Red Hat Fuse 7.8

将应用程序与 Fuse 在线集成

对于希望在不同应用程序和服务之间共享数据的业务用户

Last Updated: 2023-07-26

Red Hat Fuse 7.8 将应用程序与 Fuse 在线集成

对于希望在不同应用程序和服务之间共享数据的业务用户

法律通告

摘要

Fuse Online 提供集成即服务。

目录

前言

Red Hat Fuse 是一个分布式、云原生集成解决方案，可为架构、部署和工具提供选择。Fuse Online 是红帽基于 Web 的 Fuse 发行版。Syndesis 是 Fuse Online 的开源项目。Fuse Online 在 OpenShift Online、OpenShift Dedicated 和 OpenShift Container Platform 上运行。

本指南提供了使用 Fuse 在线 Web 界面集成应用程序的信息和说明。内容组织如下：

- 第 1 章 *Fuse Online 的高级别概述*
- 第 2 章 *如何准备创建集成*
- 第 3 章 *关于您要集成的应用程序的连接*
- 第 4 章 *创建集成*
- 第 5 章 *创建由 REST API 调用触发的集成*
- 第 6 章 *创建由 HTTP 请求(Webhook)触发的集成*
- 第 7 章 *将集成数据映射到下一连接的字段*
- 第 8 章 *管理集成*
- 第 9 章 *自定义 Fuse 在线*

要了解如何通过创建示例集成来使用 Fuse Online，请参阅示例 集成指南。

要获得支持，请在 Fuse Online 的左侧导航面板中，点 **Support**，或者在右上角点  ，然后选择 **Support**。

第 1 章 FUSE ONLINE 的高级别概述

借助 Fuse Online，您可以从应用程序或服务获取数据，如果需要，请对这些数据进行操作，然后将数据发送到完全不同的应用程序或服务。无需编码即可完成此操作。

以下主题提供了 Fuse Online 的高级别概述：

1.1. FUSE ONLINE 如何工作

Fuse Online 提供了一个 Web 浏览器界面，可让您在不编写代码的情况下集成两个或多个不同应用程序或服务。它还提供了可让您在复杂用例需要时引入代码的功能。

Fuse Online 可让您在不同应用程序间进行数据传输。例如，业务分析人员可以使用 Fuse Online 来捕获提到客户，然后利用从 Twitter 获取的数据来更新 Salesforce 帐户。另一个例子是实现股票交易建议的服务。您可以使用 Fuse Online 捕获购买或销售感兴趣的推荐，并将这些建议转发到自动化股票传输的服务。

要创建并运行简单集成，主要步骤为：

1. 创建到您要集成的每个应用程序的连接。
2. 选择启动连接。此连接是包含您要与其他应用程序共享的数据的应用程序。
   或者，您可以启动与接受 HTTP 请求的计时器或 Webhook 集成。
3. 选择完成连接。此连接是从开始连接接收数据的应用程序，并完成集成。
4. 在完成连接中映射开始连接中的数据字段。
5. 为集成指定一个名称。
6. 点 **Publish** 开始运行集成。

另一种集成是 API 供应商集成。API 提供程序集成允许 REST API 客户端调用触发集成执行的命令。要创建并运行 API 供应商集成，您可以将 OpenAPI 3 （或 2）文档上传到 Fuse Online。本文档指定客户端可以调用的操作。对于每个操作，您可以指定并配置执行该操作的连接和步骤的流程，如数据映射程序或过滤步骤。简单的集成具有一个主要流，而 API 提供程序集成对每个操作都有主要流。

Fuse Online 仪表板使您可以监控和管理集成。您可以看到哪个集成正在运行，您可以启动、停止和编辑集成。

1.2. FUSE ONLINE 适合谁

Fuse Online 适用于业务专家，例如，金融、人力资源或营销，他们不希望编写代码以在两个不同的应用程序之间共享数据。他们使用各种软件即服务(SaaS)应用程序可以了解业务需求、工作流和相关数据。

作为业务用户，您可以使用 Fuse Online 进行：

- 捕获提到您的公司，对其进行过滤，并在 Salesforce 环境中创建新联系人。

- 确定 Salesforce 领导更新，然后执行 SQL 存储的流程，以保持您的相关数据库最新。
- 订阅 AMQ 代理接收的顺序，然后使用自定义 API 对这些顺序进行操作。
- 从 Amazon S3 存储桶获取数据，并将其添加到 Dropbox 文件夹中。

这些只是在没有编写代码的情况下业务用户可以执行的操作的几个示例。

1.3. 使用 FUSE ONLINE 的好处

Fuse Online 有很多优点：

- 在不编写代码的情况下集成来自不同应用程序或服务的数据。
- 在公共云或站点的 OpenShift Container Platform 上运行集成 OpenShift Online。
- 使用可视数据映射器中的数据字段到另一个应用程序中的数据字段。
- 利用开源软件的所有优势。您可以扩展功能和自定义接口。如果 Fuse Online 没有为您要集成的应用程序或服务提供连接器，则开发人员可以创建所需的连接器。

1.4. FUSE ONLINE 结构的描述

要使用 Fuse Online，您可以使用连接器、连接、操作、步骤和流来创建集成。对每个结构都有基本了解会很有帮助。

Fuse Online 的每个安装都被称为 Fuse Online 环境。当红帽安装和管理您的 Fuse Online 环境时，它会在 OpenShift Online 或 OpenShift Dedicated 上运行。当您安装和管理 Fuse Online 环境时，它通常在 OpenShift Container Platform 上运行，但可以在 OpenShift Dedicated 上运行。

集成

在 Fuse Online 中，有简单的集成和 API 供应商集成。

简单的集成是 Fuse Online 执行的一组排序步骤。这个集合包括：

- 连接到应用程序以启动集成的步骤。此连接提供集成在其上运行的初始数据。后续连接可以提供额外的数据。
- 连接到应用程序以完成集成的步骤。此连接接收之前步骤中输出并完成集成的任何数据。
- 在开始和完成连接间连接到应用程序的可选步骤。根据集成步骤序列中附加连接的位置，额外的连接可以执行以下操作：
  - 为要操作的集成提供额外的数据
  - 处理集成数据
  - 将处理结果输出到集成
- 在连接到应用程序之间的数据上执行可选步骤。通常，有一个步骤可将上一连接中的数据字段映射到下一个连接所使用的数据字段。

API 供应商集成发布您提供 OpenAPI 模式的 REST API 服务。REST API 客户端的调用会触发 API 提供程序集成的执行。该调用可以调用 REST API 实施的任何操作。虽然简单的集成有一个执行的主要流，但 API 提供程序集成对每个操作都有一个主要流。每个操作流程都会根据您在创建集成时添加到该操作流的

步骤连接到应用程序并处理数据。每个操作流都会通过返回您指定的响应返回到触发了集成执行的客户端。

连接器

Fuse Online 提供了一组连接器。连接器代表了您要从或将数据发送到的特定应用程序。每个连接器都是用于创建与该特定应用程序连接的模板。例如，您可以使用 Salesforce 连接器创建到 Salesforce 的连接。

要连接的应用程序可能会使用 OAuth 协议来验证用户。在这种情况下，您可以将 Fuse 在线环境注册为可访问该应用程序的客户端。注册与该应用程序的连接器关联。您需要只为每个使用 OAuth 的应用程序注册特定的 Fuse 在线环境。注册会扩展到您从该连接器创建的每个连接。

如果 Fuse Online 不提供您需要的连接器，开发人员可以创建所需的连接器。

连接

在创建集成前，您必须创建一个与您要从中获取数据的每个应用程序或服务的连接。要创建连接，您可以选择连接器并添加配置信息。例如，要连接到集成中的 AMQ 代理，您可以通过选择 AMQ 连接器来创建连接，然后按照提示识别要连接的代理以及用于连接的帐户。

连接是从中创建它的连接器的一个特定实例。您可以从一个连接器创建任意数量的连接。例如，您可以使用 AMQ 连接器创建三个 AMQ 连接，每个连接访问不同的代理。

要创建简单集成，您可以选择启动集成的连接、结束集成的连接，以及访问其他应用程序的一个或多个连接。要创建 API 供应商集成，您可以在每个操作流中添加一个或多个连接。任何多个集成和操作流都可以使用相同的连接。特定集成或流可以多次使用相同的连接。

详情请参阅 关于您要集成的应用程序的连接。

Actions

在集成中，每个连接都执行一个操作。当您创建集成时，您可以选择添加到流中的连接，然后选择连接执行的操作。例如，当您向流添加 Salesforce 连接时，您可以选择包括但不仅限于创建 Salesforce 帐户、更新 Salesforce 帐户和搜索 Salesforce 的操作集合。

有些操作需要额外的配置，Fuse Online 会提示您输入此信息。

步骤

简单的集成是一组排序步骤。在 API 提供程序集成中，每个操作流都是一组排序的步骤。

每个步骤都对数据进行操作。在连接到 Fuse 在线外的应用程序或服务时，一些步骤对数据进行操作。这些步骤是连接。在连接之间，可能存在其他对 Fuse Online 中的数据进行操作的步骤。通常，一组步骤包含一个步骤，它将之前连接中使用的数据字段映射到流中下一连接中使用的数据字段。除了简单集成中的启动连接外，每个步骤对它从前面的步骤收到的数据执行操作。

要在连接间操作数据，Fuse Online 提供了以下步骤：

- 将一个应用程序中的数据字段映射到另一个应用程序中的数据字段。
- 过滤数据，以便仅在处理数据满足您定义的条件时继续集成。
- 将记录集合拆分为单个记录，以便 Fuse Online 可以迭代地执行后续步骤，每条记录执行一次。
- 将单个记录聚合到一个集合中，以便 Fuse Online 在集合后执行后续步骤。
- 通过将数据插入到 Freemarker、Melstache 或 Velocity 模板，生成等同的、一致的输出。

- 除了 Fuse Online 自动提供的默认日志记录之外，日志记录信息。

要以不内置 Fuse Online 的方式在连接间操作，您可以上传一个提供自定义步骤的扩展。请参阅 开发 Fuse 在线扩展。

流

流程是集成执行的一组排序步骤。

简单的集成有一个主要流。API 提供程序集成具有 REST API 定义的每个操作的主要流。每个操作的主要流都是处理调用该操作的步骤集合。

主要流可以具有条件流。集成会评估您指定的条件，以确定是否执行其关联的流。

在流中，每个步骤都可以对前面步骤中输出的数据进行操作。要确定流中需要的步骤，请参阅 规划您的集成的注意事项。

第 2 章 如何准备创建集成

某些规划和了解创建集成的工作流可帮助您创建符合您需求的集成。以下主题提供了准备创建集成的信息。

- 第 2.1 节 “规划您的集成的注意事项”
- 第 2.2 节 “创建简单集成的常规工作流”
- 第 2.3 节 “创建 Salesforce 到数据库简单集成的工作流示例”

2.1. 规划您的集成的注意事项

在创建集成前请考虑以下问题。

如何触发集成执行？

- 是否要设置一个计时器，以按照您指定的间隔触发执行？
- 是否要发送 HTTP 请求？
- 是否要连接到应用程序以从中获取数据？
  - 在该应用程序中，什么会触发获取数据的操作？例如，从 Twitter 获取数据开始的集成可能会在 Twitter 上触发。
  - 感兴趣的数据字段是什么？
  - Fuse Online 使用什么凭证访问此应用程序？
- 您要发布 REST API 服务，以便客户端可以调用 REST API 调用，该调用会触发对某个操作执行流？
  - 已定义了服务的 OpenAPI 模式？
  - 如果没有，服务会定义什么操作？

完成简单的集成：

- 是否有应用程序接收数据或您想要向集成日志发送信息？
- 如果您要向应用程序发送数据，则集成执行什么操作？
- 感兴趣的数据字段是什么？
- Fuse Online 使用什么凭证访问此应用程序？

在流程的一组步骤中：

- 您需要访问任何其他应用程序吗？对于需要访问的任何其他应用程序：
  - 流需要连接到哪个应用程序？
  - 连接应执行什么操作？
  - 感兴趣的数据字段是什么？

    - 连接应使用哪些凭据来连接此应用？

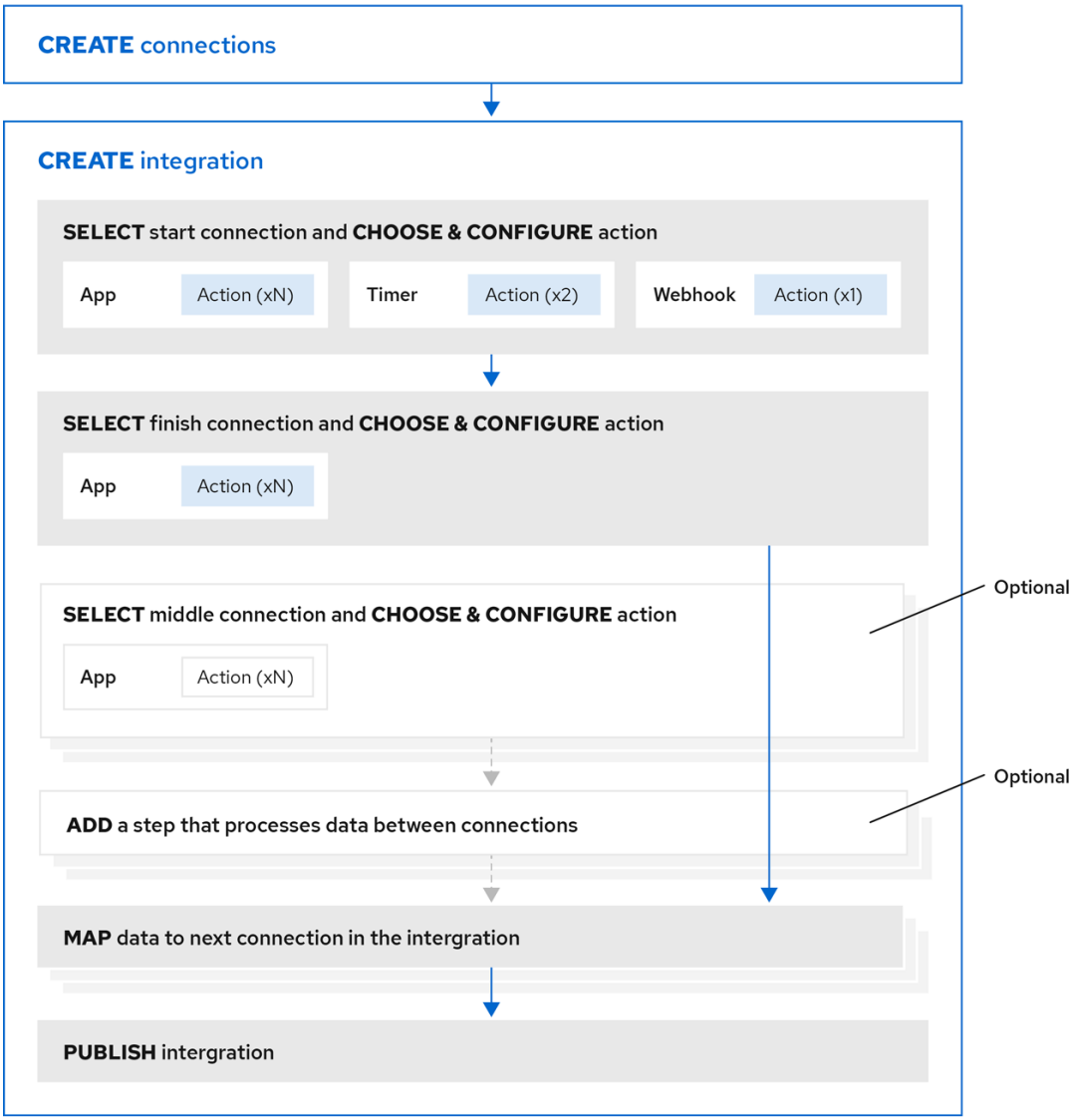
- 流是否需要在连接间的数据上运行？例如：
    - 流过滤其所在的数据吗？
    - 字段名称在源和目标应用程序之间是否不同？如果这样做，则需要数据映射。
    - 流是否在集合上运行？如果是，流可以使用数据映射器来处理集合，或者流是否需要将集合分成单独的记录？流需要将记录聚合到一个集合中？
    - 模板对以一致的形式输出数据是否有用？
    - 是否要发送有关被处理的消息信息到集成日志？
    - 流是否需要以某种自定义的方式对数据进行操作？
- 您需要根据集成数据的内容的不同执行流？也就是说，需要有条件流？

2.2. 创建简单集成的常规工作流

登录 Fuse Online 控制台后，您可以开始创建与您要集成的应用程序的连接。对于您要集成和使用 OAuth 协议的每个应用程序，请将 Fuse Online 注册为应用程序的客户端。您必须注册的应用程序包括：

- Dropbox
- Google 应用程序(Gmail、Calendar 和 Sheets)
- Salesforce
- SAP Concur
- Twitter

在为这些应用程序进行注册后，用于创建简单集成的工作流如下所示：



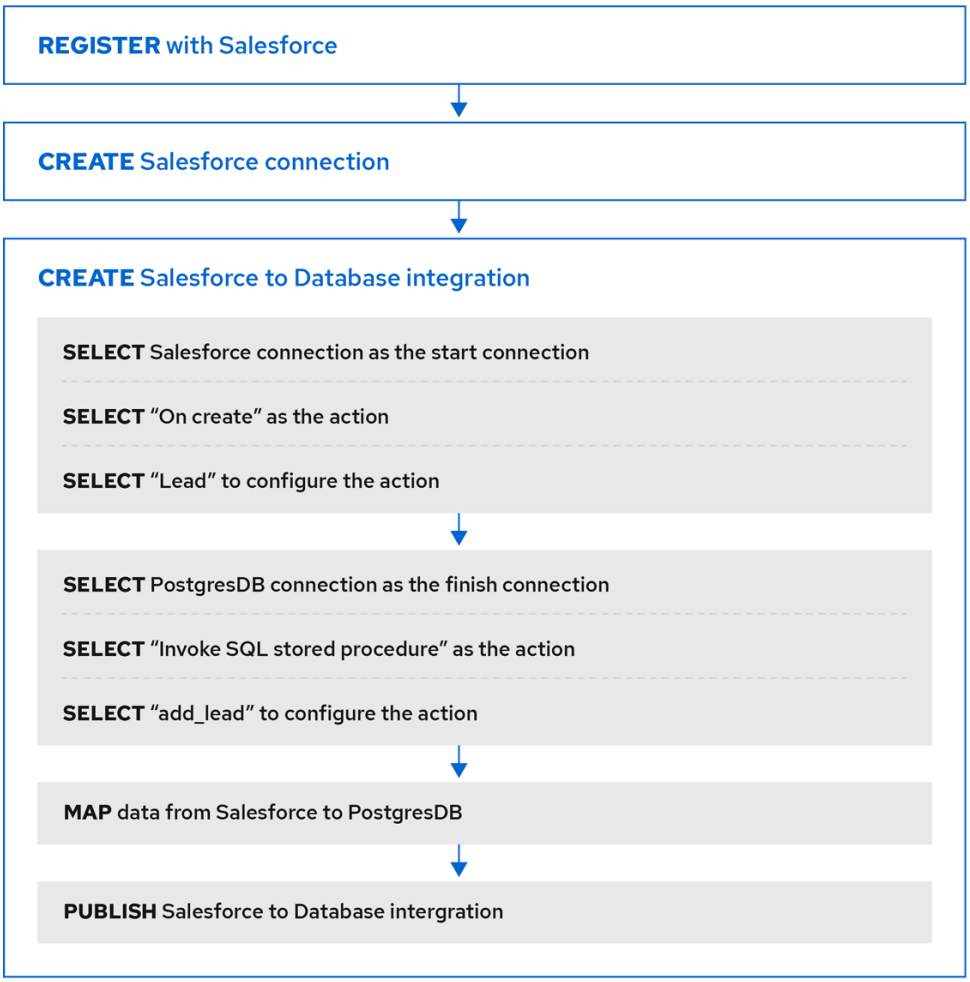
其他资源

创建 API 提供程序集成的好处、概述和工作流。

2.3. 创建 SALESFORCE 到数据库简单集成的工作流示例

了解使用 Fuse Online 创建简单集成的工作流的最佳方法是按照示例集成 教程中的说明创建示例集成。

下图显示了创建从 Salesforce 到数据库集成示例的工作流。



发布集成后，Fuse Online 仪表板会在集成准备好执行时在集成名称旁边显示 **Running**。

其他资源

导入和发布示例 API 供应商快速启动集成。

第 3 章 关于您要集成的应用程序的连接

要连接到您要集成的应用程序，主要步骤为：

1. 创建到您要集成的每个应用程序或服务的连接。
2. 创建一个集成，该集成与您要集成的每个应用程序进行连接。

创建连接的步骤因每个应用程序或服务而异。创建每种连接以及为特定集成进行配置的详情，请参见 将 Fuse 在线连接到应用程序和服务。

以下主题提供有关连接的一般信息：

- 第 3.1 节 “关于创建从 Fuse 在线到应用程序的连接”
- 第 3.2 节 “获取授权的一般流程”
- 第 3.3 节 “关于连接验证”
- 第 3.4 节 “关于在集成中添加连接”
- 第 3.5 节 “如何查看和编辑连接信息”
- 第 3.6 节 “关于从自定义连接器创建连接”

3.1. 关于创建从 FUSE 在线到应用程序的连接

要创建连接，您可以为您要连接到的应用程序选择连接器，然后在输入字段中输入值来配置与应用程序的连接。您需要为每个应用程序提供的配置详情而有所不同。配置连接后，您可以为它指定一个名称，帮助您将其与同一应用程序的任何其他连接区分开来。另外，您可以指定连接的描述。

您可以使用同一连接器来创建到该应用程序的任意数量的连接。例如，您可以使用 AMQ 连接器创建三个不同的连接。每个 AMQ 连接都可以指定不同的代理。

例如，请参阅：

- 创建 AMQ 连接
- 创建 HTTP 和 HTTPS 连接
- 创建 Slack 连接

3.2. 获取授权的一般流程

在集成中，您可能希望连接到使用 OAuth 协议验证访问请求的应用。为此，您必须注册 Fuse Online 安装，以访问该应用程序。注册授权所有从 Fuse 在线安装到给定应用程序的连接。例如，如果您通过 Salesforce 注册 Fuse 在线安装，则从 Fuse 在线安装到 Salesforce 的所有连接都使用相同的 Salesforce 客户端 ID 和注册提供的相同的 Salesforce 客户端 secret。

在每个 Fuse Online 环境中，对于使用 OAuth 的每个应用程序，只需要注册 Fuse Online，因为需要客户端。此注册可让您创建多个连接，每个连接都可以使用不同的用户凭证。

虽然您要连接到的每个 OAuth 应用程序的具体步骤有所不同，但注册总是为 Fuse Online 环境提供客户端 ID 和客户端 secret。有些应用程序将其他标签用于客户端 ID 和客户端 secret。例如，Salesforce 生成消费者密钥和消费者 secret。

对于某些 OAuth 应用程序，Fuse Online 在 **Settings** 页面中提供了一个条目，用于添加注册提供的客户端 ID 和客户端 secret。要查看此适用于哪些应用程序，请在 Fuse Online 左侧面板中点击 **Settings**。

先决条件

- 在 Fuse Online **Settings** 页面中，有一个使用 OAuth 协议授权访问的应用程序的条目。

流程概述

1. 在 Fuse Online **OAuth Application Management** 页面中，展开您要注册 Fuse Online 的应用条目。这将显示客户端 ID 和客户端 secret 字段。
2. 在您 **注册期间看到的 OAuth 应用程序管理 页面的顶部附近，输入此回调 URL：**，将该 URL 复制到剪贴板。
3. 在另一个浏览器标签页中，进入您要注册的应用程序的网站，并执行获取客户端 ID 和 secret 所需的步骤。这些步骤之一要求您输入 Fuse Online 环境的回调 URL。粘贴您在第二个步骤中复制到剪贴板的 URL。
4. 在 Fuse Online 中，在 **Settings** 页面中，粘贴客户端 ID 和客户端机密并保存设置。

其他资源

- 注册在 **Settings** 页面中具有条目的应用程序示例：
  - 将 Fuse Online 注册为 Salesforce 客户端
  - 将 Fuse Online 注册为 Twitter 客户端
- 使用在 Fuse Online **Settings** 页面中没有条目的应用程序 注册示例：将 Fuse Online 注册为 Dropbox 客户端
- 有关使用自定义连接器的信息，可让您访问使用 OAuth 协议的应用程序：关于从自定义连接器创建连接的信息。

3.3. 关于连接验证

在获得 Fuse Online 访问使用 OAuth 的应用授权后，您可以创建一个或多个与应用程序的连接。当您创建与 OAuth 应用程序的连接时，Fuse Online 会对其进行验证以确认授权已就位。在任何时候，您可以再次验证连接，以确保授权仍就位。

有些 OAuth 应用授予具有过期日期的访问令牌。如果访问令牌过期，您可以重新连接到应用程序来获取新的访问令牌。

验证使用 OAuth 或获取 OAuth 应用程序的新访问令牌的连接：

1. 在左侧面板中，单击 **Connections**。
2. 点击您要验证或要获取新访问令牌的连接。
3. 在连接的详情页面中，点 **Validate** 或点 **Reconnect**。

如果验证或重新连接失败，则检查 application/service 供应商，以确定应用的 OAuth 密钥、ID、令牌或 secret 是否仍然有效。项目可能已过期或已被撤销。

如果您发现 OAuth 项目无效、已过期或已撤销，请获取新值并将其粘贴到应用的 Fuse Online 设置中。有

关注册连接没有验证的应用程序，请参阅 将 Fuse 在线连接到 应用程序和服务中的说明。使用更新的设置时，请按照上面的说明尝试验证更新的连接。如果验证成功，并且有一个使用此连接运行的集成，请重启集成。要重启集成，请停止它，然后启动它。

如果验证失败且重新连接失败，但所有内容在服务提供商中都有效，请尝试使用应用程序重新注册 Fuse Online 环境，然后重新创建连接。在重新创建连接时，Fuse Online 会验证连接。如果您重新创建连接，且有一个使用连接的集成，则必须编辑集成以删除旧的连接并添加新连接。如果集成正在运行，则必须停止并重启它。

3.4. 关于在集成中添加连接

当您向简单集成或操作流中添加连接时，Fuse Online 会显示连接连接在连接到应用程序时可以执行的操作的列表。您必须准确选择一个操作。在运行的集成中，每个连接仅执行您选择的操作。例如，当您将 Twitter 连接添加为集成的启动连接时，您可以选择 **Mention** 操作，其中监控 Twitter 处理的 tweets。

选择某些操作会提示您指定一个或多个配置操作的参数。例如，如果您向集成添加 Salesforce 连接并选择 **On create** 操作，则必须指明创建您感兴趣的对象类型，如领导或联系。

3.5. 如何查看和编辑连接信息

创建连接后，Fuse Online 会为连接分配一个内部标识符。此标识符不会改变。您可以更改连接的名称、描述或配置值，Fuse Online 将其识别为同一连接。

查看和编辑连接信息的方法有两种：

- 在左侧面板中，点 **Connections**，然后点任何连接来查看其详情。
- 在左侧面板中，单击 **Integrations**，然后查看任何集成以查看其摘要页面。在集成流图中：
  - 对于简单集成，点连接图标查看该连接的详情。
  - 对于 API 供应商集成，点 view  以显示集成的操作列表。点击其流包含您要查看详情的连接的操作。

在 **Connection Details** 页面中，对于您要编辑的连接，点字段旁的  来编辑该字段。或者，对于某些连接，在配置字段下点 **Edit** 以更改配置值。如果更改了任何值，请务必单击 **Save**。

如果您更新集成中使用的连接，您必须重新发布集成。

对于使用 OAuth 协议授权访问的应用程序的连接，您无法更改连接使用的登录凭证。要连接到应用程序并使用不同的登录凭证，您必须创建新连接。

3.6. 关于从自定义连接器创建连接

上传定义自定义连接器的扩展后，可以使用自定义连接器。您可以使用自定义连接器来创建连接，这与使用 Fuse 在线提供的连接器创建连接的方式相同。

自定义连接器可能适用于使用 OAuth 协议的应用程序。在从这类连接器创建连接前，您必须注册 Fuse Online 环境，以访问连接器所用于的应用程序。您可以在连接器用于的应用程序的接口中完成此操作。如何注册 Fuse Online 环境的详细信息因每个应用程序而异。

例如，假设自定义连接器用于创建与 Yammer 的连接。您需要通过在 Yammer 内创建新应用程序来注册 Fuse 在线环境。注册为 Fuse Online 提供 Yammer 客户端 ID，为 Fuse Online 提供 Yammer 客户端 secret 值。从 Fuse Online 环境到 Yammer 的连接必须提供这两个值。

请注意，应用程序可能会为这些值使用不同的名称，如消费者 ID 或消费者 secret。

注册 Fuse Online 环境后，您可以创建与应用程序的连接。在配置连接时，应该有输入客户端 ID 和客户端 secret 的参数。如果这些参数不可用，您需要与扩展开发人员通信，并询问一个更新的扩展，可让您指定客户端 ID 和客户端 secret。

第 4 章 创建集成

在进行一些规划和准备后，您已准备好创建集成。在 Fuse Online Web 界面中，当您单击 **Create Integration** 时，Fuse Online 指导您完成创建集成的步骤。

先决条件

- 规划您的集成的注意事项
- 根据您要创建的集成类型：
  - 了解 用于创建简单集成的常规工作流
  - 了解 创建 API 供应商集成的一般工作流

以下主题提供了创建集成的信息和说明：

- 第 4.1 节 “准备创建集成”
- 第 4.2 节 “触发集成执行的替代方案”
- 第 4.3 节 “创建简单集成的常规流程”
- 第 4.4 节 “添加计时器连接来触发集成执行”
- 第 4.5 节 “数据位于集合中时的集成行为”
- 第 4.6 节 “关于在连接间添加步骤”
- 第 4.7 节 “评估集成数据以确定执行流”
- 第 4.8 节 “添加数据映射程序步骤”
- 第 4.9 节 “添加基本过滤器步骤”
- 第 4.10 节 “添加高级过滤器步骤”
- 第 4.11 节 “添加模板步骤”
- 第 4.12 节 “添加自定义步骤”

4.1. 准备创建集成

创建集成的准备首先是 规划您集成的注意事项 中所列问题的答案。在进行了集成计划后，您需要先进行以下操作，然后才能创建集成：

1. 确定您要连接的应用程序是否使用 OAuth 协议。对于使用 OAuth 的每个应用程序，将 Fuse Online 注册为有权访问该应用程序的客户端。使用 OAuth 协议的应用程序包括：
   - Dropbox
   - Google 应用程序(Gmail、Calendar 和 Sheets)
   - Salesforce
   - SAP Concur

- Twitter

2. 确定您要连接的应用程序是否使用 HTTP 基本身份验证。对于每个操作的应用程序，标识用于访问该应用程序的用户名和密码。创建连接时需要提供此信息。

3. 对于您要集成的每个应用程序，请创建一个连接。

其他资源

- 获取授权的一般流程
- 关于创建连接

4.2. 触发集成执行的替代方案

当您创建集成时，集成中的第一步决定了如何触发集成。集成的第一个步骤可以是以下之一：

- **连接到应用程序或服务**。您可以为特定应用程序或服务配置连接。示例：
  - 当一个包含您指定的文本时，与 Twitter 的连接可以监控 tweets 并触发简单集成的执行。
  - 当任何人创建新领导时，到 Salesforce 的连接可触发简单的集成执行。
  - 与 AWS S3 的连接可以定期轮询特定存储桶，并在存储桶包含文件时触发简单的集成执行。
- **计时器**.Fuse Online 会在您指定的间隔内触发简单集成的执行。这可以是一个简单的计时器或 **cron** 作业。
- **Webhook**.客户端可以将 HTTP **GET** 或 **POST** 请求发送到 Fuse 在线公开的 HTTP 端点。该请求会触发对简单集成的执行。
- **API 提供程序**.API 提供程序集成从 REST API 服务开始。此 REST API 服务由您在创建 API 供应商集成时提供的 OpenAPI 3 （或 2）文档定义。发布 API 供应商集成后，Fuse Online 在 OpenShift 上部署 REST API 服务。任何对集成端点有网络访问的客户端都可能会触发集成的执行。

4.3. 创建简单集成的常规流程

Fuse Online 指导您完成创建简单集成的步骤。它会提示您选择启动连接、完成连接、可选连接和其他步骤。集成完成后，您可以发布它使其正在运行，或者您可以稍后保存以进行发布。

要了解创建 API 供应商集成的步骤，请参阅 第 5.3 节 “创建 API 供应商集成”。

先决条件

- 您制定了集成步骤的计划。
- 您创建了与此集成中要连接到的每个应用程序或服务的连接。

流程

1. 在 Fuse Online 左侧面板中，单击 **Integrations**。
2. 点 **Create Integration**。

3. 选择并配置启动连接：

   a. 在 **Choose a connection** 页面上，单击您要用来启动集成的连接。当此集成运行时，Fuse Online 将连接到此应用程序，并获取您希望集成运行的数据。

   b. 在 **Choose an action** 页面中，选择您希望此连接执行的操作。每个连接都有可用的操作会有所不同。

   c. 在用于配置操作的页面中，在字段中输入值。

   d. 另外，如果连接需要数据类型规格，Fuse Online 会提示您点 **Next** 来指定操作的输入和输出类型。

   e. 点 **Next** 添加启动连接。

   作为连接到应用程序的替代选择，启动连接可以是计时器，可按照您指定的间隔触发集成执行，或者可以是接受 HTTP 请求的 webhook。

   + 在选择并配置启动连接后，Fuse Online 会提示您选择完成连接。

4. 选择并配置完成连接：

   a. 在 **Choose a connection** 页面上，点击您要用来完成集成的连接。当此集成运行时，Fuse Online 将与集成所操作的数据连接到此应用程序。

   b. 在 **Choose an action** 页面中，选择您希望此连接执行的操作。每个连接都有可用的操作会有所不同。

   c. 在用于配置操作的页面中，在字段中输入值。

   d. 另外，如果连接需要数据类型规格，Fuse Online 会提示您点 **Next** 来指定操作的输入和输出类型。

   e. 点 **Next** 添加完成连接。

   作为连接到应用程序的替代选择，结束连接可以向集成处理的消息发送信息。为此，请选择 **Log** when Fuse Online 提示您选择完成连接。

5. （可选）在启动连接和完成连接之间添加一个或多个连接。对于每个连接，选择其操作并输入任何所需的配置详情。

6. 另外，还可添加对连接之间的集成数据操作的一个或多个步骤。请参阅关于在连接之间添加步骤。

7. 在集成视觉化中，查找任何  图标。这些警告表示在此连接前需要数据映射程序步骤。添加所需的数据映射程序步骤。

8. 当集成包含所有必要的步骤时，根据您要开始运行集成，点 **Save** 或 **Publish**。

9. 在 **Name** 字段中输入可区分此集成的名称。

10. （可选）在 **Description** 字段中输入描述信息，例如，您可以指示这个集成的作用。

11. 另外，从您导入的库扩展列表中，您可以选择一个或多个库扩展来与集成关联。请注意，如果想要显示在此列表中，则必须已经将库 **.jar** 文件导入为 Fuse Online 扩展，以便您可以选择它。有关库扩展的更多信息，请参阅如何开发库扩展。

12. 如果您准备开始运行集成，请单击 **Save 并发布**。
Fuse Online 显示集成摘要。您可以看到 Fuse Online 正在发布它。可能需要稍等片刻，集成的状态才会变为 **Running**。

    如果您不想发布集成，请单击 **Save**。Fuse Online 保存集成并显示其流视觉化。您可以继续编辑它。或者，在页面顶部的面包屑导航栏中，单击 **Integrations** 以显示集成列表。如果您保存但没有发布集成，则 **Stopped** 会出现在集成的条目中。

4.4. 添加计时器连接来触发集成执行

要根据您指定的调度触发集成执行，请将计时器连接添加为简单集成的启动连接。计时器连接不能位于流的中间或流结束时。

流程

1. 在 Fuse Online 中，单击左侧的 **Integrations**。
2. 点 **Create Integration**。
3. 在 **Choose a connection** 页面上，单击 **Timer**。
Fuse Online 提供了一个 **Timer** 连接；您不需要创建计时器连接。
4. 在 **Choose an action** 页面上，选择 **Cron** 或 **Simple**。
    - **cron** 计时器需要一个 **cron** 表达式，用于指定触发集成执行的调度。
    - 一个简单的计时器会提示您输入一个句点及其时间单位，例如 **5 秒**、**1 小时**。可用的单位为毫秒、秒、分钟、小时、天。
5. 根据您要添加的计时器类型，输入 **cron** 表达式或带有所选时间单位的句点。
6. 点 **Next** 将 **Timer** 连接添加为集成的启动连接。

4.5. 数据位于集合中时的集成行为

有时，连接会返回一个集合，其中包含所有相同类型的多个值。当连接返回集合时，流可以通过多种方式在集合上运行，包括：

- 为集合执行一次每个步骤。
- 对集合中的每个元素执行一次。
- 对集合执行一些步骤，并对集合中的每个元素执行一次其他步骤。

要确定如何在流中的集合上运行，您需要知道流连接到哪些应用程序、它们是否可以处理集合，以及您希望流完成的内容。然后，您可以使用以下主题中的信息在处理集合的流中添加步骤：

- 第 4.5.1 节 “关于数据映射程序中的数据类型和集合”
- 第 4.5.2 节 “关于处理集合”
- 第 4.5.3 节 “使用数据映射器来处理集合”
- 第 4.5.4 节 “添加分割步骤”
- 第 4.5.5 节 “添加聚合步骤”

- 第 4.5.6 节 “在流中处理集合示例”

4.5.1. 关于数据映射程序中的数据类型和集合

在数据映射程序中，字段可以是：

- 存储单个值的 **原语** 类型。原语类型的示例包括 **布尔值**、**char**、**字节**、短语、**int**、长、**浮点** 和 **双引号**。**原语类型无法扩展，因为它是一个字段。**

- **由不同类型的多个字段组成 的复杂 类型。您可以在设计时定义复杂类型的子字段。在数据映射程序中，一个复杂的类型可以被扩展，以便您可以查看其子字段。**

每种类型的字段（原语和复杂）也可以是集合。集合是一个可以具有多个值的单个字段。集合中的项目数量在运行时决定。在设计时，在数据映射程序中，集合由

表示。在数据映射程序界面中可以扩展集合是否由其类型决定。当集合是一个原语类型时，它不可扩展。当集合是一个复杂的类型时，数据映射程序可以被扩展，以显示集合的子字段。您可以从/到每个字段映射。

以下是一些示例：

- **ID 是原语类型字段(int)。在运行时，员工只能有一个 ID。例如，ID=823。因此，ID 是不是集合的原语类型。在数据映射程序中，ID 无法扩展。**

- 电子邮件 **是一个原语类型字段（字符串）。在运行时，员工可以有多个 电子邮件 值。例如： email<0>=aslan@home.com 和 email<1>= aslan@business.com。因此，电子邮件也是集合的原语类型。数据映射程序使用**

  表示 电子邮件 字段是一个集合，但 电子邮件 无法扩展，因为它是一个原语类型（没有子字段）。

- 员工 **是一个复杂的对象字段，它具有多个子字段，包括 ID 和 电子邮件。在运行时，员工 也是集合，因为公司有许多员工。**
  在设计时，数据映射程序使用

  来表示 员工 是一个集合。employee 字段可以扩展，因为它是一个具有子字段的复杂类型。

4.5.2. 关于处理集合

处理集合的过程的最简单方法是使用数据映射器将源集合中的字段映射到目标集合中的字段。对于许多流，这都是需要的。例如，流可能会从数据库获取一组员工记录，然后将这些记录插入到电子表格中。

在数据库连接和 Google Sheets 连接之间，数据映射器步骤会将数据库字段映射到 Google Sheets 字段。由于源和目标都是集合，因此当 Fuse Online 执行流时，它将调用 Google Sheets 连接一次。在该调用中，Fuse Online 会迭代记录并正确填充电子表格。

在某些流中，您可能需要将集合分成单独的对象。例如，假设一个流连接到一个数据库，并获取一组员工，如果这些员工在特定日期之前不使用此时间，则这些员工将丢失。然后，流程需要向这些员工发送电子邮件通知。在此流中，您将在数据库连接后添加分割步骤。然后，您将添加一个数据映射程序步骤，它将员工记录的 source 字段映射到发送消息的 Gmail 连接中的 target 字段。当 Fuse Online 执行流时，它会为每个员工执行数据映射程序步骤以及 Gmail 连接一次。

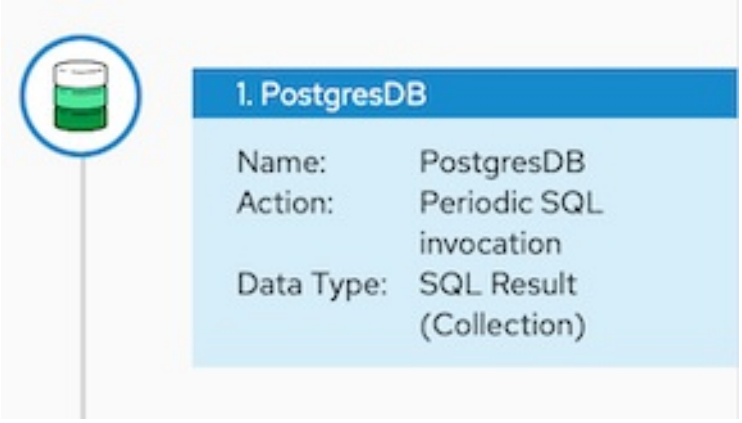
有时，在分割流中的集合后，以及对集合中的每一元素执行一次步骤后，您希望流再次对集合操作。考虑前面段落中的示例。假设 Gmail 连接在向每个员工发送电子邮件后，您想要添加通知给电子表格的员工列表。在这种情况下，在 Gmail 连接后，添加一个聚合步骤来创建一组员工名称。然后，添加一个数据映射器步骤，将源集合中的字段映射到目标 Google Sheets 连接中的字段。当 Fuse Online 执行流时，它会执行新的数据映射程序步骤，以及一次 Google Sheets 连接。

这些是在流中处理集合的最常见场景。但是，也可以进行更复杂的处理。例如，当集合中的元素本身是集合时，您可以在其他分割和聚合步骤中嵌套并汇总步骤。

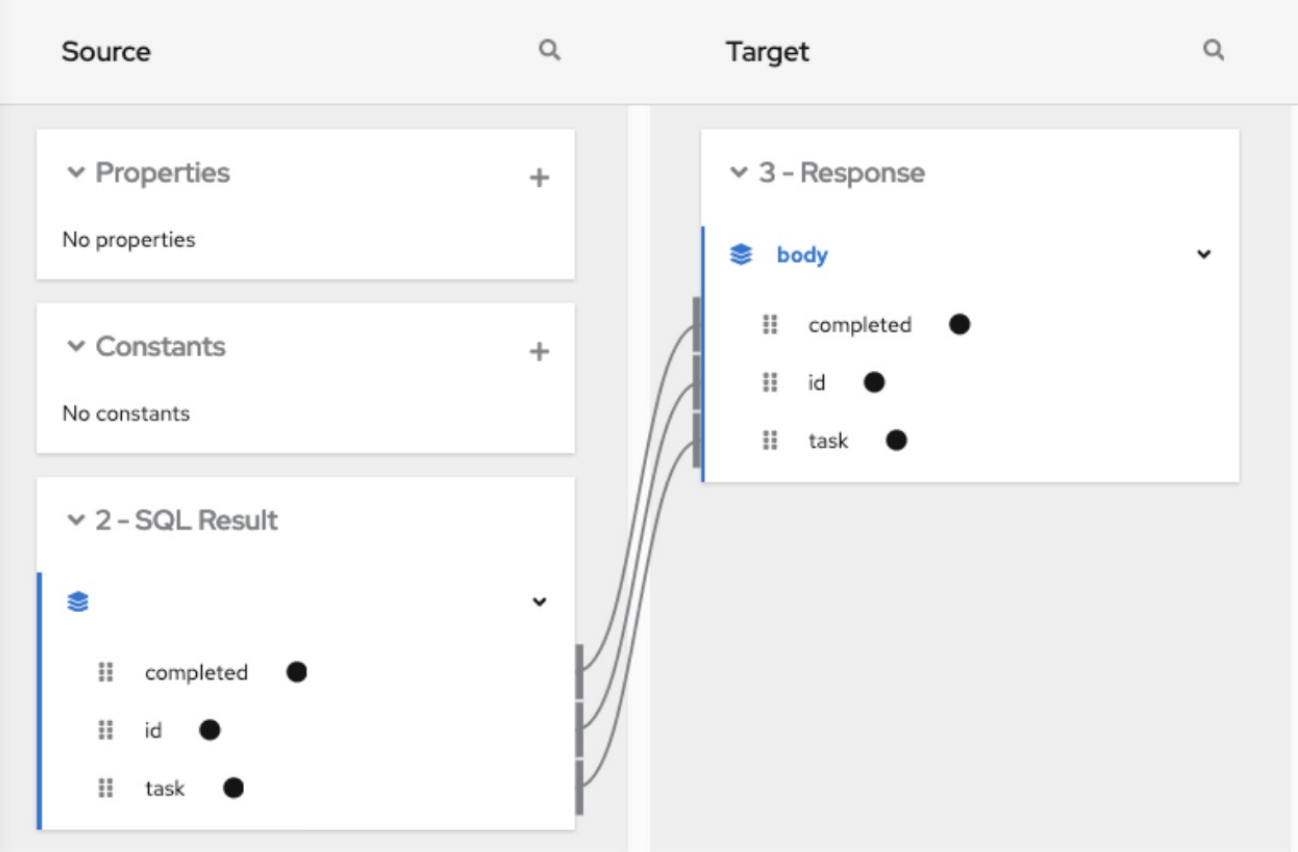
4.5.3. 使用数据映射器来处理集合

在流中，当步骤输出集合以及流中的后续连接时，您可以使用数据映射器来指定流如何处理集合。

当步骤输出集合时，流程视觉化会在步骤的详情中显示 Collection。例如：



在提供集合的步骤后添加数据映射程序步骤，并在需要映射的步骤之前添加。准确在这个数据映射程序步骤中，需要依赖于流程中的其他步骤。下图显示了从源集合字段到目标集合字段的映射：



在源和目标面板中，数据映射器会显示

来表示集合。

当集合是一个复杂的类型时，数据映射器会显示集合的子字段。您可以从/到每个字段映射。

当 source 字段嵌套在多个集合中时，您可以将其映射到满足其中一个条件的目标字段：

- target 字段嵌套在与 source 字段相同的集合数量中。例如，允许这些映射：
  - /A<>/B<>/C → /D<>/E<>/F
  - /A<>/B<>/C → /G<>/H/I<>/J

- target 字段仅嵌套在一个集合中。例如，允许这个映射：

  /A<>/B<>/C → /K<>/L

  在这种情况下，数据映射器使用深度优先算法来迭代源中的所有值。因此，数据映射程序会将源值放入单个目标集合中。

不允许以下映射：

/a<>/B<>/C cannot-map-to /M<>/N/O<>/P<>/Q

当 Fuse Online 执行流时，它会迭代源集合元素来填充目标集合元素。如果您将一个或多个源集合字段映射到目标集合字段或目标集合字段，则目标集合元素仅包含映射字段的值。

如果您将源集合中的字段映射到不在集合中的 target 字段，那么当 Fuse Online 执行流时，它将只分配源集合中最后一个元素的值。该映射步骤中会忽略集合中的任何其他元素。但是，任何后续映射步骤都可以访问源集合中的所有元素。

当连接返回 JSON 或 Java 文档中定义的集合时，数据映射器通常可以将源文档处理为集合。

4.5.4. 添加分割步骤

在执行流期间，当连接返回对象集合时，Fuse Online 会为集合执行后续步骤。如果要对集合中的每个对象执行一次后续步骤，请添加分割步骤。例如，Google Sheets 连接返回一系列行对象。要为每个行执行后续步骤，请在 Google Sheets 连接后添加拆分步骤。

确保到分割步骤的输入始终是一个集合。如果分割步骤获得一个不是集合类型的源文档，则步骤会分割每个空格的输入。例如，Fuse Online 将"Hello world!"输入分成两个元素："Hello"和"world!"，并将这两个元素传递给流中的下一步。特别是，XML 数据不是集合类型。

先决条件

- 您正在创建或编辑流。

- 流已具有需要的所有连接。
- 在流视觉化中，获取源数据的连接表示数据是一个 (Collection)。

流程

1. 在流视觉化中，点您要添加分割步骤的位置

。
2. 单击 Split。此步骤不需要任何配置。
3. 点击 Next。

附加信息

通常，您要在添加数据映射程序步骤前添加任何分割步骤和聚合步骤。这是因为数据是集合还是单个对象会影响映射。如果您添加了数据映射程序步骤，然后添加分割步骤，则通常需要重做映射。同样，如果您删除了分割或聚合步骤，则需要重做任何映射。

4.5.5. 添加聚合步骤

在流中，添加一个聚合步骤，其中您希望 Fuse Online 从单个对象创建集合。在执行聚合步骤后，在聚合步骤后，Fuse Online 会对每个对象执行一次后续步骤。

在决定是否为流中添加聚合步骤时，请考虑流中的连接。在分割步骤后，对于后续连接，Fuse Online 会针对流数据中的每个元素连接到该应用程序。对于某些连接，最好连接一次，而不是多次。

先决条件

- 您正在创建或编辑流。
- 流已具有需要的所有连接。

- 上一步将集合分成单独的对象。

流程

1. 在流视觉化中，您要在流中添加聚合步骤，点  。
2. 单击 Aggregate。此步骤不需要任何配置。
3. 点击 Next。

附加信息

通常，您要在添加数据映射程序步骤前添加任何分割和聚合步骤。这是因为数据是集合还是单个对象会影响映射。如果您添加了数据映射程序步骤，然后添加聚合步骤，则通常需要重做映射。同样，如果您删除了聚合步骤，则需要重做任何映射。

4.5.6. 在流中处理集合示例

这个简单集成从 Fuse Online 提供的示例数据库中获取一系列任务。流将集合分成单独的任务对象，然后过滤这些对象以查找已完成的任务。然后，流聚合集合中完成的任务，将该集合中的字段映射到电子表格中的字段，并通过向电子表格中添加完成的任务列表来完成。

以下流程提供了创建此简单集成的说明。

先决条件

- 您创建了 Google Sheets 连接。
- 在 Google Sheets 连接访问的帐户中，有一个接收数据库记录的电子表格。

流程

1. 点 Create Integration。

2. 添加启动连接：

   a. 在 Choose a connection 页面上，单击 PostgresDB。

   b. 在 Choose an action 页面中，选择 Periodic SQL Invocation。

   c. 在 SQL Statement 字段中，输入 select * from todo 并点 Next。

   此连接返回任务对象的集合。

3. 添加完成连接：

   a. 在 Choose a connection 页面中，点 Google Sheets 连接。

   b. 在 Choose an action 页面上，选择 Append values to a sheet。

   c. 在 SpreadsheetId 字段中，输入电子表格的 ID，以将任务列表添加到其中。

   d. 在 Range 字段中，输入 A:B 作为您要附加值的目标列。第一列 A 用于任务 ID。第二列 B 用于任务名称。

   e. 接受 Major Dimension 和 Value Input Option 的默认值，然后单击 Next。

   Google Sheets 连接通过将集合中的每个元素添加到电子表格来完成流。

4. 在流中添加分割步骤：

a. 在流视觉化中，点加号。

b. 单击 **Split**。

在流程执行分割步骤后，结果是一组单独的任务对象。Fuse Online 为每个单个任务对象执行一次流程中的后续步骤。

5. 在流中添加过滤器步骤：

a. 在流视觉化中，在分割步骤后点加号。

b. 点击 **Basic Filter** 并配置过滤器，如下所示：

i. 点击第一个字段并选择 **completed**，这是包含您要评估的数据的字段名称。

ii. 在第二个字段中，选择 等于 为 完成 字段值必须满足的条件。

iii. 在第三个字段中，指定 1 作为必须在 完成字段中的值。1 表示该任务已完成。

c. 点击 **Next**。

在执行期间，流为每个任务对象执行一次过滤器步骤。结果是一组单独的完成的任务对象。

6. 在流中添加聚合步骤：

a. 在流视觉化中，在过滤步骤后点加号。

b. 单击 **Aggregate**。

现在，结果集包含一个集合，其中包含每个完成的任务的一个元素。

7. 在流中添加数据映射程序步骤：

   a. 在流视觉化中，在聚合步骤后点加号。

   b. 点 Data Mapper，将 SQL 结果源集合中的以下字段映射到 Google Sheets 目标集合中：

      - ID 到 A
      - 任务 到 B

   c. 点 Done。

8. 单击 Publish。

结果

当集成运行时，它会每分钟从示例数据库中获取任务，然后将完成的任务添加到电子表格中的第一表格中。集成将任务 ID 映射到第一列 A，并将任务名称映射到第二列 B。

4.6. 关于在连接间添加步骤

虽然这不是要求，但建议是将所有需要的连接添加到主流中，然后根据您要执行的处理，在连接之间添加额外的步骤。在流中，每个步骤都对从上一连接获取的数据以及前面的任何步骤运行。生成的数据可用于流中的下一步。

通常，您必须映射从连接收到的数据字段和流中下一连接可以操作的数据字段。将所有连接添加到流后，检查流视觉化。对于需要数据映射的每个连接，然后再对输入数据进行操作，Fuse Online 会显示  。点此图标查看 Data Type Mismatch: Add a data mapper step before this connection to resolve difference。

您可以单击消息中的链接，以显示您添加的 Configure Mapper 页面，并指定数据映射程序步骤。但是，建议添加其他必要的步骤，然后最后添加数据映射程序步骤。

4.7. 评估集成数据以确定执行流

在流中，条件 Flows 步骤会根据您指定的条件评估集成数据。对于每个指定条件，您可以在与该条件关联的流中添加连接和其他步骤。在执行过程中，条件流 步骤评估传入的数据，以确定要执行哪些流。

以下主题提供详情：

- 第 4.7.1 节 “条件流 步骤的行为”
- 第 4.7.2 节 “条件流 步骤示例”
- 第 4.7.3 节 “配置 条件流步骤的一般 流程”
- 第 4.7.4 节 “使用基本表达式构建器指定条件”
- 第 4.7.5 节 “使用高级表达式构建器指定条件”
- 第 4.7.6 节 “在条件流中添加步骤”

4.7.1. 条件流 步骤的行为

在集成开发过程中，您可以在流中 添加条件 流步骤，并定义一个或多个条件。对于每个条件，您可以向仅与该条件关联的条件流中添加步骤。在集成执行期间，对于之前集成步骤的所有消息都传递到 Conditional Flows 步骤，条件 Flows 步骤会根据您在 Fuse Online 页面中定义它们的顺序评估消息内容，以指定条件。

在 Conditional Flows 步骤中，行为是以下之一：

- 对于评估为 true 的第一个条件，集成会执行与该条件关联的条件流。

- 如果没有条件评估为 true，并且有一个默认条件流，则集成会执行该流。
- 如果没有条件评估为 true，且没有默认条件流，则集成不会执行条件流。

执行条件流后，或者在没有条件评估为 true 且没有默认条件流后，集成会在主流中执行下一步。

4.7.2. 条件流 步骤示例

假设集成连接到 SQL 数据库，以获取每个员工的付费时间(PTO)的信息。返回的数据表示：

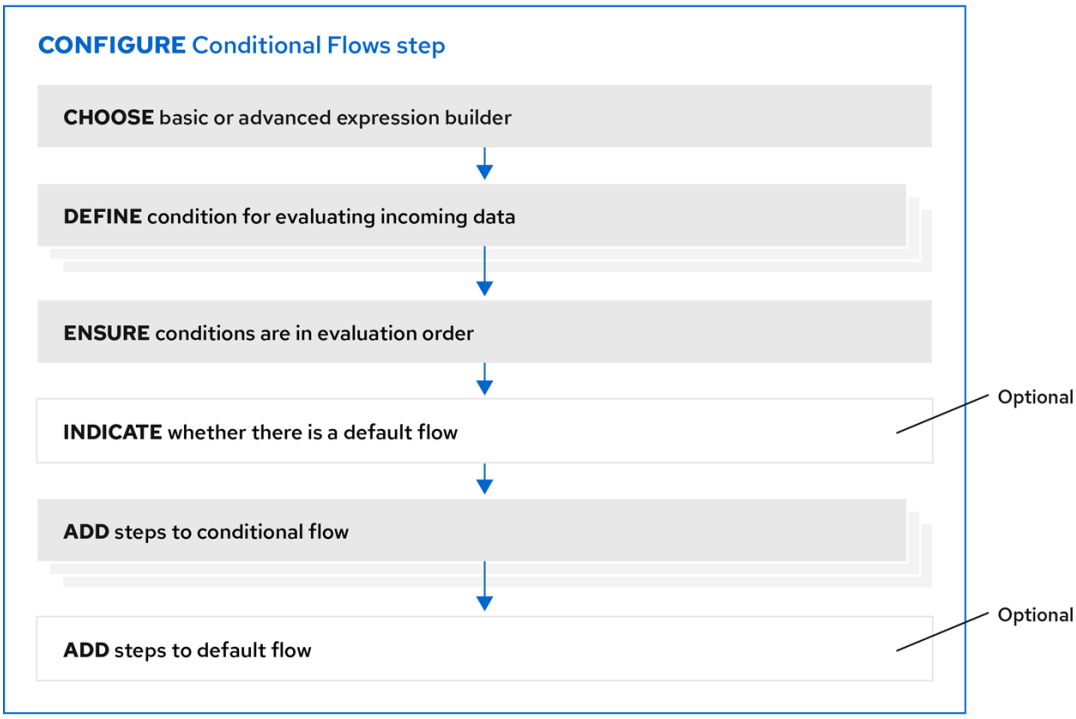
- 如果某些员工因特定日期不使用它，则他们可能会丢失 PTO。
- 其他员工已使用了比获得更多的 PTO。
- 其余的员工有 PTO 可以无时间限制地使用 PTO。

在 Conditional Flows 步骤中，此示例集成可以定义两个条件，每个条件都有一个执行流，以及默认的执行流：

- 当 PTO 大于一些数字时，这表示某些 PTO 在特定日期没有用时可能会丢失。当此条件评估为 true 时，集成会执行流，将电子邮件发送到受影响的员工。电子邮件包含必须使用的 PTO 量以及必须使用的日期。
- 当 PTO 为负数时，这表示已使用一些 PTO，但没有获得。当此条件评估为 true 时，集成会执行流，向受影响的员工发送电子邮件。该电子邮件包含员工已超过原来的 PTO 量，并指定员工再次开始进入 PTO 的日期。
- 当两个条件都没有评估为 true 时，集成会执行默认流。这个示例集成会为 PTO 不是一个负数或一些指定数量的员工执行默认条件流。默认流向员工发送一封电子邮件，并声明员工拥有的 PTO 数量。

4.7.3. 配置 条件流步骤的一般 流程

在流中添加 Conditional Flows 步骤后，配置步骤的工作流如下：



有关工作流的更多信息

- 基本表达式构建器提示您输入包含您要评估的内容的属性，以及您要测试的条件和值。基本表达式构建器适合大多数 条件流 步骤。
- 高级表达式构建器允许您在 Camel Simple Language 中指定条件表达式。
- 所有条件都必须使用相同的表达式构建器。换句话说，若要配置 条件流 步骤，您必须使用基本表达式构建器或高级表达式构建器。您不能同时使用两者。
- 在条件流中，您无法添加条件 流 步骤。

4.7.4. 使用基本表达式构建器指定条件

在流中，当您要评估传入的数据时，添加一个 Conditional Flows 步骤来确定集成的执行路径。此处描述的步骤演示了如何使用基本表达式构建器指定条件。

先决条件

- 您是创建或编辑主要流。如果这是一个简单的集成，则添加了开始和完成连接。
- 到 条件流 步骤的输入必须是单个消息。在集成视觉化中，如果上一步的数据类型显示 (Collection)，请在上一步后添加 Split 步骤，在此条件 流 步骤前添加 Split 步骤。
- 熟悉集成传递给您要添加的 Conditional Flows 步骤的消息中的字段。

流程

1. 在集成视觉化中，您要添加 条件流 步骤，点  。
2. 点 Conditional Flows。
3. 在 Basic expression builder 条目中，单击 Select。
4. 在 Configure Conditional Flows 页面中，定义一个或多个条件：
   a. 点初始 When 字段。
   b. 在属性列表中，点包含您想要 条件 Flows 步骤评估的内容的属性。
   c. 在下一字段中，接受 Contains 作为步骤评估数据或选择另一个条件的条件。您在此字段中选择的条件必须评估为 true，用于您在下一字段中输入的值。
   d. 在第三个字段中，指定条件测试的值。

e. 可选的。点 Add another condition 指定另一个条件。

f. 对您要定义的每个额外条件重复这组步骤。

g. 可选的。点条件右侧的up 或 down 箭头更改集成评估定义条件的顺序。

h. 可选的。如果要有一个默认条件流，请单击 Execute default flow。

如果您选择 Execute 默认流，在执行期间，如果没有指定条件评估为 true，则集成将执行默认条件流。如果您没有选择 Execute 默认流，在执行期间，如果没有指定评估为 true 的条件，则集成将继续执行，执行遵循这个 条件流 步骤。

5. 点击 Next。

6. 可选的。如果 Fuse Online 提示输入它，请指定输出数据类型。作为此条件流 步骤的一部分的所有条件流 必须具有相同的输出类型。

7. 点击 Next。

Fuse Online 显示流视觉化。在您添加的 Conditional Flows 步骤下方，指定了每个条件都有一个条目，如果您表示 Conditional Flows 步骤有一个默认流，则另外一个另外一个条件的默认流。

后续步骤

对于每个条件，在关联的流中添加步骤。如果存在默认流，在默认流中添加步骤。

其他资源

- 有关您可以在每个条件的中间字段中选择的条件的详情，请参阅 Camel Simple Language operator。请注意，匹配条件与 Simple Language regex 运算符对应。
- 如果您无法使用基本表达式构建器定义需要的条件，请参阅使用高级表达式构建器指定条件。

4.7.5. 使用高级表达式构建器指定条件

在流中，当您要评估传入的数据时，添加一个 Conditional Flows 步骤来确定集成的执行路径。此处描述的步骤演示了如何使用高级表达式构建器在 Camel Simple Language 中指定条件表达式。

先决条件

- 您是创建或编辑主要流。如果这是一个简单的集成，则添加了开始和完成连接。
- 到 条件流 步骤的输入必须是单个消息。在集成视觉化中，如果上一步的数据类型显示 (Collection)，请添加 Split 步骤。
- 熟悉集成传递给您要添加的 Conditional Flows 步骤的消息中的字段。
- 熟悉 Camel Simple Expression 语言或具有要评估的条件的表达式。

流程

1. 在集成视觉化中，您要添加 条件流 步骤，点  。
2. 点 Conditional Flows。
3. 在 Advanced expression builder 条目中点 Select。
4. 在 Configure Conditional Flows 页面中，定义一个或多个条件：
   a. 在初始 When 字段中，输入 Camel Simple Language 条件表达式。例如，当消息的正文包含大于 160 的 pto 字段时，以下表达式评估为 true：

```
${body.pto} > 160
```

当此表达式评估为 **true** 时，集成会执行您创建的条件流并与此条件关联。

注意

在表达式中，当 **Conditional Flows** 步骤位于以下流之一时，需要额外的属性规格：

- **API** 供应商集成操作流
- 从 **Webhook** 连接开始的简单集成
- 从自定义 **REST API** 连接开始的简单集成

在这些流中，**Fuse Online** 将实际消息内容嵌套在 **body** 属性中。这意味着，到 **Conditional Flows** 步骤的输入包含一个 **body** 属性，其中包含包含实际消息内容的另一个 **body** 属性。因此，在 条件流 步骤中，在其中一个类型的流中，您必须指定两个 正文 实例。例如，假设您想评估输入消息的 **pto** 字段中的内容。指定类似如下的表达式：

```
${body.body.pto} > 160
```

b. 可选的。单击 **Add another condition**，再重复上一步。对您要定义的每个额外条件执行此操作。

c. 可选的。在条件字段右侧点 **up** 或 **down** 箭头更改 **Conditional Flows** 步骤评估定义的条件的顺序。

d. 可选的。如果要有一个默认条件流，请单击 **Execute default flow**。

如果您选择 **Execute** 默认流，在执行期间，如果没有指定条件评估为 **true**，则集成将执行默认条件流。如果您没有选择 **Execute** 默认流，在执行期间，如果没有指定评估为 **true** 的条件，则集成将继续执行，执行遵循这个 条件流 步骤。

5. 点击 **Next**。

6. 可选的。如果 Fuse Online 提示输入它，请指定输出数据类型。作为此条件流 步骤的一部分的所有条件流 必须具有相同的输出类型。

7. 点击 Next。

Fuse Online 显示流视觉化。在您添加的 Conditional Flows 步骤下方，指定了每个条件都有一个条目，如果您表示 Conditional Flows 步骤有一个默认流，则另外一个另外一个条件的默认流。

后续步骤

对于每个条件，在关联的流中添加步骤。如果存在默认流，在默认流中添加步骤。

其他资源

Camel Simple Language 运算符.

4.7.6. 在条件流中添加步骤

在 Conditional Flows 步骤中，在为每个条件定义条件后，为与该条件关联的流添加步骤。在执行期间，当 Conditional Flows 步骤将条件评估为 true 时，它会执行与该条件关联的流。

先决条件

- 您定义了此条件 流步骤 的条件。
- 熟悉集成要传递给此条件 流步骤的消息中的字段。
- 您创建了您要添加到条件流的每个连接。

流程

1. 在集成视觉化中，对于您要添加到的条件，点 Open Flow。

Fuse Online 显示页面顶部的状况。条件流视觉化显示所有条件流具有的 Flow Start 和 Flow End 步骤。

2. 在流视觉化中，点击您要在这个条件流中添加步骤的  。

3. 点您要添加的步骤。您可以添加任何可添加到主流的连接或步骤。

   Flow Start 步骤的输出始终与这个 Conditional Flows 步骤前的主要流步骤的输出相同。例如，如果您向这个条件流中添加过滤器步骤或数据映射程序步骤，可用的字段是主流中可用的字段。

4. 根据需要配置步骤。

5. 对您要添加到此条件流的每个步骤重复前面的三个指令。

6. 在页面顶部的 Flow 字段中，点 down carat 并点 Back to primary 流，这会保存这个条件流并显示主要流。

7. 对于您要添加到的每个条件流，请重复这个过程。

结果

主流具有您在 Conditional Flows 步骤中定义的每个条件的条件流。如果您选择了 Execute default flow 选项，主流也具有默认的条件流。

在执行期间，条件流 步骤执行与评估为 true 的第一个条件关联的条件流。然后，集成执行遵循 Conditional Flows 步骤的步骤。

如果没有条件评估为 true，则 Conditional Flows 步骤执行默认条件流。然后，集成执行遵循 Conditional Flows 步骤的步骤。

如果这两者都为 true ：

- 没有条件评估为 true。

- 没有默认条件流。

然后，集成执行遵循 Conditional Flows 步骤的步骤。

4.8. 添加数据映射程序步骤

几乎所有集成都需要数据映射。数据映射器步骤将之前连接中的数据字段以及任何其他步骤映射到下一个流中可以操作的 data 字段。例如，假设集成数据包含一个 Name 字段，流中的下一个连接则有一个 CustomerName 字段。您需要将 source Name 字段映射到目标 CustomerName 字段。

重要

数据映射器显示由上一集成步骤提供的最大可能的源字段集合。但是，并非所有连接都会在显示的源字段中提供数据。例如，对第三方应用程序的更改可能会停止在特定字段中提供数据。当您创建集成时，如果您注意到数据映射没有如预期那样的行为，请确保要映射的 source 字段包含您预期的数据。

前提条件

您正在创建或编辑流。

流程

1. 在您要添加数据映射程序步骤的流视觉化中，点  。
2. 单击 Data Mapper，以显示数据映射器中的源和目标字段。

后续步骤

有关 下一个连接，请参阅将集成数据映射到字段。

4.9. 添加基本过滤器步骤

您可以为流添加一个步骤来过滤流操作的数据。在过滤器步骤中，Fuse Online 检查数据，并仅在内容

满足您定义的条件时才继续。例如，在从 Twitter 获取数据的流中，您可以指定您希望仅在包含"Red Hat"的 Teets 上继续运行。

先决条件

- 流包含需要的所有连接。
- 您正在创建或编辑流。

流程

1. 在您要添加过滤器步骤的流视觉化中，点  。
2. 单击 **Basic Filter**。
3. 在 **Configure Basic Filter Step** 页面中，只有在传入的数据匹配字段时才会在 **Continue** 中：
   - 接受必须满足所有定义的规则的默认值。
   - 或者，选择 以下 **ANY** 来指示必须满足一个规则。
4. 定义过滤器规则：
   a. 在 **Property Name** 字段中，输入或选择包含您要过滤器评估内容的字段名称。例如，假设进入步骤的数据由指出 Twitter 处理的 tweets 组成。只有在 tweet 包含某些内容时才会继续执行。tweet 位于名为 **text** 的字段中，因此您可以在属性名称字段中输入或选择文本作为值。

      您可以使用以下方法定义属性名称：

      - 开始键入：该字段具有 typeahead 功能，可在弹出窗口中为您提供可能完成的列

表。从框中选择正确的。

- 点字段。此时会出现一个可用属性列表的下拉菜单。从列表中选择感兴趣的属性。

b. 在 Operator 字段中，从下拉菜单中选择一个 operator。设置默认为 Contains 。要继续执行，您在此字段中选择的条件必须在 Keywords 字段中输入的值评估为 true。

c. 在 Keywords 字段中，输入要过滤的值。例如，假设您接受默认的 Contains 操作器，并且仅在传入的文本提到某个产品时继续集成执行。您可以在此处输入产品名称。

5. （可选）点 + Add another rule 并定义另一个规则。

您可以通过单击规则条目右上角的垃圾箱图标来删除规则。

6. 过滤器步骤完成后，点 Done 将其添加到流中。

其他资源

- 有关 Operator 和指定要评估的文本示例的详情，请参阅 Camel Simple Language operator。请注意，基本过滤器步骤 匹配 operator 对应于 Simple Language regex operator。

- 如果您无法在基本过滤器步骤中定义所需的过滤器，请参阅 添加高级过滤器步骤。

4.10. 添加高级过滤器步骤

在过滤器步骤中，Fuse Online 检查数据，并仅在内容满足您定义的条件时才继续执行流。如果基本过滤器步骤没有允许您定义您需要的确切过滤器，则添加一个高级过滤器步骤。

先决条件

- 流包含需要的所有连接。

- 您正在创建或编辑流。
- 熟悉 Camel Simple Language，或者已为您提供了一个过滤器表达式。

流程

1. 在流视觉化中，您要在流中添加高级过滤器步骤，点  。
2. 单击 **Advanced Filter**。
3. 在编辑框中，使用 Camel Simple Language 指定一个过滤器表达式。例如，当消息标头的 **type** 字段设置为 **widget** 时，以下表达式评估为 **true** ：

```
${in.header.type} == 'widget'
```

在以下示例中，当消息的正文包含 **title** 字段时，表达式评估为 **true**：

```
${in.body.title}
```

4. 点 **Next** 将高级过滤器步骤添加到流中。

某些流中的额外属性规格

在表达式中，当高级过滤器步骤位于以下流之一时，需要额外的属性规格：

- API 供应商集成操作流
- 从 Webhook 连接开始的简单集成
- 从自定义 REST API 连接开始的简单集成

在这些流中，Fuse Online 将实际消息内容嵌套在 body 属性中。这意味着高级过滤器的输入包含一个 body 属性，其中包含包含实际消息内容的另一个 body 属性。因此，在其中一个类型的流中，在高级过滤器表达式中，您必须指定两个 正文 实例。例如，假设您想评估输入消息 完成 字段中的内容。指定类似如下的表达式：

```
${body.body.completed} = 1
```

4.11. 添加模板步骤

在流中，模板步骤从源获取数据并将其插入到您上传到 Fuse Online 的模板中定义的格式。模板步骤的好处是它会以您指定的一致格式提供数据输出。

在模板中，您可以定义占位符并指定静态文本。在创建流时，您可以添加模板步骤，将 source 字段映射到模板占位符，然后将模板内容映射到流中的下一步。当 Fuse Online 执行流时，它会将映射源字段中的值插入到模板实例中，并使结果可用于流中的下一步。

如果流包含模板步骤，则很可能是该流中唯一的模板步骤。但是，允许流中的多个模板步骤。

Fuse Online 支持以下模板： Freemarker、Melociti、Vel ociy.

先决条件

- 您必须创建或编辑流。
- 如果您要创建简单集成，那么它必须已经拥有它的启动和完成连接。

流程

1. 在流视觉化中，点您要添加模板步骤的

。
2. 单击 Template。此时会打开 Upload Template 页面。

3. 指定模板类型，即 Freemarker、M Mustache 或 Velocity。

4. 要定义模板，请执行以下操作之一：

   - 将模板文件或包含您要修改的文本的文件拖放到模板编辑器中。
   - 单击 浏览以上传，导航到文件并上传该文件。
   - 在模板编辑器中，开始键入 以定义模板。

5. 在模板编辑器中，确保模板对 Fuse Online 有效。此流程后，有效模板的示例为。Fuse Online 在一行的左侧显示

，其中包含语法错误。将鼠标悬停在语法错误指示器上会显示关于如何解决错误的提示。

6. 点 Done 将模板步骤添加到流中。

   如果没有启用 Done 按钮，则至少有一个语法错误必须正确。

   到模板步骤的输入必须采用 JSON 对象的形式。因此，您必须在模板步骤前添加数据映射步骤。

7. 要在模板步骤前添加数据映射程序步骤：

   a. 在流视觉化中，点您刚刚添加的模板步骤前的

。

   b. 单击 Data Mapper。

c. 在数据映射器中，将 source 字段映射到每个模板占位符字段。

例如，使用此流程后的示例模板，将 source 字段映射到每个模板字段：

- time
- name
- text

d. 在右上角，点 Done 将数据映射程序步骤添加到流中。

模板步骤的输出始终是一个 JSON 对象。因此，您必须在模板步骤后添加数据映射程序步骤。

8. 在模板步骤后添加数据映射程序步骤：

a. 在流视觉化中，点您刚刚添加的模板步骤后立即使用  。

b. 单击 Data Mapper。

c. 在数据映射器中，将模板的 消息字段 映射，该字段始终包含将源字段插入到模板中的结果到目标字段。例如，假设 Gmail 连接在流中下一步，您想要发送模板步骤的结果作为 Gmail 消息的内容。为此，您要将 message source 字段映射到 文本 目标字段。

d. 在右上角，单击 Done。

模板示例

Mustache 模板示例：

```
At {{time}}, {{name}} tweeted:
{{text}}
```

FreeMarker 和 Velocity 支持这个示例模板：

```
At ${time}, ${name} tweeted:
${text}
```

velocity 还支持没有大括号的语法，如下例所示：

```
At $time, $name tweeted:
$text
```

占位符不能包含 .（句点）。

其他资源

有关映射字段的详情，请参阅 将集成数据映射到下一连接的字段。

4.12. 添加自定义步骤

如果 Fuse Online 不提供您在流中需要的步骤，开发人员可以在扩展中定义一个或多个自定义步骤。自定义步骤对流中连接之间的数据进行操作。

您可以像添加内置步骤一样向流添加自定义步骤。对于简单集成，请选择启动和完成连接，根据需要添加其他连接，然后添加额外的步骤。对于 API 提供程序集成，请选择其流执行自定义步骤的操作，根据需要添加连接，然后添加其他步骤。添加步骤时，Fuse Online 在流中从上一步中收到的数据上运行。

先决条件

- **您已将自定义步骤扩展上传到 Fuse Online。请参阅 使自定义功能可用。**
- **您正在创建或编辑流。**

- 流已具有需要的所有连接。

流程

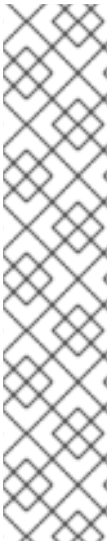
1. 在您要添加自定义步骤的流视觉化中，点  。
2. 点您要添加的自定义步骤。

   可用步骤包括上传至 Fuse Online 环境的扩展中定义的任何自定义步骤。
3. 响应 以提示输入执行步骤所需的任何信息。此信息因每个自定义步骤而异。

第 5 章 创建由 REST API 调用触发的集成

要按需触发集成执行，请开始与您提供的 REST API 描述文档集成。以这种方式开始的集成被称为 ***API 供应商集成***。API 提供程序集成允许 REST API 客户端调用触发集成执行的命令。

当 Fuse Online 发布 API 供应商集成时，任何对集成端点有网络访问权限的客户端都可以触发集成执行。



注意

如果您在 OpenShift Container Platform on OpenShift Container Platform 上使用 Fuse Online，管理员可以配置 Fuse Online 服务器，以启用 API 供应商集成 API 的 Red Hat 3scale 发现功能。默认情况下，Fuse Online 标注 API 提供程序集成的 API 服务定义，以用于 3scale，但不会公开这些 API 进行自动 3scale 发现。没有 3scale 发现功能，没有访问控制。通过 3scale 发现功能，您可以设置访问策略、集中控制并为 API 供应商集成 API 提供高可用性。如需更多信息，请参阅 Red Hat 3scale 文档页面中提供的 API 网关文档。

另请参阅：配置 Fuse Online 以启用 API 的 3scale 发现。

以下主题提供了创建 API 供应商集成的信息和说明：

- 第 5.1 节 “创建 API 供应商集成的优点、概述和工作流”
- 第 5.2 节 “OpenAPI 操作与 API 供应商集成流的关系”
- 第 5.3 节 “创建 API 供应商集成”
- 第 5.4 节 “为 API 供应商集成定义操作流”
- 第 5.5 节 “导入和发布示例 API 供应商快速启动集成”
- 第 5.6 节 “测试 API 供应商快速启动集成示例”

介绍如何创建、发布和测试 API 供应商集成的视频，请参阅 https://youtu.be/sox8SSqJ0zQ。

5.1. 创建 API 供应商集成的优点、概述和工作流

API 提供程序集成从 REST API 服务开始。此 REST API 服务由您在创建 API 供应商集成时提供的 OpenAPI 3 （或 2）文档定义。发布 API 供应商集成后，Fuse Online 在 OpenShift 上部署 REST API 服务。API 提供程序集成的好处是 REST API 客户端可以调用触发集成执行的调用。

多执行流

API 供应商集成有多个执行路径，称为流。OpenAPI 文档定义的每个操作都有自己的流。在 Fuse Online 中，对于 OpenAPI 文档定义的每个操作，您可以在该操作的执行流中添加连接和其他步骤。这些步骤会根据特定操作的要求处理数据。

执行流示例

例如，假设人工资源应用调用 Fuse Online 提供的 REST API 服务。假设调用调用添加新员工的操作。处理此调用的操作流可以：

- 连接到一个应用程序，为新的员工设备创建费用报告。
- 连接到 SQL 数据库，以添加用于设置新设备的内部票据。
- 连接到 Google 邮件，向提供信息的新员工发送消息。

触发执行的方法

可以通过多种方式调用触发集成执行的 REST API，包括：

- 使用数据输入并生成调用的 Web 浏览器页面。
- 明确调用 REST API 的应用，如 curl 工具。
- 调用 REST API 的其他 API，如 webhook。

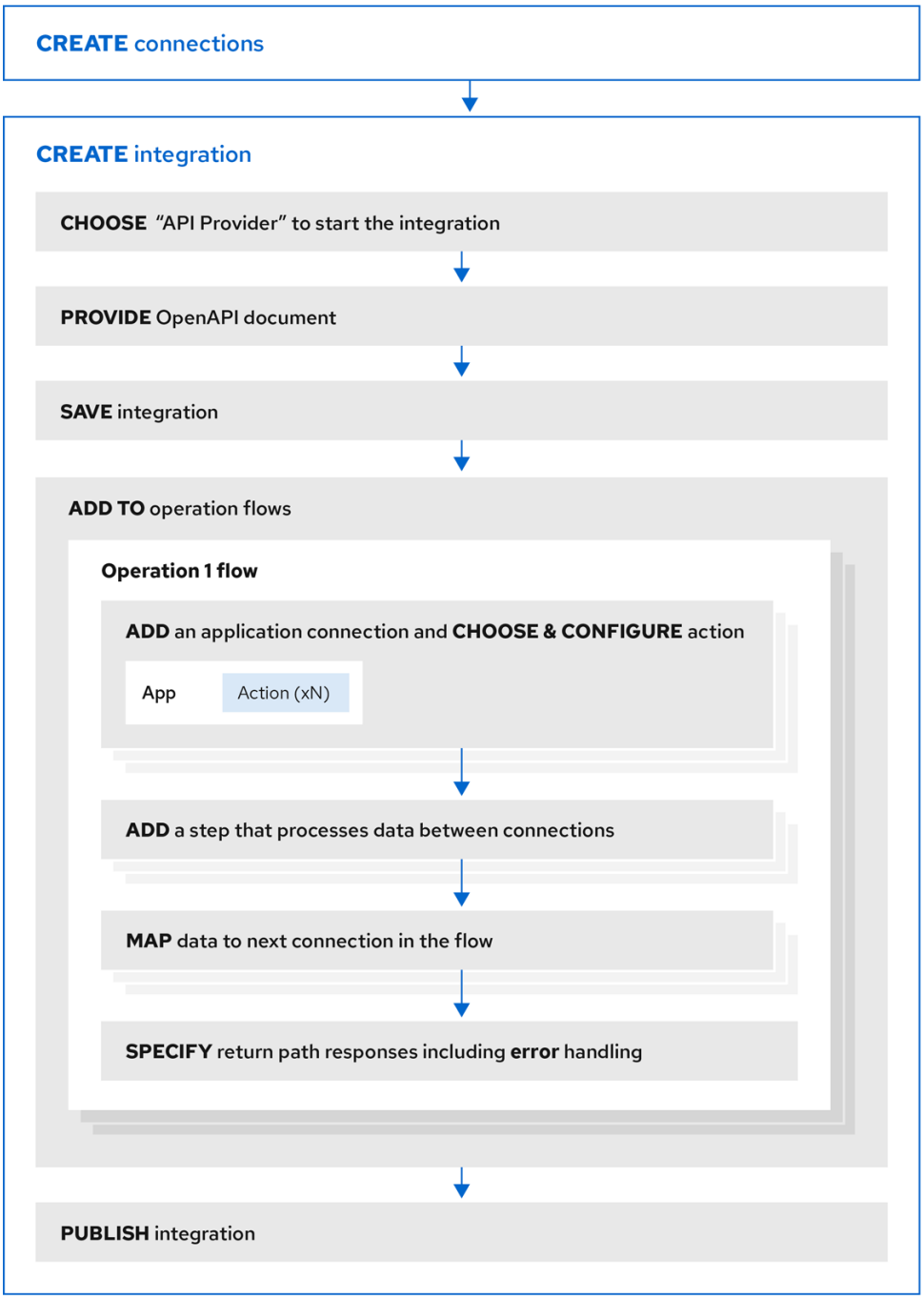
编辑流的方法

对于每个操作，您可以通过以下方法编辑其流：

- 向需要处理数据的应用程序添加连接。
- 在连接间添加步骤，包括分割、聚合和数据映射步骤。
- 映射连接错误消息，以便在完成流的 HTTP 响应中返回代码。响应发送到调用触发集成执行的调用的应用程序。

创建 API 供应商集成的工作流

以下示意图中显示了创建 API 供应商集成 的一般 工作流：



Fuse_14_1019

发布 API 供应商集成

发布 API 供应商集成后，在集成摘要页面中，Fuse Online 会显示 REST API 服务的外部 URL。此外部 URL 是客户端用来调用 REST API 服务的基本 URL。

对于 OCP 上的 Fuse Online 环境，可能会启用红帽 3scale 发现 API 提供程序集成。在本例中，3scale 发布用于调用服务的 URL。

测试 API 供应商集成

要测试 API 供应商集成的流，您可以使用 curl 工具。例如，以下 curl 命令会触发 REST API 服务 URL 的 Get Task by ID 操作执行流 ：https://i-task-api-proj319352.6a63.fuse-ignite.openshiftapps.com/api/。

HTTP GET 命令是默认请求，因此不需要指定 GET。URL 的最后一部分指定要获取的任务 ID：

```
curl -k https://i-task-api-proj319352.6a63.fuse-ignite.openshiftapps.com/api/todo/1
```

5.2. OPENAPI 操作与 API 供应商集成流的关系

API 供应商集成的 OpenAPI 文档定义 REST API 客户端可以调用的操作。每个 OpenAPI 操作都有自己的 API 供应商集成流。因此，每个操作也可以拥有自己的 REST API 服务 URL。每个 URL 由 API 服务的基本 URL 和子路径可选定义。REST API 调用指定一个操作的 URL，用于触发该操作的流执行。

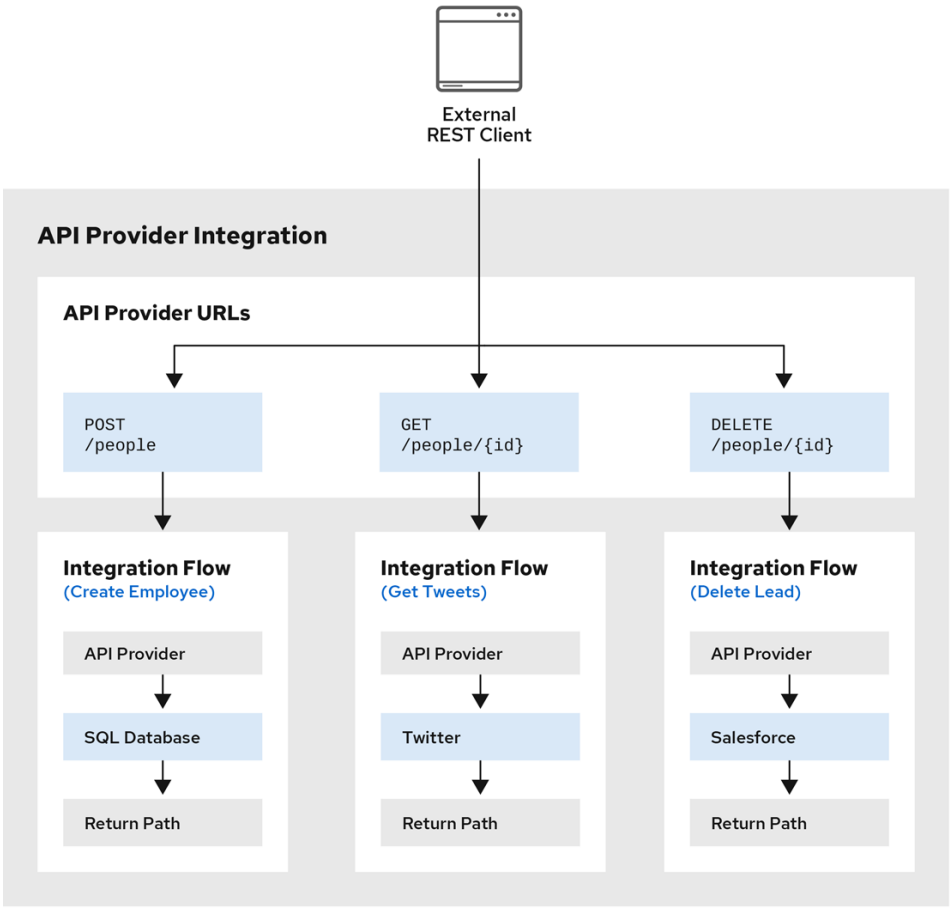
您的 OpenAPI 文档决定了哪个 HTTP 动词（如 GET、POST、DELETE 等）您可以在调用 REST API 服务 URL 中指定。调用 API 提供程序 URL 的示例是 在尝试 API 供应商快速入门示例的说明 中。

您的 OpenAPI 文档还决定操作可以返回的 HTTP 状态代码。操作的返回路径只能处理 OpenAPI 文档定义的响应。例如，基于其 ID 删除对象的操作可能会定义这些可能的响应：

```
"responses": {
  "204": {
        "description": "Task deleted"
      },
  "404": {
        "description": "No Record found with this ID"
      },
  "500": {
        "description": "Server Error"
      }
}
```

API 供应商集成示例

下图显示了处理人员数据的 API 供应商集成。外部 REST API 客户端调用由 API 提供程序集成部署的 REST API URL。调用 URL 会触发一个 REST 操作的执行流。此 API 供应商集成有 3 个流。每个流程都可以使用 Fuse Online 中提供的任何连接或步骤。REST API 及其流是一个 Fuse Online API 供应商集成，部署在一个 OpenShift pod 中。



Fuse_19_1019

在创建 API 供应商集成时编辑 OpenAPI 文档

在为 API 供应商集成指定 OpenAPI 文档后，您可以根据需要更新文档，同时为 API 操作定义执行流。要做到这一点，请点击您要编辑 API 供应商集成的页面右上角的 View/Edit API Definition。这会在 API Designer 编辑器中显示 OpenAPI 文档。编辑并保存文档，以做出反映在 Fuse Online 中的更改。

编辑 OpenAPI 文档时的注意事项：

- 用于同步的 operationId 属性

  在 API Designer 编辑器和 Fuse Online 集成编辑器中的 OpenAPI 文档版本之间同步取决于分配给文档中定义的每个操作的唯一 operationId 属性。您可以为每个操作分配特定的 operationId 属性值，或使用 Fuse Online 自动生成的值。

- 请求和响应定义

  在每个操作的定义中，您可以提供一个 JSON 模式来定义操作的请求和响应。Fuse Online 使用 JSON 模式：

- 作为操作的输入和输出数据的基础
- 显示数据映射器中的操作字段

- 无 cyclic 模式参考

  API 供应商集成操作的 JSON 模式无法具有 cyclic 模式引用。例如，指定请求或响应正文的 JSON 模式无法作为整体引用，也无法通过中间 JSON 模式引用任何部分。

5.3. 创建 API 供应商集成

要创建 API 供应商集成，请提供 OpenAPI 文档(.json、.yaml 或 .yml 文件)，用于定义集成可以执行的操作。Fuse Online 为每个操作创建一个执行流。编辑每个操作的流，以添加根据该操作的要求处理集成数据的连接和步骤。

先决条件

- 您可以为您希望集成的 REST API 操作提供或定义 OpenAPI 文档。

  要试验，请下载 task-api.json 文件 的原始版本，它是 API 供应商快速入门的 OpenAPI 文档。当 Fuse Online 提示提供 OpenAPI 文档时，您可以上传此文件。或者，您可以指定原始 task-api.json 文件的 URL，即 https://raw.githubusercontent.com/syndesisio/syndesis-quickstarts/1.11/api-provider/task-api.json。

- 每个 OpenAPI 操作都有一个计划。
- 您为您要添加到操作流的每个应用程序或服务创建一个连接。

流程

1. 在 Fuse Online 中，在左侧导航面板中，单击 Integrations。
2. 点 Create Integration。

3. 在 Choose a connection 页面上，单击 API Provider。

4. 在 Start 与 API 调用页面集成 ：

- 如果您有定义 REST API 操作的 OpenAPI 文档，请上传 OpenAPI 文档。
- 如果您需要定义 OpenAPI 文档，请选择 Create a new OpenAPI 3.x document 或 Create a new OpenAPI 2.x 文档。

5. 点击 Next。

- 如果您上传了一个文档，请查看或编辑它：
  a. 点 Review/Edit 以打开 API Designer 编辑器。
  b. 根据需要查看和编辑。

  另外，如果您的文档使用 OpenAPI 2 规格，如果您希望 API Designer 转换您的文档以符合 OpenAPI 3 规格，您可以点 Convert to OpenAPI 3。

  c. 在右上角，单击 Save 或 Cancel 以关闭编辑器。
  d. 点击 Next。
- 如果您要创建文档，则在 Fuse Online 的 API Designer 编辑器中打开：
  a. 定义 OpenAPI 文档，如使用 API Designer 设计和开发 API 定义 中所述。
  b. 在右上角，单击 Save，这将关闭编辑器。

c. 点击 **Next**。

结果

Fuse Online 显示 OpenAPI 文档定义的操作列表。

后续步骤

对于每个操作，定义执行该操作的流。

5.4. 为 API 供应商集成定义操作流

定义 REST API 服务的 OpenAPI 文档定义服务可以执行的操作。创建 API 供应商集成后，您可以编辑每个操作的流。

每个操作都只有一个流。在操作流中，您可以添加到其他应用程序和服务的连接，以及在连接间操作数据的步骤。

当您添加到操作流时，您可能会发现需要更新 API 供应商集成的 OpenAPI 文档。要做到这一点，请点击您要编辑 API 供应商集成的页面右上角的 View/Edit API Definition。这会在 API Designer 编辑器中显示您的文档。在 OpenAPI 定义中，只要每个操作都有唯一的 operationId 属性，您可以在 API Designer 中保存更新，Fuse Online 可以同步 API 供应商集成的流定义来具有您的更新。

先决条件

- 您创建了 API 供应商集成，为它提供一个名称并保存。
- 您创建了与希望操作流连接的每个应用程序或服务的连接。详情请查看 创建连接的信息。
- Fuse Online 显示 API 定义的操作列表。

流程

1. 在 Operations 列表页面中，针对您要定义的流的操作，点 Create flow。

2. 对于您要添加到此流的每个连接：

   a. 在流视觉化中，点加号在该位置添加连接。

   b. 点击要添加的连接。

   c. 选择您希望此连接执行的操作。

   d. 通过在标记的字段中输入数据来配置操作。

   e. 点击 **Next**。

   在继续操作前，将所有所需的连接添加到流。

3. 在这个操作流中，要处理连接间的数据：

   a. 在流视觉化中，点您要添加步骤的加号。

   b. 点您要添加的步骤。

   c. 通过在标记的字段中输入数据来配置步骤。

   d. 点击 **Next**。

   有关帮助，请参阅 在连接之间添加步骤。

   如果要添加另一个在连接间处理数据的步骤，请重复这个指令子集。

4. 将数据映射到下一个连接中的字段：

   a. 在流视觉化中，检查数据类型不匹配

   图标，这表示连接无法处理传入的数据。您需要在此处添加数据映射程序步骤。

   b. 对于流视觉化中的每个数据不匹配图标：

      i. 单击该步骤前的加号。

      ii. 单击 **Data Mapper**。

      iii. 定义所需的映射。有关帮助，请参阅 下一个连接中的将集成数据映射到字段。

      iv. 点 **Done** 将数据映射程序步骤添加到流中。

5. 在流视觉化中，在 **Provided API return Path** 步骤中，点 **Configure**。

每个 API 提供程序集成通过向触发操作流执行的 REST API 调用者发送响应来完成每个操作流。响应包含您为用来完成操作流的 **Provided API 返回路径** 步骤配置的返回代码之一。配置返回路径步骤，如下所示：

a. 在 **Default Response** 下，在 **return Code** 字段中，接受 Fuse Online 显示的默认响应，或者点击 **down caret**，并滚动来选择您需要的默认响应。当执行操作流时，流不会返回任何配置的错误响应，流会发送此响应。通常，默认响应返回代码表示操作成功。

b. 在 **Error Handling** 下，指示您是否要在返回消息的正文中包含错误消息。

在开发过程中，您通常想返回错误消息。但是，在生产环境中，如果包含敏感或专有信息，您可能希望隐藏错误消息。错误消息是 JSON 格式的字符串，包含 **responseCode**、类别、消息 和错误 元素，例如：

```
{
  responseCode: 404,
```

```
  category:  "ENTITY_NOT_FOUND_ERROR",
  message: "SQL SELECT did not SELECT any records"
  error: SYNDESIS_CONNECTION_ERROR
}
```

请注意，在开发过程中，了解出错的最可靠方法是检查对调用者的响应中的 HTTP_RESPONSE STATUS 标头。您还可以检查集成 pod 的日志中的 INFO 信息。集成的 Activity 日志显示成功的交换，错误并不总是在 Activity 日志中可见。

c. 在 Error Response Codes 下，Fuse Online 会显示流中连接可能会返回的每个错误的条目。对于每个错误，接受 200 All is good 默认返回代码，或者点击以选择另一个 HTTP 状态返回代码。

您可以从中选择的返回代码是 OpenAPI 文档为此流执行的操作定义的返回代码。如果 Fuse Online 没有显示您需要的返回代码，您可以编辑 OpenAPI 文档来添加它。

为此，请在右上角点 View/Edit API Definition。根据需要编辑 OpenAPI 文档。完成后，保存 OpenAPI 文档。Fuse Online 返回编辑 Provided API 返回路径，并反映您保存的任何更改。

d. 点 Next 完成返回路径的配置。

6. 当此流具有所有必需的连接和步骤，且没有数据不匹配图标时，或者您现在不再需要编辑流，请执行以下操作之一：

- 发布 criu-wagonTo 开始运行集成，在右上角点 Publish。这会构建集成，将 REST API 服务部署到 OpenShift，并使集成可用。每次完成操作流创建或每次编辑操作流时，您可以发布集成。
- 保存 mvapich-wagonTo 显示右上角的操作列表，点 Save。

重复此步骤以编辑另一个操作流。

测试 API 供应商集成

- 测试在这些平台上运行的 API 供应商集成：

- OpenShift Online
- OpenShift Dedicated
- 当 API 发现被禁用时，OpenShift Container Platform

您可以使用 curl 工具确认集成是否按预期工作。在 curl 命令中，指定 Fuse Online 发布 API 提供程序集成后显示的外部 URL。有关执行此操作的示例，请参阅 测试示例 API 提供程序快速启动集成。

- 在启用 API 发现时测试在 OpenShift Container Platform 上运行的 API供应商集成

红帽 3scale 发布您的 API 供应商集成。要测试集成，请打开 3scale 仪表板来获取集成的 URL。

例如，您可以禁用 API 供应商集成的发现，例如，您不希望 Red Hat 3scale 控制对集成 API 的访问，或者在 Fuse Online 中测试 API 供应商集成。如果您禁用发现功能，Fuse Online 重新发布集成并提供调用和测试集成执行的外部 URL。为此，在 Fuse Online 中进入集成的摘要页面。在此页面上，单击 Disable discovery。Fuse 在线重新发布集成并提供集成的 URL。有关如何测试集成的示例，请参阅 测试 API 供应商快速启动集成。测试后，您可以为 API 供应商集成重新启用发现，以便 3scale 发布它。

您可以为每个 API 供应商集成启用或禁用发现。

5.5. 导入和发布示例 API 供应商快速启动集成

Fuse Online 提供了一个 API 供应商快速入门集成，您可以导入到 Fuse Online 环境中。此快速入门包括任务管理 API 的 OpenAPI 文档。导入快速启动集成后，您可以检查流，然后发布集成。完成下面描述的步骤后，任务 API 集成将运行并准备好执行。

API 供应商快速启动可帮助您快速了解如何配置、发布和测试 API 供应商集成。但是，它不是 API 提供程序集成很有用的实际示例。对于真实示例，假设您已使用 Fuse Online 发布几个简单的集成。您可以定义 OpenAPI 文档来触发这些集成的执行。要做到这一点，您将编辑每个 OpenAPI 操作的流程，使其与您发布的简单集成几乎相同。

先决条件

- Fuse Online 在浏览器中打开。

- Fuse Online 环境必须包含 Todo 示例应用程序，如将 示例应用程序添加到 OCP 上运行的 Fuse Online 环境 中所述。

流程

1. 导入任务 API 快速启动集成：

   a. 进入 https://github.com/syndesisio/syndesis-quickstarts/tree/1.11/api-provider 并下载 TaskAPI-export.zip。

   b. 在 Fuse Online 中，在左侧导航面板中，单击 Integrations。

   c. 在右上角，单击 Import。

   d. 将您下载的 TaskAPI-export.zip 文件拖到 Import 页面中。Fuse Online 表示它已成功导入该文件。

   e. 在左侧导航面板中，点 Integrations 查看您刚才导入 的任务 API 集成的条目。虽然该条目表示需要配置，但此集成已准备好发布。

2. 点 Edit Integration 查看此 API 提供的操作列表。

3. 检查每个操作的流：

   a. 点击 Edit flow 按钮，以显示该流的视觉化。

   每个流已经有一个数据库连接、一个或多个数据映射程序步骤，以及完成流的 Provided API 返回路径 步骤。

   b. 对于 Invoke SQL 步骤，单击 Configure 以查看连接执行的 SQL 语句。然后单击 Cancel 以返回到该操作的视觉化流。

c. 对于数据映射程序步骤，点 Configure 以查看映射程序。然后单击 Cancel 以返回到视觉化。

d. 对于 Provided API return Path 步骤，这是每个操作流中的最后一步，点 Configure 以查看操作可能会发送到调用者的 HTTP 返回代码。点 Cancel 返回视觉化。

e. 检查一个操作流后，点 Integrations> Task API> Operation 下拉菜单，然后选择另一个操作。

f. 重复这一部分步骤，以检查每个流。

4. 如果需要，点 Save 并编辑集成名称。

5. 单击 Save and Publish。

集成摘要页面会在编译、构建、部署和启动集成时显示发布进度。

6. 当 Task API 集成摘要页面显示 Running 时，Fuse Online 会显示 Task API 服务的外部 URL。它类似如下：

https://i-task-api-fuseonline.apps.openshift.com/

这是 Fuse Online 使 Task API 服务可用的地方。REST API 调用指定以这个基本 URL 开头的 URL。

如果您在 OpenShift Container Platform 上使用 Fuse Online，如果外部 URL 不在集成概述页面中，则管理员启用了 Red Hat 3scale 发现功能。这意味着 Red Hat 3scale 控制对集成的 API 的访问，并发布您的 API 供应商集成。要测试集成，请打开 3scale 仪表板来获取集成的 URL。

如果您不希望 Red Hat 3scale 控制对集成 API 的访问，您可以禁用发现。通过查看集成摘要页面，在 Fuse Online 中执行此操作。在此页面上，单击 Disable discovery。Fuse 在线重新发布集成并提供外部 URL 来调用集成执行。

您可以为每个 API 供应商集成启用或禁用发现。

5.6. 测试 API 供应商快速启动集成示例

当 Fuse Online Task API quickstart 集成运行时，您可以调用 curl 实用程序命令，将 HTTP 请求发送到 Task API 服务。如何指定 HTTP 请求决定了调用触发的流。

先决条件

- Fuse Online 表示 Task API 集成 正在运行。
- 如果您的 Fuse Online 环境在 OCP 上运行，Fuse Online 没有配置为将 API 公开给 3scale，或您禁用了 Task API 集成发现。

流程

1. 在 Fuse Online 中，在左侧导航面板中，单击 Integrations。
2. 在 Task API 集成条目中，单击 View 以显示集成摘要。
3. 复制集成的外部 URL。
4. 在终端中，调用如下命令，将集成的外部 URL 分配给 externalURL 环境变量。务必将这个示例命令中的 URL 替换为您复制的 URL。

```
export externalURL="https://i-task-api-fuseonline.apps.openshift.com"
```

5. 调用 curl 命令，该命令会触发 Create new task 操作的流执行：

```
curl -k --header "Content-Type: application/json" --request POST --data '{"id":1, "task":"my
first task :)!"}' $externalURL
```

- -k 允许 curl 在出现其他被视为不安全的服务器连接时继续和操作。

- --header 表示命令正在发送 JSON 格式数据。
- --request 指定存储数据的 HTTP POST 命令。
- --data 指定要存储的 JSON 格式内容。在本例中，内容为 {"id":1, "task":"my new task!"}。请注意，不需要 完成 的字段。
- $externalURL 是要调用的 URL。

此命令将 HTTP POST 请求发送到 Task API 服务，该服务会触发 Create new task 操作流的执行。流执行向示例数据库添加新任务，并返回类似以下内容的消息，以指示它的功能：

+

```
{"completed":null,"id":1,"task":"my new task!"}
```

6. 调用以下 curl 命令以创建另外两个任务：

```
curl -k --header "Content-Type: application/json" --request POST --data '{"id":2, "task":"my second task :|"}' $externalURL

curl -k --header "Content-Type: application/json" --request POST --data '{"id":3, "task":"my third task :("}' $externalURL
```

7. 调用 curl 命令，它根据 ID 操作触发 Fetch 任务的执行 流：

```
curl -k $externalURL/1
```

要获取任务，curl 命令只需要指定 URL。HTTP GET 命令是默认请求。URL 的最后一部分指定要获取的任务 ID。

8. 调用以下 curl 命令，该命令触发 Fetch 所有任务操作 的流执行：

```
curl -k $externalURL
```

9. 另外，还可调用 curl 命令，该命令触发对 ID 操作的 Delete 任务执行流：

```
curl -k -X DELETE $externalURL/3
```

此命令调用 HTTP DELETE 命令，其 URL 与通过 ID 获取任务的命令相同。

第 6 章 创建由 HTTP 请求(WEBHOOK)触发的集成

您可以通过将 HTTP GET 或 POST 请求发送到 Fuse Online 公开的 HTTP 端点来触发简单集成的执行。以下主题提供详情：

- 第 6.1 节 “使用 Fuse 在线 Webhook 的一般流程”
- 第 6.2 节 “创建 HTTP 请求可以触发的集成”
- 第 6.3 节 “Fuse Online 如何处理 HTTP 请求”
- 第 6.4 节 “调用 Fuse Online Webhook 的 HTTP 客户端的指南”
- 第 6.5 节 “关于用于指定请求参数的 JSON 模式”
- 第 6.6 节 “如何指定 HTTP 请求”

6.1. 使用 FUSE 在线 WEBHOOK 的一般流程

要触发与 HTTP GET 或 POST 请求集成的执行，您必须执行以下操作：

1. 决定是否要将 GET 或 POST 请求发送到 Fuse Online。
2. 规划您的集成以处理此请求。
3. 创建完成集成的连接。

   Fuse Online 提供了一个 Webhook 连接，用作启动连接。
4. 创建您要添加到集成的任何其他连接。

5. 创建集成：

   a. 添加 Webhook 连接作为启动连接。

   b. 添加完成连接，然后集成所需的任何其他连接。完成连接以及任何中间连接处理 HTTP 请求，该请求会触发对集成的执行。您可以选择并指定最合适的 HTTP 请求来完成您的目标。请记住：

   - 向应用程序添加连接，其中包含您要获取或包含您要更新的数据的数据。
   - GET 请求仅限于键/值参数规格。
   - POST 请求可以提供任意正文，如 XML 或 JSON 实例。
   - Fuse Online 仅返回 HTTP 状态标头，且不会返回任何数据。因此，您可以定义由 GET 请求触发的集成，以及更新数据而不是获取数据。同样，您可以定义由 POST 请求触发的集成，并获取数据而不是更新数据。

   c. 在 Webhook 连接后添加数据映射程序步骤。

   对于 GET 请求，将 HTTP 请求中的参数字段映射到下一次连接中的 data 字段。

   对于 POST 请求，您可能会通过传递 JSON 实例、JSON 模式、XML 实例或 XML 模式来指定请求中的输出数据。如果没有，那么当您将 Webhook 连接添加为集成启动连接时，Fuse Online 会提示您指定输出数据类型。如果没有，则默认的 Webhook 连接输出数据类型采用 JSON 格式。

   d. 添加集成需要的任何其他步骤。

6. 发布集成并等待它变为 Running。

7. 进入集成摘要页面，再复制 Fuse Online 提供的外部 URL。

8. 修改此外部 URL 以构造 GET 或 POST 请求。

9. 实施将 HTTP GET 或 POST 请求发送到 Fuse Online 的应用。

6.2. 创建 HTTP 请求可以触发的集成

要触发与 HTTP GET 或 POST 请求集成的执行，请在集成启动连接中添加 Webhook 连接。

流程

1. 在左侧的 Fuse Online 面板中，单击 Integrations。

2. 点 Create Integration。

3. 在 Choose a connection 页面上，单击 Webhook 连接。

4. 在 Choose an action 页面中，选择 Incoming Webhook 操作。

   在 Webhook Configuration 页面中，Fuse Online 显示 Fuse Online 为此集成的 webhook 令牌。

   当您构建 HTTP 请求时，此令牌是 URL 的最后一部分。发布此集成并正在运行后，Fuse Online 会显示 Fuse Online 外部 URL，该 URL 的末尾有此令牌。

   Webhook 配置页面 还包括 Default Response 和 Error Handling 部分。Webhook 步骤向调用它的 HTTP 客户端发送响应。响应包含一个返回代码，默认情况下，返回消息正文中的错误消息。

5. 在 Default Response 下，在 return Code 字段中接受 Fuse Online 显示的默认响应，或使用下拉列表选择您想要的默认响应。当执行操作流时，流不会返回任何配置的错误响应，流会发送此响应。通常，默认响应返回代码表示操作成功。

6. 在 Error Handling 下，指示您是否要在返回消息的正文中包含错误消息。

在开发过程中，您通常想返回错误消息。但是，在生产环境中，如果包含敏感或专有信息，您可能希望隐藏错误消息。错误消息是一个 JSON 格式的字符串，其中包含 responseCode、类别、消息 和错误元素，例如：

```
{
  responseCode: 404,
  category:  "ENTITY_NOT_FOUND_ERROR",
  message: "SQL SELECT did not SELECT any records"
  error: SYNDESIS_CONNECTION_ERROR
 }
```

请注意，在开发过程中，了解出错的最可靠方法是检查对调用者的响应中的 HTTP_RESPONSE STATUS 标头。您还可以检查集成 pod 的日志中的 INFO 信息。集成的 Activity 日志显示成功的交换，错误并不总是在 Activity 日志中可见。

7. 对于 Webhook 步骤可能会返回的每个错误，接受默认的返回代码，或使用下拉列表选择另一个 HTTP 状态返回代码。

8. 点击 Next。

9. 在 Specify Output Data Type 页面中：

   a. 单击 Select Type 字段，然后选择 JSON 模式。

   b. 在 Definition 字段中，粘贴用于定义 HTTP 请求中参数的数据类型的 JSON 模式。请参阅关于 JSON 模式以指定请求参数。

   c. 在 Data Type Name 字段中，指定此数据类型的名称。虽然这是可选的，但如果您指定了名称，它会出现在 data mapper Sources 列表中，这样可方便正确映射字段。

   d. 另外，在 Data Type Description 字段中，提供一些可帮助您区分此数据类型的信息。

   e. 点击 Next。

10. 将完成连接添加到集成。

11. 添加任何其他所需连接。

12. 添加任何其他必要的步骤。

13. 在启动连接后立即添加数据映射程序步骤。

14. 单击 **Publish**，为集成提供名称和可选的描述，然后单击保存并发布。

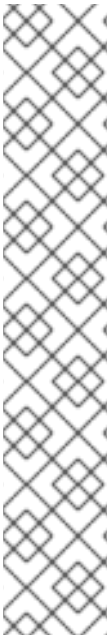
6.3. FUSE ONLINE 如何处理 HTTP 请求

您可以指定 HTTP GET 或 POST 请求，以触发简单集成的执行。虽然 GET 请求通常会获取数据和 POST 请求通常会更新数据，但您可以使用任一请求来触发执行任一操作的集成。请求中的任何参数都可用于映射到集成中的下一连接中的数据字段。详情请参阅 关于指定请求参数的 JSON 模式。

Webhook 连接只会将接收的数据传递给集成中的下一个连接。当 Fuse Online 收到 HTTP 请求时，它：

- 将 HTTP 状态标头返回到请求者。当请求成功触发集成执行时，Fuse Online 返回代码为 201。当请求无法触发集成执行时，Fuse Online 返回代码为 5xx。
- 不要将任何其他数据返回到请求者。换句话说，响应中没有包含状态标头的数据。
- 将请求中的数据传递给集成中的下一个连接。

这意味着，您可以定义由 GET 请求触发的简单集成，以及更新数据而不是获取数据。同样，您可以定义由 POST 请求触发的简单集成，并获取数据而不是更新数据。



注意

在集成的 Activity 选项卡中，webhook 步骤的状态每次都 成功。此成功状态表示 Fuse Online} Webhook 和调用它的 HTTP 客户端之间的通信状态。此成功状态并不表示集成成功传递，或者任何步骤都没有错误。HTTP 请求生成的错误在集成的 Activity 日志中不可见。

当您配置 webhook 时，默认 检查返回正文选项中的 Include 错误消息。选中此选项时，若要验证 HTTP 请求生成的错误是否包含在 Webhook 响应中，请发送要生成错误的测试请求，然后检查响应标头。您还可以检查集成 pod 的日志中的 INFO 信息。使用以下命令查看集成的 pod 日志，其中 example-integration-pod 是 pod 的名称。

```
oc logs -f pod/example-integration-pod
```

6.4. 调用 FUSE ONLINE WEBHOOK 的 HTTP 客户端的指南

当您实现向 Fuse Online 发送 HTTP 请求的客户端时，您的实施应：

- 将 添加到 Fuse Online 提供的外部 URL，以构造发出 GET 或 POST 请求的 URL。
- 在 URL 请求中，指定数据类型遵循 io:syndesis:webhook JSON 模式的 HTTP 标头和查询参数值。请参阅关于 JSON 模式以指定请求参数。当标头和查询参数遵循这个数据类型规格时，您可以将参数字段映射到集成中下一个连接可以处理的字段。
- 如果请求成功，则处理返回的成功代码 201。
- 如果请求失败，请处理 HTTP 5xx 错误代码。
- 预计 Fuse 在线的任何其他响应。换句话说，发送请求不会直接将数据返回到返回代码以外的请求客户端。

6.5. 关于用于指定请求参数的 JSON 模式

在集成中，您通常会将 HTTP 请求中的标头和查询参数映射到集成中下一连接可以处理的数据字段。要实现此目的，当您将 Webhook 连接添加到集成时，请在具有以下结构的 JSON 模式中指定输出数据类型：

-

```
{
  "$schema": "http://json-schema.org/schema#",
  "id": "io:syndesis:webhook",
  "type": "object",
  "properties": {
    "parameters": {
      "type": "object",
      "properties": { 1
      }
    },
    "body": {
      "type": "object",
      "properties": { 2
      }
    }
  }
}
```

要添加您需要的数据结构，请在 JSON 实例中为您的 HTTP 请求：

1 在 parameters 对象的 properties 部分中指定查询参数。

2 在 body 对象的 properties 部分中指定 HTTP body 模式。

虽然 HTTP 客户端发送的所有数据都位于集成中，当 Webhook 连接的数据形成符合此 JSON 模式时，就可以使用查询参数和正文内容进行映射。

例如，请参阅 如何指定 HTTP 请求。

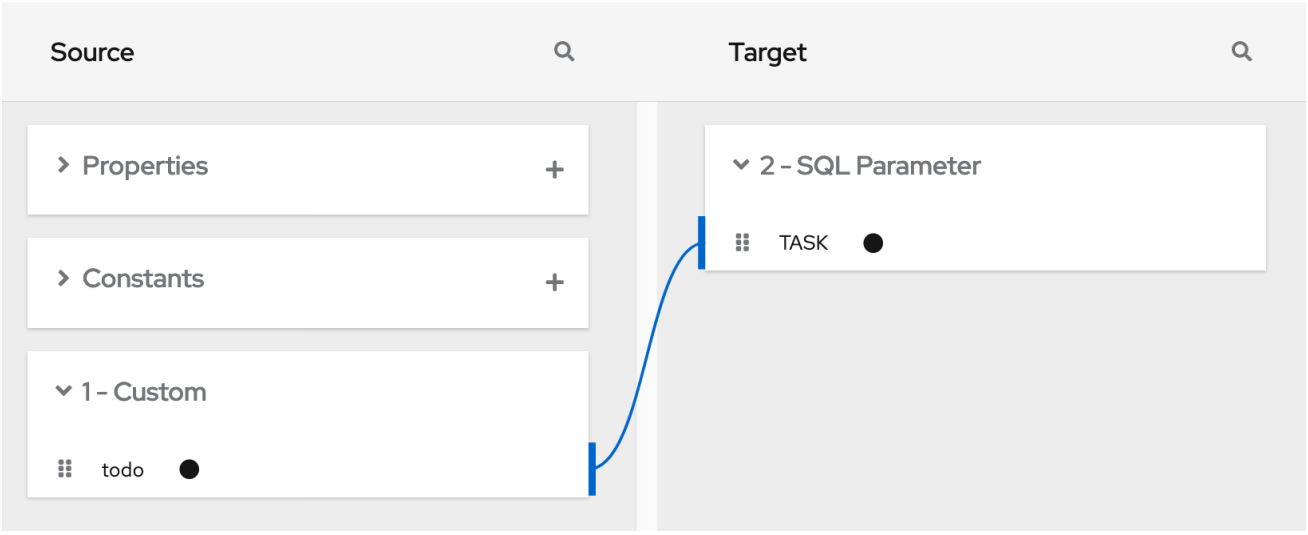
6.6. 如何指定 HTTP 请求

以下示例演示了如何为 Fuse Online Webhook 指定 HTTP 请求。

仅使用 HTTP 正文的 POST 请求的 Webhook 示例

考虑一个集成，它以 Webhook 连接开头，然后在 Fuse Online 提供的数据库的 Todo 表中创建一个行：

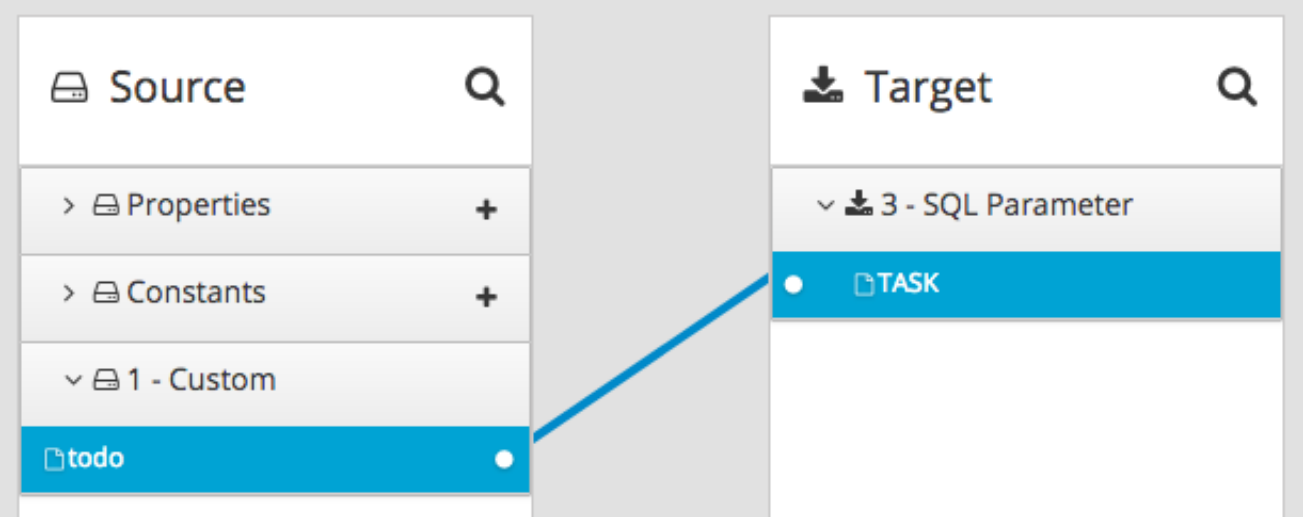
在创建此集成时，当添加 Webhook 启动连接时，您可以使用带有此内容的 JSON 实例指定其输出数据类型： **{"setuptools":"text"}:**



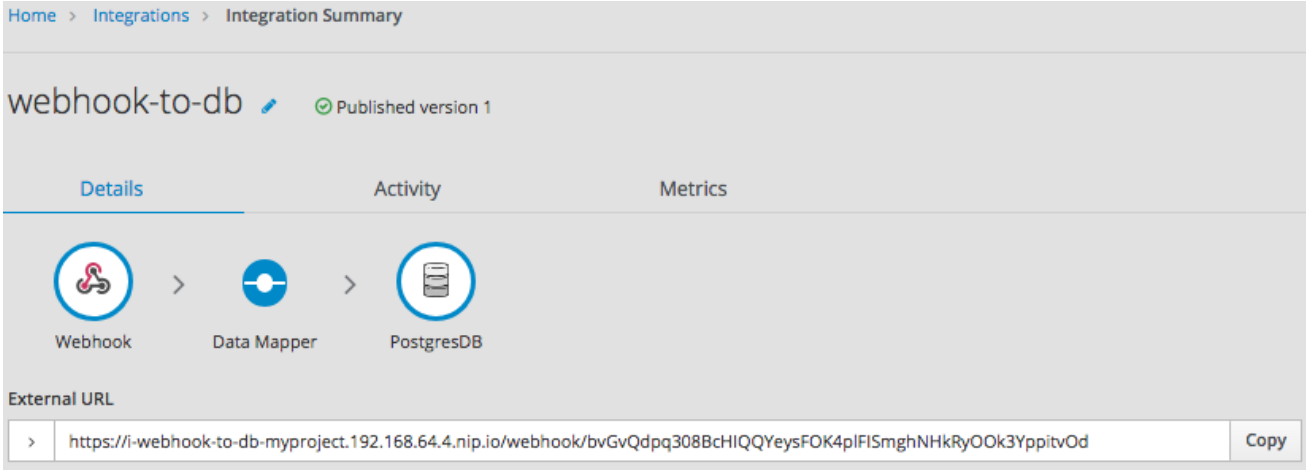
当您将 PostgresDB 连接添加为完成连接时，您可以选择 Invoke SQL 操作并指定此 SQL 语句：

插入 TODO (TASK)值(:#TASK)

添加数据库连接后，添加一个映射步骤：



您保存并发布集成。当它运行时，您可以复制 Fuse Online 提供的外部 URL：



要了解外部 URL 的部分，请考虑以下示例 URL：

https://i-webhook-to-db-myproject.192.168.64.4.nip.io/webhook/bvGvQdpq308BcHIQQYeysFOK4plFlSmghNHkRyOOk3YppitvOd

值	描述
i-	Fuse Online 始终在 URL 的开头插入这个值。
webhook-to-db	集成的名称。
myproject	包含运行集成的 pod 的 OpenShift 命名空间。
192.168.64.4.nip.io	为 OpenShift 配置的 DNS 域。这表示提供 webhook 的 Fuse Online 环境。
webhook	出现在每个 Webhook 连接 URL 中。
bvGvQdpq308BcHIQQYeysFOK4plFlSmghNHkRyOOk3YppitvOd	在向集成添加 Webhook 连接时，Fuse Online 提供的 Webhook 连接令牌。令牌是一个随机字符串，用于提供安全性，从而使 URL 难以识别，这样可防止其他人发送请求。 在请求中，您可以指定 Fuse Online 提供的令牌，或者您可以自行定义。如果您自行定义，请确保难以猜测。

正如您在外部 URL 中看到的，Fuse Online 从集成的名称、OpenShift 命名空间的名称和 OpenShift DNS 域构造主机名。Fuse Online 删除非法字符，并将空格转换为连字符。在外部 URL 示例中，这是主机名：

https://i-webhook-to-db-myproject.192.168.64.4.nip.io

要使用 curl 调用 Webhook，您可以指定命令，如下所示：

```
curl -H 'Content-Type: application/json' -d '{"setuptools":"from webhook"}' https://i-webhook-to-db-myproject.192.168.64.4.nip.io/webhook/bvGvQdpq308BcHIQQYeysFOK4plFlSmghNHkRyOOk3YppitvOd
```

- -H 选项指定 HTTP Content-Type 标头。

- 默认情况下，d 选项将 HTTP 方法设置为 POST。

执行此命令会触发集成。数据库完成连接会在 tasks 表中插入新任务。要查看这一点，请在其中显示 Todo 应用程序，例如 https://todo-myproject.192.168.64.4.nip.io，点 Update，您应该 从 Webhook 看到为新任务。

带有查询参数的 POST 请求的 Webhook 示例

在本例中，请考虑与上例中的相同集成：

但是，在这个示例中，您可以通过指定使用此内容的 JSON 模式来定义 Webhook 连接输出数据类型：

```
{
```

```
  "type": "object",
  "definitions": {},
  "$schema": "http://json-schema.org/draft-07/schema#",
  "id": "io:syndesis:webhook",
  "properties": {
    "parameters": {
      "type": "object",
      "properties": {
        "source": {
          "type": "string"
        },
        "status": {
          "type": "string"
        }
      }
    },
    "body": {
      "type": "object",
      "properties": {
        "company": {
          "type": "string"
        },
        "email": {
          "type": "string"
        },
        "phone": {
          "type": "string"
        }
      }
    }
  }
}
```

在这个 JSON 模式中：

- id 必须设置为 io.syndesis.webhook。
- parameters 部分必须指定 HTTP 查询参数。
- body 部分必须指定正文内容，并可像您需要一样复杂。例如，它可以定义嵌套属性以及数组。

这提供了 Webhook 连接器需要为集成中的下一步准备内容所需的信息。

要使用 curl 发送 HTTP 请求，请调用如下命令：

```
curl -H 'Content-Type: application/json' -d
'{"company":"Gadgets","email":"sales@gadgets.com","phone":"+1-202-555-0152"}'https://i-
webhook-params-to-db-
myproject.192.168.42.235.nip.io/webhook/ZYWrhaW7dVk097vNsLX3YJ1GyxUFMFRteLpw0z4O69
MW7d2Kjg?source=web&status=new
```

当 Webhook 连接收到此请求时，它会创建一个 JSON 实例，如下所示：

```
{
  "parameters": {
    "source": "web",
    "status": "new"
  },
  "body": {
    "company": "Gadgets",
    "email": "sales@gadgets.com",
    "phone": "+1-202-555-0152"
  }
}
```

这是启用以下映射的内部 JSON 实例：

带有 GET 的 Webhook 示例

要触发与不提供输入数据的 GET 请求集成，请将 Webhook 连接输出数据指定为带有定义 '{}' 的 JSON 实例。然后，您可以调用以下 curl 命令，该命令没有指定查询参数：

```
curl 'https://i-webhook-params-to-db-myproject.192.168.42.235.nip.io/webhook/ZYWrhaW7dVk097vNsLX3YJ1GyxUFMFRteLpw0z4O69MW7d2Kjg'
```

您可以更改前面的 POST 示例，以使用查询参数发送 GET 请求，但没有正文。您将以 JSON 模式的形式指定 Webhook 连接输出数据，如下所示。

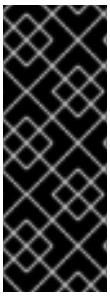
```
{
  "type": "object",
  "definitions": {},
  "$schema": "http://json-schema.org/draft-07/schema#",
  "id": "io:syndesis:webhook",
  "properties": {
    "parameters": {
      "type": "object",
      "properties": {
        "source": {
          "type": "string"
        },
        "status": {
          "type": "string"
        }
      }
    }
  }
}
```

以下是发送 GET 请求的 curl 命令：

```
curl 'https://i-webhook-params-to-db-myproject.192.168.42.235.nip.io/webhook/ZYWrhaW7dVk097vNsLX3YJ1GyxUFMFRteLpw0z4O69MW7d2Kjg?source=web&status=new`
```

第 7 章 将集成数据映射到下一连接的字段

在大多数流中，您需要将已获取或处理的数据字段映射到流中下一连接可以处理的数据字段。Fuse Online 提供了一个数据映射程序，可帮助您做到这一点。在流中，在您需要映射数据字段的每个点添加一个数据映射程序步骤。



重要

数据映射器显示由上一集成步骤提供的最大可能的源字段集合。但是，并非所有连接都会在显示的源字段中提供数据。例如，对第三方应用程序的更改可能会停止在特定字段中提供数据。当您创建集成时，如果您注意到数据映射没有如预期那样的行为，请确保要映射的 source 字段包含您预期的数据。

映射数据字段的详情位于以下主题：

- 第 7.1 节 “查看步骤中的映射”
- 第 7.2 节 “识别需要数据映射的位置”
- 第 7.3 节 “查找您要映射的数据字段”
- 第 7.4 节 “将一个 source 字段映射到一个目标字段”
- 第 7.5 节 “合并或分离字段时缺少或不需要的数据示例”
- 第 7.6 节 “将多个源字段合并到一个目标字段中”
- 第 7.7 节 “将一个 source 字段分隔成多个目标字段”
- 第 7.8 节 “关于数据映射程序中的数据类型和集合”

- 第 7.9 节 “使用数据映射器来处理集合”
- 第 7.10 节 “关于集合和非集合之间的映射”
- 第 7.11 节 “转换源或目标数据”
- 第 7.12 节 “在映射到一个目标字段前，关于在多个源值上的转换”
- 第 7.13 节 “应用条件来映射”
- 第 7.14 节 “可用转换的描述”
- 第 7.15 节 “数据映射故障排除”

7.1. 查看步骤中的映射

在添加或编辑数据映射程序步骤时，您可以查看此步骤中已定义的映射。这个选项可让您检查正确的映射是否就位。

先决条件

- 您正在创建或编辑集成。
- 您正添加数据映射程序步骤。也就是说，数据映射程序可见。

流程

1. 要查看映射的表视图，请点  。
2. 要忽略映射的表视图，并重新显示源和目标字段，请点

。

7.2. 识别需要数据映射的位置

Fuse Online 显示警告图标，以指示流需要数据映射的位置。

先决条件

- 您正在创建或编辑流。
- 流包含它所需的所有连接。

流程

1. 在流视觉化中，查找任何

图标。
2. 点图标查看 Data Type Mismatch 通知。
3. 在消息中，单击 Add a data mapping step，其中显示数据映射器。

7.3. 查找您要映射的数据字段

在相对较少的步骤的流程中，映射数据字段很容易且直观。在处理大量数据字段的流中，当您有一些有关如何使用数据映射器的背景时，从源映射到目标的过程更为简单。

数据映射器显示两列数据字段：

- source 是数据字段的列表，可在流程中的所有前面步骤中获取或处理。

- target 是流中下一连接期望的数据字段列表，并可处理。

要快速找到您要映射的 data 字段，您可以执行以下操作之一：

- 搜索它。

  Sources 面板和 Target 面板在顶部都有一个搜索字段。如果搜索字段不可见，请点击 Sources 或 Target 面板右上角的

  。

  键入您要映射的字段的名称。在 Sources 搜索字段中，输入 source 字段的名称。在 Target 搜索字段中，键入您要映射到的字段的名称。

- 使用

  和

  选项过滤可见的字段。

- 展开和折叠文件夹，以限制可见字段。

  要查看特定步骤中可用的数据字段，请展开该步骤的文件夹。

  当您向流中添加步骤时，Fuse Online number 和 renumbers 指示 Fuse Online 处理步骤的顺序。当您添加数据映射程序步骤时，步骤号会出现在 Sources 面板和 Target 面板中的文件夹标签中。

- 如果要查看每个字段的数据类型，点

  显示（或隐藏）每个字段标签中的数据类型。folder 标签还显示该步骤输出的数据类型的名称。folder 标签还显示该步骤输出的数据类型的名称。与 Twitter、Salesforce 和 SQL 等应用程序的连接定义了自己的数据类型。要连接到 Amazon S3、AMQ、AMQP、Dropbox 和 FTP/SFTP 等应用程序，您可以在将连接添加到流时定义连接的输入和输出和/或输出类型，然后选择连接执行的操作。当您指定数据类型时，您也为类型指定一个名称。您指定的类型名称显示

为 data mapper 中文件夹的名称。如果您在声明数据类型时指定了描述，则当您将鼠标悬停在映射程序中的 step 文件夹上时，类型描述会显示。

7.4. 将一个 SOURCE 字段映射到一个目标字段

默认映射行为将一个 source 字段映射到一个目标字段。例如，将 Name 字段映射到 CustomerName 字段。

流程

1. 在 Sources 面板中：
   a. 如有必要，扩展步骤以查看它提供的数据字段。

      当有很多源字段时，您可以通过点  并在搜索字段中输入 data 字段来搜索感兴趣的字段。

   b. 点击您要从中映射的数据字段，然后点  。此时会打开 Mapping Details 面板。

2. 在 Target 面板中，找到您要映射到的数据字段，然后点  。

   数据映射器显示连接您选择的两个字段的行。

3. （可选）预览数据映射结果。当您向映射添加转换或映射需要类型转换时，这个选项很有用。
   a. 在数据映射程序的右上角，点  在 source 字段中显示文本输入字段，在 target 字段中显示只读结果字段。

   b. 在 source 字段的数据输入字段中，输入示例输入值。映射结果会出现在 target 字段中

的 read-only 字段中。

c. 另外，要查看转换的结果，请在 Mapping Details 面板中添加一个转换。

d. 要隐藏预览字段，请再次点  。

4. 另外，要确认定义了映射，请点击  以显示定义的映射。

您也可以预览数据映射结果（在此视图中）。如果预览字段不可见，点  。按照上一步中所述输入数据。

+ 在定义的映射表中，preview 字段仅显示所选映射。

a. 要查看另一个映射的预览字段，请选择它。

b. 再次点击  以显示数据字段面板。

5. 在右上角，点 Done 将数据映射程序步骤添加到集成。

故障排除提示

数据映射器显示由上一集成步骤提供的最大可能的源字段集合。但是，并非所有连接都会在显示的源字段中提供数据。例如，对第三方应用程序的更改可能会停止在特定字段中提供数据。当您创建集成时，如果您注意到数据映射没有如预期那样的行为，请确保要映射的 source 字段包含您预期的数据。

7.5. 合并或分离字段时缺少或不需要的数据示例

在数据映射中，当 source 或 target 字段包含复合数据时，您可能需要识别缺少的或不需要的数据。例

如，假设一个具有此格式的 long_address 字段：

Number street apartment city state zip zip+4 country

假设您要将 long_address 字段分成多个离散字段，表示 数字、street、city、state 和 zip。要做到这一点，您可以选择 long_address 作为 source 字段，然后选择目标字段。然后，您可以在您不想的 source 字段的部分位置添加 padding 字段。在本例中，不需要的部分是、*zip+4* 和 *国家/地区*。

要识别不需要的部分，您需要了解部分的顺序。顺序在 compound 字段中指示内容的每个部分的索引。例如，long_address 字段有 8 个排序的部分。从 1 开始，每个部分的索引是：

1	*number*
2	*street*
3	*apartment*
4	*city*
5	*state*
6	*zip*
7	*zip+4*
8	*国家*

在数据映射程序中，要识别缺少的、zip+4 和 country，您可以在索引 3、7 和 8 中添加 padding 字段。请参阅将多个源字段组合为一个目标字段。

现在假设您想将源字段组合为 数字,street,city,state, 和 zip to a long_address target 字段。进一步假设没有源字段来为part ment、zip+4 和 country 提供内容。在数据映射程序中，您需要根据需要识别这些字段。同样，您可以在索引 3、7 和 8 中添加 padding 字段。请参阅 将一个 source 字段 9 个到多个目标字段中。

7.6. 将多个源字段合并到一个目标字段中

在数据映射程序步骤中，您可以将多个源字段组合为一个复合目标字段。例如，您可以将 FirstName 和 LastName 字段映射到 CustomerName 字段。

前提条件

对于 target 字段，您必须了解此复合字段中的每个部分的内容类型、内容的每个部分的顺序和索引，以及部分之间的分隔符，如空格或逗号。请参阅 缺少或不需要的数据示例。

流程

1. 在 Target 面板中，点您要将多个源字段映射到的字段，然后点

。此时会打开 Mapping Details 面板。

2. 在 Mapping Details 面板中，从 Source 下拉列表中选择您要映射到的一个或多个数据字段。

完成后，您应该从每个 source 字段看到一行到 target 字段。

在 Mapping Details 面板中，Data mapper 会显示默认的多重性转换，即 Concatenate。这表示映射的执行将 Concatenate 转换应用到所选源字段中的值，并将串联值映射到所选目标字段。

注意

有关您可以应用到多个源值的其他转换的详情，请参考

3. 在 Mapping Details 面板中，配置映射，如下所示：

a. 在 Sources 下，在 Delimiter 字段中，接受或选择不同源字段的内容在 target 字段中插入的字符。默认为一个空格。

b. 另外，在每个 source 字段条目中，您可以在映射到 target 字段前点

将转换应用到 source 字段值。

c. 在 Sources 下，检查您选择的 source 字段条目的顺序。条目必须与 compound 目标字段中的对应内容相同。

如果条目没有正确顺序，请更改字段条目的索引号，以实现相同的顺序。

如果您将 source 字段映射到 compound target 字段的每个部分，请跳过下一步。

d. 对于没有与 target 字段中对应数据相同的索引的每个 source 字段条目，请将索引编辑为相同。每个 source 字段条目都必须具有与 target 字段中对应数据相同的索引。数据映射程序根据需要自动添加 padding 字段来指示缺少数据。

如果您意外创建太多 padding 字段，请为每个额外的 padding 字段点

来删除它。

e. （可选）在 Targets 下，点

将内容映射到目标字段，然后应用转换，如 转换源或目标数据 中所述。

4. 另外，还可预览数据映射结果：

a. 点

在当前所选映射的每个源字段中显示一个文本输入字段，在当前所选映射的目标字段中显示一个只读结果字段。

b. 在源数据输入字段中，输入示例值。

如果您重新排序源字段或向映射添加转换，则 target 字段中的 result 字段会显示它。如果数据映射器检测到任何错误，它会在 Mapping Details 面板的顶部显示信息信息。

c. 再次单击

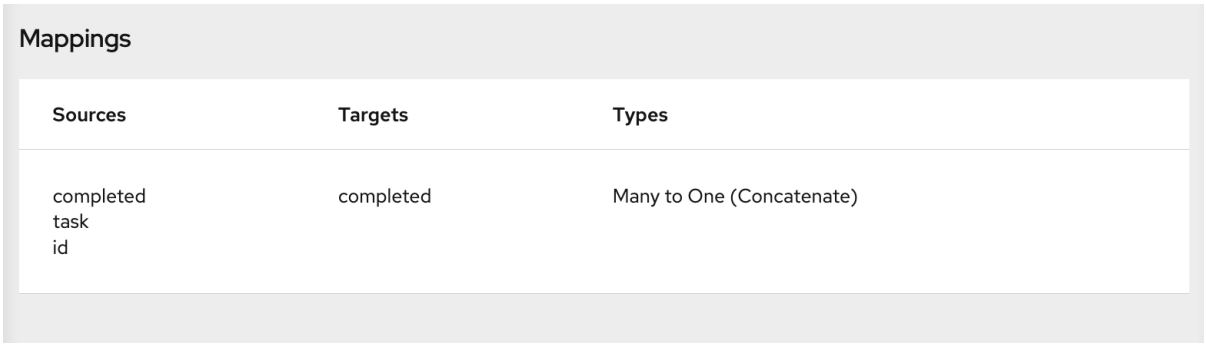
来隐藏预览字段。

如果您重新显示 preview 字段，您输入的任何数据仍会存在，且会一直存在，直到您退出数据映射程序。

5.

要确认正确定义了映射，点

显示（表格式）在此步骤中定义的映射。将多个 source 字段的值合并到一个 target 字段中的映射如下所示：



。

您还可以在此视图中预览映射结果。点

，然后输入文本，如上一步中所述。Preview 字段仅显示所选映射。点表中的另一个映射查看预览字段。

其他资源

将 padding 字段添加示例：将一个 source 字段 9 个添加到多个目标字段中。

虽然该示例适用于一对多的映射，但原则是相同的。

7.7. 将一个 SOURCE 字段分隔成多个目标字段

在数据映射程序步骤中，您可以将 compound source 字段分成多个目标字段。例如，将 Name 字段映射到 FirstName 和 LastName 字段。

前提条件

对于 source 字段，您必须了解这个 compound 字段中的每个部分的内容类型、内容的每个部分的顺序和索引，以及部分之间的分隔符，如空格或逗号。请参阅 缺少或不需要的数据示例。

流程

1. 在 Sources 面板中，点您要独立内容的字段，然后点

。

2.

在 Mapping Details 面板中，从 Target 下拉列表中选择您要映射到的数据字段。

选择目标字段后，您应该看到 source 字段中的行到您选择的每个目标字段。

在 Mapping Details 面板的顶部，数据映射器会显示 Split 来指示映射的执行分割源字段值并将其映射到多个目标字段。* 在 Targets 下，您选择的每个目标字段都有一个条目。

3.

在 Mapping Details 面板中，配置映射，如下所示：

a.

在 Sources 下，在 Delimiter 字段中接受或选择 source 字段中的字符，指示 source 字段值分开的位置。默认为一个空格。

b.

（可选）点

将转换应用到 source 字段值，然后再映射到 target 字段。

c.

在 Targets 下，检查您选择的目标字段条目的顺序。条目必须与 compound source 字段中对应的内容相同。在 source 字段中，您没有为一个或多个部分指定 target 字段无关紧要。

如果条目没有正确顺序，请更改字段条目的索引号，以实现相同的顺序。

如果您将 compound source 字段的每个部分映射到 target 字段，则跳至下一步。

d.

如果 source 字段包含您不需要的数据，然后在 Mapping Details 面板中编辑每个目标字段的索引，它们没有与 source 字段中对应的数据相同的索引。每个目标字段条目都必须具有与 source 字段中对应数据相同的索引。数据映射程序根据需要自动添加 padding 字段，以指示不需要的数据。

请参阅此流程末尾的示例。

e.

（可选）点

将内容映射到目标字段，然后应用转换，如 转换源或目标数据 中所述。

4. 另外，还可预览数据映射结果：

   a. 点击
   
   在源字段中显示一个文本输入字段，在每个 target 字段中显示只读结果字段。

   b. 在 source 字段的数据输入字段中，输入 smaple 值。务必在字段的部分之间输入分隔符字符。映射结果会出现在目标字段的只读字段中。

   如果您重新排序目标字段或向目标字段添加转换，则目标字段上的结果字段反映了这种情况。如果数据映射器检测到任何错误，它会在 Mapping Details 面板的顶部显示信息信息。

   c. 再次单击
   
   来隐藏预览字段。

   如果您重新显示 preview 字段，您输入的任何数据仍会存在，且会一直存在，直到您退出数据映射程序。

5. 要确认正确定义了映射，点
   
   显示此步骤中定义的映射。将 source 字段的值分隔到多个目标字段的映射如下所示：

Mappings

Sources	Targets	Types
completed	completed id task	One to Many (Split)

。

您还可以在此视图中预览映射结果。点


，然后键入文本，如上一步中所述。Preview 字段仅显示所选映射。点表中的另一个映射查看预览字段。

将一个字段分离到多个字段的示例

假设源数据包含一个地址字段，它使用逗号分隔内容部分，例如：

```
77 Hill Street, Brooklyn, New York, United States, 12345, 6789
```

在地址字段中，内容的部分内容有这些索引：

内容	索引
number 和 street	1
City	2
状态	3
国家	4
zip 代码	5
Zip+4	6

现在假设目标数据有四个字段用于地址：

```
number-and-street
city
state
zip
```

要定义映射，您可以执行以下操作：

- 选择 **source 字段，然后点**
  
  。

- 在 Mapping Details 面板中，在 Sources 部分中选择 delimiter，本例中为逗号。

- 选择四个目标字段。

完成此操作后，在 Targets 下的 Mapping Details 面板中，您选择的每个目标字段都有一个条目，例如：

.

数据映射器按数据映射器中显示的目标条目（按字母排序）显示目标条目。您需要更改此顺序，以便在 source 字段中镜像顺序。在本例中，source 字段在 city 内容前包含 number-and-street 内容。要更正目标条目的顺序，请将 city index 字段编辑为 2。结果类似如下：

.

在 target 字段条目中，索引号指示将映射到此目标字段的 source 字段的一部分。需要更改其中一个索引值来实现正确的目标字段值。考虑每个目标字段：

-

number-and-street - 在源字段中，number 和 street 内容具有索引 1。索引 1 源映射到 number-and-street 目标字段是正确的。此目标条目不需要任何更改。

- City - 在源字段中，城市内容的索引为 2。此目标条目也正确。
- State - 在 source 字段中，状态内容的索引为 3。此目标条目也正确。
- zip - 在 source 字段中，zip 代码内容的索引为 5。目标字段条目索引为 4 错误。如果您没有更改它，在执行期间，source 字段中的国家/地区部分将映射到 zip 目标字段。您需要将索引更改为 5。这指示数据映射器将索引 5 源内容映射到 zip 目标字段。更改索引后，数据映射器会添加一个带有索引 4 的 padding 字段。结果类似如下：

.

这个映射现已完成。虽然 source 字段在索引 6 (zip+4)上具有额外的内容，但目标不需要数据，且无需进行任何操作。

7.8. 关于数据映射程序中的数据类型和集合

在数据映射程序中，字段可以是：

- 存储单个值的 原语 类型。原语类型的示例包括 布尔值、char、字节、短语、int、长、浮点和 双引号。原语类型无法扩展，因为它是一个字段。
- 由不同类型的多个字段组成 的复杂 类型。您可以在设计时定义复杂类型的子字段。在数据映

射程序中，一个复杂的类型可以被扩展，以便您可以查看其子字段。

每种类型的字段（原语和复杂）也可以是集合。集合是一个可以具有多个值的单个字段。集合中的项目数量在运行时决定。在设计时，在数据映射程序中，集合由

表示。在数据映射程序界面中可以扩展集合是否由其类型决定。当集合是一个原语类型时，它不可扩展。当集合是一个复杂的类型时，数据映射程序可以被扩展，以显示集合的子字段。您可以从/到每个字段映射。

以下是一些示例：

- *ID 是原语类型字段(int)。在运行时，员工只能有一个 ID。例如，ID=823。因此，ID 是不是集合的原语类型。在数据映射程序中，ID 无法扩展。*

- *电子邮件 是一个原语类型字段（字符串）。在运行时，员工可以有多个 电子邮件 值。例如：email<0>=aslan@home.com 和 email<1>= aslan@business.com。因此，电子邮件 也是集合的原语类型。数据映射程序使用*

  表示 电子邮件 字段是一个集合，但 电子邮件 无法扩展，因为它是一个原语类型（没有子字段）。

- *员工 是一个复杂的对象字段，它具有多个子字段，包括 ID 和 电子邮件。在运行时，员工 也是集合，因为公司有许多员工。*
  在设计时，数据映射程序使用

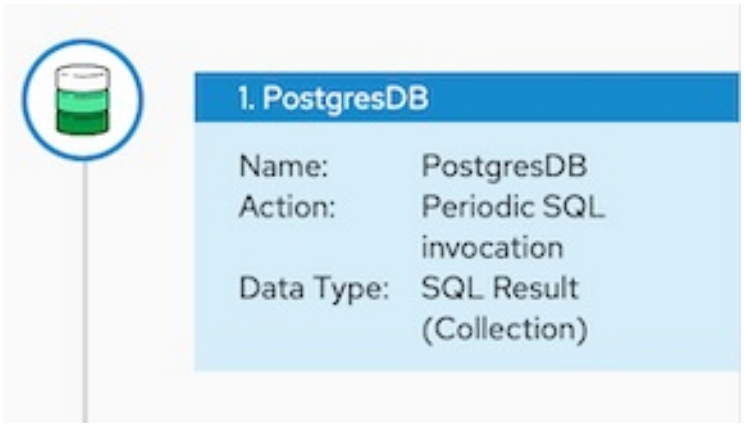
  

  来表示 员工 是一个集合。employee 字段可以扩展，因为它是一个具有子字段的复杂类型。

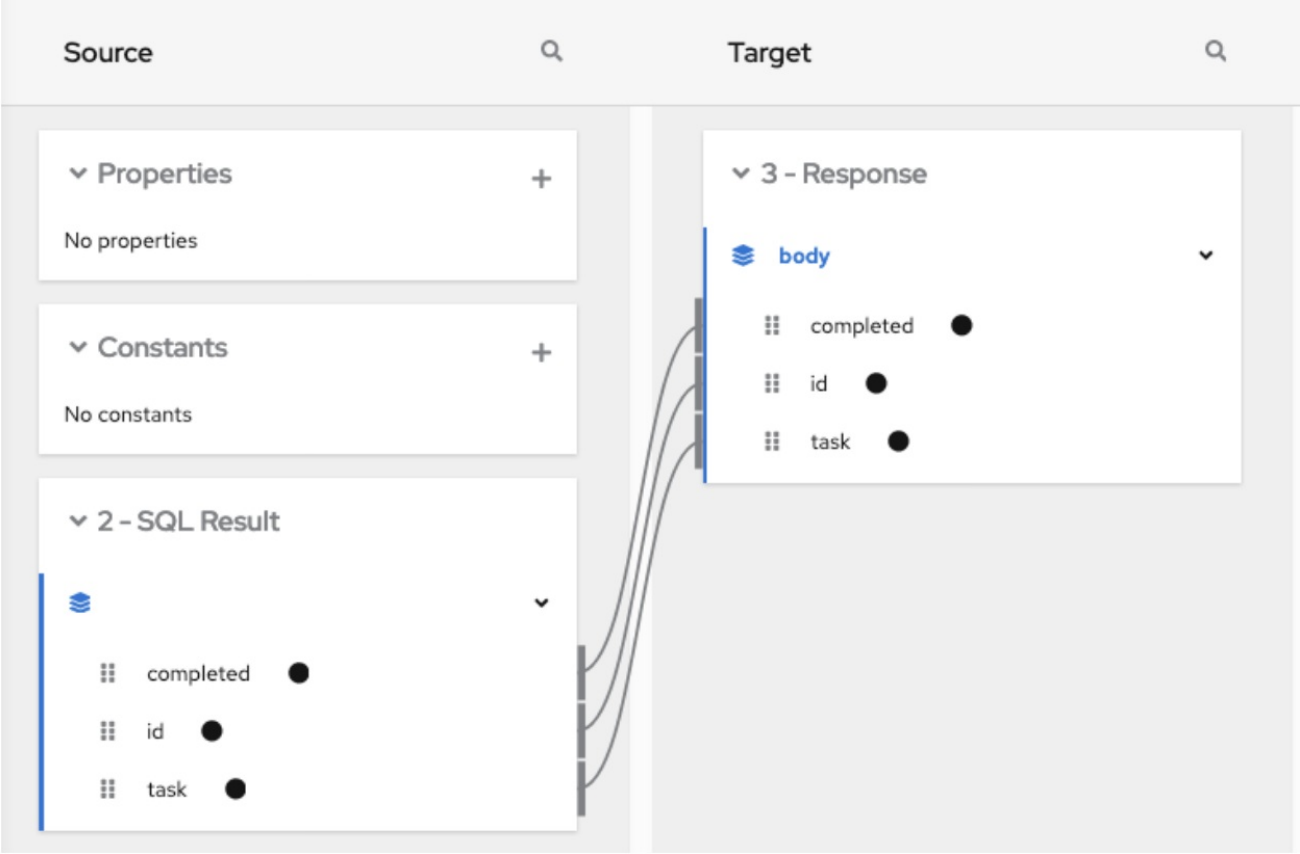
7.9. 使用数据映射器来处理集合

在流中，当步骤输出集合以及流中的后续连接时，您可以使用数据映射器来指定流如何处理集合。

当步骤输出集合时，流程视觉化会在步骤的详情中显示 Collection。例如：



在提供集合的步骤后添加数据映射程序步骤，并在需要映射的步骤之前添加。准确在这个数据映射程序步骤中，需要依赖于流程中的其他步骤。下图显示了从源集合字段到目标集合字段的映射：



在源和目标面板中，数据映射器会显示

来表示集合。

当集合是一个复杂的类型时，数据映射器会显示集合的子字段。您可以从/到每个字段映射。

当 source 字段嵌套在多个集合中时，您可以将其映射到满足其中一个条件的目标字段：

- target 字段嵌套在与 source 字段相同的集合数量中。例如，允许这些映射：
  - /A<>/B<>/C → /D<>/E<>/F
  - /A<>/B<>/C → /G<>/H/I<>/J
- target 字段仅嵌套在一个集合中。例如，允许这个映射：

  /A<>/B<>/C → /K<>/L

  在这种情况下，数据映射器使用深度优先算法来迭代源中的所有值。因此，数据映射程序会将源值放入单个目标集合中。

不允许以下映射：

/a<>/B<>/C cannot-map-to /M<>/N/O<>/P<>/Q

当 Fuse Online 执行流时，它会迭代源集合元素来填充目标集合元素。如果您将一个或多个源集合字段映射到目标集合字段或目标集合字段，则目标集合元素仅包含映射字段的值。

如果您将源集合中的字段映射到不在集合中的 target 字段，那么当 Fuse Online 执行流时，它将只分配源集合中最后一个元素的值。该映射步骤中会忽略集合中的任何其他元素。但是，任何后续映射步骤都可以访问源集合中的所有元素。

当连接返回 JSON 或 Java 文档中定义的集合时，数据映射器通常可以将源文档处理为集合。

7.10. 关于集合和非集合之间的映射

在数据 映射程序 源和目标 面板中 ：

- 

  表示集合。如果集合包含一个原语类型，您可以直接从或映射到该集合。如果集合包含两个或多个不同类型的类型，数据映射器会显示集合的子字段，您可以映射到集合的字段或从集合的字段映射。

- 

  表示是一个复杂类型的可扩展容器。复杂的类型包含不同类型的多个字段。复杂类型中的一个字段可以是集合的类型，如数组。您无法映射复杂类型容器本身。您只能映射复杂类型中的字段。

要切换数据类型的显示，如 (COMPLEX), STRING, INTEGER，点

。

集合到非集合（多到一）映射

当您从 collection 字段映射到非集合字段时，数据映射器可以识别多对一的映射。默认行为是数据映射程序将 Concatenate 转换应用到源集合或源集合字段。默认分隔符是一个空格。例如，考虑这个源集合：

- 在第一个元素中，城市 字段中的值为 Boston。
- 在第二个元素中，ity 字段中的值 是 Paris。
- 在第三个元素中，city 字段的值为 。

在执行过程中，数据映射程序使用填充目标字段

塞尔兰西亚哥伦敦

您可以通过应用不同的转换来更改此默认行为。例如，若要仅从您选择的元素映射，请应用 Item At transformation to the source 并指定索引。要映射源集合中第一个元素中的值，请为索引指定 0。

如果源集合包含您没有映射的字段，这些字段仍可提供给流中的后续步骤。

非集合到集合（一对多）映射

当您从非集合 source 字段映射到目标集合或位于 collection 元素中的 target 字段时，数据映射器会识别一对多映射。默认行为是，数据映射程序通过将空格用作分隔符并将源值拆分为多个值来应用 Split 转换。在执行过程中，数据映射器会将每个分割值插入到目标集合中自己的元素中。例如，如果 source 字段被分成 4 个值，则目标集合有 4 个元素。

在本发行版本中，Split 转换是您可以应用到一对多映射的唯一转换。

例如，考虑一个包含以下内容的非 集合 cities source 字段：

塞尔兰西亚哥伦敦

您可以将此 source 字段映射到目标集合或位于集合中的 target 字段。在执行期间，数据映射器会将 cities 字段的值 分成空格分隔符。结果是一个包含三个元素的集合。在第一个元素中，城市 字段的值是 Boston。在第二个元素中，ity 字段的值 为 Paris。在第三个元素中，qity 字段的值 为 。

其他资源

- 在映射到一个目标字段前，关于在多个源值上的转换
- 将多个源字段合并到一个目标字段中
- 将一个 source 字段分隔成多个目标字段

7.11. 转换源或目标数据

在数据映射程序中，在定义映射后，您可以转换映射中的任何字段。转换数据字段定义了如何存储数据。例如，您可以指定 Capitalize 转换，以确保 data 值的第一个字母大写。

注： 如果要向映射添加条件，您需要将任何转换放在条件表达式中，如 应用条件到映射 中所述。

流程

1. **映射字段。这可以是一对一的映射、组合映射或分离映射。**

2. **在 Mapping Details 面板中，在 Sources 或 Targets 下，在您要转换的字段的框中，点**

。这个选项显示可用转换的下拉列表。

3. 选择**您要执行数据映射程序的转换。**

4. **如果转换需要任何输入参数，请在适当的输入字段中指定它们。**

5. **要添加另一个转换，请再次点**

。

其他资源

- **可用的转换**

- **在映射到一个目标字段前，关于在多个源值上的转换**

7.12. 在映射到一个目标字段前，关于在多个源值上的转换

有些转换可用于多个源字段，或应用到包含多个值的 source 字段中的值，如集合。数据映射器将转换的结果插入到 target 字段中。下表描述了这些多重性转换。

多重性转换	描述
添加	添加数字源值，并将 sum 插入到 target 字段中。所选源字段中或所选集合中的值必须是数字。
average	计算数字源值的平均值，并将结果插入到 target 字段中。所选源字段中或所选集合中的值必须是数字。

多重性转换	描述
concatenate	加入源值，并将结果插入到 target 字段中。您可以接受空格作为分隔符或指定其他字符。数据映射器在源值之间的 target 字段中插入此字符。此转换的一个常见用途是组合多个源字段值，例如，firstName、**distributedName** 和 **LastName**，在一个目标字段中，例如 **CustomerName**。
contains	评估源值，以确定任何值是否包含您指定的参数值。如果任何源值包含指定的参数值，则 data mapper 会将 **true** 插入到 target 字段中。如果没有 source 值，则 data mapper 会将 **false** 插入到 target 字段中。 例如，假设您要跟踪与特定客户相关的活动。您可以选择每个集合成员包含客户信息的源集合字段。对于 **Value** 参数，您可以指定一个特定的电子邮件地址。当数据映射器在集合中找到指定的电子邮件地址时，它会在 target 字段中插入 **true**。
数量	在 target 字段中插入源值数。当 source 字段是集合时，这很有用。数据映射器在 target 字段中插入集合的大小。 例如，假设您选择一个是项目对象的 **Order** source 字段。应用 **Count** 转换会将按顺序排列的项数插入到目标字段中。 另外，如果您选择 4 个单独的源字段，则数据映射器会在 target 字段中插入 **4**。
划分	将第一个源值除以第二个源值，并在 target 字段中插入结果。如果还有两个源值，则执行将继续根据下一个数字来划分结果。例如，假设一个包含 **{1000, 100, 10}** 的 **numbers[]** 集合。执行将 **1000** 分为 **100** 以获得 **10**，然后将 **10** 分成 **10** 以获得 **1**。映射器在 target 字段中插入 **1**。
格式	将模板中的占位符替换为您选择的 source 字段中的值。数据映射器在 target 字段中插入生成的字符串。例如，假设您选择了三个源字段： **时间** **名称** **文本** 您可以选择 **Format** 转换并在 **Template** 参数中指定： **在 $time, $name tweeted: $text** 在 target 字段中，结果类似如下： **上午 8:00 时，Aslan tweeted: ROAR!** 这与 Java 和 C 等编程语言提供的机制类似。
项目期间	对于您选择的 source 字段，数据映射器会在您在 target 字段中指定并插入该值的索引中找到值。source 字段必须是包含带有分隔符的多个值的集合或字段。 例如，假设所选 source 字段是客户电子邮件地址的集合。在选择了 **Item At** transformation 后，在 **Index** parameter 字段中，您可以指定 **0**。数据映射器在 target 字段中插入第一个电子邮件地址，该地址为 index 0。
最大值	评估源值，并在 target 字段中插入最高值。源值必须是数字。

多重性转换	描述
最小值	评估源值，并在 target 字段中插入最低值。源值必须是数字。
multiply	将第一个源值乘以第二个源值，并在 target 字段中插入结果。如果还有两个源值，则执行将继续乘以下一个数字的结果。例如，假设一个包含 **{10, 100, 1000}** 的 **numbers[]** 集合。执行乘以 **10** 到 **100** 以获得 **1000**，然后将 **1000** 乘以 **1000** 以获得 **1000000**。映射器在 target 字段中插入 **1000000**。
减法	从第一个源值中减去第二个源值，并在 target 字段中插入结果。如果还有两个源值，则执行将继续从上一个结果中减去下一个数字。例如，假设一个包含 **{100, 90, 9}** 的 **numbers[]** 集合。执行从 **100** 减去 **90** 以获得 **10**，然后将 **9** 从 **10** 减去以获得 **1**。映射器在 target 字段中插入 **1**。

其他资源

- ***将多个源字段合并到一个目标字段中***
- ***将一个 source 字段分隔成多个目标字段***

7.13. 应用条件来映射

在某些集成中，向映射添加条件处理会很有帮助。例如，假设您将 source zip code 字段映射到目标 zip code 字段。如果 source zip code 字段为空，您可能需要使用 99999 填写目标字段。要做到这一点，您可以指定一个表达式来测试 zip 代码源字段，以确定它是否为空，如果为空，将 99999 插入 zip 代码目标字段。

重要

应用条件到映射仅是一项技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。有关红帽技术预览功能支持范围的详情，请参考 https://access.redhat.com/support/offerings/techpreview/。

数据映射程序支持与 Microsoft Excel 表达式类似的表达式，但不支持所有 Microsoft Excel 表达式语法。条件表达式可以引用单个字段或集合中的字段。

您可以为每个映射定义零个或一个条件。

以下流程从应用条件开始进行映射。

注：将条件添加到映射后，Source 和 Target 转换选项将被禁用。您必须将任何转换放在条件表达式中。

先决条件

- 您在 Data Mapper 步骤中映射字段。
- 熟悉 Microsoft Excel 表达式，或者您有您想要应用到映射的条件表达式。

流程

1. 如果数据类型还没有可见，点

来显示它们。

虽然这不是指定条件的要求，但查看数据类型会很有帮助。

2. 创建您要对其应用条件的映射，或者确保当前选择的映射是您要将条件应用到的映射。例如，考虑这个映射：

3.

在左上角，点

以显示条件表达式输入字段。

在 expression 字段中，数据映射器会自动显示当前映射中的 source 字段的名称。例如：

在表达式输入字段中，源字段的顺序是您创建映射时选择的顺序。这很重要，因为默认映射行为是数据映射程序串联字段值，以便在 target 字段中插入结果。在本例中，要创建此映射，首先选择 lastName，然后选择 firstName。

4. 编辑 expression 输入字段，以指定您希望数据映射程序应用到映射的条件表达式。有关支持的条件表达式的详情，请遵循此流程。

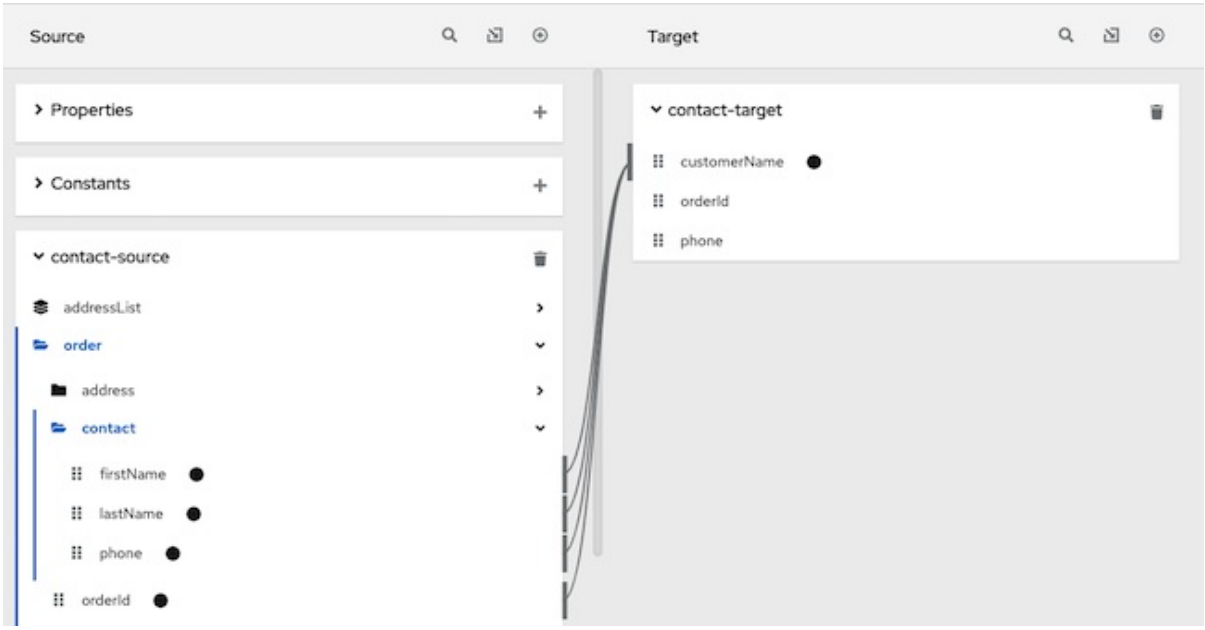
如果要在条件映射中包含转换，您必须将转换添加到条件表达式中。

在指定表达式时，您可以键入 @ 和 start 来键入字段的名称。数据映射器显示与您输入的字段列表。选择您要在表达式中指定的字段。

当您向表达式中添加字段名称时，data mapper 会将该字段添加到映射中。例如，考虑这个条件表达式：

在执行过程中，如果数据映射器决定 lastName 字段为空，它将 firstName 字段映射到目标 customerName 字段。如果 lastName 字段包含一个值，即不是空的，则 data mapper 会串联 source orderId 和 phone 字段中的值，并在 customerName 字段中插入结果。（本示例展示了逻辑的工作方式，但可能不是有用的示例，因为 lastName 字段中有一个值时，您很可能希望数据映射程序只执行映射，而不是将一些其他值映射到目标。）

在本例中，输入表达式后，数据映射为：



在条件表达式中，如果您删除表达式应用到的映射中的字段名称，则数据映射程序会从映射中删除该字段。换句话说，映射中的每个字段名称都必须在条件表达式中。

5. *如果映射预览字段尚未可见，点*  *来显示它们。*

6. *在源预览输入字段中键入示例数据，以确保 target 字段或目标字段获得正确的值。*

7. *根据需要编辑条件表达式，以获取所需的结果。*

条件表达式中支持的函数

- *ISEMPTY(source-field-name1 [+ source-field-name2])*

  ISEMPTY （） 函数的结果是一个布尔值。至少指定一个参数，这是您要将条件应用到的映射中的 source 字段的名称。当指定的 source 字段为空时，ISEMPTY （） 函数返回 true。

  另外，还可使用附加字段添加 + （分散）Operator，例如：

ISEMPTY(lastName + firstName)

如果 source 字段 lastName 和 firstName 都为空，则此表达式评估为 true。

通常，ISEMPTY （） 函数是 IF （） 函数中的第一个参数。

- *IF(boolean-expression, then, else)*

  当 boolean-expression 评估为 true 时，数据映射器会返回。 当 boolean-expression 评估为 false 时，数据映射器会返回 其他。所有这三个参数都是必需的。最后的参数可以是 null，这意味着当 boolean-expression 评估为 false 时，不会映射任何参数。

  例如，假设目标 customerName 字段中组合了 lastName 和 firstName 源字段的映射。您可以指定此条件表达式：

  IF (ISEMPTY (lastName), firstName, lastName + ',' + firstName)

  在执行过程中，数据映射器会评估 lastName 字段。

  - *如果 lastName 字段为空，即 ISEMPTY (lastName) 返回 true，则数据映射器仅将 firstName 值插入到目标 customerName 字段中。*
  - *如果 lastName 字段包含一个值，即 ISEMPTY (lastName) 返回 false，则 data mapper 会映射 lastName 值，后跟一个逗号，后接 firstName 值到目标 customerName 字段中。*

    现在，如果表达式中的第三个参数为 null，则请考虑行为：

    IF (ISEMPTY (lastName), firstName, null)

    在执行过程中，数据映射器会评估 lastName 字段。

  -

如上例所示，如果 lastName 字段为空，即 ISEMPTY (lastName) 返回 true，则数据映射器仅将 firstName 值插入到目标 customerName 字段中。

- *但是，当第三个参数为 null 时，如果 lastName 字段包含值，即 ISEMPTY (lastName) 返回 false，则数据映射器不会将任何内容映射到目标 customerName 字段中。*

- ***l (x,y) 或 & lt; (x,y)***

  数据映射程序评估 x 和 y，并返回较低值。x 和 y 都必须是数字。

- ***TOLOWER (string)***

  数据映射器将指定的字符串转换为小写并返回它。

表 7.1. 条件表达式中支持的运算符

Operator	Description
+	添加数字值或串联字符串值。
-	从另一个数字值中减去一个数字值。
*	多数字值.
\	划分数字值。
&& 和	如果左侧和右侧运算对象为 true，则返回 true。每个操作对象都必须返回一个布尔值。
|| 或者	如果左侧操作对象为 true，或者右侧操作对象为 true，或者两个操作对象都为 true，则返回 true。每个操作对象都必须返回一个布尔值。
!	not
&gt; 大于	如果左侧数字运算对象大于右侧数字运算对象，则返回 true。
< 小于	如果左侧数字运算对象小于右侧数字运算对象，则返回 true。
== Equal	如果左侧运算对象和右侧运算对象相同，则返回 true。

7.14. 可用转换的描述

下表描述了可用的转换。日期和时间类型通常是指这些概念的任何形式。也就是说，数字包括 整数，长，双。日期包括日期，如 日期、时间、区域 日期。

转换	输入类型	输出类型	参数(* = required)	描述
AbsoluteValue	number	number	None	返回数字的绝对值。
AddDays	date	date	**days**	将天数添加到日期. 默认值为 0 天。
AddSeconds	date	date	**SECONDS**	添加日期的秒数。默认值为 0 秒。
附加	字符串	字符串	字符串	将字符串附加到字符串的末尾。默认为不附加任何内容。
Camelize	字符串	字符串	None	通过删除空格将一个短语转换为 camelized 字符串，使第一个单词小写，并大写每个后续单词的第一个字母。
大写	字符串	字符串	None	在字符串中大写第一个字符。
ceiling	number	number	None	返回数字的整数。
contains	any	布尔值	**value**	如果字段包含指定的值，则返回 true。

转换	输入类型	输出类型	参数(* = required)	描述
ConvertAreaUnit	number	number	**fromUnit*** **toUnit** *	将代表区域的数字转换为另一个单元。对于 **fromUnit** 和 **toUnit** 参数，请从 From Unit 和 To unit 菜单中选择适当的单元。选择是：S **qu** are Foot、**Square Meter** 或 **Square Mile**。
ConvertDistanceUnit	number	number	**fromUnit** * **toUnit** *	将代表距离另一个单元的数字转换。对于 **fromUnit** 和 **toUnit** 参数，请从 From Unit 和 To unit 菜单中选择适当的单元。选择包括：Foot、**Inch**、Mer**ter**、**Mi** Mi 或 **Yard**。
ConvertMassUnit	number	number	**fromUnit** * **toUnit** *	将代表 mass 的数字转换为另一个单元。对于 **fromUnit** 和 **toUnit** 参数，请从 From Unit 和 To unit 菜单中选择适当的单元。选择是：**Kilogram** 或 **Pound**。
ConvertVolumeUnit	number	number	**fromUnit** * **toUnit** *	将代表卷的数字转换为另一个单元。对于 **fromUnit** 和 **toUnit** 参数，请从 From Unit 和 To unit 菜单中选择适当的单元。选择包括：**Cubic Foot**、**Cubic Meter**、**Gallon US Fluid** 或 **Liter**。

转换	输入类型	输出类型	参数(* = required)	描述
DayOfWeek	date	number	None	返回与日期对应的每周(1 到 7)的日期。
DayOfYear	date	number	None	返回与日期对应的年日(1 到 366)。
endwith	字符串	布尔值	**字符串**	如果字符串以指定字符串结尾，且两个字符串中的情况都相同，则返回 true。
等于	any	布尔值	**value**	如果输入字段等于 **指定的值**，且在字段和值中，则返回 true。
FileExtension	字符串	字符串	None	在代表文件名的字符串中，返回文件扩展名，而无需点。
floor	number	number	None	返回数字的整数。
格式	any	字符串	**模板** (Template)	在 **模板** 中，将每个占位符（如 **%s**）替换为输入字段的值并返回包含结果的字符串。这与 Java 和 C 等编程语言提供的机制类似。
IndexOf	string 中的第一个字符是 index 0。	number	**字符串** 搜索此字符串的输入字符串。	在输入字符串中返回字符的索引，该字符串是参数字符串的第一个字符。如果没有找到参数字符串，则返回 **-1**。
IsNull	any	布尔值	None	如果字段为 null，则返回 true。

转换	输入类型	输出类型	参数(* = required)	描述
LastIndexOf	string 中的第一个字符是 index 0。	number	**字符串** 搜索此字符串的输入字符串。	在输入字符串中返回字符的索引，该字符串是参数字符串的最后一个字符。如果没有找到参数字符串，则返回 **-1**。
length	any	number	None	返回字段的长度，如果字段为 null，则为 **-1**。
小写	字符串	字符串	None	将字符串转换为小写。
规范化	字符串	字符串	None	使用单个空格替换连续的空格字符，并修剪字符串中的前导和尾随空格。
PadStringLeft	字符串	字符串	**padCharacter** * **padCount** *	在字符串的开头插入 **padCharacter** 中提供的字符。执行此操作在 **padCount** 中指定的次数。
PadStringRight	字符串	字符串	**padCharacter** * **padCount** *	在字符串末尾插入 **padCharacter** 中提供的字符。执行此操作在 **padCount** 中指定的次数。
prepend	字符串	字符串	**字符串**	前缀 **字符串** 到字符串的开头。默认值是添加任何前缀。
ReplaceAll	字符串	字符串	**match** * **newString**	在字符串中，将提供的匹配字符串的所有位置替换为提供的 **newString**。默认的 **newString** 是一个空字符串。

转换	输入类型	输出类型	参数(* = required)	描述
ReplaceFirst	字符串	字符串	**match** * **newString** *	在字符串中，将指定 **匹配** 字符串的第一个出现替换为指定的 **newString**。默认的 **newString** 是一个空字符串。
round	number	number	None	返回数字的舍入整数。
SeparateByDash	字符串	字符串	None	将出现的每个空格、冒号(:)、下划线(_)、加号(=)和等号(=)替换为连字符(-)。
SeparateByUnderscore	字符串	字符串	None	将出现的每个空格、冒号(:)、连字符(-)、加号(=)和等号(=)替换为下划线(_)。
startswith	字符串	布尔值	**字符串**	如果字符串以指定字符串（包括 case）开头，则返回 true。
子字符串	字符串	字符串	**startIndex** * **endIndex**	从指定的 inclusive **startIndex** 中检索字符串片段到指定的 exclusive **endIndex**。两个索引都从零开始。**startIndex** is inclusive.**endIndex** 为 exclusive。**endIndex** 的默认值是字符串的长度。

转换	输入类型	输出类型	参数(* = required)	描述
SubstringAfter	字符串	字符串	**startIndex** * **endIndex** **match** *	在指定的 **匹配** 字符串后检索字符串的片段，从指定的 inclusive **startIndex** 到指定的 exclusive **endIndex**。两个索引都从零开始。**endIndex** 的默认值是提供的 **匹配** 字符串后面的字符串长度。
SubstringBefore	字符串	字符串	**startIndex** * **endIndex** **match** *	在提供的 inclusive **startIndex** 中的 startIndex 之前，在提供的 **匹配** 字符串前检索字符串片段到提供的 exclusive **endIndex**。两个索引都从零开始。**endIndex** 的默认值是提供 **匹配** 字符串前字符串的长度。
Trim	字符串	字符串	None	从字符串中修剪前导和尾随空格。
TrimLeft	字符串	字符串	None	从字符串中修剪前导空格。
TrimRight	字符串	字符串	None	从字符串中修剪尾部空格。
大写	字符串	字符串	None	将字符串转换为大写。

7.15. 数据映射故障排除

数据映射器显示由上一集成步骤提供的最大可能的源字段集合。但是，并非所有连接都会在显示的源字段中提供数据。例如，对第三方应用程序的更改可能会停止在特定字段中提供数据。当您创建集成时，如果您注意到数据映射没有如预期那样的行为，请确保要映射的 source 字段包含您预期的数据。

数据形成的更改会影响已映射的字段可能会阻止数据映射程序加载文档。在这种情况下，当您尝试编辑映射了受影响字段的数据映射程序步骤时，数据映射器无法显示源和目标面板。相反，它会显示一个错

误，表示它无法加载或无法找到文档。错误消息类似如下信息之一：

- Data Mapper UI 初始化错误： Could not load document '-La_rwMD_ggphAW6nE9o': undefined undefined
- 无法找到位于 URI atlas:json:-LaX4LMC1CfVJYp3JXM6 的映射字段 'last_name' 的文档

您必须删除此数据映射程序步骤，并将其替换为您映射更新字段的新数据映射程序步骤。

虽然数据形成的变化需要您删除映射字段，但您不需要删除并删除数据映射程序步骤。例如，如果 XML 实例指定了输入数据，并且更改了元素的名称，则数据映射器会从旧字段名称中删除该映射。您只需要使用更新的名称映射至/来自字段。

可以使用以下方法更改映射字段的数据：

- 在 API 供应商集成中，在编辑流时，您可以编辑定义操作的 OpenAPI 文档。

  更改操作响应的数据始终会阻止数据映射程序加载文档。

- 在流中，您可以为其中一个连接编辑输入数据类型和/或输出数据类型：
  - Amazon S3
  - AMQ
  - AMQP
  - Dropbox
  - FTP/SFTP

- ***HTTP/HTTPS***
- ***Kafka***
- ***IRC***
- ***MQTT***

第 8 章 管理集成

常见的设置是让 Fuse 在线开发环境、Fuse 在线测试环境和 Fuse 在线部署环境。为方便此目的，Fuse Online 提供从一个 Fuse Online 环境导出集成的功能，然后将该集成导入到另一个 Fuse 在线环境中。在各种 Fuse 在线环境中管理集成的信息和步骤都相同，除非特别说明。

以下主题提供了帮助您管理集成的信息：

- 第 8.1 节 “关于集成生命周期处理”
- 第 8.2 节 “将集成置于服务中和停用”
- 第 8.3 节 “有关集成执行的日志信息”
- 第 8.4 节 “监控集成”
- 第 8.5 节 “测试集成”
- 第 8.6 节 “集成执行故障排除的提示”
- 第 8.7 节 “更新集成”
- 第 8.8 节 “为集成调整内存和 CPU 配置属性”
- 第 8.9 节 “删除集成”
- 第 8.10 节 “将集成复制到另一个环境中”

8.1. 关于集成生命周期处理

创建并发布集成后，您可能需要更新集成的作用。您可以编辑已发布集成的草案，然后将正在运行的版

本替换为更新版本。为了便于实现这一目的，Fuse Online 维护多个版本以及每个版本的状态。了解集成版本和状态可帮助您管理集成。

集成版本的描述

在每个 Fuse Online 环境中，每个集成都有多个版本。对多个集成版本的支持有几个优点：

- 如果您发布了一个无法正常工作的版本，您可以返回运行正确版本的集成。要做到这一点，您可以停止不正确的版本，并启动一个运行它的版本。
- 随着要求或工具的变化，您可以逐步更新集成。您不需要创建新的集成。

Fuse Online 每次开始运行集成的新版本时分配一个新的版本号。例如，假设您将 Twitter 发布到 Salesforce 示例集成。在它运行后，您将更新集成，以使用其他帐户连接到 Twitter。然后，发布更新的集成。Fuse Online 会停止集成的运行版本，并发布与递增版本号集成的更新版本。

运行的初始集成是版本 1。现在运行的已更新集成是版本 2。如果您编辑了版本 2，例如使用不同的帐户连接到 Salesforce，您发布该版本，然后变为集成的版本 3。

集成正好有一个草案版本。Fuse Online 对集成的草案版本有一个定义，但从未运行此版本的集成。集成的草案版本没有数字。编辑集成时，您要更新集成的草案版本。

在 Fuse Online 中，您可以在集成概述页面中看到集成版本列表。要查看此页面，请在左侧导航面板中点 Integrations。在您感兴趣的集成条目中，单击 View。

集成状态的描述

在 Fuse Online 中，在集成版本列表中，每个条目都指示该版本的状态，这是以下之一：

状态	Description
运行中	Running 版本正在执行；它位于服务中。只能运行一个集成版本。也就是说，一次只能有一个版本处于 Running 状态。

stopped	Stopped 版本没有运行。集成的草案版本处于 Stopped 状态。每个一次运行集成，然后停止处于 Stopped 状态。 如果没有此集成版本处于 Running 状态，您可以启动已停止的版本。
待处理	待处理的 版本处于转换状态。Fuse Online 正在启动此版本的集成或停止此版本的集成，但集成还没有运行或停止。
Error	处于 Error 状态的集成版本在启动或运行时遇到 OpenShift 错误。错误暂停启动或执行。如果发生这种情况，请尝试启动正确运行的早期集成版本。或者，请联系技术支持以获取帮助。为此，请在右上角的任何 Fuse Online 页面中，点  图标并选择 Support。

8.2. 将集成置于服务中和停用

创建集成后，您可以将其保存为草案，或发布它以开始运行。发布集成时，Fuse Online 汇编所需的资源，构建集成运行时，部署将运行集成的 OpenShift pod，然后开始运行集成。

您可以随时点击按钮停止运行集成。当您想再次启动集成时，Fuse Online 已具有所需的内容，因此启动它的时间比首次运行它的时间要少得多。

首次启动集成版本的过程被称为发布集成。以下主题提供详情：

- 第 8.2.1 节 “关于发布集成”
- 第 8.2.2 节 “停止集成”
- 第 8.2.3 节 “启动集成”
- 第 8.2.4 节 “重启旧的集成版本”

8.2.1. 关于发布集成

要第一次运行集成的版本，请发布它。发布集成构建和部署集成运行时。集成开始运行。发布集成后，您可以停止并重新启动它。一次只能运行一个集成版本。

发布的替代方法

要第一次运行集成，请执行以下操作之一发布它：

- 在创建或编辑集成的步骤结束时，单击 **Publish**。
- 发布集成的草案版本：
  1. 在左侧 **Fuse Online** 面板中，单击 **Integrations**。
  2. 在集成列表中，在草案条目右侧点
     
     并选择 **Publish**。

关于发布进度

Fuse Online 显示发布过程的进度，它有多个阶段：

1. 装配 创建构建集成所需的 **pod** 资源。
2. 构建 可以准备部署集成。
3. 部署 会等待将运行集成的 **pod** 的部署。
4. 启动 等待 **pod** 开始运行集成。
5. **deployed** 表示集成正在运行。

在启动过程中，您可以点 View Logs 来显示提供启动信息的 OpenShift 日志。

发布后集成状态

发布集成完成后，集成名称旁边会显示 Running 状态。pod 运行集成。

8.2.2. 停止集成

每个集成都可能只有一个版本正在运行。正在运行的版本处于 Running 状态。您可以随时停止运行集成。

前提条件

要停止的集成处于 Running 状态。

流程

1. 在左侧 Fuse Online 面板中，单击 Integrations。
2. 在集成列表中，识别您要停止运行的集成条目。该条目显示此集成 正在运行。
3. 在这个集成条目的最右侧，点

并选择 Stop。

结果

Fuse Online 停止运行集成。在集成列表中，停止和 停止 会出现在集成条目中。

8.2.3. 启动集成

首次启动集成时，该过程称为发布集成，因为 Fuse Online 必须在运行集成之前构建集成运行时。您可以随时停止运行集成，然后再次启动它。

前提条件

要启动的集成处于 Stopped 状态。

流程

1. 在左侧导航面板中，单击 Integrations。
2. 在集成列表中，在您要启动的集成条目右侧，点  。
3. 选择 Start。

结果

Fuse Online 显示 Starting 作为该集成版本的状态，然后在集成再次运行时显示为 Running。

8.2.4. 重启旧的集成版本

您可以发布一个集成，该集成无法按照您想要的方式工作。在这种情况下，您可以停止不正确的版本，并将其替换为之前发布的版本以及正确运行的版本。

先决条件

- 集成的版本正在运行，但您要停止它。
- 您有另一个版本的集成，您希望运行该集成。

流程

1. 在左侧面板中，单击 Integrations 以显示此环境中的集成列表。
2. 单击您要为其发布旧版本的集成条目。Fuse Online 显示集成版本列表。
3. 在运行版本的条目中，在最右侧点

并选择 **Stop**。

4. 单击 **OK**，以确认您要停止运行此版本。

5. 等待 **Stopped** 显示在页面顶部的集成名称右侧。

6. 要像一样发布旧版本，请跳至下一步。或者，在发布旧版本前，您可以更新它：

   a. 在您要更新的集成版本的条目中，在右侧点

   并选择 **replace Draft**。

   b. 根据需要更新集成。

   c. 更新完成后，在右上角，单击 **Publish**，然后单击 **Publish** 确认。这将执行接下来的两个步骤的位置。

7. 要发布旧版本，请在您要再次开始运行的集成版本的条目中，点

   并选择 **Start**。

8. 点 **Start** 确认您要启动这个版本的集成。

结果

Fuse Online 启动集成，这需要几分钟时间。当集成运行时，**Running version n** 会显示在集成名称右侧。

8.3. 有关集成执行的日志信息

对于集成的每个执行，对于流中的每个步骤，**Fuse Online** 提供以下活动信息：

- 执行步骤的日期和时间
- 执行步骤所需的时间
- 执行是否成功
- 如果执行不成功，则错误消息

要在 Fuse Online 中查看此信息，显示集成的摘要，然后单击 Activity 选项卡。

要获取有关集成执行的更多详细信息，您可以通过向集成流中添加日志步骤来记录有关集成进程的消息的信息。对于集成接收的每个消息，日志步骤可以提供以下一个或多个消息：

- 消息的上下文，提供有关消息的元数据，包括消息的标头。
- 消息的正文，它提供消息的内容。
- 您明确指定的文本，或者通过评估 Apache Camel Simple 语言 表达式。

注意

以前版本中可用的日志连接不再可用于集成。添加日志步骤而不是日志连接。

先决条件

- 您正在创建或编辑流，Fuse Online 会提示您添加到集成。或者，Fuse Online 正在提示您选择完成连接。

流程

1. 在流视觉化中，点您要添加日志步骤的加号。如果 Fuse Online 正在提示您选择完成连接，请跳过这一步。

2. 单击 **Log**。

3. **在 Configure Log Step 页面中，选择您要记录的内容。如果您选择 Custom Text，然后在文本输入字段中输入以下之一：**

   - **要记录的文本**
   - **Camel Simple 语言表达式**

   如果您输入表达式，Fuse Online 将解析表达式并记录生成的文本。

4. **日志步骤配置完成后，点 Next 将步骤添加到流中。**

后续步骤

完成流后，发布集成以开始查看新日志步骤的输出。

其他资源

执行日志步骤的流后，日志步骤的输出会出现在集成的 Activity 选项卡中。请参阅查看集成活动信息。

8.4. 监控集成

Fuse Online 为您提供了各种监控集成执行的方法。请参阅：

- **第 8.4.1 节 “查看集成历史记录”**
- **第 8.4.2 节 “查看有关集成活动的信息”**
- **第 8.4.3 节 “查看特定集成的指标”**

- 第 8.4.4 节 “查看 Fuse 在线环境的指标”

8.4.1. 查看集成历史记录

Fuse Online 维护集成的每个版本。您始终可以查看每个集成版本的列表。

流程

1. 在左侧面板中，单击 Integrations 以显示您环境中的集成列表。
2. 在您要查看的版本集成的条目右侧，单击 View。

结果

在出现的页面中，History 部分列出了集成的版本。

图标标识当前版本，它是最近运行的版本。对于每个版本，您还可以查看它最后一次启动的日期。

要编辑、启动或停止特定版本，请点击版本条目右侧的

。选择您要执行的操作。

8.4.2. 查看有关集成活动的信息

Fuse Online 为集成的每个执行提供活动信息。此信息是集成日志的一部分。对于流中的每个步骤，Fuse Online 提供：

- 执行步骤的日期和时间
- 执行步骤所需的时间
- 执行是否成功

- 如果执行不成功，则错误消息

您可以随时查看此信息。

先决条件

- 有或有一个要查看活动信息的运行集成。
- 此集成至少执行一次。

流程

1. 在左侧面板中，单击 **Integrations**。
2. 在您要查看活动信息的集成条目右侧，单击 **View**。
3. 在集成摘要页面中，单击 **Activity** 选项卡。
4. （可选）输入 **date** 和/or 关键字过滤器来限制列出的执行。
5. 点击您要查看活动信息的集成执行。

注意

对于 **API** 提供程序集成或 **Webhook** 步骤，集成 活动 选项卡中的信息反映了 **Fuse** 在线集成与调用它的客户端之间的通信。**HTTP** 或 **REST** 请求生成的错误在集成的 **Activity** 日志中不可见。如果要查看或测试 **HTTP** 或 **REST** 请求生成的错误，当您配置 **API Provider** 或 **webhook** 步骤时，请检查 返回正文选项中的 **Include** 错误消息 （默认为检查）。然后，您可以通过检查对调用者的 **HTTP** 或 **REST** 标头来验证响应中是否包含错误消息。您还可以检查集成 **pod** 的日志文件中的 **INFO** 信息。

其他资源

- 要在任一两个步骤间获取其他信息，您可以在集成中添加日志步骤。日志步骤提供有关它收到的每个消息的信息，并提供您指定的自定义文本。如果您添加了日志步骤，则在扩展您要查看

活动信息的集成执行时，它显示为集成步骤之一。您以与查看 Fuse Online 信息的其他步骤相同的方式查看 Fuse 在线日志步骤的信息。请参阅有关集成执行的日志信息。

8.4.3. 查看特定集成的指标

Fuse Online 为每个集成提供以下指标：

- 错误总数 表示此集成在过去 30 天内遇到的所有执行运行时错误的数量。
- 最后进程 显示此集成处理消息的最新日期和时间。消息可能已被成功处理，或者可能出现错误。
- Total Messages 是此集成的所有执行在最后 30 天内处理的消息数量。这包括消息失败。
- uptime 表示此集成启动时以及它运行的时长，且没有错误。

前提条件

您要查看指标的集成正在运行或正在运行。

流程

1. 在左侧面板中，单击 Integrations。
2. 在您要查看指标的集成条目右侧，单击 View。
3. 在集成摘要页面中，单击 Metrics。

8.4.4. 查看 Fuse 在线环境的指标

Fuse Online 环境的指标会出现在 Fuse Online 主页中。

流程

在左侧面板中，单击 Home。

结果

Fuse Online 每 5 秒更新以下指标：

- 在此环境中定义的集成数量，以及正在运行的集成数量、停止的集成数量以及待处理的集成数量。Fuse Online 是停止或启动待处理的集成。红色跨度表示运行的任何集成，但遇到暂停执行的错误。
- 此环境中定义的连接数。
- 此环境中集成处理的消息总数（最后 30 天）。这包括集成处理的消息可能不再运行或者已从此环境中删除。
- 运行时间表示至少有一个集成正在运行的时间。正常运行时间启动时的日期和时间也显示。

8.5. 测试集成

创建集成后，并在 Fuse Online 开发环境中正确运行后，您可能想要在不同 Fuse Online 环境中运行来测试它。

前提条件

- 您有一个 Fuse Online 开发环境和 Fuse 在线测试环境。
- 您有一个在 Fuse 在线开发环境中正确运行的集成。

流程

1. 了解如何将集成复制到另一个环境。
2. 从开发环境导出集成。请参阅 导出集成。

3. 将集成导入到测试环境中。请参阅 导入集成。

8.6. 集成执行故障排除的提示

如果集成停止工作，请检查其活动和历史记录详情。请参阅 查看集成活动信息 和查看 集成历史记录。

集成的内存不足

如果 Fuse Online Operator 日志显示集成的状态 未就绪，或者集成 pod 事件报告不足的内存，您可能需要编辑集成的部署配置，如 调整集成的内存和 CPU 配置属性 中所述。

连接到使用 OAuth 的应用

对于到使用 OAuth 的应用的连接，您可能会看到一条错误消息，表示应用的访问令牌已过期。有时，您可能会得到更明确的 403 - Access denied 消息。消息中的信息取决于集成所连接的应用程序。在这种情况下，尝试重新连接到应用程序，然后重新发布集成：

1. 在左侧面板中，单击 Integrations。
2. 在集成列表中，单击停止正在运行的集成条目，单击 View。
3. 在集成摘要页面中，在流视觉化中，点击您要重新连接的应用程序的图标。

   如果这是 API 提供程序集成，则在集成的摘要页面中，单击其流图标以显示操作列表。然后，点击路径包含失败的连接的操作，然后在操作的路径视觉化中点失败的连接。
4. 在连接的详情页面中，点 Reconnect。
5. 响应该应用程序的 OAuth 工作流提示。

   Fuse Online 显示一条消息，以指示其对应用程序的访问权限已被授权。对于某些应用程序，这需要几秒钟，但其他应用程序可能需要更长的时间。
6. 重新连接到应用程序后，启动集成。

如果重新连接不成功，请尝试以下操作：

1. 重新注册 Fuse Online 作为应用程序的客户端。有关获取授权的信息，请参阅常规流程。
2. 创建新连接。
3. 编辑使用旧连接的每个集成：
   a. 删除旧的连接。
   b. 使用新连接替换它。
4. 发布每个更新的集成。

8.7. 更新集成

创建集成后，您可能需要更新它以添加、编辑或删除步骤。

前提条件

在 Fuse Online 环境中，有一个您要更新的集成版本。

流程

1. 在左侧 Fuse Online 面板中，单击 Integrations。
2. 在集成列表中，点您要更新的集成的 View。
3. 在集成摘要页面中，单击右上角的 Edit Integration。

   如果这是简单的集成，Fuse Online 会显示集成流。如果这是 API 供应商集成，Fuse Online 会显示操作列表。要更新特定操作的流，请点击该操作右侧的 Edit flow 来显示其流。

4. 根据需要更新流：

- 要添加步骤，在流视觉化中点您要添加步骤的加号。然后，点击代表您要添加步骤的卡。
- 要删除步骤，在流视觉化中点您要删除的步骤  。
- 要更改步骤的配置，请在流视觉化中点击您要更新的步骤的 Configure。在配置页面中，根据需要更新参数设置。

8.8. 为集成调整内存和 CPU 配置属性

您可以通过编辑集成的部署配置对象，为特定集成的 CPU 和内存指定自定义值。您可能想要为集成调整内存和 CPU 配置属性，例如，如果集成需要的内存超过其默认分配量。

前提条件

- 已安装 Red Hat OpenShift oc 客户端工具，并连接到安装了 Fuse Online 的 OCP 集群。
- 具有集群管理权限的用户为包含您要配置的集成的项目具有 admin 权限。

流程

1. 使用具有包含 Fuse Online 集成的 OpenShift 项目的 admin 权限的帐户登录 OpenShift。例如：

```
oc login -u admin -p admin
```

2. 切换到包含 Fuse Online 集成的项目。例如：

```
oc project my-fuse-online-project
```

3. 编辑集成的部署配置对象：

   a. 输入以下命令，这通常会在编辑器中打开资源：

      oc edit deploymentconfig <i-integration-name>

      例如，如果集成的名称是 my-integration，请输入这个命令：

      oc edit deploymentconfig i-my-integration

   b. 通过设置 spec.containers.resources 来编辑配置，以指定 CPU 和内存的值，如下例所示：

```
spec:
  containers:
    resources:
      limits:
        cpu: 350m
    requests:
        memory: 350Mi
```

   c. 保存配置。

结果

保存更改后，集成的 pod 会重启，新 pod 使用新值运行。例如，如果您运行 oc describe <intergration-pod-name& gt; 命令（将 <intergration-pod-name> 替换为集成 pod 的名称，如 i-my-integration），命令会返回新值，例如：

```
resources:
  limits:
    cpu: 350m
  requests:
    cpu: 350m
    memory: 350Mi
```

即使发布新版本的集成，这些值也会保留。

其他资源

要为所有集成的 CPU 和内存属性设置默认值，OpenShift 集群管理员可以更新 Fuse Online 自定义资源，如在 {NameOfFuseOnlineOnOCP} 中配置 Fuse Online 的自定义资源属性描述。

8.9. 删除集成

删除集成后，Fuse Online 仍然具有该集成的历史记录。如果您导入了已删除集成的版本，Fuse Online 会将已删除集成的历史记录与导入的集成相关联。

流程

1. 在左侧 Fuse Online 面板中，单击 **Integrations**。
2. 在集成列表中，在您要删除的集成的条目右侧，点  并选择 **Delete**。
3. 在弹出窗口中，单击 **OK** 以确认要删除该集成。

8.10. 将集成复制到另一个环境中

要在开发、暂存和生产环境中运行集成，您可以导出和导入集成。环境可以都位于单个 OpenShift 集群中，也可以分散到多个 OpenShift 集群中。

此处描述的步骤指示您在 Fuse Online 控制台中导出和导入集成。

如果您在 OpenShift Container Platform 现场运行 Fuse Online，您可能具有需要导出和导入某些集成所需的持续集成/持续部署(CI/CD)管道。有关这样做的详情，请参考 使用外部工具为 CI/CD 导出/导入集成。

请参见以下主题：

- 第 8.10.1 节 “关于复制集成”

- 第 8.10.2 节 “导出集成”
- 第 8.10.3 节 “导入集成”

8.10.1. 关于复制集成

每个 Fuse 在线安装都是您可以导出集成的环境。导出集成会下载 zip 文件，其中包含在不同 Fuse Online 环境中重新创建集成所需的信息。

在一个环境中，每个集成只能有一个 Draft 版本。

导入集成的结果取决于：

- 之前导入集成
- 之前导入了集成使用的连接

Fuse Online 为每个集成使用一个内部标识符，以及每个连接来确定它是否已存在于要导入到的环境中。如果您更改了集成或连接的名称，Fuse Online 会将它识别为与具有不同名称相同的集成或连接。

下表描述了导入集成的可能结果：

在导入环境中：	导入操作的作用：
之前没有导入集成。	创建集成。集成处于 Draft 状态。
集成之前已导入。	Fuse Online 更新集成。更新的集成处于 Draft 状态。如果此集成有 Draft 版本，它会丢失。
导入的集成使用环境中不存在的连接，然后再导入操作。	Fuse Online 创建一个连接，它具有相同的设置，但 secret 除外。您必须查看每个新连接。如果没有为其新环境完全配置连接，则必须添加缺少的设置。例如，您可能需要将此 Fuse Online 环境注册为此连接访问的应用程序的客户端来获取 secret 设置。

8.10.2. 导出集成

当 Fuse Online 导出集成时，它会将 zip 文件下载到您的本地 下载 文件夹。此 zip 文件包含在不同 Fuse 在线环境中重新创建集成所需的信息。

导出集成也是一种进行集成备份的方法。但是，Fuse Online 维护集成的版本，因此不需要导出集成。

流程

1. 在 Fuse Online 左侧面板中，单击 **Integrations**。
2. 在集成列表中，识别您要导出的集成条目。
3. 在条目右侧，点

并选择 **Export**。

后续步骤

要将集成导入到另一个 Fuse 在线环境中，请打开该环境并导入导出的 zip 文件。

8.10.3. 导入集成

在 Fuse Online 环境中，您可以导入从另一个 Fuse Online 环境导出的集成。导出集成会下载您上传以导入集成的 zip 文件。

先决条件

- 您有一个 zip 文件，其中包含从另一个 Fuse 在线环境中导出的集成。

流程

1. 打开您要将集成导入到的 Fuse Online 环境。

2. 在左侧面板中，单击 **Integrations**。

3. 在右上角，单击 **Import**。

4. 拖放一个或多个导出的集成 **zip** 文件，或者导航到包含导出的集成的 **zip** 文件并选择它。

   Fuse Online 导入文件并在导入成功后显示一条消息。

5. 在左侧面板中，单击 **Integrations**。

6. 在集成列表中，单击您刚才导入的集成条目，单击 **View**。

7. 在集成摘要中，如果需要该配置的通知，请单击 **Edit integration**。

8. 对于每个需要配置的连接：

   a. 单击 **Configure** 按钮，以显示其详细信息。

   b. 根据需要输入或更改连接详情。本页中的每个字段都正确，且只需要安全配置。

   c. 点击 **Next**。

9. 在左侧面板中，单击 **Settings**。

   Settings 页面显示使用 **OAuth** 协议的应用程序的条目。

10. 对于每个需要配置和访问使用 **OAuth** 协议的应用程序的连接，请将 **Fuse Online** 环境注册到应用程序。每个应用程序的步骤会有所不同。请查看适当的主题：

- 使用 Dropbox 注册
- 使用 Google 注册
- 使用 JIRA 注册
- 使用 REST API 注册
- 使用 Salesforce 注册
- 连接到 SAP Concur
- 使用 Twitter 注册

11. 在左侧面板中，点 Connections 并确认不再有需要配置的连接。

12. 在左侧面板中，单击 Integrations。在集成列表中，导入的集成在其条目的左上角有一个绿色三角。

13. 在集成列表中，在您导入的集成右侧，点

并选择 Edit。

14. 在右上角，单击 Save 或，如果要开始运行导入的集成，请单击 Publish。无论您将集成保存为草案，还是发布集成，Fuse Online 都会更新集成以使用更新的连接。

第 9 章 自定义 FUSE 在线

Fuse Online 提供很多连接器，可用于连接到常见的应用程序和服务。还有一些内置步骤可用于以常见方式处理数据。但是，如果 Fuse Online 不提供您需要的功能，请与开发人员讨论您的要求。经验丰富的开发人员可通过提供以下任一方式帮助您自定义集成：

- Fuse Online 可用于为 REST API 客户端创建连接器的 OpenAPI 文档。

  您可以将此模式上传到 Fuse Online，Fuse Online 根据架构创建一个连接器。然后，您可以使用连接器来创建可以添加到集成中的连接。例如，许多零售网站提供一个 REST API 客户端接口，开发人员可以在 OpenAPI 文档中捕获。

- 定义 REST API 服务的 OpenAPI 文档。

  您可以将此模式上传到 Fuse Online。Fuse Online 使 API 服务可用，并提供 API 调用的 URL。这可让您通过 API 调用 触发集成执行。

- Fuse Online 可用于为 SOAP 客户端创建连接器的 WSDL 文件。

  注意：这是一个技术预览功能。

- 实施 Fuse 在线扩展的 JAR 文件。扩展可以是以下任意一种：

  - 在连接间的集成数据上运行的一个或多个步骤
  - 应用程序或服务的连接器
  - 专有 SQL 数据库的 JDBC 驱动程序的库资源

  您可以将此 JAR 文件上传到 Fuse Online，Fuse Online 使扩展提供的自定义功能。

详情请查看以下主题：

9.1. 开发 REST API 客户端连接器

Fuse Online 可以为代表性状态传输应用程序编程接口(REST API)创建连接器，支持 Hypertext 传输协议(HTTP)。为此，Fuse Online 需要有效的 OpenAPI 3 （或 2）文档，用于描述您要连接到的 REST API。如果 API 服务提供商没有提供 OpenAPI 文档，则经验丰富的开发人员必须创建 OpenAPI 文档。

以下主题提供了开发 REST API 连接器的信息和说明：

9.1.1. REST API 客户端连接器的要求

将 OpenAPI 模式上传到 Fuse Online 后，到 REST API 的连接器将变为可用。您可以选择连接器来创建 REST API 客户端连接。然后，您可以创建新集成并添加 REST API 客户端连接，也可以编辑现有的集成以添加 REST API 客户端连接。

Fuse Online 连接器支持 OAuth 2.0、HTTP 基本授权和 API 密钥。如果访问 REST API 需要传输层安全(TLS)，API 需要使用可识别的证书颁发机构(CA)发布的有效证书。

使用 OAuth 的 REST API 必须有一个授权 URL，它将客户端回调 URL 作为输入。在 Fuse Online 创建连接器后，在使用连接器来创建连接前，您必须访问该 URL 将 Fuse Online 环境注册为 REST API 的客户端。这授权 Fuse Online 环境访问 REST API。作为注册的一部分，您可以提供 Fuse 在线回调 URL。有关执行此操作的详情，请参考 将 Fuse Online 连接到应用程序和服务，将 Fuse Online 注册为 REST API 客户端。

对于使用 OAuth 的 REST API，Fuse Online 仅支持 授权代码 授权流。在您上传以创建连接器的 OpenAPI 文档中，在 OAuth securityDefinitions 对象中，您必须将 flow 属性设置为 accessCode，例如：

```
securityDefinitions:
  OauthSecurity:
    type: oauth2
    flow: accessCode
    authorizationUrl: 'https://oauth.simple.api/authorization'
    tokenUrl: 'https://oauth.simple.api/token'
```

您不能将 流 设置为 隐式、密码 或 应用程序。

REST API 客户端连接器的 OpenAPI 模式无法具有 cyclic 模式引用。例如，指定请求或响应正文的 JSON 模式无法作为整体引用，也无法通过任意数量的中间模式引用其自身的任何部分。

Fuse Online 无法为支持 HTTP 2.0 协议的 REST API 创建连接器。

9.1.2. REST API 客户端连接器的 OpenAPI 模式指南

当 Fuse Online 创建 REST API 客户端连接器时，它会将 OpenAPI 文档中的每个资源操作映射到连接操作。操作名称和描述来自 OpenAPI 文档中的文档。

OpenAPI 文档提供的更多详细信息，在连接到 API 时，Fuse Online 提供了更多支持。例如，不需要为请求和响应声明数据类型。如果没有类型声明，Fuse Online 会将对应的连接操作定义为无类型。但是，在集成中，您无法在执行无类型操作的 API 连接前或立即添加数据映射步骤。

一个补救措施是向 OpenAPI 文档添加更多信息。识别将映射到您希望 API 连接执行的操作的 OpenAPI 资源操作。在 OpenAPI 文档中，确保有一个 YAML 或 JSON 模式，用于指定每个操作的请求和响应类型。

上传架构后，Fuse Online 为您提供了在 API Designer 中查看和编辑它的机会，这是基于 OpenAPI 文档设计 API 的可视化编辑器。您可以添加更多详情、保存更新，Fuse Online 创建一个包含更新的 API

客户端连接器。在 Fuse Online 创建客户端连接器后，您无法再编辑 OpenAPI 文档。要实现更改，必须创建新的客户端连接器。

如果 API 的 OpenAPI 文档声明了对 application/json 内容类型以及 application/xml 内容类型的支持，则连接器将使用 JSON 格式。如果 OpenAPI 文档指定 消耗 或生成 定义 application/json 和 application/xml 的参数，则连接器使用 JSON 格式。

9.1.3. 在参数中提供客户端凭证

当 Fuse Online 尝试获取访问 OAuth2 应用程序的授权时，它会使用 HTTP 基本身份验证来提供客户端凭据。如果需要，您可以更改此默认行为，以便 Fuse Online 将客户端凭证作为参数传递给提供程序，而不使用 HTTP 基本身份验证。这会影响用于获取 OAuth 访问令牌的 tokenUrl 端点。

重要

这是一个技术预览功能。

要指定 Fuse Online 应将客户端凭证作为参数传递，请在 OpenAPI 文档的 securityDefinitions 部分中，添加 x-authorize-using-parameters 厂商扩展，并设置 true。在以下示例中，最后一行指定 x-authorize-using-parameters ：

```
securityDefinitions:
  concur_oauth2:
    type: 'oauth2'
    flow: 'accessCode'
    authorizationUrl: 'https://example.com/oauth/authorize'
    tokenUrl: 'https://example.com/oauth/token'
    scopes:
      LIST: Access List API
    x-authorize-using-parameters: true
```

x-authorize-using-parameters 厂商扩展的设置是 true 或 false ：

- true 表示客户端凭据在参数中。
- 错误 （默认值）表示 Fuse Online 使用 HTTP 基本身份验证来提供客户端凭证。

9.1.4. 自动刷新访问令牌

如果访问令牌具有过期日期，则 Fuse Online 集成时使用该令牌连接到应用程序，在令牌过期时停止运行。要获取新的访问令牌，您必须重新连接到应用程序，或使用应用程序重新注册。

如果需要，您可以更改此默认行为，以便 Fuse Online 在以下情况下自动请求新的访问令牌：

- 到期日期之后。
- 当收到您指定的 HTTP 响应状态代码时。

重要

这是一个技术预览功能。

要指定 Fuse Online 应自动尝试在 OpenAPI 文档的 securityDefinitions 部分中获取新的访问令牌，请添加 x-refresh-token-retry-statuses 厂商扩展。此扩展的设置是一个以逗号分隔的列表，用于指定 HTTP 响应状态代码。当访问令牌的过期日期达到或 Fuse Online 收到来自 OAuth2 供应商的消息时，并且消息具有这些响应状态代码之一时，Fuse Online 会自动尝试获取新的访问令牌。在以下示例中，最后一行指定 x-refresh-token-retry-statuses ：

```
securityDefinitions:
  concur_oauth2:
    type: 'oauth2'
    flow: 'accessCode'
    authorizationUrl: 'https://example.com/oauth/authorize'
    tokenUrl: 'https://example.com/oauth/token'
    scopes:
      LIST: Access List API
    x-refresh-token-retry-statuses: 401,402,403
```

注意

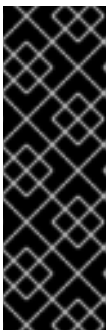
有时，API 操作会失败，且该故障的副作用是刷新访问令牌。在这种情况下，即使获取新的访问令牌成功，API 操作仍然会失败。换句话说，Fuse Online 在收到新的访问令牌后不会重试失败的 API 操作。

9.2. 添加和管理 API 客户端连接器

Fuse Online 可以创建这些 API 客户端连接器：

- OpenAPI 文档中的 REST API 客户端连接器。有关 OpenAPI 文档内容的详情，请参考 开发 REST API 客户端连接器。

- 来自 WSDL 文件的 SOAP API 客户端连接器。



重要

SOAP API 客户端连接器只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些功能可提供早期访问将来的产品功能，使客户在开发过程中测试并提供反馈。有关红帽技术预览功能支持范围的更多信息，请参阅 https://access.redhat.com/support/offerings/techpreview/

以下主题提供了添加和管理 REST API 客户端连接器的信息和说明：

- 第 9.2.1 节 “创建 REST API 客户端连接器”

- 第 9.2.2 节 “创建 SOAP API 客户端连接器”

- 第 9.2.3 节 “通过创建新 API 客户端连接器来更新 API 客户端连接器”

- 第 9.2.4 节 “删除 API 客户端连接器”

创建 API 客户端连接器后，有关使用该连接器的详情，请参阅 将 Fuse Online 连接到应用程序和服务，连接到 API 客户端。

9.2.1. 创建 REST API 客户端连接器

上传 OpenAPI 文档，以启用 Fuse Online 创建 REST API 客户端连接器。

前提条件

您有一个 OpenAPI 文档，用于希望 Fuse Online 创建的连接器。

流程

1. 在 Fuse Online 导航面板中，点 Customizations > API Client Connectors。此处列出了已可用的任何 API 客户端连接器。

2. 点 Create API Connector。

3. 在 Create API Connector 页面中，执行以下操作之一：

   - 点击点行框并选择您要上传的 OpenAPI 文件。
   - 选择 Use a URL，并在输入字段中粘贴 OpenAPI 文档的 URL。

4. 点击 Next。如果有无效或缺失的内容，Fuse Online 将显示有关需要更正内容的信息。选择要上传或点击 Cancel 的不同 OpenAPI 文件，重新调整 OpenAPI 文件，并上传更新的文件。

   如果架构有效，Fuse Online 会显示连接器提供的操作的摘要。这可能包括有关操作定义的错误和警告。

5. 如果您对概述满意，请单击 Next。

   或者，要重新发布 OpenAPI 文档，请点 Review/Edit 以打开 API Designer 编辑器。根据需要更新架构。有关使用 API 编辑器的详情，请参阅使用 API Designer 设计和开发 API 定义。完成后，保存您的更改，将更新合并到新的 API 客户端连接器中。然后，点 Next 继续创建 API 客户端连接器。

   有时，如果您为 OpenAPI 文档提供 URL，Fuse Online 可以上传它，但无法打开它进行编辑。通常，这由文件主机上的设置所致。要打开该模式进行编辑，Fuse Online 需要文件主机有：

   - https URL。（http URL 不起作用。）
   - 启用 CORS。

6.

指明 API 的安全要求。Fuse Online 读取 OpenAPI 定义，以确定配置连接器以满足 API 安全要求所需的信息。Fuse Online 可以显示以下任意一种：

a. 无安全性

b. HTTP Basic Authorization - to the API 服务使用 HTTP 基本授权，请选择此复选框。之后，当您使用此连接器创建连接时，Fuse Online 会提示您输入用户名和密码。

c. OAuth 2.0 - Fuse Online 提示您输入：

i. 授权 URL 是将 Fuse Online 注册为 API 客户端的位置。注册授权 Fuse 在线访问 API。请参阅 将 Fuse 在线连接到应用程序和服务，将 Fuse Online 注册为 REST API 客户端。API 的 OpenAPI 文档或其他文档应该指定这个 URL。如果没有，您必须联系服务提供商来获取此 URL。

ii. OAuth 授权需要访问令牌 URL。同样，API 的 OpenAPI 文档或其他文档应该提供这个 URL。如果没有，您必须联系服务提供商。

d. API Key - CamelAwsSlf API 服务需要 API 密钥，Fuse Online 会提示输入创建连接器所需的任何信息。提示基于 OpenAPI 定义。例如，您可能需要指定 API 键是否在消息标头或查询参数中。如果 OpenAPI 定义指定了 API 密钥安全性，以及另一个安全类型，请选中复选框来表示您要根据这个连接器在连接中使用 API 密钥安全性。之后，当您使用此连接器创建连接时，Fuse Online 会提示您输入 API 键的值。

7. 点击 Next。Fuse Online 显示连接器的名称、描述、主机和基本 URL，如 OpenAPI 文档所示。对于从此连接器创建的连接，

- Fuse Online 连接主机和基础 URL 值，以定义连接的端点。例如，如果主机是 https://example.com，基础 URL 为 /api/v1，则连接端点为 https://example.com/api/v1。
- Fuse Online 将 OpenAPI 文档应用到数据映射步骤。如果 OpenAPI 文档支持多个模式，Fuse Online 将使用 TLS (HTTPS)模式。

8. 查看连接器详情，并选择性地上传连接器的图标。如果没有上传图标，Fuse Online 会生成一个图标。您可以稍后上传图标。当 Fuse Online 显示集成流时，它会显示一个连接器的图标，以代表从该连接器创建的连接。

9. 要覆盖从 OpenAPI 文件获取的值，请编辑您要更改的字段值。

重要

在 Fuse Online 创建连接器后，您无法更改它。要生效更改，您需要上传更新的 OpenAPI 文档，以便 Fuse Online 能够创建新连接器，或者您可以上传相同的模式，然后在 API 编辑器中编辑它。然后，继续创建新 API 客户端连接器的过程。

10. 当您满足连接器详情时，点 Save。Fuse Online 在 API 客户端连接器列表中显示新连接器。

后续步骤

有关使用新的 API 连接器的详情，请参阅 将 Fuse 在线连接到应用程序和服务，连接到 API 客户端。

9.2.2. 创建 SOAP API 客户端连接器

上传 WSDL 文件，使 Fuse 在线创建 SOAP API 客户端连接器。

重要

SOAP API 客户端连接器只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些功能可提供早期访问将来的产品功能，使客户在开发过程中测试并提供反馈。有关红帽技术预览功能支持范围的更多信息，请参阅 https://access.redhat.com/support/offerings/techpreview/

前提条件

您有一个适用于希望 Fuse Online 创建的 SOAP 客户端连接器的 WSDL 文件。

注意：在此版本中，WSDL 文件的扩展名(.wsdl)必须为小写。如果文件扩展为大写(.WSDL)，您必须将该文件重命名为具有小写扩展名，然后才能将其导入到 Fuse Online 中。

流程

1.

在 Fuse Online 导航面板中，点 Customizations > API Client Connectors。此处列出了已可用的任何 API 客户端连接器。

2. 点 Create API Connector。

3. 在 Create API Connector 页面中，执行以下操作之一：

   - 单击点行框，再选择您要上传的 WSDL (.wsdl)文件。
   - 选择 Use a URL，并在输入字段中粘贴 WSDL (.wsdl)文件的 URL。

4. 点击 Next。

5. 在 Specify 服务和端口 页面中，验证服务和端口。

6. 点击 Next。如果有无效或缺失的内容，Fuse Online 将显示有关需要更正内容的信息。选择要上传或单击 Cancel 的不同 WSDL 文件，重新调整 WSDL 文件，然后上传更新的文件。如果架构有效，Fuse Online 会显示 API 定义摘要（名称和描述）以及导入的元素列表，如操作数量。

7. 点击 Next。

注：在此发行版本中，本页上的安全要求不活跃。相反，您可以在从连接器创建连接时定义安全选项，如 将 Fuse Online 连接到 Applications 和 Services 所述，创建 SOAP API 客户端连接。

1. 点击 Next。Fuse Online 显示连接器的名称、描述和 WSDL 端点地址。

   - 另外，还可上传连接器的图标。您还可以稍后上传图标。

     注：对于这个版本，如果没有上传图标，Fuse Online 不会 为您生成一个图标。

     当 Fuse Online 显示集成流时，它会显示一个连接器的图标，以代表从该连接器创建的

连接。

- 对于 Name，输入您选择的名称，它可帮助您将这个连接与任何其他连接区分开来。
- （可选）对于 Description，输入有助于了解此连接的信息。

2. 查看连接器详情，并覆盖从 WSDL 文件中获取的值，编辑您要更改的字段值。

重要

在 Fuse Online 创建连接器后，您无法更改它。要生效更改，您需要上传更新的 OpenAPI 文档，以便 Fuse Online 能够创建新连接器，或者您可以上传相同的模式，然后在 API 编辑器中编辑它。然后，继续创建新 API 客户端连接器的过程。

3. 当您满足连接器详情时，点 Save。Fuse Online 在 API 客户端连接器列表中显示新连接器。

后续步骤

有关使用新的 API 连接器的详情，请参阅 将 Fuse 在线连接到应用程序和服务，连接到 API 客户端。

9.2.3. 通过创建新 API 客户端连接器来更新 API 客户端连接器

当您从中创建 API 客户端连接器的 OpenAPI 文档或 WSDL 文件时，并且希望 API 客户端连接器使用这些更新，您必须创建新的 API 客户端连接器。您无法直接更新 API 客户端连接器。创建新的 API 客户端连接器后，您可以使用它创建新连接，然后编辑使用从 out-of-date 连接器创建的连接的每个集成。

先决条件

准备执行以下操作之一：

- 对于 REST API 客户端连接器：
  - 上传更新的 OpenAPI 文档。

- 再次上传过时的模式，并在 API Designer 中更新它。

- 对于 SOAP API 客户端连接器，上传更新的 WSDL 文件。

流程

1. 根据更新的 OpenAPI 文档或 WSDL 文件创建一个新的 API 客户端连接器。要方便地区分旧的连接器和新连接器，您可能需要在连接器名称或连接器描述中指定版本号。

   请参阅 开发 REST API 客户端连接器。

2. 从新连接器创建新连接。同样，您希望能够在从旧连接器创建的连接和从新连接器创建的连接之间轻松区分。连接名称或连接描述中的版本号会很有用。

3. 通过删除旧连接并添加新连接来编辑使用从旧连接器创建的连接的每个集成。

4. 发布每个更新的集成。

5. 建议但不是必需的：删除旧的连接器和旧连接。

9.2.4. 删除 API 客户端连接器

当存在从那个连接器创建的连接时，您无法删除连接器，并在集成中使用此连接。删除 API 客户端连接器后，您无法使用从该连接器创建的连接。

流程

1. 在左侧面板中，点 Customizations > API Client Connectors。

2. 在您要删除的连接器名称右侧，点 Delete。

3. 在确认弹出窗口中，如果您确定要删除连接器，请点击 Delete。

9.3. 开发 FUSE 在线扩展

如果 Fuse Online 不提供创建集成所需的功能，则专家开发人员就可以对提供所需行为的扩展进行编码。Syndesis 扩展存储库 https://github.com/syndesisio/syndesis-extensions 包含扩展示例。

商业集成商与编码扩展的开发人员共享要求。开发人员提供了一个包含 .jar 扩展的 .jar 文件。商业集成商上传 Fuse Online 中的 .jar 文件，使自定义连接器、自定义步骤或库资源可用于 Fuse Online。

Red Hat Developer Studio 的 Fuse Tooling 插件提供了一个向导，可帮助您开发步骤扩展或连接器扩展。无论您选择开发一个步骤扩展，还是在 Developer Studio 中还是在某些其他 IDE 中，最好选择开发步骤扩展。有关使用 Developer Studio 插件的详情，请参考为 Fuse Online 集成开发扩展。

在本文档中，以下主题概述了流程，描述要求，并提供您在您选择的 IDE 中开发扩展的其他示例。

- 第 9.3.1 节 “开发扩展的一般流程”
- 第 9.3.2 节 “扩展类型的描述”
- 第 9.3.3 节 “扩展内容和结构概述”
- 第 9.3.4 节 “扩展定义 JSON 文件中的要求”
- 第 9.3.5 节 “用户界面属性的描述”
- 第 9.3.6 节 “支持扩展的 Maven 插件描述”
- 第 9.3.7 节 “如何在扩展中指定数据形成”
- 第 9.3.8 节 “开发步骤扩展示例”
- 第 9.3.9 节 “开发连接器扩展示例”

- 第 9.3.10 节 “如何开发库扩展”
- 第 9.3.11 节 “创建 JDBC 驱动程序库扩展”

9.3.1. 开发扩展的一般流程

在开始开发扩展之前，熟悉您需要完成的任务。

先决条件

- 熟悉 Maven
- 如果您要开发一个提供连接器的扩展，或者提供在连接间数据上运行的集成步骤，请熟悉 Camel
- 经验编程

小心

集成 pod 在带有扁平类路径的 Java 进程中运行。要避免版本冲突，请确保扩展所使用的依赖项与这些源中导入的材料清单(BOM)一致：

- org.springframework.boot:spring-boot-dependencies:$SPRING_BOOT_VERSION
- org.apache.camel:camel-spring-boot-dependencies:$CAMEL_VERSION
- io.syndesis.integration:integration-bom:$SYNDESIS_VERSION

如果其他依赖项不是导入的 BOM 的一部分，您必须：

- 将它们打包在扩展名 JAR 文件中，该文件位于 lib 目录中。
- 从扩展的 JSON 描述符文件的 dependencies 属性中省略它们。

流程

1. 了解扩展功能必须做什么。与您的业务沟通以了解功能要求。
2. 确定您需要开发一个步骤扩展、连接器扩展还是库扩展。
3. 设置 Maven 项目，在其中开发扩展。
4. 如果您要开发一个步骤扩展：
   a. 决定是否将其作为 Camel 路由使用 Syndesis Step API 实现。Syndesis API 的信息位于 http://javadoc.io/doc/io.syndesis.extension/extension-api。
   b. 如果您选择将扩展实施为 Camel 路由，决定是否实现 XML 片段、RouteBuilder 类还是 bean。

c. 在 Maven 项目中，指定所需的元数据，如 schemaVersion、扩展 名称、extensionId 等等。

5. 对实施该功能的类进行编码。

6. 向项目的 pom.xml 文件添加依赖项。

7. 对于连接器和库扩展，以及您在 XML 中实施的步骤扩展，请创建用于定义扩展的 JSON 文件。

对于您在 Java 中实施的步骤扩展，当您在 Maven 项目中指定对应的数据结构值时，Maven 可以为您生成 JSON 扩展定义文件。

8. 运行 Maven 以构建扩展并创建扩展的 JAR 文件。

9. 通过将 JAR 文件上传到 Fuse 在线开发环境来测试扩展。

10. 提供 JAR 文件，将扩展打包给您的业务 colleague，该文件将其上传到 Fuse 在线生产环境。当您提供 JAR 文件时，请让您的业务人员了解除 Fuse 在线 Web 界面中显示的信息以外的任何需要信息的配置设置。

9.3.2. 扩展类型的描述

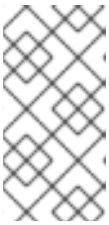
扩展定义了以下之一：

- 对连接之间的集成数据操作的一个或多个自定义步骤。每个自定义步骤都执行一个操作。这是一个步骤扩展。

- 集成运行时使用的库资源。例如，库扩展可以提供 JDBC 驱动程序以连接到专有 SQL 数据库，如 Oracle。

- 一个单一自定义连接器，用于创建到您要集成的特定应用程序或服务的连接。这是连接器扩

展。



注意

Fuse Online 可以使用 OpenAPI 文档为 REST API 客户端创建连接器。请参阅 开发 REST API 客户端连接器。

商业集成商与编码扩展的开发人员共享要求。开发人员提供了一个包含 .jar 扩展的 .jar 文件。商业集成商上传 Fuse Online 中的 .jar 文件，使自定义连接器、自定义步骤或库资源可在 Fuse Online 中使用。

上传到 Fuse Online 的扩展 .jar 文件始终包含一个扩展。

有关上传和使用提供连接间数据步骤的扩展示例，请参阅 AMQ 到 REST API 示例集成指南。

9.3.3. 扩展内容和结构概述

扩展是打包在 .jar 文件中的类、依赖项和资源的集合。

Fuse Online 使用 Spring Boot 加载扩展。因此，您必须根据 Spring Boot 的可执行 JAR 格式打包扩展。例如，确保使用 ZipEntry.STORED （） 方法保存嵌套 JAR 文件。

软件包扩展名的 .jar 文件的结构如下：

```
extension.jar
|
+- META-INF
| |
| +- syndesis
|   |
|   +- syndesis-extension-definition.json (1)
|
+- mycompany
| |
| +-project
|   |
|   +-YourClasses.class (2)
|
+- lib (3)
```

```
|
+-dependency1.jar
|
+-dependency2.jar
```

1 指定定义扩展的数据结构的 JSON 架构文件。这称为扩展定义 JSON 文件。

2 实施扩展提供的行为的 Java 类。

3 构建和执行自定义功能所需的其他依赖项。

9.3.4. 扩展定义 JSON 文件中的要求

每个扩展都必须有一个 .json 文件，该文件通过为数据结构指定值来定义扩展，如名称、描述、支持的操作和依赖项。对于每个扩展类型，下表指示 Maven 是否可以生成扩展定义 JSON 文件以及需要哪些数据结构。

扩展类型	Maven 可以生成扩展定义	所需的数据结构
Java 中的步骤扩展	是	**schemaVersion** **name** **description** **version** **extensionId** **extensionType** **actions** **dependencies** *
XML 中的步骤扩展	否	**schemaVersion** **name** **description** **version** **extensionId** **extensionType** **actions** **dependencies** *

扩展类型	Maven 可以生成扩展定义	所需的数据结构
连接器扩展	否	**schemaVersion** **name** **description** **version** **extensionId** **extensionType** **properties** **actions** **dependencies** * **componentScheme** **connectorCustomizers** **connectorFactory**
库扩展	否	**schemaVersion** **name** **description** **version** **extensionId** **extensionType** **dependencies** *

** 虽然规格不严格要求，但在实践中，您几乎需要指定依赖项。*

通常，扩展定义文件具有以下布局：

```
{
  "schemaVersion": "v1",
  "name": "",
  "description": "",
  "version": "",
  "extensionId": "",
  "extensionType": "",
  "properties": {
  },
  "actions": [
  ],
  "dependencies": [
  ],
}
```

- *schemaVersion 定义扩展定义模式的版本。在内部，Syndesis 使用 schemaVersion 来确定如何将扩展定义映射到内部模型。这允许针对旧版本 Syndesis 开发的扩展部署在较新版本的 Syndesis 中。*

- name 是扩展的名称。将扩展上传到 Fuse Online 时，将显示此名称。
- description 是您要指定的任何有用信息。Fuse Online 不对这个值进行操作。
- 版本 是为了方便您区分对扩展的更新。Fuse Online 不对这个值进行操作。
- extensionId 为扩展定义唯一 ID。这应该至少可在 Syndesis 环境中唯一。
- extensionType 表示扩展所提供的扩展。从 Syndesis 版本 1.3 开始，支持以下扩展类型：
  - 步骤
  - 连接器
  - 库
- 需要连接器扩展中的顶级 属性。它控制 Fuse Online 用户在 Fuse Online 用户选择创建连接的连接器时显示的内容。这个属性 对象包含用于创建连接的每个表单控制的一组属性。例如：

```
"myControlName": {
 "deprecated": true|false,
 "description": "",
 "displayName": "",
 "group": "",
 "kind": "",
 "label": "",
 "required": true|false,
 "secret": true|false,
 "javaType": "",
 "type": "",
 "defaultValue": "",
 "enum": {
 }
}
```

在连接器扩展中，嵌套属性对象定义了 HTML 表单控制来配置连接操作。在步骤扩展中，action 对象包含一个 properties 对象。properties 对象定义了一组属性，用于配置步骤的每个

表单控制。另请参阅：用户界面属性的描述。

- 操作 定义了连接器可以执行的操作，或者连接间步骤可以执行的操作。只有连接器和步骤扩展使用您指定的操作。操作规格的格式类似如下：

```
{
  "id": "",
  "name": "",
  "description": "",
  "actionType": "step|connector",
  "descriptor": {
  }
}
```

  - id 是操作的唯一 ID。这应该至少在 Syndesis 环境中唯一。
  - name 是 Fuse Online 中出现的操作名称。整合商将此值视为连接操作的名称，或作为连接之间集成数据运行的步骤名称。
  - description 是 Fuse Online 中显示的操作描述。使用此字段帮助集成商了解该操作的作用。
  - actionType 指示通过连接执行该操作还是连接之间的步骤。
  - 描述符 指定嵌套属性，如 kind,entrypoint,inputDataType,outputDatatype 等。

- 依赖项 定义此扩展需要 Fuse 在线提供的资源。

  定义依赖项，如下所示：

```
{
 "type": "MAVEN",
 "id"  : "org.apache.camel:camel-telegram:jar:2.21.0"
}
```

  - type 表示依赖项的类型。指定 MAVEN。（预计将来将支持其他类型。）

- ID 是 Maven 依赖项的 ID，它是一个 Maven GAV。

9.3.5. 用户界面属性的描述

在连接器扩展和步骤扩展中，在扩展定义 JSON 文件或 Java 类文件中指定用户界面属性。这些属性的设置定义 HTML 表单控制 Fuse Online 在 Fuse Online 用户创建连接时显示，配置连接操作，或者配置扩展提供的步骤。

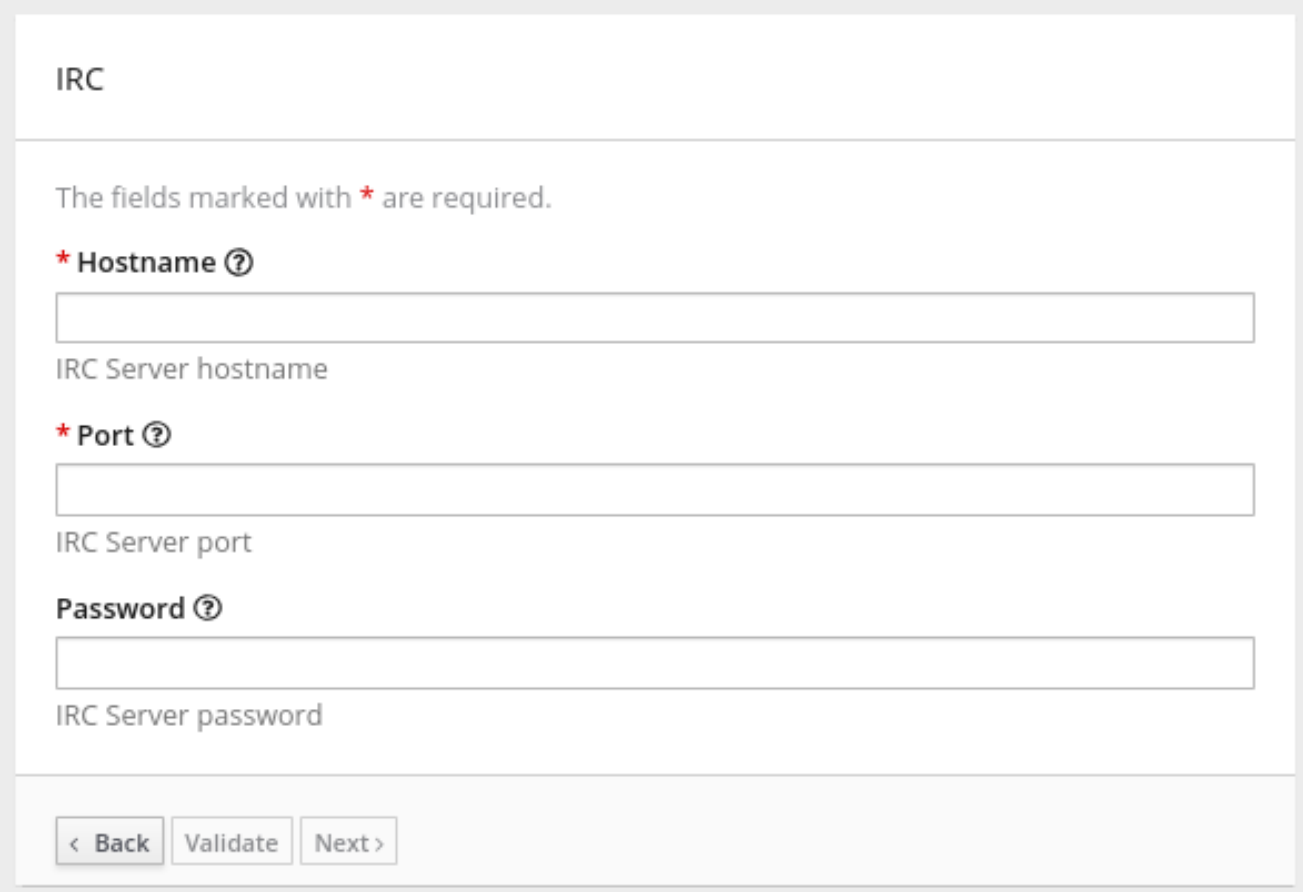
您必须为每个要出现在 Fuse Online 控制台中的扩展用户界面中的表单控制指定属性。对于每个表单控制，按任何顺序指定 some 或 all 属性。

用户界面属性规格示例

在作为 IRC 连接器一部分的 JSON 文件中，顶级 属性 对象定义了在 Fuse Online 用户选择 IRC 连接器来创建连接的 HTML 表单控制。三种形式有三组属性定义： 主机名、密码 和端口 ：

```
"properties": {
  "hostname": {
    "description": "IRC Server hostname",
    "displayName": "Hostname",
    "labelHint": "Hostname of the IRC server to connect to",
    "order": "1",
    "required": true,
    "secret": false,
    "type": "string"
  },
  "password": {
    "description": "IRC Server password",
    "displayName": "Password",
    "labelHint": "Required if IRC server requires it to join",
    "order": "3",
    "required": false,
    "secret": true,
    "type": "string"
  },
  "port": {
    "description": "IRC Server port",
    "displayName": "Port",
    "labelHint": "Port of the IRC server to connect to",
    "order": "2",
    "required": true,
    "secret": false,
    "tags": [],
    "type": "int"
  }
},
```

根据这些属性规格，当 Fuse Online 用户选择 IRC 连接器时，Fuse Online 会显示以下对话框：用户在两个必填字段中输入值并点击 Next 后，Fuse Online 创建一个 IRC 连接，该连接配置了 Fuse Online 用户输入的值。



关于扩展定义 JSON 文件中的属性对象

在连接器扩展中：

- 顶级属性对象是必需的。它控制 Fuse Online 用户在 Fuse Online 用户选择创建连接的连接器时显示的内容。这个属性对象包含用于创建连接的每个表单控制的一组属性。

- 在 actions 对象中，每个操作都有一个 properties 对象。在 每个属性 对象中，每个表单控制都有一组属性来配置该操作。

在步骤扩展中，action 对象包含一个 properties 对象。properties 对象定义了一组属性，用于配置步骤的每个表单控制。JSON 层次结构类似如下：

```
"actions": [
  {
    ...

    "propertyDefinitionSteps": [
      {
        ...

        "properties":
          {
            "control-ONE": {
              "type": "string",
              "displayName": "Topic Name",
              "order": "2",
              ...,
              }

            "control-TWO": {
              "type": "boolean",
              "displayName": "Urgent",
              "order": "3",
              ...
              }

            "control-THREE": {
              "type": "textarea",
              "displayName": "Comment",
              "order": "1",
              ...,
              }
} } ]
```

关于 Java 文件中的用户界面属性

要在 Java 文件中定义用户界面表单控制，请在定义连接、操作或步骤的每个类文件中导入 io.syndesis.extension.api.annotations.ConfigurationProperty。对于您希望 Fuse Online 控制台显示的每个表单控制，请指定 @ConfigurationProperty 注释，后跟一个属性列表。有关您可以指定的属性的详情，请参考本节末尾的用户界面属性参考表。

以下代码显示了一个表单控制的属性定义。此代码位于使用 RouteBuilder 开发 Camel 路由的示例：

```
public class LogAction extends RouteBuilder {
   @ConfigurationProperty(
      name = "prefix",
      description = "The Log body prefix message",
      displayName = "Log Prefix",
      type = "string")
```

以下代码显示了两个控制的属性定义。这个代码来自使用 Syndesis Step API 的示例：

```
@Action(id = "split", name = "Split", description = "Split your exchange")
public class SplitAction implements Step {

    @ConfigurationProperty(
        name = "language",
        displayName = "Language",
        description = "The language used for the expression")
    private String language;

    @ConfigurationProperty(
        name = "expression",
        displayName = "Expression",
        description = "The expression used to split the exchange
     private String language;
```

控制格式输入类型的描述

在每个 HTML 表单控制的一组属性中，type 属性定义 Fuse Online 显示的形式控制的输入类型。有关 HTML 表单输入类型的详情，请参考 https://www.w3schools.com/html/html_form_input_types.asp。

下表列出了 Fuse Online 表单控制的可能输入类型。在控制的一组属性中，如果您指定了一个未知的 type 值，Fuse Online 会显示一个接受一行文本的输入字段。也就是说，默认值为 "type": "text "。

type 属性的值	HTML	Fuse Online 显示
布尔值	**<input type="checkbox">**	用户可以选择或无法选择的复选框。
duration	允许 Fuse Online 用户选择一整时间的自定义控制：毫秒、秒、分钟、小时或天。用户也输入数字，Fuse Online 返回了几毫秒。例如： **"properties": {** **"period": {** **"type": "duration"** **"defaultValue": 60000,** **"description": "Period",** **"displayName": "Period",** **"labelHint": "Delay between integration executions.",** **"required": true,** **"secret": false,** **}** **}**	
hidden	**<input type="hidden">**	此字段不会出现在 Fuse Online 控制台中。您可以使用其他属性指定与此字段关联的数据，例如某些类型的文本数据。虽然 Fuse Online 用户无法查看或修改此数据，如果用户为 Fuse Online 页面选择 **View Source**，但隐藏的字段在源显示中可见。因此，不要将隐藏字段用于安全目的。

type 属性的值	HTML	Fuse Online 显示
int、整数、长、数字	**<input type="number">**	接受数字的输入字段。
password	**<input type="password">**	Fuse Online 屏蔽用户输入的字符的输入字段，通常是星号。
select	一个 & **lt;select&** gt; 元素，例如： **<select name="targets">** **<option value="queue">Queue</option>** **<option value="topic">Topic</option>** **</select>**	带有您在表单控制的 **enum** 属性中指定的每个标签/值对的下拉列表。
文本、字符串 或任何未知值	**<input type="text">`**	接受一行文本的输入字段。
textarea	**<input type="textarea"**	使用 textarea 元素

控制格式用户界面属性的描述

在连接器或步骤扩展中，对于 Fuse Online 控制台中的每个 HTML 表单控制，您可以指定下表中描述的一个或多个属性。有关 HTML 表单输入类型的详情，请参考 https://www.w3schools.com/html/html_form_input_types.asp。

属性名称	类型	描述
type	字符串	控制 Fuse 在线显示的表单控制。详情请查看上表。
cols	number	如果为 **textarea** 字段设置，则控制最初为文本控制显示的列数。
controlHint or **controlTooltip**	字符串	如果设置，该值将映射到表单控制元素的 HTML **title** 属性。与具有 **title** 属性的其他元素一样，当光标将鼠标悬停在控制上时，会显示工具提示。工具提示的内容来自 **controlHint** 或 **controlTooltip** 属性的值。
dataList	数组	如果 **type** 属性的值是 **文本**，Fuse Online 使用 **dataList** 属性的值来添加 typeahead 支持。指定字符串数组。

属性名称	类型	描述
defaultValue	根据 **type** 属性的值而有所不同。	Fuse Online 最初在表单字段中显示这个值。**defaultValue** 属性的设置类型应与 **type** 属性的值匹配。例如，当 **type** 属性设为 **number** 时，**defaultValue** 设置应该是数字。如果用户没有更改此初始字段值，Fuse Online 将使用 **defaultValue**。
description	字符串	如果设置，Fuse Online 会在表单控制的下面显示这个值。通常，这是关于控制的简短有用消息。
displayName	字符串	Fuse Online 显示这个值。
enum	数组	如果设置，Fuse Online 会覆盖 **type** 属性的任何设置，并实施 **选择** 控制。将数组指定为一组标签 **和值** 属性。**label** 属性以选择项的标签出现在用户界面中。**value** 属性成为对应选择项的值。
labelHint or **labelTooltip**	字符串	如果设置，显示名称旁边会出现 **?** 图标。当 Fuse Online 用户单击 **?** 图标时，将显示 **labelHint** 属性的值。
max	number	如果为 **数字** 字段设置，请定义最高可接受值。
åˆ†é’Ÿ	number	如果为 **数字** 字段设置，定义最低的可接受值。
multiple	布尔值	如果为 **选择字段或设置了 enum** 属性的字段设置为 **true**，Fuse Online 会显示多选择控制而不是选择下拉菜单。
order	number	决定 Fuse 在线控制台中的控制顺序。Fuse Online 应用升序，即首先出现 **"order": "1"** 的控制。如果没有指定 **order** 属性，Fuse Online 会以 JSON 文件定义它们的顺序显示控制。
placeholder	字符串	如果设置，Fuse Online 会在输入字段中的 hazed font 中显示这个值，以帮助用户了解预期的输入。
required	布尔值	控制控制上是否设置了所需的属性。**如果为 true，则 Fuse Online 用户必须为此控制输入一个值。**
rows	number	如果 **type** 属性的值是 **textarea**，则 **rows** 属性的值会控制文本控制中最初显示的行数。

属性名称	类型	描述
secret	布尔值	如果指定，Fuse Online 将控制 **type** 属性的设置改为 **password**（如果还没有设置）。

9.3.6. 支持扩展的 Maven 插件描述

extension-maven-plugin 通过将扩展打包为有效的 Spring Boot 模块来支持扩展开发。对于您在 Java 中实施的步骤扩展，此插件可以生成扩展定义 JSON 文件。

在 Maven 项目的 pom.xml 文件中，添加以下插件声明：

```
<plugin>
  <groupId>io.syndesis.extension</groupId>
  <artifactId>extension-maven-plugin</artifactId>
  <version>${syndesis.version}</version>
  <executions>
    <execution>
    <goals>
      <goal>generate-metadata</goal>
      <goal>repackage-extension</goal>
    </goals>
    </execution>
  </executions>
</plugin>
```

extension-maven-plugin 定义以下目标：

- **generate-metadata 生成 JSON 扩展定义文件，该文件将在生成的 JAR 文件中，如下所示：**

  a. **Maven 从数据结构规格开始，如果位于 META-INF/syndesis/syndesis/syndesis-extension-definition.json 文件中。**

  如果要在 XML 中编码，则必须自行定义扩展定义 JSON 文件，并且必须指定所有必需的数据结构。

  如果要开发连接器或库扩展，则必须自行定义扩展定义 JSON 文件，并且必须指定所有必需的数据结构。

如果要在 Java 中开发步骤扩展，您可以：

- 自行创建扩展定义 JSON 文件。
- 在 Java 代码中，指定定义所有所需的数据结构的注解。您不创建扩展定义 JSON 文件。
- 创建扩展定义 JSON 文件，并指定一些但并非所有数据结构。

b. 对于您使用 Java 开发的步骤扩展，Maven 从代码注解中获取缺少的规格

c. Maven 添加依赖项列表，指定提供范围提供的依赖项，并且通过 extension-bom 管理。

- repackage-extension 软件包扩展。
  - 不通过 extension-bom 管理的依赖项和相关传输依赖关系位于生成的 JAR 的 lib 文件夹中。
  - 对于库扩展，其范围为 system 的依赖项位于生成的 JAR 的 lib 文件夹中。

例如，假设您 Maven 项目有以下 pom.xml 文件：

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

  <modelVersion>4.0.0</modelVersion>
  <groupId>com.company</groupId>
  <artifactId>my-extension</artifactId>
  <version>1.0.0</version>
  <name>MyExtension</name>
  <description>A Sample Extension</description>
  <packaging>jar</packaging>
```

```
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>io.syndesis.extension</groupId>
      <artifactId>extension-bom</artifactId>
      <version>1.3.10</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>

<dependencies>
  <dependency>
    <groupId>io.syndesis.extension</groupId>
    <artifactId>extension-api</artifactId>
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>com.github.lalyos</groupId>
    <artifactId>jfiglet</artifactId>
    <version>0.0.8</version>
  </dependency>
</dependencies>

<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.7.0</version>
      <configuration>
        <source>1.8</source>
        <target>1.8</target>
      </configuration>
    </plugin>
    <plugin>
      <groupId>io.syndesis.extension</groupId>
      <artifactId>extension-maven-plugin</artifactId>
      <version>1.3.10</version>
      <executions>
        <execution>
          <goals>
            <goal>generate-metadata</goal>
            <goal>repackage-extension</goal>
          </goals>
        </execution>
      </executions>
    </plugin>
  </plugins>
</build>
</project>
```

根据这个 pom.xml 文件，生成的扩展定义 JSON 文件类似如下：

```
{
  "name": "MyExtension",
  "description": "A Sample Extension",
  "extensionId": "com.company:my-extension",
  "version": "1.0.0",
  "dependencies": [ {
    "type": "MAVEN",
    "id": "io.syndesis.extension:extension-api:jar:1.3.10"
  } ],
  "extensionType": "Libraries",
  "schemaVersion": "v1"
}
```

生成的存档具有此结构和内容：

```
my-extension-1.0.0.jar
|
+- lib
|  |
|  + jfiglet-0.0.8.jar
|
+- META-INF
   |
   +- MANIFEST.MF
      |
      +- syndesis
         |
         +- syndesis-extension-definition.json
```

9.3.7. 如何在扩展中指定数据形成

数据形成包含数据映射器使用的数据类型元数据。数据映射器将此元数据转换为内部文档，用于显示数据映射器用户界面中的源和目标数据字段。在连接器或自定义步骤的扩展定义 JSON 文件中，每个操作规格都会定义一个输入数据(inputDataShape)和一个输出数据组成(outputDataShape)。

当您开发扩展时，务必要指定数据形成的属性，允许数据映射器正确处理和显示源和目标字段。以下数据形成的属性会影响数据映射程序行为：

- **kind**
- **type**
- 规格

- ***name***

- ***description***

关于 kind 属性

data form kind 属性由 DataShapeKinds enum 表示。kind 属性的可能值为：

- ***Java 表示数据类型由 Java 类表示。通过为 type 属性指定完全限定类名称，遵循 "kind": "java" 声明。例如：***

```
"outputDataShape": {
    "kind": "java",
    "type": "org.apache.camel.component.telegram.model.IncomingMessage"
},
```

- ***json-schema 表示数据类型由 JSON 模式表示。当 kind 设为 json-schema 时，指定 JSON 模式作为 data 组成的 specification 属性的值。例如：***

```
"inputDataShape": {
  "description": "Person data",
  "kind": "json-schema",
  "name": "Person",
  "specification": "{\"$schema\":\"http://json-schema.org/draft-
04/schema#\",\"title\":\"Person\",\"type\":\"object\",\"properties\":{\"firstName\":{...}}}"
}
```

SAP Concur 连接器的代码 包含由 JSON 模式指定的数据组成的示例。

- ***json-instance 表示数据类型由 JSON 实例表示。当 kind 设为 json-instance 时，指定 JSON 实例作为 datamed 的 specification 属性的值。例如：***

```
"inputDataShape": {
  "description": "Person data",
  "kind": "json-instance",
  "name": "Person",
  "specification": "{\"firstName\":\"John\",...}"
}
```

- ***xml-schema 表示数据类型由 XML 模式表示。当 kind 设为 xml-schema 时，指定 XML Schema 作为 datamed 的 specification 属性的值。例如：***

```
"inputDataShape": {
  "description": "Person data",
  "kind": "xml-schema",
  "name": "Person",
  "specification": "<?xml version=\"1.0\" encoding=\"UTF-8\" ?><xs:schema
xmlns:xs=\"http://www.w3.org/2001/XMLSchema\">...</xs:schema>"
}
```

- ***xml-instance 表示数据类型由 XML 实例表示。当 kind 设为 xml-instance 时，将 XML 实例指定为 datamed 的 specification 属性的值。例如：***

```
"inputDataShape": {
  "description": "Person data",
  "kind": "xml-instance",
  "name": "Person",
  "specification": "<?xml version=\"1.0\" encoding=\"UTF-8\" ?><Person>
<firstName>Jane</firstName></Person>"
}
```

- ***any 表示数据类型不是结构化的。例如，它可能是字节数组或自由格式文本。当其 kind 属性设为 任何 时，数据映射程序会忽略数据。换句话说，数据不会出现在数据映射程序中，因此您无法将任何字段映射到或来自这些数据。***

  但是，对于自定义连接器，当其 kind 属性设置为 任何 时，Fuse Online 会提示您在配置从自定义连接器创建的连接时指定输入和输出数据类型。当您向集成添加连接时会出现这种情况。您可以指定数据表单的模式类型、您指定的模式类型的适当文档，以及数据类型的名称。

- ***none 表示没有数据类型。对于形成输入数据，这表示连接或步骤不会读取数据。对于一个输出数据，这表示连接或步骤不会修改数据。例如，当输入消息正文传输到输出消息正文时，将 kind 属性设为 none 表示数据只通过。当 kind 设为 none 时，数据映射程序会忽略数据形图。换句话说，数据不会出现在数据映射程序中，因此您无法将任何字段映射到或来自这些数据。***

关于 type 属性

当 kind 属性的值为 java 时，"kind": "java" 声明将后跟一个 type 声明，用于指定完全限定的 Java 类名称。例如：

```
"outputDataShape": {
    "kind": "java",
    "type": "org.apache.camel.component.telegram.model.IncomingMessage"
},
```

当 kind 属性设为 java 以外的任何设置时，会忽略 type 属性的任何设置。

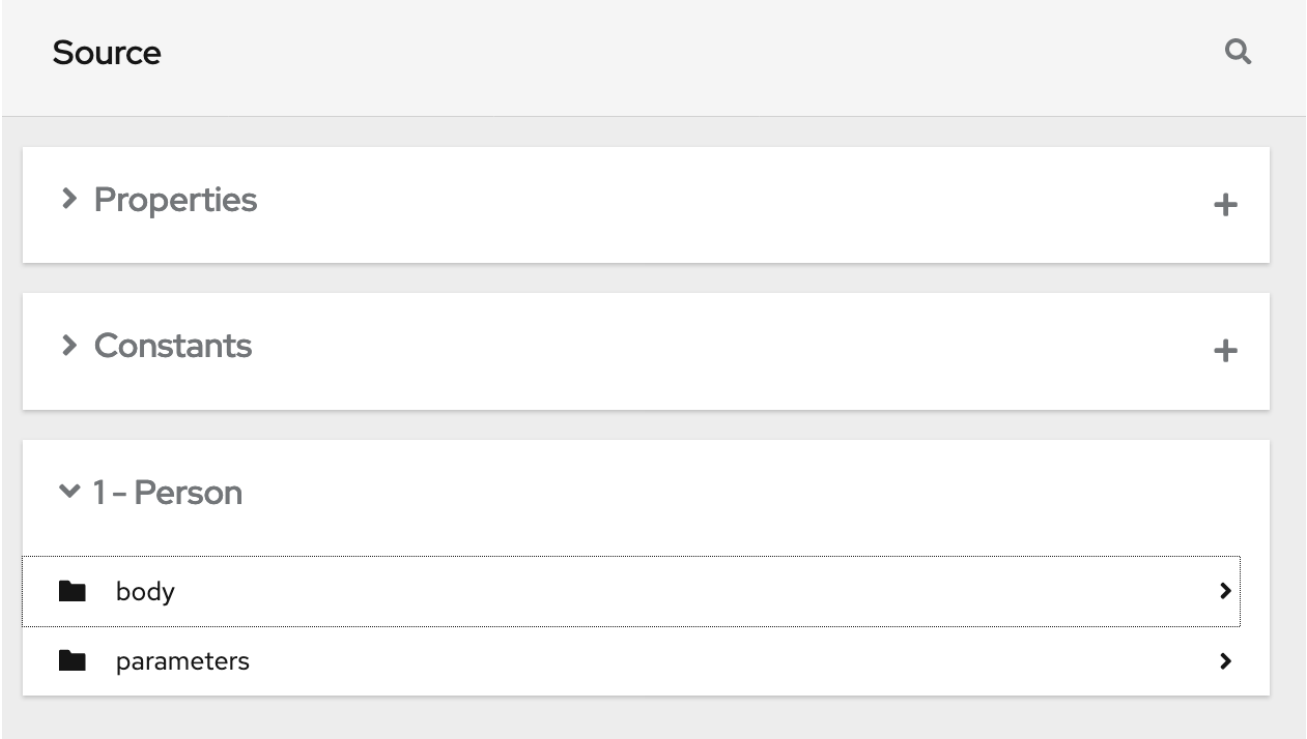
关于 规格 属性

kind 属性的设置决定了 规格 属性的设置，如下表所示。

kind 属性设置	规格 属性设置
java	Java 检查结果. 对于您在 Java 中写入的每个扩展，请使用 **extension-maven-plugin**，至少获取 Java 检查结果。该插件将 Java 检查结果插入到 JSON 扩展定义文件中，作为 **规格** 属性的设置。这是获取 Java 检查结果的唯一方法，这是 Fuse Online 中数据映射的唯一方法。 提醒，对于使用 Java 编写的步骤扩展，**extension-maven-plugin** 生成 JSON 扩展定义文件，并为它填充所需的内容。对于连接器扩展，而 **extension-maven-plugin** 在 JSON 扩展定义文件中插入 Java 检查结果，需要手动添加插件不会插入的必要内容。
json-schema	实际 JSON 模式文档。此设置不能是文档的引用，JSON 模式无法通过参考方式指向其他 JSON 架构文档。
json-instance	包含示例数据的实际 JSON 文档。数据映射器从示例数据中获取数据类型。此设置不能是文档的引用。
xml-schema	实际的 XML 模式文档。此设置不能是文档的引用，XML 模式无法通过参考方式指向其他 XML 模式文档。
xml-instance	实际的 XML 实例文档。此设置不能是文档的引用。
any	不需要 **specification** 属性。任何设置都会被忽略。
none	不需要 **specification** 属性。任何设置都会被忽略。

关于 name 属性

data form name 属性指定数据类型的人类可读名称。数据映射器在其用户界面中显示此名称作为数据字段的标签。在以下镜像中，Person 是看到 name 属性的值的示例。



此名称也会出现在 Fuse Online 流视觉化的数据类型指示符中。

关于 description 属性

datamed description 属性指定当光标悬停在数据映射器用户界面中数据类型名称时显示为工具提示的文本。

9.3.8. 开发步骤扩展示例

步骤扩展实现了一个或多个自定义步骤。每个自定义步骤都实施一个用于在连接之间处理集成数据的操作。以下示例演示了开发步骤扩展的替代方案：

- 第 9.3.8.1 节 “使用 XML 片段开发 Camel 路由示例”
- 第 9.3.8.2 节 “使用 RouteBuilder开发 Camel 路由示例”
- 第 9.3.8.3 节 “使用 RouteBuilder 和 Spring Boot 开发 Camel 路由示例”
- 第 9.3.8.4 节 “使用 Camel bean 的示例”

- 第 9.3.8.5 节 “使用 Syndesis Step API 的示例”

Syndesis 提供自定义 Java 注解，您可以将这些注解与 syndesis-extension-plugin 一起使用。当您在 Java 中实现步骤扩展或连接器扩展时，您可以指定启用 Maven 的注解来向扩展定义中添加操作定义。要启用注解处理，请在 Maven 项目中添加以下依赖项：

```
<dependency>
  <groupId>io.syndesis.extension</groupId>
  <artifactId>extension-annotation-processor</artifactId>
  <optional>true</optional>
</dependency>
```

由于 Spring Boot 是集成运行时，因此要将 Bean 注入 Camel 上下文，因此请务必遵循标准的 Spring Boot 实践。例如，创建一个自动配置类，并在那里创建 Bean。但是，默认行为是扩展代码不受到软件包扫描的影响。因此，您必须在步骤扩展中创建并填充 META-INF/spring.factories 文件。

9.3.8.1. 使用 XML 片段开发 Camel 路由示例

要开发自定义步骤，您可以将该操作作为 XML 片段实施，该片段是一个 Camel 路由，具有输入，如 direct。Syndesis 运行时调用此路由的方式与调用任何其他 Camel 路由的方式相同。

例如，假设您想创建一个步骤，该步骤使用可选前缀记录消息的正文。以下 XML 定义了执行此操作的 Camel 路由。

```
<?xml version="1.0" encoding="UTF-8"?>
<routes xmlns="http://camel.apache.org/schema/spring"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
      http://camel.apache.org/schema/spring
      http://camel.apache.org/schema/spring/camel-spring.xsd">

  <route id="log-body-with-prefix">
    <from uri="direct:log"/>
    <choice>
      <when>
        <simple>${header.prefix} != ''</simple>
        <log message="${header.prefix} ${body}"/>
      </when>
      <otherwise>
        <log message="Output ${body}"/>
      </otherwise>
    </choice>
  </route>

</routes>
```

在 XML 中开发扩展时，您必须自行创建扩展定义 JSON 文件。对于此 XML 片段，src/main/resources/META-INF/syndesis/syndesis-extension-definition.json 文件可定义操作，如下所示：

```
{
  "actionType": "step",
  "id": "log-body-with-prefix",
  "name": "Log body with prefix",
  "description": "A simple body log with a prefix",
  "descriptor": {
    "kind": "ENDPOINT", (1)
    "entrypoint": "direct:log", (2)
    "resource": "classpath:log-body-action.xml", (3)
    "inputDataShape": {
      "kind": "none"
    },
    "outputDataShape": {
      "kind": "none"
    },
  "propertyDefinitionSteps": [ {
    "description": "extension-properties",
    "name": "extension-properties",
    "properties": { (4)
      "prefix": {
        "componentProperty": false,
        "deprecated": false,
        "description": "The Log body prefix message",
        "displayName": "Log Prefix",
        "javaType": "String",
        "kind": "parameter",
        "required": false,
        "secret": false,
        "type": "string"
      }
    }
  } ]
  }
}
```

(1) 操作的 type 设置为 ENDPOINT。运行时调用 Camel 端点来执行此自定义步骤提供的操作。

(2) 要调用的 Camel 端点是 direct:log。这是 路由中的 规格。

(3) 这是 XML 片段的位置。

4

这些是此自定义步骤中定义的操作向集成添加此步骤的集成者公开的属性。在 Fuse Online 中，集成者在用户界面中指定的每个值都映射到名称与属性相同的消息标头。在本例中，集成商将看到一个输入字段，其日志 前缀 显示名称。如需了解更多详细信息，请参阅 用户界面属性的描述。

警告

Syndesis 不支持完整的 Camel XML 配置。Syndesis 仅支持 <routes> 标签。

9.3.8.2. 使用 RouteBuilder开发 Camel 路由示例

您可以通过开发一个操作作为 Camel 路由来实现自定义步骤，并支持 RouteBuilder 类。此类路由具有一个输入，如 直接。Syndesis 以其调用任何其他 Camel 路由的方式调用此路由。

要实现示例，它会创建一个步骤，该步骤使用可选前缀记录消息的正文，您可以写类似如下的内容：

```
import org.apache.camel.builder.RouteBuilder;

import io.syndesis.extension.api.annotations.Action;
import io.syndesis.extension.api.annotations.ConfigurationProperty;

@Action( 1
    id = "log-body-with-prefix",
    name = "Log body with prefix",
    description = "A simple body log with a prefix",
    entrypoint = "direct:log")
public class LogAction extends RouteBuilder {
    @ConfigurationProperty( 2
        name = "prefix",
        description = "The Log body prefix message",
        displayName = "Log Prefix",
        type = "string")
    private String prefix;

    @Override
    public void configure() throws Exception {
        from("direct::start") 3
            .choice()
                .when(simple("${header.prefix} != ''"))
                    .log("${header.prefix} ${body}")
                .otherwise()
```

```
                .log("Output ${body}")
            .endChoice();
    }
}
```

1

@Action 注释指示操作定义。

2

@ConfigurationProperty 注释指示用户界面表单控制的定义。详情请查看 用户界面属性的描述。

3

这是操作实施。

此 Java 代码使用 Syndesis 注解，这意味着 extension-maven-plugin 可以自动生成操作定义。在扩展定义 JSON 文件中，操作定义类似如下：

```
{
  "id": "log-body-with-prefix",
  "name": "Log body with prefix",
  "description": "A simple body log with a prefix",
  "descriptor": {
    "kind": "ENDPOINT", 1
    "entrypoint": "direct:log", 2
    "resource": "class:io.syndesis.extension.log.LogAction", 3
    "inputDataShape": {
      "kind": "none"
    },
    "outputDataShape": {
      "kind": "none"
    },
    "propertyDefinitionSteps": [ {
      "description": "extension-properties",
      "name": "extension-properties",
      "properties": { 4
        "prefix": {
          "componentProperty": false,
          "deprecated": false,
          "description": "The Log body prefix message",
          "displayName": "Log Prefix",
          "javaType": "java.lang.String",
          "kind": "parameter",
          "required": false,
          "secret": false,
          "type": "string",
          "raw": false
```

```
        }
      }
    } ]
  },
  "actionType": "step"
}
```

1 操作类型是 ENDPOINT。运行时调用 Camel 端点来执行此步骤实施的操作。

2 这是要调用的 Camel 端点。它是 路由中的 规格。

3 这是实施 RoutesBuilder 的类。

4 这些是此自定义步骤中定义的操作向集成添加此步骤的集成者公开的属性。在 Fuse Online 中，集成者在用户界面中指定的每个值都映射到名称与属性相同的消息标头。在本例中，集成商将看到一个输入字段，其日志 前缀 显示名称。如需更多信息，请参阅 用户界面属性的描述。

9.3.8.3. 使用 RouteBuilder 和 Spring Boot 开发 Camel 路由示例

您可以通过开发一个操作作为 Camel 路由来实现自定义步骤，并支持 RouteBuilder 类和 Spring Boot。在本例中，Spring Boot 是在 Camel 上下文中注册 RouteBuilder 对象的功能。Syndesis 以其调用任何其他 Camel 路由的方式调用此路由。

要实现示例，它会创建一个步骤，该步骤使用可选前缀记录消息的正文，您可以写类似如下的内容：

```
import io.syndesis.extension.api.annotations.Action;
import io.syndesis.extension.api.annotations.ConfigurationProperty;
import org.apache.camel.builder.RouteBuilder;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

@Configuration
public class ActionsConfiguration {

    @Action( 1
        id = "log-body-with-prefix",
        name = "Log body with prefix",
        description = "A simple body log with a prefix",
        entrypoint = "direct:log")
```

```
    @ConfigurationProperty( 2
        name = "prefix",
        description = "The Log body prefix message",
        displayName = "Log Prefix",
        type = "string")
    @Bean 3
    public RouteBuilder logBodyWithprefix() {
        return new RouteBuilder() {
            @Override
            public void configure() throws Exception {
                from("direct::start") 4
                    .choice()
                        .when(simple("${header.prefix} != ""))
                            .log("${header.prefix} ${body}")
                        .otherwise()
                            .log("Output ${body}")
                    .endChoice();
            }
        };
    }
}
```

1

@Action 注释指示操作定义。

2

@ConfigurationProperty 注释指示用户界面表单控制的定义。详情请查看 用户界面属性的描述。

3

将 RouteBuilder 对象注册为 bean。

4

这是操作实施。

此 Java 代码使用 Syndesis 注解，这意味着 extension-maven-plugin 可以自动生成操作定义。在扩展定义 JSON 文件中，操作定义类似如下：

```
{
  "id": "log-body-with-prefix",
  "name": "Log body with prefix",
  "description": "A simple body log with a prefix",
  "descriptor": {
    "kind": "ENDPOINT", 1
    "entrypoint": "direct:log", 2
```

```
  "inputDataShape": {
    "kind": "none"
  },
  "outputDataShape": {
    "kind": "none"
  },
  "propertyDefinitionSteps": [ {
    "description": "extension-properties",
    "name": "extension-properties",
    "properties": {
      "prefix": {
        "componentProperty": false,
        "deprecated": false,
        "description": "The Log body prefix message",
        "displayName": "Log Prefix",
        "javaType": "java.lang.String",
        "kind": "parameter",
        "required": false,
        "secret": false,
        "type": "string",
        "raw": false
      }
    }
  } ]
 },
 "actionType": "step"
}
```

1

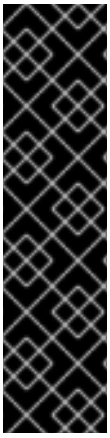
操作类型是 ENDPOINT。运行时调用 Camel 端点来执行此步骤实施的操作。

2

这是要调用的 Camel 端点。它是 路由中的 规格。

3

这些是此自定义步骤中定义的操作向集成添加此步骤的集成者公开的属性。在 Fuse Online 中，集成者在用户界面中指定的每个值都映射到名称与属性相同的消息标头。在本例中，集成商将看到一个输入字段，其日志 前缀 显示名称。如需了解更多详细信息，请参阅 用户界面属性的描述。



重要

要使配置类可以被 Spring Boot 发现，您必须将它们列在名为 META-INF/spring.factories 的文件中，例如：

```
org.springframework.boot.autoconfigure.EnableAutoConfiguration=com.company.ActionsConfiguration
```

使用 Spring Boot 时，您最终在配置类中注册的每个 bean 都可用于 Camel 上下文。详情请参阅 Spring Boot 文档来创建自己的自动配置。

9.3.8.4. 使用 Camel bean 的示例

您可以通过开发一个操作作为 Camel bean 处理器来实现自定义步骤。要实现示例，它会创建一个步骤，该步骤使用可选前缀记录消息的正文，您可以写类似如下的内容：

```
import org.apache.camel.Body;
import org.apache.camel.Handler;
import org.apache.camel.Header;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import io.syndesis.extension.api.annotations.Action;
import io.syndesis.extension.api.annotations.ConfigurationProperty;

@Action(
    id = "log-body-with-prefix",
    name = "Log body with prefix",
    description = "A simple body log with a prefix")
public class LogAction  {
    private static final Logger LOGGER = LoggerFactory.getLogger(LogAction.class);

    @ConfigurationProperty(
        name = "prefix",
        description = "The Log body prefix message",
        displayName = "Log Prefix",
        type = "string")
    private String prefix;

    @Handler 1
    public void process(@Header("prefix") String prefix, @Body Object body) {
        if (prefix == null) {
            LOGGER.info("Output {}", body);
        } else {
            LOGGER.info("{} {}", prefix, body);
```

```
        }
      }
    }
```

1

这是实施该操作的功能。

此 Java 代码使用 Syndesis 注解，这意味着 extension-maven-plugin 可以自动生成操作定义。在扩展定义 JSON 文件中，操作定义类似如下：

```
{
  "id": "log-body-with-prefix",
  "name": "Log body with prefix",
  "description": "A simple body log with a prefix",
  "descriptor": {
    "kind": "BEAN", 1
    "entrypoint": "io.syndesis.extension.log.LogAction::process", 2
    "inputDataShape": {
      "kind": "none"
    },
    "outputDataShape": {
      "kind": "none"
    },
    "propertyDefinitionSteps": [ {
      "description": "extension-properties",
      "name": "extension-properties",
      "properties": {
        "prefix": { 3
          "componentProperty": false,
          "deprecated": false,
          "description": "The Log body prefix message",
          "displayName": "Log Prefix",
          "javaType": "java.lang.String",
          "kind": "parameter",
          "required": false,
          "secret": false,
          "type": "string",
          "raw": false
        }
      }
    } ]
  },
  "actionType": "step"
}
```

1

操作的类型是 EAN。运行时调用 Camel bean 处理器，在这个自定义步骤中执行操作。

2

这是要调用的 Camel bean。

3

这些是此自定义步骤中定义的操作向集成添加此步骤的集成者公开的属性。在 Fuse Online 中，集成者在用户界面中指定的每个值都映射到名称与属性相同的消息标头。在本例中，集成商将看到一个输入字段，其日志 前缀 显示名称。如需了解更多详细信息，请参阅 用户界面属性的描述。

使用 Bean 时，您可能会发现将用户属性注入到 bean 中，而不是从交换标头中检索它们。为此，请实施您要注入的属性的 getter 和 setter 方法。操作实现类似如下：

```
import org.apache.camel.Body;
import org.apache.camel.Handler;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import io.syndesis.extension.api.annotations.Action;
import io.syndesis.extension.api.annotations.ConfigurationProperty;

@Action(
    id = "log-body-with-prefix",
    name = "Log body with prefix",
    description = "A simple body log with a prefix")
public class LogAction  {
    private static final Logger LOGGER = LoggerFactory.getLogger(LogAction.class);

    @ConfigurationProperty(
        name = "prefix",
        description = "The Log body prefix message",
        displayName = "Log Prefix",
        type = "string")
    private String prefix;

    public void setPrefix(String prefix) { 1
        this.prefix = prefix;
    }

    public String getPrefix() { 2
        return prefix;
    }

    @Handler
    public void process(@Body Object body) {
        if (this.prefix == null) {
            LOGGER.info("Output {}", body);
        } else {
            LOGGER.info("{} {}", this.prefix, body);
```

```
        }
    }
}
```

1

这是属性 setter 方法。

2

这是属性 getter 方法。

9.3.8.5. 使用 Syndesis Step API 的示例

您可以使用 Syndesis Step API 实施自定义步骤。这提供了一种与运行时路由创建交互的方法。您可以使用 ProcessorDefinition 类提供的任何方法，您可以创建更复杂的路由。Syndesis API 的信息位于 http://javadoc.io/doc/io.syndesis.extension/extension-api。

以下是使用 Syndesis Step API 实现分割操作的步骤扩展示例：

```
import java.util.Map;
import java.util.Optional;

import io.syndesis.extension.api.Step;
import io.syndesis.extension.api.annotations.Action;
import io.syndesis.extension.api.annotations.ConfigurationProperty;
import org.apache.camel.CamelContext;
import org.apache.camel.model.ProcessorDefinition;
import org.apache.camel.util.ObjectHelper;
import org.apache.camel.Expression;
import org.apache.camel.builder.Builder;
import org.apache.camel.processor.aggregate.AggregationStrategy;
import org.apache.camel.processor.aggregate.UseOriginalAggregationStrategy;
import org.apache.camel.spi.Language;

@Action(id = "split", name = "Split", description = "Split your exchange")
public class SplitAction implements Step {

    @ConfigurationProperty(
        name = "language",
        displayName = "Language",
        description = "The language used for the expression")
    private String language;

    @ConfigurationProperty(
        name = "expression",
        displayName = "Expression",
        description = "The expression used to split the exchange")
    private String expression;
```

```
    public String getLanguage() {
        return language;
    }

    public void setLanguage(String language) {
        this.language = language;
    }

    public String getExpression() {
        return expression;
    }

    public void setExpression(String expression) {
        this.expression = expression;
    }

    @Override
    public Optional<ProcessorDefinition> configure(
            CamelContext context,
            ProcessorDefinition route,
            Map<String, Object> parameters) { 1

        String languageName = language;
        String expressionDefinition = expression;

        if (ObjectHelper.isEmpty(languageName) && ObjectHelper.isEmpty(expressionDefinition))
{
            route = route.split(Builder.body());
        } else if (ObjectHelper.isNotEmpty(expressionDefinition)) {

            if (ObjectHelper.isEmpty(languageName)) {
                languageName = "simple";
            }

            final Language splitLanguage = context.resolveLanguage(languageName);
            final Expression splitExpression =
splitLanguage.createExpression(expressionDefinition);
            final AggregationStrategy aggreationStrategy = new
UseOriginalAggregationStrategy(null, false);

            route = route.split(splitExpression).aggregationStrategy(aggreationStrategy);
        }

        return Optional.of(route);
    }
}
```

1 这是自定义步骤执行的操作的实现。

此 Java 代码使用 Syndesis 注解，这意味着 extension-maven-plugin 可以自动生成操作定义。在扩

展定义 JSON 文件中，操作定义类似如下：

```
{
  "id": "split",
  "name": "Split",
  "description": "Split your exchange",
  "descriptor": {
    "kind": "STEP", (1)
    "entrypoint": "io.syndesis.extension.split.SplitAction", (2)
    "inputDataShape": {
      "kind": "none"
    },
    "outputDataShape": {
      "kind": "none"
    },
    "propertyDefinitionSteps": [ {
      "description": "extension-properties",
      "name": "extension-properties",
      "properties": {
        "language": {
          "componentProperty": false,
          "deprecated": false,
          "description": "The language used for the expression",
          "displayName": "Language",
          "javaType": "java.lang.String",
          "kind": "parameter",
          "required": false,
          "secret": false,
          "type": "string",
          "raw": false
        },
        "expression": {
          "componentProperty": false,
          "deprecated": false,
          "description": "The expression used to split the exchange",
          "displayName": "Expression",
          "javaType": "java.lang.String",
          "kind": "parameter",
          "required": false,
          "secret": false,
          "type": "string",
          "raw": false
        }
      }
    } ]
  },
  "tags": [],
  "actionType": "step"
}
```

(1) 操作的类型是 STEP。

2

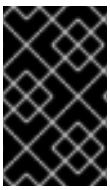
这是实施 Step 接口的类。

其他资源

有关用户界面属性的详情，请参阅 用户界面属性的描述。

9.3.9. 开发连接器扩展示例

如果 Fuse Online 没有提供您要连接到集成的应用程序或服务的连接器，经验丰富的开发人员就可以对向 Fuse Online 贡献新连接器的扩展进行编码。本文档介绍了开发连接器扩展。有关开发连接器的详情，请参阅 Syndesis 社区站点上的开发 Syndesis 连接器。



重要

对于连接器扩展，还没有从 Java 代码自动生成扩展定义 JSON 文件。

连接器本质上是 Camel 组件的代理。连接器配置底层组件，并根据在扩展定义中定义的选项以及 Fuse Online Web 界面收集的用户提供的选项来创建端点。

连接器扩展定义通过以下附加数据结构扩展步骤扩展所需的扩展定义：

- componentScheme

  定义连接器使用的 Camel 组件。您可以为连接器或操作设置 componentScheme。如果您为连接器和操作设置了 componentScheme，则操作的设置具有优先权。

- connectorCustomizers

  指定实现 component ProxyCustomizer 类的类列表。每个类自定义连接器的行为。例如，一个类可能会在应用到底层组件/端点前操作属性，或者类可能会添加 pre/post 端点逻辑。对于每个类，指定实现的完整类名称，例如 com.mycomponent.MyCustomizer。您可以对操作以及连接器设置 connectorCustomizers。根据设置的内容，Fuse Online 首先对连接器应用定制程序，然后用于操作。

- *connectorFactory*

  定义实现 ComponentProxyFactory 类的类，它创建和/或配置底层组件/端点。指定实现的完整类名称。您可以为连接器或操作设置 connectorFactory。操作具有优先权。

Customizer 示例

以下自定义器示例从独立选项设置 DataSource ：

```
public class DataSourceCustomizer implements ComponentProxyCustomizer,
CamelContextAware {
    private final static Logger LOGGER =
LoggerFactory.getLogger(DataSourceCustomizer.class);

    private CamelContext camelContext;

    @Override
    public void setCamelContext(CamelContext camelContext) { (1)
        this.camelContext = camelContext;
    }

    @Override
    public CamelContext getCamelContext() { (2)
        return this.camelContext;
    }

    @Override
    public void customize(ComponentProxyComponent component, Map<String, Object>
options) {
        if (!options.containsKey("dataSource")) {
            if (options.containsKey("user") && options.containsKey("password") &&
options.containsKey("url")) {
                try {
                    BasicDataSource ds = new BasicDataSource();

                    consumeOption(camelContext, options, "user", String.class, ds::setUsername);
(3)
                    consumeOption(camelContext, options, "password", String.class,
ds::setPassword); (4)
                    consumeOption(camelContext, options, "url", String.class, ds::setUrl); (5)

                    options.put("dataSource", ds);
                } catch (@SuppressWarnings("PMD.AvoidCatchingGenericException") Exception e)
{
                    throw new IllegalArgumentException(e);
                }
            } else {
                LOGGER.debug("Not enough information provided to set-up the DataSource");
            }
```

```
        }
    }
}
```

1 2

通过实施 CamelContextAware，Syndesis 注入 Camel 上下文，然后调用自定义方法。

3 4 5

进程选项，然后将它们从选项映射中删除。

注入属性的示例

如果自定义器遵循 Java bean 惯例，您也可以注入属性，如上例的修订版本所示：

```
public class DataSourceCustomizer implements ComponentProxyCustomizer,
CamelContextAware {
    private final static Logger LOGGER =
LoggerFactory.getLogger(DataSourceCustomizer.class);

    private CamelContext camelContext;
    private String userName;
    private String password;
    private String url;

    @Override
    public void setCamelContext(CamelContext camelContext) { 1
        this.camelContext = camelContext;
    }

    @Override
    public CamelContext getCamelContext() { 2
        return this.camelContext;
    }

    public void setUserName(String userName) { 3
      this.userName = userName;
    }

    public String getUserName() { 4
      return this.userName;
    }

    public void setPassword(String password) { 5
      this.password = password;
    }

    public String getPassword() { 6
      return this.password;
```

```
    }

    public void setUrl(String url) { 7
      this.url = url;
    }

    public String getUrl() { 8
      return this.url;
    }

    @Override
    public void customize(ComponentProxyComponent component, Map<String, Object>
options) {
        if (!options.containsKey("dataSource")) {
            if (userName != null && password != null && url != null) {
                try {
                    BasicDataSource ds = new BasicDataSource();
                    ds.setUserName(userName);
                    ds.setPassword(password);
                    ds.setUrl(url);

                    options.put("dataSource", ds);
                } catch (@SuppressWarnings("PMD.AvoidCatchingGenericException") Exception e)
{
                    throw new IllegalArgumentException(e);
                }
            } else {
                LOGGER.debug("Not enough information provided to set-up the DataSource");
            }
        }
    }
}
```

1 2 3

通过实施 CamelContextAware，Syndesis 注入 Camel 上下文，然后调用自定义方法。这个示例代码会覆盖 setCamelContext （） 和 getCamelContext （） 方法，并设置用户名。

4 5 6 7 8

示例代码处理注入的选项，并从选项映射中自动删除它们。

使用自定义器配置 before/after 逻辑

您可以使用自定义器配置 before/after 逻辑，如下例所示：

```
public class AWSS3DeleteObjectCustomizer implements ComponentProxyCustomizer {
  private String filenameKey;

  public void setFilenameKey(String filenameKey) {
    this.filenameKey = filenameKey;
```

```
    }

    public String getFilenameKey() {
        return this.filenameKey;
    }

    @Override
    public void customize(ComponentProxyComponent component, Map<String, Object>
options) {
        component.setBeforeProducer(this::beforeProducer);
    }

    public void beforeProducer(final Exchange exchange) throws IOException {
        exchange.getIn().setHeader(S3Constants.S3_OPERATION, S3Operations.deleteObject);

        if (filenameKey != null) {
            exchange.getIn().setHeader(S3Constants.KEY, filenameKey);
        }
    }
}
```

自定义 component ProxyComponent的行为

ComponentProxyFactory 类创建和/或配置底层组件/端点。要自定义 component ProxyComponent 对象 的行为，您可以使用以下方法覆盖任何方法：

- createDelegateComponent()

  当代理启动时，Syndesis 调用此方法，并最终使用 componentScheme 选项定义的方案来创建专用组件实例。

  这个方法的默认行为是确定任何连接器/操作选项是否适用于组件级别。只有在端点中无法应用同一选项时，该方法才会创建一个自定义组件实例，并根据适用的选项进行配置。

- configureDelegateComponent()

  只有在创建了自定义组件实例来配置委派的组件实例的额外行为时，才会调用此方法。

- createDelegateEndpoint()

  当代理创建端点并且默认使用 Camel 目录设施创建端点时，Syndesis 调用此方法。

- *configureDelegateEndpoint()*

  创建委派的端点后，Syndesis 调用此方法来配置委派的端点实例的额外行为，例如：

```
public class IrcComponentProxyFactory implements ComponentProxyFactory {

   @Override
   public ComponentProxyComponent newInstance(String componentId, String
componentScheme) {
      return new ComponentProxyComponent(componentId, componentScheme) {
         @Override
         protected void configureDelegateEndpoint(ComponentDefinition definition,
Endpoint endpoint, Map<String, Object> options) throws Exception {
            if (!(endpoint instanceof IrcEndpoint)) {
               throw new IllegalStateException("Endpoint should be of type
IrcEndpoint");
            }

            final IrcEndpoint ircEndpoint = (IrcEndpoint)endpoint;
            final String channels = (String)options.remove("channels");

            if (ObjectHelper.isNotEmpty(channels)) {
               ircEndpoint.getConfiguration().setChannel(
                  Arrays.asList(channels.split(","))
               );
            }
         }
      };
   }
}
```

9.3.10. 如何开发库扩展

库扩展提供集成在运行时需要的资源。库扩展不向 Fuse Online 贡献步骤或连接器。

当您保存集成时，您可以选择一个或多个要与集成中包含的导入的库扩展。

库扩展不定义任何操作。以下是库扩展的示例定义：

```
{
  "schemaVersion" : "v1",
  "name" : "Example Library Extension",
  "description" : "Syndesis Extension for adding a runtime library",
  "extensionId" : "io.syndesis.extensions:syndesis-library",
  "version" : "1.0.0",
```

```
  "tags" : [ "my-libraries-extension" ],
  "extensionType" : "Libraries"
}
```

另请参阅示例库扩展 ：https://github.com/syndesisio/syndesis-extensions

除了缺少操作外，库扩展的结构与步骤或连接器扩展的结构相同。

在创建库扩展的 Maven 项目中，若要添加 Maven 存储库不可用的依赖项，请指定系统依赖项，例如：

```
<dependency>
  <groupId>com.company</groupId>
  <artifactId>my-library-extension</artifactId>
  <version>1.0</version>
  <scope>system</scope>
  <systemPath>${project.basedir}/lib/my-library-extension.jar</systemPath>
</dependency>
```

9.3.11. 创建 JDBC 驱动程序库扩展

要连接到 Apache Derby、MySQL 和 PostgreSQL 以外的 SQL 数据库，您可以创建一个库扩展，用于打包您要连接的数据库的 JDBC 驱动程序。将此扩展上传到 Fuse Online 后，Fuse Online 提供的数据库连接器可以访问驱动程序来验证和创建与专有数据库的连接。您不会为特定数据库创建新连接器。

Syndesis 开源社区提供了一个项目，用于创建打包 JDBC 驱动程序的扩展。

仅在扩展中打包一个驱动程序。这样可以更轻松地管理扩展，作为管理特定数据库的一部分。但是，您可以创建一个库扩展来打包多个驱动程序。

先决条件

- 要使用 Syndesis 项目，您必须有一个 GitHub 帐户。

流程

1. 通过执行以下操作之一来确保访问您要连接的数据库的 JDBC 驱动程序：

a. 确认驱动程序位于 Maven 存储库中。

b. 下载驱动程序。

2. 在浏览器标签页中，进入 https://github.com/syndesisio/syndesis-extensions

3. 将 syndesis-extensions 存储库派生到您的 GitHub 帐户。

4. 从 fork 创建本地克隆。

5. 在 syndesis-extensions clone 中：

a. 如果驱动程序不在 Maven 存储库中，请将驱动程序复制到 syndesis-library-jdbc-driver/lib 文件夹中。

b. 编辑 syndesis-library-jdbc-driver/pom.xml 文件：

i. 将 Name 元素的值更新为您为这个扩展选择的名称。

ii. 更新 Description 元素的值，以提供有关此扩展的有用信息。

iii. 如果您将驱动程序复制到 syndesis-library-jdbc-driver/lib 中，请确保 pom.xml 中的 systemPath 指向该驱动程序文件。（可选）更改 groupId、artifactId 和 version，以根据驱动程序反映正确的值。

iv. 如果驱动程序位于 Maven 存储库中，请确保对 Maven 依赖项的引用位于 pom.xml 文件中。

v. 检查 pom.xml 文件的其余部分，并根据需要更改所有相关元数据。

c.

执行 ./mvnw -pl :syndesis-library-jdbc-driver clean 软件包 以构建扩展。

生成的 .jar 文件位于 syndesis-library-jdbc-driver/target 文件夹中。将此 .jar 文件作为 Fuse Online 中的扩展导入。

导入库扩展后，当您在 Fuse Online 中保存集成时，您可以选择导入的库扩展并将其与集成相关联。

9.4. 添加并管理扩展

扩展可让您在 Fuse Online 中添加自定义，以便您可以以您想要的方式集成应用程序。使用扩展中提供的自定义后，您可以识别使用这些自定义的集成。这在更新或删除扩展前可以正常工作。

以下主题提供详情：

- 第 9.4.1 节 “使自定义功能可用”
- 第 9.4.2 节 “识别使用扩展的集成”
- 第 9.4.3 节 “更新扩展”
- 第 9.4.4 节 “删除扩展”

9.4.1. 使自定义功能可用

要使自定义功能可用于集成，请将扩展上传到 Fuse Online。

先决条件

- 开发人员提供了一个包含 Fuse Online 扩展的 .jar 文件。

流程

1. 在左侧 Fuse Online 面板中，点 Customizations > Extensions。

2. 点 Import Extension。

3. 拖放或选择包含您要上传的扩展名的 .jar 文件。

   Fuse Online 立即尝试验证文件是否包含扩展。如果出现问题，Fuse Online 会显示关于该错误的消息。您必须与扩展开发人员协调，以获取更新的 .jar 文件，然后您可以尝试上传该文件。

4. 查看扩展详情。

   在 Fuse Online 验证文件后，它会提取并显示扩展名的名称、ID、描述和类型。该类型指示扩展是否定义了自定义连接器，还是在连接之间运行一个或多个自定义步骤，还是运行时库扩展（包括 JDBC 驱动程序）。

   对于连接器扩展，Fuse Online 显示从此自定义连接器创建的连接可用的操作。在扩展中，开发人员可能提供了一个图标，Fuse Online 可以用来代表从此连接器创建的应用程序连接。虽然您在扩展详情页面中没有看到此图标，但在从自定义连接器创建连接时会出现它。如果扩展开发人员没有在扩展中提供图标，则 Fuse Online 会生成图标。

   对于步骤扩展，Fuse Online 显示扩展所定义的每个自定义步骤的名称。

   对于库扩展，Fuse Online 在集成运行时类路径中包含导入的 Maven 依赖项。您必须确保导入的 Maven 依赖项不会与集成中使用的其他依赖项冲突（包括任何其他库扩展，如 JDBC 驱动程序）。

5. 点 Import Extension。Fuse Online 使自定义连接器或自定义步骤可用，并显示扩展的详情页面。

其他资源

- 从自定义连接器创建连接。

- 添加自定义步骤
- 连接到专有数据库

9.4.2. 识别使用扩展的集成

在更新或删除扩展前，您应该识别使用该扩展提供的自定义集成。

流程

1. 在左侧 Fuse Online 面板中，点 Customizations > Extensions。
2. 在扩展列表中，找到您要更新或删除的扩展条目，然后点其 Details 按钮。

结果

Fuse Online 显示有关扩展的详细信息，包括使用扩展提供的自定义的任何集成列表。

9.4.3. 更新扩展

当开发人员更新扩展时，您可以上传更新的 .jar 文件，以在集成中实施更新。

前提条件

开发人员为您提供了一个更新的 .jar 文件，用于之前上传的扩展。

流程

1. 在 Fuse Online 中，在左侧面板中点 Customizations > Extensions。
2. 在您要更新的扩展的条目右侧，点 Update。
3. 单击点行框以导航到并选择更新的 .jar 文件，然后单击 Open。

4. 确认扩展详情正确，然后单击 **Import Extension**。

5. 在更新扩展的详情页面中，决定使用扩展中定义的连接器或自定义步骤。

从更新的扩展中，您可以准确了解更新每个使用自定义连接器或自定义步骤的集成所需的内容。您至少必须重新发布使用更新扩展中定义的自定义的每个集成。

在某些情况下，您可能需要编辑集成以更改或添加自定义配置详情。您必须与扩展开发人员通信，了解如何更新集成。

9.4.4. 删除扩展

即使正在运行的集成使用了该扩展提供的步骤，您也可以删除扩展，或使用从该扩展提供的连接器创建的连接。删除扩展后，您无法启动使用该扩展提供的自定义的集成。

流程

1. 在左侧 **Fuse Online** 面板中，点 **Customizations > Extensions**。

2. 在扩展列表中，找到您要删除的扩展的条目，并点击 **Delete**，它出现在条目右侧。

结果

可能已停止或草案集成，该集成使用您删除的扩展提供的自定义。要运行其中一个集成，您需要编辑集成来删除自定义。

其他资源

- 识别扩展使用
- 更新扩展