 登录面板。

要打开 AWS 工具包登录 来自的面板 AWS 探险家：

1. 从中 AWS Toolkit for Visual Studio Code，展开 EXPLORER。
2. 选择 ... 图标来展开更多操作... 菜单。
3. 在更多操作... 菜单中，选择连接到 AWS 以打开 AWS Toolkit 登录面板。

要打开 AWS 工具包登录 使用 VS Code 命令模块的面板：

1. 按下 **Shift+Command+P** ( 对于 Windows , 按 **Ctrl+Shift+P** ) 打开命令面板。
2. 在搜索字段中输入 **AWS: Add a New Connection**。
3. 选择 **AWS: Add a New Connection** 以打开 AWS Toolkit 登录面板。

## 正在连接到 AWS 来自工具包

### 使用 SSO 进行身份验证和连接

要 AWS 使用进行身份验证和连接 AWS IAM Identity Center , 请完成以下步骤。

#### Note

使用 AWS 构建器 ID 或 IAM Identity Center 进行身份验证会在您的默认 Web 浏览器中启动 AWS 授权门户。每次您的凭证到期时，都必须重复此过程才能续订您的 AWS 账户与之间的连接 AWS Toolkit for Visual Studio Code。

### 进行身份验证并连接 AWS IAM Identity Center

1. 在 AWS Toolkit 登录面板中，选择员工选项卡，然后选择继续按钮继续。
2. 在使用 IAM Identity Center 登录面板中，输入贵公司的起始 URL。此 URL 由贵公司的管理员或帮助台提供给您。
3. 从下拉菜单中选择您 AWS 所在的地区。这是托管您的身份目录的 AWS 区域。
4. 选择继续按钮，并确认您想要在默认的 Web 浏览器中打开 AWS 授权请求网站。
5. 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 VS Code。

### 使用 IAM 凭证进行身份验证和连接

要 AWS 使用 IAM 凭证进行身份验证和连接，请完成以下步骤。

#### 使用 IAM 凭证进行身份验证和连接

1. 在 AWS Toolkit 登录面板中，选择 IAM 凭证，然后选择继续按钮继续。

2. 在提供的字段中输入您 AWS 账户的 **Profile Name**、**Access Key**、和 **Secret Key**，然后选择“继续”按钮将配置文件添加到您的配置文件中，并将 Toolkit 与您的 AWS 帐户关联起来。
3. 完成身份验证并建立连接后，Toolkit AWS Explorer 将会更新以显示您的 AWS 服务和资源。

## Amazon 身份验证 CodeCatalyst

要开始使用 Toolkit，请使用 CodeCatalyst 您的 AWS 构建器 ID 或 IAM Identity Center 凭证进行身份验证并进行连接。

以下过程介绍了如何进行身份验证并将 Toolkit 与您的 AWS 帐户相连接。

### 进行身份验证并连接 AWS Builder ID

1. 在 AWS Toolkit 登录面板中，选择员工选项卡，然后选择继续按钮继续。
2. 在使用 SSO 登录面板顶部，选择跳至登录链接。
3. 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 VS Code。

### 通过 IAM Identity Center 进行身份验证和连接

1. 在 AWS Toolkit 登录面板中，选择员工选项卡，然后选择继续按钮继续。
2. 在使用 IAM Identity Center 登录面板中，输入贵公司的起始 URL。此 URL 由贵公司的管理员或帮助台提供给您。
3. 从下拉菜单中选择您 AWS 所在的地区。这是托管您的身份目录的 AWS 区域。
4. 选择继续按钮，并确认您想要在默认的 Web 浏览器中打开 AWS 授权请求网站。
5. 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 VS Code。

## 更改 AWS 区域

AWS 区域指定 AWS 资源的管理位置。从 AWS Toolkit for Visual Studio Code 连接到 AWS 帐户时，系统会检测到默认 AWS 区域，并自动显示在 AWS Explorer 中。

以下各节将介绍如何从 AWS Explorer 添加或隐藏区域。

## 向 AWS 各区服务浏览器添加区域

完成以下步骤，以向 AWS 各区服务浏览器添加区域。

1. 在 VS Code 中，展开主菜单上的视图，然后选择命令面板以打开命令面板。或者使用以下快捷键：
  - Windows 和 Linux：按下 **Ctrl+Shift+P**。
  - macOS：按下 **Shift+Command+P**。
2. 在命令面板中，搜索 **AWS: Show or Hide Regions** 并选择 **AWS: 显示或隐藏区域**，即可显示可用区域列表。
3. 在列表中，选择要添加到 AWS 各区服务浏览器的 AWS 区域。
4. 选择确定按钮，确认您的选择并更新 AWS 各区服务浏览器。

## 从 AWS Explorer 中隐藏区域

要从 AWS Explorer 视图中隐藏区域，请完成以下流程。

1. 在 AWS Explorer 中，找到要隐藏的 AWS 区域。
2. 打开要隐藏的区域的下文菜单（右键单击）。
3. 选择显示或隐藏区域，在 VS Code 中打开 **AWS: 显示或隐藏区域** 选项。
4. 在“AWS 各区服务浏览器”视图中取消选择要隐藏的区域。

## 配置工具链

AWS Toolkit for Visual Studio Code 支持所有 AWS 服务中的多种语言。以下各节将介绍如何为不同的语言配置工具链。

### 为 .NET Core 配置工具链

1. 确保已[安装](#)适用于 VS Code 的 AWS Toolkit。
2. 安装 [C# 扩展](#)。此扩展使 VS Code 能够调试 .NET Core 应用程序。
3. 打开 AWS Serverless Application Model (AWS SAM) 应用程序，或[创建一个](#)。
4. 打开包含 `template.yaml` 的文件夹。

## 为以下各项配置工具链 Node.js

1. 确保已[安装](#)适用于 VS Code 的 AWS Toolkit。
2. 打开或[创建 AWS SAM](#)应用程序。
3. 打开包含 `template.yaml` 的文件夹。

### Note

直接从源代码调试 TypeScript Lambda 函数时（启动配置有 `"target": "code"`），TypeScript 编译器必须安装在全局或项目中。package.json

## 为 Python 配置工具链

1. 确保已[安装](#)适用于 VS Code 的 AWS Toolkit。
2. 安装[适用于 Visual Studio Code 的 Python 扩展](#)。此扩展使 VS Code 能够调试 Python 应用程序。
3. 打开或[创建 AWS SAM](#)应用程序。
4. 打开包含 `template.yaml` 的文件夹。
5. 在应用程序的根目录下打开一个终端，然后通过运行 `python -m venv ./venv` 配置 `virtualenv`。

### Note

每个系统只需要配置一次 `virtualenv`。

6. 通过运行以下任一项激活 `virtualenv`：
  - Linux 或 macOS (Bash) : `source ./venv/bin/activate`
  - Windows (PowerShell) : `./venv/Scripts/Activate.ps1`
  - Windows ( 命令提示符 ) : `./venv\Scripts\activate.bat`

## 为 Java 配置工具链

1. 确保已[安装](#)适用于 VS Code 的 AWS Toolkit。
2. 安装 [Java 扩展和 Java 11](#)。此扩展使 VS Code 能够识别 Java 函数。

3. 安装 [Java 调试程序扩展](#)。此扩展使 VS Code 能够调试 Java 应用程序。
4. 打开或[创建 AWS SAM](#)应用程序。
5. 打开包含 `template.yaml` 的文件夹。

## 为 Go 配置工具链

1. 确保已[安装](#)适用于 VS Code 的 AWS Toolkit。
2. 调试 Go Lambda 函数需要 Go 1.14 或更高版本。
3. 安装 [Go 扩展](#)。

### Note

调试 Go1.15 + 运行时需要版本 0.25.0 或更高版本。

4. 使用[命令面板](#)安装 Go 工具：
  - a. 从命令面板中选择 Go: Install/Update Tools。
  - b. 从一组复选框中，选择 `dlv` 和 `gopls`。
5. 打开或[创建 AWS SAM](#)应用程序。
6. 打开包含 `template.yaml` 的文件夹。

## 使用工具链

设置好工具链后，就可以用它来[运行或调试](#) AWS SAM 应用程序。

# 的身份验证和访问权限 AWS Toolkit for Visual Studio Code

您无需进行身份验证 AWS 即可开始使用 AWS Toolkit for Visual Studio Code。但是，大多数 AWS 资源都是通过 AWS 账户管理的。要访问所有 AWS Toolkit for Visual Studio Code 服务和功能，您需要使用 AWS 构建器 ID 或 IAM 凭证进行身份验证。AWS IAM Identity Center

以下主题包含每个凭证类型的更多详细信息。

有关如何 AWS Toolkit for Visual Studio Code 使用现有凭据连接 AWS 的详细信息，请参阅本用户指南中的[连接 AWS](#)主题。

## 主题

- [AWS IAM 身份中心](#)
- [AWS IAM 证书](#)
- [面向开发人员的 AWS Builder ID](#)
- [使用外部凭证流程](#)
- [更新防火墙和网关以允许访问](#)

## AWS IAM 身份中心

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设置 IAM 身份中心 (SDKs) 的详细说明，请参阅AWS SDKs 和工具参考指南的[IAM 身份中心身份验证](#)部分。

有关如何进行身份验证以及如何将 AWS 工具包与现有 IAM Identity Center 证书关联的详细信息，请参阅本用户指南中的 [Connect](#) to AWS主题。

## AWS IAM 证书

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

有关如何进行身份验证以及如何将 AWS 工具包与现有 AWS IAM 证书关联的详细信息，请参阅本用户指南中的 [Connect](#) to AWS主题。

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 Code。

### ⚠ 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 AWS Toolkit for Visual Studio Code 会自动检测这些证书并在 VS Code 中提供这些证书。
- AWS 建议使用 IAM 身份中心身份验证。有关 AWS IAM 最佳实践的更多信息，请参阅 Identity and Access Management AWS 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，改为使用与身份提供商的联合身份验证，例如《AWS IAM Identity Center 用户指南》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 Code 进行身份验证之前，您需要在和工具参考指南的[使用长期证书进行身份验证](#)主题中完成步骤 1：创建您的 IAM 用户 AWS SDKs 和步骤 2：获取访问密钥。

### 📘 Note

第 3 步：更新共享凭证文件 AWS SDKs 和《工具参考指南》是可选的。

如果您完成了步骤 3，则 AWS Toolkit for Visual Studio Code 会在以下[the section called “从中创建共享凭证文件 AWS Toolkit for Visual Studio Code”](#)位置自动检测您的凭证。

如果您尚未完成步骤 3，则将 AWS Toolkit for Visual Studio Code 引导您完成创建的过程，credentials file 如下[the section called “从中创建共享凭证文件 AWS Toolkit for Visual Studio Code”](#)所述。

## 从中创建共享凭证文件 AWS Toolkit for Visual Studio Code

您的共享配置文件和共享凭证文件存储您 AWS 账户的配置和凭证信息。有关共享配置和凭证的更多信息，请参阅《AWS Command Line Interface 用户指南》中的[配置设置存储在何处？](#)

通过创建共享凭证文件 AWS Toolkit for Visual Studio Code

1. 按下 **Shift+Command+P**（对于 Windows，按 **Ctrl+Shift+P**）打开命令面板。
2. 在搜索字段中输入 **AWS: Add a New Connection**。

3. 选择 **AWS: Add a New Connection** 以打开 AWS Toolkit 登录面板。
4. 在 AWS Toolkit 登录面板中，选择 IAM 凭证，然后选择继续按钮继续。
5. 在提供的字段中输入您 AWS 账户的 **Profile Name**、**Access Key**、和 **Secret Key**，然后选择“继续”按钮将配置文件添加到您的配置文件中，并将 Toolkit 与您的 AWS 帐户关联起来。
6. 完成身份验证并建立连接后，Toolkit AWS Explorer 将会更新以显示您的 AWS 服务和资源。

### Note

在此示例中，假设 `[Profile_Name]` 其中包含语法错误并导致身份验证失败。

```
[Profile_Name]
xaws_access_key_id= AKIAI44QH8DHBEXAMPLE
xaws_secret_access_key= wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

以下是为响应身份验证尝试失败而生成的日志消息示例。

```
2022-11-02 22:01:54 [ERROR]: Profile [Profile_Name] is not a valid Credential
Profile: not supported by the Toolkit
2022-11-02 22:01:54 [WARN]: Shared Credentials Profile [Profile_Name] is not
valid. It will not be used by the toolkit.
```

## 添加额外的凭证配置文件

您可以向配置文件中添加多个凭证。为此，请打开命令面板并选择 AWS Toolkit 创建凭证配置文件。这将打开凭证文件。在此页面上，您可以在第一个配置文件下方添加一个新的配置文件，如以下示例所示：

```
# Amazon Web Services Credentials File used by AWS CLI, SDKs, and tools
# This file was created by the AWS Toolkit for Visual Studio Code extension.
#
# Your AWS credentials are represented by access keys associated with IAM users.
# For information about how to create and manage AWS access keys for a user, see:
# https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_access-keys.html
#
# This credential file can store multiple access keys by placing each one in a
```

```
# named "profile". For information about how to change the access keys in a
# profile or to add a new profile with a different access key, see:
# https://docs.aws.amazon.com/cli/latest/userguide/cli-config-files.html
#
[Profile1_Name]
# The access key and secret key pair identify your account and grant access to AWS.
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
# Treat your secret key like a password. Never share your secret key with anyone. Do
# not post it in online forums, or store it in a source control system. If your secret
# key is ever disclosed, immediately use IAM to delete the access key and secret key
# and create a new key pair. Then, update this file with the replacement key details.
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
[Profile2_Name]
aws_access_key_id = AKIAI44QH8DHBEXAMPLE
aws_secret_access_key = je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
```

## 面向开发人员的 AWS Builder ID

AWS Builder ID 对某些 AWS 服务来说，是额外的可选或必需 AWS 账户。有关 AWS 构建者 ID 身份验证方法的详细信息，请参阅《AWS 登录用户指南》中的[使用 AWS 构建者 ID 登录](#)主题。

有关如何使用现有 AWS 构建者 ID 进行身份验证并连接 AWS Toolkit 的详细信息，请参阅本用户指南中的[连接到 AWS](#)主题。

## 使用外部凭证流程

您可以通过修改您的，AWS Toolkit for Visual Studio Code 为不直接支持的 AWS 凭证流程进行配置。shared config file

修改您的证书流程shared config file对 AWS Toolkit for Visual Studio Code 和来说都是一样的。AWS Command Line Interface有关如何设置外部凭证的详细信息，请参阅《AWS Command Line Interface 用户指南》中的[使用外部流程获取凭证](#)主题。

## 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 AWS Toolkit for Visual Studio Code Amazon Q 提供的所有服务和功能。

## AWS Toolkit for Visual Studio Code 端点

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

## 端点

```
https://idetoolkits.amazonwebservices.com/endpoints.json
```

## 托管文件

```
https://idetoolkits-hostedfiles.amazonaws.com/Notifications/VSCode/startup/1.x.json  
https://idetoolkits-hostedfiles.amazonaws.com/Notifications/VSCode/emergency/1.x.json
```

## 架构支持

```
https://raw.githubusercontent.com/aws/serverless-application-model/main/samtranslator/  
schema/schema.json  
https://api.github.com/repos/devfile/api/releases/latest  
https://raw.githubusercontent.com/devfile/api/${devfileSchemaVersion}/schemas/latest/  
devfile.json
```

## cSharpSam调试安装脚本

```
https://aka.ms/getvsdbgps1  
https://aka.ms/getvsdbgsh
```

## 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 autocomplete 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身份端点，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.
```

# 使用 AWS 服务和工具

AWS Toolkit for Visual Studio Code 使您可以直接在 VS Code 中使用 AWS 服务、工具和资源。以下是介绍每个 Toolkit for VS Code 服务及其功能的指南主题列表。选择一项服务或工具，详细了解该服务或工具的用途、设置方法及其基本功能的使用方式。

## 主题

- [使用实验性功能](#)
- [在 AWS Explorer 中使用 AWS 服务](#)
- [AWS 文档](#)
- [亚马逊 V CodeCatalyst S Code 版](#)
- [使用 Amazon API Gateway](#)
- [AWS App Runner 与一起使用 AWS Toolkit for Visual Studio Code](#)
- [AWS Application Builder](#)
- [AWS 基础架构编辑器](#)
- [AWS CDK for VS Code](#)
- [使用 AWS CloudFormation 堆栈](#)
- [通过 AWS Toolkit for Visual Studio Code 使用 CloudWatch Logs](#)
- [Amazon DocumentDB](#)
- [Amazon Elastic Compute Cloud](#)
- [使用 Amazon Elastic Container Registry](#)
- [使用 Amazon Elastic Container Service](#)
- [使用 Amazon EventBridge](#)
- [AWS IAM 访问分析器](#)
- [在 AWS IoT 中使用 AWS Toolkit for Visual Studio Code](#)
- [AWS Lambda 函数](#)
- [Toolkit for VS Code 中的 Amazon Redshift](#)
- [使用 Amazon S3](#)
- [适用于 VS Code 的 Amazon SageMaker 融通式合作开发工作室](#)
- [使用无服务器应用程序](#)

- [使用 Systems Manager 自动化文档](#)
- [AWS Step Functions](#)
- [使用 Threat Composer](#)
- [使用资源](#)

## 使用实验性功能

实验性功能允许您在功能正式发布之前，抢先体验 AWS Toolkit for Visual Studio Code 中的功能。

### Warning

由于实验性功能仍有待测试和更新，可能存在可用性问题。实验性功能可能会从 AWS Toolkit for Visual Studio Code 中删除，恕不另行通知。

您可以在 VS Code IDE 中设置窗格的 AWS Toolkit 部分，启用特定 AWS 服务的实验性功能。

1. 要在 VS Code 中编辑 AWS 设置，请依次选择文件、首选项、设置。
2. 在设置窗格中，展开扩展，然后选择 AWS Toolkit。
3. 在 AWS：实验下，选中要在发布之前抢先体验的实验性功能对应的复选框。如果要关闭实验性功能，请清除相关复选框。

## 在 AWS Explorer 中使用 AWS 服务

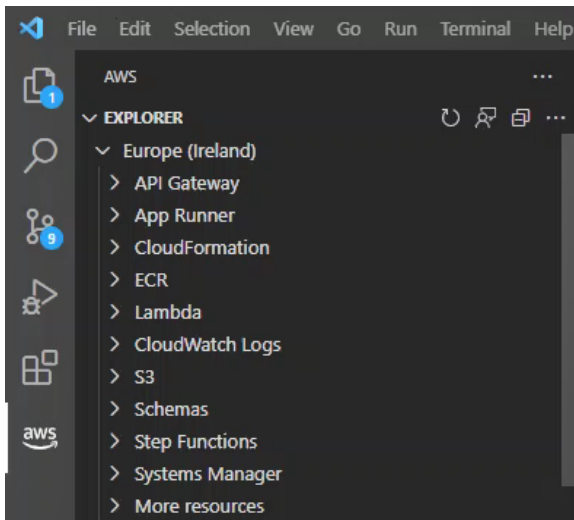
AWS Explorer 为您提供了使用 AWS Toolkit for Visual Studio Code 时可以使用的一些 AWS 服务的视图。

本节提供有关如何在 VS Code 中访问和使用 AWS Explorer 的信息。此过程假定您已在系统上[安装和配置](#)了 Toolkit for VS Code。

一些要点：

- 如果该工具包已正确安装和配置，您应该会在 AWS Explorer 中看到相应项目。要查看 AWS Explorer，请在活动栏中选择 AWS 图标。

例如：



- 某些功能需要特定的 AWS 权限。例如，要查看 AWS 账户中的 AWS Lambda 函数，您在 [身份验证和访问](#) 中配置的凭证必须至少包含只读 Lambda 权限。有关每个功能所需权限的更多信息，请参阅以下主题。
- 如果您想与在 AWS Explorer 中无法立即看到的 AWS 服务进行交互，可以转到更多资源，然后从数百种可以添加到界面的资源中进行选择。

例如，您可以从可用资源类型选择中 `AWS Toolkit:CodeArtifact::Repository`。将此资源类型添加到更多资源后，您可以展开该条目以查看使用自己的属性创建不同 CodeArtifact 存储库的资源列表。此外，您可以在 JSON 格式的模板中描述资源的属性，这些模板可以保存，以便在 AWS Cloud 中创建新资源。

## AWS 文档

AWS Toolkit for Visual Studio Code 支持 for AWS Serverless Application Model JSON Schema AWS SAM templates，通过直接在 VS Code 中启用定义、自动完成和验证来增强模板创作体验。AWS 文档支持所有 CloudFormation 资源 AWS SAM 和资源。有关更多详细信息，请参阅以下资源：

- 有关 JSON 架构的具体信息，请参阅 [JSON 架构](#) JSON-Schema.org 网站。
- 有关 AWS SAM 模板的更多信息，请参阅《AWS Serverless Application Model 开发人员指南》中的 [AWS SAM 模板解剖学](#) 主题。
- 有关 AWS 资源和属性类型的更多信息，请参阅 CloudFormation 用户指南中的 [AWS 资源和属性类型参考](#) 主题。
- 有关 AWS Toolkit 使用的 AWS SAM 架构的详细信息，请参阅 AWS GitHub 存储库中的 [AWS Serverless Application Model](#) 架构。

## 入门 AWS 文档

要开始使用 VS Code 中的 AWS 文档，请从 IDE 或 VS Code Marketplace 中安装 AWS Toolkit for Visual Studio Code 扩展程序，然后打开任何 AWS SAM 模板。

### 在 VS Code 中查看文档、自动完成和验证

查看文档、自动完成和验证是 AWS Toolkit 中包含的功能。有关这些功能在 VS Code 中的具体呈现效果示例，请参见下图。

- 要从打开的 AWS SAM 模板中查看文档，请将指针悬停在文档中的行条目上。
- 要进行自动补全，请开始在 AWS SAM 模板中键入内容以激活一个弹出窗口，其中包含基于您的输入的建议。
- 系统会自动扫描您的 AWS SAM 模板以进行验证，错误会以灯泡图标突出显示，您可以选择该图标以获取其他建议。

有关这些功能在 VS Code 中的具体呈现效果示例，请参见下图。

The screenshot shows a VS Code editor with an AWS SAM template. The code snippet is:

```

Authorizer: myCognitoAuthMultipleUserPools
WithCognitoMultipleUserPoolsAuthorizerAnyMethod:
  Type: Api
  RestApiId: !Ref MyApi
  Path: /any/lambda/token
  Method: any

```

A tooltip is displayed over the `RestApiId` property, providing the following information:

- RestApiId**
- Identifier of a RestApi resource, which must contain an operation with the given path and method. Typically, this is set to reference an `AWS::Serverless::Api` resource defined in this template.
- With Type Property**: If you don't define this property, AWS SAM creates a default `AWS::Serverless::Api` resource using a generated `OpenApi` document. That resource contains a union of all paths and methods defined by `Api` events in the same template that do not specify a `RestApiId`.
- This cannot reference an `AWS::Serverless::Api` resource defined in another template.
- Type**: String
- Required**: No
- AWS CloudFormation compatibility**: This property is unique to AWS SAM and doesn't have an AWS CloudFormation equivalent.
- Source**: `schema.json`

Below the tooltip, the code snippet is partially visible, showing the `RestApiId` property being referenced as `!Ref MyApi`, and the `Path` and `Method` properties.

# 亚马逊 V CodeCatalyst S Code 版

## 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。通过 AWS Toolkit for Visual Studio Code，您可以直接从 VS Code 中查看和管理您的 CodeCatalyst 资源。您也可以使用 AWS 工具包直接在云端工作，启动运行 VS Code 的开发环境虚拟计算环境。有关该 CodeCatalyst 服务的更多信息，请参阅 [Amazon CodeCatalyst](#) 用户指南。

以下主题介绍如何连接 VS Code CodeCatalyst，以及如何使用 VS Code 工具包 CodeCatalyst 中的 VS Code。

### 主题

- [入门 CodeCatalyst 和 VS Code 的 Toolkit for Code](#)
- [在 VS Code 中使用亚马逊 CodeCatalyst 资源](#)
- [在开发环境中使用 Toolkit](#)
- [对亚马逊 CodeCatalyst 和 VS Code 进行故障排除](#)

## 入门 CodeCatalyst 和 VS Code 的 Toolkit for Code

要开始 CodeCatalyst 在 VS Code 中使用，请按照以下步骤操作。

### 主题

- [创建 CodeCatalyst 账户](#)
- [将 AWS 工具包与 CodeCatalyst](#)

## 创建 CodeCatalyst 账户

要 CodeCatalyst 从 VS Code 的 Toolkit for VS Code 进行连接，您必须拥有有效的 AWS 生成器 ID 或 AWS IAM Identity Center 凭证。要了解有关 AWS 构建器 ID、IAM Identity Center 和 CodeCatalyst 证书的更多信息，请参阅 CodeCatalyst 用户指南中的 [设置](#) 方式 CodeCatalyst 部分。

## 将 AWS 工具包与 CodeCatalyst

要将 AWS Toolkit 与您的 CodeCatalyst 账户关联，请参阅本用户指南连接 AWS 主题 CodeCatalyst 中的 [Amazon 身份验证](#) 部分。

## 在 VS Code 中使用亚马逊 CodeCatalyst 资源

以下各节概述了 VS Code Toolkit 提供的亚马逊 CodeCatalyst 资源管理功能。

有关开发环境以及如何访问开发环境的更多信息 CodeCatalyst，请参阅 Amazon CodeCatalyst 用户指南中的[开发环境](#)部分。

以下各节介绍了如何从 VS Code 创建、打开和使用开发环境。

### 主题

- [克隆存储库](#)
- [打开开发环境](#)
- [创建 CodeCatalyst 开发环境](#)
- [从第三方存储库创建开发环境](#)
- [CodeCatalyst VS 代码中的命令](#)

### 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。如果您更喜欢在本地处理项目，则可以将 CodeCatalyst 存储库克隆到本地计算机上，并在下次连接到云端时将其与您的 CodeCatalyst 项目在线同步。

要使用 AWS 工具包将存储库从您的 CodeCatalyst 账户克隆到 VS Code，请完成以下步骤：

#### Note

如果您要通过第三方服务克隆存储库，系统可能会提示您使用该服务的凭证进行身份验证。克隆存储库时，VS Code 会在正在克隆存储库状态窗口中显示进度。克隆存储库后，系统会显示是否要打开克隆的存储库？消息。

1. 在 Toolkit for VS Code 中，展开开发人员工具资源管理器。
2. 展开 CodeCatalyst，选择“克隆存储库”。
3. 在“选择 CodeCatalyst 存储库”对话框中，搜索要克隆的存储库，然后将其选中以打开“选择要克隆的文件夹”对话框。
4. 选择选择存储库位置，以关闭提示并开始克隆存储库。

5. 在对话框窗口中，选择以下任一方式以完成克隆流程：
  - 要在当前的 VS Code 窗口中打开存储库，请选择打开。
  - 要在新的 VS Code 窗口中打开存储库，请选择在新窗口中打开。
  - 要在不打开存储库的情况下完成克隆流程，请关闭对话框窗口。

## 打开开发环境

要在 VS Code 中打开现有开发环境，请完成以下步骤。

### Note

选择开发环境后，打开开发环境，即可开始连接 VS Code 的过程。CodeCatalyst在此过程中，VS Code 会在 CodeCatalyst 状态窗口中显示进度更新。当流程完成后，状态窗口即会更新。

- 如果开发环境无法打开，状态将会更新，并提供有关流程失败原因的信息以及一个用于打开流程日志的链接。
- 如果该流程成功完成，开发环境将会从 VS Code 打开一个新的窗口。

1. 在 Toolkit for VS Code 中，展开开发人员工具资源管理器。
2. 展开CodeCatalyst并选择“打开开发环境”，在 VS Code 中打开“选择 CodeCatalyst开发环境”对话框。
3. 从“选择 CodeCatalyst 开发环境”对话框中，选择要打开的开发环境。

## 创建 CodeCatalyst 开发环境

要创建新的开发环境，请完成以下步骤。

### Note

在创建新的开发环境时，请注意以下事项：

- AWS 建议您指定别名，因为它可以简化组织并提高开发环境的搜索功能。
- 开发环境可以持续保存您的工作。这意味着您可以停止开发环境，而不会丢失您的工作。停止开发环境可以降低确保开发环境正常运行所需的费用。

- 存储是创建开发环境后唯一无法更改的设置。
- VS Code 会在状态窗口中显示开发环境的创建进度。创建开发环境后，VS Code 会在新窗口中打开该开发环境，并且还会显示您信任此文件夹中文件的作者吗？提示。同意条款和条件以继续在您的开发环境中工作。

1. 在 Toolkit for VS Code 中，展开开发人员工具资源管理器。
2. 展开并选择“创建开发环境”选项 CodeCatalyst，在 VS Code 中打开“创建 CodeCatalyst 开发环境”菜单。
3. 在源代码部分中，选择以下选项之一：
  - 使用现有 CodeCatalyst 存储库：使用现有 CodeCatalyst 存储库创建开发环境。必须选择“CodeCatalyst 项目”和“分支”。
  - 创建空的开发环境：创建一个空的开发环境。
4. (可选) 在别名部分，输入开发环境的备用名称。
5. (可选) 在开发环境配置部分中，更改以下设置以满足您的特定需求。
  - 计算：选择编辑计算以更改分配给系统的处理能力和 RAM 量。
  - 超时：选择编辑超时以更改开发环境停止之前允许的系统空闲时间。
  - 存储：选择编辑存储大小以更改分配给系统的存储空间量。
6. 要创建新的云开发环境，请选择创建开发环境。

## 从第三方存储库创建开发环境

您可以通过链接到作为源的存储库来创建开发环境。

链接到第三方存储库作为源代码是在中的项目级别处理的 CodeCatalyst。有关如何将第三方存储库连接到您的开发环境的说明和其他详细信息，请参阅 Amazon CodeCatalyst 用户指南中的[链接源存储库](#)主题。

## CodeCatalyst VS 代码中的命令

还有一些分配给 CodeCatalyst 相关功能的其他 VS Code 命令不会直接显示在 AWS 工具包中。

要查看 CodeCatalyst 从命令面板分配给命令列表，请完成以下步骤：

1. 在 Toolkit for VS Code 中，展开开发人员工具资源管理器。

2. 选择“显示 CodeCatalyst 命令”以打开命令面板，其中包含预先填充的搜索内容。CodeCatalyst
3. 从列表中选择 一个 CodeCatalyst 命令将其激活。

## 在开发环境中使用 Toolkit

开发环境是 Amazon 的虚拟计算环境 CodeCatalyst。以下各节介绍了如何通过 AWS Toolkit for Visual Studio Code 创建、启动和使用开发环境。

有关开发环境的详细信息，请参阅 Amazon CodeCatalyst 用户指南中的[开发环境](#)主题。

### 使用 devfile 配置您的开发环境

devfile 规范是 YAML 的开放标准格式，可用于定义开发环境的配置。每个开发环境都有一个 devfile。如果您创建的开发环境没有存储库，或者存储库中未包含 devfile，则系统会自动将默认设置应用于源。可以从 CodeCatalyst 或您的 IDE 更新开发文件。在 VS Code 的本地或远程实例中更新 devfile 的流程完全相同，但是如果您在本地更新 devfile，这些更新只有在推送到源存储库之后才会生效。

有关使用开发文件配置开发环境的详细信息，请参阅 Amazon CodeCatalyst 用户指南中的[配置开发环境](#)主题。

以下过程介绍当 devfile 在开发环境中运行时，如何通过 Toolkit 的远程实例对其进行编辑。

#### Important

如果您通过 VS Code 编辑 Devfile，请注意以下事项：

- 更改 devfile 的名称或 devfile 组件名称会替换根目录的内容。之前的所有内容均已丢失且不可恢复。
- 如果您创建的开发环境根文件夹中没有 devfile，或者创建的开发环境未与源存储库相关联，则在创建开发环境时，系统会为其生成一个具有默认配置设置的 devfile。
- 有关如何定义和配置 Devfile 的说明，请参阅 [devfile.io](https://devfile.io) 网站上的[添加命令](#)文档。

1. 在 Toolkit for VS Code 中，展开开发人员工具资源管理器。
2. 展开 CodeCatalyst 并选择“打开 Devfile”，devfile.yaml 在当前开发环境中的新编辑器窗口中打开。
3. 在 VS Code 编辑器中，更新您的 devfile，然后保存您所做的更改。

4. 下次启动开发环境时，配置会更新以匹配您在 Devfile 中定义的规范。

## 通过您的开发环境进行身份验证和连接 AWS

要从开发环境访问所有 AWS 资源，您必须进行身份验证并将工具包的远程实例与您的 AWS 账户连接起来。当开发环境启动时，Toolkit 的远程实例会自动使用从您的 Toolkit 本地实例继承的凭证进行身份验证。

更新 Toolkit 远程实例凭证的过程与您的 Toolkit 本地实例中的身份验证体验完全相同。有关如何通过 Toolkit 更新凭证、进行身份验证和连接到 AWS 的详细说明，请参阅本用户指南的“开始使用”主题中的[连接到 AWS](#)部分。

有关与兼容的每种 AWS 身份验证方法的更多信息 AWS Toolkit for Visual Studio Code，请参阅本用户指南中的[身份验证和访问](#)主题。

## 在开发环境中使用 Toolkit for VS Code

在 VS Code 中打开或创建开发环境后，您可以使用 Toolkit for VS Code 进行工作，这类似于在 VS Code 的本地实例中的工作方式。运行 VS Code 的开发环境配置为自动安装 AWS 工具包并使用您的 AWS 生成器 ID 进行连接。

### 停止开发环境

要停止当前的开发环境，请执行以下操作：

1. 在 Toolkit for VS Code 中，展开开发人员工具资源管理器。
2. 展开 CodeCatalyst 并选择“停止开发环境”。
3. 当 VS Code 显示提示时，请确认您是否要停止开发环境。
4. 当 VS Code 关闭远程连接并返回到本地开发实例时，您的开发环境已成功停止。

### 打开开发环境设置

要打开当前开发环境的设置，请完成以下步骤：

#### Note

开发环境在创建之后就无法更改分配到其中的存储空间量。

1. 在 Toolkit for VS Code 中，展开开发人员工具资源管理器。
2. 展开 CodeCatalyst 并选择“打开设置”，打开当前开发环境的“开发环境设置”视图。
3. 在 Dev Environment Settings (开发环境设置) 视图中，以下部分包含开发环境的选项：
  - Alias (别名)：查看和更改分配给您的开发环境的 Alias (别名)。
  - 状态：查看您当前的开发环境状态、已分配给开发环境的项目以及停止开发环境。
  - Devfile：查看开发环境的 Devfile 的名称和位置。选择在编辑器中打开按钮可打开 Devfile。
  - 计算设置：更改开发环境的大小和默认超时长度。

## 对亚马逊 CodeCatalyst 和 VS Code 进行故障排除

以下主题解决了在使用 Amazon CodeCatalyst 和 VS Code 时可能遇到的技术问题。

### 主题

- [VS Code 版本](#)
- [Amazon 的权限 CodeCatalyst](#)
- [从 Toolkit for VS Code 连接到开发环境](#)

## VS Code 版本

你的 VS Code 版本应该会在你的系统 `vscode://` URIs 上设置一个处理程序。如果没有这个处理程序，你就无法访问 AWS 工具包中的所有 CodeCatalyst 功能。例如，从 VS Code Insiders 启动开发环境时遇到错误。这是因为 VS Code Insiders `vscode-insiders://` URIs 可以处理但不处理 `vscode://` URIs。

## Amazon 的权限 CodeCatalyst

以下是使用 CodeCatalyst 中的文件权限要求 AWS Toolkit for Visual Studio Code：

- 将您自己针对 `~/.ssh/config` 文件的访问权限设置为 `read` 和 `write`。限制所有其他用户的 `write` 权限。
- 将您针对 `~/.ssh/id_dsa` 和 `~/.ssh/id_rsa` 文件的访问权限设置为仅限 `read`。限制所有其他用户的 `read`、`write` 和 `execute` 权限。
- 您的 `globals.context.globalStorageUri.fsPath` 文件必须位于可写的位置。

## 从 Toolkit for VS Code 连接到开发环境

如果您在尝试通过 AWS Toolkit for Visual Studio Code 连接到开发环境时收到以下错误：

您的 `~/.ssh/config` 的 `aws-devenv-*` 部分可能已过时。

- 选择 **打开配置...** 按钮，以在 VS Code 编辑器中打开您的 `~/.ssh/config` 文件。
- 在编辑器中，选择并删除 `Host aws-devenv-*` 部分的内容。
- 保存您对 `~/.ssh/config` 的 `Host aws-devenv-*` 所做的更改。然后，关闭该文件。
- 重新尝试从 Toolkit for VS Code 连接到开发环境。

## 使用 Amazon API Gateway

您可以使用 AWS Toolkit for Visual Studio Code 在连接的 AWS 账户中浏览和运行远程 API Gateway 资源。

### Note

此特征不支持调试。

要浏览和运行远程 API Gateway 资源，请执行以下操作

1. 在 AWS Explorer 中，选择 API Gateway 以展开菜单。此时页面上会列出远程 API Gateway 资源。
2. 找到您想要调用的 API Gateway 资源，打开其上下文菜单（右键单击），然后选择在 AWS 上调用。
3. 在参数表单中，指定调用参数。
4. 要运行远程 API Gateway 资源，请选择调用。结果将显示在 VS Code 输出视图中。

## AWS App Runner 与一起使用 AWS Toolkit for Visual Studio Code

[AWS App Runner](#) 提供了一种快速、简单且经济实惠的方式，可将源代码或容器映像直接部署到 AWS 云中可扩展且安全的 Web 应用程序。使用它，您无需学习新技术、决定使用哪种计算服务，也不需要知道如何配置和配置 AWS 资源。

您可以使用 AWS App Runner 基于源图像或源代码创建和管理服务。如果您使用源镜像，则可以选择存储在镜像存储库中的公有或私有容器镜像。App Runner 支持以下镜像存储库提供商：

- 亚马逊 Elastic Container Registry (Amazon ECR)：在您的账户中存储私有图片。AWS
- Amazon Elastic Container Registry Public (Amazon ECR Public)：存储公开可读的镜像。

如果选择源代码选项，则可以从受支持的存储库提供商维护的源代码存储库进行部署。目前，App Runner 支持[GitHub](#)作为源代码存储库提供者。

## 先决条件

要使用与 App Runner 进行交互，AWS Toolkit for Visual Studio Code 需要满足以下条件：

- 一个 AWS 账户
- 该功能 AWS Toolkit for Visual Studio Code 的一个版本 AWS App Runner

除了这些核心要求之外，请确保所有相关的 IAM 用户都有权与 App Runner 服务进行交互。此外，您还需要获取有关服务源的特定信息，例如容器镜像 URI 或与 GitHub 存储库的连接。创建 App Runner 服务时，您需要使用此信息。

### 为 App Runner 配置 IAM 权限

授予 App Runner 所需权限的最简单方法是将现有 AWS 托管策略附加到相关 AWS Identity and Access Management (IAM) 实体，特别是用户或群组。App Runner 提供两种可附加到 IAM 用户的托管策略：

- `AWSAppRunnerFullAccess`：允许用户执行所有 App Runner 操作。
- `AWSAppRunnerReadOnlyAccess`：允许用户列出和查看有关 App Runner 资源的详细信息。

此外，如果您从 Amazon Elastic Container Registry (Amazon ECR) 中选择私有存储库作为服务源，则必须为 App Runner 服务创建以下访问角色：

- `AWSAppRunnerServicePolicyForECRAccess`：允许 App Runner 访问您账户中的 Amazon Elastic Container Registry (Amazon ECR) 镜像。

使用 VS Code 的命令面板配置服务实例时，您可以自动创建此角色。

**Note**

AWSServiceRoleForAppRunner服务相关角色 AWS App Runner 允许完成以下任务：

- 将日志推送到 Amazon CloudWatch 日志组。
- 创建亚马逊 CloudWatch 活动规则以订阅亚马逊弹性容器注册表 (Amazon ECR) Container Registry 图片推送。

无需手动创建 服务关联角色。当您使用调用的 API 操作 AWS App Runner 在 AWS 管理控制台 或中创建时 AWS Toolkit for Visual Studio Code ， AWS App Runner 会为您创建此服务相关角色。

有关更多信息，请参阅 AWS App Runner 开发人员指南中的[适用于 App Runner 的 Identity and Access Management](#)。

获取 App Runner 的服务源

您可以使用 AWS App Runner 从源图像或源代码部署服务。

Source image

如果您从源映像进行部署，则可以从私有或公共映像注册表中获取该映 AWS 像存储库的链接。

- [Amazon ECR 私有注册表：复制在存储库中使用 Amazon ECR 控制台的私有存储库的 URI。https://console.aws.amazon.com/ecr/](#)
- Amazon ECR 公共注册表：复制使用 Amazon ECR Public Gallery 的公有存储库的 URI：<https://gallery.ecr.aws/>。

**Note**

您还可以直接从 Toolkit for VS Code 中的 AWS Explorer 获取私有 Amazon ECR 存储库的 URI：

- 打开AWS 资源管理器并展开 ECR 节点以查看该 AWS 区域的存储库列表。
- 右键单击存储库，然后选择 Copy Repository URI ( 复制存储库 URI ) 以将链接复制到剪贴板。

使用 VS Code 的命令面板配置服务实例时，您可以指定映像存储库的 URI。

有关更多信息，请参阅 AWS App Runner 开发人员指南中的[基于源镜像的 App Runner 服务](#)。

## Source code

要将源代码部署到 AWS App Runner 服务，该代码必须存储在由支持的存储库提供商维护的 Git 存储库中。App Runner 支持一个源代码存储库提供商：[GitHub](#)。

有关设置 GitHub 存储库的信息，请参阅上的“[入门](#)”文档 [GitHub](#)。

要将您的源代码从 GitHub 存储库部署到 App Runner 服务，App Runner 需要与建立连接 GitHub。如果您的存储库是私有的（也就是说，它不能在上公开访问 GitHub），则必须向 App Runner 提供连接详细信息。

### Important

要创建 GitHub 连接，必须使用 App Runner 控制台 (<https://console.aws.amazon.com/apprunner>) 创建链接到的连接。GitHub AWS 使用 VS Code 的命令面板配置服务实例时，您可以选择 GitHub 连接页面上可用的连接。

有关更多信息，请参阅 AWS App Runner 开发人员指南中的[管理 App Runner 连接](#)。

App Runner 服务实例提供托管运行时，允许您的代码生成和运行。AWS App Runner 目前支持以下运行时：

- Python 托管运行时
- Node.js 托管运行时

作为服务配置的一部分，您可以提供有关 App Runner 服务如何构建和启动服务的信息。您可以使用命令调色板输入此信息，或指定 YAML 格式的 [App Runner 配置文件](#)。此文件中的值指示 App Runner 如何构建和启动服务以及提供运行时上下文。这包括相关的网络设置和环境变量。配置文件名为 `apprunner.yaml`。它会自动添加到应用程序存储库的根目录中。

## 定价

您需要为应用程序使用的计算和内存资源付费。此外，如果选择自动执行部署，则还需要为每个应用程序支付一笔固定的月费，其中涵盖该月的所有自动化部署。如果您选择从源代码进行部署，则还需要为 App Runner 从源代码构建容器所需的时间支付构建费用。

有关更多信息，请参阅[AWS App Runner 定价](#)。

## 主题

- [创建 App Runner 服务](#)
- [管理 App Runner 服务](#)

## 创建 App Runner 服务

您可以使用 AWS Explorer 和 VS Code 的命令面板，在 Toolkit for VS Code 中创建 App Runner 服务。选择在特定 AWS 区域创建服务后，命令面板提供的带编号的步骤将引导您完成配置应用程序运行的服务实例的过程。

在创建 App Runner 服务之前，请确保您已完成[先决条件](#)。这包括提供相关的 IAM 权限以及确认要部署的特定源存储库。

### 创建 App Runner 服务

1. 如果 AWS 资源管理器尚未打开，请将其打开。
2. 右键单击 App Runner 节点，然后选择 Create Service ( 创建服务 )。

此时将显示命令面板。

3. 对于 Select a source code location type ( 选择源代码位置类型 )，请选择 ECR 或 Repository ( 存储库 )。

如果选择 ECR，则可以在 Amazon Elastic Container Registry 维护的存储库中指定容器镜像。如果选择 Repository ( 存储库 )，则可以指定由受支持的存储库提供商维护的源代码存储库。目前，App Runner 支持[GitHub](#)作为源代码存储库提供者。

### 从 ECR 部署

1. 对于 Select or enter an image repository ( 选择或输入镜像存储库 )，请选择或输入由 Amazon ECR 私有注册表或 Amazon ECR Public Gallery 维护的镜像存储库的 URL。

#### Note

如果您从 Amazon ECR Public Gallery 中指定存储库，请确保关闭自动部署，因为 App Runner 不支持对 Amazon ECR Public Gallery 中的镜像进行进行自动部署。

默认情况下，自动部署处于关闭状态，当命令面板标题上的图标显示一条对角线时，会显示此状态。如果选择开启自动部署，则会显示一条消息，通知您此选项可能会产生额外费用。

2. 如果命令面板中的步骤报告未找到标签，则您需要后退一步才能选择包含已设置标签的容器镜像的存储库。
3. 如果您使用的是 Amazon ECR 私有注册表，则需要 ECR 访问角色角色，该角色允许 App Runner 访问您账户 AppRunnerECRAccess 中的亚马逊弹性容器注册表 (Amazon ECR) Container Registry 镜像。选择命令面板标题上的“+”图标以自动创建此角色。（如果您的镜像存储在 Amazon ECR Public 中，其中镜像是公开可用的，则不需要访问角色。）
4. 对于 Port（端口），输入服务使用的 IP 端口（例如端口 8000）。
5. 对于 Configure environment variables（配置环境变量），您可以指定一个文件，其中包含用于自定义服务实例中的行为的环境变量。您也可以跳过此步骤。
6. 对于 Name your service（为服务命名），请输入一个不含空格的唯一名称，然后按 Enter。
7. 对于 Select instance configuration（选择实例配置），请为您的服务实例选择 CPU 单位和内存的组合（以 GB 为单位）。

创建服务时，其状态将从正在创建更改为正在运行。

8. 服务开始运行后，右键单击它并选择 Copy Service URL（复制服务 URL）。
9. 要访问已部署的应用程序，请将复制的 URL 粘贴到 Web 浏览器的地址栏中。

## 从远程存储库进行部署

1. 在“选择连接”中，选择链接 GitHub 到的连接 AWS。可供选择的连接列在 App Runner 控制台的 GitHub 连接页面上。
2. 在“选择远程 GitHub 存储库”中，选择或输入远程存储库的 URL。

已配置 Visual Studio Code 的源控制管理（SCM）的远程存储库可供选择。如果没有列出存储库，也可以粘贴指向存储库的链接。

3. 对于 Select a branch（选择分支），请选择要将源代码部署到哪个 Git 分支。
4. 对于 Choose configuration source（选择配置源），请指定希望如何定义运行时配置。

如果选择 Use configuration file（使用配置文件），则您的服务实例将通过 `apprunner.yaml` 配置文件定义的设置进行配置。此文件位于应用程序存储库的根目录中。

如果选择在此配置所有设置，请使用命令面板指定以下项：

- Runtime (运行时) : 选择 Python 3 或 Nodejs 12。
  - Build command (构建命令) : 输入命令以在服务实例的运行时环境中构建应用程序。
  - Start command (启动命令) : 输入命令以在服务实例的运行时环境中启动应用程序。
5. 对于 Port (端口), 输入服务使用的 IP 端口 (例如端口 8000)。
  6. 对于 Configure environment variables (配置环境变量), 您可以指定一个文件, 其中包含用于自定义服务实例中的行为的环境变量。您也可以跳过此步骤。
  7. 对于 Name your service (为服务命名), 请输入一个不含空格的唯一名称, 然后按 Enter。
  8. 对于 Select instance configuration (选择实例配置), 请为您的服务实例选择 CPU 单位和内存的组合 (以 GB 为单位)。

创建服务时, 其状态将从正在创建更改为正在运行。

9. 服务开始运行后, 右键单击它并选择 Copy Service URL (复制服务 URL)。
10. 要访问已部署的应用程序, 请将复制的 URL 粘贴到 Web 浏览器的地址栏中。

#### Note

如果您尝试创建 App Runner 服务失败, 则该服务将在 AWS Explorer 中显示状态 Create failed (创建失败)。有关故障排除技巧, 请参阅 App Runner 开发人员指南中的[服务创建失败时](#)。

## 管理 App Runner 服务

创建 App Runner 服务后, 您可以使用 AWS 资源管理器窗格对其进行管理, 以执行以下活动:

- [暂停和恢复 App Runner 服务](#)
- [部署 App Runner 服务](#)
- [查看 App Runner 的日志流](#)
- [删除 App Runner 服务](#)

## 暂停和恢复 App Runner 服务

如果您需要暂时禁用 Web 应用程序并停止代码运行，则可以暂停 AWS App Runner 服务。App Runner 会将服务的计算容量降至零。当你准备好再次运行应用程序时，请恢复 App Runner 服务。App Runner 将预置新的计算容量，为其部署应用程序，然后运行该应用程序。

### Important

仅当 App Runner 运行时，您才需要为它付费。因此，您可以根据需要暂停和恢复应用程序，以便控制成本。这在开发和测试方案中特别有用。

### 暂停 App Runner 服务

1. 如果 AWS 资源管理器尚未打开，请将其打开。
2. 展开 App Runner 以查看服务列表。
3. 右键单击服务并选择 Pause ( 暂停 )。
4. 在显示的对话框中，选择 Confirm ( 确认 )。

在服务暂停期间，服务状态将从 Running ( 正在运行 ) 变为 Pausing ( 正在暂停 )，然后变为 Paused ( 已暂停 )。

### 恢复 App Runner 服务

1. 如果 AWS 资源管理器尚未打开，请将其打开。
2. 展开 App Runner 以查看服务列表。
3. 右键单击服务并选择 Resume ( 恢复 )。

在服务恢复期间，服务状态将从 Resuming ( 正在恢复 ) 变为 Running ( 正在运行 )。

## 部署 App Runner 服务

如果为服务选择手动部署选项，则需要明确启动服务的每个部署。

1. 如果 AWS 资源管理器尚未打开，请将其打开。
2. 展开 App Runner 以查看服务列表。
3. 右键单击服务并选择 Start Deployment ( 开始部署 )。

4. 在应用程序部署期间，服务状态将从 Deploying (正在部署) 变为 Running (正在运行)。
5. 要确认应用程序已成功部署，请右键单击同一服务，然后选择 Copy Service URL (复制服务 URL)。
6. 要访问已部署的 Web 应用程序，请将复制的 URL 粘贴到 Web 浏览器的地址栏中。

## 查看 App Runner 的日志流

使用 CloudWatch 日志来监控、存储和访问诸如 App Runner 之类的服务的日志流。日志流是共享同一来源的一系列日志事件。

1. 展开 App Runner 以查看服务实例列表。
2. 展开特定服务实例，以查看日志组列表。(日志组是一组具有相同保留期、监控和访问控制设置的日志流。)
3. 右键单击日志组并选择 View Log Streams (查看日志流)。
4. 在命令面板中，从组中选择一个日志流。

VS Code 编辑器将显示组成流的日志事件列表。您可以选择将较旧或更新的事件加载到编辑器中。

## 删除 App Runner 服务

### Important

如果您删除 App Runner 服务，则它将被永久删除，并且您存储的数据也将被删除。如果您需要重新创建服务，则 App Runner 需要重新获取您的源代码并构建它(如果它是代码存储库)。您的 Web 应用程序将获得一个新的 App Runner 域。

1. 如果 AWS 资源管理器尚未打开，请将其打开。
2. 展开 App Runner 以查看服务列表。
3. 右键单击服务，然后选择 Delete Service (删除服务)。
4. 在命令面板中，输入删除，然后按输入键以确认。

已删除的服务将显示 Deleting (正在删除) 状态，然后这些服务将从列表中消失。

# AWS Application Builder

AWS Toolkit for Visual Studio Code 的 AWS Application Builder 可指导您直观地构建项目、在本地迭代项目以及将应用程序部署到 AWS。

以下主题介绍了如何通过 AWS Toolkit for Visual Studio Code 使用 AWS Application Builder。

## 主题

- [使用 App AWS 开发器](#)

## 使用 App AWS 开发器

以下各节介绍如何在 VS Code 中访问 AWS 应用程序生成器 AWS Toolkit for Visual Studio Code。借助 Application Builder，您可以直观地构建项目，在本地对其进行迭代，然后将其部署到 AWS。有关 App Builder 的功能和潜在用例的概述以及您的本地 AWS Lambda 体验，请观看 AWS 开发者 YouTube 视频 [\\*新增\\* Lambda Local ID AWS E 体验！](#)。

## 使用 App AWS 开发器资源管理器

要在 AWS 工具包中访问应用程序生成器，请在 VS Code 中打开 AWS 工具包，然后展开 AWS 应用程序生成器资源管理器。App AWS 开发器资源管理器包含一个在 VS Code 编辑器选项卡中打开应用程序生成器演练的链接，并显示当前 VS Code 工作区中包含 AWS 应用程序生成器相关资源的文件夹。

在 AWS Toolkit 的 Application Builder 资源管理器中，有 4 个 project-folder-level 操作可以通过项目文件夹旁边的按钮图标或打开（右键单击）项目文件夹的上下文菜单来访问：

- 打开模板文件：在 VS Code 浏览器中打开模板文件。
- 使用基础架构编排器打开：在 VS Code 编辑器中使用 AWS 基础设施编排器打开您的模板文件。有关使用 AWS 基础架构 Composer 的详细信息，请参阅 [《AWS 基础设施编排开发者指南》](#) 中的 [“AWS 基础设施编排器是什么”](#) 主题。
- 构建 SAM 模板：在 AWS 工具包中打开“为生成指定参数”对话框。您可以选择为构建指定构建标志或使用 samconfig 中的默认值。有关 AWS SAM 模板的详细信息，请参阅 [《AWS Serverless Application Model 开发人员指南》](#) 中的 [模板解剖学](#) 主题。
- 部署 SAM 应用程序：在 VS Code 中打开选择部署命令对话框，您可以在其中选择部署应用程序或同步来更新已部署的应用程序。有关部署 AWS SAM 应用程序的详细信息，请参阅 [《AWS Serverless Application Model 开发人员指南》](#) 中的 [“部署您的应用程序和资源”](#) 主题。

有两个操作可以通过项目文件夹中 AWS Lambda 函数旁边的按钮图标或右键单击该 AWS Lambda 函数来访问：

- 本地调用和调试配置：在 VS Code 编辑器中打开本地调用和调试配置表单。使用此表单，您可以创建、编辑和运行以下类型的启动配置：`aws-sam`。有关 SAM 调试配置的更多信息，请参阅本用户指南中的[调试无服务器应用程序的配置选项](#)主题。

#### Note

目前，VS Code 不支持在 ARM64 架构上调试 .NET Core 应用程序。如果您尝试调试 .NET Core 应用程序，则会显示以下错误：

```
The vsdbg debugger does not currently support the arm64 architecture. Function will run locally without debug.
```

有关此问题的更多详细信息，请参阅存储库中的这个 [VSCode-csharp](#) 问题。DotNet GitHub

- 打开函数处理程序：打开包含函数处理程序的项目文件。

对于已部署的 AWS Lambda 函数，还有 2 个额外操作可用。

- 远程调用：在 VS Code 编辑器中打开远程调用配置菜单。
- 搜索日志：在 VS Code 中打开搜索日志对话框。

## Application Builder 演练

Application Builder 演练是一份 step-by-step 交互式指南，它将引导您完成使用 Application Builder 构建新 AWS 应用程序的过程。您可以从两个位置访问 Application Builder 演练：位于“App Builder”资源管理器 AWS Toolkit for Visual Studio Code 和“VS Code 欢迎使用”选项卡。在 AWS 工具包的“应用程序生成器资源管理器”中选择“应用程序生成器演练”时，它会在 VS Code Editor 窗口的“VS Code 欢迎”选项卡中打开“应用程序生成器演练”。

Application Builder 演练由 5 个主要部分组成：

### 1. 安装

“安装”部分会检查您是否安装了 Application Builder 所需的 AWS CLI 工具和其他可选工具。如果您未安装所需工具或工具已过期，则系统会指导您安装正确的版本。

要查看是否安装了正确 AWS CLI 和可选的工具，请选择要测试的工具 AWS CLI 或其他工具的按钮。选择按钮后，AWS Toolkit 日志会更新，VS Code 会显示一条提醒消息，告知您工具状态。如果您需要安装或更新工具，Application Builder 演练会更新您继续操作所需的说明和资源。

有关安装的详细信息 AWS CLI，请参阅《AWS CLI 开发人员指南》中的[“安装或更新至最新版本”](#) AWS CLI 主题。有关安装 AWS SAM CLI 的详细信息，请参阅 C [AWS SAM CLI 开发人员指南中的安装 AWS SAM CLI](#) 主题。

## 2. 选择应用程序模板

“选择应用程序模板”部分将指导您基于模板构建新应用程序。

要选择模板并初始化应用程序，请完成以下步骤。

1. 在 Application Builder 演练中，选择选择应用程序模板部分，以在屏幕上显示模板选项列表。
2. 从列表中选择一个模板，然后选择初始化项目按钮以打开 VS Code 对话框。
3. 完成 VS Code 对话框中的步骤以初始化新应用程序。
4. 在初始化过程中，AWS Toolkit 日志会根据应用程序的状态进行更新。
5. 要在 Application Builder 浏览器中查看应用程序，请选择刷新 Application Builder 浏览器图标，使用所做的更改更新浏览器。

## 3. 本地迭代

本地迭代部分包含示例图片，这些图像演示了如何使用 VS Code 和 AWS Toolkit 资源管理器中提供的应用程序生成器功能进行迭代。

有关 VS Code 和 AWS Toolkit 资源管理器中所有可用的 App Builder 功能的更多信息，请参阅本用户指南主题中的“使用应用程序生成器资源管理器”部分。

## 4. 部署到 AWS

“部署到 AWS”部分包含有关如何配置用于部署应用程序的连接凭 AWS 据的信息，以及如何使用 Application Builder 部署应用程序的示例。

要 AWS 使用 Application Builder 演练中的现有凭据进行连接，请完成以下过程之一。

员工：AWS 使用单点登录登录。

1. 从 Application Builder 演练的“部署到”AWS 部分，选择“配置凭据”按钮，在 AWS Toolkit 资源管理器中打开 AWS:LOGIN 菜单。

2. 在 AWS : 登录菜单中选择员工，然后选择继续按钮继续。
3. 在提供的字段中输入起始 URL，从下拉菜单中选择 AWS 区域，然后选择继续按钮继续。
4. 在 VS Code 弹出窗口中，确认您要在默认浏览器中打开 AWS 身份验证站点。
5. 使用默认浏览器完成身份验证步骤，当身份验证完成时您会收到通知，此时即可安全地关闭浏览器窗口。

IAM 证书：存储用于 AWS CLI 工具的密钥。

1. 从 Application Builder 演练的“部署到”AWS部分，选择“配置凭据”按钮，在 AWS Toolkit 资源管理器中打开AWS:LOGIN 菜单。
2. 在 AWS : 登录菜单中，选择 IAM 凭证，然后选择继续按钮继续。
3. 在提供的字段中输入配置文件名称，然后输入您的 **Access Key** 和 **Secret Key**，接下来选择继续按钮继续。
4. VS Code 会显示身份验证状态，并在身份验证完成或凭证无效时通知您。

有关使用配置部署凭证的详细信息 AWS CLI，请参阅《AWS CLI开发人员指南》中的[配置 AWS CLI](#)主题。有关使用现有凭据 AWS 从 AWS Toolkit [连接](#)的更多信息，请参阅本用户指南中的[连接 AWS](#)主题。

## AWS 基础架构编辑器

您可以将 AWS Toolkit for Visual Studio Code 与 AWS 基础架构编辑器结合使用。AWS基础架构编辑器是一款用于 AWS 应用程序的可视化生成器，可帮助您设计应用程序架构并直观呈现 CloudFormation 基础架构。

有关 AWS 基础架构编辑器的详细信息，请参阅《[AWS 基础架构编辑器](#)用户指南》。

以下主题介绍了如何通过 AWS Toolkit for Visual Studio Code 使用 AWS 基础架构编辑器。

主题

- [使用 Toolkit 中的 AWS 基础架构编辑器](#)

## 使用 Toolkit 中的 AWS 基础架构编辑器

AWS Toolkit for Visual Studio Code 的 AWS 基础架构编辑器支持您通过交互式画布直观地设计应用程序。您还可以使用基础架构编辑器来直观地查看和修改 CloudFormation 和 AWS Serverless

Application Model ( AWS SAM ) 模板。在使用基础架构编辑器时，您所做的更改会永久存储，以确保您能够在直接在 VS Code 编辑器中编辑文件或使用交互式画布之间无缝切换。

有关 AWS 基础架构编辑器的详细信息、入门信息和教程，请参阅《[AWS 基础架构编辑器用户指南](#)》。

以下各节介绍了如何通过 AWS Toolkit for Visual Studio Code 访问 AWS 基础架构编辑器。

## 通过 Toolkit 访问 AWS 基础架构编辑器

通过 Toolkit 访问 AWS 基础架构编辑器主要有 3 种方式。

### 通过现有模板访问 AWS 基础架构编辑器

1. 进入 VS Code，在 VS Code 编辑器中打开现有的模板文件。
2. 在编辑器窗口中，单击位于编辑器窗口右上角的“AWS 基础架构编辑器”按钮。
3. AWS 基础架构编辑器随即打开，并在 VS Code 编辑器窗口中直观呈现您的模板文件。

### 通过上下文菜单 ( 右键单击 ) 访问 AWS 基础架构编辑器

1. 在 VS Code 中，右键单击您想要使用 AWS 基础架构编辑器打开的模板文件。
2. 在上下文菜单中，选择使用应用程序编辑器打开选项。
3. 系统会随即打开 AWS 基础架构编辑器，并在新的 VS Code 编辑器窗口中直观呈现您的模板文件。

### 通过命令面板访问 AWS 基础架构编辑器

1. 在 VS Code 中，按下 **Cmd + Shift + P** ( 在 Windows 中，则需按下 **Ctrl + Shift + P** ) 以打开命令面板。
2. 在搜索字段中，输入 **AWS Infrastructure Composer**，然后在搜索结果中出现 AWS 基础架构编辑器时选择它。
3. 选择您想要打开的模板文件，系统会随即打开 AWS 基础架构编辑器，并在新的 VS Code 编辑器窗口中直观呈现您的模板文件。

# AWS CDK for VS Code

这是适用于预览版中功能的预发布文档。本文档随时可能更改。

通过 AWS CDK 服务，您可以使用 [AWS Cloud Development Kit \(AWS CDK\)](#) 应用程序或应用。您可以在《[AWS Cloud Development Kit \(AWS CDK\) 开发人员指南](#)》中找到有关 AWS CDK 的详细信息。

AWS CDK 应用程序由称为 [构造](#) 的构建块组成，其中包括您的 CloudFormation 堆栈和其中 AWS 资源的定义。使用 AWS CDK Explorer，您能够以可视化形式呈现在 AWS CDK 构造中定义的 [堆栈](#) 和 [资源](#)。此可视化视图在 Visual Studio Code (VS Code) 编辑器内“开发人员工具”窗格中的树视图中提供。

本节提供有关如何在 VS Code 编辑器中访问和使用 AWS CDK 的信息。此过程假定您已为本地 IDE [安装和配置](#) Toolkit for VS Code。

## 主题

- [与 AWS CDK 应用程序](#)

## 与 AWS CDK 应用程序

这是适用于预览版中功能的预发布文档。本文档随时可能更改。

使用 VS Code AWS 工具包中的 AWS CDK Explorer for VS Code 来可视化和使用 AWS CDK 应用程序。

## 先决条件

- 确保您的系统满足 [安装 Toolkit for VS Code](#) 中指定的先决条件。
- 按照《[AWS Cloud Development Kit \(AWS CDK\) 开发人员指南](#)》中 [AWS CDK 入门的前几节所述](#)，安装 AWS CDK 命令行界面。

### Important

AWS CDK 版本必须是 1.17.0 或更高版本。在命令行上使用 `cdk --version` 可查看您正在运行的版本。

## 可视化 AWS CDK 应用程序

使用适用于 VS Code AWS CDK Explorer 的 AWS Toolkit，您可以管理存储在应用程序的 CDK 结构中的[堆栈](#)和[资源](#)。资源 AWS CDK 管理器使用在运行 `cdk synth` 命令时创建 `tree.json` 的文件中定义的信息，在树视图中显示您的资源。默认情况下，`tree.json` 文件位于应用程序的 `cdk.out` 目录中。

要开始使用 AWS Toolkit Explorer，您需要创建一个 CDK 应用程序。

1. 完成 [《AWS CDK 开发人员指南》](#) 中 [Hello World 教程](#) 的前几个步骤。

### Important

当您到达教程步骤部署堆栈时，请停下来返回本指南。

### Note

您可以在操作系统命令行上或在 VS Code 编辑器内的终端窗口中，运行本教程中提供的命令（例如 `mkdir` 和 `cdk init`）。

2. 完成 CDK 教程的所需步骤后，打开您在 VS Code 编辑器中创建的 CDK 内容。
3. 在 AWS 导航窗格中，展开 CDK（预览）标题。您的 CDK 应用程序及其关联资源现在显示在 CDK Explorer 树视图中。

### 重要提示

- 将 CDK 应用程序加载到 VS Code 编辑器中后，您可以一次加载多个文件夹。每个文件夹可以包含多个 CDK 应用程序，如上图所示。AWS CDK 资源管理器可在项目根目录及其直接子目录中查找应用程序。
- 执行教程的前几个步骤时，您可能会注意到所执行的最后一个命令是 `cdk synth`，该命令会生成 `tree.json` 文件。如果您对 CDK 应用程序的一些部分做出更改（例如添加更多资源），则需要再次执行该命令，以便查看树视图中反映的更改。

## 在上执行其他操作 AWS CDK 应用程序

您可以使用 VS Code 编辑器对 CDK 应用程序执行其他操作，就像使用操作系统命令行或其他工具一样。例如，您可以更新编辑器中的代码文件，并使用 VS Code 终端窗口部署应用程序。

要尝试这些类型的操作，请使用 VS Code 编辑器继续查看《AWS CDK 开发人员指南》中的 [Hello World 教程](#)。请务必执行最后一步，销毁应用程序的资源，这样您的 AWS 账户就不会产生意想不到的费用。

## 使用 AWS CloudFormation 堆栈

AWS Toolkit for Visual Studio Code 提供对 [AWS CloudFormation](#) 堆栈的支持。使用 Toolkit for VS Code，您可以对 AWS CloudFormation 堆栈执行特定的任务，例如删除这些堆栈。

### 主题

- [删除 CloudFormation 堆栈](#)
- [创建一个 AWS CloudFormation 模板使用 AWS Toolkit for Visual Studio Code](#)

## 删除 CloudFormation 堆栈

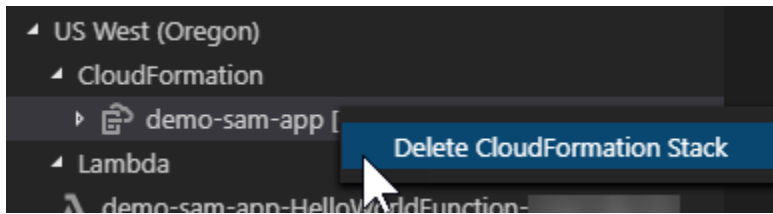
您可以使用 AWS Toolkit for Visual Studio Code 删除 CloudFormation 堆栈。

### 先决条件

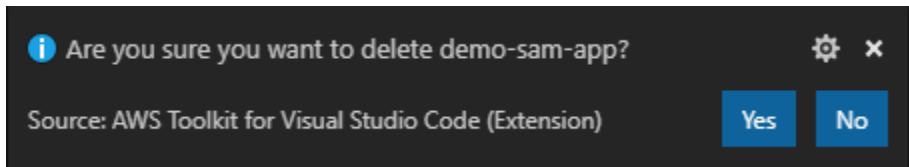
- 确保您的系统满足[安装 Toolkit for VS Code](#) 中指定的先决条件。
- 确保您在[身份验证和访问](#)中配置的凭证包含对 CloudFormation 服务的适当读/写访问权限。如果在 AWS Explorer 中的 CloudFormation 下，您看到类似于“Error loading CloudFormation resources ( 加载 CloudFormation 资源时出错 )”的消息，请检查附加到这些凭证的权限。对权限所做的更改需要几分钟时间才会反映在 VS Code 中的 AWS Explorer 内。

## 删除 CloudFormation 堆栈

1. 在 AWS Explorer 中，打开要删除的 CloudFormation 堆栈的上下文菜单。



2. 选择 Delete CloudFormation Stack ( 删除 CloudFormation 堆栈 )。
3. 在出现的消息中，选择 Yes ( 是 ) 以确认删除。



删除堆栈后，它不再在 AWS Explorer 中列出。

## 创建一个 AWS CloudFormation 模板使用 AWS Toolkit for Visual Studio Code

AWS Toolkit for Visual Studio Code 可以帮助您编写 AWS CloudFormation 和 SAM 模板。

### 先决条件

Toolkit for VS Code 和凭证先决条件

- 在从 VS Code Toolkit 访问该 CloudFormation 服务之前，您需要满足 [《安装适用于 VS Code 的工具包》](#) 用户指南中列出的要求。
- 您在中创建的凭据 [身份验证和访问](#) 必须包括对 AWS CloudFormation 服务的适当 read/write 访问权限。

#### Note

如果 CloudFormation 服务显示加载 CloudFormation 资源时出错消息，请检查您附加到这些证书的权限。另请注意，对权限所做的更改可能需要几分钟才会在 AWS Explorer 中更新。

CloudFormation 模板先决条件

- 安装并启用 [Redhat Developer YAML VS Code](#) 扩展。

- 使用 Redhat Developer YAML VS Code 扩展程序时，您需要连接到互联网，因为它用于在您的计算机上下载和缓存 JSON 架构。

## 使用 YAML Schema Support 编写 CloudFormation 模板

该工具包使用 YAML 语言支持和 JSON 架构来简化编写过程 CloudFormation 和 SAM 模板。语法验证和自动补全等功能不仅可以加快流程，还可以帮助提升模板的质量。为模板选择架构时，以下是推荐的最佳实践。

### CloudFormation 模板

- 文件具有 .yaml 或 .yml 扩展名。
- 该文件具有顶级 AWSTemplateFormatVersion 或 Resources 节点。

### SAM 模板

- 已经描述的所有标准 CloudFormation
- 该文件具有顶级 Transform 节点，其中包含以 AWS::Serverless 开头的值。

该架构将在文件修改后应用。例如，在向模板添加无服务器转换并保存文件后，将应用 SAM CloudFormation 模板架构。

### 语法验证

YAML 扩展将自动对您的模板应用类型验证。这会突出显示给定属性的类型无效的条目。如果将鼠标悬停在突出显示的条目上，则扩展会显示纠正措施。

### 自动补全

添加新字段、枚举值或其他[资源类型](#)时，您可以通过按 Ctrl + 空格键来启动 YAML 扩展的自动补全功能。

## 通过 AWS Toolkit for Visual Studio Code 使用 CloudWatch Logs

Amazon CloudWatch Logs 使您能够将所有系统、应用程序和 AWS 服务中的日志集中在高度可扩展的单个服务中。您可以轻松地查看它们、在其中搜索特定错误代码或模式、根据特定字段对其进行筛选，或者安全地将其归档以供将来分析。有关更多信息，请参阅 Amazon CloudWatch 用户指南中的[什么是 Amazon CloudWatch Logs ?](#)。

以下主题介绍如何通过 AWS Toolkit for Visual Studio Code 使用 AWS 账户中的 CloudWatch Logs。

## 主题

- [使用查看 CloudWatch 日志组和日志流 AWS Toolkit for Visual Studio Code](#)
- [使用处理 CloudWatch 日志流中的日志事件 AWS Toolkit for Visual Studio Code](#)
- [搜索 CloudWatch 日志组](#)
- [Amazon CloudWatch Logs Live Tail](#)

## 使用查看 CloudWatch 日志组和日志流 AWS Toolkit for Visual Studio Code

日志流是共享同一来源的一系列日志事件。Logs 中每个单独的 CloudWatch 日志源构成一个单独的日志流。

日志组是一组具有相同保留期、监控和访问控制设置的日志流。您可以定义日志组并指定向各组中放入哪些流。对可属于一个日志组的日志流数没有限制。

有关更多信息，请参阅 Amazon CloudWatch 用户指南中的[使用日志组和日志流](#)。

## 主题

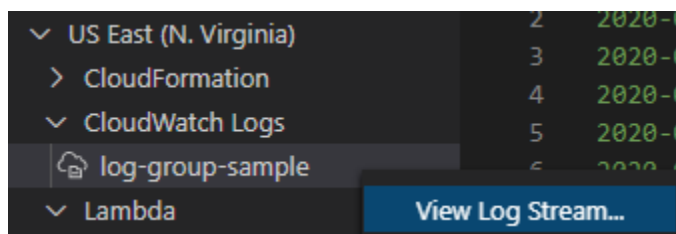
- [使用“日志”节点查看日志组和CloudWatch 日志流](#)

## 使用“日志”节点查看日志组和CloudWatch 日志流

1. 在 VS Code 中，依次选择视图和 Explorer，以打开 AWS Explorer。
2. 单击“CloudWatch 日志”节点展开日志组列表。

当前 AWS 区域的日志组显示在“CloudWatch 日志”节点下。

3. 要查看日志组中的日志流，请右键单击该日志组的名称，然后选择查看日志流。



4. 在命令面板中，从组中选择一个日志流进行查看。

**Note**

命令面板中会显示每个流中最后一个事件的时间戳。

[日志流编辑器](#)会随即启动，以显示流的日志事件。

## 使用处理 CloudWatch 日志流中的日志事件 AWS Toolkit for Visual Studio Code

在打开日志流窗口后，您可以访问每个流中的日志事件。日志事件是对受监控的应用程序或资源记录的活动的记录。

### 主题

- [查看和复制日志流信息](#)
- [将日志流编辑器的内容保存到本地文件](#)

### 查看和复制日志流信息

当您打开日志流后，日志流编辑器将显示该流的日志事件序列。

1. 要查找要查看的日志流，请打开日志流编辑器（请参阅 [查看 CloudWatch 日志组和日志流](#)）。

列出事件的每一行都有时间戳，以显示事件的录入时间。

2. 您可以使用以下选项查看和复制有关流的事件信息：

- 按时间查看事件：通过选择 Load newer events（加载更新的事件）或 Load older events（加载更早的事件）以显示最新的更早的事件。

**Note**

Log Stream（日志流）编辑器最初会加载一批最近 1 万行日志事件或 1MB 的日志数据（以较小者为准）。如果选择 Load newer events（加载更新的事件），编辑器会显示上一批事件加载后记录的事件。如果选择 Load older events（加载更早的事件），编辑器会显示在当前显示的事件之前发生的一批事件。

- 复制日志事件：选择要复制的事件，然后右键单击并从菜单中选择 Copy（复制）。
- 复制日志流的名称：右键单击日志流编辑器的选项卡，然后选择复制日志流名称。

#### Note

您也可以使用命令面板来运行 AWS Toolkit 复制日志流名称。

## 将日志流编辑器的内容保存到本地文件

您可以将 CloudWatch 日志流编辑器的内容下载到本地计算机上的 log 文件中。

#### Note

借助此选项，您可以仅将日志流编辑器中当前显示的日志事件保存到文件中。例如，如果日志流的总大小为 5MB，编辑器中仅加载了 2MB，则保存的文件也将只包含 2MB 的日志数据。要显示更多要保存的数据，请在编辑器中选择 Load newer events（加载更新的事件）或 Load older event（加载更早的事件）。

1. 要查找要复制的日志流，请打开日志流编辑器（请参阅 [查看 CloudWatch 日志组和日志流](#)）。
2. 选择显示日志流名称的选项卡旁边的保存图标。

#### Note

您也可以使用命令面板来运行 AWS Toolkit 保存当前的日志流内容。

3. 使用该对话框为日志文件选择或创建下载文件夹，然后单击 Save（保存）。

## 搜索 CloudWatch 日志组

您可以使用“搜索日志组”功能搜索日志组中的所有日志流。

有关 Amazon CloudWatch Logs 服务的详细信息，请参阅《Amazon CloudWatch 用户指南》中的[使用日志组和日志流](#)主题。

### 从 VS Code 命令面板中搜索日志组

要从 VS Code 命令面板中搜索日志组，请完成以下步骤。

有关 Amazon CloudWatch Logs 筛选条件和模式的更多信息，请参阅《Amazon CloudWatch 用户指南》中的[筛选条件和模式语法](#)部分。

1. 在 VS Code 中，按下 **cmd+shift+p** (在 Windows 中，则需按下 **ctrl+shift+p**) 以打开命令面板。
2. 在命令面板中，输入命令 **AWS: Search Log Group**，然后选择它以在 VS Code 中打开“搜索日志组”对话框，并按照提示继续操作。

#### Note

在第一个提示中，您可以选择切换 AWS 区域，然后再继续执行后续步骤。

3. 从选择日志组 ( 1/3 ) 提示中，选择要搜索的日志组。
4. 从选择时间筛选器 ( 2/3 ) 提示中，选择要应用于搜索的时间筛选器。
5. 从搜索日志组... ( 3/3 ) 提示中，在提供的字段中输入您的搜索模式，然后按下 **Enter** 键以继续操作，或者按下 **ESC** 键以取消搜索。
6. 搜索完成后，您的搜索结果将在 VS Code 编辑器中打开。


## 从 AWS Explorer 中搜索日志组

要从 AWS Toolkit for Visual Studio Code Explorer 中搜索日志组，请完成以下步骤。

1. 从 AWS Toolkit for Visual Studio Code Explorer 中展开 CloudWatch。
2. 打开要搜索的“搜索日志组”的上下文菜单（右键单击），然后选择搜索日志组以打开搜索提示。
3. 按照提示选择要继续的时间范围。
4. 当出现提示时，在提供的字段中输入您的搜索模式，然后按下 **Enter** 键以继续操作，或者按下 **ESC** 键以取消搜索。
5. 搜索完成后，您的搜索结果将在 VS Code 编辑器中打开。

## 处理搜索日志结果

成功完成 CloudWatch 日志组搜索后，您的搜索结果将在 VS Code 编辑器中打开。以下过程将介绍如何处理搜索日志结果。

 Note

查看单个日志流时，以下功能仅限于当前处于活跃状态的日志流中的结果。

### 保存搜索日志组结果

要在本地保存搜索日志组结果，请完成以下步骤。

1. 从搜索日志组结果中，选择位于 VS Code 编辑器右上角的将日志保存为文件图标按钮。
2. 从另存为提示中，指定要保存的文件名称和要将文件保存到的位置。
3. 选择确定以将文件保存到本地计算机。

### 更改时间范围

要更改搜索日志组结果中处于活跃状态的时间范围，请完成以下步骤。

1. 从搜索日志组结果中，选择位于 VS Code 编辑器右上角的按日期搜索...图标按钮。
2. 从选择时间筛选器提示中，为搜索日志结果选择新的时间范围。
3. 当选择时间筛选器提示关闭后，您的结果就会更新。

### 更改搜索模式

要更改搜索日志组结果中处于活跃状态的搜索模式，请完成以下步骤。

1. 从搜索日志组结果中，选择位于 VS Code 编辑器右上角的按模式搜索...图标按钮。
2. 在搜索日志组提示中，在提供的字段中输入新的搜索模式。
3. 按 **Enter** 键关闭提示并使用新的搜索模式更新结果。

## Amazon CloudWatch Logs Live Tail

使用 Amazon CloudWatch Logs Live Tail 在 CloudWatch 日志事件摄入特定日志组时实时予以流式处理。

有关 Live Tail 功能的详细信息，请参阅《Amazon CloudWatch Logs 用户指南》中的[使用 CloudWatch Logs Live Tail 进行故障排除](#)。

Live Tail 会话按会话使用时间每分钟产生费用。有关定价的信息，请查看 [Amazon CloudWatch 定价指南](#) 付费套餐部分的日志选项卡。

## 从 VS Code 命令面板启动 Live Tail 会话

要从 VS Code 命令面板启动 Live Tail 会话，请完成以下步骤。

有关 Amazon CloudWatch Logs 筛选条件和模式的更多信息，请参阅《Amazon CloudWatch 用户指南》中的 [筛选条件和模式语法](#) 部分。

### 通过命令面板启动跟踪会话

1. 在 VS Code 中，按 **cmd+shift+p**（在 Windows 中，则需按 **ctrl+shift+p**）以打开命令面板。
2. 在命令面板中，输入命令 `AWS: Tail Log Group`，然后选择它以在 VS Code 中打开跟踪日志组对话框，并按照提示继续操作。

#### Note

在第一个提示中，您可以选择切换 AWS 区域，然后再继续执行后续步骤。

3. 在跟踪日志组（1/3）提示中，选择要跟踪的日志组。
4. 在包含来自...的日志事件（2/3）提示中，选择要应用于跟踪会话的日志流筛选条件。
5. 在提供日志事件筛选条件模式...（3/3）提示中，在提供的字段中输入您的筛选条件模式，然后按 **Enter** 键以继续操作，或者按 **ESC** 键以取消搜索。
6. 完成后，结果将流式传输到 VS Code 编辑器中

#### Note

如果在 VS Code 窗口中运行的 Live Tail 会话与新提交的 Tail Log Group 命令的配置相匹配，则新会话不会启动。相反，您的现有会话将变为活动的文本编辑器。

## 通过 AWS 各区服务浏览器启动 Live Tail 会话

要通过 AWS Toolkit 浏览器启动 Live Tail 会话，请完成以下步骤。

## 通过 AWS 各区服务浏览器启动跟踪会话

1. 在 AWS Toolkit 浏览器中，展开 CloudWatch。
2. 打开要跟踪的日志组的上下文菜单（右键单击），然后选择跟踪日志组以打开跟踪提示。
3. 按照提示继续操作。
4. 结果将流式传输到 VS Code 编辑器中。

## 停止 Live Tail 会话

有两种方法可以停止正在运行的跟踪会话。

### 停止跟踪会话

1. 单击跟踪会话文本文档底部的停止跟踪 CodeLens。
2. 关闭所有包含跟踪会话文本文档的编辑器。

# Amazon DocumentDB

您可以使用 AWS Toolkit for Visual Studio Code 在 VS Code 中直接管理 Amazon DocumentDB 集群和实例。Amazon DocumentDB（兼容 MongoDB）是一项快速、可靠的完全托管式数据库服务，可简化在云中设置、操作和扩展与 MongoDB 兼容的数据库的过程。有关 Amazon DocumentDB 服务的详细信息，请参阅《[Amazon DocumentDB 开发人员指南](#)》。

以下主题介绍了如何通过 AWS Toolkit for Visual Studio Code 使用 Amazon DocumentDB。

### 主题

- [在 Toolkit 中使用 Amazon DocumentDB](#)

## 在 Toolkit 中使用 Amazon DocumentDB

Amazon DocumentDB（兼容 MongoDB）是一项快速、可靠的完全托管式数据库服务，可简化在云中设置、操作和扩展与 MongoDB 兼容的数据库的过程。

有关 Amazon DocumentDB 的详细信息、入门信息和教程，请参阅《[Amazon DocumentDB 开发人员指南](#)》。

以下各节介绍了如何通过 AWS Toolkit for Visual Studio Code 使用 Amazon DocumentDB。

## 通过工具包访问亚马逊 DocumentDB AWS

要使用该 AWS 工具包访问 Amazon DocumentDB，请完成以下步骤。

在工具包中访问亚马逊 DocumentDB AWS

1. 从 VS Code 中，打开 AWS Toolkit for Visual Studio Code。
2. 在 AWS 工具包中，展开资源管理器。
3. 在浏览器中，展开 Amazon DocumentDB 以显示您现有的 Amazon DocumentDB 资源。

创建基于实例的集群。

要开始使用 Amazon DocumentDB，请完成以下过程来创建一个集群。

创建基于实例的集群

1. 从中 AWS Toolkit for Visual Studio Code，打开（右键单击）Amazon DocumentDB 的快捷菜单，然后选择创建集群，在 VS Code 中打开“创建亚马逊文档数据库集群”对话框。
2. 在集群类型屏幕上，选择基于实例的集群。
3. 在集群名称屏幕上，指定新集群的名称。
4. 在选择引擎版本屏幕上，选择您的首选 Amazon DocumentDB 引擎版本。
5. 在管理员用户名和密码屏幕上，指定管理员用户名和密码以保护集群。
6. 在指定存储加密屏幕上，选择是否加密集群。
7. 在实例数量屏幕上，配置您的首选实例数量。
8. 在选择实例类屏幕上，选择您的首选实例类，然后继续创建新集群。

### Note

启动集群可能需要几分钟的时间。

复制集群端点

要复制 Amazon DocumentDB 集群端点，请完成以下过程。

## 复制集群端点

1. 从中 AWS Toolkit for Visual Studio Code，展开 Amazon DocumentDB 以显示您的亚马逊 DocumentDB 集群。
2. 右键单击要从中复制连接详细信息的集群，然后选择复制端点以将集群端点信息复制到剪贴板。
3. 您的集群端点现在可以粘贴到您的文档中。

## 在浏览器中打开

在 AWS 控制台中打开您的 Amazon DocumentDB 集群，以获得更多集群管理功能。要在默认网络浏览器中打开您的 Amazon DocumentDB 集群的 AWS 控制台，请完成以下步骤。

### 在 AWS 控制台中打开集群

1. 从中 AWS Toolkit for Visual Studio Code，展开 Amazon DocumentDB 以显示您的亚马逊 DocumentDB 集群。
2. 在 AWS 控制台中右键单击要查看的集群，然后选择在浏览器中打开。
3. AWS 控制台将在您的默认网络浏览器中打开 Amazon DocumentDB 集群。

## 扩展现有集群

要通过添加实例来扩展 Amazon DocumentDB 集群，请完成以下过程。

### 添加实例以扩展集群

1. 从中 AWS Toolkit for Visual Studio Code，展开 Amazon DocumentDB 以显示您的亚马逊 DocumentDB 集群。
2. 右键单击要扩展的集群，然后选择添加实例以在 VS Code 中打开添加实例对话框。
3. 出现提示时，在文本字段中输入新实例的名称，然后按 **Enter** 键继续。
4. 出现提示时，从列表中选择实例类以继续。
5. AWS 各区服务浏览器会显示创建状态，并在新实例准备就绪后进行更新。

## 停止集群

要停止 Amazon DocumentDB 集群，请完成以下过程。

**Note**

当集群停止时，大多数集群管理功能将不可用。

## 停止 Amazon DocumentDB 集群

1. 从中 AWS Toolkit for Visual Studio Code，展开 Amazon DocumentDB 以显示您的亚马逊 DocumentDB 集群。
2. 选择位于要停止的集群旁边的停止集群按钮，或者右键单击集群并选择停止集群。
3. 出现提示时，选择是以停止集群，或者选择取消以取消停止过程并保持集群运行。
4. AWS 各区服务浏览器会显示集群状态，并在集群停止后进行更新。

## 重启实例

重启实例对于在不影响整个集群的情况下进行故障排除和进行细微更改非常有用。要重启 Amazon DocumentDB 实例，请完成以下过程。

### 重启集群实例

1. 从中 AWS Toolkit for Visual Studio Code，展开 Amazon DocumentDB 以显示您的亚马逊 DocumentDB 集群。
2. 右键单击要重启的集群实例，然后选择重启实例。
3. 出现提示时，选择是以重启实例，或者选择取消以取消重启过程并保持实例停止。
4. AWS 各区服务浏览器会显示集群状态，并在实例重启后进行更新。

## 删除实例

要删除 Amazon DocumentDB 集群实例，请完成以下过程。

**Note**

删除实例并不会影响集群中的数据。如果您删除主实例，其中一个副本实例将作为可写实例进行接管。

## 删除集群实例

1. 从中 AWS Toolkit for Visual Studio Code ，展开 Amazon DocumentDB 以显示您的亚马逊 DocumentDB 集群。
2. 右键单击要删除的集群实例，然后选择删除以在 VS Code 中打开 delete-cluster-instance 确认对话框。
3. 按照确认提示操作，然后按 **Enter** 键删除集群实例。
4. AWS 各区服务浏览器会显示集群实例状态，并在实例被删除后进行更新。

## 查看、添加和删除标签

标签用于组织和跟踪环境中的资源。要查看或编辑与 Amazon DocumentDB 集群关联的标签，请完成下列过程之一。

### 查看集群标签

1. 从中 AWS Toolkit for Visual Studio Code ，展开 Amazon DocumentDB 以显示您的亚马逊 DocumentDB 集群。
2. 右键单击要查看其标签的集群，然后选择标签... 以打开 **your cluster name** 的标签对话框。
3. 标签会显示在对话框窗口中，如果没有标签与集群相关联，则会显示 [未分配标签] 这一消息。

### 向集群添加标签

1. 从中 AWS Toolkit for Visual Studio Code ，展开 Amazon DocumentDB 以显示您的亚马逊 DocumentDB 集群。
2. 右键单击要为其添加标签的集群，然后选择标签... 以打开 **your cluster name** 的标签对话框。
3. 选择添加标签... 按钮以在 VS Code 中打开添加标签对话框。
4. 在文本字段中输入新标签，然后按 Enter 键继续。
5. 在文本字段中输入一个值，然后按 Enter 将该 key/value 对添加到您的集群中。

### 从集群中删除标签

1. 从中 AWS Toolkit for Visual Studio Code ，展开 Amazon DocumentDB 以显示您的亚马逊 DocumentDB 集群。
2. 右键单击要从中删除标签的集群，然后选择标签... 以打开 **your cluster name** 的标签对话框。

3. 选择删除标签... 按钮以在 VS Code 中打开从 **your cluster name** 删除标签对话框。
4. 从提供的列表中选择要删除的标签，将该标签从集群中删除。

## 修改实例类

要修改 Amazon DocumentDB 集群实例类，请完成以下过程。

### 修改实例类

1. 从中 AWS Toolkit for Visual Studio Code，展开 Amazon DocumentDB 以显示您的亚马逊 DocumentDB 集群。
2. 右键单击要修改的集群实例，然后选择修改类... 以在 VS Code 中打开选择实例类对话框。
3. 从列表中为实例选择一个新类来更新该类。
4. AWS 各区服务浏览器会显示集群实例状态，并在实例类更新后进行更新。

## Amazon Elastic Compute Cloud

借助适用于 AWS Toolkit for Visual Studio Code 的 Amazon Elastic Compute Cloud，您可以从 VS Code 启动和连接您的 Amazon EC2 实例。有关 Amazon EC2 的详细信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的[什么是 Amazon EC2 ?](#)。

以下主题介绍了如何通过 AWS Toolkit for Visual Studio Code 使用 AWS Application Builder。

### 主题

- [使用 Amazon Elastic Cloud](#)
- [Amazon Elastic Compute Cloud 问题排查](#)

## 使用 Amazon Elastic Cloud

以下各部分介绍了如何在 AWS Toolkit for Visual Studio Code 中使用 Amazon Elastic Compute Cloud。

### 先决条件

本用户指南主题中描述的功能已在使用以下操作系统的 Amazon EC2 实例上进行了测试：

- Windows 2016+

**Note**

此操作系统仅在连接 VS Code 终端时起才可运行。它连接完整 VS Code 远程实例时无法运行。有关 VS Code 终端和远程实例的更多信息，请参阅 VS Code 文档中的 [Getting started with the terminal](#) 和 [VS Code Remote Development](#) 主题。

- Amazon Linux 2023
- Ubuntu 22.04

要打开与 Amazon EC2 实例的远程连接，需要本地安装的 SSH，但无需使用本地安装的 SSH 即可打开与 Amazon EC2 实例的终端。

您的 Amazon EC2 实例配置文件必须包含以下 AWS Identity and Access Management (IAM) 权限。

```
"ssmmessages:CreateControlChannel",  
"ssmmessages:CreateDataChannel",  
"ssmmessages:OpenControlChannel",  
"ssmmessages:OpenDataChannel",  
"ssm:DescribeAssociation",  
"ssm:ListAssociations",  
"ssm:UpdateInstanceInformation"
```

**Note**

所需的权限包含在以下 AWS 托管策略中。

- AmazonSSMManagedInstanceCore
- AmazonSSMManagedEC2InstanceDefaultPolicy

## 查看现有的 Amazon EC2 实例

要通过 AWS 工具包查看您的现有 Amazon EC2 实例，请完成以下步骤。

1. 在 AWS 工具包中，展开 AWS 工具包资源管理器。
2. 扩展包含您要查看的 Amazon EC2 实例的区域。

3. 展开标EC2题以显示您的现有 Amazon EC2 实例。

## 启动一个新的 Amazon EC2 实例

有三种方法可以使用该 AWS 工具包创建新的 Amazon EC2 实例。

每个工作流都会在 AWS 管理控制台中打开启动实例向导。有关通过启动实例向导启动新 Amazon EC2 实例的详细信息，请参阅 Amazon Elastic Compute Cloud 用户指南中[控制台中的使用启动实例向导启动实例主题](#)。EC2要启动新的 Amazon EC2 实例，请完成以下步骤之一。

### 从 VS Code 命令面板启动一个新的 Amazon EC2 实例

1. 在 VS Code 中，按下 **command + shift + P (Windows: ctrl + shift + P)** 以打开 VS Code 命令面板
2. 在 VS Code 命令面板中，搜索该**AWS: Launch EC2**命令并在其填充到列表中时将其选中，以在 VS Code 中打开 Launch EC2 实例选择区域提示符。
3. 在启动 EC2 实例选择区域提示中，选择要在其中启动新实例的区域，然后确认要在默认 Web 浏览器中打开 AWS 控制台。
4. 在默认 Web 浏览器的 AWS 控制台中，完成身份验证过程以继续启动实例向导。
5. 在启动实例向导中，完成必填部分，然后选择启动实例按钮以启动您的新 Amazon EC2 实例。
6. AWS Explorer 会更新以显示您的新 Amazon EC2 实例。

### 从 AWS Explorer 启动一个新的 Amazon EC2 实例

1. 展开 AWS Toolkit Explorer，然后展开要在其中创建新 Amazon EC2 实例的区域。
2. 展开或将鼠标悬停在EC2标题上，然后选择 + (启动 EC2 实例) 图标。
3. 出现提示时，确认您要在默认 Web 浏览器中打开 AWS 控制台。
4. 在 Web 浏览器的 AWS 控制台中，完成身份验证过程以继续启动实例向导。
5. 在启动实例向导中，完成必填部分，然后选择启动实例按钮以启动您的新 Amazon EC2 实例。
6. AWS Explorer 会更新以显示您的新 Amazon EC2 实例。

### 从上下文 (右键单击) 菜单启动新的 Amazon EC2 实例

1. 展开 AWS Toolkit Explorer，然后展开要在其中创建新 Amazon EC2 实例的区域。
2. 右键单击标EC2题，然后选择启动 EC2 实例。

3. 出现提示时，确认您要在默认 Web 浏览器中打开 AWS 控制台。
4. 在 Web 浏览器的 AWS 控制台中，完成身份验证过程以继续启动实例向导。
5. 在启动实例向导中，完成必填部分，然后选择启动实例按钮以启动您的新 Amazon EC2 实例。
6. AWS Explorer 会更新以显示您的新 Amazon EC2 实例。

## 将 VS Code 连接到亚马逊 EC2 实例

通过 VS Code 连接亚马逊 EC2 实例有三种方法。要将 VS Code 连接到您的 EC2 实例，请完成以下过程之一。

### 通过命令面板将 VS Code 连接到 Amazon EC2 实例

1. 在 VS Code 中，按下 **command + shift + P (Windows: ctrl + shift + P)** 以打开 VS Code 命令面板
2. 从 VS Code 命令面板中搜索该 **AWS: Connect VS Code to EC2 instance...** 命令，并在该命令填充到列表中时将其选中，以在 VS Code 中打开“选择 EC2 实例”提示符。
3. 在选择 EC2 实例提示中，选择包含您要连接的实例的区域，然后选择要连接的实例。
4. VS Code 在建立连接时会显示状态。
5. 连接完成后，将打开一个新窗口，显示您的 Amazon EC2 实例。

### 通过 AWS 资源管理器将 VS Code 连接到 Amazon EC2 实例。

1. 展开 AWS Toolkit Explorer，然后展开包含您要连接的 Amazon EC2 实例的区域。
2. 将鼠标悬停在 Amazon EC2 实例上，然后选择（将 VS 代码连接到 EC2 实例）图标。

#### Note


您也可以从 AWS 资源管理器的 EC2 服务标题中选择（Connect VS Code 到 EC2 实例）图标。

3. VS Code 在建立连接时会显示状态。
4. 连接完成后，将打开一个新窗口，显示您的 Amazon EC2 实例。

### 通过右键单击菜单将 VS Code 连接到 Amazon EC2 实例

1. 展开 AWS Toolkit Explorer，然后展开包含您要连接的 Amazon EC2 实例的区域。

2. 右键单击要连接的 Amazon EC2 实例，然后选择 **Connect VS Code** 到 EC2 实例。

 Note

您也可以在 AWS 资源管理器中右键单击 EC2 服务标题，然后选择要 EC2 实例的 **Connect VS Code**。

3. VS Code 在建立连接时会显示状态。
4. 连接完成后，将打开一个新窗口，显示您的 Amazon EC2 实例。

## 打开一个连接到 Amazon EC2 实例的终端。


有三种方法可以从 VS Code 终端连接到 Amazon EC2 实例。

### 通过命令面板将 VS Code 连接到 Amazon EC2 实例

1. 在 VS Code 中，按下 **command + shift + P (Windows: ctrl + shift + P)** 以打开 VS Code 命令面板
2. 从 VS Code 命令面板中搜索该 **AWS:Open terminal to EC2 instance...** 命令，并在该命令填充到列表中时将其选中，以在 VS Code 中打开“选择 EC2 实例”提示符。
3. 从“选择 EC2 实例”提示中，选择包含要在终端中打开的实例的区域，然后选择该实例。
4. VS Code 在建立连接时会显示状态。
5. 连接完成后，系统将打开 VS Code 终端，显示您的新会话。

通过 AWS 资源管理器在 VS Code 终端中打开一个 Amazon EC2 实例。

1. 展开 AWS Toolkit Explorer，然后展开包含您要连接的 Amazon EC2 实例的区域。
2. 将鼠标悬停在 Amazon EC2 实例上，然后选择（打开终端到 EC2 实例...）图标。

 Note

你也可以选择（打开终端到 EC2 实例...）AWS 资源管理器中 EC2 服务标题中的图标。

3. VS Code 在建立连接时会显示状态。
4. 连接完成后，系统将打开 VS Code 终端，显示您的新会话。

## 通过右键单击菜单在 VS Code 终端中打开 Amazon EC2 实例

1. 展开 AWS Toolkit Explorer，然后展开包含要在 VS Code 终端中打开的 Amazon EC2 实例的区域。
2. 右键单击要在终端中打开的 Amazon EC2 实例，然后选择打开终端到 EC2实例...。

### Note

您也可以在 AWS 资源管理器中右键单击 EC2 服务标题，然后选择打开终端进行 EC2 实例...。

3. VS Code 在建立连接时会显示状态。
4. 连接完成后，系统将打开 VS Code 终端，显示您的新会话。

## 启动或重启 Amazon 实例 EC2

有三种方法可以启动或重启 Amazon EC2 实例。

### 从命令面板重启 Amazon EC2 实例

1. 在 VS Code 中，按下 **command + shift + P (Windows: ctrl + shift + P)** 以打开 VS Code 命令面板
2. 从 VS Code 命令面板中搜索该 **AWS: Reboot EC2 instance** 命令，并在该命令填充到列表中时将其选中，以在 VS Code 中打开“选择 EC2 实例”提示符。

### Note


要启动未运行的实例，必须选择 **AWS: Start EC2 instance** 命令。**AWS: Reboot EC2 instance** 命令仅重启当前正在运行的实例。

3. 从“选择 EC2 实例”提示中，选择包含您要启动或重启的实例的区域。
4. VS Code 会在实例重启时显示状态。
5. E AWS xplorer 会更新以显示您的实例在完成重启后正在运行。

### 从资源管理器启动或重启 Amazon EC2 实例 AWS

1. 展开 AWS Toolkit Explorer，然后展开包含您要启动或重启的 Amazon EC2 实例的区域。

2. 将鼠标悬停在 Amazon EC2 实例上，然后选择 ( 重启 EC2 实例 ) 图标。


 Note

如果实例已停止，则唯一的选项是 ( 启动 EC2 实例 ) 图标

3. VS Code 会在实例重启时显示状态。
4. E AWS xplorer 会更新以显示您的实例在完成重启后正在运行。

### 通过右键单击菜单启动或重启 Amazon EC2 实例

1. 展开 AWS Toolkit Explorer，然后展开包含您要启动或重启的 Amazon EC2 实例的区域。
2. 右键单击要连接的 Amazon EC2 实例，然后选择重启 EC2 实例。

 Note

如果实例已停止，则唯一的选项是启动 EC2 实例。

3. VS Code 会在实例重启时显示状态。
4. E AWS xplorer 会更新以显示您的实例在完成重启后正在运行。

### 停止 Amazon EC2 实例

有三种方法可以停止 Amazon EC2 实例。

#### 从命令面板中停止 Amazon EC2 实例

1. 在 VS Code 中，按下 **command + shift + P (Windows: ctrl + shift + P)** 以打开 VS Code 命令面板
2. 从 VS Code 命令面板中搜索该 **AWS: Stop EC2 instance** 命令，并在该命令填充到列表中时将其选中，以在 VS Code 中打开“选择 EC2 实例”提示符。
3. 从“选择 EC2 实例”提示中，选择包含您要停止的实例的区域。
4. VS Code 会在实例停止时显示状态。
5. E AWS xplorer 会更新以显示您的实例已停止。

## 从 AWS Explorer 中停止 Amazon EC2 实例

1. 展开 AWS Toolkit Explorer，然后展开包含您要停止的 Amazon EC2 实例的区域。
2. 将鼠标悬停在 Amazon EC2 实例上，然后选择 ( 停止 EC2 实例 ) 图标。
3. VS Code 会在实例停止时显示状态。
4. E AWS xplorer 会更新以显示您的实例已停止。

## 通过右键单击菜单停止 Amazon EC2 实例

1. 展开 AWS Toolkit Explorer，然后展开包含您要停止的 Amazon EC2 实例的区域。
2. 右键单击要连接的 Amazon EC2 实例，然后选择重启 EC2 实例。
3. VS Code 会在实例停止时显示状态。
4. E AWS xplorer 会更新以显示您的实例已停止。

## 复制实例 ID

要复制实例 ID，请完成以下步骤。

1. 右键单击要从中复制 ID 的实例。
2. 选择复制实例 ID。
3. 系统会将实例 ID 复制到您的本地剪贴板。

## 复制名称

要复制实例名称，请完成以下步骤。

1. 右键单击要从中复制名称的实例。
2. 选择复制实例名称。
3. 系统会将实例名称复制到您的本地剪贴板。

## 复制 ARN

要复制实例 ARN，请完成以下步骤。

1. 右键单击要从中复制 ARN 的实例。

2. 选择复制实例 ARN。
3. 系统会将实例 ARN 复制到您的本地剪贴板。

## Amazon Elastic Compute Cloud 问题排查

以下各部分介绍了如何排查在 AWS Toolkit for Visual Studio Code 中使用 Amazon Elastic Compute Cloud 时可能出现的已知问题。有关排查特定于 Amazon EC2 服务的问题的详细信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的[排查 Amazon EC2 实例的问题](#)主题。

### 通用调试

如果您出于任何原因遇到远程连接问题，请首先检查是否可以从 AWS 控制台建立 AWS Systems Manager 连接。

要从 AWS 控制台通过 Systems Manager 连接到 Amazon EC2 实例，请完成以下步骤。

1. 在 Web 浏览器中，导航到 [AWS 管理控制台](#)。
2. 完成身份验证以继续登录 AWS 控制台 EC2。
3. 在 Amazon EC2 导航窗格中，选择实例。
4. 选中要连接的实例旁边的框。
5. 选择连接按钮，系统将在新的浏览器标签页中打开连接到实例屏幕。

#### Note

仅当实例处于运行状态时，您才能连接到该实例。如果您无法选择连接按钮，请进行检查以确保您的实例处于运行状态。

6. 在连接到实例屏幕中，选择会话管理器选项卡，然后选择连接按钮，系统将在当前浏览器标签页中打开 Systems Manager 连接。

#### Note

如果您最近启动了实例，但该选项不可用于连接 Systems Manager，则可能需要等待几分钟后该选项才会可用。

## 目标实例未运行

要通过终端或远程连接来连接到 Amazon EC2 实例，该实例必须处于运行状态。在您尝试从 AWS Toolkit 连接到您的实例之前，请先从 AWS Explorer 启动实例 AWS 管理控制台、或 AWS Command Line Interface。

## 目标实例没有 IAM 角色或其 IAM 角色权限不正确

要连接到您的 Amazon EC2 实例，该实例必须具有附加正确权限的 IAM 角色。如果您尝试连接到未附加 IAM 角色的实例，VS Code 将向您发出通知。

如果您尝试连接到具有 IAM 角色但缺少必要权限的实例，系统会提示您向现有 IAM 角色添加最低限度的必要操作作为内联策略。更新内联策略后，您即可连接到您的实例。有关 IAM 角色、权限及向实例附加角色的详细信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的[适用于 Amazon EC2 的 IAM 角色](#)主题和《AWS Systems Manager 用户指南》中的[步骤 2：验证或添加 Session Manager 的实例权限](#)主题。

以下示例包含最低限度的必要操作。

```
"ssmmessages:CreateControlChannel",  
"ssmmessages:CreateDataChannel",  
"ssmmessages:OpenControlChannel",  
"ssmmessages:OpenDataChannel",  
"ssm:DescribeAssociation",  
"ssm:ListAssociations",  
"ssm:UpdateInstanceInformation"
```

### Note

所需的权限包含在以下 AWS 托管策略中。

- AmazonSSMManagedEC2InstanceDefaultPolicy
- AmazonSSMManagedInstanceCore

## 目标实例没有运行中的 Systems Manager Agent

您可能会因为多种不同原因而遇到此问题。要解决此问题，请首先重启实例并再次尝试连接。或者，通过非 ssm 连接方法手动启动初始连接。有关 Systems Manager 的更多详细信息，请参阅《AWS Systems Manager 用户指南》中的[使用 Systems Manager Agent](#) 主题。

启动时，Amazon EC2 状态显示为正在运行，但无法建立连接

如果您最近为实例启动或创建了新的 IAM 角色但无法建立连接，请再等待几分钟，然后再尝试建立连接。

## 使用 Amazon Elastic Container Registry

Amazon Elastic Container Registry ( Amazon ECR ) 是 AWS 托管式容器注册表服务，它安全且可扩展。多项 Amazon ECR 服务功能可以通过 Toolkit for VS Code Explorer 访问。

- 创建存储库。
- 为您的存储库或已标记的映像创建 AWS App Runner 服务。
- 访问映像标签和存储库 URI 或 ARN。
- 删除映像标签和存储库。

还可以通过将 AWS CLI 和其他平台与 VS Code 集成，通过 VS Code 控制台访问各项 Amazon ECR 功能。

有关 Amazon ECR 的更多信息，请参阅《Amazon Elastic Container Registry 用户指南》中的[什么是 Amazon ECR ?](#)

主题

- [使用 Amazon Elastic Container Registry](#)
- [通过 Amazon ECR 创建 App Runner 服务](#)

## 使用 Amazon Elastic Container Registry

您可以在 VS Code 中直接从 AWS Explorer 访问亚马逊弹性容器注册表 (Amazon ECR) Registry 服务，然后使用它将程序映像推送到亚马逊 ECR 存储库。要开始使用，您需要执行以下步骤：

1. 创建一个 Dockerfile，其中包含构建映像所需的信息。

2. 从该 Dockerfile 生成映像并标记该映像以供处理。
3. 在 Amazon ECR 实例内创建一个存储库。
4. 将标记的映像推送到此存储库。

## 先决条件

要通过 VS Code Explorer 中访问 Amazon ECR 服务，您需要完成下面这些步骤。

### 创建 IAM 用户

在访问某项 AWS 服务（例如 Amazon ECR）之前，您必须提供凭证。这样，相应服务便可以确定您是否有权访问其资源。我们不建议您 AWS 直接通过根 AWS 账户的证书进行访问。而是使用 AWS Identity and Access Management (IAM) 创建 IAM 用户，然后将该用户添加到具有管理权限的 IAM 群组。然后，您可以使用特殊的 URL 和 IAM 用户的证书进行访问 AWS。

如果您已注册 AWS 但没有为自己创建 IAM 用户，则可以使用 IAM 控制台创建一个。

要创建管理员用户，请选择以下选项之一。

选择一种方法来管理您的管理员	目标	方式	您也可以
在 IAM Identity Center 中 (推荐)	使用短期凭证访问 AWS。  这符合安全最佳实践。有关最佳实践的信息，请参阅《IAM 用户指南》中的 <a href="#">IAM 中的安全最佳实践</a> 。	有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 <a href="#">入门</a> 。	通过在《AWS Command Line Interface 用户指南》 <a href="#">AWS IAM Identity Center 中配置 AWS CLI 要使用的来配置编程访问权限</a> 。
在 IAM 中	使用长期凭证访问 AWS。	按照《IAM 用户指南》中的 <a href="#">创建用于紧急访问的 IAM 用户</a> 中的说明进行操作。	按照《IAM 用户指南》中的 <a href="#">管理 IAM 用户的访问密钥</a> ，配置程式访问。

选择一种方法来管理您的管理员	目标	方式	您也可以
(不推荐使用)			

要以此新 IAM 用户的身份登录，请退出 AWS 控制台，然后使用以下 URL。在以下 URL 中，`your_aws_account_id` 是没有连字符的 AWS 账号（例如，如果您的 AWS 账号是，则您的账户 ID 是）：`1234-5678-9012 AWS 123456789012`

```
https://your_aws_account_id.signin.aws.amazon.com/console/
```

输入您刚创建的 IAM 用户名和密码。登录后，导航栏显示“`your_user_name @ your_aws_account_id`”。

如果您不希望登录页面的 URL 包含您的 AWS 账户 ID，则可以创建账户别名。从 IAM 控制面板中，选择自定义，然后输入账户别名。账户别名可以是您的公司名称。有关更多信息，请参阅 IAM 用户指南中的[您的 AWS 账户 ID 及其别名](#)。

要在创建账户别名后登录，请使用以下 URL：

```
https://your_account_alias.signin.aws.amazon.com/console/
```

要为您的账户验证 IAM 用户的登录链接，请打开 IAM 控制台并在控制面板的 IAM 用户登录链接下进行检查。

有关 IAM 的更多信息，请参阅 [AWS Identity and Access Management 用户指南](#)。

## 安装和配置 Docker

您可以通过从[安装 Docker 引擎](#)用户指南中选择首选操作系统，并按照说明来安装和配置 Docker。

## 安装和配置 AWS CLI 版本 2

通过从安装、[更新和卸载 AWS CLI 版本 2 用户指南](#)中选择您的首选操作系统来安装和配置 AWS CLI 版本 2。

## 1. 创建 Dockerfile

Docker 使用名为“Dockerfile”的文件来定义可以推送和存储在远程存储库中的映像。您必须先创建 Dockerfile，然后通过该 Dockerfile 构建映像，然后才能将映像上传到 ECR 存储库。

### 创建 Dockerfile

1. 使用 Toolkit for VS Code 资源管理器导航至要将 Dockerfile 存储到的目录。
2. 创建一个名为 Dockerfile 的新文件。

#### Note

VS Code 可能会提示您选择文件类型或文件扩展名。如果出现该提示，请选择纯文本。Vs Code 具有“dockerfile”扩展名。但是，我们建议您不要使用它。这是因为该扩展名可能会导致与某些版本的 Docker 或其他关联应用程序发生冲突。

### 使用 VS Code 编辑 Dockerfile

如果 Dockerfile 具有文件扩展名，请打开该文件的上下文（右键单击）菜单，然后移除文件扩展名。

从 Dockerfile 中删除文件扩展名后：

1. 直接在 VS Code 中打开空的 Dockerfile。
2. 将以下示例的内容复制到您的 Dockerfile 中：

#### Example Dockerfile 映像模板

```
FROM ubuntu:18.04

# Install dependencies
RUN apt-get update && \
    apt-get -y install apache2

# Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure apache
RUN echo '. /etc/apache2/envvars' > /root/run_apache.sh && \
    echo 'mkdir -p /var/run/apache2' >> /root/run_apache.sh && \
```

```
echo 'mkdir -p /var/lock/apache2' >> /root/run_apache.sh && \  
echo '/usr/sbin/apache2 -D FOREGROUND' >> /root/run_apache.sh && \  
chmod 755 /root/run_apache.sh
```

```
EXPOSE 80
```

```
CMD /root/run_apache.sh
```

这是使用 Ubuntu 18.04 映像的 Dockerfile。RUN 指令将更新软件包缓存。安装一些适用于 Web 服务器的软件包，然后将“Hello World!” 内容写入到 Web 服务器的文档根目录。EXPOSE 指令在容器上公开端口 80，而 CMD 指令启动 Web 服务器。

### 3. 保存您的 Dockerfile。

#### Important

确保您的 Dockerfile 名称上没有附加扩展名。带有扩展名的 Dockerfile 可能会导致与某些版本的 Docker 或其他关联应用程序发生冲突。

## 2. 通过 Dockerfile 构建映像

您创建的 Dockerfile 包含为程序构建映像所需的信息。您必须先构建映像，然后才能将该映像推送到 Amazon ECR 实例。

### 通过 Dockerfile 构建映像

1. 要导航到包含您的 Dockerfile 的目录，请使用 Docker CLI 或与您的 Docker 实例集成的 CLI。
2. 要构建在 Dockerfile 中定义的映像，请运行 Docker build 命令。

```
docker build -t hello-world .
```

3. 要验证是否已正确创建映像，请运行 Docker images 命令。

```
docker images --filter reference=hello-world
```

Example输出示例：

REPOSITORY SIZE	TAG	IMAGE ID	CREATED
hello-world 241MB	latest	e9ffedc8c286	4 minutes ago

4.

#### Note

创建或推送映像无需执行此步骤，但是您可以看到程序映像在运行时的工作情况。

要运行新构建的映像，请使用 Docker run 命令。

```
docker run -t -i -p 80:80 hello-world
```

前面示例中指定的 `-p` 选项将容器上暴露的端口 80 映射到主机系统的端口 80。如果您正在本地运行 Docker，可使用 Web 浏览器导航至 <http://localhost:80>。如果程序运行正常，系统将显示“Hello World!”语句。

有关 Docker run 命令的更多信息，请参阅 Docker 网站上的 [Docker Run 参考](#)。

### 3. 创建新存储库

要将您的映像上载到您的 Amazon ECR 实例，请创建一个新的存储库来存储它。

创建新的 Amazon ECR 存储库

1. 从 VS Code 活动栏中，选择 AWS Toolkit 图标。
2. 展开“AWS 资源管理器”菜单。
3. 找到与您的 AWS 账户关联的默认 AWS 区域。然后，选择它以查看通过 Toolkit for VS Cod 提供的服务列表。
4. 选择 ECR + 选项以开始创建新存储库流程。
5. 要完成该流程，请按照提示操作。
6. 完成后，您可以从 AWS 资源管理器菜单的 ECR 部分访问您的新存储库。

## 4. 推送、拉取和删除映像

从 Dockerfile 构建映像并创建存储库后，您可以将映像推送到 Amazon ECR 存储库中。此外，使用带有 Docker 和 AWS CLI 的 AWS 资源管理器可以执行以下操作：

- 从存储库中提取映像。
- 删除存储在存储库中的映像。
- 删除存储库。

### 使用默认存储库对 Docker 进行身份验证

在 Amazon ECR 实例和 Docker 实例之间交换数据要求进行身份验证。使用注册表对 Docker 进行身份验证：

1. 打开连接到您的 AWS CLI 实例的命令行操作系统。
2. 使用该 `get-login-password` 方法向您的私有 ECR 注册表进行身份验证。

```
aws ecr get-login-password --region region | docker login --username AWS --password-stdin AWS_account_id.dkr.ecr.region.amazonaws.com
```

#### Important

在上述命令中，您必须将 **region** 和 **AWS\_account\_id** 更新为您的 AWS 账户的特定信息。

### 标记映像并将其推送到存储库

使用您的实例对 Docker 进行身份验证后 AWS，将映像推送到您的存储库。

1. 使用 `Docker images` 命令查看您在本地存储的映像，并识别要标记的映像。

```
docker images
```

Example 输出示例：

REPOSITORY SIZE	TAG	IMAGE ID	CREATED
hello-world 241MB	latest	e9ffedc8c286	4 minutes ago

## 2. 使用 Docker tag 命令标记映像。

```
docker tag hello-world:latest AWS_account_id.dkr.ecr.region.amazonaws.com/hello-world:latest
```

## 3. 使用 Docker tag 命令将标记的映像推送到您的存储库。

```
docker push AWS_account_id.dkr.ecr.region.amazonaws.com/hello-world:latest
```

Example 输出示例：

```
The push refers to a repository [AWS_account_id.dkr.ecr.region.amazonaws.com/hello-world] (len: 1)
e9ae3c220b23: Pushed
a6785352b25c: Pushed
0998bf8fb9e9: Pushed
0a85502c06c9: Pushed
latest: digest:
  sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636b95aed25f52c89b size: 6774
```

将带标签的图像成功上传到存储库后，它就会显示在 AWS 资源管理器菜单中。

## 从 Amazon ECR 拉取映像

- 您可以将映像拉取到您的 Docker tag 命令的本地实例。

```
docker pull AWS_account_id.dkr.ecr.region.amazonaws.com/hello-world:latest
```

Example 输出示例：

```
The push refers to a repository [AWS_account_id.dkr.ecr.region.amazonaws.com/hello-world] (len: 1)
e9ae3c220b23: Pushed
a6785352b25c: Pushed
0998bf8fb9e9: Pushed
0a85502c06c9: Pushed
latest: digest:
  sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636b95aed25f52c89b size: 6774
```

## 从 Amazon ECR 存储库中删除映像

从 VS Code 中删除映像的方法有两种。第一种方法是使用 AWS 资源管理器。

1. 在 AWS 资源管理器中，展开 ECR 菜单
2. 展开要从中删除映像的存储库
3. 打开上下文菜单（右键单击），选择与您想要删除的映像关联的映像标签
4. 要删除与该标签关联的所有存储的映像，请选择删除标签...

## 使用 AWS CLI 删除镜像

- 您也可以使用 AWS `ecr batch-delete-image` 命令从存储库中删除图像。

```
AWS ecr batch-delete-image \  
  --repository-name hello-world \  
  --image-ids imageTag=latest
```

Example 输出示例：

```
{  
  "failures": [],  
  "imageIds": [  
    {  
      "imageTag": "latest",  
      "imageDigest":  
      "sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636b95aed25f52c89b"  
    }  
  ]  
}
```

```
}
```

## 从 Amazon ECR 实例中删除存储库

从 VS Code 中删除存储库的方法有两种。第一种方法是使用 AWS 资源管理器。

1. 在 AWS 资源管理器中，展开 ECR 菜单
2. 打开上下文（右键单击）菜单，选择您要删除的存储库
3. 选择删除存储库...选项，以选择相应存储库

## 从 CLI 中删除 Amazon ECR 存储库 AWS

- 您可以使用 `AWS ecr delete-repository` 命令删除存储库。

### Note

默认情况下，您不能删除包含映像的存储库。但是，`--force` 标记允许这样做。

```
AWS ecr delete-repository \  
  --repository-name hello-world \  
  --force
```

Example 输出示例：

```
{  
  "failures": [],  
  "imageIds": [  
    {  
      "imageTag": "latest",  
      "imageDigest":  
"sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636b95aed25f52c89b"  
    }  
  ]  
}
```

## 通过 Amazon ECR 创建 App Runner 服务

以下主题介绍如何从亚马逊弹性容器注册表 (Amazon ECR) Container Registry 节点创建和启动 AWS App Runner 服务，位于中。AWS Toolkit for Visual Studio Code 有关 AWS App Runner 和 Amazon ECR 服务的详细信息，请参阅[AWS App Runner](#)和 [Amazon ECR 用户指南](#)。

### 先决条件

在 AWS 工具包中创建和启动 AWS App Runner 来自 Amazon ECR 之前，必须完成以下操作。有关如何完成这些步骤的详细指南，请参阅本用户指南中的[使用 Amazon Elastic Container Registry](#) 主题。

1. 创建 dockerfile。
2. 通过 dockerfile 构建映像。
3. 创建新存储库。
4. 标记映像并将其推送到存储库。

## 从现有 Amazon ECR 存储库创建 AWS App Runner 服务

以下过程介绍如何使用 AWS 工具包中的现有 Amazon ECR 存储库创建 AWS App Runner 服务。

1. 在 AWS 资源管理器中，展开包含您要从中创建 AWS App Runner 服务的 Amazon ECR 存储库的区域。
2. 展开 Amazon ECR 服务节点，查看 Amazon ECR 存储库。
3. 打开（右键单击）要创建 AWS App Runner 服务的 Amazon ECR 存储库或存储库映像的快捷菜单。
4. 从上下文菜单中，选择创建 App Runner 服务，在 VS Code 中打开 AWS App Runner 创建向导
5. 在输入新服务的端口（1/5）中，输入要使用的端口号，然后按 **Enter** 键继续。
6. 在配置环境变量（2/5）中，选择使用文件...，进行浏览以选择浏览您的本地文件，或选择跳过以跳过此步骤。
7. 在选择要从 ECR 拉取的角色（3/5）中，从列表中选择一个现有 IAM 角色。

**Note**

需要AppRunnerECRAccess角色访问角色才能从 Amazon ECR 私有注册表创建 AWS App Runner 服务。如果列表中没有有效的角色，请选择 + ( 创建角色... ) 图标可自动创建AppRunnerECRAccess角色并将其分配给您的注册表。

8. 在为您的服务命名 ( 4/5 ) 中，输入新服务的名称，然后按 **Enter** 键继续。
9. 在选择实例配置 ( 5/5 ) 中，从列表中选择 **vCPU** 和 **Memory** 配置以创建新服务。
10. 在 AWS 资源管理器中，展开 App Runner 服务节点以查看您的 AWS App Runner 资源。新服务成功创建后，状态会自动更新为正在运行。

## 使用 Amazon Elastic Container Service

AWS Toolkit for Visual Studio Code 为 [Amazon Elastic Container Service \( Amazon ECS \)](#) 提供了一些支持。Toolkit for VS Code 可协助您完成某些与 Amazon ECS 相关的工作，例如创建任务定义。

### 主题

- [用 IntelliSense 于 Amazon ECS 任务定义文件](#)
- [Amazon 弹性容器服务执行官 AWS Toolkit for Visual Studio Code](#)

## 用 IntelliSense 于 Amazon ECS 任务定义文件

使用 Amazon Elastic Container Service ( Amazon ECS ) 时，您可能需要创建任务定义，如《Amazon Elastic Container Service 开发人员指南》中的[创建任务定义](#)部分所述。安装时 AWS Toolkit for Visual Studio Code，安装中包含了 Amazon ECS 任务定义文件的 IntelliSense 功能。

### 先决条件

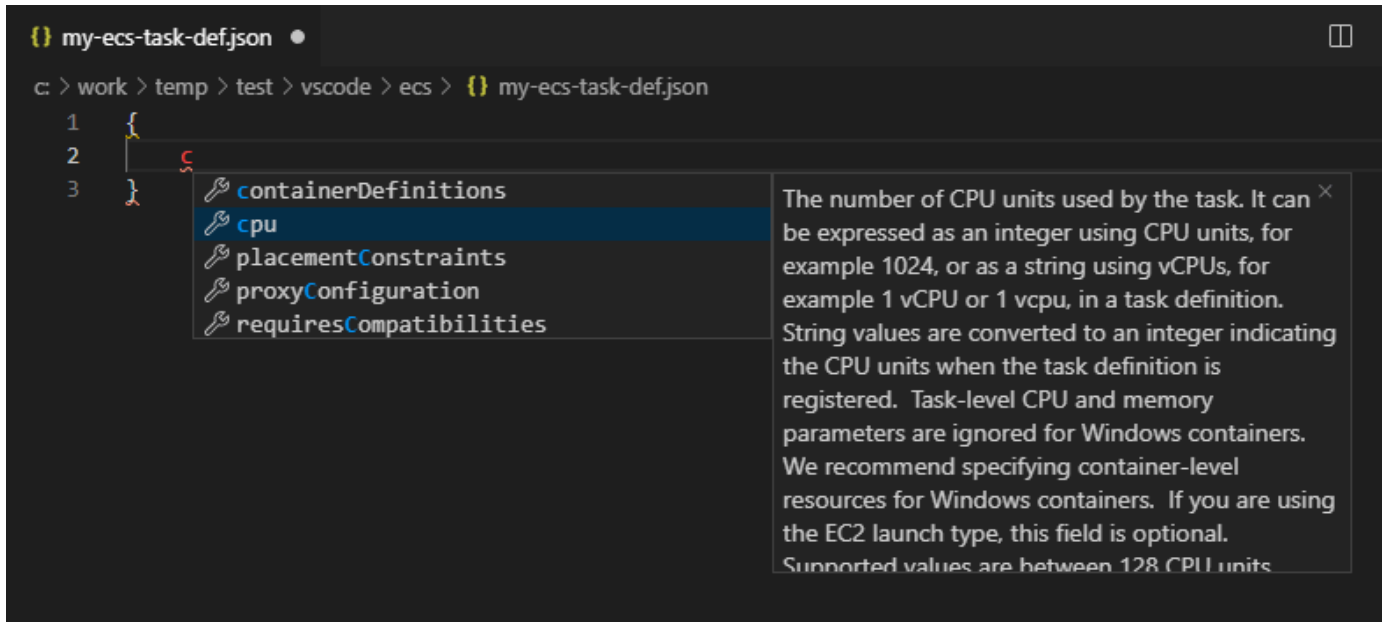
- 确保您的系统满足[安装 Toolkit for VS Code](#) 中指定的先决条件。

## IntelliSense 在 Amazon ECS 任务定义文件中使用

以下示例向您展示了如何在 Amazon ECS 任务定义文件 IntelliSense 中加以利用。

1. 为您的 Amazon ECS 任务定义创建 JSON 文件。文件名称的结尾必须包含 `ecs-task-def.json`，但在开头可以包含其他字符。

- 在本示例中，创建一个名为 `my-ecs-task-def.json` 的文件
- 在 VS Code 编辑器中打开该文件，然后输入开头的花括号。
  - 输入字母“c”，就好像您希望将 `cpu` 添加到定义中一样。观察打开的 IntelliSense 对话框，该对话框类似于以下内容。



## Amazon 弹性容器服务执行官 AWS Toolkit for Visual Studio Code

您可以使用 Amazon ECS Exec 功能在带有 AWS Toolkit for Visual Studio Code 的亚马逊弹性容器服务 (Amazon ECS) 容器中发出单个命令。

### Important

启用和禁用 Amazon ECS Exec 会更改您 AWS 账户中资源的状态。这包括停止和重新启动服务。在 Amazon ECS Exec 处于启用状态时更改资源状态可能会导致不可预测的结果。有关 Amazon ECS 的更多信息，请参阅[使用 Amazon ECS Exec 进行调试](#)开发人员指南。

## Amazon ECS Exec 先决条件

您需要满足一些先决条件才能使用 Amazon ECS Exec 功能。

### Amazon ECS 要求

根据您的任务是托管在 Amazon EC2 上还是 AWS Fargate，Amazon ECS Exec 有不同的版本要求。

- 如果您使用的是 Amazon EC2，必须使用 2021 年 1 月 20 日之后发布的经 Amazon ECS 优化的 AMI，代理版本为 1.50.2 或更高。其他信息可在开发者指南 [Amazon ECS 优化](#) 版中找到 AMIs。
- 如果您使用的是 AWS Fargate，则必须使用平台版本 1.4.0 或更高版本。开发人员指南 [AWS Fargate 平台版本](#) 中提供了有关 Fargate 要求的更多信息。

## AWS 账户配置和 IAM 权限

要使用 Amazon ECS Exec 功能，您需要将现有的 Amazon ECS 集群与您的 AWS 账户相关联。Amazon ECS Exec 使用 Systems Manager 与集群中的容器建立连接，并且需要特定的任务 IAM 角色权限才能与 SSM 服务通信。

您可以在 [ECS Exec 所需的 IAM 权限](#) 开发人员指南中找到特定于 Amazon ECS Exec 的 IAM 角色和策略信息。

## 使用 Amazon ECS Exec

您可以直接从 VS Code 工具包中的 AWS 资源管理器中启用或禁用 Amazon ECS Exec。启用 Amazon ECS Exec 后，您可以从 Amazon ECS 菜单中选择容器，然后对它们运行命令。

### 启用 Amazon ECS Exec

1. 在 AWS 资源管理器中，找到并展开 Amazon ECS 菜单。
2. 展开包含您要修改的服务的集群。
3. 打开服务的上下文菜单（右键单击），然后选择 Enable Command Execution（启用命令执行）。

#### Important

此操作将启动服务的新部署，可能需要几分钟。有关更多信息，请参阅本部分开头的注释。

### 禁用 Amazon ECS Exec

1. 在 AWS 资源管理器中，找到并展开 Amazon ECS 菜单。
2. 扩展具有所需服务的集群。
3. 打开服务的上下文菜单（右键单击），然后选择 Disable Command Execution（禁用命令执行）。

### Important

此操作将启动服务的新部署，可能需要几分钟。有关更多信息，请参阅本部分开头的注释。

## 对容器运行命令

要使用 AWS 资源管理器对容器运行命令，必须启用 Amazon ECS Exec。如果未启用，请参阅本部分中的启用 Amazon ECS Exec 流程。

1. 在 AWS 资源管理器中，找到并展开 Amazon ECS 菜单。
2. 扩展具有所需服务的集群。
3. 扩展此服务以列出关联的容器。
4. 打开容器的上下文菜单（右键单击），然后选择 Run Command in Container（在容器中运行命令）。
5. 系统会随即将打开一个提示，其中包含正在运行的任务列表，请选择所需的任务 ARN。

### Note

如果该服务仅运行一个任务，则系统会自动选择该任务并跳过此步骤。

6. 出现提示时，请输入要运行的命令，然后按下 Enter 键以继续。

## 使用 Amazon EventBridge

AWS Toolkit for Visual Studio Code (VS Code) 为 [Amazon EventBridge](#) 提供支持。使用 Toolkit for VS Code，您可以处理 EventBridge 的某些方面，例如架构。

### 主题

- [使用 Amazon EventBridge 架构](#)

## 使用 Amazon EventBridge 架构

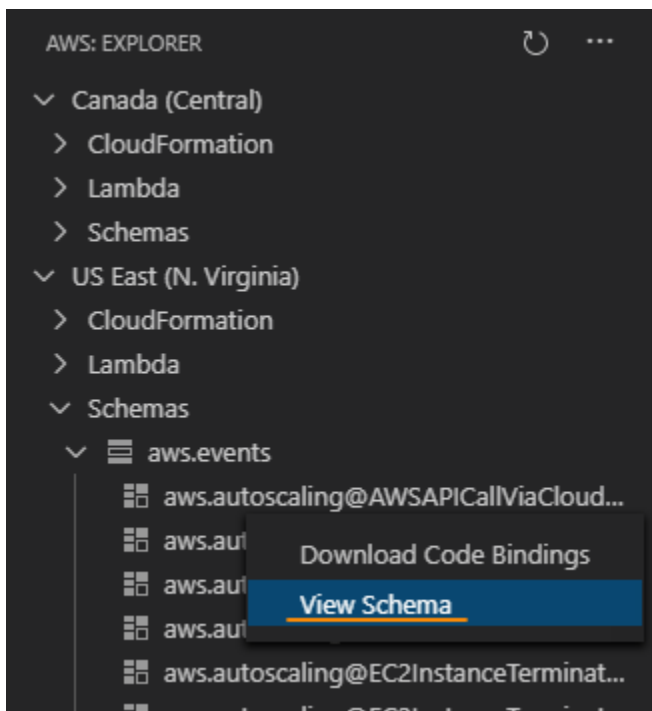
您可以使用 AWS Toolkit for Visual Studio Code (VS 代码) 在 [Amazon EventBridge 架构](#) 上执行各种操作。

## 先决条件

- 确保您的系统满足[安装 Toolkit for VS Code](#) 中指定的先决条件。
- 您要使用的 EventBridge 架构必须在您的 AWS 账户中可用。如果不可用，请创建或上传该架构。参见《[亚马逊 EventBridge 用户指南 EventBridge](#)》中的[亚马逊架构](#)。

## 查看可用架构

1. 在 AWS Explorer 中，展开架构。
2. 展开包含您希望查看的架构的注册表的名称。例如，提供的许多架构都位于 aws.events 注册表中。AWS
3. 要查看编辑器中的架构，请打开该架构的上下文菜单，然后选择 View Schema (查看架构)。



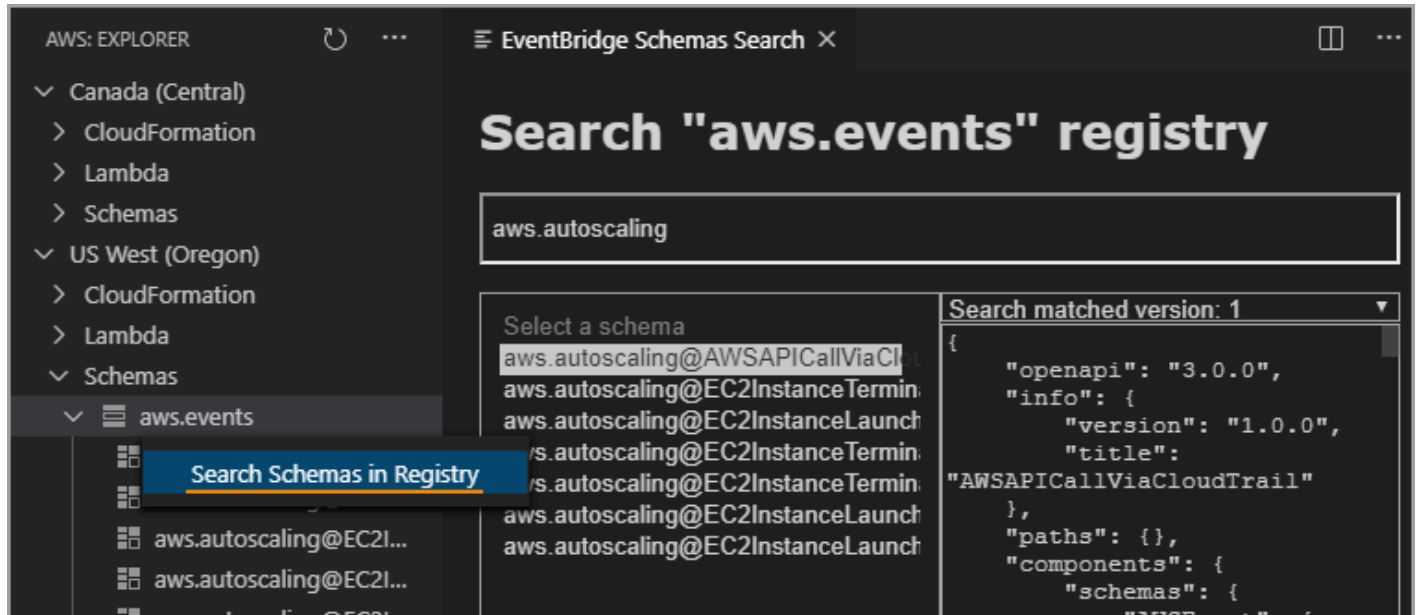
## 查找可用架构

在 AWS Explorer 中，执行以下一项或多项操作：

- 开始键入您希望查找的架构的标题。AWS Explorer 突出显示包含匹配项的架构标题。（必须扩展注册表，您才能看到突出显示的标题。）

- 打开 Schemas (架构) 的上下文菜单，然后选择 Search Schemas (搜索架构)。或者展开 Schemas (架构)，打开包含您希望查找的架构的注册表的上下文菜单，然后选择 Search Schemas in Registry (在注册表中搜索架构)。在 EventBridge Schemas Search (EventBridge 架构搜索) 对话框中，开始键入要查找的架构的标题。对话框中将显示包含匹配项的架构标题。

要在对话框中显示架构，请选择架构标题。



## 为可用架构生成代码

1. 在 AWS Explorer 中，展开架构。
2. 展开包含您希望生成代码的架构的注册表的名称。
3. 右键单击架构的标题，然后选择 Download code bindings (下载代码绑定)。
4. 在显示的向导页面中，选择以下内容：
  - 架构的版本
  - 代码绑定语言
  - 本地开发计算机上要将生成的代码存储到的工作区文件夹

# AWS IAM 访问分析器

您可以使用[中的 IAM 访问分析器对 CloudFormation 在模板、Terraform 计划和 JSON 策略文档中编写的 IAM 策略进行AWS 身份与访问管理 \(IAM\) Access Analyzer 策略检查](#)。AWS Toolkit for Visual Studio Code

IAM 访问权限分析器策略检查包括策略验证和自定义策略检查。策略验证有助于根据AWS Identity and Access Management用户指南中的 IAM [JSON 策略语言语法和 IAM](#) 主题中的 AWS [安全最佳实践中详述的标准验证您的 IAM](#) 策略。策略验证检查结果包括安全警告、错误、常规警告和策略建议。

您还可以使用自定义策略检查，根据安全标准检查新的访问。每次针对新访问的自定义策略检查都会产生费用。有关定价的详细信息，请参阅 [AWS IAM 访问权限分析器定价](#) 网站。有关 IAM 访问权限分析器策略检查的详细信息，请参阅《AWS Identity and Access Management用户指南》中的 [检查验证策略](#) 主题。

以下主题介绍了如何在 AWS Toolkit for Visual Studio Code中使用 IAM 访问权限分析器策略检查。

## 主题

- [与 AWS IAM 访问分析器](#)

## 与 AWS IAM 访问分析器

以下各部分介绍了如何在 AWS Toolkit for Visual Studio Code中执行 IAM 策略验证和自定义策略检查。有关更多详细信息，请参阅 AWS Identity and Access Management 用户指南中的以下主题：[IAM Access Analyzer 策略验证](#)和 [IAM Access Analyzer 自定义策略检查](#)。

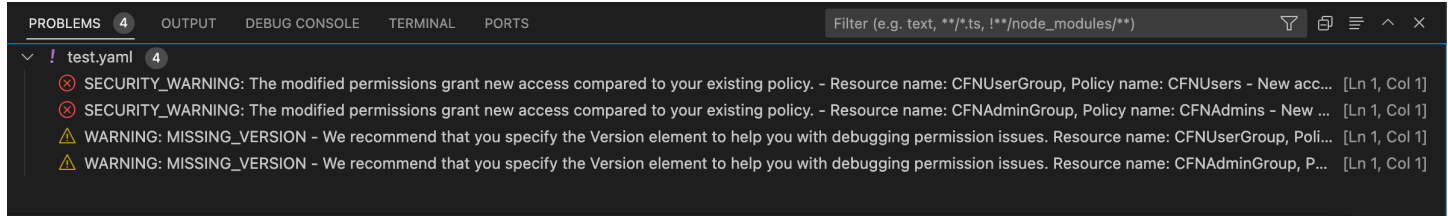
## 先决条件

要在 Toolkit 中使用 IAM 访问权限分析器策略检查，就必须先满足以下先决条件。

- 安装 Python 版本 3.6 或更高版本。
- 安装[适用于 CloudFormation的 IAM 策略验证器](#)或[适用于 Terraform 的 IAM 策略验证器](#)，这是 Python CLI 工具所必需的，并在“IAM 策略检查”窗口中指定。
- 配置您的 AWS 角色证书。

## IAM 访问权限分析器策略检查

您可以使用对 CloudFormation 模板、Terraform 计划和 JSON 策略文档执行策略检查。AWS Toolkit for Visual Studio Code 您的检查结果可在 VS Code 问题面板中查看。下图显示的是 VS Code 问题面板。



IAM 访问权限分析器提供 4 种类型的检查：

- 验证策略
- CheckAccessNotGranted
- CheckNoNewAccess
- CheckNoPublicAccess

以下各部分介绍了如何运行每种类型的检查。

### Note

在运行任何类型的检查之前，请配置您的 AWS 角色证书。支持的文件包括以下文档类型：CloudFormation 模板、Terraform 计划和 JSON 策略文档。文件路径引用通常由管理员或安全团队提供，可以是系统文件路径或 Amazon S3 存储桶 URI。要使用 Amazon S3 存储桶 URI，您的当前角色必须可以访问 Amazon S3 存储桶。每次自定义策略检查都会产生费用。有关自定义策略检查定价的详细信息，请参阅 [AWS IAM 访问权限分析器定价指南](#)。

## 运行验证策略

验证策略检查（也称为策略验证）可根据 IAM 策略语法和 AWS 最佳实践验证您的策略。有关更多信息，请参阅 [AWS Identity and Access Management 用户指南](#) 中的 [IAM JSON 策略语言语法](#) 和 [IAM 主题](#) 中的 [AWS 安全最佳实践](#)。

1. 在 VS Code 中，在 VS Code 编辑器中打开一个包含 AWS IAM 策略的支持文件。

2. 要打开 IAM 访问权限分析器策略检查，请在 VS Code 编辑器中按下 **CRTL+Shift+P** 打开 VS Code 命令面板，搜索 **IAM Policy Checks** 并进行单击以打开 IAM 策略检查窗格。
3. 在 IAM 策略检查窗格中，从下拉菜单中选择您的文档类型。
4. 从验证策略部分，选择运行策略验证按钮以运行“验证策略”检查。
5. 在 VS Code 的问题面板中，查看策略检查结果。
6. 更新策略并重复此过程，重新运行“验证策略”检查，直到策略检查结果不再显示安全警告或错误。

## 正在跑步 CheckAccessNotGranted

CheckAccessNotGranted 是一项自定义策略检查，用于验证您的策略是否不允许特定 IAM 操作。

### Note

文件路径引用通常由管理员或安全团队提供，可以是系统文件路径或 Amazon S3 存储桶 URI。要使用 Amazon S3 存储桶 URI，您的当前角色必须可以访问 Amazon S3 存储桶。必须至少指定一个操作或资源，并且文件结构应遵循以下示例：

```
    {"actions": ["action1", "action2", "action3"], "resources":  
    ["resource1", "resource2", "resource3"]}
```

1. 在 VS Code 中，在 VS Code 编辑器中打开一个包含 AWS IAM 策略的支持文件。
2. 要打开 IAM 访问权限分析器策略检查，请在 VS Code 编辑器中按下 **CRTL+Shift+P** 打开 VS Code 命令面板，搜索 **IAM Policy Checks** 并进行单击以打开 IAM 策略检查窗格。
3. 在 IAM 策略检查窗格中，从下拉菜单中选择您的文档类型。
4. 从“自定义策略检查”部分，选择 CheckAccessNotGranted。
5. 在文本输入字段中，您可以输入包含操作和资源 ARN 的以逗号分隔的列表。必须至少提供一项操作或资源。
6. 选择运行自定义策略检查按钮。
7. 在 VS Code 的问题面板中，查看策略检查结果。自定义策略检查会返回 PASS 或 FAIL 结果。
8. 更新您的政策并重复此过程，重新运行 CheckAccessNotGranted 支票直到它返回PASS。

## 正在跑步 CheckNoNewAccess

CheckNoNewAccess 是一项自定义策略检查，用于验证与参考策略相比，您的策略是否授予了新的访问权限。

1. 在 VS Code 中，在 VS Code 编辑器中打开一个包含 AWS IAM 策略的支持文件。
2. 要打开 IAM 访问权限分析器策略检查，请在 VS Code 编辑器中按下 **CRTL+Shift+P** 打开 VS Code 命令面板，搜索 **IAM Policy Checks** 并进行单击以打开 IAM 策略检查窗格。
3. 在 IAM 策略检查窗格中，从下拉菜单中选择您的文档类型。
4. 从“自定义策略检查”部分，选择 CheckNoNewAccess。
5. 输入引用 JSON 策略文档。或者，您也可以提供引用 JSON 策略文档的文件路径。
6. 选择与您的引用文档类型相匹配的引用策略类型。
7. 选择运行自定义策略检查按钮。
8. 在 VS Code 的问题面板中，查看策略检查结果。自定义策略检查会返回 PASS 或 FAIL 结果。
9. 更新您的政策并重复此过程，重新运行 CheckNoNewAccess 支票直到它返回PASS。

## 正在跑步 CheckNoPublicAccess

CheckNoPublicAccess 是一项自定义策略检查，用于验证您的策略是否授予对模板中支持的资源类型的公共访问权限。

[有关支持的资源类型的具体信息，请参阅 `cloudformation-iam-policy-validator` 和 `terraform-iam-policy-validator` 存储库。](#) GitHub

1. 在 VS Code 中，在 VS Code 编辑器中打开一个包含 AWS IAM 策略的支持文件。
2. 要打开 IAM 访问权限分析器策略检查，请在 VS Code 编辑器中按下 **CRTL+Shift+P** 打开 VS Code 命令面板，搜索 **IAM Policy Checks** 并进行单击以打开 IAM 策略检查窗格。
3. 在 IAM 策略检查窗格中，从下拉菜单中选择您的文档类型。
4. 从“自定义策略检查”部分，选择 CheckNoPublicAccess。
5. 选择运行自定义策略检查按钮。
6. 在 VS Code 的问题面板中，查看策略检查结果。自定义策略检查会返回 PASS 或 FAIL 结果。
7. 更新您的政策并重复此过程，重新运行 CheckNoNewAccess 支票直到它返回PASS。

# 在 AWS IoT 中使用 AWS Toolkit for Visual Studio Code

利用 AWS Toolkit for Visual Studio Code 中的 AWS IoT，您可以与 AWS IoT 服务交互，同时最大限度地减少对 VS Code 中工作流的干扰。本用户指南旨在帮助您开始使用 AWS Toolkit for Visual Studio Code 中提供的 AWS IoT 服务功能。有关 AWS IoT 服务的更多信息，请参阅开发者指南 [什么是 AWS IoT ?](#)

## AWS IoT 先决条件

要开始通过 Toolkit for VS Code 使用 AWS IoT，请确保您的 AWS 账户和 VS Code 符合以下指南中的要求：

- 有关特定于 AWS IoT 服务的 AWS 账户要求以及 AWS 用户权限，请参阅[开始使用 AWS IoT Core 开发人员指南](#)。
- 有关 Toolkit for VS Code 的具体要求，请参阅[设置 Toolkit for VS Code 用户指南](#)。

## AWS IoT 事物

AWS IoT 将设备连接到 AWS 云服务和资源。您可以使用名为事物的对象，将设备连接到 AWS IoT。事物是特定设备或逻辑实体的表示形式。它可以是物理设备或传感器（例如，灯泡或墙壁上的开关）。有关 AWS IoT 事物的更多信息，请参阅开发人员指南[通过 AWS IoT 管理设备](#)。

## 管理 AWS IoT 事物

Toolkit for VS Code 有多项功能可以提高您的 AWS IoT 事物的管理效率。您可以通过以下方式使用 VS Code 工具包来管理您的 AWS IoT 事物：

- [Create a thing](#)
- [Attach a certificate to a thing](#)
- [Detach a certificate from a thing](#)
- [Delete a thing](#)

## 创建事物

1. 在 AWS Explorer 中，展开 IoT 服务标题，然后打开事物的上下文菜单（右键单击）。
2. 从上下文菜单中选择创建事物，以打开对话框。

3. 按照提示在事物名称字段中输入 IoT 事物的名称。
4. 此步骤完成后，在事物部分中可以看到事物图标后跟您指定的名称。

### 将证书附加到事物

1. 在 AWS Explorer 中，展开 IoT 服务部分。
2. 在事物子部分中，找到您要将证书附加到的事物。
3. 打开该事物的上下文菜单（右键单击），然后从上下文菜单中选择附加证书以打开输入选择器，其中包含您的证书列表。
4. 从列表中，选择您要附加到事物的证书所对应的证书 ID。
5. 此步骤完成后，即可在 AWS Explorer 中将该证书作为它所附加到的事物的项目进行访问。

### 从事物分离证书

1. 在 AWS Explorer 中，展开 IoT 服务部分。
2. 在 Things（事物）子部分中，找到您要与之分离证书的事物。
3. 打开该事物的上下文菜单（右键单击），然后从上下文菜单中选择分离证书。
4. 此步骤完成后，分离的证书将不再显示在 AWS Explorer 中的该事物下，但仍可以从证书子部分进行访问。

### 删除事物

1. 在 AWS Explorer 中，展开 IoT 服务部分。
2. 在事物子部分中，找到您要删除的事物。
3. 打开该事物的上下文菜单（右键单击），然后从上下文菜单中选择删除事物。
4. 完成此步骤后，事物子部分中不再显示已删除的事物。

#### Note

注意：您只能删除没有附加证书的事物。

# AWS IoT 证书

证书是用于在您的 AWS IoT 服务与设备之间创建安全连接的常用方法。X.509 证书属于数字证书，它按照 X.509 公有密钥基础设施标准将公有密钥与证书所含的身份相关联。有关 AWS IoT 证书的更多信息，请参阅开发人员指南 [身份验证 \(IoT\)](#)。

## 管理证书

VS Code 工具包提供了多种直接从 AWS Explorer 管理 AWS IoT 证书的方法。

- [Create a certificate](#)
- [Change a certificate status](#)
- [Attach a policy to a certificate](#)
- [Delete a certificate](#)

## 创建 AWS IoT 证书

X.509 证书用于连接到您的 AWS IoT 实例。

1. 在 AWS Explorer 中，展开 IoT 服务部分，打开证书的上下文菜单（右键单击）。
2. 从上下文菜单中选择创建证书，以打开对话框。
3. 在本地文件系统中选择一个目录，以要保存 RSA 密钥对和 X.509 证书。

### Note

- 默认文件名包含证书 ID 作为前缀。
- 只有 X.509 证书通过 AWS IoT 服务存储在您的 AWS 账户中。
- 您的 RSA 密钥对只能颁发一次，请在出现提示时将其保存到文件系统的安全位置。
- 如果证书或密钥对此时无法保存到您的文件系统，则 AWS Toolkit 会从您的 AWS 账户中删除证书。

## 修改证书状态

单个证书的状态显示在 AWS Explorer 中其 ID 的旁边，可以设置为“活跃”“不活跃”或“已撤销”。

**Note**

- 您的证书需要处于 active (活动) 状态才能用来将设备连接到 AWS IoT 服务。
- 处于不活跃状态的证书可以激活，无论该证书是之前已停用还是默认处于不活跃状态。
- Revoked (已撤销) 的证书无法重新激活。

1. 在 AWS Explorer 中，展开 IoT 服务部分。
  2. 在证书子部分中，找到要修改的证书。
  3. 打开证书的上下文菜单 (右键单击)，该菜单显示对该证书可用的状态更改选项。
- 如果证书的状态为 inactive (不活动)，选择 activate (激活) 可将其状态更改为 active (活动)。
  - 如果证书的状态为 active (活动)，选择 deactivate (停用) 可将其状态更改为 inactive (不活动)。
  - 如果证书的状态为 active (活动) 或 inactive (不活动)，选择 revoke (撤销) 可将其状态更改为 revoked (已撤销)。

**Note**

如果您选择了附加到事物子部分所显示事物的证书，则所有这些状态更改操作也都可用。

### 将 IoT 策略附加到证书

1. 在 AWS Explorer 中，展开 IoT 服务部分。
2. 在证书子部分中，找到要修改的证书。
3. 打开证书的上下文菜单 (右键单击)，然后从上下文菜单中附加策略以打开输入选择器，其中提供了可用策略的列表。
4. 选择要附加到证书的策略。
5. 完成此步骤后，您选择的策略将作为子菜单项添加到证书中。

### 将 IoT 策略与证书分离

1. 在 AWS Explorer 中，展开 IoT 服务部分。

2. 在证书子部分中，找到要修改的证书。
3. 展开证书并找到要分离的策略。
4. 打开策略的上下文菜单（右键单击），然后从上下文菜单中选择分离。
5. 完成此步骤后，该策略将不再是可从您的证书访问的项目，但您可以从策略子部分访问它。

## 删除证书

1. 在 AWS Explorer 中，展开 IoT 服务标题。
2. 在证书子部分中，找到要删除的证书。
3. 打开证书的上下文菜单（右键单击），然后从上下文菜单中选择删除证书。

### Note

无法删除已经附加到某个事物或者处于活动状态的证书。您可以删除附加了策略的证书。

## AWS IoT 策略

AWS IoT Core 策略通过 JSON 文档定义，其中每个文档包含一个或多个策略声明。策略定义了 AWS IoT、AWS 和您的设备如何相互交互。有关如何创建策略文档的更多信息，请参阅开发人员指南 [IoT 策略](#)。

### Note

命名策略采取了版本控制，因此您可以回滚它们。在 AWS Explorer 中，您的 IoT 策略列在 IoT 服务中的策略子部分下。您可以通过展开策略来查看策略版本。默认版本由星号表示。

## 管理策略

Toolkit for VS Code 为您提供了多种管理 AWS IoT 服务策略的方法。以下是直接在 VS Code 中通过 AWS Explorer 管理或修改策略的方法：

- [Create a policy](#)
- [Upload a new policy version](#)

- [Edit a policy version](#)
- [Change the policy version default](#)
- [Change the policy version default](#)

## 创建 AWS IoT 策略

### Note

您可以从 AWS Explorer 创建新策略，但定义该策略的 JSON 文档必须已经存在于您的文件系统中。

1. 在 AWS Explorer 中，展开 IoT 服务部分。
2. 打开策略子部分的上下文菜单（右键单击），然后选择从文档创建策略，以打开策略名称输入字段。
3. 输入名称并按照提示打开一个对话框，其中会要求您从文件系统中选择 JSON 文档。
4. 选择包含您的策略定义的 JSON 文件，完成此步骤后，该策略将在 AWS Explorer 中可用。

## 上传新的 AWS IoT 策略版本

您可以通过向策略上传 JSON 文档来创建该策略的新版本。

### Note

要使用 AWS Explorer 创建新版本，新的 JSON 文档必须存在于您的文件系统上。

1. 在 AWS Explorer 中，展开 IoT 服务部分。
2. 展开策略子部分以查看您的 AWS IoT 策略。
3. 打开要更新的策略的上下文菜单（右键单击），然后选择从文档创建新版本。
4. 对话框打开时，选择包含策略定义更新的 JSON 文件。
5. 新版本可在 AWS Explorer 中从策略进行访问。

## 编辑 AWS IoT 策略版本

策略文档可以使用 VS Code 打开和编辑。完成文档编辑后，请将其保存到文件系统中。然后，从 AWS Explorer 将其上传到您的 AWS IoT 服务。

1. 在 AWS Explorer 中，展开 IoT 服务部分。
2. 展开策略子部分并找到要更新的策略。选择从文档创建策略，以打开策略名称输入字段。
3. 展开要更新的策略，然后打开要编辑的策略版本的上下文菜单（右键单击）。
4. 从上下文菜单中选择查看，以在 VS Code 中打开相应策略版本
5. 打开策略文档后，进行所需的更改并保存。

### Note

此时，您对策略所做的更改仅会保存到本地文件系统中。要更新版本并使用 AWS Explorer 对其进行跟踪，请重复 [Upload a new policy version](#) 流程中所述的步骤。

## 选择新的默认策略版本

1. 在 AWS Explorer 中，展开 IoT 服务部分。
2. 展开策略子部分并找到要更新的策略。
3. 展开要更新的策略，然后打开要设置的策略版本的上下文菜单（右键单击）并选择设为默认。
4. 此步骤完成后，您选择的新默认版本旁边会显示一个星号。

## 删除策略

### Note

在删除某个策略或策略版本之前，需要满足一些条件。

- 您无法删除已附加到证书的策略。
- 如果策略有任何非默认版本，则无法删除该策略。
- 只有选择了新的默认版本或删除了整个策略，您才能删除策略的默认版本。
- 在删除整个策略之前，必须先删除该策略的所有非默认版本。

1. 在 AWS Explorer 中，展开 IoT 服务部分。
2. 展开策略子部分并找到要更新的策略。
3. 展开要更新的策略，然后打开要删除的策略版本的上下文菜单（右键单击）并选择删除。
4. 删除版本后，它在 Explorer 中将不再可见。
5. 如果策略只剩下默认版本，请打开父策略的上下文菜单（右键单击），然后选择删除以将其删除。

## AWS Lambda 函数

AWS Toolkit for Visual Studio Code 为 AWS Lambda 函数提供全面支持，使您能够直接从 VS Code 构建、测试和部署。

Lambda 是一项完全托管的、事件驱动型计算服务，可自动运行代码以响应来自 200 多个 AWS 服务和软件即服务 (SaaS) 应用程序的事件。有关 AWS Lambda 服务的详细信息，请参阅《[AWS Lambda 开发人员指南](#)》。

以下主题介绍了如何在 AWS Lambda 中使用 AWS Toolkit for Visual Studio Code。

### 主题

- [使用 AWS Lambda 函数](#)
- [AWS Lambda console 到 IDE](#)
- [AWS Lambda 有 LocalStack 支撑](#)
- [AWS Lambda 远程调试](#)

## 使用 AWS Lambda 函数

AWS Toolkit for Visual Studio Code 允许您在本地 VS Code 环境中使用您的 AWS Lambda 函数。使用该 AWS 工具包，您无需离开 IDE 即可创建、编辑、测试、调试和部署您的 Lambda 函数。有关该 AWS Lambda 服务的详细信息，请参阅[AWS Lambda](#)开发人员指南。

以下各节介绍了如何开始在 AWS Toolkit for Visual Studio Code 中使用 Lambda 函数。

### Note

如果您已经使用创建了 Lambda 函数 AWS 管理控制台，则可以从工具包中调用它们。此外，您可以从在 VS Code 中打开 Lambda 函数 AWS Lambda console，有关更多信息，请参阅本

用户[AWS Lambda console 到 IDE](#)指南中的主题。要在 VS Code 中创建新的 Lambda 函数，请按照本用户指南中[创建新的无服务器应用程序（本地）](#)主题中概述的步骤进行操作。

## 先决条件

必须满足以下条件才能使用 AWS 工具包中的 AWS Lambda 服务。

- 已安装最新版本 AWS Toolkit for Visual Studio Code 的，并使用您的 AWS 凭据进行设置。
- 您的 AWS Identity and Access Management (IAM) 托管权限和策略已配置为与该 AWS Lambda 服务配合使用。[有关如何配置权限和创建兼容的 AWS 托管策略的详细信息，请参阅《AWS Identity and Access Management AWS Lambda 开发人员指南》中的 AWS Lambda 主题。](#)
- 您已有 AWS Lambda 函数或熟悉如何创建函数。有关如何创建 Lambda 函数的说明，请参阅《AWS Lambda 开发人员指南》中的[创建第一个 Lambda 函数](#)主题。

## 调用 Lambda 函数

要从您的 AWS 账户在 VS Code 中调用 Lambda 函数，请完成以下步骤。

1. 从中 AWS Toolkit for Visual Studio Code，展开 AWS 浏览器。
2. 在 AWS 资源管理器中，展开 Lambda 以查看您的 Lambda 资源。
3. 打开要调用的 Lambda 函数的上下文菜单（右键单击），然后选择在云中调用或选择在云中调用图标以在 VS Code 中打开远程调用配置菜单。
4. 在远程调用配置菜单中，指定有效载荷设置并添加事件所需的任何其他信息。

### Note

在 AWS 资源管理器中选择“云端调用”后，第一个调用进程可能会立即开始运行。输出将显示在 VS Code 终端的输出选项卡中。

5. 选择远程调用按钮来调用函数。输出将显示在 VS Code 终端的输出选项卡中。

## 删除 Lambda 函数

要删除 Lambda 函数，请完成以下过程。

**⚠ Warning**

请勿使用此过程删除与 [CloudFormation](#) 关联的 Lambda 函数。这些函数必须通过 CloudFormation 堆栈删除。

1. 从中 AWS Toolkit for Visual Studio Code，展开 AWS 浏览器。
2. 在 AWS 资源管理器中，展开 Lambda 以查看您的 Lambda 资源。
3. 右键单击要删除的 Lambda 函数，然后选择删除。
4. 出现提示时，确认您要删除函数。

删除该函数后，它将不再在 AWS 资源管理器中列出。

## 下载 Lambda 函数

您可以将远程 Lambda 函数的代码下载到 VS Code 工作区中进行编辑和调试。


**i Note**

要下载你的 Lambda 函数，你必须在带有可访问文件夹的 VS Code 工作区中工作，而且该 AWS 工具包仅支持使用 Node.js 和 Python 运行时的 Lambda 函数的此功能。

1. 从中 AWS Toolkit for Visual Studio Code，展开 AWS 浏览器。
2. 在 AWS 资源管理器中，展开 Lambda 以查看您的 Lambda 资源。
3. 右键单击要下载的 Lambda 函数，然后选择下载。
4. 您的 Lambda 函数将在 VS Code 编辑器中打开，并在下载完成后显示在 AWS 资源管理器中。该 AWS 工具包还在 VS Code 运行面板中创建启动配置，允许您使用在本地运行和调试 Lambda 函数。AWS Serverless Application Model 有关使用的更多信息 AWS SAM，请参阅 [the section called “从模板 \(本地\) 运行和调试无服务器应用程序”](#)。

## 为新的 Lambda 函数部署更新

您可以从本地计算机上未指定的临时位置部署新 Lambda 函数的更新。

 Note

当您的 Lambda 文件有未部署的更改时，VS Code 编辑器和 AWS 各区服务浏览器中修改后的文件旁边的 M 图标会通知您。

### 通过 VS Code 编辑器部署


1. 在 VS Code 编辑器中打开 Lambda 函数中的一个文件，然后对该文件进行更改。
2. 从 VS Code 主菜单中手动保存或按 **option+s** ( Mac ) **ctrl+s** ( Windows )。
3. VS Code 会自动提示您将更改部署到云中，选择部署按钮以确认部署。
4. VS Code 会实时更新部署状态，并在该过程完成时通知您。

### 从 AWS 资源管理器部署

1. 在 VS Code 编辑器中打开 Lambda 函数中的一个文件，然后对该文件进行更改。
2. 在 AWS 工具包中，展开 AWS 资源管理器。
3. 在 AWS 资源管理器中，使用要为其部署更改的 Lambda 函数扩展 AWS 区域。
4. 从该 AWS 区域展开 Lambda，然后浏览要为其部署更改的函数。
5. 在函数旁边的快捷菜单中，选择保存并部署代码图标。
6. VS Code 会实时更新部署状态，并在该过程完成时通知您。

### 上传现有 Lambda 函数的更新

以下过程说明了如何上传对现有 Lambda 函数所做的本地更改。此功能支持使用任何 Lambda 支持的运行时进行上传。

 Warning

在上传 Lambda 函数之前，请注意以下几点：

- 以这种方式更新代码不会使用 AWS SAM CLI 进行部署或创建 CloudFormation 堆栈
- 该 AWS 工具包不验证代码。在将任何更改上传到云中之前，请验证代码并测试函数。

## 上传 Zip 格式归档

1. 从中 AWS Toolkit for Visual Studio Code ，展开 AWS 浏览器。
2. 在 AWS 资源管理器中，展开 Lambda 以查看您的 Lambda 资源。
3. 右键单击要将更改上传到的 Lambda 函数，然后选择上传 Lambda... 以打开选择上传类型菜单。
4. 选择 ZIP 归档，在本地目录中找到 ZIP Archive。
5. 出现提示时，确认上传以开始上传所选 ZIP Archive。
6. 上传状态显示在 VS Code 中，当上传过程完成时，系统会通知您。

## 上传不含构建的目录

1. 从中 AWS Toolkit for Visual Studio Code ，展开 AWS 浏览器。
2. 在 AWS 资源管理器中，展开 Lambda 以查看您的 Lambda 资源。
3. 右键单击要将更改上传到的 Lambda 函数，然后选择上传 Lambda... 以打开选择上传类型菜单。
4. 选择目录以进入构建目录屏幕。
5. 在构建目录屏幕上，选择否以选择要上传的本地目录。
6. 出现提示时，确认上传以上传所选目录。
7. 上传状态显示在 VS Code 中，当上传过程完成时，系统会通知您。

## 上传包含构建的目录

### Note

请注意以下事项：

- 此过程需要 AWS Serverless Application Model CLI。
- AWS Toolkit 会在上传之前通知您无法检测到匹配的处理程序。
- 要更改附加到 Lambda 函数的处理程序，请使用 AWS Lambda console 或。 AWS Command Line Interface

1. 从中 AWS Toolkit for Visual Studio Code ，展开 AWS 浏览器。
2. 在 AWS 资源管理器中，展开 Lambda 以查看您的 Lambda 资源。
3. 右键单击要将更改上传到的 Lambda 函数，然后选择上传 Lambda... 以打开选择上传类型菜单。

4. 选择目录以进入构建目录屏幕。
5. 在构建目录屏幕上，选择是，然后选择要上传的本地目录。
6. 出现提示时，确认上传以开始构建和上传所选目录。
7. 上传状态显示在 VS Code 中，当上传过程完成时，系统会通知您。

## 将您的 Lambda 函数转换为项目 AWS SAM

要将您的 Lambda 函数转换为 AWS SAM 堆栈，请完成以下步骤。

### Warning

目前，将 Lambda 函数转换为 AWS SAM 项目时，仅支持部分资源。要在转换后找到任何缺失的资源，请查看 Lambda 控制台并将其手动添加到您的 AWS SAM 模板中。有关支持和不支持的资源的更多详细信息，请参阅《AWS CloudFormation 开发人员指南》中的[资源类型支持](#)主题。

1. 在 AWS 工具包中，展开 AWS 资源管理器。
2. 在 AWS 资源管理器中，使用要转换为项目的 Lambda 函数扩展 AWS 区域。AWS SAM
3. AWS 在该区域中，展开 Lambda，然后浏览要转换为堆栈的函数。AWS SAM
4. 在 Lambda 函数旁边的快捷菜单中，选择转换为 SAM 应用程序图标以浏览您的本地文件系统并作为新 AWS SAM 项目指定位置。
5. 指定位置后，AWS 工具包开始将您的 Lambda 函数转换为 AWS SAM 项目，VS Code 会提供有关流程状态的更新。

### Note

此过程可能需要几分钟时间。

6. 当 VS Code 显示提示时，输入堆栈名称，然后按 **Enter** 键继续。
7. VS Code 会继续更新您的项目状态，然后在流程完成时通知您，并将您的新 AWS SAM 项目作为 VS Code 工作区打开。

## AWS Lambda console 到 IDE

从 AWS Lambda console 到 IDE 功能允许您将 AWS Lambda 函数从下载到 VS Code AWS Lambda console 中。在 VS Code 中使用您的 Lambda 函数可以访问其他本地开发选项，例如 AWS Serverless Application Model (SAM) 和 AWS Cloud Development Kit (AWS CDK)。

有关的更多信息 AWS Lambda，请参阅《[AWS Lambda 开发人员指南](#)》。要开始使用 AWS 工具包中的 Lambda 函数，请参阅本用户指南中的[使用 AWS Lambda 函数](#)主题。以下各节介绍了如何将工作流从 Lambda 控制台移至 VS Code。有关将 Lambda 函数从 Lambda 控制台移至 VS Code 的详细信息（包括如何开始使用 Lambda 控制台），请参阅《AWS Lambda 开发人员指南》中的[使用 VS Code 在本地开发 Lambda 函数](#)主题。

### 从控制台转向本地开发

要在 VS Code 中从 Lambda 控制台打开 Lambda 函数，请完成以下步骤：

1. 在 Web 浏览器中打开 [Lambda 控制台](#)。
2. 在 Lambda 控制台中，选择要在 VS Code 中打开的函数。
3. 在函数视图中，导航到代码源选项卡。
4. 在代码源选项卡中，选择在 VS Code 中打开。

### 在 VS Code 中使用 Lambda 函数

当 Lambda 函数通过 Lambda 控制台在 VS Code 中打开时：

- VS Code 会在本地计算机上自动启动。
- Lambda 函数作为 VS Code 工作区打开。
- Lambda handler file 在 VS Code 编辑器中打开。

#### Note

如果工作区中没有正确配置的 handler file，则不会在 VS Code 编辑器中打开任何文件。

通过 Lambda 控制台在 VS Code 中打开 Lambda 函数后，您就可以访问所有现有 AWS Toolkit Lambda 功能，包括使用完全语言支持编辑函数代码、本地测试、远程调试、部署支持和依赖项管理。

有关该 AWS 工具包支持的 Lambda 功能的更多信息，请参阅本用户指南中的[AWS Lambda](#)服务目录。

## AWS Lambda 有 LocalStack 支撑

在中的 LocalStack 支持下构建、测试和调试您的无服务器应用程序。AWS Toolkit for Visual Studio Code LocalStack 是一款 AWS 云端仿真器，允许对无服务器应用程序进行本地测试。

有关的其他信息 AWS Lambda，请参阅《[AWS Lambda](#)开发人员指南》。要了解更多信息 LocalStack，请访问他们的网站[LocalStack](#)。

### 先决条件

以下是在 VS Code LocalStack 中使用的先决条件。

#### Note

LocalStack CLI 是在安装过程中安装的，但如果您更喜欢其他版本的 LocalStack CLI，则最低要求版本为 4.8.0。

- 需要一个 LocalStack Web 应用程序帐户才能访问免费和付费 LocalStack 套餐的所有可用功能。LocalStack 社区版无需账号即可使用。
- 需要 LocalStack 在 VS Code 中使用 Docker。有关 Docker LocalStack 要求的更多信息，请参阅文档中的 LocalStack [Docker 镜像](#)主题。LocalStack
- 推荐：AWS Command Line Interface (AWS CLI) 可帮助您在模拟云环境中使用服务。

### 正在安装 LocalStack

要安装 LocalStack 免费和付费分层版本，请完成以下步骤。

#### Note

有关如何设置 LocalStack 社区版的说明，请参阅本主题设置 LocalStack 部分中的 LocalStack 社区内容。

1. 在 AWS 工具包中，展开“应用程序生成器”资源管理器。

2. 选择打开演练按钮，在 VS Code 编辑器中打开开始构建应用程序演练选项卡。
3. 在演练中，选择“安装” LocalStack 以启动 VS Code 中的 LocalStack 安装过程。

## 设置 LocalStack

安装 VS Code LocalStack 扩展程序后，在需要安装时，您可能会看到以下指示器之一：

- 默认情况下，在 IDE 左下角的 VS Code 状态栏中，LocalStack 状态为红色。
- VS Code 会提示您进行设置 LocalStack。

有两种类型的设置和配置 LocalStack，具体取决于 LocalStack 您使用的是哪个版本。以下选项卡部分描述了每个 LocalStack 设置过程。

### Note

LocalStack 的免费和付费套餐版本需要身份验证令牌。LocalStack 有关 LocalStack 定价的具体信息，请参阅他们的 [“选择您的套餐”](#) 定价指南。

## LocalStack 免费和付费等级

有两种设置方式 LocalStack。

- 从 VS Code 安装 LocalStack 到入门提示中，选择设置按钮。
- 从 VS Code 状态栏中，选择 LocalStack 状态图标 LocalStack 以打开安装程序开始提示符，然后选择设置按钮。

在设置过程中，系统将执行以下步骤：

1. 安装 C LocalStack LI。
2. 检查您是否有 LocalStack 账户。
3. 如果您有 LocalStack 帐户，系统会引导您在默认的 Web 浏览器中完成身份验证过程。同样，如果您没有 LocalStack 帐户，系统会在身份验证过程之前指导您完成帐户设置。

设置完成后 LocalStack，VS Code LocalStack 状态栏中的状态会更新。

**Note**

如果您尚未为创建 AWS 配置文件 LocalStack，则在 LocalStack 设置过程中会自动为您创建一个新的配置文件。

## LocalStack 社区

的 LocalStack 社区版可以免费使用，不需要您注册帐户，它使用不需要许可证的 Docker 镜像运行。有关 LocalStack 社区版的更多详细信息，请参阅[LocalStack 社区图片](#)文档。以下各节介绍在 VS Code 中使用 LocalStack 社区版所需的先决条件和基本设置。

### 启动新实例

要启动新的 LocalStack 社区实例，请完成以下步骤。

**Note**

以下示例在端口 4566 LocalStack 上启动一个容器实例。如果指定不同的端口值，则必须更新在配置 AWS CLI 和 AWS 工具包部分的过程中指定的端口值。

1. 从 VS Code 中打开 VS Code 终端 (按 **ctrl + ` (backtick)**)。
2. 在终端窗口中输入以下内容。

Mac :

```
docker run -d --name localstack_main \  
>> -p 4566:4566 \  
>> -v /var/run/docker.sock:/var/run/docker.sock \  
>> localstack/localstack
```

Windows :

```
docker run -d --name localstack_main \  
>> -p 4566:4566 \  
>> -v /var/run/docker.sock:/var/run/docker.sock \  
>> localstack/localstack
```

3. 当流程完成后，终端会更新您的 Docker 实例的状态。

的这个容器化实例 LocalStack 允许您访问您在下载过程中指定的 AWS 服务。

为 LocalStack 和 Docker 配置 CLI。

要在 Docker LocalStack 中配置要使用的 AWS CLI 和 AWS Toolkit，请完成以下步骤来设置新的配置文件：

1. 从 VS Code 中打开 VS Code 终端（按 **ctrl + `**（backtick））。
2. 在终端窗口中输入以下内容。

```
~/.aws/credentials
[localstack]
aws_access_key_id = test
aws_secret_access_key = test
~/.aws/config
[profile localstack]
region = us-east-1
output = json
endpoint_url = http://localhost:4566 [default localstack endpoint]
```

3. AWS Toolkit 会检测您的 LocalStack 个人资料并更新连接状态菜单。

设置完成后，从状态栏的 LocalStack 配置 AWS 文件部分选择您的个人资料后，您的 LocalStack 资源便会在资源 AWS 管理器中可见。此外，您可以在 VS Code 终端的“输出”选项卡中查看 LocalStack 日志。

## 从 LocalStack VS 代码开始

你可以开始 LocalStack 使用以下任何一种方法：

### LocalStack 从 VS 代码状态栏开始

1. 在 VS Code 中，导航到状态栏，然后选择 LocalStack “开始” 按钮启动 LocalStack。
2. 成功启动后，VS Code 状态栏会 LocalStack 更新。

### LocalStack 从 VS Code 命令面板开始

1. 在 VS Code 中，按下 **Cmd + Shift + P** (Mac) 或 **Control + Shift + P** (Windows) 打开命令面板。
2. 在命令面板的搜索栏中输入 **Start LocalStack**，并在结果列表中选中它。

3. 成功启动后，VS Code 状态栏会 LocalStack 更新。

### LocalStack 从 VS Code 终端开始

1. 从 VS Code 中打开 VS Code 终端 ( 按 **ctrl + ` (backtick)** )。
2. 在 VS Code 终端中输入 **localstack start** CLI 命令。
3. 成功启动后，VS Code 状态栏会 LocalStack 更新。

### 构建示例无服务器应用程序

要开始 LocalStack 在 VS Code 中使用，您需要一个示例无服务器应用程序。如果您的 AWS 账户中已有应用程序，则可以使用将其部署到本地，LocalStack 也可以使用 AWS Serverless Land 创建新应用程序。

如需了解在 AWS Toolkit 中使用 Serverless Land 创建应用程序的更多信息，请参阅本用户指南中的[使用 AWS Serverless Land](#) 主题。有关 Serverless Land 的详细信息，请参阅 [Serverless Land](#) Web 应用程序主登录页面。

### 使用测试和调试 Lambda 函数 LocalStack

在 LocalStack VS Code 扩展中测试和调试 Lambda 函数与使用部署到云端的 AWS 函数类似。主要区别在于，您的 AWS Toolkit 实例必须使用您的 LocalStack 账户进行身份验证才能部署和调试函数 LocalStack。

#### Note

本节中描述的测试和调试功能不适用于 LocalStack 社区版。

要 LocalStack 在 VS Code 中使用，请在 AWS 工具包中连接到您的 LocalStack 个人资料。当您的 LocalStack 个人资料处于活动状态时，VS Code 状态栏会显示 AWS：带复选标记的 profile: localstack ( 自定义端点 )。

有关在 AWS 工具包中使用 Lambda 函数的详细信息，请参阅本用户指南中的[使用 AWS Lambda 函数](#)主题。

## AWS Lambda 远程调试

AWS Toolkit for Visual Studio Code 使您可以直接在 VS Code 中调试在云中运行的 AWS Lambda 函数。通过 AWS Lambda 远程调试，您可以检查正在运行的函数、设置断点、检查变量和逐步调试，而无需修改其现有开发工作流程。

以下各部分介绍如何在 AWS Toolkit for Visual Studio Code 中使用 Lambda 远程调试。

### Lambda 远程调试的工作原理

该 AWS 工具包支持远程调试，方法是使用额外的 Lambda 调试层临时修改您的 Lambda 函数，并将 Lambda 调用超时限制延长至 900 秒。通过 AWS IoT 安全隧道，您的本地调试器会与 Lambda 运行时环境之间建立一条安全连接。通过这条连接，您可以使用本地代码中的断点，对远程运行的函数进行逐步调试。调试会话结束后，所有临时修改都会被自动恢复到原始设置。

### 开始使用

#### 支持的运行时

Lambda 远程调试支持以下运行时：

- Python ( Amazon Linux 2023 )
- Java
- Typescript/JavaScript/Node.js ( 亚马逊 Linux 2023 )

#### Note

Lambda 远程调试不支持 Lambda 托管实例和 OCI 图像函数类型。

#### 先决条件

在您开始之前，必须满足以下先决条件。

- 您必须在 AWS Toolkit 中配置有效的 AWS 凭据。有关安装 AWS Toolkit 和配置凭据的更多详细信息，请参阅本用户指南中的[入门](#)主题。
- Lambda 函数已部署到您的 AWS 账户。有关部署 Lambda 函数的详细信息，请参阅《AWS Lambda 开发人员指南》中的[创建第一个 Lambda 函数](#)主题。

- 您必须拥有相应的 AWS Identity and Access Management (IAM) 策略和权限才能调试您的函数。有关 Lambda 权限的更多信息，请参阅《AWS Lambda开发人员指南》中[适用于 AWS Lambda的 AWS 托管式策略](#)主题。以下是一个示例策略，包含在 AWS Toolkit 中使用 Lambda 远程调试所需的最低权限。

#### Note

远程调试是通过 AWS IoT 安全隧道启用的。这可让您的本地调试器与 Lambda 运行时环境建立安全连接。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:ListFunctions",
        "lambda:GetFunction",
        "lambda:GetFunctionConfiguration",
        "lambda:GetLayerVersion",
        "lambda:UpdateFunctionConfiguration",
        "lambda:InvokeFunction",
        "lambda:PublishVersion",
        "lambda>DeleteFunction",
        "iot:OpenTunnel",
        "iot:RotateTunnelAccessToken",
        "iot:ListTunnels"
      ],
      "Resource": "*"
    }
  ]
}
```

## 访问 Lambda 远程调试

在 AWS 工具包中访问 Lambda 远程调试有两种主要路径：AWS 资源管理器或应用程序生成器资源管理器。在 AWS 资源管理器中，您可以通过节点访问 Lambda 远程调试。AWS Lambda 在 Application Builder 浏览器中，您可以通过本地 AWS SAM 项目访问 Lambda 远程调试。

从资源管理器访问 Lambda 远程调试 AWS

1. 在 VS Code 中，打开 AWS 工具包扩展。
2. 在 AWS 工具包中，展开 AWS 资源管理器。
3. 在浏览器中，展开 Lambda 节点。
4. 找到您想调试的函数，然后在右键菜单中选择远程调用图标，以打开远程调用配置界面。

从 Application Builder 浏览器访问 Lambda 远程调试。

1. 在 VS Code 中，打开 AWS 工具包扩展。
2. 在 AWS 工具包中，展开应用程序生成器资源管理器。
3. 在浏览器中，展开包含您想调试的 Lambda 项目的 AWS SAM 项目。
4. 展开您想调试的已部署 Lambda 函数。
5. 找到该函数的 remote 选项，然后在右键菜单中选择远程调用图标，以打开远程调用配置界面。

## 使用 Lambda 远程调试

以下各部分介绍如何在 AWS Toolkit for Visual Studio Code 中使用 Lambda 远程调试。

### Note

Lambda 函数最多支持 5 个层，并且函数代码和所有附加层的总大小不得超过 250MB。Lambda 远程调试需要至少保留 1 个空闲层才能运行。

## 设置调试会话

在开始之前，请按照以下步骤配置您的调试会话。

1. 通过完成前一节中的“从资源管理器访问 Lambda 远程调试”或“从应用程序生成器资源管理器访问 Lambda 远程调试”过程，打开远程调用配置菜单。

2. 在远程调用配置菜单中，选中远程调试复选框以显示远程调试属性。
3. 指定您的本地处理程序文件的本地根路径。

#### Note

本地根路径指的是与您已部署的 Lambda 函数相对应的本地源代码位置。如果您是从 Application Builder 浏览器中的已部署函数开始操作，本地根路径会自动检测。如果您在本地没有保存源代码，选择下载远程代码按钮来获取 Lambda 函数的源代码。这将在 VS Code 编辑器中打开您的 handler file。

4. 在有效载荷部分，指定您的测试事件数据的来源。

## 设置断点并调试

按以下步骤设置断点并开始调试。

1. 在 VS Code 编辑器中的 handler file 文件，单击行号旁的边距区域，即可在需要暂停调试的位置设置断点。
2. 设置好断点后，返回远程调用配置菜单，确认您的设置是正确的，然后选择远程调用按钮开始调试。
3. AWS Toolkit 使用调试功能更新您的 Lambda 函数，为调试会话建立安全隧道，使用指定的有效负载调用您的函数，然后在遇到断点时暂停该进程。
4. 在断点暂停时，使用 RUN AND DEBUG 面板查看 VARIABLES、CALL STACK 和 BREAKPOINTS。

## 更新并测试您的函数

如需修改代码并通过快速部署测试变更，请按以下步骤执行操作。

1. 在调试会话仍处于活动状态时，在 VS Code 编辑器中修改您的 handler file。
2. 保存您的更改 ( **Command+S on macOS** , **Ctrl+S on Windows** )
3. 出现提示时，确认您希望继续部署这些更改。该 AWS 工具包将使用修改后的代码更新您的 Lambda 函数。
4. 设置新的断点，然后再次选择远程调用按钮，即可继续调试并测试您的变更。

**Note**

或者，您可以在 VS Code 调试控制中取消选中附加调试器选项，然后选择远程调用按钮，让函数在不调试的情况下运行。

## 结束调试会话

以下任一操作都会结束您的远程调试会话，并从项目中移除调试层。

- 在远程调用配置界面选择移除调试设置选项。
- 在 VS Code 调试控制栏中选择断开连接图标。
- 在 VS Code 编辑器中关闭 handler file。

**Note**

记录以下内容：

- Lambda 调试层会在 60 秒处于活动状态后自动移除。这个计时会在上一次调用完成时开始。
- 如果您在调试过程中对 infrastructure-as-code (IaC) 托管 (AWS SAM,, AWS CDK, Terraform) 函数进行了代码更改，请将其保存到本地项目中，并考虑更新源代码控制存储库。未保存的更改会在 IaC 函数重新部署时被覆盖。
- 如果您仅出于调试目的进行了临时更改，则可能需要从源代码控制中重新部署函数，以确保其与生产环境中的代码保持一致。

## 使用源映 TypeScript 射调试 Lambda 函数

以下各节介绍如何使用源映射调试 TypeScript Lambda 函数。

### 先决条件

要调试您的 TypeScript Lambda 函数，必须满足以下先决条件。

- TypeScript 必须在启用源映射选项的情况下进行编译。有关更多信息，请参阅 VS Code 文档中的[JavaScript 源映射支持](#)主题。

- 不支持内联源地图。必须使用单独 `.js.map` 的文件来存储源地图。

## 配置

要为 AWS 工具包中的 Lambda 函数配置 TypeScript lambda 远程调试，请完成以下步骤。

1. 在 AWS 工具包中，展开 AWS 资源管理器。
2. 在浏览器中，展开 Lambda 节点。
3. 导航到要为其配置的功能 TypeScript，然后从上下文菜单中选择“远程调用”图标以打开“远程调用”配置屏幕。
4. 选中远程调试复选框以启用远程调试。
5. 通过指向包含 TypeScript handler file 的目录来配置您的本地根路径。

### Note

TypeScript handler file 是您设置调试断点的位置。

6. 展开远程调试附加配置设置。
7. 选中源映射复选框来启用源映射。
8. 将输出文件字段设置为您的 Lambda 函数副本所在的本地目录。

### Example

如果 `app.js` 和 `app.map` 位于 `.aws-sam/build/HelloWorldFunction` 中，则将输出文件位置为 `/Users/user/project/aws-sam/build/HelloWorldFunction/*`。

### Note

输出文件路径应为绝对路径。

对于 AWS SAM 和 AWS CDK 项目，该 AWS 工具包支持自动源映射检测。如果将这些项目的 `Output files` 字段留空，则该工具包将自动尝试检测源地图的位置。

9. 如果您对设置感到满意，请选择“远程调用”按钮开始调试您的 TypeScript 函数。

## 故障排除与高级使用案例

如果您的调试会话失败，请按以下步骤开始故障排除：

1. 将 AWS 工具包更新到最新版本。
2. 关闭远程调用配置 Web 视图并重新打开，以刷新 Web 视图。
3. 完全关闭 VS Code 后再重新打开它。
4. 打开 VS Code 命令面板，输入命令 **AWS: Reset Lambda Remote Debugging Snapshot**，在结果中选择它，以重置 Lambda 远程调试快照。
5. 如果您无法解决问题，请向 Issues 提交[AWS Toolkit for Visual Studio Code GitHub问题](#)。

### 高级使用案例：代码签名配置

远程调试需要向您的 Lambda 函数附加调试层。如果您的函数启用并强制执行了代码签名配置，则 AWS Toolkit 无法自动将调试层附加到您的函数。

您可以通过以下两种方式解决代码签名配置问题。

- 暂时移除代码签名。
- 使用已签名的调试层。

暂时移除代码签名。

提供设置 `UntrustedArtifactOnDeployment` : Warn 更新代码签名配置，在调试过程结束后再将其重置回 `Enforced`。

有关更多信息，请参阅 AWS Lambda API [UpdateCodeSigningConfig](#) 参考中的参考资料。

使用已签名的调试层。

1. 在 AWS 工具包的 Lambda 远程调试中，展开远程调试其他配置部分。
2. 在远程调试附加配置部分中，从层覆盖字段复制您所在区域的层 ARN。
3. 从中 AWS CLI，使用以下命令下载图层版本 `aws lambda get-layer-version-by-arn --arn layer-arn`，用您的图层 ARN 替换 `layer-arn`。有关如何下载已签名的调试层的详细说明，请参阅《AWS CLI 命令参考》中的 [get-layer-version-by-arn](#) 参考。
4. 使用您的代码签名配置对该层进行签名，并将其发布到您的账户。有关签名和发布指南，请参阅《AWS Serverless Application Model 开发人员指南》中的[为您的 AWS SAM 应用程序设置代码签名](#)主题。
5. 在已签名的层成功发布到您的账户后，返回 Lambda 远程调试的远程调试附加配置部分，然后层覆盖字段中输入新的层 ARN。完成后，Lambda 远程调试将使用您签名的调试层，而不再使用默认层。

## 高级用例：使用 SnapStart 或预配置并发调试函数

对于配置 SnapStart 或预配置并发的 Lambda 函数，发布新版本所需的时间要长得多。为了加快调试工作流程，您可以将 Lambda 远程调试配置为仅更新函数的 \$LATEST 版本，而不是发布新版本。

1. 在远程调用配置屏幕上，展开远程调试其他配置设置。
2. 取消选择“发布版本”选项。
3. 现在，该 AWS 工具包将仅更新您的函数 \$LATEST 版本并使用它进行调试。

### Note

使用 \$LATEST 版本进行调试的副作用是，您应避免其他可能调用您的 \$LATEST 版本的流量，以确保调试环境不受干扰。

## 支持的区域

当某个区域不支持远程调试时，会出现以下错误。

```
Region ${region} doesn't support remote debugging yet
```

以下是受支持的区域列表。

- ap-east-1
- ap-northeast-1
- ap-northeast-2
- ap-south-1
- ap-southeast-1
- ap-southeast-2
- ca-central-1
- eu-central-1
- eu-north-1
- eu-west-1
- eu-west-2
- eu-west-3

- me-central-1
- me-south-1
- sa-east-1
- us-east-1
- us-east-2
- us-west-1
- us-west-2

## Lambda RequestEntityTooLargeException

Lambda 函数最多支持 5 个层，并且函数代码和所有附加层的总大小不得超过 250MB。远程调试层的大小约为 40MB。如果您的函数包较大或已附加多个层，这可能会导致总大小超出限制。有关更多详细信息，请参阅《AWS Lambda 开发者指南》中的 [Lambda: InvalidParameterValueException 或 RequestEntityTooLargeException](#) 主题部分。

以下列表介绍了排查并解决此错误的一些方法。

- 减小函数大小：优化函数的代码，移除不必要的依赖关系。
- 移除未使用的层：在调试期间临时移除非必要层。
- 使用外部依赖项：将较大的依赖项移至外部存储，例如 Amazon S3，并在运行时加载它们。

## Java 调试故障排

要调试 Java Lambda 函数，您必须在本地安装与您的 Lambda 函数的运行时版本相匹配的 Java 版本。

例如，在调试 Java 25 函数时，必须在运行 AWS 工具包的本地环境中安装 Java 25。如果您尝试在本地安装了 Java 21 或更早版本的情况下调试 Java 25 函数，则远程调试将无法在您设置的断点处停止。

在开始调试会话之前，请确保您的本地 Java 版本与 Lambda 函数的运行时版本相匹配。

## 已超出 IoT 安全隧道配额

以下是在 Lambda 远程调试中达到 AWS IoT 安全隧道连接的每日限制时发生的隧道配额超出错误的示例。

```
Error creating/reusing tunnel: LimitExceededException: Exceeded quota of Lambda debugging tunnels
```

AWS IoT 安全隧道连接具有以下配额：

- 免费层的 IoT 安全隧道每天最多分配 10 个连接。
- 每个隧道支持一个 VS Code 实例，最长 12 小时。
- 配额适用于每个 AWS 账户、每天。

如果您遇到 AWS IoT 安全隧道错误，请等待每日配额重置或联系 AWS 支持部门申请提高配额限制。有关 AWS 支持联系人信息，请访问[AWS 支持联系人门户](#)。有关 AWS IoT 安全隧道的详细信息，请参阅《开发人员指南》中的[AWS IoT 安全隧道](#)主题。AWS IoT

## Toolkit for VS Code 中的 Amazon Redshift

Amazon Redshift 是云中一种完全托管的 PB 级数据仓库服务。有关 Amazon Redshift 服务的详细信息，请参阅《[Amazon Redshift 用户指南](#)》目录。

以下主题介绍了如何通过 AWS Toolkit for Visual Studio Code 使用 Amazon Redshift。

主题

- [通过 Toolkit for VS Code 使用 Amazon Redshift](#)

## 通过 Toolkit for VS Code 使用 Amazon Redshift

以下各节介绍了如何开始通过 AWS Toolkit for Visual Studio Code 使用 Amazon Redshift。

有关 Amazon Redshift 服务的详细信息，请参阅《[Amazon Redshift 用户指南](#)》主题。

入门

要开始通过 AWS Toolkit for Visual Studio Code 使用 Amazon Redshift，必须满足以下要求。

1. 您已通过 Toolkit 连接到您的 AWS 账户。有关通过 Toolkit 连接到 AWS 账户的更多信息，请参阅本用户指南中的[连接到 AWS](#)主题。
2. 您已经创建了预置数据仓库或无服务器数据仓库。

如果您尚未创建 Amazon Redshift Serverless 或 Amazon Redshift 预置集群，以下流程介绍了如何通过 AWS 控制台使用示例数据集创建数据仓库。

## 创建预置数据仓库

有关创建 Amazon Redshift 预置集群数据仓库的更多详细信息，请参阅《Amazon Redshift 入门用户指南》中的[创建示例 Amazon Redshift 集群](#)主题。

1. 在您的首选互联网浏览器中，登录 AWS 管理控制台，然后打开 Amazon Redshift 控制台，网址为：<https://console.aws.amazon.com/redshift/>。
2. 在 Amazon Redshift 控制台中，选择预置集群控制面板。
3. 在预置集群控制面板中，选择创建集群按钮，以打开创建集群窗格。
4. 填写集群配置部分中的必填字段。
5. 在示例数据部分，选中加载示例数据框，以将示例数据集 **Tickit** 加载到包含 **public** 架构的默认数据库 **Dev** 中。
6. 在数据库配置部分中，为管理员用户名和管理员用户密码字段输入值。
7. 选择创建集群，以创建您的预置数据仓库。

## 创建无服务器数据仓库

有关创建 Amazon Redshift Serverless 数据仓库的更多详细信息，请参阅《Amazon Redshift 入门用户指南》中的[使用 Amazon Redshift Serverless 创建数据仓库](#)主题。

1. 在您的首选互联网浏览器中，登录 AWS 管理控制台，然后打开 Amazon Redshift 控制台，网址为：<https://console.aws.amazon.com/redshift/>。
2. 在 Amazon Redshift 控制台中，选择试用 Amazon Redshift Serverless 按钮，以打开开始使用 Amazon Redshift Serverless 窗格。
3. 在配置部分中，选择使用默认设置径向。
4. 在开始使用 Amazon Redshift Serverless 窗格的底部，选择保存配置，以使用默认工作组、命名空间、凭证和加密设置创建无服务器数据仓库。

## 通过 Toolkit 连接到数据仓库

通过 Toolkit 连接到数据库的方法有 3 种：

- 数据库用户名和密码

- AWS Secrets Manager
- 临时凭证

要通过该 Toolkit 连接到位于现有预置集群或无服务器数据仓库中的数据库，请完成以下步骤。

#### Important

如果您已完成本用户指南主题的“先决条件”部分中的步骤，并且您的数据仓库在 Toolkit 资源管理器中不可见，请确保在资源管理器中正确的 AWS 区域内工作。

#### 使用数据库用户名和密码方法连接到数据仓库

1. 在 Toolkit 资源管理器中，展开数据仓库所在的 AWS 区域。
2. 展开 Redshift 并选择您的数据仓库，以在 VS Code 中打开选择连接类型对话框。
3. 从选择连接类型对话框中，选择数据库用户名和密码，并提供每个提示所需的信息。
4. 当 Toolkit 连接到您的数据仓库并且该流程完成后，您的可用数据库、表和架构将显示在 Toolkit 资源管理器中。

#### 使用 AWS Secrets Manager 连接到您的数据仓库

#### Note

此流程需要 AWS Secrets Manager 数据库密钥才能完成。有关如何设置数据库密钥的说明，请参阅《AWS Secrets Manager 用户指南》中的[创建 AWS Secrets Manager 数据库密钥](#)。

1. 在 Toolkit 资源管理器中，展开数据仓库所在的 AWS 区域。
2. 展开 Redshift 并选择您的数据仓库，以在 VS Code 中打开选择连接类型对话框。
3. 从选择连接类型对话框中，选择 Secrets Manager，然后提供每个提示所需的信息。
4. 当 Toolkit 连接到您的数据仓库并且该流程完成后，您的可用数据库、表和架构将显示在 Toolkit 资源管理器中。

#### 使用临时凭证连接到您的数据仓库

1. 在 Toolkit 资源管理器中，展开数据仓库所在的 AWS 区域。

2. 展开 Redshift 并选择您的数据仓库，以在 VS Code 中打开选择连接类型对话框。
3. 从选择连接类型对话框中，选择临时凭证，然后提供每个提示所需的信息。
4. 当 Toolkit 连接到您的数据仓库并且该流程完成后，您的可用数据库、表和架构将显示在 Toolkit 资源管理器中。

## 编辑与数据仓库的连接

您可以编辑与数据仓库的连接以更改要连接到的数据库。

1. 在 Toolkit 资源管理器中，展开数据仓库所在的 AWS 区域。
2. 展开 Redshift，右键单击您要连接到的数据仓库，选择编辑连接，然后提供您要连接到的数据库的名称。
3. 当 Toolkit 连接到您的数据仓库并且该流程完成后，您的可用数据库、表和架构将显示在 Toolkit 资源管理器中。

## 删除与数据仓库的连接

1. 在 Toolkit 资源管理器中，展开数据仓库所在的 AWS 区域。
2. 展开 Redshift，右键单击包含要删除的连接的数据仓库，然后选择删除连接。这样做会从 Toolkit 资源管理器中移除可用的数据库、表和架构。
3. 要重新连接到您的数据仓库，请选择单击以连接，然后提供每个提示所需的信息。默认情况下，重新连接使用以前的身份验证方法来连接到数据仓库。要使用其他方法，请在对话框中选择返回箭头，直到出现身份验证提示为止。

## 运行 SQL 语句

以下流程描述了如何通过 AWS Toolkit for Visual Studio Code 在数据库中创建和运行 SQL 语句。

### Note

要完成以下每个流程中的步骤，必须先完成本用户指南主题中的从 Toolkit 连接到数据仓库部分。

1. 在 Toolkit 资源管理器中，展开 Redshift，然后展开包含要查询的数据库的数据仓库。

2. 选择创建笔记本以指定要在本地存储笔记本的文件名和位置，然后选择确定以在 VS Code 编辑器中打开笔记本。
3. 在 VS Code 编辑器中，输入要存储在此笔记本中的 SQL 语句。
4. 选择全部运行按钮，以运行您输入的 SQL 语句。
5. 您的 SQL 语句的输出显示在您输入的语句下方。

### 向笔记本中添加 Markdown

1. 在 VS Code 编辑器中的笔记本中，选择 Markdown 按钮，以向笔记本中添加“Markdown”单元格。
2. 将您的 Markdown 输入到提供的单元格中。
3. Markdown 单元格可以通过使用位于 Markdown 单元格右上角的编辑器工具来编辑。

### 向笔记本中添加代码

1. 在 VS Code 编辑器中的笔记本中，选择代码 按钮，以向笔记本中添加“代码”单元格。
2. 将您的代码输入到提供的单元格中。
3. 您可以选择在“代码”单元格的上方或下方运行代码，方法是从位于“代码”单元格右上角的单元格编辑器工具中选择相应的按钮。

## 使用 Amazon S3

Amazon Simple Storage Service ( Amazon S3 ) 是一项可扩展的数据存储服务。AWS Toolkit for Visual Studio Code 允许您直接通过 VS Code 代码管理您的 Amazon S3 对象和资源。

有关 Amazon S3 服务的详细信息，请参阅 [Amazon S3](#) 用户指南。

以下主题介绍了如何通过 AWS Toolkit for Visual Studio Code 使用 Amazon S3 对象和资源。

### 主题

- [使用 Amazon S3 资源](#)
- [使用 Amazon S3 对象](#)

## 使用 Amazon S3 资源

您可以通过 AWS Toolkit for Visual Studio Code 使用 Amazon S3 来查看、管理和编辑您的 Amazon S3 存储桶和其他资源。

以下各节介绍了如何通过 AWS Toolkit for Visual Studio Code 使用 Amazon S3 资源。有关通过 AWS Toolkit for Visual Studio Code 使用 Amazon S3 对象（例如文件夹和文件）的信息，请参阅本用户指南中的[使用 Amazon S3 对象](#)主题。

### 创建 Amazon S3 存储桶

1. 在 Toolkit 资源管理器中，打开 Amazon S3 服务的上下文菜单（右键单击），然后选择创建存储桶...。或者，选择创建存储桶图标，以打开创建存储桶对话框。
2. 在 Bucket Name（存储桶名称）字段中，输入一个有效的存储桶名称。

按下 Enter 键以创建存储桶并关闭对话框。然后，您的新存储桶将显示在 Toolkit 中的 Amazon S3 服务下。

#### Note

由于 Amazon S3 允许您的存储桶用作可公开访问的 URL，因此您选择的存储桶名称必须具有全局唯一性。如果您想要使用的名称已被其他账户用于创建存储桶，那么您必须使用其他名称。

如果您无法创建新的存储桶，可以检查输出选项卡中的 AWS Toolkit 日志。如果您尝试使用无效的存储桶名称，则会发生 `BucketAlreadyExists` 错误。

有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的[存储桶限制](#)。

### 在 Amazon S3 存储桶中添加文件夹

您可以通过将对象分组到文件夹中来整理 S3 存储桶的内容。您也可以在文件夹中创建文件夹。

1. 在 Toolkit 资源管理器中，展开 Amazon S3 服务以查看 Amazon S3 资源列表。
2. 选择“创建文件夹”图标，以打开创建文件夹对话框。或者，打开存储桶或文件夹的上下文菜单（右键单击），然后选择创建文件夹。
3. 在文件夹名称字段中输入一个值，然后按下 Enter 键以创建文件夹并关闭对话框。您的新文件夹显示在 Toolkit 菜单中相应的 Amazon S3 资源下。

## 删除 Amazon S3 存储桶

删除 Amazon S3 存储桶时，还会删除其中包含的文件夹和对象。因此，当您尝试删除存储桶时，系统会要求您确认是否要删除它。

1. 在 Toolkit 资源管理器中，展开 Amazon S3 服务以查看 Amazon S3 资源列表。
2. 打开存储桶或文件夹的上下文菜单（右键单击），然后选择删除 Amazon S3 存储桶。
3. 当系统提示时，在文本字段中输入存储桶的名称，然后按下 Enter 键以删除存储桶并关闭确认提示。

### Note

如果您的存储桶包含对象，则系统会在删除该存储桶之前将其清空。如果您尝试一次性删除大量资源或对象，则系统可能需要一些时间才会删除这些资源或对象。当这些资源或对象被删除后，系统会向您发送一条通知，告知您它们已被成功删除。

## 使用 Amazon S3 对象

您存储在 Amazon S3 资源存储桶中的文件、文件夹以及任何其他数据都称为 Amazon S3 对象。

以下各节介绍了如何通过 AWS Toolkit for Visual Studio Code 使用 Amazon S3 对象。有关使用 Amazon S3 资源（例如 S3 存储桶）的详细信息 AWS Toolkit for Visual Studio Code，[请参阅本用户指南中的使用 S3 资源](#)主题。

### 对象分页

如果您正在使用大量 Amazon S3 对象和文件夹，分页允许您指定要在页面上显示的项目数量。

1. 导航到 VS Code 活动栏并选择扩展。
2. 从 AWS Toolkit 扩展程序中，选择设置图标，然后选择扩展设置。
3. 在设置页面上，向下滚动至 AWS > Amazon S3：每页最大项目数设置。
4. 在“加载更多”选项显示之前，将默认值更改为您要显示的 Amazon S3 项目数量。

**Note**

有效值包括 3 到 1000 之间的任意数字。此设置仅适用于同时显示的对象或文件夹的数量。您创建的所有存储桶都会一次性显示。默认情况下，您可以在每个 AWS 账户中创建多达 100 个存储桶。

5. 关闭设置页面以确认您的更改。

您还可以通过选择设置页面右上角的打开设置 ( JSON ) 图标，更新 JSON 格式文件中的设置。

## 上传和下载 Amazon S3 对象

您可以通过 AWS Toolkit for Visual Studio Code 将本地存储的文件上传到您的 Amazon S3 存储桶，也可以将远程 Amazon S3 对象下载到您的本地系统中。

### 使用 Toolkit 上传文件

1. 在 Toolkit 资源管理器中，展开 Amazon S3 服务以查看 Amazon S3 资源列表。
2. 选择位于存储桶或文件夹旁边的“上传文件”图标，以打开“上传文件”对话框。或者，您也可以打开文件的上下文菜单（右键单击），然后选择上传文件。

**Note**

要将文件上传到对象的父文件夹或资源，请打开任意 Amazon S3 对象的上下文菜单（右键单击），然后选择上传至父级。

3. 使用系统的文件管理器选择一个文件，然后选择上传文件以关闭对话框并上传文件。

### 使用“命令面板”上传文件

您可以使用 Toolkit 界面或命令面板将文件上传到存储桶。

1. 要选择要上传的文件，请在 VS Code 中选择该文件的选项卡。
2. 按下 Ctrl+Shift+P 以显示命令面板。
3. 在命令面板中，输入短语 `upload file` 以显示推荐的命令列表。
4. 选择 `AWS : 上传文件` 命令以打开 `AWS : 上传文件` 对话框。
5. 出现提示时，选择要上传的文件，然后选择要将该文件上传到的存储桶。

6. 确认您的上传，以关闭对话框并开始上传流程。上传完毕后，该对象将显示在 Toolkit 菜单中，并且包含对象大小、上次修改日期和路径等元数据。

## 下载 Amazon S3 对象

1. 在 Toolkit 资源管理器中，展开 Amazon S3 服务。
2. 在存储桶或文件夹中，打开要下载的对象的下文菜单（右键单击）。然后，选择下载为以打开“下载为”对话框。或者，也可以选择对象旁边的下载为图标。
3. 使用系统的文件管理器，选择目标文件夹，输入文件名，然后选择下载以关闭对话框并开始下载。

## 编辑远程对象

您可以使用 AWS Toolkit for Visual Studio Code 来编辑存储在远程 Amazon S3 资源中的 Amazon S3 对象。

1. 在 Toolkit 资源管理器中，展开 Amazon S3 服务。
2. 展开包含要编辑的文件的 Amazon S3 资源。
3. 要编辑文件，请选择铅笔图标（“编辑文件”）。
4. 要编辑以只读模式打开的文件，请在 VS Code 编辑器中查看该文件，然后选择位于 UI 右上角的铅笔图标。

### Note

- 如果您重新启动或退出 VS Code，则您的 IDE 将会与 Amazon S3 资源断开连接。如果您断开连接时正在编辑任何远程 Amazon S3 文件，则编辑操作将会停止。您必须重新启动 VS Code 并重新打开编辑选项卡才能恢复编辑。
- 编辑文件按钮位于 UI 的右上角。只有当您在 VS Code 编辑器中主动查看只读文件时，它才可见。
- 无法以只读模式打开非文本文件。它们始终以编辑模式打开。
- 您无法从仅限编辑模式切换回只读模式，只能从只读模式切换回仅限编辑模式。

## 复制 Amazon S3 对象的路径

以下流程了如何从 AWS Toolkit for Visual Studio Code 复制 Amazon S3 对象的路径。

1. 在 Toolkit 资源管理器中，展开 Amazon S3 服务。
2. 展开包含要复制路径的对象的资源存储桶。
3. 打开要复制路径的对象的上下文菜单（右键单击），然后选择复制路径，以将对象路径复制到本地剪贴板。

## 为 Amazon S3 对象生成预签名 URL

您可以通过预签名 URL 功能授予限时下载权限，从而与其他人共享私有 Amazon S3 对象。有关更多信息，请参阅[使用预签名 URL 共享对象](#)。

1. 在 Toolkit 资源管理器中，展开 Amazon S3 服务。
2. 在存储桶或文件夹中，打开要共享的对象的上下文菜单（右键单击）。然后，选择生成预签名 URL 以打开命令面板。
3. 在命令面板中，输入可使用该 URL 访问对象的分钟数。然后，选择 Enter 键以确认并关闭对话框。
4. 生成预签名 URL 后，VS Code 状态栏会显示已复制到本地剪贴板的对象的预签名 URL。

## 删除 Amazon S3 对象

如果对象位于不受版本控制的存储桶中，您可以永久删除它。对于启用版本控制的存储桶，删除请求不会永久删除该对象。但是，Amazon S3 将在存储桶中插入一个删除标记。有关更多信息，请参阅[删除对象版本](#)。

1. 在 Toolkit 资源管理器中，展开 Amazon S3 服务以查看 Amazon S3 资源列表。
2. 打开要删除的对象的上下文菜单（右键单击），然后选择删除以打开确认对话框。
3. 选择删除...，以确认您想要删除这个 Amazon S3 对象。然后，关闭对话框。

## 适用于 VS Code 的 Amazon SageMaker 融通式合作开发工作室

作为下一代 Amazon SageMaker 的一部分，Amazon SageMaker 融通式合作开发工作室是一种统一的开发体验，它汇集了 AWS 数据、分析、人工智能（AI）和机器学习（ML）服务。它提供了从单一界面构建、部署、执行和监控工作流的位置。有关设置 Amazon SageMaker 融通式合作开发工作室与 VS Code IDE 的集成的更多信息，请参阅《Amazon SageMaker 融通式合作开发工作室用户指南》中的[在 VS Code 中设置 Amazon SageMaker 融通式合作开发工作室集成](#)。

## 使用无服务器应用程序

AWS Toolkit for Visual Studio Code 提供对 [AWS Serverless Application](#) 的支持。以下主题介绍了如何从 AWS Toolkit for Visual Studio Code 开始创建和使用 AWS Serverless Application Model (AWS SAM) 应用程序。

### 主题

- [开始使用无服务器应用程序](#)
- [使用 AWS 无服务器土地](#)
- [直接通过代码运行和调试 Lambda 函数](#)
- [运行和调试本地 Amazon API Gateway 资源](#)
- [调试无服务器应用程序的配置选项](#)
- [排查无服务器应用程序的问题](#)

## 开始使用无服务器应用程序

以下各节介绍如何开始 AWS Serverless Application 从 AWS Toolkit for Visual Studio Code、使用 AWS Serverless Application Model (AWS SAM) 和 CloudFormation 堆栈创建。

### 先决条件

在创建或使用之前 AWS Serverless Application，必须先满足以下先决条件。

#### Note

以下操作可能需要您在更改完成之前退出或重新启动 VS Code。

- 安装 AWS SAM 命令行界面 (CLI)。有关如何安装 AWS SAM CLI 的其他信息和说明，请参阅本AWS Serverless Application Model 用户指南中的[安装 AWS SAM CLI](#) 主题。
- 在您的 AWS 配置文件中，确定您的默认 AWS 区域。有关配置文件的更多信息，请参阅《AWS Command Line Interface 用户指南》中的[配置和凭证文件设置](#)主题。
- 安装语言 SDK 并配置工具链。有关如何配置工具链的更多信息，AWS Toolkit for Visual Studio Code 请参阅本用户指南中的[配置您的工具链](#)主题。
- 从 VS Code Marketplace 安装 [YAML 语言支持扩展](#)。这是 AWS SAM 模板文件可访问的 CodeLens 功能所必需的。有关的其他信息 CodeLens，请参阅 VS Code 文档中的[CodeLens](#)部分

## 无服务器应用程序的 IAM 权限

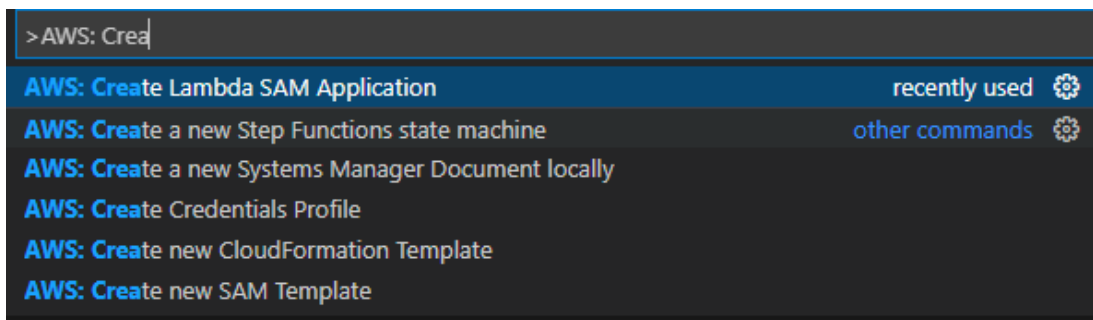
在 Toolkit for VS Code 中，您必须具有一个凭证配置文件，其中包含部署和运行无服务器应用程序所必需的 AWS Identity and Access Management (IAM) 权限。您必须有权 read/write 访问以下服务：CloudFormation、IAM、Lambda、Amazon API Gateway、亚马逊简单存储服务 (Amazon S3) 和亚马逊弹性容器注册表 (Amazon ECR) Container Registry (Amazon ECR) Elastic Registry。

有关设置部署和运行无服务器应用程序所需的身份验证的更多信息，请参阅《AWS Serverless Application Model 开发人员指南》中的 [管理资源访问和权限](#)。有关如何设置凭证的信息，请参阅本用户指南中的 [AWS IAM 证书](#)。

## 创建新的无服务器应用程序 (本地)

此过程演示如何使用 AWS SAM Toolkit for VS Code 创建无服务器应用程序。此过程的输出是开发主机上的本地目录，其中包含一个示例无服务器应用程序，您可以构建、本地测试、修改该应用程序并将其部署到 AWS 云中。

1. 要打开命令面板，请依次选择视图和命令面板，然后输入 AWS。
2. 选择 AWS Toolkit 创建 Lambda SAM 应用程序。



### Note

如果未安装 AWS SAM CLI，则在 VS Code 编辑器的右下角会出现错误。如果发生这种情况，请验证是否满足所有的 [假设和先决条件](#)。

3. 为您的 AWS SAM 应用程序选择运行时。

**Note**

如果您选择一个带有“( 图像 )”的运行时，则您的应用程序是 Image 型软件包。如果您选择一个不带有“( 图像 )”的运行时，则您的应用程序是 Zip 型。有关 Image 和 Zip 软件包类型差异的更多信息，请参阅 AWS Lambda 开发人员指南中的 [Lambda 部署软件包](#)。

4. 根据您选择的运行时，您可能需要为 SAM 应用程序选择依赖项管理器和运行时架构。

### Dependency Manager

选择 Gradle 或 Maven。

**Note**

这个构建自动化工具的选择仅适用于 Java 运行时。

### Architecture

选择 x86\_64 或 arm64。

以下运行时可以选择在 ARM64 基于模拟的环境中运行无服务器应用程序，而不是默认的基于 x86\_64 的环境：

- nodejs12.x ( ZIP 和图像 )
- nodejs14.x ( ZIP 和图像 )
- python3.8 ( ZIP 和图像 )
- python3.9 ( ZIP 和图像 )
- python3.10 ( ZIP 和图像 )
- python3.11 ( ZIP 和图像 )
- python3.12 ( ZIP 和图像 )
- 带有 Gradle 的 java8.al2 ( ZIP 和图像 )
- 带有 Maven 的 java8.al2 ( 仅限 ZIP )
- 带有 Gradle 的 java11 ( ZIP 和图像 )
- 带有 Maven 的 java11 ( 仅限 ZIP )

**⚠ Important**

必须安装 AWS CLI 版本 1.33.0 或更高版本才能允许应用程序在 ARM64 基于基础的环境中运行。有关更多信息，请参阅 [先决条件](#)。

5. 为新项目选择一个位置。您可以使用现有工作区文件夹（如果已打开一个此类文件夹），Select a different folder (选择已存在的其他文件夹)，或创建新文件夹并将其选中。对于此示例，选择 There are no workspace folders open (没有打开的工作区文件夹) 以创建名为 MY-SAM-APP 的文件夹。
6. 输入新项目的名称。对于本示例，请使用 my-sam-app-nodejs。按下 Enter 键后，Toolkit for VS Code 需要几分钟才能创建项目。

创建项目后，应用程序将添加到当前工作区中。您应该看到它在 Explorer 窗口中列出。

## 打开无服务器应用程序（本地）

要在本地开发主机上打开无服务器应用程序，请打开包含该应用程序的模板文件的文件夹。

1. 从文件中选择打开文件夹...
2. 在打开文件夹对话框中，导航到要打开的无服务器应用程序文件夹。
3. 选择选择文件夹按钮。

当您打开应用程序的文件夹时，它会被添加到资源管理器窗口中。

## 从模板（本地）运行和调试无服务器应用程序

您可以使用 Toolkit for VS Code 配置如何调试无服务器应用程序，并在开发环境中本地运行它们。

您可以使用 VS Code [CodeLens](#) 功能来识别符合条件的 Lambda 函数，从而开始配置调试行为。CodeLens 启用与源代码的内容感知交互。有关确保您可以访问该 CodeLens 功能的信息，请查看本主题前 [先决条件](#) 面的部分。

**📘 Note**

在此示例中，您将调试使用的一个应用程序 JavaScript。但是，您可以使用具有以下语言和运行时系统的 Toolkit for VS Code 调试功能：

- C# : .NET Core 2.1, 3.1 ; .NET 5.0
- JavaScript/TypeScript — Node.js 12。 x、 14。 x
- Python : 3.6、 3.7、 3.8、 3.9、 3.10、 3.11、 3.12
- Java : 8、 8.al2、 11
- Go : 1.x

您的语言选择还会影响 CodeLens 检测符合条件的 Lambda 处理程序的方式。有关更多信息，请参阅 [直接通过代码运行和调试 Lambda 函数](#)。

在此流程中，您需要使用在本主题前面的 [创建新的无服务器应用程序 \(本地\)](#) 部分中创建的示例应用程序。

1. 要在 VS Code 的文件资源管理器中查看应用程序文件，请依次选择视图和资源管理器。
2. 在应用程序文件夹 (例如 my-sample-app) 中，打开该 template.yaml 文件。

#### Note

如果您使用的模板名称与之不同 template.yaml，则该 CodeLens 指标不会自动在 YAML 文件中可用。这意味着您必须手动添加调试配置。

3. 在 template.yaml 的编辑器中，转到模板中定义无服务器资源的 Resources 部分。在本例中，这是类型 AWS::Serverless::Function 的 HelloWorldFunction 资源。

在此资源的 CodeLens 指示器中，选择“添加调试配置”。

使用 template.yaml 文件中的 CodeLens 指示器添加调试配置。

4. 在命令面板中，选择将运行您的 AWS SAM 应用程序的运行时系统。
5. 在 launch.json 文件的编辑器中，编辑或确认以下配置属性的值：
  - "name" – 输入一个易于阅读的名称，以显示在 Run (运行) 视图中的 Configuration (配置) 下拉字段中。
  - "target" — 确保该值为，"template" 以便 AWS SAM 模板成为调试会话的入口点。
  - "templatePath" – 输入 template.yaml 文件的相对路径或绝对路径。
  - "logicalId" — 确保名称与 AWS SAM 模板的“资源”部分中指定的名称相匹配。在这种情况下，它是类型 AWS::Serverless::Function 的 HelloWorldFunction。

配置 `launch.json` 文件以进行基于模板的调试。

有关 `launch.json` 文件中这些以及其他条目的更多信息，请参阅 [调试无服务器应用程序的配置选项](#)。

6. 如果您对调试配置满意，请保存 `launch.json`。然后，选择运行视图旁边的“播放”按钮以启动调试。

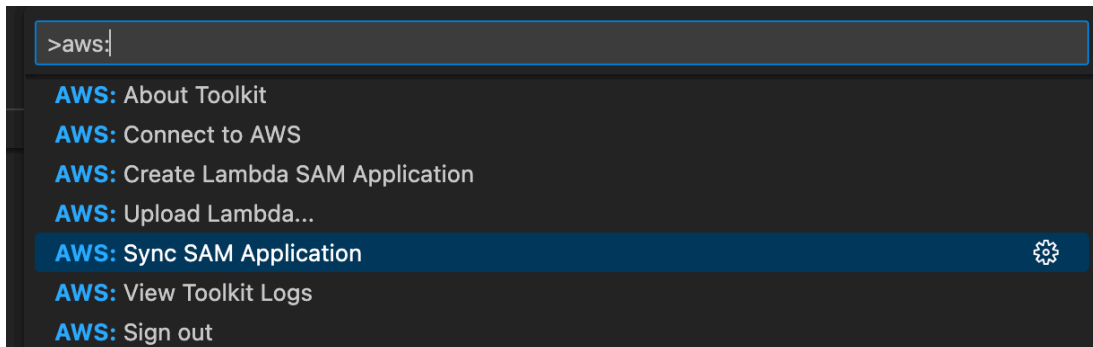
当调试会话启动时，DEBUG CONSOLE（调试控制台）面板显示调试输出，并显示 Lambda 函数返回的任何值。（调试 AWS SAM 应用程序时，AWS Toolkit 在输出面板中被选择作为输出通道。）

## 同步应用程序 AWS SAM

AWS Toolkit for Visual Studio Code 运行 AWS SAM CLI 命令 `aws sam sync` 将您的无服务器应用程序部署到。AWS Cloud 有关 AWS SAM 同步的其他信息，请参阅《AWS Serverless Application Model 开发者指南》中的 [AWS SAM CLI 命令参考](#) 主题

以下过程介绍如何通过 Toolkit for VS Code 将您的无服务器应用程序部署 `aws sam sync` 到。AWS Cloud

1. 在 VS Code 内的主菜单中，展开视图并选择命令面板，以打开命令面板。
2. 从命令面板中搜索 AWS 并选择同步 SAM 应用程序，以开始设置同步。



3. 选择要将您的无服务器应用程序同步到的 AWS 区域。
4. 选择要用于部署的 `template.yaml` 文件。
5. 选择现有的 Amazon S3 存储桶或输入新的 Amazon S3 存储桶名称，以便将您的应用程序部署到其中。

**⚠ Important**

Amazon S3 存储桶包必须符合以下要求：

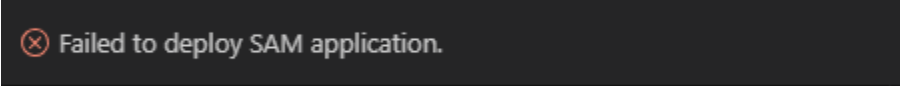
- 存储桶必须位于您要同步到的区域中。
- Amazon S3 存储桶名称在 Amazon S3 中的所有现有存储桶名称之间必须是全局唯一的。

6. 如果您的无服务器应用程序包含具有类型 Image 的函数，输入此部署可使用的 Amazon ECR 存储库的名称。存储库必须位于您要部署到的区域中。
7. 从之前的部署列表中选择部署堆栈，或者通过输入新的堆栈名称来创建新的部署堆栈。然后，继续操作以开始同步流程。

**ℹ Note**

先前部署中使用的堆栈会按工作空间和区域召回。

8. 在同步流程期间，部署状态会在 VS Code 的终端选项卡中捕获。从“终端”选项卡验证同步是否成功，如果出现错误，您会收到通知。



```
⊗ Failed to deploy SAM application.
```

**ℹ Note**

有关同步的更多详细信息，可从命令面板访问 AWS Toolkit for Visual Studio Code 日志。

要从命令面板访问您的 AWS Toolkit for Visual Studio Code 日志，请展开视图，选择命令面板，然后搜索**AWS: View AWS Toolkits Logs**并在列表中填充日志时将其选中。

一旦完成部署，您将看到 AWS Explorer 窗口中列出了您的应用程序。有关如何调用作为应用程序的一部分创建的 Lambda 函数的更多信息，请参阅本《用户指南》中的 [使用 AWS Lambda 函数](#) 主题。

## 从中删除无服务器应用程序 AWS Cloud

删除无服务器应用程序涉及删除您之前部署到 AWS 云端的 CloudFormation 堆栈。请注意，此过程不会从本地主机中删除您的应用程序目录。

1. 打开 [AWS Explorer](#)。
2. 在 AWS Toolkit Explorer 窗口中，展开包含您想要删除的已部署应用程序的区域，然后展开 CloudFormation。
3. 打开与要删除的无服务器应用程序相对应的 CloudFormation 堆栈名称的上下文（右键单击）菜单，然后选择删除 CloudFormation 堆栈。
4. 要确认删除已选择的堆栈，请选择是。

如果堆栈删除成功，Toolkit for VS Code 将从 AWS Explorer 中的 CloudFormation 列表中移除堆栈名称。

## 使用 AWS 无服务器土地

AWS 中的 Serverless Land AWS Toolkit for Visual Studio Code 是一系列功能，可帮助您构建事件驱动型架构。以下主题部分描述了如何在 AWS 工具包中使用 Serverless Land。有关 Serverless Land 的详细信息，请参阅 [Serverless Land](#) Web 应用程序。

### 访问 Serverless Land

AWS 工具包中有 3 个访问无服务器版块的主要入口点：

- VS Code 命令面板
- AWS 工具包浏览器
- AWS 工具包应用程序生成器资源管理器

从 VS Code 命令面板中打开 Serverless Land

要从 VS Code 命令面板中打开 Serverless Land，请完成以下步骤。

1. 在 VS Code 中，按下 **option+shift+p** ( Mac ) 或 **control+shift+p** ( Windows ) 打开命令面板。
2. 在 VS Code 命令面板的搜索栏中，输入 **AWS Create application with Serverless template**。
3. 在 AWS：使用无服务器模板创建应用程序出现在列表时选择此命令。
4. 该过程完成后，Serverless Land 向导将在 VS Code 中打开，显示为您的应用程序选择模式 ( 1/5 ) 屏幕。

从“AWS 工具包资源管理器”中打开无服务器版块。

要从 AWS Toolkit 资源管理器中打开 Serverless Land，请完成以下步骤。

1. 在 AWS Toolkit Explorer 中，展开要在其中打开 Serverless Land 的区域。
2. 打开 Lambda 节点的上下文菜单（右键单击）。
3. 从快捷菜单中选择使用无服务器模板创建应用程序。
4. 该过程完成后，Serverless Land 向导将在 VS Code 中打开，显示为您的应用程序选择模式（1/5）屏幕。

从 Application Builder 浏览器打开 Serverless Land

要从 AWS Toolkit 应用程序生成器资源管理器中打开 Serverless Land，请完成以下步骤。

1. 在 AWS 工具包资源管理器中，导航到应用程序生成器资源管理器。
2. 右键单击 Application Builder 浏览器，然后从上下文菜单中选择使用无服务器模板创建应用程序。
3. 该过程完成后，Serverless Land 向导将在 VS Code 中打开，显示为您的应用程序选择模式（1/5）屏幕。

## 使用无服务器模板创建应用程序

要使用无服务器模板创建应用程序，请完成以下步骤。

1. 在 Serverless Land 向导的为您的应用程序选择模式（1/5）屏幕中，为您的应用程序基础选择一个模式。

### Note

要查看特定模式的预览和更多详细信息，请选择要查看的模式旁边的在 Serverless Land 中打开图标。Serverless Land 模式将在您的默认 Web 浏览器中打开。

2. 从选择运行时（2/5）屏幕中，为项目选择运行时。
3. 在选择 IaC（3/5）屏幕中，为您的项目选择一个 IaC 选项。
4. 在选择项目位置（4/5）屏幕中，选择存储项目的位置。
5. 在输入项目名称（5/5）屏幕中，输入新应用程序的名称。
6. 过程完成后，您的新应用程序将显示在 VS Code 浏览器中，您的项目 `readme.md` 将在 VS Code 编辑器中打开。

**Note**

创建新应用程序后，可以在 `readme.md` 文件中找到特定于应用程序类型的其他操作。此外，您的 AWS Serverless Application Model (AWS SAM) 应用程序可以使用 Application Builder 打开，以进行本地测试、调试等。

有关在 AWS Toolkit 中使用 Application Builder 的详细信息，请参阅本用户指南中的 [使用 AWS 应用程序生成器资源管理器](#) 主题。

## 直接通过代码运行和调试 Lambda 函数

测试 AWS SAM 应用程序时，您可以选择仅运行和调试 Lambda 函数，并排除 AWS SAM 模板定义的其他资源。这种方法包括使用该 [CodeLens](#) 功能在源代码中识别您可以直接调用的 Lambda 函数处理程序。

检测到的 Lambda 处理程序 CodeLens 取决于您用于应用程序的语言和运行时间。

Language/runtime	用指标识别 Lambda 函数的标准 CodeLens
C# ( dotnetcore2.1、3.1 ; .NET 5.0 )	<p>该函数具有以下功能：</p> <ul style="list-style-type: none"> <li>• 这是公有类的公有函数。</li> <li>• 它有一个或两个参数。如果有两个参数，则第二个参数必须实现 <code>ILambdaContext</code> 接口。</li> <li>• 它在 VS Code 工作区文件夹内的父文件夹中具有 <code>*.csproj</code> 文件。</li> </ul> <p><a href="#">ms-dotnettools.csharp 扩展</a> ( 或为 C# 提供语言符号的任何扩展 ) 已安装且已启用。</p>
JavaScript/TypeScript (Node.js 12.x、14.x)	<p>该函数具有以下功能：</p> <ul style="list-style-type: none"> <li>• 这是一个导出的函数，最多有三个参数。</li> <li>• 它在 VS Code 工作区文件夹内的父文件夹中具有 <code>package.json</code> 文件。</li> </ul>

Language/runtime	用指标识别 Lambda 函数的标准 CodeLens
Python ( 3.7、3.8、3.9、3.10、3.11、3.12 )	<p>该函数具有以下功能：</p> <ul style="list-style-type: none"><li>• 这是一个顶级函数。</li><li>• 它在 VS Code 工作区文件夹内的父文件夹中具有 <code>requirements.txt</code> 文件。</li></ul> <p><a href="#">ms-python.python 扩展</a> ( 或为 Python 提供语言符号的任何扩展 ) 已安装且已启用。</p>

Language/runtime	用指标识别 Lambda 函数的标准 CodeLens
Java ( 8、8.al2、11 )	<p>该函数具有以下功能：</p> <ul style="list-style-type: none"><li>• 这是公有非抽象类的公有函数。</li><li>• 它有一个、两个或三个参数。<ul style="list-style-type: none"><li>• 一个参数：参数可以是任意值。</li><li>• 两个参数：参数必须是 <code>java.io.InputStream</code> 和 <code>java.io.OutputStream</code>，或者最后一个参数必须是 <code>com.amazonaws.services.lambda.runtime.Context</code>。</li><li>• 三个参数：参数必须是 <code>java.io.InputStream</code> 和 <code>java.io.OutputStream</code>，或者最后一个参数必须是 <code>com.amazonaws.services.lambda.runtime.Context</code>。</li></ul></li><li>• 它在 VS Code 工作区文件夹内的父文件夹中具有 <code>build.gradle</code> ( Gradle ) 或 <code>pom.xml</code> ( Maven ) 文件。</li></ul> <p><a href="#">redhat.java 扩展</a> ( 或为 Python 提供语言符号的任何扩展 ) 已安装且已启用。无论你使用的是哪个 Java 运行时系统，该扩展都需要 Java 11。</p> <p><a href="#">vscjava.vscode-java-debug</a> 扩展 ( 或提供 Java 调试器的任何扩展 ) 已安装且已启用。</p>

Language/runtime	用指标识别 Lambda 函数的标准 CodeLens
Go (1.x)	<p>该函数具有以下功能：</p> <ul style="list-style-type: none"> <li>• 这是一个顶级函数。</li> <li>• 它需要 0 到 2 个参数。如果有两个参数，则第一个参数必须实现 <code>context.Context</code>。</li> <li>• 它会返回 0 到 2 个参数。如果有 0 个以上的参数，则最后一个参数必须实现 <code>error</code>。</li> <li>• 它在 VS Code 工作区文件夹中具有 <code>go.mod</code> 文件。</li> </ul> <p><a href="#">golang.go 扩展</a> 已安装和配置且已启用。</p>

## 直接从应用程序代码运行和调试无服务器应用程序

1. 要在 VS Code 文件资源管理器中查看应用程序文件，请选择视图、资源管理器。
2. 从应用程序文件夹（例如 `my-sample-app`），展开函数文件夹（在此案例中，为 `hello-world`），然后打开 `app.js` 文件。
3. 在标识符合条件的 Lambda 函数处理程序的 CodeLens 指标中，选择 **Add Debug Configuration** 访问 Lambda 函数处理程序 CodeLens 指标中的“添加调试配置”选项。
4. 在命令面板中，选择将运行您的 AWS SAM 应用程序的运行时系统。
5. 在 `launch.json` 文件的编辑器中，编辑或确认以下配置属性的值：
  - `"name"` – 输入一个易于阅读的名称，以显示在 Run（运行）视图中的 Configuration（配置）下拉字段中。
  - `"target"` – 确保值为 `"code"`，以便直接调用 Lambda 函数处理程序。
  - `"lambdaHandler"` – 输入代码中方法的名称，Lambda 可使用该方法来调用您的函数。例如，对于中的应用程序 JavaScript，默认值为 `app.lambdaHandler`。
  - `"projectRoot"` – 输入包含 Lambda 函数的应用程序文件的路径。
  - `"runtime"` – 输入或确认 Lambda 执行环境的有效运行时，例如 `"nodejs12.x"`。
  - `"payload"` – 选择以下选项之一以定义要作为输入提供给 Lambda 函数的事件负载：

- "json" : 定义事件负载的 JSON-formatted 键值对。
- "path" : 用作事件负载的文件路径。

在下面的示例中，"json" 选项定义了负载。

配置 `launch.json` 文件以直接调用 Lambda 函数。

有关 `launch.json` 文件中这些以及其他条目的更多信息，请参阅 [调试无服务器应用程序的配置选项](#)。

6. 如果您对调试配置满意，请选择运行 旁边的绿色播放箭头以启动调试。

当调试会话启动时，调试控制台面板会显示调试输出，并显示 Lambda 函数返回的任何值。（调试 AWS SAM 应用程序时，会在“输出”面板中选择 AWS Toolkit 作为输出通道。）

## 运行和调试本地 Amazon API Gateway 资源

您可以通过使用 `invokeTarget.target=api` 运行 `type=aws-sam` 的 VS Code 启动配置，运行或调试 `template.yaml` 中指定的 AWS SAM API Gateway 本地资源。

### Note

API Gateway 支持两种类型的 API：REST 和 HTTP。但是，带有 AWS Toolkit for Visual Studio Code 的 API Gateway 功能仅支持 REST API。有时候 HTTP API 被称为“API Gateway V2 API”。

## 运行和调试本地 API Gateway 资源

1. 选择以下方法之一以创建 AWS SAM API Gateway 资源的启动配置：

- 选项 1：访问 AWS SAM 项目中的处理程序源代码（.js、.cs 或 .py 文件），将鼠标悬停在 Lambda 处理程序上，然后选择添加调试配置 CodeLens。然后，在菜单中，选择标记为 API 事件的项目。
- 选项 2：编辑 `launch.json` 并使用以下语法创建新的启动配置。

```
{
```

```
"type": "aws-sam",
"request": "direct-invoke",
"name": "myConfig",
"invokeTarget": {
  "target": "api",
  "templatePath": "n12/template.yaml",
  "logicalId": "HelloWorldFunction"
},
"api": {
  "path": "/hello",
  "httpMethod": "post",
  "payload": {
    "json": {}
  }
},
"sam": {},
"aws": {}
}
```

2. 在 VS Code 运行面板中，选择启动配置（在上面的示例中名为 myConfig）。
3. （可选）将断点添加到您的 Lambda 项目代码中。
4. 在运行面板中，输入 F5 或选择播放。
5. 在输出窗格中，查看结果。

## 配置

在使用 `invokeTarget.target` 属性值 `api` 时，Toolkit 会更改启动配置验证和行为，以支持 `api` 字段。

```
{
  "type": "aws-sam",
  "request": "direct-invoke",
  "name": "myConfig",
  "invokeTarget": {
    "target": "api",
    "templatePath": "n12/template.yaml",
    "logicalId": "HelloWorldFunction"
  },
  "api": {
```

```
"path": "/hello",
"httpMethod": "post",
"payload": {
  "json": {}
},
"queryString": "abc=def&qrs=tuv",
"headers": {
  "cookie": "name=value; name2=value2; name3=value3"
}
},
"sam": {},
"aws": {}
}
```

按以下示例的方式替换值：

`invokeTarget.logicalId`

API 资源。

`path`

启动 Config 请求的 API 路径，如 `"path": "/hello"`。

必须是从 `invokeTarget.templatePath` 指定的 `template.yaml` 解析出的有效 API 路径。

`httpMethod`

以下任一动词：“delete”、“get”、“head”、“options”、“patch”、“post”和“put”。

`payload`

要在请求中发送的 JSON 负载（HTTP 正文），其结构和规则与 [lambda.payload](#) 字段相同。

`payload.path` 指向包含 JSON 负载的文件。

`payload.json` 指定内联 JSON 负载。

标头

可选名称-值对映射，用于指定要包含在请求中的 HTTP 标头，如以下示例中所示。

```
"headers": {
  "accept-encoding": "deflate, gzip;q=1.0, *;q=0.5",
  "accept-language": "fr-CH, fr;q=0.9, en;q=0.8, de;q=0.7, *;q=0.5",
```

```

    "cookie": "name=value; name2=value2; name3=value3",
    "user-agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.198 Safari/537.36",
  }

```

## querystring

可选字符串，设置请求的 querystring，如 "querystring": "abc=def&ghi=jkl"。

## AWS

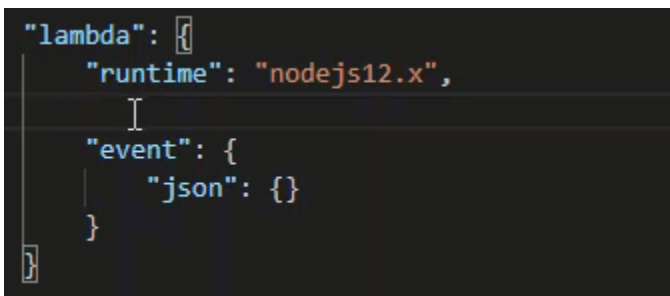
AWS 连接信息提供的方式。有关更多信息，请参阅 [调试无服务器应用程序的配置选项](#) 部分中的 AWS 连接 (“aws”) 属性表。

## sam

AWS SAM CLI 构建应用程序的方式。有关更多信息，请参阅 [调试无服务器应用程序的配置选项](#) 部分中的 AWS SAM CLI (“sam”) 属性表。

## 调试无服务器应用程序的配置选项

打开 launch.json 文件编辑调试配置时，可以使用 VS Code [IntelliSense](#) 功能查看并自动完成有效属性。要在编辑器 IntelliSense 中触发，请按 Ctrl + 空格键。



```

"lambda": {
  "runtime": "nodejs12.x",
  "event": {
    "json": {}
  }
}

```

IntelliSense 允许您直接或使用模板查找和定义用于调用 Lambda 函数的 AWS SAM 属性。您还可以为 "lambda" (函数的运行方式)、"sam" (AWS SAM CLI 如何构建应用程序) 和 "aws" (如何提供 AWS 连接信息) 定义属性。

### AWS SAM : Lambda 处理程序直接调用/基于模板的 Lambda 调用

属性	说明
type	指定哪个扩展管理启动配置。始终设置 aws-sam 为使用 AWS SAM CLI 在本地进行构建和调试。

属性	说明
name	指定一个易于阅读的名称，以显示在 Debug launch configuration ( 调试启动配置 ) 列表中。
request	指定要由指定扩展程序执行的配置类型 (aws-sam)。始终设置为 direct-invoke 以启动 Lambda 函数。
invokeTarget	<p>指定资源调用的入口点。</p> <p>为了直接调用 Lambda 函数，请为以下 invokeTarget 字段设置值：</p> <ul style="list-style-type: none"><li>• target – 设置为 code。</li><li>• lambdaHandler – 要调用的 Lambda 函数处理程序的名称。</li><li>• projectRoot : 指向包含 Lambda 函数处理程序的应用程序文件的路径。</li><li>• architecture : 运行本地 SAM Lambda 应用程序的模拟环境的处理器架构。对于某些运行时系统，您可以选择 arm64，而不是默认的 x86_64 架构。有关更多信息，请参阅 <a href="#">创建新的无服务器应用程序 (本地)</a>。</li></ul> <p>要使用模板调用 Lambda 资源 AWS SAM，请为以下 invokeTarget 字段设置值：</p> <ul style="list-style-type: none"><li>• target – 设置为 template。</li><li>• templatePath : 指向 AWS SAM 模板文件的路径。</li><li>• logicalId – 要调用的 AWS::Lambda::Function 或 AWS::Serverless::Function 的资源名称。您可以在 YAML 格式 AWS SAM 的模板中找到资源名称。请注意，AWS Toolkit 隐式将 AWS SAM 模板 PackageType: Image 中定义的函数识别为基于 <a href="#">图像的 Lambda</a> 函数。有关更多信息，请参阅《AWS Lambda 开发人员指南》中的 <a href="#">Lambda 部署包</a>。</li></ul>

Lambda ("**lambda**") 属性

属性	说明
environmentVariables	<p>将操作参数传递到您的 Lambda 函数。例如，您在写入 Amazon S3 存储桶时，不应对要写入的存储桶名称进行硬编码，而应将存储桶名称配置为环境变量。</p> <div data-bbox="620 510 742 548"><p> Note</p></div> <p>为无服务器应用程序指定环境变量时，必须同时向 AWS SAM 模板 (template.yaml ) 和 launch.json 文件中添加配置。</p> <p>AWS SAM 模板中环境变量的格式化示例：</p> <pre data-bbox="688 808 1299 1192">Resources:   HelloWorldFunction:     Type: AWS::Serverless::Function   Properties:     CodeUri: hello-world/     Handler: app.lambdaHandlerN10     Runtime: nodejs10.x     Environment:       Variables:         SAMPLE1: Default Sample 1 Value</pre> <p>launch.json 文件中环境变量的格式化示例：</p> <pre data-bbox="688 1350 1234 1457">"environmentVariables": {   "SAMPLE1": "My sample 1 value" }</pre>

属性	说明
runtime	指定 Lambda 函数使用的运行时系统。有关更多信息，请参阅 <a href="#">AWS Lambda 运行时</a> 。
timeoutSec	设置调试会话超时之前的允许时间（以秒为单位）。
pathMappings	<p>指定本地代码在容器中的运行位置。</p> <p>默认情况下，Toolkit for VS Code 将 localRoot 设置为本地工作区中 Lambda 函数的代码根目录，并将 remoteRoot 设置为 /var/task（在 Lambda 中运行的代码的默认工作目录）。如果在 Dockerfile 中或 CloudFormation 模板文件中的 WorkingDirectory 参数更改了工作目录，则必须至少指定一个 pathMapping 条目，这样调试器才能成功地将本地设置的断点映射到 Lambda 容器中运行的代码。</p> <p>launch.json 文件中 pathMappings 的格式化示例：</p> <pre> "pathMappings": [   {     "localRoot": " \${workspaceFolder}/sam-app/HelloWorldFunction ",     "remoteRoot": " /var/task "   } ] </pre> <p>注意事项：</p> <ul style="list-style-type: none"> <li>对于基于 .NET 映像的 Lambda 函数，remoteRoot 条目必须是构建目录。</li> <li>对于基于 Node.js 的 Lambda 函数，您只能指定一个路径映射条目。</li> </ul>

VS Code 工具包使用 AWS SAM CLI 在本地构建和调试无服务器应用程序。您可以使用 launch.json 文件中配置的属性来 "sam" 配置 AWS SAM CLI 命令的行为。

## AWS SAM CLI ("sam") 属性

属性	说明	默认值
buildArguments	配置 sam build 命令构建 Lambda 源代码的方式。若要查看构建选项，请参阅 AWS Serverless Application Model 开发人员指南中的 <a href="#">sam 构建</a> 。	空字符串
containerBuild	指示是否在类似于 Lambda 的 Docker 容器内构建函数。	false
dockerNetwork	Lambda Docker 容器应连接到的现有 Docker 网络的名称或 ID，以及默认桥接网络。如果未指定此项，Lambda 容器将仅连接到默认的桥接 Docker 网络。	空字符串
localArguments	指定其他本地调用参数。	空字符串
skipNewImageCheck	指定命令是否应跳过下拉最新 Docker 镜像获取 Lambda 运行时的操作。	false
template	使用参数自定义 AWS SAM 模板以输入客户价值。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的 <a href="#">参数</a> 。	"parameters":{}

## AWS 连接 ("aws") 属性

属性	说明	默认值
credentials	从您的凭证文件中选择特定的个人资料 (例如 profile:d	您的现有 <a href="#">共享 AWS 配置文件</a> 或 <a href="#">共享 AWSAWS 凭据文件</a> 提

属性	说明	默认值
	default ) 以获取 AWS 凭证。	供给 Toolkit for VS Code 的凭证。
region	设置服务的 AWS 区域 ( 例如 us-east-1 ) 。	与活动凭证配置文件关联的默认 AWS 区域。

## 示例：模板启动配置

以下是 AWS SAM 模板目标的启动配置文件示例：

```
{
  "configurations": [
    {
      "type": "aws-sam",
      "request": "direct-invoke",
      "name": "my-example:HelloWorldFunction",
      "invokeTarget": {
        "target": "template",
        "templatePath": "template.yaml",
        "logicalId": "HelloWorldFunction"
      },
      "lambda": {
        "payload": {},
        "environmentVariables": {}
      }
    }
  ]
}
```

## 示例：代码启动配置

以下是 Lambda 函数目标的启动配置文件示例：

```
{
  "configurations": [
    {
      "type": "aws-sam",
      "request": "direct-invoke",
      "name": "my-example:app.lambda_handler (python3.7)",
```

```
    "invokeTarget": {
      "target": "code",
      "projectRoot": "hello_world",
      "lambdaHandler": "app.lambda_handler"
    },
    "lambda": {
      "runtime": "python3.7",
      "payload": {},
      "environmentVariables": {}
    }
  }
]
```

## 排查无服务器应用程序的问题

本主题详细介绍了在使用 Toolkit for VS Code 创建无服务器应用程序时可能遇到的常见错误以及如何解决这些问题。

### 主题

- [如何使用带有 SAM 启动配置的 samconfig.toml ?](#)
- [错误：“RuntimeError：容器不存在”](#)
- [错误：“docker.errors。APIError: 500 服务器错误... 您已达到拉取率上限。”](#)
- [错误：“500 服务器错误：正在挂载 C:\Users\...”](#)
- [使用 WSL，网络视图（例如，“调用 AWS” 表单）已损坏](#)
- [正在调试 TypeScript 应用程序，但断点不起作用](#)

### 如何使用带有 SAM 启动配置的 samconfig.toml ?

通过在启动配置的 `sam.localArguments` 属性中配置 `--config-file` 参数来指定 SAM CLI [samconfig.toml](#) 的位置。例如，如果 `samconfig.toml` 文件位于工作区的顶层：

```
"sam": {
  "localArguments": ["--config-file", "${workspaceFolder}/samconfig.toml"],
}
```

## 错误：“RuntimeError：容器不存在”

如果您的系统没有足够的磁盘空间容纳 Docker 容器，则 `sam build` 命令可能会显示此错误。如果您的系统存储空间只有 1-2 GB 的可用空间，则在处理过程中 `sam build` 可能会失败，即使在构建开始之前系统存储空间未完全满也是如此。有关更多信息，请参阅[此 GitHub 问题](#)。

## 错误：“docker.errors。APIError: 500 服务器错误... 您已达到拉取率上限。”

Docker Hub 限制匿名用户可以发出的请求数量。如果系统达到该上限，Docker 就会失败，并且此错误消息会出现在 VS Code 的输出视图中：

```
docker.errors.APIError: 500 Server Error: Internal Server Error ("toomanyrequests: You
have
reached your pull rate limit. You may increase the limit by authenticating and
upgrading:
https://www.docker.com/increase-rate-limit")
```

确保系统 Docker 服务已使用 Docker Hub 凭据进行身份验证。

## 错误：“500 服务器错误：正在挂载 C:\Users\...”

Windows 用户在调试 AWS SAM 应用程序时可能会看到此 Docker 挂载错误：

```
Fetching lambci/lambda:nodejs10.x Docker container image.....
2019-07-12 13:36:58 Mounting C:\Users\\AppData\Local\Temp\ ... as /var/
task:ro,delegated inside runtime container
Traceback (most recent call last):
...
requests.exceptions.HTTPError: 500 Server Error: Internal Server Error ...
```

尝试刷新共享驱动器的凭证（在 Docker 设置中）。

## 使用 WSL，网络视图（例如，“调用 AWS”表单）已损坏

对于 Cisco VPN 用户来说，这是一个已知的 VS Code 问题。有关更多信息，请参阅[此 GitHub 问题](#)。

[此 WSL 跟踪问题](#)中推荐了一种解决方法。

## 正在调试 TypeScript 应用程序，但断点不起作用

如果没有源映射可以将编译后的 JavaScript 文件链接到源文件，就会发生这种情况。

TypeScript 要修正此问题，请打开您的 `tsconfig.json` 文件，并确保设置了以下选项和值：`"inlineSourceMap": true`。

## 使用 Systems Manager 自动化文档

AWS Systems Manager 让您查看和控制自己的基础架构 AWS。Systems Manager 提供了统一的用户界面，因此您可以查看来自多个 AWS 服务的操作数据，并自动执行跨 AWS 资源的操作任务。

[Systems Manager 文档](#) 定义 Systems Manager 对您的托管式实例执行的操作。自动化文档是一种 Systems Manager 文档，用于执行常见的维护和部署任务，如创建或更新亚马逊机器映像 (AMI)。本主题概述了如何使用创建、编辑、发布和删除自动化文档 AWS Toolkit for Visual Studio Code。

### 主题

- [假设和先决条件](#)
- [Systems Manager 自动化文档的 IAM 权限](#)
- [创建新的 Systems Manager 自动化文档](#)
- [打开现有的 Systems Manager 自动化文档](#)
- [编辑 Systems Manager 自动化文档](#)
- [发布 Systems Manager 自动化文档](#)
- [删除 Systems Manager 自动化文档](#)
- [删除 Systems Manager 自动化文档](#)
- [排查 Toolkit for VS Code 中 Systems Manager 自动化文档的问题](#)

## 假设和先决条件

在您开始之前，请确保：

- 您已经安装了 Visual Studio Code 和最新版本的 AWS Toolkit for Visual Studio Code。有关更多信息，请参阅 [安装 AWS Toolkit for Visual Studio Code](#)。
- 您熟悉 Systems Manager。有关更多信息，请参阅 [AWS Systems Manager 用户指南](#)。
- 熟悉 Systems Manager Automation 使用案例。有关更多信息，请参阅 AWS Systems Manager 用户指南中的 [AWS Systems Manager 自动化](#)。

## Systems Manager 自动化文档的 IAM 权限

在 Toolkit for VS Code 中，您必须拥有一个凭证配置文件，其中包含创建、编辑、发布和删除 Systems Manager 自动化文档所必需的 AWS Identity and Access Management ( IAM ) 权限。以下策略文档定义了可在委托人策略中使用的必要 IAM 权限：

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:ListDocuments",
        "ssm:ListDocumentVersions",
        "ssm:DescribeDocument",
        "ssm:GetDocument",
        "ssm:CreateDocument",
        "ssm:UpdateDocument",
        "ssm:UpdateDocumentDefaultVersion",
        "ssm>DeleteDocument"
      ],
      "Resource": "*"
    }
  ]
}
```

有关如何更新策略的信息，请参阅 IAM 用户指南中的[创建 IAM 策略](#)。有关如何设置凭证配置文件的信息，请参阅[AWS IAM 证书](#)。

## 创建新的 Systems Manager 自动化文档

您可以在 JSON 或 YAML 中使用 Visual Studio Code 创建新的自动化文档。当您创建新的自动化文档时，它将显示在无标题文件中。你可以命名你的文件并将其保存在 VS Code 中，但是文件名是不可见的 AWS。

### 创建新的自动化文档

1. 打开 VS Code。

2. 在视图菜单中，选择命令面板以打开“命令面板”。
3. 在“命令面板”中，输入 AWS Toolkit 在本地创建新的 Systems Manager 文档。
4. 为 Hello World 示例选择其中一个初学者模板。
5. 选择 JSON 或 YAML。

新的自动化文档已成功创建。

#### Note

您在 VS Code 中创建的新自动化文档不会自动显示在 AWS 中。必须先将其发布到，AWS 然后才能运行它。

## 打开现有的 Systems Manager 自动化文档

您可以使用 AWS 资源管理器查找现有的 Systems Manager 自动化文档。当您打开现有的自动化文档后，它将在 VS Code 中显示为无标题文件。

### 打开自动化文档

1. 打开 VS Code。
2. 从左侧导航栏中，选择 AWS 以打开 AWS Explorer。
3. 在 AWS 资源管理器中，对于 Systems Manager，选择要打开的文档上的下载图标，然后选择文档版本。该文件将以该版本的格式打开。或者，您也可以选择以 JSON 格式下载或以 YAML 格式下载。

#### Note

在 VS Code 中将自动化文档保存为本地文件，不会使其显示在 AWS 中。在执行 AWS 之前，需要将其发布到。

## 编辑 Systems Manager 自动化文档

如果您拥有任何自动化文档，则它们会显示在 AWS 资源管理器中 Systems Manager 文档的“归我所有”类别中。您可以拥有中已存在的自动化文档 AWS，也可以拥有之前 AWS 从 VS Code 发布到的新文档或更新的文档。

当您在 VS Code 中打开自动化文档进行编辑时，您可以利用它完成的操作会比在 AWS 管理控制台中更多。例如：

- JSON 和 YAML 格式均有架构验证。
- 文档编辑器中有一些代码段可供您创建任何自动化步骤类型。
- JSON 和 YAML 中的各种选项都支持自动完成。

### 使用版本

Systems Manager 自动化文档使用版本进行变更管理。您可以在 VS Code 中为 Systems Manager 自动化文档选择默认版本。

#### 要设置默认版本

- 在 AWS 资源管理器中，导航到要设置默认版本的文档，打开该文档的上下文（右键单击）菜单，然后选择设置默认版本。

#### Note

如果选择的文档只有一个版本，则无法更改默认版本。

## 发布 Systems Manager 自动化文档

在 VS Code 中编辑自动化文档后，可以将其发布到 AWS。

#### 发布您的自动化文档

1. 打开您想要使用[打开现有的 Systems Manager 自动化文档](#)中概述的流程进行发布的自动化文档。
2. 执行您想要发布的更改。有关更多信息，请参阅[编辑 Systems Manager 自动化文档](#)。
3. 在打开文件的右上角，选择上传图标。
4. 在发布工作流程对话框中，选择要将自动化文档发布到的 AWS 区域。

5. 如果您要发布新文档，请选择快速创建。否则，请选择快速更新以更新该 AWS 区域中的现有自动化文档。
6. 输入此自动化文档的名称。

当您将现有 Automation 文档的更新发布到 AWS，会向该文档添加一个新版本。

## 删除 Systems Manager 自动化文档

您可以删除 VS Code 中的自动化文档。删除自动化文档将删除该文档以及该文档的所有版本。

### Important

- 删除是一个无法撤消的破坏性操作。
- 删除已经运行的自动化文档不会删除启动时创建或修改的 AWS 资源。

### 删除您的自动化文档

1. 打开 VS Code。
2. 从左侧导航栏中，选择 AWS 以打开 AWS Explorer。
3. 在 AWS 资源管理器中，对于 Systems Manager，打开要删除的文档的上下文（右键单击）菜单，然后选择删除文档。

## 删除 Systems Manager 自动化文档

将 Automation 文档发布到后 AWS，您可以运行该文档以在您的 AWS 账户中代表您执行任务。要运行自动化文档，可以使用 Systems Manager APIs、AWS CLI、或 AWS Tools for PowerShell。AWS 管理控制台有关如何运行自动化文档的说明，请参阅《AWS Systems Manager 用户指南》中的[运行简单的自动化](#)。

或者，如果您想在 Systems Manager 中 AWS SDKs 使用其中一个 APIs 来运行自动化文档，请参阅[AWS SDK 参考资料](#)。

**Note**

执行自动化文档可能会在中创建新资源，AWS 并可能产生账单费用。我们强烈建议您在启动自动化文档之前先了解自动化文档将在您的账户中创建什么。

## 排查 Toolkit for VS Code 中 Systems Manager 自动化文档的问题

我已将自动化文档保存到 VS Code 中，但未在 AWS 管理控制台中看到它。

将自动化文档保存到 VS Code 中，不会将自动化文档发布到 AWS。有关发布自动化文档的更多信息，请参阅[发布 Systems Manager 自动化文档](#)。

发布自动化文档失败，出现权限错误。

确保您的 AWS 凭证配置文件具有发布 Automation 文档所需的权限。有关权限策略的示例，请参阅[Systems Manager 自动化文档的 IAM 权限](#)。

我已将我的自动化文档发布到 AWS，但我没有在中看到它 AWS 管理控制台。

请确保您已将文档发布到您正在浏览的同一 AWS 区域 AWS 管理控制台。

我已经删除了自动化文档，但仍在为其创建的资源付费。

删除自动化文档不会删除它创建或修改的资源。您可以通过[AWS 账单管理控制台](#)识别您创建的 AWS 资源，查看您的费用，然后从中选择要删除的资源。

## AWS Step Functions

借 AWS Step Functions 助，您可以创建工作流（也称为状态机）来构建分布式应用程序、实现流程自动化、协调微服务以及创建数据和机器学习管道。以下主题描述了如何在 AWS Step Functions 中使用 AWS Toolkit for Visual Studio Code。有关该 AWS Step Functions 服务的详细信息，请参阅[AWS Step Functions](#)开发人员指南。

### 主题

- [与 AWS Step Functions](#)
- [使用工作 AWS Step Functions 流工作室](#)

## 与 AWS Step Functions

以下各节介绍如何使用 AWS 工具包中包含状态机定义的 AWS Step Functions Amazon State Language (ASL) 文件。有关 AWS Step Functions 状态机的详细信息，请参阅《AWS Step Functions 开发者指南》中的“[在 Step Functions 中了解状态机](#)”主题。

### 查看 Step Functions 状态机

要在 AWS Toolkit Explorer 中查看包含状态机定义的现有 ASL 文件，请完成以下步骤。

1. 在 AWS Toolkit Explorer 中，展开包含要查看的 ASL 文件的区域。
2. 展开 Step Functions 标题。
3. 您的 ASL 文件将显示在 AWS 资源管理器中。

### 创建 Step Functions 状态机

在 AWS 工具包中，您可以根据文件创建新的 Step Functions 状态机，也可以使用模板。以下步骤介绍了如何使用文件创建 Step Functions 状态机。有关使用模板创建 SFN 状态机的详细信息，请参阅本用户指南主题中下方的状态机模板部分。

#### Note

要在 VS Code 中使用 Step Functions，包含状态机定义的 Amazon State Language (ASL) 文件的扩展名必须以 `asl.json`、`asl.yml` 或 `.asl.yaml` 结尾。默认情况下，相关的 Step Functions 文件会在 Workflow Studio 中打开。有关通过 AWS 工具包在 Workflow Studio 中[工作的详细信息](#)，请参阅本用户指南中的[使用 Workflow Studio](#) 主题。

1. 在 VS Code 的工作区中，创建新文件。
2. 为文件命名并将文件扩展名指定为 `asl.json`、`asl.yml` 或 `.asl.yaml`。
3. AWS 工具包创建后，将在 Workflow AWS Step Functions w Studio 中打开新文件。
4. 在 Workflow Studio 的实用工具菜单中，选择保存按钮，保存您的新 ASL 文件。

## 使用模板创建 Step Functions 状态机

在 AWS 工具包中，您可以根据模板创建 Step Functions 状态机。模板过程会创建一个包含状态机定义的 ASL 文件，从而为您的项目提供一个起点。以下过程介绍如何根据 AWS 工具包中的模板创建 Step Functions 状态机。

1. 在 AWS Toolkit Explorer 中，展开要在其中创建 Step Functions 状态机的区域。
2. 打开（右键单击）Step Functions 的上下文菜单，然后选择创建新的 Step Functions 状态机以在 VS Code 中打开选择初学者模板（1/2）向导。
3. 从选择初学者模板（1/2）向导中，选择 Step Functions 状态机的模板类型以继续。
4. 在选择模板格式（2/2）屏幕中，为您的模板格式选择 YAML 或 JSON。
5. 系统将在 VS Code 编辑器中打开一个包含状态机定义的新 ASL 文件。

## 下载 Step Functions 状态机

要将远程存储的 Step Functions 状态机下载到本地的 VS Code 实例，请完成以下步骤。

1. 在 AWS Toolkit Explorer 中，展开包含要下载的 Step Functions 状态机的区域。
2. 展开 Step Functions，然后右键单击要下载的 Step Functions 状态机并选择下载定义...
3. 指定存储 Step Functions 状态机的本地位置以继续。
4. 步骤完成后，Step Functions 状态机将在 Workflow Studio 中打开。

## 将更改保存到 Step Functions 状态机

以下步骤介绍了如何保存对 Step Functions 状态机的更改。

### Note

在 Workflow Studio 中进行的编辑会同步到您的本地文件，但工作保存到 VS Code 编辑器或 Workflow Studio 中之前，编辑内容将保持未保存状态。如果在打开 Workflow Studio 时修改并保存了本地文件，并且在 ASL 文件中未检测到任何错误，则保存完成后，您将在 Workflow Studio 中收到成功通知。但是，如果您的本地文件包含无效 JSON 或 YAML 并且您尝试了保存，则您的本地文件将无法同步，并且您会在 Workflow Studio 中收到警告通知。

1. 从 Workflow Studio 中打开的包含状态机定义 ASL 的文件，导航到实用工具按钮。

2. 选择保存按钮。
3. 文件保存后，VS Code 会通知您。

## 运行 Step Functions 状态机

以下过程描述了如何在 AWS 工具包中运行 Step Functions 状态机。

1. 在 AWS Toolkit Explorer 中，展开包含要运行的 Step Functions 状态机的区域。
2. 展开 Step Functions，然后右键单击要运行的 Step Functions 状态机。
3. 从上下文菜单中，选择启动执行以启动该启动流程。
4. 启动状态显示在 VS Code 的 AWS Toolkit 输出窗口中。

## 使用代码段

代码段是根据您正在处理的代码生成的自动建议。要在 Toolkit 中将代码段与 Step Functions 结合使用，请完成以下步骤。

### Note

要在 VS Code 中使用 Step Functions 代码段，包含状态机定义的 ASL 文件的扩展名必须以 `.asl.json`、`.asl.yml` 或 `.asl.yaml` 结尾。

默认情况下，您的相关 Step Functions 文件会在 Workflow Studio 中打开。

1. 在 VS Code 中，打开包含要修改的状态机定义的 ASL 文件或创建一个新 ASL 文件。
2. 在 Workflow Studio 中，如果您处于设计模式，请切换为代码模式。
3. 在 Workflow Studio 代码编辑器中，将光标放在 "States" 属性中。
4. 按下 **control + space**，打开代码段菜单，按下 **control + space** 即可访问其他属性，这些属性基于 "State" "Type"。
5. 从列表中选择所需的代码段。

## 代码验证

当您在 Workflow Studio 中处理 Step Functions 时，代码验证会主动识别错误并针对以下错误提出建议：

- 缺少属性
- 值不正确
- 非终端状态
- 指向不存在的状态

## 使用工作 AWS Step Functions 流工作室

以下各节介绍如何在中使用 AWS Step Functions Workflow Studio AWS Toolkit for Visual Studio Code。有关 AWS Step Functions Workflow Studio 的详细信息，请参阅AWS Step Functions开发人员指南中的开发[工作流程](#)主题

### 打开 Workflow Studio

以下列表介绍了可用于在 VS Code 中打开 Workflow Studio 的不同路径。

#### Note

要在 VS Code 中使用 Workflow Studio，包含状态机定义的 Amazon State Language ( ASL ) 文件的扩展名必须以 `asl.json`、`asl.yml` 或 `asl.yaml` 结尾。有关在 AWS Toolkit 中下载或创建新状态机定义的详细信息，请参阅本用户指南的“[使用](#)”AWS Step Functions主题中的“下载状态机”和“创建状态机”部分。

- 在 AWS 资源管理器中，打开（右键单击）包含状态机定义的 ASL 文件的上下文菜单，然后选择在 Workflow Studio 中打开。
- 在包含状态机定义的已打开 ASL 文件中，选择 VS Code 编辑器窗口中选项卡旁边的使用 Workflow Studio 打开图标。
- 从包含状态机定义的已打开 ASL 文件中，选择位于文件顶部的使用 Workflow Studio 打开 CodeLens 命令。
- 除非手动禁用默认 Workflow Studio，否则关闭并重新打开包含状态机定义的 ASL 文件会自动在 Workflow Studio 中重新打开该文件。

## 设计模式和代码模式

Workflow Studio 有两种模式可用于处理包含状态机定义的 ASL 文件：设计模式和代码模式。设计模式提供了一个图形界面，可在您构建原型时对工作流进行可视化。代码模式具有集成的代码编辑器，您可以在其中查看、编写和编辑工作流中的 ASL 定义。

### Note

有关设计和代码模式下每个 UI 部分的详细信息，请参阅《AWS Step Functions开发人员指南》中的[使用 Workflow Studio](#) 主题。并非所有 Workflow Studio 功能都可以在 AWS 工具包中使用，例如配置模式。

设计模式 UI 有 7 个主要部分，如下图所示的标签和描述。

1. 模式按钮：用于在设计和代码模式之间切换的按钮。
2. 实用工具按钮：一组用于执行任务的按钮，例如退出 Workflow Studio、保存工作流或将 ASL 定义导出为 JSON 或 YAML 文件。
3. 设计工具栏：包含一组用于执行常见操作的按钮（例如“撤消”、“删除”和“放大”）的工具栏。
4. 状态浏览器：包含工作流程画布 drag-and-drop 状态的浏览器。状态被整理成选项卡，并定义为操作、流程和模式。
5. 画布和工作流图：工作流的可视化渲染，您可以在其中删除、重组和选择状态进行配置。
6. 检查器面板：查看和编辑画布上所选的任何状态的属性。根据画布工作流图中选择的状态，选项卡中会填充特定于状态的配置、输入/输出、变量和错误处理选项。
7. 信息链接：要帮助时，用于打开一个包含上下文信息的面板。这些面板还包括指向《AWS Step Functions开发人员指南》中相关主题的连接。

## 在设计期间使用单状态测试

在 Workflow Studio 测试状态 UI 中，您可以测试状态机的各个状态。这包括提供状态输入、设置变量以及两者兼而有之 AWS SAM 和 CloudFormation 定义替换的能力。

要了解有关基础设施即代码 (IaC)、资源定义和数据转换的更多信息，请参阅 AWS Step Functions 开发者指南中的 [使用 AWS SAM 构建 Step Functions 工作流程](#) 和 [JSONata 在 Step Functions 中使用转换数据](#) 主题。

以下步骤介绍了如何在 Workflow Studio 中打开测试状态 UI。

### 打开测试状态 UI

1. 在 Workflow Studio 的设计模式选项卡中，导航到画布并选择一种状态以在检查器面板中将其打开。
2. 在检查器面板中，选择测试状态按钮。
3. 测试状态 UI 将在 VS Code 中打开。

测试状态 UI 有 3 个主要选项卡：测试输入、参数和输出、状态定义。“测试输入”选项卡还有 3 个其他字段，允许您从 AWS SAM 或 CloudFormation 模板中提供状态输入、设置变量和指定定义替换。在状

态定义选项卡中，您可以调整 workflow 并重新测试。运行完测试后，可以应用和保存对状态机定义的改变。

以下屏幕截图显示的是测试状态 UI，其中包括主题资源定义。

**Test state**

Test a state in isolation using the TestState API to ensure that it works correctly. [Learn more](#)

**Test input** | Arguments & Output | State definition

**Execution role**  
Testing a state requires an execution role. Enter an IAM role ARN, select an existing IAM role from the list, or [learn how to create a new IAM role with permissions for an optimized service integration](#)

Admin

**Definition substitutions**  
Enter values for any definition substitutions.

Key	Value
#{topic}	arn:aws:sns:ca-central-1:652323157371:mySnsT...

**State input - optional**  
Enter input values for this state.

```
1 {
  "key": "value"
}
```

Must be in valid JSON format.

**Variables - optional**  
Enter values for any variables referenced.

```
1 {
  "variableName": "value"
}
```

Must be in valid JSON format.

**Start test**

**Basic** | **Advanced**

**Task state request**  
Request that will be sent to the Task state.

```
1 Start a test to view the output.
```

**Task state response**  
Response received from the Task state.

```
1 Start a test to view the output.
```

**State output**  
Output that will be passed to the next state.

```
1 Start a test to view the output.
```

**Variables**  
Variables at end of test.

```
1 Start a test to view the output.
```

[Copy TestState API response](#) [Apply changes and close](#)

## 默认禁用 Workflow Studio

默认情况下，Workflow Studio 是包含状态机定义的 ASL 文件的默认编辑器。您可以通过修改本地 `.vscode` 目录中的 `settings.json` 文件来禁用默认设置。如果您默认禁用 Workflow Studio，则仍可通过本主题的打开 Workflow Studio 部分中列出的方法对其进行访问。

要在 VS Code 中编辑 `settings.json` 文件，请完成以下步骤。

1. 在 VS Code 中，按下 **option+shift+p** ( Mac ) 或 **ctrl+shift+p** ( Windows ) 打开命令面板。
2. 在 VS Code 命令面板的搜索字段中，输入 **Open User Settings (JSON)**，并在其出现在列表时选择它。
3. 在编辑器的 `settings.json` 中，将以下修改添加到文件。

```
    {
      "workbench.editorAssociations": {
        // Use all the following overrides or a specific one for a
        certain file type
        "*.asl.json": "default",
        "*.asl.yaml": "default",
        "*.asl.yml": "default"
      }
    }
  }
```

4. 保存您对 `settings.json` 的更改，然后刷新或重启 VS Code。

## 使用 Threat Composer

您可以通过 AWS Toolkit for Visual Studio Code 来使用 Threat Composer 工具。Threat Composer 是一款威胁建模工具，可以简化威胁建模过程。

有关 Threat Composer 工具的详细信息，请参阅 [Threat Composer GitHub 存储库](#)。

以下主题介绍了如何在 AWS Toolkit for Visual Studio Code 中使用 Threat Composer。

### 主题

- [通过 Toolkit 使用 Threat Composer](#)

## 通过 Toolkit 使用 Threat Composer

借助 Threat Composer，您可以直接在 VS Code 中创建、查看和编辑 Threat Composer 威胁模型。有关 Threat Composer 工具的详细信息，请参阅 [Threat Composer GitHub 存储库](#)。

以下各部分介绍了如何在 AWS Toolkit for Visual Studio Code 中访问 Threat Composer 工具。

### 通过 Toolkit 访问 Threat Composer

通过 Toolkit 访问 Threat Composer 的主要方法有 3 种。

通过现有威胁模型访问 Threat Composer

要打开 Threat Composer，请在 VS Code 中打开现有的威胁模型文件（扩展名为 `.tc.json`）。Threat Composer 将自动打开，并在 VS Code 编辑器窗口中直观呈现您的威胁模型文件。

### 创建新的 Threat Composer 威胁模型

1. 在 VS Code 主菜单中，展开文件，然后选择新建文件。
2. 在新建文件对话框中，选择 Threat Composer 文件...。
3. 出现提示时，输入 file name，然后按 **enter** 键打开 Threat Composer，并在新的 VS Code 编辑器窗口中创建空威胁模型文件的可视化图表。

### 通过命令面板创建新的 Threat Composer 威胁模型

1. 在 VS Code 中，按下 **Cmd + Shift + P**（在 Windows 中，则需按下 **Ctrl + Shift + P**）以打开命令面板。
2. 在搜索字段中，输入 **Threat Composer**，然后在搜索结果中出现创建新的 Threat Composer 文件时选择它。
3. 出现提示时，输入 file name，然后按 **enter** 键打开 Threat Composer，并在新的 VS Code 编辑器窗口中创建空威胁模型文件的可视化图表。

## 使用资源

除了访问资源 AWS 管理器中默认列出的 AWS 服务外，您还可以转到“资源”，然后从数百种资源中进行选择，以添加到界面中。在中 AWS，资源是您可以使用的实体。可以添加的一些资源包括亚马逊 AppFlow、Amazon Kinesis Data Streams AWS、IAM 角色、亚马逊 VPC 和 CloudFront 亚马逊分配。

进行选择后，您可以前往资源，展开资源类型，以列出该类型的可用资源。例如，如果选择 `AWS Toolkit:Lambda::Function` 资源类型，则可以访问用于定义不同函数、其属性和特性的资源。

将资源类型添加到 Resources（资源）后，您可以通过以下方式与它及其资源进行交互：

- 查看当前 AWS 区域中可用于该资源类型的现有资源列表。
- 查看用于描述资源的 JSON 文件的只读版本。
- 复制资源的资源标识符。
- 查看说明资源类型用途和资源建模架构（采用 JSON 和 YAML 格式）的 AWS 文档。

- 通过编辑和保存符合架构的 JSON 格式模板来创建新资源。\*
- 更新或删除现有资源。\*

### ⚠ Important

\* 在当前版本 AWS Toolkit for Visual Studio Code 中，创建、编辑和删除资源的选项是一项实验性功能。由于实验性功能仍有待测试和更新，可能存在可用性问题。实验性功能可能会在 AWS Toolkit for Visual Studio Code 不另行通知的情况下从中删除。

要允许对资源使用实验性功能，请在 VS Code IDE 中打开设置窗格，然后展开扩展并选择 AWS Toolkit。

在 AWS Toolkit 实验下，选择 jsonResourceModification 允许您创建、更新和删除资源。

有关更多信息，请参阅 [使用实验性功能](#)。

## 用于访问资源的 IAM 权限

您需要特定的 AWS Identity and Access Management 权限才能访问与 AWS 服务关联的资源。例如，IAM 实体（如用户或角色）需要 Lambda 权限才能访问 `AWS Toolkit::Lambda::Function` 资源。

除了服务资源的权限外，IAM 实体还需要权限才能允许 Toolkit for VS Code 代表其调用 AWS Cloud Control API 操作。Cloud Control API 操作允许 IAM 用户或角色访问和更新远程资源。

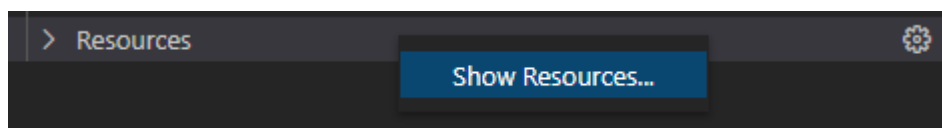
授予权限的最简单方法是使用 Toolkit 接口将 AWS 托管策略附加到调用这些 API 操作的 IAM 实体。PowerUserAccess 此 [托管策略](#) 将授予执行应用程序开发任务的一系列权限，包括调用 API 操作。

有关定义远程资源上允许的 API 操作的特定权限，请参阅 [AWS Cloud Control API 用户指南](#)。

## 添加现有资源并与之交互

1. 在 AWS Explorer 中，右键单击资源，然后选择显示资源。

此时会出现一个窗格，显示可供选择的资源类型列表。



2. 在选择面板中，选择要添加到 AWS Explorer 中的资源类型，然后按下返回或选择确认以进行确认。

您选择的资源类型将列在资源下方。

#### Note

如果您已经将资源类型添加到 AWS Explorer，然后清除该类型的复选框，则您选择确认后，相应类型将不再列于资源下方。只有当前选定的资源类型才会显示在 AWS Explorer 中。

3. 要查看某种资源类型已存在的资源，请展开该类型的条目。

可用资源列表显示在其资源类型下方。

4. 要与特定资源交互，请右键单击其名称，然后选择以下选项之一：

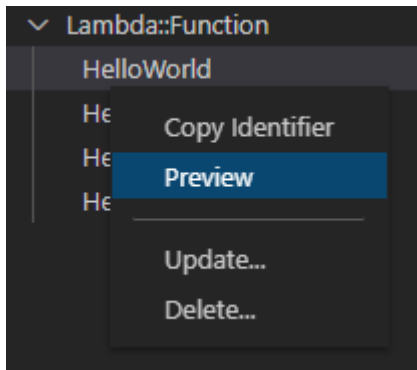
- 复制资源标识符：将特定资源的标识符复制到剪贴板。（例如，可以使用 `TableName` 属性来标识 `AWS Toolkit:DynamoDB::Table` 资源。）
- Preview（预览）：查看用于描述资源的 JSON 格式模板的只读版本。

资源模板显示后，您可以通过选择编辑器选项卡右侧的更新图标来对其进行修改。要更新资源，您必须启用所需的 [???](#)。

- 更新：在 VS Code 编辑器中编辑资源的 JSON 格式模板。有关更多信息，请参阅 [创建和编辑资源](#)。
- 删除：通过在显示的对话框中确认删除操作来删除资源。（在此版本 [???](#) 中，删除资源目前是一项操作 AWS Toolkit for Visual Studio Code。）

#### Warning

如果您删除资源，则使用该资源的任何 AWS CloudFormation 堆栈都将无法更新。要修复此更新失败，您需要重新创建资源或删除堆栈 CloudFormation 模板中对其的引用。有关更多信息，请参阅这篇 [知识中心文章](#)。



## 创建和编辑资源

### Important

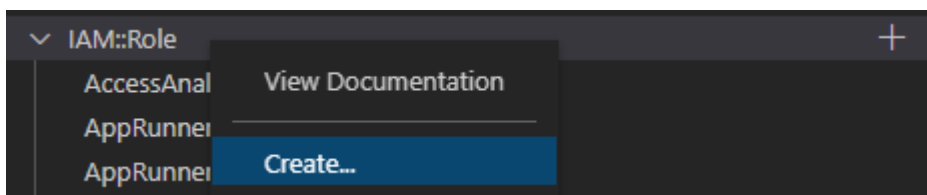
目前，资源的创建和更新在 AWS Toolkit for Visual Studio Code 的此版本中是 [???](#)。

创建新资源涉及将资源类型添加到资源列表，然后编辑 JSON 格式的模板，该模板可定义资源及其属性和特性。

例如，属于该资源类型的 `AWS Toolkit:SageMaker::UserProfile` 资源是使用为 Amazon A SageMaker I Studio 创建用户配置文件的模板定义的。定义此用户配置文件资源的模板必须符合 `AWS Toolkit:SageMaker::UserProfile` 的资源类型架构。如果由于属性缺失或不正确等原因，模板不符合架构，则无法创建或更新资源。

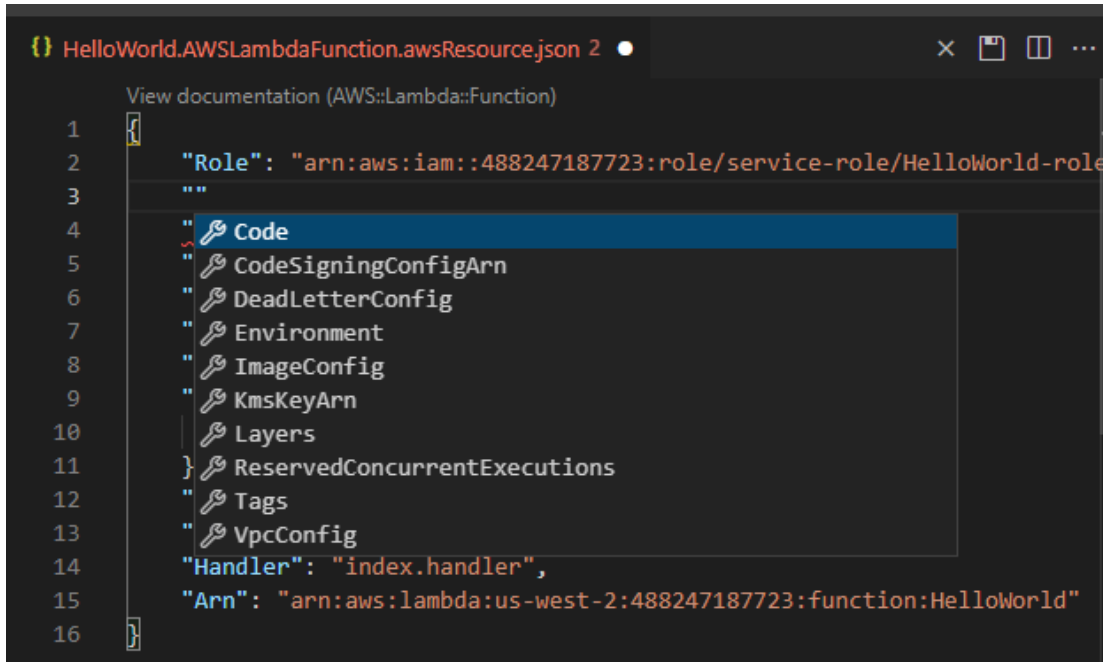
1. 要为您要创建的资源添加资源类型，请右键单击资源，然后选择显示资源。
2. 在资源下添加资源类型后，选择加号 (“+”) 图标，以在新编辑器中打开模板文件。

或者，您可以右键单击资源类型的名称并选择创建。您还可以通过选择查看文档，访问有关如何对资源进行建模的信息。



3. 在编辑器中，首先定义构成资源模板的属性。自动完成功能会建议符合模板架构的属性名称。当您悬停在某个属性类型上时，一个窗格会随即显示，其中显示了有关其用途的描述。有关架构的详细信息，请选择查看文档。

任何不符合资源架构的文本都用红色波浪下划线表示。



4. 声明完资源后，选择“保存”图标以验证您的模板并将资源保存到远程 AWS 云端。

如果您的模板根据其架构定义资源，系统会显示一条消息，确认资源已创建。（如果资源已经存在，则消息将确认资源已更新。）

资源创建完成后，会添加至资源类型标题下的列表中。

5. 如果您的文件包含错误，系统会显示一条消息，说明无法创建或更新资源。打开查看日志以确定需要修复的模板元素。

# 疑难解答 AWS Toolkit for Visual Studio Code

以下各节包含有关 AWS Toolkit for Visual Studio Code 和使用该工具包中的 AWS 服务的一般疑难解答信息。有关与解决 AWS Toolkit 中的 SAM 问题具体相关的问题，请参阅本用户指南中的[无服务器应用程序疑难解答](#)主题。

## 主题

- [问题排查最佳实践](#)
- [配置文件 ... 在配置文件中找不到](#)
- [SAM json 架构：无法更改 template.yaml 文件中的架构](#)

## 问题排查最佳实践

以下是疑难解答 AWS Toolkit for Visual Studio Code 问题时推荐的最佳做法。有关如何为做出贡献的详细信息 AWS Toolkit for Visual Studio Code，请参阅 AWS Toolkit for Visual Studio Code GitHub 存储库中的[参与 AWS Toolkit for Visual Studio Code](#)主题。

- 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。
- 收集您的 AWS 工具包调试日志。有关如何查找 AWS Toolkit 调试日志的详细说明，请参阅本用户指南主题中的“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 Code GitHub 存储库的[“AWS Toolkit for Visual Studio Code 问题”](#)部分报告未解决的问题。

### Note

以下过程介绍如何查看 AWS Toolkit 调试日志。与查看 Amazon Q 调试日志的过程完全相同，唯一的区别是您从 VS Code 命令面板中选择 Amazon Q：查看日志。

## 如何找到你的 AWS Toolkit for Visual Studio Code 调试日志

1. 在 VS Code 中，按 **Cmd + Shift + P** 或 **Ctrl + Shift + P** (Windows) 打开命令面板，然后在搜索字段中输入 **AWS View Logs**。
2. 选择“AWS 查看日志”，在 VS Code 终端输出窗口中打开您的 AWS Toolkit 日志。

3. 在 VS Code 终端输出窗口中，展开齿轮图标菜单并选择调试。
4. 再次展开齿轮图标菜单，然后选择设为默认。
5. Re-open 按 **Cmd + Shift + P** 或 **Ctrl + Shift + P** (Windows) 进入命令面板，然后搜索 **Reload Window**，然后选择“开发人员：重新加载窗口”。
6. VS Code 会重新加载，VS Code 终端输出窗口会显示您更新的 AWS Toolkit 调试日志。

## 配置文件 ... 在配置文件中找不到

### 问题

#### Note

此问题仅适用于 `~/.aws/config` 文件而不适用于 `~/.aws/credentials` 文件。有关 AWS 配置和 AWS 凭据文件的详细信息，请参阅 AWS SDK 和工具参考指南中的共享配置和[凭据文件](#)主题。

选择凭据时，AWS Toolkit 日志会显示一条具有以下结构的消息：`Profile name could not be found in shared credentials file.`

以下是该错误在 AWS Toolkit 日志中的示例：

```
2023-08-08 18:20:45 [ERROR]: _aws.auth.reauthenticate: Error: Unable to
authenticate connection
-> CredentialsProviderError: Profile vscode-prod-readonly could not be found
in shared credentials file.
```

### 解决方案

如果您的配置文件已存在于 `~/.aws/config` 中，请检查其开头是否为 `[profile`。以下是结构正确的用户配置文件示例：

```
[profile example]
region=us-west-2
credential_process=...
```

以下是结构不正确的用户配置文件示例：

```
[example]
region=us-west-2
credential_process=...
```

## SAM json 架构：无法更改 template.yaml 文件中的架构

### 问题

您无法在 SAM template.yaml 中手动选择不同的 json 架构

### 解决方案

更新至 vscode-yaml 版本 1.11+ 后，您可以在 YAML 文件的顶部添加一个 **yaml-language-server** 模式行来强制通过 URI 使用架构。有关在 Redhat 的 [开发者存储库的 yaml 语言服务器主题中使用内联架构](#) 部分的更多信息。GitHub 以下是 **yaml-language-server** 模式行的示例。

```
# yaml-language-server: $schema=https://raw.githubusercontent.com/aws/serverless-application-model/main/samtranslator/schema/schema.json
```

## 中的安全性 AWS Toolkit for VS Code

云安全 AWS 是重中之重。作为 AWS 客户，您可以受益于专为满足大多数安全敏感型组织的要求而构建的数据中心和网络架构。

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 负责保护在云中运行 AWS 服务的基础架构 AWS Cloud。AWS 还为您提供可以安全使用的服务。Third-party 作为[AWS 合规计划](#)的一部分，审计师定期测试和验证我们安全的有效性。要了解适用于 AWS Toolkit for VS Code 的合规计划，请参阅[AWS 按合规计划划分的范围内 AWS 服务按合规计划](#)。
- 云端安全-您的责任由您使用的 AWS 服务决定。您还需要对其它因素负责，包括您的数据的敏感性、您的公司的要求以及适用的法律法规。

本文档可帮助您了解在使用 Toolkit for VS Code 时如何应用责任共担模型。以下主题向您介绍如何配置 VS Code 的 Toolkit 以满足您的安全和合规目标。您还将学习如何使用其他 AWS 服务来帮助您监控和保护您的 Toolkit for VS Code 资源。

### 主题

- [中的数据保护 AWS Toolkit for VS Code](#)

## 中的数据保护 AWS Toolkit for VS Code

责任共担模型 AWS [分](#)适用于 AWS Toolkit for VS Code 中的数据保护。如本模型所述 AWS，负责保护运行所有内容的全球基础架构 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[数据隐私常见问题解答](#)[AWS](#)条款。有关欧洲数据保护的信息，请参阅[通用数据保护条例 \(GDPR\) 中心](#)。

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）。这包括当你使用控制台、API 或 SDK AWS 服务使用 Toolkit for VS Code 或其他 AWS 工具包时。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。

# 的文档历史记录 AWS Toolkit for Visual Studio Code 用户指南

下表介绍每一个 AWS Toolkit for Visual Studio Code 发行版中的重大更改。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
<a href="#">亚马逊 SageMaker 联合工作室</a>	与亚马逊 SageMaker Unified Studio 服务集成。	2025 年 9 月 18 日
<a href="#">LocalStack</a>	添加了新的《用户指南》主题以支持发布 LocalStack。	2025 年 9 月 11 日
<a href="#">使用 AWS Lambda 函数</a>	更新用户指南主题，在工具包中加入更新后的 Lambda 功能。	2025 年 7 月 17 日
<a href="#">AWS Lambda 远程调试</a>	在《AWS Toolkit for Visual Studio Code 用户指南》中添加了新的 AWS Lambda 远程调试主题。	2025 年 7 月 17 日
<a href="#">AWS Lambda 控制台到 IDE</a>	在《AWS Toolkit for Visual Studio Code 用户指南》中添加了 IDE 的新 AWS Lambda console 主题。	2025 年 7 月 17 日
<a href="#">AWS Step Functions 内容更新并增加了对 Workflow Studio 的支持</a>	为 Workflo AWS Step Functions w Studio 的现有内容 AWS Step Functions 和用户指南主题添加了更新，以支持功能发布。	2025 年 3 月 6 日
<a href="#">AWS 无服务器土地</a>	在 Ap AWS p 开发器目录中添加了新的 AWS 无服务器版块主题。	2025 年 3 月 6 日

<a href="#">更新防火墙和网关以允许访问</a>	必须允许列出的终端节点和资源列表，才能访问 AWS Toolkit for Visual Studio Code 和 Amazon Q for VS Code 扩展中的所有服务和功能。	2025 年 2 月 28 日
<a href="#">支持 Amazon ECR App Runner</a>	在 AWS 工具包中增加了对从 Amazon 弹性容器注册表节点启动 AWS App Runner 服务的文档支持。	2025 年 2 月 6 日
<a href="#">Amazon DocumentDB</a>	在 AWS Toolkit for Visual Studio Code 用户指南中添加了新的亚马逊 DocumentDB 主题。	2025 年 2 月 6 日
<a href="#">EC2 支持</a>	Toolkit 新增了对 Amazon Elastic Compute Cloud 服务的支持。	2025 年 1 月 31 日
<a href="#">AWS Documents</a>	为“AWS 文档”添加了新的用户指南主题。	2025 年 1 月 20 日
<a href="#">亚马逊 CloudWatch 日志 Live Tail</a>	在中添加了新的副主题以支持 Amazon L CloudWatch ogs Live Tail 功能。AWS Toolkit for Visual Studio Code	2024 年 12 月 15 日
<a href="#">AWS 应用程序生成器</a>	在《AWS Toolkit for Visual Studio Code 用户指南》中添加了新的“AWS 应用程序生成器”主题。	2024 年 10 月 30 日
<a href="#">基础架构编辑器</a>	AWS 应用程序编排器现在是 AWS 基础架构编排器。	2024 年 10 月 3 日

<a href="#">AWS 身份和访问管理 (IAM) Access Management 访问分析器更新</a>	更新了 IAM 访问权限分析器内容，增加了新的 API 参考。	2024 年 7 月 10 日
<a href="#">AWS 身份和访问管理 (IAM) Access Management 访问分析器</a>	新增了关于 IAM 访问权限分析器的用户指南主题。	2024 年 5 月 23 日
<a href="#">Connect 到 AWS 授权流程已更新</a>	更新了授权流程，以反映对身份验证流程的更改以及 Amazon Q 与 AWS Toolkit for Visual Studio Code 的隔离。	2024 年 4 月 30 日
<a href="#">适用于 VS Code 的 Amazon Q 扩展程序</a>	截至 2024 年 4 月 30 日，CodeWhisperer 现已成为亚马逊 Q 的一部分，亚马逊 Q 已作为 VS Code 的扩展提供。	2024 年 4 月 30 日
<a href="#">支持开发环境中的虚拟私有云</a>	更新了内容，其中包含支持开发环境中 VPC 的 UI 更改。	2024 年 1 月 21 日
<a href="#">基础架构编辑器</a>	在《AWS Toolkit for Visual Studio Code 用户指南》中添加了新的基础设施编排器主题。	2023 年 11 月 28 日
<a href="#">单点登录支持 CodeCatalyst</a>	更新了内容，涵盖了 IAM 身份中心对开发环境的 CodeCatalyst 支持。	2023 年 11 月 17 日
<a href="#">添加了 VS Code 和 Toolkit 下载链接</a>	更新了内容，其中包含 VS Code 和 AWS Toolkit for Visual Studio Code 的下载链接。	2023 年 11 月 1 日

<a href="#">Amazon Redshift 主题</a>	在《AWS Toolkit for Visual Studio Code 用户指南》中添加了新的 Amazon Redshift 主题。	2023 年 10 月 17 日
<a href="#">Connect 到 AWS 授权流程已更新</a>	更新了授权流程，将重点放在服务特定的身份验证方法上。	2023 年 9 月 29 日
<a href="#">已创建用户指南：创建模板 CloudFormation</a>	创建了新的用户指南，描述了如何使用 Toolkit for VS Code 创建 CloudFormation 模板	2021 年 12 月 17 日
<a href="#">次要 UI 更新</a>	将现有文本“预览机器状态”更新为“呈现图表”，以便更好地匹配 UI。	2021 年 12 月 14 日
<a href="#">为 Amazon Elastic Container Service Exec 创建了用户指南</a>	这是 Amazon ECS Exec 的概述。	2021 年 12 月 13 日
<a href="#">为适用于 VS Code 的 AWS IoT Toolkit 服务创建了用户指南</a>	本用户指南旨在帮助您开始使用适用于 VS Code 的 Toolkit AWS IoT 服务。	2021 年 11 月 22 日
<a href="#">支持实验性功能</a>	增加了对开启 AWS 服务实验性功能的支持。	2021 年 10 月 14 日
<a href="#">Support 对 AWS 资源的支持</a>	增加了对访问资源类型的支持以及用于创建、编辑和删除资源的界面选项。	2021 年 10 月 14 日
<a href="#">Amazon ECR 服务概述 AWS Toolkit for Visual Studio Code</a>	增加了可在 VS Code 中访问的 Amazon ECR 服务的功能概述和演练	2021 年 10 月 14 日
<a href="#">支持 ARM64 环境</a>	现在，您可以在 ARM64-based 模拟环境和基于 x86_64 的环境中运行无服务器应用程序。	2021 年 10 月 1 日

<a href="#">AWS Serverless Application</a>	增加了对在 ARM64 平台上运行 AWS SAM 应用程序的支持	2021 年 9 月 30 日
<a href="#">格式更新 Node.js 部分</a>	根据客户的反馈，更新了 Node.js / 的格式 TypeScript。	2021 年 8 月 12 日
<a href="#">App Runner 支持</a>	增加了对 AWS App Runner 的支持 AWS Toolkit for Visual Studio Code。	2021 年 8 月 11 日
<a href="#">调试 Go 函数</a>	增加了调试本地 Go 函数的支持。	2021 年 5 月 10 日
<a href="#">调试 Java 函数</a>	增加了调试本地 Java 函数的支持。	2021 年 4 月 22 日
<a href="#">YAML 支持 AWS Step Functions</a>	增加了对 YAML 的支持。AWS Step Functions	2021 年 3 月 4 日
<a href="#">调试 Amazon API Gateway 资源</a>	增加了调试本地 Amazon API Gateway 资源的支持	2020 年 12 月 1 日
<a href="#">Amazon API Gateway</a>	增加了对 Amazon API Gateway 的支持。	2020 年 12 月 1 日
<a href="#">AWS Serverless Application</a>	增加了将 Lambda 容器映像与无服务器应用程序结合使用的支持	2020 年 12 月 1 日
<a href="#">AWS Systems Manager 支持</a>	增加了 Systems Manager 自动化文档的支持。	2020 年 9 月 30 日
<a href="#">CloudWatch 日志</a>	增加了对 CloudWatch 日志的支持。	2020 年 8 月 24 日
<a href="#">Amazon S3</a>	增加了对 Amazon S3 的支持。	2020 年 7 月 30 日
<a href="#">AWS Step Functions 支持</a>	增加了对的支持 AWS Step Functions。	2020 年 3 月 31 日

<a href="#">使用 Amazon EventBridge 架构</a>	增加了对 Amazon EventBridge 架构的支持	2019 年 12 月 1 日
<a href="#">AWS CDK</a>	该 AWS CDK 服务的预览版。	2019 年 11 月 25 日
<a href="#">使用外部凭证流程</a>	添加了有关使用外部凭证流程来获取 AWS 凭证的信息。	2019 年 9 月 25 日
<a href="#">IntelliSense 用于任务定义文件</a>	IntelliSense 增加了对使用 Amazon ECS 任务定义文件的支持。	2019 年 9 月 24 日
<a href="#">的用户指南 AWS Toolkit for Visual Studio Code</a>	公开发行版本。	2019 年 7 月 11 日
<a href="#">的用户指南 AWS Toolkit for Visual Studio Code</a>	更新了文档结构，以使其更加清晰易用。	2019 年 7 月 3 日
<a href="#">正在安装 AWS Toolkit for Visual Studio Code</a>	添加了有关安装语言开发工具包以支持各种工具链的信息。	2019 年 6 月 12 日
<a href="#">配置工具链</a>	添加了有关配置各种工具链的信息。	2019 年 6 月 12 日
<a href="#">首次发布</a>	AWS Toolkit for Visual Studio Code 用户指南的初始版本。	2019 年 3 月 28 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。