Red Hat Fuse 7.13

工具用户指南

在 CodeReady Studio 中使用 Fuse 工具来开发和部署 Fuse 应用程序

Last Updated: 2024-07-01

Red Hat Fuse 7.13 工具用户指南

在 CodeReady Studio 中使用 Fuse 工具来开发和部署 Fuse 应用程序

法律通告

摘要

本指南介绍了如何使用红帽 Fuse 工具，该工具提供开发人员工具，用于在设计、开发、测试和调试您的集成应用程序时提高您的工作效率。

目录

前言

红帽 Fuse 工具是基于 Eclipse 的 IDE，可简化并简化在红帽 CodeReady Studio 中开发集成应用程序的过程。Fuse 工具提供了一组开发人员工具，它们专门设计为使用：

- Red Hat Fuse
- Red Hat JBoss EAP
- Apache Camel
- Apache CXF
- Apache Karaf
- Spring Boot

本指南提供有关如何使用 Fuse 工具进行以下操作的信息：

- 为应用程序创建一个项目，包括 Maven 依赖项
- 连接并配置企业集成模式以构建路由
- 浏览端点和路由
- 将消息拖放到运行的路由中
- 通过 JMX 浏览和视觉化运行时进程
- 调试本地运行 Camel 上下文和路由
- 通过以下方法测试应用程序：
  - 在 Apache Camel 路由中创建和使用 JUnit 测试案例
  - 使用 JMX 分析正在运行的组件
  - 通过 Apache Camel 路由追踪消息
- 部署应用程序

对于新用户，工具 Tutorials 提供了有关如何创建、调试、测试和部署示例 Camel 应用程序的分步说明。

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。这些更改将在即将发行的几个发行本中逐渐实施。详情请查看我们的 CTO Chris Wright 信息。

部分 I. 开发应用程序

第 1 章 创建新的 FUSE 集成项目

概述

创建新的 Fuse 集成项目涉及以下主要步骤：

- 指定项目名称和工作区
- 配置项目部署环境
- 选择项目模板
- 如有必要，解决 Maven 依赖项错误

配置项目后，工具会下载所有必需的 Maven 依赖项，并创建运行和发布项目所需的 POM 文件。

开始前

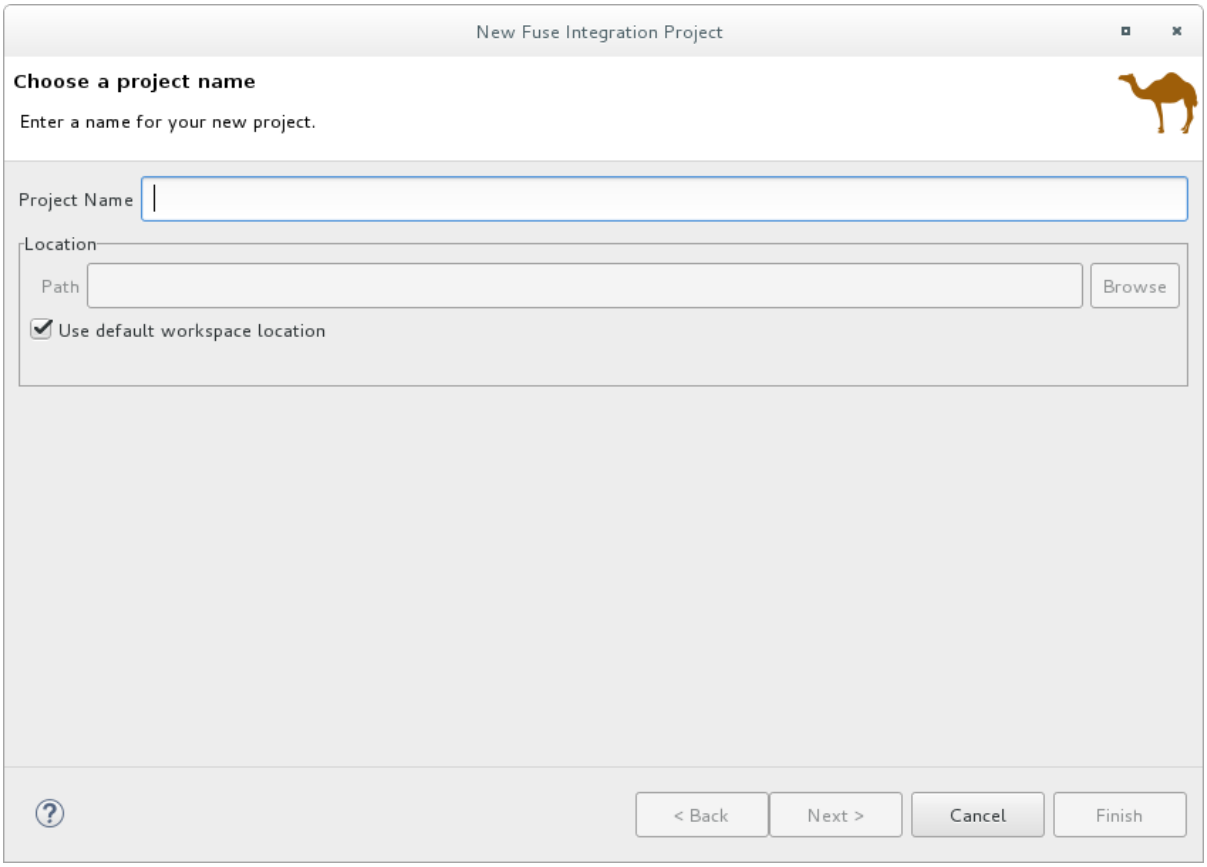
在创建新的 Fuse 集成项目前，您应该有以下信息：

- 您的目标运行时环境：OpenShift 或 Fuse 单机上的 Fuse (Spring Boot、Fuse on Karaf 或 Fuse on EAP)
- Camel 版本（如果不是工具使用的默认值）

指定项目名称和工作区

要创建新的 Fuse 集成项目，请按照以下步骤操作：

1. 选择 **New → Project → Red Hat Fuse → Fuse Integration Project** 以打开 **New Fuse Integration Project** 向导。
   这个向导会打开，并在 **Location** 窗格中选择 **Use default workspace location** 选项。



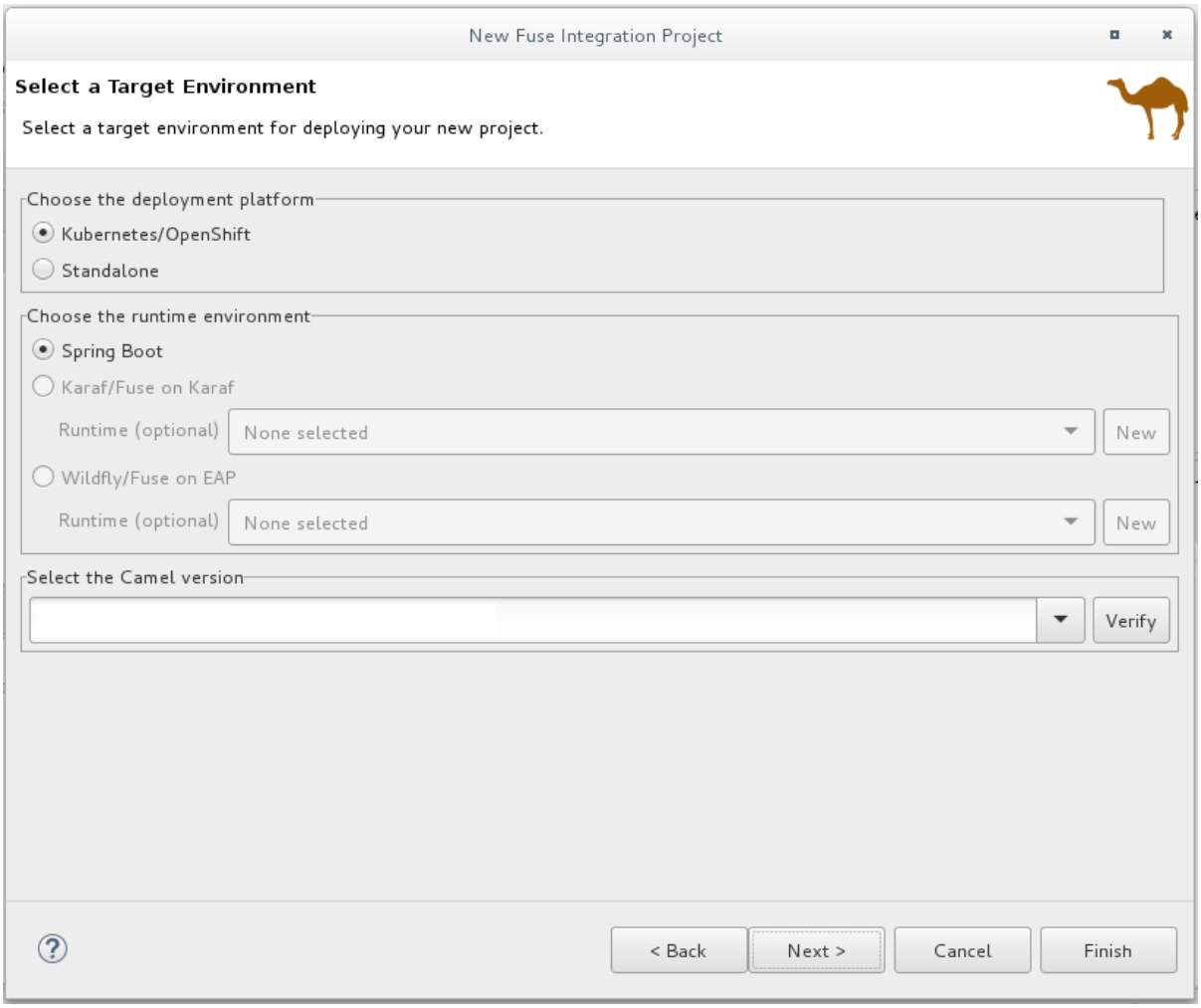
2. 在 **Project Name** 中，输入新项目的名称，如 **MySampleProject**。
3. 指定要存储项目数据的工作区位置。
   - 要使用默认工作区，请启用 **Use default workspace location** 选项。
   - 要使用其他位置清除 **Use default workspace location** 选项，并在 **Path** 字段中指定位置。点 **Browse...** 快速查找并选择备用工作区。
4. 点 **Next** 以打开 **Select a Target Environment** 页面。

配置项目部署环境

在创建新项目时，您可以指定项目的目标部署环境，以便您的项目在运行时具有所需的资源。您必须选择一个部署平台和 Camel 版本。另外，您还可以指定运行时配置。

在 **Select a Target Environment** 页面中打开：

1. 选择您要在 **Kubernetes/OpenShift** 或 **独立** 平台上部署项目。



如果为部署平台选择 **Kubernetes/OpenShift**，则会自动选择 **Sprint Boot** 运行时，您可以跳过到第 3 步。

2. 如果您为部署平台选择 **Standalone** ：

   a. 选择目标运行时环境：

      - **Spring Boot**
      - **Karaf/Fuse on Karaf**
      - **EAP 中的 WildFly/Fuse**

   b. 对于 Karaf 和 EAP 独立运行时环境，为运行时配置选择以下选项之一：

      - 接受 **None selected** 选项（您可以稍后定义运行时配置）。
      - 从下拉菜单中选择现有的运行时配置。
      - 创建新的运行时配置，如 “创建新目标运行时（可选）”一节 所述。

3. 在 **Select the Camel version for your new project**窗格中，接受与运行时关联的默认 Camel 版本或更改默认值：

   - 从下拉列表中选择 Camel 版本。Fuse 工具支持列出的产品化版本。
   - 如果要使用非产品化版本（不支持），请输入不同的 Camel 版本。

您可以点 **Verify** 按钮来检查工具是否可以访问指定的版本。如果没有，在 **Select a Target Runtime** 页面标头中会显示类似以下示例的通知：

注意

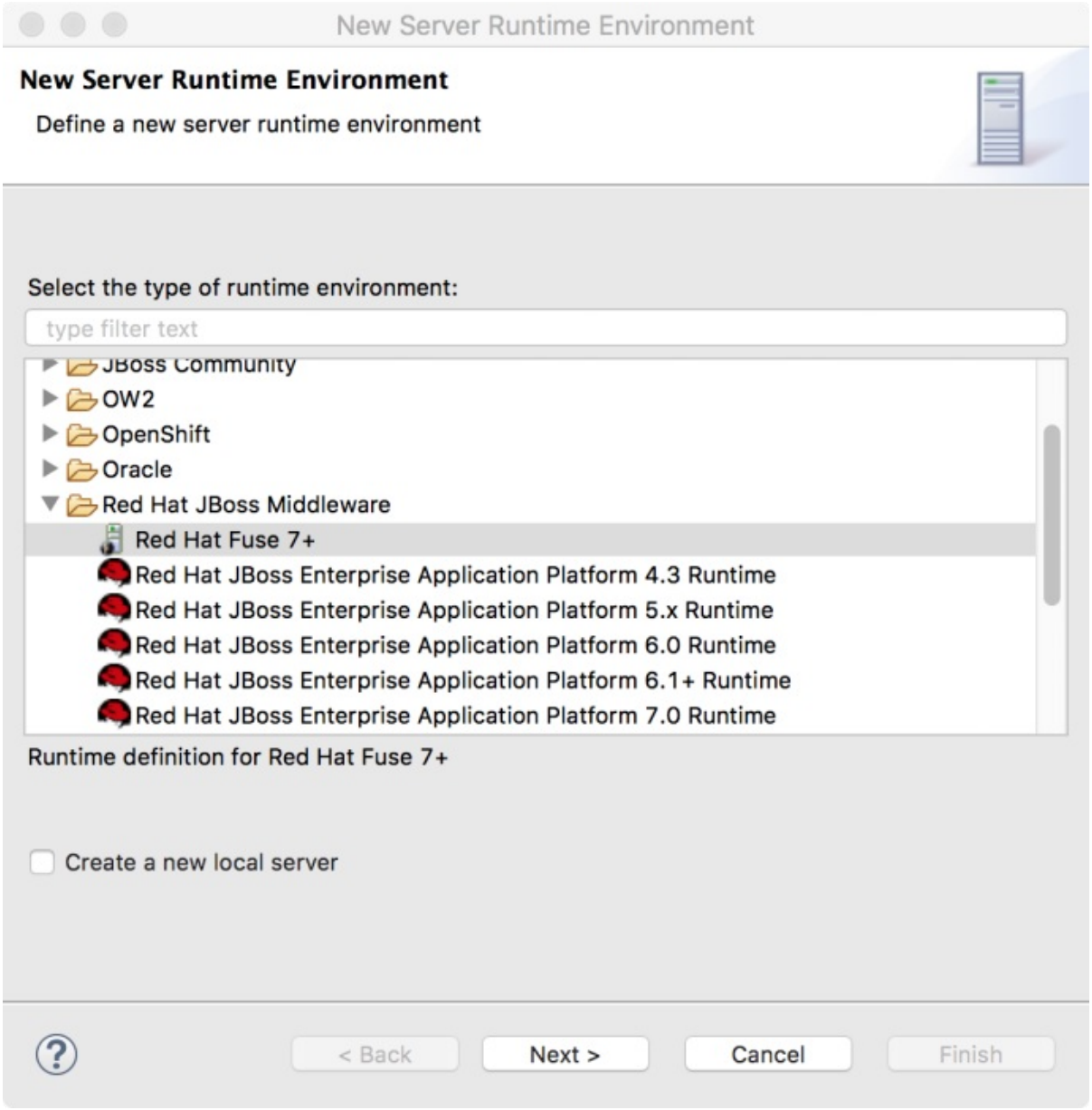
创建、配置和保存项目后，您可以更改 Camel 版本。请参阅 第 11 章 *更改 Camel 版本*。

4. 选择运行时环境和用于新 Fuse 集成项目的 Camel 版本后，点 **Next** 打开向导的高级 **项目** 设置页面，然后按照 "选择项目模板"一节 中的步骤操作。

创建新目标运行时（可选）

对于 Karaf 和 EAP 单机运行时环境，您可以选择从 **New Fuse Integration Project** 向导创建一个新的运行时配置。

1. 在向导的 **Select a Target Runtime** 页面中，点 **New** 打开 **New server runtime environment** 页面：

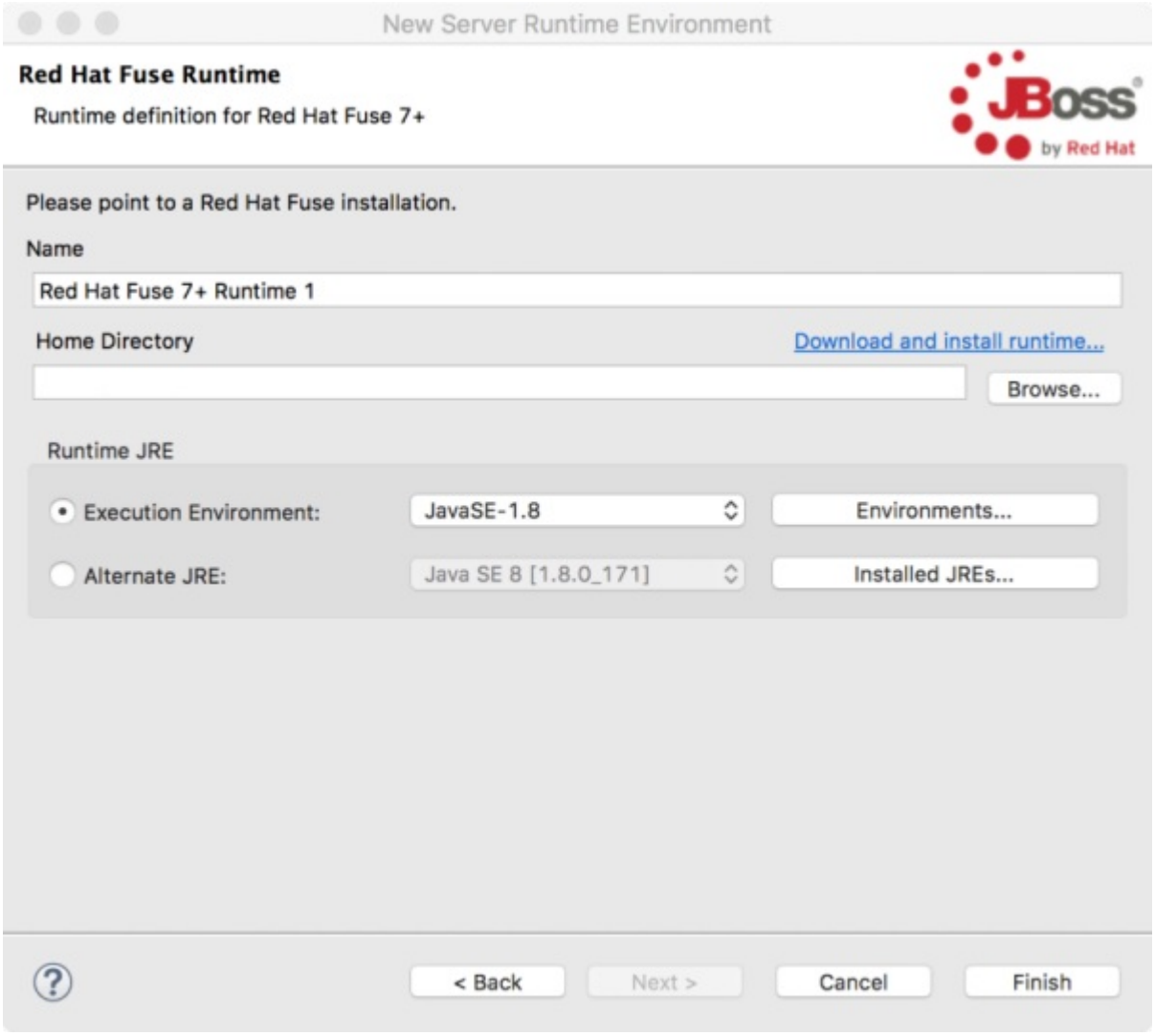


2. 扩展 **Red Hat JBoss Middleware** 文件夹，然后选择红帽 Fuse 运行时环境。
不选中 **Create a new local server**选项。当您准备好发布项目时，您可以稍后创建本地服务器（请参阅 第 27.1 节 “添加服务器”）。

注意

如果您检查 **Create a new local server**选项，**New Fuse Integration Project** 向导将引导您完成额外的步骤来定义和配置 Fuse 服务器运行时（如 第 27.1 节 “添加服务器”所述）。然后，当它构建项目时，它还会将服务器运行时添加到 **Fuse Integration** 视角中的 **服务器** 视图中。

3. 点 **Next** 以打开服务器的 **New Server Runtime Environment**页面：



4. 指定服务器运行时的名称、主目录、执行环境 ：

   - 名称 wagon- swigAccept default 或为运行时环境输入新名称。
   - Home Directory HEKETIClick Browse 按钮，以查找并选择服务器运行时的安装目录。

     注意

     如果服务器上尚未安装服务器，您可以点 Download and install runtime链接安装它，然后按照站点的下载说明进行操作。根据站点，您可能需要提供有效的凭证，然后才能继续下载过程。

   - Runtime JRE: Execution EnvironmentHEKETIAccept default 或从下拉列表中选择另一个 JavaSE 版本。
     如果您希望的版本没有出现在列表中，点 Environments 按钮，然后从该列表中选择版本。您选择的 JRE 版本必须安装在您的计算机上。

     注意

     Fuse 7.x 需要 JRE 版本 1.8.

   - Runtime JRE: Alternate JRE- 如果您的项目需要不同的 Java 版本，您可以使用这个选项。

5. 点 Finish 返回到 New Fuse Integration Project 向导的 Select a Target Runtme 页面：

新配置的目标运行时会出现在 Target Runtime 窗格的下拉菜单中，运行时支持的 Camel 版本会出现在 Camel Version 窗格中，灰显。

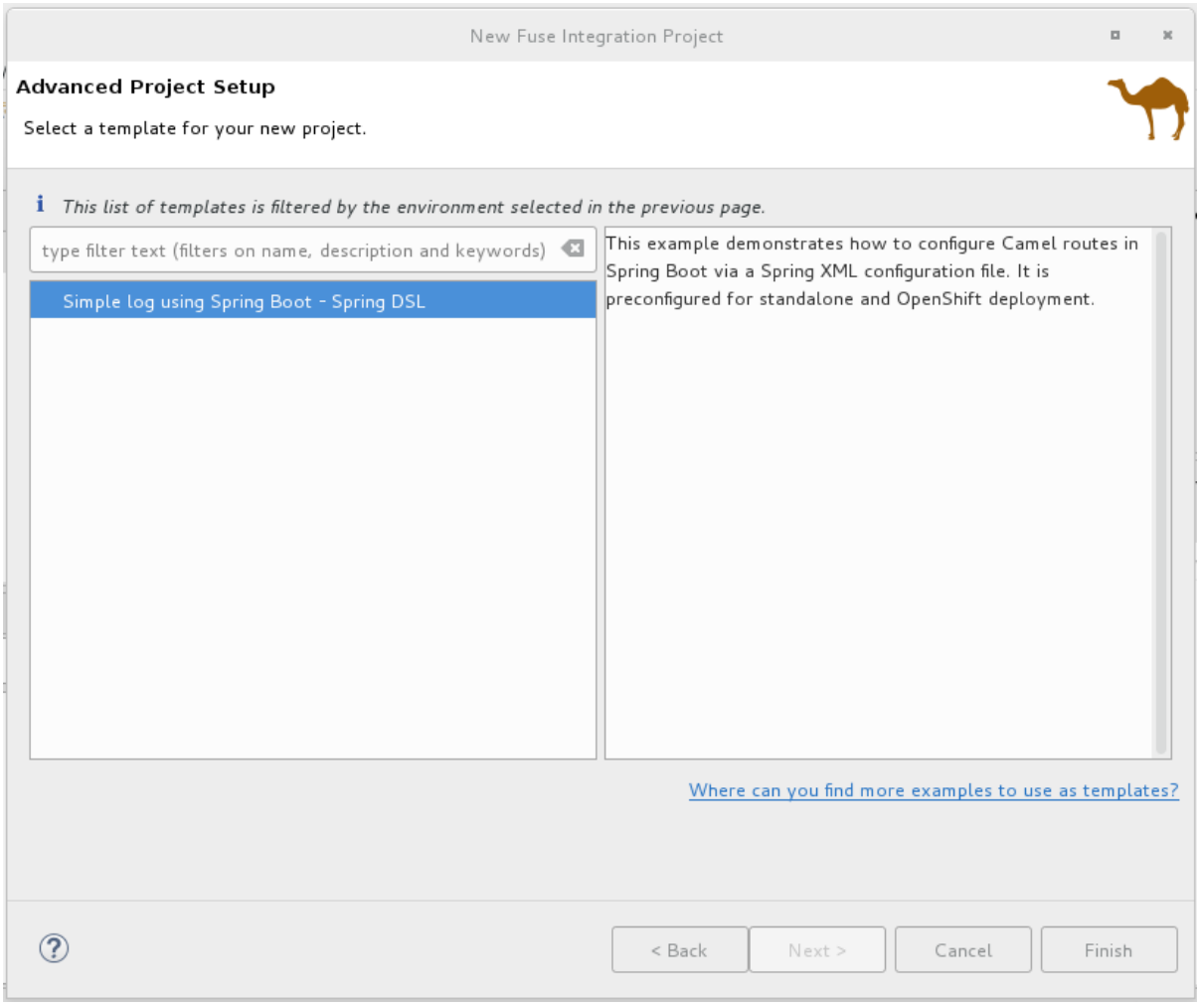
创建 Fuse 集成项目后，可以更改 Camel 版本。请参阅 第 11 章 *更改 Camel 版本*。

6. 点 Next 为项目指定一个模板，如 "选择项目模板"一节 所述。

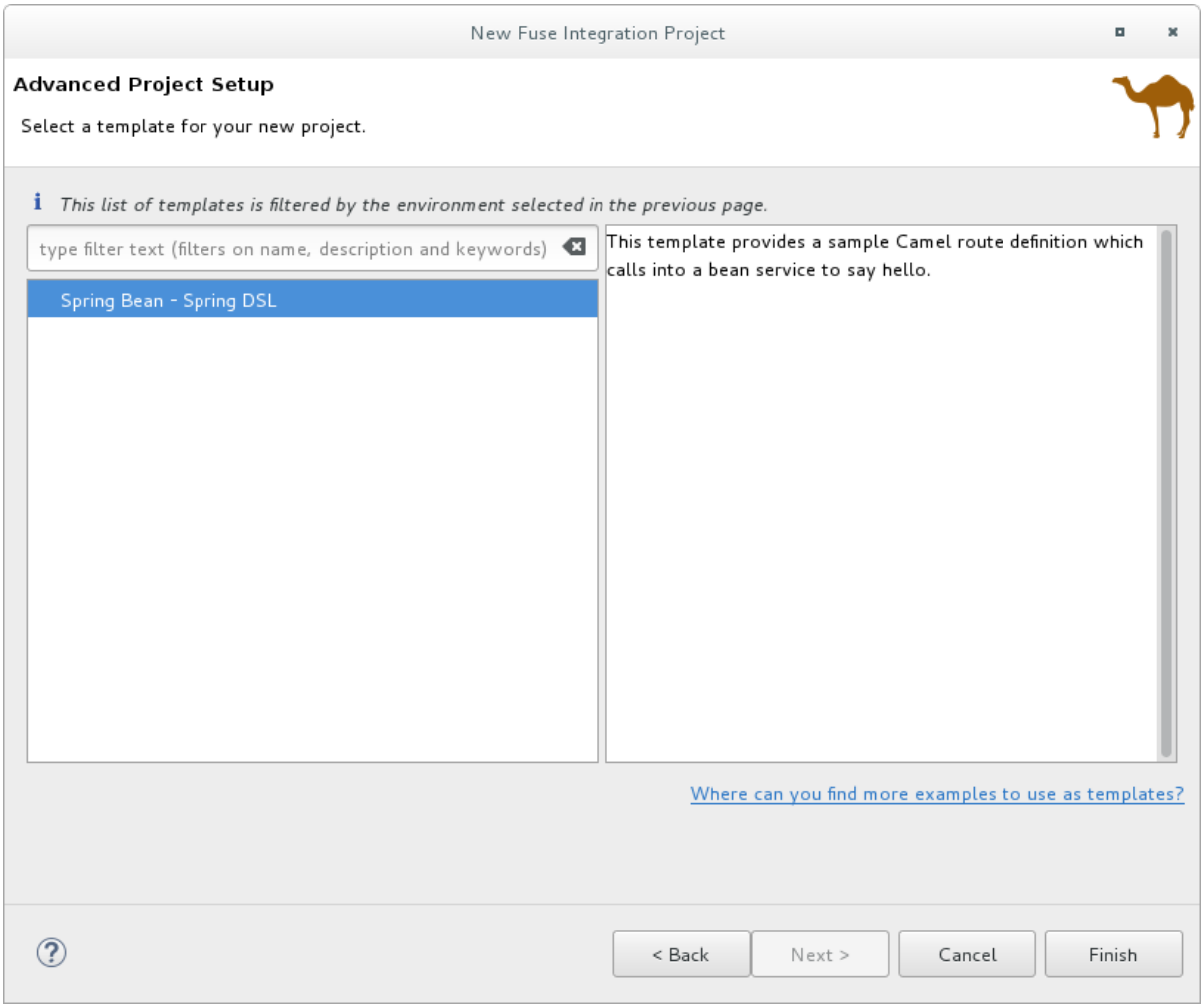
选择项目模板

Advanced Project Setup 页面提供了一个模板列表，您可以用作新项目的起点。模板基于常见用例提供示例代码和数据，以便您快速开始。可用的模板列表取决于您在上页中选择的运行时环境。选择一个模板以在右侧窗格中查看其描述。

- 对于 OpenShift 上的 Fuse，有单一模板演示了如何使用 Spring XML 配置文件在 Spring Boot 中配置 Camel 路由。此模板会创建一个 Fuse 集成项目，并需要一个比 *2.18.1.redhat-000012* 更新的 Camel 版本。
  此模板会创建一个在 OpenShift 服务器上运行的项目，仅支持 Spring DSL。有关使用此模板的详情，请参考 第 6 章 *在 OpenShift 中使用 Fuse*。



- 对于 EAP 上的 Wildfly 或 Fuse，有一个模板提供调用 Bean 服务的示例 Camel 路由，以说"Hello"。此模板会创建一个在 Red Hat EAP 服务器上运行的项目，它仅支持 Spring DSL。



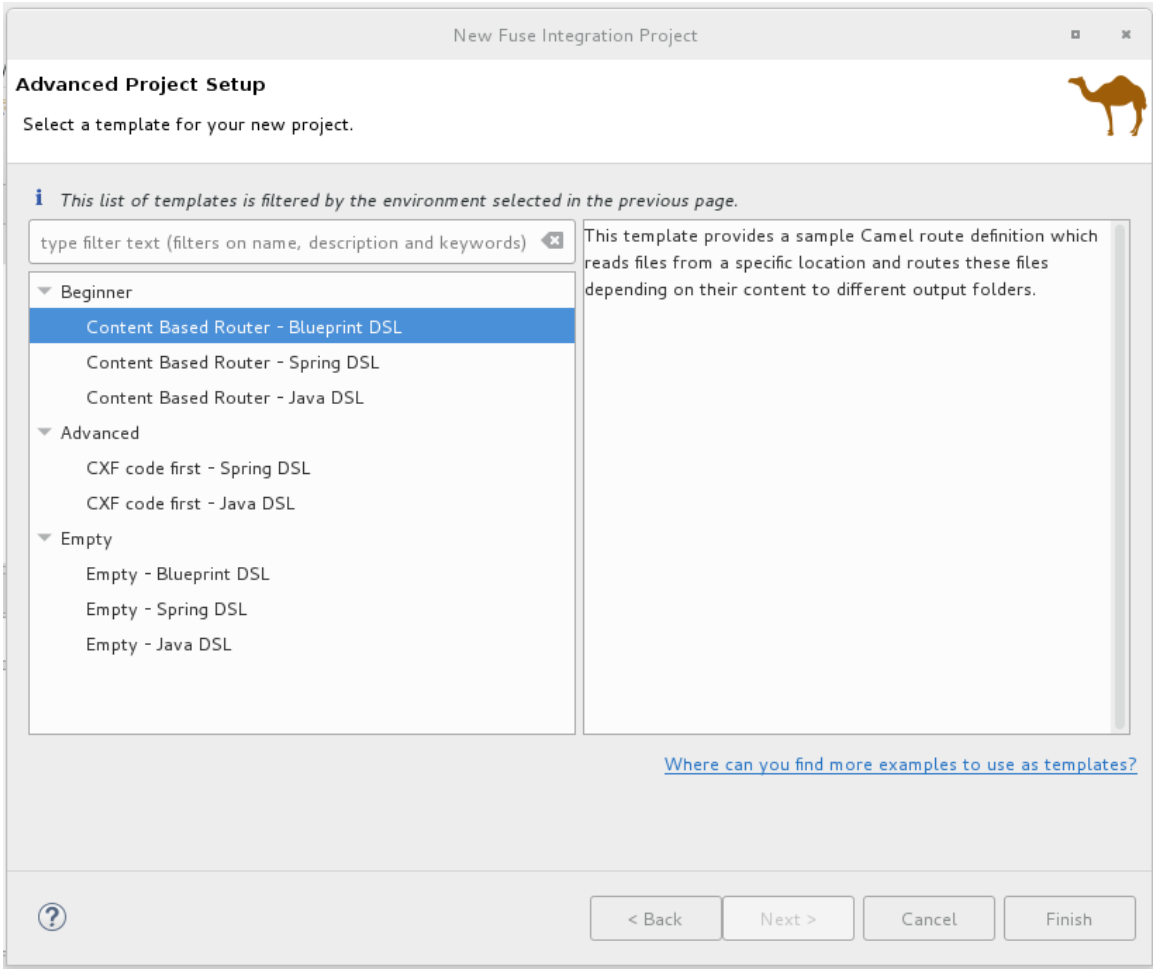
- 对于 Karaf 上的 Karaf 或 Fuse，您可以选择模板。您可以创建一个空项目，它根据三个支持的域特定语言(DSL)之一创建一个框架 Camel 上下文路由文件，或使用预定义的模板，每个项目都基于常见的用例。单个模板可能不支持所有 DSL 选项。

注意

对于 Java DSL，工具会生成一个 **CamelRoute.java** 文件，您可以在工具的 Java 编辑器中编辑该文件，但不会生成它的图形图表示。

  - 基于内容 Based Router swig-wagonProvides 是一个 Camel 路由示例，该路由从特定位置读取文件，并根据消息内容将它们路由到不同的输出文件夹。
    此模板会创建一个在红帽 Fuse 服务器上运行的项目，并支持所有三个 DSL。

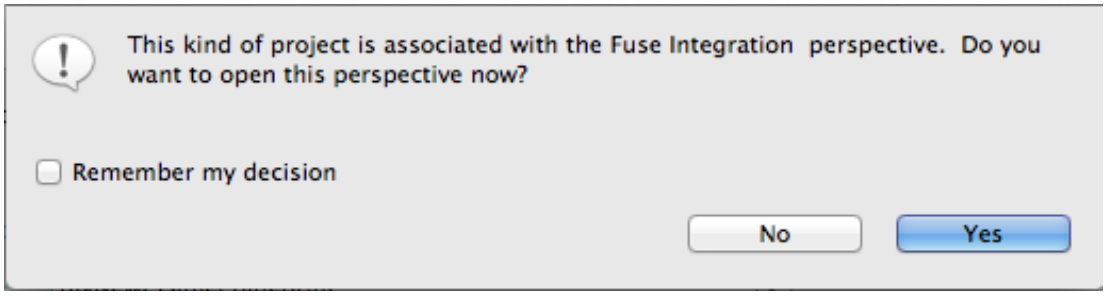
  - CXF 代码首先为 swig-wagonProvides 是一个由 CXF Web 服务调用启动的 Camel 路由示例。
    此模板会创建一个在红帽 Fuse 服务器上运行的项目，并且仅支持 Spring 和 Java DSL。



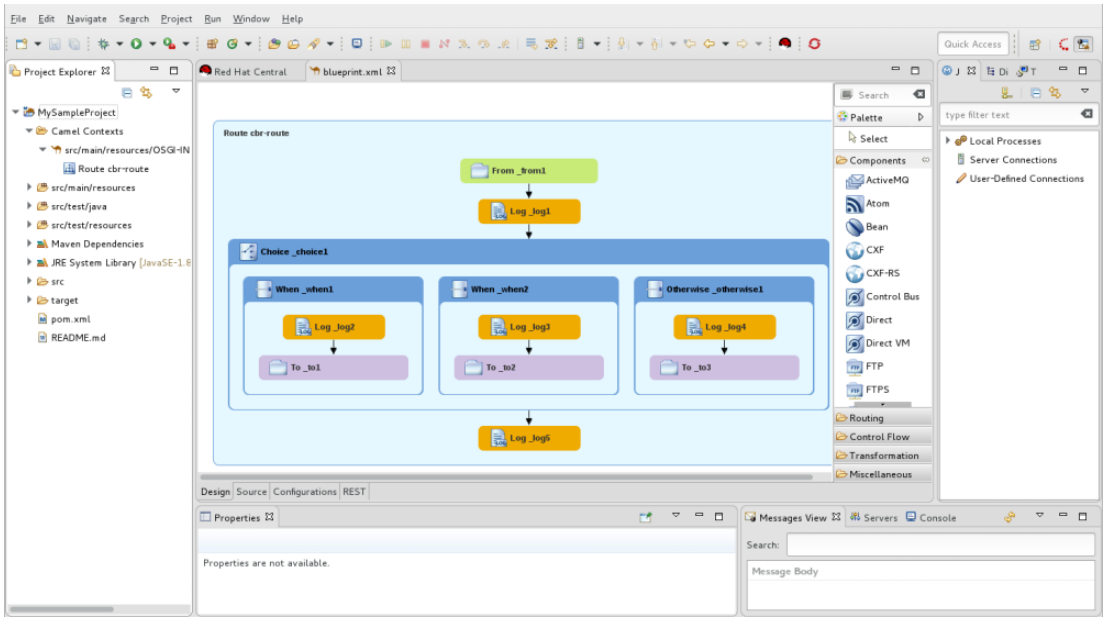
1. **从列表中选择模板。**

2. **点 Finish。**
   该工具开始构建项目，并将其添加到 Project Explorer 视图中。

   如果 Fuse Integration 视角还没有打开，工具会询问您现在是否要切换到它：

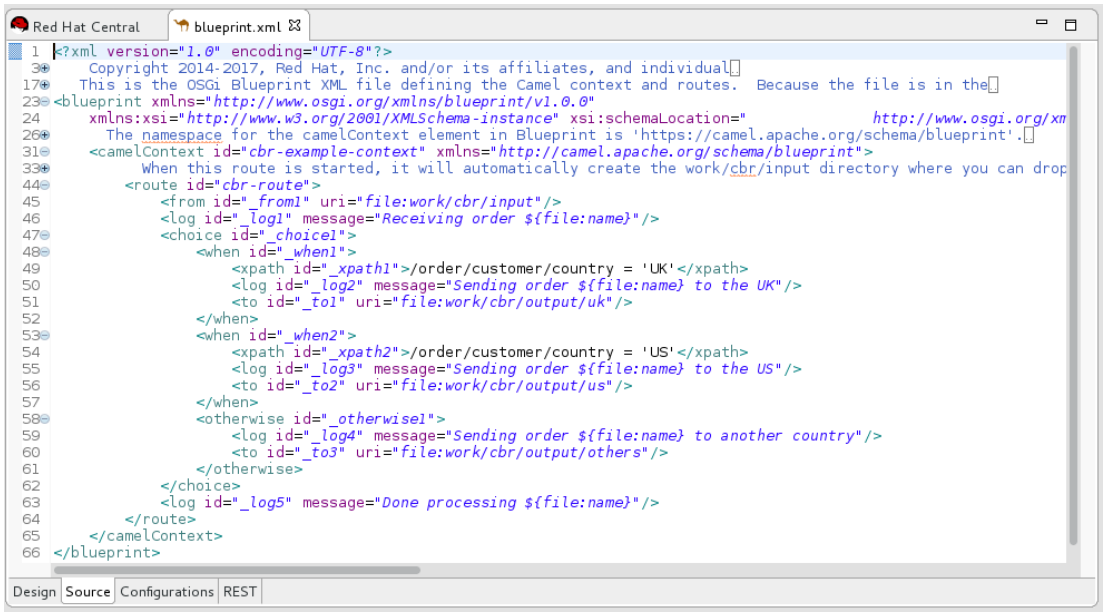


3. **点 Yes 在 Fuse Integration 视角中打开新项目：**



该项目会出现在 Project Explorer 视图中。默认情况下，项目包含一个 Apache Camel 上下文(XML)文件。

4. 点 canvas 底部的 Source 选项卡查看生成的 Camel 上下文文件：



注意

如果要在项目中添加另一个新的 Camel 上下文文件，请参阅 第 10 章 *创建新的 Camel XML 文件*。

当您构建使用 CXF 的项目时，您可能希望构建过程自动操作 Java 文件来生成 WSDL 文件。为此，请在项目的 **.pom** 文件中配置 **java2ws** Maven 插件。请参阅：Apache CXF 开发指南、Maven 工具参考、java2ws。

解决 MAVEN 依赖项错误

在创建新的 Fuse 集成项目后，您可能会遇到 Maven 依赖项错误。

虽然它可能会在其他时间发生，但在进程完成前取消项目构建时通常会发生。以这种方式中断进程通常可防止所有项目的依赖项从 Maven 存储库下载，这可能需要一些时间。

您通常可以通过更新 Maven 依赖项来解决这些依赖项错误，如下所示：

1. 在 Project Explorer 视图中，右键单击 root 项目，以打开上下文菜单。
2. 选择 Maven → Update Project。
3. 在 Update Maven Project 向导中：
   - 如果向导的列表中出现多个您要更新的项目，请选择您要更新的项目。
   - 单击 强制更新 Snapshots/Releases 选项以启用它。
4. 点击 确定。
   在工作台的右上角，您可以查看进度状态栏，因为 Maven 存储库中下载缺少的依赖项。

第 2 章 在路由编辑器中编辑路由上下文

The following sections describe how to edit a routing context.

2.1. 在路由中添加模式

路由由一系列连接模式组成，当节点放置在 Route 容器节点上时，就称为节点。完整的路由通常由起始端点、处理节点字符串和一个或多个目标端点组成。

当您向 canvas 上的 Route 容器中添加模式时，模式会采用表示其节点类型的颜色：

- **BLUE-5-4Route 容器，与上下文文件中的路由元素和其他容器节点对应，如何时和包含完成其逻辑的其他 EIP 时的 EIP**
- **Gregeyncy to input data entered routes （进入路由）的 green swig-wagonConsumer 端点**
- **带入路由、转换、进程或控制数据传输路由的流**
- **输出数据退出路由的 purple5-4Producer 端点**

流程

将模式添加到路由中：

1. **在 swig 中，找到您要添加到路由的模式。**
2. **使用以下方法之一：**
   - **单击 swig 中的模式，然后在 canvas 中单击路由容器。**
   - **将模式拖动到目标 Route 容器并丢弃它。**

   或者，您可以在没有传出连接的现有节点上或两个节点间现有的连接中添加一个模式，以便工具在涉及所有节点之间的连接自动连接。

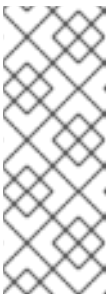
该工具会检查生成的连接是否有效，然后允许或阻止您在目标中添加模式。对于有效的连接，该工具的行为根据目标是节点还是连接而有所不同：

- 对于 现有节点，工具会将新节点添加到目标节点的传出端（根据 编辑器首选项如何设置），并在它们之间自动连接
- 对于 现有连接，工具在两个连接的节点之间插入新节点，并自动重新连接三个节点之间的连接

3. 另外，您可以手动连接两个节点：

   a. 在 canvas 上的 Route 容器中，选择源节点来显示其连接器箭头。

   b. 将源节点的连接器箭头()拖到目标节点上，并释放鼠标按钮来丢弃连接器。



注意

并非所有节点都能够连接。当您试图将源节点连接到无效的目标节点时，工具会显示附加到鼠标光标的  符号，连接器无法代表到目标节点。

4. 在 Route 容器中添加模式后，您可以将其拖到路由容器内的不同位置，或者将其拖到 canvas 上的另一个路由容器，只要它可以建立有效的连接。您还可以重新定位已连接的现有节点，只要移动可以建立另一个有效连接。

   要查看说明如何重新定位端点的简短视频，请单击此处。

5. 选择 File → Save。该工具会在上下文文件中保存路由，无论它们是否完成。

新模式会出现在 Route 容器中的 canvas 中，并成为所选节点。Properties 视图显示您可以编辑的新节点属性的列表。

更改布局方向

当您将一个节点连接到另一个节点时，工具会根据路由编辑器的布局首选项更新布局。默认值为 Down。

要访问路由编辑器的布局首选项：

- 在 Linux 和 Windows 机器上，选择 Windows → Preferences → Fuse Tooling → Editor → 选择图编辑器的布局方向。

相关主题

- 第 2.2 节 “配置模式”
- 第 2.3 节 “从路由中删除模式”

2.2. 配置模式

概述

大多数模式需要一些显式配置。例如，端点需要明确输入的 URI。

工具的 Properties 视图提供了一个表单，用于列出特定模式支持的所有配置详情。Properties 视图还提供以下方便的功能：

- 验证所有必要属性是否具有值
- 验证提供的值是否为属性的正确数据类型
- 具有固定值集的属性的下拉列表
- 下拉列表填充 Apache Camel Spring 配置中可用的 bean 引用

流程

配置模式：

1. 在 canvas 上，选择您要配置的节点。

   Properties 视图列出了您要编辑的所有所选节点属性。对于 EIPs，Details 选项卡列出了模式的所有属性。对于 Components drawer 的组件，Details 选项卡列出了常规属性和需要值的选项，高级 选项卡则列出了根据功能分组的其他属性。

   Documentation 选项卡描述模式及其每个属性。

2. 编辑 Properties 视图中的字段以配置节点。

3. 完成后，通过从菜单栏中选择 File → Save 保存您的工作。

2.3. 从路由中删除模式

概述

在开发和更新路由时，您可能需要删除一个或多个路由的节点。节点的  图标可以轻松地这样做。当您从 canvas 中删除节点时，其与路由中其他节点的连接也会被删除，节点会从上下文文件中对应的 route 元素中删除。

注意

您还可以通过打开其上下文菜单并选择 Remove 来删除节点。

流程

从路由中删除节点：

1. 选择您要删除的节点。

2. 点其

图标。

3. 在询问您是否要删除此元素时，请单击 **Yes**。

节点及其所有连接都从 canvas 中删除，节点会从上下文文件中对应的 route 元素中删除。

相关主题

- 第 2.1 节 “在路由中添加模式”

2.4. 在路由上下文中添加路由

概述

XML 上下文文件中的 camelContext 元素会创建一个路由上下文。camelContext 元素可以包含一个或多个路由，每个路由都显示为 Route 容器节点，映射到生成的 camelContext 元素中的 route 元素。

流程

将另一个路由添加到 camelContext 中：

1. 在 Design 选项卡中，执行以下操作之一：

   - 单击 Schedule 的Routing drawer 中的 Route 模式，然后单击您要放置路由的 canvas。
   - 将 Route 模式从 Dashboards 的Routing drawer 拖放到 canvas。

   Properties 视图显示您可以编辑的新路由属性的列表。

2. 在 Properties 视图中，输入：

- 路由 Id 字段中新路由的 ID （如 Route2）

注意

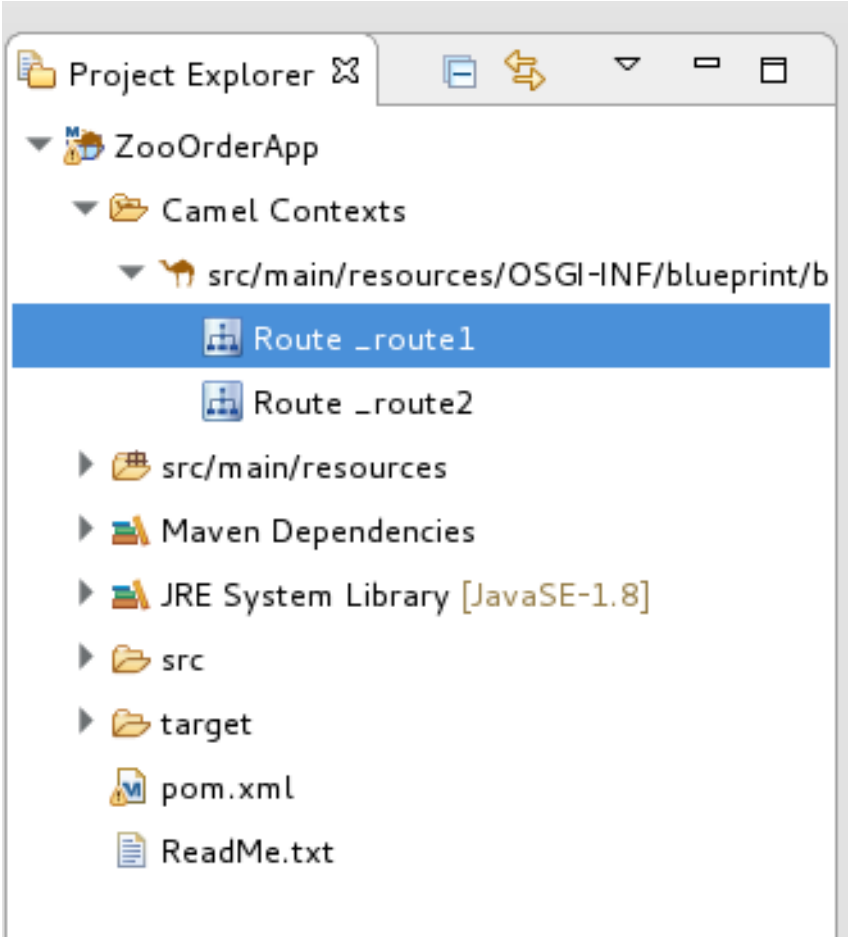
该工具会自动为 canvas 上丢弃的 EIP 和组件模式分配 ID。您可以将这些自动生成的 ID 替换为您自己的 ID，以区分项目中的路由。

- Description 字段中路由的描述

- 根据需要，任何其他属性的值。必要属性由星号 HEKETI 表示。

3. 在菜单栏中，选择 File → Save 以保存您对路由上下文文件所做的更改。

4. 要在多个路由间切换，请点击 Project Explorer 视图中项目的 Camel Contexts 文件夹下的条目，选择要在 canvas 上显示的路由。



5.

要显示上下文中的所有路由，如空间允许，请单击 **Project Explorer** 视图中的上下文文件条目。

6. 要在向 **canvas** 添加路由时查看工具生成的代码，请点 **Source** 选项卡。

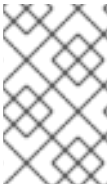
注意

您还可以通过将 **<route/>** 元素添加到 **camelContext** 元素中的现有列表中，在 **Source** 选项卡中添加路由。

2.5. 删除路由

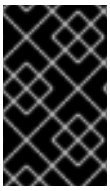
概述

在某些情况下，您需要从路由上下文中删除整个路由。**Route** 容器的  图标可以更容易实现。当您删除路由时，**Route** 容器中的所有节点也会被删除，上下文文件中对应的 **route** 元素会被删除。



注意

您还可以使用 **Route** 容器的上下文菜单并选择 **Remove** 来删除路由。



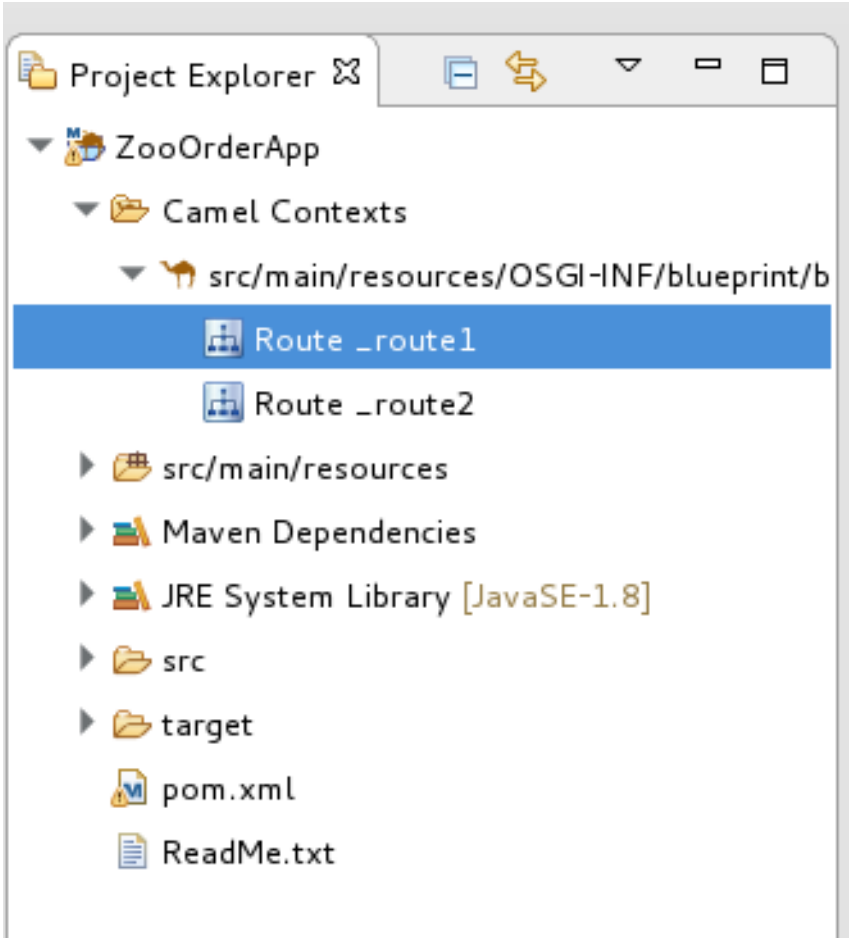
重要

您无法撤销此操作。

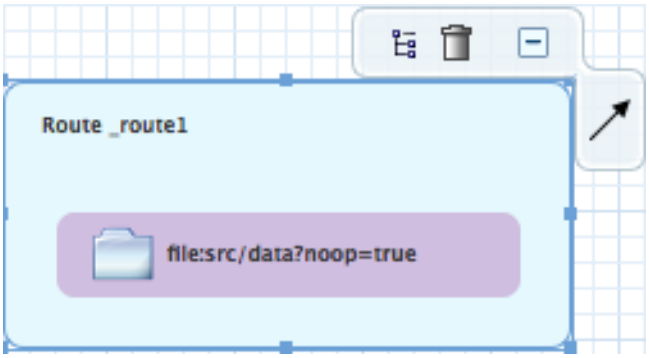
流程

删除路由：

1. 如果路由上下文包含多个路由，则首先在 **Project Explorer** 视图中选择您要删除的路由。



2. **在 canvas 中，点 Route 容器的**  **图标。**



3. **在询问您是否要删除此元素时，请单击 Yes。**

该路由从 canvas、context 文件和 Project Explorer 视图中删除。

2.6. 添加全局端点、数据格式或 BEAN

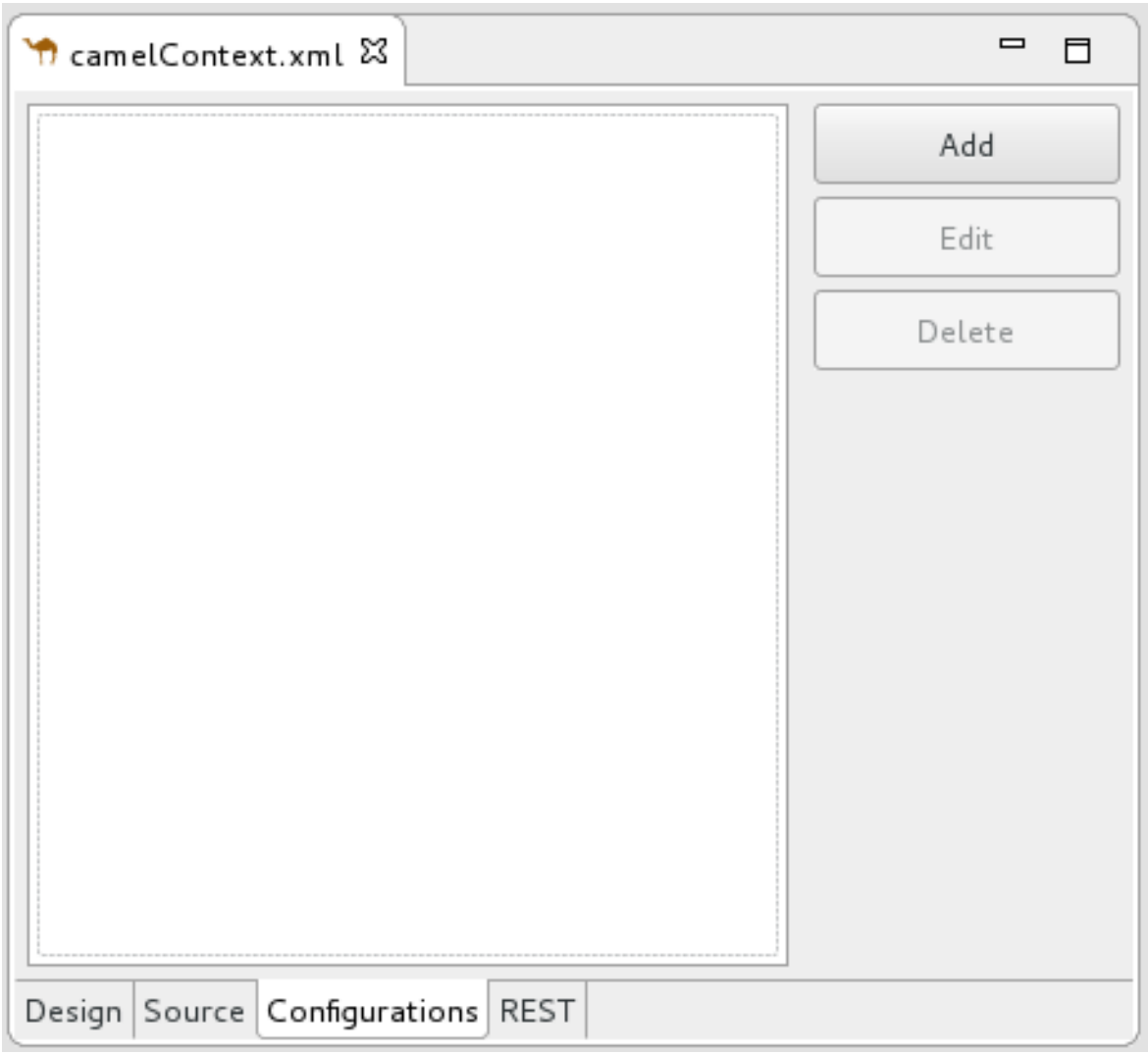
概述

有些路由依赖于全局端点、全局数据格式或全局 Bean 提供的共享配置。您可以使用路由编辑器的 Configuration 选项卡将全局元素添加到项目的路由上下文文件中。

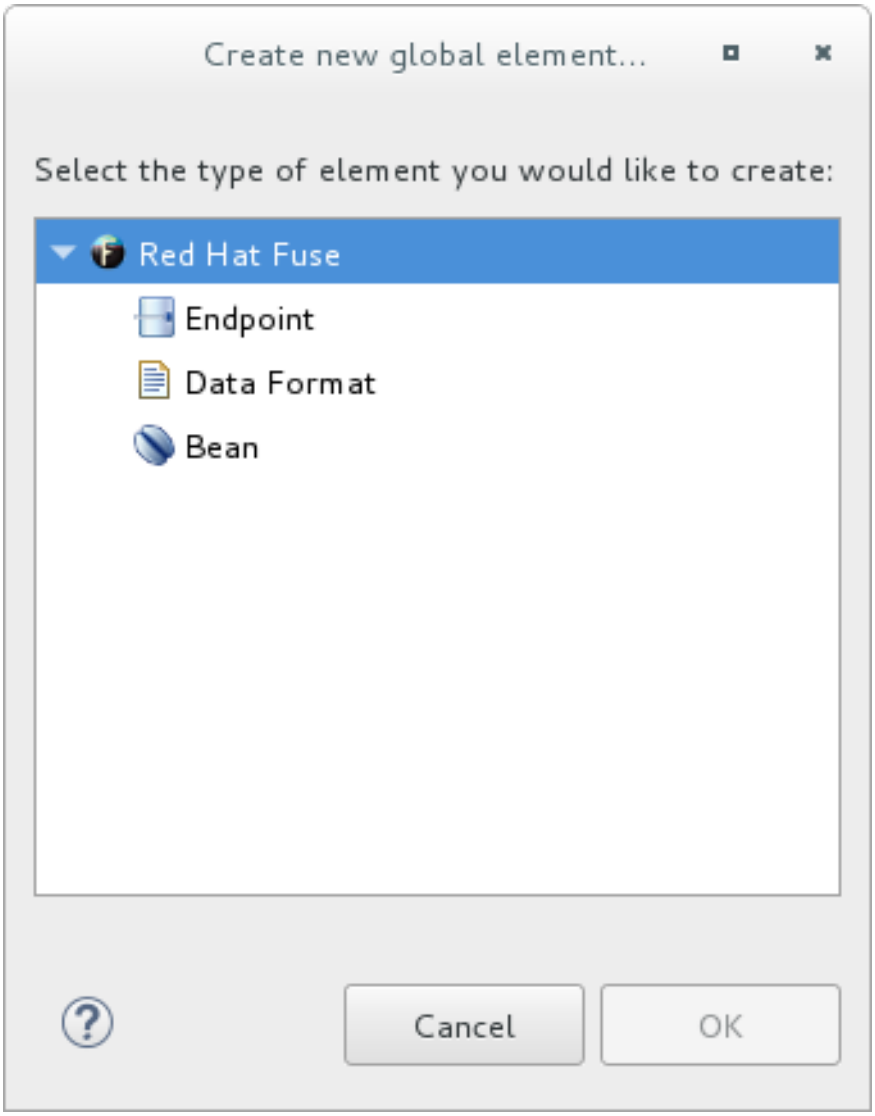
在您的路由上下文文件中添加全局元素：

1. 在路由编辑器中打开您的路由上下文文件。

2. 在路由编辑器的底部，单击 Configuration 选项卡，以显示全局配置（若有）。



3. 单击 Add 以打开 Create a new global element 对话框。

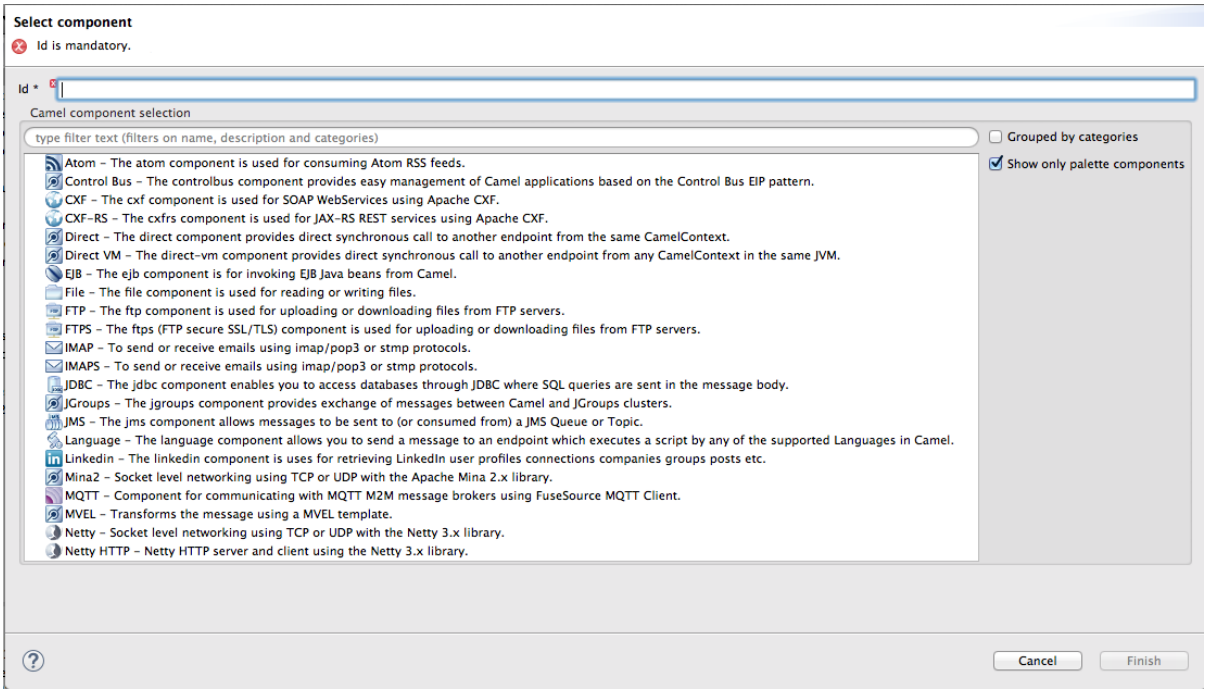


这些选项是：

- **endpoint swig-wagon** 请参见 **“添加全局端点”一节**。
- **数据格式 swig-wagon** 请参见 **“添加全局数据格式”一节**。
- **bean swig-wagon** 请参见 **“添加全局 bean”一节**。

添加全局端点

1. **在 Create a new global element 对话框中，选择 Endpoint 并点 OK 以打开 Select component 对话框。**

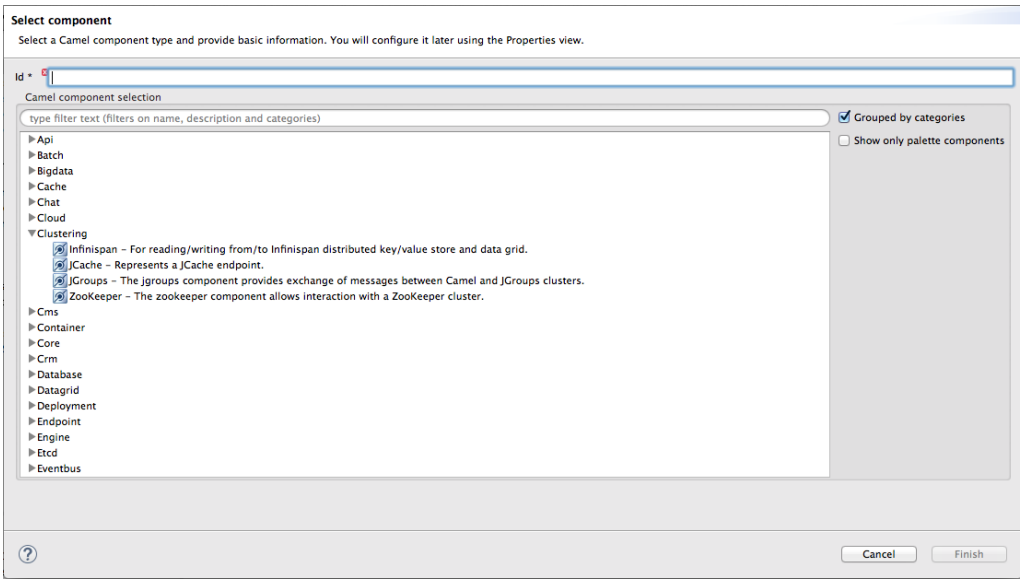


注意

默认情况下，会打开 Select component 对话框，并启用了 Show only palette 组件 选项。要查看所有可用的组件，请取消选中这个选项。

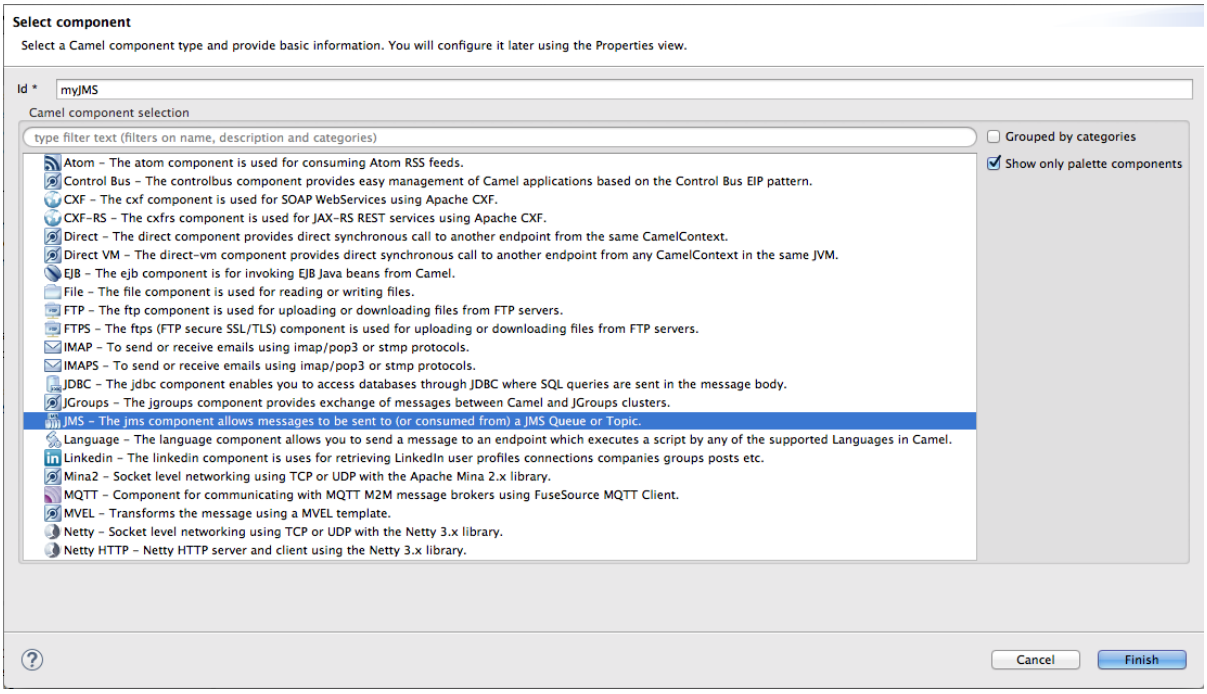
注意

由类别 选项组按类型分组组件。



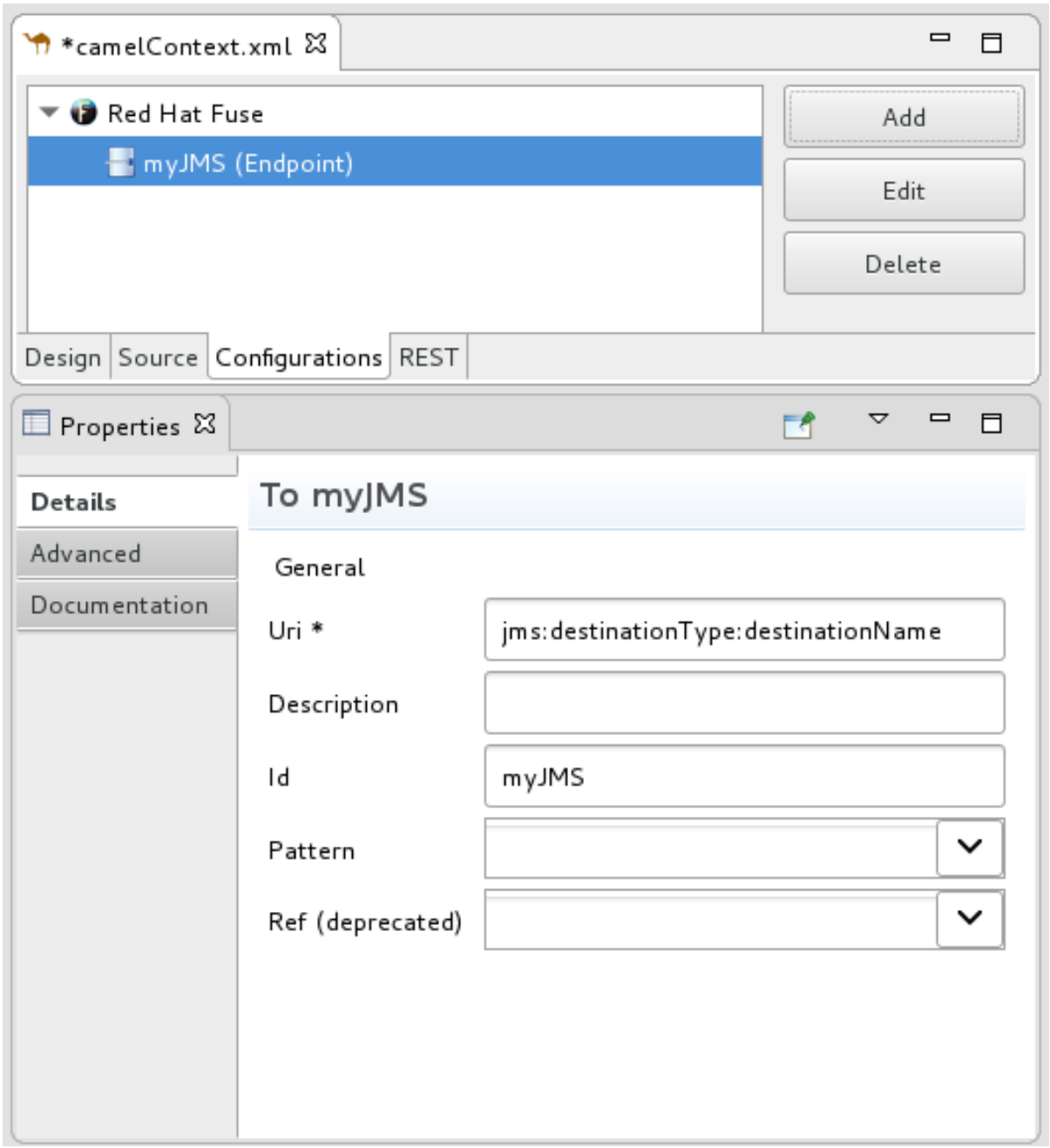
2.

在 Select component 对话框中，滚动浏览 Camel 组件列表以查找并选择您要添加到上下文文件的组件，然后在 Id 字段中输入其 ID。



在本例中，选择 JMS 组件，myJMS 是 Id 值。

3. **点 Finish。**



现在，您可以根据需要在 **Properties** 视图中设置属性。

工具会在 [globalEndptSelect] 中的组件 **Id** 字段中输入的值自动填充 **Id**。在本例中，**Camel** 构建 **uri** （必需字段），从组件的模式开始（本例中为 **jms:**），但您必须指定 **destinationName** 和 **destinationType** 以完成组件的 **uri**。

注意

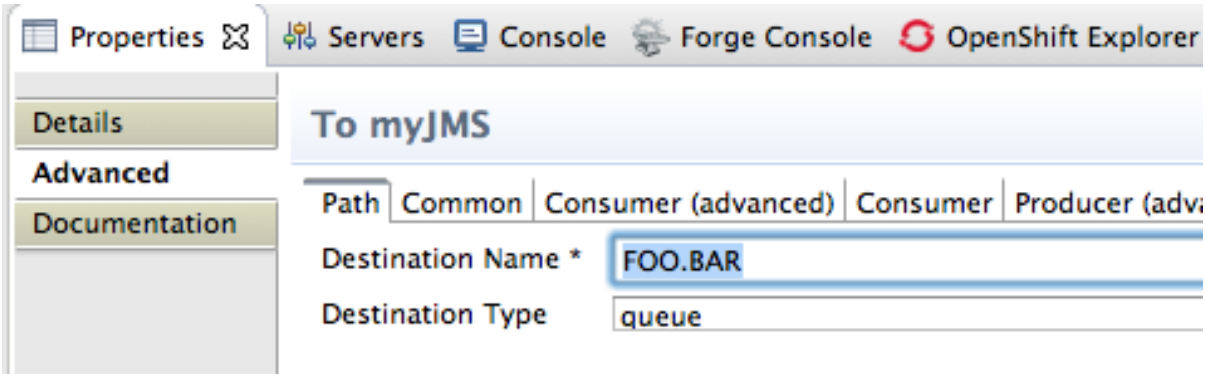
对于 **JMS** 组件，目的地类型默认为 **queue**。在 **Destination Name** （必需字段）中输入了一个值，此默认值不会出现在 **Details** 页面中的 **uri** 字段中。

4.

要完成组件的 uri，请单击 **Advanced**。

5.

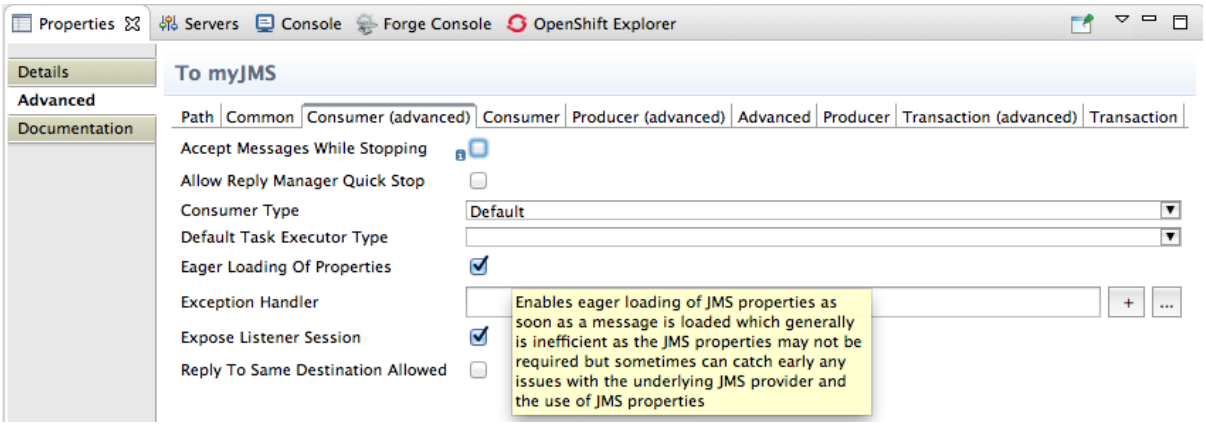
在 **Destination Name** 字段中，输入目标端点的名称（如 **FOO.BAR**）。在 **Destination Type** 字段中，输入端点目的地的类型（如 **queue**、**topic**、**temp:queue** 或 **temp:topic**）。



Properties 视图的 **Details** 和 **Advanced** 选项卡提供对可用于配置特定组件的所有属性的访问。

6.

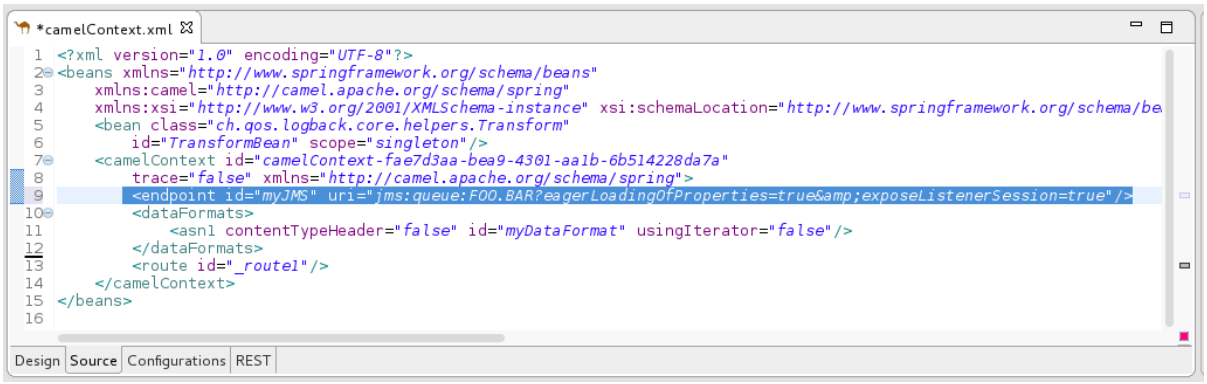
点 **Consumer (advanced)** 选项卡。



启用属性 **Eager Loading Of Properties** 和 **Expose Listener Session**。

7.

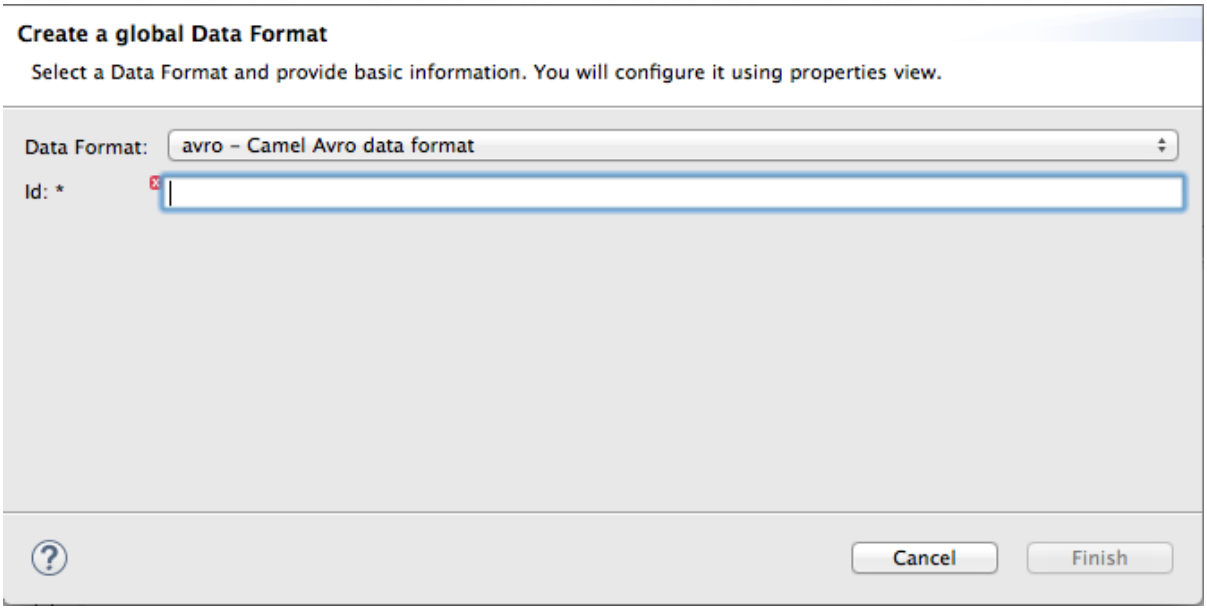
在路由编辑器中，切换到 **Source** 选项卡，在第一个 **route** 元素之前，查看添加到上下文文件的工具（本例中为配置的 **JMS** 端点）。



8. **完成后，通过选择 File → Save 来保存您的更改。**

添加全局数据格式

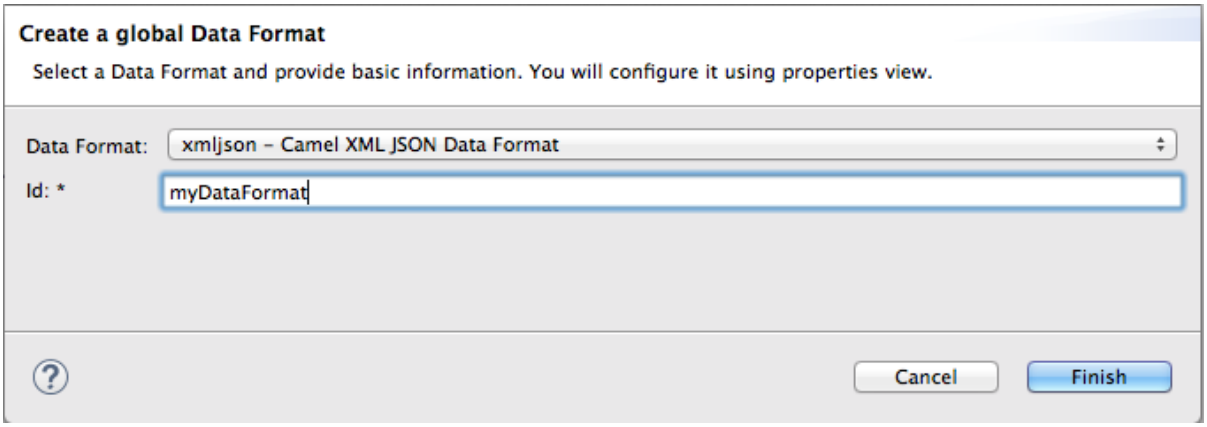
1. **在 Create a new global element 对话框中，选择 Data Format 并点 OK 以打开 Create a global Data Format 对话框。**



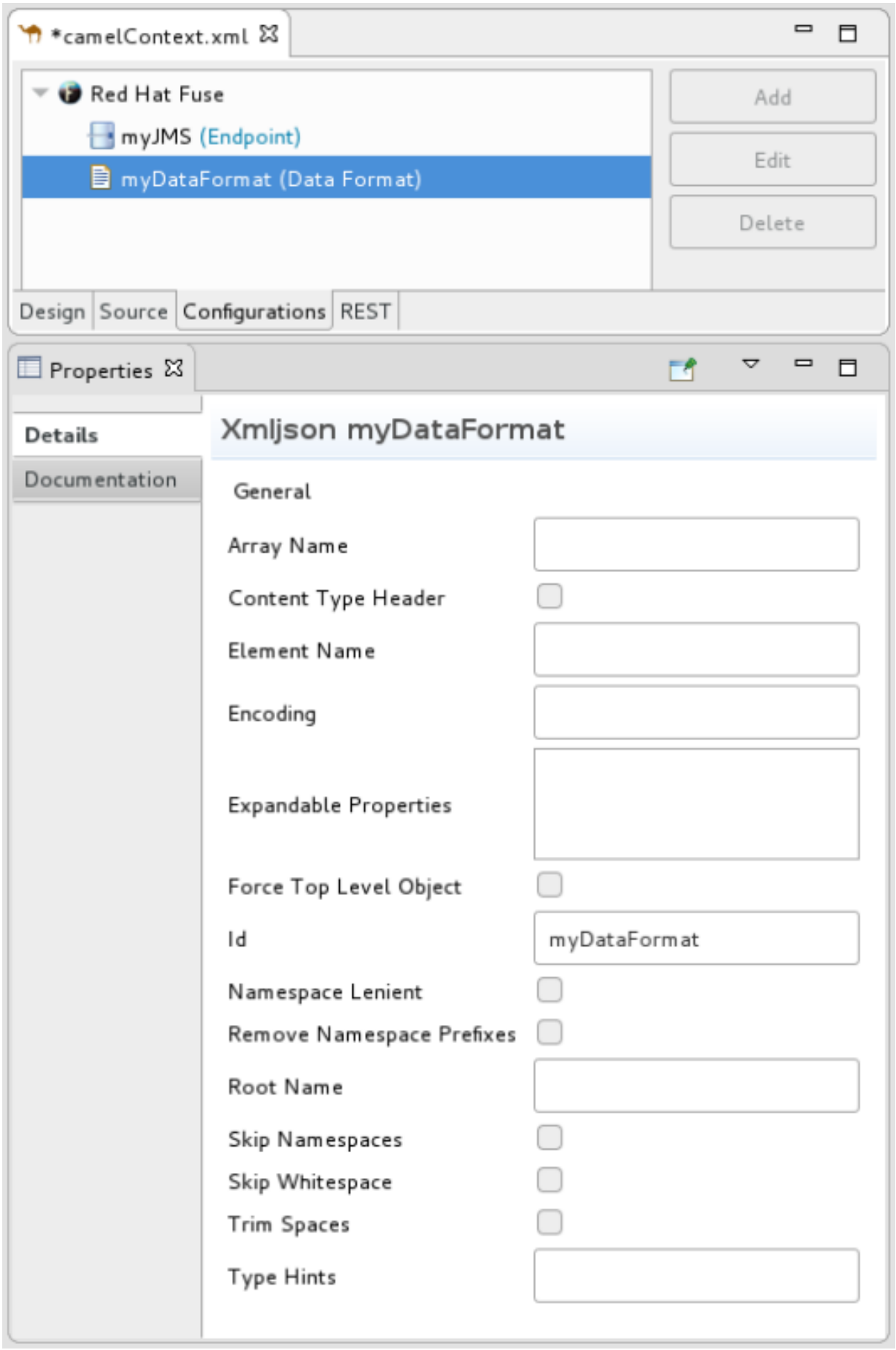
数据格式默认为 avro，其格式位于可用列表的顶部。

2. **打开 Data Format 下拉菜单，再选择您想要的格式，如 xmljson。**

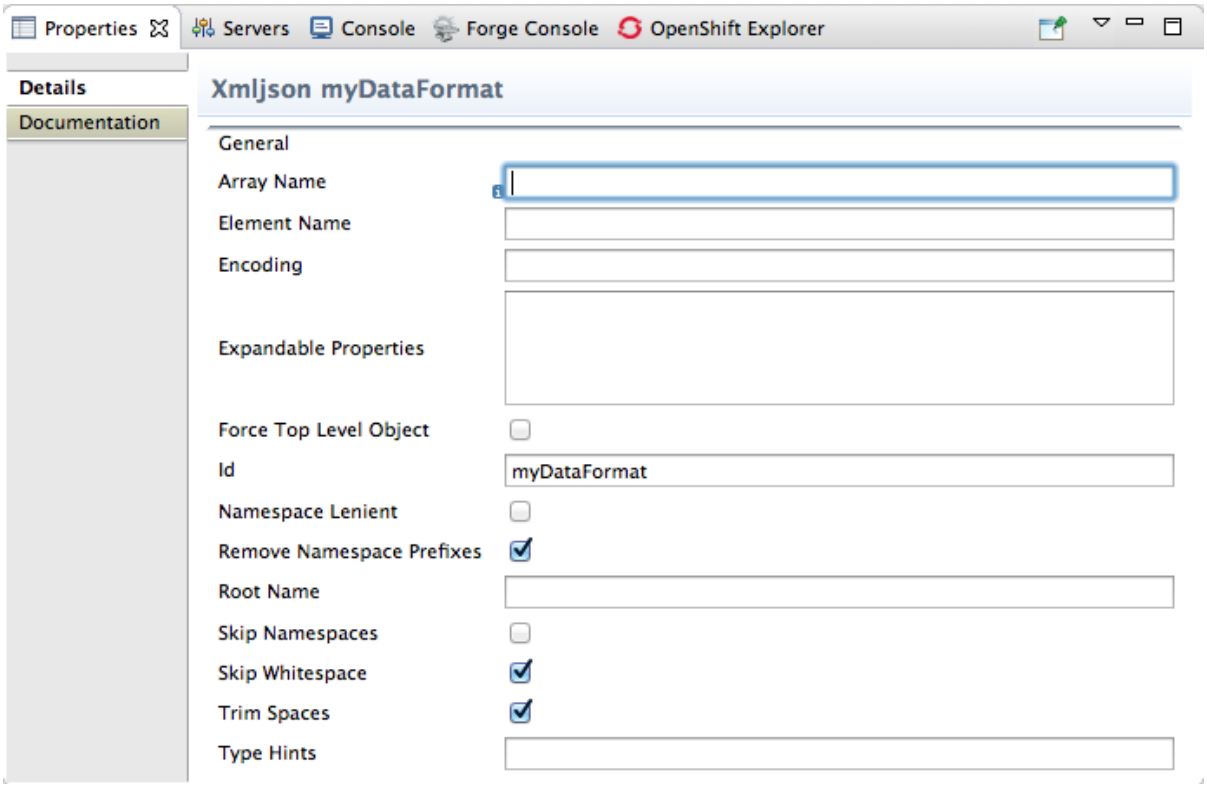
3. **在 Id 字段中输入格式的名称，如 *myDataFormat*。**


Create a global Data Format
Select a Data Format and provide basic information. You will configure it using properties view.
Data Format:
xmljson – Camel XML JSON Data Format
Id: *
myDataFormat
Cancel
Finish

4. 点 **Finish**。



5. **在 Properties 视图中，根据项目设置属性值，例如：**



6. **在路由编辑器中，点 Source 选项卡查看工具添加到上下文文件中的代码。在本例中，配置的 xmljson 数据格式是第一个 route 元素的前面。**

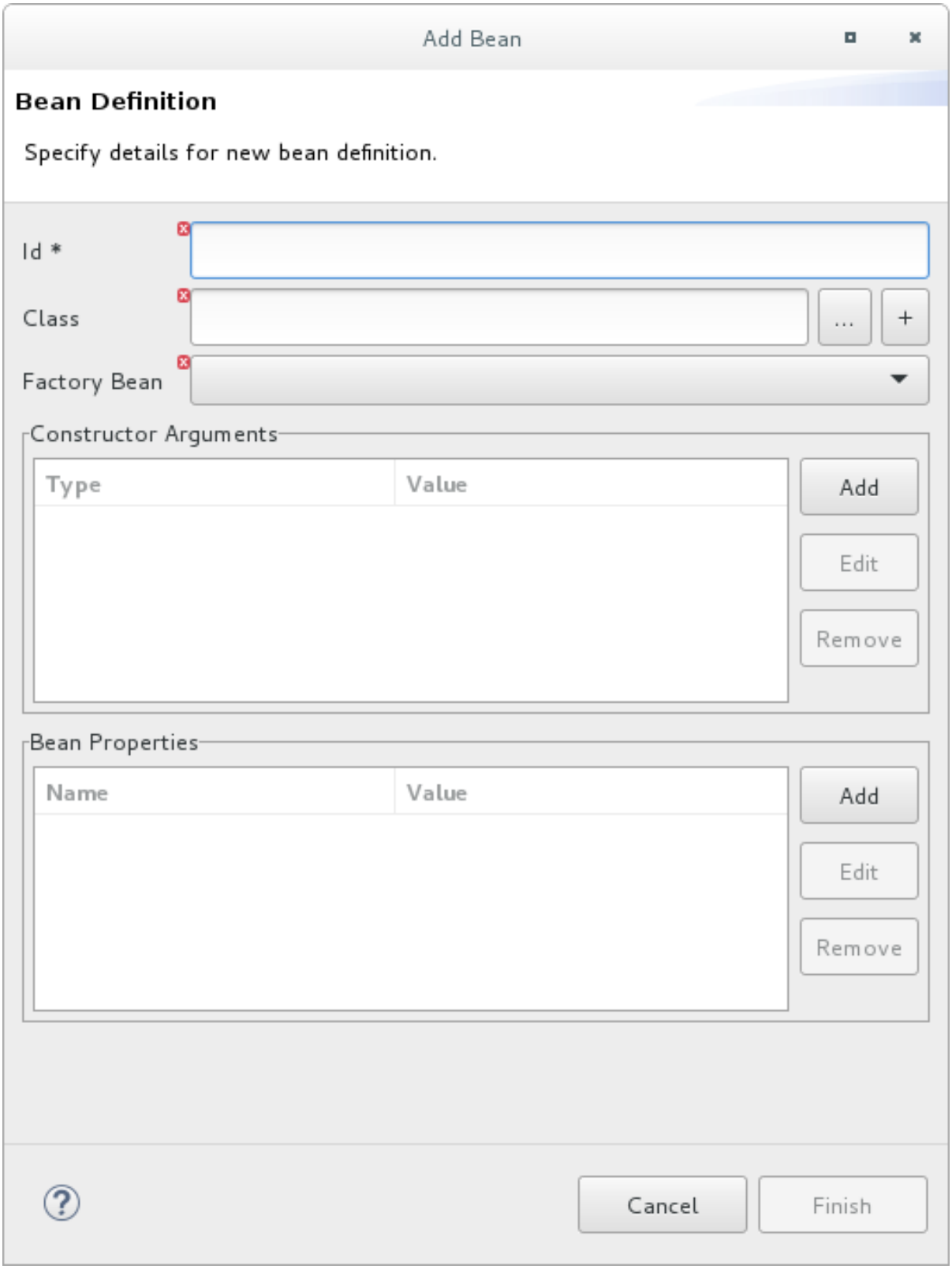
7. **完成后，通过选择 File → Save 来保存您的更改。**

添加全局 bean

全局 bean 启用路由外定义，这些定义可以从路由中的任何位置引用。当您将 Bean 组件从面板复制到路由时，您可以在 Properties 视图的 Ref 下拉菜单中找到定义的全局 Bean。选择您希望 Bean 组件引用的全局 bean。

添加全局 bean 元素：

1. **在 Create a new global element 窗口中，选择 Bean 并点 OK 来打开 Bean Definition 对话框。**



2. **在 Id 字段中输入全局 bean 的 ID，例如 TransformBean。该 ID 在配置中必须是唯一的。**

3.

3.

识别 bean 类或 factory bean。

要指定 factory bean，您必须已添加了另外一个指定了 factory 类的全局 bean。然后，您可以选择该全局 bean 来声明它作为全局 Bean 工厂。bean 工厂类的一个实例将处于运行时。其他全局 Bean 可以调用该类的工厂方法，以创建自己的其他类实例。

要填充 Class 字段，请执行以下操作之一：

- 输入项目或被引用项目中的类名称。
- 点 ... 导航到并选择位于项目或引用的项目中的类。
- 单击 + 以定义新的 bean 类，并将它添加为全局 Bean。

4.

如果要添加的 bean 需要一个或多个参数，请在 Constructor Arguments 部分中为每个参数：

a.

点击 Add。

b.

（可选）在 Type 字段中输入参数的类型。默认值为 java.lang.String。

c.

在 Value 字段中输入 参数的值。

d.

点击 确定。

5.

（可选）指定全局 bean 可以访问的一个或多个属性。在 Bean Properties 部分中，为每个属性执行以下操作：

a.

点击 Add。

b.

b.

在 Name 字段中输入属性的名称。

c.

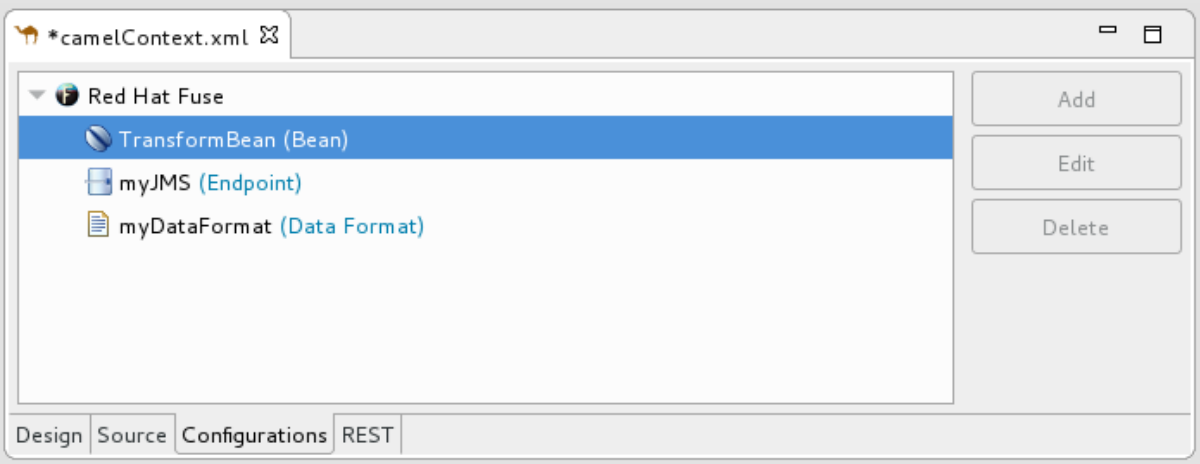
在 Value 字段中输入 属性的值。

d.

点击 确定。

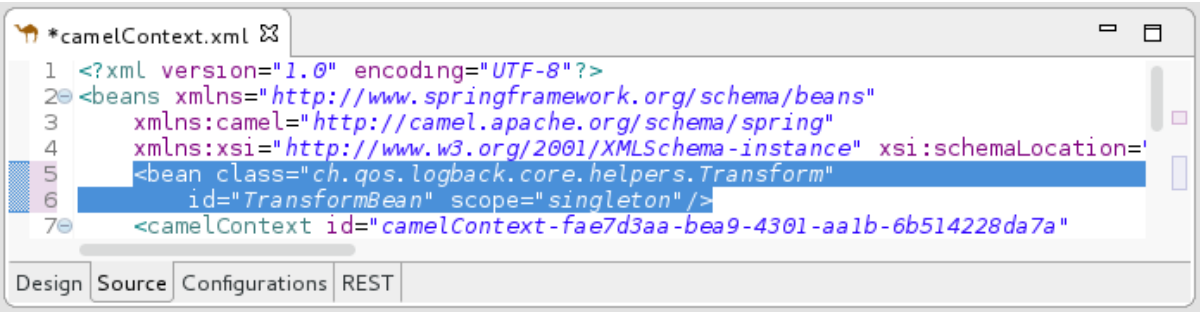
6.

单击 Finish，将全局 bean 添加到配置。您指定的全局 bean ID 会出现在 configuration 选项卡中，例如：



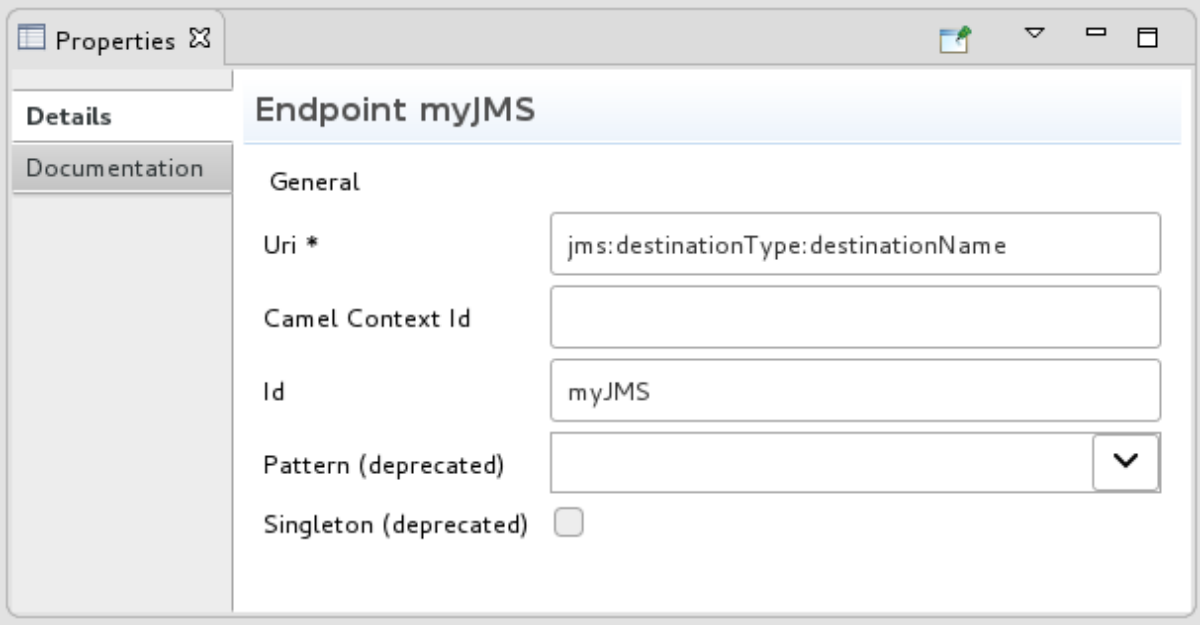
7.

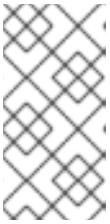
切换到 Source 选项卡，以查看工具添加到上下文文件中的 bean 元素。例如：



8.

点 Configuration 选项卡返回到全局元素列表，并选择全局 bean 在 Properties 视图中显示其标准属性，例如：





注意

要查看或编辑您在添加全局 Bean 时指定的属性，请在 Configuration 选项卡中选择 bean，然后单击 Edit。

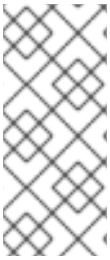
9. **根据需要设置全局 bean 属性：**

- **dependent-on 是一个字符串，可用于识别必须在此全局 Bean 之前创建的 Bean。指定依赖 Bean 的 ID （名称）。例如，如果您要添加 TransformBean，并将 Depends-on 设置为 ChangeCaseBean，则必须创建 ChangeCaseBean，然后可以创建 TransformBean。当 Bean 被销毁时，TransformBean 会首先被销毁。**
- **factory-method 仅在全局 bean 是工厂类时很有用。在这种情况下，指定或选择在引用 bean 时调用的静态工厂方法。**
- **范围 是 单例 或 原型。默认 单例 指示 Camel 每次调用 Bean 时都使用相同的 bean 实例。当您希望 Camel 每次被调用时，指定 bean 的新实例。**
- **init -method 可让您指定或选择在引用 bean 时要调用的 bean 的 init （） 方法。**
- **destroy-method 可让您指定或选择 bean 在 bean 执行处理完成时调用哪个 destory 方法。**

10. 完成后，通过选择 **File** → **Save** 来保存您的更改。

删除全局元素

该流程是删除之前添加到路由上下文中的端点、数据格式或 bean 的过程相同。

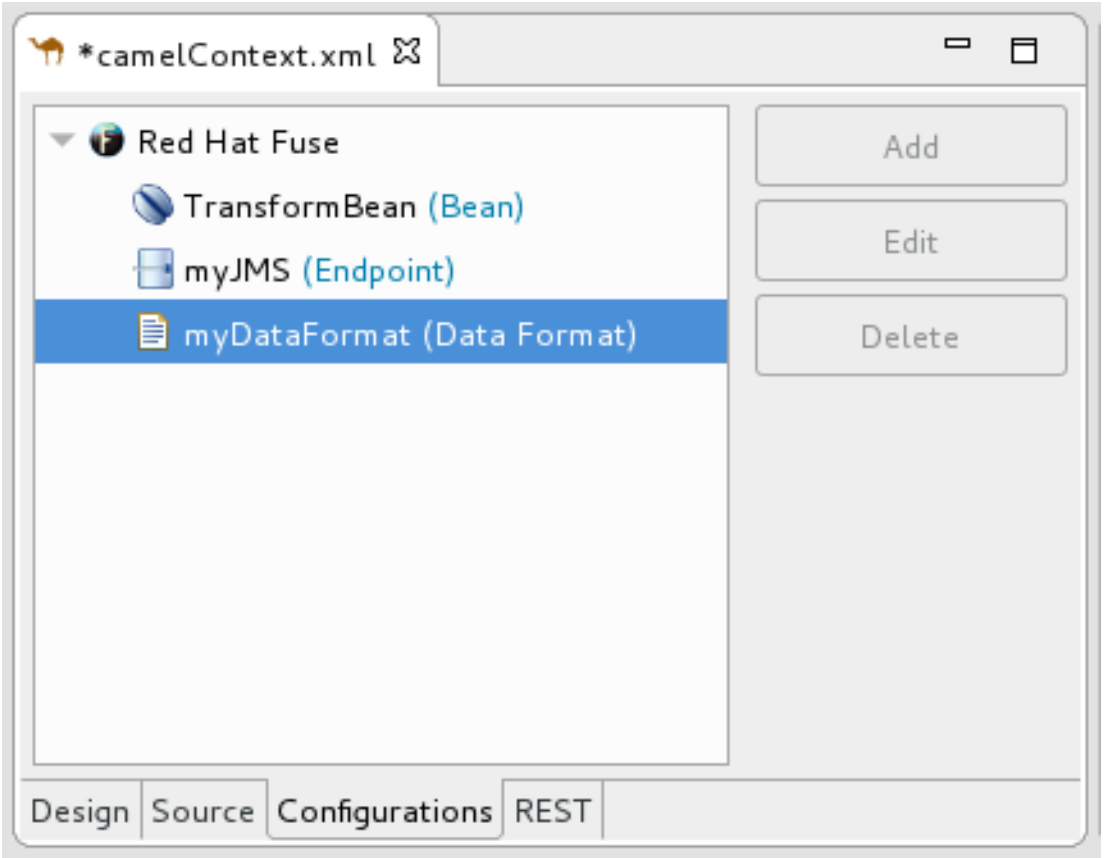


注意

您无法执行撤销操作来删除全局元素。如果您意外删除您要保留在配置中的全局元素，则您可能可以通过关闭上下文文件来撤销删除，而无需保存它。如果这不可行，则重新添加意外删除的全局元素。

1. 在 **Configuration** 选项卡中，选择您要删除的全局元素。

例如，假设您要删除在 “添加全局数据格式”一节 中添加的数据格式 **myDataFormat** ：



2. 点击 **Delete**。

global 元素 myDataFormat 从 Configurations 选项卡中消失。

3. 切换到 Source 选项卡，以检查工具是否从路由上下文中删除了 XML 代码。

4. 完成后，通过选择 File → Save 来保存您的更改。

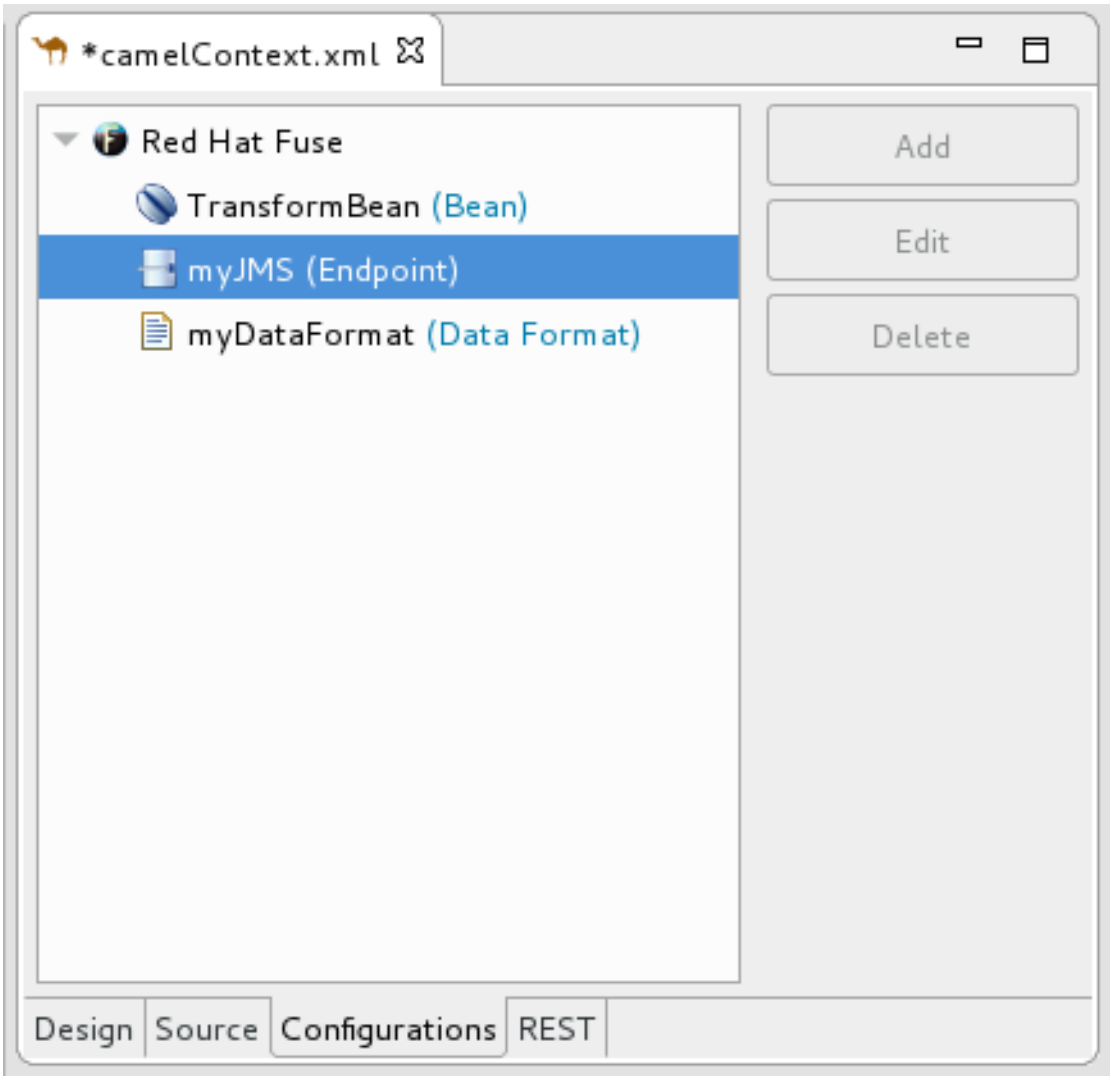
编辑全局元素

无论修改添加到路由上下文中的端点、数据格式或 bean 的属性是相同的。

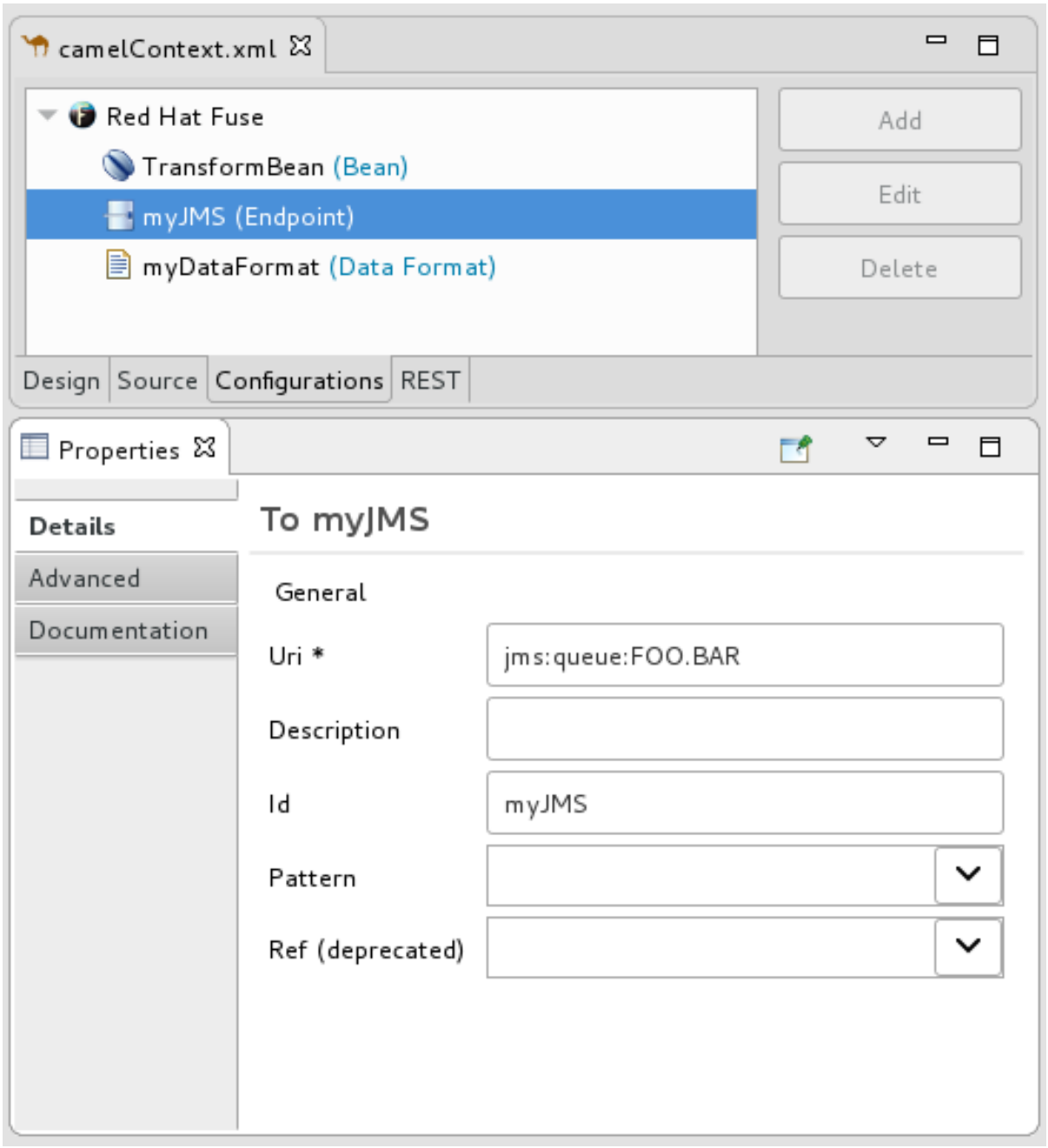
通常，您不想更改全局元素的 ID。如果全局元素已在正在运行的路由中使用，更改 ID 可能会破坏对全局元素的引用。

1. 在 Configuration 选项卡中，选择您要编辑的全局元素。

例如，要编辑 “添加全局端点”一节 中添加的端点 myJMS，请选择它：

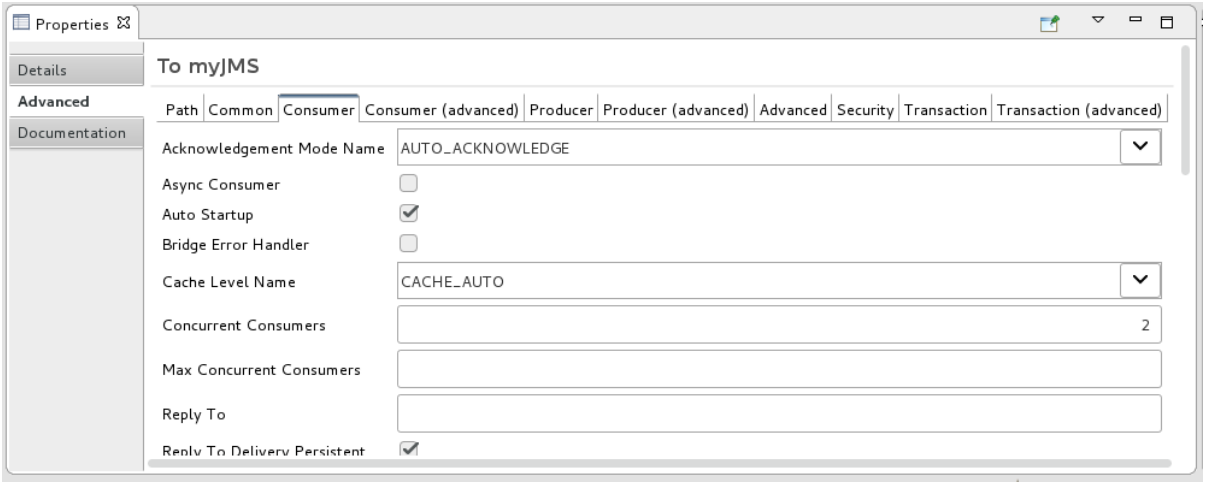

*camelContext.xml
Red Hat Fuse
TransformBean (Bean)
myJMS (Endpoint)
myDataFormat (Data Format)
Add
Edit
Delete
Design
Source
Configurations
REST

2. **点 Edit。**



在 Properties 视图中，根据需要修改元素的属性。

3. **例如，打开 Advanced → Consumer 选项卡，并将 Concurrent Consumers 的值改为 2 ：**



4. **在路由编辑器中，点 Source 选项卡，检查工具将属性 concurrentConsumers=2 添加到路由上下文中：**

5. **完成后，通过选择 File → Save 来保存您的更改。**

2.7. 配置路由编辑器

概述

使用 Fuse preference settings，您可以为路由编辑器的行为和用户界面指定选项：

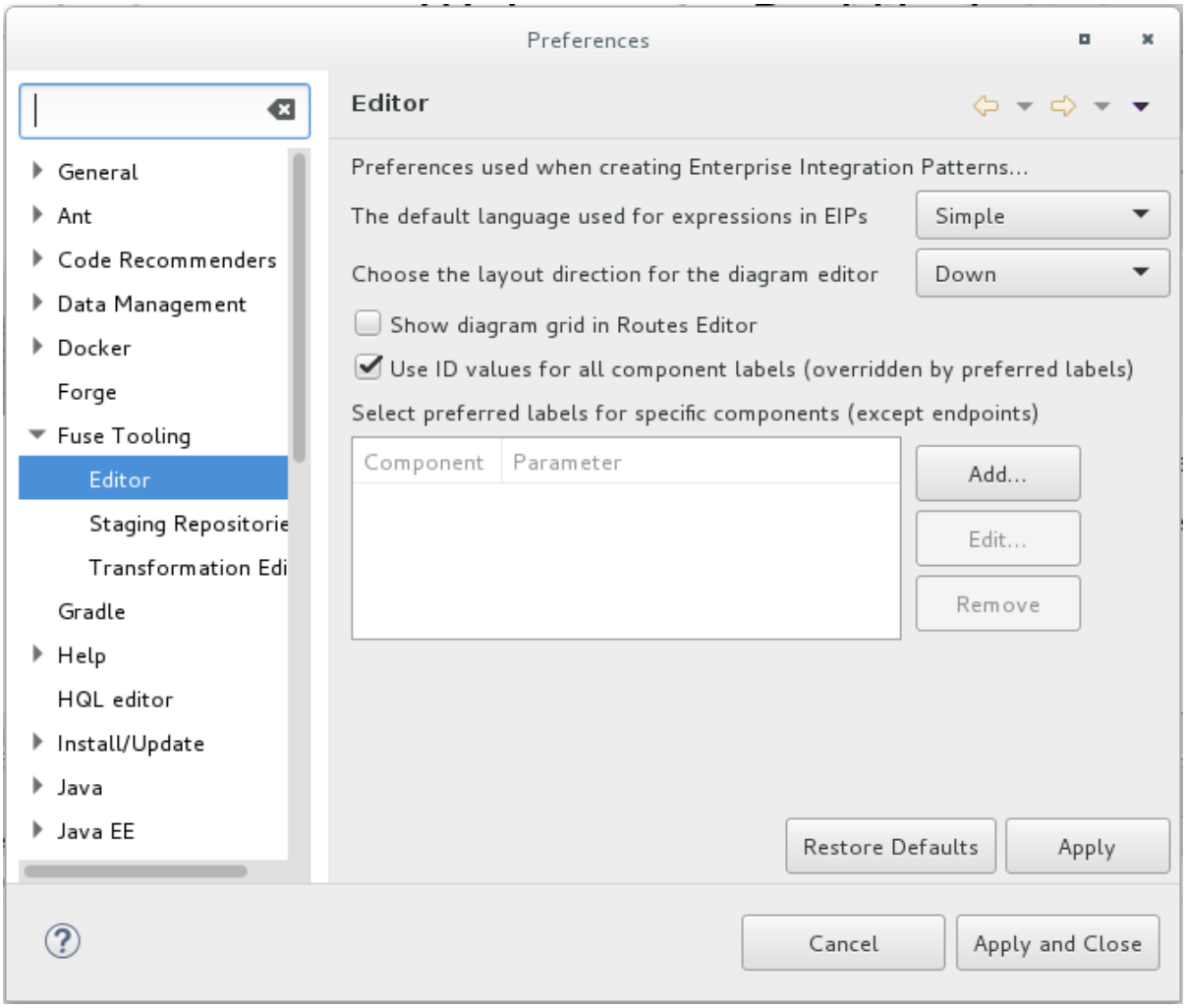
- **在企业集成模式(EIP)中用于表达式的默认语言**

- Design canvas 上的模式流的方向（到右下或向下）在创建路由时
- Design canvas 是否在后台显示网格覆盖。
- 在 Design canvas 中标记节点的方法

流程

配置路由编辑器：

1. 打开 Editor 首选项窗口：
   - 在 Linux 和 Windows 机器上，选择 Windows → Preferences → Fuse Tooling → Editor。



2. **要在企业集成模式(EIP)组件中选择您要用于表达式的默认语言，请从下拉列表中选择一个语言。默认值为 *Simple*。**

3. **要指定您希望路由编辑器在路由中的模式对齐的方向，请选择 Down 或 Right。默认值为 *Down*。**

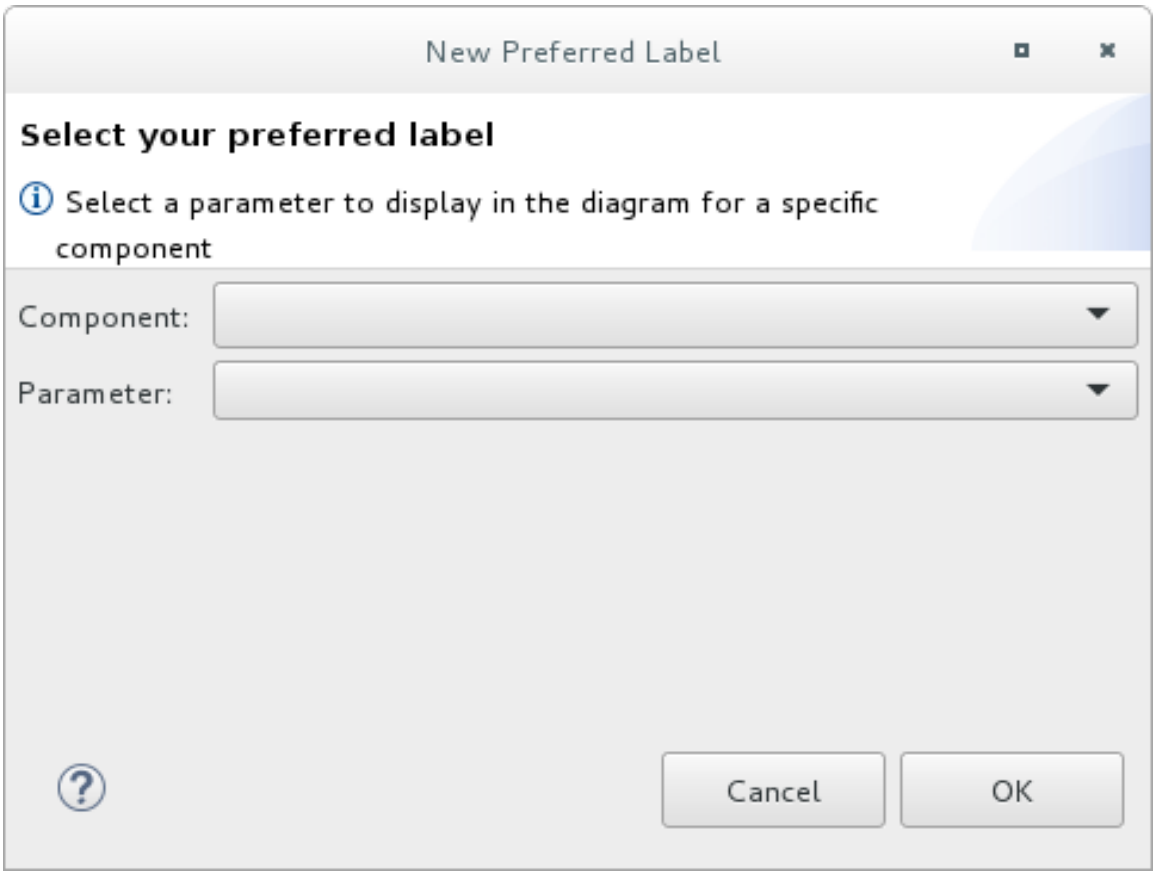
4. **要在 canvas 背景上启用或禁用显示网格覆盖，请在 Routes Editor 中选中 Show diagram grid 旁边的复选框。*默认为启用*。**

5. **要在路由编辑器的 Design 选项卡中启用或禁用使用组件 ID 作为标签，请选中将 ID 值用于组件标签 旁边的框。默认值 *是禁用的*。**

   如果您检查这个选项，同时为组件指定首选标签（请参阅第 6 步），则会为该组件使用首选标签而不是 ID 值。

6. **在路由编辑器的 Design 选项卡中使用参数作为组件标签（端点除外，如文件节点）：**

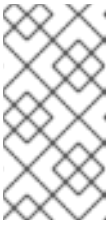
   a. **在 Preferred labels 部分中，单击 Add。此时会打开 New Preferred Label 对话框。**



   b. **选择一个 组件，然后选择 要用作组件标签的参数。**

   c. **点击 确定。组件和参数对在 Editor Preferences 窗口中列出。**

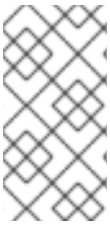
您可以选择 **Edit** 和 **Remove** 组件标签。



注意

如果您检查了 组件标签的 **Use ID** 值，它会适用于所有组件，但 **Preferred labels** 部分中列出的组件除外。

7. 点 **Apply** 和 **Close** 将更改应用到 **Editor** 首选项并关闭 **Preferences** 窗口。



注意

您可以随时恢复路由编辑器的原始默认值，方法是返回到 **Editor** 首选项对话框并单击 **Restore Defaults**。

第 3 章 查看和编辑 REST DSL 组件

Apache Camel 支持多种方法来定义 REST 服务。特别是，Apache Camel 提供 REST DSL （域特定语言），它是一个简单但强大的 API，可以分层在任何 REST 组件上，并提供与 OpenAPI 2.0 规格 的集成。OpenAPI （以前称为 Swagger）是 API 服务的厂商中立且可移植的开放描述格式。

有关使用 Camel Rest DSL 的详情，请参考 Apache Camel 开发指南中的 "定义 REST 服务"章节。

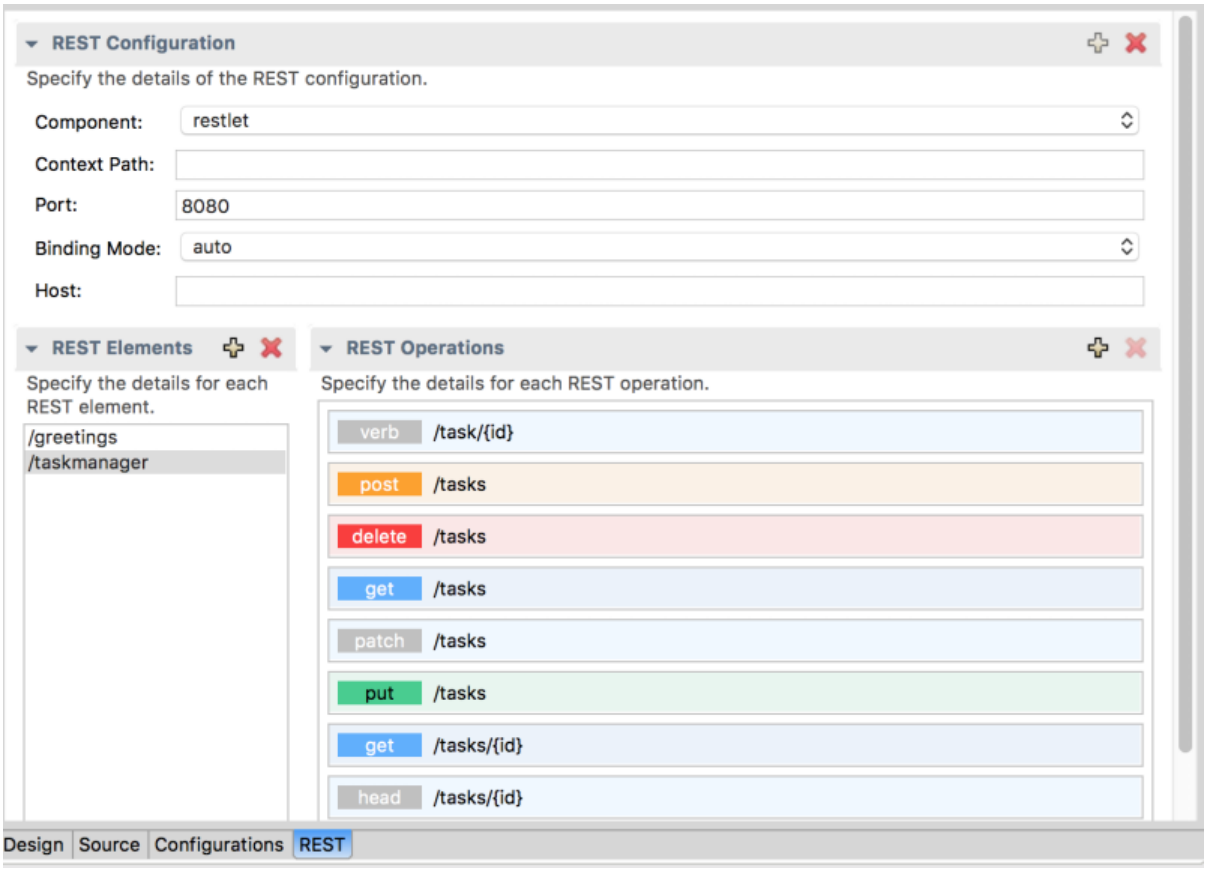
使用 Fuse 工具，您可以查看和编辑 Camel 上下文文件中的 Rest DSL 组件。

您还可以配置 Fuse 集成项目，将 REST API 公开给 OpenAPI 客户端，如下所述：https://access.redhat.com/articles/4296981。

3.1. 查看 REST DSL 组件的图形表示

在图形模式中查看 Camel 上下文文件中的 REST DSL 组件：

1. 在路由编辑器中打开 Camel 上下文文件。
2. 单击 REST 选项卡，以查看 Rest DSL 组件。

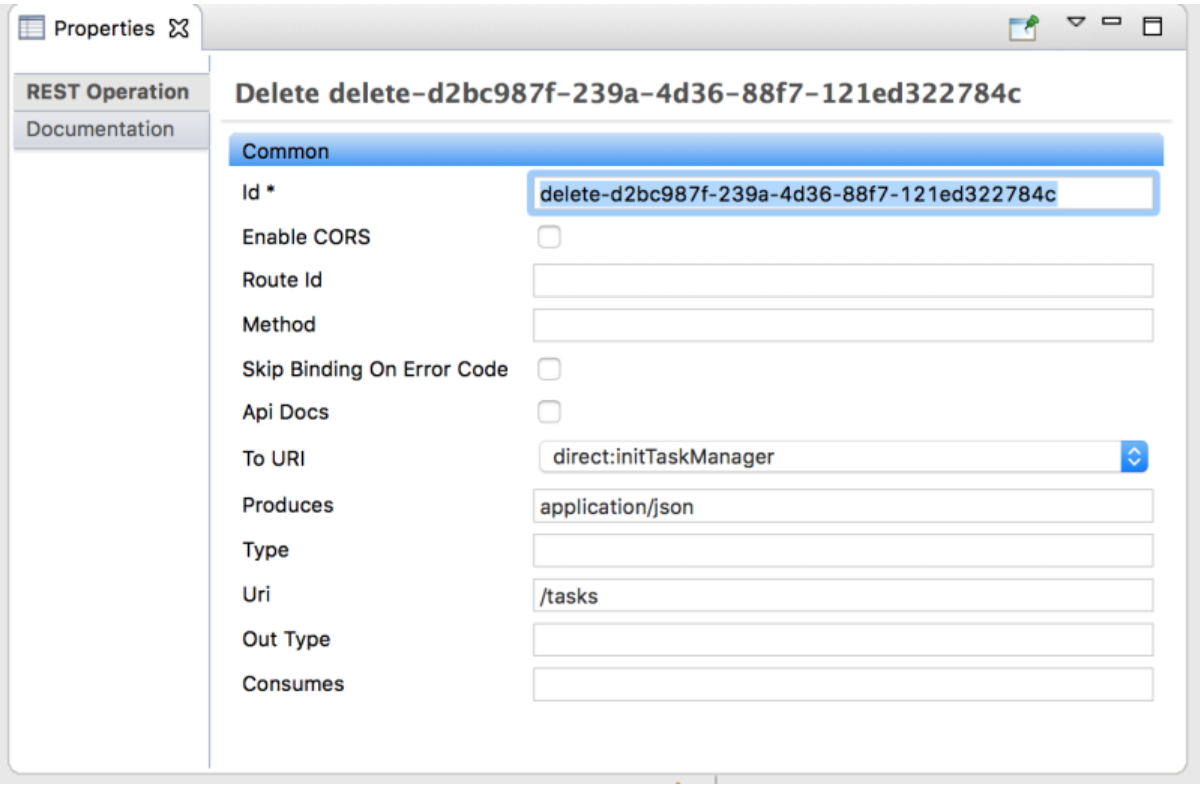


REST Configuration 部分显示这些配置选项：

- **组件 HEKETI-rhacm Camel 组件用于 REST 传输。**
- **上下文路径 HEKETI-wagon REST 服务的主要上下文路径。您可以将这个选项用于组件，如 Servlet，其中使用 context-path 部署 web 应用程序。**
- **端口 HEKETI-rhacm 公开 REST 服务的端口号。**
- **用于 JSON 或 XML 格式的绑定模式。可能的值有： off （默认值）、auto、json、xml 或 json_xml。**
- **主机 HEKETI-rhacm 用于公开 REST 服务的主机名。**

3. **单击 REST 元素，以在 REST 操作 部分中查看其关联的操作（如 GET、POST、PUT 和 DELETE）。**

4. 单击 **REST 元素或 REST 操作，以在 Properties 视图中查看其属性。**

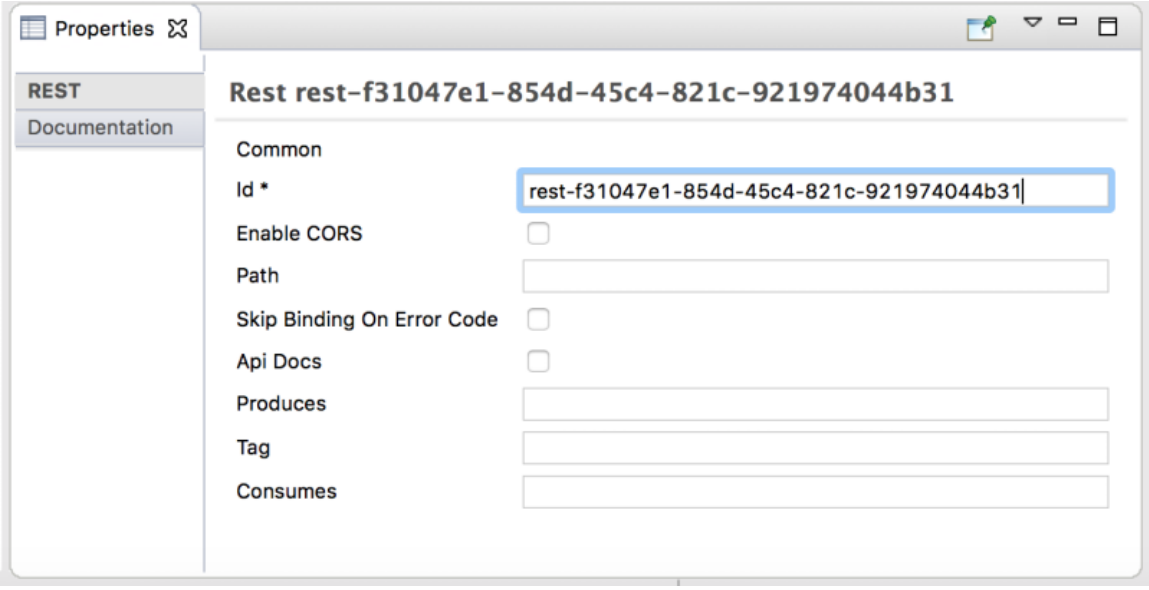


3.2. 在图形视图中编辑 REST DSL 组件

您可以在 REST 选项卡中的项目的 Camel 上下文文件中添加、编辑或删除 REST 元素。

- **添加新 REST 元素：**

  1. **在 REST elements 部分中，单击 + 按钮。REST 元素添加到 REST 元素 列表中。**
  2. **在 Properties 视图中，编辑 REST 元素属性。**

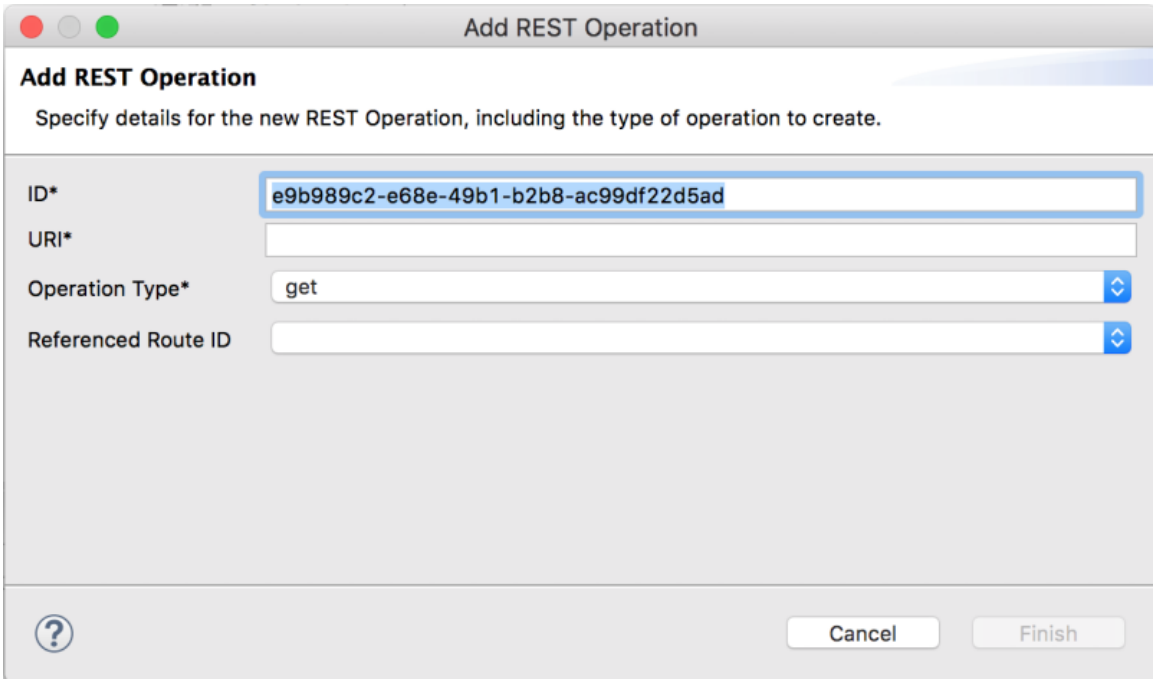


- **在 REST 元素中添加 REST 操作：**

1. **在 REST 元素 列表中，选择一个 REST 元素。**

2. **在 REST 操作 部分中，单击 + 按钮。**

   此时会打开 Add REST Operation 对话框。



3. **指定 ID、URI 和 Operation 类型。（可选）选择一个 参考的路由 ID。**

4. 点 Finish。新操作显示在所选 REST 元素的 REST 操作列表中。

- 要编辑 REST 元素或操作，请在 REST 选项卡中选择它，然后在 Properties 选项卡中编辑其属性值。

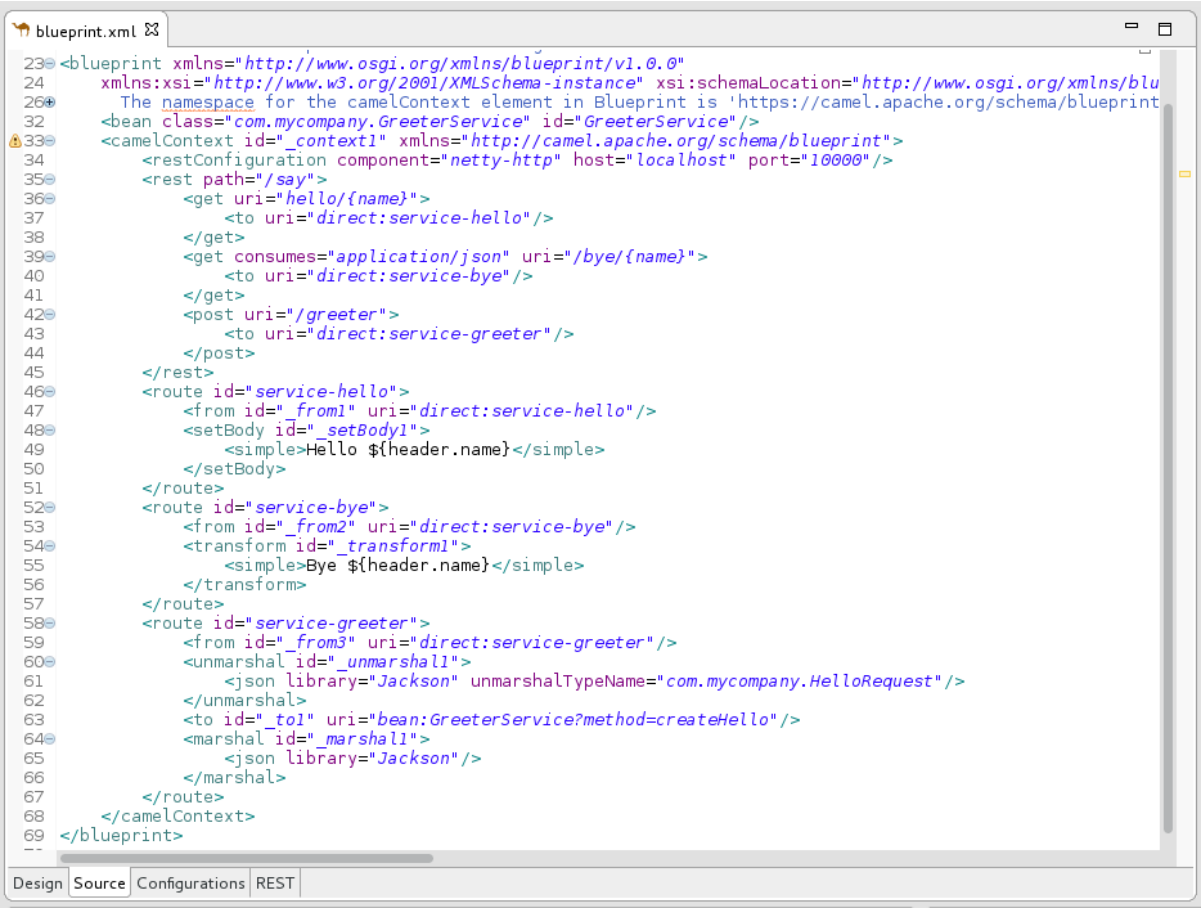
- 要删除所选 REST 元素或操作，请单击 x 按钮。

3.3. 查看和编辑 REST DSL 源代码

您还可以在 Source 选项卡中查看并编辑 Rest DSL 组件：

1. 在路由编辑器中打开 Camel 上下文文件。

2. 单击路由编辑器的 Source 选项卡，然后编辑代码。




```
<blueprint xmlns="http://www.osgi.org/xmlns/blueprint/v1.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.osgi.org/xmlns/blu
      The namespace for the camelContext element in Blueprint is 'https://camel.apache.org/schema/blueprint
    <bean class="com.mycompany.GreeterService" id="GreeterService"/>
    <camelContext id="_context1" xmlns="http://camel.apache.org/schema/blueprint">
        <restConfiguration component="netty-http" host="localhost" port="10000"/>
        <rest path="/say">
            <get uri="hello/{name}">
                <to uri="direct:service-hello"/>
            </get>
            <get consumes="application/json" uri="/bye/{name}">
                <to uri="direct:service-bye"/>
            </get>
            <post uri="/greeter">
                <to uri="direct:service-greeter"/>
            </post>
        </rest>
        <route id="service-hello">
            <from id="_from1" uri="direct:service-hello"/>
            <setBody id="_setBody1">
                <simple>Hello ${header.name}</simple>
            </setBody>
        </route>
        <route id="service-bye">
            <from id="_from2" uri="direct:service-bye"/>
            <transform id="_transform1">
                <simple>Bye ${header.name}</simple>
            </transform>
        </route>
        <route id="service-greeter">
            <from id="_from3" uri="direct:service-greeter"/>
            <unmarshal id="_unmarshal1">
                <json library="Jackson" unmarshalTypeName="com.mycompany.HelloRequest"/>
            </unmarshal>
            <to id="_to1" uri="bean:GreeterService?method=createHello"/>
            <marshal id="_marshal1">
                <json library="Jackson"/>
            </marshal>
        </route>
    </camelContext>
</blueprint>
```

3. （可选）点 REST 选项卡查看图形视图中的更改。

4.

要保存您的更改，请选择 **File → Save。**

第 4 章 创建新的 APACHE CAMEL JUNIT 测试案例

概述

测试路由的常见方法是使用 JUnit。设计时间工具包含一个向导，简化了为您的路由创建 JUnit 测试案例。向导使用您指定的端点来生成测试的起点代码和配置。

注意

创建 boilerplate JUnit 测试案例后，您需要修改它以添加特定于您创建或修改的路由的预期和断言，因此测试对路由有效。

先决条件

在创建新的 JUnit 测试案例前，您需要执行初始任务：

- 如果您要替换现有的 JUnit 测试案例，则需要在创建新测试之前将其删除。请参阅 “删除和现有的 JUnit 测试案例”一节。
- 如果您要在没有登录的项目中创建新的 JUnit 测试案例，您需要首先为构建路径中包含的测试案例创建 *project_root*/src/test/java 文件夹。请参阅 “创建 src/test/java 文件夹并添加到构建路径中”一节。

删除和现有的 JUNIT 测试案例

1. 在 Project Explorer 视图中，展开项目的 root 节点，以公开 < *root_project*>/src/test/java 文件夹。
2. 在 /src/test/java 文件夹中找到 JUnit 测试案例文件。

   根据项目基于的 DSL，JUnit 测试案例文件名为 BlueprintXmlTest.java 或 CamelContextXmlTest.java。
3. 右键单击 JUnit 测试案例 .java 文件，以打开上下文菜单，然后选择 Delete。

JUnit 测试案例 .java 文件会从 Project Explorer 视图中消失。

现在，您可以创建一个新的 JUnit 测试案例。

创建 SRC/TEST/JAVA 文件夹并添加到构建路径中

1. 在 Project Explorer 视图中，右键单击项目的 root 以打开上下文菜单。
2. 选择 New → Folder 以打开 Create a new folder resource 向导。
3. 在向导的项目树窗格中，展开项目的 root 节点，再选择 src 文件夹。

   确保 & *lt;project_root*>/src 出现在 Enter 或选择父文件夹 字段中。
4. 在 Folder name 中，输入 /test/java。此文件夹将存储您创建的新的 JUnit 测试案例。
5. 点 Finish。

在 Project Explorer 视图中，新的 src/test/java 文件夹显示在 src/main/resources 文件夹下。您可以通过打开其上下文菜单并选择 Build Path 来验证此文件夹是否在类路径中。如果 Remove from Build Path 是菜单选项，则您知道 src/test/java 文件夹位于类路径上。

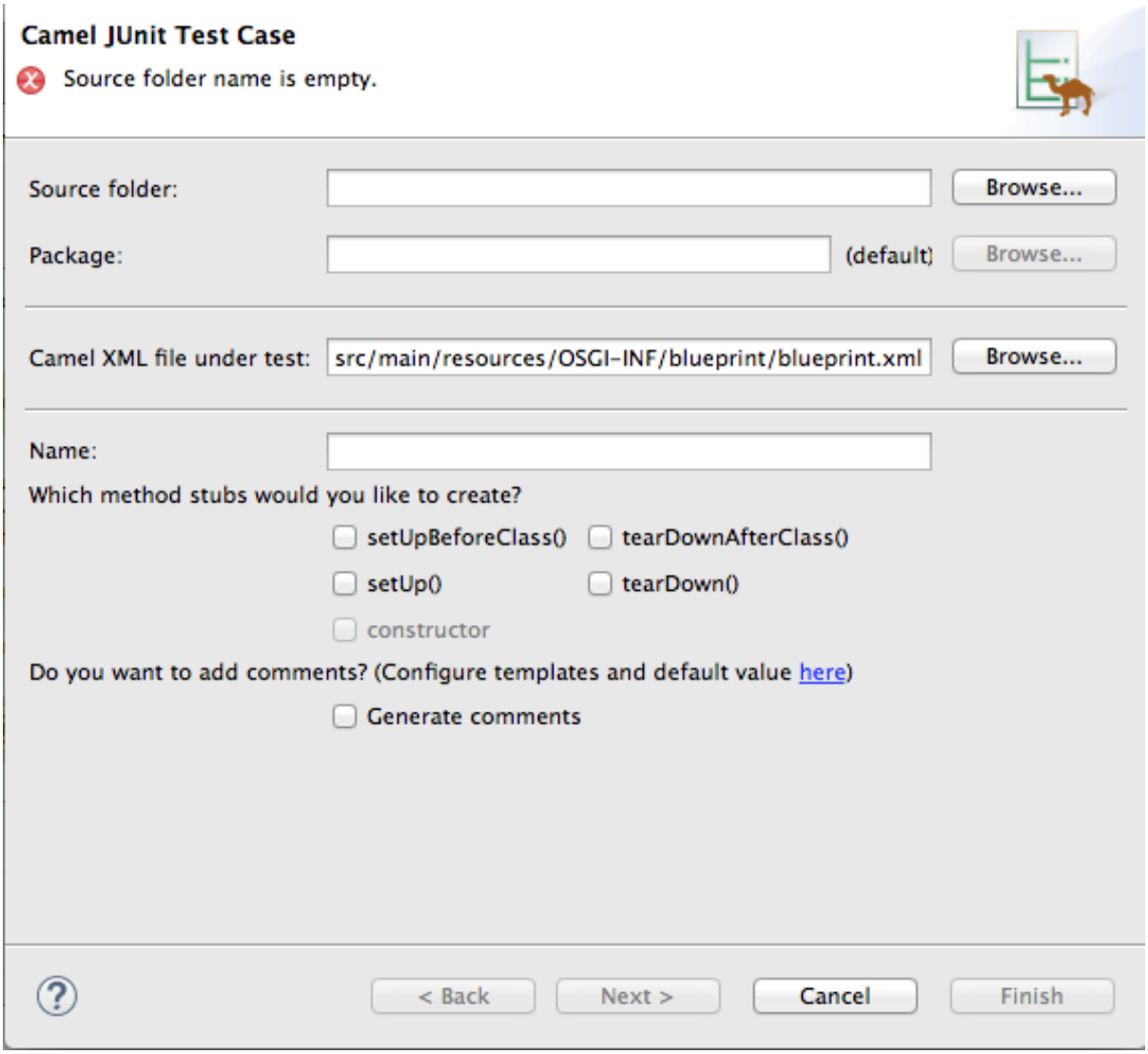
现在，您可以创建一个新的 JUnit 测试案例。

创建 JUNIT 测试案例

为您的路由创建新的 JUnit 测试案例：

1. 在 Project Explorer 视图中，选择项目中的路由上下文 .xml 文件。
2. 右键单击该上下文菜单，然后选择 New → Camel Test Case 以打开 New Camel JUnit Test Case 向导，如 图 4.1 “新的 Camel JUnit Test Case 向导” 所示。

图 4.1. 新的 Camel JUnit Test Case 向导



或者，您可以通过从菜单栏中选择 File → New → Other > Fuse > Camel Test Case 来打开向导。

3. 在 Source 文件夹中，接受测试案例的源代码的默认位置，或者输入其他位置。

   您可以点

   搜索位置。

4. 在 Package 中，接受生成的测试代码的默认软件包名称，或者输入其他软件包名称。

   您可以点

搜索软件包。

5. 在测试 下的 Camel XML 文件中，接受包含您要测试 的路由上下文文件的默认路径名称，或者输入另一个路径名称。

您可以点

搜索上下文文件。

6. 在 Name 中，接受生成的测试类的默认名称，或者输入其他名称。

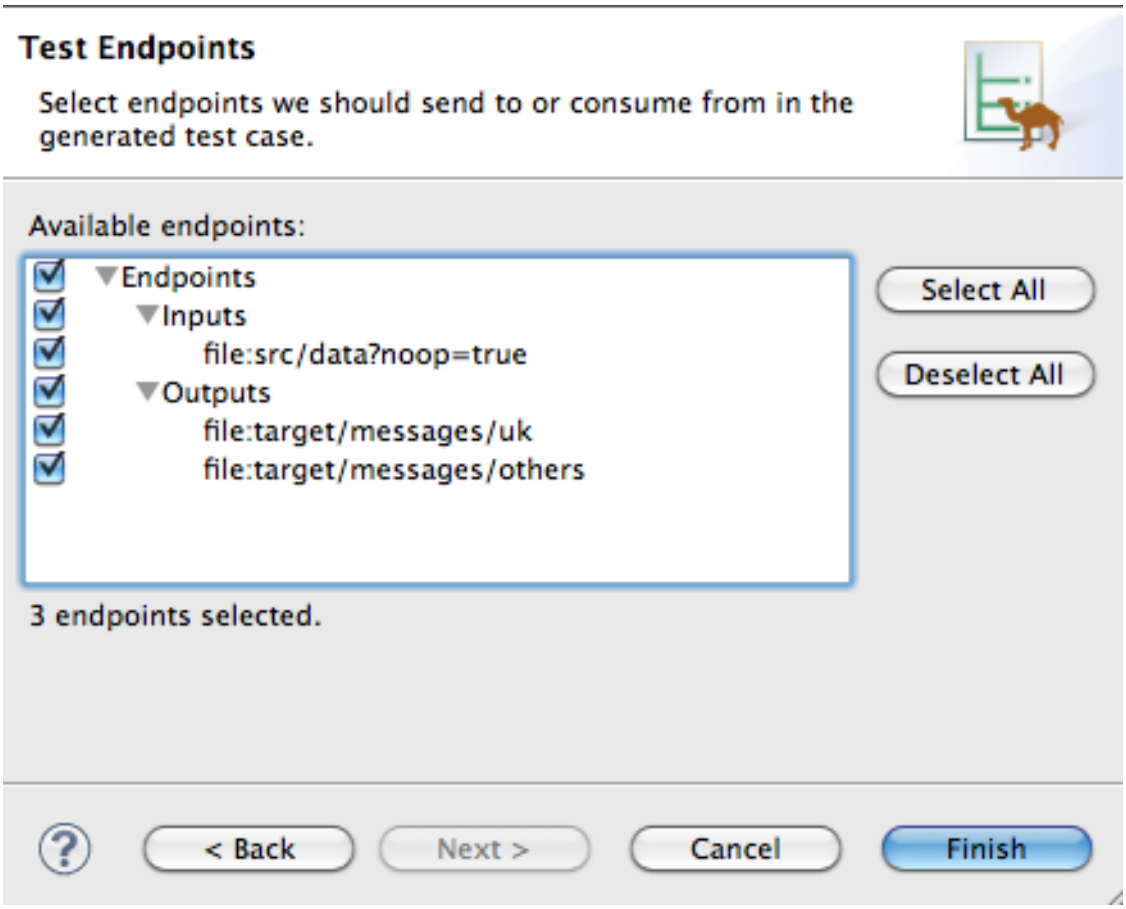
7. 选择您要包含在生成的代码中的方法根。

8. 如果要在生成的代码中包括默认生成的注释，请选中 Generate comments box。

9. 点

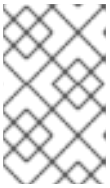
打开 Test Endpoints 页面。例如： 图 4.2 “新的 Camel JUnit Test Case 页面” 显示所选路由的输入和输出文件端点。

图 4.2. 新的 Camel JUnit Test Case 页面



10. 在 Available endpoint 下，选择您要测试的端点。单击任何所选端点旁边的复选框，以取消选择它。

11. 点击 Finish 。



注意

若有提示，请将 JUnit 添加到构建路径中。

测试的工件添加到项目中，并出现在 src/test/java 下的 Project Explorer 视图中。实施测试案例的类在 Java 编辑器中打开。

第 5 章 在红帽 FUSE 工具内运行路由

使用工具运行路由的方法有两种：

- 第 5.1 节 “将路由作为本地 Camel 上下文运行”
- 第 5.2 节 “使用 Maven 运行路由”

5.1. 将路由作为本地 CAMEL 上下文运行

概述

运行 Apache Camel 路由的最简单方法是 本地 Camel 上下文。这个方法可让您直接从 Project Explorer 视图的上下文菜单启动路由。从上下文菜单运行路由时，工具会自动为您创建运行时配置集。您还可以创建自定义运行时配置集来运行路由。

您的路由会像从命令行直接调用一样，并使用 Apache Camel 的嵌入式 Spring 容器。您可以通过编辑运行时配置集来配置多个运行时参数。

流程

要将路由作为本地 Camel 上下文运行：

1. 在 Project Explorer 视图中，选择一个路由上下文文件。
2. 右键单击该上下文菜单，然后选择 Run As → Local Camel Context。

注意

选择 Local Camel Context （没有测试） 指示工具在不执行验证测试的情况下运行项目，这可能更快。

结果

Console 视图显示从 运行路由生成的输出。

相关主题

- 第 5.3.1 节 “编辑本地 Camel 上下文运行时配置集”

5.2. 使用 MAVEN 运行路由

概述

如果包含路由的项目是 Maven 项目，您可以使用 m2e 插件来运行路由。使用这个选项，您可以在路由运行前执行任何 Maven 目标。

流程

使用 Maven 运行路由：

1. 在 Project Explorer 视图中，选择项目 的根目录。
2. 右键单击该上下文菜单，然后选择 Run As → Maven build。
   a. 第一次使用 Maven 运行项目时，会打开 Edit Configuration 和启动 编辑器，以便您可以创建 Maven 运行时配置集。

      要创建运行时配置集，在 Maven 标签页中：

      i. 确保 Apache Camel 项目的路由目录显示在 Base directory: 字段中。

         例如，在 Linux 上，项目的根目录类似于 ~/workspace/simple-router。

      ii. 在 Goals: 字段中，输入 camel:run。

重要

如果您使用 Java DSL 创建项目，请在 Goals: 字段中输入 exec:java。

iii. 单击应用，然后单击 运行。

b. 后续 Maven 运行会使用此配置集，除非您在运行之间修改它。

结果

Console 视图显示 Maven run 的输出。

相关主题

- 第 5.3.2 节 “编辑 Maven 运行时配置集”

5.3. 使用运行时配置集

红帽 Fuse 工具将每个项目的运行时环境信息存储在 *运行时配置集中*。运行时配置集跟踪此类信息，如要调用的 Maven 目标、要使用的 Java 运行时环境、需要设置的任何系统变量等。一个项目可以有多个运行时配置集。

5.3.1. 编辑本地 Camel 上下文运行时配置集

概述

Local Camel 上下文 运行时配置集配置如何调用 Apache Camel 来执行路由。Local Camel 上下文运行时配置文件存储您定义路由的上下文文件的名称、要调用的主要 名称、传递给 JVM 的命令行选项、要使用的类路径、需要设置的任何环境变量以及一些其他信息。

Local Camel 上下文 运行时配置集的运行时配置编辑器包含以下标签页：

- Camel Context File HEKETI-wagonspec 会显示新配置的名称和包含路由上下文文件的完整路径。

- JMX HEKETI-wagonspecifies JMX 连接详情，包括 JMX URI 和要使用的用户名和密码（可选） 用于访问它。

- 主 HEKETI-wagonspec 指定项目基础目录的完全限定名称、几个选项，用于查找基础目录、运行路由前执行的任何目标，以及要使用的 Maven 运行时的版本。

- JRE wagon-wagonspec 在启动 JVM 时要使用的 JRE 和命令行参数。

- 刷新 HEKETI-wagonspec 指示 Maven 在运行终止后如何刷新项目的资源文件。

- 环境 HEKETI-均指定需要设置的任何环境变量。

- 常见 HEKETI-wagonspecs 如何存储配置集以及显示的输出。

第一次将 Apache Camel 路由作为 本地 Camel 上下文 运行时，红帽 Fuse 工具会为路由上下文文件创建默认运行时配置文件，不需要编辑。

访问本地 Camel 上下文运行时配置编辑器

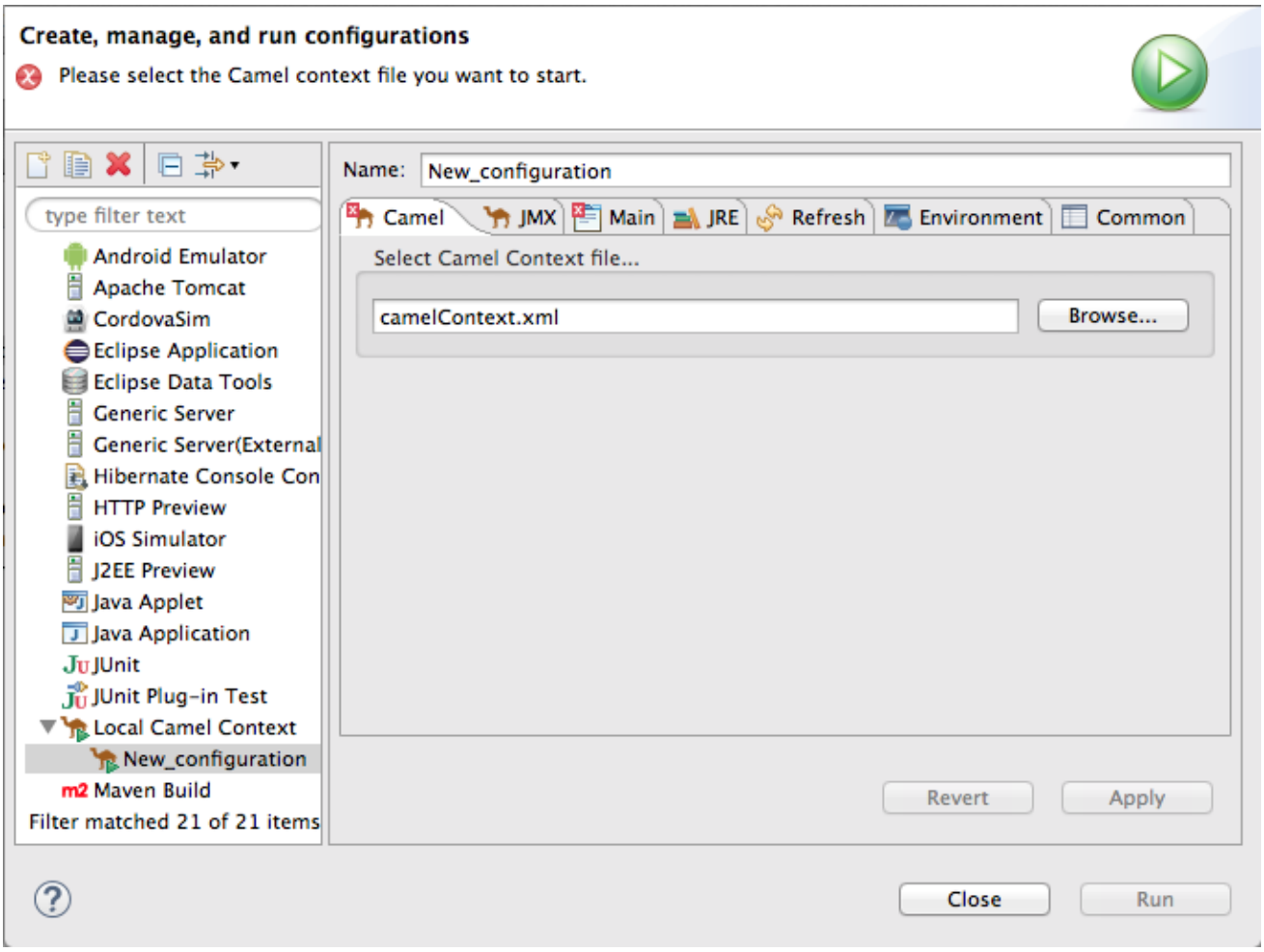
1. 在 Project Explorer 视图中，选择要编辑或创建自定义运行时配置文件的 Camel 上下文文件。

2. 右键单击它以打开上下文菜单，然后选择 Run As → Run Configuration 以打开 Run Configuration 对话框。

3. 在 上下文选择 窗格中，选择 Local Camel Context，然后点击 上下文选择 窗格顶部的
。

4. 在 Name 字段中输入运行时配置集的新名称。

图 5.1. 本地 Camel 上下文的运行时配置编辑器



设置 camel 上下文文件

Camel Context File 选项卡有一个字段，选择 Camel Context file...。输入包含路由定义的路由上下文文件的完整路径。

Browse 按钮访问 Open Resource 对话框，这有助于查找目标路由上下文文件。此对话框已预先配置为搜索包含 Apache Camel 路由的文件。

更改命令行选项

默认情况下，传递给 JVM 的唯一命令行选项是：

```
-fa context-file
```

如果您使用自定义主类，您可能需要传递不同的选项。要做到这一点，在 Main 选项卡中，点 Add 按钮来输入参数的名称和值。您可以点 Add Parameter 对话框的 Variables... 按钮来显示您可以选择的变量列表。

要添加或修改特定于 JVM 的参数，请编辑 JRE 选项卡上的 VM 参数 字段。

更改发送输出的位置

默认情况下，运行路由生成的输出发送到 Console 视图。但是您可以改为将其重定向到文件。

将输出重定向到文件：

1. 选择 Common 选项卡。
2. 在 Standard Input and Output 窗格中，单击 Output File: 字段旁边的复选框，然后输入您要发送输出的文件的路径。

   Workspace、文件系统 和 Variables 按钮有助于构建到输出文件的路径。

相关主题

- 第 5.1 节 “将路由作为本地 Camel 上下文运行”

5.3.2. 编辑 Maven 运行时配置集

概述

Maven 运行时配置集配置 Maven 如何调用 Apache Camel。Maven 运行时配置集存储要执行的 Maven 目标、要使用的 Maven 配置文件、要使用的 Maven 版本、要使用的 JRE、类路径、需要设置的任何环境变量以及其它部分信息。

重要

第一次使用 Maven 运行 Apache Camel 路由时，您必须为它创建一个默认运行时配置集。

Fuse 运行时配置集的运行时配置编辑器包含以下标签页：

-

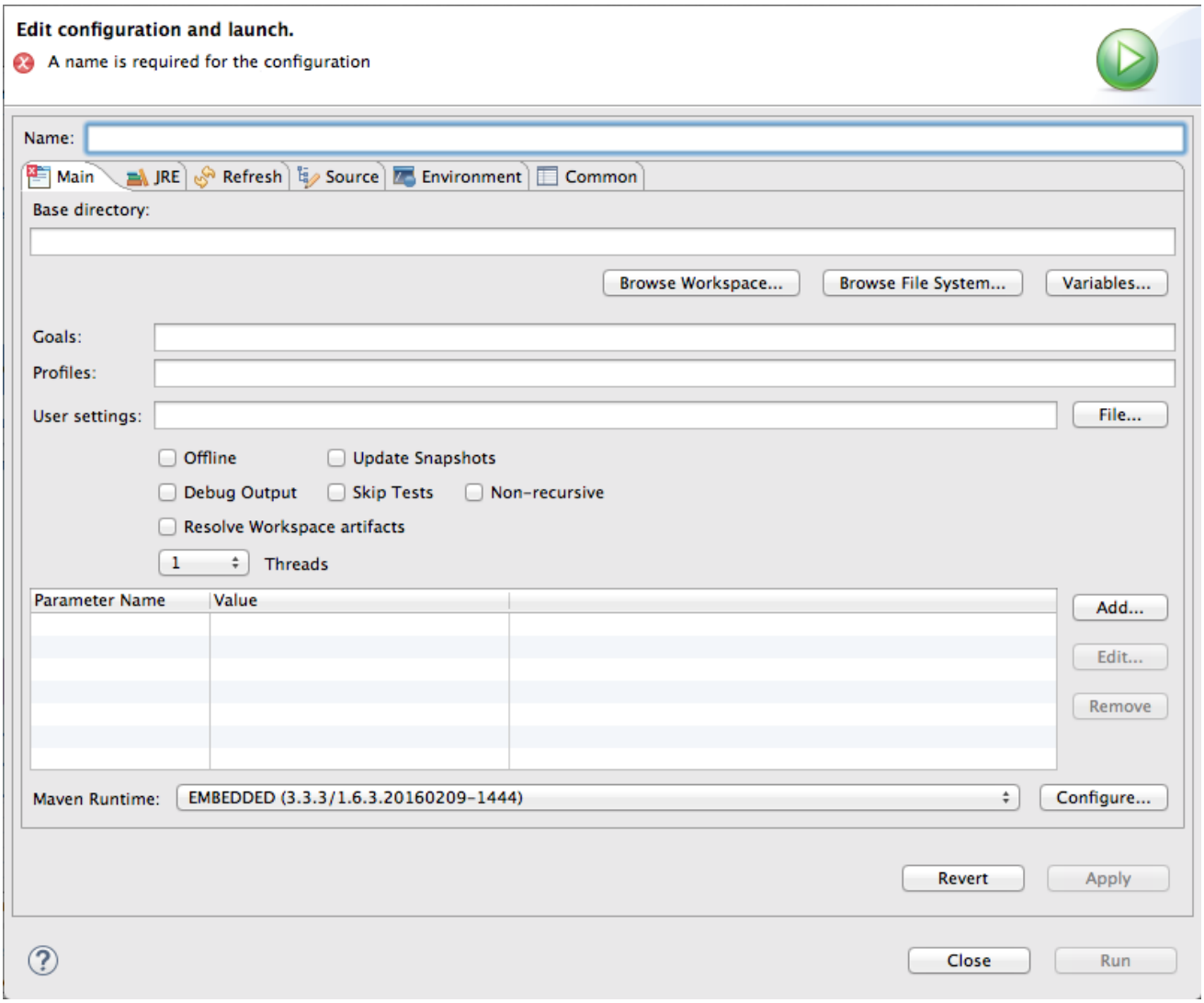
主 rhacm 检查新配置的名称、项目基础目录的完全限定名称、查找基础目录的几个选项、运行路由前需要执行的任何目标以及要使用的 Maven 运行时版本。

- JRE wagon-wagonspec 在启动 JVM 时要使用的 JRE 和命令行参数。
- 刷新 HEKETI-wagonspec 指示 Maven 在运行终止后如何刷新项目的资源文件。
- Source HEKETI-wagonspec 用来指定项目所需的任何其他源的位置。
- 环境 HEKETI-均指定需要设置的任何环境变量。
- 常见 HEKETI-wagonspecs 如何存储配置集以及显示的输出。

访问 Maven 运行时配置编辑器

1. 在 Project Explorer 视图中，选择要编辑或创建自定义运行时配置集的项目的根目录。
2. 右键单击它以打开上下文菜单，然后选择 Run As → Run Configuration 以打开 Run Configuration 对话框。
3. 在 上下文选择 窗格中，选择 Maven Build，然后单击 上下文选择 窗格左上角的  。

图 5.2. Maven 的运行时配置编辑器



更改 Maven 目标

运行路由时最常用的目标为 **camel:run**。它将路由加载到在其自己的 JVM 中运行的 Spring 容器中。

Apache Camel 插件还支持 camel:embedded 目标，它将 Spring 容器加载到 Maven 使用的同一 JVM 中。这样做的好处是路由应该更快地引导。

基于 Java DSL 的项目使用 exec:java 目标。

如果您的 POM 包含其他目标，可以通过单击 Main 选项卡中的 Maven Runtime 字段旁边的 Configure... 按钮来更改所用的 Maven 目标。在 Installations 对话框中，您可以编辑 <selected_runtime> 安装字段的 Global 设置。

更改 Maven 的版本

默认情况下，Red Hat Fuse Tooling for Eclipse 使用 m2e，它嵌入在 Eclipse 中。如果要使用其他版本的 Maven，或者在开发机器上安装更新的版本，您可以从 Main 选项卡中的 Maven Runtime 下拉菜单中选择它。

更改发送输出的位置

默认情况下，路由执行的输出发送到 Console 视图。但是您可以改为将其重定向到文件。

将输出重定向到文件：

1. 选择 Common 选项卡。
2. 单击 Output File: 字段旁边的复选框，然后输入您要发送输出的文件的路径。

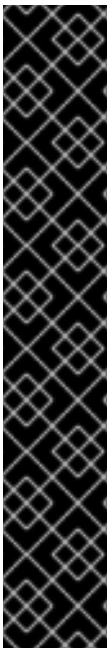
   Workspace、文件系统 和 Variables 按钮有助于构建到输出文件的路径。

相关主题

- 第 5.2 节 “使用 Maven 运行路由”

第 6 章 在 OPENSHIFT 中使用 FUSE

OpenShift 上的 Fuse （自 7.0 起的 Fuse 集成服务的名称）可让您在 OpenShift Container Platform 上部署 Fuse 应用程序。



重要

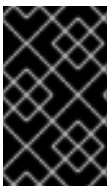
对于 Fuse 集成项目（在 OpenShift 项目上使用），Fuse 工具需要安装 Red Hat Container Development Kit (CDK) v3*x*。具体步骤请查看 入门指南。除了本指南中指定的先决条件外，如果您没有帐户，您需要建立红帽帐户。启动 Red Hat Container Development Kit 中提供的虚拟 OpenShift 实例，需要您的红帽用户名和密码。

在红帽客户门户上 注册，您可以轻松地获得帐户。点白色横幅右上角的

，然后点击 Login to your Red Hat Account 页的 

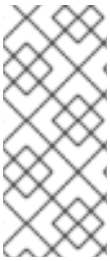
。

Fuse 工具可让您使用 s2i 二进制工作流开发和部署 Fuse 集成项目。在此工作流中，工具会在本地构建项目，将其编译到镜像流中，然后将镜像流推送到 OpenShift，在其中构建 Docker 容器。构建 Docker 容器后，OpenShift 会在 pod 中部署。



重要

Fuse 工具只适用于 S2I 二进制工作流，且仅适用于基于 Spring Boot 框架的项目。



注意

虽然 Fuse 工具可以使用工具将创建的 Fuse 集成项目部署到远程 OpenShift 服务器，但本章描述了创建和部署 Fuse 集成项目到虚拟 OpenShift 实例，使用 Red Hat Container Development Kit (CDK) v3.*x* 安装。

以下小节介绍了如何创建和部署第一个 Fuse 集成项目：

- 第 6.1 节 “添加 Red Hat Container Development Kit 服务器”

- 第 6.2 节 “启动容器开发环境(CDE)和虚拟 OpenShift 服务器”
- 第 6.3 节 “创建新的 OpenShift 项目”
- 第 6.4 节 “创建新的 Fuse 集成项目”
- 第 6.5 节 “将 Fuse 集成项目部署到 OpenShift”

注意

您还可以将 Fuse Integration 项目作为本地 Camel 上下文运行，查看 第 5.1 节 “将路由作为本地 Camel 上下文运行”，然后在 JMX Navigator 视图中连接它，您可以在其中监控并测试路由上下文。您还可以在 Fuse Integration 项目(第 II 部分 “调试路由上下文”)上运行 Camel debugger，以公开和修复路由上下文中的任何逻辑错误。

6.1. 添加 RED HAT CONTAINER DEVELOPMENT KIT 服务器

将 Red Hat Container Development Kit 添加到 Servers 视图中：

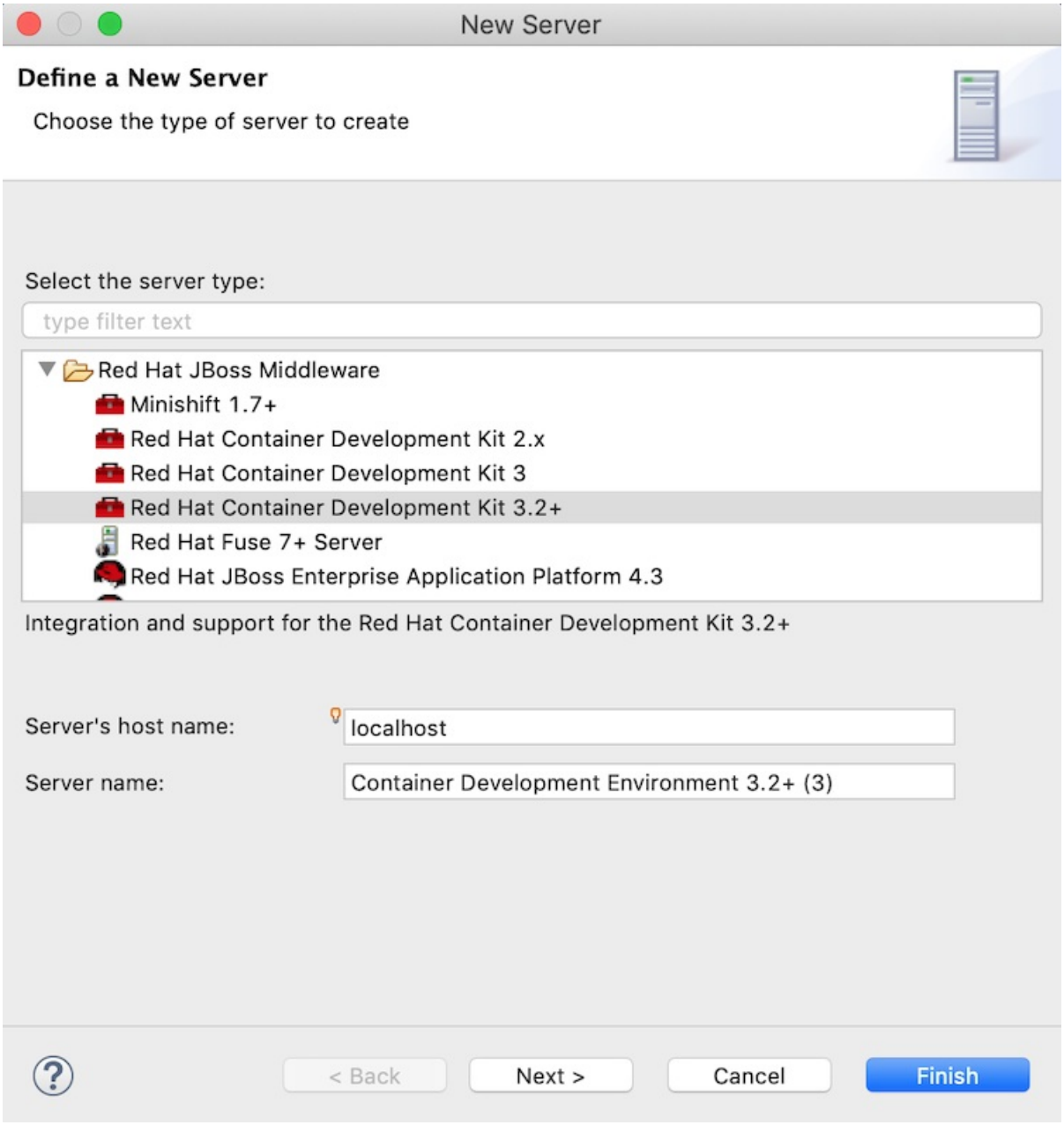
1. 如有必要，选择 Window → Perspective → Open Perspective → Fuse Integration 来切换到 Fuse Integration。

注意

如果没有打开这个过程中描述的视图，您可以选择 Window → Show View → Other，然后选择您要打开的视图的名称。

2. 在 Servers 视图中，点链接 No servers are available。点这个链接来创建新 server... 以打开 Define a New Server 向导。只有 Servers 视图不包含服务器条目时，才会显示此链接。

   否则，在 Servers 视图中右键单击以打开上下文菜单，然后选择 New → Server 以打开 Define a New Server 向导。



3. 选择 **Red Hat JBoss Middleware → Red Hat Container Development Kit 3.2+**。

   接受以下的默认值：

   - **服务器的主机名 HEKETI - localhost**
   - **服务器名称 HEKETI - Container Development 环境**

4. **点 Next 以打开 Red Hat Container Development Environment 页面。**

5.

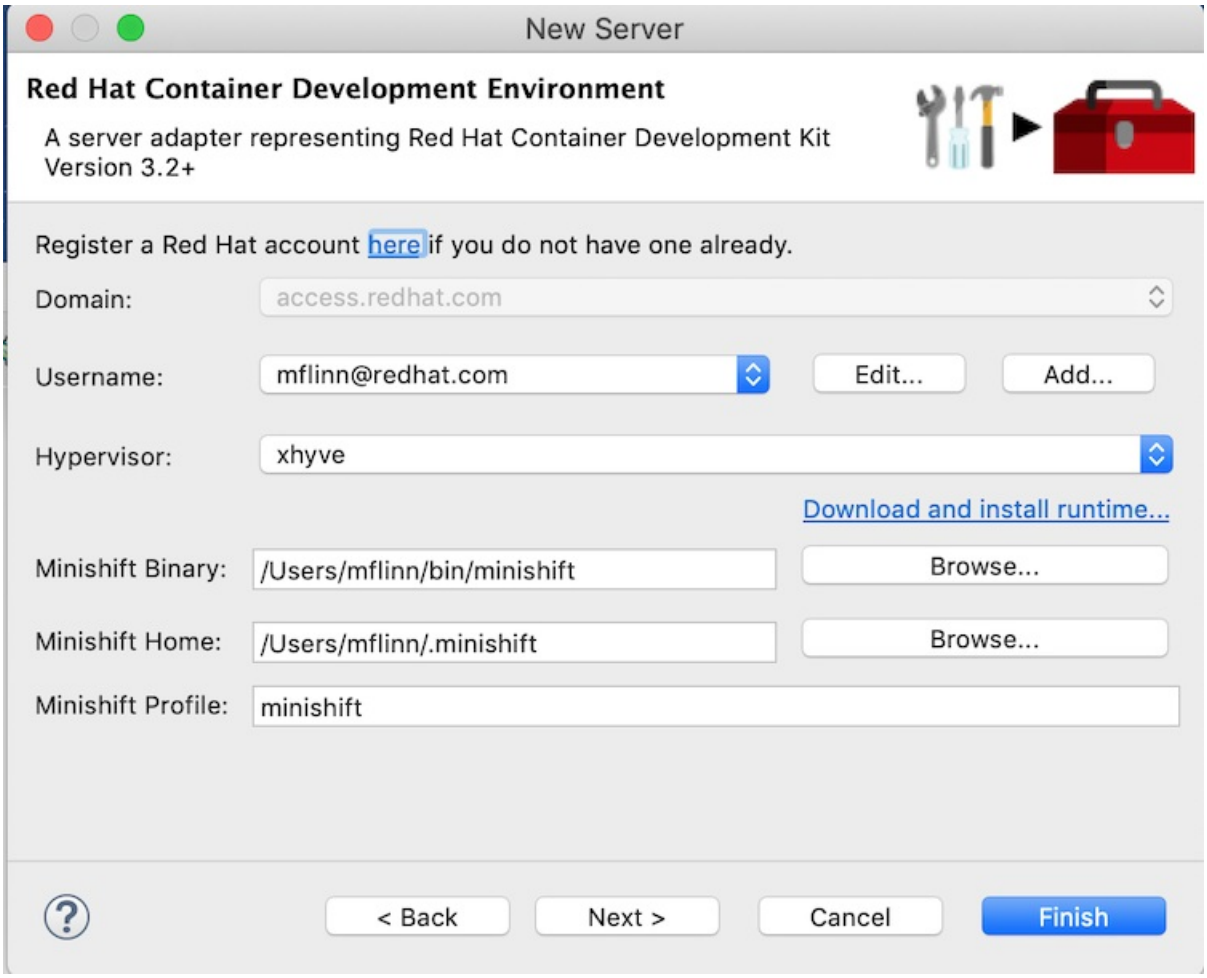
在 MiniShift Binary 旁边，单击 Browse，进入安装 Red Hat Container Development Kit 3.*x* 的位置，然后单击 Open。

6. 在 Username 旁边，单击 Add 以打开 Add a Credential 页面。

7. 以这种方式设置凭证：

   - 用户名 HEKETI-rhacmEnter，用于登录到您的红帽帐户的名称。
   - 始终提示输入密码时 （禁用）
   - 密码 HEKETI-rhacmEnter，用于登录到您的红帽帐户的密码。

8. 单击 OK 以返回到 Red Hat Container Development Environment 页面，该页面现已填充。例如：



9. 点 Finish。容器开发环境 3.2+ [Stopped, Synchronized] 会出现在 Servers 视图中。当您添加 CDK 3.*x* 服务器时，容器开发环境 3.2+ 是默认的服务器名称。

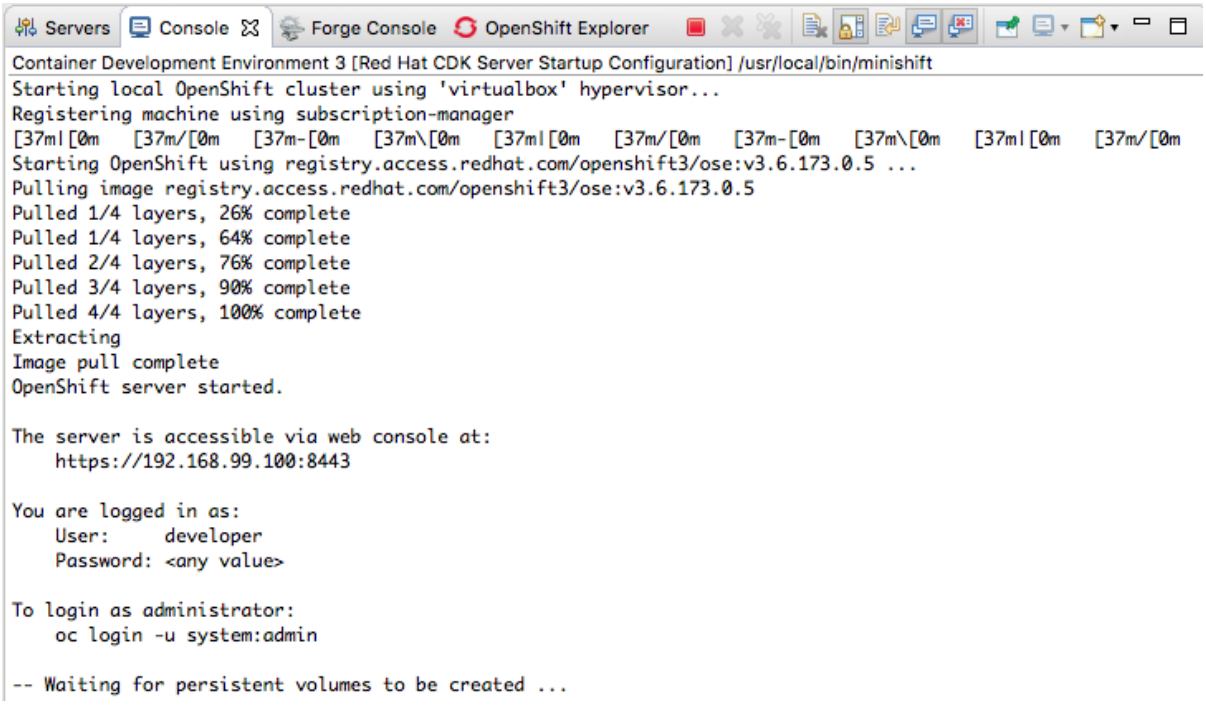
6.2. 启动容器开发环境(CDE)和虚拟 OPENSHIFT 服务器

启动容器开发环境(CDE)也会启动虚拟 OpenShift 服务器。停止 CDE 也会停止虚拟 OpenShift 服务器。

1. 在 Servers 视图中，选择 Container Development Environment 3 [stopped, Synchronized]，然后点击 Servers 菜单栏中的

。

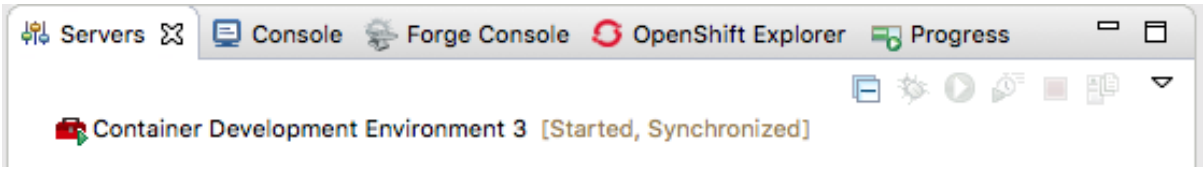
控制台 视图会打开并显示启动过程的状态：



注意

在初始启动时，CDE 会询问您是否接受不受信任的 SSL 证书。单击 Yes 。

启动过程完成后，服务器 视图会显示：



2. 切换到 OpenShift Explorer 视图。

   虚拟 OpenShift 服务器实例 developer 也在运行：

   https://192.168.99.100:8443 是 OpenShift 开发人员 Web 控制台的 URL 示例。您的安装会显示实例的 URL。如需了解更多详细信息，请参阅 第 6.6 节 “访问 OpenShift Web 控制台”。

6.3. 创建新的 OPENSHIFT 项目

将 Fuse Integration 项目部署到 OpenShift 时，它将发布到您创建的 OpenShift 项目中。

1. 在 OpenShift Explorer 视图中，右键单击 developer 条目，以打开上下文菜单。

2. 选择 New → Project 以打开 New OpenShift Project 向导。

3. 以这种方式设置新项目的属性：

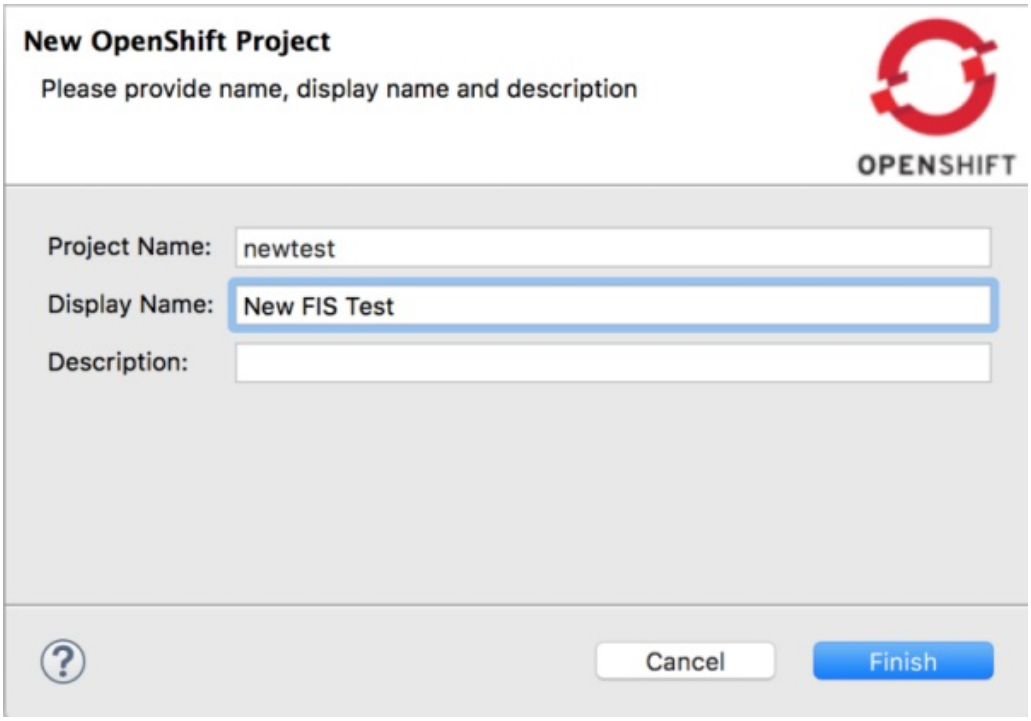
   - 在 Project Name 字段中输入虚拟 OpenShift 服务器上项目命名空间的名称。

     只有小写字母、数字和短划线有效。

   - 在 Display Name 字段中输入要在虚拟 OpenShift Web 控制台的 Overview 页面中显示的名称。

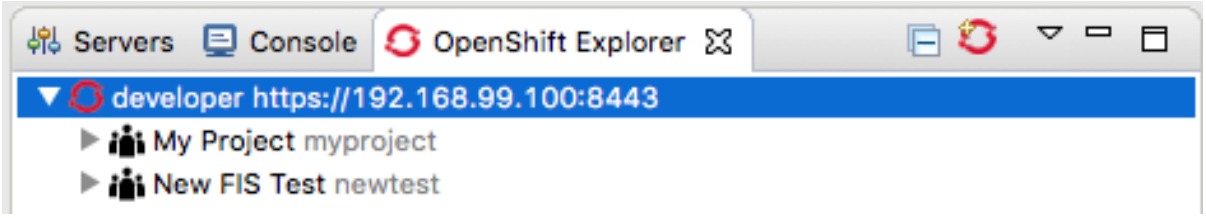
   - Description 字段保留原样。

例如：



4. 点 Finish。

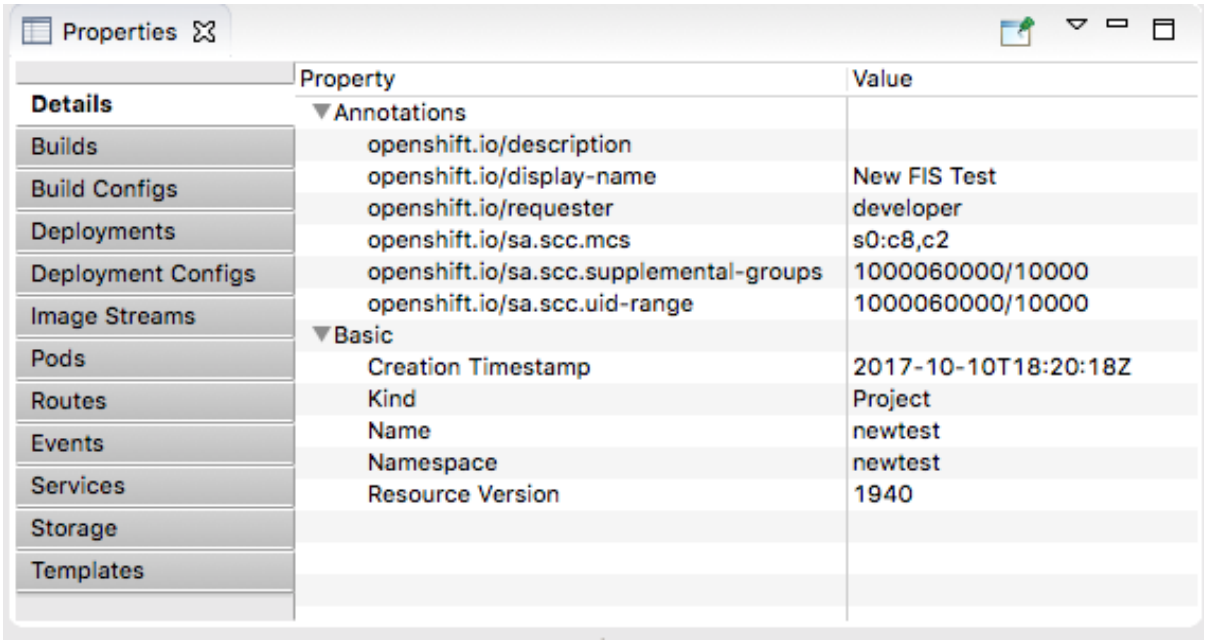
新的 OpenShift 项目（本例中为 New FIS Test newtest）会出现在 OpenShift Explorer 选项卡中，在本例中为 developer https://192.168.99.100:8443 ：



注意

myproject myproject 是 OpenShift 中包含的初始示例项目。

在 OpenShift Explorer 视图中选择了 New FIS Test newtest 时，Properties 视图会显示项目的详细信息。例如：



注意

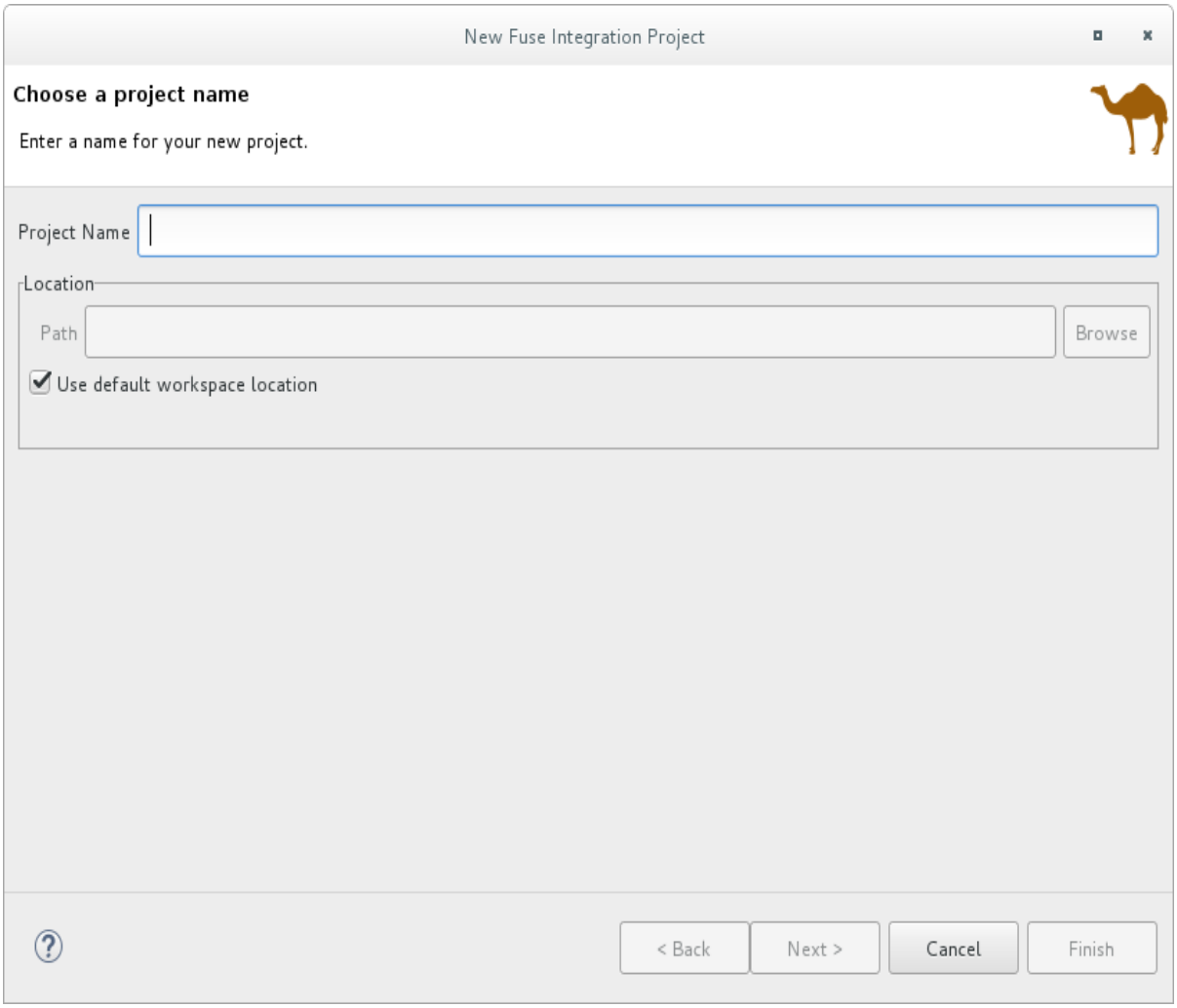
将项目部署到 OpenShift 时，Properties 视图会收集并显示与 OpenShift Web 控制台执行的项目相同的信息。

6.4. 创建新的 FUSE 集成项目

在创建新的 Fuse 集成项目前，您应该启用暂存存储库。这是必要的，因为一些 Maven 工件不在默认的 Maven 存储库中。要启用暂存存储库，请选择 Window → Preferences → Fuse Tooling → Staging Repositories。

要创建 Fuse Integration 项目，请使用 OpenShift 模板上的 Spring Boot ：

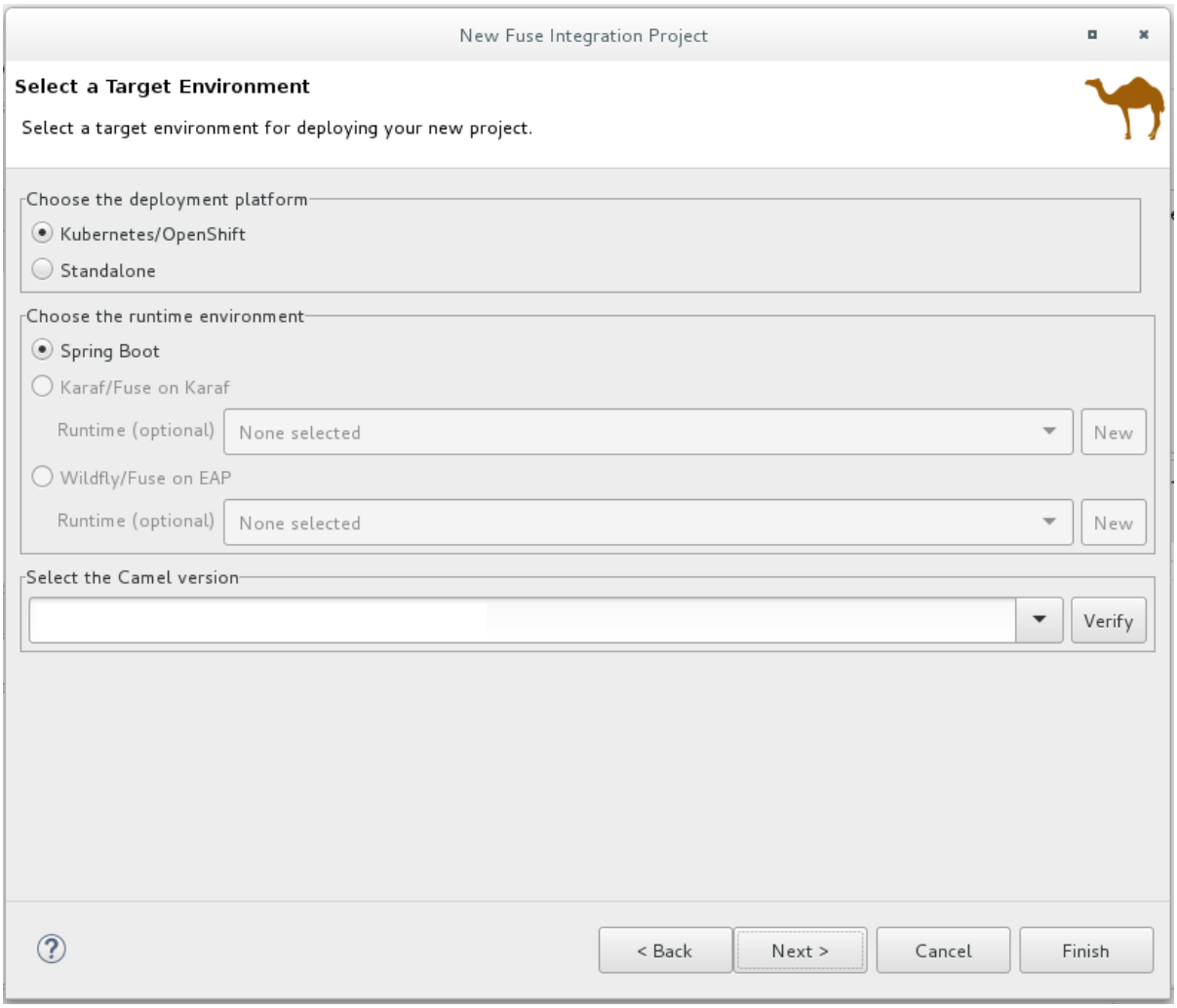
1. 在 Project Explorer 视图中，右键单击打开上下文菜单，然后选择 New → Fuse Integration Project 以打开向导的 Choose a project 名称 页面：



2. **在 Project Name 字段中，输入您正在使用的工作区独有的名称，例如 myFISproject。**

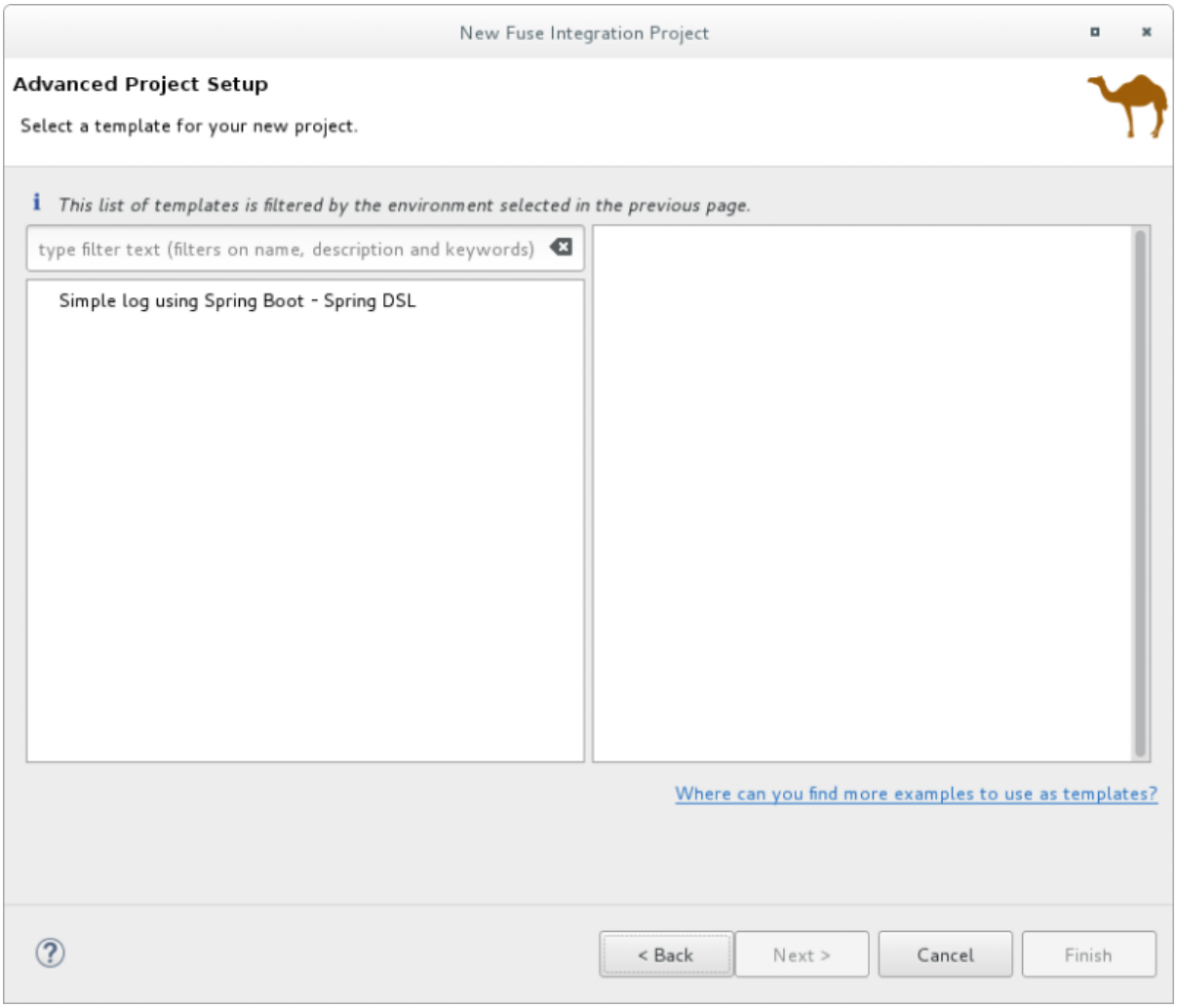
   接受其他选项的默认值。

3. **点 Next 以打开 Select a Target Runtime 页面：**

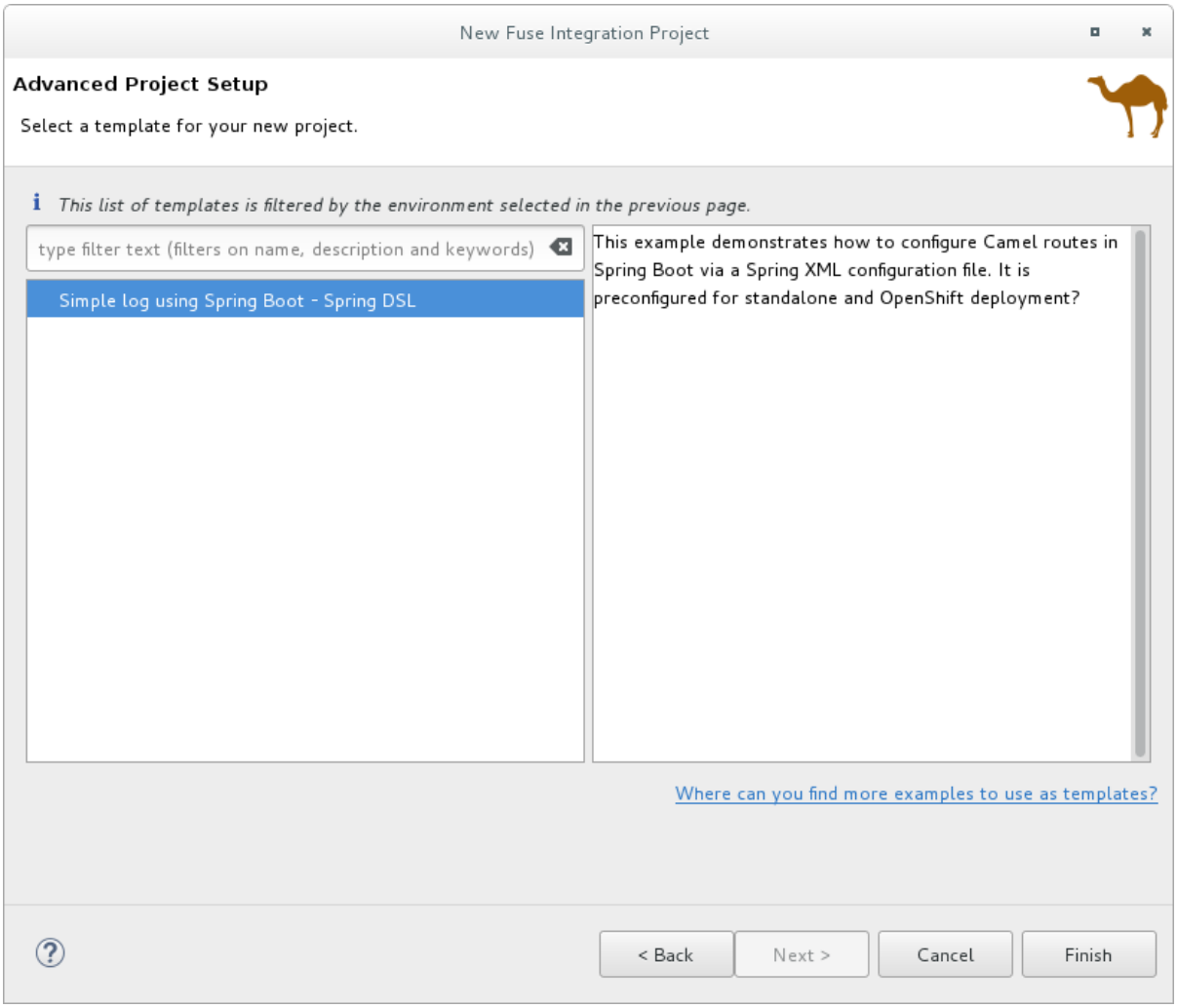


保留 Target Runtime 的默认值(*未选择运行时*)和 Camel Version。

4. **点 Next 打开 Advanced Project Setup 页面：**



5. **使用 Spring Boot - Spring DSL 模板选择 Simple log。**



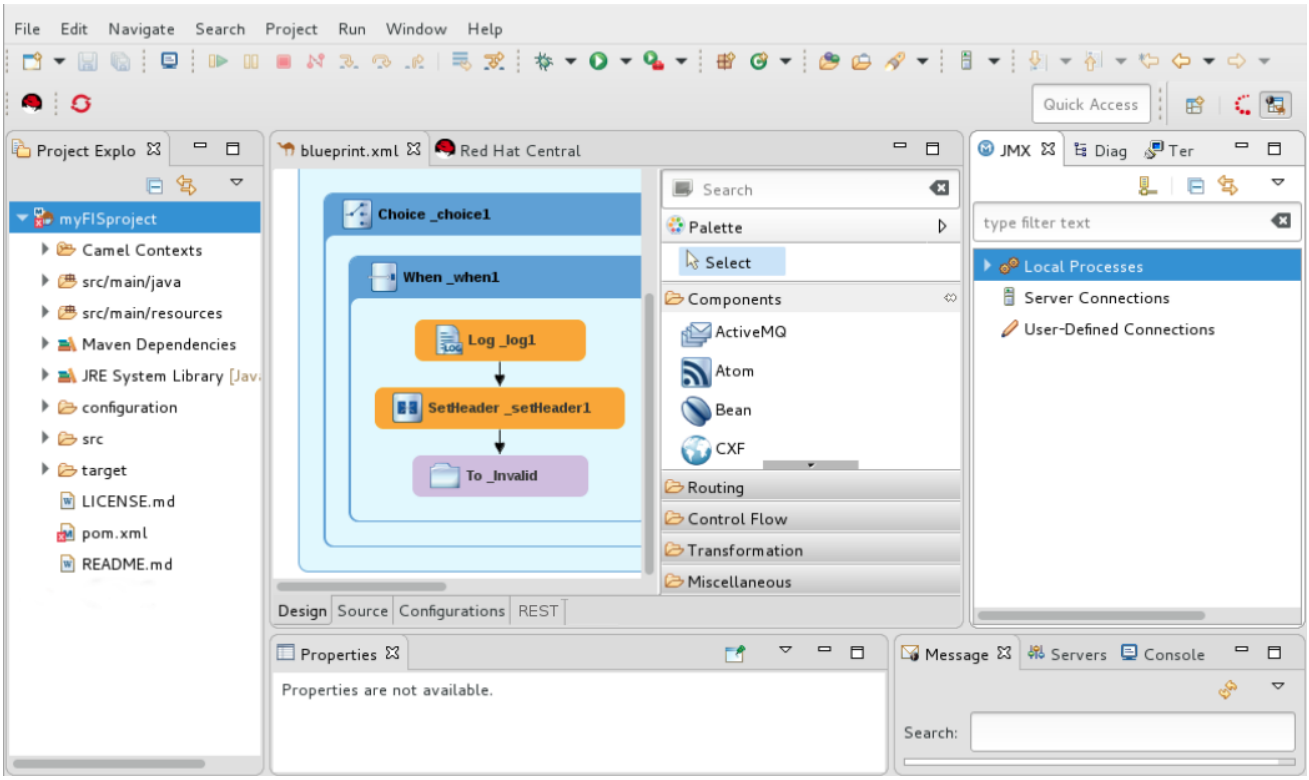
6. **点 Finish。**

注意

由于首次为 Fuse 集成项目下载的依赖项数量，因此构建可能需要一段时间。

如果 Fuse Integration 透视图尚未打开，Developer Studio 会提示您指明您现在是否要打开它。单击 Yes 。

构建完成后，Fuse Integration 视角会显示项目，例如：



此时，您可以：

- 在 OpenShift 上部署项目

- 第 5.1 节 “将路由作为本地 Camel 上下文运行” 验证路由上下文是否在您的本地机器上成功运行

  连接到 JMX Navigator 视图中运行的上下文（请参阅 “查看本地 JMX 服务器中的进程”一节），您可以监控路由组件并测试路由组件是否按预期执行：

  - 查看路由组件的 JMX 统计 JMX 统计see 第 20 章 *查看组件的 JMX 统计信息*。
  - 编辑正在运行的 route swig-wagonsee 第 24 章 *管理路由端点*。
  - suspend/resume the running the running route swig-wagonsee 第 26 章 *管理路由上下文*
  - 在运行的 routewagon-wagon 上启动/停止追踪 第 22 章 *追踪路由*

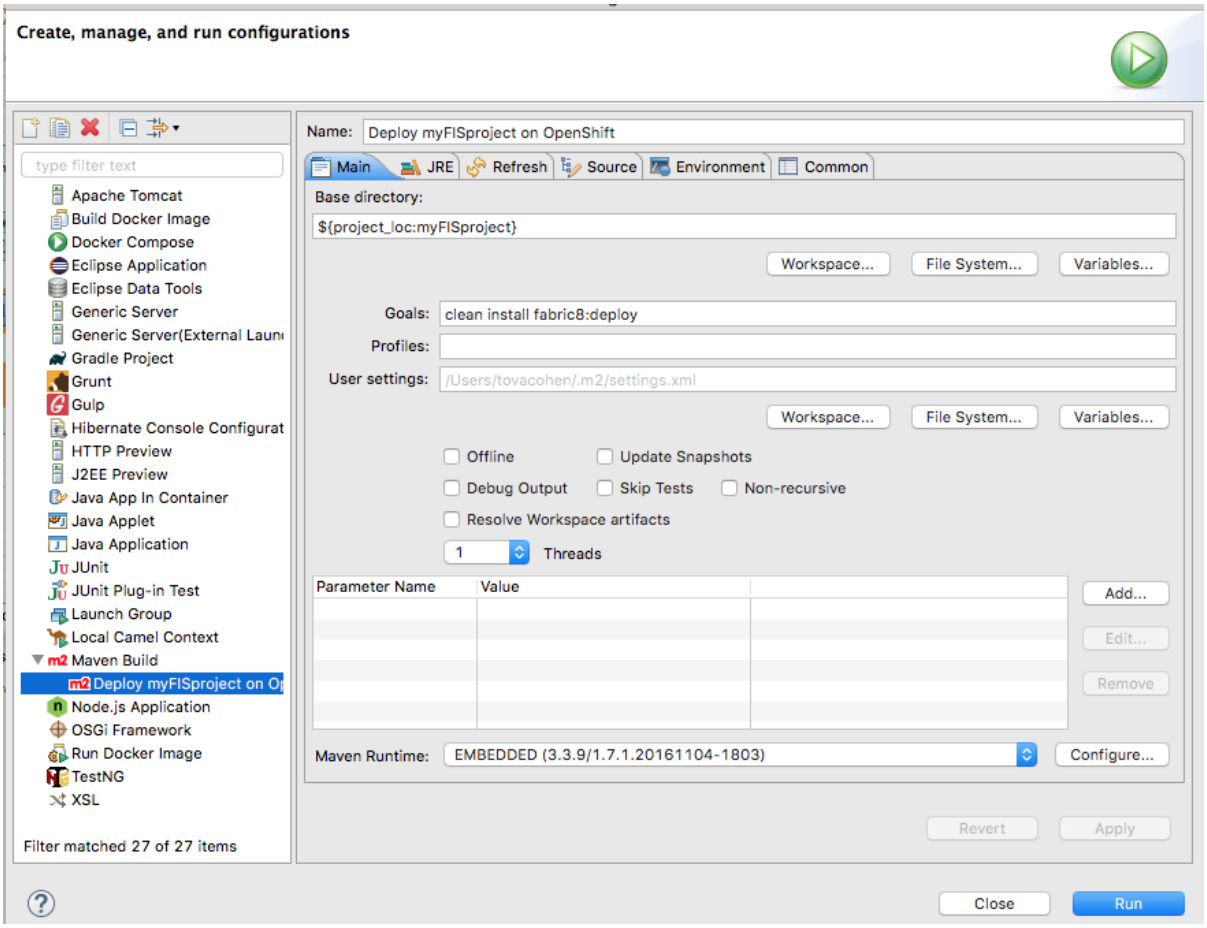
- 在项目的 camel-context.xml 文件上运行 Camel debugger，以发现和修复逻辑错误 swig-mtcsee 第 II 部分 “调试路由上下文”

6.5. 将 FUSE 集成项目部署到 OPENSHIFT

1. 在 Project Explorer 视图中，右键单击项目的根（本例中为 myFISproject）以打开上下文菜单。

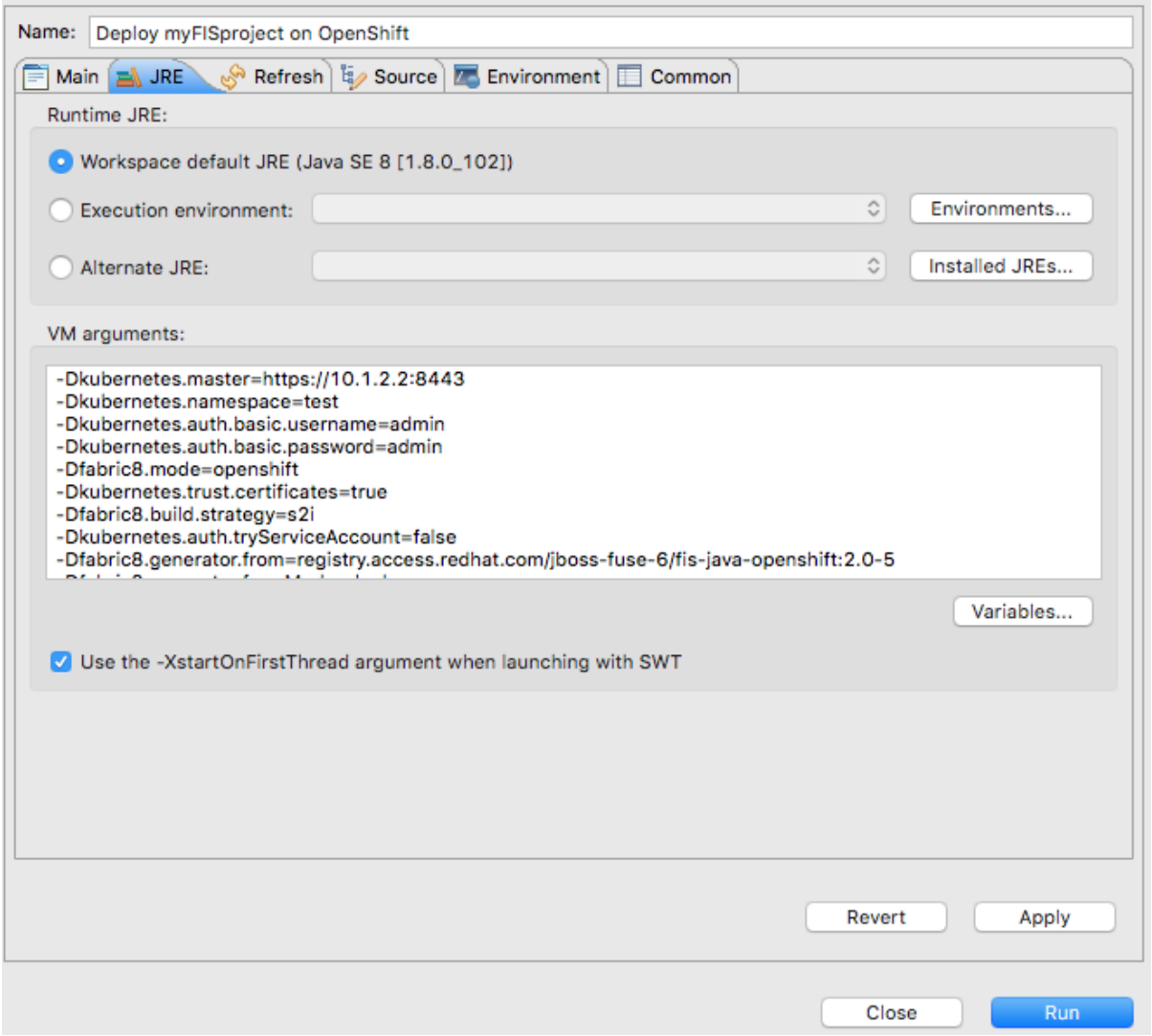
2. 选择 Run As → Run Configuration 以打开 Run Configuration 向导。

3. 在侧边栏中，选择 Maven Build → Deploy <projectname> on OpenShift （本例中为 Deploy myFISproject on OpenShift）以打开项目的默认运行配置：



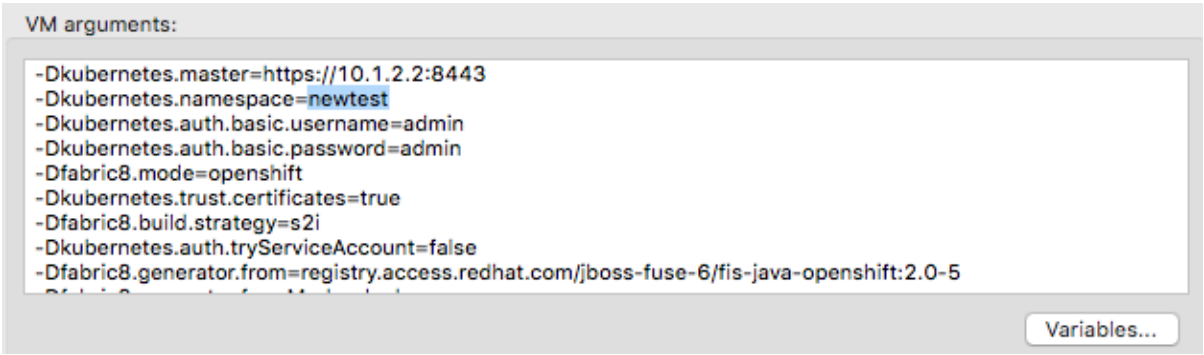
保留默认设置，因为它们位于 Main 选项卡中。

4. 打开 JRE 选项卡以访问虚拟机参数：



5. **在 VM 参数 窗格中，更改 -Dkubernetes.namespace=test 参数的值，使其与您在创建 OpenShift 项目时使用的项目名称匹配（在 第 6.3 节 “创建新的 OpenShift 项目” 中OpenShift 项目名称）。**

在本例中，将默认值 test 改为 newtest ：



根据 OpenShift 配置，您可能需要修改其他的VM 参数来支持它：

- -Dkubernetes.master=https://192.168.99.1:8443

  在运行多个 OpenShift 实例或使用远程实例时，您需要指定用于部署的 OpenShift 实例的 URL。上面的 URL 是示例。

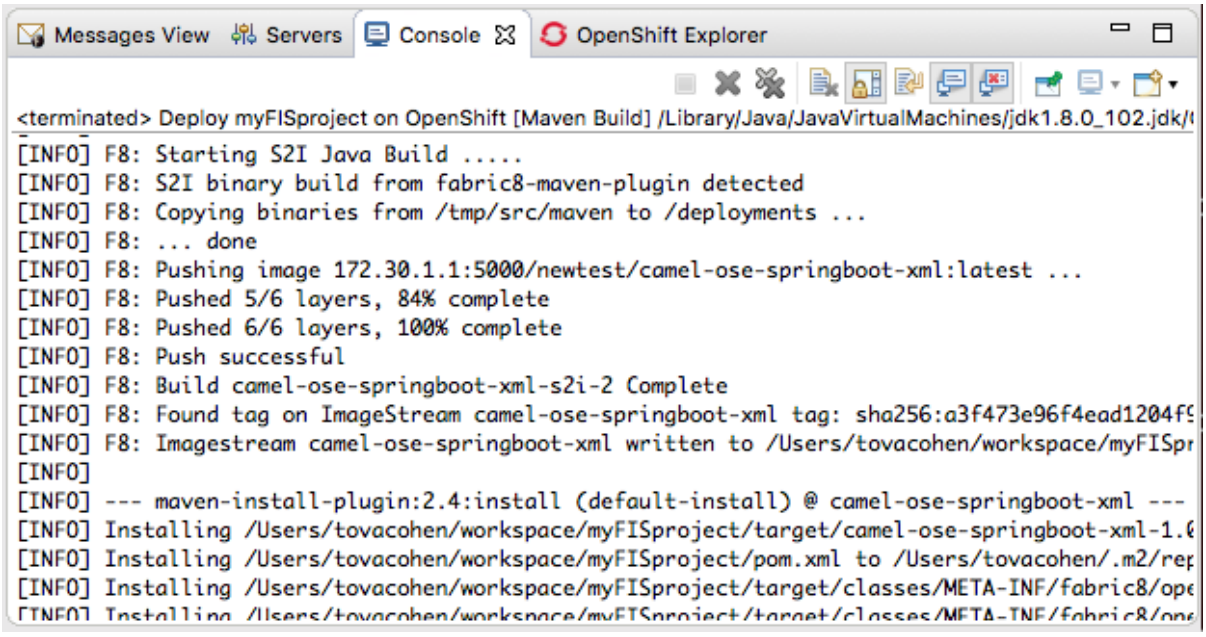
- -Dkubernetes.trust.certificates=true

  - 在使用 CDK 时，需要此参数。将它保留为 true。
  - 如果您使用具有有效 SSL 证书的 OpenShift 实例，请将此参数的值更改为 false。

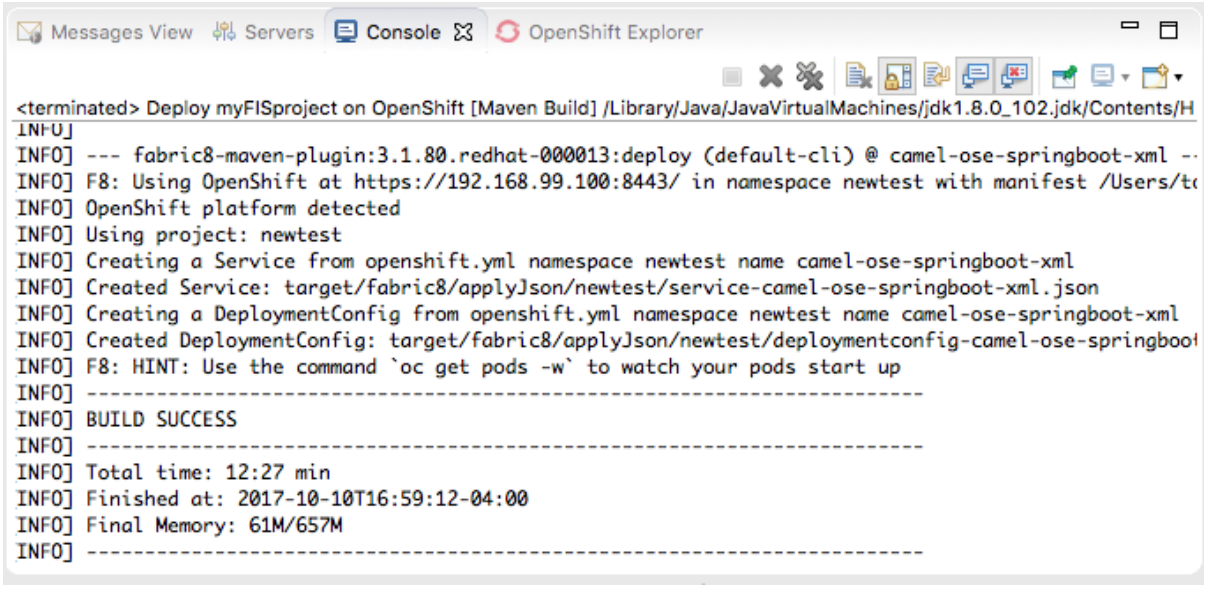
6. 单击 Apply，然后单击 Run。

由于要下载的依赖项数量，第一次部署可能需要一些时间。您的计算机和互联网连接的速度是做出贡献的因素。通常，完成第一次部署需要 25 到 35 分钟。

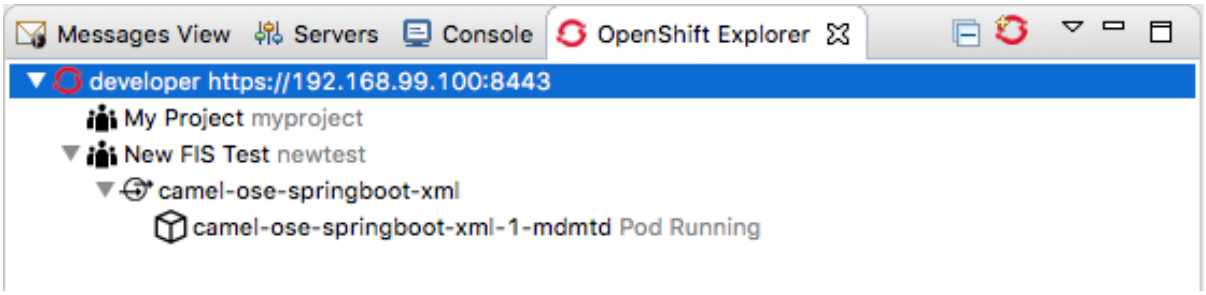
在 Console 视图中，您可以跟踪部署过程的进度。在以下输出中，条目为 *Pushing image 172.30.1 ...。* 表示项目成功构建，应用程序镜像正在推送到 OpenShift，以用于构建 Docker 容器。



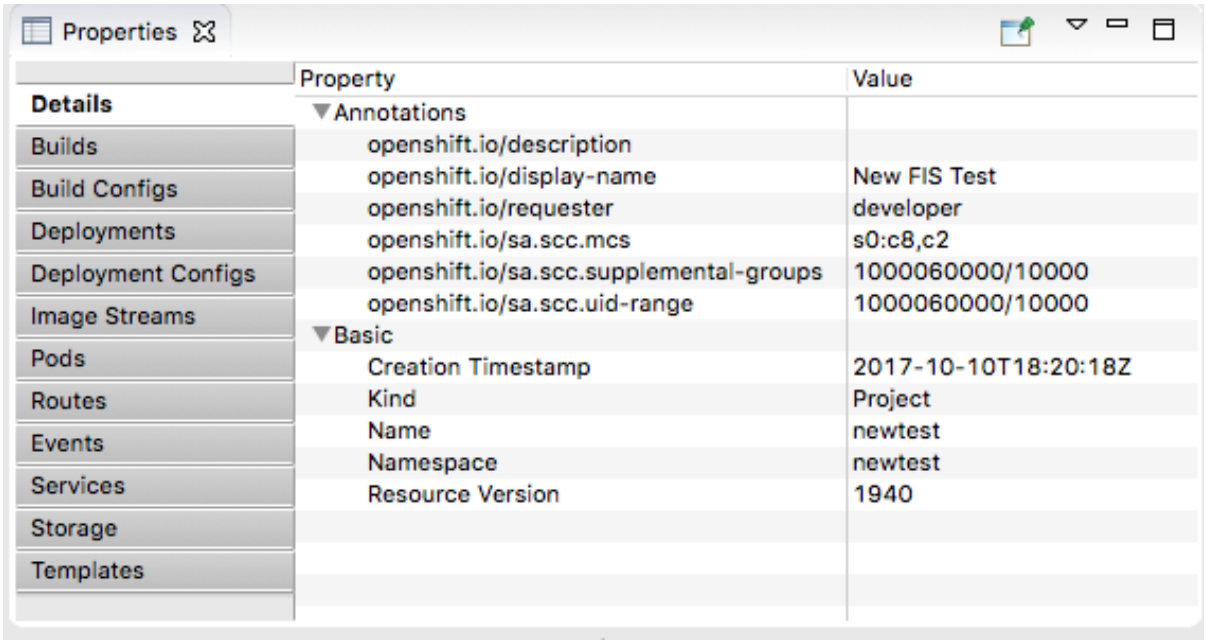
部署成功完成后，控制台 视图会显示 BUILD SUCCESS ：



7. 切换到 **OpenShift Explorer** 视图并选择 **New FIS Test newtest** ：



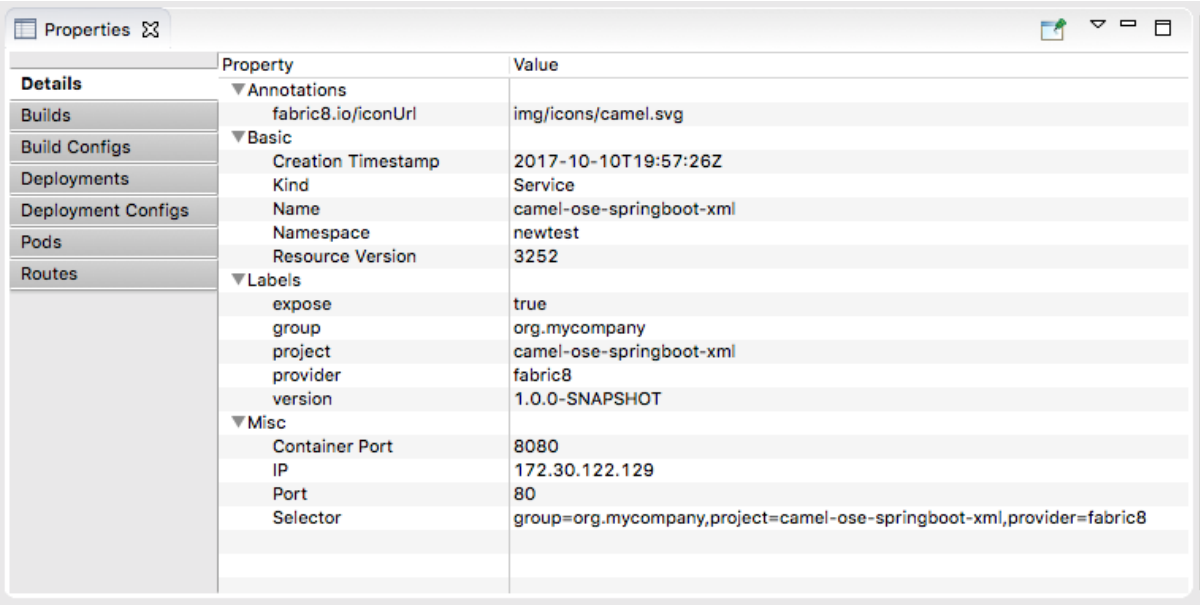
在 **Properties** 视图中，**Details** 页面会显示项目的属性值。



打开其他标签页(**Builds**、**Build Configs**、**Deployment**、...)以查看项目的其他属性。**Properties** 视图提供与 **OpenShift Web** 控制台相同的信息。

8.

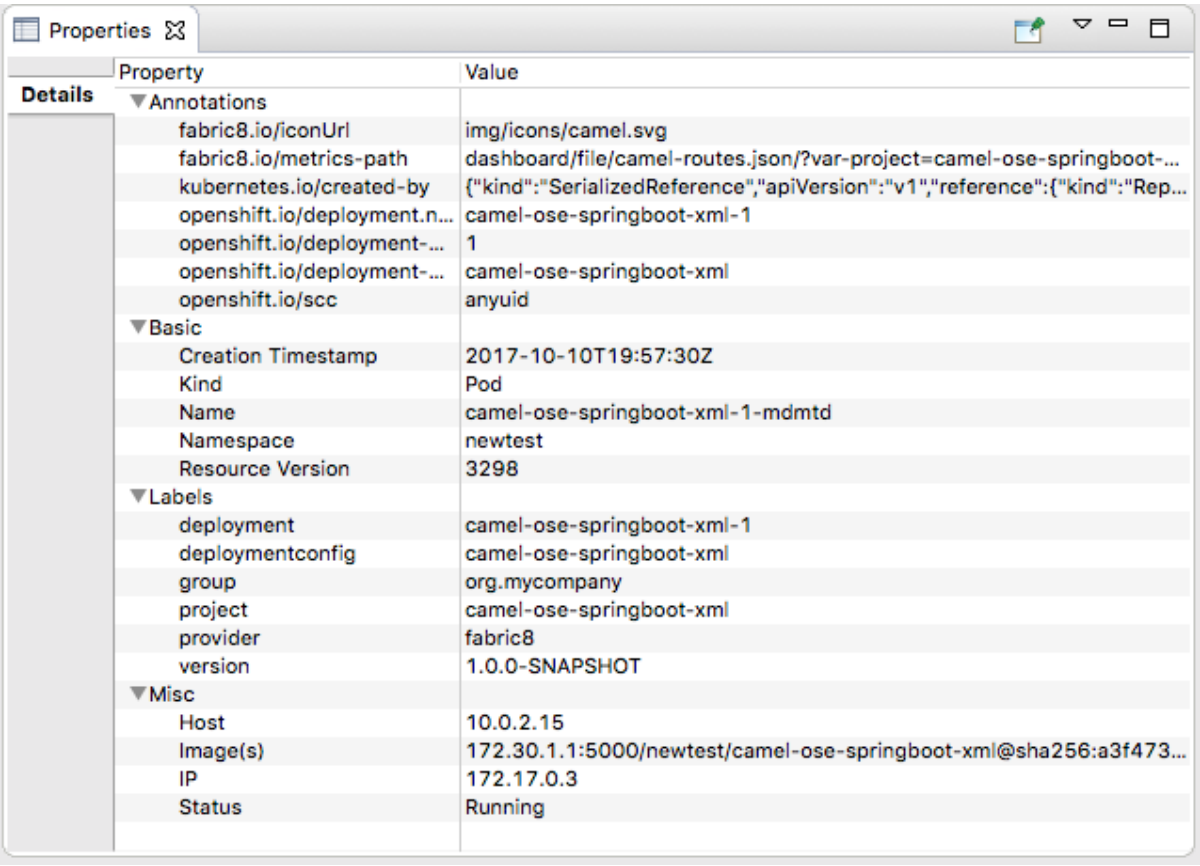
在 OpenShift Explorer 视图中，选择 camel-ose-springboot-xml 以在 Properties 视图中查看其详情：



滚动浏览其他标签页以查看部署配置的其他属性。

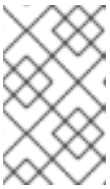
9.

在 OpenShift Explorer 视图中，选择 camel-ose-springboot-xml-1-mdmtd Pod Running，然后在 Properties 视图中查看正在运行的实例的详情：



10.

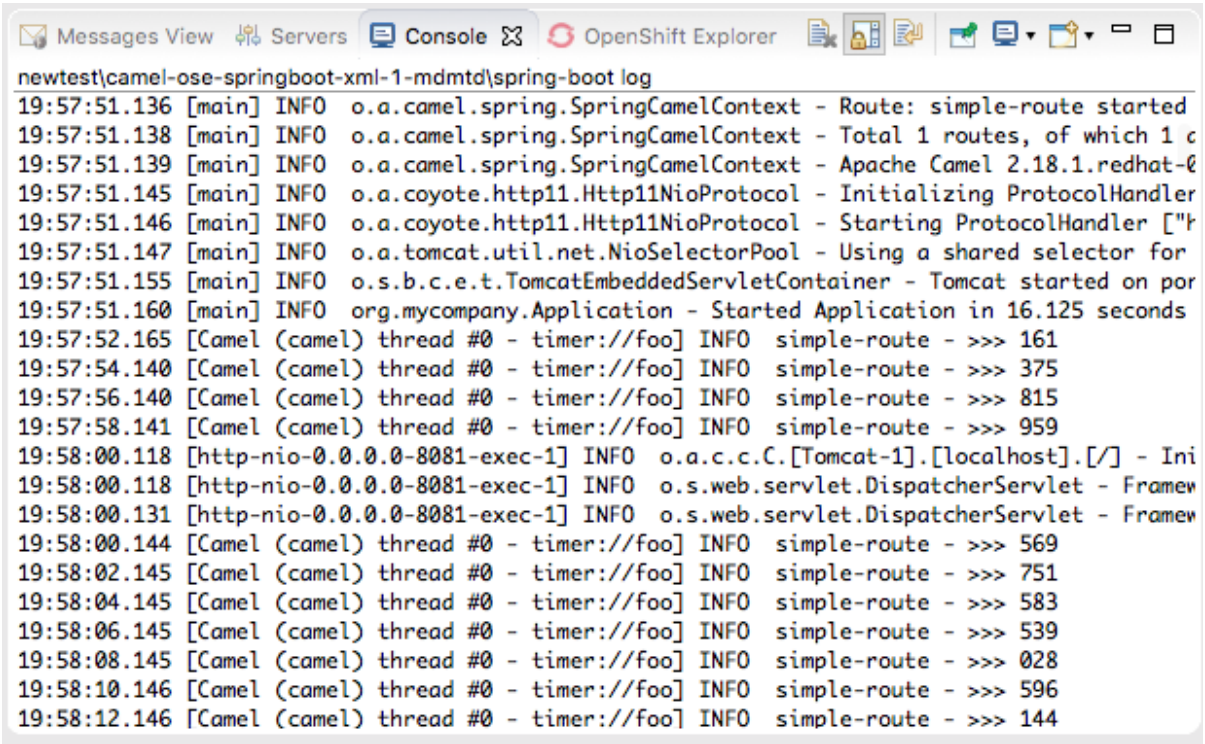
在 OpenShift Explorer 视图中，右键单击 camel-ose-springboot-xml-1-mdmtd Pod Running，然后选择 Pod Logs...。



注意

如有提示，输入安装的 oc 可执行文件的路径。需要检索 Pod 日志。

Console 视图会自动打开，显示来自正在运行的 pod 的日志：



点击 Console 视图的菜单栏中的

来终止会话并清除控制台输出。

6.6. 访问 OPENSHIFT WEB 控制台

注意

此信息仅适用于 Red Hat Container Development Kit 安装。

要访问 OpenShift Web 控制台，请打开浏览器并输入特定于您的实例和计算机的 OpenShift 服务器的 URL。例如，在浏览器的地址字段中输入 https://192.168.99.100:8443。

您可以使用默认凭证以开发者或管理员身份登录 Web 控制台：

- 默认开发人员角色

  开发人员用户只能查看自己的项目和提供的 OpenShift 示例项目，它们演示了 OpenShift v3 功能。开发人员用户可以创建、编辑和删除他们拥有部署在 OpenShift 上的任何项目。

  - username - developer
  - password - developer

- 默认管理员角色

  管理员用户可以查看和访问 OpenShift (CDK)中的所有项目。管理员用户可以创建、编辑和删除 OpenShift 中部署的任何项目。

  - username - admin
  - password - admin

有关使用 OpenShift Web 控制台的更多信息，请参阅 入门指南。

第 7 章 使用红帽 FUSE SAP 工具套件

红帽 Fuse SAP Tool 套件可使您的 Camel 路由与远程 SAP Application Server 集成。提供了各种 SAP 组件来支持远程功能调用(RFC)和接收中间文档(IDocs)。SAP 工具套件依赖于 SAP 的 JCo 和 IDoc 客户端库。要安装和使用这些库，您必须有一个 SAP Service Marketplace 帐户。

7.1. 安装 RED HAT FUSE SAP TOOL SUITE

概述

Red Hat Fuse SAP Tool Suite 提供了 Edit SAP Connection Configuration 对话框，它可帮助您创建和管理 SAP Application Server 和 Destination 连接。默认不安装该套件，因为它需要第三方 JCo 和 IDoc 客户端库，这些库由 SAP 单独许可。

SAP 工具的平台限制

因为 SAP 工具套件取决于第三方 JCo 和 IDoc 库，因此只能安装到这些库支持的平台上。有关 SAP 工具的平台限制的详情，请参阅 Red Hat Fuse 7.13 支持的配置。

先决条件

- 在安装 Fuse SAP Tool 套件前，您必须从以下位置下载 JCo 和 IDoc 库 ：http://service.sap.com/connectors
- 要确定适用于您的操作系统的适当 JCo 和 IDoc 库，请参阅 Red Hat Fuse 支持的配置 页面。
- 要下载 JCo 和 IDoc 库，您需要一个 SAP Service Marketplace 帐户。
- 对于这个安装过程，您可以使用归档格式保存下载的文件。不需要提取内容。

创建并测试 SAP DESTINATION CONNECTION

概述

在 Fuse SAP Tool 套件中，Edit SAP Connection Configuration 对话框可帮助您创建和管理 SAP 应用目标连接。本节论述了如何创建和测试 SAP 目标连接。

流程

要创建并测试 SAP 目标连接，请执行以下步骤：

1. **导航到路由编辑器的 全局配置 选项卡，再单击 添加。**

   此时会出现 Create new global element 视图。

2. **在 SAP 下，选择您要创建的连接类型。选择 SAP Connection，再单击 确定。**

   此时会出现 Edit SAP Connection Configuration 对话框。它允许您创建、编辑和删除 Destination 和 Server Connection 配置。

3. **要创建新的 Destination Data Store，请单击 Add Destination 选项卡。**

   此时会出现 Create Destination 对话框。

4. **在 Destination Name 字段中输入目的地的名称，然后单击确定。**

5. **在 属性对话框中，**

   a. **单击 Basic 选项卡，以配置连接到 SAP 目的地所需的基本属性。在这个标签页中填写以下属性字段来配置连接：**

      - **SAP Application Server**
      - **SAP 系统号**
      - **SAP Client**

- **logon User**
- **Logon Password**
- **logon Language**

b. 单击 **Connection** 选项卡，以添加连接到 **SAP** 目的地所需的值。填写以下属性字段来配置连接：

- **SAP 系统号**
- **SAP Router 字符串**
- **SAP Application Server**
- **SAP Message Server**
- **SAP Message Server Port**
- **网关主机**
- **网关端口**
- **SAP 系统 ID**
- **SAP Application Server Group**

c. 单击 **Authenticate** 选项卡，以添加验证 **SAP** 目的地所需的值。填写以下属性字段以配置连接。

- **SAP Authentication type**
- **SAP Client**
- **logon User**
- **logon User Alias**
- **Logon Password**
- **SAP SSO Logon Ticket**
- **SAP X509 Login Ticket**
- **logon Language**

d. **点 Special** 选项卡。在这个标签页中填写以下属性字段来配置连接：

- 选择 **CPIC Trace**
- **初始代码页面**

e. **点 Pool** 选项卡，并填写以下属性字段来配置连接：

- **连接池调整限制**
- **连接池容量**

- 连接池过期时间
- 连接池过期检查期
- 连接池最大获取客户端时间

f. 点 SNC 选项卡并填写以下属性字段来配置连接：

- SNC 合作伙伴名称
- SNC 级别安全
- SNC 名称
- SNC 库路径

g. 点 Repository 选项卡，并填写以下属性字段来配置连接：

- 仓库目标
- 仓库日志用户
- Repository Logon Password

注意

如果您需要有关这些设置的更多信息，请参阅 SAP 文档。

6. 您现在已准备好测试目标连接。在 Edit SAP Connection Configuration 对话框中，右键单击目的地名称，再选择 Test。

此时会打开 **Test Destination Connection** 对话框。

7. 对话框使用当前的目标配置设置来连接 **SAP Destination Data Store**。如果测试成功，您将在状态区中看到以下信息：

```
Connection test for destination 'YourDestination' succeeded.
```

否则，错误报告会出现在状态区中。

8. 点 **Close** 关闭 **Test Destination Connection** 对话框。

9. 点 **Finish**。新创建的 **SAP Destination Connection** 会出现在 **SAP** 下。

7.2. 创建并测试 SAP 服务器连接

概述

在 **Fuse SAP Tool** 套件中，**Edit SAP Connection Configuration** 对话框可帮助您创建和管理 **SAP** 应用服务器连接。本节论述了如何创建和测试 **SAP** 服务器连接。

流程

要创建并测试 **SAP** 服务器连接，请执行以下步骤：

1. 导航到路由编辑器的 **全局配置** 选项卡，再单击 **Add**。

此时会出现 **Create new global element** 视图。

2. 在 **SAP** 下，选择您要创建的连接类型。选择 **SAP Connection**，再单击 **确定**。

此时会出现 **Edit SAP Connection Configuration** 对话框。它允许您创建、编辑和删除 **Destination** 和 **Server Connection** 配置。

3. 要创建新的 Server Data Store，请单击 Add Server 选项卡。

   此时会出现 Create Server 对话框。

4. 在 Server Name 字段中输入服务器的名称，然后单击确定。

5. 在 属性对话框中，

   a. 点 Mandatory 选项卡配置连接到 SAP 服务器所需的基本属性。在这个标签页中填写以下属性字段来配置连接：

   - 网关主机
   - 网关端口
   - 程序 ID
   - 仓库目标
   - 连接数

   b. 点 Optional 选项卡，并填写以下属性字段来配置连接：

   - SAP Router 字符串
   - worker 线程数
   - 最小 worker 线程数

- 最大启动延迟
- 仓库映射

c. 单击 SNC 选项卡，并填写以下属性字段来配置连接。

- SNC 级别安全
- SNC 名称
- SNC 库路径

注意

有关设置的更多信息，请参阅 SAP 文档。

6. 您现在已准备好测试服务器连接。在 Edit SAP Connection Configuration 对话框中，右键单击服务器名称，再选择 Test。

此时会打开 Test Server Connection 对话框。

7. 对话框使用当前服务器配置设置来连接 SAP 服务器数据存储。如果测试成功，您将在状态区中看到以下信息：

```
Server state: STARTED
Server state: ALIVE
```

如果测试失败，服务器状态会报告为 DEAD。

8. 点 Stop 关闭 Test Sever。

9. 单击 **Close** 以关闭 **Test Server Connection** 对话框。

10. 点 **Finish**。新创建的 **SAP Server Connection** 会出现在 **SAP** 下。

7.3. 删除目标和服务器连接

概述

下面的部分论述了如何在 **Edit SAP Connection Configuration** 对话框中删除 **SAP Destination** 和 **Server** 连接。

流程

如果要删除 **Destination** 和 **Server** 连接，请执行以下步骤：

1. 导航到路由编辑器的 **全局配置** 选项卡，再单击 **Add**。

   此时会出现 **Create new global element** 视图。

2. 在 **SAP** 下，选择 **SAP Connection**，再单击 **Ok**。

   此时会出现 **Edit SAP Connection Configuration** 对话框。它允许您创建、编辑和删除 **Destination** 和 **Server Connection** 配置。

3. 在 **Edit SAP Connection Configuration** 对话框中，选择您要删除的 **Destination** 和 **Server Data Stores**。

4. 点击 **Delete**。它将删除所选连接。

   Atlast，单击 **Finish**。它将保存所有更改。

7.4. 创建新 SAP 端点

概述

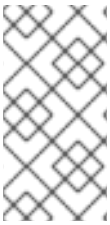
您可以使用路由编辑器中的组件面板，通过 Edit SAP Connection Configuration 对话框将 SAP 组件添加到路由中。

注意

如果您使用 SAP Connection 视图，请记得将必要的 SAP 连接配置数据粘贴到 Blueprint XML 或 Spring XML 代码中。

先决条件

您必须已创建了一些 SAP 目标连接和/或服务器连接以及 Edit SAP Connection Configuration 对话框的帮助。



注意

如果您使用 SAP Connection 视图，请将此配置导出到适当类型的文件（打印 XML 或 Spring XML）。

流程

要创建新的 SAP 端点，请执行以下步骤：

1. 假设您已经有一个 Fuse 项目和一个 Camel XML 文件可供使用（可以是 Blueprint XML 或 Spring XML 格式）。

2. 在路由编辑器中打开 Camel XML 文件。如果您已经安装了 Red Hat Fuse SAP Tool Suite，则应该可以在路由编辑器中的 Components palette 下看到 SAP 组件。以下 SAP 组件由工具套件提供：

   - SAP IDoc Destination
   - SAP IDoc List Destination
   - SAP IDoc List Server

- SAP qRFC 目的地
- SAP Queued IDoc Destination
- SAP Queued IDoc List Destination
- SAP sRFC 目的地
- SAP sRFC 服务器
- SAP tRFC Destination
- SAP tRFC Server

在路由编辑器的 Design 选项卡中，将其中一个组件拖到 canvas 中，以在当前的 camelContext 中创建新的 SAP 端点。

注意

SAP Netweaver 组件不属于红帽 Fuse SAP 工具套件。它托管在 Apache Camel 项目中。

3. 单击 canvas 底部的 Source 选项卡，以切换到路由编辑器的 Source 选项卡。您可以查看路由的 XML 源。

4. 在指定 SAP 端点 URI 时，您必须以 URI 格式嵌入目标名称或服务器连接名称。例如，sap-srfc-destination 组件具有以下 URI 格式：

```
sap-srfc-destination:destinationName:rfcName
```

要引用特定目的地，请使用相关 entry 元素的 key 属性的值作为此 URI 中的 destinationName。

第 8 章 数据转换入门

与系统和数据集成相关的挑战之一是组件系统通常使用不同的数据格式。您不能简单地将信息从一个系统发送到另一个系统，而无需将其转换为接收系统可识别的格式（或语言）。数据转换是提供给此转换的术语。

在本章中，您将了解如何在预定义的 Camel 路由中包含数据转换。Camel 路由将消息从源端点定向到使用 JSON 数据的目标端点。您可以添加和定义数据转换组件，该组件将源的 XML 数据格式映射到目标的 JSON 数据格式。

8.1. 为数据转换示例创建项目

1. 创建一个新的 Fuse Integration Project （选择 File → New → Fuse Integration Project）。

   在向导中提供以下信息：

   - 项目名称： 初学者
   - 部署平台： 独立
   - 运行时环境：Na hia/Fuse on Karaf
   - Camel 版本：使用默认
   - 模板： Empty - Blueprint DSL

2. 从以下位置下载准备的数据示例 ：https://github.com/FuseByExample/fuse-tooling-tutorials/archive/user-guide-11.1.zip

3. 将 data 文件夹和其包含的三个文件从 user-guide-11.1.zip 存档提取到 Fuse Integration 项目的 src 目录(初学者/src/data)。

4. **在 Project Explorer 视图中，展开 starter 项目。**

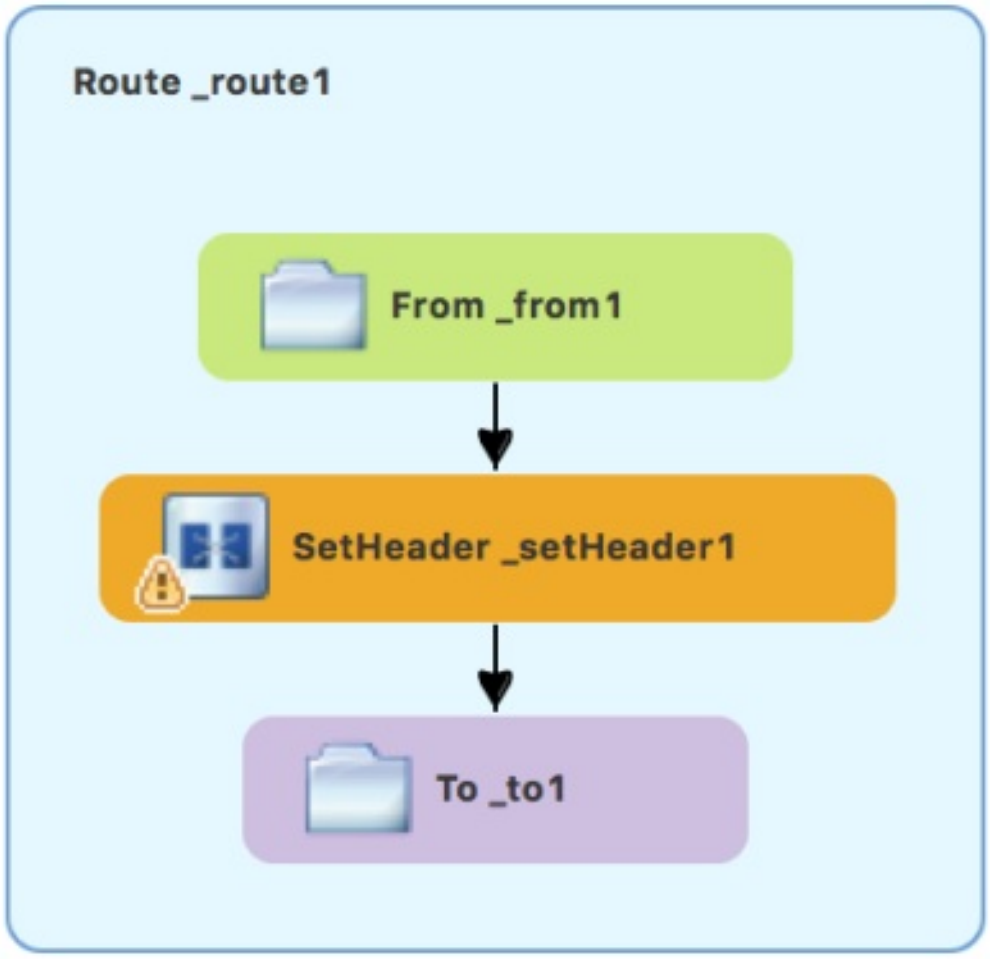
5. **双击 Camel Contexts → src/main/resources/OSGI-INF/blueprint/blueprint.xml，以在路由编辑器的 Design 选项卡中打开 路由。**

6. **点 Source 选项卡查看底层 XML。**

7. **将 <route id="_route1"/& gt; 替换为以下代码：**

```
<route id="_route1">
  <from id="_from1" uri="file:src/data?fileName=abc-order.xml&amp;noop=true"/>
  <setHeader headerName="approvalID" id="_setHeader1">
    <simple>AUTO_OK</simple>
  </setHeader>
  <to id="_to1" uri="file:target/messages?fileName=xyz-order.json"/>
</route>
```

8. **点 Design 选项卡返回路由的图形显示：**

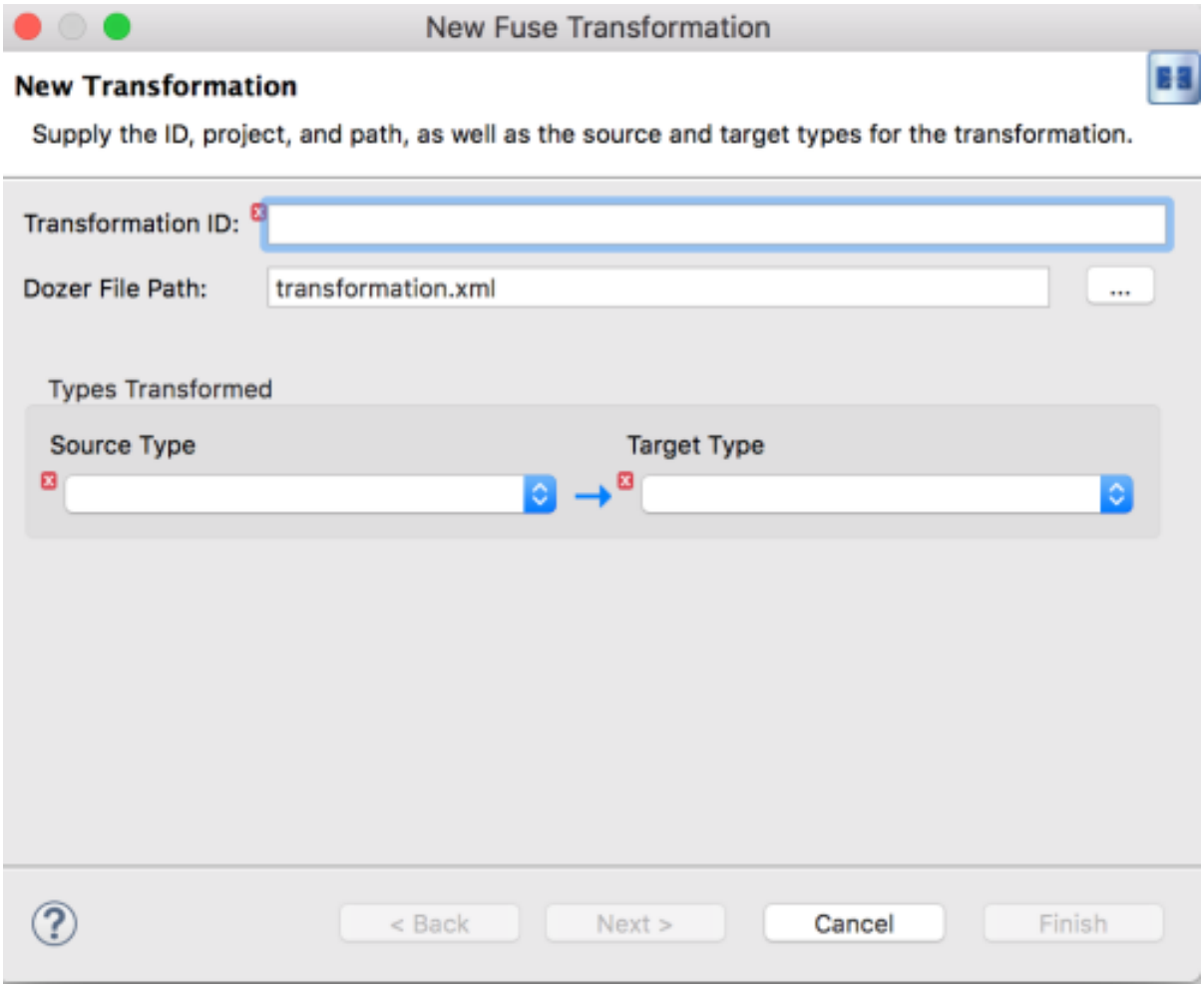


8.2. 将数据转换节点添加到 CAMEL 路由

1. 在 Prod 中，展开 Transformation drawer。

2. 单击 Data Transformation 模式，然后在 canvas 中点 SetHeader _setHeader1 和 To_to1 节点之间的箭头。

   New Transformation 向导会在 Dozer File Path 字段自动填充时打开。



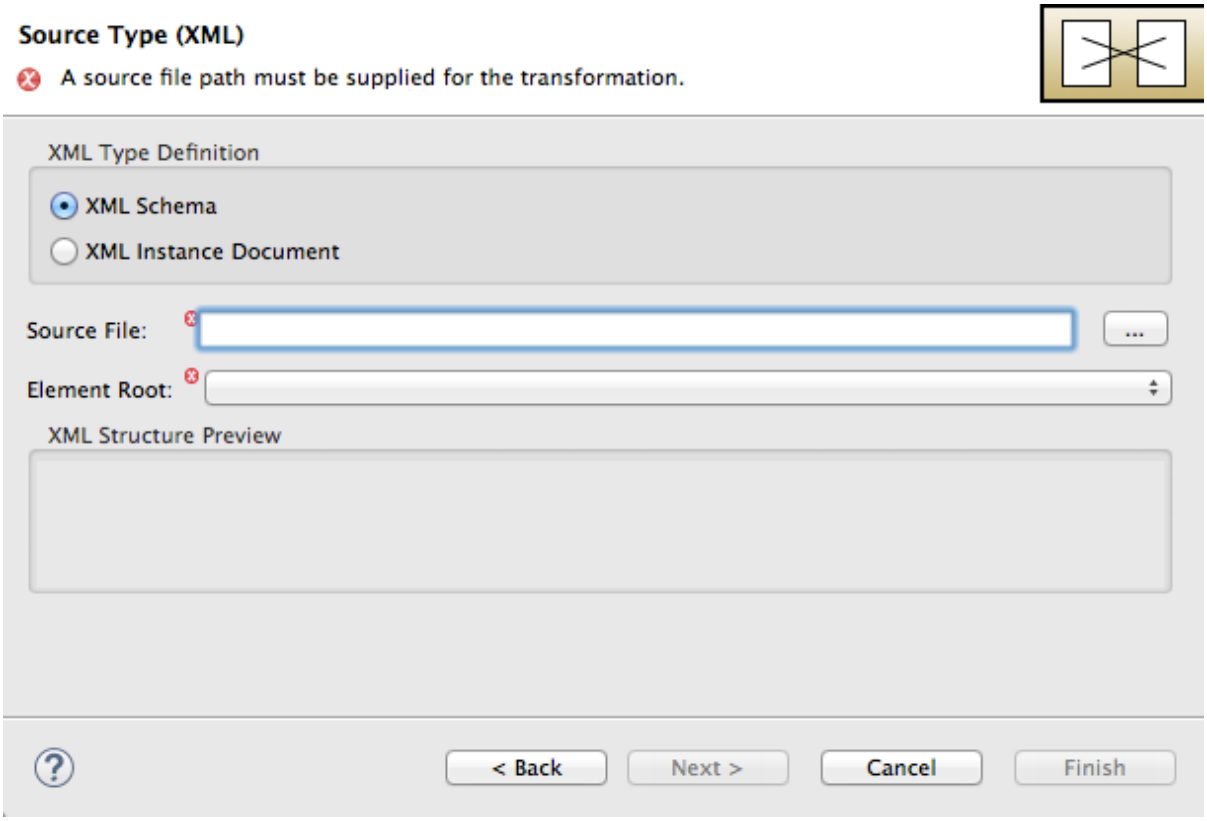
3. 填写剩余的字段：

   - 在 Transformation ID 字段中，输入 xml2json。
   - 对于 Source Type，从下拉菜单中选择 XML。

- 对于 **Target Type**，从下拉菜单中选择 **JSON**。

4. 点击 **Next**。

源类型 (XML) 定义页面会打开，您可以在其中指定 XML 架构 （默认）或示例 XML 实例文档 来提供源数据的类型定义：



5. 使 XML 架构 保持启用状态。

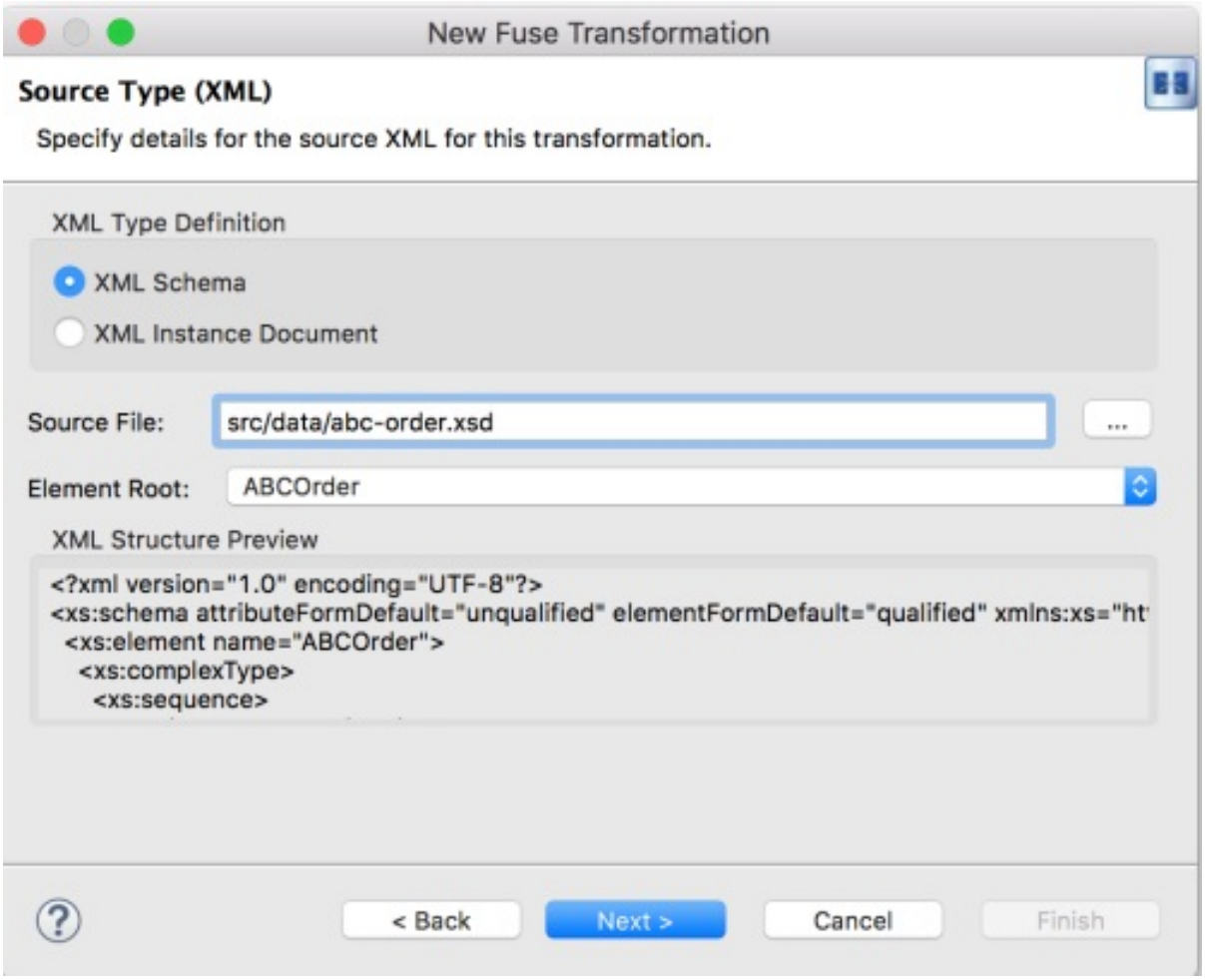
6. 对于 Source 文件，浏览 XML 架构文件的位置或用于源数据的类型定义的 XML 实例文件的位置，然后选择（本例中为 abc-order.xsd）。

XML 结构预览 窗格显示 XML 结构的预览。

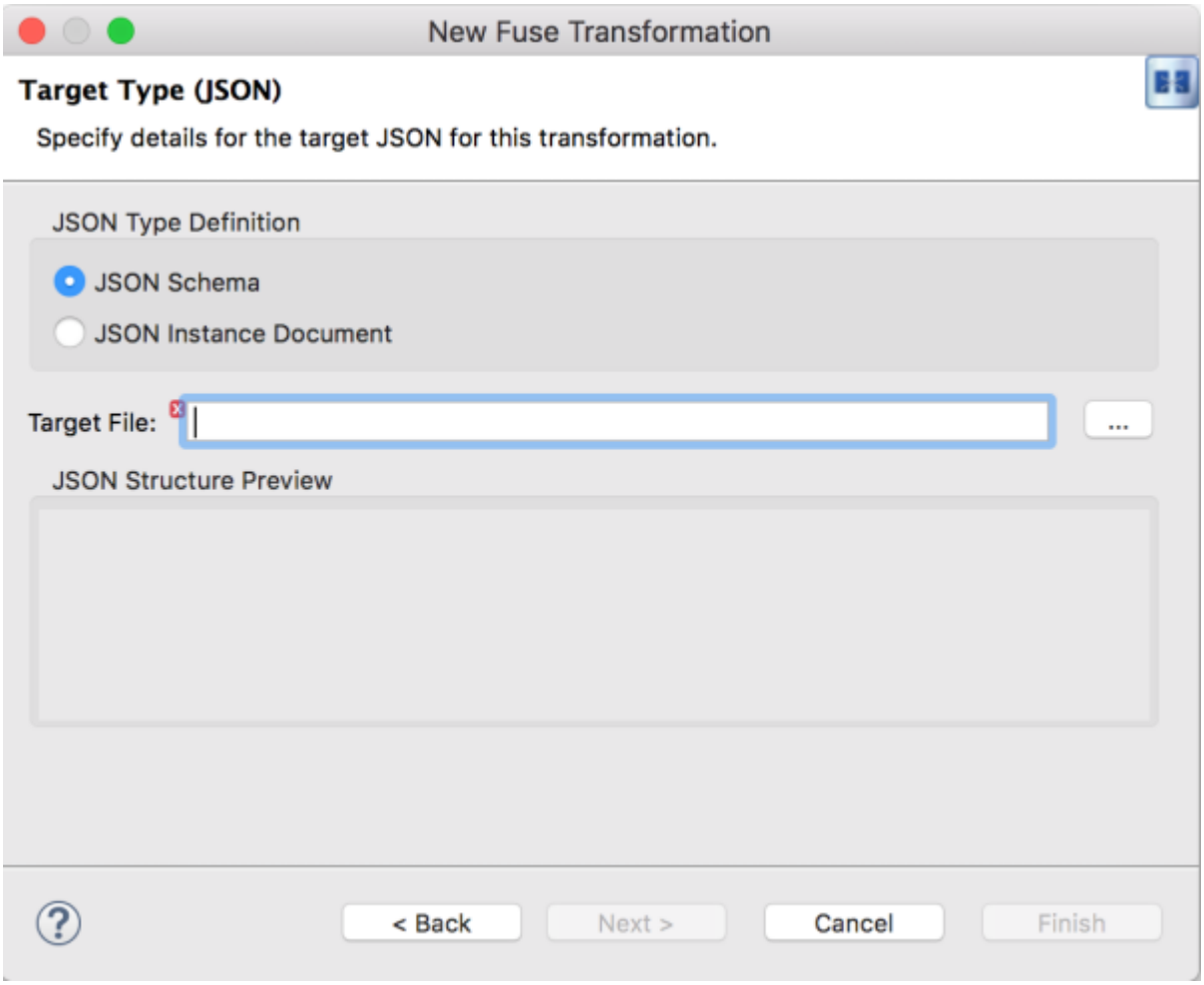
7. 在 Element root 字段中，输入 ABCOrder。

工具使用此文本来标记显示要映射的源数据项的窗格。

Source Type (XML) 定义页面现在应类似如下：

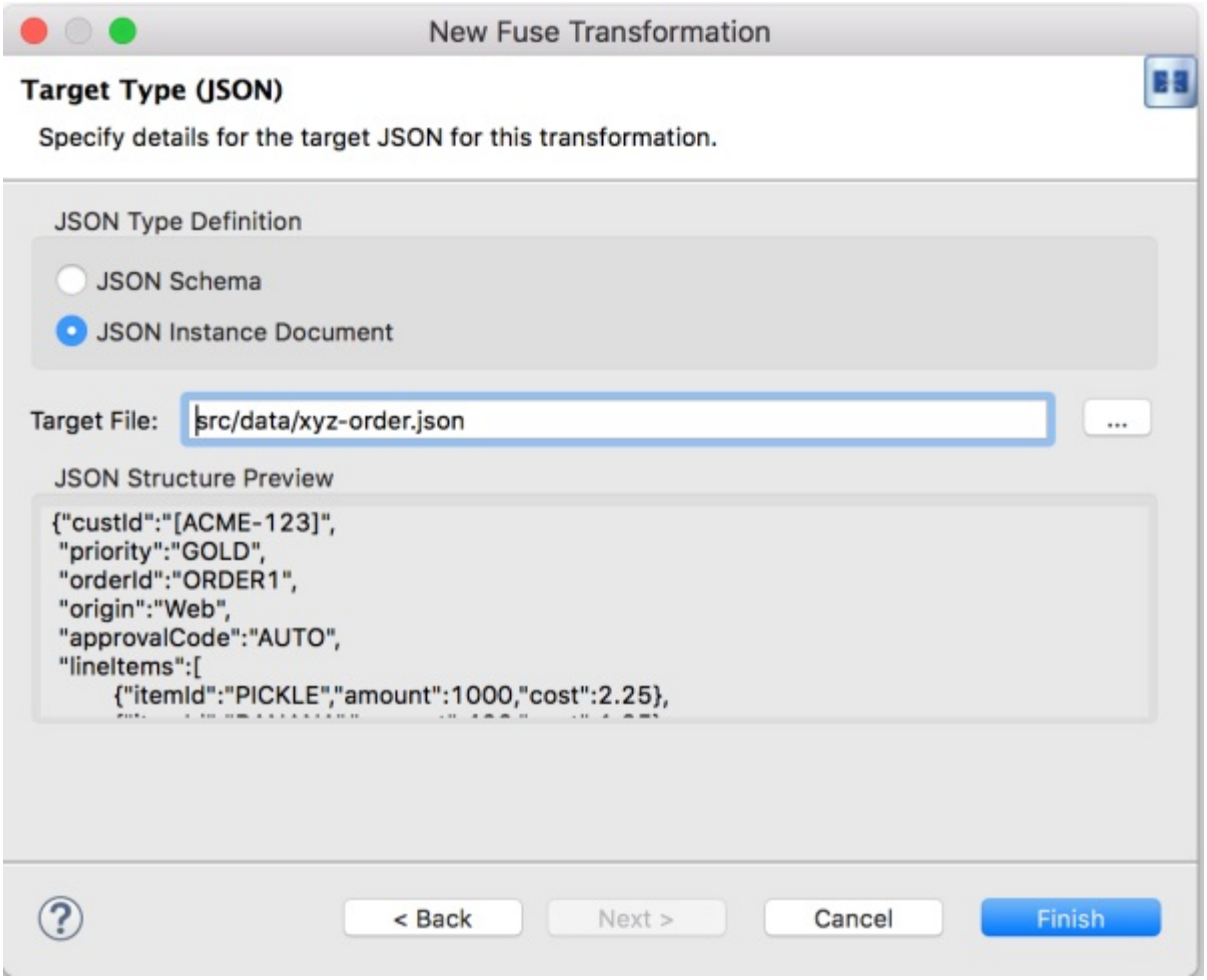


8. **点 Next 以打开 Target Type (JSON) 定义页面。这是您为目标数据指定类型定义的位置。**



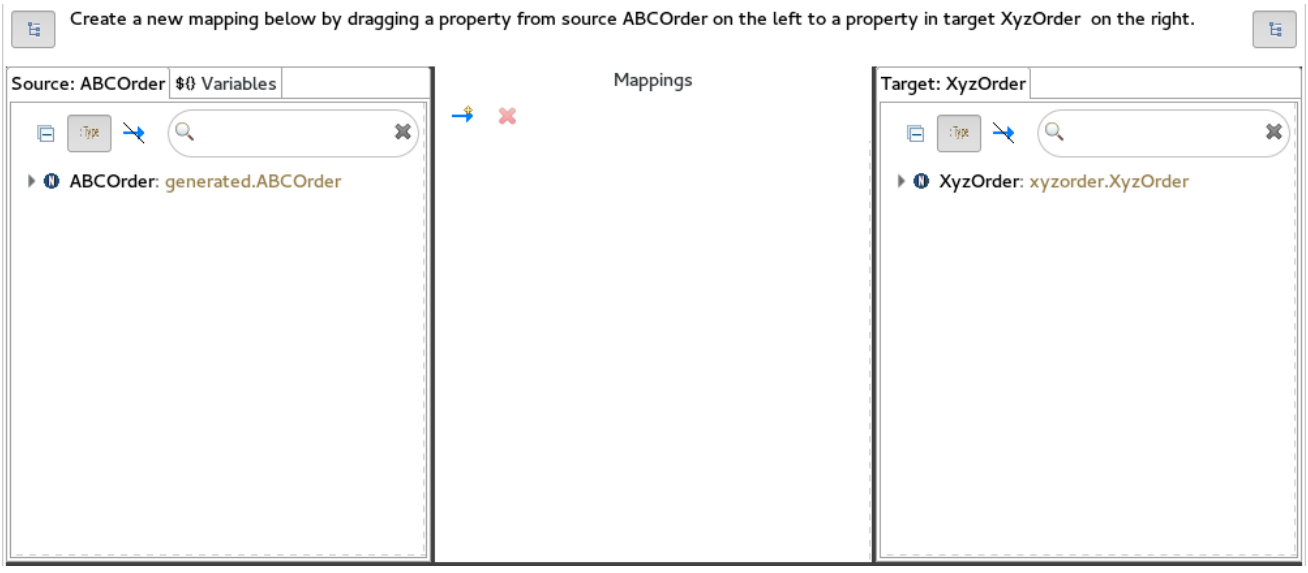
9. 单击 **JSON Instance Document**。

在 Target File 字段中，输入 xyz-order.json 实例文档的路径，或者浏览到它。JSON 结构预览 窗格显示 JSON 数据结构的预览：



10. 点 **Finish**。

此时会打开转换编辑器。在这里，您可以将 XML 源中的数据项映射到 JSON 目标中的数据项。



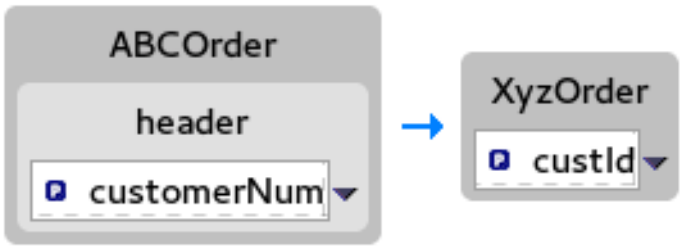
转换编辑器由三个面板组成：

- source wagon-wagonlists 源的可用数据项

- 映射 HEKETI- swig 显示源和目标数据项之间的映射

- 目标 HEKETI-wagonlist 为目标的可用数据项

此外，编辑器的详细信息窗格（位于编辑器的三个面板下面）（一旦进行了第一个映射），以图形方式显示当前选择的映射源和目标数据项目的分层级别。例如：



使用详情窗格，您可以自定义所选源和目标数据项目的映射：

- 设置属性 wagon-wagonModify 现有映射，或将简单的数据项映射到集合中的一个映射（请参阅 第 8.8 节 “将简单的数据项映射到集合中的数据项”）。

- 为 数据项设置一个常量值（请参阅 第 8.5 节 “将恒定变量映射到数据项”）。

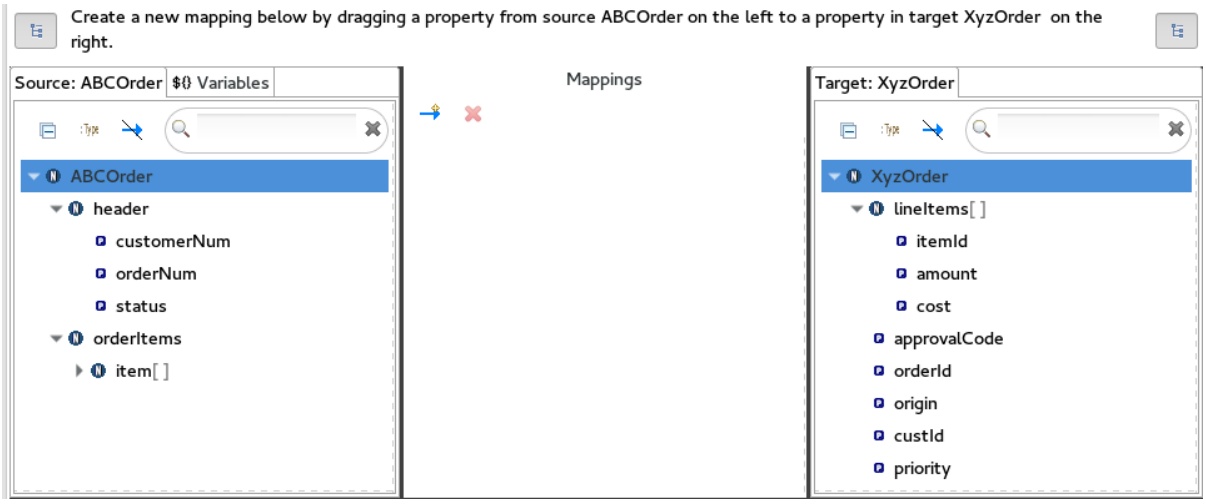
- 将 expression swig- swigMap 一个数据项设置为指定表达式的动态评估（请参阅 第 8.6 节 “将表达式映射到数据项”）。

- 使用内置功能（请参阅 第 8.9 节 “将内置功能添加到映射的数据项中”） 添加转换 的属性数据项的值。

- 使用您创建的 Java 方法，或添加您之前创建的 Java 方法（请参阅 第 8.7 节 “将自定义转换添加到映射的数据项”），添加自定义转换 信息项的值。

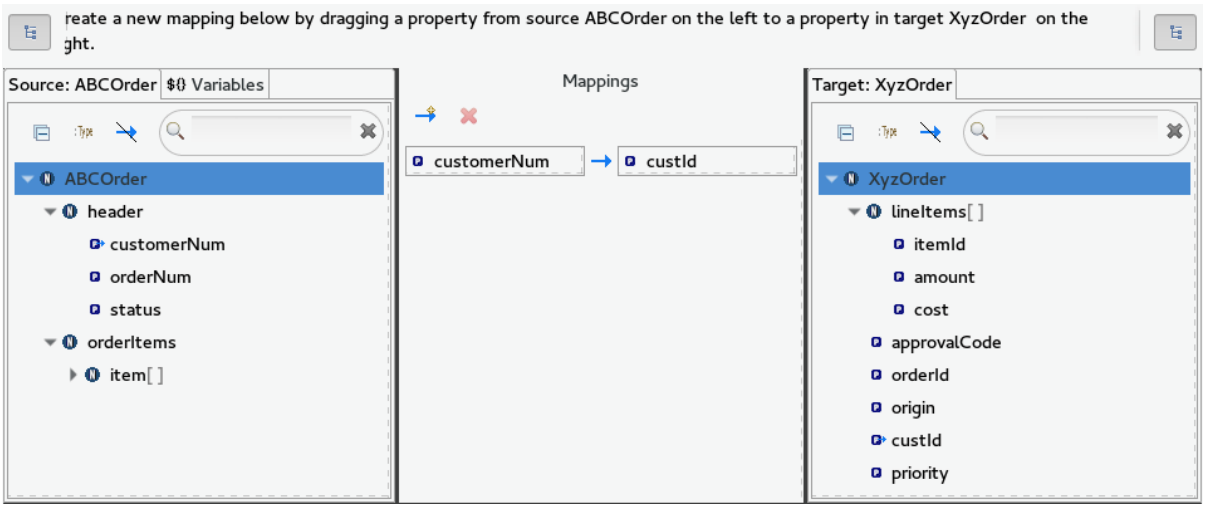
8.3. 将源数据项映射到目标数据项

1. 展开位于左侧的 Source 和 Target 面板中的所有项，并在 Mappings 面板的右侧展开。



2. 从 Source 面板中拖动数据项，并将它放到 Target 面板中的对应数据项上。

例如，将 customerNum 数据项拖到 Source 面板中，并将其放到 Target 面板中的 custId 数据项中。

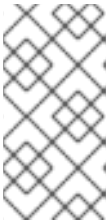


映射会出现在映射面板中，Source 和 Target data 项的详情都会出现在详情窗格中。

3. 继续将源数据项拖放到对应的目标数据项，直到您完成所有基本映射。

在 初学者 示例中，要映射的其余数据项是：

Source	目标
orderNum	orderId
status	priority
id	itemId
价格	cost
quantity	amount



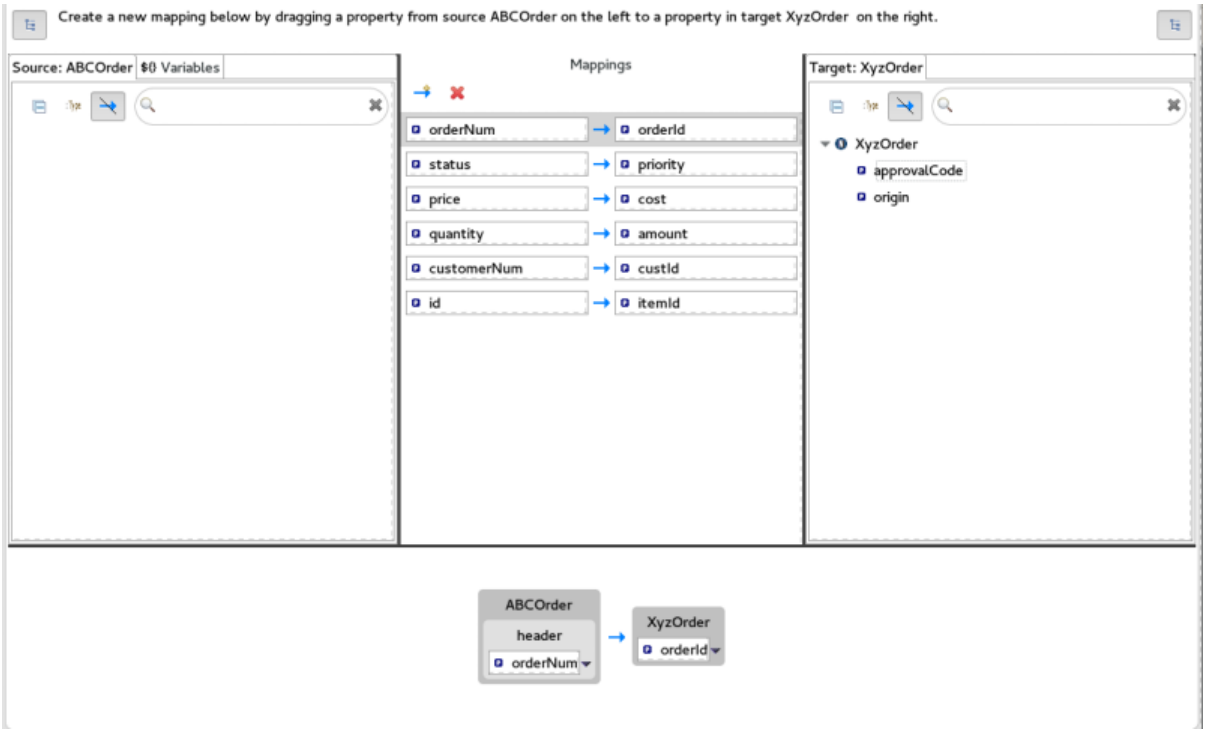
注意

您可以将集合（包含列表或集合的数据项）映射到非集合数据项，反之亦然，但不能将集合映射到其他集合。

4.

在 Source 和 Target 面板上点

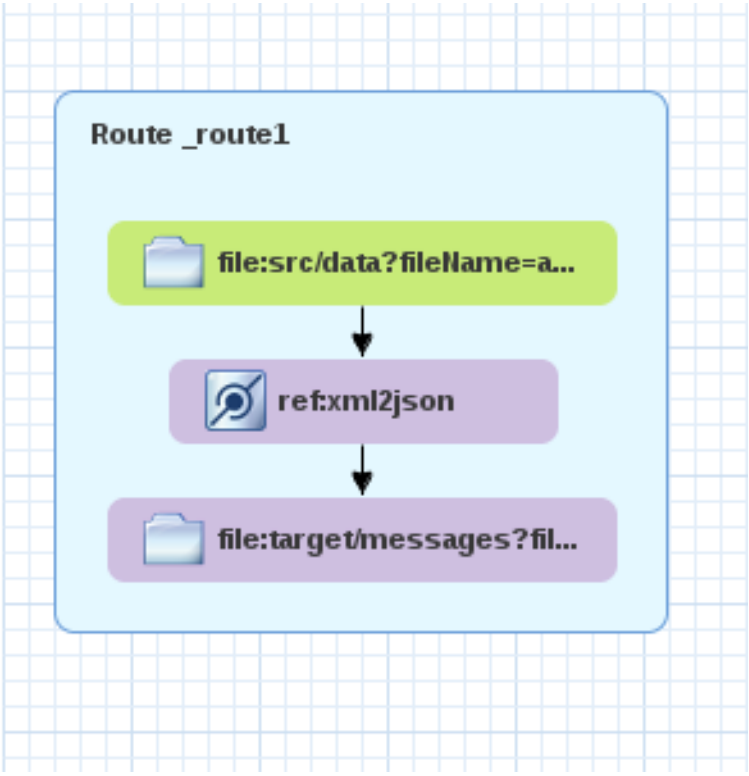
来快速确定所有数据项是否已映射。



Source 和 Target 面板中仅列出尚未映射的数据项。

在 初学者 示例中，剩余的未映射 目标 属性是 approvalCode 和 origin。

5. 点 blueprint.xml 标签页返回路由的图形显示：



6. 点 File → Save。

在创建转换测试后，您可以在转换文件上运行 JUnit 测试。详情请查看 第 8.4 节 “创建转换测试文件并运行 JUnit 测试”。此时，您将在 Console 视图中看到此输出：

对于源 XML 数据：

```
<?xml version="1.0" encoding="UTF-8"?>
<ABCOrder xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:java="http://java.sun.com">
    <header>
        <status>GOLD</status>
        <customer-num>ACME-123</customer-num>
        <order-num>ORDER1</order-num>
    </header>
    <order-items>
        <item id="PICKLE">
            <price>2.25</price>
            <quantity>1000</quantity>
        </item>
        <item id="BANANA">
```

```
        <price>1.25</price>
        <quantity>400</quantity>
      </item>
    </order-items>
</ABCOrder>
```

对于目标 **JSON** 数据：

```
{"custId":"ACME-123","priority":"GOLD","orderId":"ORDER1","lineItems":[{"itemId":"PICKLE",
"amount":1000,"cost":2.25},{"itemId":"BANANA","amount":400,"cost":1.25
```

8.4. 创建转换测试文件并运行 JUNIT 测试

1. 右键单击 **Project Explorer** 视图中的 初学者 项目，然后选择 **New → Other → Fuse Tooling → Fuse Transformation Test**。

2. 选择 **Next** 以打开 **New Transformation Test** 向导。

3. 在 **New Transformation Test** 向导中，设置以下值：

字段	value
软件包	示例
Camel 文件路径	OSGI-INF/blueprint/blueprint.xml
转换 ID	xml2json

4. 点 **Finish**。

5. 在 **Project Explorer** 视图中，导航到 **starter/src/test/java/example**，然后打开 **TransformationTest.java** 文件。

6. 将以下代码添加到 转换方法中 ：

```
startEndpoint.sendBodyAndHeader(readFile("src/data/abc-order.xml"), "approvalID",
"AUTO_OK");
```

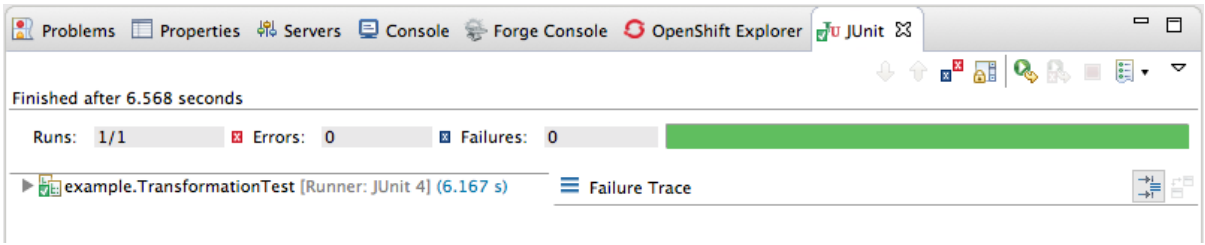
7. 点 File → Save。

现在，您可以在这些教程的任意点对转换文件运行 JUnit 测试。

8. 在 Project Explorer 视图中，展开 starter 项目，以公开 /src/test/java/example/TransformationTest.java 文件。

9. 右键单击该上下文菜单，再选择 Run as JUnit Test。

此时会打开 JUnit Test 窗格，以显示测试的状态。为避免消除工作区，请将窗格拖放到 控制台 视图旁边的底部面板中。



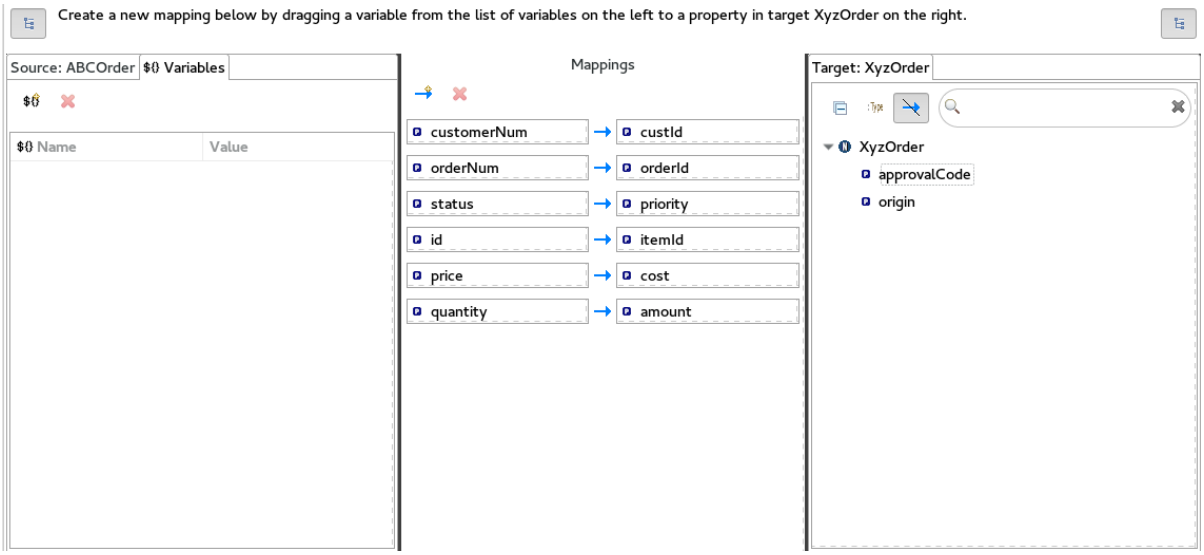
10. 打开 Console 视图 以查看日志输出。

8.5. 将恒定变量映射到数据项

当源/目标数据项没有对应的目标/源数据项时，您可以将常量变量映射到现有数据项。

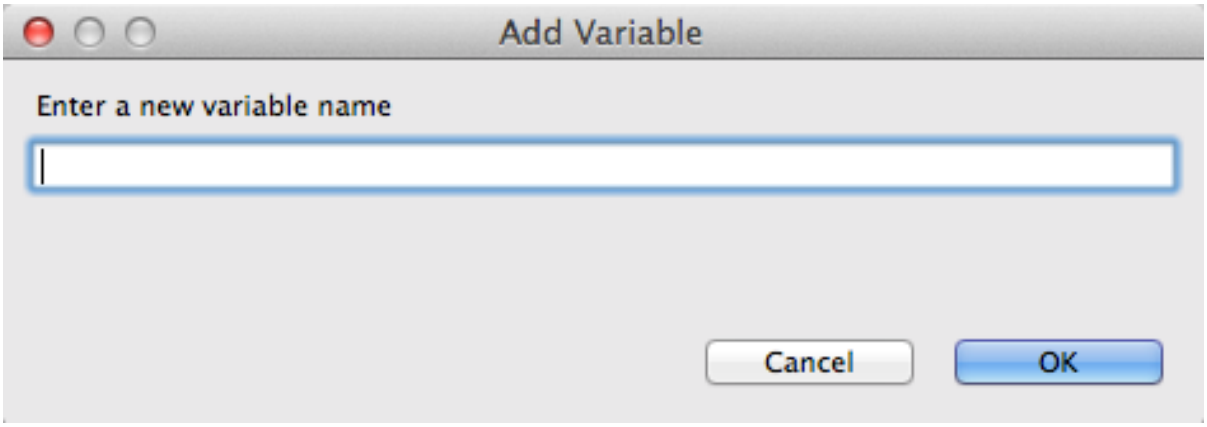
在 初学者 示例中，目标数据项 origin 没有对应的源数据项。将 origin 属性映射到常量变量：

1. 在 Source 面板中，点 Variables 视图。



2. **在 Variables 视图中，点**

打开 Enter a new variable name 对话框。



3. **输入您要创建的变量的名称。**

   对于 **初学者 示例，**请输入 **ORIGIN。**

4. **点击 确定。**

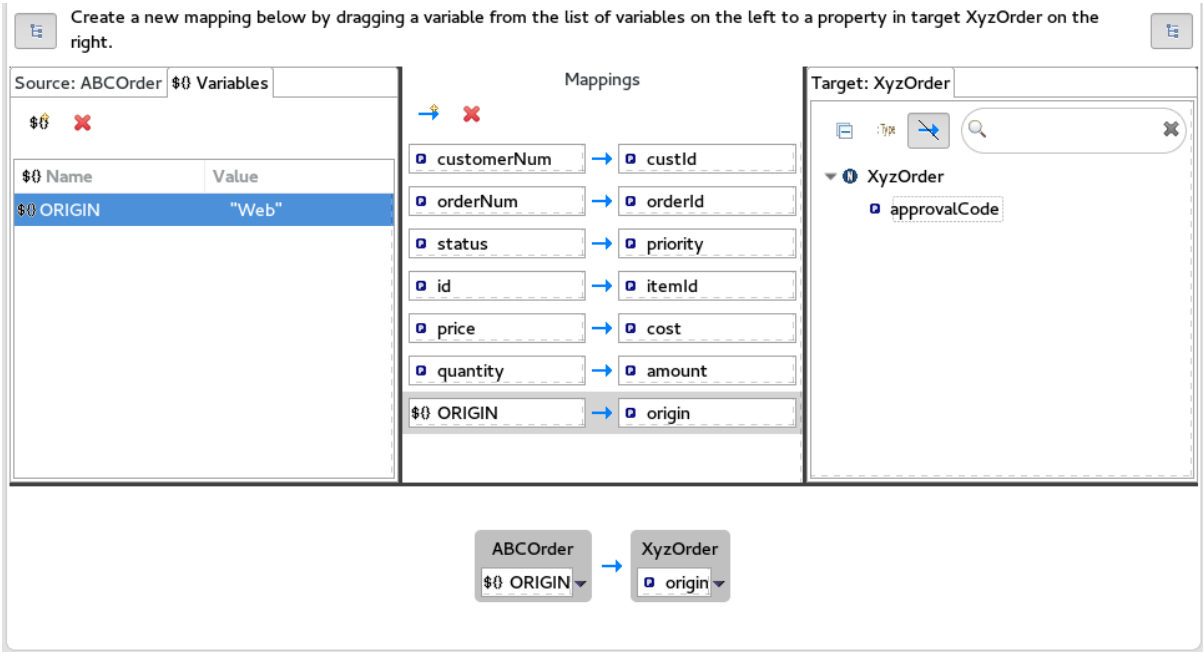
   新创建的变量 ORIGIN 会出现在 Name 列中的 Variables 视图中，默认值为 ORIGIN。

5. 单击**默认值进行编辑，并将值更改为 Web。**

6. **按 Enter 键。**

7.

将新变量 ORIGIN 拖放到 Target 面板中的原始数据项。



变量 $(ORIGIN) 的新映射会出现在 Mappings 面板和详情窗格中。

8.

在 TransformationTest.java 文件中运行 JUnit 测试。详情请查看 第 8.4 节 “创建转换测试文件并运行 JUnit 测试”。

Console 视图显示 JSON 格式的输出数据：

```
{"custId":"ACME-123","priority":"GOLD","orderId":"ORDER1","origin":"Web",
"approvalCode":"AUTO_OK","lineItems":[{"itemId":"PICKLE","amount":1000,"cost":2.25},
{"itemId":"BANANA","amount":400,"cost":1.25}]}
```

8.6. 将表达式映射到数据项

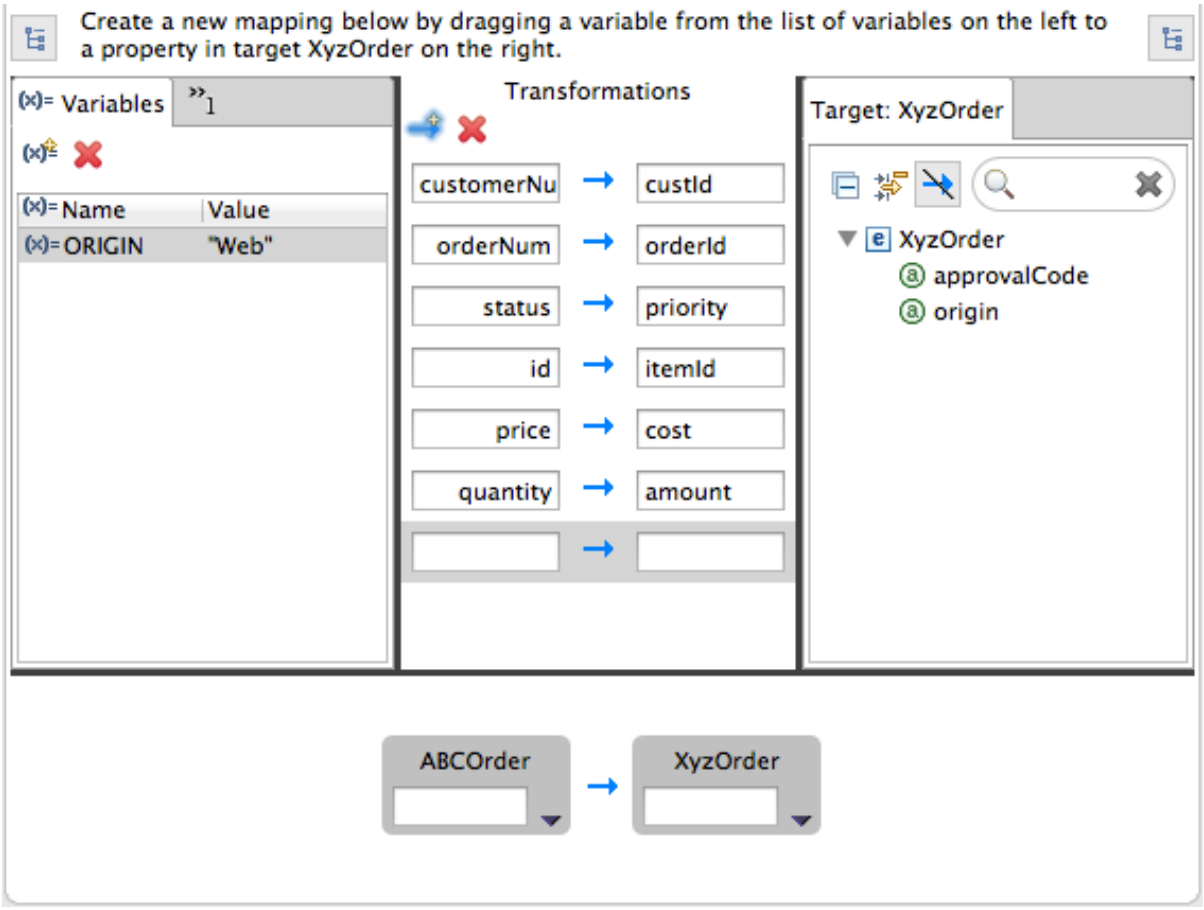
例如，这个功能允许您将目标数据项映射到 Camel 语言表达式的动态评估中。

使用目标 approvalCode 数据项，它缺少对应的源数据项：

1.

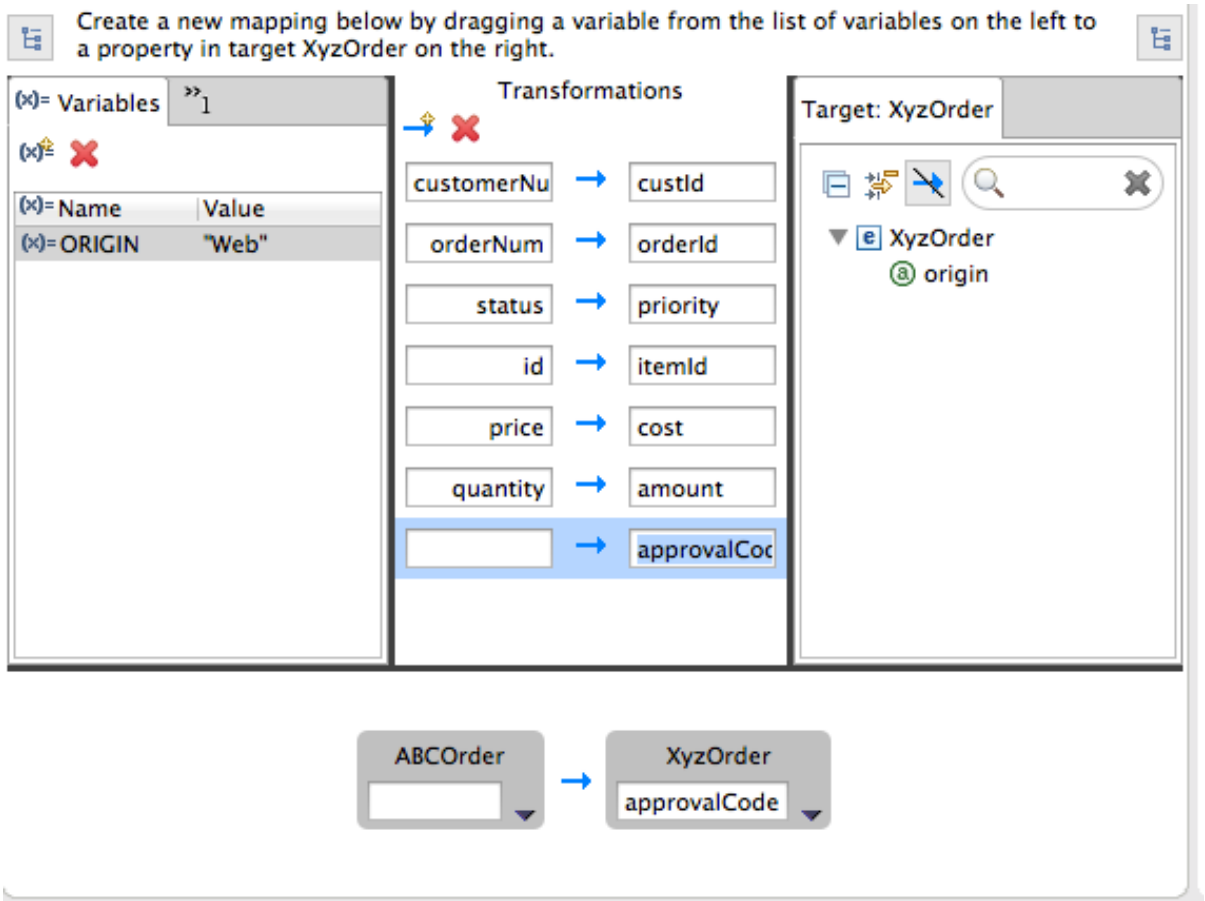
点击

在 Mappings 面板中添加空转换映射。



2.

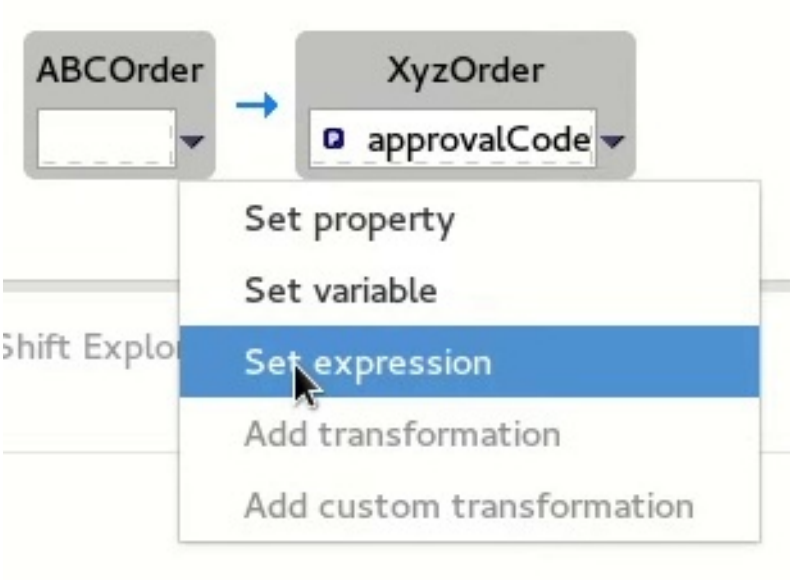
在 Target 面板中，将 approvalCode 数据项拖放到 Mappings 面板中新创建的映射的目标字段。



批准代码 数据项也会出现在详情窗格的目标框中。

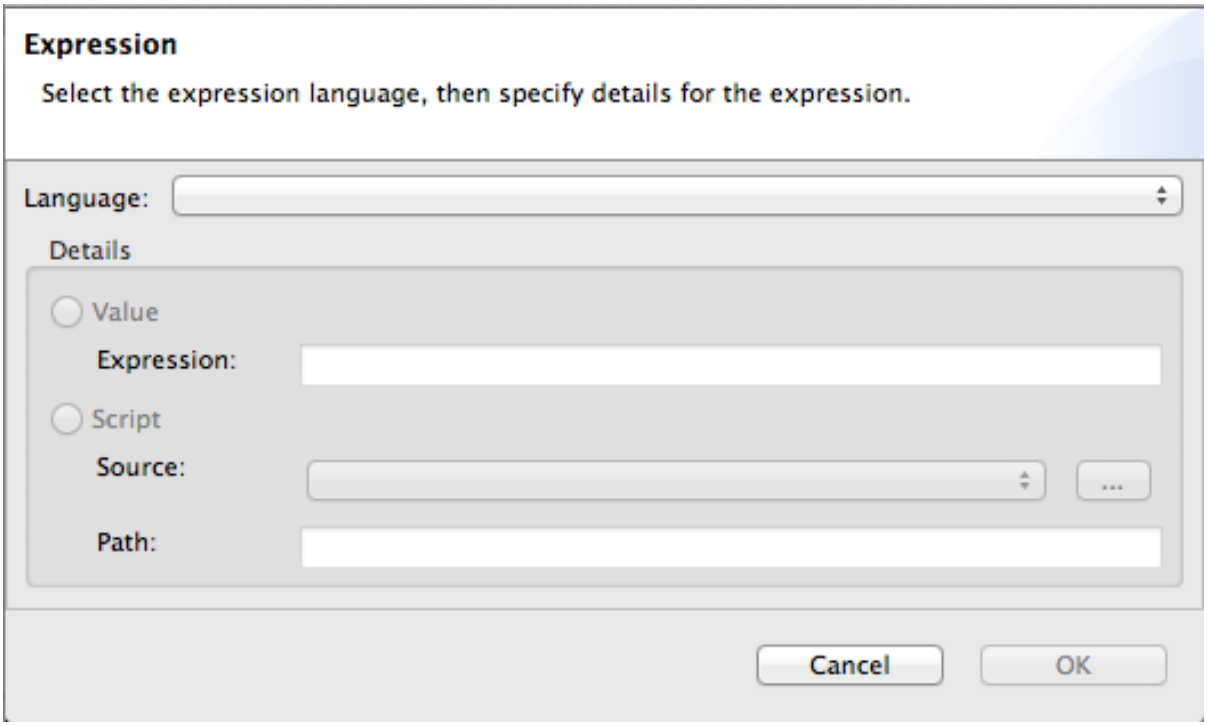
3. 在详情窗格中，点 ABCOrder 源框中的

来打开下拉菜单。



菜单选项取决于所选数据项目的数据类型。可用选项被粗体显示。

4. 选择 Set expression 以打开 Expression 对话框。



5. 在 Language 中，从可用的列表中选择要使用的表达式语言。可用的选项取决于 data 项的

数据类型。

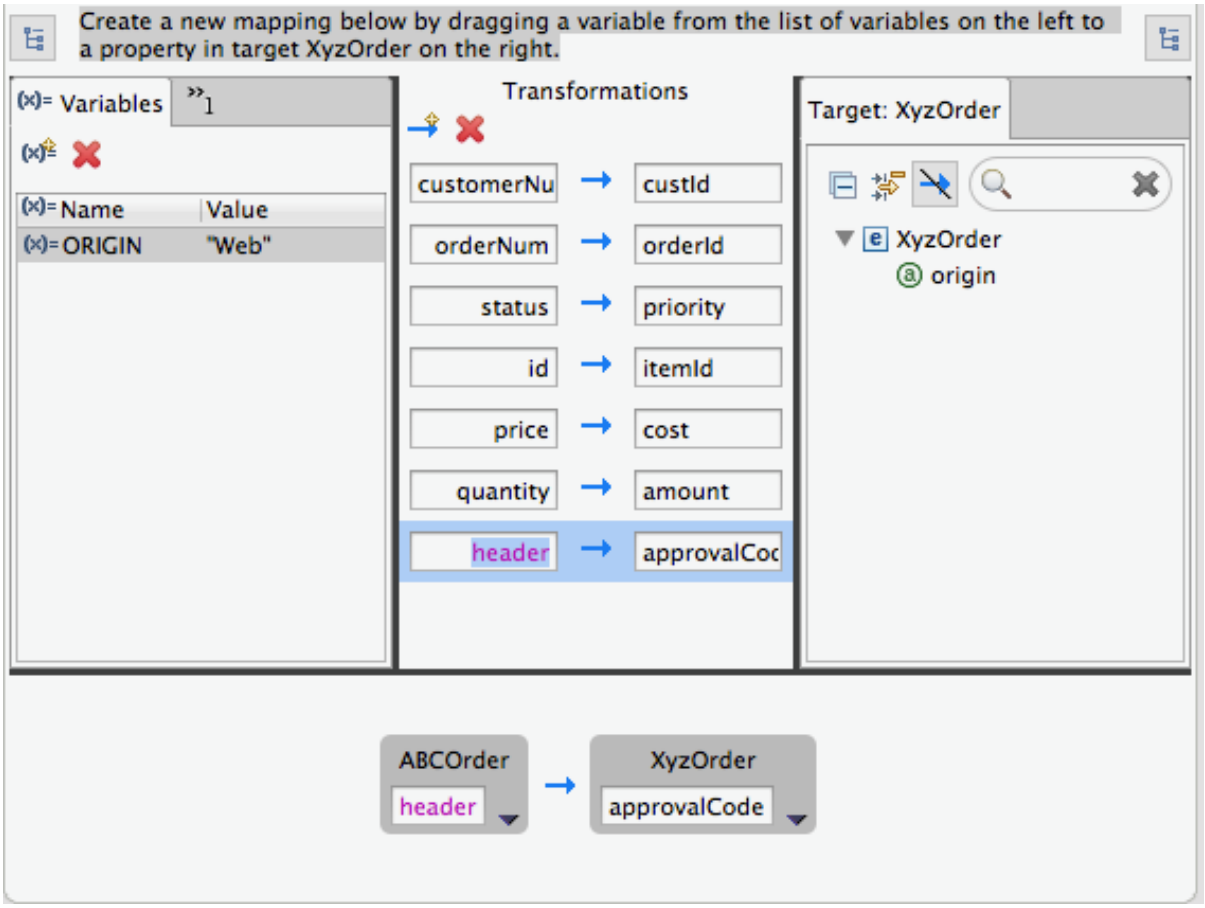
对于 **初学者 示例，** 请选择 **Header。**

6. **在详细信息窗格中，选择要使用的表达式源。**

选项为 **Value 和 Script。**

对于 **初学者 示例，** 单击 **Value，然后输入 ApprovalID。**

7. **点击 确定。**



Mappings 面板和详情窗格都显示目标数据项 approvalCode 的新映射。

8. **在 TransformationTest.java 文件中运行 JUnit 测试。详情请查看 第 8.4 节 “创建转换测试文件并运行 JUnit 测试”。**

Console 视图显示 JSON 格式的输出数据：

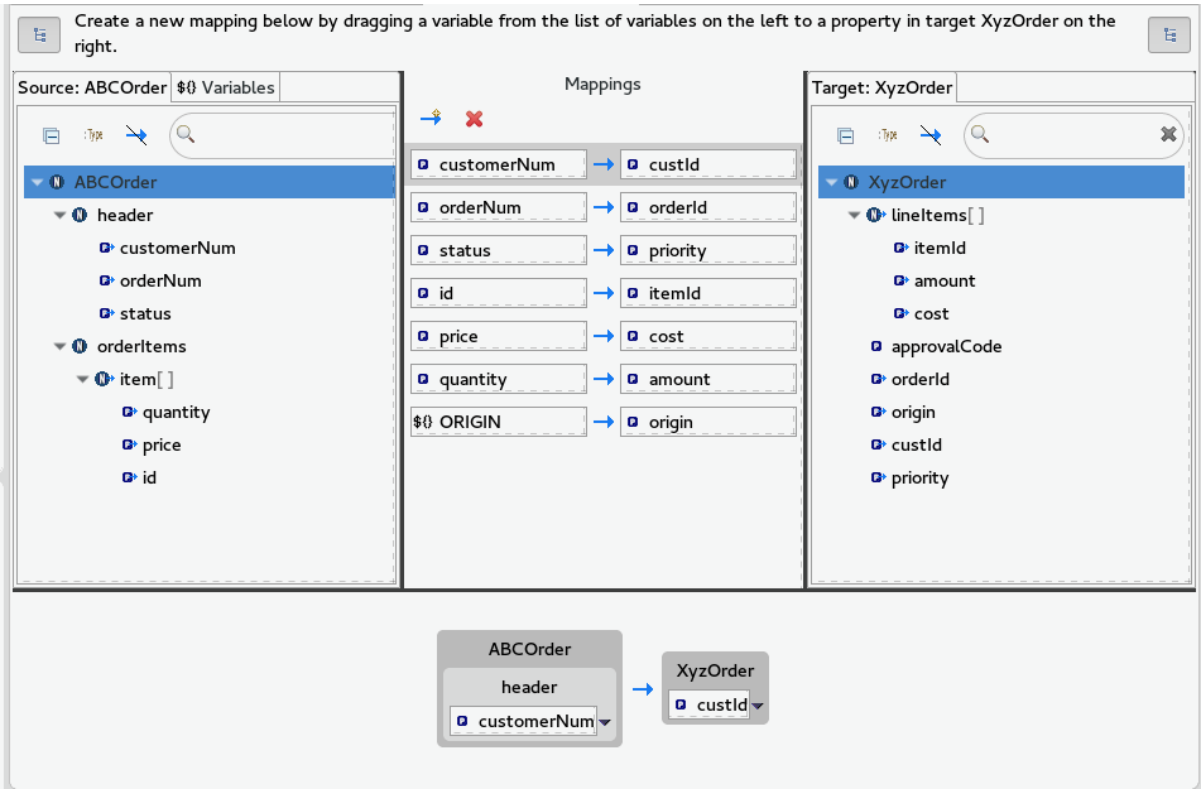
```
{"custId":"ACME-123","priority":"GOLD","orderId":"ORDER1","origin":"Web",
"approvalCode":"AUTO_OK","lineItems":[{"itemId":"PICKLE","amount":1000,"cost":2.25},
{"itemId":"BANANA","amount":400,"cost":1.25}]}
```

8.7. 将自定义转换添加到映射的数据项

当源数据项无法满足目标系统的要求时，您可能需要修改源数据项的格式。

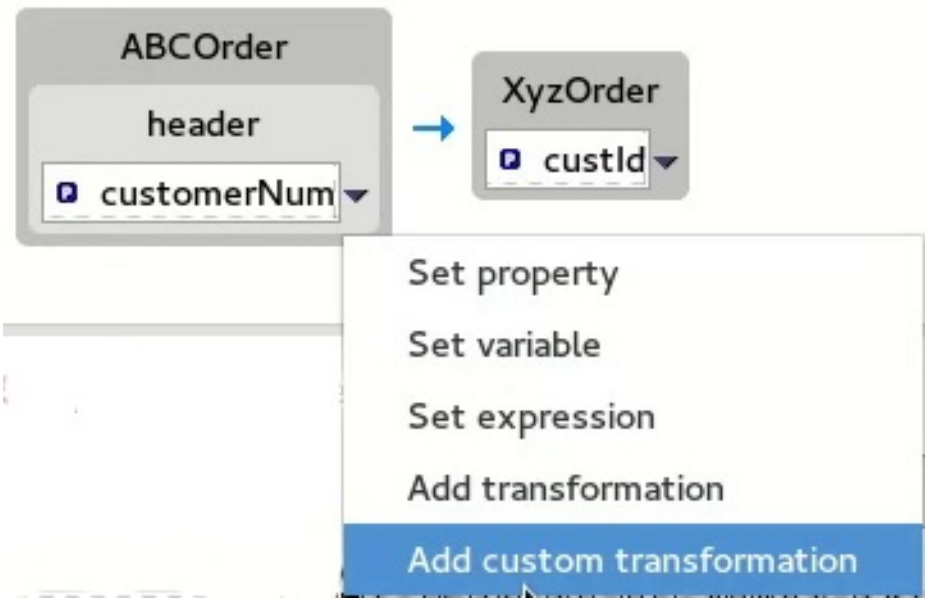
例如，要满足目标系统的要求，所有客户 ID 都用括号括起来：

1. **在 Mappings 面板中，选择 customerNum 映射来填充详情窗格。**

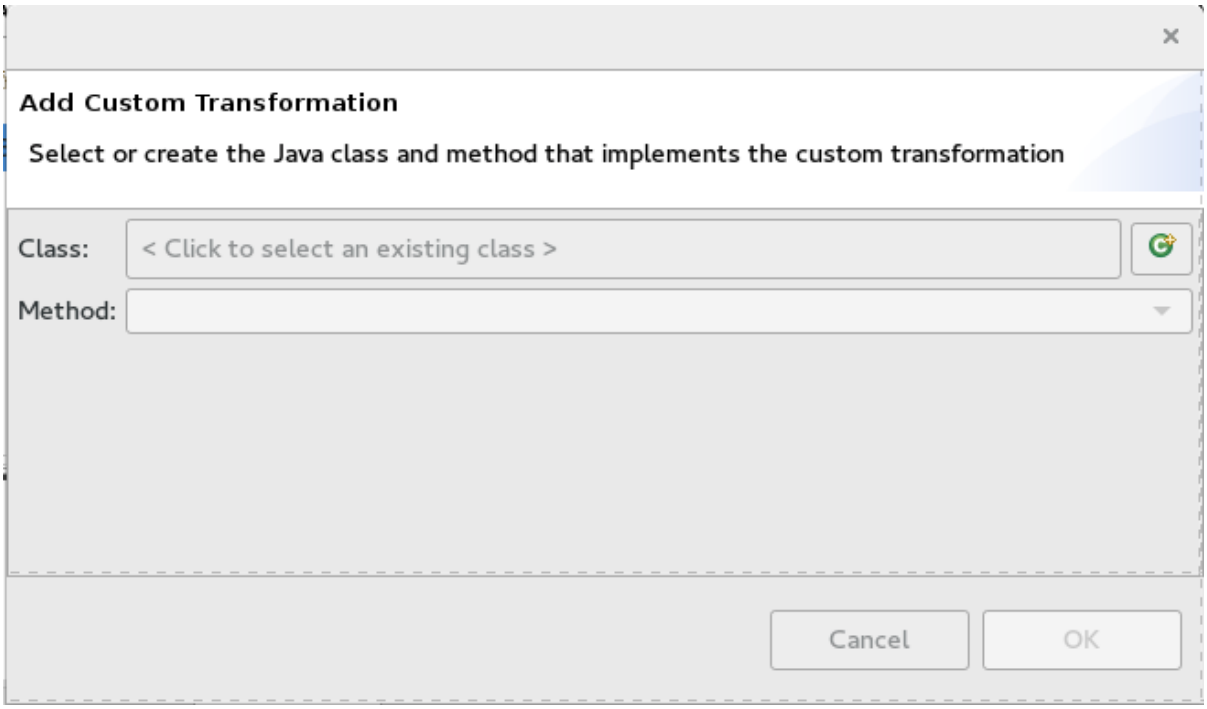


2. **在详情窗格中，点 ABCOrder 源框中的**

来打开下拉菜单。

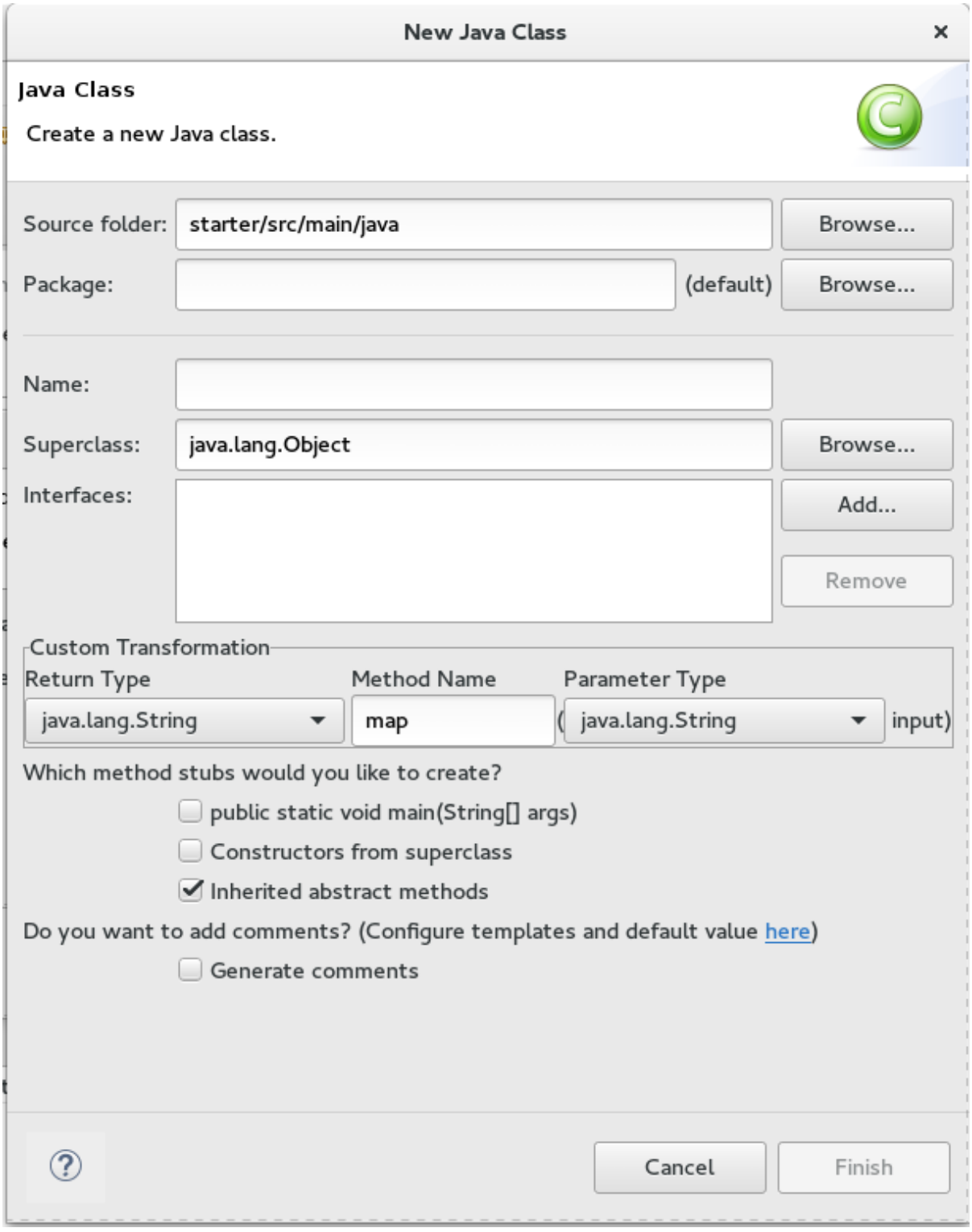


3. 选择 **Add custom transformation** 以打开 **Add Custom Transformation** 页面。



4. **点 Class 字段旁的**

打开 Create a New Java Class 向导。



5. **修改以下字段：**

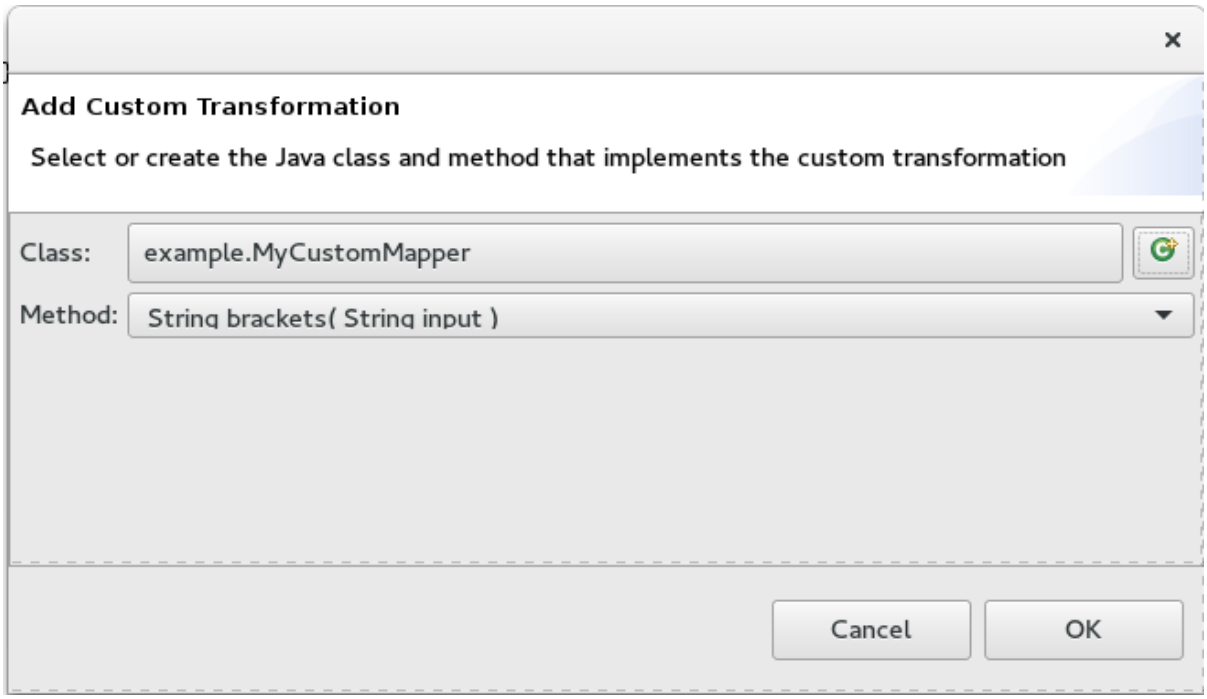
- **软件包 wagon-wagonEnter 示例.**
- **名称 wagon-wagonEnter MyCustomMapper.**

- 方法名称 HEKETI-wagonChange 映射到 方括号。

  所有其他字段保留原样。

6. 点 Finish。

   Add Custom Transformation 页面将打开，并自动填充 Class 和 Method 字段：



7. 单击 OK 以打开 Java 编辑器中的 MyCustomMapper.java 文件：

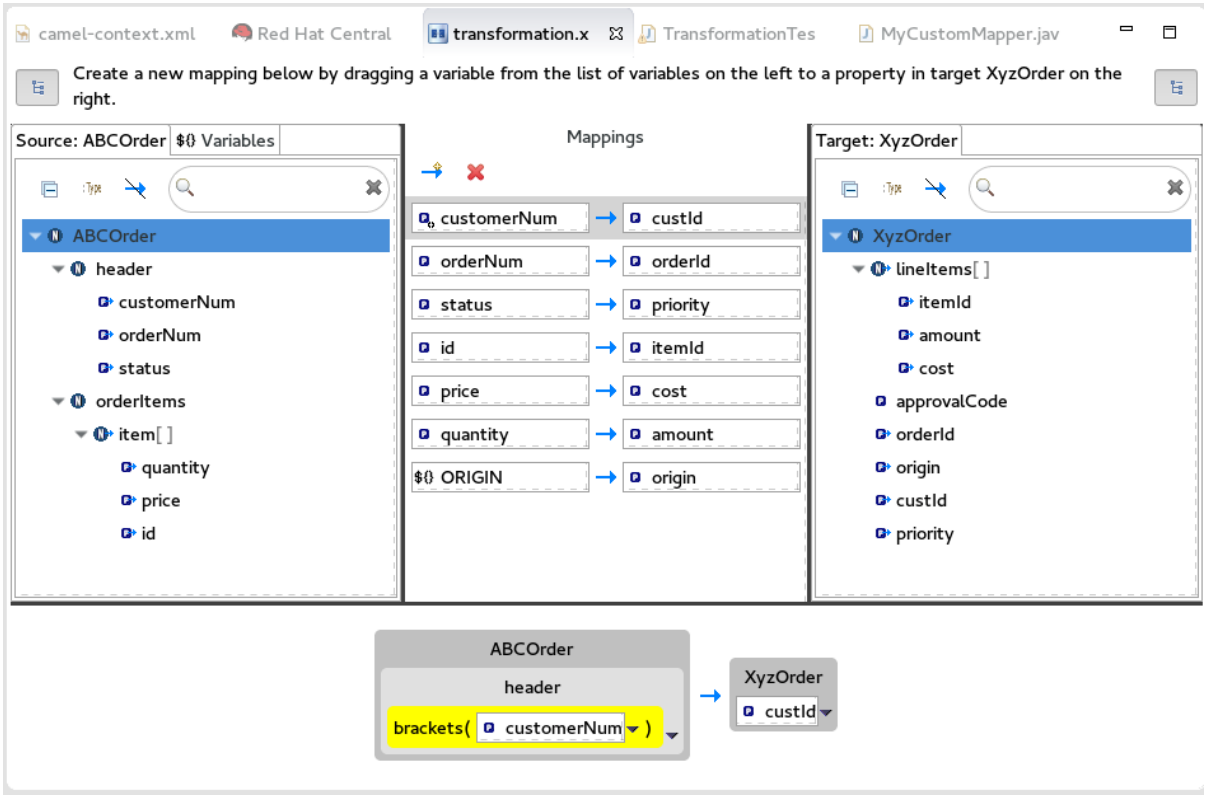
```
package example;

public class MyCustomMapper {

    public String brackets(String input) {
        return null;
    }

}
```

8. 编辑 方括号 方法，将最后一行 返回 null； 例如：

```
return "[" + input + "]";
```

9.

点 transformation.xml 选项卡，以切回到转换编辑器。



详情窗格显示 方括号 方法已与 customerNum 数据项关联。

括号 方法在将源输入发送到目标系统之前对它执行。

10.

在 TransformationTest.java 文件中运行 JUnit 测试。详情请查看 第 8.4 节 “创建转换测试文件并运行 JUnit 测试”。

Console 视图显示 JSON 格式的输出数据：

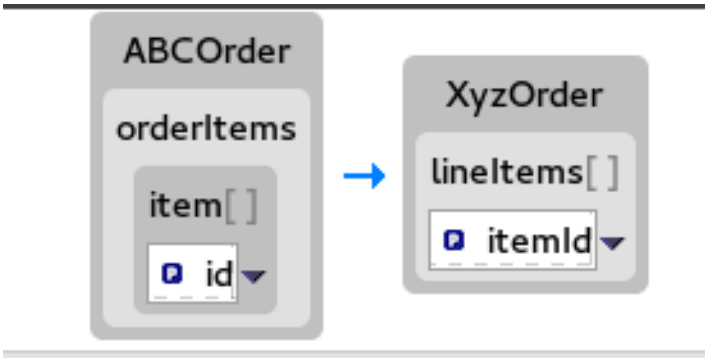
```
{"custId":"[ACME-123]","priority":"GOLD","orderId":"ORDER1","origin":"Web",
"approvalCode":"AUTO_OK","lineItems":[{"itemId":"PICKLE","amount":1000,"cost":2.25},
{"itemId":"BANANA","amount":400,"cost":1.25}]}
```

8.8. 将简单的数据项映射到集合中的数据项

在本教程中，您将修改一个现有映射，该映射将 Source 中的所有 ids 映射到 Target 中的 itemIds。新映射将 Source 中的 customerNum 数据项映射到 Target 中 lineItems 集合中第二个项目的 itemId。

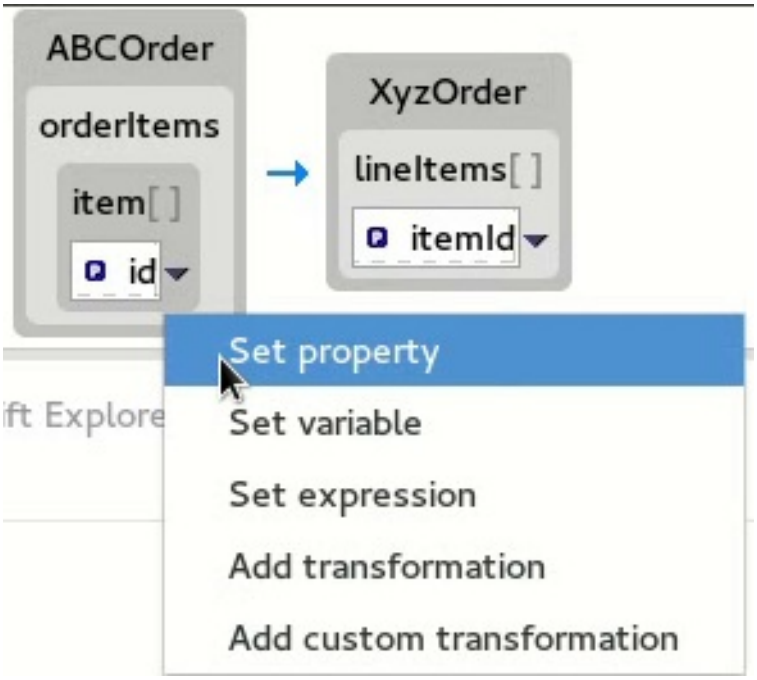
在这个版本中，Source 中没有 ID s 映射到 Target 中的 itemIds。

1. 在 Mappings 面板中，选择映射 id HEKETI- swig& gt; 项Id 以在详情窗格中显示映射。

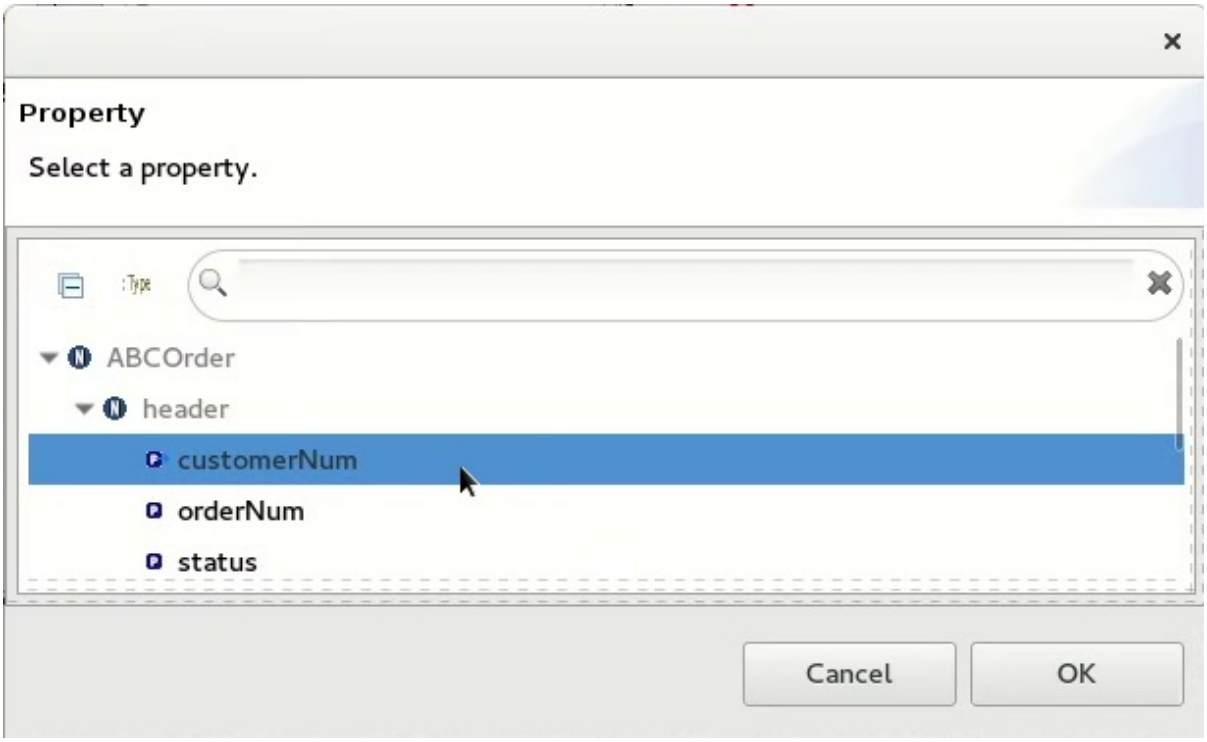


2. 在 Source 框中，点

打开下拉菜单，然后选择 Set property。



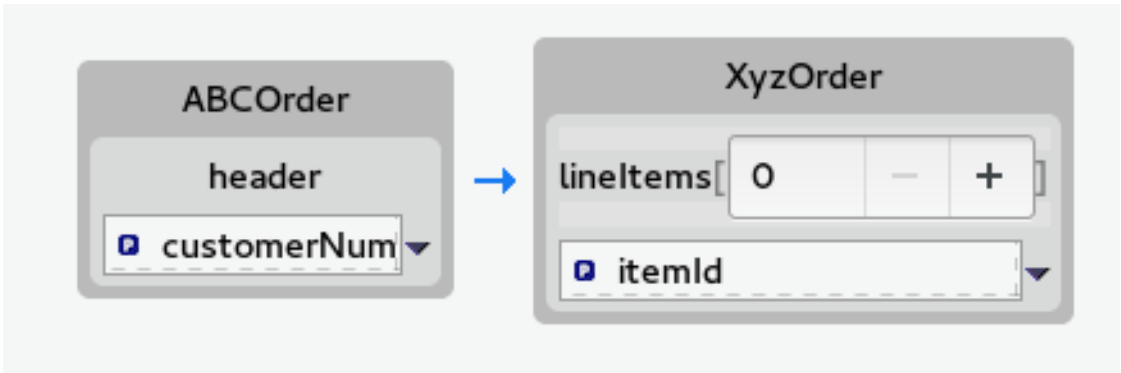
3. 在 Select a property 页面中，展开 标头 节点并选择 customerNum。点 OK 保存更改。



4. 详情窗格现在显示 XyzOrder 有一个 lineItems 字段。单击 lineItems 旁边的切换按钮，将其值增加到 1。

注意

索引基于零，因此值 1 在集合中选择 itemId 的第二个实例。



请注意，详细信息窗格显示 customerNum 映射到 lineItems 集合中第二个项目的 itemId。

5. 在 TransformationTest.java 文件中运行 JUnit 测试。详情请查看 第 8.4 节 “创建转换测试文件并运行 JUnit 测试”。

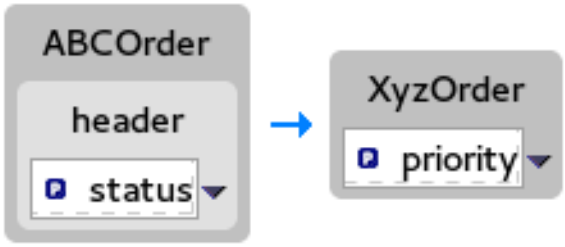
Console 视图显示 JSON 格式的输出数据：

{"custId":"[ACME-123]","priority":"GOLD","orderId":"ORDER1","origin":"Web",
"approvalCode":"AUTO_OK","lineItems":[{"amount":1000,"cost":2.25},
{"itemId":"ACME-123","amount":400,"cost":1.25}]}

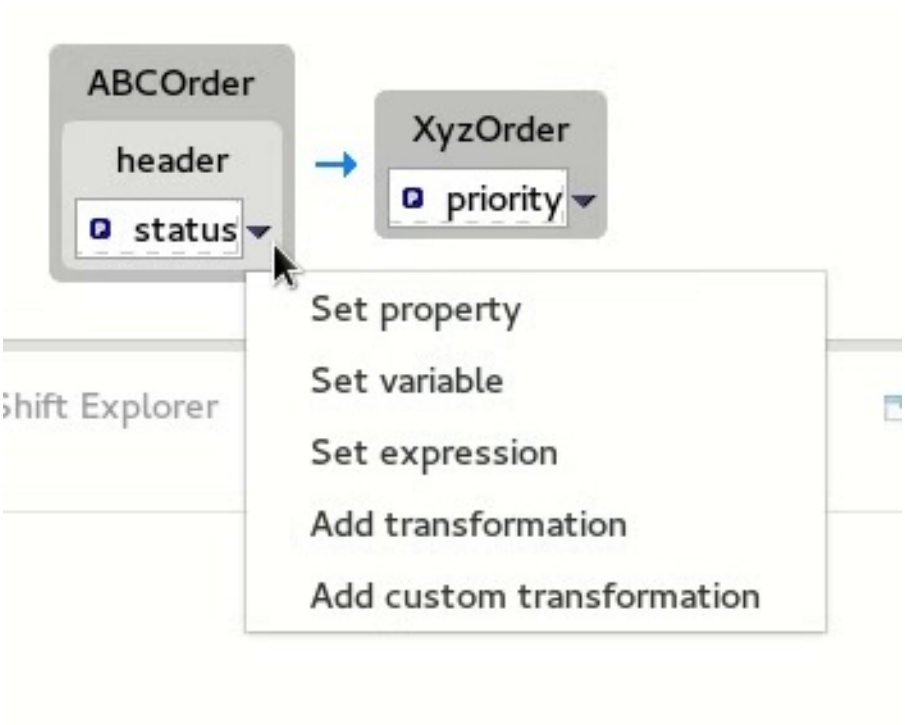
8.9. 将内置功能添加到映射的数据项中

您可以使用内置的字符串相关功能将转换应用到映射的数据项。

1. 在 Transformations 面板中，选择 优先级 映射以填充详情窗格 的状态。

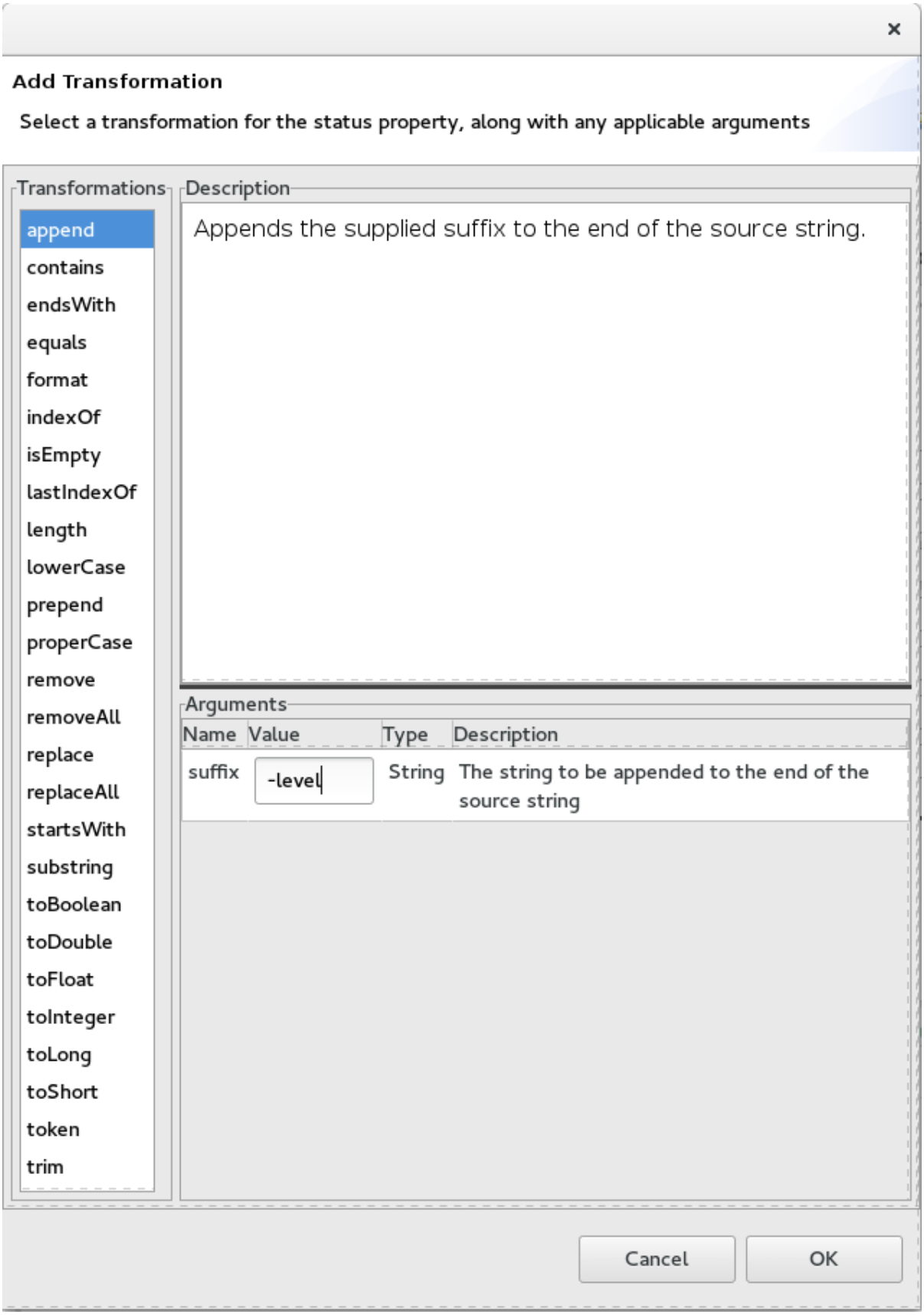


2. 在 Source 框中，点 打开下拉菜单，然后选择 Add transformation。

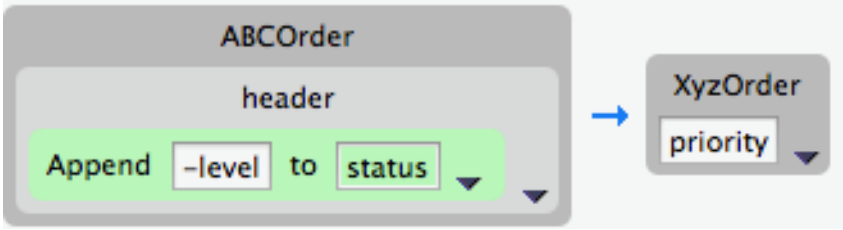


3. 在 Transformations 窗格中，在 Arguments 窗格中选择 附加，输入 后缀 值 级别。

此附加功能将指定的后缀添加到 状态 字符串的末尾，然后将其映射到目标 优先级 数据项。

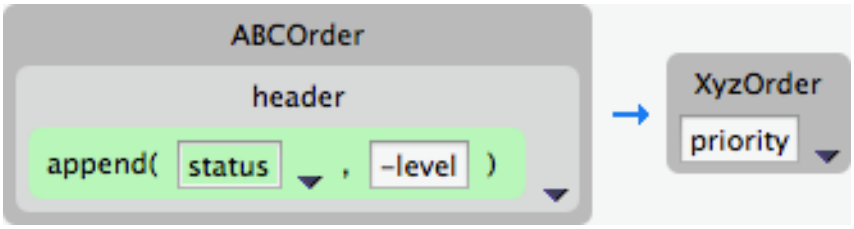

Add Transformation
Select a transformation for the status property, along with any applicable arguments
Transformations
append
contains
endsWith
equals
format
indexOf
isEmpty
lastIndexOf
length
lowerCase
prepend
properCase
remove
removeAll
replace
replaceAll
startsWith
substring
toBoolean
toDouble
toFloat
toInteger
toLong
toShort
token
trim
Description
Appends the supplied suffix to the end of the source string.
Arguments
Name
Value
Type
Description
suffix
-level
String
The string to be appended to the end of the source string
Cancel
OK

4. 点击 **确定**。



默认情况下，详细信息窗格以用户友好的格式显示将 append 功能添加到 状态 数据项的结果。您可以在 Source 框中点右边

，然后选择 Show standard Format 来更改此格式。



5. 在 TransformationTest.java 文件中运行 JUnit 测试。详情请查看 第 8.4 节 “创建转换测试文件并运行 JUnit 测试”。

Console 视图显示 JSON 格式的输出数据：

```
{"custId":"[ACME-123]","priority":"GOLD-level","orderId":"ORDER1","origin":"Web",
"approvalCode":"AUTO_OK","lineItems":[{"amount":1000,"cost":2.25},{"itemId":"ACME-123",
"amount":400,"cost":1.25}]}
```

8.10. 将 FUSE 集成项目与数据转换发布到红帽 FUSE 服务器

在将数据转换项目发布到 Fuse 服务器（请参阅 第 28 章 *将 Fuse 集成项目发布到服务器*）前，您需要在 Fuse 运行时中安装以下功能：

- camel-dozer
- camel-jackson
- camel-jaxb

在 Fuse 运行时安装所需的功能：

1. 如果还没有存在，切换到 Fuse Integration 视角。

2. 如有必要，将 Fuse 服务器添加到 服务器 列表中（请参阅 第 27.1 节 “添加服务器”）。

3. 启动 Fuse 服务器（请参阅 第 27.2 节 “启动服务器”），并等待 JBoss Fuse shell 出现在 Terminal 视图中。

4. 对于每个所需的 camel- 功能，在 JBossFuse:admin@root > 提示类型中：

   features:install camel-<featureName>

   其中 *featureName* 是 dozer,jackson, 或 jaxb 之一。

5. 要验证每个功能是否已成功安装，在 JBossFuse:admin@root > 提示类型中：

   features:list --ordered --installed

   您应该在输出列表中看到您刚才安装的 camel 功能：

```
[installed  ] [2.17.0.redhat-630159] camel-dozer                 camel-2.17.0.redhat-630159
[installed  ] [2.17.0.redhat-630159] camel-exec                  camel-2.17.0.redhat-630159
[installed  ] [2.17.0.redhat-630159] camel-ftp                   camel-2.17.0.redhat-630159
[installed  ] [2.17.0.redhat-630159] camel-jackson               camel-2.17.0.redhat-630159
[installed  ] [2.17.0.redhat-630159] camel-jasypt                camel-2.17.0.redhat-630159
[installed  ] [2.17.0.redhat-630159] camel-jaxb                  camel-2.17.0.redhat-630159
```

第 9 章 为 FUSE ONLINE 集成开发扩展

Fuse Online 是红帽 Fuse 功能，为集成应用程序提供 Web 界面。如果没有编写代码，业务专家可以使用 Fuse Online 连接到应用程序，并在不同应用程序的连接之间选择性地操作数据。如果 Fuse Online 不提供集成商所需的功能，则开发人员可以创建定义所需行为的扩展。

您可以使用 Fuse 工具开发提供 Fuse Online 中使用功能的扩展。扩展定义：

- 在集成中的连接间对数据进行操作的一个或多个自定义 步骤

  or

- 一个自定义 连接器

在 Fuse Online 中，连接器 代表一个特定的应用程序，用于从数据获取或发送数据。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，Salesforce 连接器是用于创建与 Salesforce 的连接的模板。如果 Fuse Online 没有提供 Fuse Online 用户需要的连接器，您可以开发定义自定义连接器的扩展。

在 Fuse Online 中，集成中连接之间发生的数据操作称为 步骤。Fuse Online 为过滤和映射数据等操作提供步骤。要以不是由 Fuse Online 内置步骤提供的方式在连接之间操作数据，您可以开发一个定义一个或多个自定义步骤的 Fuse Online 扩展。

注意

您可能希望在您选择的 IDE 中开发扩展。无论您使用 Fuse 工具还是另一个 IDE，都完全是个人首选项。有关在任何 IDE 中开发扩展的信息，请参考 将应用程序与 Fuse Online 集成。

9.1. 任务概述

以下是用于开发 Fuse Online 扩展的任务概述：

1. 创建 Fuse Online 扩展项目，然后选择 Custom Connector 或 Custom Step 作为扩展类型。

2. 根据扩展类型，编写扩展代码：

   - 对于 Custom Connector ：定义基本 Camel 组件、连接器图标、全局连接器属性和连接器操作。
   - 对于自定义 步骤 ： 添加路由、定义操作并指定任何依赖项。

3. 构建 .jar 文件。

4. 向 Fuse Online 用户提供 .jar 文件。

Fuse Online 用户将 .jar 文件上传到 Fuse Online，这使得自定义连接器或自定义步骤可供使用。有关 Fuse 在线以及如何创建集成的详情，请参考 将应用程序与 Fuse Online 集成。

9.2. 先决条件

开始之前，您需要以下信息和知识：

- 有关 Fuse Online 自定义连接器或步骤（来自 Fuse Online 用户）所需功能的说明。
- 扩展的 Fuse Online 版本号。
- 对于自定义连接器，一个 PNG 或 SVG 格式的图标镜像文件。Fuse Online 显示集成流时使用此图标。如果没有提供图标，则 Fuse Online 会在上传包含扩展名的 .jar 时生成一个。
- 您应该熟悉：
  - Fuse Online
  - Spring Boot XML 或 Java

- Apache Camel 路由（如果要创建基于路由的步骤扩展）
- JSON
- Maven

9.3. 创建自定义连接器

在 Fuse Online 中，自定义连接器由一个或多个连接配置参数、一个或多个连接操作以及每个操作的可选配置参数组成。

以下是用于开发自定义连接器的任务概述：

1. 创建 Fuse Online 扩展项目，再选择 Custom Connector 作为扩展类型。
2. 编写扩展代码。定义基本 Camel 组件、连接器图标、全局连接器属性和连接器操作。

9.3.1. 为自定义连接器编写代码

创建 Fuse Online 扩展项目后，您将根据 Fuse Online 用户为您提供所需功能的描述编写定义自定义连接器元素的代码。表 9.1 “自定义连接器元素” 表显示您在 Fuse 工具中所创建的自定义连接器的元素如何与 Fuse Online 中的元素对应。

表 9.1. 自定义连接器元素

Fuse Tooling 元素	Fuse Online 元素	描述
global （顶级）属性	连接配置参数	当 Fuse Online 用户从此连接器创建连接时，用户将此属性的值指定为连接的配置的一部分。
操作	连接操作	在 Fuse Online 中，对于从此连接器创建的连接，Fuse Online 用户选择其中之一。
操作中定义的属性	操作配置参数	当 Fuse Online 用户配置连接执行的操作时，Fuse Online 用户将此属性的值指定为操作的一部分。

Fuse Tooling 元素	Fuse Online 元素	描述

编写为 **Fuse Online** 实施自定义连接器的代码：

1. 在 **Editor** 视图中打开 **syndesis-extension-definition.json** 文件，并编写定义全局属性的代码、自定义连接器可以执行的操作，以及每个操作的属性。

每个 全局 属性都对应于 **Fuse Online** 中的连接配置参数。每个 **action** 属性都对应于 **Fuse Online** 连接操作配置参数。在 **Fuse Online** 中，当用户选择自定义连接器时，**Fuse Online** 会提示输入每个连接配置参数的值。自定义连接器可以是使用 **OAuth** 协议的应用程序。在本例中，请务必为 **OAuth** 客户端 **ID** 指定全局属性，并为 **OAuth** 客户端 **secret** 指定另一个全局属性。**Fuse Online** 用户需要为从此连接器创建的连接指定这些参数值，才能工作。

每个连接器操作都 声明一个基本 **Camel** 组件方案。

New Fuse Online Extension Project 向导提供的示例使用 **telegram Camel** 组件方案：

```
{
  "schemaVersion" : "v1",
  "name" : "Example Fuse Online Extension",
  "extensionId" : "fuse.online.extension.example",
  "version" : "1.0.0",
  "actions" : [ {
    "id" : "io.syndesis:telegram-chat-from-action",
    "name" : "Chat Messages",
    "description" : "Receive all messages sent to the chat bot",
    "descriptor" : {
      "componentScheme" : "telegram",
      "inputDataShape" : {
        "kind" : "none"
      },
      "outputDataShape" : {
        "kind" : "java",
        "type" : "org.apache.camel.component.telegram.model.IncomingMessage"
      },
      "configuredProperties" : {
        "type" : "bots"
      }
    },
    "actionType" : "connector",
    "pattern" : "From"
```

```
  }, {
    "id" : "io.syndesis:telegram-chat-to-action",
    "name" : "Send a chat Messages",
    "description" : "Send messages to the chat (through the bot).",
    "descriptor" : {
      "componentScheme" : "telegram",
      "inputDataShape" : {
        "kind" : "java",
        "type" : "java.lang.String"
      },
      "outputDataShape" : {
        "kind" : "none"
      },
      "propertyDefinitionSteps" : [ {
        "description" : "Chat id",
        "name" : "chatId",
        "properties" : {
          "chatId" : {
            "kind" : "parameter",
            "displayName" : "Chat Id",
            "type" : "string",
            "javaType" : "String",
            "description" : "The telegram's Chat Id, if not set will use CamelTelegramChatId from
the incoming exchange."
          }
        }
      } ],
      "configuredProperties" : {
        "type" : "bots"
      }
    },
    "actionType" : "connector",
    "pattern" : "To"
  } ],
  "properties" : {
    "authorizationToken" : {
      "kind" : "property",
      "displayName" : "Authorization Token",
      "group" : "security",
      "label" : "security",
      "required" : true,
      "type" : "string",
      "javaType" : "java.lang.String",
      "secret" : true,
      "description" : "Telegram Bot Authorization Token"
    }
  }
}
```

2.

如果自定义连接器需要额外的依赖项，请将它们添加到项目的 pom.xml 文件中。依赖项的默认范围是 runtime。如果您添加了红帽提供的依赖项，请定义其范围，例如：

```
<dependencies>
  <dependency>
```

```
        <groupId>org.apache.camel</groupId>
        <artifactId>camel-telegram</artifactId>
        <scope>provided</scope>
    </dependency>
  </dependencies>
```

完成为自定义连接器编写代码后，构建 .jar 文件，如 第 9.5 节 “构建 Fuse Online 扩展 JAR 文件” 所述。

9.4. 创建自定义步骤

创建 Fuse Online 扩展项目后，您将根据 Fuse Online 用户为您提供所需功能的描述编写定义自定义步骤的代码。在一个扩展中，您可以定义多个自定义步骤，您可以使用 Camel 路由或使用 Java Bean 定义每个自定义步骤。

9.4.1. 为自定义步骤编写代码

创建 Fuse Online 扩展项目后，您将编写用于定义自定义步骤的代码，该代码基于 Fuse Online 用户所提供的所需功能的描述。

表 9.2 “自定义步骤元素” 显示在 Fuse Tooling 中创建的自定义步骤的元素如何与 Fuse Online 中的元素对应。

表 9.2. 自定义步骤元素

Fuse Tooling 元素	Fuse Online 元素	描述
操作	自定义步骤	在 Fuse Online 中，在用户导入步骤扩展后，会在 Choose a step 页面上显示自定义步骤。
操作中定义的属性	自定义步骤配置参数	在 Fuse Online 中，当用户选择自定义步骤时，Fuse Online 会提示输入配置参数的值。

编写为 Fuse Online 实施自定义步骤的代码：

1. 对于基于 Camel 路由的步骤，在 extension.xml 文件中，创建用于处理扩展目的的路由。每个路由的入口点必须与您在 syndesis-extension-definition.json 文件中定义的入口点匹配，如第 2 步所述。

对于基于 Java Bean 的步骤，请编辑 java 文件。

2. 在 syndesis-extension-definition.json 文件中，编写定义操作及其属性的代码。每个入口点都需要一个新的操作。

您创建的每个操作都对应于 Fuse Online 中的自定义步骤。您可以为每个操作使用不同类型的代码。也就是说，您可以将 Camel 路由用于一个操作，而 Java bean 用于另一个操作。

每个属性对应于 Fuse Online step 配置参数。在 Fuse Online 中，当用户选择自定义步骤时，Fuse Online 会提示输入配置参数的值。例如，自定义日志步骤可能有一个 level 参数，用于指示要发送到日志的信息量。

以下是包含扩展元数据的 .json 文件模板，包括在上传扩展并将其自定义步骤添加到集成后，用户在 Fuse Online 中填写的属性：

```
{
  "actions": [
    {
      "actionType": "extension",
      "id": "${actionId}",
      "name": "Action Name",
      "description": "Action Description",
      "tags": [
          "xml"
      ],
      "descriptor": {
        "kind": "ENDPOINT|BEAN|STEP",
        "entrypoint": "direct:${actionId}",
        "inputDataShape": {
          "kind": "any"
        },
        "outputDataShape": {
          "kind": "any"
        },
        "propertyDefinitionSteps": []
      }
    }
  ],
  "tags": [
    "feature",
    "experimental"
  ]
}
```

注意

本发行版本中会忽略标签。它们被保留给以后使用。

3. 要编辑扩展依赖项，请在编辑器中打开 'pom.xml 'file。如果添加依赖项，则必须定义其范围。

完成自定义步骤编写代码后，构建 .jar 文件，如 第 9.5 节 “构建 Fuse Online 扩展 JAR 文件” 所述。

9.5. 构建 FUSE ONLINE 扩展 JAR 文件

为扩展名构建 .jar 文件：

1. 在 Project Explorer 视图中，右键点击该项目。
2. 在上下文菜单中，选择 Run As → Maven clean verify。
3. 在 Console 视图中，您可以监控构建的进度。
4. 构建完成后，刷新 Project Explorer 视图中的目标文件夹（选择项目，然后按 F5）。
5. 在 Project Explorer 视图中，打开目标文件夹以查看生成的 .jar 文件：

   .jar 文件的名称遵循 Maven defaults: ${artifactId}-${version}.jar

   例如： custom:step-camel-1.0.0.jar

   此 .jar 文件定义扩展名、其所需的依赖项及其元数据：Extension Id、Name、Version、Tags 和 Description。例如：

```
{
  "schemaVersion" : "v1",
```

```
    "name" : "Example Fuse Online Extension",
    "description" : "Logs a message body with a prefix",
    "extensionId" : "fuse.online.extension.example",
    "version" : "1.0.0",
    "actions" : [ {
      "id" : "Log-body",
      "name" : "Log Body",
      "description" : "A simple xml Body Log with a prefix",
      "descriptor" : {
        "kind" : "ENDPOINT",
        "entrypoint" : "direct:log-xml",
        "resource" : "classpath:META-INF/syndesis/extensions/log-body-action.xml",
        "inputDataShape" : {
          "kind" : "any"
        },
        "outputDataShape" : {
          "kind" : "any"
        },
        "propertyDefinitionSteps" : [ {
          "description" : "Define your Log message",
          "name" : "Log Body",
          "properties" : {
            "prefix" : {
              "componentProperty" : false,
              "deprecated" : false,
              "description" : "The Log body prefix message",
              "displayName" : "Log Prefix",
              "javaType" : "String",
              "kind" : "parameter",
              "required" : false,
              "secret" : false,
              "type" : "string"
            }
          }
        } ]
      },
      "tags" : [ "xml" ],
      "actionType" : "step"
    } ],
    "dependencies" : [ {
      "type" : "MAVEN",
      "id" : "io.syndesis.extension:extension-api:jar:1.3.0.fuse-000014"
    } ],
    "extensionType" : "Steps"
}
```

9.6. 为 FUSE ONLINE 用户提供 JAR 文件

为 **Fuse Online** 用户提供以下内容：

- **.jar 文件**

- 描述扩展的文档。对于步骤扩展，请在步骤扩展中包含每个操作所需的数据信息，或者作为输出（用于数据映射）提供。

在 Fuse Online 中，用户上传 .jar 文件，如将应用 与 Fuse Online 集成 中所述。

第 10 章 创建新的 CAMEL XML 文件

概述

Apache Camel 将路由存储在包含 camelContext 元素的 XML 文件中。当您创建新的 Fuse 集成项目时，工具默认提供 Apache Camel 上下文(XML)文件作为项目的一部分。

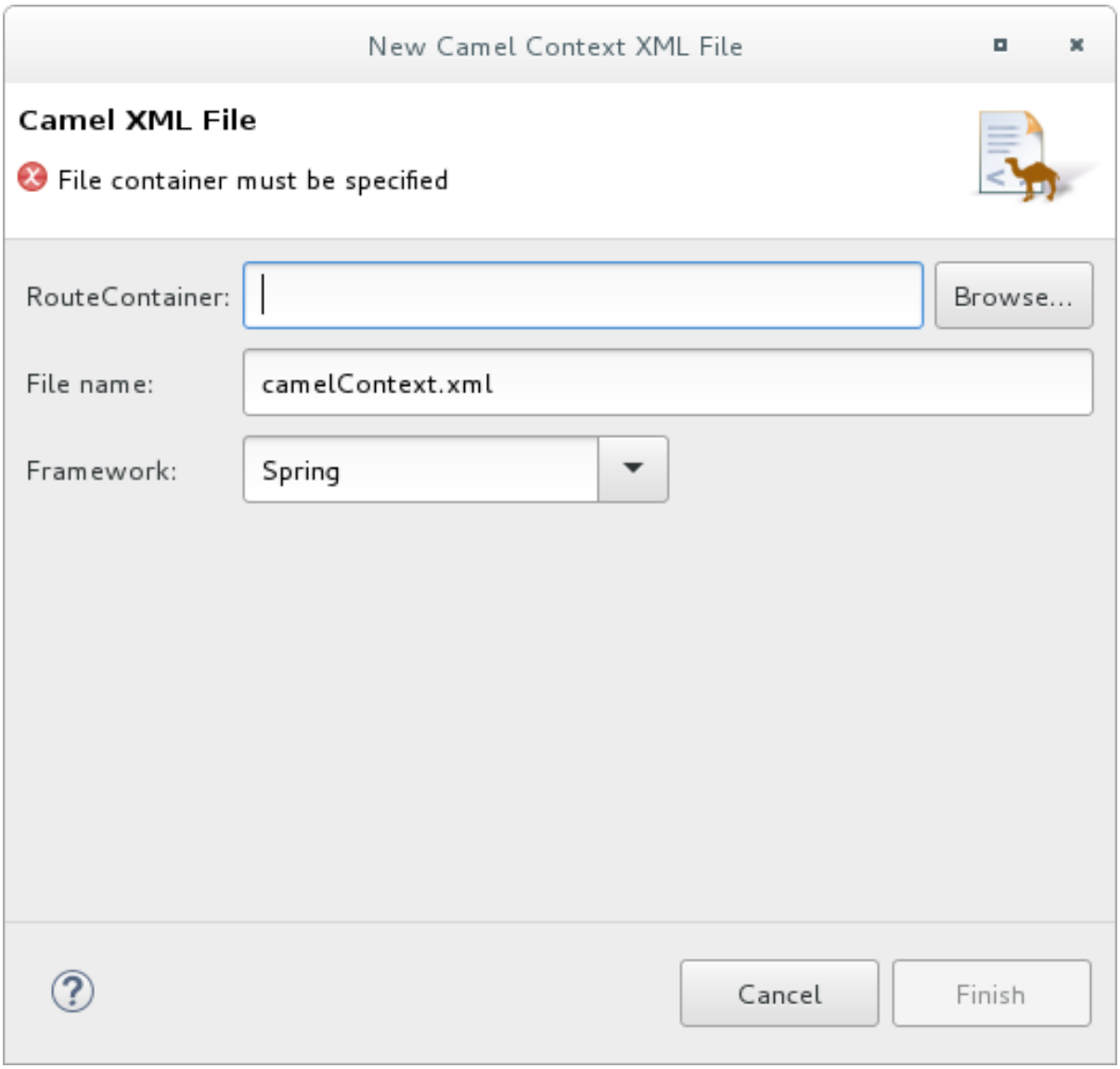
您还可以添加新的 Camel XML 文件，其中包含预配置的所有必需命名空间和模板 camelContext 元素。

流程

在项目中添加新的 Apache Camel 上下文文件：

1. 从主菜单中选择 File → New → Camel XML File 来打开 Camel XML File 向导，如 图 10.1 “Camel XML File 向导” 所示。

图 10.1. Camel XML File 向导

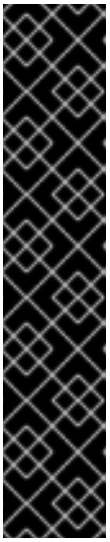


2. **在 RouteContainer 中，输入新文件的位置，或者接受默认值。**

   您可以点

   搜索适当的位置。



重要

Spring 框架和 OSGi 蓝图框架要求将所有 Apache Camel 文件放在项目的 META-INF 或 OSGI-INF 文件夹下的特定位置：

- **spring - projectName/src/main/resources/META-INF/spring/**
- **OSGi Blueprint - projectName/src/main/resources/OSGI-INF/blueprint/**

3. **在 File Name 中，输入新上下文文件的名称或接受默认值(camelContext.xml)。**

   文件名不能包含空格或特殊字符，且在 JVM 中必须是唯一的。

4. **在 Framework 中，接受默认值，或者选择路由要使用的框架：**

   - **在 Spring 容器、非OSGi 容器或独立应用程序中部署的路由的 Spring HEKETI-wagon[default]**
   - **OSGi Blueprint 5-4-wagon 路由，这些路由将在 OSGi 容器中部署**
   - **路由到 您可以加载并添加到现有的 camelContexts 中的路由**

5. **点 Finish。**

   新上下文文件添加到项目中，并在路由编辑器中打开。

第 11 章 更改 CAMEL 版本

当您使用 Fuse 工具项目时，您可能希望更改它使用的 Camel 版本。这很有用，例如，如果要使用较新的 Camel 版本支持的功能，或者使用社区版本。

更改项目使用的 Camel 版本：

1. 在 Project Explorer 中，右键单击您要更改 Camel 版本的项目，然后选择 Configure → Change Camel Version。

2. 在 Change Camel Version 窗口中，在 Camel Version 字段右侧，单击 down caret 以显示可用的 Camel 版本。

   要使用 Apache Camel 的社区版本，请输入其版本号，如 2.19.2。

3. 选择或输入您想要的版本，然后单击 完成。

Fuse 工具检查您选择的版本是否可用，并由 Fuse 工具支持。如果是 Fuse Tooling 更改 Camel 版本，并保存项目更新的 pom.xml 文件。如果您选择的 Camel 版本不可用或不支持，您会收到一条错误消息。

您可以在 < camel.version> 元素的 pom.xml 文件中检查项目的 Camel 版本。

第 12 章 导入现有的 MAVEN 项目

概述

例如，您可能希望导入现有项目，例如，用作模板或开发应用程序的起点。

例如，New Fuse Integration Project 向导指向以下 Github 存储库作为示例源：

- https://github.com/apache/camel/tree/master/examples
- https://github.com/fabric8-quickstarts
- https://github.com/wildfly-extras/wildfly-camel-examples
- https://github.com/jboss-fuse/quickstarts

流程

导入现有的 Maven 项目：

1. 选择 File → Import → Maven → Existing Maven Projects，然后点 Next。
2. 对于 Root Directory，请选择包含下载的示例项目的文件夹。
3. 在项目列表中，检查您要导入的项目，然后单击 Finish。

部分 II. 调试路由上下文

Camel debugger 包括许多用于本地和远程运行路由上下文的功能：

- 在路由编辑器中的节点设置条件和无条件断点
- 自动启动调试器并切换到 Debug 视角
- 与正在运行的路由上下文交互：
  - 在断点间切换，以快速比较消息实例的变量值
  - 检查并更改感兴趣的变量的值
  - 在监视列表中添加感兴趣的变量，以便在 debug 会话中跟踪它们
  - 在运行时禁用和重新启用断点
  - 在路由上下文运行时以图形方式跟踪消息流
  - 检查控制台日志以跟踪 Camel 和调试器操作

注意

在运行 Camel 调试器之前，您必须在路由编辑器的 canvas 上显示的感兴趣的节点上设置断点。然后，您可以在项目的路由上下文 .xml 文件上运行 Camel debugger，以查找其中的逻辑错误并修复它们。调用 Camel debugger 在调试模式下运行路由上下文，并打开 Debug Perspective。

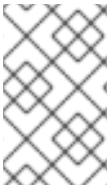
第 13 章 设置 BREAKPOINTS

概述

要设置断点，项目的路由上下文 .xml 文件必须在路由编辑器的 Design 选项卡中打开。

Camel debugger 支持两种类型的断点：

- 当调试会话过程中遇到时，未条件断点中的未条件中断点
- 只有在调试会话过程中满足断点的指定条件时，条件断点才会在发生断点时触发



注意

您不能在消费者端点或何时或其他节点上设置断点。

设置无条件断点

在 Design 选项卡中，您的路由上下文在 canvas 中显示：

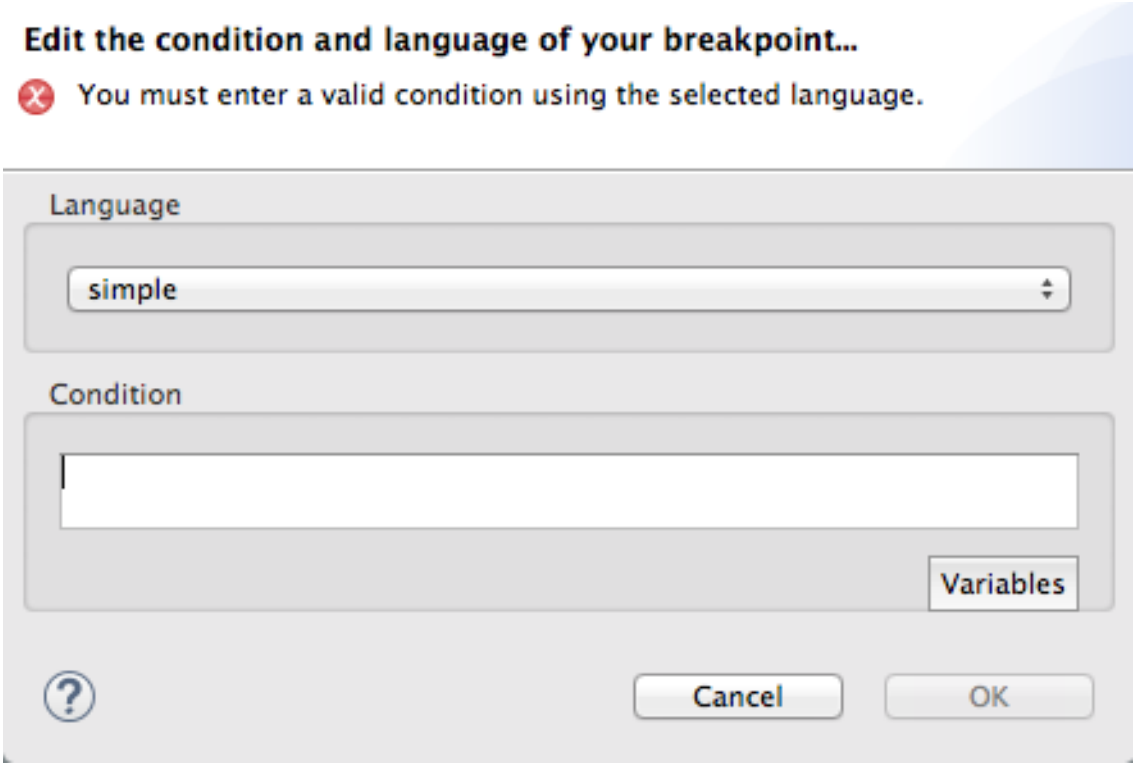
1. 选择您要在调试会话期间检查的状态的节点。
2. 点击其  图标设置无条件断点。
3. 对您要在其上设置无条件断点的每个节点重复这些步骤。

设置条件断点

在 Design 选项卡中，您的路由上下文在 canvas 中显示：

1. 选择一个在调试会话期间要检查的状态的节点。

2. 点击其  图标设置条件断点，并打开 Edit the condition and language of your breakpoint... 对话框：



3. 单击 Language 下拉菜单，再选择用于创建触发断点的条件的表达式语言。

   Fuse 工具支持二十四表达式语言。其中一些语言提供用于创建条件表达式的变量，而其他语言则不提供。

4. Click Variables 以显示所选语言支持变量的列表。

   如果显示列表，请按顺序选择一个或多个变量来创建触发断点的条件。您选择的变量会出现在 Condition 文本框中。

   如果显示 

   ，在 Condition 文本框中直接输入表达式。

5. 对您要设置条件断点的每个节点重复 [condBpLast] 到 [condBpLast] 的步骤。

禁用断点

您可以临时禁用断点，保留它，然后稍后启用它。

按钮在调试会话过程中跳过禁用的断点。

要禁用断点，请选择 canvas 上的节点并点其

图标。断点会灰显，表示它已被禁用。

要启用禁用的断点，请选择 canvas 中的节点并点其

图标。根据禁用的断点是条件还是无条件，它会分别打开 yellow 或 red 来指示它已被重新启用。

注意

您还可以在调试会话过程中禁用和重新启用断点。详情请查看 第 18 章 *在运行的上下文中禁用 Breakpoints*。

删除断点

您可以删除单个断点或所有断点。

- 要删除单个断点和路由容器，请选择您要删除的断点的节点，然后点其

  图标。

- 要删除特定路由的所有断点中的所有断点，请右键单击目标路由的容器，然后选择 Delete all breakpoints。

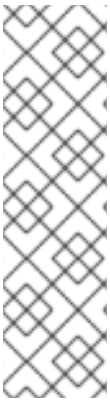
- 要删除所有路由都 breakpoints 中的所有断点，请单击 canvas，然后选择 Delete all breakpoints。

相关主题

- 第 14 章 *运行 Camel Debugger*

第 14 章 运行 CAMEL DEBUGGER

您可以在本地运行的路由上下文上运行 Camel debugger。



注意

- 不再支持远程调试。对于远程调试，您必须配置 Jolokia，并通过 Jolokia 创建特定的 JMX 连接。
- 如果您的项目包含 Java 代码，您可以使用标准 Eclipse Java 调试工具进行调试。

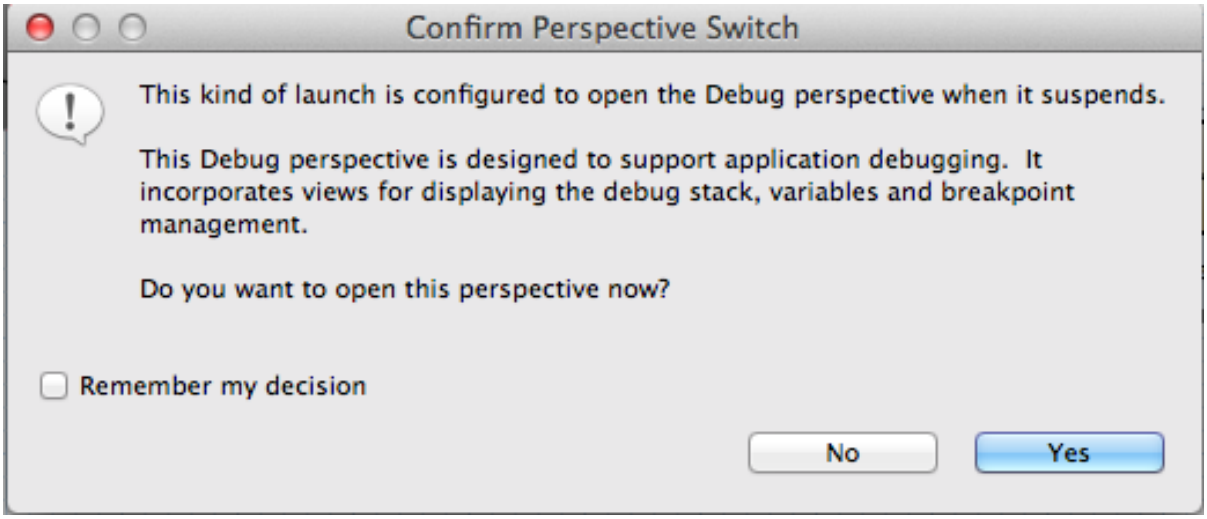
您必须在路由上下文文件中设置断点，然后才能启动 Camel 调试器。

流程

1. 在 Project Explorer 视图中，选择您要调试的路由上下文文件。
2. 右键单击所选文件以打开上下文菜单，然后选择 Debug As → Local Camel Context。

Fuse 工具构建 Camel 路由，启动 Apache Camel，启动路由上下文，启用 JMX，在路由上下文中启动路由，向节点添加断点，并启用 Camel 调试器。

Camel debugger 会在第一个断点命中时挂起执行路由上下文（收到消息），并提示您指明是否要打开 Debug 视角。

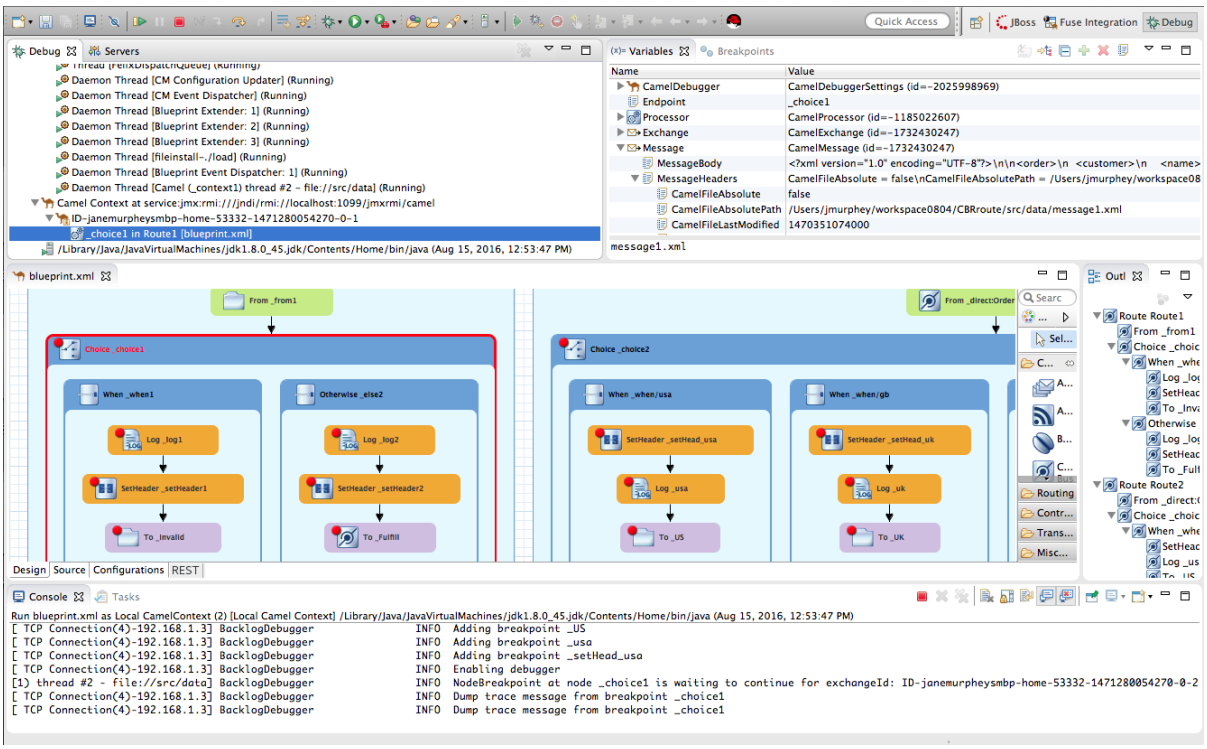


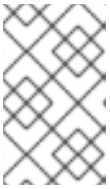
3. 单击 **Yes** 以打开 **Debug** 视角。

Debug 视角会打开，并在运行的路由上下文中遇到的第一个断点暂停路由上下文。

重要

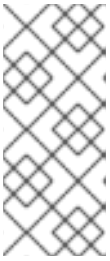
断点最多保留五分钟，之后调试会自动恢复，进入下一个断点或路由上下文的末尾。





注意

要查看控制台输出，请在切换视角时打开 Console 视图。



注意

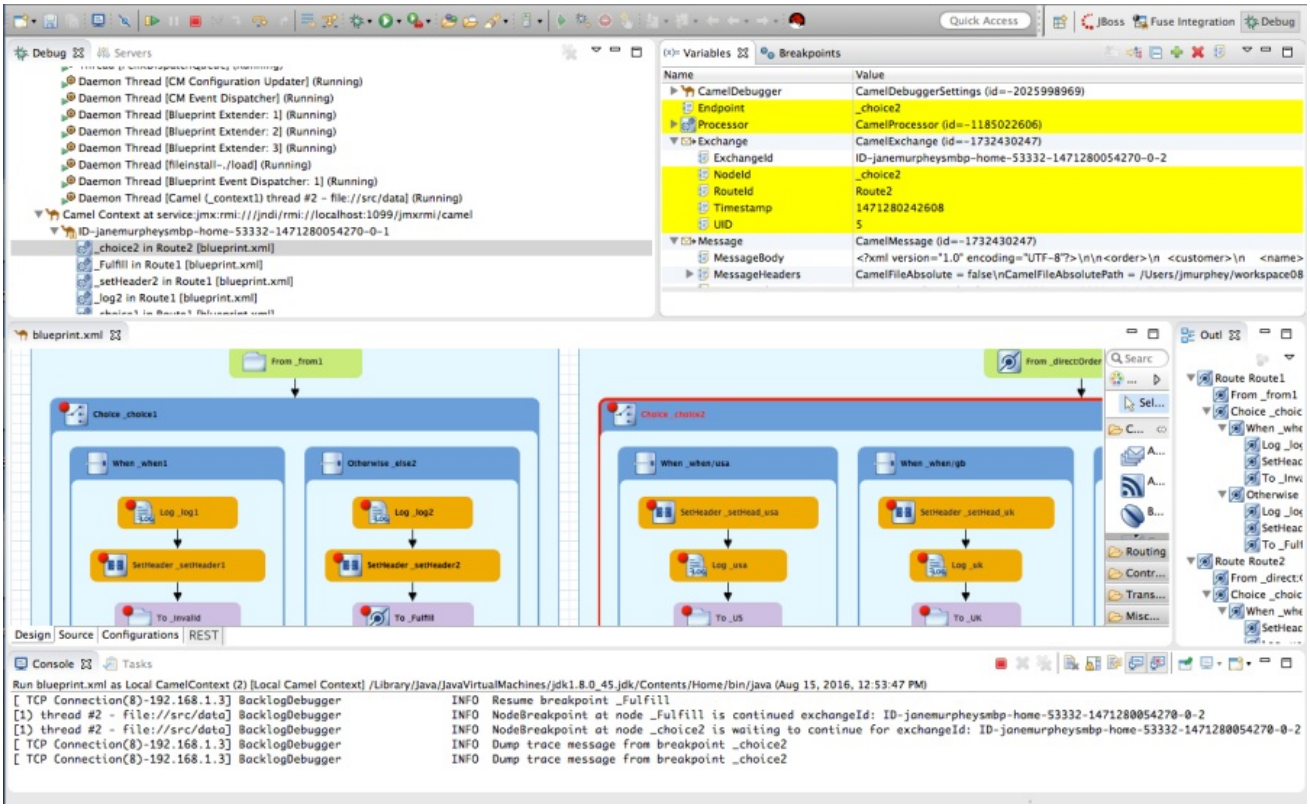
默认情况下，Debug 视角显示 foundation 视图，它提供在运行的路由上下文中在单独的路由间切换的方法。如果您的路由上下文包含单个路由，请关闭 outline 视图释放空间以扩展其他视图，从而更轻松地访问和检查调试器输出。

通过路由上下文监视消息交换进度

点

(Step Over)在路由上下文中跳到执行下一个节点。点

（恢复）在路由上下文中的下一个活跃断点继续执行。



相关主题

- *第 15 章 停止 Camel Debugger*

第 15 章 停止 CAMEL DEBUGGER

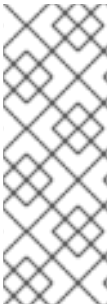
概述

要停止 Camel 调试器，请在调试会话终止后点菜单栏中的

。否则，点

两次：一次终止当前运行的节点线程，一次终止 Camel Context thread （在 Debug 视图中显示）。

+



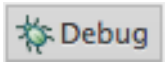
注意

终止 Camel 调试器也会终止控制台，但不会清除其输出。要清除输出，请点击 Console 视图 菜单栏中的

(Clear Console)。

关闭 CAMEL DEBUGGER

在完成项目调试后，您可能需要关闭 Debug 视角，为您的工作台提供更多空间。

要做到这一点，在工具栏中右键单击


，然后选择 Close。

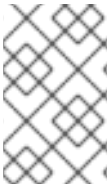
相关主题

- 第 14 章 *运行 Camel Debugger*

第 16 章 更改变量值

概述

当 Camel 调试器到达断点时，Variables 视图将显示路由上下文中在该点上可用的所有变量的值。有些变量可编辑，允许您更改其值。这可让您了解应用程序如何处理程序状态的更改。



注意

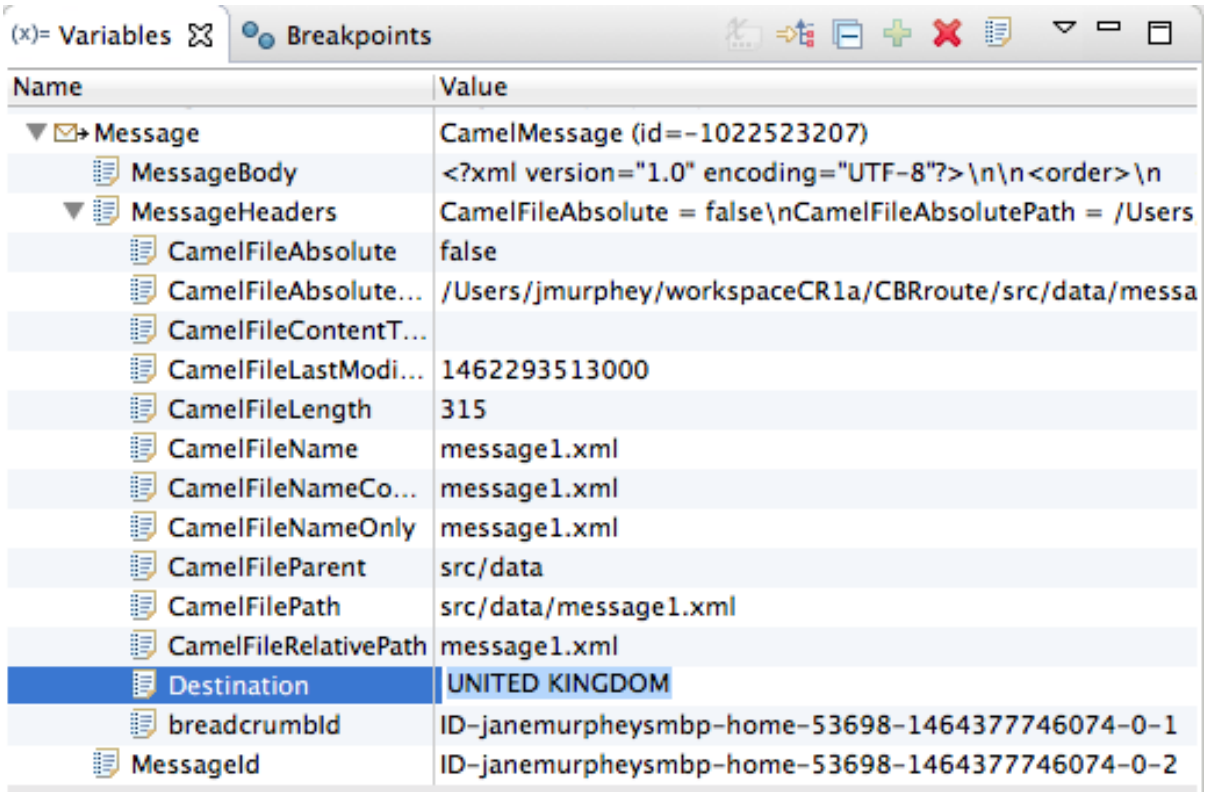
并非所有变量都可编辑。可编辑的变量的上下文菜单显示 Change Value... 选项。

流程

更改变量的值：

1. 如有必要，启动 debugger。请参阅 第 14 章 *运行 Camel Debugger*。

2. 在 Variables 视图中，选择一个您要更改的值的变量，然后点其 Value 字段。



变量的 value 字段会更轻的蓝色点，表示它处于编辑模式。

注意

或者，您可以右键点击变量以打开其上下文菜单，然后选择 Change Value... 来编辑其值。

3. 输入新值，然后点 Enter。

Console 视图显示一个 INFO 级别日志条目，它 指出变量的值的变化（例如，节点 to1 的明细点正在更新 ExchangeId: ID-dhcp-97-16-bos-redhat-com-52574-1417298894070-0-2 with header: Destination and value: UNITED KINGDOM）。

4. 继续逐步浏览断点，检查消息是否如预期处理。在每个步骤中，检查 Debug 视图、Variables 视图和 Console 输出。

相关主题

- *第 18 章 在运行的上下文中禁用 Breakpoints*
- *第 17 章 在 Watch 列表中添加变量*

第 17 章 在 WATCH 列表中添加变量

概述

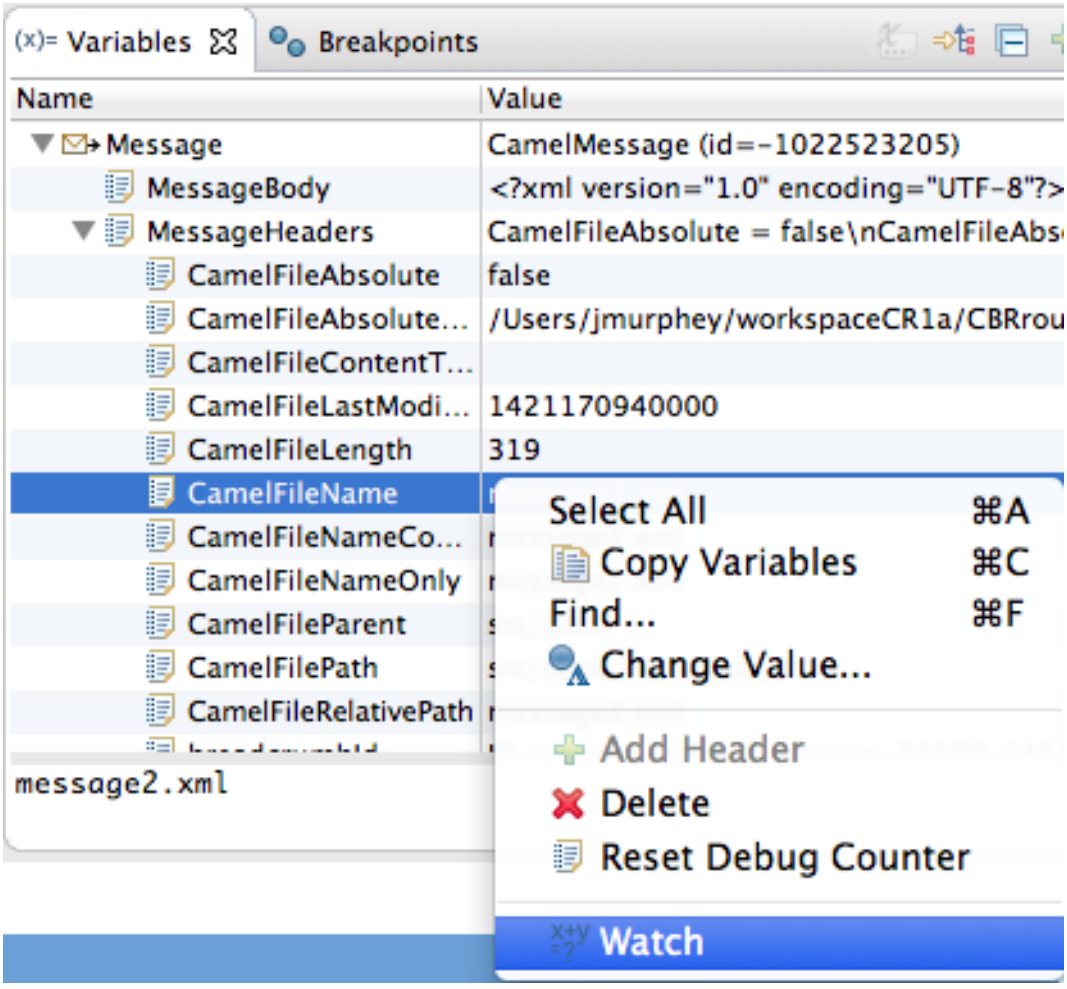
通过添加变量到监视列表中，您可以专注于特定的变量来查看它们的值是否如通过路由上下文所要更改。

流程

在 watch 列表中添加一个变量：

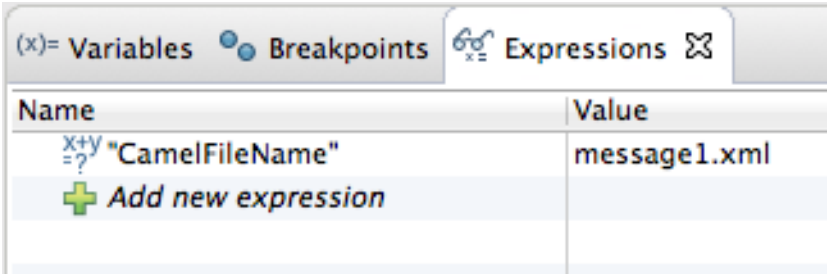
1. 如有必要，启动 debugger。请参阅 第 14 章 *运行 Camel Debugger*。

2. 在 Variables 视图中，右键单击您要跟踪的变量以打开上下文菜单。



3. 选择 Watch。

在 Breakpoints 视图旁打开一个新的视图 Expressions。Expressions 视图显示正在监视的变量名称及其当前值，例如：



4. 重复 [watch1] 和 [watch2] 在 watch 列表中添加额外的变量。

注意

您添加的变量保留在 watch 列表中，直到您删除它们。要停止监视变量，请在列表中右键单击以打开上下文菜单，然后单击 删除。

5. 在 Expressions 视图打开后，逐步浏览路由上下文，以跟踪监视列表中每个变量的值在路由中的每个步骤如何改变。

相关主题

- 第 16 章 *更改变量值*

第 18 章 在运行的上下文中禁用 BREAKPOINTS

概述

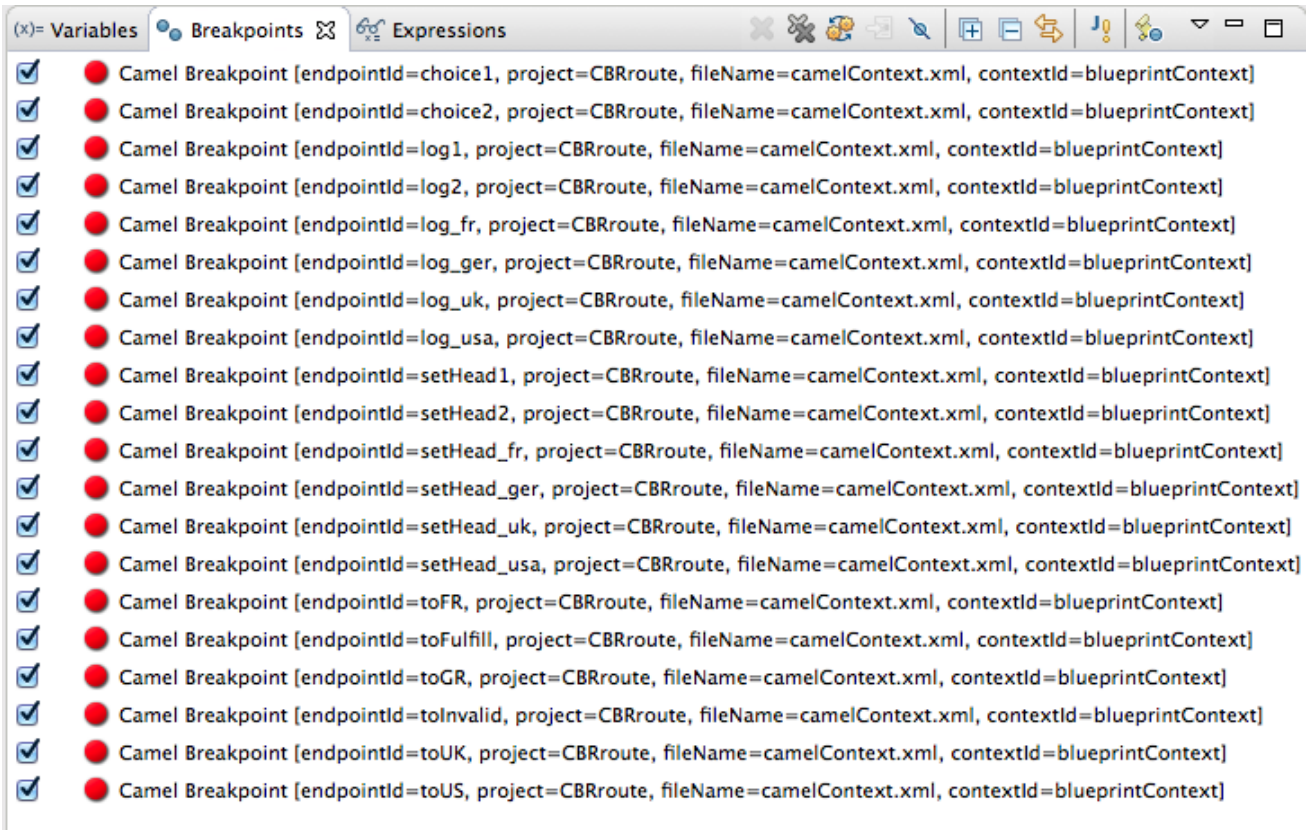
您可以在 Breakpoints 视图中运行的路由上下文中禁用和重新启用断点。

当禁用断点时，

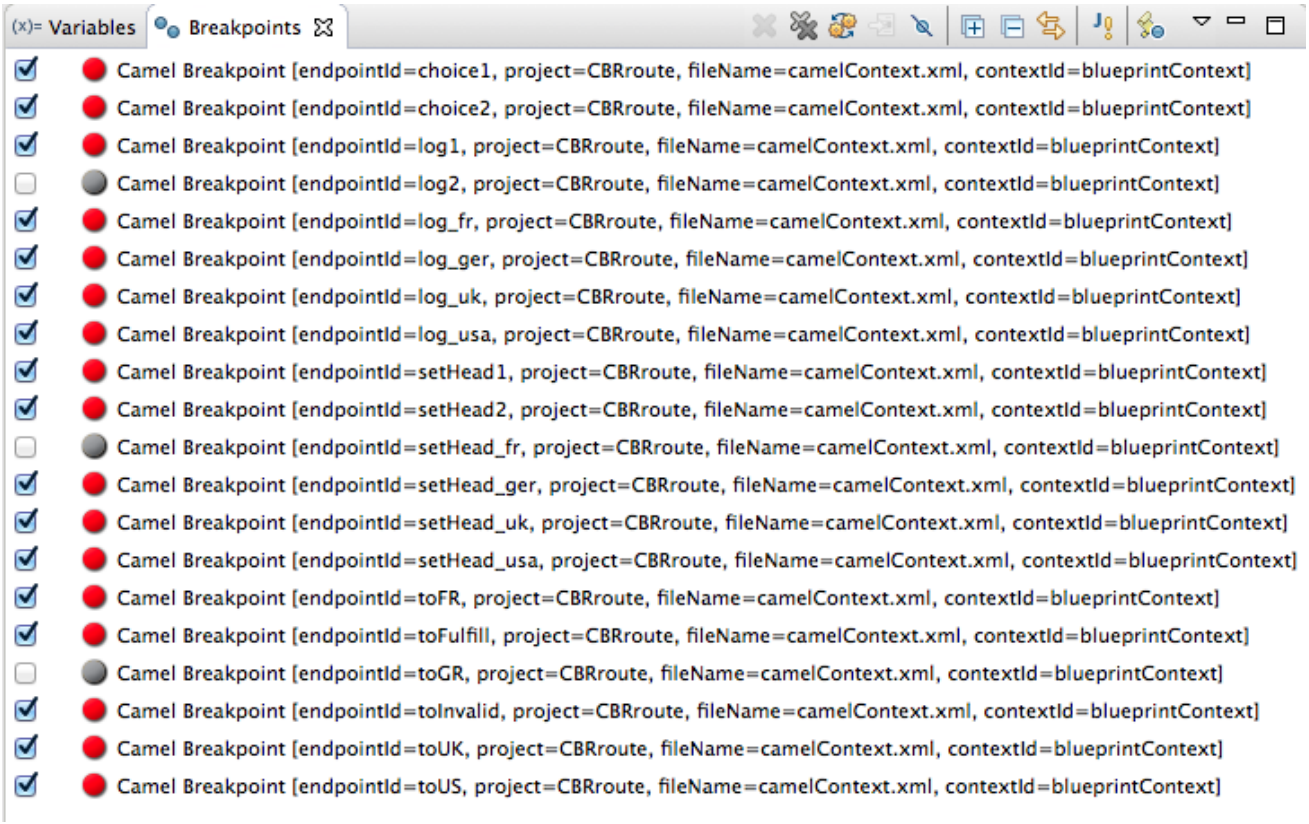
按钮会导致调试器在调试会话中跳过它。

在 BREAKPOINTS 视图中禁用并启用断点

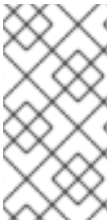
Breakpoints 视图会打开，并启用了所有设置断点。



要禁用断点，请清除其复选框。



对于禁用的每个断点，Console 视图会显示一个 INFO 级别日志条目，它已被禁用（例如，删除断点 log2）。同样，对于您重新启用的每个断点，Console 视图会显示一个 INFO 级别日志条目，指出它已经启用（例如，添加断点 log2）。



注意

要重新启用禁用的断点，请点其复选框。Console 视图显示一个 INFO 级别日志条目，表示将断点添加到所选节点。

部分 III. 监控和测试应用程序

JMX Navigator 视图提供了大量监控和测试 Fuse 应用程序的方法。

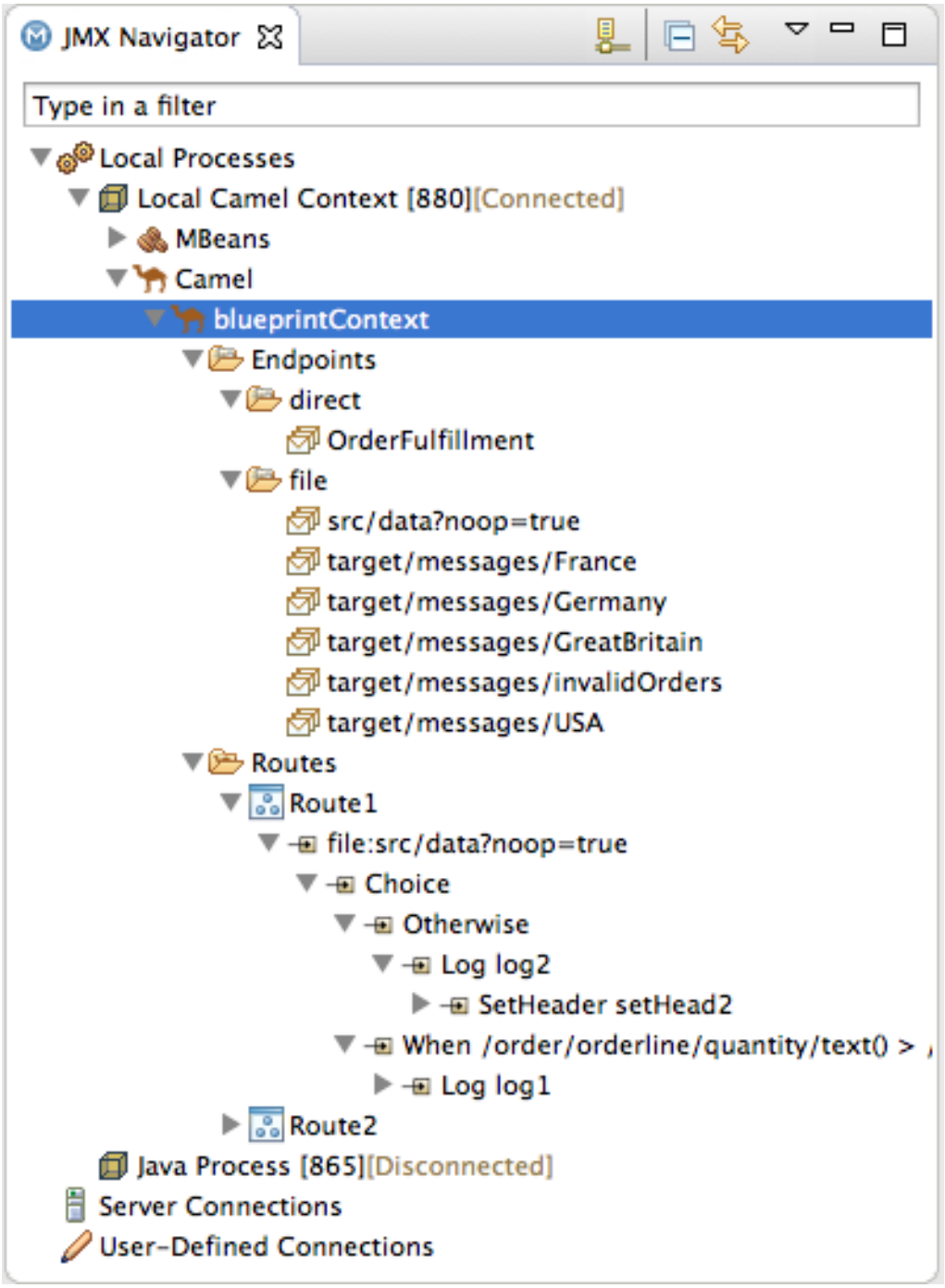
注意

您还可以使用 Fuse 控制台监控 Fuse 应用程序，如 管理 Fuse 所述。

第 19 章 JMX NAVIGATOR

JMX Navigator 视图（如 图 19.1 “JMX Navigator 视图” 所示）显示应用程序中运行的所有进程，并驱动与监控和测试功能的所有交互。Fuse Integration 透视图的其他区域适应以显示与 JMX Navigator 视图中选择的节点相关的信息。在 JMX Navigator 视图中，其上下文菜单提供了激活路由追踪和添加 JMS 目的地所需的命令。

图 19.1. JMX Navigator 视图



默认情况下，JMX Navigator 视图发现在本地计算机上运行的所有 JMX 服务器，并在以下类别下列出它们：

- 本地进程
- 服务器连接
- 用户定义的连接

注意

您可以使用服务器的 JMX URL 添加其他 JMX 服务器。详情请查看 第 19.2 节 “添加 JMX 服务器”。

19.1. 查看 JMX 中的进程

概述

JMX Navigator 视图列出了一系列树中的所有已知进程。每个树的根都是 JMX 服务器。

列表中的第一个树是一个特殊的本地进程树，其中包含本地计算机上运行的所有 JMX 服务器。您必须连接到 JMX 服务器中的一个，才能查看其包含的进程。

查看本地 JMX 服务器中的进程

查看本地 JMX 服务器中进程的信息：

1. 在 JMX Navigator 视图中，展开 Local process。
2. 在本地进程 下，双击其中一个顶级条目来连接它。
3. 点条目旁边的

图标显示其在 JVM 中运行的组件列表。

查看备用 JMX 服务器中的进程

查看备用 JMX 服务器中进程的信息：

1. 第 19.2 节 “添加 JMX 服务器” JMX 服务器到 JMX 导航器 视图。

2. 在 JMX Navigator 视图中，使用条目旁边出现的 ▶ 图标扩展服务器条目。这将显示 JVM 中运行的 JMX 服务器组件的列表。

19.2. 添加 JMX 服务器

概述

在 JMX Navigator 视图中，在树的 Local process 分支下，您可以看到所有本地 JMX 服务器的列表。您可能需要连接到特定的 JMX 服务器，以查看其他计算机上部署的组件。

要添加 JMX 服务器，您必须知道要添加的服务器的 JMX URL。

流程

将 JMX 服务器添加到 JMX Navigator 视图中：

1. 在 JMX Navigator 视图中，点 New Connection  。

2. 在 Create a new JMX connection 向导中，选择 Default JMX Connection。

3. 点击 Next。

4. 选择 Advanced 选项卡。

5. 在 Name 字段中输入 JMX 服务器的名称。

名称可以是任意字符串。它用于标记 JMX Navigator 树中的条目。

6. 在 JMX URL 字段中，输入服务器的 JMX URL。

7. 如果 JMX 服务器需要身份验证，请在 Username 和 Password 字段中输入您的用户名和密码。

8. 点 Finish。

新的 JMX 服务器在 User-Defined Connections 树中作为分支显示。

第 20 章 查看组件的 JMX 统计信息

概述

工具收集 Fuse 组件报告的所有 JMX 统计信息，并将它们显示在 Properties 视图中。这种统计信息可以深入了解集成应用程序中的情况。

JMX 统计数据分为三个类别： 属性 、 处理器 和配置文件。

流程

查看 Fuse 组件的统计信息：

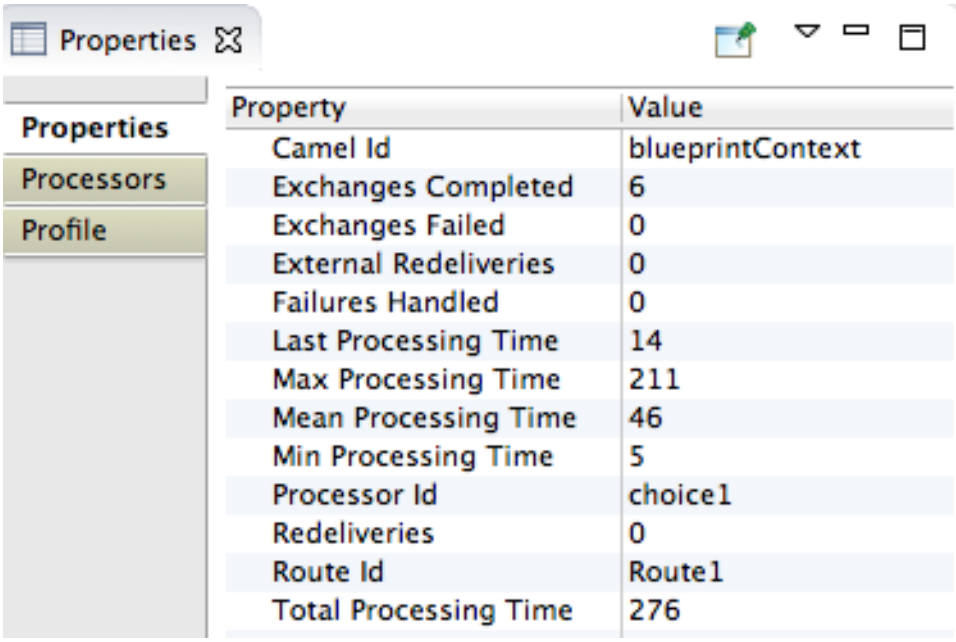
1. 在 JMX Navigator 视图中，找到组件的节点。

   您可能需要扩展树上的节点，以查找低级组件。

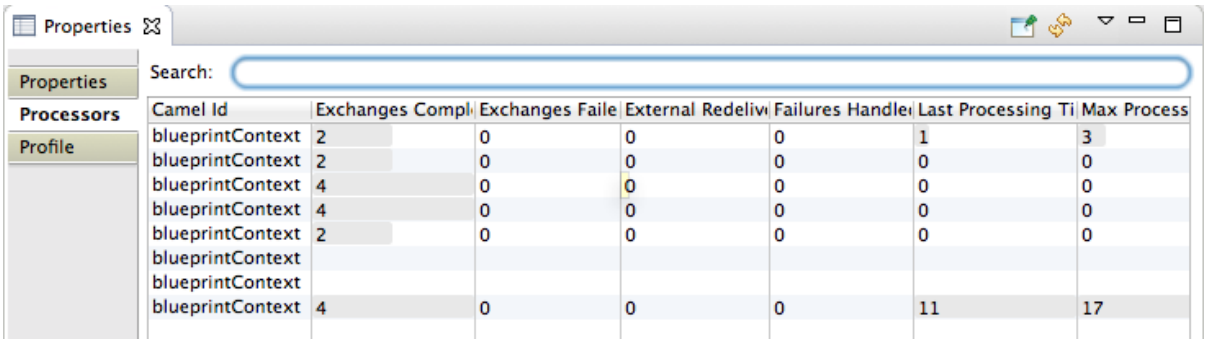
2. 选择您要查看其统计信息的 Fuse 组件的节点。

3. 打开 Properties 视图。

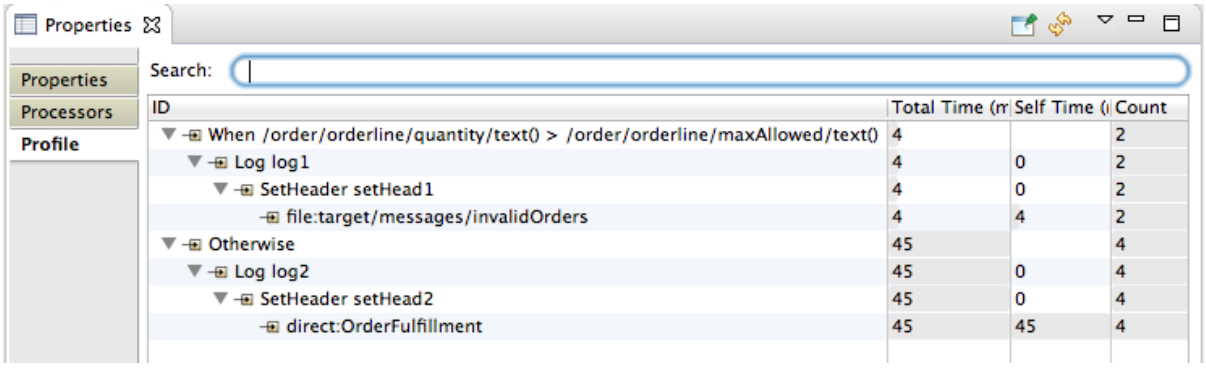
4. Properties 页面显示所选组件的 JMX 属性：



5. **点 Processors 检查所选组件的交换指标：**



6. **点 Profile 检查所选节点及其子节点的消息指标：**



第 21 章 浏览消息

概述

分布式环境中调试应用的关键工具是看到应用中存储在 JMS 目的地和路由端点的所有消息。该工具可浏览以下内容：

- JMS 目的地
- JMS 路由端点
- Apache Camel 路由端点
- SEDA 路由端点
- 浏览路由端点
- 模拟路由端点
- VM 路由端点
- Dataset 路由端点

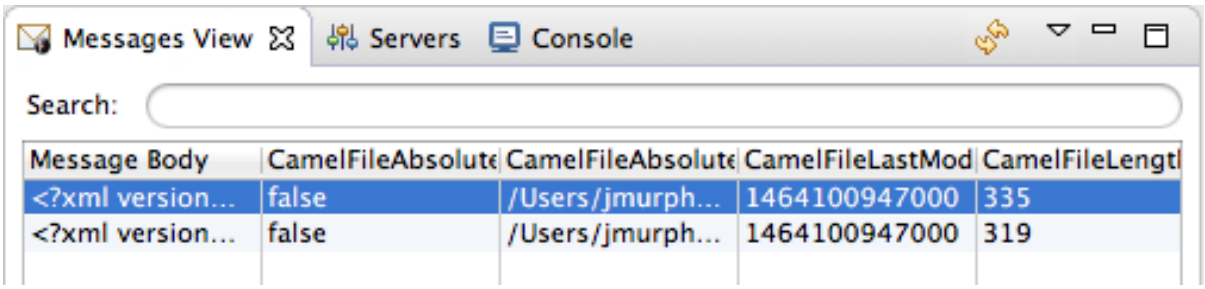
流程

浏览信息：

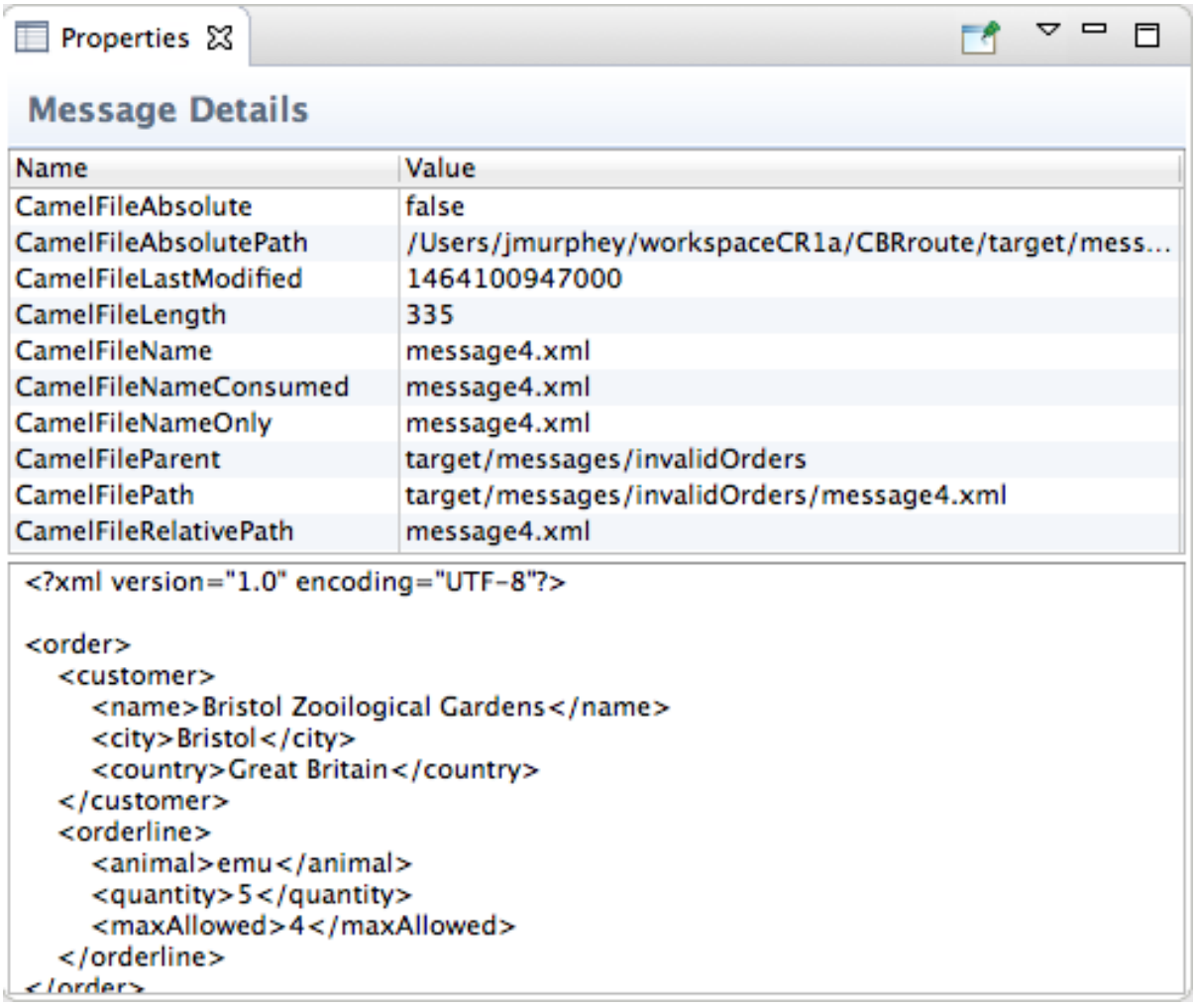
1. 在 JMX Navigator 视图中，选择您要浏览的 JMS 目的地或端点。

   消息列表会出现在 Messages View 中。

2. **在 Messages View 中，选择要检查的单个消息。**



Properties 视图中会显示消息详情和内容：



相关主题

- 第 22.3 节 “通过路由上下文追踪消息”

第 22 章 追踪路由

调试路由通常涉及解决以下两个问题之一：

- 消息被错误地转换。
- 消息无法到达其目标端点。

通过路由跟踪一个或多个测试消息是发现此类问题源的最简单方法。

工具的路由追踪功能可让您监控消息通过路由的路径，并查看消息如何从处理器传递到处理器的方式。

Diagram View 显示路由的图形表示，它可让您看到消息通过它的路径。对于路由中的每个处理器，它还显示自路由启动以来所有消息的平均处理时间（以毫秒为单位），以及自路由启动后处理的消息数量（以毫秒为单位）。

Messages View 显示 JMX Navigator 树中选择的 JMS 目标或路由端点处理的消息。在 Messages View 中选择单个消息跟踪会在 Properties 视图中显示消息的完整详情和内容，并突出显示 Diagram View 中的 correspoding 节点。

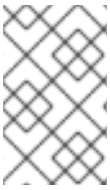
通过路由追踪消息涉及以下步骤：

1. 第 22.1 节 “为路由追踪创建测试消息”
2. 第 22.2 节 “激活路由追踪”
3. 第 22.3 节 “通过路由上下文追踪消息”
4. 第 22.4 节 “停用路由追踪”

22.1. 为路由追踪创建测试消息

概述

路由追踪可用于任何类型的消息结构。Fuse Message 向导会创建一个空的 .xml 消息，使消息的结构完全为您保留。



注意

如果要存储测试消息的文件夹不存在，您需要在创建消息前创建它。

创建新文件夹以存储测试信息

要创建新文件夹，请执行以下操作：

1. 在 Project Explorer 视图中，右键单击项目 root 以打开上下文菜单。
2. 选择 New → Folder 以打开 New Folder 向导。

   项目 root 会出现在 Enter 或选择父文件夹 字段中。
3. 以项目层次结构的图形表示中扩展节点，然后选择您要成为父文件夹的节点。
4. 在 Folder name 字段中，输入新文件夹的名称。
5. 点 Finish。

   新文件夹显示在所选父文件夹下的 Project Explorer 视图中。

注意

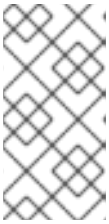
如果没有出现新文件夹，请右键单击父项 foler 并选择 Refresh。

创建测试信息

创建测试信息：

1. 在 Project Explorer 视图中，右键单击项目以打开上下文菜单。
2. 选择 New → Fuse Message 以打开 New File 向导。
3. 以项目层次结构的图形表示形式扩展节点，然后选择要存储新测试消息的文件夹。
4. 在 File name 字段中输入消息的名称，或接受默认值(message.xml)。
5. 点 Finish。

   新消息将在 XML 编辑器中打开。
6. 输入消息内容，包括正文和标题文本。



注意

您可能会看到 文档 中引用的警告 No grammar 约束(DTD 或 XML Schema)， 具体取决于您输入的标头文本。您可以安全地忽略这个警告。

相关主题

- 第 22.3 节 “通过路由上下文追踪消息”

22.2. 激活路由追踪

概述

您必须激活路由上下文的路由追踪，然后才能通过该路由上下文跟踪消息。

流程

要在路由上下文中激活追踪：

1. 在 JMX Navigator 视图中，选择要开始追踪的正在运行的路由上下文。

注意

您可以选择上下文中的任何路由开始追踪整个上下文。

2. 右键单击所选路由上下文以打开上下文菜单，然后选择 Start Tracing 以启动 trace。

如果在上下文菜单中启用了 Stop Tracing Context，则追踪已经活跃。

相关主题

- 第 22.3 节 “通过路由上下文追踪消息”
- 第 22.4 节 “停用路由追踪”

22.3. 通过路由上下文追踪消息

概述

查看路由上下文中发生的情况的最佳方法是监控每个停止信息发生的情况。该工具提供了一种将消息丢弃到正在运行的路由上下文中的机制，并追踪消息通过它的路径。

流程

通过路由上下文跟踪信息：

1. 创建一个或多个测试信息，如 第 22.1 节 “为路由追踪创建测试消息” 所述。

2. 在 Project Explorer 视图中，右键单击项目的 Camel 上下文文件以打开上下文菜单，然后选择 Run As → Local Camel Context (without Tests)。

注意

不要将它作为 Local Camel Context 运行，除非您为项目创建了全面的 JUnit 测试。

3. 激活正在运行的路由上下文的追踪，如 第 22.2 节 “激活路由追踪” 所述。

4. 将一个测试消息从 Project Explorer 视图拖到 JMX Navigator 视图中的路由上下文的起点。

5. 在 JMX Navigator 视图中，选择要跟踪的路由上下文。

   该工具填充 Messages View，其消息实例代表 traced 上下文的每个阶段的消息实例。

   图表视图 显示所选路由上下文的图形表示。

6. 在 Messages View 中，选择其中一个消息实例。

   Properties 视图显示消息实例的详情和内容。

   在 Diagram View 中，突出显示与所选消息实例对应的路由步骤。如果路由步骤是处理步骤，工具会使用计时和处理指标标记退出路径。

7. 根据需要重复此生产。

相关主题

- 第 22.1 节 “为路由追踪创建测试消息”
- 第 22.2 节 “激活路由追踪”

- 第 22.4 节 “停用路由追踪”

22.4. 停用路由追踪

概述

在路由上下文中调试路由后，您应该停用追踪。

重要

停用追踪会停止追踪并清空路由上下文中所有路由的 trace 数据。这意味着您无法查看任何过去的追踪会话。

流程

停止路由上下文的追踪：

1. 在 JMX Navigator 视图中，选择要取消激活追踪的正在运行的路由上下文。

注意

您可以选择上下文中的任何路由来停止追踪上下文。

2. 右键单击所选路由上下文以打开上下文菜单，然后选择 Stop Tracing Context。

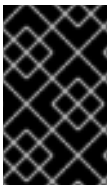
   如果 Start Tracing 出现在上下文菜单中，则不会为路由上下文激活追踪。

相关主题

- 第 22.2 节 “激活路由追踪”
- 第 22.3 节 “通过路由上下文追踪消息”

第 23 章 管理 JMS 目的地

JMX Navigator 视图可让您在运行的 Red Hat Fuse 实例中添加或删除 JMS 目的地。



重要

这些更改在代理重启后不会保留。

23.1. 添加 JMS 目的地

概述

在测试新方案时，可以方便地向其中一个代理添加新的 JMS 目的地。

流程

将 JMS 目的地添加到代理：

1. 在 JMX Navigator 视图中，在您要添加目的地的代理节点下，选择 Queues 子或 Topics 子节点。
2. 右键单击所选节点以打开上下文菜单，然后选择 Create Queue 或 Create Topic。
3. 在 Create Queue 或 Create Topic 对话框中，为新目的地输入一个名称。
4. 点击 确定。
5. 右键单击 Queues 或 Topics 子级，然后选择 Refresh。

   新目的地会出现在 Queues 子或主题子下的 JMX Navigator 视图中。

相关主题

- 第 23.2 节 “删除 JMS 目的地”

23.2. 删除 JMS 目的地

概述

在测试故障转移方案或其他涉及处理故障的场景时，可以轻松地删除 JMS 目的地会很有帮助。

流程

删除 JMS 目的地：

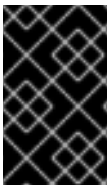
1. 在 JMX Navigator 视图中，在 Queues 子或主题下，选择您要删除的 JMS 目的地。
2. 右键单击所选目的地以打开上下文菜单，然后选择 Delete Queue/Topic。

相关主题

- 第 23.1 节 “添加 JMS 目的地”

第 24 章 管理路由端点

JMX Navigator 视图可让您添加或删除路由端点。



重要

这些更改在路由上下文重启后不会保留。

24.1. 添加路由端点

概述

在测试新场景时，您可能希望添加新端点到路由上下文中。

流程

将端点添加到路由上下文：

1. 在 JMX Navigator 视图中，在路由上下文节点下，选择要向其添加端点的 Endpoints 子级。
2. 右键单击所选节点以打开上下文菜单，然后选择 Create Endpoint。
3. 在 Create Endpoint 对话框中，输入定义新端点的 URL，例如 file://target/messages/validOrders。
4. 点击 确定。
5. 右键单击路由上下文节点，然后选择 Refresh。

新目的地会出现在 Endpoints 节点的 JMX Navigator 视图中，位于与端点类型对应的文件夹中，例如 file。

相关主题

- 第 24.2 节 “删除路由端点”

24.2. 删除路由端点

概述

测试故障转移方案或其他涉及处理故障的场景时，可以从路由上下文中删除端点会很有帮助。

流程

删除路由端点：

1. 在 JMX Navigator 视图中，选择您要删除的端点。
2. 右键单击所选端点以打开上下文菜单，然后选择 Delete Endpoint。

   工具删除端点。
3. 要从视图中删除已删除的端点，请右键单击 Endpoints 节点，再选择 Refresh。

   端点会从 JMX Navigator 视图中消失。

注意

要从 Project Explorer 视图中删除端点的节点，而无需重新运行项目，您需要通过右键单击节点并选择 Delete 来显式删除它。若要将其从视图中移除，可刷新项目显示。

相关主题

- 第 24.1 节 “添加路由端点”

第 25 章 编辑运行的路由

概述

您可以试验对正在运行的路由的更改，而无需更改项目的路由上下文。

要做到这一点：

- 在 JMX Navigator 视图中，在运行的路由上下文中启用 Edit Routes 选项。

  这会打开一个内存模型，它对 route 编辑器进行一个 in-memory model of the Remote CamelContext:<camelContextId > HEKETI-wagonin。

- 在路由编辑器中，对路由上下文的内存模型进行更改。同时，您可以在相关节点上设置断点，以使用 Camel 调试器及其所有功能。

  您可以编辑内存模型以添加、删除或重新排列节点；添加或删除现有节点的属性；以及修改现有节点上设置的属性值。您必须保存对内存模型所做的更改，以更新运行的上下文，并在设置了断点时查看 Debug 视角。

- 在 JMX Navigator 视图中，在运行的路由上下文中丢弃消息，或者等待消息从计时器、ActiveMQ、文件或其他持续输入节点到达。

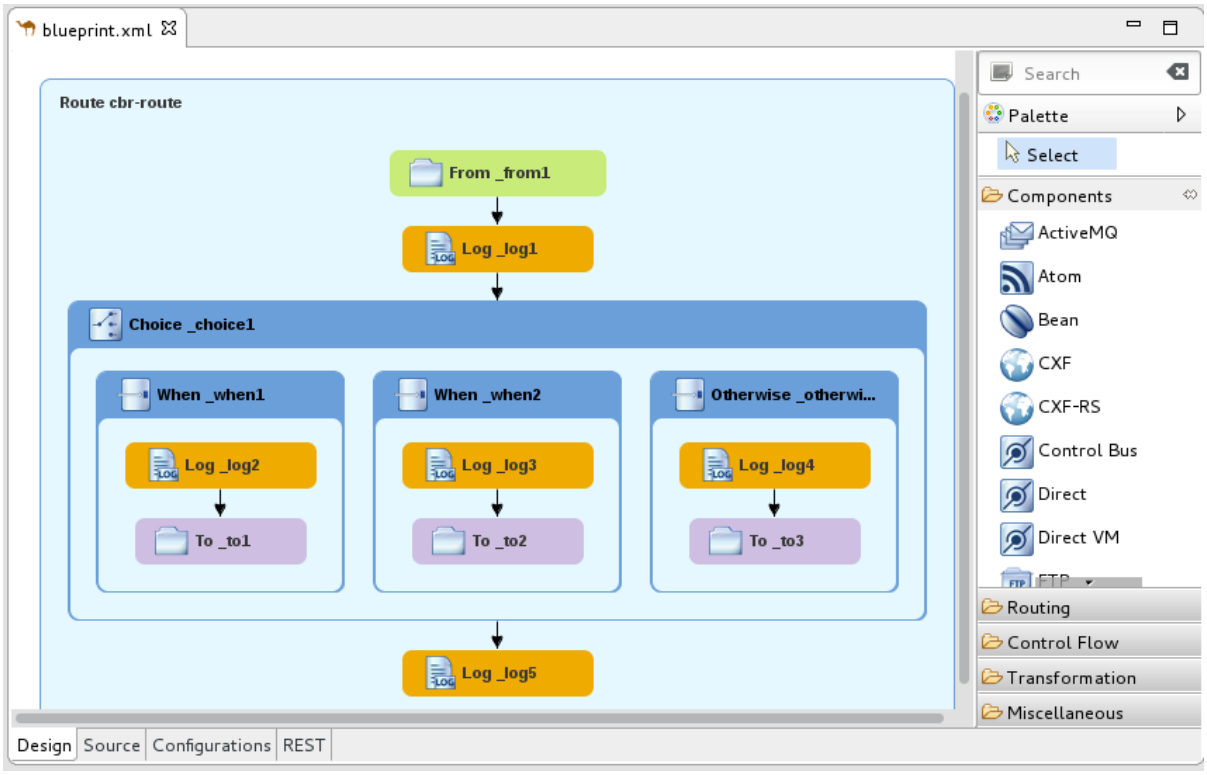
- 在 Debug 透视图中，评估结果并使用 Camel 调试器深入了解路由上下文。

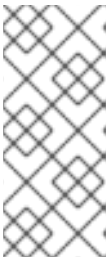
修改正在运行的路由并评估结果

1. 在 JMX Navigator 视图中，选择包含您要编辑的路由上下文。

2. 右键单击所选路由上下文以打开上下文菜单，然后选择 Edit Routes。

   路由编辑器会打开路由上下文的内存中模型 Remote CamelContext:<contextId& gt;，并显示了上下文中的所有路由，例如：





注意

<contextId > 是项目路由上下文 .xml 文件中的 camelContext 元素的 ID。在本例中，基于 Fuse → Content Based Router 内置模板，ID 是 cbr-example-context。

3. **按照 第 2 章 *在路由编辑器中编辑路由上下文* 所述编辑路由，然后选择 File → Save 以保存您对内存模型所做的更改，并更新正在运行的路由上下文。**

4. **在相关节点上设置断点，如 第 13 章 *设置 Breakpoints* 所述。**

5. **在 JMX Navigator 视图中，丢弃正在运行的路由上下文输入节点上的消息。**

   如果您的项目不包含测试信息，您可以创建它们，如 第 22.1 节 “为路由追踪创建测试消息” 所述。

6. **单击 Yes 以确认切换到 Debug 视角。**

7. **在 Camel debugger 中，像常规一样，逐步浏览断点的消息（请参阅 第 14 章 *运行 Camel Debugger*），以查看您生成的更改的结果。**

Camel debugger 在 Edit Routes 模式中的行为与通常调试模式相同，因此您可以在消息传输路由上下文时使用任何 Camel 调试器的功能。

注意

当消息到达路由上下文结束时，调试器将暂停。要继续调试，请切换回 Fuse Integration 视角，并在 JMX Navigator 视图中的输入节点上丢弃另一个消息。每次这样做时，工具会要求您确认切换到 Debug 视角。

注意

在路由编辑会话过程中，可能会丢失与正在运行的路由上下文的连接。如果发生这种情况，则在 JMX Navigator 视图中，您会看到如下内容： Local Process → maven[xxxx][Disconnected]。要继续会话，您必须重新连接到正在运行的路由上下文，在 JMX Navigator 视图中选择它，然后重新选择 Edit Routes。

终止路由编辑会话

1. 在 Debug 视角的 Debug 视图中，选择 Remote Camel Debug - camelContext--<contextId>--xxxxxxxxxxxxxxxxxx.xml [Remote Camel Context] thread，然后点击菜单栏上的

来终止调试会话。

2. 在 Console 视图的菜单栏中，点

终止路由上下文。

3. 如果要清除控制台输出，请点击 Console 视图的菜单栏中的

。

4. 切换到 Fuse Integration 视角，在路由编辑器中点 Remote CamelContext:<contextId> 标签页中的

来关闭路由上下文文件的内存中模型。

相关主题

- 第 2 章 *在路由编辑器中编辑路由上下文*
- 第 II 部分 “调试路由上下文”

第 26 章 管理路由上下文

JMX Navigator 视图可让您暂停和恢复运行路由上下文。

26.1. 挂起路由上下文的操作

概述

该工具允许您在 JMX Navigator 视图中暂停路由上下文的操作。暂停上下文操作会正常关闭上下文中的所有路由，但将其加载到内存中，以便他们可以恢复操作。

流程

挂起路由上下文的操作：

1. 在 JMX Navigator 视图中，展开项目的 Camel 上下文 节点，然后选择您要暂停其操作的路由上下文。

2. 右键单击所选路由上下文以打开上下文菜单，然后选择 Suspend Camel Context。

注意

如果 Resume Camel Context 出现在上下文菜单中，则上下文的操作已被暂停。

相关主题

- 第 26.2 节 “恢复路由上下文的操作”

26.2. 恢复路由上下文的操作

概述

该工具可让您在 JMX Navigator 视图中恢复运行暂停的路由上下文。恢复上下文的操作会重启其中的所有路由，以便它们可以处理消息。

流程

恢复路由上下文的操作：

1. 在 **JMX Navigator** 视图中，展开项目的 **Camel** 上下文 节点，然后选择您要恢复其操作的路由上下文。

2. 右键单击所选上下文以打开上下文菜单，然后选择 **Resume Camel Context**。

注意

如果 **Suspend Camel Context** 出现在上下文菜单中，则上下文及其路由正在运行。

相关主题

- 第 26.1 节 “挂起路由上下文的操作”

部分 IV. 将应用程序发布到容器

要将 Fuse 集成项目发布到服务器容器，您必须首先将服务器及其 runime 定义添加到工具的 Servers 列表中。然后，您可以将项目分配给服务器运行时，并为它设置发布选项。

第 27 章 管理服务器

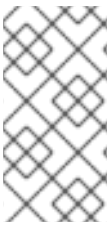
注意

有关如何将 Camel 项目发布到红帽 Fuse 的步骤，请参考 *第 28 章 将 Fuse 集成项目发布到服务器*。

27.1. 添加服务器

概述

对于用于管理服务器的工具，您需要将服务器添加到 Servers 列表中。添加后，服务器将显示在 Servers 视图中，您可以在其中连接它并发布 Fuse 集成项目。



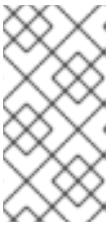
注意

如果添加红帽 Fuse 服务器，建议您编辑 *installDir*/etc/users.properties 文件并添加用户信息，格式为 user=password,role，以便工具建立与服务器的 SSH 连接。

流程

在服务器视图中添加新服务器的方法有三种：

- 在 Servers 视图中，点 No servers are available。点击这个链接来创建新 server...。

  注意

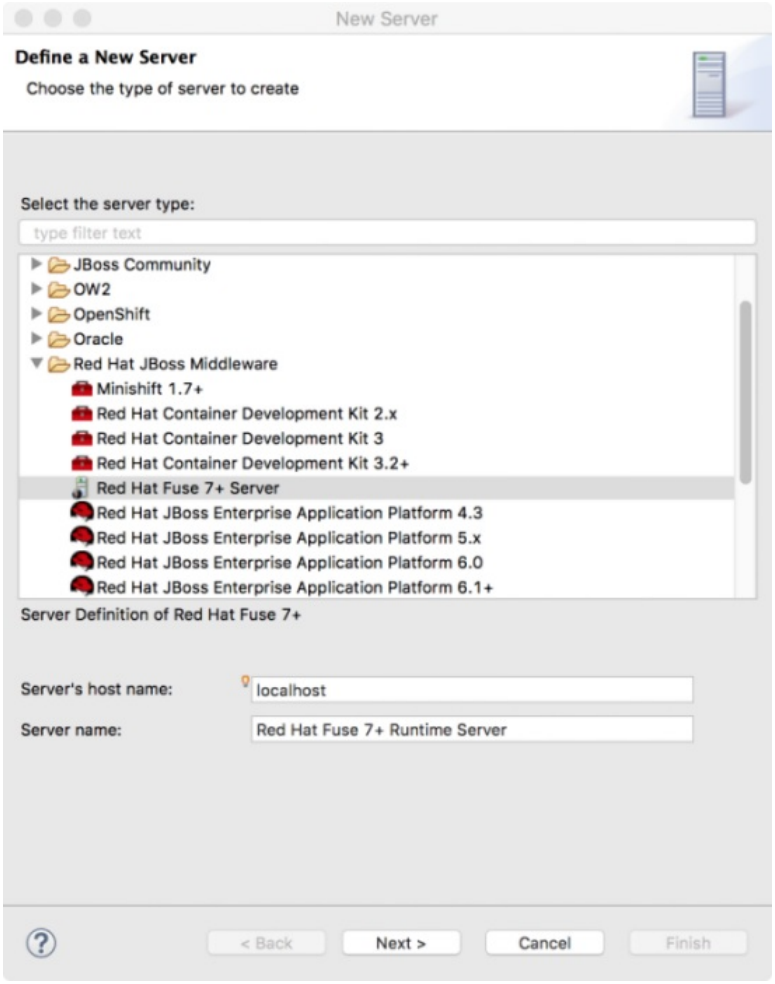
  只有未定义服务器时，此链接才会出现在 Servers 视图中。如果您在首次创建项目时定义了并添加了服务器，则服务器视图将显示该服务器。

- 在 Servers 视图中，右键单击打开上下文菜单并选择 New → Server。

- 在菜单栏中，选择 File → New → Other → Server → Server。

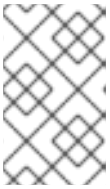
在 Define a New Server 对话框中添加新服务器：

1. 扩展 Red Hat JBoss Middleware 节点以公开可用服务器选项列表：



2. 点您要添加的服务器。

3. 在 服务器主机名 字段中，接受默认值(localhost)。

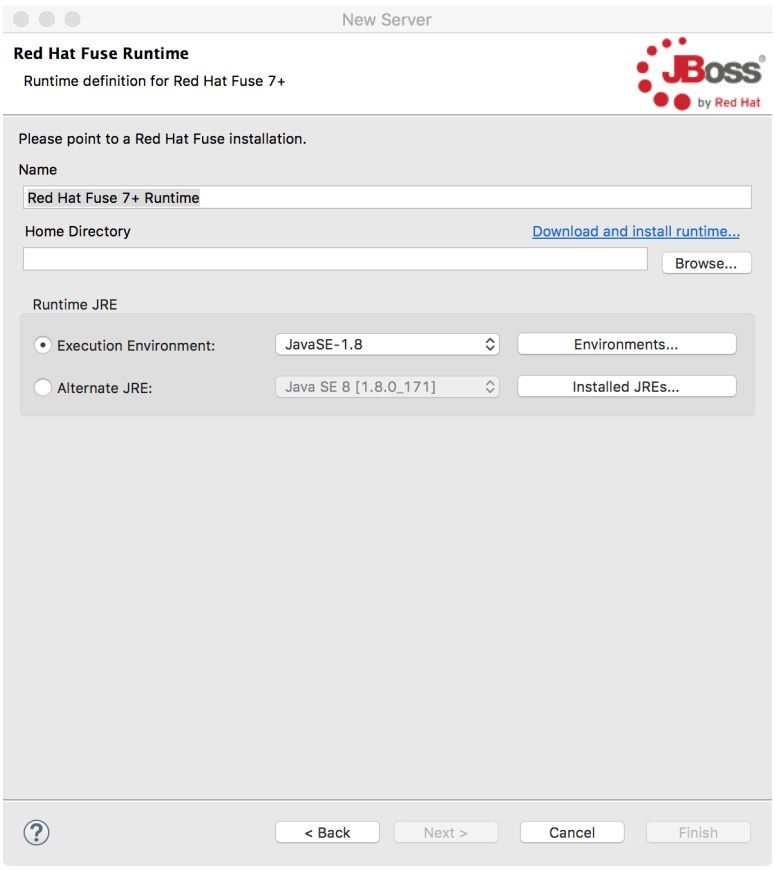


注意

localhost 的地址是 0.0.0.0。

4. 在 Server name 字段中，接受默认值，或者为运行时服务器输入不同的名称。

5. 对于 服务器运行时环境，接受默认值或单击 Add 以打开服务器的运行时定义页面：



注意

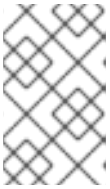
如果服务器上尚未安装服务器，您可以点 Download and install runtime... 并按照站点的下载说明安装它。根据站点，您可能需要提供有效的凭证，然后才能继续下载过程。

6. 接受安装 名称 的默认内容。

7. 在 Home Directory 字段中，输入安装服务器运行时的路径，或者点击 Browse 查找并选择它。

8. 在 Execution Environment 旁边，从下拉菜单中选择运行时 JRE。

如果您希望的版本没有出现在列表中，点 Environments，然后从出现的列表中选择版本。您选择的 JRE 版本必须安装在您的计算机上。

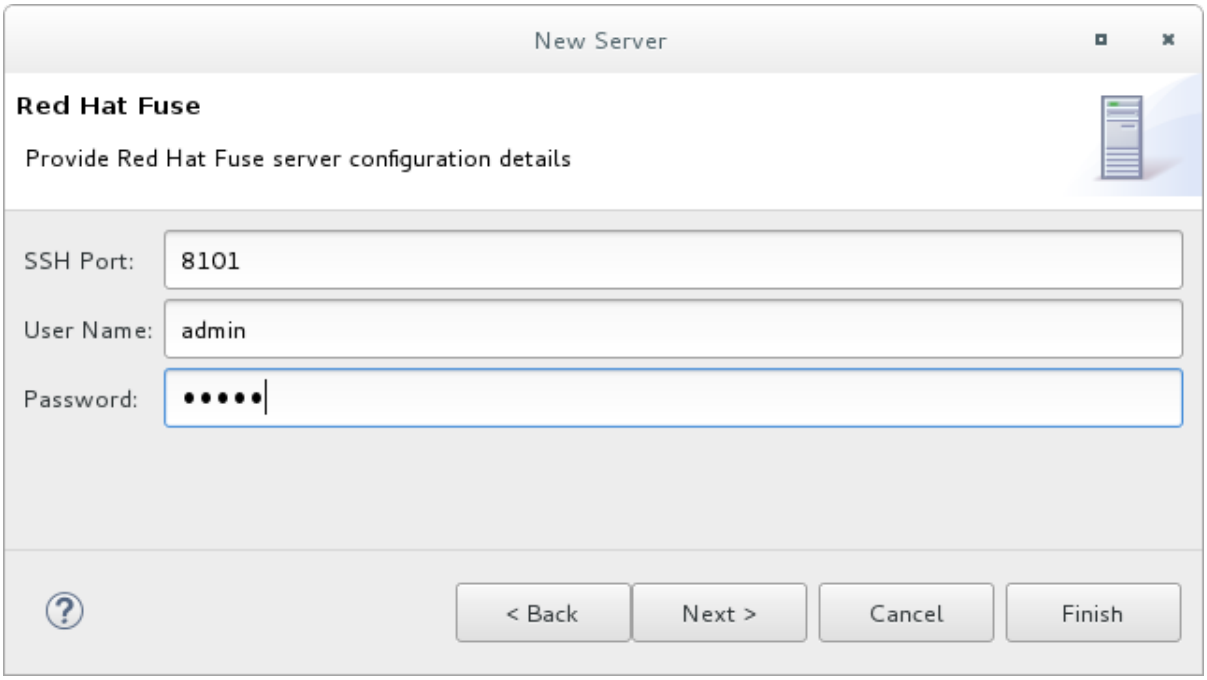


注意

有关所需 Java 版本，请参阅 Red Hat Fuse 支持的配置。

9. Alternate JRE 选项保留原样。

10. 点 Next 保存服务器的运行时定义并打开其 Configuration details 页面：



11. 接受 SSH 端口 的默认端口(8101)。

运行时使用 SSH 端口来连接服务器的 Karaf shell。如果此默认值适合您的设置，您可以通过查找服务器的 *installDir*/etc/org.apache.karaf.shell.cfg 文件来发现正确的端口号。

12. 在 User Name 字段中输入用于登录服务器的名称。

对于 Red Hat Fuse，这是存储在 Red Hat Fuse *installDir*/etc/users.properties 文件中的用户名。

注意

如果在 /etc/users.properties 文件中激活了默认用户（未注释），工具会自动使用默认用户名和密码填充 User Name 和 Password 字段，如 [servCnfigDetails] 所示。

如果用户尚未设置，您可以使用格式 user=password,role （如 joe=secret,Administrator）向该文件添加一个，或者您可以使用 karaf jaas 命令设置：

- jaas:realms swig-wagonto 列出域
- jaas:manage --index 1 swig- swigto 编辑第一个(server)域
- jaas:useradd <username> <password > HEKETI- swigto 添加用户和关联的密码
- jaas:roleadd <username> Administrator wagon-rhacmto 指定新用户的角色
- jaas:update HEKETI- swigto 使用新的用户信息更新域

如果已经为服务器选择了 jaas 域，您可以通过发出 JBossFuse:karaf@root>jaas:users 来发现用户名。

13. 在 Password 字段中，输入 User Name 以登录到服务器所需的密码。

14. 点 Finish 保存服务器的配置详情。

服务器运行时会出现在 Servers 视图中。

扩展服务器节点会公开服务器的 JMX 节点：

27.2. 启动服务器

概述

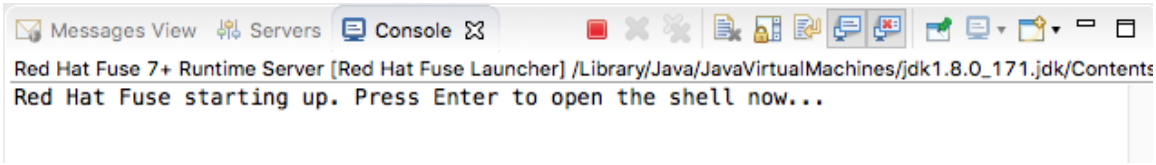
当您启动配置的服务器时，工具会在 Terminal 视图中打开服务器的远程管理控制台。这可让您在测试应用程序时轻松地管理容器。

流程

启动服务器：

1. 在 Servers 视图中，选择您要启动的服务器。
2. 点击  。

- Console 视图会打开并显示一条消息，要求您在容器启动时等待，例如：



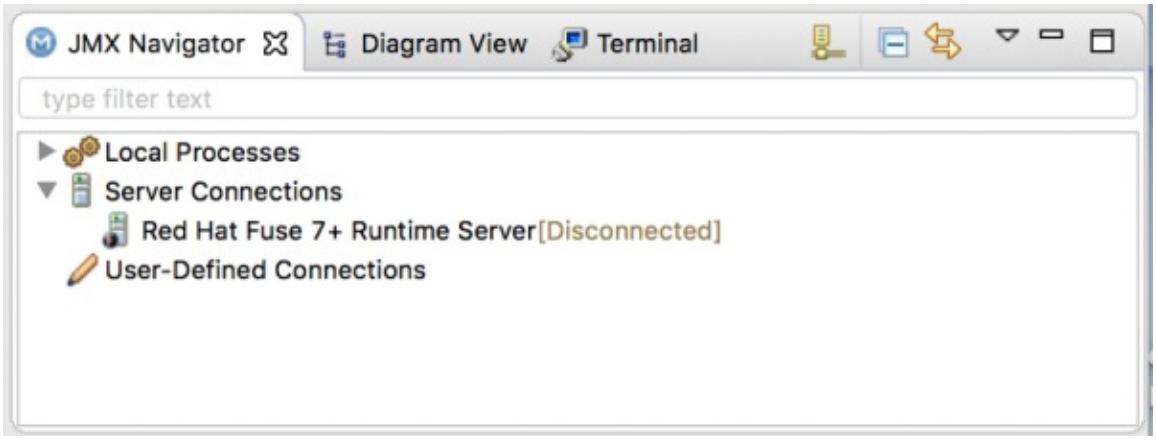
注意

如果您没有正确配置用户名和密码来打开远程控制台，则会打开一个对话框，要求您输入正确的凭证。请参阅 第 27.1 节 “添加服务器”。

- 容器启动后，会打开 Terminal 视图来显示容器的管理控制台。

- 运行的服务器会出现在 **Servers** 视图中：

- 运行的服务器也会出现在 **Server Connections** 下的 **JMX Navigator** 视图中：



注意

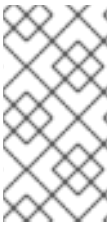
如果服务器在与工具相同的计算机上运行，则服务器在本地 进程 下也有一个条目。

27.3. 连接到正在运行的服务器

概述

启动配置的服务器后，它会出现在 **Servers** 视图中，并在 **Server Connections** 节点的 **JMX Navigator** 视图中显示。您可能需要扩展 **Server Connections** 节点来查看服务器。

要在运行的服务器上发布并测试 **Fuse** 项目应用程序，您必须首先与其连接。您可以在 **Servers** 视图中或 **JMX Navigator** 视图中连接到正在运行的服务器。



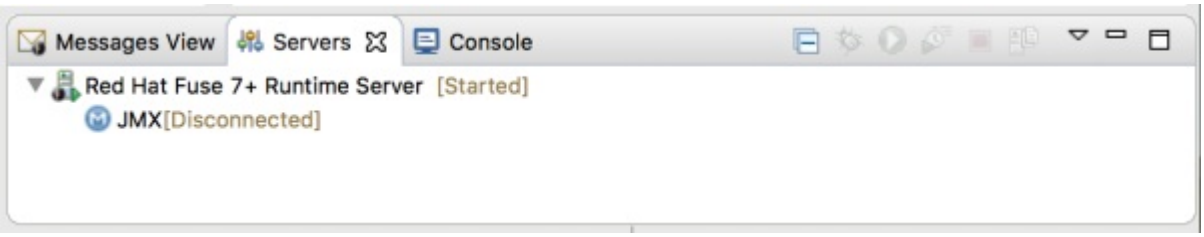
注意

服务器 视图和 **JMX Navigator** 视图与服务器连接同步。也就是说，在服务器视图中连接 服务器 也会在 **JMX Navigator** 视图中连接，反之亦然。

在 Servers 视图中连接到正在运行的服务器

1. **在 Servers 视图中，扩展服务器运行时以公开其 JMX[Disconnected] 节点。**

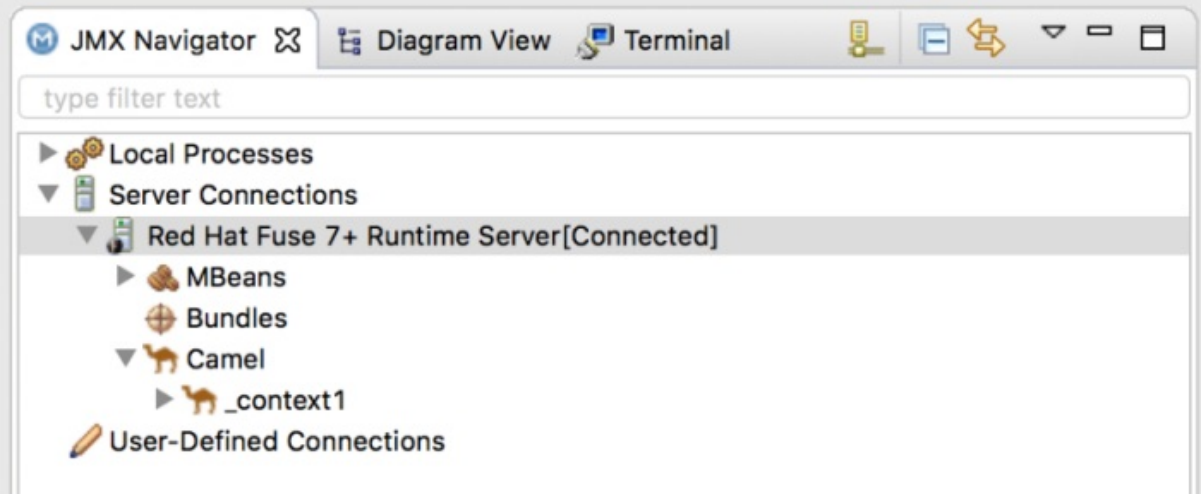
2. **双击 JMX[Disconnected] 节点：**



在 JMX Navigator 视图中连接到正在运行的服务器

1. **在 JMX Navigator 视图中，在 Server Connections 节点下，选择您要连接的服务器。**

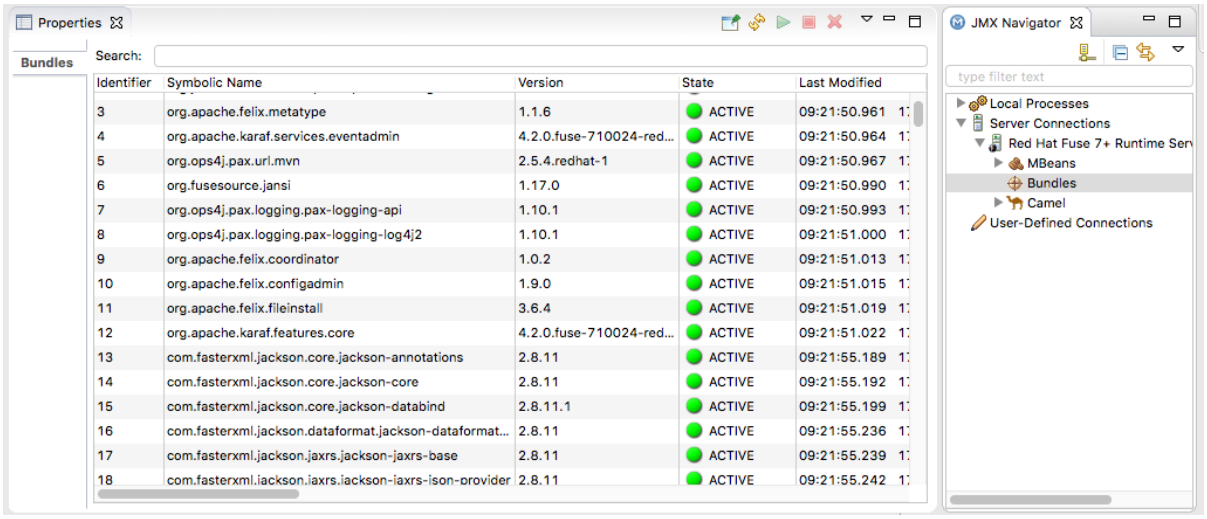
2. **双击所选服务器：**



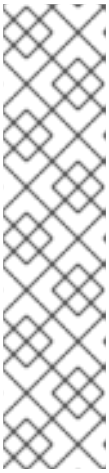
查看连接的服务器上安装的捆绑包

1. **在 Servers 视图或 JMX Navigator 视图中，展开服务器运行时树以公开 Bundles 节点，然后选择它。**

2. **这些工具使用服务器上安装的捆绑包列表填充 Properties 视图：**



使用 Properties 视图的 搜索工具，您可以根据其 Symbolic Name 或其 标识符 搜索捆绑包（如果您知道它）。当您输入符号名称或标识符时，列表更新仅显示与当前搜索字符串匹配的捆绑包。



注意

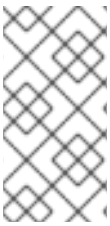
或者，您可以在 Terminal 视图中发出 osgi:list 命令，以查看在红帽 Fuse 服务器运行时上安装的生成的捆绑包列表。该工具对 OSGi 捆绑包使用不同的命名方案，由 osgi:list 命令显示。

在项目的 pom.xml 文件的 < build > 部分中，您可以找到捆绑包的符号链接名称及其捆绑包名称(OSGi)，在 maven-bundle-plugin 条目中列出的。如需了解更多详细信息，请参阅 “验证项目是否已发布到服务器”一节。

27.4. 断开与服务器的连接

概述

测试完应用程序后，您可以在不停止的情况下断开与服务器的连接。



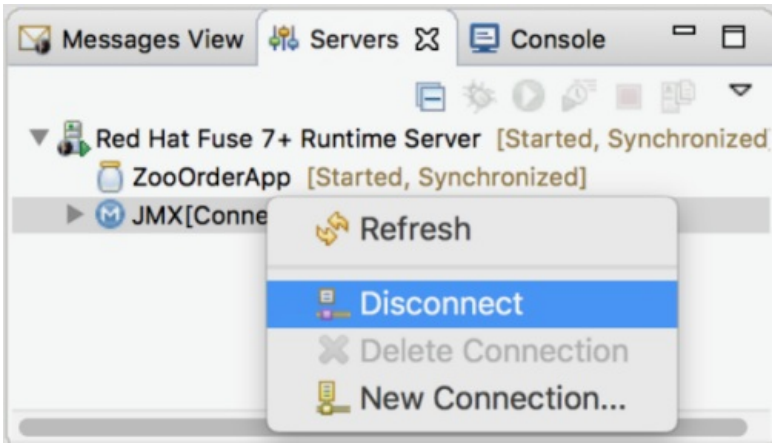
注意

服务器 视图和 JMX Navigator 视图与服务器连接同步。也就是说，与 Servers 视图中的服务器断开连接，也会在 JMX Navigator 视图中断开它，反之亦然。

在 Servers 视图中断开与服务器的连接

1. 在 Servers 视图中，扩展服务器运行时以公开其 JMX[Connected] 节点。

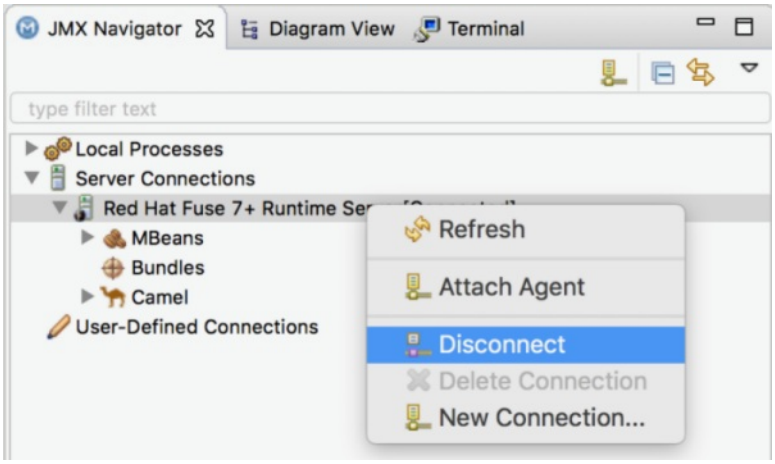
2. 右键单击 JMX[Connected] 节点以打开上下文菜单，然后选择 Disconnect。



在 JMX Navigator 视图中断开与服务器的连接

1. 在 JMX Navigator 视图中，在 Server Connections 下选择要断开连接的服务器。

2. 右键单击所选服务器以打开上下文菜单，然后选择 Disconnect。



27.5. 停止服务器

概述

您可以在 Servers 视图中关闭服务器，或者在 Terminal 视图中服务器的远程控制台中关闭。

使用 Servers 视图

停止服务器：

1. 在 Servers 视图中，选择您要停止的服务器。
2. 点击  。

使用远程控制台

停止服务器：

1. 打开托管服务器远程控制台的 Terminal 视图。
2. 按： CTRL+D

27.6. 删除服务器

概述

当您使用配置的服务器完成，或者错误配置服务器时，您可以将其及其配置删除。

首先，从 Servers 视图或从 JMX Navigator 视图中删除服务器。接下来，删除服务器的配置。

删除服务器

1. 在 Servers 视图中，右键单击您要删除的服务器，以打开上下文菜单。
2. 选择 Delete。
3. 点击 确定。

删除服务器配置

在 Linux 和 Windows 机器上，选择 Window → Preferences。

1. 展开 Server 文件夹，然后选择 Runtime Environments 以打开 Server Runtime Environments 页面。

2. 从列表中，选择您之前从 Servers 视图删除的服务器的运行时环境，然后单击 Remove。

3. 点击 确定。

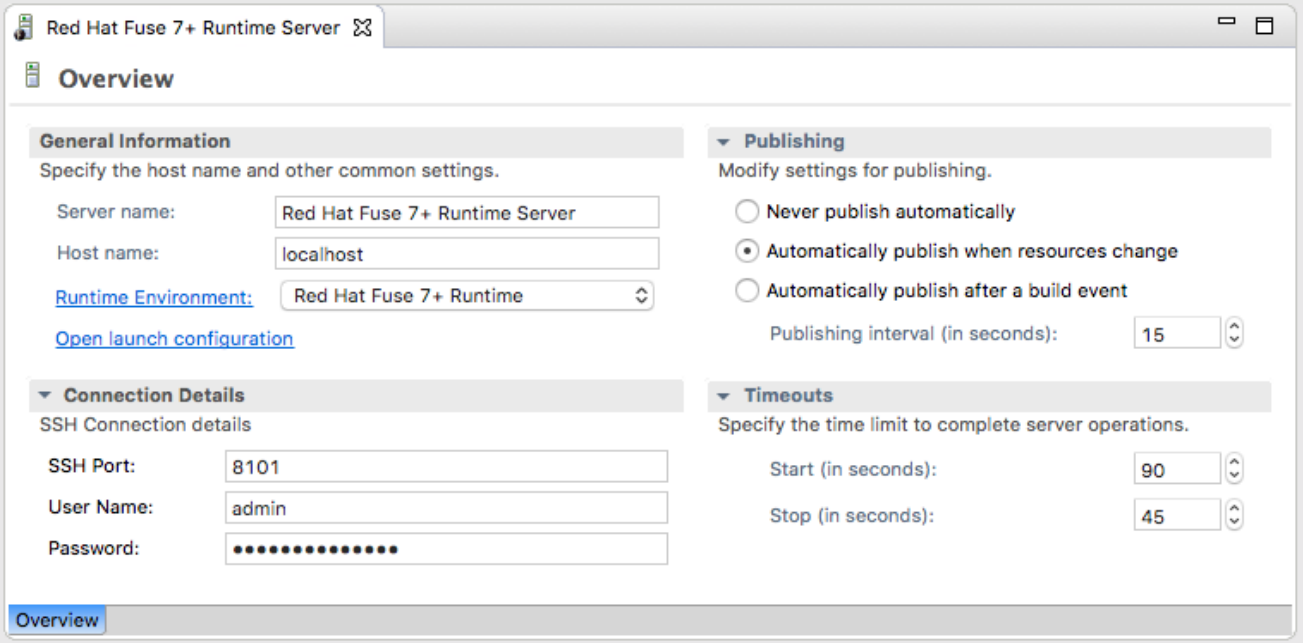
第 28 章 将 FUSE 集成项目发布到服务器

您可以使用 Eclipse 发布机制将 Fuse 集成项目部署到服务器运行时。为此，您必须定义服务器，并将服务器添加到 Fuse 集成 视角中的 Servers 视图中。有关逐步演示，请参阅。

概述

您可以设置受支持的服务器来自动发布分配的 Fuse 项目，或者仅在手动调用 publish 命令时发布它们。

添加到 Servers 视图的每个服务器运行时都有自己的 Overview 页面，其中包含其配置、连接和发布详情：



您可能需要扩展 发布 以公开服务器运行时发布选项和默认设置：

- 永远不会自动发布 HEKETI- swigYou，才能选择此选项来手动发布项目。

重要

您还必须禁用 如果服务器启动，在服务器的 Add 和 Remove 页面中立即发布更改 选项（详情请参阅 “手动发布 Fuse 项目”一节。

-

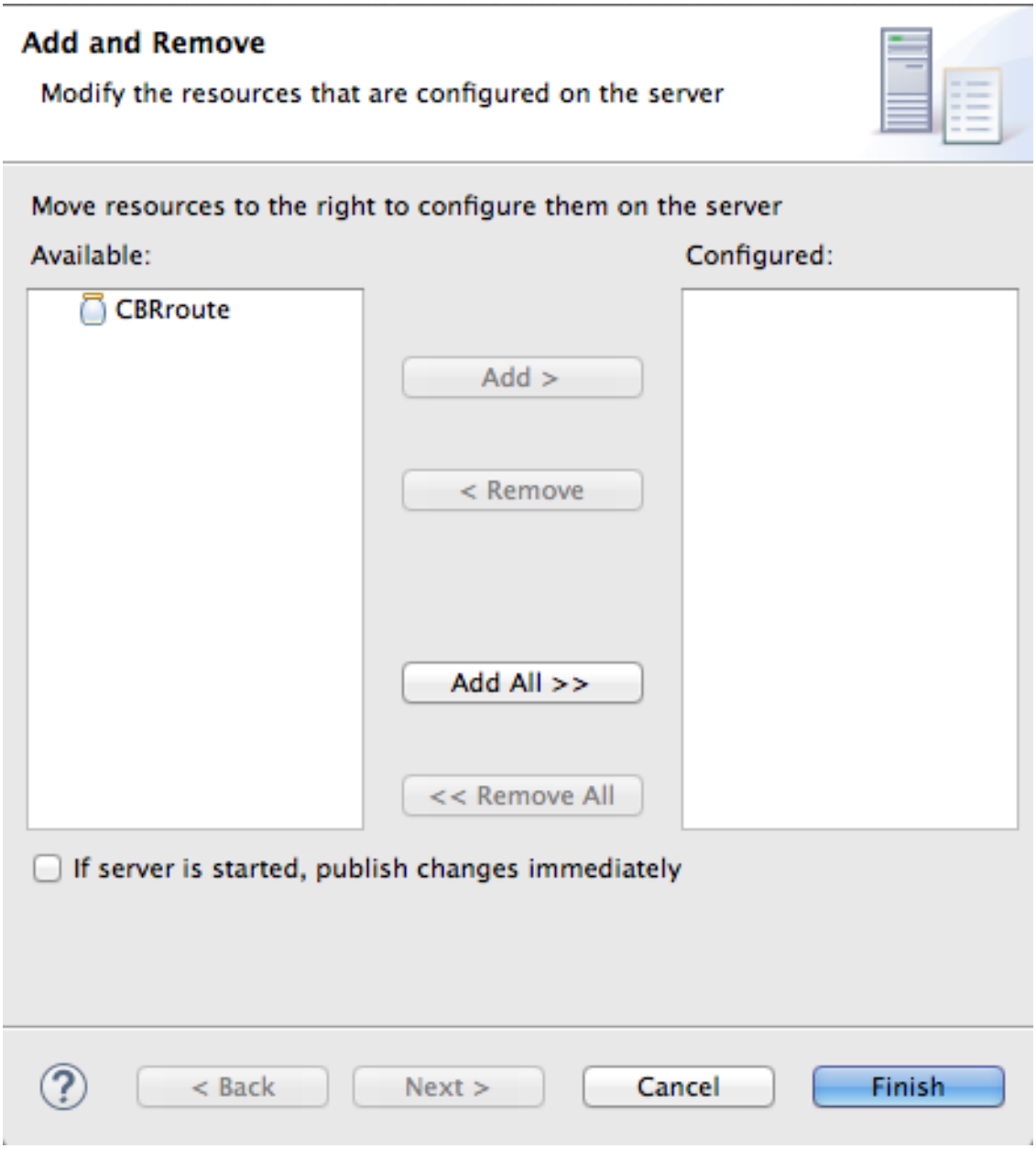
当资源更改 InventoryService-wagon[default] 时，自动发布此选项，以在保存对它所做的更改时自动发布或重新发布 Fuse 项目。发布项目的速度取决于 发布间隔 （默认为 15 秒）。

- 在构建事件 swig -wagonFor Fuse 项目后自动发布，在资源 更改时与 Automatically publish 相同。

资源更改时自动发布 FUSE 项目

在资源 更改时，服务器运行时的默认发布选项是自动发布。

1. 如有必要，启动您要发布 Fuse 项目的服务器运行时。详情请查看 第 27.2 节 “启动服务器”。
2. 在 Servers 视图中，双击服务器运行时以打开其 Overview 页面。
3. 展开 发布，然后在 资源更改时选择 Automatically publish。
4. 要在发布周期之间增加或减少间隔，请单击 发布间隔（以秒为单位） 上或缩减的单选按钮。
5. 在 Servers 视图中，右键单击服务器运行时以打开上下文菜单，然后选择" 添加和删除 "。

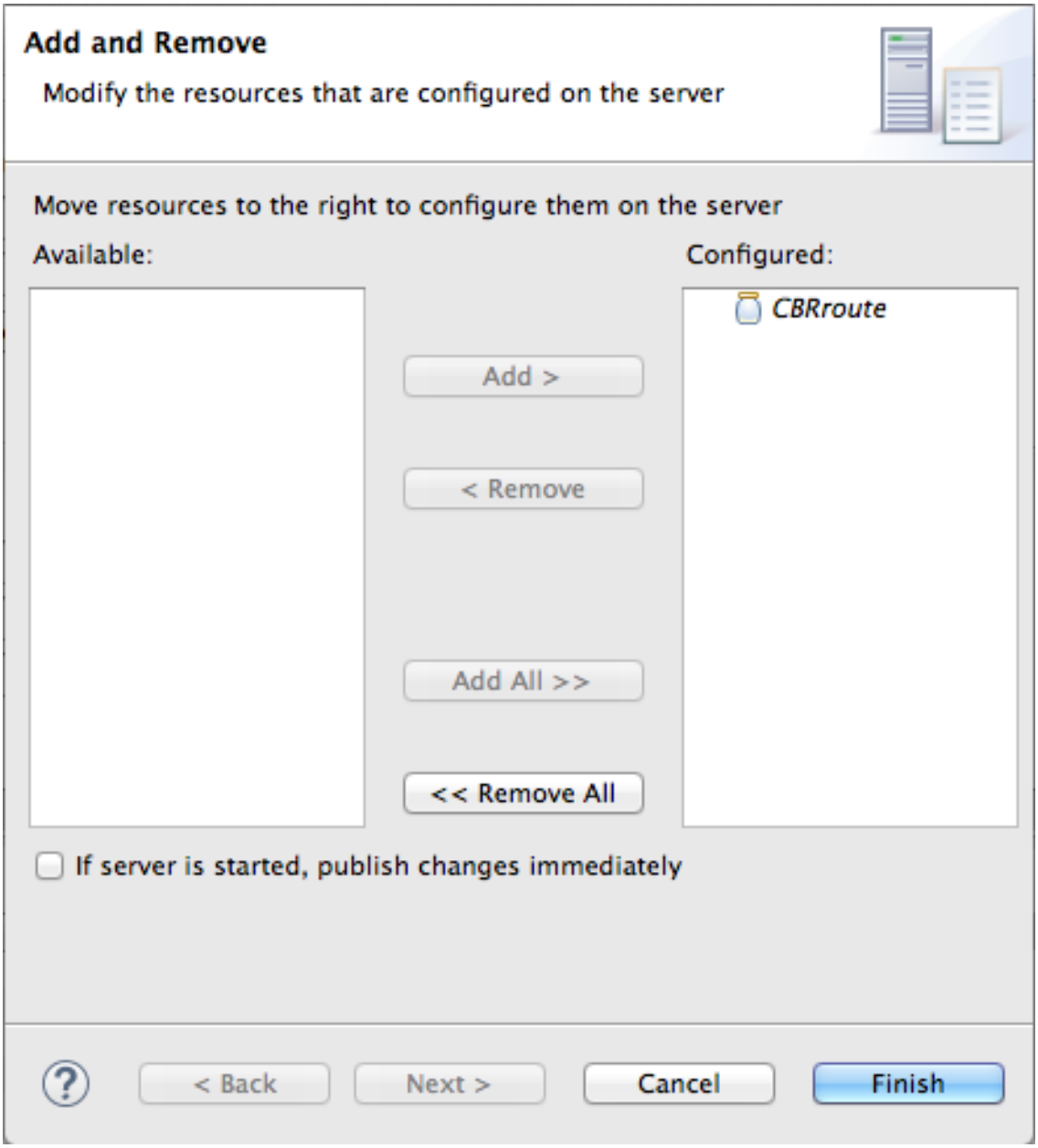


所有可用于发布的资源都显示在 **Available** 列中。

6. 为服务器运行时分配一个资源（本例中为 **CBRroute Fuse** 项目）：

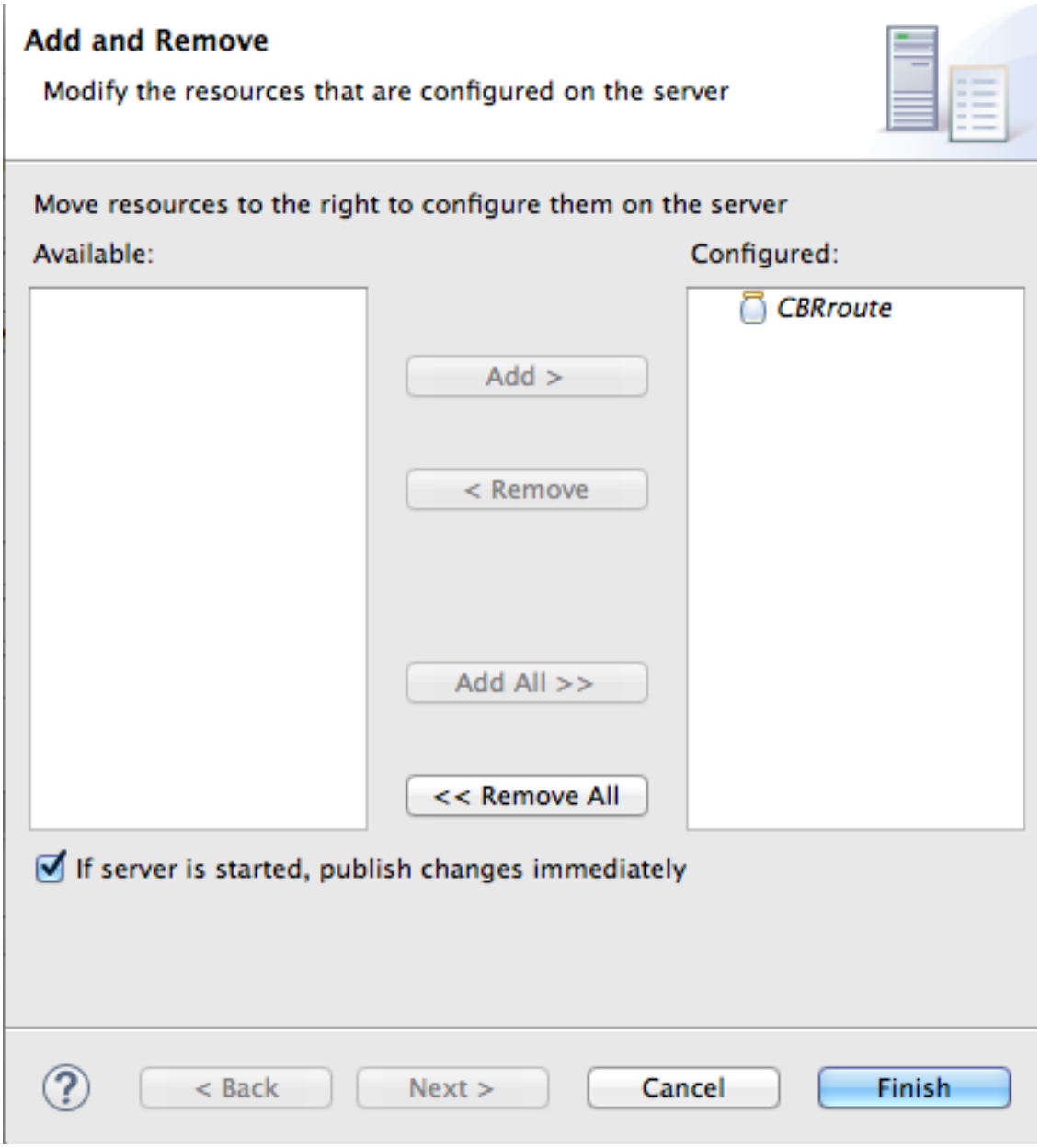
- 双击它，或者
- 选择它，然后单击 **Add**。

所选资源移到 **Configured** 列：



在这个阶段，实际发布资源的时间取决于服务器运行时是否在运行和发布 间隔 设置。但是，如果服务器停止，则必须在启动服务器后手动发布项目（详情请参阅 “手动发布 Fuse 项目”一节）。

7. 点 If server started，发布立即更改 选项来启用它：



此选项可确保当您单击 Finish 后，配置的项目会立即发布。当服务器运行时 Overview 页面中的 资源更改选项自动 发布时，会在保存对本地项目更改时重新发布配置的项目。

8. 点 Finish。

该项目会出现在服务器运行时节点的 Servers 视图中，服务器运行时状态报告 [Started,Publishing…].

发布完成后，服务器运行时和项目报告的状态为 [Started,Synchronized] ：

注意

对于服务器运行时，同步同步 意味着服务器上的所有发布的资源都与其本地对应的资源相同。对于公布的资源，同步 表示它与其本地对应的项相同。

手动发布 FUSE 项目

1. 如有必要，启动您要发布 Fuse 项目的服务器运行时。详情请查看 第 27.2 节 “启动服务器”。
2. 在 Servers 视图中，双击服务器运行时以打开其 Overview 页面。
3. 展开 Publishing，然后选择 Never publish automatically。
4. 点 File → Save 保存发布选项更改。
5. 如果 Fuse 项目已分配给服务器运行时，请确保此选项被禁用：如果服务器 启动，请立即发布更改 ：
   a. 在 Servers 视图中，右键单击服务器运行时以打开上下文菜单。
   b. 点 Add and Remove... 打开服务器的 Add 和 Remove 页面。
   c. 如果启用了以下选项，请禁用该选项： 如果服务器启动，请立即发布更改。
   d. 跳至 [finish]。

6. 如果 Fuse 项目还没有分配给服务器运行时，请立即分配它：

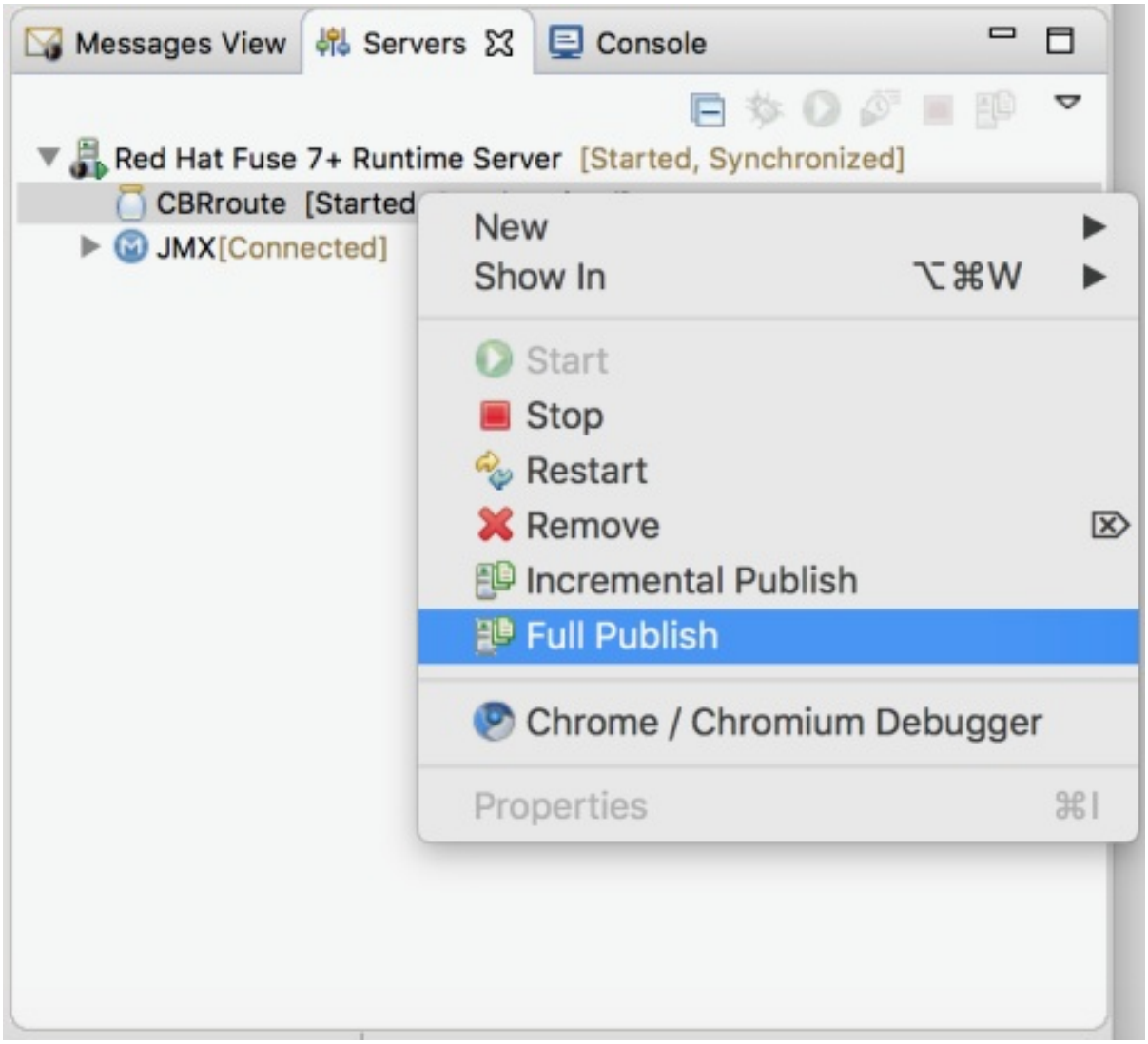
   a. 在 [startAssignResource] 中遵循 [stopAssignResource] 到 “资源更改时自动发布 Fuse 项目”一节 的 “资源更改时自动发布 Fuse 项目”一节。

   b. 不要启用 服务器启动，请立即发布更改。

7. 点 Finish。

   该项目会出现在服务器运行时节点的 Servers 视图中，服务器运行时状态报告 [Started]:

8. 在 Servers 视图中，右键单击项目的节点。在本例中，选择 CBRroute Fuse 项目以打开上下文菜单：



9. 选择 **Full Publish**。

在发布操作期间，服务器运行时和项目报告 [Started,Republish] 的状态。

发布完成后，服务器运行时的状态和项目报告 [Started,Synchronized]:

注意

该工具不支持 Incremental Publish 选项。点击 Incremental Publish 结果全面发布。

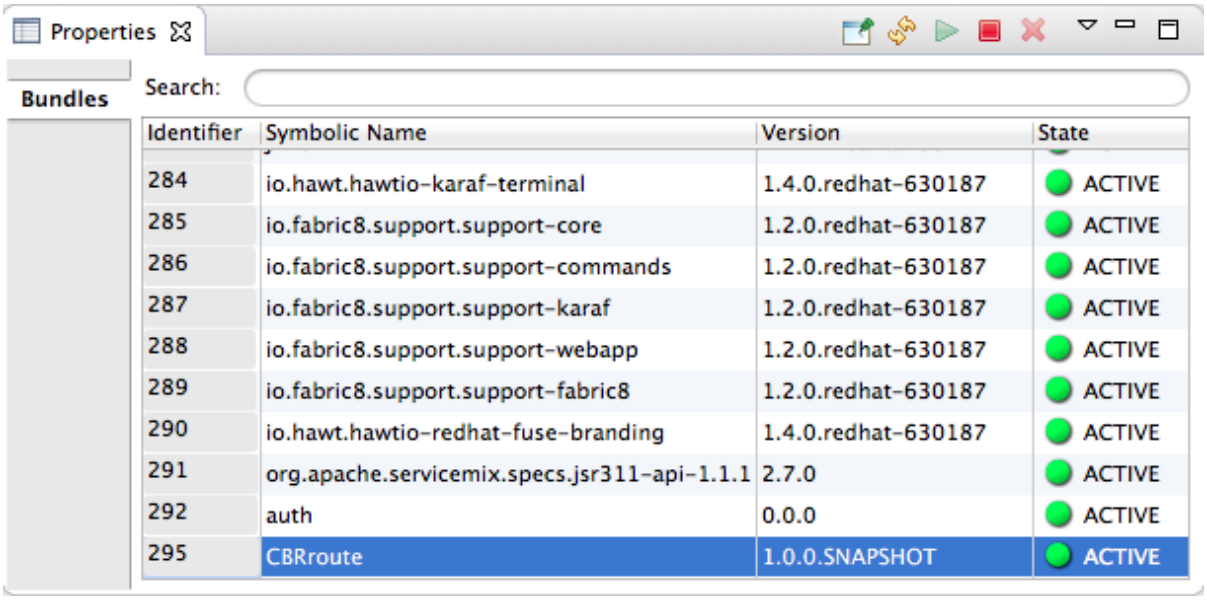
验证项目是否已发布到服务器

将 Fuse 项目发布到服务器运行时后，您可以连接到服务器并检查该项目的捆绑包是否已安装。

1. 连接到服务器运行时。详情请查看 “在 Servers 视图中连接到正在运行的服务器”一节。

2. 在 Servers 视图中，展开服务器运行时树，以公开 Bundles 节点并选择它。

   这些工具使用服务器上安装的捆绑包列表填充 Properties 视图：



3. 要查找项目的捆绑包，可向下滚动到列表的底部，或者在 Properties 视图的 Search 框中开始输入捆绑包的 Symbolic Name。捆绑包的 Symbolic Name 是您在创建项目时给出项目的名称。

注意

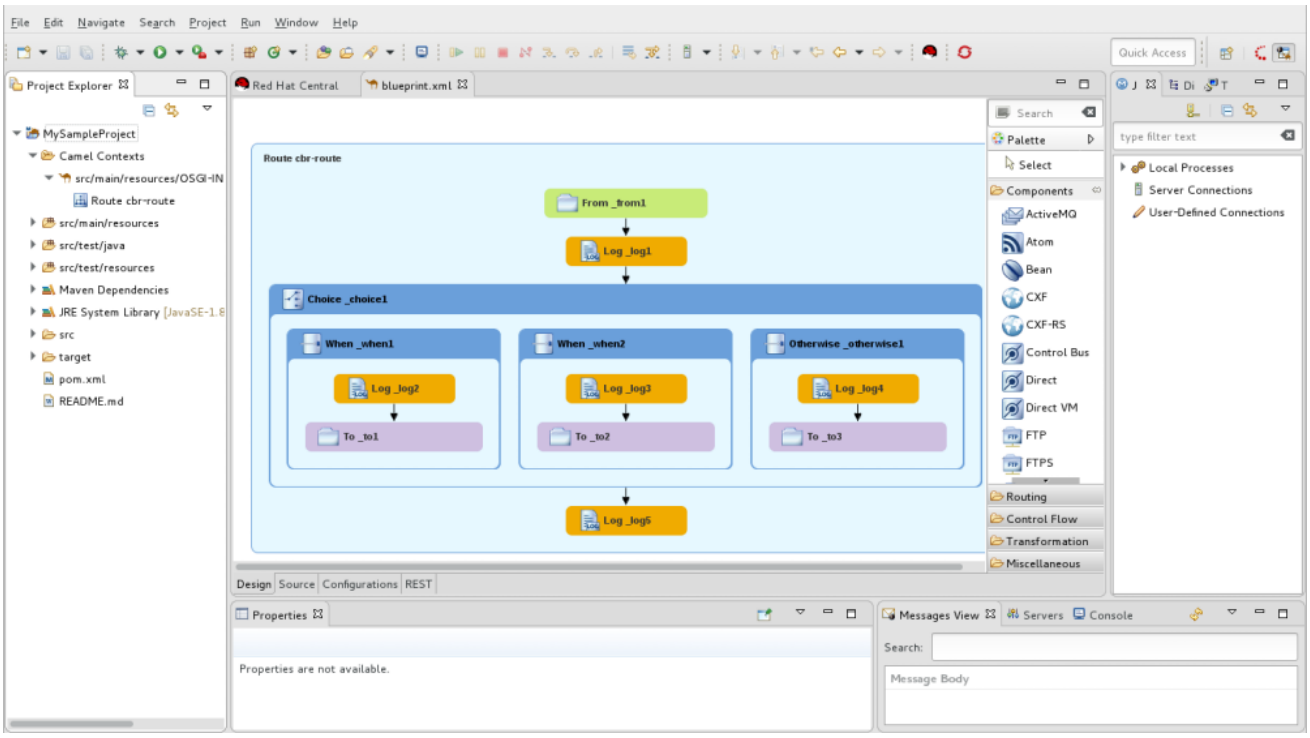
或者，您可以在 Terminal 视图中发出 osgi:list 命令，以查看在 Fuse 服务器运行时上安装的捆绑包列表。该工具对 OSGi 捆绑包使用不同的命名方案，由 osgi:list 命令显示。

在项目的 pom.xml 文件的 < build > 部分中，您可以找到捆绑包的符号链接名称及其捆绑包名称(OSGi)在 maven-bundle-plugin 条目中列出的，例如：

```
<build>
  <defaultGoal>install</defaultGoal>
  <plugins>
    <plugin>
      <groupId>org.apache.felix</groupId>
      <artifactId>maven-bundle-plugin</artifactId>
      <version>${version.maven-bundle-plugin}</version>
      <extensions>true</extensions>
      <configuration>
        <instructions>
          <Bundle-SymbolicName>CBRroute</Bundle-SymbolicName>
          <Bundle-Name>Empty Camel Blueprint Example [CBRroute]</Bundle-Name></instructions>
      </configuration>
    </plugin>
```

附录 A. FUSE INTEGRATION PERSPECTIVE

使用 Fuse Integration 视角设计、监控、测试和发布您的集成应用程序。



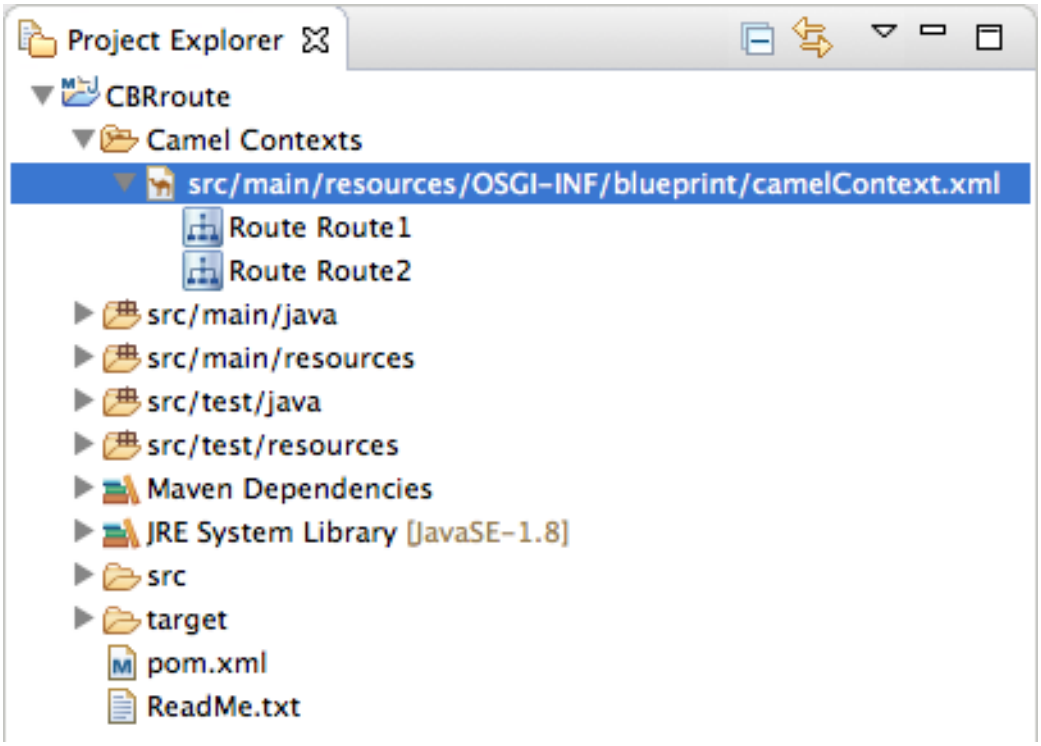
您可以使用以下方法打开 Fuse Integration 视角：

- 当您创建新的 Fuse 集成项目（请参阅 第 1 章 *创建新的 Fuse 集成项目*）时，工具切换到 Fuse Integration 视角。
- 点工具栏中的  。如果工具栏中没有  图标，点  ，然后从可用视角列表中选择 Fuse Integration。
- 选择 Window → Perspective → Open Perspective → Fuse Integration。

Fuse 集成 视角由 9 个主要区域组成：

- **Project Explorer 视图**

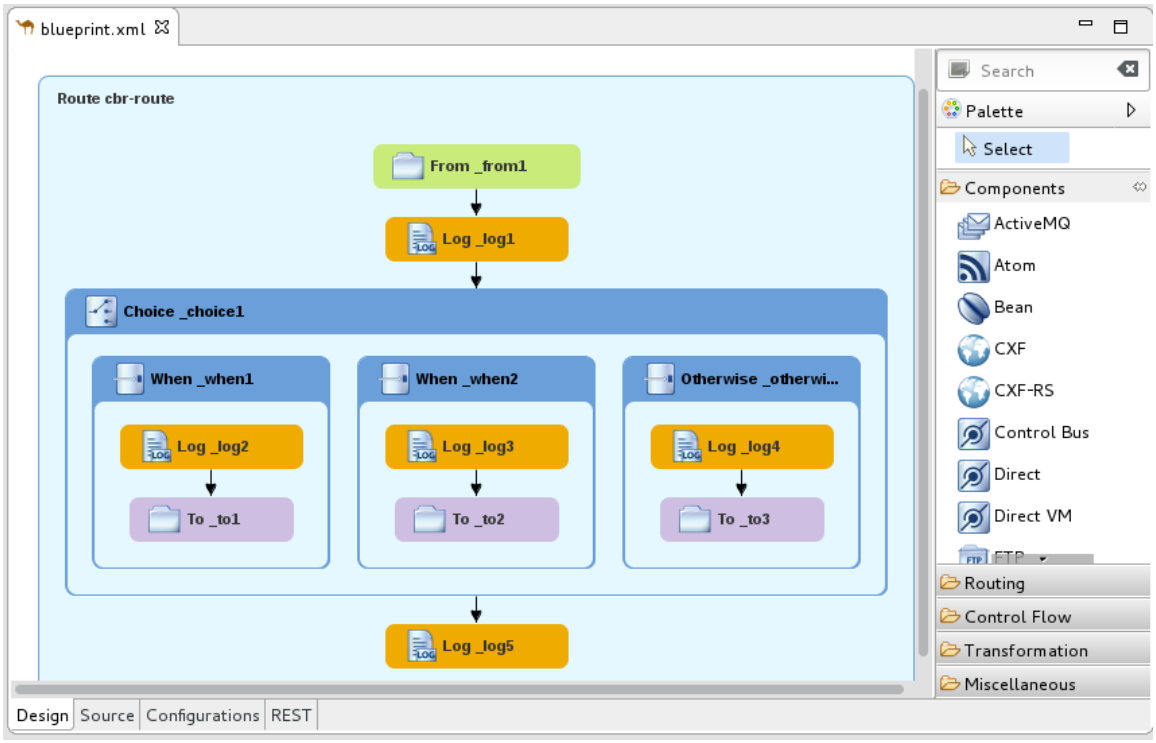
  显示工具已知的所有项目。您可以查看组成每个项目的所有工件。Project Explorer 视图还在其 Camel 上下文节点下显示项目的所有路由上下文 .xml. 文件。这可让您查找并打开项目中包含的路由上下文文件。在每个路由上下文 .xml 文件下，Project Explorer 视图会显示上下文中定义的所有路由。对于多路由上下文，您可以专注于 canvas 上的特定路由。



- **路由编辑器**

  提供主要的设计时间工具，由三个标签组成：

  - 设计 HEKETIDisplays 是一个大型网格区域，在其中构建路由以及从中选择企业集成模式(EIP)和 Camel 组件的面板，然后在 canvas 上连接以形成路由。



canvas 是路由编辑器的工作台以及您的大部分工作。它显示一个或多个路由的图形表示，这些路由由连接的 **EIP** 和 **Camel** 组件组成（在将节点放置在 **canvas** 上时称为节点）。

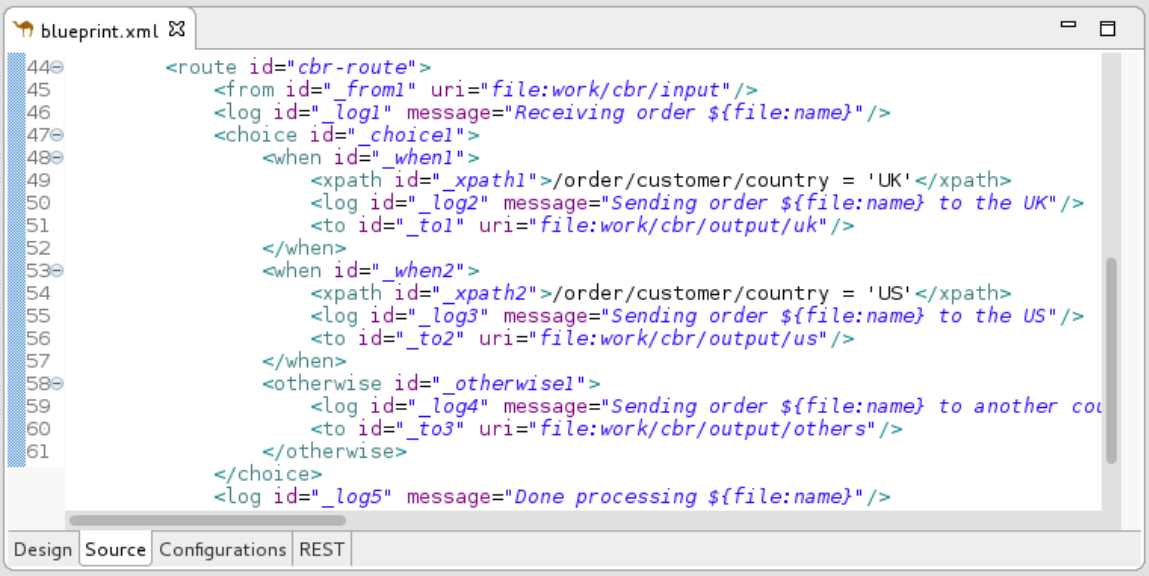
在 **canvas** 上选择一个节点使用应用到所选节点的属性填充 **Properties** 视图，以便您可以编辑它们。

swig 包含构建路由所需的所有模式和 **Camel** 组件，并根据功能 **swig- Components**、路由、控制流、转换和 杂项 对它们进行分组。

- **Source**

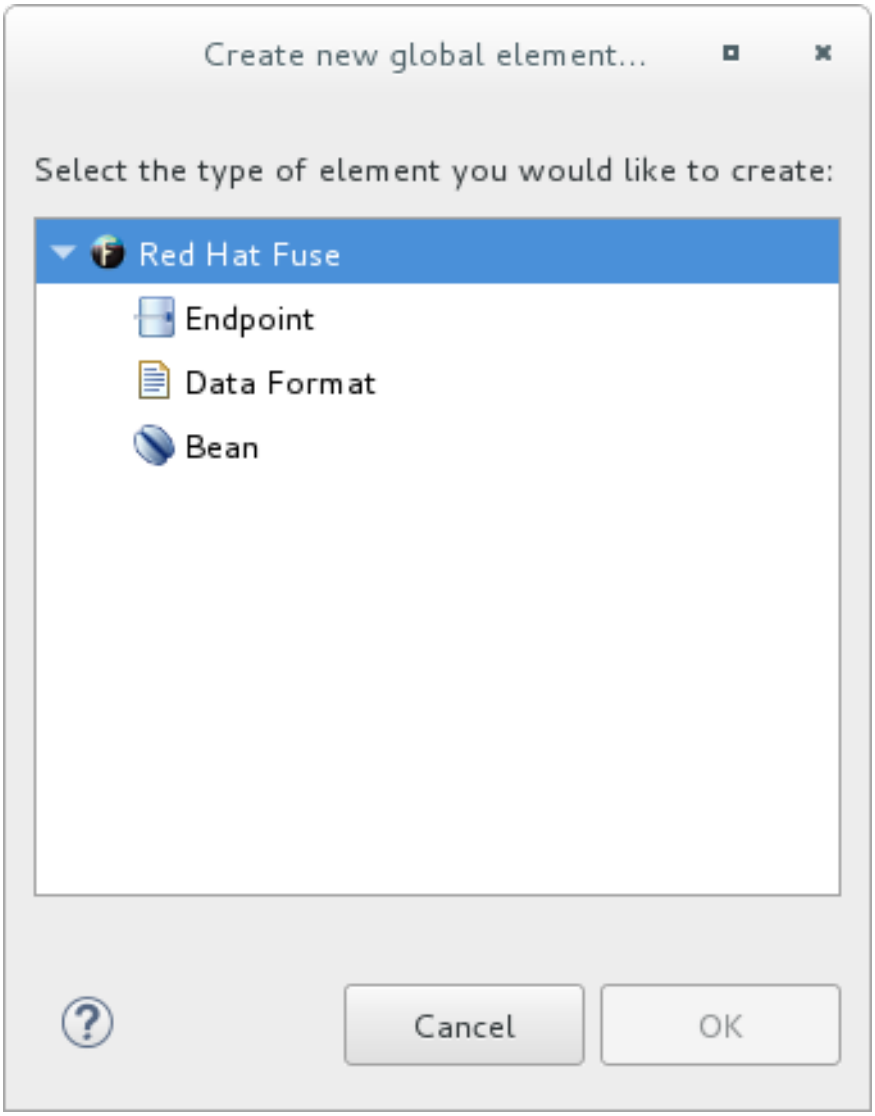
  显示路由编辑器 **canvas** 上构建的路由的 **.xml** 文件的内容。

  您可以在 **Source** 选项卡中和 **Design** 选项卡中编辑路由上下文。**Source** 选项卡可用于编辑并添加任何配置、注释或 **Bean** 到路由上下文文件。内容协助功能可帮助您使用配置文件。在 **Source** 选项卡中，按 **Ctrl+Space** 以查看可插入到项目中的可能值列表。



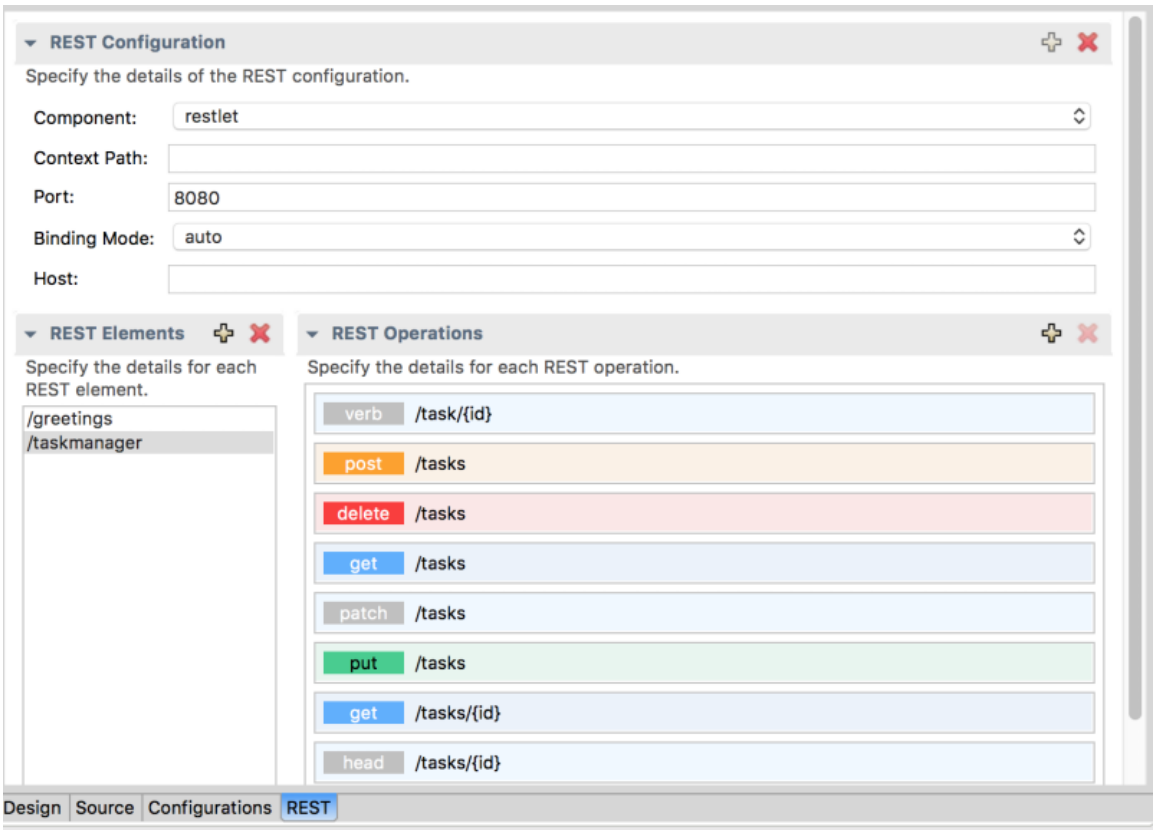
- **配置**

提供了一种简单的方法，将共享配置（全局端点、数据格式、Bean）添加到多路由路由上下文。详情请查看 第 2.6 节 “添加全局端点、数据格式或 Bean”。



- **REST**

  提供 Rest DSL 组件的图形表示。



- **属性 视图**

  显示 canvas 上所选节点的属性。

- **JMX Navigator 视图**

  列出 JMX 服务器及其监控的基础架构。它可让您浏览 JMX 服务器，以及它们正在监控的缺陷。它还标识红帽流程的实例。

  JMX Navigator 视图驱动 Fuse 集成 视角中的所有监控和测试活动。它决定了 图表 视图、属性 视图和 Messages View 中显示的路由。它还提供用于激活路由追踪、添加和删除 JMS 目的地的菜单命令，以及启动和暂停路由。这也是将消息拖放到路由的目标。

  默认情况下，JMX Navigator 视图显示您本地机器上运行的所有 Java 进程。您可以根据需要添加 JMX 服务器来查看其他机器上的基础架构。

- 图表视图

  显示代表 JMX Navigator 视图中选择的节点的图形树。当您选择进程、服务器、端点或其他节点时，Diagram View 以根身份显示所选节点，并分支到其子级和 grandchildren。

  当您选择代理时，Diagram View 会显示最多三个子项： connection、topics 和 queues。它还显示配置的连接和目的地为 grandchildren。

  当您选择路由时，Diagram View 会显示路由中的所有节点，并显示消息可以通过路由的不同路径。它还显示启用路由追踪时路由中每个处理步骤的时间指标。

- 消息视图

  列出在启用路由追踪时通过所选 JMS 目的地或 Apache Camel 端点传递的消息。

  在 JMX Navigator 视图中选择了 JMS 目的地时，该视图将列出位于目的地的所有消息。

  启用路由追踪后，Messages View 列出了自追踪开始后通过路由中的节点传递的所有消息。您可以将 Messages View 配置为仅显示您感兴趣的数据和您首选的序列。

  当选择了 Messages View 的消息追踪时，其详情（消息正文和所有消息标头）会出现在 Properties 视图中。在 Diagram View 中，与所选消息 trace 关联的路由中的步骤被高亮显示。

- 服务器 视图

  显示由工具管理的服务器列表。它显示它们的运行时状态，并为添加、启动和停止它们以及向它们发布项目提供控制。

- Terminal 视图

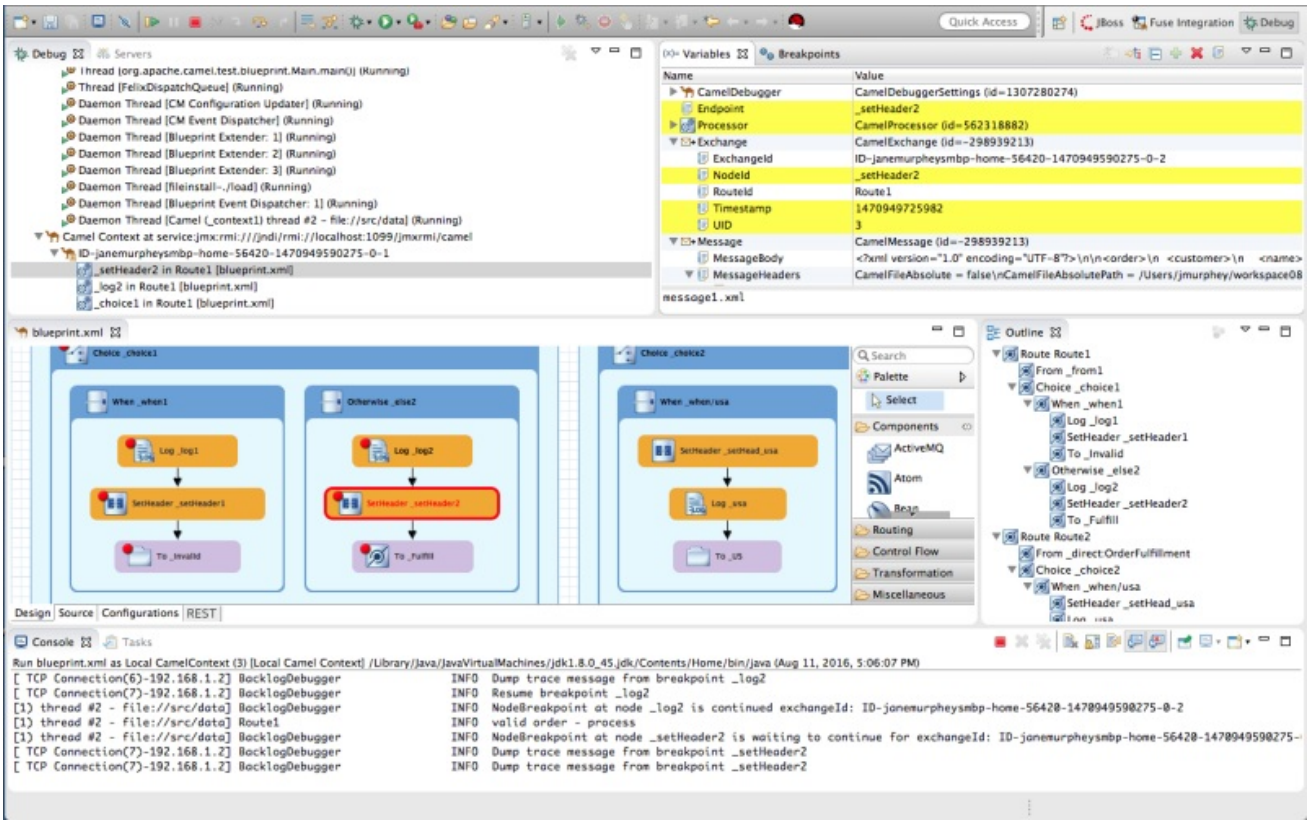
  显示连接的容器的命令控制台。您可以通过在 Terminal 视图中输入命令来控制容器。

- **控制台** 视图

  显示最近执行的操作的控制台输出。

附录 B. 调试视角

使用 Debug 视角来监控和调试正在运行的 Camel 上下文。



- 调试 视图

  对于正在运行的 Camel 上下文，Debug 视图会显示 debug 堆栈。

  您可以在同一消息流中切换断点，在 位于 service:jmx:rmi://jndi/rmi://localhost:1099/jmxrmi/camel 条目下列出，以检查和比较 Variables 视图中的变量值。

  消息流通过其唯一面包屑图标 ID 标识，每个后续消息流的面包屑导航栏 ID 则以 2 递增。例如，如果第一个消息流的面包 ID 是 ID-janemurpheysmbp-home-54620-1470949590275-0-1，则第二个消息流的面包屑 ID将是 ID-janemurpheysmbp-home-54620-1470949590-0-3。

- 变量 视图

  对于设置了断点的路由上下文中的每个节点，在断点点时 Variables 视图会显示可用变量的值。由于前面的断点被突出显示，因此值值更改的每个变量都会以黄色突出显示。

您可以更改可编辑变量的值，以检查此类更改是否产生预期的结果，并测试路由上下文的稳健性。

您还可以将变量添加到 watch 列表中，以便您可以快速轻松地查看它们的值是否如消息流的预期点更改。

- **breakpoints** 视图

  显示路由上下文中设置的断点列表，并显示它们是 enabled 或 disabled。您可以通过检查（启用）或取消选中（禁用）它们来启用和禁用单个断点。这可让您临时专注于路由上下文中的节点。

  
  按钮跳过禁用的断点，以跳到路由上下文中的下一个活跃断点。相反，
  
  按钮会在路由上下文中跳到执行下一个节点，而不考虑断点。

- **Camel Context.xml view**

  在图形模式中显示正在运行的路由上下文文件。对于使用断点设置的节点，它会显示设定的断点类型以及是否启用或禁用断点。当达到断点时，在 canvas 中它对应的节点会在红色列出。

  要检查节点的配置，请打开 Properties 视图，然后在 camel Context.xml 中选择 canvas 中的节点。

- **控制台** 视图

  显示 Camel debugger 生成的日志输出，因为它执行路由上下文。

- **属性** 视图

  在 CamelContext.xml 中显示在 canvas 中所选节点的属性。