用户指南

AWS 应用程序工作室

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS 应用程序工作室: 用户指南

Copyright © 2025 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

什么是 AWS App Studio？ 1
你是首次使用 App Studio 的用户吗？ 1
概念 2
管理员角色 2
应用程序（应用程序） 2
自动化 3
自动化操作 3
建造者角色 3
组件 3
Connector 3
开发环境 3
实体 4
实例 4
页面 4
触发器 4
应用工作室的工作原理 5
将您的应用程序连接到其他服务 6
配置应用程序的数据模型 7
构建应用程序的用户界面 7
实现应用程序的逻辑或行为 9
应用程序的开发生命周期 11
了解更多 11
设置并登录 App Studio 13
首次创建和设置 App Studio 实例 13
注册一个 AWS 账户 13
创建管理用户来管理 AWS 资源 13
在中创建 App Studio 实例 AWS Management Console 14
接受加入 App Studio 的邀请 18
入门 19
教程：使用 AI 生成应用程序 19
先决条件 20
第 1 步：创建 应用程序 20
第 2 步：探索您的新应用程序 21
第 3 步：预览您的应用程序 22

后续步骤 ... 23
教程：从一个空的应用程序开始构建 ... 23
先决条件 ... 25
第 1 步：创建 应用程序 ... 25
第 2 步：创建一个实体来定义应用程序的数据 ... 26
第 3 步：设计用户界面 (UI) 和逻辑 ... 28
步骤 4：预览应用程序 ... 31
步骤 5：将应用程序发布到测试环境 ... 31
后续步骤 ... 32
管理员文档 ... 33
使用群组和角色管理用户访问权限 ... 33
角色和权限 ... 33
查看群组 ... 34
添加用户或群组 ... 34
更改群组的角色 ... 35
移除用户或群组 ... 36
使用连接器连接到其他服务 ... 36
Connect 连接到 AWS 服务 ... 37
Connect 连接到第三方服务 ... 77
查看、编辑和删除连接器 ... 84
删除 App Studio 实例 ... 85
生成器文档 ... 86
教程 ... 86
使用 Amazon Bedrock 构建文本摘要器应用程序 ... 86
与亚马逊 S3 互动 ... 93
调用 Lambda 函数 ... 101
使用生成式 AI 构建您的应用程序 ... 103
正在生成您的应用程序 ... 103
构建或编辑您的应用程序 ... 104
生成您的数据模型 ... 104
生成样本数据 ... 104
为 AWS 服务配置操作 ... 104
嘲笑的回应 ... 105
在建造时向 AI 寻求帮助 ... 105
创建、编辑和删除应用程序 ... 105
创建 应用程序 ... 105

导入应用程序 106
复制应用程序 110
编辑或构建应用程序 110
编辑之前发布的应用程序版本 111
重命名应用程序 112
删除 应用程序 112
预览、发布和共享应用程序 112
预览应用程序 113
发布应用程序 113
共享已发布的应用程序 117
回滚到之前发布的版本 118
导出应用程序 119
页面和组件：构建应用程序的用户界面 120
管理页面 120
管理组件 122
配置基于角色的页面可见性 124
在应用程序导航中对页面进行排序和整理 125
使用应用程序主题更改应用程序中的颜色 126
组件参考 127
自动化和操作：定义应用程序的业务逻辑 167
自动化概念 168
创建、编辑和删除自动化 169
添加、编辑和删除自动化操作 170
自动化操作参考 172
实体和数据操作：配置应用程序的数据模型 188
设计数据模型时的最佳实践 189
创建实体 190
配置实体 192
删除实体 199
托管数据实体 200
页面和自动化参数 201
页面参数 201
自动化参数 202
JavaScript 用于编写表达式 206
基本语法 206
插值 207

联接 207
日期和时间 207
代码块 208
全局变量和函数 208
引用或更新 UI 组件值 208
处理表格数据 210
访问自动化 211
数据依赖关系和时序注意事项 213
示例：订单详情和客户信息 213
数据依赖和计时最佳实践 213
构建包含多个用户的应用程序 214
邀请构建者编辑应用程序 215
正在尝试编辑正在由其他用户编辑的应用程序 215
更新应用程序的内容安全设置 216
问题排查和调试 218
设置、权限和入职 218
选择 “为我创建账户实例” 选项时 App Studio 设置失败 218
设置后无法访问 App Studio 218
不确定登录 App Studio 时要使用什么用户名或密码 218
设置 App Studio 时出现系统错误 219
我找不到我的 App Studio 实例网址 219
我无法在 App Studio 中修改群组或角色 219
如何退出 App Studio 218
对应用程序进行故障排除和调试 220
AI 生成器助手 220
在应用工作室里 220
预览应用程序 221
在测试环境中 222
使用登录 CloudWatch 223
连接器 225
发布和共享应用程序 228
我在 “共享” 对话框中看不到新创建的应用程序角色 228
我的应用程序发布完成后我没有收到电子邮件 228
我的应用程序的最终用户无法访问已发布的应用程序 228
安全性 229
安全注意事项和缓解措施 229

安全性注意事项 229
安全风险缓解建议 230
数据保护 230
数据加密 231
传输中加密 232
密钥管理 232
互联网络流量隐私 232
App Studio 和身份与访问管理 232
基于身份的策略 234
基于资源的策略 234
策略操作 235
策略资源 236
策略条件键 236
ACLs 236
ABAC 237
临时凭证 237
主体权限 237
服务角色 237
服务相关角色 238
AWS 托管策略 238
服务相关角色 242
基于身份的策略示例 244
合规性验证 247
恢复能力 248
基础设施安全性 248
配置和漏洞分析 248
防止跨服务混淆座席 249
跨区域数据传输 250
支持的浏览器 251
支持和推荐的用于构建应用程序的浏览器 251
应用程序最终用户支持和推荐的浏览器 251
更新浏览器设置以在 App Studio 上构建应用程序 251
限额 253
文档历史记录 254
..... cclxii

什么是 AWS App Studio？

AWS App Studio 是一项由人工智能驱动的生成式服务，它使用自然语言来帮助您创建企业级应用程序。App Studio 向没有软件开发技能的技术专业人员（例如 IT 项目经理、数据工程师和企业架构师）开放应用程序开发。借助 App Studio，您无需操作专业知识即可快速构建安全且完全由 AWS其管理的应用程序。

开发者可以使用 App Studio 创建和部署应用程序，以实现内部业务流程的现代化。一些用例包括库存管理和跟踪、索赔处理和复杂的审批，以提高员工工作效率和客户成果。

主题

- 你是首次使用 App Studio 的用户吗？

你是首次使用 App Studio 的用户吗？

如果您是首次使用 App Studio 的用户，我们建议您先阅读以下章节：

- 对于具有管理员角色且将设置 App Studio、管理用户和访问权限以及配置与其他 AWS 或第三方服务的连接器的用户，请参阅AWS 应用工作室的概念和设置并登录 AWS App Studio。
- 对于将要创建和开发应用程序的构建者，请参阅AWS 应用工作室的概念和AWS App Studio 入门。

AWS 应用工作室的概念

熟悉 App Studio 的关键概念，以帮助加快团队创建应用程序和自动化流程。这些概念包括整个 App Studio 中针对管理员和构建者使用的术语。

主题

- 管理员角色
- 应用程序（应用程序）
- 自动化
- 自动化操作
- 建造者角色
- 组件
- Connector
- 开发环境
- 实体
- 实例
- 页面
- 触发器

管理员角色

管理员是一个可以在 App Studio 中分配给群组的角色。管理员可以在 App Studio 中管理用户和群组、添加和管理连接器以及管理构建器创建的应用程序。此外，具有管理员角色的用户拥有构建者角色中包含的所有权限。

只有具有 “管理员” 角色的用户才能访问 Admin Hub，其中包含用于管理角色、数据源和应用程序的工具。

应用程序（应用程序）

应用程序（应用程序）是为最终用户开发的用于完成特定任务的单个软件程序。App Studio 中的应用程序包括用户界面页面和组件、自动化以及用户可以与之交互的数据源等资产。

自动化

自动化是您定义应用程序业务逻辑的方式。自动化的主要组成部分是：启动自动化的触发器、一个或多个操作的序列、用于向自动化传递数据的输入参数以及输出。

自动化操作

自动化操作，通常称为操作，是构成自动化的单个逻辑步骤。每个操作都会执行特定的任务，无论是发送电子邮件、创建数据记录、调用 Lambda 函数还是调用。 APIs操作通过操作库添加到自动化中，并且可以分组为条件语句或循环。

建造者角色

生成器是一个可以在 App Studio 中分配给群组的角色。构建者可以创建和构建应用程序。构建器无法管理用户或群组、添加或编辑连接器实例，也无法管理其他构建器的应用程序。

具有 Builder 角色的用户可以访问 Builder Hub，其中包含有关资源的详细信息，例如构建者可以访问的应用程序，以及学习资源等有用信息。

组件

组件是应用程序用户界面中的单个功能项。组件包含在页面中，有些组件可以用作其他组件的容器。组件将用户界面元素与您希望该用户界面元素执行的业务逻辑相结合。例如，一种类型的组件是表单，用户可以在其中输入字段中的信息，提交后，该信息将作为数据库记录添加。

Connector

连接器是 App Studio 与其他 AWS 服务（例如 AWS Lambda 和 Amazon Redshift）或第三方服务之间的连接。创建并配置连接器后，构建者可以在其应用程序中使用该连接器及其连接到 App Studio 的资源。

只有具有管理员角色的用户才能创建、管理或删除连接器。

开发环境

开发环境是用于构建应用程序的可视化工具。此环境包括以下用于构建应用程序的选项卡：

- 页面：构建者使用页面和组件设计应用程序的地方。
- 自动化：构建者使用自动化来设计其应用程序的业务逻辑。
- 数据：构建者使用实体设计应用程序数据模型的地方。

开发环境还包含一个调试控制台和一个 AI 聊天窗口，用于在构建时获取上下文帮助。构建者可以从开发环境中预览其正在进行的应用程序。

实体

实体是 App Studio 中的数据表。实体直接与数据源中的表交互。实体包括用于描述其中的数据的字段、用于查找和返回数据的查询以及用于将实体字段连接到数据源列的映射。

实例

实例是您的所有 App Studio 资源的逻辑容器。它代表您、您的公司、团队或组织，并包含您的所有 App Studio 资源，例如应用程序、连接器以及用户和群组的角色分配。大型组织或企业通常有多个 App Studio 实例，例如：沙盒、测试和生产实例。在设置 App Studio 的过程中，您需要创建一个实例。

页面

页面是组件的容器，这些组件构成 App Studio 中应用程序的用户界面。每个页面都代表您的应用程序的用户界面 (UI) 屏幕，您的用户将与之交互。页面是在应用程序工作室的 “页面” 选项卡中创建和编辑的。

触发器

触发器决定何时以及在什么条件下运行自动化。触发器的一些`On click`示例包括按钮和`On select`文本输入。组件的类型决定了该组件的可用触发器列表。触发器被添加到组件中，并在应用程序工作室中进行配置。

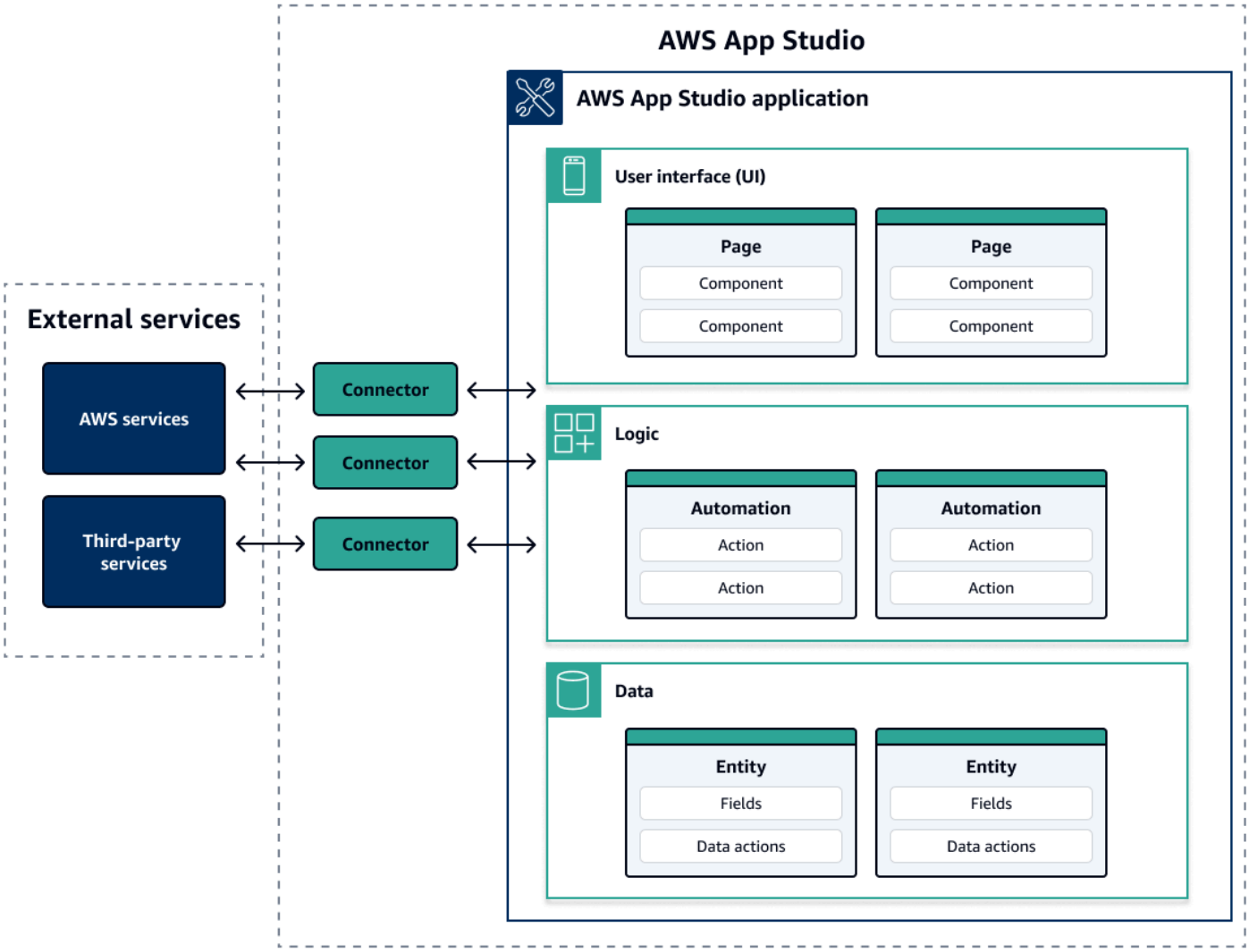
AWS 应用工作室的工作原理

使用 A AWS pp Studio 构建应用程序时，需要了解几个关键概念。本主题涵盖以下概念或资源的基础知识：

- 使用连接器连接到其他服务，以便在您的应用程序中使用它们的资源或 API 调用。例如，您可以使用连接器来存储和访问数据，或者从您的应用程序发送通知。
- 使用实体配置应用程序的数据模型，该模型将您的应用程序与外部数据源连接起来。
- 使用页面和组件来构建应用程序的用户界面 (UI)。
- 使用自动化和操作来实现应用程序的逻辑或行为。
- App Studio 中的应用程序开发生命周期：构建、测试和发布。

有关 App Studio 概念的更多信息，请参阅AWS 应用工作室的概念。

下图是 App Studio 及其资源的组织方式的简单示意图。



在 App Studio 的应用程序中，页面、自动化和实体都相互交互。您可以使用连接器将这些资源连接到外部服务，例如数据、存储或通知提供程序。要成功构建应用程序，了解所有这些概念和资源是如何相互作用的，这一点至关重要。

将您的应用程序连接到其他服务

使用 App Studio 构建应用程序的最大好处之一是能够轻松地将您的应用程序与其他服务集成。在 App Studio 中，您可以使用特定于该服务的连接器以及您要在应用程序中使用的资源或 API 调用连接到其他服务。

您可以在 App Studio 实例级别创建连接器，而不是在单个应用程序中创建连接器。创建连接器后，您可以在应用程序的各个部分中使用它们，具体取决于所连接的服务和应用程序。

以下是使用连接器连接到其他服务的应用程序中的功能示例：

- 几乎所有应用程序中都使用的最常见用例是通过连接到 Amazon Redshift、Amazon DynamoDB 或 Amazon Aurora 等 AWS 数据服务来存储和访问应用程序中使用的数据。
- 允许上传和查看图像（例如收据）的应用程序可以使用 Amazon S3 来存储和访问图像文件。
- 文本摘要器应用程序可以向 Amazon Bedrock 发送文本输入并显示返回的摘要。

> **Note**
>
> 您必须在 App Studio 中拥有管理员角色才能创建连接器。创建连接器时，必须包含正确的凭据以及有关要使用的资源或 API 调用的信息。

配置应用程序的数据模型

您的应用程序的数据是为应用程序提供动力的信息。在 App Studio 中，您可以创建和使用代表您存储和使用的不同类型的数据的实体。例如，在客户会议的跟踪应用程序中，您可能有三个实体分别代表客户会议、议程和与会者。

实体包含描述所存储数据的类型（例如整数或字符串）的字段。尽管您使用实体来定义数据模型，但必须将您的实体连接到 Amazon Redshift 或 Amazon DynamoDB 等外部数据存储服务才能存储数据。您可以将实体视为 App Studio 应用程序和外部服务中数据之间的中介。

您可以使用数据操作通过组件和自动化与应用程序中的数据进行交互。最常用的两个数据操作是`getAll`操作和`getByID`操作。例如，您的应用程序可以使用`getAll`数据操作在表格中填充数据，使用`getByID`操作在详细信息组件中填充有关特定数据条目的更多信息。

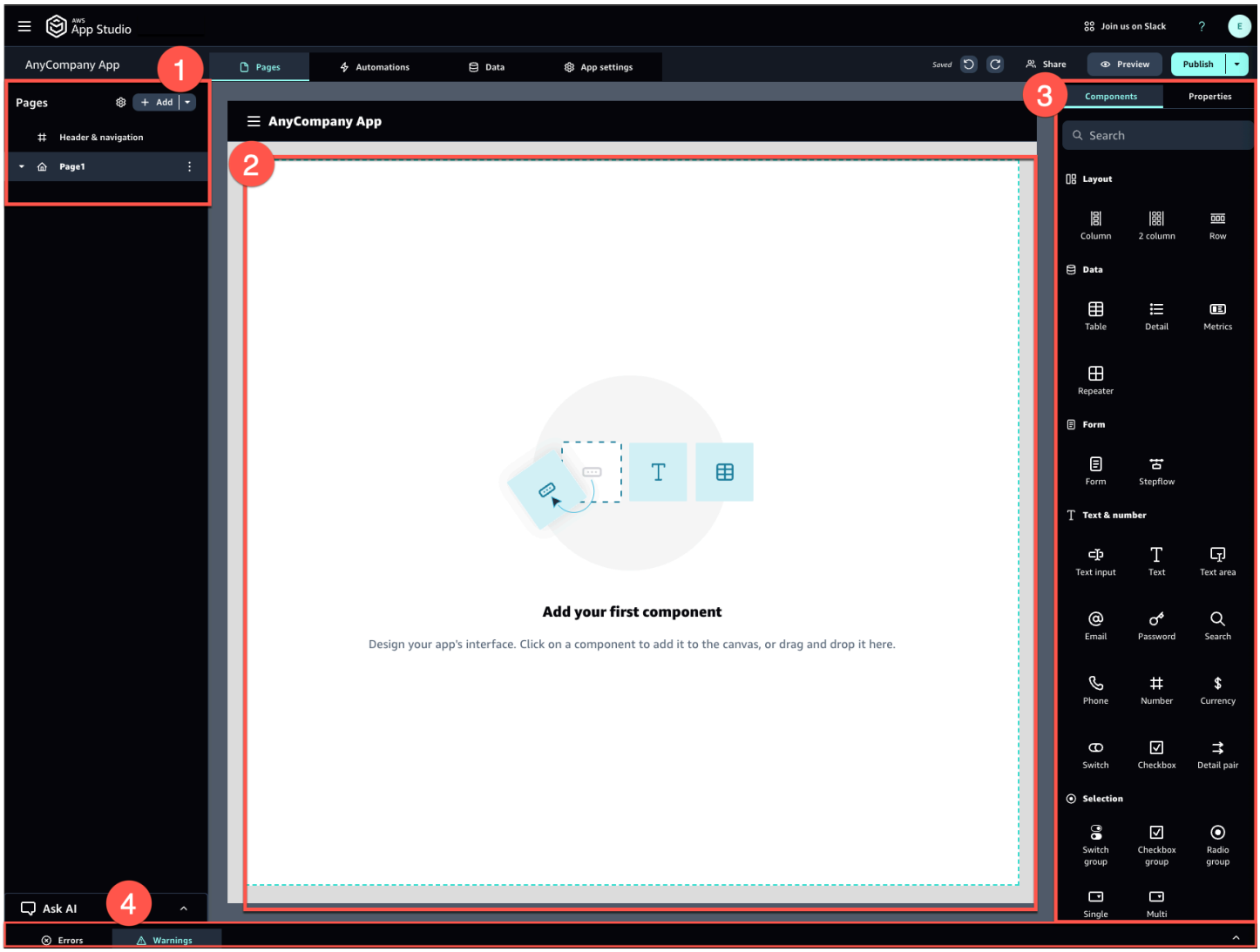
您还可以向实体中添加示例数据，以便更轻松地测试应用程序，而无需调用外部服务。

构建应用程序的用户界面

在 App Studio 中，您可以使用页面和组件来构建应用程序的用户界面。页面是应用程序的各个屏幕，也是组件的容器。组件是应用程序用户界面的构建块。有许多类型的组件，例如表格、表单、图像查看器和按钮。

下图显示了应用程序工作室的页面选项卡，您可以在其中添加或配置应用程序中的页面和组件。以下关键区域被突出显示并编号：

1. 左侧的 “页面” 面板。您可以在此处管理页面、应用程序标题和导航设置。您可以查看应用程序的所有页面和组件。
2. 画布，用于显示当前页面的组件。您可以在画布中选择组件来配置其属性。
3. 右侧的 “组件” 或 “属性” 面板。如果未选择任何内容，则会显示 “组件” 面板，其中显示了可以添加到页面的组件列表。如果选择页面或组件，则会显示 “属性” 面板，您可以在其中配置页面或组件。
4. 底部的 “错误和警告” 面板。这些面板显示应用程序中的任何错误或警告，这些错误或警告最常见的是配置问题。您可以选择面板将其展开并查看消息。



例如，用户必须输入信息的应用程序可能包含以下页面和组件：

- 一种输入页面，包括用户用来填写和提交信息的表单组件。
- 包含表格组件的列表视图页面，其中包含有关每个输入的信息。

- 详细视图页面，其中包含详细信息组件，其中包含有关每个输入的更多信息。

组件可以包含静态信息或数据，例如带有已定义字段的表单。它们还可以使用自动化功能来包含动态信息，例如图像查看器，它可以从 Amazon S3 存储桶中检索图像并将其显示给用户。

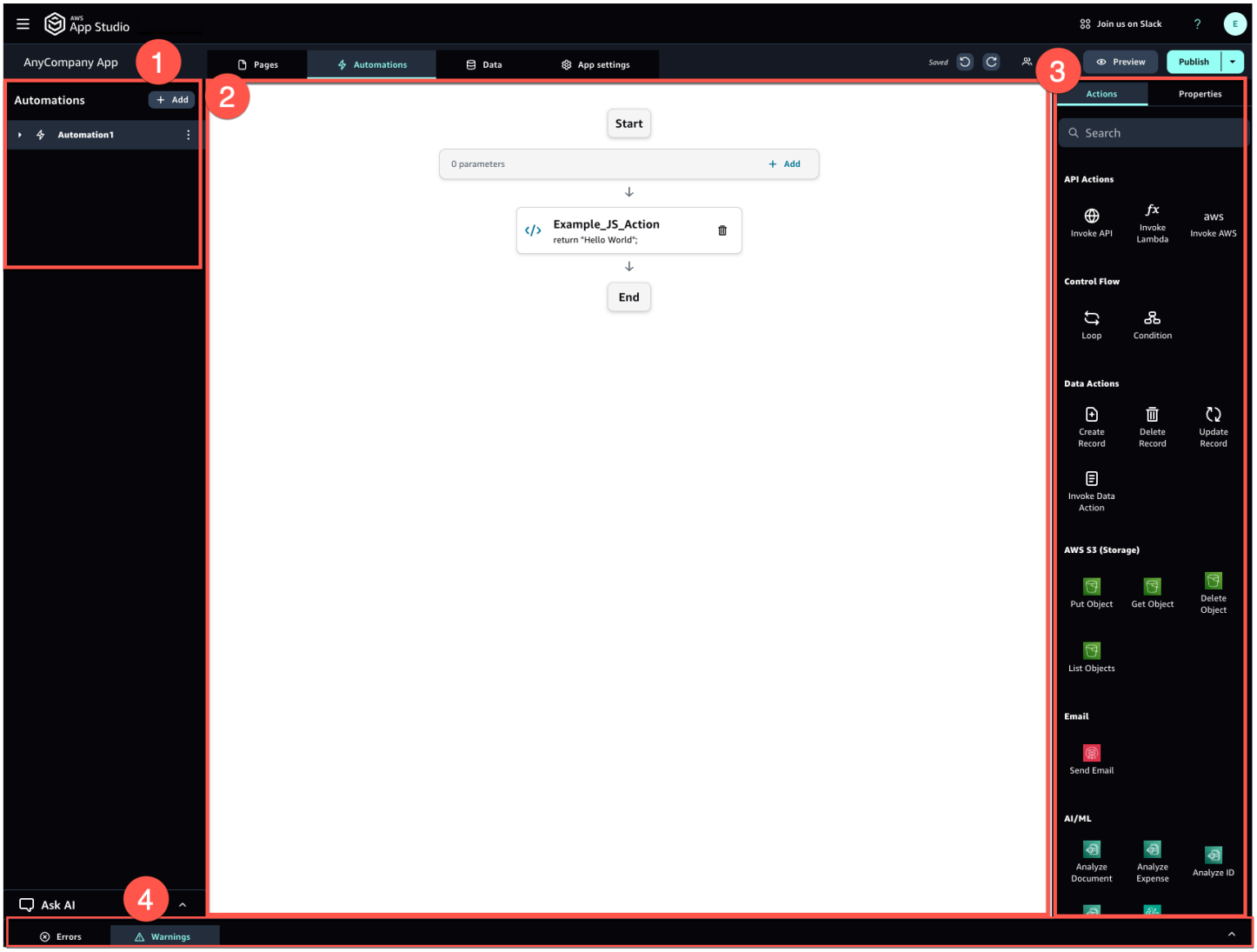
了解页面参数的概念很重要。您可以使用页面参数将信息从一个页面发送到另一个页面。页面参数的一个常见用例是搜索和筛选，其中一个页面的搜索词被发送到表格或项目列表中，以便在另一个页面中进行筛选。另一个用例是查看项目详细信息，其中项目标识符被发送到详细的查看者页面。

实现应用程序的逻辑或行为

您可以将应用程序的逻辑或行为视为应用程序的功能。您可以定义当用户选择按钮、提交信息、导航到新页面或以其他方式进行交互时会发生什么。在 App Studio 中，您可以使用自动化和操作来定义应用程序的逻辑。自动化是操作的容器，操作是自动化功能的基石。

下图显示了 Applications Studio 的 “自动化” 选项卡，您可以在其中添加或配置应用程序中的自动化及其操作。以下关键区域被突出显示并编号：

- 左侧的 “自动化” 面板。这是您管理自动化的地方。您可以查看应用程序的所有自动化和操作。
- 画布，显示当前的自动化。它显示配置的自动化参数（将在本节后面介绍）和操作。您可以在画布中选择组件来配置其属性。
- 右侧的 “动作” 和 “属性” 面板。如果未选择任何内容，则会显示 “动作” 面板。它会显示可添加到自动化中的操作列表。如果选择自动化，则可以查看和配置其属性，例如自动化的输入和输出。如果选择某项操作，则可以查看和配置该动作的属性。
- 底部的 “错误和警告” 面板。此面板显示应用程序中的任何错误或警告（最常见的是配置问题）。您可以选择面板将其展开并查看消息。



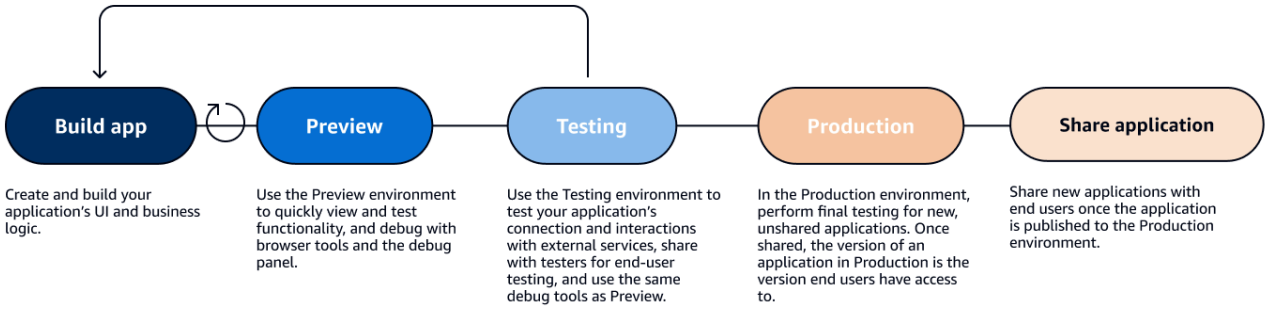
自动化可以很简单（例如添加数字并返回结果），也可以更强大（例如将输入发送到其他服务并返回结果）。自动化的主要组成部分如下：

- 触发器，它定义何时运行自动化。例如，当用户在用户界面中按下按钮时。
- 自动化输入，用于向自动化发送信息。您可以使用自动化参数定义自动化输入。例如，如果您想使用 Amazon Bedrock 向用户返回文本摘要，则可以将要摘要的文本配置为自动化参数。
- 操作，它们是自动化功能的组成部分。您可以将每个操作视为自动化过程中的一个步骤。操作可以调用 APIs、调用自定义 JavaScript、创建数据记录和执行其他函数。您也可以将操作分组为循环或条件，以进一步自定义功能。您也可以通过操作调用其他自动化。
- 自动化输出，可以在组件甚至其他自动化中使用。例如，自动化输出可以是显示在文本组件中的文本、要在图像查看器组件中显示的图像或其他自动化的输入。

应用程序的开发生命周期

应用程序的开发生命周期包括以下阶段：构建、测试和发布。之所以称之为循环，是因为在创建和迭代应用程序时，你很可能会在这些阶段之间进行迭代。

下图显示了 App Studio 中应用程序开发生命周期的简化时间表：



App Studio 提供各种工具来支持应用程序的生命周期。这些工具包括以下三个不同的环境，如上图所示：

- 预览环境，您可以在其中预览应用程序以查看最终用户的外观，并测试特定功能。使用预览环境可以快速测试和迭代应用程序，而无需发布应用程序。预览环境中的应用程序不与外部服务通信或传输数据。这意味着您无法在预览环境中测试依赖外部服务的交互和功能。
- 测试环境，您可以在其中测试应用程序的连接以及与外部服务的交互。在这里，您还可以通过将发布到测试环境的版本共享给一组测试人员来进行最终用户测试。
- 生产环境，您可以在其中对新应用程序进行最终测试，然后再与最终用户共享。共享应用程序后，发布到生产环境的应用程序版本就是最终用户将要查看和使用的版本。

了解更多

既然您已经了解了 App Studio 中应用程序开发的基本原理，您可以开始构建自己的应用程序，也可以更深入地了解有关概念和资源的更多信息。

要开始构建，我们建议您尝试以下入门教程之一：

- 请关注教程：使用 AI 生成应用程序以了解如何使用 AI 构建器助手抢先开发应用程序。
- 跟随学习教程：从一个空的应用程序开始构建如何从头开始构建应用程序，同时学习基础知识。

要了解有关本主题中提及的资源或概念的更多信息，请参阅以下主题：

- 使用连接器将 App Studio 连接到其他服务
- 实体和数据操作：配置应用程序的数据模型
- 页面和组件：构建应用程序的用户界面
- 自动化和操作：定义应用程序的业务逻辑
- 预览、发布和共享应用程序

设置并登录 AWS App Studio

根据你的角色，设置 AWS App Studio 会有所不同：

- 首次以 AWS 或组织管理员身份进行设置：要首次以管理员身份设置 App Studio，您需要创建一个 AWS 账户（如果没有），创建 App Studio 实例，然后使用 IAM Identity Center 群组配置用户访问权限。创建实例后，在 App Studio 中拥有管理员角色的任何人都可以进一步设置任务，例如，配置连接器以将其他服务（例如数据源）连接到您的 App Studio 实例。有关首次安装的信息，请参见首次创建和设置 App Studio 实例。
- 构建者@@ 入门：当您收到以构建者身份加入 App Studio 的邀请时，您必须接受邀请并通过提供密码激活您的 IAM Identity Center 用户证书。之后，您可以登录 App Studio 并开始构建应用程序。有关接受邀请和加入 App Studio 实例的信息，请参阅接受加入 App Studio 的邀请。

首次创建和设置 App Studio 实例

注册一个 AWS 账户

需要一个 AWS 账号才能设置 App Studio。使用 App Studio 只需要一个 AWS 账户——构建者和管理员不需要 AWS 账户即可使用 App Studio，因为访问权限由 AWS IAM Iditentity Center 管理。

要创建 AWS 账户

1. 打开https://portal.aws.amazon.com/billing/注册。
2. 按照屏幕上的说明操作。

   在注册时，将接到电话，要求使用电话键盘输入一个验证码。

   当您注册时 AWS 账户，就会创建AWS 账户根用户一个。根用户有权访问该账户中的所有 AWS 服务 和资源。作为最佳安全实践，请为用户分配管理访问权限，并且只使用根用户来执行需要根用户访问权限的任务。

创建管理用户来管理 AWS 资源

首次创建 AWS 账户时，首先使用一组默认证书，可以完全访问账户中的所有 AWS 资源。此身份称为AWS 账户根用户。要创建用于 App Studio 的 AWS 角色和资源，我们强烈建议您不要使用 root 用户 AWS 账户。相反，我们建议您创建和使用管理用户。

使用以下主题创建管理用户来管理 AWS 角色和资源，以便在 App Studio 中使用。

- 对于单个独立 AWS 账户，请参阅 IAM 用户指南中的创建您的第一个 IAM 用户。您可以提供任何用户名，但该用户名必须具有`AdministratorAccess`权限策略。
- 对于通过管理的多个 AWS 账户 AWS Organizations，请参阅用户指南中的为 IA M Identity Center 管理用户设置 AWS 账户访问权限。AWS IAM Identity Center

在中创建 App Studio 实例 AWS Management Console

要使用 App Studio，您必须通过中的 App Studio 登录页面创建实例 AWS Management Console。有两种方法可用于创建 App Studio 实例：

1. 轻松创建：通过这种简化的方法，您只需将一个用户设置为在设置过程中访问和使用 App Studio。如果您正在为您的组织或团队评估 App Studio，或者您只打算自己使用 App Studio，则应使用此方法。设置完成后，您可以向 App Studio 添加更多用户或群组。请注意，如果您有 IAM Identity Center 的组织实例，则无法使用此方法。
2. 标准创建：使用此方法，您可以添加用户或群组，并在设置过程中在 App Studio 中为他们分配角色。如果您想在设置时向 App Studio 添加多个用户，则应使用此方法。

> **Note**
>
> 您只能在所有 AWS 地区创建一个 App Studio 实例。如果您已有实例，则必须先将其删除，然后再创建另一个实例。有关更多信息，请参阅 删除 App Studio 实例。

Easy create

要在中创建 App Studio 实例，只需 AWS Management Console 轻松创建即可

1. 在以下位置打开 App Studio 控制台https://console.aws.amazon.com/appstudio/。
2. 导航到您要在其中创建 App Studio 实例的 AWS 区域。
3. 选择开始。
4. 选择 “轻松创建”，然后选择 “下一步”。
5. 设置 App Studio 的后续步骤取决于您是否拥有 IAM 身份中心账户实例。要了解有关 IAM Identity Center 实例的更多信息，包括不同的类型以及如何查找您拥有的类型，请参阅 IAM Identity Center 用户指南中的管理 IAM Identity Center 的组织和账户实例。AWS

- 如果您有 IAM 身份中心的账户实例：

  a. 在账户权限中，查看启用 App Studio 所需的权限。如果你的账户没有所需的权限，你将无法启用 App Studio。您必须将所需权限添加到您的账户中，或者切换到拥有这些权限的账户。

  b. 在添加用户中，搜索并选择您的 IAM Identity Center 账户实例中将访问 App Studio 的用户的电子邮件地址。此用户将在 App Studio 实例中扮演管理员角色。如果您没有看到想要提供对 App Studio 访问权限的用户，则可能需要将其添加到您的 IAM 身份中心实例。

- 如果您没有 IAM 身份中心的账户实例：

  > **Note**
  >
  > 设置 App Studio 会自动使用您在设置过程中配置的用户创建一个 IAM Identity Center 账户实例。设置完成后，您可以在 IAM Identity Center 控制台中添加或管理用户和群组，网址为https://console.aws.amazon.com/singlesignon/。

  a. 在账户权限中，查看启用 App Studio 所需的权限。如果您的账户没有所需的权限，您将无法启用 App Studio。您必须将所需权限添加到您的账户中，或者切换到拥有这些权限的账户。

  b. 在添加用户中，为访问 App Studio 的用户提供电子邮件地址、名字、姓氏和用户名。此用户将在 App Studio 实例中扮演管理员角色。

6. 在服务访问权限和角色中，查看在设置 App Studio 为服务提供必要权限时自动创建的服务角色和服务相关角色。选择查看权限以查看为服务角色授予的确切权限，或选择查看策略以查看附加到服务相关角色的权限策略。
7. 在 “确认” 中，通过选中对声明的复选框进行确认。
8. 选择设置以创建您的实例。

> **Note**
>
> 要在设置后向 App Studio 实例添加更多用户或群组，您必须将其添加到 IAM Identity Center 实例。

Standard create

使用标准方法在中创建 App Studio 实例 AWS Management Console

1. 在以下位置打开 App Studio 控制台https://console.aws.amazon.com/appstudio/。
2. 导航到您要在其中创建 App Studio 实例的 AWS 区域。
3. 选择开始。
4. 选择标准创建，然后选择下一步。
5. 设置 App Studio 的步骤取决于您是否拥有 IAM 身份中心实例以及实例类型。要了解有关 IAM Identity Center 实例的更多信息，包括不同的类型以及如何查找您拥有的类型，请参阅 IAM Identity Center 用户指南中的管理 IAM Identity Center 的组织和账户实例。AWS
   - 如果您有 IAM 身份中心的组织实例：
     - 在使用单点登录配置对 App Studio 的访问权限中，选择现有的 IAM 身份中心群组，为他们提供对 App Studio 的访问权限。将根据指定的配置创建 App Studio 群组。添加到管理员群组的成员将具有管理员角色，而添加到构建器群组的成员将在 App Studio 中拥有构建者角色。角色定义如下：
       - 管理员可以在 App Studio 中管理用户和群组、添加和管理连接器以及管理构建器创建的应用程序。此外，具有管理员角色的用户拥有构建者角色中包含的所有权限。
       - 构建者可以创建和构建应用程序。构建器无法管理用户或群组、添加或编辑连接器实例，也无法管理其他构建器的应用程序。
   - 如果您有 IAM 身份中心实例的账户实例：
     a. 在账户权限中，查看启用 App Studio 所需的权限。如果您的账户没有所需的权限，您将无法启用 App Studio。您必须将所需权限添加到您的账户中，或者切换到拥有这些权限的账户。
     b. 在 “通过单点登录配置对 App Studio 的访问权限” 中，在 IAM Identity Center 账户中，选择使用现有账户实例
     c. 在AWS 区域中，选择您的 IAM Identity Center 账户实例所在的区域。
     d. 选择现有的 IAM 身份中心群组，为他们提供访问 App Studio 的权限。将根据指定的配置创建 App Studio 群组。添加到管理员群组的成员将具有管理员角色，而添加到构建器群组的成员将在 App Studio 中拥有构建者角色。角色定义如下：
       - 管理员可以在 App Studio 中管理用户和群组、添加和管理连接器以及管理构建器创建的应用程序。此外，具有管理员角色的用户拥有构建者角色中包含的所有权限。

- 构建者可以创建和构建应用程序。构建器无法管理用户或群组、添加或编辑连接器实例，也无法管理其他构建器的应用程序。

- 如果您没有 IAM 身份中心实例：

> **Note**
>
> 设置 App Studio 会自动使用您在设置过程中配置的群组创建一个 IAM Identity Center 账户实例。设置完成后，您可以在 IAM Identity Center 控制台中添加或管理用户和群组，网址为https://console.aws.amazon.com/singlesignon/。

a. 在账户权限中，查看启用 App Studio 所需的权限。如果你的账户没有所需的权限，你将无法启用 App Studio。您必须将所需权限添加到您的账户中，或者切换到拥有这些权限的账户。

b. 在 “使用单点登录配置对 App Studio 的访问权限” 中，在 IAM Identity Center 账户中，选择为我创建账户实例。

c. 在创建用户和群组并将其添加到 App Studio 中，提供名称并将用户添加到管理员群组和构建者群组。添加到管理员组的用户将在 App Studio 中拥有管理员角色，而添加到构建者组的用户将拥有构建者角色。角色定义如下：

   - 管理员可以在 App Studio 中管理用户和群组、添加和管理连接器以及管理构建器创建的应用程序。此外，具有管理员角色的用户拥有构建者角色中包含的所有权限。
   - 构建者可以创建和构建应用程序。构建器无法管理用户或群组、添加或编辑连接器实例，也无法管理其他构建器的应用程序。

> **Important**
>
> 您必须将自己添加为管理员组的用户才能设置 App Studio，并在设置后拥有管理员访问权限。

6. 在服务访问权限和角色中，查看在设置 App Studio 为服务提供必要权限时自动创建的服务角色和服务相关角色。选择查看权限以查看为服务角色授予的确切权限，或选择查看策略以查看附加到服务相关角色的权限策略。
7. 在 “确认” 中，通过选中对声明的复选框进行确认。
8. 选择设置以创建您的实例。

接受加入 App Studio 的邀请

App Studio 的访问权限由 IAM 身份中心管理。这意味着，每个想要使用 App Studio 的用户都必须在 IAM Identity Center 中配置一个用户，并且必须属于管理员已添加到 App Studio 的群组。当管理员邀请您加入 IAM Identity Center 时，您将收到一封电子邮件，要求您接受邀请并激活您的用户证书。激活后，您可以使用这些凭据登录 App Studio。

接受 IAM 身份中心访问 App Studio 的邀请

1. 收到邀请电子邮件后，请按照步骤在 IAM Identity Center 中提供密码并激活您的用户证书。有关更多信息，请参阅接受加入 IAM 身份中心的邀请。
2. 激活用户凭证后，使用它们登录您的 App Studio 实例。

AWS App Studio 入门

以下入门教程将引导您完成在 App Studio 中构建第一个应用程序的过程。

- 推荐：要使用生成式 AI 来描述您要创建的应用程序，并自动创建该应用程序及其资源，请参阅教程：使用 AI 生成应用程序。
- 要从空应用程序开始构建，请参阅教程：从一个空的应用程序开始构建。

教程：使用 AI 生成应用程序

AWS App Studio 在整个服务中都包含生成式 AI 功能，可帮助加快应用程序构建。在本教程中，您将通过使用自然语言描述您的应用程序，学习如何使用 AI 生成应用程序。

使用 AI 生成应用程序是开始构建的好方法，因为应用程序的许多资源都是为您创建的。使用现有资源从生成的应用程序开始构建通常比从空应用程序开始构建要容易得多。

> **Note**
>
> 您可以查看博客文章使用 AWS App Studio（预览版）使用自然语言构建企业级应用程序，以查看包含图像的类似演练。博客文章还包含有关设置和配置管理员相关资源的信息，但如果需要，您可以跳到有关构建应用程序的部分。

当 App Studio 生成带有 AI 的应用程序时，它会使用以下资源来创建应用程序，这些资源是针对您所描述的应用程序量身定制的：

- 页面和组件：组件是应用程序用户界面的基块。它们代表表格、表单和按钮等视觉元素。每个组件都有自己的一组属性，您可以根据自己的特定要求自定义组件。页面是组件的容器。
- 自动化：您可以使用自动化来定义控制应用程序行为的逻辑和工作流程。例如，您可以使用自动化来创建、更新、读取或删除数据表中的行，或者与 Amazon S3 存储桶中的对象进行交互。您还可以使用它们来处理诸如数据验证、通知或与其他系统的集成之类的任务。
- 实体：数据是为您的应用程序提供动力的信息。生成的应用程序会创建类似于表格的实体。实体代表您需要存储和处理的不同类型的数据，例如客户、产品或订单。您可以使用 App Studio 连接器将这些数据模型连接到各种数据源 APIs，包括 AWS 服务和外部数据源。

目录

- 先决条件
- 第 1 步：创建 应用程序
- 第 2 步：探索您的新应用程序
  - 浏览页面和组件
  - 探索自动化和操作
  - 使用实体浏览数据
- 第 3 步：预览您的应用程序
- 后续步骤

先决条件

在开始之前，请查看并完成以下先决条件：

- 访问 AWS App Studio。有关更多信息，请参阅 设置并登录 AWS App Studio。
- 可选：查看AWS 应用工作室的概念以熟悉 App Studio 的重要概念。

第 1 步：创建 应用程序

生成应用程序的第一步是在 App Studio 中向 AI 助手描述要创建的应用程序。您可以查看将生成的应用程序，并在生成应用程序之前根据需要进行迭代。

使用 AI 生成应用程序

1. 登录 App Studio。
2. 在左侧导航栏中，选择 Builder Hub，然后选择 + 创建应用程序。
3. 选择 “使用 AI 生成应用程序”。
4. 在应用程序名称字段中，为您的应用程序提供一个名称。
5. 在 “选择数据源” 对话框中，选择 “跳过”。
6. 您可以通过在文本框中描述要生成的应用程序或在示例提示上选择 “自定义” 来开始定义要生成的应用程序。描述您的应用程序后，App Studio 会生成应用程序要求和详细信息供您查看。这包括用例、用户流和数据模型。
7. 使用文本框根据需要对应用程序进行迭代，直到您对要求和细节感到满意。
8. 准备好生成应用程序并开始构建时，请选择生成应用程序。
9. 或者，您可以观看一段简短的视频，详细介绍如何在您的新应用程序中导航。

10. 选择 “编辑应用程序” 以进入应用程序的开发环境。

第 2 步：探索您的新应用程序

在开发环境中，您将找到以下资源：

- 用于查看或编辑应用程序的画布。画布会根据所选资源而变化。
- 画布顶部的导航选项卡。以下列表对这些选项卡进行了描述：
  - 页面：使用页面和组件来设计应用程序界面的位置。
  - 自动化：在自动化中使用操作来定义应用程序的业务逻辑。
  - 数据：在其中定义实体、其字段、示例数据和数据操作，以定义应用程序的数据模型。
  - 应用程序设置：用于定义应用程序设置，包括应用程序角色，用于为最终用户定义基于角色的页面可见性。
- 左侧导航菜单，其中包含基于您正在查看的选项卡的资源。
- 右侧菜单，列出 “页面” 和 “自动化” 选项卡中选定资源的资源和属性。
- 在生成器底部显示警告和错误的调试控制台。您生成的应用程序中可能存在错误。这可能是由于自动化需要配置的连接器才能执行操作，例如使用 Amazon Simple Email Service 发送电子邮件。
- “询问 AI” 聊天窗口，可从 AI 生成器助手那里获取情境帮助。

让我们仔细看看 “页面”、“自动化” 和 “数据” 选项卡。

浏览页面和组件

“页面” 选项卡显示为您生成的页面及其组件。

每个页面都代表您的应用程序的用户界面 (UI) 屏幕，您的用户将与之交互。在这些页面上，您可以找到各种组件（例如表格、表单和按钮）来创建所需的布局和功能。

请花点时间使用左侧导航菜单查看页面及其组件。选择页面或组件时，可以在右侧菜单中选择 “属性”。

探索自动化和操作

“自动化” 选项卡显示为您生成的自动化及其操作。

自动化定义应用程序的业务逻辑，例如创建、查看、更新或删除数据条目、发送电子邮件，甚至调用或 APIs Lambda 函数。

花点时间使用左侧导航菜单查看自动化。选择自动化后，可以在右侧的 “属性” 菜单上查看其属性。自动化包含以下资源：

- 自动化由单个操作组成，这些操作是应用程序业务逻辑的基石。您可以在左侧导航菜单上或所选自动化的画布中查看自动化的操作。选择操作后，可以在右侧的 “属性” 菜单上查看其属性。
- 自动化参数是将数据传递到自动化的方式。参数充当占位符，当自动化运行时，这些占位符会被实际值替换。这允许您每次使用相同的自动化，输入也不同。
- 自动化输出是您配置自动化结果的地方。默认情况下，自动化没有输出，因此要在组件或其他自动化中使用自动化的结果，必须在此处定义它们。

有关更多信息，请参阅 自动化概念。

使用实体浏览数据

“数据” 选项卡显示为您生成的实体。

实体表示保存应用程序数据的表，类似于数据库中的表。它们不是将应用程序的用户界面 (UI) 和自动化直接连接到数据源，而是先连接到实体。实体充当您的实际数据源和 App Studio 应用程序之间的中介。这为管理和访问您的数据提供了一个单一的地方。

花点时间查看通过从左侧导航菜单中选择生成的实体。您可以查看以下详细信息：

- “配置” 选项卡显示实体名称及其字段，这些字段代表您的实体的列。
- 数据操作选项卡显示您的实体生成的数据操作。组件和自动化可以使用数据操作从您的实体中获取数据。
- 示例数据选项卡显示了示例数据，您可以使用这些数据在开发环境（不与外部服务通信）中测试您的应用程序。有关环境的更多信息，请参阅应用程序环境。
- “连接” 选项卡显示有关实体所连接的外部数据源的信息。App Studio 提供了一种使用 DynamoDB 表的托管数据存储解决方案。有关更多信息，请参阅 AWS App Studio 中的管理数据实体。

第 3 步：预览您的应用程序

您可以在 App Studio 中预览应用程序，以查看其对用户的显示效果。您还可以通过使用它并在调试面板中查看日志来测试其功能。

应用程序预览环境不支持显示实时数据或使用连接器与外部资源（例如数据源）的连接。相反，您可以使用样本数据和模拟输出来测试功能。

预览您的应用程序以进行测试

1. 在应用程序构建器的右上角，选择预览。
2. 与您的应用程序的页面互动。

后续步骤

现在，您已经创建了第一个应用程序，接下来是一些后续步骤：

- 有关包含图像的另一篇入门演练，请参阅博客文章使用 AWS App Studio 使用自然语言构建企业级应用程序（预览版）。
- 应用程序使用连接器发送和接收数据，或与外部服务（包括 AWS 服务和第三方服务）通信。有必要详细了解连接器以及如何对其进行配置以构建应用程序。请注意，您必须具有管理员角色才能管理连接器。要了解更多信息，请参阅使用连接器将 App Studio 连接到其他服务。
- 要了解有关预览、发布以及最终与最终用户共享应用程序的更多信息，请参阅预览、发布和共享应用程序。
- 继续探索和更新您生成的应用程序，以获得一些动手体验。
- 要了解有关构建应用程序的更多信息，请查看生成器文档。具体而言，以下主题可能有助于探索：
  - 自动化操作参考
  - 组件参考
  - 使用组件和自动化与 Amazon 简单存储服务交互
  - 安全注意事项和缓解措施

教程：从一个空的应用程序开始构建

在本教程中，您将使用 AWS App Studio 构建内部客户会议请求应用程序。你将学习如何在 App Studio 中构建应用程序，同时重点介绍现实世界的用例和动手示例。此外，你还将学习如何定义数据结构、UI 设计和应用程序部署。

> **Note**
>
> 本教程详细介绍了如何从头开始构建应用程序，从空应用程序开始。通常，通过提供要创建的应用程序的描述，使用 AI 来帮助生成应用程序及其资源会更快、更容易。有关更多信息，请参阅 教程：使用 AI 生成应用程序。

要了解如何使用 App Studio 构建应用程序，关键是要了解以下四个核心概念以及它们是如何协同工作的：组件、自动化、数据和连接器。

- 组件：组件是应用程序用户界面的基块。它们代表表格、表单和按钮等视觉元素。每个组件都有自己的属性集，您可以根据自己的特定要求对其进行自定义。
- 自动化：通过自动化，您可以定义控制应用程序行为的逻辑和工作流程。您可以使用自动化来创建、更新、读取或删除数据表中的行，或者与 Amazon S3 存储桶中的对象进行交互。您还可以使用它们来处理诸如数据验证、通知或与其他系统的集成之类的任务。
- 数据：数据是为您的应用程序提供动力的信息。在 App Studio 中，您可以定义称为实体的数据模型。实体代表您需要存储和处理的不同类型的数据，例如客户会议请求、议程或与会者。您可以使用 App Studio 连接器将这些数据模型连接到各种数据源 APIs，包括 AWS 服务和外部数据源。
- 连接器：App Studio 提供与各种数据源的连接，其中包括 Aurora、DynamoDB 和 Amazon Redshift 等 AWS 服务。数据源还包括第三方服务，例如Salesforce，或者许多其他使用 OpenAPI 或通用 API 连接器的服务。您可以使用 App Studio 连接器轻松地将这些企业级服务和外部应用程序中的数据和功能整合到您的应用程序中。

在学习本教程的过程中，您将探索组件、数据和自动化的关键概念是如何结合在一起构建您的内部客户会议请求应用程序的。

以下是描述你将在本教程中做什么的高级步骤：

1. 从@@ 数据开始：许多应用程序都是从数据模型开始的，因此本教程也从数据开始。要构建 “客户会议请求” 应用程序，您需要先创建一个`MeetingRequests`实体。此实体代表用于存储所有相关会议请求信息（例如客户姓名、会议日期、议程和与会者）的数据结构。该数据模型是您的应用程序的基础，并为您将要构建的各种组件和自动化提供支持。
2. 创建您的用户界面 (UI)：有了数据模型后，本教程将指导您完成用户界面 (UI) 的构建。在 App Studio 中，您可以通过添加页面并向其中添加组件来构建界面。您将向会议请求仪表板页面添加表格、详细信息视图和日历等组件。这些组件将设计为显示存储在`MeetingRequests`实体中的数据并与之交互。这允许您的用户查看、管理和安排客户会议。您还将创建一个会议请求创建页面。该页面包括一个用于收集数据的表单组件和一个用于提交数据的按钮组件。
3. 使用自动化功能添加业务逻辑：为了增强应用程序的功能，您需要配置一些组件以启用用户交互。一些示例是导航到页面或在`MeetingRequests`实体中创建新的会议请求记录。
4. 使用验证和表达式进行增强：为确保数据的完整性和准确性，您需要向表单组件添加验证规则。这将有助于确保用户在创建新的会议请求记录时提供完整有效的信息。此外，你还将使用表达式来引用和操作应用程序中的数据，这样你就可以在整个用户界面中显示动态和上下文信息。

5. 预览和测试：在部署应用程序之前，您将有机会对其进行全面的预览和测试。这将允许您验证组件、数据和自动化是否全部无缝协作。这为您的用户提供了流畅直观的体验。
6. 发布应用程序：最后，您将部署已完成的内部客户会议请求应用程序，并使其可供用户访问。借助 App Studio 中低代码方法的强大功能，您无需大量的编程专业知识即可构建满足组织特定需求的自定义应用程序。

目录

- 先决条件
- 第 1 步：创建 应用程序
- 第 2 步：创建一个实体来定义应用程序的数据
  - 创建托管实体
  - 向您的实体添加字段
- 第 3 步：设计用户界面 (UI) 和逻辑
  - 添加会议请求控制面板页面
  - 添加会议请求创建页面
- 步骤 4：预览应用程序
- 步骤 5：将应用程序发布到测试环境
- 后续步骤

先决条件

在开始之前，请查看并完成以下先决条件：

- 访问 AWS App Studio。有关更多信息，请参阅 设置并登录 AWS App Studio。
- 可选：查看AWS 应用工作室的概念以熟悉 App Studio 的重要概念。
- 可选：了解基本的 Web 开发概念，例如 JavaScript语法。
- 可选：熟悉 AWS 服务。

第 1 步：创建 应用程序

1. 登录 App Studio。
2. 在左侧导航栏中，选择 Builder 中心，然后选择 + 创建应用程序。

3. 选择从头开始。
4. 在应用程序名称字段中，为您的应用程序提供一个名称，例如**`Customer Meeting Requests`**。
5. 如果系统要求选择数据源或连接器，请选择跳过以了解本教程。
6. 选择下一步以继续。
7. （可选）：观看视频教程，快速了解如何在 App Studio 中构建应用程序。
8. 选择 “编辑应用程序”，即可进入 App Studio 应用程序生成器。

第 2 步：创建一个实体来定义应用程序的数据

实体表示保存应用程序数据的表，类似于数据库中的表。它们不是应用程序的用户界面 (UI) 和自动化直接连接到数据源，而是先连接到实体。实体充当您的实际数据源和 App Studio 应用程序之间的中介，并提供管理和访问数据的单一位置。

有四种方法可以创建实体。在本教程中，您将使用 App Studio 托管实体。

创建托管实体

创建托管实体还会创建由 App Studio 管理的相应的 DynamoDB 表。在 App Studio 应用程序中对实体进行更改时，DynamoDB 表会自动更新。使用此选项，您无需手动创建、管理或连接到第三方数据源，也不必指定从实体字段到表列的映射。

创建实体时，必须定义主键字段。主键是实体中每条记录或每行的唯一标识符。这确保了可以轻松识别和检索每条记录，而不会产生模棱两可。主键由以下属性组成：

- 主键名称：实体主键字段的名称。
- 主键数据类型：主键字段的类型。在 App Studio 中，支持的主键类型为字符串（文本）和浮点型（表示数字）。文本主键（比如*`meetingName`*）的类型为 String，数字主键（比如*`meetingId`*）的类型为 Float。

主键是实体的关键组成部分，因为它可以确保数据完整性，防止数据重复，并实现高效的数据检索和查询。

创建托管实体

1. 从顶部栏菜单中选择 “数据”。
2. 选择 + 创建实体。
3. 选择创建 App Studio 管理实体。

4. 在实体名称字段中，为您的实体提供一个名称。在本教程中，请输入 **MeetingRequests**。
5. 在主键字段中，输入要分配给实体中主键列的主键名称标签。在本教程中，请输入 **requestID**。
6. 对于主键数据类型，选择 Float。
7. 选择 Create entity (创建实体)。

向您的实体添加字段

对于每个字段，您将指定显示名称，即应用程序用户可见的标签。显示名称可以包含空格和特殊字符，但在实体中必须是唯一的。显示名称可作为该字段的用户友好型标签，可帮助用户轻松识别和理解其用途。

接下来，您将提供系统名称，这是应用程序内部用来引用该字段的唯一标识符。系统名称应简洁，不含空格或特殊字符。系统名称允许应用程序更改字段的数据。它充当应用程序中该字段的唯一参考点。

最后，您将选择最能代表要在字段中存储的数据类型的数据类型，例如字符串（文本）、布尔值（真/假）、日期、十进制、浮点数、整数或。 DateTime定义适当的数据类型可确保数据的完整性，并允许正确处理和处理字段的值。例如，如果您在会议请求中存储客户姓名，则需要选择适合文本值`String`的数据类型。

向您的**MeetingRequests**实体添加字段

- 选择 + 添加字段以添加以下四个字段：
  a. 添加一个代表客户姓名的字段，其中包含以下信息：
     - 显示名称：**Customer name**
     - 系统名称：**customerName**
     - 数据类型：**String**
  b. 添加一个代表会议日期的字段，其中包含以下信息：
     - 显示名称：**Meeting date**
     - 系统名称：**meetingDate**
     - 数据类型：**DateTime**
  c. 添加一个代表会议议程的字段，其中包含以下信息：
     - 显示名称：**Agenda**
     - 系统名称：**agenda**

- 数据类型：**String**

d. 添加一个字段，用以下信息代表会议参与者：

- 显示名称：**Attendees**
- 系统名称：**attendees**
- 数据类型：**String**

您可以向实体添加示例数据，以便在发布应用程序之前使用这些数据来测试和预览应用程序。通过添加多达 500 行的模拟数据，您可以模拟真实场景并检查您的应用程序如何处理和显示各种类型的数据，而无需依赖实际数据或连接到外部服务。这可以帮助您在开发过程的早期发现并解决任何问题或不一致之处。这样可以确保您的应用程序在处理实际数据时按预期运行。

向您的实体添加示例数据

1. 在横幅中选择 “样本数据” 选项卡。
2. 选择 “生成更多样本数据”。
3. 选择保存。

或者，在横幅中选择 “连接”，查看有关连接器和为您创建的 DynamoDB 表的详细信息。

第 3 步：设计用户界面 (UI) 和逻辑

添加会议请求控制面板页面

在 App Studio 中，每个页面都代表用户将与之交互的应用程序用户界面 (UI) 屏幕。在这些页面中，您可以添加各种组件，例如表格、表单和按钮，以创建所需的布局和功能。

新创建的应用程序带有默认页面，因此您可以重命名该页面，而不是添加一个新页面来用作简单的会议请求仪表板页面。

重命名默认页面

1. 在顶部栏导航菜单中，选择页面。
2. 在左侧面板中，双击 Page1，将其重命名为**MeetingRequestsDashboard**，然后按 Enter。

现在，向页面添加一个用于显示会议请求的表格组件。

向会议请求仪表板页面添加表格组件

1. 在右侧的 “组件” 面板中，找到表格组件并将其拖到画布上。
2. 在画布中选择表格以将其选中。
3. 在右侧的 “属性” 面板中，更新以下设置：
   a. 选择铅笔图标将表格重命名为**meetingRequestsTable**。
   b. 在来源下拉列表中，选择实体。
   c. 在数据操作下拉列表中，选择您创建的实体 (**MeetingRequests**)，然后选择 + 添加数据操作。
4. 如果出现提示，请选择 getAll。

> **Note**
>
> getAll 数据操作是一种特定类型的数据操作，用于检索指定实体中的所有记录（行）。例如，当您将 getAll 数据操作与表组件关联时，表格会自动填充来自连接实体的所有数据，并将每条记录显示为表中的一行。

添加会议请求创建页面

接下来，创建一个包含最终用户用于创建会议请求的表单的页面。您还将添加一个提交按钮，用于在`MeetingRequests`实体中创建记录，然后将最终用户导航回`MeetingRequestsDashboard`页面。

添加会议请求创建页面

1. 在顶部横幅中，选择页面。
2. 在左侧面板中，选择 + 添加。
3. 在右侧的属性面板中，选择铅笔图标并将页面重命名为。**CreateMeetingRequest**

现在，页面已添加，您将在页面中添加一个表单，最终用户将使用该表单输入信息，以便在`MeetingRequests`实体中创建会议请求。App Studio 提供了一种从现有实体生成表单的方法，该方法可以根据实体的字段自动填充表单字段，还可以生成一个提交按钮，用于在实体中使用表单输入创建记录。

在会议请求创建页面上自动生成来自实体的表单

1. 在右侧的 “组件” 菜单上，找到表单组件并将其拖到画布上。
2. 选择 “生成表单”。
3. 从下拉列表中选择实`MeetingRequests`体。
4. 选择 生成。
5. 选择画布上的 “提交” 按钮将其选中。
6. 在右侧属性面板的 “触发器” 部分，选择 + 添加。
7. 选择 “导航”。
8. 在右侧的属性面板中，将操作名称更改为描述性名称，例如。**`Navigate to MeetingRequestsDashboard`**
9. 将导航类型更改为页面。在 “导航至” 下拉列表中，选择**`MeetingRequestsDashboard`**。

现在我们有了会议请求创建页面和表单，我们希望可以轻松地从页面导航到此`MeetingRequestsDashboard`页面，以便查看仪表板的最终用户可以轻松创建会议请求。使用以下步骤在页面上创建一个用于导航到该`MeetingRequestsDashboardCreateMeetingRequest`页面的按钮。

添加从**`MeetingRequestsDashboard`**导航到的按钮 **`CreateMeetingRequest`**

1. 在顶部横幅中，选择页面。
2. 选择`MeetingRequestsDashboard`页面。
3. 在右侧的 “组件” 面板中，找到 B ut ton 组件，将其拖到画布上，然后将其放在表格上方。
4. 选择新添加的按钮将其选中。
5. 在右侧的 “属性” 面板中，更新以下设置：
   a. 选择铅笔图标将按钮重命名为**`createMeetingRequestButton`**。
   b. 按钮标签:**`Create Meeting Request`**. 这是最终用户将看到的名称。
   c. 在 “图标” 下拉列表中，选择 + Plus。
   d. 创建一个触发器，将最终用户导航到该`MeetingRequestsDashboard`页面：
      1. 在 “触发器” 部分中，选择 + 添加。
      2. 在操作类型中，选择导航。
      3. 选择您刚刚创建的触发器进行配置。

4. 在操作名称中，提供描述性名称，例如。**NavigateToCreateMeetingRequest**
5. 在导航类型下拉列表中，选择页面。
6. 在 “导航至” 下拉列表中，选择`CreateMeetingRequest`页面。

步骤 4：预览应用程序

您可以在 App Studio 中预览应用程序，以查看其对用户的显示效果。此外，您还可以通过使用它并在调试面板中查看日志来测试其功能。

应用程序预览环境不支持显示实时数据。它也不支持通过连接器与外部资源（例如数据源）进行连接。相反，您可以使用样本数据和模拟输出来测试功能。

预览您的应用程序以进行测试

1. 在应用程序构建器的右上角，选择预览。
2. 与`MeetingRequestsDashboard`页面互动，测试表格、表单和按钮。

步骤 5：将应用程序发布到测试环境

现在，您已经完成了应用程序的创建、配置和测试，是时候将其发布到测试环境以执行最终测试，然后与用户共享了。

将您的应用程序发布到测试环境

1. 在应用程序构建器的右上角，选择发布。
2. 为测试环境添加版本描述。
3. 查看并选中与 SLA 相关的复选框。
4. 选择启动。发布最多可能需要 15 分钟。
5. （可选）准备就绪后，您可以通过选择 “共享” 并按照提示向其他人授予访问权限。

> **Note**
>
> 要共享应用程序，管理员必须已创建最终用户组。

测试后，再次选择 “发布”，将应用程序提升到生产环境。有关不同应用程序环境的更多信息，请参阅应用程序环境。

后续步骤

现在您已经创建了第一个应用程序，接下来是一些后续步骤：

1. 继续构建教程应用程序。现在，您已经配置了数据、一些页面和自动化，您可以添加其他页面和添加组件，以了解有关构建应用程序的更多信息。
2. 要了解有关构建应用程序的更多信息，请参阅生成器文档。具体而言，以下主题可能有助于探索：
   - 自动化操作参考
   - 组件参考
   - 使用组件和自动化与 Amazon 简单存储服务交互
   - 安全注意事项和缓解措施

   此外，以下主题包含有关本教程中讨论的概念的更多信息：
   - 预览、发布和共享应用程序
   - 在 App Studio 应用程序中创建实体

管理员文档

以下主题包含的信息可帮助在 App Studio 中管理第三方服务连接和访问权限的用户、用户和角色。

主题

- 在 App Studio 中管理访问权限和角色
- 使用连接器将 App Studio 连接到其他服务
- 删除 App Studio 实例

在 App Studio 中管理访问权限和角色

App Studio 管理员的职责之一是管理访问权限、角色和权限。以下主题包含有关 App Studio 中的角色以及如何添加用户、移除用户或更改其角色的信息。

使用 IAM 身份中心群组管理对 A AWS pp Studio 的访问权限。要将用户添加到您的 App Studio 实例，您必须：

- 将他们添加到已添加到 App Studio 的现有 IAM 身份中心群组中。
- 将他们添加到未添加到 App Studio 的新的或现有的 IAM 身份中心群组中，然后将其添加到 App Studio。

由于角色适用于群组，因此 IAM Identity Center 群组应代表您要分配给群组成员的访问权限（或角色）。有关 IAM Identity Center 的更多信息，包括有关管理用户和群组的信息，请参阅 IAM 身份中心用户指南。

角色和权限

App Studio 中有三个角色。以下列表包含每个角色及其描述。

- 管理员：管理员可以在 App Studio 中管理用户和群组、添加和管理连接器以及管理构建器创建的应用程序。此外，具有管理员角色的用户拥有构建者角色中包含的所有权限。
- 生成器：生成器可以创建和构建应用程序。构建器无法管理用户或群组、添加或编辑连接器实例，也无法管理其他构建器的应用程序。
- 应用程序用户：应用程序用户可以访问和使用已发布的应用程序，但不能访问您的 App Studio 实例来构建应用程序或管理资源。

在 App Studio 中，角色被分配给群组，因此，将为已添加的 IAM Identity Center 群组中的每位成员分配分配给该群组的角色。

查看群组

执行以下步骤以查看添加到您的 App Studio 实例的群组。

> **Note**
> 您必须是管理员才能在 App Studio 实例中查看群组。

查看添加到您的 App Studio 实例的群组

- 在导航窗格中，选择管理部分中的角色。您将进入一个页面，其中显示了现有群组的列表以及每个群组的分配角色。

  有关管理群组的信息，请参阅添加用户或群组更改群组的角色、或从 App Studio 移除用户或群组。

添加用户或群组

要将用户添加到 App Studio，您必须将他们添加到 IAM 身份中心群组并将该群组添加到 App Studio。执行以下步骤，通过添加 IAM 身份中心群组并分配角色，将用户添加到 App Studio。

> **Note**
> 您必须是管理员才能将用户添加到您的 App Studio 实例。

将用户或群组添加到您的 App Studio 实例

1. 要将用户添加到您的 App Studio 实例，您必须将他们添加到已添加到 App Studio 的现有 IAM 身份中心群组，或者创建一个新的 IAM Identity Center 群组，向其中添加新用户，然后将新群组添加到 App Studio。

   有关管理 IAM Identity Center 用户和群组的信息，请参阅AWS IAM Identity Center 用户指南中的在 IAM 身份中心管理身份。

2. 如果您将用户添加到已添加到 App Studio 的现有 IAM 身份中心群组，则新用户在完成 IAM 身份中心权限设置后，可以使用指定权限访问 App Studio。如果您创建了新的 IAM Identity Center 群组，请执行以下步骤将该群组添加到 App Studio 并为该群组的成员指定一个角色。
3. 在导航窗格中，选择管理部分中的角色。
4. 在 “角色” 页面上，选择 + 添加群组。这将打开 “添加群组” 对话框供您输入群组的相关信息。
5. 在添加群组对话框中，输入以下信息：
   - 在下拉列表中选择现有的 IAM 身份中心组。
   - 为群组选择一个角色。
     - 管理员：管理员可以在 App Studio 中管理用户和群组、添加和管理连接器以及管理构建器创建的应用程序。此外，具有管理员角色的用户拥有构建者角色中包含的所有权限。
     - 生成器：生成器可以创建和构建应用程序。构建器无法管理用户或群组、添加或编辑连接器实例，也无法管理其他构建器的应用程序。
     - 应用程序用户：应用程序用户可以访问和使用已发布的应用程序，但不能访问您的 App Studio 实例来构建应用程序或管理资源。
6. 选择 “分配” 将群组添加到 App Studio 并为其成员提供已配置的角色。

更改群组的角色

按照以下步骤在 App Studio 中更改分配给群组的角色。更改群组的角色将改变该群组中每个成员的角色。

> **Note**
>
> 您必须是管理员才能在 App Studio 中更改群组的角色。

更改群组的角色

1. 在导航窗格中，选择管理部分中的角色。您将进入一个页面，其中显示了现有群组的列表以及每个群组的分配角色。
2. 选择省略号图标 (...)，然后选择 “更改角色”。
3. 在 “更改角色” 对话框中，为该组选择一个新角色：
   - 管理@@ 员：管理员可以在 App Studio 中管理用户和群组、添加和管理连接器以及管理构建器创建的应用程序。此外，具有管理员角色的用户拥有构建者角色中包含的所有权限。

- 生成器：生成器可以创建和构建应用程序。构建器无法管理用户或群组、添加或编辑连接器实例，也无法管理其他构建器的应用程序。
- 应用程序用户：应用程序用户可以访问和使用已发布的应用程序，但不能访问您的 App Studio 实例来构建应用程序或管理资源。

4. 选择更改更改群组的角色。

从 App Studio 移除用户或群组

您无法从 App Studio 中移除 IAM 身份中心群组。相反，执行以下说明会将群组的角色降级为应用程序用户。群组成员仍然可以访问已发布的 App Studio 应用程序。

要移除对 App Studio 及其应用程序的所有访问权限，您必须在 AWS IAM Identity Center 控制台中删除 IAM Identity Center 群组或用户。有关管理 IAM Identity Center 用户和群组的信息，请参阅AWS IAM Identity Center 用户指南中的在 IAM 身份中心管理身份。

> **Note**
>
> 您必须是管理员才能在 App Studio 中降级群组的访问权限。

移除群组

1. 在导航窗格中，选择管理部分中的角色。您将进入一个页面，其中显示了现有群组的列表以及每个群组的分配角色。
2. 选择省略号图标 (...)，然后选择 “撤消角色”。
3. 在 “撤消角色” 对话框中，选择 “撤消”，将群组的角色降级为 “应用程序用户”。

使用连接器将 App Studio 连接到其他服务

连接器是 App Studio 与其他 AWS 服务（例如 AWS Lambda 和 Amazon Redshift）或第三方服务之间的连接。创建并配置连接器后，构建者可以在其应用程序中使用该连接器及其连接到 App Studio 的资源。

只有具有管理员角色的用户才能创建、管理或删除连接器。

主题

- Connect 连接到 AWS 服务
- Connect 连接到第三方服务
- 查看、编辑和删除连接器

Connect 连接到 AWS 服务

主题

- 连接亚马逊 Redshift
- 连接亚马逊 DynamoDB
- 连接到 AWS Lambda
- 连接到亚马逊简单存储服务 (Amazon S3) Service
- 连接亚马逊 Aurora
- Connect 到 Amazon Bedrock
- Connect 到 Amazon 简单电子邮件服务
- 使用 “其他 AWS 服务” 连接器连接到 AWS 服务
- 将加密数据源与 CMKs

连接亚马逊 Redshift

要将 App Studio 与 Amazon Redshift 连接以使构建者能够在应用程序中访问和使用 Amazon Redshift 资源，您必须执行以下步骤：

1. 第 1 步：创建和配置亚马逊 Redshift 资源
2. 第 2 步：创建具有相应亚马逊 Redshift 权限的 IAM 策略和角色
3. 第 3 步：创建亚马逊 Redshift 连接器

第 1 步：创建和配置亚马逊 Redshift 资源

使用以下过程创建和配置要用于 App Studio 的 Amazon Redshift 资源。

设置 Amazon Redshift 以与 App Studio 配合使用

1. 登录 AWS Management Console 并打开亚马逊 Redshift 控制台，网址为。https://console.aws.amazon.com/redshiftv2/

我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。

2. 创建 Redshift 无服务器数据仓库或预配置集群。有关更多信息，请参阅《亚马逊 Redshift 用户指南》中的 “使用 Redshift Serverless 创建数据仓库” 或 “创建集群”。
3. 配置完成后，选择 “查询数据” 以打开查询编辑器。连接到数据库。
4. 更改以下设置：

   1. 将 “隔离会话” 切换开关设置为`OFF`。这是必需的，这样您才能看到其他用户所做的数据更改，例如来自正在运行的 App Studio 应用程序所做的更改。
   2. 选择 “齿轮” 图标。选择 Account settings (账户设置)。将最大并发连接数增加到`10`。这是可以连接到 Amazon Redshift 数据库的查询编辑器会话数量的限制。它不适用于其他客户端，例如 App Studio 应用程序。
5. 在`public`架构下创建您的数据表。 `INSERT`这些表中的任何初始数据。
6. 在查询编辑器中运行以下命令：

   以下命令创建数据库用户并将其与 App Studio *`AppBuilderDataAccessRole`* 使用的名为 IAM 角色关联起来。您将在后面的步骤中创建 IAM 角色，此处的名称必须与为该角色指定的名称相匹配。

```
CREATE USER "IAMR:AppBuilderDataAccessRole" WITH PASSWORD DISABLE;
```

   以下命令向 App Studio 授予所有表的所有权限。

> **Note**
>
> 为了获得最佳安全实践，您应将此处的权限范围缩小到相应表所需的最低权限。有关该`GRANT`命令的更多信息，请参阅 Amazon Redshift 数据库开发者指南中的授权。

```
GRANT ALL ON ALL TABLES IN SCHEMA public to "IAMR:AppBuilderDataAccessRole";
```

第 2 步：创建具有相应亚马逊 Redshift 权限的 IAM 策略和角色

要在 App Studio 中使用 Amazon Redshift 资源，管理员必须创建 IAM 策略和角色来授予 App Studio 访问资源的权限。IAM 策略控制构建者可以使用的数据范围以及可以针对这些数据调用的操作，例如创建、读取、更新或删除。然后，IAM 策略将附加到 App Studio 使用的 IAM 角色。

我们建议为每个服务和策略至少创建一个 IAM 角色。例如，如果构建者在 Amazon Redshift 中创建两个由不同表支持的应用程序，则管理员应创建两个 IAM 策略和角色，分别为 Amazon Redshift 中的每个表创建一个。

步骤 2a：创建具有相应亚马逊 Redshift 权限的 IAM 策略

您在 App Studio 中创建和使用的 IAM 策略应仅包含应用程序遵循最佳安全实践所需的相应资源的最低限度权限。

创建具有相应亚马逊 Redshift 权限的 IAM 策略

1. 使用有权创建 IAM 策略的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在左侧导航栏中，选择策略。
3. 选择创建策略。
4. 在策略编辑器部分，选择 JSON 选项。
5. 键入或粘贴 JSON 策略文档。以下选项卡包含预配置和无服务器 Amazon Redshift 的示例策略。

> **Note**
>
> 以下政策适用于所有使用通配符 () 的 Amazon Redshift 资源。*为了获得最佳安全实践，您应将通配符替换为要在 App Studio 中使用的资源的亚马逊资源名称 (ARN)。

Provisioned

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "ProvisionedRedshiftForAppStudio",
      "Effect": "Allow",
         "Action": [
            "redshift:DescribeClusters",
            "redshift:GetClusterCredentialsWithIAM",
            "redshift-data:ListDatabases",
            "redshift-data:ListTables",
            "redshift-data:DescribeTable",
            "redshift-data:DescribeStatement",
            "redshift-data:ExecuteStatement",
```

```
            "redshift-data:GetStatementResult"
         ],
         "Resource": "*"
      }
   ]
}
```

Serverless

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "ServerlessRedshiftForAppStudio",
      "Effect": "Allow",
         "Action": [
            "redshift-serverless:ListNamespaces",
            "redshift-serverless:GetCredentials",
            "redshift-serverless:ListWorkgroups",
            "redshift-data:ListDatabases",
            "redshift-data:ListTables",
            "redshift-data:DescribeTable",
            "redshift-data:DescribeStatement",
            "redshift-data:ExecuteStatement",
            "redshift-data:GetStatementResult"
         ],
         "Resource": "*"
      }
   ]
}
```

6. 选择下一步。
7. 在查看并创建页面上，提供策略名称（例如**RedshiftServerlessForAppStudio**或**RedshiftProvisionedForAppStudio**）和描述（可选）。
8. 选择创建策略以创建策略。

步骤 2b：创建一个 IAM 角色以授予 App Studio 访问亚马逊 Redshift 资源的权限

现在，创建一个使用您之前创建的策略的 IAM 角色。App Studio 将使用此策略来访问已配置的 Amazon Redshift 资源。

创建 IAM 角色以授予 App Studio 访问亚马逊 Redshift 资源的权限

1. 使用有权创建 IAM 角色的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在左侧导航栏中，选择角色
3. 选择创建角色。
4. 在可信实体类型中，选择自定义信任策略。
5. 将默认策略替换为以下策略，以允许 App Studio 应用程序在您的账户中扮演此角色。

   您必须替换策略中的以下占位符。要使用的值可以在 App Studio 的 “帐户设置” 页面中找到。

   - *111122223333*替换为 AWS 用于设置 App Studio 实例的账户的账号，该AWS 账号在 App Studio 实例的账户设置中作为账户 ID 列出。
   - *11111111-2222-3333-4444-555555555555*替换为您的 App Studio 实例 ID，该实例在 App Studi o 实例的账户设置中作为实例 ID 列出。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

   选择下一步。

6. 在添加权限中，搜索并选择您在上一步中创建的策略（**RedshiftServerlessForAppStudio**或**RedshiftProvisionedForAppStudio**）。选择策略旁边的 + 将展开策略以显示其授予的权限，选中该复选框将选择该策略。

   选择下一步。
7. 在 “名称、查看和创建” 页面上，提供角色名称和描述。

   > **Important**
   >
   > 此处的角色名称必须与 第 1 步：创建和配置亚马逊 Redshift 资源 (*AppBuilderDataAccessRole*) 中GRANT命令中使用的角色名称相匹配。

8. 在步骤 3：添加标签中，选择添加新标签以添加以下标签以提供 App Studio 访问权限：
   - 密钥：`IsAppStudioDataAccessRole`
   - 值：`true`
9. 选择 “创建角色” 并记下生成的亚马逊资源名称 (ARN)，在 App Studio 中创建 Amazon Redshift 连接器时，您将需要该名称。

第 3 步：创建亚马逊 Redshift 连接器

现在，您已经配置了 Amazon Redshift 资源以及 IAM 策略和角色，请使用这些信息在 App Studio 中创建连接器，构建者可以使用该连接器将其应用程序连接到 Amazon Redshift。

> **Note**
>
> 您必须在 App Studio 中拥有管理员角色才能创建连接器。

为亚马逊 Redshift 创建连接器

1. 导航到 App Studio。
2. 在左侧导航栏的管理区域，选择连接器。您将进入一个页面，其中显示了现有连接器的列表，其中包含每个连接器的一些详细信息。
3. 选择 + 创建连接器。
4. 选择亚马逊 Redshift 连接器。
5. 通过填写以下字段来配置您的连接器：

- 名称：为您的连接器提供一个名称。
- 描述：为您的连接器提供描述。
- IAM 角色：输入中创建的 IAM 角色的亚马逊资源名称 (ARN)。步骤 2b：创建一个 IAM 角色以授予 App Studio 访问亚马逊 Redshift 资源的权限有关 IAM 的更多信息，请参阅《IAM 用户指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html。
- 区域：选择您的亚马逊 Redshift 资源所在的地 AWS 区。
- 计算类型：选择您使用的是 Amazon Redshift Serverless 还是预配置集群。
- 选择@@ 集群或工作组：如果选择 “已配置”，请选择要连接到 App Studio 的集群。如果选择了无服务器，请选择工作组。
- 数据库选择：选择要连接到 App Studio 的数据库。
- 可用表：选择要连接到 App Studio 的表。

6. 选择下一步。查看连接信息并选择创建。
7. 新创建的连接器将出现在连接器列表中。

连接亚马逊 DynamoDB

要将 App Studio 与 DynamoDB 连接以使构建者能够在应用程序中访问和使用 DynamoDB 资源，您必须执行以下步骤：

1. 步骤 1：创建和配置 DynamoDB 资源
2. 步骤 2：创建具有相应 DynamoDB 权限的 IAM 策略和角色
3. 创建 DynamoDB 连接器

步骤 1：创建和配置 DynamoDB 资源

使用以下过程创建和配置要用于 App Studio 的 DynamoDB 资源。

设置 DynamoDB 以与 App Studio 配合使用

1. 登录 AWS Management Console 并打开 DynamoDB 控制台，网址为。https://console.aws.amazon.com/dynamodb/

   我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在左侧导航窗格中，选择 表。
3. 选择创建表。

4. 输入桌子的名称和密钥。
5. 选择创建表。
6. 创建表格后，向其中添加一些项目，以便在表格连接到 App Studio 后它们就会出现。
   a. 选择您的表格，选择操作，然后选择浏览项目。
   b. 在退回的商品中，选择创建商品。
   c. （可选）：选择添加新属性可向表中添加更多属性。
   d. 为每个属性输入值，然后选择创建项目。

步骤 2：创建具有相应 DynamoDB 权限的 IAM 策略和角色

要在 App Studio 中使用 DynamoDB 资源，管理员必须创建 IAM 策略和角色来授予 App Studio 访问资源的权限。IAM 策略控制构建者可以使用的数据范围以及可以对这些数据调用的操作，例如创建、读取、更新或删除。然后，IAM 策略将附加到 App Studio 使用的 IAM 角色。

我们建议为每项服务和策略至少创建一个 IAM 角色。例如，如果构建者要在 DynamoDB 中创建两个由相同表支持的应用程序，一个只需要读取权限，另一个需要读取、创建、更新和删除；管理员应创建两个 IAM 角色，一个使用只读权限，另一个拥有对 DynamoDB 中适用表的完整 CRUD 权限。

步骤 2a：创建具有相应的 DynamoDB 权限的 IAM 策略

您在 App Studio 中创建和使用的 IAM 策略应仅包含应用程序遵循最佳安全实践所需的相应资源的最低限度权限。

创建具有相应 DynamoDB 权限的 IAM 策略

1. 使用有权创建 IAM 策略的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在左侧导航栏中，选择策略。
3. 选择创建策略。
4. 在策略编辑器部分，选择 JSON 选项。
5. 键入或粘贴 JSON 策略文档。以下选项卡包含对 DynamoDB 表的只读和完全访问权限的策略示例，以及包含使用客户管理密钥 (CMK) 加密的 DynamoDB 表的 AWS KMS 权限的策略示例。AWS KMS

> **Note**
>
> 以下策略适用于使用通配符 () 的所有 DynamoDB 资源。*为了获得最佳安全实践，您应将通配符替换为要在 App Studio 中使用的资源的亚马逊资源名称 (ARN)。

Read only

以下策略授予对已配置的 DynamoDB 资源的读取权限。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "ReadOnlyDDBForAppStudio",
      "Effect": "Allow",
         "Action": [
            "dynamodb:ListTables",
            "dynamodb:DescribeTable",
            "dynamodb:PartiQLSelect"
         ],
         "Resource": "*"
      }
   ]
}
```

Full access

以下策略授予对已配置的 DynamoDB 资源的创建、读取、更新和删除权限。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "FullAccessDDBForAppStudio",
      "Effect": "Allow",
         "Action": [
           "dynamodb:ListTables",
           "dynamodb:DescribeTable",
           "dynamodb:PartiQLSelect",
           "dynamodb:PartiQLInsert",
```

```
            "dynamodb:PartiQLUpdate",
            "dynamodb:PartiQLDelete"
         ],
         "Resource": "*"
      }
   ]
}
```

Read only - KMS encrypted

以下策略通过提供权限来授予对配置的加密 DynamoDB 资源的读取权限。 AWS KMS 您必须将 ARN 替换为密钥的 ARN。 AWS KMS

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "ReadOnlyDDBForAppStudio",
      "Effect": "Allow",
         "Action": [
            "dynamodb:ListTables",
            "dynamodb:DescribeTable",
            "dynamodb:PartiQLSelect"
         ],
         "Resource": "*"
      },
      {
      "Sid": "KMSPermissionsForEncryptedTable",
      "Effect": "Allow",
         "Action": [
            "kms:Decrypt",
            "kms:DescribeKey"
         ],
         "Resource": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      },

   ]
}
```

Full access - KMS encrypted

以下策略通过提供权限来授予对配置的加密 DynamoDB 资源的读取权限。 AWS KMS 您必须将 ARN 替换为密钥的 ARN。 AWS KMS

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "ReadOnlyDDBForAppStudio",
      "Effect": "Allow",
         "Action": [
           "dynamodb:ListTables",
           "dynamodb:DescribeTable",
           "dynamodb:PartiQLSelect",
           "dynamodb:PartiQLInsert",
           "dynamodb:PartiQLUpdate",
           "dynamodb:PartiQLDelete"
         ],
         "Resource": "*"
      },
      {
      "Sid": "KMSPermissionsForEncryptedTable",
      "Effect": "Allow",
         "Action": [
            "kms:Decrypt",
            "kms:DescribeKey"
         ],
         "Resource": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      },

   ]
}
```

6. 选择下一步。
7. 在查看并创建页面上，提供策略名称（例如**ReadOnlyDDBForAppStudio**或**FullAccessDDBForAppStudio**）和描述（可选）。
8. 选择创建策略以创建策略。

步骤 2b：创建一个 IAM 角色以授予 App Studio 对 DynamoDB 资源的访问权限

现在，创建一个使用您之前创建的策略的 IAM 角色。App Studio 将使用此策略来访问已配置的 DynamoDB 资源。

创建 IAM 角色以授予 App Studio 访问 DynamoDB 资源的权限

1. 使用有权创建 IAM 角色的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在控制台的导航窗格中，选择 Roles，然后选择 Create role。
3. 在可信实体类型中，选择自定义信任策略。
4. 将默认策略替换为以下策略，以允许 App Studio 应用程序在您的账户中扮演此角色。

   您必须替换策略中的以下占位符。要使用的值可以在 App Studio 的 “帐户设置” 页面中找到。

   - *111122223333*替换为 AWS 用于设置 App Studio 实例的账户的账号，该AWS 账号在 App Studio 实例的账户设置中作为账户 ID 列出。
   - *11111111-2222-3333-4444-555555555555*替换为您的 App Studio 实例 ID，该实例在 App Studi o 实例的账户设置中作为实例 ID 列出。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

选择下一步。

5. 在添加权限中，搜索并选择您在上一步中创建的策略（**ReadOnlyDDBForAppStudio**或**FullAccessDDBForAppStudio**）。选择策略旁边的 + 将展开策略以显示其授予的权限，选中该复选框将选择该策略。

   选择下一步。
6. 在 “名称、查看和创建” 页面上，提供角色名称和描述。
7. 在步骤 3：添加标签中，选择添加新标签以添加以下标签以提供 App Studio 访问权限：

   - 密钥：`IsAppStudioDataAccessRole`
   - 值：`true`
8. 选择创建角色并记下生成的亚马逊资源名称 (ARN)，在 App Studio 中创建 DynamoDB 连接器时，您将需要该名称。

创建 DynamoDB 连接器

现在，您已经配置了 DynamoDB 资源以及 IAM 策略和角色，请使用这些信息在 App Studio 中创建连接器，构建者可以使用该连接器将其应用程序连接到 DynamoDB。

> **Note**
>
> 您必须在 App Studio 中拥有管理员角色才能创建连接器。

为 DynamoDB 创建连接器

1. 导航到 App Studio。
2. 在左侧导航栏的管理区域，选择连接器。您将进入一个页面，其中显示了现有连接器的列表，其中包含每个连接器的一些详细信息。
3. 选择 + 创建连接器。
4. 从连接器类型列表中选择 Amazon DynamoDB。
5. 通过填写以下字段来配置您的连接器：

   - 名称：输入您的 DynamoDB 连接器的名称。
   - 描述：输入您的 DynamoDB 连接器的描述。
   - IAM 角色：输入中创建的 IAM 角色的亚马逊资源名称 (ARN)。步骤 2b：创建一个 IAM 角色以授予 App Studio 对 DynamoDB 资源的访问权限有关 IAM 的更多信息，请参阅《IAM 用户指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html。

- 区域：选择您的 DynamoDB 资源所在的 AWS 区域。
- 可用表：选择要连接到 App Studio 的表。

6. 选择下一步。查看连接信息，然后选择创建。
7. 新创建的连接器将出现在连接器列表中。

连接到 AWS Lambda

要将 App Studio 与 Lambda 连接以使构建者能够在应用程序中访问和使用 Lambda 资源，您必须执行以下步骤：

1. 步骤 1：创建和配置 Lambda 函数
2. 第 2 步：创建一个 IAM 角色以授予 App Studio 访问 Lambda 资源的权限
3. 步骤 3：创建 Lambda 连接器

步骤 1：创建和配置 Lambda 函数

如果您没有现有 Lambda 函数，则必须先创建它们。要了解有关创建 Lambda 函数的更多信息，请参阅AWS Lambda 开发人员指南。

第 2 步：创建一个 IAM 角色以授予 App Studio 访问 Lambda 资源的权限

要在 App Studio 中使用 Lambda 资源，管理员必须创建一个 IAM 角色来授予 App Studio 访问资源的权限。IAM 角色控制应用程序可以从 Lambda 访问的资源或操作。

我们建议为每项服务和策略至少创建一个 IAM 角色。

创建 IAM 角色以授予 App Studio 访问 Lambda 资源的权限

1. 使用有权创建 IAM 角色的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在控制台的导航窗格中，选择 Roles，然后选择 Create role。
3. 在可信实体类型中，选择自定义信任策略。
4. 将默认策略替换为以下策略，以允许 App Studio 应用程序在您的账户中扮演此角色。

   您必须替换策略中的以下占位符。要使用的值可以在 App Studio 的 “帐户设置” 页面中找到。

   - *111122223333*替换为 AWS 用于设置 App Studio 实例的账户的账号，该AWS 账号在 App Studio 实例的账户设置中作为账户 ID 列出。

- *11111111-2222-3333-4444-555555555555*替换为您的 App Studio 实例 ID，该实例在 App Studi o 实例的账户设置中作为实例 ID 列出。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

选择下一步。

5. 在添加权限中，搜索并选择为角色授予相应权限的策略。选择策略旁边的 + 将展开策略以显示其授予的权限，选中该复选框将选择该策略。对于 Lambda，您可以考虑添加策略，该`AWSLambdaRole`策略授予调用 Lambda 函数的权限。

   有关在 Lambda 中使用 IAM 策略的更多信息，包括托管策略列表及其描述，请参阅AWS Lambda 开发人员指南 AWS Lambda中的身份和访问管理。

   选择下一步。

6. 在 “名称、查看和创建” 页面上，提供角色名称和描述。
7. 在步骤 3：添加标签中，选择添加新标签以添加以下标签以提供 App Studio 访问权限：
   - 密钥：`IsAppStudioDataAccessRole`
   - 值：`true`
8. 选择创建角色并记下生成的亚马逊资源名称 (ARN)，在 App Studio 中创建 Lambda 连接器时，您将需要该名称。

步骤 3：创建 Lambda 连接器

现在，您已经配置了 Lambda 资源以及 IAM 策略和角色，请使用这些信息在 App Studio 中创建连接器，构建者可以使用该连接器将其应用程序连接到 Lambda。

> **Note**
>
> 您必须在 App Studio 中拥有管理员角色才能创建连接器。

为 Lambda 创建连接器

1. 导航到 App Studio。
2. 在左侧导航栏的管理区域，选择连接器。您将进入一个页面，其中显示了现有连接器的列表，其中包含每个连接器的一些详细信息。
3. 选择 + 创建连接器。
4. 从连接器类型列表中选择 “其他 AWS 服务”。
5. 通过填写以下字段来配置您的连接器：
   - 名称：输入您的 Lambda 连接器的名称。
   - 描述：输入您的 Lambda 连接器的描述。
   - IAM 角色：输入中创建的 IAM 角色的亚马逊资源名称 (ARN)。第 2 步：创建一个 IAM 角色以授予 App Studio 访问 Lambda 资源的权限有关 IAM 的更多信息，请参阅《IAM 用户指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html。
   - 服务：选择 Lambda。
   - 区域：选择您的 Lambda 资源所在的 AWS 区域。
6. 选择创建。
7. 新创建的连接器将出现在连接器列表中。

连接到亚马逊简单存储服务 (Amazon S3) Service

要将 App Studio 与 Amazon S3 连接以使构建者能够在应用程序中访问和使用 Amazon S3 资源，请执行以下步骤：

1. 步骤 1：创建和配置 Amazon S3 资源
2. 第 2 步：创建具有相应的 Amazon S3 权限的 IAM 策略和角色

3. 步骤 3：创建 Amazon S3 连接器

在您完成步骤并使用适当权限创建连接器后，构建者可以使用连接器创建与 Amazon S3 资源交互的应用程序。有关在 App Studio 应用程序中与 Amazon S3 交互的更多信息，请参阅使用组件和自动化与 Amazon 简单存储服务交互。

步骤 1：创建和配置 Amazon S3 资源

根据应用程序的需求和现有资源，您可能需要创建一个 Amazon S3 存储桶，供应用程序写入和读取。有关创建 Amazon S3 资源（包括存储桶）的信息，请参阅《亚马逊简单存储服务用户指南》中的 Amazon S3 入门。

要在应用程序中使用该S3 上传组件，您必须将跨源资源共享 (CORS) 配置添加到要上传到的任何 Amazon S3 存储桶。CORS 配置授予 App Studio 向存储桶推送对象的权限。以下过程详细介绍了如何使用控制台向 Amazon S3 存储桶添加 CORS 配置。有关 CORS 及其配置的更多信息，请参阅 Amazon 简单存储服务用户指南中的使用跨源资源共享 (CORS)。

在控制台中向 Amazon S3 存储桶添加 CORS 配置

1. 在中导航到您的存储桶https://console.aws.amazon.com/s3/。
2. 选择 Permissions（权限）选项卡。
3. 在跨源资源共享 (CORS) 中，选择编辑。
4. 添加以下片段：

```
[
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "PUT",
            "POST"
        ],
        "AllowedOrigins": [
            "*"
        ],
        "ExposeHeaders": []
    }
]
```

5. 选择保存更改。

第 2 步：创建具有相应的 Amazon S3 权限的 IAM 策略和角色

要在 App Studio 中使用 Amazon S3 资源，管理员必须创建 IAM 策略和角色来授予 App Studio 访问资源的权限。IAM 策略控制构建者可以使用的数据范围以及可以针对这些数据调用的操作，例如创建、读取、更新或删除。然后，IAM 策略将附加到 App Studio 使用的 IAM 角色。

我们建议为每项服务和策略至少创建一个 IAM 角色。例如，如果构建者在 Amazon S3 中创建两个由不同存储桶支持的应用程序，则管理员应创建两个 IAM 策略和角色，每个策略和角色对应一个存储桶。

步骤 2a：创建具有相应的 Amazon S3 权限的 IAM 策略

您在 App Studio 中创建和使用的 IAM 策略应仅包含应用程序遵循最佳安全实践所需的相应资源的最低限度权限。

创建具有相应 Amazon S3 权限的 IAM 策略

1. 使用有权创建 IAM 策略的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在左侧导航栏中，选择策略。
3. 选择创建策略。
4. 在策略编辑器部分，选择 JSON 选项。
5. 键入或粘贴 JSON 策略文档。以下选项卡包含针对 Amazon S3 资源的只读和完全访问权限的示例策略。

> **Note**
>
> 以下策略适用于所有使用通配符 (*) 的 Amazon S3 资源。为了获得最佳安全实践，您应将通配符替换为要用于 App Studio 的资源（例如存储桶或文件夹）的亚马逊资源名称 (ARN)。

Read only

以下策略授予对已配置的 Amazon S3 存储桶或文件夹的只读访问权限（获取和列出）。

```
{
```

```
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "S3ReadOnlyForAppStudio",
         "Effect": "Allow",
         "Action": [
            "s3:GetObject",
            "s3:ListBucket"
         ],
         "Resource": "*"
      }
   ]
}
```

Full access

以下策略授予对已配置的 Amazon S3 存储桶或文件夹的完全访问权限（放置、获取、列出和删除）。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "S3FullAccessForAppStudio",
         "Effect": "Allow",
         "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:ListBucket",
            "s3:DeleteObject"
         ],
         "Resource": "*"
      }
   ]
}
```

6. 选择下一步。
7. 在 “查看并创建” 页面上，提供策略名称（例如**AWSAppStudioS3FullAccess**）和描述（可选）。
8. 选择创建策略以创建策略。

步骤 2b：创建一个 IAM 角色以授予 App Studio 访问亚马逊 S3 资源的权限

要在 App Studio 中使用 Amazon S3 资源，管理员必须创建一个 IAM 角色来授予 App Studio 访问资源的权限。IAM 角色控制构建者可以使用的数据范围以及可以针对这些数据调用的操作，例如创建、读取、更新或删除。

我们建议为每项服务和策略至少创建一个 IAM 角色。

创建 IAM 角色以授予 App Studio 访问 Amazon S3 资源的权限

1. 使用有权创建 IAM 角色的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在控制台的导航窗格中，选择 Roles，然后选择 Create role。
3. 在可信实体类型中，选择自定义信任策略。
4. 将默认策略替换为以下策略，以允许 App Studio 应用程序在您的账户中扮演此角色。

   您必须替换策略中的以下占位符。要使用的值可以在 App Studio 的 “帐户设置” 页面中找到。

   - *111122223333*替换为 AWS 用于设置 App Studio 实例的账户的账号，该AWS 账号在 App Studio 实例的账户设置中作为账户 ID 列出。
   - *11111111-2222-3333-4444-555555555555*替换为您的 App Studio 实例 ID，该实例在 App Studi o 实例的账户设置中作为实例 ID 列出。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
```

```
}
```

选择下一步。

5. 在添加权限中，搜索并选择您在上一步中创建的策略（**S3ReadOnlyForAppStudio**或**S3FullAccessForAppStudio**）。选择策略旁边的 + 将展开策略以显示其授予的权限，选中该复选框将选择该策略。

   选择下一步。

6. 在 “名称、查看和创建” 页面上，提供角色名称和描述。
7. 在步骤 3：添加标签中，选择添加新标签以添加以下标签以提供 App Studio 访问权限：
   - 密钥：`IsAppStudioDataAccessRole`
   - 值：`true`
8. 选择创建角色并记下生成的亚马逊资源名称 (ARN)，下一步您将需要它来在 App Studio 中创建 Amazon S3 连接器。

步骤 3：创建 Amazon S3 连接器

现在，您已经配置了 Amazon S3 资源以及 IAM 策略和角色，请使用这些信息在 App Studio 中创建连接器，构建者可以使用该连接器将其应用程序连接到 Amazon S3。

> **Note**
>
> 您必须在 App Studio 中拥有管理员角色才能创建连接器。

为 Amazon S3 创建连接器

1. 导航到 App Studio。
2. 在左侧导航栏的管理区域，选择连接器。您将进入一个页面，其中显示了现有连接器的列表，其中包含每个连接器的一些详细信息。
3. 选择 + 创建连接器。
4. 选择 Amazon S3 连接器。
5. 通过填写以下字段来配置您的连接器：
   - 名称：输入您的 Amazon S3 连接器的名称。

- 描述：输入您的 Amazon S3 连接器的描述。
- IAM 角色：输入中创建的 IAM 角色的亚马逊资源名称 (ARN)。步骤 2b：创建一个 IAM 角色以授予 App Studio 访问亚马逊 S3 资源的权限有关 IAM 的更多信息，请参阅《IAM 用户指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html。
- 区域：选择您的 Amazon S3 资源所在的 AWS 区域。

6. 选择创建。
7. 新创建的连接器将出现在连接器列表中。

连接亚马逊 Aurora

要将 App Studio 与 Aurora 连接以使构建者能够在应用程序中访问和使用 Aurora 资源，您必须执行以下步骤：

1. 步骤 1：创建和配置 Aurora 资源
2. 步骤 2：创建具有相应 Aurora 权限的 IAM 策略和角色
3. 步骤 3：在 App Studio 中创建 Aurora 连接器

App Studio 支持以下 Aurora 版本：

- Aurora MySQL 无服务器 V1：5.72
- Aurora PostgreSQL Serverless V1：11.18、13.9
- Aurora MySQL Serverless V2：13.11 或更高版本、14.8 或更高版本以及 15.3 或更高版本
- Aurora PostgreSQL Serverless V2：13.11 或更高版本、14.8 或更高版本以及 15.3 或更高版本

步骤 1：创建和配置 Aurora 资源

要在 App Studio 中使用 Aurora 数据库，必须先创建这些数据库并对其进行适当的配置。App Studio 支持两种 Aurora 数据库类型：Aurora PostgreSQL 和 Aurora MySQL。要比较这些类型，请参阅 MySQL 和 PostgreSQL 有什么区别？ 。选择相应的选项卡，然后按照步骤设置 Aurora 以与 App Studio 应用程序配合使用。

Aurora PostgreSQL

使用以下过程创建和配置要与 App Studio 配合使用的 Aurora PostgreSQL 数据库集群。

设置 Aurora 以便在 App Studio 中使用

1. 登录 AWS Management Console 并打开 Amazon RDS 控制台，网址为https://console.aws.amazon.com/rds/。
2. 选择创建数据库。
3. 选择 Aurora（兼容 PostgreSQL）。
4. 在可用版本中，选择任何大于或等于版本的版本`13.1``14.8`、和`15.3`。
5. 在设置中，输入数据库集群标识符。
6. 在实例配置中，选择 Serverless v2 并选择适当的容量。
7. 在 “连接” 中，选择启用 RDS 数据 API。
8. 在数据库身份验证中，选择 IAM 数据库身份验证。
9. 在其他配置中，在初始数据库名称中，输入数据库的初始数据库名称。

Aurora MySQL

使用以下过程创建和配置要与 App Studio 配合使用的 Aurora MySQL 数据库集群。

对于支持数据 API 或无服务器 v1 的版本，Aurora MySQL 不支持通过用户界面创建。要创建支持数据 API 的 Aurora MySQL 集群，必须使用 AWS CLI。

> **Note**
>
> 要在 App Studio 中使用 Aurora MySQL 数据库，它们必须位于虚拟私有云 (VPC) 中。有关更多信息，请参阅 Amazon Aurora 用户指南中的在 VPC 中使用数据库集群。

设置 Aurora MySQL 以与 App Studio 配合使用

1. 如有必要，请 AWS CLI 按照AWS Command Line Interface 用户指南中的安装或更新到最新版本 AWS CLI中的说明进行安装。
2. 登录 AWS Management Console 并打开 Amazon RDS 控制台，网址为https://console.aws.amazon.com/rds/。
3. 在左侧导航栏中，选择子网组。
4. 选择 Create DB subnet group（创建数据库子网组）。
5. 填写信息并创建子网组。有关子网组和使用的更多信息 VPCs，请参阅 Amazon Aurora 用户指南中的在 VPC 中使用数据库集群。

6. 运行以下 AWS CLI 命令：

```
aws rds create-db-cluster --database-name db_name \
    --db-cluster-identifier db_cluster_identifier \
    --engine aurora-mysql \
    --engine-version 5.7.mysql_aurora.2.08.3 \
    --engine-mode serverless \
    --scaling-configuration
 MinCapacity=4,MaxCapacity=32,SecondsUntilAutoPause=1000,AutoPause=true \
    --master-username userName \
    --master-user-password userPass \
    --availability-zones us-west-2b us-west-2c \
    --db-subnet-group-name subnet-group-name
```

替换以下字段：

- *db_name*替换为所需的数据库名称。
- *db_cluster_identifier*替换为所需的数据库集群标识符。
- （可选）根据需要替换`scaling-configuration`字段中的数字。
- *userName*替换为所需的用户名。
- *userPass*替换为所需的密码。
- 在中`availability-zones`，添加您创建的子网组中的可用区。
- *subnet-group-name*替换为您创建的子网组的名称。

步骤 2：创建具有相应 Aurora 权限的 IAM 策略和角色

要在 App Studio 中使用 Aurora 资源，管理员必须创建一个 IAM 策略并将其关联到一个 IAM 角色，该角色用于授予 App Studio 访问已配置资源的权限。IAM 策略和角色控制构建者可以使用的数据范围以及可以针对这些数据调用的操作，例如创建、读取、更新或删除。

我们建议为每个服务和策略至少创建一个 IAM 角色。

步骤 2a：创建具有相应 Aurora 权限的 IAM 策略

您在 App Studio 中创建和使用的 IAM 策略应仅包含应用程序遵循最佳安全实践所需的相应资源的最低限度权限。

创建具有相应 Aurora 权限的 IAM 策略

1. 使用有权创建 IAM 角色的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在左侧导航栏中，选择策略。
3. 选择创建策略。
4. 在策略编辑器部分，选择 JSON 选项。
5. 将现有代码段替换为以下代码段，*111122223333*替换为包含亚马逊 Redshift 和 Aurora 资源的 AWS 账号。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "BaselineAuroraForAppStudio",
            "Effect": "Allow",
            "Action": [
                "rds-data:ExecuteStatement",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:rds:*:111122223333:cluster:*",
                "arn:aws:secretsmanager:*:111122223333:secret:rds*"
            ]
        }
    ]
}
```

6. 选择下一步。
7. 在查看并创建页面上，提供策略名称，例如**Aurora_AppStudio**和描述（可选）。
8. 选择创建策略以创建策略。

步骤 2b：创建一个 IAM 角色以授予 App Studio 访问 Aurora 资源的权限

现在，创建一个使用您之前创建的策略的 IAM 角色。App Studio 将使用此策略来访问已配置的 Aurora 资源。

创建 IAM 角色以授予 App Studio 访问 Aurora 资源的权限

1. 使用有权创建 IAM 角色的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在控制台的导航窗格中，选择 Roles，然后选择 Create role。
3. 在可信实体类型中，选择自定义信任策略。
4. 将默认策略替换为以下策略，以允许 App Studio 应用程序在您的账户中扮演此角色。

   您必须替换策略中的以下占位符。要使用的值可以在 App Studio 的 “帐户设置” 页面中找到。

   - *111122223333*替换为 AWS 用于设置 App Studio 实例的账户的账号，该AWS 账号在 App Studio 实例的账户设置中作为账户 ID 列出。
   - *11111111-2222-3333-4444-555555555555*替换为您的 App Studio 实例 ID，该实例在 App Studi o 实例的账户设置中作为实例 ID 列出。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

   选择下一步。

5. 在添加权限中，搜索并选择您之前创建的策略（**Aurora_AppStudio**）。选择策略旁边的 + 将展开策略以显示其授予的权限，选中该复选框将选择该策略。

   选择下一步。

6. 在 “名称、查看和创建” 页面上，提供角色名称和描述。
7. 在步骤 3：添加标签中，选择添加新标签以添加以下标签以提供 App Studio 访问权限：
   - 密钥：`IsAppStudioDataAccessRole`
   - 值：`true`
8. 选择创建角色并记下生成的亚马逊资源名称 (ARN)，在 A pp Studio 中创建 Aurora 连接器时将需要该名称。

步骤 3：在 App Studio 中创建 Aurora 连接器

现在，您已经配置了 Aurora 资源以及 IAM 策略和角色，请使用这些信息在 App Studio 中创建连接器，构建者可以使用该连接器将其应用程序连接到 Aurora。

> **Note**
> 您必须在 App Studio 中拥有管理员角色才能创建连接器。

为 Aurora 创建连接器

1. 导航到 App Studio。
2. 在左侧导航栏的管理区域，选择连接器。您将进入一个页面，其中显示了现有连接器的列表，其中包含每个连接器的一些详细信息。
3. 选择 + 创建连接器。
4. 选择亚马逊 Aurora 连接器。
5. 通过填写以下字段来配置您的连接器：
   - 名称：输入 Aurora 连接器的名称。
   - 描述：输入您的 Aurora 连接器的描述。
   - IAM 角色：输入中创建的 IAM 角色的亚马逊资源名称 (ARN)。步骤 2b：创建一个 IAM 角色以授予 App Studio 访问 Aurora 资源的权限有关 IAM 的更多信息，请参阅《IAM 用户指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html。
   - 密钥 ARN：输入数据库集群的秘密 ARN。有关在何处可以找到密钥 ARN 的信息，请参阅 A mazon Aurora 用户指南中的查看有关数据库集群密钥的详细信息。
   - 区域：选择您的 Aurora 资源所在的 AWS 区域。

- 数据库 ARN：输入数据库集群的 ARN。ARN 可以在数据库集群的 “配置” 选项卡中找到，类似于秘密 ARN。
- 数据库类型：选择与中创建的数据库类型相匹配的数据库类型 M yS Q L 或 PostgreSQL。步骤 1：创建和配置 Aurora 资源
- 数据库名称：输入数据库的名称，该名称也可以在数据库集群的 “配置” 选项卡中找到。
- 可用表：使用此连接器选择要在 App Studio 中使用的表。

6. 选择 “下一步” 以查看或定义实体映射。
7. 选择创建以创建 Aurora 连接器。新创建的连接器将出现在连接器列表中。

Connect 到 Amazon Bedrock

要将 App Studio 与 Amazon Bedrock 连接起来，以便构建者可以在应用程序中访问和使用 Amazon Bedrock，您必须执行以下步骤：

1. 第 1 步：启用 Amazon Bedrock 模型
2. 第 2 步：创建具有相应的 Amazon Bedrock 权限的 IAM 策略和角色
3. 第 3 步：创建 Amazon Bedrock 连接器

第 1 步：启用 Amazon Bedrock 模型

使用以下步骤启用 Amazon Bedrock 模型。

启用 Amazon Bedrock 模型

1. 登录 AWS Management Console 并打开 Amazon Bedrock 控制台，网址为https://console.aws.amazon.com/bedrock/。
2. 在左侧导航窗格中，选择模型访问权限。
3. 启用要使用的模型。有关更多信息，请参阅管理对 Amazon Bedrock 基础模型的访问权限。

第 2 步：创建具有相应的 Amazon Bedrock 权限的 IAM 策略和角色

要在 App Studio 中使用 Amazon Bedrock 资源，管理员必须创建 IAM 策略和角色来授予 App Studio 访问资源的权限。IAM 策略控制可以对这些资源调用哪些资源以及哪些操作，例如`InvokeModel`。然后，IAM 策略将附加到 App Studio 使用的 IAM 角色。

步骤 2a：创建具有相应的 Amazon Bedrock 权限的 IAM 策略

您在 App Studio 中创建和使用的 IAM 策略应仅包含应用程序遵循最佳安全实践所需的相应资源的最低限度权限。

创建具有相应 Amazon Bedrock 权限的 IAM 策略

1. 使用有权创建 IAM 策略的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在左侧导航栏中，选择策略。
3. 选择创建策略。
4. 在策略编辑器部分，选择 JSON 选项。
5. 键入或粘贴 JSON 策略文档。以下示例策略使用通配符 () `InvokeModel` `*` 对所有 Amazon Bedrock 资源进行了规定。

   为了获得最佳安全实践，您应将通配符替换为要在 App Studio 中使用的资源的亚马逊资源名称 (ARN)。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "BedrockAccessForAppStudio",
         "Effect": "Allow",
         "Action": [
            "bedrock:InvokeModel"
         ],
         "Resource": "*"
      }
   ]
}
```

6. 选择下一步。
7. 在查看并创建页面上，提供策略名称（例如**`BedrockAccessForAppStudio`**）和描述（可选）。
8. 选择创建策略以创建策略。

步骤 2b：创建一个 IAM 角色以授予 App Studio 访问亚马逊 Bedrock 的权限

要将 Amazon Bedrock 与 App Studio 配合使用，管理员必须创建一个 IAM 角色来授予 App Studio 访问资源的权限。IAM 角色控制 App Studio 应用程序使用的权限范围，并在创建连接器时使用。我们建议为每个服务和策略至少创建一个 IAM 角色。

创建 IAM 角色以授予 App Studio 访问亚马逊 Bedrock 的权限

1. 使用有权创建 IAM 角色的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在控制台的导航窗格中，选择 Roles，然后选择 Create role。
3. 在可信实体类型中，选择自定义信任策略。
4. 将默认策略替换为以下策略，以允许 App Studio 应用程序在您的账户中扮演此角色。

   您必须替换策略中的以下占位符。要使用的值可以在 App Studio 的 “帐户设置” 页面中找到。

   - *111122223333*替换为 AWS 用于设置 App Studio 实例的账户的账号，该AWS 账号在 App Studio 实例的账户设置中作为账户 ID 列出。
   - *11111111-2222-3333-4444-555555555555*替换为您的 App Studio 实例 ID，该实例在 App Studi o 实例的账户设置中作为实例 ID 列出。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

选择下一步。

5. 在添加权限中，搜索并选择您在上一步中创建的策略（**`BedrockAccessForAppStudio`**）。选择策略旁边的 + 将展开策略以显示其授予的权限，选中该复选框将选择该策略。

   选择下一步。

6. 在 “名称、查看和创建” 页面上，提供角色名称和描述。
7. 在步骤 3：添加标签中，选择添加新标签以添加以下标签以提供 App Studio 访问权限：
   - 密钥：`IsAppStudioDataAccessRole`
   - 值：`true`
8. 选择创建角色并记下生成的亚马逊资源名称 (ARN)，下一步在 App Studio 中创建 Amazon Bedrock 连接器时将需要该名称。

第 3 步：创建 Amazon Bedrock 连接器

现在，您已经配置了 Amazon Bedrock 资源以及 IAM 策略和角色，请使用这些信息在 App Studio 中创建连接器，构建者可以使用该连接器将其应用程序连接到 Amazon Bedrock。

> **Note**
>
> 您必须在 App Studio 中拥有管理员角色才能创建连接器。

为 Amazon Bedrock 创建连接器

1. 导航到 App Studio。
2. 在左侧导航栏的管理区域，选择连接器。您将进入一个页面，其中显示了现有连接器的列表，其中包含每个连接器的一些详细信息。
3. 选择 + 创建连接器。
4. 从连接器类型列表中选择其他 AWS 服务。
5. 通过填写以下字段来配置您的连接器：
   - 名称：输入您的 Amazon Bedrock 连接器的名称。
   - 描述：输入您的 Amazon Bedrock 连接器的描述。

- IAM 角色：输入中创建的 IAM 角色的亚马逊资源名称 (ARN)。步骤 2b：创建一个 IAM 角色以授予 App Studio 访问亚马逊 Bedrock 的权限有关 IAM 的更多信息，请参阅《IAM 用户指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html。
- 服务：选择 B edrock 运行时。

> **Note**
>
> 基岩运行时用于对托管在 Amazon Bedrock 中的模型发出推理请求，而 B edrock 则用于管理、训练和部署模型。

- 区域：选择您的 AWS Amazon Bedrock 资源所在的地区。

6. 选择创建。
7. 新创建的连接器将出现在连接器列表中。

Connect 到 Amazon 简单电子邮件服务

要将 App Studio 与 Amazon SES 连接起来，使构建者能够使用它从其应用程序发送电子邮件通知，您必须执行以下步骤：

1. 步骤 1：配置 Amazon SES 资源
2. 第 2 步：创建具有相应的 Amazon SES 权限的 IAM 策略和角色
3. 第 3 步：创建 Amazon SES 连接器

步骤 1：配置 Amazon SES 资源

如果没有，则必须先将 Amazon SES 配置为使用它来发送电子邮件。要了解有关设置 Amazon SES 的更多信息，请参阅《亚马逊简单电子邮件服务开发者指南》中的 “亚马逊简单电子邮件服务入门”。

第 2 步：创建具有相应的 Amazon SES 权限的 IAM 策略和角色

要在 App Studio 中使用 Amazon SES 资源，管理员必须创建一个 IAM 角色来授予 App Studio 访问资源的权限。IAM 角色控制可以在 App Studio 应用程序中使用哪些 Amazon SES 功能或资源。

我们建议为每项服务和策略至少创建一个 IAM 角色。

步骤 2a：创建具有相应的 Amazon SES 权限的 IAM 策略

您在 App Studio 中创建和使用的 IAM 策略应仅包含应用程序遵循最佳安全实践所需的相应资源的最低限度权限。

创建具有相应 Amazon SES 权限的 IAM 策略

1. 使用有权创建 IAM 策略的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在左侧导航栏中，选择策略。
3. 选择创建策略。
4. 在策略编辑器部分，选择 JSON 选项。
5. 键入或粘贴以下 JSON 策略文档。

> **Note**
>
> 以下策略适用于所有使用通配符 (*) 的 Amazon SES 资源。为了获得最佳安全实践，您应将通配符替换为要在 App Studio 中使用的资源的亚马逊资源名称 (ARN)。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "ses:SendEmail",
            "Resource": "*"
        }
    ]
}
```

6. 选择下一步。
7. 在 “查看并创建” 页面上，提供策略名称（例如**SESForAppStudioPolicy**）和描述（可选）。
8. 选择创建策略以创建策略。

步骤 2b：创建一个 IAM 角色以授予 App Studio 访问亚马逊 SES 的权限

现在，创建一个使用您之前创建的策略的 IAM 角色。App Studio 将使用此政策来访问亚马逊 SES。

创建 IAM 角色以授予 App Studio 访问亚马逊 SES 的权限

1. 使用有权创建 IAM 角色的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在左侧导航栏中，选择角色
3. 选择创建角色。
4. 在可信实体类型中，选择自定义信任策略。
5. 将默认策略替换为以下策略，以允许 App Studio 应用程序在您的账户中扮演此角色。

   您必须替换策略中的以下占位符。要使用的值可以在 App Studio 的 “帐户设置” 页面中找到。

   - *111122223333*替换为 AWS 用于设置 App Studio 实例的账户的账号，该AWS 账号在 App Studio 实例的账户设置中作为账户 ID 列出。
   - *11111111-2222-3333-4444-555555555555*替换为您的 App Studio 实例 ID，该实例在 App Studi o 实例的账户设置中作为实例 ID 列出。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

   选择下一步。

6. 在添加权限中，搜索并选择您在上一步中创建的策略（**SESForAppStudioPolicy**）。选择策略旁边的 + 将展开策略以显示其授予的权限，选中该复选框将选择该策略。

选择下一步。

7. 在 “名称、查看和创建” 页面上，提供角色名称和描述。
8. 在步骤 3：添加标签中，选择添加新标签以添加以下标签以提供 App Studio 访问权限：
   - 密钥：`IsAppStudioDataAccessRole`
   - 值：`true`
9. 选择 “创建角色” 并记下生成的亚马逊资源名称 (ARN)，在 A pp Studio 中创建 Amazon SES 连接器时，您将需要该名称。

第 3 步：创建 Amazon SES 连接器

现在，您已经配置了 Amazon SES 和 IAM 策略和角色，请使用这些信息在 App Studio 中创建连接器，构建者可以使用该连接器在其应用程序中使用 Amazon SES。

> **Note**
>
> 您必须在 App Studio 中拥有管理员角色才能创建连接器。

为 Amazon SES 创建连接器

1. 导航到 App Studio。
2. 在左侧导航栏的管理区域，选择连接器。您将进入一个页面，其中显示了现有连接器的列表，其中包含每个连接器的一些详细信息。
3. 选择 + 创建连接器。
4. 从连接器类型列表中选择 “其他 AWS 服务”。
5. 通过填写以下字段来配置您的连接器：
   - 名称：输入您的 Amazon SES 连接器的名称。
   - 描述：输入您的 Amazon SES 连接器的描述。
   - IAM 角色：输入中创建的 IAM 角色的亚马逊资源名称 (ARN)。步骤 2b：创建一个 IAM 角色以授予 App Studio 访问亚马逊 SES 的权限有关 IAM 的更多信息，请参阅《IAM 用户指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html。
   - 服务：选择 “简单电子邮件服务”。
   - 区域：选择您的 AWS Amazon SES 资源所在的地区。

6. 选择创建。
7. 新创建的连接器将出现在连接器列表中。

使用 “其他 AWS 服务” 连接器连接到 AWS 服务

虽然 App Studio 提供了一些特定于某些 AWS 服务的连接器，但您也可以使用其他 AWS 服务连接器连接到其他 AWS 服务。

> **Note**
>
> 如果 AWS 服务专用的连接器可用，建议使用该连接器。

要将 App Studio 与 AWS 服务连接以使构建者能够在应用程序中访问和使用该服务的资源，您必须执行以下步骤：

1. 创建 IAM 角色以授予 App Studio 对 AWS 资源的访问权限
2. 创建其他 AWS 服务连接器

创建 IAM 角色以授予 App Studio 对 AWS 资源的访问权限

要在 App Studio 中使用 AWS 服务和资源，管理员必须创建一个 IAM 角色来授予 App Studio 访问资源的权限。IAM 角色控制构建者可以访问的资源范围以及可以对这些资源调用的操作。我们建议为每项服务和策略至少创建一个 IAM 角色。

创建 IAM 角色以授予 App Studio 对 AWS 资源的访问权限

1. 使用有权创建 IAM 角色的用户登录 IAM 控制台。我们建议使用中创建的管理用户创建管理用户来管理 AWS 资源。
2. 在控制台的导航窗格中，选择 Roles，然后选择 Create role。
3. 在可信实体类型中，选择自定义信任策略。
4. 将默认策略替换为以下策略，以允许 App Studio 应用程序在您的账户中扮演此角色。

   您必须替换策略中的以下占位符。要使用的值可以在 App Studio 的 “帐户设置” 页面中找到。

   - *111122223333*替换为 AWS 用于设置 App Studio 实例的账户的账号，该AWS 账号在 App Studio 实例的账户设置中作为账户 ID 列出。

- *11111111-2222-3333-4444-555555555555*替换为您的 App Studio 实例 ID，该实例在 App Studi o 实例的账户设置中作为实例 ID 列出。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

选择下一步。

5. 在添加权限中，搜索并选择为角色授予相应权限的策略。选择策略旁边的 + 将展开策略以显示其授予的权限，选中该复选框将选择该策略。有关 IAM 的更多信息，请参阅《IAM 用户指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html。

   选择下一步。

6. 在角色详细信息中，提供名称和描述。
7. 在步骤 3：添加标签中，选择添加新标签以添加以下标签以提供 App Studio 访问权限：
   - 密钥：`IsAppStudioDataAccessRole`
   - 值：`true`
8. 选择创建角色并记下生成的亚马逊资源名称 (ARN)，在 A pp Studio 中创建其他 AWS 服务连接器时，您将需要该名称。

创建其他 AWS 服务连接器

现在，您已经配置了 IAM 角色，请使用该信息在 App Studio 中创建连接器，构建者可以使用该连接器将其应用程序连接到服务和资源。

> **Note**
> 您必须在 App Studio 中拥有管理员角色才能创建连接器。

使用 “其他 AWS 服务” 连接器连接到 AWS 服务

1. 导航到 App Studio。
2. 在左侧导航栏的管理区域，选择连接器。
3. 选择 + 创建连接器。
4. 在 “支持的 AWS 服务” 列表的 “AWS 连接器” 部分中选择 “其他服务”。
5. 通过填写以下字段来配置您的 AWS 服务连接器：
   - 名称：为您的连接器提供一个名称。
   - 描述：为您的连接器提供描述。
   - IAM 角色：输入中创建的 IAM 角色的亚马逊资源名称 (ARN)。创建 IAM 角色以授予 App Studio 对 AWS 资源的访问权限
   - 服务：选择要连接到 App Studio 的 AWS 服务。
   - 区域：选择 AWS 您的 AWS 资源所在的地区。
6. 选择创建。新创建的连接器将出现在连接器列表中。

将加密数据源与 CMKs

本主题包含有关设置 App Studio 并将其连接到使用AWS KMS 客户托管密钥 (CMK) 加密的数据源的信息。

目录

- 使用加密的托管数据存储表
- 使用加密的 DynamoDB 表

使用加密的托管数据存储表

使用以下过程对 App Studio 应用程序中的托管存储实体使用的 DynamoDB 表进行加密。有关托管数据实体的更多信息，请参阅AWS App Studio 中的管理数据实体。

使用加密的托管数据存储表

1. 如有必要，在 App Studio 的应用程序中创建托管数据实体。有关更多信息，请参阅 使用 App Studio 托管数据源创建实体。
2. 通过执行以下步骤，向 `AppStudioManagedStorageDDBAccess` IAM 角色添加具有使用您的 CMK 加密和解密表数据的权限的策略声明：

   a. 使用 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。

   > **Important**
   > 您必须使用用于创建 App Studio 实例的相同帐户。

   b. 在 IAM 控制台的导航窗格中，选择角色。

   c. 选择 `AppStudioManagedStorageDDBAccess`。

   d. 在 “权限策略” 中，选择 “添加权限”，然后选择 “创建内联策略”。

   e. 选择 JSON 并将内容替换为以下策略，替换以下内容：

      - *111122223333*替换为 AWS 用于设置 App Studio 实例的账户的账号，该AWS 账号在 App Studio 实例的账户设置中作为账户 ID 列出。
      - 替换*CMK_id*为 CMK ID。要找到它，请参阅查找密钥 ID 和密钥 ARN。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "connector_cmk_support",
      "Effect": "Allow",
      "Action": [ "kms:Decrypt", "kms:Encrypt"],
      "Resource": "arn:aws:kms:us-west-2:111122223333:key/CMK_id"
    }
  ]
}
```

3. 通过执行以下步骤对您的 App Studio 托管数据实体使用的 DynamoDB 表进行加密：
   a. 打开亚马逊 DynamoDB 控制台，网址为。https://console.aws.amazon.com/dynamodbv2/
   b. 选择要加密的表。您可以在 App Studio 中相应实体的 “连接” 选项卡中找到表名。
   c. 选择其他设置。
   d. 在加密中，选择管理加密。
   e. 选择 “存储在您的账户中，由您拥有和管理”，然后选择您的 CMK。
4. 通过重新发布应用程序来测试您的更改，确保在测试和生产环境中都能读取和写入数据，并在其他实体中使用此表按预期工作。

> **Note**
> 默认情况下，任何新添加的托管数据实体都使用 DynamoDB 托管密钥，并且必须按照前面的步骤更新为使用 CMK。

使用加密的 DynamoDB 表

使用以下步骤配置要在您的 App Studio 应用程序中使用的加密 DynamoDB 表。

使用加密的 DynamoDB 表

1. 按照中的说明步骤 1：创建和配置 DynamoDB 资源进行以下更改：
   - 将您的表配置为加密。有关更多信息，请参阅 Amazon DynamoD B 开发者指南中的为新表指定加密密钥。
2. 按照中的说明进行操作步骤 2：创建具有相应 DynamoDB 权限的 IAM 策略和角色，然后通过添加新的策略声明来更新新角色的权限策略，允许该角色使用您的 CMK 加密和解密表数据，方法是执行以下步骤：
   a. 如有必要，请在 IAM 控制台中导航到您的角色。
   b. 在 “权限策略” 中，选择 “添加权限”，然后选择 “创建内联策略”。
   c. 选择 JSON 并将内容替换为以下策略，替换以下内容：
      - *team_account_id*替换为您的 App Studio 团队标识，该ID可以在您的帐户设置中找到。
      - 替换*CMK_id*为 CMK ID。要找到它，请参阅查找密钥 ID 和密钥 ARN。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "connector_cmk_support",
      "Effect": "Allow",
      "Action": [ "kms:Decrypt", "kms:Encrypt"],
      "Resource": "arn:aws:kms:us-west-2:team_account_id:key/CMK_id"
    }
  ]
}
```

3. 按照中的说明创建连接器，创建 DynamoDB 连接器并使用您之前创建的角色。
4. 通过将使用 DynamoDB 连接器和表格的应用程序发布到 “测试” 或 “生产” 来测试配置。确保读取和写入数据正常工作，并且使用此表创建另一个实体也能正常工作。

> **Note**
>
> 创建任何新的 DynamoDB 表时，必须按照前面的步骤将其配置为使用 CMK 进行加密。

Connect 连接到第三方服务

主题

- OpenAPI 连接器与 API 连接器
- Connect 连接到第三方服务和 APIs （通用）
- 使用 OpenAPI 连接到服务
- 连接 Salesforce

OpenAPI 连接器与 API 连接器

要从 App Studio 应用程序向第三方服务发送 API 请求，您必须创建和配置连接器，应用程序使用该连接器对服务进行身份验证并配置 API 调用。App Studio 提供了`API Connector`和`OpenAPI Connector`连接器类型来实现此目的，如下所述：

- API 连接器：用于为任何类型的 REST API 配置身份验证和请求信息。

- OpenAPI 连接器：用于配置已采用 OpenAPI 规范 (OAS) 的身份验证和请求信息。 APIs APIs 遵守美洲国家组织可以带来多项好处，包括标准化、安全、治理和文档。

App Studio 建议`OpenAPI Connector`对任何 APIs 符合 OAS 的用户使用，并提供 OpenAPI 规范文件。有关 OpenAPI 的更多信息，请参阅什么是 Open API？ 在 Swagger 文档中。

Connect 连接到第三方服务和 APIs （通用）

使用以下步骤在 App Studio 中创建通用 API 连接器。API 连接器用于向 App Studio 应用程序提供对第三方服务、资源或操作的访问权限。

使用 API 连接器连接到第三方服务

1. 在左侧导航栏的管理区域，选择连接器。您将进入一个页面，其中显示了现有连接器的列表，其中包含每个连接器的一些详细信息。
2. 选择 + 创建连接器。
3. 选择 API 连接器。现在，通过填写以下字段来配置您的连接器。
4. 连接器名称：为您的连接器提供一个名称。
5. 连接器描述：为您的连接器提供描述。
6. 基本 URL：第三方连接的网站或主机。例如 `www.slack.com`。
7. 身份验证方法：选择使用目标服务进行身份验证的方法。
   - 无：无需身份验证即可访问目标服务。
   - 基本：使用从所连接的服务中获取的用户名和密码访问目标服务。
   - 不记名令牌：使用从服务的用户账户或 API 设置中获取的身份验证令牌的令牌值访问目标服务。
   - OAuth 2.0：使用 OAuth 2.0 协议访问目标服务，该协议授予 App Studio 访问服务和资源的权限，而无需共享任何凭据或身份。要使用 OAuth 2.0 身份验证方法，必须先从所连接的服务中创建一个代表 App Studio 的应用程序，以获取必要的信息。使用这些信息，填写以下字段：

     a. 客户凭证流：非常适合应用程序在没有用户 system-to-system交互的情况下代表自己操作的交互。例如，根据用户添加的新记录自动更新 Salesforce 记录的 CRM 应用程序，或者在报告中检索和显示交易数据的应用程序。

        1. 在 “客户端 ID” 中，输入从目标服务中创建的 OAuth 应用程序中获取的 ID。
        2. 在客户端密钥中，输入从目标服务中创建的 OAuth 应用程序中获取的密钥。

3. 在访问令牌 URL 中，输入从目标服务中创建的 OAuth 应用程序中获取的令牌 URL。
4. 或者，在范围中，输入应用程序的范围。范围是应用程序所需的权限或访问级别。请参阅目标服务的 API 文档以了解其范围，并仅配置您的 App Studio 应用程序所需的范围。

选择 “验证连接” 以测试身份验证和连接。

b. 授权码流：非常适合需要代表用户执行操作的应用程序。例如，一个客户支持应用程序，用户可以在其中登录并查看和更新支持工单，或者一个销售应用程序，每个团队成员都可以在其中登录以查看和管理其销售数据。

1. 在 “客户端 ID” 中，输入从目标服务中创建的 OAuth 应用程序中获取的 ID。
2. 在客户端密钥中，输入从目标服务中创建的 OAuth 应用程序中获取的密钥。
3. 在授权 URL 中，输入来自目标服务的授权 URL。
4. 在访问令牌 URL 中，输入从目标服务中创建的 OAuth 应用程序中获取的令牌 URL。
5. 或者，在范围中，输入应用程序的范围。范围是应用程序所需的权限或访问级别。请参阅目标服务的 API 文档以了解其范围，并仅配置您的 App Studio 应用程序所需的范围。

8. 标头：添加用于提供有关请求或响应的元数据的 HTTP 标头。您可以同时添加键和值，也可以仅提供生成器可以在应用程序中为其提供值的密钥。
9. 查询参数：添加用于将选项、过滤器或数据作为请求网址的一部分传递的查询参数。与标题类似，您可以同时提供键和值，也可以仅提供生成器可以在应用程序中为其提供值的密钥。
10. 选择创建。新创建的连接器将出现在连接器列表中。

现在，连接器已创建，构建者可以在他们的应用程序中使用它。

使用 OpenAPI 连接到服务

要使用 OpenAPI 将 App Studio 与服务连接起来，使构建者能够构建能够发送请求和接收来自服务的响应的应用程序，请执行以下步骤：

1. 获取 OpenAPI 规范文件并收集服务信息
2. 创建 OpenAPI 连接器

获取 OpenAPI 规范文件并收集服务信息

要使用 OpenAPI 将服务连接到 App Studio，请执行以下步骤：

1. 转到你想要连接到 App Studio 的服务，然后找到 OpenAPI 规范 JSON 文件。

> **Note**
>
> App Studio 支持符合 OpenAPI 规范版本 3.0.0 或更高版本的 OpenAPI 规范文件。

2. 收集配置 OpenAPI 连接器所需的数据，包括以下内容：
   - 用于连接到服务的基本 URL。
   - 身份验证凭证，例如令牌或用户名/密码。
   - 任何标题（如果适用）。
   - 任何查询参数（如果适用）。

创建 OpenAPI 连接器

为 OpenAPI 创建连接器

1. 导航到 App Studio。
2. 在左侧导航栏的管理区域，选择连接器。您将进入一个页面，其中显示了现有连接器的列表，其中包含每个连接器的一些详细信息。
3. 选择 + 创建连接器。
4. 从连接器类型列表中选择 OpenAPI 连接器。现在，通过填写以下字段来配置您的连接器。
5. 名称：输入 OpenAPI 连接器的名称。
6. 描述：输入您的 OpenAPI 连接器的描述。
7. 基本 URL：输入用于连接到服务的基本 URL。
8. 身份验证方法：选择使用目标服务进行身份验证的方法。
   - 无：无需身份验证即可访问目标服务。
   - 基本：使用从所连接的服务中获取的用户名和密码访问目标服务。
   - 不记名令牌：使用从服务的用户账户或 API 设置中获取的身份验证令牌的令牌值访问目标服务。
   - OAuth 2.0：使用 OAuth 2.0 协议访问目标服务，该协议授予 App Studio 访问服务和资源的权限，而无需共享任何凭据或身份。要使用 OAuth 2.0 身份验证方法，必须先从所连接的服

务中创建一个代表 App Studio 的应用程序，以获取必要的信息。使用这些信息，填写以下字段：

a. 客户凭证流程：

1. 在客户端 ID 中，输入目标服务的 ID。
2. 在客户端密钥中，输入来自目标服务的密钥。
3. 在访问令牌 URL 中，输入来自目标服务的令牌 URL。
4. 或者，在范围中，输入应用程序的范围。范围是应用程序所需的权限或访问级别。请参阅目标服务的 API 文档以了解其范围，并仅配置您的 App Studio 应用程序所需的范围。

在每次调用时添加要随服务发送的所有变量，然后选择验证连接以测试身份验证和连接。

b. 授权码流程：

1. 在客户端 ID 中，输入目标服务的 ID。
2. 在客户端密钥中，输入来自目标服务的密钥。
3. 在授权 URL 中，输入来自目标服务的授权 URL。
4. 在访问令牌 URL 中，输入来自目标服务的令牌 URL。
5. 或者，在范围中，输入应用程序的范围。范围是应用程序所需的权限或访问级别。请参阅目标服务的 API 文档以了解其范围，并仅配置您的 App Studio 应用程序所需的范围。

9. 变量：添加每次调用时要发送到服务的变量。配置期间添加的变量会被安全存储，并且只有在使用连接的应用程序的运行时才能访问。
10. 标头：添加用于提供有关请求或响应的元数据的 HTTP 标头。您可以同时添加键和值，也可以仅提供生成器可以在应用程序中为其提供值的密钥。
11. 查询参数：添加用于将选项、过滤器或数据作为请求网址的一部分传递的查询参数。与标题类似，您可以同时提供键和值，也可以仅提供生成器可以在应用程序中为其提供值的密钥。
12. OpenAPI 规范文件：通过拖放来上传 OpenAPI 规范 JSON 文件，或者选择选择文件来浏览本地文件系统并选择要上传的文件。

    添加后，文件即被处理并显示可用选项列表。为您的连接器选择必要的操作。
13. 选择创建。新创建的连接器将出现在连接器列表中。

现在，连接器已创建，构建者可以在他们的应用程序中使用它。

连接 Salesforce

要将 App Studio 与 Salesforce 连接以使构建者能够在应用程序中访问和使用 Salesforce 资源，您必须在 Salesforce 中创建和配置连接的应用程序，并在 App Studio 中创建 Salesforce 连接器。

将 Salesforce 与 App Stud

1. 在 App Studio 的导航窗格中，选择管理部分的连接器。您将进入一个页面，其中显示了现有连接器的列表，其中包含每个连接器的一些详细信息。
2. 选择 + 创建连接器。
3. 从连接器类型列表中选择 Salesforce 以打开连接器创建页面。
4. 记下重定向网址，您将在以下步骤中使用该网址来配置 Salesforce。
5. 下一步是在 Salesforce 中创建一个联网应用程序。在另一个选项卡或窗口中，导航到您的 Salesforce 实例。
6. 在 “快速查找” 框中，搜索，**`App Manager`**然后选择 “应用程序管理器”。
7. 选择 “新建连接的应用程序”。
8. 在已连接的应用程序名称和 API 名称中，输入您的应用程序的名称。它不必与你的 App Studio 应用程序名称相匹配。
9. 根据需要提供联系信息。
10. 在 API（启用 OAuth 设置）部分，启用启用 OAuth 设置。
11. 在回调网址中，输入你之前在 App Studio 中记下的重定向网址。
12. 在选定 OAuth 范围中，从列表中添加必要的权限范围。App Studio 可以与 Salesforce REST APIs 交互，对五个对象执行 CRUD 操作：客户、案例、联系人、潜在客户和机会。建议添加完全访问权限（完整），以确保您的 App Studio 应用程序具有所有相关的权限或范围。
13. 禁用 “支持的授权流需要验证密钥进行代码交换 (PKCE) 扩展” 选项。App Studio 不支持 PKCE。
14. 启用 Web 服务器流需要密钥和刷新令牌流需要密钥以遵循最佳安全实践。
15. App Studio 支持以下两个身份验证流程：

    - 客户凭证流程：非常适合应用程序在没有用户 server-to-server交互的情况下代表自己操作的交互。例如，列出没有 Salesforce 访问权限的临时员工团队的所有潜在客户信息。
    - 授权码流程：适用于代表用户行事的应用程序，例如个人数据访问或操作。例如，列出每位销售经理获取或拥有的潜在客户，以便通过此应用程序执行其他任务。

    - 对于客户凭证流程：

a. 启用 “启用客户机凭证流”。查看并确认消息。

b. 保存应用程序。

c. 尽管流程中没有用户交互，但您必须选择执行用户。通过选择执行用户，Salesforce 代表该用户返回访问令牌。

   1. 在应用程序管理器中，从应用程序列表中选择 App Studio 应用程序的箭头，然后选择管理。
   2. 选择 “编辑策略”
   3. 在 “客户凭证流” 中，添加相应的用户。

- 对于授权码流程，启用启用授权码和凭证流程

16. Salesforce 提供了客户端 ID 和客户端密钥，在以下步骤中，必须使用它们在 App Studio 中配置连接器。

    a. 在应用程序管理器中，选择 App Studio 应用程序的箭头，然后选择查看。

    b. 在 API（启用 OAuth 设置）部分中，选择管理消费者详细信息。这可能会发送一封电子邮件以获取验证密钥，您需要输入验证密钥进行确认。

    c. 记下消费者密钥（客户端 ID）和消费者密钥（客户机密）。

17. 返回 App Studio，通过填写以下字段来配置和创建您的连接器。
18. 在名称中，输入您的 Salesforce 连接器的名称。
19. 在描述中，输入您的 Salesforce 连接器的描述。
20. 在基本网址中，输入您的 Salesforce 实例的基本网址。它应该看起来像这样:`https://hostname.salesforce.com/services/data/v60.0`，替换`hostname`为你的 Salesforce 实例名称。
21. 在 “身份验证方法” 中，确保选择 OAuth 2.0。
22. 在 OAuth 2.0 Flow 中，选择 OAuth 身份验证方法并填写相关字段：

    - 选择 “客户凭证流”，以便在代表自己行事的应用程序中使用以进行 system-to-system集成。

      a. 在 “客户端 ID” 中，输入之前从 Salesforce 获得的使用者密钥。

      b. 在客户密钥中，输入之前从 Salesforce 获得的消费者机密。

      c. 在访问令牌 URL 中，输入 OAuth 2.0 令牌端点。它应该看起来像这样:`https://hostname/services/oauth2/token`，替换`hostname`为你的 Salesforce 实例名称。有关更多信息，请参阅 Salesforce OAuth 终端节点文档。

      d. 选择 “验证连接” 以测试身份验证和连接。

- 选择 “授权代码流”，以便在代表用户执行操作的应用程序中使用。

  a. 在 “客户端 ID” 中，输入之前从 Salesforce 获得的使用者密钥。

  b. 在客户密钥中，输入之前从 Salesforce 获得的消费者机密。

  c. 在授权 URL 中，输入授权端点。它应该看起来像这样:`https://hostname/services/oauth2/authorize`，替换*hostname*为你的 Salesforce 实例名称。有关更多信息，请参阅 Salesforce OAuth 终端节点文档。

  d. 在访问令牌 URL 中，输入 OAuth 2.0 令牌端点。它应该看起来像这样:`https://hostname/services/oauth2/token`，替换*hostname*为你的 Salesforce 实例名称。有关更多信息，请参阅 Salesforce OAuth 终端节点文档。

23. 在操作中，选择您的连接器将支持的 Salesforce 操作。此列表中的操作是预定义的，代表 Salesforce 中的常见任务，例如创建、检索、更新或删除常用对象中的记录。

24. 选择创建。新创建的连接器将出现在连接器列表中。

查看、编辑和删除连接器

查看、编辑或删除现有连接器

1. 在导航窗格中，选择管理部分的连接器。您将进入一个显示现有连接器列表的页面，其中包含每个连接器的以下详细信息：

   - 名称：创建期间提供的连接器的名称。
   - 描述：创建期间提供的连接器的描述。
   - 已连接到：连接器连接到 App Studio 的服务。API 的值表示与第三方服务的连接。
   - 创建者：创建连接器的用户。
   - 创建日期：连接器的创建日期。

2. 要查看有关连接器的更多详细信息，或者编辑或删除连接器，请按照以下说明进行操作：

   - 要查看有关特定连接器的更多信息，请为该连接器选择查看。
   - 要编辑连接器，请选择 “查看” 旁边的下拉菜单，然后选择 “编辑”。
   - 要删除连接器，请选择 “查看” 旁边的下拉菜单，然后选择 “删除”。

删除 App Studio 实例

使用本主题中的步骤删除您的 App Studio 实例。如果您在其他服务中创建了用于 App Studio 的资源，请根据需要查看并删除这些资源，以免被收费。

出于以下原因，您可能需要删除 App Studio 实例：

- 你不想再使用 App Studio 了。
- 您想在其他 AWS 地区创建 App Studio 实例。由于 App Studio 一次仅支持在一个区域中拥有一个实例，因此您必须删除所有现有实例才能创建另一个实例。

> **Warning**
>
> 删除 App Studio 实例还会删除所有 App Studio 资源，例如应用程序和连接器。删除实例的操作无法撤消。

删除你的 App Studio 实例

1. 在以下位置打开 App Studio 控制台https://console.aws.amazon.com/appstudio/。
2. 选择您的 App Studio 实例所在的区域。
3. 在导航窗格中，选择实例。
4. 选择操作以打开包含其他实例操作的下拉列表。
5. 选择删除 App Studio 实例。
6. 输入 **`confirm`**，然后选择删除。
7. 您的实例删除可能需要一段时间才能处理。删除后，您将收到一封确认电子邮件。收到电子邮件后，您可以根据需要创建另一个实例。

生成器文档

以下主题包含的信息可帮助在 App Studio 中创建、编辑和发布应用程序的用户。

主题

- 教程
- 使用生成式 AI 构建你的 App Studio 应用程序
- 创建、编辑和删除应用程序
- 预览、发布和共享应用程序
- 页面和组件：构建应用程序的用户界面
- 自动化和操作：定义应用程序的业务逻辑
- 实体和数据操作：配置应用程序的数据模型
- 页面和自动化参数
- JavaScript 用于在 App Studio 中编写表达式
- 数据依赖关系和时序注意事项
- 构建包含多个用户的应用程序
- 查看或更新应用程序的内容安全设置

教程

主题

- 使用 Amazon Bedrock 构建人工智能文本摘要器应用程序
- 使用组件和自动化与 Amazon 简单存储服务交互
- 在 App Studio 应用程序中调用 Lambda 函数

使用 Amazon Bedrock 构建人工智能文本摘要器应用程序

在本教程中，您将在 App Studio 中构建一个应用程序，该应用程序使用 Amazon Bedrock 来提供最终用户输入的文本的简明摘要。该应用程序包含一个简单的用户界面，用户可以在其中输入他们想要摘要的任何文本。这可能是会议记录、文章内容、研究结果或任何其他文字信息。用户输入文本后，他们可以按下按钮将文本发送到 Amazon Bedrock，Amazon Bedrock 将使用 Claude 3 Sonnet 模型对其进行处理并返回摘要版本。

目录

- 先决条件
- 步骤 1：创建和配置 IAM 角色和 App Studio 连接器
- 步骤 2：创建应用程序
- 步骤 3：创建和配置自动化
- 步骤 4：创建页面和组件
  - 重命名默认页面
  - 向页面添加组件
  - 配置页面组件
- 步骤 5：将应用程序发布到测试环境
- (可选)清除

先决条件

在开始之前，请查看并完成以下先决条件：

- 访问 AWS App Studio。请注意，您必须具有管理员角色才能在本教程中创建连接器。
- 可选：查看AWS 应用工作室的概念并熟悉 App Studio 的重要概念。教程：从一个空的应用程序开始构建

步骤 1：创建和配置 IAM 角色和 App Studio 连接器

要让 App Studio 访问亚马逊 Bedrock 型号，您必须：

1. 启用要在应用程序中使用的 Amazon Bedrock 模型。在本教程中，您将使用 Claude 3 Sonnet，因此请确保启用该模型。
2. 创建具有相应的 Amazon Bedrock 权限的 IAM 角色。
3. 使用将在您的应用程序中使用的 IAM 角色创建 App Studio 连接器。

Connect 到 Amazon Bedrock有关详细说明，请在按照步骤操作并创建连接器后返回本教程。

步骤 2：创建应用程序

使用以下步骤在 App Studio 中创建一个空应用程序，然后将其内置到文本摘要器应用程序中。

1. 登录 App Studio。
2. 导航到生成器中心，然后选择 + 创建应用程序。
3. 选择从头开始。
4. 在应用程序名称字段中，为您的应用程序提供一个名称，例如**Text Summarizer**。
5. 如果系统要求您选择数据源或连接器，请在本教程中选择 “跳过”。
6. 选择下一步以继续。
7. （可选）：观看视频教程，快速了解如何在 App Studio 中构建应用程序。
8. 选择 “编辑应用程序”，这将带您进入应用程序工作室。

步骤 3：创建和配置自动化

您可以在自动化中定义 App Studio 应用程序的逻辑和行为。自动化由称为操作的各个步骤、用于将数据从其他资源传递给操作的参数以及可供其他自动化或组件使用的输出组成。在此步骤中，您将创建一个自动化系统，通过以下方式处理与 Amazon Bedrock 的互动：

- 输入：用于将用户输入的文本传递给自动化的参数。
- 操作：一个 GenAI 提示操作，用于将文本输入发送到 Amazon Bedrock 并返回输出文本摘要。
- 输出：由来自 Amazon Bedrock 的已处理摘要组成的自动化输出，可在您的应用程序中使用。

创建和配置自动化，以便向 Amazon Bedrock 发送提示并处理和返回摘要

1. 选择画布顶部的 “自动化” 选项卡。
2. 选择 + 添加自动化。
3. 在右侧面板中，选择 “属性”。
4. 通过选择铅笔图标来更新自动化名称。输入 **InvokeBedrock**，键入按 Enter。
5. 通过执行以下步骤，向自动化添加一个参数，该参数将用于将用户输入的文本提示传递到自动化中，以便在向 Amazon Bedrock 的请求中使用：
   a. 在画布的参数框中，选择 + 添加。
   b. 在名称中，输入 **input**。
   c. 在描述中，输入任何描述，例如**Text to be sent to Amazon Bedrock**。
   d. 在类型中，选择字符串。
   e. 选择 “添加” 以创建参数。

6. 通过执行以下步骤添加 GenAI 提示操作：
   a. 在右侧面板中，选择动作。
   b. 选择 GenAI 提示以添加操作。
7. 通过执行以下步骤来配置操作：
   a. 从画布中选择操作以打开右侧的 “属性” 菜单。
   b. **`PromptBedrock`**通过选择铅笔图标，输入名称并按 Enter 键将动作重命名为。
   c. 在连接器中，选择在中创建的连接器步骤 1：创建和配置 IAM 角色和 App Studio 连接器。
   d. 在模型中，选择要用于处理提示的 Amazon Bedrock 型号。在本教程中，你将选择 Claude 3.5 Sonn et。
   e. 在用户提示符中，输入`{{params.input}}`。这表示您之前创建的`input`参数，并将包含您的应用程序用户输入的文本。
   f. 在系统提示符中，输入您要发送给 Amazon Bedrock 的系统提示符说明。在本教程中，输入以下内容：

      ```
      You are a highly efficient text summarizer. Provide a concise summary of the
       prompted text, capturing the key points and main ideas.
      ```

   g. 选择 “请求设置” 将其展开，然后更新以下字段：
      - 在温度中，输入`0`。温度在 0 到 10 的范围内决定输出的随机性或创造性。数字越高，回应就越有创意。
      - 在最大代币数中，输入`4096`以限制响应的长度。
8. 此自动化的输出将是摘要文本，但是，默认情况下，自动化不会创建输出。通过执行以下步骤配置自动化以创建自动化输出：
   a. 在左侧导航栏中，选择InvokeBedrock自动化。
   b. 在右侧的 “属性” 菜单的 “输出” 中，选择 + 添加。
   c. 在输出中，输入**`{{results.PromptBedrock.text}}`**。此表达式返回`processResults`操作的内容。

步骤 4：创建页面和组件

在 App Studio 中，每个页面都代表用户将与之交互的应用程序用户界面 (UI) 屏幕。在这些页面中，您可以添加各种组件，例如表格、表单、按钮等，以创建所需的布局和功能。

重命名默认页面

本教程中的文本摘要器应用程序将仅包含一页。新创建的应用程序带有默认页面，因此您可以重命名该页面，而不是添加一个页面。

重命名默认页面

1. 在顶部栏导航菜单中，选择页面。
2. 在左侧面板中，选择 Page1，然后在右侧面板中选择 “属性” 面板。
3. 选择铅笔图标，输入**TextSummarizationTool**，然后按 Enter。
4. 在导航标签中输入**TextSummarizationTool**。

向页面添加组件

在本教程中，文本摘要器应用程序有一个包含以下组件的页面：

- 一种文本输入组件，最终用户使用它来输入要汇总的提示。
- 用于向 Amazon Bedrock 发送提示的 B ut ton 组件。
- 一个文本区域组件，用于显示来自 Amazon Bedrock 的摘要。

向页面添加文本输入组件，用户将使用该组件输入要汇总的文本提示。

添加文本输入组件

1. 在右侧的 “组件” 面板中，找到文本输入组件并将其拖到画布上。
2. 选择画布中的文本输入以将其选中。
3. 在右侧的 “属性” 面板中，更新以下设置：
   a. 选择铅笔图标将文本输入重命名为**inputPrompt**。
   b. 在标签中，输入**Prompt**。
   c. 在占位符中，输入**Enter text to be summarized**。

现在，添加一个 B ut ton 组件，用户可以选择将其发送到 Amazon Bedrock。

添加按钮组件

1. 在右侧的 “组件” 面板中，找到 B ut ton 组件并将其拖到画布上。

2. 在画布中选择按钮将其选中。
3. 在右侧的 “属性” 面板中，更新以下设置：
   a. 选择要将按钮重命名为的铅笔图标**sendButton**。
   b. 在 “按钮标签” 中，输入**Send**。

现在，添加一个文本区域组件，该组件将显示 Amazon Bedrock 返回的摘要。

添加文本区域组件

1. 在右侧的 “组件” 面板中，找到文本区域组件并将其拖到画布上。
2. 在画布中选择文本区域将其选中。
3. 在右侧的 “属性” 面板中，更新以下设置：
   a. 选择要将按钮重命名为的铅笔图标**textSummary**。
   b. 在标签中，输入**Summary**。

配置页面组件

现在，该应用程序包含一个包含组件的页面，下一步是配置组件以执行适当的行为。要将组件（例如按钮）配置为在与之交互时执行操作，必须向其添加触发器。对于本教程中的应用程序，您将在`sendButton`按钮中添加两个触发器来执行以下操作：

- 第一个触发器将`textPrompt`组件中的文本发送到 Amazon Bedrock 进行分析。
- 第二个触发器在`textSummary`组件中显示从 Amazon Bedrock 返回的摘要。

添加将提示发送到 Amazon Bedrock 的触发器

1. 在画布中选择按钮将其选中。
2. 在右侧的 “属性” 面板的 “触发器” 部分，选择 + 添加。
3. 选择 “调用自动化”。
4. 选择为对其进行配置而创建的 InvokeAutomation1 触发器。
5. 在操作名称中，输入**invokeBedrockAutomation**。
6. 在 “调用自动化” 中，选择之前创建的InvokeBedrock自动化。

7. 在参数框中，在之前创建的输入参数中输入**{{ui.inputPrompt.value}}**，它会传递`inputPrompt`文本输入组件中的内容。
8. 选择面板顶部的左箭头返回到组件属性菜单。

现在，您已经配置了一个触发器，该触发器会在点击按钮时调用自动向 Amazon Bedrock 发送请求，下一步是配置第二个触发器，在组件中显示结果。`textSummary`

添加在文本区域组件中显示 Amazon Bedrock 结果的触发器

1. 在按钮右侧的 “属性” 面板的 “触发器” 部分，选择 + 添加。
2. 选择 “运行组件操作”。
3. 选择为对其进行配置而创建的 Runcomponentaction1 触发器。
4. 在操作名称中，输入**setTextSummary**。
5. 在组件中，选择文本摘要组件。
6. 在 “操作” 中，选择 “设置值”。
7. 在 “将值设置为” 中，输入**{{results.invokeBedrockAutomation}}**。

步骤 5：将应用程序发布到测试环境

通常，在构建应用程序时，最好先对其进行预览以查看其外观并对其功能进行初步测试。但是，由于应用程序在预览环境中不与外部服务交互，因此您可以将应用程序发布到测试环境，以便能够测试发送请求和接收来自 Amazon Bedrock 的响应。

将您的应用程序发布到测试环境

1. 在应用程序构建器的右上角，选择发布。
2. 为测试环境添加版本描述。
3. 查看并选中与 SLA 相关的复选框。
4. 选择启动。发布最多可能需要 15 分钟。
5. （可选）准备就绪后，您可以通过选择 “共享” 并按照提示向其他人授予访问权限。有关共享 App Studio 应用程序的更多信息，请参阅共享已发布的应用程序。

测试应用程序后，再次选择 Publish 以将该应用程序提升到生产环境。请注意，生产环境中的应用程序只有在共享后才可供最终用户使用。有关不同应用程序环境的更多信息，请参阅应用程序环境。

(可选)清除

现在，您已成功完成本教程，并使用 Amazon Bedrock 在 App Studio 中构建了一个文本摘要应用程序。您可以继续使用您的应用程序，也可以清理在本教程中创建的资源。以下列表包含要清理的资源列表：

- 在 App Studio 中创建的亚马逊 Bedrock 连接器。有关更多信息，请参阅 查看、编辑和删除连接器。
- App Studio 中的文本摘要器应用程序。有关更多信息，请参阅 删除 应用程序。
- 在 IAM 控制台中创建的 IAM 角色。有关更多信息，请参阅AWS Identity and Access Management 用户指南中的删除角色或实例配置文件。
- 如果您请求模型访问权限以使用 Claude 3 Sonnet 并希望恢复访问权限，请参阅《亚马逊 Bedrock 用户指南》中的 “管理对亚马逊 Bedrock 基础模型的访问权限”。

使用组件和自动化与 Amazon 简单存储服务交互

您可以从 App Studio 应用程序中调用各种 Amazon S3 操作。例如，您可以创建一个简单的管理面板来管理您的用户和订单，并显示来自 Amazon S3 的媒体。虽然您可以使用调用操作调用任何 Amazon S3 AWS操作，但您可以将四个专用 Amazon S3 操作添加到应用程序的自动化中，以便对 Amazon S3 存储桶和对象执行常见操作。这四个动作及其操作如下：

- 放置对象：使用`Amazon S3 PutObject`操作将对象添加到 Amazon S3 存储桶。
- 获取对象：使用`Amazon S3 GetObject`操作从 Amazon S3 存储桶中检索对象。
- 列出对象：使用`Amazon S3 ListObjects`操作列出 Amazon S3 存储桶中的对象。
- 删除对象：使用`Amazon S3 DeleteObject`操作从 Amazon S3 存储桶中删除对象。

除了操作之外，还有一个 S3 上传组件，您可以将其添加到应用程序的页面中。用户可以使用此组件选择要上传的文件，该组件调用将文件上传`Amazon S3 PutObject`到配置的存储桶和文件夹。本教程将使用此组件代替独立的 Pu t Object 自动化操作。（独立操作应用于更复杂的场景，这些场景涉及在上传之前或之后要执行的额外逻辑或操作。）

先决条件

本指南假设您已完成以下先决条件：

1. 创建并配置了亚马逊 S3 存储桶、IAM 角色和策略以及亚马逊 S3 连接器，以便成功将 Amazon S3 与 App Studio 集成。要创建连接器，必须具有管理员角色。有关更多信息，请参阅 连接到亚马逊简单存储服务 (Amazon S3) Service。

创建一个空的应用程序

执行以下步骤，创建一个要在本指南中使用的空应用程序。

创建空应用程序

1. 在导航窗格中，选择我的应用程序。
2. 选择 + 创建应用程序。
3. 在 “创建应用程序” 对话框中，为应用程序命名，选择 “从头开始”，然后选择 “下一步”。
4. 在 “连接现有数据” 对话框中，选择 “跳过” 以创建应用程序。
5. 选择 “编辑应用程序”，进入新应用程序的画布，您可以在其中使用组件、自动化和数据来配置应用程序的外观和功能。

创建页面

在应用程序中创建三个页面以收集或显示信息。

创建页面

1. 如有必要，请选择画布顶部的 “页面” 选项卡。
2. 在左侧导航栏中，有一个使用您的应用程序创建的页面。选择 “+ 添加” 两次可再创建两个页面。导航窗格总共应显示三个页面。
3. 通过执行以下步骤更新 Page1 页面的名称：
   a. 选择省略号图标并选择页面属性。
   b. 在右侧的 “属性” 菜单中，选择铅笔图标以编辑名称。
   c. 输入 **`FileList`**，键入按 Enter。
4. 重复前面的步骤更新第二页和第三页，如下所示：
   - 将 Page2 重命名为。**`UploadFile`**
   - 将 Page3 重命名为。**`FailUpload`**

现在，您的应用程序应该有三个名为FileList、和UploadFile、的页面 FailUpload，它们显示在左侧的 “页面” 面板中。

接下来，您将创建和配置与 Amazon S3 交互的自动化。

创建和配置自动化

创建与 Amazon S3 交互的应用程序的自动化程序。使用以下过程创建以下自动化：

- GetFiles 自动化，用于列出您的 Amazon S3 存储桶中的对象，这些对象将用于填充表格组件。
- DeleteFile 自动化，用于从您的 Amazon S3 存储桶中删除对象，该存储桶将用于向表格组件添加删除按钮。
- ViewFile 自动化，可从您的 Amazon S3 存储桶中获取对象并显示该对象，用于显示有关从表格组件中选择的单个对象的更多详细信息。

创建**`getFiles`**自动化

创建自动化，列出指定 Amazon S3 存储桶中的文件。

1. 选择画布顶部的 “自动化” 选项卡。
2. 选择 + 添加自动化。
3. 在右侧面板中，选择 “属性”。
4. 通过选择铅笔图标来更新自动化名称。输入 **`getFiles`**，键入按 Enter。
5. 通过执行以下步骤添加 “列出对象” 操作：
   a. 在右侧面板中，选择动作。
   b. 选择 “列出对象” 以添加动作。该动作应命名`ListObjects1`。
6. 通过执行以下步骤来配置操作：
   a. 从画布中选择操作以打开右侧的 “属性” 菜单。
   b. 对于连接器，请选择您根据先决条件创建的 Amazon S3 连接器。
   c. 在配置中，输入以下文本，*`bucket_name`*替换为您在先决条件中创建的存储桶：

```
{
  "Bucket": "bucket_name",
  "Prefix": ""
}
```

> **Note**
>
> 您可以使用该`Prefix`字段将响应限制为以指定字符串开头的对象。

7. 此自动化的输出将用于使用您的 Amazon S3 存储桶中的对象填充表组件。但是，默认情况下，自动化不会创建输出。通过执行以下步骤配置自动化以创建自动化输出：
   a. 在左侧导航栏中，选择 GetFiles 自动化。
   b. 在右侧的 “属性” 菜单上，在 “自动输出” 中，选择 “+ 添加输出”。
   c. 在 “输出” 中，输入**`{{results.ListObjects1.Contents}}`**。此表达式返回操作的内容，现在可用于填充表格组件。

创建**`deleteFile`**自动化

创建自动化，从指定的 Amazon S3 存储桶中删除对象。

1. 在左侧的 “自动化” 面板中，选择 + 添加。
2. 选择 + 添加自动化。
3. 在右侧面板中，选择 “属性”。
4. 通过选择铅笔图标来更新自动化名称。输入 **`deleteFile`**，键入按 Enter。
5. 通过执行以下步骤，添加用于向自动化传递数据的自动化参数：
   a. 在右侧的 “属性” 菜单上的 “自动化参数” 中，选择 + 添加。
   b. 选择铅笔图标编辑自动化参数。将参数名称更新为，**`fileName`**然后按 Enter。
6. 通过执行以下步骤添加 “删除对象” 操作：
   a. 在右侧面板中，选择动作。
   b. 选择 “删除对象” 以添加动作。该动作应命名`DeleteObject1`。
7. 通过执行以下步骤来配置操作：
   a. 从画布中选择操作以打开右侧的 “属性” 菜单。
   b. 对于连接器，请选择您根据先决条件创建的 Amazon S3 连接器。
   c. 在配置中，输入以下文本，*`bucket_name`*替换为您在先决条件中创建的存储桶：

```
{
  "Bucket": "bucket_name",
  "Key": params.fileName
}
```

创建**viewFile**自动化

创建自动化，从指定的 Amazon S3 存储桶中检索单个对象。稍后，您将使用文件查看器组件来配置此自动化，以显示该对象。

1. 在左侧的 “自动化” 面板中，选择 + 添加。
2. 选择 + 添加自动化。
3. 在右侧面板中，选择 “属性”。
4. 通过选择铅笔图标来更新自动化名称。输入 **viewFile**，键入按 Enter。
5. 通过执行以下步骤，添加用于向自动化传递数据的自动化参数：
   a. 在右侧的 “属性” 菜单上的 “自动化参数” 中，选择 + 添加。
   b. 选择铅笔图标编辑自动化参数。将参数名称更新为，**fileName**然后按 Enter。
6. 通过执行以下步骤添加 “获取对象” 操作：
   a. 在右侧面板中，选择动作。
   b. 选择 “获取对象” 以添加动作。该动作应命名`GetObject1`。
7. 通过执行以下步骤来配置操作：
   a. 从画布中选择操作以打开右侧的 “属性” 菜单。
   b. 对于连接器，请选择您根据先决条件创建的 Amazon S3 连接器。
   c. 在配置中，输入以下文本，*bucket_name*替换为您在先决条件中创建的存储桶：

```
{
  "Bucket": "bucket_name",
  "Key": params.fileName
}
```

8. 默认情况下，自动化不会创建输出。通过执行以下步骤配置自动化以创建自动化输出：
   a. 在左侧导航栏中，选择 ViewFile 自动化。
   b. 在右侧的 “属性” 菜单上，在 “自动输出” 中，选择 “+ 添加输出”。
   c. 在 “输出” 中，输入**{{results.GetObject1.Body.transformToWebStream()}}**。此表达式返回操作的内容。

> **Note**
>
> 您可以通过以下方式阅读`S3 GetObject`的回复：

- `transformToWebStream`：返回一个流，必须使用该流才能检索数据。如果用作自动化输出，则自动化会处理此问题，并且输出可用作图像或 PDF 查看器组件的数据源。它也可以用作其他操作的输入，例如`S3 PutObject`。
- `transformToString`：返回自动化的原始数据，如果您的文件包含文本内容（例如 JSON 数据），则应在 JavaScript 操作中使用该数据。必须等待，例如：`await results.GetObject1.Body.transformToString();`
- `transformToByteArray`: 返回一个 8 位无符号整数的数组。此响应用于字节数组，该数组允许存储和操作二进制数据。必须等待，例如：`await results.GetObject1.Body.transformToByteArray();`

接下来，您将向之前创建的页面添加组件，并使用您的自动化功能对其进行配置，以便用户可以使用您的应用程序查看和删除文件。

添加和配置页面组件

现在，您已经创建了定义应用程序业务逻辑和功能的自动化，接下来您将创建组件并将它们连接起来。

向FileList页面添加组件

您之前创建的FileList页面将用于显示已配置的 Amazon S3 存储桶中的文件列表以及有关从列表中选择的任何文件的更多详细信息。为此，您将执行以下操作：

1. 创建表格组件以显示文件列表。您需要将表的行配置为使用之前创建的 getFiles 自动化的输出进行填充。
2. 创建 PDF 查看器组件以显示单个 PDF。您将使用之前创建的 ViewFile 自动化功能将组件配置为查看从表中选择的文件，从存储桶中提取文件。

向FileList页面添加组件

1. 选择画布顶部的 “页面” 选项卡。
2. 在左侧的 “页面” 面板中，选择FileList页面。
3. 在右侧的 “组件” 页面上，找到表格组件并将其拖动到画布的中央。
4. 选择您刚刚添加到页面的表格组件。
5. 在右侧的 “属性” 菜单上，选择 “来源” 下拉列表并选择 “自动化”。
6. 选择 “自动化” 下拉列表并选择 GetFiles 自动化。该表将使用 getFiles 自动化的输出作为其内容。

7. 添加一列，用文件名填充。

   a. 在右侧的 “属性” 菜单上，选择 “列” 旁边，选择 “+ 添加”。

   b. 选择刚刚添加的 Column1 列右侧的箭头图标。

   c. 对于列标签，将列重命名为**Filename**。

   d. 对于值，请输入 **{{currentRow.Key}}**。

   e. 选择面板顶部的箭头图标可返回主属性面板。

8. 添加表格操作以连续删除文件。

   a. 在右侧的 “属性” 菜单上，选择 “操作” 旁边的 “+ 添加”。

   b. 在 “操作” 中，将 “按钮” 重命名为**Delete**。

   c. 选择刚刚重命名的 “删除” 操作右侧的箭头图标。

   d. 在 “点击时” 中，选择 “+ 添加操作”，然后选择 “调用自动化”。

   e. 选择为对其进行配置而添加的操作。

   f. 对于操作名称，输入 **DeleteRecord**。

   g. 在 “调用自动化” 中，选择**deleteFile**。

   h. 在参数文本框中，输入**{{currentRow.Key}}**。

   i. 对于值，请输入 **{{currentRow.Key}}**。

9. 在右侧面板中，选择 “组件” 以查看组件菜单。显示文件有两种选择：

   - 图像查看器，用于查看`.jpg`扩展名为`.png`.`jpeg`、或的文件。
   - 用于查看 PDF 文件的 PDF 查看器组件。

   在本教程中，您将添加和配置 PDF 查看器组件。

10. 添加 PDF 查看器组件。

    a. 在右侧的 “组件” 页面上，找到 PDF 查看器组件，然后将其拖到表格组件下方的画布上。

    b. 选择刚刚添加的 PDF 查看器组件。

    c. 在右侧的 “属性” 菜单上，选择 “来源” 下拉列表并选择 “自动化”。

    d. 选择 “自动化” 下拉列表并选择 ViewFile 自动化。该表将使用 ViewFile 自动化的输出作为其内容。

    e. 在参数文本框中，输入**{{ui.table1.selectedRow["Filename"]}}**。

f. 在右侧面板中，还有一个 “文件名” 字段。此字段的值用作 PDF 查看器组件的标题。输入与上一步相同的文本:**{{ui.table1.selectedRow["Filename"]}}**.

向UploadFile页面添加组件

该UploadFile页面将包含一个文件选择器，可用于选择文件并将其上传到已配置的 Amazon S3 存储桶。您将在页面中添加 S3 上传组件，用户可以使用该组件来选择和上传文件。

1. 在左侧的 “页面” 面板中，选择UploadFile页面。
2. 在右侧的 “组件” 页面上，找到 S3 上传组件并将其拖到画布中央。
3. 选择您刚刚添加到页面的 S3 上传组件。
4. 在右侧的 “属性” 菜单上，配置组件：

   a. 在连接器下拉列表中，选择先决条件中创建的 Amazon S3 连接器。

   b. 在存储桶中，输入您的 Amazon S3 存储桶的名称。

   c. 对于文件名，输入 **{{ui.s3Upload1.files[0]?.nameWithExtension}}**。

   d. 对于 “最大文件大小”，**5**在文本框中输入，并确保**MB**在下拉列表中将其选中。

   e. 在 “触发器” 部分，通过执行以下步骤添加在上传成功或失败后运行的操作：

      要添加在成功上传后运行的操作，请执行以下操作：

      1. 在 “成功时” 中，选择 “+ 添加操作”，然后选择 “导航”。
      2. 选择为对其进行配置而添加的操作。
      3. 对于导航类型，请选择页面。
      4. 在 “导航到” 中，选择**FileList**。
      5. 选择面板顶部的箭头图标可返回主属性面板。

      要添加在上传失败后运行的操作，请执行以下操作：

      1. 在 “失败时” 中，选择 “+ 添加操作”，然后选择 “导航”。
      2. 选择为对其进行配置而添加的操作。
      3. 对于导航类型，请选择页面。
      4. 在 “导航到” 中，选择**FailUpload**。
      5. 选择面板顶部的箭头图标可返回主属性面板。

向FailUpload页面添加组件

该FailUpload页面是一个简单的页面，其中包含一个文本框，用于通知用户其上传失败。

1. 在左侧的 “页面” 面板中，选择FailUpload页面。
2. 在右侧的 “组件” 页面上，找到文本组件并将其拖动到画布的中央。
3. 选择您刚刚添加到页面的文本组件。
4. 在右侧的 “属性” 菜单上，在 “值” 中输入**`Failed to upload, try again`**。

更新您的应用程序安全设置

App Studio 中的每个应用程序都有内容安全设置，您可以使用这些设置来限制外部媒体或资源，或者限制您可以向 Amazon S3 中的哪些域上传对象。默认设置是屏蔽所有域名。要从您的应用程序将对象上传到 Amazon S3，您必须更新设置以允许您要向其上传对象的域。

允许域名将对象上传到 Amazon S3

1. 选择应用程序设置选项卡。
2. 选择 “内容安全设置” 选项卡。
3. 对于 Connect 来源，选择 “允许所有连接”。
4. 选择保存。

后续步骤：预览并发布应用程序以供测试

该应用程序现已准备就绪，可以进行测试了。有关预览和发布应用程序的更多信息，请参阅预览、发布和共享应用程序。

在 App Studio 应用程序中调用 Lambda 函数

本教程向您展示如何将 App Studio 连接到 Lambda 以及如何从您的应用程序调用 Lambda 函数。

先决条件

本指南假设您已完成以下先决条件：

1. 已创建 App Studio 应用程序。如果没有，则可以创建一个空的应用程序以在本教程中使用。有关更多信息，请参阅 创建 应用程序。

> **Note**
>
> 虽然您不需要 Lambda 函数即可按照本教程学习如何对其进行配置，但拥有一个 Lambda 函数可能会有所帮助，以确保您已正确配置应用程序。本教程不包含有关创建 Lambda 函数的信息。有关更多信息，请参阅开发人员指南。AWS Lambda

创建 Lambda 连接器

要在您的 App Studio 应用程序中使用 Lambda 函数，您必须使用连接器将 App Studio 连接到 Lambda，以提供对您的函数的访问权限。只有管理员才能在 App Studio 中创建连接器。有关创建 Lambda 连接器的更多信息，包括创建连接器的步骤，请参阅。连接到 AWS Lambda

创建和配置自动化

自动化用于定义应用程序的逻辑，由操作组成。要在应用程序中调用 Lambda 函数，您需要先向自动化添加并配置 “调用 Lambda” 操作。使用以下步骤创建自动化并向其添加 “调用 Lambda” 操作。

1. 编辑应用程序时，选择 “自动化” 选项卡。
2. 选择 + 添加自动化。
3. 在右侧的 “操作” 菜单中，选择 “调用 Lambd a”，将该步骤添加到您的自动化中。
4. 在画布中选择新的 Lambda 步骤以查看和配置其属性。
5. 在右侧的 “属性” 菜单中，通过执行以下步骤来配置该步骤：
   a. 在 C onnector 中，选择为将 App Studio 连接到您的 Lambda 函数而创建的连接器。
   b. 在函数名称中，输入您的 Lambda 函数的名称。
   c. 在函数事件中，输入要传递给 Lambda 函数的事件。以下列表提供了一些常见的用例示例：
      - 传递自动化参数的值，例如文件名或其他字符串：`varName: params.paramName`
      - 传递先前操作的结果：`varName: results.actionName1.data[0].fieldName`
      - 如果您在 L oop 操作中添加 Invoke Lambda 操作，则可以从每个迭代项目中发送与参数类似的字段：`varName: currentItem.fieldName`
   d. Mocked ou tput 字段可用于提供模拟输出，以便在预览时测试应用程序，其中连接器未处于活动状态。

配置用户界面元素以运行自动化

现在，您的自动化配置了调用您的 Lambda 函数的操作，您可以配置一个用户界面元素来运行自动化。在本教程中，您将创建一个按钮，单击该按钮即可运行自动化。

> **Tip**
> 您还可以使用 “调用自动化” 操作从其他自动化中运行自动化。

通过按钮运行自动化

1. 编辑应用程序时，选择 “页面” 选项卡。
2. 在右侧菜单中，选择 B ut ton 组件以向页面添加按钮。
3. 选择新按钮进行配置。
4. 在右侧的 “属性” 菜单的 “触发器” 中，选择 “+ 添加”，然后选择 “调用自动化”。
5. 选择新的自动化调用触发器进行配置。
6. 在调用自动化中，选择调用您的 Lambda 函数的自动化，并配置要发送给自动化的任何参数。

现在，任何在您的应用程序中选择此按钮的用户都将使配置的自动化运行。

后续步骤：预览并发布应用程序以供测试

您的应用程序现已准备好进行测试。在开发环境中预览应用程序时，连接器处于非活动状态，因此您无法在预览时测试自动化，因为它使用连接器进行连接。 AWS Lambda要测试您的应用程序依赖于连接器的功能，您必须将应用程序发布到测试环境。有关预览和发布应用程序的更多信息，请参阅预览、发布和共享应用程序。

使用生成式 AI 构建你的 App Studio 应用程序

AWS App Studio 提供集成的生成式 AI 功能，可加快开发速度并简化常见任务。你可以利用生成式 AI 来生成和编辑应用程序、数据模型、样本数据，甚至可以在构建应用程序时获得情境帮助。

正在生成您的应用程序

为了加快启动速度，您可以使用由 AI 支持的自然语言提示生成整个应用程序。此功能允许您描述所需的应用程序功能，AI 将自动构建数据模型、用户界面、工作流程和连接器。有关使用 AI 生成应用程序的更多信息，请参阅创建 应用程序。

构建或编辑您的应用程序

在编辑应用程序时，您可以使用聊天来描述您要进行的更改，并且您的应用程序会自动更新。您可以从现有的示例提示中进行选择，也可以输入自己的提示。聊天可用于添加、编辑和删除支持的组件，还可以创建和配置自动化和操作。使用以下步骤使用 AI 编辑或构建您的应用程序。

使用 AI 编辑您的应用程序

1. 如有必要，请编辑您的应用程序以导航到应用程序工作室。
2. （可选）选择要使用 AI 编辑的页面或组件。
3. 选择左下角的 “使用 AI 构建” 以打开聊天。
4. 输入要进行的更改，或从示例提示中进行选择。
5. 查看要进行的更改。如果要进行更改，请选择 “确认”。否则，请输入另一个提示。
6. 查看变更摘要。

生成您的数据模型

您可以根据提供的实体名称自动生成包含字段、数据类型和数据操作的实体。有关创建实体（包括使用创建实体）的更多信息 GenAi，请参阅在 App Studio 应用程序中创建实体。

您也可以通过以下方式更新现有实体：

- 向实体添加更多字段。有关更多信息，请参阅 添加、编辑或删除实体字段。
- 向实体添加数据操作。有关更多信息，请参阅 创建数据操作。

生成样本数据

您可以根据实体的字段为实体生成示例数据。这对于在连接外部数据源之前测试您的应用程序，或者在不与外部数据源通信的开发环境中测试您的应用程序非常有用。有关更多信息，请参阅 添加或删除示例数据。

将应用程序发布到测试或生产环境后，您的实时数据源和连接器将在这些环境中使用。

为 AWS 服务配置操作

在与 Amazon Simple Email Service 等 AWS 服务集成时，您可以使用 AI 生成示例配置，其中包含基于所选服务的预填字段。要尝试一下，请在 “调用 AWS 自动化” 操作的 “属性” 菜单中，选择双面箭头展开 “配置” 字段。然后，选择 “生成示例配置”。

嘲笑的回应

您可以为 AWS 服务操作生成模拟响应。这有助于在开发环境中测试您的应用程序，因为开发环境不与外部数据源通信。

在建造时向 AI 寻求帮助

在应用程序工作室中，你可以在支持的资源或属性上找到 AI 寻求帮助按钮。使用它来获取与当前视图或所选组件相关的上下文建议、文档和指导。询问有关 App Studio、应用程序构建最佳实践或您的特定应用程序用例的一般问题，以获得量身定制的信息和建议。

创建、编辑和删除应用程序

目录

- 创建 应用程序
- 导入应用程序
  - App Studio 提供的可导入应用程序
- 复制应用程序
- 编辑或构建应用程序
- 编辑之前发布的应用程序版本
- 重命名应用程序
- 删除 应用程序

创建 应用程序

按照以下步骤在 App Studio 中创建应用程序。

创建应用程序

1. 在导航窗格中，在 “构建” 部分中选择 “我的应用程序”，以导航到您的应用程序列表。
2. 选择 + 创建应用程序。
3. 在创建应用程序对话框中，为您的应用程序命名，然后选择以下应用程序创建方法之一：
   - 使用 AI 生成应用程序：选择此选项可使用自然语言描述您的应用程序，然后让 AI 为您生成应用程序及其资源。

- 从头开始：选择此选项可从空应用程序开始构建。

4. 选择下一步。
5. 如果您选择了 “使用 AI 生成应用程序”：

   a. 在 “连接到现有数据” 对话框中，通过选择提供 App Studio 数据源访问权限的连接器，将任何现有数据源添加到您的应用程序，然后选择表，然后选择下一步。在此处添加数据源有助于 AI 为您生成经过优化的应用程序。您可以跳过此步骤，稍后通过选择 “跳过” 来添加数据源。

   b. 短暂延迟（几分钟）后，您将被带到使用 AI 生成您的应用程序页面，您可以在其中描述要创建的应用程序。

   c. 您可以开始在聊天中描述您的应用程序，也可以选择和自定义提供的示例提示。

   d. 对您的提示进行分析后，请查看应用程序要求和概述。使用聊天请求任何更改，或者选择重新开始，从空白的提示开始。

   e. 准备就绪后，选择 “生成应用程序”。

   f. 生成后，选择预览应用程序，在另一个选项卡中预览您的应用程序。准备好开始编辑时，可以选择 “编辑应用程序”。浏览应用程序的页面、自动化和数据，以熟悉它。在底部的调试面板中查看所有错误或警告。要了解如何使用 AI 生成应用程序，请参阅教程：使用 AI 生成应用程序。有关如何在 App Studio 中进行构建的一般信息，请参阅AWS 应用工作室的工作原理。

6. 如果您选择了从头开始：

   a. 在 “连接到现有数据” 对话框中，通过选择提供 App Studio 数据源访问权限的连接器，将任何现有数据源添加到您的应用程序，然后选择表，然后选择下一步。您可以跳过此步骤，稍后通过选择 “跳过” 来添加数据源。

   b. 创建应用程序后，选择 “编辑应用程序” 以开始编辑您的应用程序。要了解如何使用空应用程序进行构建，请参阅教程：从一个空的应用程序开始构建。有关如何在 App Studio 中进行构建的一般信息，请参阅AWS 应用工作室的工作原理。

导入应用程序

您可以将导出的应用程序的副本导入到您的 App Studio 实例。您可以导入从其他 App Studio 实例导出的应用程序，也可以导入从 App Studio 提供的目录中导出的应用程序。从 App Studio 应用程序目录中导入应用程序可以帮助您开始使用具有类似功能的应用程序，或者通过浏览导入的应用程序来帮助您了解 App Studio 中的应用程序构建。

当您将应用程序导入实例时，将在您的实例中创建原始应用程序的副本。创建新应用程序后，您将导航到该应用程序的开发环境，您可以在其中预览该应用程序并浏览应用程序的功能。

> **Warning**
>
> 导入应用程序时，您正在从应用程序中导入所有逻辑，这可能会导致意外或意外的行为。例如，可能存在破坏性查询，这些查询会从您连接到应用程序的数据库中删除数据。我们建议在将生产数据连接到应用程序之前，彻底审查应用程序及其配置，并在非生产资产上对其进行测试。

导入应用程序

1. 在导航窗格中，在 “构建” 部分中选择 “我的应用程序”，以导航到您的应用程序列表。
2. 选择 + 创建应用程序旁边的下拉箭头。
3. 选择 “导入应用程序”。
4. 在导入应用程序对话框的导入代码中，输入要导入的应用程序的导入代码。有关 App Studio 提供的可以导入的应用程序列表，包括应用程序描述和导入代码，请参阅App Studio 提供的可导入应用程序。
5. 选择导入以导入应用程序，然后转到导入的应用程序的开发环境进行查看或编辑。有关在 App Studio 中构建应用程序的信息，请参阅 AWS 应用工作室的工作原理

App Studio 提供的可导入应用程序

App Studio 提供了可以导入到您的实例中的应用程序，以帮助您了解应用程序构建。要查看 App Studio 中如何实现特定的应用程序功能，您可以预览应用程序，然后在开发环境中浏览其配置。

下表包含应用程序列表、其导入代码以及应用程序的简要描述。每个应用程序都包含一个`README`页面，其中包含有关该应用程序的信息，您可以在导入应用程序后查看这些信息。

应用程序名称	描述	导入代码
Swag 申请调查	一款内部礼品申请应用程序，专为员工订购公司品牌商品而设计。员工可以选择物品并指定尺寸，然后通过简单的表格提交申请。该应用程序通过内置存储处理所有数据，无需外部连接。	Swag Request survey/ec4f5faf-e2f8-42ee-ab8d-6723d8ca21B2

应用程序名称	描述	导入代码
冲刺跟踪	一款冲刺管理应用程序，团队可以使用它来组织和跟踪软件开发工作。用户可以通过专用的冲刺、跟踪和任务视图创建冲刺、添加任务、分配工作以及监控进度。该应用程序通过内置存储处理所有数据，无需外部连接。	Sprint tracking/8f31e160-771f-48d7-87b0-374e285e2fbc
亚马逊评论情绪追踪器	此应用程序是一种客户反馈分析工具，可从产品评论中生成情绪分数，以帮助企业了解客户满意度。该应用程序包括用于快速测试的示例数据生成工具，并且需要使用 Amazon Bedrock 连接器来获得人工智能驱动的见解，同时将所有其他数据保存在内置存储系统中。	亚马逊评论情绪追踪器/60f0dae4-f8e2-4c20-9583-fa456f5ebFab
发票和收据处理	这款收据处理应用程序通过自动进行手动数据输入和简化文件审批工作流程，节省时间并减少错误。该解决方案需要亚马逊 Textract、Amazon S3 和 Amazon SES 连接器。它使用 Amazon Textract 来分析存储在 Amazon S3 中的收据并从中提取数据，然后使用 Amazon SES 处理提取的信息并将其通过电子邮件发送给审批人。	发票和收据处理/98bde3ae-e454-4b18-a1e6-6f23e8b2a4F1

应用程序名称	描述	导入代码
检验和库存审计	用于管理仓库检查和设备跟踪的应用程序。用户可以按房间位置进行设备通过/不合格评估，监控库存水平并查看检查历史记录。该应用程序提供了一个用于跟踪设施检查和设备状态的集中系统。该应用程序通过内置存储处理所有数据，无需外部连接。	检验和库存审计/cf570a06-1c5e-4dd7-9ea8-5c04723d687F
产品采用率追踪器	一款用于管理产品开发的综合应用程序，可集中客户反馈、功能请求和客户会议记录。团队可以跟踪客户互动，组织需求，并生成基于人工智能的报告，用于季度路线图规划。该应用程序包括示例数据实用程序，利用亚马逊贝德罗克获取人工智能见解，利用 Amazon Aurora PostgreSQL 进行数据管理。	产品采用追踪器/9b3a4437-bb50-467f-ae9e-d108776b7ca1
QuickSight 嵌入	一款演示应用程序，使用户能够在处理底层数据的同时查看分析。该应用程序包含两种在 App Studio 中嵌入亚马逊 QuickSight 仪表板的方法：一种针对注册用户和匿名用户的基于 API 的方法（需要 QuickSight 连接器），以及用于公共仪表板的 iFrame 集成。	Quicksight embedding/0cdc15fc-ca8b-41b7-869e-ed13c9072bc8

应用程序名称	描述	导入代码
厨房水槽	一款展示高级 App Studio 开发技巧和最佳实践的参考应用程序。包括状态管理、CSV 数据处理、浏览器 API 集成和用户界面模式的工作示例，构建者可以在自己的应用程序中研究和实现这些模式。所有示例都不需要外部连接。	App Studio Kitchen sink/1cfe6b2f-544c-4611-b82c-80eadc76a0c8

复制应用程序

应用程序所有者和共同所有者可以复制其应用程序，以创建应用程序的精确副本。如果您想保留当前状态以用于测试目的，或者将复制的应用程序用作创建新应用程序的入门工具，则复制应用程序会很有用。

复制应用程序

1. 在导航窗格的 “构建” 部分中，选择 “我的应用程序”。您将被带到一个页面，其中显示了您可以访问的应用程序列表。
2. 选择要复制的应用程序的 “操作” 列中的下拉列表。
3. 选择 Duplicate (复制)。如果复制（Duplicate）选项不可用，则您可能不是该应用程序的所有者或共同所有者。
4. （可选）提供重复应用程序的名称。默认名称为 *Current_App_Name COPY*。
5. 选择 Duplicate (复制)。

编辑或构建应用程序

使用以下步骤在 App Studio 中编辑应用程序。

编辑（构建）应用程序

1. 在导航窗格的 “构建” 部分中，选择 “我的应用程序”。您将被带到一个页面，其中显示了您可以访问的应用程序列表。

2. 在要编辑的应用程序的 “操作” 列中，选择 “编辑”。这将带您进入开发环境，在那里您可以使用组件、自动化和数据来配置应用程序的外观和功能。有关构建应用程序的信息，请参阅AWS App Studio 入门。

编辑之前发布的应用程序版本

使用以下步骤编辑先前发布的 App Studio 应用程序版本。选择编辑先前发布的版本后，可以在开发环境中编辑该应用程序，或者将其发布到测试版然后发布到正式版。

> **Warning**
>
> 在开发环境中，先前发布的版本取代了正在开发的应用程序版本。对您的应用程序所做的任何未发布更改都将丢失。

如果您不小心发布了不想要的更改或对用户造成应用程序损坏的更改，并且您想在以前的应用程序版本的基础上进一步构建或编辑，则编辑先前发布的版本非常有用。

> **Note**
>
> 如果您发现已发布的应用程序存在问题并需要立即发布以前运行的版本，或者您想在开发环境中发布先前版本但保留该应用程序的最新更新，则应改为回滚该应用程序。有关更多信息，请参阅 回滚到之前发布的版本。

编辑之前发布的应用程序版本

1. 如有必要，请导航到应用程序的应用程序工作室。
2. 选择 “发布” 按钮旁边的下拉箭头，然后选择 “发布中心”。
3. 选择 “版本历史记录” 以查看应用程序先前发布的版本列表。
4. 找到要编辑的版本，然后选择编辑。
5. 查看信息，然后选择 “恢复”。
6. 您选择编辑的版本现在是开发环境中的当前版本。您可以对其进行更改，也可以通过选择 “发布” 将其按原样发布到测试环境。发布到 Testing 后，如果需要，您可以再次发布到生产环境。

重命名应用程序

使用以下步骤在 App Studio 中重命名应用程序。在构建应用程序时，您可以从应用程序列表或开发环境中重命名应用程序。

重命名应用程序

1. 在导航窗格的 “构建” 部分中，选择 “我的应用程序”。您将被带到一个页面，其中显示了您可以访问的应用程序列表。
2. 可以在编辑时重命名此列表中的应用程序，也可以在开发环境中重命名应用程序。
   - 要从此列表中重命名：
     a. 选择要重命名的应用程序的 “操作” 列中的下拉列表，然后选择 “重命名”。
     b. 为您的应用程序指定一个新名称，然后选择 “重命名”。
   - 要在开发环境中重命名，请执行以下操作：
     a. 在要编辑的应用程序的 “操作” 列中，选择 “编辑”。
     b. 在开发环境中，选择应用程序名称并对其进行更新，然后按 Enter 键或离开文本字段以保存更改。

删除 应用程序

使用以下步骤在 App Studio 中删除应用程序。

删除 应用程序

1. 在导航窗格的 “构建” 部分中，选择 “我的应用程序”。您将被带到一个页面，其中显示了您可以访问的应用程序列表。
2. 选择要删除的应用程序的 “操作” 列中的下拉列表。
3. 选择删除。
4. 在删除应用程序对话框中，仔细查看有关删除应用程序的信息。如果要删除应用程序，请选择删除。

预览、发布和共享应用程序

主题

- 预览应用程序
- 发布应用程序
- 共享已发布的应用程序
- 回滚到之前发布的版本
- 导出应用程序

预览应用程序

您可以在 App Studio 中预览应用程序，以了解它们在用户面前的显示效果，还可以通过使用它和在调试面板中查看日志来测试其功能。

应用程序预览环境不支持显示实时数据或使用连接器与外部资源（例如数据源）的连接。要在预览环境中测试功能，可以在自动化中使用模拟输出，在实体中使用示例数据。要使用实时数据查看应用程序，必须发布应用程序。有关更多信息，请参阅 发布应用程序。

预览或开发环境不会更新在其他环境中发布的应用程序。如果应用程序尚未发布，则在发布和共享应用程序之前，用户将无法访问该应用程序。如果应用程序已经发布和共享，则用户仍将访问已发布的版本，而不是预览环境中使用的版本。

预览您的应用程序

1. 如有必要，请导航到要预览的应用程序的应用程序工作室：
   a. 在导航窗格的 “构建” 部分中，选择 “我的应用程序”。
   b. 为应用程序选择编辑。
2. 选择 “预览” 以打开应用程序的预览环境。
3. （可选）通过选择屏幕底部附近的标题来展开调试面板。通过在 “筛选日志” 部分选择消息类型，可以按消息类型筛选面板。您可以通过选择 “清除控制台” 来清除面板的日志。
4. 在预览环境中，您可以通过浏览应用程序的页面、使用其组件并选择其按钮来启动传输数据的自动化来测试应用程序。由于预览环境不支持实时数据或与外部源的连接，因此您可以在调试面板中查看正在传输的数据的示例。

发布应用程序

创建和配置完应用程序后，下一步是将其发布以测试数据传输或与最终用户共享。要了解在 App Studio 中发布应用程序，了解可用环境非常重要。App Studio 提供了三个独立的环境，如下表所述：

1. 开发：在哪里构建和预览应用程序。您无需发布到开发环境，因为最新版本的应用程序会自动托管在开发环境中。此环境中没有实时数据或第三方服务或资源。
2. 测试：您可以在其中对应用程序进行全面测试。在测试环境中，您可以连接其他服务，向其发送数据，也可以从中接收数据。
3. 生产：供最终用户使用的实时操作环境。

您的所有应用程序构建都是在开发环境中进行的。然后，发布到测试环境以测试其他服务之间的数据传输，并通过向最终用户提供访问网址来测试用户验收测试 (UAT)。之后，将您的应用程序发布到生产环境以进行最终测试，然后再与用户共享。有关应用程序环境的更多信息，请参阅应用程序环境。

发布应用程序时，只有在共享应用程序后才可供用户使用。这使您有机会在用户访问应用程序之前在测试和生产环境中使用和测试该应用程序。当您将之前发布和共享的应用程序发布到生产环境时，可供用户使用的版本会更新。

发布应用程序

使用以下步骤将 App Studio 应用程序发布到测试或生产环境。

将应用程序发布到测试或生产环境

1. 在导航窗格的 “构建” 部分中，选择 “我的应用程序”。您将被带到一个页面，其中显示了您可以访问的应用程序列表。
2. 为要发布的应用程序选择 “编辑”。
3. 选择右上角的 “发布”。
4. 在 “发布您的更新” 对话框中：
   a. 查看有关发布应用程序的信息。
   b. （可选）在版本描述中，包括此版本应用程序的描述。
   c. 选中复选框以确认有关环境的信息。
   d. 选择启动。在实时环境中更新应用程序最多可能需要 15 分钟。
5. 有关在测试或生产环境中查看应用程序的信息，请参阅查看已发布的应用程序。

> **Note**
> 在测试或生产环境中使用该应用程序将导致实时数据传输，例如在已通过连接器连接的数据源表中创建记录。

用户或其他构建者无法使用从未共享过的已发布应用程序。要向用户提供应用程序，必须在发布后共享该应用程序。有关更多信息，请参阅 共享已发布的应用程序。

查看已发布的应用程序

您可以查看发布到测试和生产环境的应用程序，以便在与最终用户或其他构建者共享应用程序之前对其进行测试。

在测试或生产环境中查看已发布的应用程序

1. 如有必要，请导航到要预览的应用程序的应用程序工作室：
   a. 在导航窗格的 “构建” 部分中，选择 “我的应用程序”。
   b. 为应用程序选择编辑。
2. 选择右上角发布旁边的下拉箭头，然后选择发布中心。
3. 在发布中心，您可以查看应用程序发布到的环境。如果您的应用程序已发布到测试或生产环境，则可以使用每个环境的 URL 链接查看应用程序。

> **Note**
>
> 在测试或生产环境中使用该应用程序将导致实时数据传输，例如在已通过连接器连接的数据源表中创建记录。

应用程序环境

AWS App Studio 通过三个独立的环境（开发、测试和生产）提供应用程序生命周期管理 (ALM) 功能。这可以帮助您更轻松地进行最佳实践，例如在整个应用程序生命周期中维护单独的环境、版本控制、共享和监控。

开发环境

开发环境是一个隔离的沙箱，您无需使用应用程序工作室和示例数据连接到任何实时数据源或服务即可构建应用程序。在开发环境中，您可以预览应用程序以查看和测试应用程序，而不会影响生产数据。

尽管您的应用程序无法连接到开发环境中的其他服务，但您可以在应用程序中配置不同的资源以模仿实时数据连接器和自动化。

开发环境中应用程序工作室底部有一个可折叠的调试面板，其中包含错误和警告，可帮助您在构建时检查和调试应用程序。有关应用程序故障排除和调试的更多信息，请参阅App Studio 故障排除和。

测试环境

初始应用程序开发完成后，下一步是发布到测试环境。在测试环境中，您的应用程序可以连接到其他服务，也可以向其他服务发送数据，也可以从中接收数据。因此，您可以使用此环境通过向最终用户提供访问网址来执行全面测试，包括用户验收测试 (UAT)。

> **Note**
>
> 首次发布到测试环境最多可能需要 15 分钟。

发布到测试环境的应用程序版本将在最终用户处于非活动状态 3 小时后移除。但是，所有版本都将保留，可以从 “版本历史记录” 选项卡中恢复。

测试环境的主要功能如下：

- 与实时数据源的集成测试以及 APIs
- 通过受控访问简化用户验收测试 (UAT)
- 收集反馈和解决问题的环境
- 能够使用浏览器控制台和开发者工具检查和调试客户端和服务器端活动。

有关应用程序故障排除和调试的更多信息，请参阅App Studio 故障排除和。

生产环境

在测试并修复了所有问题之后，您可以将应用程序版本从测试环境升级到生产环境以供实际操作。尽管生产环境是供最终用户使用的实时操作环境，但您可以在与用户共享发布的版本之前对其进行测试。

在最终用户处于非活动状态 14 天后，您在生产环境中发布的版本将被删除。但是，所有版本都将保留，可以从 “版本历史记录” 选项卡中恢复。

生产环境的主要功能如下：

- 供最终用户使用的实时操作环境
- 基于角色的精细访问控制
- 版本控制和回滚功能
- 只能检查和调试客户端活动
- 使用实时连接器、数据、自动化和 APIs

版本控制和发布管理

App Studio 通过其发布中心的版本控制系统提供版本控制和发布管理功能。

关键版本控制功能：

- 发布到测试环境会生成新的版本号（1.0、2.0、3.0...）。
- 从测试环境升级到生产环境时，版本号不会更改。
- 您可以从 “版本历史记录” 中回滚到任何以前的版本。
- 发布到测试环境的应用程序将在处于非活动状态 3 小时后暂停。版本会保留，可以从 “版本历史记录” 中恢复。
- 发布到生产环境的应用程序在处于非活动状态 14 天后将被删除。版本会保留，可以从 “版本历史记录” 中恢复。

这种版本控制模型允许快速迭代，同时在整个应用程序开发和测试周期中保持可追溯性、回滚功能和最佳性能。

维护和运营

App Studio 可能需要自动重新发布您的应用程序，以完成某些维护任务、操作活动并整合新的软件库。作为构建者，您无需执行任何操作，但最终用户可能需要重新登录应用程序。在某些情况下，我们可能需要您重新发布您的应用程序，以加入我们无法自动添加的新功能和库。在重新发布之前，您需要解决所有错误并查看警告。

共享已发布的应用程序

当您发布尚未发布的应用程序时，只有共享该应用程序才可供用户使用。共享已发布的应用程序后，用户即可使用该应用程序，如果发布了其他版本，则无需再次共享。

> **Note**
>
> 本节介绍与最终用户或测试人员共享已发布的应用程序。有关邀请其他用户开发应用程序的信息，请参阅构建包含多个用户的应用程序。

共享已发布的应用程序

1. 按照以下说明从应用程序列表或应用程序的应用程序工作室访问共享对话框：

- 要从应用程序列表中访问 “共享” 对话框，请执行以下操作：在导航窗格中，在 “构建” 部分中选择 “我的应用程序”。选择要共享的应用程序的 “操作” 列中的下拉列表，然后选择 “共享”。
- 要从应用程序工作室访问共享对话框，请执行以下操作：在应用程序的应用程序工作室中，选择顶部标题中的共享。

2. 在 “共享” 对话框中，选择要共享的环境的选项卡。如果您看不到 “测试” 或 “生产” 选项卡，则您的应用程序可能无法发布到相应的环境。有关发布的更多信息，请参阅发布应用程序。
3. 在相应的选项卡中，从下拉菜单中选择要与其共享环境的群组。
4. （可选）为群组分配应用程序级角色以测试或配置有条件的页面可见性。有关更多信息，请参阅配置基于角色的页面可见性。
5. 选择共享。
6. （可选）复制链接并与用户共享。只有与之共享应用程序和环境的用户才能访问相应环境中的应用程序。

回滚到之前发布的版本

使用以下步骤将 App Studio 应用程序的生产环境还原到之前发布的版本。您的应用程序最终用户将受到影响，并在应用程序部署后看到其回滚版本。回滚应用程序时，它还会将组件代码回滚到上次发布时的版本，并影响整个应用程序部署堆栈（用户代码、组件配置状态）。这意味着，App Studio 对组件代码所做的任何更新（例如字段或其他配置更改）都将被回滚，以确保回滚的应用程序版本能够像最初发布时一样运行。

回滚已发布版本时，开发环境中正在进行的应用程序版本不会受到影响。

如果您发现已发布的应用程序存在问题并需要立即发布以前运行的版本，或者您想在开发环境中发布先前版本并保留该应用程序的最新更新，则回滚已发布的应用程序版本会很有帮助。

> **Note**
>
> 如果要将应用程序的开发环境恢复到之前发布的版本，则应恢复该应用程序。有关更多信息，请参阅 编辑之前发布的应用程序版本。

将生产环境版本回滚到之前发布的应用程序版本

1. 如有必要，可通过编辑应用程序将其导航到其开发环境。有关更多信息，请参阅 编辑或构建应用程序。

2. 选择生产环境图块顶部的版本下拉箭头，查看可供回滚的可用版本。下拉列表包含过去 30 天内发布的版本。如果禁用此下拉列表，则可能是因为应用程序发布已在进行中，并且只能同时发布一个应用程序。
3. 选择要回滚到的版本。
4. 输入回滚的原因，然后选择回滚。回滚发布将开始，完成后，您的应用程序的生产环境将更新为所选版本。

> **Note**
>
> 回滚后，您也可以向前滚动到之前发布的应用程序版本。

导出应用程序

您可以导出应用程序的快照，以便与其他 App Studio 实例共享。导出应用程序时，会从该应用程序的开发环境中创建快照，并生成导入代码。然后，可以使用导入代码将应用程序导入其他 App Studio 实例，在那里可以查看和构建该应用程序。

导出的应用程序可以导入到 App Studio AWS 区域 支持的任何实例中。

导出应用程序

1. 在导航窗格中，在 “构建” 部分中选择 “我的应用程序”，以导航到您的应用程序列表。
2. 选择要导出的应用程序的 “操作” 列中的下拉列表。
3. 选择导出。
4. 生成和共享导入代码的过程会有所不同，具体取决于是否已经为应用程序创建了导入代码。
   - 如果尚未创建导入代码：
     a. 在应用程序导入权限中，指定哪些实例可以导入导出的应用程序。您可以向所有实例授予导入权限，也可以通过输入其实例来添加特定的 App Studio 实例 IDs。 IDs 用逗号分隔多个实例。

        要查找您的实例 ID，请在 App Studio 控制台中选择账户设置，导航到您的实例的账户设置。
     b. 选择 “生成导入代码”。
     c. 复制并共享生成的导入代码。
   - 如果已经创建了导入代码：

- 要共享当前导出的应用程序，请复制并共享现有的导入代码。要使用应用程序的最新更改创建新的导出应用程序，请选择生成新代码。如果需要，您也可以更新导入权限。

页面和组件：构建应用程序的用户界面

主题

- 管理页面
- 管理组件
- 配置基于角色的页面可见性
- 在应用程序导航中对页面进行排序和整理
- 使用应用程序主题更改应用程序中的颜色
- 组件参考

管理页面

使用以下过程在您的 AWS App Studio 应用程序中创建、编辑或删除页面。

页面是组件的容器，这些组件构成 App Studio 中应用程序的用户界面。每个页面都代表您的应用程序的用户界面 (UI) 屏幕，您的用户将与之交互。页面是在应用程序工作室的 “页面” 选项卡中创建和编辑的。

创建页面

使用以下步骤在 App Studio 的应用程序中创建页面。有关复制现有页面的信息，请参见复制页面。

创建页面

1. 如有必要，可通过编辑应用程序将其导航到其开发环境。
2. 导航到 “页面” 选项卡。
3. 在左侧的 “页面” 菜单中，选择 + 添加。

复制页面

使用以下步骤在 App Studio 的应用程序中复制页面。

复制页面

1. 如有必要，可通过编辑应用程序将其导航到其开发环境。
2. 导航到 “页面” 选项卡。
3. 在左侧的 “页面” 菜单中，选择要复制的页面名称旁边的省略号菜单，然后选择 “复制”。复制的页面直接添加到原始页面之后。

查看和编辑页面属性

使用以下步骤在 App Studio 的应用程序中编辑页面。您可以编辑诸如页面名称、其参数和布局之类的属性。

查看或编辑页面属性

1. 如有必要，可通过编辑应用程序将其导航到其开发环境。
2. 导航到 “页面” 选项卡。
3. 在左侧的 “页面” 菜单中，选择要编辑的页面名称旁边的省略号菜单，然后选择页面属性。这将打开右侧的 “属性” 菜单。
4. 要编辑页面名称，请执行以下操作：

   > **Note**
   >
   > 有效的页面名称字符：A-Z、a -z、0-9、_、$

   a. 选择 “属性” 菜单顶部附近名称旁边的铅笔图标。
   b. 输入页面的新名称，然后按 Enter。
5. 要创建、编辑或删除页面参数，请执行以下操作：

   a. 要创建页面参数，请在 “页面参数” 部分中选择 “+ 新增”。
   b. 要编辑页面参数的 “关键字” 或 “描述” 值，请选择要更改的属性的输入字段，然后输入新值。您的更改将在您编辑时保存。
   c. 要删除页面参数，请选择要删除的页面参数的垃圾图标。
6. 要添加、编辑或移除页面的徽标或横幅，请执行以下操作：

a. 要添加页面徽标或横幅，请启用样式部分中的相应选项。配置图像的来源，并可选择提供替代文本。

b. 要编辑页面徽标或横幅，请更新 “样式” 部分中的字段。

c. 要删除页面徽标或横幅，请禁用 “样式” 部分中的相应选项。

7. 要编辑页面的布局，请执行以下操作：

- 更新 “布局” 部分中的字段。

删除页面

使用以下步骤从 App Studio 中的应用程序中删除页面。

删除页面

1. 如有必要，可通过编辑应用程序将其导航到其开发环境。
2. 导航到 “页面” 选项卡。
3. 在左侧的 “页面” 菜单中，选择要删除的页面名称旁边的省略号菜单，然后选择删除。

管理组件

使用以下过程在 App Studio 应用程序工作室的页面中或页面中添加、编辑和删除组件，为您的应用程序制作所需的用户界面。

向页面添加组件

使用以下步骤向 App Studio 中的页面添加组件。有关复制现有组件的信息，请参见复制组件。

1. 如有必要，可通过编辑应用程序将其导航到其开发环境。
2. 导航到 “页面” 选项卡。
3. 组件面板位于右侧菜单中，其中包含可用组件。
4. 将所需的组件从面板拖放到画布上。或者，您可以双击面板中的组件，将其自动添加到当前页面的中心。
5. 现在，您已经添加了一个组件，请使用右侧的 “属性” 面板来调整其设置，例如数据源、布局和行为。有关配置每种组件类型的详细信息，请参见组件参考。

复制组件

使用以下步骤在 App Studio 应用程序中复制组件。复制的组件包含任何链接的自动化或原始组件中的实体。

1. 如有必要，可通过编辑应用程序将其导航到其开发环境。
2. 导航到 “页面” 选项卡。
3. 有两种方法可以复制组件：
   - 在左侧的 “页面” 菜单中，展开包含要复制的组件的页面。选择要复制的组件名称旁边的省略号菜单，然后选择 “复制”。
   - 选择要复制的组件，然后选择复制图标。

   复制的组件直接添加到原始组件之后。

   > **Tip**
   > 您可以使用 CTRL+Z 或 CMD+Z 键盘快捷键撤消组件复制以及开发环境中的许多其他操作。

查看和编辑组件属性

1. 如有必要，可通过编辑应用程序将其导航到其开发环境。
2. 导航到 “页面” 选项卡。
3. 在左侧的 “页面” 菜单中，展开包含该组件的页面，然后选择要查看或编辑的组件。或者，您可以选择页面，然后从画布中选择组件。
4. 右侧的 “属性” 面板显示选定组件的可配置设置。
5. 浏览各种可用的属性和选项，并根据需要对其进行更新以配置组件的外观和行为。例如，您可能想要更改数据源、配置布局或启用其他功能。

   有关配置每种组件类型的详细信息，请参见组件参考。

删除组件

1. 如有必要，可通过编辑应用程序将其导航到其开发环境。
2. 导航到 “页面” 选项卡。

3. 在左侧的 “页面” 菜单中，选择要删除的组件以将其选中。
4. 在右侧的 “属性” 菜单中，选择垃圾桶图标。
5. 在确认对话框中，选择删除。

配置基于角色的页面可见性

您可以在 App Studio 应用程序中创建角色，并根据这些角色配置页面的可见性。例如，您可以根据用户需求或访问权限级别为提供项目批准或索赔处理等功能的应用程序创建角色，例如管理员、经理或用户，并使某些页面对特定角色可见。在此示例中，管理员可能拥有完全访问权限，经理可能有权查看报告仪表板，而用户可以访问带有输入表单的任务页面。

使用以下步骤在 App Studio 应用程序中配置基于角色的页面可见性。

1. 如有必要，请导航到应用程序的应用程序工作室。从左侧导航菜单中，选择我的应用程序，找到您的应用程序，然后选择编辑。
2. 在应用程序工作室中创建应用程序级角色。
   a. 选择应用程序工作室顶部的应用程序设置选项卡。
   b. 选择 + 添加角色
   c. 在角色名称中，提供一个名称来标识您的角色。我们建议使用能够描述群组访问级别或职责的名称，因为您将使用该名称来设置页面可见性。
   d. 或者，在描述中，添加角色的描述。
   e. 重复这些步骤，根据需要创建任意数量的角色。
3. 配置页面的可见性
   a. 选择应用程序工作室顶部的 “页面” 选项卡。
   b. 从左侧的 “页面” 菜单中，选择要为其配置基于角色的可见性的页面。
   c. 在右侧菜单中，选择 “属性” 选项卡。
   d. 在 “可见性” 中，禁用 “向所有最终用户开放”。
   e. 选择 “角色” 以从您在上一步中创建的角色列表中进行选择。选择 “自定义”，为更复杂的可见性配置编写 JavaScript 表达式。
      1. 选择 “角色” 后，选中要显示页面的应用程序角色的复选框。
      2. 选择 “自定义” 后，输入解析为 “真” 或 “假” 的 JavaScript 表达式。使用以下示例来检查当前用户是否具有管理员的角色:`{{currentUser.roles.includes('manager')}}`.

4. 现在，您的可见性已配置完毕，您可以通过预览应用程序来测试页面的可见性。

   a. 选择 “预览” 以打开应用程序的预览。

   b. 在预览的右上角，选择 “预览为” 菜单，然后选中要测试的角色的复选框。可见的页面应反映所选的角色。

5. 现在，将群组分配给已发布应用程序的应用程序角色。必须为每个环境分别配置组和角色分配。有关应用程序环境的更多信息，请参阅应用程序环境。

   > **Note**
   >
   > 您的应用程序必须发布到测试或生产环境，才能将 App Studio 群组分配给您创建和配置的角色。如有必要，请发布您的应用程序，为角色分配群组。有关发布的更多信息，请参阅发布应用程序。

   a. 在应用程序工作室的右上角，选择共享。

   b. 选择要配置页面可见性的环境的选项卡。

   c. 选择搜索群组输入框，然后选择要与之共享应用程序版本的群组。您可以输入文本来搜索群组。

   d. 在下拉菜单中，选择要分配给群组的角色。您可以选择 “无角色” 来共享应用程序版本，也可以不为群组分配角色。只有所有用户都可见的页面才会对没有角色的群组可见。

   e. 选择共享。重复这些步骤，根据需要添加任意数量的群组。

在应用程序导航中对页面进行排序和整理

本主题包括有关在 App Studio 应用程序中重新排序和组织页面的信息。在产品的两个区域可以看到应用程序页面：在应用程序工作室编辑应用程序时在左侧的 “页面” 菜单中，以及在已发布应用程序预览的左侧导航栏中。

编辑应用程序时在左侧的 “页面” 菜单中对页面进行排序

在应用程序工作室编辑应用程序时，页面在左侧的 “页面” 菜单中按创建时间排序。您无法重新排序此菜单中的页面。

在预览或已发布的应用程序的导航中排序、显示或隐藏页面

您可以编辑预览或已发布应用程序左侧导航的以下设置：

- 整个导航的可见性
- 特定页面在导航中的可见性
- 导航中页面的顺序

编辑预览或已发布应用程序的左侧导航

1. 如有必要，请导航到应用程序的应用程序工作室进行编辑。
2. 在左侧的 “页面” 菜单中，选择 “标题和导航”。
3. 在右侧的标题和导航菜单中，查看或编辑以下内容：
   a. 要隐藏或显示应用程序中的导航，请使用应用程序导航开关。
   b. 要在应用程序导航中隐藏页面，请将页面拖到 “未链接页面” 部分。
   c. 要对应用程序导航中的页面进行重新排序，请在 “链接页面” 部分中将它们拖到所需的顺序。

使用应用程序主题更改应用程序中的颜色

使用以下步骤通过配置应用程序主题来更新应用程序中的颜色。

1. 如有必要，请导航到应用程序的应用程序工作室进行编辑。
2. 在应用程序工作室中，导航到页面选项卡。
3. 在左侧导航栏中，选择应用程序主题以打开右侧的应用程序主题设置。
4. 在基础主题中，选择浅色模式或深色模式。
5. 要向应用程序添加自定义颜色，请启用 “自定义” 开关并更新以下设置：
   a. 在 Prim ary color 中，选择应用于某些组件和应用程序导航的颜色。您可以使用颜色选择器、RGB、HSL 或 HEX 代码选择颜色。

      > **Note**
      > App Studio 将自动确保你的颜色易于访问。例如，如果您在灯光模式下选择浅色，则会对其进行更新以使其更易于访问。

   b. 在标题颜色中，选择应用于应用程序标题的颜色。您可以使用颜色选择器、RGB、HSL 或 HEX 代码选择颜色。
   c. 选择 “默认主题” 以查看预定义主题并从中进行选择，或者选择 “随机化” 以生成随机主色和标题颜色。

6. 选择 “保存更改” 以更新您的应用程序主题。

组件参考

本主题详细介绍了 App Studio 的每个组件及其属性，并包括配置示例。

常用组件属性

本节概述了应用程序工作室中多个组件共享的一般属性和功能。每种属性类型的具体实现细节和用例可能因组件而异，但这些属性的总体概念在 App Studio 中保持一致。

名称

将为每个组件生成一个默认名称；但是，您可以进行编辑以更改为每个组件的唯一名称。您将使用此名称从同一页面中的其他组件或表达式中引用该组件及其数据。限制：不要在组件名称中包含空格；它只能包含字母、数字、下划线和美元符号。示例：`userNameInput`、`ordersTable`、`metricCard1`。

主要值、次要值和值

Application Studio 中的许多组件都提供了用于指定值或表达式的字段，这些值或表达式决定了组件中显示的内容或数据。根据组件类型和用途 `Primary value``Secondary value`，这些字段通常被标记为`Value`、或简单标记。

该`Primary value`字段通常用于定义应在组件中突出显示的主值、数据点或内容。

该`Secondary value`字段（如果可用）用于在主要值旁边显示附加值或支持值或信息。

该`Value`字段允许您指定应在组件中显示的值或表达式。

这些字段支持静态文本输入和动态表达式。通过使用表达式，您可以引用应用程序中其他组件、数据源或变量的数据，从而实现动态和数据驱动的内容显示。

表达式的语法

在这些字段中输入表达式的语法遵循一致的模式：

```
{{expression}}
```

其中，*`expression`*是计算结果为所需值或要显示的数据的有效表达式。

示例：静态文本

- 主要值：您可以直接输入静态数字或值，例如"123"或"$1,999.99"。
- 次要值：您可以输入静态文本标签，例如"Goal"或"Projected Revenue"。
- 值：您可以输入静态字符串，例如"since last month"或"Total Quantity"。

示例：表达式

- Hello, {{currentUser.firstName}}：显示带有当前登录用户的名字的问候语。
- {{currentUser.role === 'Admin' ? 'Admin Dashboard' : 'User Dashboard'}}：根据用户的角色有条件地显示不同的仪表板标题。
- {{ui.componentName.data?.[0]?.fieldName}}：从 ID 为 ID componentName 的组件数据中的第一项中检索该fieldName字段的值。
- {{ui.componentName.value * 100}}：对带有 ID 的组件的值进行计算componentName。
- {{ui.componentName.value + ' items'}}：将组件的值与 ID componentName 和字符串连接起来。' items'
- {{ui.ordersTable.data?.[0]?.orderNumber}}：从ordersTable组件的第一行数据中检索订单号。
- {{ui.salesMetrics.data?.[0]?.totalRevenue * 1.15}}：通过将salesMetrics组件中第一行数据的总收入增加 15% 来计算预计收入。
- {{ui.customerProfile.data?.[0]?.firstName + ' ' + ui.customerProfile.data?.lastName}}：连接组件中数据的名字和姓氏。customerProfile
- {{new Date(ui.orderDetails.data?.orderDate).toLocaleDateString()}}：将orderDetails组件中的订单日期格式化为更具可读性的日期字符串。
- {{ui.productList.data?.length}}：显示连接到productList组件的数据中的产品总数。
- {{ui.discountPercentage.value * ui.orderTotal.value}}：根据折扣百分比和订单总额计算折扣金额。
- {{ui.cartItemCount.value + ' items in cart'}}：显示购物车中的商品数量以及标签items in cart。

通过使用这些表达式字段，可以在应用程序中创建动态和数据驱动的内容，从而显示根据用户上下文或应用程序状态量身定制的信息。这可以实现更加个性化和交互性的用户体验。

标签

L abel 属性允许您为组件指定标题或标题。此标签通常显示在组件旁边或上方，以帮助用户了解其用途。

您可以使用静态文本和表达式来定义标签。

示例：静态文本

如果您在 “标签” 字段中输入文本 “名字”，则组件的标签将显示 “名字”。

示例：表达式

示例：零售店

以下示例为每个用户个性化标签，使界面更适合个人：

```
{{currentUser.firstName}} {{currentUser.lastName}}'s Account
```

示例：SaaS 项目管理

以下示例从选定项目中提取数据以提供特定于上下文的标签，从而帮助用户在应用程序中保持定向：

```
Project {{ui.projectsTable.selectedRow.id}} - {{ui.projectsTable.selectedRow.name}}
```

示例：医疗诊所

以下示例引用了当前用户的个人资料和医生的信息，为患者提供了更加个性化的体验。

```
Dr. {{ui.doctorProfileTable.data.firstName}}
       {{ui.doctorProfileTable.data.lastName}}
```

Placeholder

Placeholder 属性允许您指定当组件为空时显示的提示或指导文本。这可以帮助用户理解预期的输入格式或提供其他上下文。

您可以使用静态文本和表达式来定义占位符。

示例：静态文本

如果在 “占位符” 字段`Enter your name`中输入文本，则该组件将显示`Enter your name`为占位符文本。

示例：表达式

示例：金融服务

`Enter the amount you'd like to deposit into your {{ui.accountsTable.selectedRow.balance}} account`这些示例从所选账户中提取数据以显示相关提示，从而使银行客户的界面变得直观。

示例：电子商务

`Enter the coupon code for {{ui.cartTable.data.currency}} total`此处的占位符会根据用户的购物车内容动态更新，从而提供无缝的结账体验。

示例：医疗诊所

`Enter your {{ui.patientProfile.data.age}}-year-old patient's symptoms`通过使用引用患者年龄的表达式，应用程序可以创建更加个性化和有用的占位符。

来源

S ourc e 属性允许您为组件选择数据源。选择后，您可以从以下数据源类型中进行选择：`entity`、`expression`、或`automation`。

实体

选择 “实体” 作为数据源可以将组件连接到应用程序中的现有数据实体或模型。当您想要在整个应用程序中使用定义明确的数据结构或架构时，这很有用。

何时使用实体数据源：

- 当您的数据模型或实体包含要在组件中显示的信息时（例如，带有 “名称”、“描述”、“价格” 等字段的 “产品” 实体）。
- 当您需要从数据库、API 或其他外部数据源动态获取数据并将其呈现在组件中时。
- 当您想利用应用程序的数据模型中定义的关系和关联时。

选择对实体的查询

有时，您可能希望将组件连接到从实体而不是整个实体检索数据的特定查询。在实体数据源中，您可以选择从现有查询中进行选择或创建新查询。

通过选择查询，您可以：

- 根据特定条件筛选组件中显示的数据。
- 向查询传递参数以动态筛选或排序数据。
- 利用查询中定义的复杂联接、聚合或其他数据操作技术。

例如，如果您的应用程序中有一个`Customers`实体，其中包含诸如`Name``Email`、和之类的字段`Phone``Number`。您可以将表格组件连接到该实体，然后选择一个预定义`Active``Customers`的数据操作，根据客户的状态筛选客户。这允许您仅显示表格中的活跃客户，而不是整个客户数据库。

向实体数据源添加参数

使用实体作为数据源时，也可以向组件添加参数。这些参数可用于筛选、排序或转换组件中显示的数据。

例如，如果您的`Products`实体包含诸如`Name`、`Description``Price`、和之类的字段`Category`。您可以将名为`category`的参数添加到显示产品列表的表格组件。当用户从下拉列表中选择类别时，表格将自动更新为仅显示属于所选类别的产品，使用数据操作中的`{{params.category}}`表达式。

Expression

选择 E x pression 作为数据源，输入自定义表达式或计算，为组件动态生成数据。当您需要执行转换、合并来自多个源的数据或根据特定的业务逻辑生成数据时，这非常有用。

何时使用表达式数据源：

- 当您需要计算或导出数据模型中无法直接提供的数据时（例如，根据数量和价格计算订单总价值）。
- 当您想要合并来自多个实体或数据源的数据以创建复合视图时（例如，显示客户的订单历史记录及其联系信息）。
- 当您需要根据特定规则或条件生成数据时（例如，根据用户的浏览历史显示 “推荐产品” 列表）。

例如，如果您有一个*`Metrics`*组件需要显示当月的总收入，则可以使用类似以下的表达式来计算和显示每月收入：

```
{{ui.table1.orders.concat(ui.table1.orderDetails).filter(o => o.orderDate.getMonth()
 === new Date().getMonth()).reduce((a, b) => a + (b.quantity * b.unitPrice), 0)}}
```

自动化

选择 Automation 作为数据源，将组件连接到应用程序中的现有自动化或工作流程。当组件的数据或功能是作为特定流程或工作流的一部分生成或更新时，这很有用。

何时使用自动化数据源：

- 当组件中显示的数据是特定自动化或工作流程的结果时（例如，作为批准流程一部分更新的 “待批准” 表）。
- 当您想根据自动化中的事件或条件触发对组件的操作或更新时（例如，使用某个 SKU 的最新销售数据更新指标）。
- 当您需要通过自动化将组件与应用程序中的其他服务或系统集成时（例如，从第三方 API 获取数据并将其显示在表格中）。

例如，如果您有一个用于指导用户完成求职流程的 stepflow 组件。stepflow 组件可以连接到处理求职申请提交、背景调查和录用生成自动化。随着这些步骤的自动化，stepflow 组件可以动态更新以反映应用程序的当前状态。

通过为每个组件仔细选择适当的数据源，您可以确保应用程序的用户界面由正确的数据和逻辑提供支持，从而为用户提供无缝且引人入胜的体验。

在以下情况下可见

使用 V isible if 属性根据特定条件或数据值显示或隐藏组件或元素。当您想要动态控制应用程序用户界面某些部分的可见性时，这很有用。

Visi ble if 属性使用以下语法：

```
{{expression ? true : false}}
```

或

```
{{expression}}
```

其中*expression*，是计算结果为`true`或`false`的布尔表达式。

如果表达式的计算结果为`true`，则该组件将可见。如果表达式的计算结果为`false`，则该组件将被隐藏。该表达式可以引用应用程序中其他组件、数据源或变量的值。

if 表达式示例可见

示例：根据电子邮件输入显示或隐藏密码输入字段

想象一下，你有一个带有电子邮件输入字段和密码输入字段的登录表单。只有当用户输入了电子邮件地址时，才需要显示密码输入字段。你可以使用以下 Visible if 表达式：

```
{{ui.emailInput.value !== ""}}
```

此表达式检查emailInput组件的值是否不是空字符串。如果用户输入了电子邮件地址，则表达式的计算结果为true，密码输入字段将可见。如果电子邮件字段为空，则表达式的计算结果为false，密码输入字段将被隐藏。

示例：根据下拉列表选择显示其他表单域

假设你有一个表单，用户可以在其中从下拉列表中选择一个类别。根据所选类别，您想要显示或隐藏其他表单域以收集更具体的信息。

例如，如果用户选择*Products*类别，则可以使用以下表达式来显示其他*Product Details*字段：

```
{{ui.categoryDropdown.value === "Products"}}
```

如果用户选择*Services*或*Consulting*类别，则可以使用此表达式来显示一组不同的附加字段：

```
{{ui.categoryDropdown.value === "Services" || ui.categoryDropdown.value ===
 "Consulting"}}
```

示例：其他

要在组件的值不是空字符串时使该textInput1组件可见，请执行以下操作：

```
{{ui.textInput1.value === "" ? false : true}}
```

要使组件始终可见，请执行以下操作：

```
{{true}}
```

要在组件的值不是空字符串时使该emailInput组件可见，请执行以下操作：

```
{{ui.emailInput.value !== ""}}
```

在以下情况下禁用

“如果 Disabled if” 功能允许您根据特定条件或数据值有条件地启用或禁用组件。这是通过使用 Disabled i f 属性来实现的，该属性接受一个布尔表达式，用于确定应启用还是禁用组件。

如果 Disab led if 属性使用以下语法：

```
{{expression ? true : false}}
```

或

```
{{expression}}
```

如果表达式示例，则禁用

示例：根据表单验证禁用提交按钮

如果您的表单包含多个输入字段，并且想要在正确填写所有必填字段之前禁用提交按钮，则可以使用以下 Disabled I f 表达式：

```
{{ui.nameInput.value === "" || ui.emailInput.value === "" || ui.passwordInput.value ===
 ""}}
```

此表达式检查必填的输入字段（`nameInput`、`emailInput`、`passwordInput`）是否为空。如果任何字段为空，则表达式的计算结果为`true`，提交按钮将被禁用。填写完所有必填字段后，表达式的计算结果为`false`，提交按钮将启用。

通过在 Visib le if 和 Dis abled ff 属性中使用这些类型的条件表达式，您可以创建适应用户输入的动态响应式用户界面，从而为应用程序的用户提供更加精简和相关的体验。

其中*`expression`*，是计算结果为真或假的布尔表达式。

示例：

```
{{ui.textInput1.value === "" ? true : false}}: The component will be Disabled if the
 textInput1 component's value is an empty string.
{{!ui.nameInput.isValid || !ui.emailInput.isValid || !ui.passwordInput.isValid}}: The
 component will be Disabled if any of the named input fields are invalid.
```

容器布局

布局属性决定了组件中一个或多个内容的排列和定位方式。有几个布局选项可供选择，每个选项都由一个图标表示：

- 列布局：此布局将内容或元素垂直排列在一列中。

- 双列布局：此布局将组件分成两个等宽的列，允许您并排放置内容或元素。
- 行布局：此布局将内容或元素水平排列在一行中。

方向

- 横向：此布局将内容或元素水平排列成一行。
- 垂直：此布局将内容或元素垂直排列在一列中。
- 内联换行：此布局将内容或元素水平排列，但如果元素超过可用宽度，则换行到下一行。

一致性

- 左：将一个或多个内容与组件的左侧对齐。
- 居中：将内容或元素在组件内水平居中。
- 右：将一个或多个内容与组件的右侧对齐。

宽度

宽度属性指定组件的水平大小。您可以输入介于 0% 和 100% 之间的百分比值，表示组件相对于其父容器或可用空间的宽度。

高度

He ig ht 属性指定组件的垂直大小。“auto” 值根据组件的内容或可用空间自动调整组件的高度。

两者之间的空间

“间距” 属性决定组件内内容或元素之间的间距或间距。你可以选择一个从 0px（无间距）到 64px 的值，增量为 4px（例如，4px、8px、12px 等）。

Padding

Paddin g 属性控制一个或多个内容与组件边缘之间的空间。你可以选择一个从 0px（无填充）到 64px 的值，增量为 4px（例如 4px、8px、12px 等）。

背景

背景启用或禁用组件的背景颜色或样式。

这些布局属性可以灵活地排列和定位组件内的内容，以及控制组件本身的大小、间距和视觉外观。

数据组件

本节介绍应用程序工作室中可用的各种数据组件，包括表、详细信息、指标、表单和中继器组件。这些组件用于在应用程序中显示、收集和操作数据。

表

表格组件以表格格式显示数据，包括行和列。它用于以有条理 easy-to-read的方式呈现结构化数据，例如数据库中的项目列表或记录。

表属性

表组件与其他组件共享多个共同属性，例如`NameSource`、和`Actions`。有关这些属性的更多信息，请参阅常用组件属性。

除了常用属性外，表组件还具有特定的属性和配置选项，包括`ColumnsSearch and export`、和`Expressions`。

Columns

在本节中，您可以定义要在表格中显示的列。可以为每列配置以下属性：

- 格式：字段的数据类型，例如：文本、数字、日期。
- 列标签：列的标题文本。
- 值：数据源中应显示在此列中的字段。

  此字段允许您指定应在列单元格中显示的值或表达式。您可以使用表达式来引用来自连接源或其他组件的数据。

  示例：`{{currentRow.title}}`-此表达式将在列单元格中显示当前行的`title`字段值。
- 启用排序：此开关允许您启用或禁用特定列的排序功能。启用后，用户可以根据此列中的值对表格数据进行排序。

搜索和导出

表格组件提供以下开关，用于启用或禁用搜索和导出功能：

- 显示搜索启用后，此开关会向表格中添加搜索输入字段，允许用户搜索和筛选显示的数据。
- 显示导出启用后，此开关会向表格添加导出选项，允许用户下载各种格式的表格数据，例如：CSV。

> **Note**
>
> 默认情况下，搜索功能仅限于已加载到表中的数据。要全面使用搜索，您需要加载所有页面的数据。

每页行数

您可以指定表格中每页要显示的行数。然后，用户可以在页面之间导航以查看完整的数据集。

预取限制

指定每个查询请求中要预取的最大记录数。最大值为 3000。

操作

在 “操作” 部分中，配置以下属性：

- 操作位置：启用 “固定到右侧” 后，无论用户如何滚动，任何添加的操作都将始终显示在表格的右侧。
- 操作：向表格中添加操作按钮。您可以将这些按钮配置为在用户单击时执行指定的操作，例如：
  - 运行组件操作
  - 导航到其他页面
  - 调用数据操作
  - 运行自定义 JavaScript
  - 调用自动化

Expressions

表格组件提供了多个使用表达式和行级操作功能的区域，允许您自定义和增强表格的功能和交互性。它们允许您在表格中动态引用和显示数据。通过利用这些表达式字段，您可以创建动态列，将数据传递给行级操作，以及引用应用程序中其他组件或表达式中的表数据。

示例：引用行值

`{{currentRow.columnName}}`或者`{{currentRow["Column Name"]}}`这些表达式允许您为正在呈现的当前行引用特定列的值。`Column Name`用要引用的列的实际名称替换`columnName`或。

示例：

- {{currentRow.productName}}显示当前行的产品名称。
- {{currentRow["*Supplier Name*"]}}显示当前行（列标题所在行）的供应商名称*Supplier Name*。
- {{currentRow.orderDate}}显示当前行的订单日期。

示例：引用所选行

{{ui.*table1*.selectedRow["columnName"]}}此表达式允许您使用 ID 引用表中当前选定行的特定列的值*table1*。*table1*替换为表组件的实际 ID *columnName* 以及要引用的列的名称。

示例：

- {{ui.*ordersTable*.selectedRow["totalAmount"]}}显示带有 ID 的表中当前选定行的总金额*ordersTable*。
- {{ui.*customersTable*.selectedRow["email"]}}显示表格中当前选定行的电子邮件地址和 ID *customersTable*。
- {{ui.*employeesTable*.selectedRow["department"]}}显示表格中当前选定行的部门，标识为 ID *employeesTable*。

示例：创建自定义列

您可以根据从基础数据操作、自动化或表达式返回的数据向表中添加自定义列。您可以使用现有的列值和 JavaScript 表达式来创建新列。

示例：

- {{currentRow.quantity * currentRow.unitPrice}}通过将数量和单价列相乘来创建显示总价的新列。
- {{new Date(currentRow.orderDate).toLocaleDateString()}}创建新列，以更易读的格式显示订单日期。
- {{currentRow.firstName + ' ' + currentRow.lastName + ' (' + currentRow.email + ')' }}创建新列，显示每行的全名和电子邮件地址。

示例：自定义列显示值：

您可以通过设置列映射的字段来自定义表列中Value字段的显示值。这允许您对显示的数据应用自定义格式或转换。

示例：

- {{ currentRow.rating >= 4 ? '##'.repeat(currentRow.rating) : currentRow.rating }}根据每行的评分值显示星形表情符号。
- {{ currentRow.category.toLowerCase().replace(/\b\w/g, c => c.toUpperCase()) }}显示类别值，每行的每个单词都大写。
- {{ currentRow.status === 'Active' ? '# Active' : '# Inactive' }}：根据每行的状态值显示彩色圆圈表情符号和文本。

行级按钮操作

{{currentRow.columnName}}或者，{{currentRow["Column Name"]}}您可以使用这些表达式在行级操作中传递被引用的行的上下文，例如使用选定行的数据导航到另一页或触发行数据的自动化。

示例：

- 如果您在行操作列中有一个编辑按钮，则可以{{currentRow.orderId}}作为参数传递，以导航到带有所选订单编号的订单编辑页面。
- 如果您在行操作列中有一个删除按钮，则可以转{{currentRow.customerName}}到自动化，该自动化会在删除订单之前向客户发送确认电子邮件。
- 如果您在行操作列中有 “查看详情” 按钮，则可以转{{currentRow.employeeId}}到记录查看订单详细信息的员工的自动化。

通过利用这些表达式字段和行级操作功能，您可以创建高度自定义的交互式表格，根据您的特定要求显示和操作数据。此外，您可以将行级操作与应用程序中的其他组件或自动化连接起来，从而实现无缝的数据流和功能。

Detail

详细信息组件旨在显示有关特定记录或项目的详细信息。它提供了一个专门的空间，用于展示与单个实体或行相关的全面数据，非常适合展示深入的细节或简化数据输入和编辑任务。

细节属性

D et ail 组件与其他组件共享多个共同属性Name，例如Source、和Actions。有关这些属性的更多信息，请参阅常用组件属性。

细节组件还具有特定的属性和配置选项，包括FieldsLayout、和Expressions。

布局

布局部分允许您自定义详细信息组件中字段的排列和显示方式。您可以配置选项，例如：

- 列数：指定要在其中显示字段的列数。
- 字段排序：拖放字段以对其外观进行重新排序。
- 间距和对齐方式：调整组件内字段的间距和对齐方式。

表达式和示例

D et ail 组件提供了各种表达式字段，允许您动态引用和显示组件内的数据。这些表达式使您能够创建与应用程序的数据和逻辑无缝连接的自定义交互式细节组件。

示例：引用数据

`{{ui.details.data[0]?.["colName"]}}`：此表达式允许您为连接到 Detail 组件的数据数组中的第一项（索引 0）引用名为 “colName” 的列的值，ID 为 “details”。将 “colName” 替换为要引用的列的实际名称。例如，以下表达式将显示连接到 “详情” 组件的数据数组中第一项的 “CustomerName” 列的值：

```
{{ui.details.data[0]?.["customerName"]}}
```

> **Note**
> 当详细信息组件与被引用的表位于同一页上，并且您想在详细信息组件中显示表格第一行的数据时，此表达式很有用。

示例：条件渲染

`{{ui.table1.selectedRow["colName"]}}`：如果表中选定的 ID *table1* 行包含名为的列的数据，则此表达式返回 true *colName*。它可用于根据表格的选定行是否为空有条件地显示或隐藏 Detail 组件。

示例：

您可以在 Detail 组件的**Visible if**属性中使用此表达式，根据表格中的选定行有条件地显示或隐藏该表达式。

```
{{ui.table1.selectedRow["customerName"]}}
```

如果此表达式的计算结果为 true（*table1*组件中的选定行具有该*customerName*列的值），则 Detai l 组件将可见。如果表达式的计算结果为 false（即所选行为空或没有 “CustomerName” 的值），则详细信息组件将被隐藏。

示例：条件显示

`{{(ui.Component.value === "green" ? "#" : ui.Component.value === "yellow" ? "#" : ui.detail1.data?.[0]?.CustomerStatus)}}`：此表达式根据组件或数据字段的值有条件地显示表情符号。

细分：

- `ui.`*`Component`*`.value`：引用带有 ID 的组件的值*Component*。
- `=== "green"`：检查组件的值是否等于字符串 “green”。
- `? "#"`: 如果条件为真，则显示绿色圆圈表情符号。
- `: ui.Component.value === "yellow" ? "#"`: 如果第一个条件为 false，则检查组件的值是否等于字符串 “yellow”。
- `? "#"`: 如果第二个条件为真，则显示黄色方块表情符号。
- `: ui.detail1.data?.[0]?.CustomerStatus`：如果两个条件都为假，则它会引用连接到 Detail 组件的数据数组中第一个项目的 “CustomerStatus” 值，ID 为 “detail1”。

此表达式可用于根据详细信息组件中的组件或数据字段的值显示表情符号或特定值。

Metrics

M etrics 组件是一个视觉元素，它以类似卡片的格式显示关键指标或数据点。它旨在提供一种简洁且具有视觉吸引力的方式来呈现重要信息或绩效指标。

指标属性

Metrics 组件与其他组件共享多个共同属性`Name`，例如`Source`、和`Actions`。有关这些属性的更多信息，请参阅常用组件属性。

趋势

指标的趋势功能允许您显示绩效的可视指标，或者显示的指标会随着时间的推移而发生变化。

趋势值

此字段允许您指定用于确定趋势方向和幅度的值或表达式。通常，这是一个代表特定时间段内的变化或性能的值。

示例：

```
{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue}}
```

此表达式检索与 “SalesMetrics” 指标关联的数据中第一项的 month-over-month收入值。

积极的趋势

此字段允许您输入定义正向趋势条件的表达式。表达式的计算结果应为真或假。

示例：

```
{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue > 0}}
```

此表达式检查 month-over-month收入值是否大于 0，表示呈正趋势。

负面趋势

此字段允许您输入定义负趋势条件的表达式。表达式的计算结果应为真或假。

示例：

```
{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue < 0}}
```

此表达式检查 month-over-month收入值是否小于 0，表示呈负趋势。

色条

此切换允许您启用或禁用彩条的显示，以直观地指示趋势状态。

颜色条示例：

示例：销售指标趋势

- 趋势值：`{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue}}`
- 积极趋势：`{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue > 0}}`
- 负面趋势：`{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue < 0}}`
- 颜色栏：已启用

示例：库存指标趋势

- 趋势值：`{{ui.inventoryMetrics.data?.[0]?.currentInventory - ui.inventoryMetrics.data?.[1]?.currentInventory}}`
- 积极趋势：`{{ui.inventoryMetrics.data?.[0]?.currentInventory > ui.inventoryMetrics.data?.[1]?.currentInventory}}`
- 负面趋势：`{{ui.inventoryMetrics.data?.[0]?.currentInventory < ui.inventoryMetrics.data?.[1]?.currentInventory}}`
- 颜色栏：已启用

示例：客户满意度趋势

- 趋势值：`{{ui.customerSatisfactionMetrics.data?.[0]?.npsScore}}`
- 积极趋势：`{{ui.customerSatisfactionMetrics.data?.[0]?.npsScore >= 8}}`
- 负面趋势：`{{ui.customerSatisfactionMetrics.data?.[0]?.npsScore < 7}}`
- 颜色栏：已启用

通过配置这些与趋势相关的属性，您可以创建指标组件，这些组件可以直观地呈现所显示的指标的性能或随时间推移而发生的变化。

通过利用这些表达式，您可以创建高度自定义的交互式指标组件，这些组件可以动态引用和显示数据，从而允许您在应用程序中展示关键指标、绩效指标和数据驱动的可视化效果。

指标表达式示例

在属性面板中，您可以输入表达式来显示标题、主要值、次要值和值标题以动态显示值。

示例：引用主值

`{{ui.`*`metric1`*`.primaryValue}}`：此表达式允许您使用*`metric1`*来自同一页面中其他组件或表达式的 ID 来引用 M etrics 组件的主要值。

示例：`{{ui.`*`salesMetrics`*`.primaryValue}}`将显示*`salesMetrics`*指标组件的主要值。

示例：引用次要值

`{{ui.`*`metric1`*`.secondaryValue}}`：此表达式允许您使用*`metric1`*来自同一页面中其他组件或表达式的 ID 来引用 M etrics 组件的辅助值。

示例：{{ui.*revenueMetrics*.secondaryValue}}将显示*revenueMetrics*指标组件的次要值。

示例：引用数据

{{ui.*metric1*.data}}：此表达式允许您使用*metric1*来自同一页面中其他组件或表达式的 ID 来引用 M etrics 组件的数据。

示例：{{ui.*kpiMetrics*.data}}将引用与*kpiMetrics*指标组件相关的数据。

示例：显示特定的数据值：

{{ui.*metric1*.data?.[0]?.*id*}}：此表达式是一个示例，说明如何显示连接到 M etrics 组件的数据中的特定信息，ID 为*metric1*。当您想要显示数据中第一项的特定属性时，它很有用。

细分：

- ui.*metric1*：引用带有 ID 的指标组件*metric1*。
- data：指与该组件相关的信息或数据集。
- ?.[0]: 表示该数据集中的第一个项目或条目。
- ?.id：显示第一个项目或条目的*id*值或标识符。

示例：{{ui.*orderMetrics*.data?.[0]?.*orderId*}}将显示连接到 M *orderMetrics* etrics 组件的数据中第一项的*orderId*值。

示例：显示数据长度

{{ui.*metric1*.data?.length}}：此表达式演示如何显示与 ID 为 M etrics 组件关联的数据中的长度（项目数）*metric1*。当您想要显示数据中的项目数时，它很有用。

细分：

- ui.*metric1*.data：引用连接到组件的数据集。
- ?.length：访问该数据集中项目或条目的总数或数量。

示例：{{ui.*productMetrics*.data?.length}}将显示数据中与 M *productMetrics* etrics 组件关联的项目数。

中继器

Repeater 组件是一个动态组件，允许您根据提供的数据源生成和显示元素集合。它旨在便于在应用程序的用户界面中创建列表、网格或重复模式。一些示例用例包括：

- 为账户中的每位用户显示一张卡片
- 显示包含图片的产品列表和将其添加到购物车的按钮
- 显示用户可以访问的文件列表

Repeater 组件与内容丰富的表格组件区分开来。表组件具有严格的行和列格式。中继器可以更灵活地显示您的数据。

中继器属性

Repeater 组件与其他组件共享多个共同属性，例如`NameSource`、和。`Actions`有关这些属性的更多信息，请参阅常用组件属性。

除了常用属性外，Repeater 组件还具有以下其他属性和配置选项。

商品模板

项目模板是一个容器，您可以在其中定义结构和组件，这些结构和组件将为数据源中的每个项目重复使用。您可以将其他组件拖放到此容器中，例如文本、图像、按钮或表示每个项目所需的任何其他组件。

在项目模板中，您可以使用格式中的表达式引用当前项目的属性或值`{{currentItem.propertyName}}`。

例如，如果您的数据源包含一个`itemName`属性，则可以使用`{{currentItem.itemName}}`来显示当前项目的项目名称。

布局

布局部分允许您配置中继器组件中重复元素的排列。

方向

- 列表：将重复的元素垂直排列在一列中。
- 网格：在具有多列的网格布局中排列重复的元素。

每页行数

指定列表布局中每页显示的行数。为溢出指定行数的项目提供分页功能。

每页列数和每页行数（网格）

- 列：指定网格布局中的列数。

- 每页行数：指定网格布局中每页显示的行数。为超出指定网格尺寸的项目提供分页。

表达式和示例

Repeater 组件提供了各种表达式字段，允许您动态引用和显示组件内的数据。这些表达式使您能够创建自定义的交互式 Repeater 组件，这些组件可以与应用程序的数据和逻辑无缝连接。

示例：引用项目

- `{{currentItem.propertyName}}`：引用项目模板中当前项目的属性或值。
- `{{ui.repeaterID[index]}}`：通过索引引用 Repeater 组件中的特定项目。

示例：呈现产品列表

- 来源：选择*Products*实体作为数据源。
- 商品模板：添加一个容器组件，里面有一个文本组件来显示产品名称（`{{currentItem.productName}}`），一个图片组件来显示产品图片（`{{currentItem.productImageUrl}}`）。
- 布局：将设置为`Orientation`，`List`然后根据`Rows per Page`需要进行调整。

示例：生成用户头像网格

- 来源：使用表达式生成用户数据数组（例如`[{name: 'John', avatarUrl: '...'}, {...}, {...}]`）。
- 项目模板：添加图像组件并将其`Source`属性设置为`{{currentItem.avatarUrl}}`。
- 布局：`Orientation`将设置为`Grid`，指定`Columns`和的数量`Rows per Page`，然后根据需要调整`Space Between`和`Padding`。

通过使用该`Repeater`组件，您可以创建动态和数据驱动的用户界面，从而简化元素集合的渲染过程并减少手动重复或硬编码的需求。

表单

表单组件旨在捕获用户输入并简化应用程序中的数据输入任务。它提供了用于显示输入字段、下拉菜单、复选框和其他表单控件的结构化布局，允许用户无缝输入或修改数据。可以在表单组件（例如表格）中嵌套其他组件。

表单属性

表单组件与其他组件共享多个共同属性，例如`NameSource`、和`Actions`。有关这些属性的更多信息，请参阅常用组件属性。

生成表单

使用生成表单功能，可以根据所选数据源自动填充表单域，从而轻松快速创建表单域。在构建需要显示大量字段的表单时，这可以节省时间和精力。

要使用 “生成表单” 功能，请执行以下操作：

1. 在表单组件的属性中，找到 “生成表单” 部分。
2. 选择要用于生成表单域的数据源。这可以是您的应用程序中可用的实体、工作流程或任何其他数据源。
3. 表单字段将根据所选数据源自动生成，包括字段标签、类型和数据映射。
4. 查看生成的字段并进行任何必要的自定义，例如添加验证规则或更改字段顺序。
5. 对表单配置感到满意后，选择 “提交”，将生成的字段应用于您的表单组件。

当应用程序中有一个定义明确的数据模型或一组需要捕获用户输入的实体时，生成表单功能特别有用。通过自动生成表单字段，您可以节省时间并确保应用程序表单之间的一致性。

使用 “生成表单” 功能后，您可以进一步自定义表单组件的布局、操作和表达式以满足您的特定要求。

表达式和示例

与其他组件一样，您可以使用表达式来引用和显示表单组件中的数据。例如：

- `{{ui.userForm.data.email}}`：引用连接到 ID 为 Form 组件的数据源中的`email`字段值`userForm`。

> **Note**
>
> 有关常常用组件属性用属性的更多表达式示例，请参阅。

通过配置这些属性并利用表达式，您可以创建与应用程序数据源和逻辑无缝集成的自定义交互式表单组件。这些组件可用于捕获用户输入、显示预先填充的数据，以及根据表单提交或用户互动触发操作。

阶梯流

Stepflow 组件旨在指导用户完成应用程序中的多步骤流程或工作流程。它提供了一个结构化且直观的界面，用于呈现一系列步骤，每个步骤都有自己的输入、验证和操作集。

Stepflow 组件与其他组件共享多个共同属性，例如`NameSource`、和。`Actions`有关这些属性的更多信息，请参阅常用组件属性。

Stepflow 组件具有其他属性和配置选项，例如`Step NavigationValidation`、和。`Expressions`

AI 组件

人工智能世代

Gen AI 组件是一个分组容器，用于对组件及其随附的逻辑进行分组，以便在应用程序工作室中使用聊天功能使用 AI 轻松编辑它们。当你使用聊天创建组件时，它们将被分组到一个 Gen AI 容器中。有关编辑或使用此组件的信息，请参见构建或编辑您的应用程序。

文本和数字组件

文本输入

文本输入组件允许用户在您的应用程序中输入和提交文本数据。它提供了一种简单直观的方式来捕获用户输入，例如姓名、地址或任何其他文本信息。

- `{{ui.inputTextID.value}}`：返回输入字段中提供的值。
- `{{ui.inputTextID.isValid}}`：返回输入字段中提供的值的有效性。

文本

文本组件用于在应用程序中显示文本信息。它可用于显示静态文本、动态值或由表达式生成的内容。

文字区域

文本区域组件旨在捕获用户的多行文本输入。它提供了更大的输入字段区域，供用户输入更长的文本条目，例如描述、注释或注释。

- `{{ui.textAreaID.value}}`：返回文本区域中提供的值。
- `{{ui.textAreaID.isValid}}`：返回文本区域中提供的值的有效性。

电子邮件

电子邮件组件是一个专门的输入字段，旨在捕获用户的电子邮件地址。它可以强制执行特定的验证规则，以确保输入的值符合正确的电子邮件格式。

- {{ui.*emailID*.value}}：返回电子邮件输入字段中提供的值。
- {{ui.*emailID*.isValid}}：返回电子邮件输入字段中提供的值的有效性。

密码

P as sword 组件是一个专门为用户输入敏感信息（例如密码或 PIN 码）而设计的输入字段。它会屏蔽输入的字符以维护隐私和安全。

- {{ui.*passwordID*.value}}：返回密码输入字段中提供的值。
- {{ui.*passwordID*.isValid}}：返回密码输入字段中提供的值的有效性。

Search

搜索组件为用户提供了一个专用的输入字段，用于在应用程序中填充的数据中执行搜索查询或输入搜索词。

- {{ui.*searchID*.value}}：返回搜索字段中提供的值。

Phone

“电话” 组件是一个输入字段，专为捕获用户的电话号码或其他联系信息而量身定制。它可以包括特定的验证规则和格式选项，以确保输入的值符合正确的电话号码格式。

- {{ui.*phoneID*.value}}：返回电话输入字段中提供的值。
- {{ui.*phoneID*.isValid}}：返回电话输入字段中提供的值的有效性。

数字

数字分量是一个专门为用户输入数值而设计的输入字段。它可以强制执行验证规则，以确保输入的值是指定范围或格式内的有效数字。

- {{ui.*numberID*.value}}：返回数字输入字段中提供的值。
- {{ui.*numberID*.isValid}}：返回数字输入字段中提供的值的有效性。

货币

货币部分是一个专门的输入字段，用于捕获货币价值或金额。它可以包括用于显示货币符号的格式化选项、十进制分隔符，并强制执行特定于货币输入的验证规则。

- `{{ui.currencyID.value}}`：返回货币输入字段中提供的值。
- `{{ui.currencyID.isValid}}`：返回货币输入字段中提供的值的有效性。

细节对

D etail pair 组件用于以结构化和可读的格式显示键值对或相关信息对。它通常用于呈现与特定项目或实体相关的详细信息或元数据。

选择组件

Switch

S witch 组件是一个用户界面控件，允许用户在两种状态或选项之间切换，例如on/off, true/false, or enabled/disabled。它提供了当前状态的可视化表示，并允许用户通过单击或点击进行更改。

交换机组

Switch 组组件是单个开关控件的集合，允许用户从预定义的集合中选择一个或多个选项。它提供了所选选项和未选中选项的可视化表示，使用户更容易理解可用选项并与之交互。

切换群组表达式字段

- `{{ui.switchGroupID.value}}`：返回一个字符串数组，其中包含应用程序用户启用的每个开关的值。

复选框组

复选框组组件为用户提供一组复选框，允许他们同时选择多个选项。当您想让用户能够从选项列表中选择一个或多个项目时，它很有用。

复选框组表达式字段

- `{{ui.checkboxGroupID.value}}`：返回一个字符串数组，其中包含应用程序用户选择的每个复选框的值。

电台小组

Radio group 组件是一组单选按钮，允许用户从多个互斥的选项中选择一个选项。它确保一次只能选择一个选项，从而为用户提供了一种清晰而明确的选择方式。

无线电群组表达式字段

以下字段可以在表达式中使用。

- `{{ui.radioGroupID.value}}`：返回应用程序用户选择的单选按钮的值。

单选

S ing le select 组件为用户提供选项列表，用户可以从中选择单个项目。它通常用于用户需要从一组预定义的选项中进行选择的场景，例如选择类别、位置或首选项。

单选表达式字段

- `{{ui.singleSelectID.value}}`：返回应用程序用户选择的列表项的值。

多选

多选组件与单选组件类似，但允许用户从选项列表中同时选择多个选项。当用户需要从一组预定义的选项中进行多个选择（例如选择多个标签、兴趣或偏好）时，它很有用。

多选表达式字段

- `{{ui.multiSelectID.value}}`：返回一个字符串数组，其中包含应用程序用户选择的每个列表项的值。

按钮和导航组件

Application Studio 提供了各种按钮和导航组件，允许用户触发操作并在您的应用程序中导航。

按钮组件

可用的按钮组件有：

- 按钮
- 轮廓按钮

- 图标按钮
- “文本” 按钮

这些按钮组件具有以下共同属性：

内容

- 按钮标签：要在按钮上显示的文本。

类型

- 按钮：标准按钮。
- 轮廓：带有轮廓样式的纽扣。
- 图标：带有图标的按钮。
- 文本：纯文字按钮。

大小

按钮的大小。可能的值为 `Small`、`Medium` 和 `Large`。

图标

您可以从按钮上显示的各种图标中进行选择，包括：

- 信封已关闭
- 贝尔
- 人员
- 汉堡菜单
- Search
- 信息圈出
- 装备
- V 形左
- V 形右
- 水平圆点
- 垃圾桶

- 编辑
- Check
- Close
- 主页
- 再加上

触发

单击按钮时，您可以配置一个或多个要触发的操作。可用的操作类型有：

- 基本
  - 运行组件操作：在组件内执行特定操作。
  - 导航：导航到其他页面或视图。
  - 调用数据操作：触发与数据相关的操作，例如创建、更新或删除记录。
- 高级
  - JavaScript: 运行自定义 JavaScript 代码。
  - 调用自动化：启动现有的自动化或工作流程。

JavaScript 操作按钮属性

选择`JavaScript`操作类型以在单击按钮时运行自定义 JavaScript 代码。

源代码

在该`Source code`字段中，您可以输入您的 JavaScript 表达式或函数。例如：

```
return "Hello World";
```

这只会在点击按钮`Hello World`时返回字符串。

条件：如果 “运行”

您还可以提供一个布尔表达式来确定是否应执行 JavaScript 操作。它使用以下语法：

```
{{ui.textinput1.value !== ""}}
```

在此示例中，只有当`textinput1`组件的值不是空字符串时，该 JavaScript 操作才会运行。

通过使用这些高级触发器选项，您可以创建高度自定义的按钮行为，这些行为直接与应用程序的逻辑和数据集成。这使您可以扩展按钮的内置功能，并根据您的特定要求定制用户体验。

> **Note**
>
> 务必彻底测试您的 JavaScript 操作，确保它们按预期运行。

超链接

Hyperlink 组件提供了一个可点击的链接，用于导航到外部 URLs 或内部应用程序路由。

超链接属性

内容

- 超链接标签：要显示为超链接标签的文本。

URL

超链接的目标 URL，可以是外部网站或内部应用程序路由。

触发

单击超链接时，您可以配置一个或多个要触发的操作。可用的操作类型与按钮组件的操作类型相同。

日期和时间组件

日期

日期组件允许用户选择和输入日期。

日期组件与其他组件共享多个共同属性，例如`NameSource`、和`Validation`。有关这些属性的更多信息，请参阅常用组件属性。

除了常用属性外，Dat e 组件还具有以下特定属性：

日期属性

格式

- YYYY/MM/DD、DD/MM/YYYY、YYYY/MM/DD、YYYY/DD/MM、MM/DD、DD/MM：显示日期的格式。

值

- YYYY-MM-DD：内部存储日期值的格式。

最小日期

- YYYY-MM-DD：可以选择的最小日期。

> **Note**
>
> 此值必须与的格式匹配YYYY-MM-DD。

最大日期

- YYYY-MM-DD：可以选择的最大日期。

> **Note**
>
> 此值必须与的格式匹配YYYY-MM-DD。

日历类型

- 1 个月，2 个月：要显示的日历用户界面的类型。

禁用日期

- 来源：应禁用的日期的数据源。例如：无，表达式。
- 禁用日期：用于确定应禁用哪些日期的表达式，例如：
  - `{{currentRow.column}}`: 禁用与此表达式计算结果相匹配的日期。
  - `{{new Date(currentRow.dateColumn) < new Date("2023-01-01")}}`: 禁用 2023 年 1 月 1 日之前的日期
  - `{{new Date(currentRow.dateColumn).getDay() === 0 || new Date(currentRow.dateColumn).getDay() === 6}}`: 禁用周末。

行为

- 在以下@@ 条件下可见：确定日期组件可见性的表达式。
- i@@ f 禁用：用于确定是否应禁用 Dat e 组件的表达式。

验证

“验证” 部分允许您为日期输入定义其他规则和约束。通过配置这些验证规则，可以确保用户输入的日期值符合应用程序的特定要求。您可以添加以下类型的验证：

- 必填：此开关可确保用户在提交表单之前必须输入日期值。
- 自定义：您可以使用 JavaScript表达式创建自定义验证规则。例如：

```
{{new Date(ui.dateInput.value) < new Date("2023-01-01")}}
```

  此表达式检查输入的日期是否在 2023 年 1 月 1 日之前。如果条件为真，则验证将失败。

  您还可以提供一条自定义验证消息，以便在不满足验证时显示：

```
"Validation not met. The date must be on or after January 1, 2023."
```

通过配置这些验证规则，可以确保用户输入的日期值符合应用程序的特定要求。

表达式和示例

日期组件提供以下表达式字段：

- `{{ui.dateID.value}}`：返回用户以格式输入的日期值`YYYY-MM-DD`。

Time

时间组件允许用户选择和输入时间值。通过配置时间组件的各种属性，您可以创建满足应用程序特定要求的时间输入字段，例如限制可选的时间范围、禁用某些时间以及控制组件的可见性和交互性。

时间属性

时间组件与其他组件共享几个共同的属性，例如`NameSource`、和`Validation`。有关这些属性的更多信息，请参阅常用组件属性。

除了常用属性外，T im e 组件还具有以下特定属性：

时间间隔

- 5 分钟、10 分钟、15 分钟、20 分钟、25 分钟、30 分钟、60 分钟：可用于选择时间的时间间隔。

值

- HH: MM AA：内部存储时间值的格式。

> **Note**
> 此值必须与的格式匹配`HH:MM AA`。

Placeholder

- 日历设置：时间字段为空时显示的占位符文本。

最小时间

- HH: MM AA：可以选择的最短时间。

> **Note**
> 此值必须与的格式匹配`HH:MM AA`。

最大时间

- HH: MM AA：可以选择的最长时间。

> **Note**
> 此值必须与的格式匹配`HH:MM AA`。

禁用时间

- 来源：应禁用的时间的数据源（例如，无、表达式）。
- 禁用时间：用于确定应禁用哪些时间的表达式，例如`{{currentRow.column}}`。

禁用时间配置

您可以使用 “禁用时间” 部分来指定哪些时间值不可选择。

来源

- 无：不禁用任何时间。
- 表达式：您可以使用 JavaScript 表达式来确定应禁用哪些时间，例如`{{currentRow.column}}`。

表达式示例：

```
{{currentRow.column === "Lunch Break"}}
```

当当前行的 “午休时间” 列为真时，此表达式将禁用。

通过配置这些验证规则和禁用的时间表达式，可以确保用户输入的时间值满足应用程序的特定要求。

行为

- 在以下@@ 条件下可见：确定时间组件可见性的表达式。
- i@@ f 禁用：决定是否应禁用时间组件的表达式。

验证

- 必需：用于确保用户在提交表单之前必须输入时间值的切换开关。
- 自定义：允许您使用 JavaScript 表达式创建自定义验证规则。

  自定义验证消息：不满足自定义验证时要显示的消息。

例如：

```
{{ui.timeInput.value === "09:00 AM" || ui.timeInput.value === "09:30 AM"}}
```

此表达式检查输入的时间是上午 9:00 还是上午 9:30。如果条件为真，则验证将失败。

您还可以提供一条自定义验证消息，以便在不满足验证时显示：

```
Validation not met. The time must be 9:00 AM or 9:30 AM.
```

表达式和示例

时间组件提供以下表达式字段：

- `{{ui.timeID.value}}`：返回用户以 HH: MM AA 格式输入的时间值。

示例：时间值

- `{{ui.timeID.value}}`：返回用户以格式输入的时间值`HH:MM AA`。

示例：时间比较

- `{{ui.timeInput.value > "10:00 AM"}}`：检查时间值是否大于上午 10:00。
- `{{ui.timeInput.value < "05:00 pM"}}`：检查时间值是否小于 05:00 PM。

日期范围

日期范围组件允许用户选择和输入日期范围。通过配置日期范围组件的各种属性，您可以创建满足应用程序特定要求的日期范围输入字段，例如限制可选日期范围、禁用某些日期以及控制组件的可见性和交互性。

日期范围属性

日期范围组件与其他组件共享多个共同属性，例如`NameSource`、和`Validation`。有关这些属性的更多信息，请参阅常用组件属性。

除了常用属性外，日期范围组件还具有以下特定属性：

格式

- MM/DD/YYYY：显示日期范围的格式。

开始日期

- YYYY-MM-DD：可以选择作为范围起点的最小日期。

> **Note**
> 此值必须与的格式匹配YYYY-MM-DD。

结束日期

- YYYY-MM-DD：可以选择作为范围终点的最大日期。

> **Note**
> 此值必须与的格式匹配YYYY-MM-DD。

Placeholder

- 日历设置：日期范围字段为空时显示的占位符文本。

最小日期

- YYYY-MM-DD：可以选择的最小日期。

> **Note**
> 此值必须与的格式匹配YYYY-MM-DD。

最大日期

- YYYY-MM-DD：可以选择的最大日期。

> **Note**
> 此值必须与的格式匹配YYYY-MM-DD。

日历类型

- 1 个月：要显示的日历用户界面的类型。例如，单月。
- 2 个月：要显示的日历用户界面的类型。例如，两个月。

已选择必修日期

- 0：在日期范围内必须选择的必填天数。

禁用日期

- 来源：应禁用的日期（例如 “无”、“表达式”、“实体” 或 “自动”）的数据源。
- 禁用日期：用于确定应禁用哪些日期的表达式，例如`{{currentRow.column}}`。

验证

“验证” 部分允许您为日期范围输入定义其他规则和约束。

表达式和示例

日期范围组件提供以下表达式字段：

- `{{ui.dateRangeID.startDate}}`：以格式返回所选范围的开始日期`YYYY-MM-DD`。
- `{{ui.dateRangeID.endDate}}`：以格式返回所选范围的结束日期`YYYY-MM-DD`。

示例：计算日期差

- `{(new Date(ui.dateRangeID.endDate) - new Date(ui.dateRangeID.startDate)) / (1000 * 60 * 60 * 24)}}`计算起始日期和结束日期之间的天数。

示例：基于日期范围的条件可见性

- `{{new Date(ui.dateRangeID.startDate) < new Date("2023-01-01") || new Date(ui.dateRangeID.endDate) > new Date("2023-12-31")}}`检查所选日期范围是否在 2023 年之外。

示例：基于当前行数据的禁用日期

- `{{currentRow.isHoliday}}`禁用当前行中 “isHoliday” 列为真的日期。
- `{{new Date(currentRow.dateColumn) < new Date("2023-01-01")}}`根据当前行中的 “日期列” 禁用 2023 年 1 月 1 日之前的日期。
- `{{new Date(currentRow.dateColumn).getDay() === 0 || new Date(currentRow.dateColumn).getDay() === 6}}`根据当前行中的 “日期列” 禁用周末。

自定义验证

- `{{new Date(ui.dateRangeID.startDate) > new Date(ui.dateRangeID.endDate)}}`检查开始日期是否晚于结束日期，这将使自定义验证失败。

媒体组件

应用程序工作室提供了多个组件，用于在应用程序中嵌入和显示各种媒体类型。

iFrame 嵌入

iFrame 嵌入组件允许您使用 iFrame 在应用程序中嵌入外部 Web 内容或应用程序。

iFrame 嵌入属性

URL

> **Note**
>
> 应用程序的内容安全设置中必须允许使用此组件中显示的媒体来源。有关更多信息，请参阅 查看或更新应用程序的内容安全设置。

要嵌入的外部内容或应用程序的 URL。

布局

- 宽度：iFrame 的宽度，指定为百分比 (%) 或固定像素值（例如 300 像素）。
- 高度：iFrame 的高度，指定为百分比 (%) 或固定像素值。

S3 上传

S 3 上传组件允许用户将文件上传到 Amazon S3 存储桶。通过配置 S3 上传组件，您可以让用户轻松地将文件上传到应用程序的 Amazon S3 存储，然后在应用程序的逻辑和用户界面中利用上传的文件信息。

> **Note**
>
> 请务必确保具备必要的权限和 Amazon S3 存储桶配置，以支持应用程序的文件上传和存储要求。

S3 上传属性

S3 配置

- 连接器：选择用于文件上传的预配置的 Amazon S3 连接器。
- 存储桶：用于上传文件的 Amazon S3 存储桶。
- 文件夹：Amazon S3 存储桶中用于存储文件的文件夹。
- 文件名：上传文件的命名惯例。

文件上传配置

- 标签：文件上传区域上方显示的标签或说明。
- 描述：有关文件上传的其他说明或信息。
- 文件类型：允许上传的文件类型。例如：图像、文档或视频。
- 大小：可以上传的单个文件的最大大小。
- 按钮标签：文件选择按钮上显示的文本。
- 按钮样式：文件选择按钮的样式。例如，勾勒或填充。
- 按钮大小：文件选择按钮的大小。

验证

- 最大文件数：一次可以上传的最大文件数。
- 最大文件大小：每个文件允许的最大大小。

触发

- 成功时：文件上传成功时要触发的操作。
- 失败时：文件上传失败时要触发的操作。

S3 上传表达式字段

S 3 上传组件提供以下表达式字段：

- `{{ui.s3uploadID.files}}`：返回已上传文件的数组。
- `{{ui.s3uploadID.files[0]?.size}}`：返回指定索引处文件的大小。
- `{{ui.s3uploadID.files[0]?.type}}`：返回指定索引处的文件类型。
- `{{ui.s3uploadID.files[0]?.nameOnly}}`: 返回指定索引处的文件名，不带扩展名后缀。
- `{{ui.s3uploadID.files[0]?.nameWithExtension}}`: 返回文件名及其扩展名后缀位于指定索引处。

表达式和示例

示例：访问上传的文件

- `{{ui.s3uploadID.files.length}}`：返回已上传的文件数。
- `{{ui.s3uploadID.files.map(f => f.name).join(', ')}}`：返回以逗号分隔的已上传文件名列表。
- `{{ui.s3uploadID.files.filter(f => f.type.startsWith('image/'))}}`：返回仅包含已上传图像文件的数组。

示例：验证文件上传

- `{{ui.s3uploadID.files.some(f => f.size > 5 * 1024 * 1024)}}`：检查上传的文件大小是否超过 5 MB。
- `{{ui.s3uploadID.files.every(f => f.type === 'image/png')}}`：检查所有上传的文件是否都是 PNG 图片。
- `{{ui.s3uploadID.files.length > 3}}`：检查上传的文件是否超过 3 个。

示例：触发动作

- `{{ui.s3uploadID.files.length > 0 ? 'Upload Successful' : 'No files uploaded'}}`：如果至少上传了一个文件，则显示成功消息。
- `{{ui.s3uploadID.files.some(f => f.type.startsWith('video/')) ? triggerVideoProcessing() : null}}`：如果已上传任何视频文件，则触发视频处理自动化。
- `{{ui.s3uploadID.files.map(f => f.url)}}`：检索已上传 URLs 的文件，这些文件可用于显示或进一步处理这些文件。

这些表达式允许您访问上传的文件、验证文件上传以及根据文件上传结果触发操作。通过利用这些表达式，您可以在应用程序的文件上传功能中创建更加动态和智能的行为。

> **Note**
>
> *s3uploadID*替换为您的 S3 上传组件的 ID。

PDF 查看器组件

PDF 查看器组件允许用户在您的应用程序中查看 PDF 文档并与之交互。App Studio 支持这些不同的 PDF 源输入类型，PDF 查看器组件让您可以灵活地将 PDF 文档集成到应用程序中，无论是来自静态 URL、内联数据 URI 还是动态生成的内容。

PDF 查看器属性

来源

> **Note**
>
> 应用程序的内容安全设置中必须允许使用此组件中显示的媒体来源。有关更多信息，请参阅 查看或更新应用程序的内容安全设置。

PDF 文档的来源，可以是表达式、实体、URL 或自动化。

Expression

使用表达式动态生成 PDF 源。

实体

将 PDF 查看器组件连接到包含 PDF 文档的数据实体。

URL

指定 PDF 文档的网址。

URL

您可以输入指向要显示的 PDF 文档的 URL。这可以是公共网址，也可以是您自己的应用程序中的网址。

示例：`https://example.com/document.pdf`

数据 URI

数据 URI 是一种在应用程序中嵌入小型数据文件（如图像或 PDFs）的紧凑方法。PDF 文档被编码为 base64 字符串，并直接包含在组件的配置中。

Blob 或 ArrayBuffer

您还可以以 Blob 或 ArrayBuffer 对象的形式提供 PDF 文档，这样您就可以在应用程序中动态生成或检索 PDF 数据。

自动化

将 PDF 查看器组件连接到提供 PDF 文档的自动化系统。

操作

- 下载：添加允许用户下载 PDF 文档的按钮或链接。

布局

- 宽度：PDF 查看器的宽度，指定为百分比 (%) 或固定像素值（例如 600 像素）。
- 高度：PDF 查看器的高度，指定为固定像素值。

图像查看器

图像查看器组件允许用户查看应用程序中的图像文件并与之交互。

图像查看器属性

来源

> **Note**
>
> 应用程序的内容安全设置中必须允许使用此组件中显示的媒体来源。有关更多信息，请参阅 查看或更新应用程序的内容安全设置。

- 实体：将图像查看器组件连接到包含图像文件的数据实体。
- URL：指定图像文件的 URL。
- 表达式：使用表达式动态生成图像源。
- 自动化：将图像查看器组件连接到提供图像文件的自动化。

替代文本

图片的替代文字描述，用于无障碍访问目的。

布局

- 图像拟合：确定应如何调整图像大小并在组件中显示图像。例如，`Contain`、`Cover` 或 `Fill`。
- 宽度：图像查看器组件的宽度，指定为百分比 (%) 或固定像素值（例如 300px）。
- 高度：图像查看器组件的高度，指定为固定像素值。
- 背景：允许您为图像查看器组件设置背景颜色或图像。

自动化和操作：定义应用程序的业务逻辑

自动化是您定义应用程序业务逻辑的方式。自动化的主要组成部分是：启动自动化的触发器、一个或多个操作的序列、用于向自动化传递数据的输入参数以及输出。

主题

- 自动化概念
- 创建、编辑和删除自动化
- 添加、编辑和删除自动化操作

- 自动化操作参考

自动化概念

以下是在 App Studio 中使用自动化来定义和配置应用程序的业务逻辑时需要了解的一些概念和术语。

自动化

自动化是您定义应用程序业务逻辑的方式。自动化的主要组成部分是：启动自动化的触发器、一个或多个操作的序列、用于向自动化传递数据的输入参数以及输出。

操作

自动化操作，通常称为操作，是构成自动化的单个逻辑步骤。每个操作都会执行特定的任务，无论是发送电子邮件、创建数据记录、调用 Lambda 函数还是调用。 APIs操作可通过操作库添加到自动化中，并且可以分组为条件语句或循环。

自动化输入参数

自动化输入参数是动态输入值，您可以将其从组件传递给自动化，以使其灵活且可重复使用。将参数视为自动化的变量，您可以定义参数并在需要时提供不同的值，而不是将值硬编码到自动化中。参数允许您在每次运行时使用具有不同输入的相同自动化。

模拟输出

某些操作使用连接器与外部资源或服务进行交互。使用预览环境时，应用程序不与外部服务交互。要测试在预览环境中使用连接器的操作，可以使用模拟输出来模拟连接器的行为和输出。模拟输出是使用配置的 JavaScript，结果存储在操作的结果中，就像连接器的响应存储在已发布的应用程序中一样。

通过使用模拟，您可以使用预览环境测试各种场景及其对自动化操作的影响，例如模拟不同的结果值、错误场景、边缘情况或不愉快的路径，而无需通过连接器调用外部服务。

自动化输出

自动化输出用于将值从一个自动化传递到应用程序的其他资源，例如组件或其他自动化。自动化输出配置为表达式，表达式可以返回静态值或根据自动化参数和操作计算出的动态值。默认情况下，自动化不返回任何数据，包括自动化中的操作结果。

以下是如何使用自动化输出的几个示例：

- 您可以将自动化输出配置为返回数组，然后传递该数组以填充数据组件。

- 您可以使用自动化来计算一个值，并将该值传递给其他多个自动化，以此作为集中和重复使用业务逻辑的一种方式。

触发

触发器决定何时以及在什么条件下运行自动化。触发器的一些`On click`示例包括按钮和`On select`文本输入。组件的类型决定了该组件的可用触发器列表。触发器被添加到组件中，并在应用程序工作室中进行配置。

创建、编辑和删除自动化

目录

- 创建自动化
- 查看或编辑自动化属性
- 删除自动化

创建自动化

使用以下步骤在 App Studio 应用程序中创建自动化。创建自动化后，必须通过编辑其属性并向其添加操作来对其进行配置。

创建自动化

1. 如有必要，请导航到应用程序的应用程序工作室。
2. 选择自动化选项卡。
3. 如果您没有自动化，请在画布中选择 + 添加自动化。否则，在左侧的 “自动化” 菜单中，选择 + 添加。
4. 将创建一个新的自动化，您可以开始编辑其属性或添加和配置操作来定义应用程序的业务逻辑。

查看或编辑自动化属性

使用以下步骤查看或编辑自动化属性。

查看或编辑自动化属性

1. 如有必要，请导航到应用程序的应用程序工作室。
2. 选择自动化选项卡。

3. 在左侧的 “自动化” 菜单中，选择要查看或编辑其属性的自动化以打开右侧的 “属性” 菜单。
4. 在 “属性” 菜单中，您可以查看以下属性：
   - 自动化标识符：自动化的唯一名称。要对其进行编辑，请在文本字段中输入新的标识符。
   - 自动化参数：自动化参数用于将动态值从应用程序的界面传递给自动化和数据操作。要添加参数，请选择 + 添加。选择铅笔图标可更改参数的名称、描述或类型。要移除参数，请选择垃圾桶图标。

     > **Tip**
     > 您也可以直接从画布添加自动化参数。

   - 自动化输出：自动化输出用于配置哪些自动化数据可以在其他自动化或组件中引用。默认情况下，自动化不会创建输出。要添加自动化输出，请选择 + 添加。要删除输出，请选择垃圾桶图标。
5. 您可以通过添加和配置操作来定义自动化的用途。有关操作的更多信息，请参阅添加、编辑和删除自动化操作和自动化操作参考。

删除自动化

使用以下步骤删除 App Studio 应用程序中的自动化。

删除自动化

1. 如有必要，请导航到应用程序的应用程序工作室。
2. 选择自动化选项卡。
3. 在左侧的 “自动化” 菜单中，选择要删除的自动化的省略号菜单，然后选择 “删除”。或者，您可以从自动化右侧的 “属性” 菜单中选择垃圾桶图标。
4. 在确认对话框中，选择删除。

添加、编辑和删除自动化操作

自动化操作，通常称为操作，是构成自动化的单个逻辑步骤。每个操作都会执行特定的任务，无论是发送电子邮件、创建数据记录、调用 Lambda 函数还是调用。 APIs操作可通过操作库添加到自动化中，并且可以分组为条件语句或循环。

目录

- 添加自动化操作
- 查看和编辑自动化操作属性
- 删除自动化操作

添加自动化操作

使用以下步骤向 App Studio 应用程序中的自动化添加操作。

添加自动化操作

1. 如有必要，请导航到应用程序的应用程序工作室。
2. 选择自动化选项卡。
3. 在左侧的 “自动化” 菜单中，选择要向其添加操作的自动化。
4. 在右侧的 “操作” 菜单中，选择要添加的动作，或者将该动作拖放到画布中。创建动作后，您可以选择动作来配置动作属性以定义动作的功能。有关操作属性及其配置的更多信息，请参阅自动化操作参考。

查看和编辑自动化操作属性

使用以下步骤在 App Studio 应用程序中查看或编辑自动化操作的属性。

查看或编辑自动化操作属性

1. 如有必要，请导航到应用程序的应用程序工作室。
2. 选择自动化选项卡。
3. 在左侧的 “自动化” 菜单中，选择要查看或编辑属性的操作。或者，您可以在查看包含该操作的自动化时在画布中选择操作。
4. 可以在右侧的 “属性” 菜单中查看或编辑操作属性。每种动作类型的动作属性都不同。有关操作属性及其配置的更多信息，请参阅自动化操作参考。

删除自动化操作

使用以下步骤从 App Studio 应用程序的自动化中删除操作。

删除自动化操作

1. 如有必要，请导航到应用程序的应用程序工作室。

2. 选择自动化选项卡。
3. 在左侧的 “自动化” 菜单中，选择包含要删除的操作的自动化。
4. 在画布中，选择要删除的动作中的垃圾桶图标，然后选择删除。

自动化操作参考

以下是 App Studio 中使用的自动化操作的参考文档。

自动化操作，通常称为操作，是构成自动化的单个逻辑步骤。每个操作都会执行特定的任务，无论是发送电子邮件、创建数据记录、调用 Lambda 函数还是调用。 APIs操作通过操作库添加到自动化中，并且可以分组为条件语句或循环。

有关创建和配置自动化及其操作的信息，请参阅中的主题。自动化和操作：定义应用程序的业务逻辑

调用 API

调用 HTTP REST API 请求。构建者可以使用此操作将请求从 App Studio 发送到其他系统或服务 APIs。例如，您可以使用它连接到第三方系统或本土应用程序以访问关键业务数据，或者调用 App Studio 专用 App Studio 操作无法调用的 API 端点。

有关 REST 的更多信息 APIs，请参阅什么是 RESTful API？ 。

属性

Connector

用于此操作发出的 API 请求的连接器。连接器下拉列表仅包含以下类型的连接器：`API Connector`和`OpenAPI Connector`。根据连接器的配置方式，它可能包含重要信息，例如凭据和默认标头或查询参数。

有关 API 连接器的更多信息，包括使用`API Connector`和之间的比较`OpenAPI Connector`，请参阅Connect 连接到第三方服务。

API 请求配置属性

从属性面板中选择配置 API 请求以打开请求配置对话框。如果选择了 API 连接器，则对话框将包含连接器信息。

方法：API 调用的方法。可能值如下所示：

- `DELETE`：删除指定资源。

- GET：检索信息或数据。
- HEAD：仅检索不带正文的响应标头。
- POST：提交待处理的数据。
- PUSH：提交待处理的数据。
- PATCH：部分更新指定资源。

路径：资源的相对路径。

标头：与 API 请求一起发送的键值对形式的任何标头。如果选择了连接器，则其配置的标头将自动添加且无法删除。无法编辑配置的标题，但您可以通过添加另一个同名的标题来覆盖它们。

查询参数：任何要与 API 请求一起发送的键值对形式的查询参数。如果选择了连接器，则会自动添加其配置的查询参数，且无法对其进行编辑或删除。

正文：与 API 请求一起发送的信息，采用 JSON 格式。没有受理GET请求的机构。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作results的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

调用 AWS

从服务调用操作。 AWS 这是调用 AWS 服务或操作的一般操作，如果没有针对所需 AWS 服务或操作的专用操作，则应使用此操作。

属性

服务

包含要运行的操作的 AWS 服务。

操作

要运行的操作。

Connector

用于此操作运行的操作的连接器。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

配置

运行指定操作时的 JSON 输入。有关配置 AWS 操作输入的更多信息，请参阅适用于 JavaScript 的 AWS SDK。

调用 Lambda

调用现有的 Lambda 函数。

属性

Connector

用于此操作运行的 Lambda 函数的连接器。配置的连接器应使用访问 Lambda 函数的适当凭证以及其他配置信息（例如包含 Lambda 函数的 AWS 区域）进行设置。有关为 Lambda 配置连接器的更多信息，请参阅。步骤 3：创建 Lambda 连接器

函数名称

要运行的 Lambda 函数的名称。请注意，这是函数名称，而不是函数 ARN（Amazon 资源名称）。

函数事件

要作为事件负载传递给 Lambda 函数的键值对。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

循环

重复运行嵌套操作以遍历项目列表，一次只能遍历一个项目。例如，将该创建记录操作添加到循环操作中以创建多条记录。

循环操作可以嵌套在其他循环或条件动作中。循环操作是按顺序运行的，而不是并行运行的。循环中每个操作的结果只能访问同一循环迭代中的后续操作。不能在循环之外或循环的不同迭代中访问它们。

属性

来源

要迭代的项目列表，一次只能遍历一项。源可以是先前操作的结果，也可以是您可以使用 JavaScript 表达式提供的字符串、数字或对象的静态列表。

示例

以下列表包含源输入的示例。

- 先前操作的结果：`{{results.actionName.data}}`
- 数字清单：`{{[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]}}`
- 字符串列表：`{{["apple", "banana", "orange", "grape", "kiwi"]}}`
- 计算值：`{{params.actionName.split("\n")}}`

当前项目名称

可用于引用当前正在迭代的项目的变量名称。当前项目名称是可配置的，因此您可以嵌套两个或多个循环并访问每个循环中的变量。例如，如果您要使用两个环路循环遍历国家和城市，则可以配置并引用`currentCountry`和`currentCity`。

状况

根据自动化运行时评估的一个或多个指定逻辑条件的结果运行操作。条件动作由以下组件组成：

- 一个条件字段，用于提供计算结果为`true`或`false`的 JavaScript 表达式。
- 一个真正的分支，其中包含条件计算结果为时运行的操作。`true`
- 一个 false 分支，其中包含条件计算结果为时运行的操作。`false`

通过将动作拖动到条件动作中，为真分支和假分支添加动作。

属性

状况

操作运行时要评估的 JavaScript 表达式。

创建记录

在现有 App Studio 实体中创建一条记录。

属性

实体

要在其中创建记录的实体。选择实体后，必须向该实体的字段中添加值才能创建记录。字段的类型以及字段是必填字段还是可选字段都是在实体中定义的。

更新记录

更新 App Studio 实体中的现有记录。

属性

实体

包含要更新的记录的实体。

Conditions

定义操作更新哪些记录的标准。您可以对条件进行分组以创建一条逻辑语句。您可以将组或条件与AND或OR语句合并。

字段

条件指定的记录中要更新的字段。

值

指定字段中要更新的值。

删除记录

从 App Studio 实体中删除一条记录。

属性

实体

包含要删除的记录的实体。

Conditions

定义操作删除哪些记录的标准。您可以对条件进行分组以创建一条逻辑语句。您可以将组或条件与AND或OR语句合并。

调用数据操作

使用可选参数运行数据操作。

属性

数据操作

该操作要运行的数据操作。

参数

数据操作要使用的数据操作参数。数据操作参数用于发送用作数据操作输入的值。配置自动化操作时可以添加数据操作参数，但必须在 “数据” 选项卡中进行编辑。

高级设置

该Invoke data action操作包含以下高级设置：

- 页面大小：每次查询中要提取的最大记录数。默认值为 500，最大值为 3000。
- 分页令牌：用于从查询中获取其他记录的标记。例如，如果设置Page size为 500，但有超过 500 条记录，则将分页令牌传递给后续查询将获取接下来的 500 条记录。如果不再存在记录或页面，则标记将处于未定义状态。

亚马逊 S3：放置对象

使用Amazon S3 PutObject操作将由密钥（文件路径）标识的对象添加到指定的 Amazon S3 存储桶。

属性

Connector

用于此操作运行的操作的连接器。应使用运行操作的相应凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）来设置已配置的连接器。

配置

要在PutObject命令中使用的必需选项。这些方法如下所示：

> **Note**
>
> 有关Amazon S3 PutObject操作的更多信息，请参阅PutObject《亚马逊简单存储服务 API 参考》。

- 存储桶：用于放置对象的 Amazon S3 存储桶的名称。
- 密钥：要放入 Amazon S3 存储桶的对象的唯一名称。
- 正文：要放入 Amazon S3 存储桶的数据元的内容。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作results的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

亚马逊 S3：删除对象

使用Amazon S3 DeleteObject操作从指定的 Amazon S3 存储桶中删除由密钥（文件路径）标识的对象。

属性

Connector

用于此操作运行的操作的连接器。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

配置

要在DeleteObject命令中使用的必需选项。这些方法如下所示：

> **Note**
>
> 有关`Amazon S3 DeleteObject`操作的更多信息，请参阅DeleteObject《亚马逊简单存储服务 API 参考》。

- 存储桶：要从中删除对象的 Amazon S3 存储桶的名称。
- 密钥：要从 Amazon S3 存储桶中删除的对象的唯一名称。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

亚马逊 S3：获取对象

使用`Amazon S3 GetObject`操作从指定的 Amazon S3 存储桶中检索由密钥（文件路径）标识的对象。

属性

Connector

用于此操作运行的操作的连接器。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

配置

要在`GetObject`命令中使用的必需选项。这些方法如下所示：

> **Note**
>
> 有关`Amazon S3 GetObject`操作的更多信息，请参阅GetObject《亚马逊简单存储服务 API 参考》。

- 存储桶：要从中检索对象的 Amazon S3 存储桶的名称。
- 密钥：要从 Amazon S3 存储桶中检索的对象的唯一名称。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

亚马逊 S3：列出对象

使用`Amazon S3 ListObjects`操作列出指定 Amazon S3 存储桶中的对象。

属性

Connector

用于此操作运行的操作的连接器。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

配置

要在`ListObjects`命令中使用的必需选项。这些方法如下所示：

> **Note**
>
> 有关`Amazon S3 ListObjects`操作的更多信息，请参阅ListObjects《亚马逊简单存储服务 API 参考》。

- 存储桶：用于列出对象的 Amazon S3 存储桶的名称。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

Amazon Textract：分析文档

使用`Amazon Textract AnalyzeDocument`操作来分析输入文档中检测到的项目之间的关系。

属性

Connector

用于此操作运行的操作的连接器。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

配置

要在`AnalyzeDocument`命令中使用的请求内容。这些方法如下所示：

> **Note**
>
> 有关`Amazon Textract AnalyzeDocument`操作的更多信息，请参阅 AnalyzeDocumentAmazon Textract 开发者指南。

- 文档/S3Object/Bucket：亚马逊 S3 存储桶的名称。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。
- 文档/S3Object/名称：输入文档的文件名。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。
- 文档/S3Object/版本：如果 Amazon S3 存储桶启用了版本控制，则可以指定对象的版本。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。
- FeatureTypes：要执行的分析类型列表。有效值为：`TABLES`、`FORMS`、`QUERIES`、`SIGNATURES` 和 `LAYOUT`。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

Amazon Textract：分析支出

使用`Amazon Textract AnalyzeExpense`操作来分析输入文档，了解文本之间与财务相关的关系。

属性

Connector

用于此操作运行的操作的连接器。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

配置

要在`AnalyzeExpense`命令中使用的请求内容。这些方法如下所示：

> **Note**
>
> 有关`Amazon Textract AnalyzeExpense`操作的更多信息，请参阅 AnalyzeExpenseAmazon Textract 开发者指南。

- 文档/S3Object/Bucket：亚马逊 S3 存储桶的名称。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。
- 文档/S3Object/名称：输入文档的文件名。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。
- 文档/S3Object/版本：如果 Amazon S3 存储桶启用了版本控制，则可以指定对象的版本。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

Amazon Textract：分析身份证

使用该`Amazon Textract AnalyzeID`操作来分析身份证件以获取相关信息。

属性

Connector

用于此操作运行的操作的连接器。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

配置

要在`AnalyzeID`命令中使用的请求内容。这些方法如下所示：

> **Note**
>
> 有关该`Amazon Textract AnalyzeID`操作的更多信息，请参阅亚马逊 Textrac t 开发者指南中的 analyzeID。

- 文档/S3Object/Bucket：亚马逊 S3 存储桶的名称。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。
- 文档/S3Object/名称：输入文档的文件名。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。
- 文档/S3Object/版本：如果 Amazon S3 存储桶启用了版本控制，则可以指定对象的版本。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

Amazon Textract：检测文档文本

使用该`Amazon Textract DetectDocumentText`操作来检测输入文档中的文本行和构成一行文本的单词。

属性

Connector

用于此操作运行的操作的连接器。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

配置

要在`DetectDocumentText`命令中使用的请求内容。这些方法如下所示：

> **Note**
>
> 有关`Amazon Textract DetectDocumentText`操作的更多信息，请参阅 DetectDocumentTextAmazon Textract 开发者指南。

- 文档/S3Object/Bucket：亚马逊 S3 存储桶的名称。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。
- 文档/S3Object/名称：输入文档的文件名。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。
- 文档/S3Object/版本：如果 Amazon S3 存储桶启用了版本控制，则可以指定对象的版本。如果使用 S3 上传组件将文件传递给操作，则此参数可以留空。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

亚马逊 Bedrock：GenAI Prompt

使用 Amazon Bedrock InvokeModel 操作使用操作属性中提供的提示和推理参数运行推理。该操作可以生成文本、图像和嵌入内容。

属性

Connector

用于此操作运行的操作的连接器。要成功使用此操作，必须将连接器配置为 Amazon Bedrock Runtime 作为服务。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

模型

Amazon Bedrock 用于处理请求的基础模型。有关亚马逊 Bedrock 中模型的更多信息，请参阅亚马逊 Bedrock 用户指南中的亚马逊 Bedrock 基础模型信息。

输入类型

输入的输入类型将发送到 Amazon Bedrock 模型。可能的值为 “文本”、“文档” 和 “图像”。如果输入类型不可选，则配置的模型可能不支持该类型。

用户提示

要发送给 Amazon Bedrock 模型的提示，以进行处理以生成响应。您可以输入静态文本，也可以传入来自应用程序其他部分的输入，例如使用参数的组件、自动化中的先前操作或其他自动化。以下示例说明如何传入来自组件或先前操作的值：

- 要使用参数从组件传递值，请执行以下操作：`{{params.paramName}}`
- 要传递先前操作中的值，请执行以下操作：`{{results.actionName}}`

系统提示符（克劳德模型）

Amazon Bedrock 模型在处理请求时要使用的系统提示。系统提示符用于为 Claude 模型提供上下文、说明或指南。

请求设置

配置各种请求设置和模型推理参数。您可以配置以下设置：

- 温度：Amazon Bedrock 模型在处理请求时使用的温度。温度决定了 Bedrock 模型输出的随机性或创造力。温度越高，反应的创造性就越强，分析性就越低。可能的值是`[0-10]`。
- 最大代币数量：限制 Amazon Bedrock 模型的输出长度。
- TopP：在 nucleus 采样中，模型按概率递减顺序计算每个后续代币在所有期权上的累积分布，并在其达到 TopP 指定的特定概率时将其切断。你应该改变温度或 TopP，但不能两者兼而有之

- 停止序列：导致模型停止处理请求和生成输出的序列。

有关更多信息，请参阅 Amazon Bedrock 用户指南中的基础模型的推理请求参数和响应字段。

停止序列

输入 Amazon Bedrock Guardrail ID 和版本。护栏用于根据您的用例和负责任的人工智能政策实施保障措施。有关更多信息，请参阅《亚马逊 Bedrock 用户指南》中的使用 Amazon Bedrock Guardrails 阻止模型中的有害内容。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

Amazon Bedrock：调用模型

使用 Amazon Bedrock InvokeModel 操作使用请求正文中提供的提示和推理参数运行推理。您可以使用模型推理来生成文本、图像和嵌入内容。

属性

Connector

用于此操作运行的操作的连接器。要成功使用此操作，必须将连接器配置为 Amazon Bedrock Runtime 作为服务。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

配置

要在`InvokeModel`命令中使用的请求内容。

> **Note**
>
> 有关`Amazon Bedrock InvokeModel`操作的更多信息，包括示例命令，请参阅 Amazon Bedrock API 参考InvokeModel中的。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

JavaScript

运行自定义 JavaScript 函数以返回指定值。

> **Important**
>
> App Studio 不支持使用第三方库或自定义 JavaScript 库。

属性

源代码

操作要运行的 JavaScript 代码片段。

> **Tip**
>
> 您可以通过执行以下步骤使用 AI 来帮助您生成 JavaScript ：
>
> 1. 选择展开图标以打开展开的 JavaScript 编辑器。
> 2. （可选）：启用修改代码开关以修改任何现有代码 JavaScript。否则，人工智能将取代任何现有 JavaScript的。
> 3. 在 “生成” 中 JavaScript，描述你要用什么来做 JavaScript，例如：**`Add two numbers`**。
> 4. 选择发送图标以生成您的 JavaScript.

调用自动化

运行指定的自动化。

属性

调用自动化

要由操作运行的自动化。

发送电子邮件

使用`Amazon SES SendEmail`操作发送电子邮件。

属性

Connector

用于此操作运行的操作的连接器。配置的连接器应使用运行操作的正确凭据以及其他配置信息（例如包含操作中引用的任何资源的 AWS 区域）进行设置。

配置

要在`SendEmail`命令中使用的请求内容。这些方法如下所示：

> **Note**
>
> 有关该`Amazon SES SendEmail`操作的更多信息，请参阅SendEmail《亚马逊简单电子邮件服务 API 参考》。

模拟输出

在预览环境中，操作不会与外部服务或资源交互。Mocked 输出字段用于提供一个 JSON 表达式，该表达式可模拟连接器在预览环境中的行为，以用于测试目的。此代码段存储在操作`results`的地图中，就像连接器响应在实时环境中发布的应用程序一样。

使用此字段，您可以测试各种场景及其对自动化中其他操作的影响，例如模拟不同的结果值、错误场景、边缘案例或不愉快的路径，而无需通过连接器与外部服务进行通信。

实体和数据操作：配置应用程序的数据模型

实体是 App Studio 中的数据表。实体直接与数据源中的表交互。实体包括用于描述其中的数据的字段、用于查找和返回数据的查询以及用于将实体字段连接到数据源列的映射。

主题

- 设计数据模型时的最佳实践
- 在 App Studio 应用程序中创建实体
- 在 App Studio 应用程序中配置或编辑实体
- 删除实体
- AWS App Studio 中的管理数据实体

设计数据模型时的最佳实践

使用以下最佳实践在中创建强大、可扩展且安全的关系数据模型， AWS 以便在 App Studio 应用程序中使用，该模型既要满足应用程序的要求，又能确保数据基础架构的长期可靠性和性能。

- 选择正确 AWS 的数据服务：根据您的要求，选择合适 AWS 的数据服务。例如，对于在线事务处理 (OLTP) 应用程序，您可以考虑使用诸如 Amazon Aurora 之类的数据库 (DB)，它是一种云原生、关系型和完全托管的数据库服务，支持 MySQL 和 PostgreSQL 等各种数据库引擎。有关 App Studio 支持的 Aurora 版本的完整列表，请参阅连接亚马逊 Aurora。另一方面，对于在线分析处理 (OLAP) 用例，可以考虑使用 Amazon Redshift，这是一个云数据仓库，可让您对非常大的数据集运行复杂查询。这些查询通常需要时间（数秒）才能完成，这使得 Amazon Redshift 不太适合需要低延迟数据访问的 OLTP 应用程序。
- 为可扩展性而设计：在规划数据模型时考虑未来的增长和可扩展性。在选择适当的数据服务以及数据库实例类型和配置（例如预配置容量）时，请考虑预期数据量、访问模式和性能要求等因素。
  - 有关使用 Aurora 无服务器进行扩展的更多信息，请参阅 Aurora Server less V2 的性能和扩展。
- 标准化数据：遵循数据库标准化原则，最大限度地减少数据冗余并提高数据完整性。这包括创建相应的表、定义主键和外键以及在实体之间建立关系。在 App Studio 中，在查询来自一个实体的数据时，您可以通过在查询中指定`join`子句来从另一个实体检索相关数据。
- 实施适当的索引：确定最重要的查询和访问模式，并创建适当的索引以优化性能。
- 利用 AWS 数据服务功能：利用您选择 AWS 的数据服务提供的功能，例如自动备份、多可用区部署和自动软件更新。
- 保护您的数据：实施强大的安全措施，例如 IAM (AWS Identity and Access Management) 策略，创建对表和架构具有有限权限的数据库用户，以及在静态和传输中强制加密。
- 监控和优化性能：持续监控数据库的性能并根据需要进行调整，例如扩展资源、优化查询或调整数据库配置。
- 自动管理数据库：利用 AWS Aurora Autoscaling、Performance Insights for Aurora 和 AWS Database Migration Service 等服务来自动执行数据库管理任务并减少运营开销。

- 实施灾难恢复和备份策略：利用 Aurora 自动备份、恢复和跨区域副本配置等功能，确保您有一个明确定义的备份和 point-in-time恢复计划。
- 遵循 AWS 最佳实践和文档：随时 up-to-date了解所选数据服务的最新 AWS 最佳实践、指南和文档，以确保您的数据模型和实施与 AWS 建议保持一致。

有关每种 AWS 数据服务的更多详细指导，请参阅以下主题：

- 亚马逊 Aurora 的最佳实践
- 亚马逊 Aurora MySQL 的最佳实践
- 亚马逊 Redshift 查询性能调整
- 在 Amazon DynamoDB 中查询和扫描数据的最佳实践

在 App Studio 应用程序中创建实体

在 App Studio 应用程序中创建实体的方法有四种。以下列表包含每种方法及其优点，以及使用该方法创建和配置实体的说明的链接。

- 从现有数据源创建实体：根据现有数据源表自动创建实体及其字段，并将这些字段映射到数据源表的列。如果您有要在 App Studio 应用程序中使用的现有数据源，则最好使用此选项。
- 使用 App Studio 托管数据源创建实体：创建由 App Studio 为您管理的实体和一个 DynamoDB 表。当你更新实体时，DynamoDB 表会自动更新。使用此选项，您无需手动创建、管理或连接第三方数据源，也不必指定从实体字段到表列的映射。您的应用程序的所有数据建模和配置都在 App Studio 中完成。如果您不想管理自己的数据源和 DynamoDB 表，并且其功能足以满足您的应用程序的需求，则最好使用此选项。
- 创建空实体: 完全从头开始创建一个空实体。如果您没有任何由管理员创建的现有数据源或连接器，并且您想在不受外部数据源限制的情况下灵活设计应用程序的数据模型，则最好使用此选项。创建实体后，您可以将实体连接到数据源。
- 使用 AI 创建实体：根据指定的实体名称生成实体、字段、数据操作和示例数据。如果您对应用程序的数据模型有所了解，但需要帮助将其转换为实体，则最好使用此选项。

从现有数据源创建实体

使用数据源中的表自动创建实体及其字段，并将实体字段映射到表的列。如果您有要在 App Studio 应用程序中使用的现有数据源，则最好使用此选项。

1. 如有必要，请导航到您的应用程序。

2. 选择画布顶部的 “数据” 选项卡。
3. 如果您的应用程序中没有实体，请选择 + 创建实体。否则，在左侧的实体菜单中，选择 + 添加。
4. 选择 “使用现有数据源中的表”。
5. 在 C onnector 中，选择包含要用于创建实体的表格的连接器。
6. 在表格中，选择要用于创建实体的表。
7. 选中 “创建数据操作” 复选框以创建数据操作。
8. 选择 Create entity (创建实体)。您的实体现已创建，您可以在左侧的实体面板中看到它。
9. 按照中的步骤配置您的新实体在 App Studio 应用程序中配置或编辑实体。请注意，由于您的实体是使用现有数据源创建的，因此已经创建了一些属性或资源，例如字段、连接的数据源和字段映射。此外，如果您在创建过程中选中了创建数据操作复选框，则您的实体将包含数据操作。

使用 App Studio 托管数据源创建实体

创建由 App Studio 管理的托管实体和相应的 DynamoDB 表。虽然 DynamoDB 表存在于 AWS 关联账户中，但在 App Studio 应用程序中对该实体进行更改时，DynamoDB 表会自动更新。使用此选项，您无需手动创建、管理或连接第三方数据源，也不必指定从实体字段到表列的映射。如果您不想管理自己的数据源和 DynamoDB 表，并且其功能足以满足您的应用程序的需求，则最好使用此选项。有关托管实体的更多信息，请参阅AWS App Studio 中的管理数据实体。

您可以在多个应用程序中使用相同的托管实体。有关说明，请参阅从现有数据源创建实体。

1. 如有必要，请导航到您的应用程序。
2. 选择画布顶部的 “数据” 选项卡。
3. 如果您的应用程序中没有实体，请选择 + 创建实体。否则，在左侧的实体菜单中，选择 + 添加。
4. 选择创建 App Studio 管理实体。
5. 在实体名称中，为您的实体提供一个名称。
6. 在主键中，提供实体的主键的名称。主键是实体的唯一标识符，在创建实体后无法更改。
7. 在主键数据类型中，选择实体主键的数据类型。实体创建后无法更改数据类型。
8. 选择 Create entity (创建实体)。您的实体现已创建，您可以在左侧的实体面板中看到它。
9. 按照中的步骤配置您的新实体在 App Studio 应用程序中配置或编辑实体。请注意，由于您的实体是使用托管数据创建的，因此已经创建了一些属性或资源，例如主键字段和连接的数据源。

创建空实体

完全从头开始创建一个空实体。如果您没有任何由管理员创建的现有数据源或连接器，则最好使用此选项。创建空实体提供了灵活性，因为您可以在 App Studio 应用程序中设计实体，而不受外部数据源的限制。在设计应用程序的数据模型并相应地配置实体之后，您仍然可以将其连接到外部数据源。

1. 如有必要，请导航到您的应用程序。
2. 选择画布顶部的 “数据” 选项卡。
3. 如果您的应用程序中没有实体，请选择 + 创建实体。否则，在左侧的实体菜单中，选择 + 添加。
4. 选择 “创建实体”。
5. 选择 Create entity (创建实体)。您的实体现已创建，您可以在左侧的实体面板中看到它。
6. 按照中的步骤配置您的新实体在 App Studio 应用程序中配置或编辑实体。

使用 AI 创建实体

根据指定的实体名称生成实体、字段、数据操作和示例数据。如果您对应用程序的数据模型有所了解，但需要帮助将其转换为实体，则最好使用此选项。

1. 如有必要，请导航到您的应用程序。
2. 选择画布顶部的 “数据” 选项卡。
3. 如果您的应用程序中没有实体，请选择 + 创建实体。否则，在左侧的实体菜单中，选择 + 添加。
4. 选择 “使用 AI 创建实体”。
5. 在实体名称中，为您的实体提供一个名称。此名称用于生成实体的字段、数据操作和示例数据。
6. 选中 “创建数据操作” 复选框以创建数据操作。
7. 选择 “生成实体”。您的实体现已创建，您可以在左侧的实体面板中看到它。
8. 按照中的步骤配置您的新实体在 App Studio 应用程序中配置或编辑实体。请注意，由于您的实体是使用 AI 创建的，因此您的实体将已经包含生成的字段。此外，如果您在创建过程中选中了创建数据操作复选框，则您的实体将包含数据操作。

在 App Studio 应用程序中配置或编辑实体

使用以下主题在 App Studio 应用程序中配置实体。

主题

- 编辑实体名称
- 添加、编辑或删除实体字段
- 创建、编辑或删除数据操作
- 添加或删除示例数据
- 添加或编辑连接的数据源和地图字段

编辑实体名称

1. 如有必要，请导航到要编辑的实体。
2. 在 “配置” 选项卡的 “实体名称” 中，更新实体名称，然后在文本框之外选择以保存更改。

添加、编辑或删除实体字段

> **Tip**
> 您可以按 CTRL+Z 撤消最近对实体所做的更改。

1. 如有必要，请导航到要编辑的实体。
2. 在 “配置” 选项卡的 “字段” 中，您可以查看实体字段的表。实体字段包含以下列：
   - 显示名称：显示名称类似于表标题或表单域，可供应用程序用户查看。它可以包含空格和特殊字符，但在实体中必须是唯一的。
   - 系统名称：系统名称是代码中用于引用字段的唯一标识符。映射到亚马逊 Redshift 表中的列时，该列必须与亚马逊 Redshift 表的列名相匹配。
   - 数据类型：将存储在此字段中的数据类型，例如`Integer``Boolean`、或`String`。
3. 要添加字段：
   a. 要使用 AI 根据实体名称和连接的数据源生成字段，请选择生成更多字段。
   b. 要添加单个字段，请选择 + 添加字段。
4. 要编辑字段，请执行以下操作：
   a. 要编辑显示名称，请在显示名称文本框中输入所需的值。如果尚未编辑该字段的系统名称，则会将其更新为显示名称的新值。
   b. 要编辑系统名称，请在系统名称文本框中输入所需的值。

c. 要编辑数据类型，请选择数据类型下拉菜单，然后从列表中选择所需的类型。

d. 要编辑字段的属性，请选择该字段的齿轮图标。以下列表详细说明了字段的属性：

- 必填：如果您的数据源需要该字段，请启用此选项。
- 主键：如果字段映射到数据源中的主键，请启用此选项。
- 唯一：如果此字段的值必须是唯一的，请启用此选项。
- 使用数据源默认值：如果字段的值由数据源提供，例如使用自动增量或事件时间戳，则启用此选项。
- 数据类型选项：某些数据类型的字段可以配置数据类型选项，例如最小值或最大值。

5. 要删除字段，请选择要删除的字段的垃圾桶图标。

创建、编辑或删除数据操作

应用程序中使用数据操作对实体的数据执行操作，例如获取所有记录或按 ID 获取记录。数据操作可用于查找和返回与指定条件相匹配的数据，以便在表格或详细视图等组件中查看。

目录

- 创建数据操作
- 编辑或配置数据操作
- 数据操作条件运算符和示例
  - 数据库支持条件运算符
  - 数据操作条件示例
- 删除数据操作

创建数据操作

> **Tip**
> 您可以按 CTRL+Z 撤消最近对实体所做的更改。

1. 如有必要，请导航到要为其创建数据操作的实体。
2. 选择 “数据操作” 选项卡。
3. 创建数据操作的方法有两种：

- （推荐）要使用 AI 根据您的实体名称、字段和连接的数据源为您生成数据操作，请选择生成数据操作。将生成以下操作：

  1. `getAll`：检索实体的所有记录。当您需要显示记录列表或同时对多条记录执行操作时，此操作非常有用。
  2. `getByID`：根据实体的唯一标识符（ID 或主键）从其检索单个记录。当您需要显示或对特定记录执行操作时，此操作非常有用。
- 要添加单个数据操作，请选择 + 添加数据操作。

4. 要查看或配置新的数据操作，请参阅以下部分编辑或配置数据操作。

编辑或配置数据操作

1. 如有必要，请导航到要为其创建数据操作的实体。
2. 选择 “数据操作” 选项卡。
3. 在字段中，配置查询要返回的字段。默认情况下，实体中所有已配置的字段都处于选中状态。

   您也可以通过执行以下步骤将联接添加到数据操作中：

   1. 选择 + 添加加入以打开一个对话框。
   2. 在相关实体中，选择要与当前实体联接的实体。
   3. 在别名中，可以选择输入相关实体的临时别名。
   4. 在联接类型中，选择所需的联接类型。
   5. 通过从每个实体中选择字段来定义联接子句。
   6. 选择 “添加” 以创建联接。

   创建后，联接将显示在 “联接” 部分，从而在 “要返回的字段” 下拉列表中提供其他字段。您可以添加多个联接，包括跨实体的链式联接。您还可以按联接实体中的字段进行筛选和排序。

   要删除联接，请选择其旁边的垃圾桶图标。这将从该联接中移除所有字段，并使用这些字段打破任何依赖的联接或约束。
4. 在条件中，添加、编辑或删除筛选查询输出的规则。您可以将规则组织成组，也可以使用AND或OR语句将多个规则链接在一起。有关您可以使用的运算符的更多信息，请参阅数据操作条件运算符和示例。
5. 在排序中，通过选择属性并选择升序或降序来配置查询结果的排序方式。您可以通过选择排序规则旁边的垃圾桶图标来删除排序配置。

6. 在转换结果中，您可以输入 custom JavaScript 来修改或格式化结果，然后再显示结果或将其发送到自动化。
7. 在输出预览中，根据配置的字段、筛选器、排序和查看查询输出的预览表 JavaScript。

数据操作条件运算符和示例

您可以使用条件运算符将配置的表达式值与实体列进行比较，以返回数据库对象的子集。您可以使用的运算符取决于列的数据类型以及实体所连接的数据库的类型，例如 Amazon Redshift、Amazon Aurora 或 Amazon DynamoDB。

以下条件运算符可用于所有数据库服务：

- =和!=：适用于所有数据类型（不包括主键列）。
- <=、>=<、和>=：仅适用于数字列。
- `IS NULL`an `IS NOT NULL` d：用于匹配具有空值或空值的列。在每个数据库中，空值的解释通常有所不同，但是在 App Studio 中，`NULL`运算符匹配并返回连接的数据库表中包含空值的记录。

以下条件运算符只能在连接到支持它们的数据库服务的实体中使用：

- `LIKE`和`NOT LIKE`（Redshift，Aurora）：用于在连接的数据库中执行基于模式的查询。该`LIKE`运算符为搜索功能提供了灵活性，因为它可以查找并返回符合指定模式的记录。您可以使用与模式中的任何字符或字符序列相匹配的通配符来定义模式。每个数据库管理系统都有一组唯一的通配符，但最常用的两个通配符是`%`表示任意数量的字符（包括 0），以及`_`表示单个字符。
- `Contains`和 `Not Contains` (DynamoDB)：用于执行区分大小写的搜索以确定是否在列值中找到给定文本。
- `Starts With`和 `Not Starts With` (DynamoDB)：用于执行区分大小写的搜索，以确定是否在列值的开头找到给定文本。

数据库支持条件运算符

下表显示了每个可以连接到 App Studio 的数据库支持哪些数据操作条件运算符。

	=, !=, <, >, <=, >=	喜欢，不喜欢	包含，不包含	开头为，不开头	为空，不为空
DynamoDB	是	否	是	是	是

	=, !=, <, >, <=, >=	喜欢，不喜欢	包含，不包含	开头为，不开头	为空，不为空
Aurora	支持	是	否	否	是
Redshift	支持	是	否	否	是

数据操作条件示例

考虑以下数据库表，其中包含多个带有`namecity`、和`hireDate`字段的项目。

名称	city	HireDate
亚当	Seattle	2025-03-01
艾德丽安	波士顿	2025-03-05
Bob	阿尔伯克基	2025-03-06
Carlos	芝加哥	2025-03-10
卡罗琳	NULL	2025-03-12
丽塔	迈阿密	2025-03-15

现在，考虑在 App Studio 中创建数据操作，以返回符合指定条件的项目的`name`字段。以下列表包含条件示例以及该表为每个示例返回的值。

> **Note**
>
> 这些示例的格式为 SQL 示例，它们可能不像在 App Studio 中那样出现，但用于说明运算符的行为。

- `WHERE name LIKE 'Adam'`: 退货`Adam`。
- `WHERE name LIKE 'A%'`: 退货`Adam`和`Adrienne`.
- `WHERE name NOT LIKE 'B_B'`: 返回`Adam``Adrienne`、`Carlos`、`Caroline`、和`Rita`。

- `WHERE contains(name, 'ita')`: 退货`Rita`。
- `WHERE begins_with(name, 'Car')`: 退货`Carlos`和`Caroline`.
- `WHERE city IS NULL`: 退货`Caroline`。
- `WHERE hireDate < "2025-03-06"`: 退货`Adam`和`Adrienne`.
- `WHERE hireDate >= DateTime.now().toISODate()`: 请注意，`DateTime.now().toISODate()`返回当前日期。在当前日期为 2025-03-10 的场景中，表达式返回`Carlos`、和。`Caroline` `Rita`

> **Tip**
> 有关比较表达式中的日期和时间的更多信息，请参阅日期和时间。

删除数据操作

使用以下步骤从 App Studio 实体中删除数据操作。

1. 如有必要，请导航到要删除其数据操作的实体。
2. 选择 “数据操作” 选项卡。
3. 对于要删除的每个数据操作，请选择 “编辑” 旁边的下拉菜单，然后选择 “删除”。
4. 在对话框中选择 “确认”。

添加或删除示例数据

您可以向 App Studio 应用程序中的实体添加示例数据。由于应用程序在发布之前不会与外部服务通信，因此可以使用示例数据在预览环境中测试您的应用程序和实体。

1. 如有必要，请导航到要编辑的实体。
2. 选择 “样本数据” 选项卡。
3. 要生成样本数据，请选择生成更多样本数据。
4. 要删除样本数据，请选中要删除的数据的复选框，然后按 Delete 或 Backspace 键。选择 “保存” 以保存更改。

添加或编辑连接的数据源和地图字段

> **Tip**
>
> 您可以按 CTRL+Z 撤消最近对实体所做的更改。

1. 如有必要，请导航到要编辑的实体。
2. 选择 “连接” 选项卡可查看或管理实体与发布应用程序时存储数据的数据源表之间的连接。连接数据源表后，您可以将实体字段映射到表的列。
3. 在连接器中，选择包含与所需数据源表的连接的连接器。有关连接器的更多信息，请参见使用连接器将 App Studio 连接到其他服务。
4. 在表中，选择要用作实体数据源的表。
5. 该表显示了实体的字段以及它们映射到的数据源列。选择 “自动映射” 可自动将实体字段与数据源列映射。您也可以通过在每个实体字段的下拉列表中选择数据源列来手动映射表中的字段。

删除实体

使用以下步骤从 App Studio 应用程序中删除实体。

> **Note**
>
> 从 App Studio 应用程序中删除实体不会删除连接的数据源表，包括相应的 DynamoDB 托管实体表。数据源表将保留在关联 AWS 账户中，如果需要，需要从相应的服务中删除。

删除实体

1. 如有必要，请导航到您的应用程序。
2. 选择数据选项卡。
3. 在左侧的 “实体” 菜单中，选择要删除的实体旁边的省略号菜单，然后选择 “删除”。
4. 查看对话框中的信息，输入**`confirm`**并选择删除以删除实体。

AWS App Studio 中的管理数据实体

通常，您可以在 App Studio 中为实体配置与外部数据库表的连接，并且必须创建每个实体字段并将其映射到连接的数据库表中的一列。更改数据模型时，必须更新外部数据库表和实体，并且必须重新映射已更改的字段。虽然这种方法很灵活，可以使用不同类型的数据源，但需要更多的前期规划和持续维护。

托管实体是 App Studio 为您管理整个数据存储和配置过程的一种实体。创建托管实体时，将在关联账户中创建相应的 DynamoDB 表。 AWS 这确保了内部安全透明的数据管理 AWS。对于托管实体，您可以在 App Studio 中配置该实体的架构，相应的 DynamoDB 表也会自动更新。

在多个应用程序中使用托管实体

在 App Studio 应用程序中创建托管实体后，该实体就可以在其他 App Studio 应用程序中使用。通过提供单个底层资源进行维护，这有助于为具有相同数据模型和架构的应用程序配置数据存储。

在多个应用程序中使用托管实体时，必须使用创建托管实体的原始应用程序对相应的 DynamoDB 表进行所有架构更新。在其他应用程序中对该实体所做的任何架构更改都不会更新相应的 DynamoDB 表。

托管实体限制

主键更新限制：创建实体后无法更改其主键名称或类型，因为这是 DynamoDB 中的破坏性更改，会导致现有数据丢失。

重@@ 命名列：在 DynamoDB 中重命名列时，实际上是在创建新列，而原始列仍保留原始数据。原始数据不会自动复制到新列或从原始列中删除。您可以重命名托管实体字段（称为系统名称），但您将无法访问原始列及其数据。重命名显示名称没有限制。

更改数据类型：尽管 DynamoDB 允许在创建表后灵活修改列数据类型，但此类更改可能会严重影响现有数据以及查询逻辑和准确性。数据类型更改需要转换所有现有数据以符合新格式，这对于大型活动表来说很复杂。此外，在数据迁移完成之前，数据操作可能会返回意想不到的结果。您可以切换字段的数据类型，但现有数据不会迁移到新的数据类型。

排序列：DynamoDB 支持通过排序键检索已排序的数据。排序键必须与分区键一起定义为复合主键的一部分。限制包括强制性的排序键、仅限于一个分区内的排序以及没有跨分区的全局排序。需要对排序键进行仔细的数据建模，以避免出现热分区。我们将不支持排序预览里程碑。

加入：DynamoDB 不支持联接。为了避免昂贵的联接操作，表在设计上是非规范化的。为了对 one-to-many关系进行建模，子表包含一个引用父表主键的属性。多表数据查询涉及从父表中查找项目以检索详细信息。作为预览里程碑的一部分，我们将不支持托管实体的本机联接。作为一种解决方法，我们将

引入一个自动化步骤，该步骤可以执行 2 个实体的数据合并。这将与一级查询非常相似。我们将不支持排序预览里程碑。

Env Stag e：我们将允许发布进行测试，但在两个环境中使用相同的托管存储

页面和自动化参数

参数是 AWS App Studio 中的一项强大功能，用于在应用程序中的不同组件、页面和自动化之间传递动态值。使用参数，您可以打造灵活且具有情境感知能力的体验，从而提高应用程序的响应速度和个性化程度。本文涵盖两种类型的参数：页面参数和自动化参数。

主题

- 页面参数
- 自动化参数

页面参数

页面参数是一种在页面之间发送信息的方式，通常用于在 App Studio 应用程序中从一个页面导航到另一个页面以维护上下文或传递数据。页面参数通常由名称和值组成。

页面参数用例

页面参数用于在 App Studio 应用程序中的不同页面和组件之间传递数据。它们对以下用例特别有用：

1. 搜索和筛选：当用户在您的应用程序的主页上搜索时，可以将搜索词作为参数传递给结果页面，从而使其仅显示相关的筛选项目。例如，如果用户搜索*noise-cancelling headphones*，则*noise-cancelling headphones*可以将带有值的参数传递到产品列表页面。
2. 查看商品详情：如果用户点击商品（例如产品），则该商品的唯一标识符可以作为参数传递到详情页面。这允许详细信息页面显示有关特定项目的所有信息。例如，当用户点击耳机产品时，该产品的唯一 ID 将作为参数传递到产品详情页面。
3. 在页面导航中传递用户上下文：当用户在页面之间导航时，参数可以传递重要的上下文，例如用户的位置、首选产品类别、购物车内容和其他设置。例如，当用户在您的应用程序上浏览不同的产品类别时，他们的位置和首选类别会作为参数保留，从而提供个性化和一致的体验。
4. 深度链接：使用页面参数将指向应用程序内特定页面的链接共享或添加书签。
5. 数据操作：您可以创建接受参数值的数据操作，以便根据传递的参数筛选和查询数据源。例如，在产品列表页面上，您可以创建一个接受`category`参数以获取相关产品的数据操作。

页面参数安全注意事项

虽然页面参数提供了一种在页面之间传递数据的强大方法，但您必须谨慎使用它们，因为如果使用不当，它们可能会暴露敏感信息。以下是需要记住的重要安全注意事项：

1. 避免在中暴露敏感数据 URLs
   a. 风险： URLs（包括数据操作参数）通常出现在服务器日志、浏览器历史记录和其他地方。因此，必须避免在页面参数值中暴露敏感数据，例如用户凭证、个人身份信息 (PII) 或任何其他机密数据。
   b. 缓解措施：考虑使用可以安全映射到敏感数据的标识符。例如，您可以传递一个可用于获取用户名或电子邮件地址的随机唯一标识符，而不是将用户的姓名或电子邮件地址作为参数传递。

自动化参数

自动化参数是 App Studio 中的一项强大功能，可用于通过传递来自各种来源（例如用户界面、其他自动化或数据操作）的动态值来创建灵活且可重复使用的自动化。它们充当占位符，在自动化运行时会被实际值替换，从而允许您每次使用具有不同输入的相同自动化。

在自动化中，参数具有唯一的名称，您可以使用 params 变量和参数名称来引用参数的值，例如，`{{params.customerId}}`。

本文深入了解自动化参数，包括其基本概念、用法和最佳实践。

自动化参数的好处

自动化参数具有多种优点，包括以下列表：

1. 可@@ 重用性：通过使用参数，您可以创建可重复使用的自动化，这些自动化可以使用不同的输入值进行自定义，从而允许您在不同的输入中重复使用相同的自动化逻辑。
2. 灵活性：无需将值硬编码到自动化中，而是可以定义参数并在需要时提供不同的值，从而使自动化更具动态性和适应性。
3. 关注点分离：参数有助于将自动化逻辑与使用的特定值区分开来，从而促进代码的组织和可维护性。
4. 验证：每个参数都有一种数据类型，例如字符串、数字或布尔值，将在运行时进行验证。这样可以确保无需自定义验证代码即可拒绝数据类型不正确的请求。
5. 可选参数和必填参数：您可以将自动化参数指定为可选参数或必填参数。运行自动化时必须提供必需的参数，而可选参数可以具有默认值或省略。这种灵活性使您可以创建更多功能的自动化，这些自动化可以根据提供的参数处理不同的场景。

场景和用例

场景：检索商品详情

想象一下，您有一个自动化系统，可以根据产品 ID 从数据库中检索产品详细信息。此自动化可能有一个名为的参数productId。

该productId参数充当占位符，您可以在运行自动化时使用实际的产品 ID 值填充该占位符。您可以定义productId参数并在每次运行自动化时传入不同的产品 ID 值，而不必将特定的产品 ID 硬编码到自动化中。

您可以从组件的数据源调用此自动化，使用双大括号语法将所选产品的 ID 作为productId参数传递：{{ui.productsTable.selectedRow.id}}。这样，当用户从表 (ui.productsTable) 中选择产品时，自动化将通过传递所选行的 ID 作为productId参数来检索所选产品的详细信息。

或者，您可以从另一个自动化中调用此自动化，该自动化会循环浏览产品列表，并通过传递产品的 ID 作为productId参数来检索每个产品的详细信息。在这种情况下，productId参数值将在循环的每次迭代中从{{product.id}}表达式中动态提供。

通过使用productId参数和双大括号语法，可以使这种自动化更加灵活和可重复使用。无需为每个产品创建单独的自动化，只需提供来自不同来源（例如用户界面组件或其他自动化）的相应产品 ID 作为参数值即可检索任何产品的详细信息。

场景：使用备用值处理可选参数

让我们考虑一个场景，其中你有一个 “任务” 实体，其中包含必填的 “所有者” 列，但你希望该字段在自动化中成为可选字段，如果未选择所有者，则提供备用值。

1. 使用名为的参数创建自动化Owner，该参数映射到OwnerTask实体的字段。
2. 由于该Owner字段在实体中是必填字段，因此该Owner参数将与所需的设置同步。
3. 要在自动化中将该Owner参数设为可选，请关闭该参数的required设置。
4. 在你的自动化逻辑中，你可以使用类似的表达式{{params.Owner || currentUser.userId}}。此表达式检查是否提供了Owner参数。如果未提供，它将回退到当前用户的 ID 作为所有者。
5. 这样，如果用户没有在表单或组件中选择所有者，则自动化会自动将当前用户指定为任务的所有者。

通过切换Owner参数的required设置并使用后备表达式，您可以将其与实体字段要求分离，使其在自动化中成为可选的，并在未提供参数时提供默认值。

定义自动化参数类型

通过使用参数类型来指定数据类型和设置要求，您可以控制自动化的输入。这有助于确保您的自动化在预期输入下可靠运行。

同步实体中的类型

动态同步实体字段定义中的参数类型和要求可以简化与实体数据交互的自动化构建，从而确保参数始终反映最新的实体字段类型和要求。

以下过程详细说明了同步实体参数类型的一般步骤：

1. 使用键入字段（例如布尔值、数字等）创建实体，并根据需要标记字段。
2. 创建新的自动化。
3. 向自动化添加参数，然后在选择 “类型” 时，选择要与之同步的实体字段。数据类型和所需设置将自动从映射的实体字段同步
4. 如果需要，您可以通过为每个参数打开/关闭来覆盖 “必需” 设置。这意味着所需的状态将不会与实体字段保持同步，但除此之外，它将保持同步。

手动定义类型

您也可以手动定义参数类型，而无需从实体进行同步

通过定义自定义参数类型，您可以创建接受特定输入类型并根据需要处理可选或必需参数的自动化，而不必依赖实体字段映射。

1. 使用键入字段（例如布尔值、数字等）创建实体，并根据需要标记字段。
2. 创建新的自动化。
3. 向自动化添加参数，然后在选择类型时，选择所需的类型。

配置要传递给自动化参数的动态值

为自动化定义参数后，可以在调用自动化时向其传递值。您可以通过两种方式传递参数值：

1. 组件触发器：如果您通过组件触发器（例如单击按钮）调用自动化，则可以使用 JavaScript 表达式来传递组件上下文中的值。例如，如果您有一个名为的文本输入字段`emailInput`，则可以使用以下表达式将其值传递给电子邮件参数：`ui.emailInput.value`。
2. 其他自动化：如果您要从其他自动化中调用自动化，则可以使用 JavaScript 表达式来传递自动化上下文中的值。例如，您可以传递另一个参数的值或上一个操作步骤的结果。

类型安全

通过使用特定数据类型（例如字符串、数字或布尔值）定义参数，可以确保传递到自动化的值属于预期类型。

> **Note**
>
> 在 App Studio 中，日期是 ISO 字符串日期，这些日期也将被验证。

这种类型安全有助于防止类型不匹配，这可能会导致自动化逻辑中出现错误或意外行为。例如，如果您将参数定义为 `aNumber`，则可以确信传递给该参数的任何值都将是一个数字，并且无需在自动化中执行额外的类型检查或转换。

验证

您可以向参数添加验证规则，确保传递到自动化的值符合特定标准。

虽然 App Studio 不提供内置的参数验证设置，但您可以通过在自动化中添加一个 JavaScript 操作来实现自定义验证，该操作在违反特定限制时会引发错误。

对于实体字段，是验证规则的子集，例如minimum/maximum values, are supported. However, those are not validated at the automation level, only at the data layer, when running Create/Update/Delete 记录操作。

自动化参数的最佳实践

为确保您的自动化参数经过精心设计、可维护且易于使用，请遵循以下最佳实践：

1. 使用描述性参数名称：选择能清楚描述参数目的或上下文的参数名称。
2. 提供参数描述：在定义参数时，请使用 “描述” 字段来解释其用途、限制和期望。这些描述将出现在引用参数时的 JSDoc 注释中，以及用户在调用自动化时需要为参数提供值的任何用户界面中。
3. 使用适当的数据类型：根据预期的输入值仔细考虑每个参数的数据类型，例如：字符串、数字、布尔值、对象。
4. 验证参数值：在继续执行进一步操作之前，在自动化中实施适当的验证检查，以确保参数值满足特定要求。
5. 使用后备值或默认值：虽然 App Studio 目前不支持为参数设置默认值，但您可以在使用自动化逻辑中的参数时实现后备值或默认值。例如，如果未提供参数或`param1`参数值为假，则可以使用类似的表达式`{{ params.param1 || "default value" }}`来提供默认值。

6. 保持参数一致性：如果您有多个自动化需要相似的参数，请尝试在这些自动化中保持参数名称和数据类型的一致性。
7. 文档参数的使用：为自动化保留清晰的文档，包括每个参数的描述、其用途、预期值以及任何相关的示例或边缘情况。
8. 经常检查和重构：定期检查您的自动化及其参数，根据需要重构或合并参数，以提高清晰度、可维护性和可重用性。
9. 限制参数数量：虽然参数提供了灵活性，但参数过多会使自动化变得复杂且难以使用。通过将参数数量限制在必要的范围内，力求在灵活性和简单性之间取得平衡。
10. 考虑参数分组：如果您发现自己定义了多个相关参数，请考虑将它们分组为单个`Object`参数。
11. 单独关注的问题：避免将单个参数用于多种用途，或将不相关的值组合成单个参数。每个参数都应代表一个不同的关注点或一段数据。
12. 使用参数别名：如果参数名称较长或复杂，请考虑在自动化逻辑中使用别名或速记版本，以提高可读性和可维护性。

通过遵循这些最佳实践，您可以确保自动化参数经过精心设计、可维护且易于使用，从而最终提高自动化的整体质量和效率。

JavaScript 用于在 App Studio 中编写表达式

在 AWS App Studio 中，您可以使用 JavaScript 表达式来动态控制应用程序的行为和外观。单行 JavaScript 表达式用双花括号、`{{ }}`、编写，可用于各种上下文，例如自动化、用户界面组件和数据查询。这些表达式在运行时进行求值，可用于执行计算、操作数据和控制应用程序逻辑。

App Studio 为三个 JavaScript 开源库提供原生支持：Luxon、UUID、Lodash 以及 SDK 集成，用于检测应用程序配置中的 JavaScript 语法和类型检查错误。

> **Important**
>
> App Studio 不支持使用第三方库或自定义 JavaScript 库。

基本语法

JavaScript 表达式可以包括变量、文字、运算符和函数调用。表达式通常用于执行计算或评估条件。

请参阅以下示例：

- `{{ 2 + 3 }}`将评估为 5。
- `{{ "Hello, " + "World!" }}`将评估为 “你好，世界！”。
- `{{ Math.max(5, 10) }}`将评估为 10。
- `{{ Math.random() * 10 }}`返回介于 [0-10) 之间的随机数（含小数）。

插值

也可以使用 JavaScript 在静态文本中插入动态值。这是通过将 JavaScript 表达式用双大括号括起来实现的，如下例所示：

```
Hello {{ currentUser.firstName }}, welcome to App Studio!
```

在此示例中，`currentUser.firstName`是一个 JavaScript 表达式，用于检索当前用户的名字，然后将其动态插入到问候消息中。

联接

您可以使用中的+运算符连接字符串和变量 JavaScript，如下例所示。

```
{{ currentRow.FirstName + " " + currentRow.LastName }}
```

此表达式将`currentRow.FirstName`和`currentRow.LastName`的值组合在一起，中间有一个空格，从而得出当前行的全名。例如，如果`currentRow.FirstName`是`John`、`currentRow.LastName`是`Doe`，则表达式将解析为`John Doe`。

日期和时间

JavaScript 提供了用于处理日期和时间的各种函数和对象。例如：

- `{{ new Date().toLocaleDateString() }}`：以本地化格式返回当前日期。
- `{{ DateTime.now().toISODate() }}`：以 YYYY-MM-DD格式返回当前日期，以用于 “日期” 组件。

日期和时间比较

使用诸如=、、><>=、或之类的运算符<=来比较日期或时间值。例如：

- `{{ui.timeInput.value > "10:00 AM"}}`: 检查时间是否在上午 10:00 之后。
- `{{ui.timeInput.value <= "5:00 PM"}}`: 检查时间是否在下午 5:00 或之前。
- `{{ui.timeInput.value > DateTime.now().toISOTime()}}`：检查时间是否在当前时间之后。
- `{{ui.dateInput.value > DateTime.now().toISODate()}}`：检查日期是否早于当前日期。
- `{{ DateTime.fromISO(ui.dateInput.value).diff(DateTime.now(), "days").days >= 5 }}`：检查该日期是否距离当前日期至少有 5 天。

代码块

除了表达式之外，您还可以编写多行 JavaScript 代码块。与表达式不同，代码块不需要花括号。相反，您可以直接在 JavaScript 代码块编辑器中编写代码。

> **Note**
>
> 在计算表达式并显示其值时，运行代码块并显示其输出（如果有）。

全局变量和函数

App Studio 提供对某些全局变量和函数的访问权限，这些变量和函数可以在 JavaScript 表达式和代码块中使用。例如，`currentUser`是一个表示当前登录用户的全局变量，您可以访问诸如检索用户角色之类`currentUser.role`的属性。

引用或更新 UI 组件值

您可以在组件和自动化操作中使用表达式来引用和更新界面组件值。通过以编程方式引用和更新组件值，您可以创建响应应用户输入和数据更改的动态交互式用户界面。

引用 UI 组件值

您可以通过访问用户界面组件中的值来实现动态行为，从而创建交互式和数据驱动的应用程序。

通过在表达式中使用`ui`命名空间，可以在同一页面上访问用户界面组件的值和属性。通过引用组件的名称，您可以检索其值或根据其状态执行操作。

> **Note**
>
> ui命名空间将仅显示当前页面上的组件，因为组件的作用域仅限于其各自的页面。

在 App Studio 应用程序中引用组件的基本语法是:{{ui.*componentName*}}.

以下列表包含使用ui命名空间访问界面组件值的示例：

- {{ui.*textInputName*.value}}: 表示名为的文本输入组件的值*textInputName*。
- {{ui.*formName*.isValid}}：根据您提供的验证标准，检查名*formName*为的表单中的所有字段是否有效。
- {{ui.*tableName*.currentRow.*columnName*}}：表示名为的表组件当前行中特定列的值*tableName*。
- {{ui.*tableName*.selectedRowData.*fieldName*}}：表示名为的表组件中选定行中指定字段的值*tableName*。然后，您可以追加诸如 ID ({{ui.*tableName*.selectedRowData.*ID*}}) 之类的字段名称，以引用选定行中该字段的值。

以下列表包含更多引用组件值的具体示例：

- {{ui.*inputText1*.value.trim().length > 0}}：在修剪任何前导或尾随空格后，检查*inputText1*组件的值是否具有非空字符串。这对于验证用户输入或根据输入文本字段的值启用/禁用其他组件非常有用。
- {{ui.*multiSelect1*.value.join(", ")}}: 对于名为的多选组件*multiSelect1*，此表达式将所选选项值的数组转换为逗号分隔的字符串。这对于以用户友好的格式显示所选选项或将选择传递给其他组件或自动化非常有用。
- {{ui.*multiSelect1*.value.includes("*option1*")}}：此表达式检查该值*option1*是否包含在*multiSelect1*组件的选定选项数组中。如果*option1*选择则返回 true，否则返回 false。这对于有条件地渲染组件或根据特定选项选择采取操作非常有用。
- {{ui.*s3Upload1*.files.length > 0}}：对于名为的 Amazon S3 文件上传组件*s3Upload1*，此表达式通过检查文件数组的长度来检查是否已上传任何文件。它可用于根据文件是否已上传来启用/禁用其他组件或操作。
- {{ui.*s3Upload1*.files.filter(file => file.type === "*image/png*").length}}：此表达式筛选*s3Upload1*组件中已上传文件的列表，使其仅包含 PNG 图像文件，并返回这些文件的数量。这可能有助于验证或显示有关上传文件类型的信息。

更新 UI 组件值

要更新或操作组件的值，请在自动化RunComponentAction中使用。以下是可用于更新使用RunComponentAction操作命名的*myInput*文本输入组件值的语法示例：

```
RunComponentAction(ui.myInput, "setValue", "New Value")
```

在此示例中，该RunComponentAction步骤调用*myInput*组件上的setValue操作，并传入新值*New Value*。

处理表格数据

您可以访问表数据和值以执行操作。您可以使用以下表达式来访问表数据：

- currentRow：用于访问表中当前行的表数据。例如，设置表格操作的名称，将行中的值发送到从操作启动的自动化，或者使用表中现有列中的值来创建新列。
- ui.*tableName*.selectedRow和ui.*tableName*.selectedRowData都用于访问页面上其他组件的表数据。例如，根据所选行在表格外设置按钮的名称。返回的值相同，但selectedRow和之间的区别selectedRowData如下：
  - selectedRow：此命名空间包括每个字段的列标题中显示的名称。selectedRow在引用表中可见列中的值时，应使用。例如，如果您的表中有一个自定义列或计算列，但该列在实体中不作为字段存在。
  - selectedRowData：此命名空间包括实体中用作表源的字段。您应该使用selectedRowData来引用实体中的一个值，该值在表格中不可见，但对于应用程序中的其他组件或自动化很有用。

以下列表包含在表达式中访问表数据的示例：

- {{ui.*tableName*.selectedRow.*columnNameWithNoSpace*}}：返回表中选定行中该*columnNameWithNoSpace*列的值。
- {{ui.*tableName*.selectedRow.['*Column Name With Space*']}}：返回表中选定行中该*Column Name With Space*列的值。
- {{ui.*tableName*.selectedRowData.*fieldName*}}：返回表格中选定行中*fieldName*实体字段的值。
- {{ui.*tableName*.selectedRows[0].*columnMappingName*}}：从同一页面上的其他组件或表达式中引用选定行的列名。

- {{currentRow.*firstName* + ' ' + currentRow.*lastNamecolumnMapping*}}：连接多列中的值以在表中创建新列。
- {{ { "Blocked": "#", "Delayed": "#", "On track": "#" } [currentRow.*statuscolumnMapping*] + " " + currentRow.*statuscolumnMapping*}}：根据存储的状态值自定义表中字段的显示值。
- {{currentRow.*colName*}}、{{currentRow["*First Name*"]}}{{currentRow}}、或{{ui.*tableName*.selectedRows[0]}}：在行操作中传递被引用的行的上下文。

访问自动化

你可以在 App Studio 中使用自动化来运行服务器端逻辑和操作。在自动化操作中，您可以使用表达式来处理数据、生成动态值以及合并先前操作的结果。

访问自动化参数

您可以将来自用户界面组件和其他自动化的动态值传递到自动化中，从而使其可重复使用且灵活。这是使用params命名空间的自动化参数完成的，如下所示：

{{params.*parameterName*}}：引用从用户界面组件或其他来源传递到自动化的值。例如，{{params.*ID*}}将引用名为的参数*ID*。

操纵自动化参数

您可以使用 JavaScript 来操作自动化参数。请参阅以下示例：

- {{params.*firstName*}} {{params.*lastName*}}：连接作为参数传递的值。
- {{params.*numberParam1* + params.*numberParam2*}}：添加两个数字参数。
- {{params.*valueProvided*?.length > 0 ? params.*valueProvided* : '*Default*'}}：检查参数是否不为空或未定义，且长度是否为非零。如果为 true，则使用提供的值；否则，设置默认值。
- {{params.*rootCause* || "*No root cause provided*"}}：如果params.*rootCause*参数为 false（空、未定义或空字符串），请使用提供的默认值。
- {{Math.min(params.numberOfProducts, 100)}}：将参数的值限制为最大值（在本例中为100）。
- {{ DateTime.fromISO(params.startDate).plus({ days: 7 }).toISO() }}：如果params.*startDate*参数为"2023-06-15T10:30:00.000Z"，则此表达式的计算结果将为"2023-06-22T10:30:00.000Z"，即开始日期一周后的日期。

访问先前操作的自动化结果

自动化允许应用程序运行服务器端的逻辑和操作，例如查询数据库 APIs、与之交互或执行数据转换。results命名空间提供对同一自动化中先前操作返回的输出和数据的访问权限。关于访问自动化结果，请注意以下几点：

1. 您只能访问同一个自动化中先前自动化步骤的结果。
2. 如果您有按该顺序命名的*action1*操作，则*action1*无法引用任何结果，*action2*只能访问results.action1。*action2*
3. 这也适用于客户端操作。例如，如果你有一个使用InvokeAutomation操作触发自动化的按钮。然后，您可以设置一个导航步骤，results.*myInvokeAutomation1*.fileType === "pdf"其Run If条件是，如果自动化显示文件是 PDF，则使用 PDF 查看器导航到页面。

以下列表包含使用results命名空间访问先前操作的自动化结果的语法。

- {{results.*stepName*.data}}：从名为的自动化步骤中检索数据数组*stepName*。
- {{results.*stepName*.output}}：检索名为的自动化步骤的输出*stepName*。

访问自动化步骤结果的方式取决于操作的类型及其返回的数据。不同的操作可能会返回不同的属性或数据结构。以下是一些常见的示例：

- 对于数据操作，您可以使用访问返回的数据数组results.*stepName*.data。
- 对于 API 调用操作，您可以使用访问响应正文results.*stepName*.body。
- 对于 Amazon S3 操作，您可以使用访问文件内容results.*stepName*.Body.transformToWebStream()。

请参阅文档，了解您正在使用的特定操作类型，以了解它们返回的数据的形状以及如何在results命名空间中访问这些数据。以下列表包含一些示例

- {{results.*getDataStep*.data.filter(row => row.status === "pending").length}}：假设*getDataStep*是一个返回数据行数组的Invoke Data Action自动化操作，则此表达式会筛选数据数组，使其仅包括状态字段等于的行pending，并返回筛选数组的长度（计数）。这对于根据特定条件查询或处理数据非常有用。
- {{params.*email*.split("@")[0]}}：如果params.*email*参数包含电子邮件地址，则此表达式会在 @ 符号处拆分字符串，并返回 @ 符号之前的部分，从而有效地提取电子邮件地址的用户名部分。

- `{{new Date(params.timestamp * 1000)}}`：此表达式采用 Unix 时间戳参数 (`params.timestamp`) 并将其转换为 JavaScript Date 对象。它假设时间戳以秒为单位，因此将其乘以1000，将其转换为毫秒，这是构造函数所期望的格式。`Date`这对于在自动化中处理日期和时间值非常有用。
- `{{results.stepName.Body}}`: 对于名为的`Amazon S3 GetObject`自动化操作`stepName`，此表达式会检索文件内容，用户界面组件（例如 I mag e 或 PDF Viewer）可以使用这些内容来显示检索到的文件。请注意，需要在自动化的自动化输出中配置此表达式才能在组件中使用。

数据依赖关系和时序注意事项

在 App Studio 中构建复杂的应用程序时，了解和管理不同数据组件（例如表单、详细信息视图和自动化驱动的组件）之间的数据依赖关系至关重要。数据组件和自动化可能无法同时完成数据检索或执行，这可能会导致计时问题、错误和意外行为。通过了解潜在的计时问题并遵循最佳实践，您可以在 App Studio 应用程序中创建更可靠、更一致的用户体验。

一些潜在的问题如下：

1. 渲染时序冲突：数据组件的渲染顺序可能与其数据依赖关系不一致，从而可能导致视觉不一致或错误。
2. 自动化运行时间：自动化任务可能在组件完全加载之前完成，从而导致运行时执行错误。
3. 组件崩溃：由自动化提供支持的组件可能会在响应无效或自动化尚未完成运行时崩溃。

示例：订单详情和客户信息

此示例演示了数据组件之间的依赖关系如何导致数据显示中的计时问题和潜在错误。

假设一个在同一页面上包含以下两个数据组件的应用程序：

- 用于获取订单数据的详情组件 (`orderDetails`)。
- 显示与订单相关的客户详细信息的详情组件 (`customerDetails`)。

在此应用程序中，`orderDetails`详细信息组件中有两个字段，配置了以下值：

```
// 2 text fields within the orderDetails detail component

// Info from orderDetails Component
```

```
{{ui.orderDetails.data[0].name}}

// Info from customerDetails component
{{ui.customerDetails.data[0].name}} // Problematic reference
```

在此示例中，orderDetails组件试图通过引用customerDetails组件中的数据来显示客户名称。这是有问题的，因为orderDetails组件可能会在customerDetails组件获取其数据之前进行渲染。如果customerDetails组件数据获取延迟或失败，则orderDetails组件将显示不完整或不正确的信息。

数据依赖和计时最佳实践

使用以下最佳实践来缓解 App Studio 应用程序中的数据依赖和计时问题：

1. 使用条件渲染：只有在确认组件或数据可用时才渲染组件或显示数据。在显示数据之前，使用条件语句检查数据是否存在。以下代码段显示了一个条件语句的示例：

   ```
   {{ui.someComponent.data ? ui.someComponent.data.fieldName : "Loading..."}}
   ```

2. 管理子组件的可见性：对于在加载数据之前呈现子组件的 Stepflow、Form 或 Detail 等组件，请手动设置子组件的可见性。以下代码段显示了根据父组件数据可用性设置可见性的示例：

   ```
   {{ui.parentComponent.data ? true : false}}
   ```

3. 使用联接查询：如果可能，使用联接查询在单个查询中提取相关数据。这减少了单独的数据提取次数，并最大限度地减少了数据组件之间的时序问题。
4. 在自动化中@@ 实现错误处理：在自动化中实施强大的错误处理，以优雅地管理预期数据不可用或收到无效响应的场景。
5. 使用可选链接：访问嵌套属性时，使用可选链接以防止父属性未定义时出错。以下片段显示了可选链接的示例：

   ```
   {{ui.component.data?.[0]?.fieldSystemName}}
   ```

构建包含多个用户的应用程序

多个用户可以在单个 App Studio 应用程序上工作，但是一次只能有一个用户编辑一个应用程序。有关邀请其他用户编辑应用程序以及多个用户尝试同时编辑应用程序时的行为的信息，请参阅以下部分。

邀请构建者编辑应用程序

按照以下说明邀请其他构建者编辑 App Studio 应用程序。

邀请其他构建者编辑应用程序

1. 如有必要，请导航到应用程序的应用程序工作室。
2. 选择共享。
3. 在 “开发” 选项卡中，使用文本框搜索并选择要邀请其编辑应用程序的群组或个人用户。
4. 对于每个用户或组，选择下拉列表并选择要授予该用户或组的权限。
   - 共同所有者：共同所有者拥有与应用程序所有者相同的权限。
   - 仅限编辑：角色为 “仅限编辑” 的用户拥有与所有者和共同所有者相同的权限，但以下权限除外：
     - 他们不能邀请其他用户编辑应用程序。
     - 他们无法将应用程序发布到测试或生产环境。
     - 他们无法向应用程序添加数据源。
     - 他们无法删除或复制该应用程序。

正在尝试编辑正在由其他用户编辑的应用程序

一个 App Studio 应用程序一次只能由一个用户编辑。请参阅以下示例，了解当多个用户尝试同时编辑一个应用程序时会发生什么。

在此示例中，当前User A正在编辑一个应用程序，并已与其共享该应用程序User B。 User B然后尝试编辑正在编辑的应用程序User A。

User B尝试编辑应用程序时，将出现一个对话框，通知他们当前User A正在编辑应用程序，继续编辑应用程序将退User A出应用程序工作室，所有更改都将被保存。 User B可以选择取消并User A继续操作，也可以选择继续并进入应用程序工作室编辑应用程序。在此示例中，他们选择编辑应用程序。

当User B选择编辑应用程序时，User A会收到一条通知，告知应用程序User B已开始编辑，其会话已结束。请注意，如果User A在非活动浏览器选项卡中打开应用程序，他们可能不会收到通知。在这种情况下，如果他们尝试返回应用程序并尝试进行编辑，他们将收到一条错误消息并被引导刷新页面，这将使他们返回到应用程序列表。

查看或更新应用程序的内容安全设置

App Studio 中的每个应用程序都具有内容安全设置，可用于限制外部媒体或资源（例如图像、iFrame 和 PDFs iFrame）的加载，或者仅允许来自指定域或 URLs（包括 Amazon S3 存储桶）。您还可以指定您的应用程序可以将对象上传到 Amazon S3 的域。

所有应用程序的默认内容安全设置是阻止加载来自外部来源的所有媒体，包括 Amazon S3 存储桶，并阻止将对象上传到 Amazon S3。因此，要加载图像、iFrame 或类似媒体，必须编辑设置以允许媒体来源。 PDFs此外，要允许将对象上传到 Amazon S3，您必须编辑设置以允许上传到的域。

> **Note**
>
> 内容安全设置用于在应用程序中配置内容安全策略 (CSP) 标头。CSP 是一种安全标准，可帮助保护您的应用程序免受跨站脚本 (XSS)、点击劫持和其他代码注入攻击。有关 CSP 的更多信息，请参阅 MDN Web 文档中的内容安全政策 (CSP)。

更新应用程序的内容安全设置

1. 如有必要，可通过从应用程序列表中选择编辑应用程序来导航到该应用程序的应用程序工作室。
2. 选择应用程序设置。
3. 选择 “内容安全设置” 选项卡以查看以下设置：
   - 帧源：用于管理您的应用程序可以从中加载框架和 iframe（例如交互式内容或 PDFs）的域。此设置会影响以下组件或应用程序资源：
     - iFrame 嵌入组件
     - PDF 查看器组件
   - 图片来源：用于管理您的应用程序可以从中加载图像的网域。此设置会影响以下组件或应用程序资源：
     - 应用程序徽标和横幅
     - 图像查看器组件
   - Connect sourc e：用于管理您的应用程序可以将 Amazon S3 对象上传到的域。
4. 对于每个设置，从下拉列表中选择所需的设置：
   - 全部阻止 frames/images/connections：不允许加载任何媒体（图像、框架 PDFs）或将任何对象上传到 Amazon S3。

- 全部允许frames/images/connections：允许加载来自所有域的所有媒体（图像、框架 PDFs），或允许将所有域的对象上传到 Amazon S3。
- 允许特定域：允许从指定域加载媒体或向指定域上传媒体。域或指定 URLs 为以空格分隔的表达式列表，其中通配符 (*) 可用于子域、主机地址或端口号，以表示每个域名的所有合法值均有效。指定`http`也匹配`https`。以下列表包含有效条目的示例：
  - `blob:`：匹配所有 blob，其中包括自动化操作返回的文件数据，例如从 Amazon S3 存储桶`GetObject`返回的项目，或者由 Amazon Bedrock 生成的图像。

    > **Important**
    >
    > 您必须在提供的表达式中包含`blob:`以允许操作返回文件数据，即使您的表达式是`*`，也应将其更新为 `* blob:`

  - `http://*.example.com`: 匹配所有从的子域加载的`example.com`尝试。还匹配`https`资源。
  - `https://source1.example.com https//source2.example.com`: 匹配所有从`https://source1.example.com`和中加载的尝试 `https://source2.example.com`
  - `https://example.com/subdirectory/`: 匹配所有尝试加载子目录下文件的尝试。例如`https://example.com/subdirectory/path/to/file.jpeg`。它不匹配`https://example.com/path/to/file.jpeg`。

5. 选择 保存 以保存您的更改。

App Studio 故障排除和

主题

- App Studio 设置、权限和入门问题疑难解答
- 对应用程序进行故障排除和调试
- 发布和共享应用程序疑难解答

App Studio 设置、权限和入门问题疑难解答

本主题包括有关在设置或加入 App Studio 时解决常见问题以及管理权限的信息。

选择 “为我创建账户实例” 选项时 App Studio 设置失败

问题：如果您在任何 AWS 地区都有账户级 IAM 身份中心实例，则使用 “为我创建账户” 实例设置 App Studio 将失败，因为 IAM Identity Center 仅支持一个实例。

解决方案：导航到 IAM 身份中心控制台 https://console.aws.amazon.com/singlesignon/，检查您是否有 IAM 身份中心实例。检查每个支持的 AWS 区域，直到找到实例。您可以在设置 App Studio 时使用该实例，也可以删除 IAM Identity Center 实例，然后使用为我创建账户实例选项重试。

> **Warning**
>
> 删除 IAM 身份中心实例将影响所有现有用例。在删除实例之前，请确保未使用该实例，或者使用该实例设置 App Studio。

设置后无法访问 App Studio

问题：在设置 App Studio 时，您可能提供了您不是其成员的 IAM 身份中心群组。您必须是至少一个群组的成员才能访问 App Studio。

解决方案：导航到 IAM 身份中心控制台，将自己https://console.aws.amazon.com/singlesignon/添加到设置时添加到 App Studio 的群组中。

不确定登录 App Studio 时要使用什么用户名或密码

问题：您可能不确定如何登录 App Studio，这可能是因为您尚未设置 IAM 身份中心证书，或者您忘记了 IAM 身份中心的用户名或密码。

解决方案：在没有 IAM 身份中心实例的情况下设置 App Studio 时，会为每个用户提供电子邮件和用户名，用于创建 IAM Identity Center 用户。提供的每个电子邮件地址都收到一封电子邮件，邀请其加入 IAM Identity Center。每个用户都必须接受邀请并为其 IAM Identity Center 用户证书创建密码。然后，每位用户都可以使用 IAM 身份中心的用户名和密码登录 App Studio。

如果您已经设置了证书，但忘记了用户名或密码，则必须要求管理员使用 IAM Identity Center 控制台查看和提供您的用户名，或者重置您的密码。

设置 App Studio 时出现系统错误

问题：设置 App Studio 时出现以下错误：

```
System error. We encountered a problem. Report the issue and the App Studio service
 team will get back to you.
```

当服务遇到未知错误时，就会发生此错误。

解决方案：加入社区 Slack，联系支持团队，方法是在左侧导航栏的 “学习” 部分或编辑应用程序时在顶部横幅中选择 “在 Slack 上加入我们”。

我找不到我的 App Studio 实例网址

如果您找不到访问您的 App Studio 实例的网址，请联系设置 App Studio 的管理员。管理员可以在 App Studio 控制台的中查看该 URL AWS Management Console。

我无法在 App Studio 中修改群组或角色

问题：您无法在左侧导航栏中看到 “角色” 链接。这是因为只有具有管理员角色的用户才能在 App Studio 中修改群组和角色。

解决方案：联系具有管理员角色的用户以更改群组或角色，或者联系您的管理员以将其添加到管理员群组。

如何退出 App Studio

此时你无法退出 App Studio。建议移除所有资源，例如应用程序和连接器，并将群组的角色更改为应用程序用户，以防止访问或使用。您还应该删除专门用于 App Studio 的第三方资源，例如 IAM 角色或数据库表。

对应用程序进行故障排除和调试

以下主题包含有关对 App Studio 应用程序进行故障排除和调试的信息。

主题

- 对 AI 生成器助手和聊天进行故障排除
- 在应用工作室进行故障排除
- 预览应用程序疑难解答
- 在测试环境中进行故障排除
- 使用 Amazon 日志中已发布应用程序的 CloudWatch 日志进行调试
- 连接器故障排除

对 AI 生成器助手和聊天进行故障排除

本主题包含使用 AI 生成器助手时常见问题的故障排除指南。

使用 AI 创建应用程序时出错

使用 AI 提示创建应用程序时，可能会出现以下错误：

```
We apologize, but we cannot proceed with your request. The request may contain content
 that violates our policies and guidelines. Please revise your prompt before trying
 again.
```

问题：由于内容可能有害，请求被阻止。

解决方案：改写提示并重试。

使用 AI 生成的应用程序是空的应用程序或缺少组件。

问题：这可能是由意外的服务错误引起的。

解决方案：重试使用 AI 创建应用程序，或者在生成的应用程序中手动创建组件。

在应用工作室进行故障排除

本主题包含构建应用程序时出现问题的故障排除和调试指南。

使用调试面板

为了帮助您在构建应用程序时进行实时调试，App Studio 提供了一个可折叠的生成器调试面板，该面板横跨应用程序工作室的页面、自动化和数据选项卡。此面板同时显示错误和警告。虽然警告可以作为可操作的建议，例如尚未配置的资源，但必须解决错误才能成功预览或发布您的应用程序。每个错误或警告都包含一个 “查看” 链接，可用于导航到问题所在地。

调试面板会在出现新的错误或警告时自动更新，错误或警告在解决后自动消失。当您离开构建器时，这些警告和错误消息的状态将保持不变。

JavaScript 表达式语法和数据类型处理

App Studio 具有 JavaScript 错误检测功能，通过用红线在代码下划线来突出显示错误。这些编译错误会阻止应用程序成功构建，它们表明了诸如错别字、无效引用、无效操作和所需数据类型的错误输出等问题。有关常见问题，请参阅以下列表：

1. 重命名资源导致的错误：当 JavaScript 表达式在 App Studio 中引用资源名称时，更改这些名称将导致表达式不正确并产生错误。您可以在调试面板中查看这些错误。
2. 数据类型问题：数据类型不匹配将在您的应用程序中产生错误。例如，如果将自动化配置为接受类型的参数`String`，但将组件配置为发送类型的值`Integer`，则会发生错误。检查相应资源（包括组件、自动化以及数据实体和操作）之间的数据类型是否匹配。您可能需要更改 JavaScript 表达式中值的类型。

预览应用程序疑难解答

本主题包含有关在尝试预览应用程序时解决问题的信息。

无法加载预览，并出现以下错误：`Your app failed to build and cannot be previewed`

问题：您的应用程序必须成功构建才能进行预览。当出现编译错误导致您的应用程序无法成功构建时，就会发生此错误。

解决方案：使用应用程序工作室中的调试面板查看并解决错误。

加载预览需要很长时间

问题：某些类型的应用程序更新需要很长时间才能编译和构建。

解决方案：让该选项卡保持打开状态，然后等待生成更新。您应该在应用程序的应用程序工作室的右上角看到 “已保存”，预览将重新加载。

预览未反映最新更改

问题：当您的应用程序编辑会话被其他用户接管但未收到通知时，可能会发生这种情况。这可能会导致正在编辑的应用程序与预览环境不匹配。

解决方案：刷新 Application Studio 浏览器选项卡，并在需要时接管编辑会话。

在测试环境中进行故障排除

本主题包含有关对发布到测试环境的应用程序进行故障排除的信息。

> **Note**
>
> 来自自动化或数据操作的 HTTP 500 响应可能是由于表达式的运行时崩溃、连接器故障或来自连接到应用程序的数据源的限制所致。使用中的说明查看使用浏览器控制台进行调试将显示潜在错误详细信息的调试日志。

使用调试面板

与构建应用程序时使用的构建调试面板类似，App Studio 在测试环境中提供了一个可折叠的调试面板。此面板显示信息性消息，例如页面加载时间、用户导航和应用程序事件。它还包含错误和警告。事件发生时，调试面板会自动更新新消息。

使用浏览器控制台进行调试

由于在预览应用程序时不会调用操作，因此需要将您的应用程序发布到测试环境以测试其呼叫和响应处理。如果在执行自动化过程中出现错误，或者您想了解应用程序为何以某种方式运行，则可以使用浏览器的控制台进行实时调试。

使用浏览器控制台在测试环境中调试应用程序

1. 追加`?debug=true`到网址的末尾并按回车键。请注意，如果 URL 已有查询字符串（其中包含?），则应将其附加`&debug=true`到 URL 的末尾。
2. 打开浏览器控制台，通过浏览您的操作或 API 输入和输出开始调试。
   - 在 Chrome 中：在浏览器中右键单击，然后选择 “检查”。有关使用 Chrome 进行调试的更多信息 DevTools，请参阅 Chrome DevTools 文档。

- 在 Firefox 中：长按或右键单击网页元素，然后选择 “检查元素”。有关使用 Firefox 进行调试的更多信息 DevTools，请参阅 F irefox DevTools 用户文档。

以下列表包含一些会导致错误的常见问题：

- 运行时错误
  - 问题：如果自动化或表达式配置不正确，则可能会在自动化运行时导致错误。常见的错误是重命名资产，从而导致表达式不正确、其他 JavaScript 编译错误或尝试使用这些数据或资产。`undefined`
  - 解决方案：检查自定义代码输入（表达式 JavaScript、和 JSON）的每种用法，并确保代码编辑器或调试面板中没有编译错误。
- 连接器问题
  - 问题：由于 App Studio 应用程序在发布之前不会通过连接器与外部服务通信，因此测试环境中可能会出现预览期间未发生的错误。如果自动化中使用连接器的操作失败，则可能是由于向连接器发送请求的操作配置错误，也可能是连接器配置本身导致的。
  - 解决方案：您应该在预览环境的早期使用 Mocked 输出来测试自动化，以防止出现这些错误。确保您的连接器配置正确，有关更多信息，请参阅连接器故障排除。最后，您可以使用 CloudWatch 查看日志。有关更多信息，请参阅 使用 Amazon 日志中已发布应用程序的 CloudWatch 日志进行调试。在`ConnectorService`命名空间日志中，应该有源自连接器的错误消息或元数据。

使用 Amazon 日志中已发布应用程序的 CloudWatch 日志进行调试

Ama CloudWatch zon Logs 会实时监控您的 AWS 资源和您运行 AWS 的应用程序。您可以使用 CloudWatch 日志来收集和跟踪指标，这些指标是您可以衡量资源和应用程序的变量。

对于调试 App Studio 应用程序， CloudWatch 日志可用于跟踪应用程序执行过程中发生的错误、审计信息以及提供有关用户操作和专有交互的上下文。这些日志提供了历史数据，您可以使用这些数据来审核应用程序的使用情况和访问模式，以及查看用户遇到的错误。

> **Note**
>
> CloudWatch 日志不提供从应用程序用户界面传递的参数值的实时跟踪。

使用以下步骤在 “日志” 中访问您的 App Studio 应用程序的 CloudWatch 日志。

1. 在您的应用程序的 App Studio 应用程序工作室中，通过查看网址来找到并记下您的应用程序 ID。应用程序ID可能看起来像这样:`802a3bd6-ed4d-424c-9f6b-405aa42a62c5`.
2. 打开 CloudWatch 控制台，网址为https://console.aws.amazon.com/cloudwatch/。
3. 在导航窗格中，选择 日志组。
4. 在这里，您可以找到每个应用程序的五个日志组。根据您感兴趣的信息类型，选择一个群组，然后为要发现的数据编写查询。

   以下列表包含日志组以及有关何时使用每个日志组的信息：

   1. `/aws/appstudio/teamId/appId/TEST/app`：用于调试自动化响应、组件错误或与当前发布到测试环境的应用程序版本相关的 JavaScript 代码。
   2. `/aws/appstudio/teamId/appId/TEST/audit`：用于调试 JavaScript 代码错误，例如条件可见性或转换、查询失败以及与当前发布到测试环境的应用程序版本相关的登录或权限用户错误。
   3. `/aws/appstudio/teamId/setup`：用于监视生成器或管理员的操作。
   4. `/aws/appstudio/teamId/appId/PRODUCTION/app`：用于调试自动化响应、查询失败、组件错误或与当前发布到生产环境的应用程序版本相关的 JavaScript 代码。
   5. `/aws/appstudio/teamId/appId/PRODUCTION/audit`：用于调试 JavaScript 代码错误，例如条件可见性或转换，以及与当前发布到生产环境的应用程序版本相关的登录或权限用户错误。

   > **Note**
   >
   > 大多数用于调试的日志都归类在`DebugLogClient`命名空间下。

5. 进入日志组后，您可以选择最新的日志流，也可以选择上次事件时间最接近感兴趣时间的日志流，也可以选择搜索所有日志流以搜索该日志组中的所有事件。有关在日志中 CloudWatch 查看日志数据的更多信息，请参阅查看发送到日志的 CloudWatch 日志数据。

使用 L CloudWatch ogs Insights 查询对日志进行筛选和排序

您可以使用 CloudWatch Logs Insights 同时查询多个日志组。确定包含会话信息的日志组列表后，导航到 Lo CloudWatch gs Insights 并选择日志组。然后，通过自定义查询进一步缩小目标日志条目的范围。以下是一些示例查询：

包含关键字的日志列表：***error***

```
fields @timestamp, @message
| filter @message like 'error'
| sort @timestamp desc
```

来自测试环境的调试日志：

```
fields @timestamp, @message
| filter namespace = "DebugLogClient"
| sort @timestamp desc
```

每隔 5 分钟，504/404/500 总错误计数：

```
filter @message like '/api/automation' and (@message like ': 404' or @message like ':
 500' or @message like ': 504')
| fields @timestamp, method, path, statusCode
| stats count(*) as errorCount by bin(5m)
```

有关 “ CloudWatch 日志见解”、“使用 Logs Insights 分析 CloudWatch 日志数据” 的更多信息，请参阅 Amazon CloudWatch Logs 用户指南。

连接器故障排除

本主题包含常见连接器问题的故障排除指南。您必须是管理员组的成员才能查看或编辑连接器。

检查您的 IAM 角色是否具有正确的自定义信任策略和标签

在为连接器设置 IAM 角色时，请确保正确配置自定义信任策略以提供对 App Studio 的访问权限。如果 AWS 资源位于用于设置 App Studio 的同一个 AWS 账户中，则仍需要使用此自定义信任策略。

- 确保该`Principal`部分中的 AWS 账号是用于设置 App Studio 的账户的账户 ID。 AWS 此账号并不总是资源所在的账户。
- 确保`"aws:PrincipalTag/IsAppStudioAccessRole": "true"`已正确添加到该`sts:AssumeRole`部分。

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

```
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::111122223333:root"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:PrincipalTag/IsAppStudioAccessRole": "true"
      }
    }
  }
  ]
}
```

此外，请确保已将具有以下键和值的标签添加到 IAM 角色，有关添加标签的更多信息，请参阅标记 IAM 角色：

> **Note**
>
> 请注意，标签的值为`IsAppStudioDataAccessRole`，这与自定义信任策略中的值 (`IsAppStudioAccessRole`) 略有不同。

- 键：`IsAppStudioDataAccessRole`
- 值：`true`

检查您的连接器所连接的产品或服务中资源的配置。某些资源，例如亚马逊 Redshift 表，需要额外的配置才能在 App Studio 中使用。

检查您的连接器配置。对于 AWS 服务，请访问 App Studio 中的连接器，确保包含正确的亚马逊资源名称 (ARN)，并且指定的 AWS 区域是包含您的资源的区域。

检查您的 IAM 角色是否具有正确的权限

要向 App Studio 提供对 AWS 资源的访问权限，您必须为连接器使用的 IAM 角色分配适当的权限。所需的权限是要执行的服务、资源和操作所独有的。例如，从 Amazon Redshift 表中读取数据所需的权限与将对象上传到 Amazon S3 存储桶所需的权限不同。有关更多信息，Connect 连接到 AWS 服务请参阅中的相应主题。

对亚马逊 Redshift 连接器进行故障排除

本节包括 Amazon Redshift 连接器常见问题的疑难解答指南。有关配置 Amazon Redshift 连接器和资源的信息，请参阅。连接亚马逊 Redshift

1. 确保在 Amazon Redshift 编辑`OFF`器上将`Isolated Session`切换开关设置为。必须使用此设置才能看到其他用户（例如 App Studio 应用程序）所做的数据更改。
2. 确保在 Amazon Redshift 表上授予了相应的权限。
3. 在连接器配置中，确保选择了与 Amazon Redshift 表类型相匹配的相应计算类型（`Provisioned`或`Serverless`）。

Aurora 连接器故障排除

本节包括 Aurora 连接器常见问题的故障排除指南。有关配置 Aurora 连接器和资源的信息，请参阅连接亚马逊 Aurora。

1. 创建表时，请确保选择适当且受支持的 Aurora 版本。
2. 确认已启用 Amazon RDS 数据 API，因为这是允许 App Studio 对 Aurora 表执行操作的必要条件。有关更多信息，请参阅启用 Amazon RDS 数据 API。
3. 验证是否提供了 AWS Secrets Manager 权限。

DynamoDB 连接器疑难解答

本节包括 DynamoDB 连接器常见问题的疑难解答指南。有关配置 DynamoDB 连接器和资源的信息，请参阅。连接亚马逊 DynamoDB

如果在创建连接器时未显示您的 DynamoDB 表架构，则可能是因为您的 DynamoDB 表已使用客户管理的密钥 (CMK) 加密，并且如果没有描述密钥和解密表的权限，则无法访问表数据。要使用使用 CMK 加密的表创建 DynamoDB 连接器，您必须将`kms:describeKey`和权限添加到`kms:decrypt`您的 IAM 角色。

对 Amazon S3 连接器进行故障排除

本节包括 Amazon S3 连接器常见问题的疑难解答指南。有关配置 Amazon S3 连接器和资源的信息，请参阅连接到亚马逊简单存储服务 (Amazon S3) Service。

一般故障排除指南包括检查以下内容：

1. 确保将 Amazon S3 连接器配置为使用 Amazon S3 资源所在的 AWS 区域。
2. 确保 IAM 角色配置正确。
3. 在 Amazon S3 存储桶中，确保 CORS 配置授予了相应的权限。有关更多信息，请参阅 步骤 1：创建和配置 Amazon S3 资源。

Amazon S3 文件上传错误：无法计算预签名 URL

尝试使用 S3 上传组件将文件上传到 Amazon S3 存储桶时，可能会遇到以下错误：

```
Error while uploading file to S3: Failed to calculate presigned URL.
```

此错误通常是由错误的 IAM 角色配置或 Amazon S3 存储桶上的 CORS 配置不正确造成的，可以通过使用中的连接到亚马逊简单存储服务 (Amazon S3) Service信息修复这些配置来解决。

发布和共享应用程序疑难解答

本主题包含发布或共享 App Studio 应用程序时常见问题的疑难解答指南。

我在 “共享” 对话框中看不到新创建的应用程序角色

只有在重新发布应用程序后，新创建的应用程序级别角色才会显示在 “共享” 对话框中。创建新角色后发布应用程序以使用它们。

我的应用程序发布完成后我没有收到电子邮件

发布应用程序时，只有应用程序所有者会收到一封电子邮件。

我的应用程序的最终用户无法访问已发布的应用程序

如果您的最终用户无法访问您发布的应用程序，并且在尝试访问该应用程序时收到`Forbidden`消息，则很可能未与尝试访问该应用程序的用户共享已发布的应用程序。必须与群组共享已发布的应用程序，才能向群组中的用户授予访问权限。

要了解有关共享应用程序的更多信息，请参阅共享已发布的应用程序。

AWS 应用工作室的安全性

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 AWS 云中运行 AWS 服务的基础架构。 AWS 还为您提供可以安全使用的服务。作为 AWS 合规性计划的一部分，我们的安全措施的有效性定期由第三方审计员进行测试和验证。要了解适用于 App Studio 的合规计划，请参阅按合规性计划划分的范围内AWS 服务。
- 云端安全-您的责任由您使用的 AWS 服务决定。您还需要对其他因素负责，包括您的数据的敏感性、您组织的要求以及适用的法律法规。

本文档将帮助您了解在使用 App Studio 时如何应用分担责任模型。以下主题向您介绍如何配置 App Studio 以满足您的安全与合规性目标。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 App Studio 资源。

主题

- 安全注意事项和缓解措施
- AWS App Studio 中的数据保护
- AWS 应用工作室和 AWS Identity and Access Management (IAM)
- AWS 应用工作室合规性验证
- AWS 应用工作室的弹性
- AWS App Studio 中的基础设施安全
- AWS App Studio 中的配置和漏洞分析
- 防止跨服务混淆座席
- AWS App Studio 中的跨区域数据传输

安全注意事项和缓解措施

安全性注意事项

在处理数据连接器、数据模型和已发布的应用程序时，会出现一些与数据泄露、访问控制和潜在漏洞相关的安全问题。以下列表包括主要的安全问题。

IAM 角色配置不正确

数据连接器的 IAM 角色配置不正确可能会导致未经授权的访问和数据泄露。向数据连接器的 IAM 角色授予过于宽松的访问权限可能会允许未经授权的用户访问和修改敏感数据。

使用 IAM 角色执行数据操作

由于 App Studio 应用程序的最终用户使用连接器配置中提供的 IAM 角色来执行操作，因此这些最终用户可能会访问他们通常无权访问的数据。

删除已发布应用程序的数据连接器

删除数据连接器后，关联的机密凭据不会自动从已使用该连接器的已发布应用程序中删除。在这种情况下，如果使用某些连接器发布了应用程序，并且其中一个连接器已从 App Studio 中删除，则已发布的应用程序将继续使用先前存储的连接器凭据运行。值得注意的是，尽管删除了连接器，但已发布的应用程序仍将不受影响并可以运行。

在已发布的应用程序上编辑数据连接器

编辑数据连接器时，所做的更改不会自动反映在使用该连接器的已发布应用程序中。如果使用某些连接器发布了应用程序，并且其中一个连接器在 App Studio 中进行了修改，则发布的应用程序将继续使用先前存储的连接器配置和凭据。要合并更新的连接器更改，必须重新发布应用程序。在重新发布应用程序之前，它将保持不正确且无法运行，或者不受影响且可以运行，但不会反映最新的连接器修改。

安全风险缓解建议

本节列出了避免安全风险的缓解建议，详见前面的 “安全注意事项” 部分。

1. 正确的 IAM 角色配置：确保按照最小权限原则正确配置数据连接器的 IAM 角色，以防止未经授权的访问和数据泄露。
2. 受限的应用程序访问权限：仅与有权查看应用程序数据或对应用程序数据执行操作的用户共享您的应用程序。
3. 应用程序发布：确保在更新或删除连接器时重新发布应用程序。

AWS App Studio 中的数据保护

AWS 分担责任模型分适用于 AWS App Studio 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 云。您负责维护对托管在此基础结构上的内容的控制。您还负责您所使用的 AWS 服务 的安全配置和管理任务。有关数据隐私的更多信息，请参阅数据隐私常见问题。有关欧

洲数据保护的信息，请参阅 AWS Security Blog 上的 AWS Shared Responsibility Model and GDPR 博客文章。

出于数据保护目的，我们建议您保护 AWS 账户 凭证并使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 设置个人用户。这样，每个用户只获得履行其工作职责所需的权限。还建议您通过以下方式保护数据：

- 对每个账户使用多重身份验证（MFA）。
- 使用 SSL/TLS 与资源通信。 AWS 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》。

强烈建议您切勿将机密信息或敏感信息（如您客户的电子邮件地址）放入标签或自由格式文本字段（如名称字段）。这包括您 AWS 服务 使用控制台、API 或与 AWS App Studio 或其他人合作时 AWS SDKs。 AWS CLI在用于名称的标签或自由格式文本字段中输入的任何数据都可能会用于计费或诊断日志。如果您向外部服务器提供网址，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。

数据加密

App Studio 通过加密静态和传输中的数据来安全地存储和传输数据。

静态加密

静态加密是指通过对存储数据进行加密来保护您的数据免受未经授权的访问。App Studio 默认使用 AWS KMS 密钥提供静态加密，您无需对静态数据加密进行任何其他配置。

App Studio 为您的应用程序安全存储以下数据：源代码、构建构件、元数据和权限信息。

使用使用 AWS KMS 客户托管密钥 (CMK) 加密的数据源时，App Studio 资源将继续使用 AWS 托管密钥进行加密，而加密数据源中的数据则由 CMK 加密。有关在 App Studio 应用程序中使用加密数据源的更多信息，请参阅将加密数据源与 CMKs。

App Studio 使用亚马逊 CloudFront 向您的用户提供您的应用程序。 CloudFront 用于边缘接入点 (POPs)，使用 SSDs 加密的 EBS 卷用于区域边缘缓存 (RECs)。Functi CloudFront ons 中的功能代码和配置始终以加密格式存储 SSDs 在边缘位置 POPs和使用的其他存储位置 CloudFront。

传输中加密

传输中加密是指在通信终端节点之间移动数据时，保护您的数据免遭拦截。默认情况下，App Studio 为传输中的数据提供加密。客户与 App Studio 之间以及 App Studio 与其下游依赖项之间的所有通信都使用使用签名版本 4 签名流程签名的 TLS 连接进行保护。所有 App Studio 端点都使用由 AWS Certificate Manager 私有证书颁发机构管理的 SHA-256 证书。

密钥管理

App Studio 不支持管理加密密钥。

互联网络流量隐私

在 App Studio 中创建实例时，您可以选择存储该实例的数据和资源的 AWS 区域。应用程序构建工件和元数据永远不会离开该 AWS 区域。

但是，请注意以下信息：

- 由于 App Studio 使用亚马逊 CloudFront 为您的应用程序提供服务，并使用 Lambda @Edge 来管理应用程序的身份验证，因此可以从 CloudFront 边缘站点（可能位于不同的区域）访问有限的身份验证数据、授权数据和应用程序元数据。
- AWS App Studio 跨 AWS 区域传输数据，以便在服务中启用某些生成式 AI 功能。有关跨区域数据传输所启用的功能、跨区域移动的数据类型以及如何选择退出的更多信息，请参阅AWS App Studio 中的跨区域数据传输。

AWS 应用工作室和 AWS Identity and Access Management (IAM)

在 AWS App Studio 中，您可以通过将 IAM Identity Center 中的群组分配给 App Studio 中的相应角色来管理服务中的访问权限和权限。群组成员的权限由分配的角色决定，而不是直接在 AWS Identity and Access Management (IAM) 中配置用户、角色或权限。有关在 App Studio 中管理访问权限和权限的更多信息，请参阅在 App Studio 中管理访问权限和角色。

App Studio 在出于计费目的验证实例时，以及在连接到 AWS 账户以创建和使用该 AWS 账户中的资源时，会与 IAM 集成。有关将 App Studio 连接到其他 AWS 服务以便在您的应用程序中使用的信息，请参阅Connect 连接到 AWS 服务。

在 App Studio 中创建实例时，必须关联一个 AWS 账户作为实例的账单和管理账户。为了启用关键功能，App Studio 还会创建 IAM 服务角色，为服务提供代表您执行任务所需的权限。

AWS Identity and Access Management (IAM) AWS 服务 可帮助管理员安全地控制对 AWS 资源的访问权限。IAM 管理员控制谁可以进行身份验证（登录）和授权（有权限）使用 App Studio 资源。您可以使用 IAM AWS 服务 ，无需支付额外费用。

主题

- App Studio 基于身份的政策
- App Studio 内部基于资源的政策
- App Studio 的政策措施
- App Studio 的政策资源
- App Studio 的策略条件密钥
- ACLs 在应用工作室中
- 带有 App Studio 的
- 在 App Studio 中使用临时证书
- App Studio 的跨服务主体权限
- App Studio 的服务角色
- App Studio 的服务相关角色
- AWSAWS App Studio 托管策略
- App Studio 的服务相关角色
- App Studio 基于身份的 AWS 策略示例

在使用 IAM 管理对 App Studio 的访问权限之前，请先了解有哪些 IAM 功能可用于 App Studio。

您可以在 A AWS pp Studio 中使用的 IAM 功能

IAM 特征	应用工作室支持
基于身份的策略	是
基于资源的策略	否
策略操作	是

IAM 特征	应用工作室支持
策略资源	是
策略条件密钥	否
ACLs	否
ABAC（策略中的标签）	否
临时凭证	是
主体权限	是
服务角色	是
服务相关角色	是

要全面了解 App Studio 和其他 AWS 服务如何与大多数 IAM 功能配合使用，请参阅 IAM 用户指南中的与 IAM 配合使用的AWS 服务。

App Studio 基于身份的政策

支持基于身份的策略：是

基于身份的策略是可附加到身份（如 IAM 用户、用户组或角色）的 JSON 权限策略文档。这些策略控制用户和角色可在何种条件下对哪些资源执行哪些操作。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的使用客户管理型策略定义自定义 IAM 权限。

通过使用 IAM 基于身份的策略，您可以指定允许或拒绝的操作和资源以及允许或拒绝操作的条件。您无法在基于身份的策略中指定主体，因为它适用于其附加的用户或角色。要了解可在 JSON 策略中使用的所有元素，请参阅《IAM 用户指南》中的 IAM JSON 策略元素引用。

App Studio 基于身份的策略示例

要查看 App Studio 基于身份的策略的示例，请参阅。App Studio 基于身份的 AWS 策略示例

App Studio 内部基于资源的政策

支持基于资源的策略：否

基于资源的策略是附加到资源的 JSON 策略文档。基于资源的策略的示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。对于在其中附加策略的资源，策略定义指定主体可以对该资源执行哪些操作以及在什么条件下执行。您必须在基于资源的策略中指定主体。委托人可以包括账户、用户、角色、联合用户或 AWS 服务。

要启用跨账户访问，您可以将整个账户或其他账户中的 IAM 实体指定为基于资源的策略中的主体。将跨账户主体添加到基于资源的策略只是建立信任关系工作的一半而已。当委托人和资源处于不同位置时 AWS 账户，可信账户中的 IAM 管理员还必须向委托人实体（用户或角色）授予访问资源的权限。他们通过将基于身份的策略附加到实体以授予权限。但是，如果基于资源的策略向同一个账户中的主体授予访问权限，则不需要额外的基于身份的策略。有关更多信息，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

App Studio 的政策措施

支持策略操作：是

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

JSON 策略的 `Action` 元素描述可用于在策略中允许或拒绝访问的操作。策略操作通常与关联的 AWS API 操作同名。有一些例外情况，例如没有匹配 API 操作的仅限权限 操作。还有一些操作需要在策略中执行多个操作。这些附加操作称为相关操作。

在策略中包含操作以授予执行关联操作的权限。

要查看 App Studio 操作列表，请参阅《服务授权参考》中的 AWS App Studio 定义的操作。

App Studio 中的策略操作在操作前使用以下前缀：

```
appstudio
```

要在单个语句中指定多项操作，请使用逗号将它们隔开。

```
"Action": [
      "appstudio:action1",
      "appstudio:action2"
         ]
```

以下语句列出了 App Studio 中的所有操作：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWS App Studio permissions",
            "Effect": "Allow",
            "Action": [
                "appstudio:GetAccountStatus", // Required to get the current account's
 App Studio instance status
                "appstudio:GetEnablementJobStatus", // Required to get the status of an
 enablement job of an App Studio instance
                "appstudio:StartEnablementJob", // Required to start the enablement of
 an App Studio instance
                "appstudio:StartRollbackEnablementJob", // Required to disable an
 enabled App Studio instance
                "appstudio:StartTeamDeployment" // Required to start deployment in
 order to update the App Studio instance infrastructure
            ],
            "Resource": "*"
        }
    ]
}
```

App Studio 的政策资源

支持策略资源：是

App Studio 权限仅支持策略`Resource`元素中的通配符 (`*`)。

App Studio 的策略条件密钥

支持特定于服务的策略条件键：否

App Studio 不支持策略条件密钥。

ACLs 在应用工作室中

支持 ACLs：否

访问控制列表 (ACLs) 控制哪些委托人（账户成员、用户或角色）有权访问资源。 ACLs 与基于资源的策略类似，尽管它们不使用 JSON 策略文档格式。

带有 App Studio 的

支持 ABAC（策略中的标签）：否

App Studio 不支持基于属性的访问控制 (ABAC)。

在 App Studio 中使用临时证书

支持临时凭证：是

当你使用临时凭证登录时，有些 AWS 服务 不起作用。有关更多信息，包括哪些 AWS 服务 适用于临时证书，请参阅 IAM 用户指南中的AWS 服务 与 IAM 配合使用的信息。

如果您使用除用户名和密码之外的任何方法登录，则 AWS Management Console 使用的是临时证书。例如，当您 AWS 使用公司的单点登录 (SSO) 链接进行访问时，该过程会自动创建临时证书。当您以用户身份登录控制台，然后切换角色时，您还会自动创建临时凭证。有关切换角色的更多信息，请参阅《IAM 用户指南》中的从用户切换到 IAM 角色（控制台）。

您可以使用 AWS CLI 或 AWS API 手动创建临时证书。然后，您可以使用这些临时证书进行访问 AWS。 AWS 建议您动态生成临时证书，而不是使用长期访问密钥。有关更多信息，请参阅 IAM 中的临时安全凭证。

App Studio 的跨服务主体权限

支持转发访问会话（FAS）：是

当您使用 IAM 用户或角色在中执行操作时 AWS，您被视为委托人。使用某些服务时，您可能会执行一个操作，然后此操作在其他服务中启动另一个操作。FAS 使用调用委托人的权限以及 AWS 服务 向下游服务发出请求的请求。 AWS 服务只有当服务收到需要与其他 AWS 服务 或资源交互才能完成的请求时，才会发出 FAS 请求。在这种情况下，您必须具有执行这两项操作的权限。有关发出 FAS 请求时的策略详情，请参阅转发访问会话。

App Studio 的服务角色

支持服务角色：是

服务角色是由一项服务担任、代表您执行操作的 IAM 角色。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的创建向 AWS 服务委派权限的角色。

AWS App Studio 对某些功能使用 IAM 服务角色来授予 App Studio 代表你执行任务的权限。设置 App Studio 时，控制台会自动为支持的功能创建服务角色。

> **Warning**
>
> 更改服务角色的权限可能会中断 App Studio 的功能。仅当 App Studio 提供相关指导时，才可编辑服务角色。

App Studio 的服务相关角色

支持服务相关角色：是

服务相关角色是一种与服务相关联的 AWS 服务服务角色。服务可以代入代表您执行操作的角色。服务相关角色出现在您的中 AWS 账户 ，并且归服务所有。IAM 管理员可以查看但不能编辑服务相关角色的权限。

有关创建或管理服务相关角色的详细信息，请参阅能够与 IAM 搭配使用的AWS 服务。在表中查找服务相关角色列中包含 Yes 的表。选择是链接以查看该服务的服务相关角色文档。

AWSAWS App Studio 托管策略

要向用户、群组和角色添加权限，使用 AWS 托管策略比自己编写策略要容易得多。创建仅为团队提供所需权限的 IAM 客户管理型策略需要时间和专业知识。要快速入门，您可以使用我们的 AWS 托管策略。这些策略涵盖常见使用案例，可在您的 AWS 账户中使用。有关 AWS 托管策略的更多信息，请参阅 IAM 用户指南中的AWS 托管策略。

AWS 服务维护和更新 AWS 托管策略。您无法更改 AWS 托管策略中的权限。服务偶尔会向 AWS 托管式策略添加额外权限以支持新特征。此类更新会影响附加策略的所有身份（用户、组和角色）。当启动新特征或新操作可用时，服务最有可能会更新 AWS 托管式策略。服务不会从 AWS 托管策略中移除权限，因此策略更新不会破坏您的现有权限。

此外，还 AWS 支持跨多个服务的工作职能的托管策略。例如，ReadOnlyAccess AWS 托管策略提供对所有 AWS 服务和资源的只读访问权限。当服务启动一项新功能时， AWS 会为新操作和资源添加只读权限。有关工作职能策略的列表和说明，请参阅 IAM 用户指南中的适用于工作职能的AWS 托管式策略。

AWS 托管策略： AppStudioServiceRolePolicy

您不能将 `AppStudioServiceRolePolicy` 附加到自己的 IAM 实体。此政策附加到服务相关角色，允许 App Studio 代表您执行操作。有关更多信息，请参阅 App Studio 的服务相关角色。

此策略授予允许服务相关角色管理 AWS 资源的权限。

权限详细信息

此策略包括以下权限：

- `logs`-创建 CloudWatch 日志组和日志流。还允许在这些日志组和流中创建日志事件。
- `secretsmanager`-创建、读取、更新和删除由 App Studio 管理的托管密钥。
- `sso`-检索应用程序实例。
- `sso-directory`-检索用户信息并检索群组中的成员列表。

```
{
 "Version": "2012-10-17",
 "Statement": [
     {
         "Sid": "AppStudioResourcePermissionsForCloudWatch",
         "Effect": "Allow",
         "Action": [
             "logs:CreateLogGroup",
             "logs:CreateLogStream",
             "logs:PutLogEvents"
         ],
         "Resource": [
             "arn:aws:logs:*:*:log-group:/aws/appstudio/*"
         ],
         "Condition": {
             "StringEquals": {
                 "aws:ResourceAccount": "${aws:PrincipalAccount}"
             }
         }
     },
     {
         "Sid": "AppStudioResourcePermissionsForSecretsManager",
         "Effect": "Allow",
         "Action": [
             "secretsmanager:CreateSecret",
```

```
            "secretsmanager:DeleteSecret",
            "secretsmanager:DescribeSecret",
            "secretsmanager:GetSecretValue",
            "secretsmanager:PutSecretValue",
            "secretsmanager:UpdateSecret",
            "secretsmanager:TagResource"
        ],
        "Resource": "arn:aws:secretsmanager:*:*:secret:appstudio-*",
        "Condition": {
            "ForAllValues:StringEquals": {
                "aws:TagKeys": [
                    "IsAppStudioSecret"
                ]
            },
            "StringEquals": {
                "aws:ResourceAccount": "${aws:PrincipalAccount}",
                "aws:ResourceTag/IsAppStudioSecret": "true"
            }
        }
    },
    {
        "Sid": "AppStudioResourcePermissionsForManagedSecrets",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:DeleteSecret",
            "secretsmanager:DescribeSecret",
            "secretsmanager:GetSecretValue",
            "secretsmanager:PutSecretValue",
            "secretsmanager:UpdateSecret"
        ],
        "Resource": "arn:aws:secretsmanager:*:*:secret:appstudio!*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceAccount": "${aws:PrincipalAccount}",
                "secretsmanager:ResourceTag/aws:secretsmanager:owningService":
 "appstudio"
            }
        }
    },
    {
        "Sid": "AppStudioResourceWritePermissionsForManagedSecrets",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret"
```

```
            ],
            "Resource": "arn:aws:secretsmanager:*:*:secret:appstudio!*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        },
        {
            "Sid": "AppStudioResourcePermissionsForSSO",
            "Effect": "Allow",
            "Action": [
                "sso:GetManagedApplicationInstance",
                "sso-directory:DescribeUsers",
                "sso-directory:ListMembersInGroup"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        }
    ]
}
```

App Studio 更新 AWS 了托管政策

查看自该服务开始跟踪这些更改以来 App Studio AWS 托管政策更新的详细信息。

更改	描述	日期
AppStudioServiceRolePolicy – 对现有策略的更新	App Studio 添加了新的权限，允许在中管理 App Studio 托管的密钥 AWS Secrets Manager。	2025 年 3 月 14 日
应用工作室已开始跟踪更改	App Studio 已开始跟踪其 AWS 托管策略的变更。	2024 年 6 月 28 日

App Studio 的服务相关角色

App Studio 使用 AWS Identity and Access Management (IAM) 服务相关角色。服务相关角色是一种独特的 IAM 角色，直接关联到 App Studio。服务相关角色由 App Studio 预定义，包括该服务代表您调用其他 AWS 服务所需的所有权限。

服务相关角色使设置 App Studio 变得更加容易，因为您不必手动添加必要的权限。App Studio 定义了其服务相关角色的权限，除非另有定义，否则只有 App Studio 可以担任其角色。定义的权限包括信任策略和权限策略，以及不能附加到任何其他 IAM 实体的权限策略。

只有在首先删除相关资源后，您才能删除服务相关角色。这样可以保护您的 App Studio 资源，因为您不会无意中移除访问这些资源的权限。

内容

- App Studio 的服务相关角色权限
- 为 App Studio 创建服务相关角色
- 编辑 App Studio 的服务相关角色
- 删除 App Studio 的服务相关角色

App Studio 的服务相关角色权限

App Studio 使用名为`AWSServiceRoleForAppStudio`的服务相关角色。这是 App Studio 持续管理 AWS 服务、维护应用程序构建体验所需的服务相关角色。

`AWSServiceRoleForAppStudio`服务相关角色使用以下信任策略，该策略仅信任`appstudio-service.amazonaws.com`服务：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "appstudio-service.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

对于权限，AWSServiceRoleForAppStudio服务相关角色提供对以下服务的权限：

- 亚马逊 CloudWatch：发送有关 App Studio 使用情况的日志和指标。
- AWS Secrets Manager：在 App Studio 中管理连接器的凭据，用于将应用程序连接到其他服务。
- IAM 身份中心：只读访问权限以管理用户访问权限。

具体而言，授予的AWSServiceRoleForAppStudio权限由附加的AppStudioServiceRolePolicy托管策略定义。有关托管策略的更多信息（包括其包含的权限），请参阅AWS 托管策略： AppStudioServiceRolePolicy。

为 App Studio 创建服务相关角色

您无需手动创建服务相关角色。当您创建 App Studio 实例时，App Studio 会为您创建服务相关角色。

如果您删除此服务相关角色，建议您创建一个 App Studio 实例，以便自动为您创建另一个实例。

尽管不是必需的，但您也可以使用 IAM 控制台或 AWS CLI 通过使用服务名称创建服务相关角色来创建服务相关角色，如前appstudio-service.amazonaws.com面显示的信任策略片段所示。有关更多信息，请参阅 IAM 用户指南 中的创建服务相关角色。

编辑 App Studio 的服务相关角色

App Studio 不允许您编辑AWSServiceRoleForAppStudio服务相关角色。在创建服务相关角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。但是可以使用 IAM 编辑角色说明。有关更多信息，请参阅《IAM 用户指南》中的编辑服务相关角色。

删除 App Studio 的服务相关角色

您无需删除该AWSServiceRoleForAppStudio角色。当您删除 App Studio 实例时，App Studio 会自动清理资源并删除与服务相关的角色。

虽然不建议这样做，但您可以使用 IAM 控制台或删除服务相关角色。 AWS CLI 为此，您必须先清理服务相关角色的资源，然后才能将其删除。

> **Note**
>
> 如果您尝试删除资源时 App Studio 正在使用该角色，则删除可能会失败。如果发生这种情况，请等待几分钟后重试。

使用 IAM 手动删除服务相关角色

1. 从 App Studio 实例中删除应用程序和连接器。
2. 使用 IAM 控制台、IAM CLI 或 IAM API 删除 `AWSServiceRoleForAppStudio` 服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的删除服务相关角色。

App Studio 基于身份的 AWS 策略示例

默认情况下，用户和角色无权创建或修改 App Studio 资源。他们也无法使用 AWS Management Console、 AWS Command Line Interface (AWS CLI) 或 AWS API 执行任务。要授予用户对所需资源执行操作的权限，IAM 管理员可以创建 IAM 策略。管理员随后可以向角色添加 IAM 策略，用户可以代入角色。

要了解如何使用这些示例 JSON 策略文档创建基于 IAM 身份的策略，请参阅《IAM 用户指南》中的创建 IAM 策略（控制台）。

有关 App Studio 定义的操作和资源类型（包括每种资源类型的格式）的详细信息，请参阅《服务授权参考》中的 AWS App Studio 操作、资源和条件密钥。 ARNs

主题

- 策略最佳实践
- 使用 App Studio 控制台
- 允许用户查看他们自己的权限
- 示例 1：允许用户设置 App Studio 实例
- 示例 2：拒绝用户设置 App Studio 实例

策略最佳实践

基于身份的策略决定了某人是否可以在您的账户中创建、访问或删除 App Studio 资源。这些操作可能会使 AWS 账户产生成本。创建或编辑基于身份的策略时，请遵循以下指南和建议：

- 开始使用 AWS 托管策略并转向最低权限权限 — 要开始向用户和工作负载授予权限，请使用为许多常见用例授予权限的AWS 托管策略。它们在你的版本中可用 AWS 账户。我们建议您通过定义针对您的用例的 AWS 客户托管策略来进一步减少权限。有关更多信息，请参阅《IAM 用户指南》中的 AWS 托管式策略或工作职能的AWS 托管式策略。

- 应用最低权限：在使用 IAM 策略设置权限时，请仅授予执行任务所需的权限。为此，您可以定义在特定条件下可以对特定资源执行的操作，也称为最低权限许可。有关使用 IAM 应用权限的更多信息，请参阅《IAM 用户指南》中的 IAM 中的策略和权限。
- 使用 IAM 策略中的条件进一步限制访问权限：您可以向策略添加条件来限制对操作和资源的访问。例如，您可以编写策略条件来指定必须使用 SSL 发送所有请求。如果服务操作是通过特定的方式使用的，则也可以使用条件来授予对服务操作的访问权限 AWS 服务，例如 AWS CloudFormation。有关更多信息，请参阅《IAM 用户指南》中的 IAM JSON 策略元素：条件。
- 使用 IAM Access Analyzer 验证您的 IAM 策略，以确保权限的安全性和功能性 – IAM Access Analyzer 会验证新策略和现有策略，以确保策略符合 IAM 策略语言（JSON）和 IAM 最佳实践。IAM Access Analyzer 提供 100 多项策略检查和可操作的建议，以帮助您制定安全且功能性强的策略。有关更多信息，请参阅《IAM 用户指南》中的使用 IAM Access Analyzer 验证策略。
- 需要多重身份验证 (MFA)-如果 AWS 账户您的场景需要 IAM 用户或根用户，请启用 MFA 以提高安全性。若要在调用 API 操作时需要 MFA，请将 MFA 条件添加到您的策略中。有关更多信息，请参阅《IAM 用户指南》中的使用 MFA 保护 API 访问。

有关 IAM 中的最佳实操的更多信息，请参阅《IAM 用户指南》中的 IAM 中的安全最佳实践。

使用 App Studio 控制台

要访问 AWS App Studio 控制台，您必须拥有一组最低权限。这些权限必须允许您在中列出和查看有关 App Studio 资源的详细信息 AWS 账户。如果创建比必需的最低权限更为严格的基于身份的策略，对于附加了该策略的实体（用户或角色），控制台将无法按预期正常运行。

对于仅调用 AWS CLI 或 AWS API 的用户，您无需为其设置最低控制台权限。相反，只允许访问与其尝试执行的 API 操作相匹配的操作。

为确保用户和角色仍然可以使用 App Studio 控制台，还需要将 App Studio `ConsoleAccess` 或`ReadOnly` AWS 托管策略附加到实体。有关更多信息，请参阅《IAM 用户指南》中的为用户添加权限。

允许用户查看他们自己的权限

该示例说明了您如何创建策略，以允许 IAM 用户查看附加到其用户身份的内联和托管式策略。此策略包括在控制台上或使用 AWS CLI 或 AWS API 以编程方式完成此操作的权限。

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

```
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

示例 1：允许用户设置 App Studio 实例

以下示例显示了允许角色设置 App Studio 实例的基于身份的策略。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "appstudio:GetAccountStatus",
            "appstudio:GetEnablementJobStatus",
            "appstudio:StartEnablementJob",
            "appstudio:StartRollbackEnablementJob",
            "appstudio:StartTeamDeployment"
```

```
        ],
        "Resource": "*"
    }]
}
```

示例 2：拒绝用户设置 App Studio 实例

以下示例显示了一种基于身份的策略，该策略用于拒绝角色设置 App Studio 实例。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "Action": [
            "appstudio:*"
        ],
        "Resource": "*"
    }]
}
```

AWS 应用工作室合规性验证

要了解是否属于特定合规计划的范围，请参阅AWS 服务 “按合规计划划分的范围” ”，然后选择您感兴趣的合规计划。 AWS 服务 有关一般信息，请参阅AWS 合规计划AWS。

您可以使用下载第三方审计报告 AWS Artifact。有关更多信息，请参阅中的 “下载报告” 中的 “ AWS Artifact。

您在使用 AWS 服务 时的合规责任取决于您的数据的敏感性、贵公司的合规目标以及适用的法律和法规。 AWS 提供了以下资源来帮助实现合规性：

- Security Compliance & Governance：这些解决方案实施指南讨论了架构考虑因素，并提供了部署安全性和合规性功能的步骤。
- 符合 HIPAA 要求的服务参考：列出符合 HIPAA 要求的服务。并非所有 AWS 服务 人都符合 HIPAA 资格。
- AWS 合AWS 规资源 — 此工作簿和指南集可能适用于您的行业和所在地区。
- AWS 客户合规指南 — 从合规角度了解责任共担模式。这些指南总结了保护的最佳实践， AWS 服务 并将指南映射到跨多个框架（包括美国国家标准与技术研究院 (NIST)、支付卡行业安全标准委员会 (PCI) 和国际标准化组织 (ISO)）的安全控制。

- 使用AWS Config 开发人员指南中的规则评估资源 — 该 AWS Config 服务评估您的资源配置在多大程度上符合内部实践、行业准则和法规。
- AWS Security Hub— 这 AWS 服务 可以全面了解您的安全状态 AWS。Security Hub 通过安全控制措施评估您的 AWS 资源并检查其是否符合安全行业标准和最佳实践。有关受支持服务及控制措施的列表，请参阅 Security Hub 控制措施参考。
- Amazon GuardDuty — 它通过监控您的 AWS 账户环境中是否存在可疑和恶意活动，来 AWS 服务检测您的工作负载、容器和数据面临的潜在威胁。 GuardDuty 通过满足某些合规性框架规定的入侵检测要求，可以帮助您满足各种合规性要求，例如 PCI DSS。
- AWS Audit Manager— 这 AWS 服务 可以帮助您持续审计 AWS 使用情况，从而简化风险管理以及对法规和行业标准的合规性。

AWS 应用工作室的弹性

AWS 全球基础设施是围绕 AWS 区域 可用区构建的。 AWS 区域 提供多个物理隔离和隔离的可用区，这些可用区通过低延迟、高吞吐量和高度冗余的网络连接。利用可用区，您可以设计和操作在可用区之间无中断地自动实现失效转移的应用程序和数据库。与传统的单个或多个数据中心基础结构相比，可用区具有更高的可用性、容错性和可扩展性。

有关 AWS 区域 和可用区的更多信息，请参阅AWS 全球基础设施。

除了 AWS 全球基础架构外， AWS App Studio 还提供多项功能来帮助支持您的数据弹性和备份需求。

AWS App Studio 中的基础设施安全

作为一项托管服务， AWS App Studio 受亚马逊网络服务：安全流程概述白皮书中描述的 AWS 全球网络安全程序的保护。

您可以使用 AWS 已发布的 API 调用通过网络访问 App Studio。客户端必须至少支持传输层安全 (TLS) 1.2，但建议使用 TLS 1.3。客户端还必须支持具有完全向前保密 (PFS) 的密码套件，例如 DHE (Ephemeral Diffie-Hellman) 或 ECDHE (Elliptic Curve Ephemeral Diffie-Hellman)。大多数现代系统（如 Java 7 及更高版本）都支持这些模式。

此外，必须使用访问密钥 ID 和与 IAM 主体关联的秘密访问密钥来对请求进行签名。或者，您可以使用 AWS Security Token Service（AWS STS）生成临时安全凭证来对请求进行签名。

AWS App Studio 中的配置和漏洞分析

配置和 IT 控制由您（我们的客户）共同 AWS 负责。有关更多信息，请参阅责任 AWS 共担模型。

防止跨服务混淆座席

混淆代理问题是一个安全性问题，即不具有操作执行权限的实体可能会迫使具有更高权限的实体执行该操作。在中 AWS，跨服务模仿可能会导致混乱的副手问题。一个服务（呼叫服务）调用另一项服务（所谓的服务）时，可能会发生跨服务模拟。可以操纵调用服务，使用其权限以在其他情况下该服务不应有访问权限的方式对另一个客户的资源进行操作。为防止这种情况， AWS 提供可帮助您保护所有服务的数据的工具，而这些服务中的服务主体有权限访问账户中的资源。

我们建议在资源策略中使用`aws:SourceArn`和`aws:SourceAccount`全局条件上下文密钥来限制为资源提供其他服务的权限。如果您只希望将一个资源与跨服务访问相关联，请使用 `aws:SourceArn`。如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 `aws:SourceAccount`。

防范混淆代理问题最有效的方法是使用 `aws:SourceArn` 全局条件上下文键和资源的完整 ARN。如果不知道资源的完整 ARN，或者正在指定多个资源，请针对 ARN 未知部分使用带有通配符字符 (*) 的 `aws:SourceArn` 全局上下文条件键。例如 `arn:aws:servicename:*:123456789012:*`。

如果 `aws:SourceArn` 值不包含账户 ID，例如 Amazon S3 存储桶 ARN，您必须使用两个全局条件上下文键来限制权限。

`aws:SourceArn` 的值必须为 ResourceDescription。

以下示例演示如何使用 中的 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键来防范混淆代理问题。

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "servicename.amazonaws.com"
    },
    "Action": "servicename:ActionName",
    "Resource": [
      "arn:aws:servicename:::ResourceName/*"
    ],
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:servicename:*:123456789012:*"
      },
      "StringEquals": {
```

```
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

AWS App Studio 中的跨区域数据传输

AWS App Studio 跨 AWS 区域传输数据，以便在服务中启用某些生成式 AI 功能。本主题包含有关跨区域数据传输所启用的功能、跨区域移动的数据类型以及如何选择退出的信息。

以下功能由跨区域数据传输启用，如果您选择退出，则无法在您的实例中使用这些功能：

1. 使用 AI 创建应用程序，用于通过使用自然语言描述您的应用程序并为您创建资源来启动应用程序构建。
2. 应用工作室中的 AI 聊天用于询问有关应用程序构建、发布和共享的问题。

以下数据跨区域传输：

1. 前面描述的功能中的提示或用户输入。

要选择退出跨区域数据传输及其启用的功能，请使用以下步骤从控制台填写退出请求表：

1. 在以下位置打开 App Studio 控制台https://console.aws.amazon.com/appstudio/。
2. 选择退出数据传输。
3. 输入您的 AWS 账户 ID，并提供您的电子邮件地址。
4. 选择提交。
5. 提交后，您的选择退出跨区域数据传输的请求将得到处理，最长可能需要 60 天。

AWS App Studio 支持的浏览器

本主题包含有关 AWS App Studio 支持和推荐的浏览器的信息，包括对访问已发布应用程序的最终用户和应用程序构建者的浏览器支持。

支持和推荐的用于构建应用程序的浏览器

为了获得最佳的应用程序构建体验，App Studio 支持并强烈建议您使用谷歌浏览器。

> **Note**
>
> 虽然不推荐，但你也可以使用其他流行的网络浏览器（例如 Mozilla Firefox、Microsoft Edge 或适用于 macOS 的 Apple Safari）来构建应用程序，但请注意，官方不支持或验证这些浏览器，你可能需要更新设置才能访问某些构建器功能。有关更多信息，请参阅 更新浏览器设置以在 App Studio 上构建应用程序。

App Studio 不支持通过移动平台构建应用程序。

应用程序最终用户支持和推荐的浏览器

对于访问已发布应用程序的最终用户，App Studio 强烈建议使用谷歌浏览器或 Mozilla Firefox。虽然这些是推荐的浏览器，但最终用户也可以使用其他流行的网络浏览器（例如Microsoft Edge或适用于macOS的Apple Safari）访问已发布的应用程序。

最终用户还可以从移动平台访问已发布的应用程序。

更新浏览器设置以在 App Studio 上构建应用程序

App Studio 正式支持并推荐使用谷歌浏览器来构建应用程序。但是，如果您想使用其他浏览器来构建应用程序，则可能需要更新与跨站跟踪相关的某些设置或 Cookie，才能访问 App Studio 中的某些页面。

对于 Mozilla Firefox：要预览应用程序，请将以下设置更新为。`Firefox Settings > Privacy & Security > Enhanced Tracking Protection Custom > Cookies > Cross-site tracking cookies`

对于适用于 macOS 的 Apple Safari：要构建或预览应用程序，请禁用以下设置：。`Settings > Privacy > Prevent cross-site tracking`

AWS 应用工作室的配额

下表介绍了 AWS App Studio 的配额和限制。

一个 App Studio 实例中的最大应用程序数量	20
在 App Studio 实例中发布到测试或生产环境的应用程序的最大数量。同时发布到测试版和生产版的单个应用程序算作两个已发布的应用程序。	6
每个应用程序的最大托管实体数量	20
每次查询返回的最大行数	3000
每个实体的最大样本数据行数	500
自动化的最大运行时间	2 分钟。运行时间超过 2 分钟的自动化操作将失败。
最大自动化输入和输出尺寸	每个输入或输出 5GB。
自动化或数据操作使用的最大数据大小	每次自动化或数据操作运行 450 MB。
页面名称和组件名称	必须是非空且唯一的。只能包含字母、数字、下划线 (_) 和美元符号 ($)。不能包含空格。

AWS App Studio 用户指南的文档历史记录

下表介绍了 AWS App Studio 的文档版本。

变更	说明	日期
新主题：回滚应用程序的生产环境版本	添加了有关在检测到问题时将已发布的应用程序版本回滚到之前发布的版本的信息。有关更多信息，请参阅回滚到之前发布的版本。	2025 年 4 月 10 日
新主题：复制应用程序、页面和组件	添加了新主题，其中包含有关在 App Studio 中复制应用程序、页面和组件的信息。有关更多信息，请参阅复制应用程序、复制页面和复制组件。	2025 年 4 月 7 日
新主题：在 App Studio 实例之间导入和导出应用程序	添加了新主题，其中包含有关在 App Studio 实例之间导入和导出应用程序的信息，包括 App Studio 提供的可用于学习应用程序构建概念的可导入应用程序列表。有关更多信息，请参阅导入应用程序和导出应用程序。	2025年3月30日
更新的主题： AWS 托管策略： AppStudioServiceRolePolicy	更新了每项服务的`AppStudioServiceRolePolicy` 权限并添加了策略描述信息。有关更多信息，请参阅AWS 托管策略： AppStudioServiceRolePolicy。	2025 年 3 月 14 日
更新的主题：编辑或配置数据操作	添加了有关数据操作条件中使用的现有运算符和新运算符的信息，这些运算符用于从数据	2025 年 2 月 27 日

	库表中检索符合条件的数据子集。有关更多信息，请参阅编辑或配置数据操作。	
更新了主题： JavaScript 使用编写表达式	重组并添加了有关在应用程序中使用表达式引用或更新用户界面组件和表数据的信息。有关更多信息，请参阅在 App Studio 中使用 JavaScript 编写表达式。	2025 年 2 月 18 日
更新的主题：应用程序内容安全设置	添加了有关内容源应用程序内容安全设置的信息。您可以使用此设置来限制您的应用程序可以将对象上传到 Amazon S3 的域。有关更多信息，请参阅查看或更新应用程序的内容安全设置。	2025 年 2 月 14 日
新主题：在 App Studio 应用程序中调用 Lambda 函数	添加了一个简短的教程，详细介绍了如何在 App Studio 应用程序中调用 Lambda 函数。有关更多信息，请参阅调用 Lambda 函数。	2025年1月24日
新主题：Connect 到 Amazon SES	添加了创建 Amazon SES 连接器以在 App Studio 应用程序中使用该服务的说明。有关更多信息，请参阅 Connect 到 Amazon 简单电子邮件服务。	2025 年 1 月 16 日
更新的主题：首次创建和设置 App Studio 实例	添加了使用简易创建方法创建 App Studio 实例以更快地入门的说明。有关更多信息，请参阅首次创建和设置 App Studio 实例。	2024 年 12 月 13 日

新主题：管理数据依赖关系和计时问题的最佳实践	添加了有关在 App Studio 应用程序中优雅地管理数据依赖关系和计时问题的文档。有关更多信息，请参阅数据依赖关系和计时注意事项。	2024 年 11 月 20 日
更新的主题：使用 AI 编辑应用程序	添加了文档，其中包含有关在应用程序工作室中使用 AI 聊天编辑应用程序的信息。有关更多信息，请参阅使用生成式 AI 构建 App Studio 应用程序。	2024 年 11 月 18 日
更新的主题：使用 AI JavaScript 为你生成	更新了 JavaScript 自动化操作参考以包含有关使用 AI JavaScript 为您生成的信息。有关更多信息，请参阅 JavaScript 自动化操作。	2024 年 11 月 18 日
更新主题：使用 Amazon Bedrock 构建 AI 文本摘要器应用程序	更新了 Amazon Bedrock 提示教程以使用新发布的 GenAI 提示操作。有关更多信息，请参阅使用 A mazon Bedrock 构建 AI 文本摘要器应用程序。	2024 年 11 月 18 日
新主题：使用应用主题更改应用的颜色	添加了一个主题，其中包含有关使用应用程序主题更改应用程序中颜色的信息。有关更多信息，请参阅使用应用程序主题更改应用程序中的颜色。	2024 年 11 月 18 日
新主题：数据模型最佳实践	添加了一个主题，其中包含创建安全、稳健且可扩展的数据模型以用于 App Studio 应用程序的最佳实践。有关更多信息，请参阅设计数据模型时的最佳实践。	2024 年 11 月 15 日

更新的主题：连接到 AWS 服务	更新了信任策略，使其包含在内`sts:ExternalId` ，用于创建 AWS 服务连接器的 IAM 角色需要使用该策略。有关更多信息，请参阅 Connect 连接到 AWS 服务。	2024 年 11 月 13 日
新主题：回滚或恢复到之前发布的应用程序版本	添加了一个主题，其中包含有关将应用程序回滚或恢复到先前发布的版本的信息。有关更多信息，请参阅回滚到之前发布的版本。	2024 年 11 月 13 日
新主题：删除 App Studio 实例	添加了一个主题，其中包含有关删除 App Studio 实例的信息，包括如何删除实例的说明。有关更多信息，请参阅删除 App Studio 实例。	2024 年 11 月 12 日
新主题：更新应用程序内容安全设置	添加了一个主题，其中包含有关 App Studio 中应用程序内容安全设置的信息，包括如何更新这些设置。有关更多信息，请参阅查看或更新应用程序的内容安全设置。	2024 年 11 月 8 日
更新的主题： AWS App Studio 中的安全性	扩展了安全文档，包括有关数据保护以及 App Studio 如何与 IAM 交互的信息。有关更多信息，请参阅 AWS App Studio 中的安全性。	2024 年 11 月 6 日
更新的主题： AWS App Studio 中的配额	更新了 App Studio 服务配额和限制文档，修复了错误的值并删除了一些配额。有关更多信息，请参阅 AWS App Studio 中的配额。	2024 年 10 月 21 日

更新的主题：将 App Studio 连接到其他 AWS 服务	更新了连接 AWS 服务的文档，提供了有关向 App Studio 授予对服务或资源的最低必要权限的说明和示例，以更好地遵守最佳安全实践。有关更多信息，请参阅 Connect 连接到 AWS 服务。	2024 年 10 月 18 日
更新主题：为 Aurora 连接器文档添加了版本支持	在 Aurora 连接器文档中添加了支持的版本列表。有关更多信息，请参阅 Connect 到 Amazon Aurora。	2024 年 10 月 16 日
新主题：App Studio 支持的浏览器	添加了一个主题，其中包括浏览器支持和使用 App Studio 的建议。有关更多信息，请参阅支持的浏览器。	2024 年 10 月 10 日
新主题： AWS App Studio 的工作原理	添加了一个主题，介绍了 App Studio 中应用程序开发的关键概念，包括图表和屏幕截图。有关更多信息，请参阅 App Studio 的工作原理。	2024 年 10 月 10 日
新主题：对页面进行排序和整理	添加了一个主题，其中包含有关在预览或已发布应用程序的导航中重新排序以及隐藏或显示页面的信息。有关更多信息，请参阅排序和整理页面。	2024 年 9 月 24 日
新主题： AWS App Studio 中的配额	添加了一个包含与 App Studio 相关的服务配额和限制的主题。有关更多信息，请参阅 AWS App Studio 中的配额。	2024 年 9 月 11 日

更新的主题：Connect 连接到加密的 DynamoDB 表	添加了在 App Studio 中使用使用使用客户管理密钥 CMKs () 加密 AWS KMS 的 DynamoDB 表所需的权限等信息。有关更多信息，请参阅 Connect 到 DynamoDB。	2024 年 9 月 6 日
更新的主题：连接到 DynamoDB、Amazon Redshift 和 Aurora	添加了添加到 IAM 角色在 App Studio 应用程序中使用 DynamoDB、Amazon Redshift 和 Aurora 资源所需的最低权限。有关更多信息，请参阅 Connect 连接到 AWS 服务。	2024 年 9 月 5 日
更新的主题：连接亚马逊 Aurora	更新了有关创建和配置用于 App Studio 应用程序的 Amazon Aurora 数据库和表的文档。有关更多信息，请参阅 Connect 到 Amazon Aurora。	2024 年 9 月 5 日
新增和更新的主题：故障排除和调试	扩展了故障排除和调试文档，以帮助解决 App Studio 的常见问题，包括构建应用程序的调试信息。有关更多信息，请参阅故障排除和调试 App Studio。	2024 年 8 月 26 日
新主题：教程：使用 Amazon Bedrock 构建 AI 文本摘要器应用程序	按照教程中的步骤构建一个应用程序，该应用程序接收来自最终用户的输入提示，将其发送到 Amazon Bedrock，然后返回并显示摘要版本。有关更多信息，请参阅使用 A mazon Bedrock 构建 AI 文本摘要器应用程序。	2024 年 8 月 20 日

更新的主题：预览、发布和共享 App Studio 应用程序	扩展了预览、发布和共享文档的范围，以增加清晰度，与服务体验相匹配，并提供有关发布环境和在其中查看应用程序的更多信息。有关更多信息，请参阅预览、发布和共享应用程序。	2024 年 8 月 2 日
新主题：构建包含多个用户的应用程序	扩展了预览、发布和共享文档的范围，以增加清晰度，与服务体验相匹配，并提供有关发布环境和在其中查看应用程序的更多信息。有关更多信息，请参阅构建包含多个用户的应用程序。	2024 年 8 月 2 日
更新的主题：将 App Studio 连接到 AWS 服务	添加了有关创建和提供 IAM 角色以在创建其他 AWS 服务连接器时提供 AWS 资源访问权限的信息。有关更多信息，请参阅使用 “其他 AWS 服务” 连接器连接到 AWS 服务。	2024 年 7 月 29 日
更新的主题：添加有关在设置过程中创建 AWS 管理用户的说明	在设置 App Studio 文档中添加了创建管理用户来管理 AWS 资源的说明。还更新了整个连接器文档，以推荐使用该用户。	2024 年 7 月 24 日
新话题：Connect to Amazon Bedrock	添加了一个主题，其中包含如何为 Amazon Bedrock 创建连接器的说明。构建者可以使用该连接器来构建使用 Amazon Bedrock 的应用程序。有关更多信息，请参阅 Connect 到 Amazon Bedrock。	2024 年 7 月 24 日

初始版本	AWS App Studio 用户指南的初始版本	2024 年 7 月 10 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。