Red Hat Fuse 7.5

设计 API

在 OpenShift 上为 Fuse 应用程序设计 REST API

Last Updated: 2023-10-22

Red Hat Fuse 7.5 设计 API

在 OpenShift 上为 Fuse 应用程序设计 REST API

法律通告

摘要

使用基于 Web 的 REST API Designer 的指南

目录

第 1 章 概述

Red Hat Fuse on OpenShift 提供了 API Designer，它是一个基于 Web 的 API 编辑器，可用于设计符合 OpenAPI 2.0 规格的 REST API，这是 API 服务的厂商中立且可移植的开放描述格式。API Designer 是 Apicurio Studio 开源项目的"亮点"版本(https://www.apicur.io/)。这意味着您的 API Designer 会话是无状态的，您必须在每个会话末尾将 API 定义保存为 JSON 文件。

您还可以使用 API Designer 根据 REST API 定义生成初始 Fuse 项目。在 Fuse 开发环境中，您可以完成项目的 Camel 路由并构建项目。最后，您可以在 OpenShift 上的 Fuse 上部署生成的 REST 服务。

以下是如何使用 API Designer 在 OpenShift 应用程序解决方案中的 Fuse 中纳入 REST API 的概述：

1. 添加 API Designer 作为服务到 OpenShift 项目。
2. 在 API Designer 中：
   - 使用 API Designer 创建 API 定义。将 REST API 定义保存为 JSON 文件到本地文件系统。您可以在编辑会话的任意点保存 API 定义，即使 API 定义没有完成。
   - 将 API 定义上传到 API Designer。
   - 根据当前的 REST API 定义生成 Fuse Camel 项目。API Designer 提供了一个可下载 zip 文件，其中包含完整的 Maven 项目。
3. 在 Fuse 开发环境中，完成由生成的 Fuse 项目提供的框架实施。
4. 构建 Fuse 应用并将其部署到 OpenShift。
5. （可选）将 Fuse 应用程序与红帽 3scale API 管理集成，使用 3scale 服务发现功能查找并配置 Fuse 应用程序。

1.1. 将 API DESIGNER 作为服务添加到 OPENSHIFT 项目中

您可以从 OpenShift 服务目录将 API Designer 作为服务添加到 OpenShift 项目中。您可以通过 OpenShift 环境外部的 URL 访问此实例。

先决条件

- 根据 OpenShift 系统管理员的建议准则，获取允许您访问 API 设计器的主机名。
- 在命令窗口中运行以下命令来验证 OpenShift 镜像和模板上的 Fuse （包括 **apidesigner-ui** 和 **fuse-apidesigner-generator** ）是否安装在 OpenShift 集群上：

  ```
  oc get is -n openshift
  ```

  如果没有预安装镜像和模板，或者所提供的版本已过时，安装（或更新）OpenShift 镜像和模板上的 Fuse，如 OpenShift 指南中的 Fuse 所述。

流程

将 API Designer 作为一个服务添加到 OpenShift 项目中：

1. 在命令窗口中登录到 OpenShift 服务器：

   ```
   oc login -u developer -p developer
   ```

2. 创建新项目命名空间。例如，以下命令创建一个名为 **test** 的新项目：

```
oc new-project test
```

3. 在 Web 浏览器中，打开 OpenShift 控制台并使用您的凭据（例如，用户名 **developer** 和密码 **developer**）登录。

4. 单击 **Catalog**。在 Catalog 搜索字段中，键入 **API Designer**，然后选择 **Red Hat Fuse API Designer**。
Red Hat Fuse API Designer 向导 **的信息** 步骤将打开。

5. 点击 **Next**。
Red Hat Fuse API Designer 向导的 **Configuration** 步骤将打开。

6. 在 **Image Stream Namespace** 字段中，键入 **openshift**。

7. 在 **ROUTE_HOSTNAME** 字段中，键入允许您访问 API Designer 实例的外部主机名，如 **apidesigner-myproject.192.168.64.43.nip.io**。

8. 接受 **Configuration** 步骤中其余设置的默认值，再单击 **Create**。
模板向导的 **Results** 步骤将打开。

9. 单击 **Close**。

10. 在 OpenShift Web 控制台中，在 **My Projects** 窗格中，选择项目，例如，选择 **test**。
项目的 **Overview** 选项卡将打开，显示 **apidesigner-ui** 应用。

11. 单击 **apidesigner-ui** 部署左侧的箭头，以展开并查看部署详情。

12. 点 API Designer 实例的链接，例如 **https://apidesigner-myproject.192.168.64.43.nip.io**。
API Designer 在新的 Web 浏览器窗口或标签页中打开。

注意

如果您无法打开 API Designer 实例，您可能需要编辑计算机的 **/etc/hosts** 文件来添加 ROUTE_HOSTNAME，其中 **$OPENSHIFT_IP_ADDR** 是 OpenShift 服务器和 **apidesigner.my-minishift.apicurio.io** 的 IP 地址。

```
$OPENSHIFT_IP_ADDR apidesigner.my-minishift.apicurio.io
```

第 2 章 使用 API DESIGNER 设计和开发 API 定义

您可以使用 API Designer 设计和开发符合 OpenAPI 2.0 规格的 REST API 定义。

先决条件

- 您创建了 OpenShift 项目。
- 您已将 API Designer 服务添加到 OpenShift 项目中。

2.1. 创建 REST API 定义

下列步骤描述了如何创建 REST API 定义。

注意

- 您可以从 OpenShift 上的 Fuse Online 或 Fuse 访问 API Designer 用户界面。
- 对于 OpenShift 上的 Fuse，API Designer 是无状态的，这意味着它不会在 OpenShift 会话之间保存您的工作。您需要在会话之间将 API 保存到本地文件系统中。

关于示例

Task Management API 示例模拟一个简单的 API，而销售顾问可能会用来跟踪在与客户联系交互时所需的任务。"to-do"任务示例可能是"为新联系人创建帐户"或"现有联系人的订单"。要实现 Task Management API 示例，您需要创建两个路径 - 一个用于任务，另一个用于特定任务。然后，您可以定义操作来创建任务，检索所有任务或特定任务，更新任务，以及删除任务。

先决条件

- 您知道您要创建的 API 的端点。对于任务管理 API 示例，有两个端点：/ setuptools 和 **/setuptools/{id}**。
- 对于 OpenShift 上的 Fuse，您只创建了 OpenShift 项目，并将 API Designer 服务添加到 OpenShift 项目中。

流程

1. 如果您使用 Fuse Online，请跳至第 2 步。
   如果您在 OpenShift 中使用 Fuse：
   a. 登录您的 OpenShift Web 控制台，然后打开包含 API Designer 的项目。
   b. 在应用程序列表中，点 API Designer 的 URL，例如 **https://apidesigner-myproject.192.168.64.43.nip.io**
   为 API Designer 打开一个新的浏览器窗口或标签页。

   注意

   因为 API Designer 是 Apicurio Studio 开源项目的 "亮点"版本，所以 "Apicurio" 显示在 API Designer 界面中。

2. 单击 **New API**。这会打开新的 API 页面。

3. 更改 API 名称：

   a. 将光标悬停在名称上，然后点显示的编辑图标()。

   b. 编辑名称。例如，键入 **Task API**。

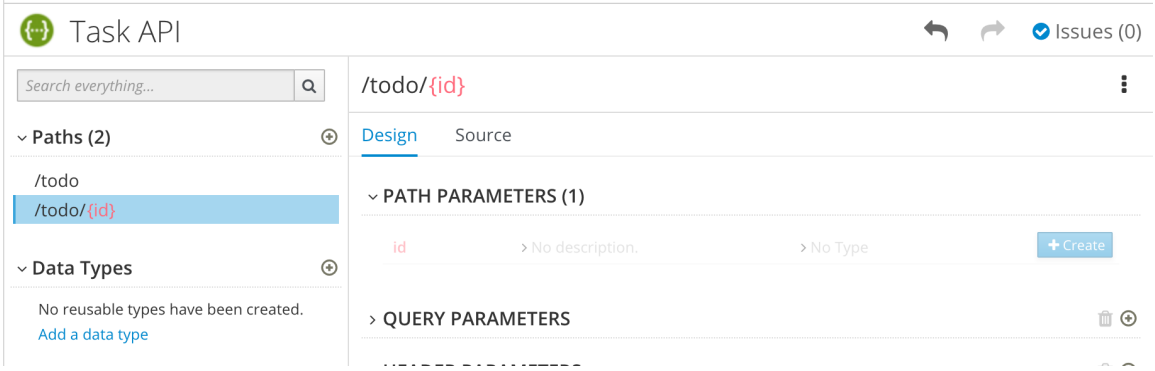
   c. 确认名称更改。

4. （可选）：

   - 添加您的联系信息（姓名、电子邮件地址和 URL）。
   - 选择许可证。
   - 定义标签。
   - 定义安全方案。
   - 指定安全要求。

5. 定义 API 的每个端点的相对路径。字段名称必须以斜杠(/)开头。
   对于 Task Management API 示例，创建两个路径：

   - 任务的路径： **/setuptools**
   - 一个特定任务的路径，按 ID: / **covers/{id}**



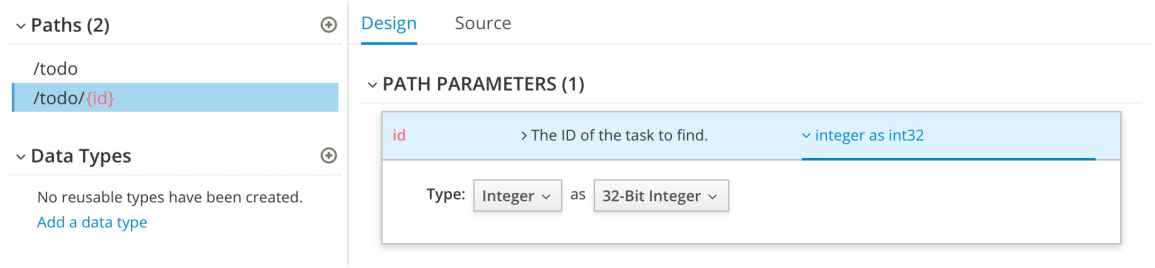
6. 指定任何路径参数的类型。
   对于示例 **id** 参数：

   a. 在 **Paths** 列表中，单击 **/cleanup/{id}**。
   id 参数会出现在 **PATH PARAMETERS** 部分中。

   b. 点 **Create**。

   c. 对于描述，type： **要查找的任务 ID。**

   d. 对于类型，将 **整数** 选为 **32 位整数。**



7. 在 **Data Types** 部分中，为 API 定义可重复使用的类型。

   a. 单击 **Add a data type**。

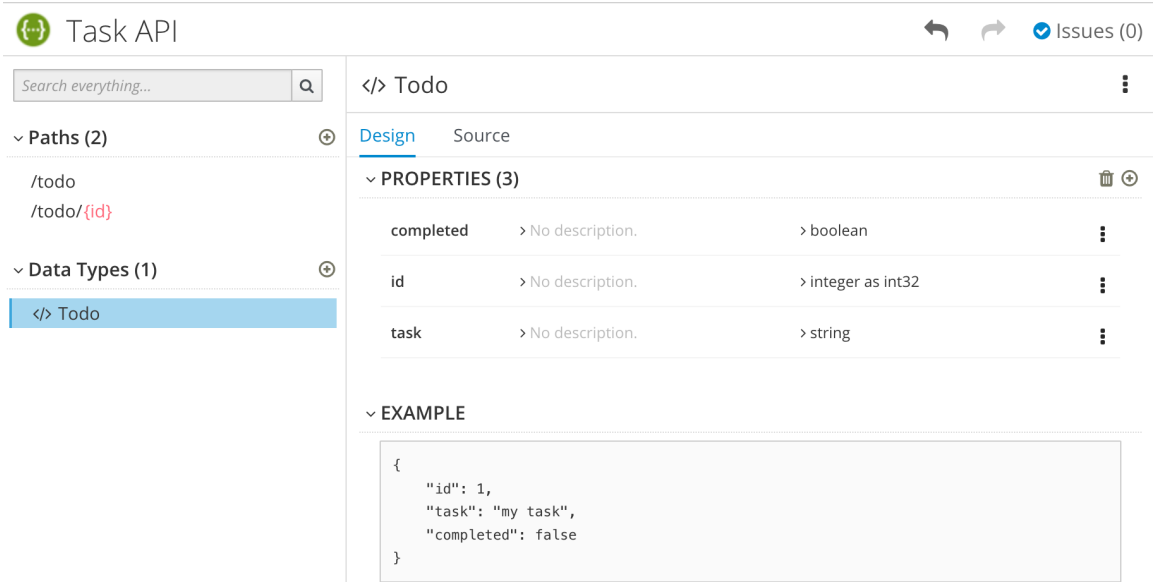
   b. 在 **Add Data Type** 对话框中输入名称。对于 Task Management API 示例，键入 **Todo**。

   c. 另外，您可以提供一个示例，API Designer 创建数据类型的 schema。然后，您可以编辑生成的模式。
   对于 Task Management API 示例，从以下 JSON 示例开始：

```
{
    "id": 1,
    "task": "my task",
    "completed": false
}
```

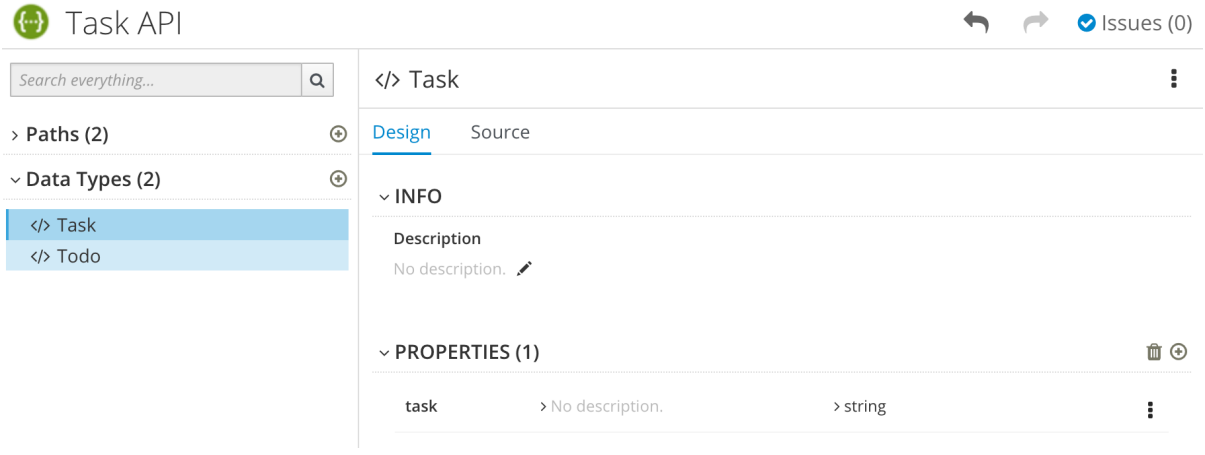
   d. 另外，您可以选择使用数据类型创建 REST 资源。

   e. 点 **Save**。如果您提供了示例，API Designer 从示例中生成一个模式：



8. 另外，您可以添加编辑模式属性并添加新模式属性。

9. 对于 Task Management API 示例，创建另一个名为 **Task** 的数据类型，其名为 **string 的任务**。



10. 对于每个路径，定义操作(GET、PUT、POST、DELETE、OPTIONS、HEAD 或 PATCH)。
    对于 Task Management API 示例，请按照下表所述定义操作：

表 2.1. 任务管理 API 操作

路径	操作	描述	Request（请求）	响应
/todo	POST	创建新任务。	request Body type: **Task**	状态代码： **201** Description: **Task created** 响应正文： **Todo** 类型
/todo	GET	获取所有任务。	*Not applicable*	• 状态代码： **200** 描述： **任务已删除** • 状态代码： **400** Description: **Task not deleted**
/todo/{id}	GET	按 ID 获取任务。	*Not applicable*	状态代码： **200** 描述 ： **为 ID 找到的任务** 响应正文： **Todo** 类型
/todo/{id}	UPDATE （更新）	按 ID 更新任务。	request Body type: **Task**	• 状态代码： **200** 描述： **完成** • 状态代码： **400** Description: **Task not updated**
/todo/{id}	DELETE	按 ID 删除任务。	*Not applicable*	• 状态代码： **200** 描述： **任务已删除** • 状态代码： **400** Description: **Task not deleted**

11. 解决任何问题，如 第 2.2 节 “解决 API 设计程序中的问题” 所述。

12. 对于 OpenShift 上的 Fuse，点 **Save As**，然后选择 **JSON** 或 **YAML** 格式来保存您的 API 规格。JSON 或 YAML 文件下载到您的本地下载文件夹中。默认文件名为 **openapi-spec**，带有适当的文件扩展名。

其他资源

- 有关 OpenAPI 2.0 规格的详情，请参考：https://github.com/OAI/OpenAPI-Specification/blob/master/versions/2.0.md

2.2. 解决 API 设计程序中的问题

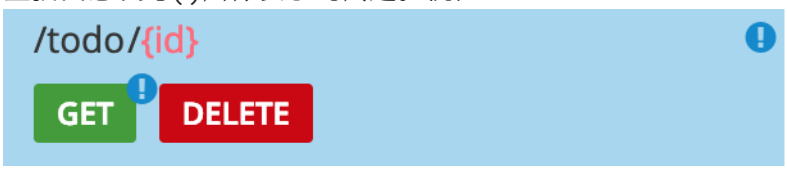
创建并编辑 API 时，API Designer 会识别您必须使用感叹号(!)图标解决的问题。

先决条件

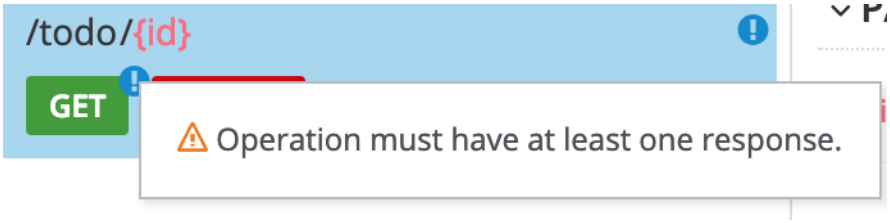
- 在 API Designer 中打开 API。

流程

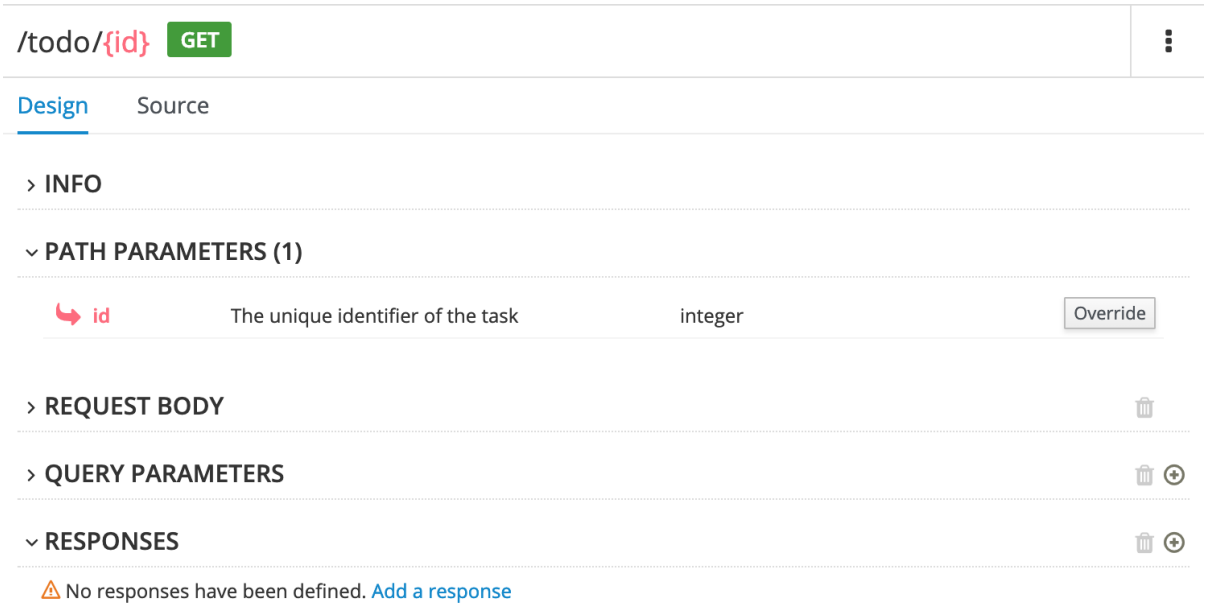
1. 查找由感叹号(!)图标表示的问题。例如：



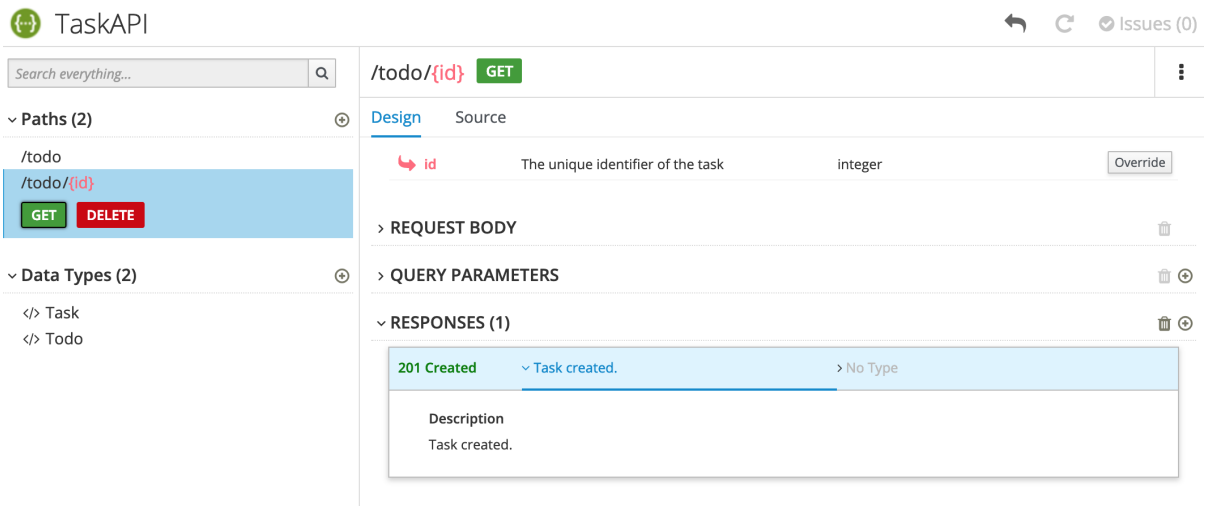
2. 点感叹号图标查看问题的描述。例如：



3. 根据问题描述提供的信息，导航到问题的位置并修复它。
例如，要修复"Operation 必须至少有一个响应"问题，请单击 **GET** 操作来打开它，然后单击 **Add a response**。



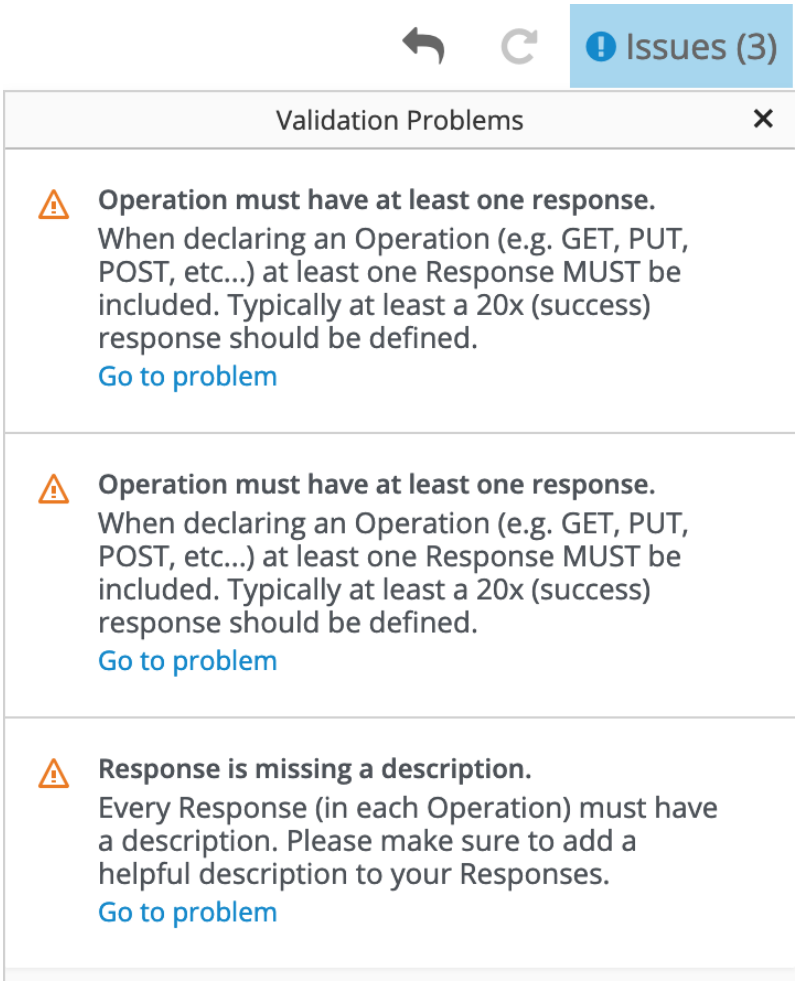
输入响应的描述后，问题会被解决，感叹号图标会消失：



4. 有关所有问题的总结：

   a. 单击右上角的 **issues** 链接。

   b. 点 **Go 解决特定问题** 以进入问题的位置，以便您可以解决这个问题。


Issues (3)
Validation Problems
Operation must have at least one response.
When declaring an Operation (e.g. GET, PUT, POST, etc...) at least one Response MUST be included. Typically at least a 20x (success) response should be defined.
Go to problem
Operation must have at least one response.
When declaring an Operation (e.g. GET, PUT, POST, etc...) at least one Response MUST be included. Typically at least a 20x (success) response should be defined.
Go to problem
Response is missing a description.
Every Response (in each Operation) must have a description. Please make sure to add a helpful description to your Responses.
Go to problem

第 3 章 根据 REST API 实施、构建和部署 FUSE 应用程序

您可以使用 Red Hat Fuse API Designer 根据 REST API 定义生成 Camel Fuse 项目。在 Fuse 开发环境中，您可以完成 Camel 路由和 Rest DSL API。最后，您可以构建项目，并将生成的应用部署到 OpenShift 上的 Fuse。

先决条件

- 您有一个现有的 API 定义，它符合 OpenAPI 2.0 规格。例如，使用 API Designer 创建的 **openapi-spec.json** 文件。
- API Designer 在本地 OpenShift 集群上安装并运行。
- 您有一个现有的 OpenShift 项目，其 API Designer 添加为服务。
- 您已安装了 Maven 和 Red Hat Fuse。

以下主题描述了如何根据 REST API 实施、构建和部署 Fuse 应用程序：

- 第 3.1 节 “将 API 定义上传到 API Designer”
- 第 3.2 节 “从 API Designer 生成 Fuse Camel 项目”
- 第 3.3 节 “完成 API Designer 生成的 Camel 项目”
- 第 3.4 节 “构建和部署 REST 服务”

3.1. 将 API 定义上传到 API DESIGNER

您可以将现有的 API 定义上传到 API Designer。

先决条件

- 您有一个现有的 API 定义，它符合 OpenAPI 2.0 规格。例如，使用 API Designer 创建的 **openapi.json** 文件。
- API Designer 在本地 OpenShift 集群上安装并运行。
- 您有一个现有的 OpenShift 项目，其 API Designer 添加为应用。

流程

1. 在 OpenShift Web 控制台中，打开包含 API Designer 的项目。
2. 打开 API Designer 控制台。在项目的应用列表中，单击 apidesigner 下的 URL。例如：
   https://apidesigner-myproject.192.168.64.38.nip.io
   API Designer 控制台在一个单独的 Web 浏览器标签页或窗口中打开。
3. 单击 **Open API**。
   此时会打开文件管理器窗口。
4. 在文件管理器窗口中：
   a. 导航到包含现有 OpenAPI 定义文件的文件夹，如 **openapi.json**。
   b. 选择 OpenAPI 定义文件，然后点 **Open**。

OpenAPI 定义在 API Designer 控制台中打开。

3.2. 从 API DESIGNER 生成 FUSE CAMEL 项目

您可以使用 API Designer 根据 API 定义生成 Fuse Camel 项目。

先决条件

- API Designer 在本地 OpenShift 集群上安装并运行。
- 您有一个现有的 OpenShift 项目，其 API Designer 添加为应用。
- 您已在 API Designer 控制台中创建或打开了 API 定义文件。

流程

在 API Designer 控制台中：

1. 点 **Generate**。
2. 从下拉列表中选择 **Fuse Camel Project**。

API Designer 生成 **camel-project.zip** 文件，并将其下载到您的本地默认下载文件夹。

zip 文件包含 Fuse Camel 项目，该项目使用 Camel 的 Rest DSL 提供 API 定义的默认框架实施，并包含所有资源操作。该项目还包括您用于生成项目的原始 OpenAPI 定义文件。

3.3. 完成 API DESIGNER 生成的 CAMEL 项目

API Designer 生成 Fuse 项目，该项目使用 Camel 的 Rest DSL 提供 API 定义的默认框架实现，并涵盖所有资源操作。在 Fuse 开发环境中，您将完成该项目。

先决条件

- 您有一个由 API Designer 生成的 **camel-project.zip** 文件。
- （可选）您已安装了带有 Fuse 工具的 Red Hat Developer Studio。

流程

1. 将 API Designer-generated **camel-project.zip** 文件解压缩到临时文件夹。
2. 打开 Red Hat Developer Studio。
3. 在 Developer Studio 中，选择 **File → Import**。
4. 在 **Import** 对话框中，选择 **Maven → Existing Maven Projects**。
5. 在编辑器视图中打开项目的 **camel-context.xml** 文件。
6. 单击 **REST** 选项卡，以编辑 Rest DSL 组件。
   有关定义 REST 服务的详情，请参考 Apache Camel 开发指南中的 "定义 REST 服务"部分。

   有关使用 Swagger 支持扩展 JAX-RS 端点的详情，请参考 Apache CXF 开发指南。

有关使用 Fuse 工具 REST 编辑器的详情，请参考工具 用户指南中的 "查看和编辑 Rest DSL 组件"部分。

7. 在 Design 选项卡中，编辑 Camel 路由。
有关编辑 Camel 路由的详情，请参考 工具用户指南中的"编辑路由上下文"部分。

3.4. 构建和部署 REST 服务

完成 Fuse 项目后，您可以在 OpenShift 中构建和部署项目。

先决条件

- 您有一个定义 REST 服务的完整、无 error-free Fuse 项目。
- 已安装 Java 8 JDK （或更新版本）和 Maven 3.3.x （或更新版本）。

流程

如果您有一个单节点 OpenShift 集群，如 Minishift 或 Red Hat Container Development Kit，您可以在其中部署您的项目。http://appdev.openshift.io/docs/minishift-installation.html

将此项目部署到正在运行的单节点 OpenShift 集群：

1. 登录到您的 OpenShift 集群：

```
$ oc login -u developer -p developer
```

2. 为项目创建新的 OpenShift 项目。例如，以下命令会创建一个名为 **test-deploy** 的新项目。

```
$ oc new-project test-deploy
```

3. 将目录更改为包含 Fuse Camel 项目的文件夹（例如，**myworkspace/camel-project**）：

```
$ cd myworkspace/camel-project
```

4. 构建和部署项目到 OpenShift 集群：

```
$ mvn clean fabric8:deploy -Popenshift
```

5. 在您的浏览器中，打开 OpenShift 控制台并导航到项目（如 **test-deploy**）。等待您看到 **camel-project** 应用的 pod 已启动。

6. 在项目的 **Overview** 页面中，找到 **camel-project** 应用的 URL。URL 使用此形式：
http://camel-project-MY_PROJECT_NAME.OPENSHIFT_IP_ADDR.nip.io。

7. 点 URL 访问该服务。

第 4 章 为 3SCALE 发现准备 API 服务

Red Hat 3scale API Management 是红帽提供的，可让您规范对公共互联网上的 API 服务的访问。3scale 的功能包括执行服务级别协议(SLA)的功能，管理 API 版本，提供安全性和身份验证服务等。Fuse 支持 3scale 的服务发现功能，这有助于从 3scale 管理门户 UI 发现 Fuse 服务。*使用服务发现，您可以扫描在同一 OpenShift 集群中运行的 Fuse 应用程序，并将关联的 API 定义自动导入到 3scale 中。*

先决条件

- *在 OpenShift 中部署和运行 API 服务的 Fuse 应用。*
- *Fuse 应用程序标注了必要的注解，以使 3scale 能够发现它。*

注意

由 API Designer 生成的 Fuse 项目预先配置为自动提供必要注解。

对于不是由 API Designer 生成的 Fuse 项目，您必须配置您的项目，如 第 4.1 节 "为不是由 API Designer 生成的 Fuse 项目添加注解" 所述。

- *3scale API 管理系统部署在与要发现的 API 服务 相同的 OpenShift 集群中。*

有关在 3scale 中发现 API 服务的步骤的详情，请查看 Red Hat 3scale API 管理门户指南中的 服务发现部分。

其他资源

- *Red Hat 3scale API Management 产品页*
- *Red Hat 3scale API Management 文档*

4.1. 为不是由 API DESIGNER 生成的 FUSE 项目添加注解

为了让 3scale 发现 API 服务，提供 API 服务的 Fuse 应用程序必须包含 Kubernetes Service Annotations，使它可以被发现。这些注释由 Fabric8 Service Discovery Enricher 提供，后者是 Fabric8 Maven 插件的一部分。

对于 Apache Camel Rest DSL 项目，Fa Fabric8 Maven 插件默认运行 Fabric8 服务发现功能。

由 API Designer 生成的 Fuse 项目预先配置为自动提供所需注解。

流程

对于不是由 API Designer 生成的 Fuse Rest DSL 项目，请按如下所示配置项目：

1. *编辑 Fuse 项目的 **pom.xml** 文件，使其包含 **fabric8-maven-plugin** 依赖项，如下例所示：*

```
<plugin>
    <groupId>org.jboss.redhat-fuse</groupId>
    <artifactId>fabric8-maven-plugin</artifactId>
    <version>${fuse.version}</version>
    <executions>
      <execution>
        <goals>
          <goal>resource</goal>
```

```
            <goal>build</goal>
          </goals>
        </execution>
      </executions>
    </plugin>
```

如果满足某些项目级别条件（例如，项目必须是 Camel Rest DSL 项目），则 Fabric8 Maven 插件将运行 Fabric8 服务发现功能。您不需要将 Fabric8 Service Discovery Enricher 指定为 **pom.xml** 文件中的依赖项，除非您想自定义增强器的行为（如 第 4.2 节 “自定义 API 服务注解值” 所述）。

2. 在 Fuse Rest DSL 项目的 **camel-context.xml** 文件中，在 **restConfiguration** 元素中指定以下属性：

   - **方案** ：托管服务的 URL 的方案部分。您可以指定 "http" 或 "https"。
   - **contextPath** ：托管 API 服务的 URL 的路径部分。
   - **apiContextPath**: 托管 API 服务描述文档的位置的路径。如果文档在外部托管，您可以指定一个相对路径，也可以指定完整的 URL。
     以下来自 **camel-context.xml** 示例的摘录显示了 **restConfiguration** 元素中的注解属性值：

     ```
     <?xml version="1.0" encoding="UTF-8"?>
     <beans xmlns="http://www.springframework.org/schema/beans"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="
           http://www.springframework.org/schema/beans
     http://www.springframework.org/schema/beans/spring-beans.xsd
           http://camel.apache.org/schema/spring
     http://camel.apache.org/schema/spring/camel-spring.xsd">

        <camelContext xmlns="http://camel.apache.org/schema/spring">
           <restConfiguration component="servlet" scheme="https"
                contextPath="myapi" apiContextPath="myapi/openapi.json"/>
     ...
     ```

增强器使用这些 **restConfiguration** 元素属性值提供的信息来为 **discovery.3scale.net/scheme**、**discovery.3scale.net/path** 和 **discovery.3scale.net/description-path** 注解创建值，如 Red Hat 3scale API 管理门户指南的 服务发现部分所述。

增强程序添加以下标签和注解，以使服务可以被 3scale 发现：

- **discovery.3scale.net** 标签：默认情况下，增强器将此值设置为 "true"。3scale 执行选择器定义以查找需要发现的所有服务时，它会使用此标签。
- 以下注解：
  - **discovery.3scale.net/discovery-version**:（可选）3scale 发现过程的版本。增强器将此值默认设置为 "v1"。
  - **discovery.3scale.net/scheme** ：托管该服务的 URL 的方案部分。增强器使用默认的 "http"，除非您在 **restConfiguration** 元素的 **scheme** 属性中覆盖它。其他可能的值是 "https"。
  - **discovery.3scale.net/path** ：托管该服务的 URL 的路径部分。当路径位于 root, "/" 时，会省略此注解。增强器从 **restConfiguration** 元素的 **path** 属性中获取这个值。

- ***discovery.3scale.net/port*** ：服务的端口。增强器从 Kubernetes 服务定义中获取这个值，该服务包含它公开的服务的端口号。如果 Kubernetes 服务定义公开多个服务，则增强器将使用列出的第一个端口。
- ***discovery.3scale.net/description-path***: （可选） OpenAPI 服务描述文档的路径。增强器从 ***restConfiguration*** 元素的 ***contextPath*** 属性中获取这个值。

您可以自定义 Fabric8 Service Discovery Enricher 的行为，如 第 4.2 节 "自定义 API 服务注解值" 所述。

4.2. 自定义 API 服务注解值

Maven Fabric8 插件默认运行 Fabric8 服务发现功能。增强器向 Fuse Rest DSL 项目的 API 服务添加了注解，以便 API 服务可以被 3scale 发现，如 Red Hat 3scale API 管理 管理门户 指南中的使用 服务发现 所述。

增强器对一些注解使用默认值，并从项目的 ***camel-context.xml*** 文件中获取其他注解的值。

您可以通过在 Fuse 项目 ***pom.xml*** 文件或 ***service.yml*** 文件中定义值来覆盖 ***camel-context.xml*** 文件中定义的默认值和值。（如果您在两个文件中定义值，增强器将使用 ***service.yml*** 文件中的值。）有关您可以为 Fabric8 Service Discovery Enricher 指定的元素的描述，请参阅 第 4.3 节 "Fabric8 Service Discovery Enricher 元素"。

流程

在 Fuse 项目 ***pom.xml*** 文件中指定注解值：

1. 在您选择的编辑器中打开 Fuse 项目的 ***pom.xml*** 文件。
2. 找到 ***fabric8-maven-plugin*** 依赖项，如下例所示：

```
<plugin>
      <groupId>org.jboss.redhat-fuse</groupId>
      <artifactId>fabric8-maven-plugin</artifactId>
      <version>${fuse.version}</version>
      <executions>
        <execution>
          <goals>
            <goal>resource</goal>
            <goal>build</goal>
          </goals>
        </execution>
      </executions>
 </plugin>
```

3. 将 Fabric8 服务发现功能作为依赖项添加到 Fabric8-Maven 插件，如下例所示。

```
<plugin>
  <groupId>org.jboss.redhat-fuse</groupId>
  <artifactId>fabric8-maven-plugin</artifactId>
  <version>${fuse.version}</version>
  <executions>
    <execution>
      <goals>
        <goal>resource</goal>
        <goal>build</goal>
      </goals>
```

```
    </execution>
   </executions>
   <dependencies>
    <dependency>
     <groupId>io.acme</groupId>
     <artifactId>myenricher</artifactId>
     <version>1.0</version>
     <configuration>
      <enricher>
       <config>
        <f8-service-discovery>
         <scheme>https</scheme>
         <path>/api</path>
         <descriptionPath>/api/openapi.json</descriptionPath>
        </f8-service-discovery>
       </config>
      </enricher>
     </configuration>
     </dependency>
    </dependencies>
   </plugin>
```

4. *保存您的更改。*

或者，您可以使用 ***src/main/fabric8/service.yml*** *片段覆盖注解值，如下例所示：*

```
kind: Service
name:
metadata:
  labels:
    discovery.3scale.net/discoverable : "true"
  annotations:
    discovery.3scale.net/discovery-version : "v1"
    discovery.3scale.net/scheme : "https"
    discovery.3scale.net/path : "/api"
    discovery.3scale.net/port : "443"
    discovery.3scale.net/description-path : "/api/openapi.json"
spec:
  type: LoadBalancer
```

4.3. FABRIC8 SERVICE DISCOVERY ENRICHER 元素

下表描述了您可以为 Fabric8 Service Discovery Enricher 指定的元素，如果要覆盖默认值和 ***camel-context.xml*** *文件中定义的值。*

您可以在 Fuse Rest DSL 项目的 ***pom.xml*** *文件或* ***src/main/fabric8/service.yml*** *文件中定义这些值。（如果您在两个文件中定义值，增强器将使用* ***service.yml*** *文件中的值。）如需示例，请参阅 第 4.2 节 "自定义 API 服务注解值"。*

表 4.1. Fabric8 Service Discovery Enricher 元素

元素	描述	default

元素	描述	default
springDir	包含 **camel-context.xml** 文件的 spring 配置目录的路径。	用于识别 Camel Rest DSL 项目的 **/src/main/resources/spring** 路径。
scheme	托管该服务的 URL 的方案部分。您可以指定 "http" 或 "https"。	http
path	托管 API 服务的 URL 的路径部分。	
端口	托管 API 服务的 URL 的端口部分。	80
descriptionPath	托管 API 服务描述文档的位置的路径。如果文档在外部托管，您可以指定一个相对路径，也可以指定完整的 URL。	
discoveryVersion	3scale 发现实现的版本。	v1
discoverable	将 **discovery.3scale.net** 标签设为 **true** 或 **false** 的元素。 如果设置为 **true**，3scale 将尝试发现此服务。 如果设置为 **false**，3scale 不会尝试发现此服务。 您可以使用此元素作为交换机，通过将其设置为 "false" 来临时关闭 3scale 发现集成。	如果没有指定值，增强器会尝试自动检测该服务是否可以发现。如果增强器决定它无法发现该服务，3scale 不会尝试发现此服务。