# Lab: Reusing Mediation Sequences

**Training Objective**

Identify how Templates can manage Sequences better.

**High-level Steps**

* Create a Sequence Template.
* Update the REST API with the Call Template mediator to pass the parameters.
* Test your configuration.

**Detailed Instructions**

# Reusing Mediation Sequences

## **What you'll build**

In this sample scenario, you will use a **Sequence Template** and reuse it in multiple places of the medation flow. You can reuse the mediation flow that was defined in the [Service Orchestration](https://ei.docs.wso2.com/en/latest/micro-integrator/use-cases/tutorials/exposing-several-services-as-a-single-service) tutorial and then replace its sections with the sequence template. See [Creating Templates](https://ei.docs.wso2.com/en/latest/micro-integrator/develop/creating-artifacts/creating-sequence-templates/) for details on how to work with templates using WSO2 Integration Studio.

## **Let's get started!**

### **Step 1: Set up the workspace**

Set up WSO2 Integration Studio as follows:

1. Download the relevant [WSO2 Integration Studio](https://wso2.com/integration/tooling/) based on your operating system.
2. Set up the project from the [Service Orchestration](https://ei.docs.wso2.com/en/latest/micro-integrator/use-cases/tutorials/exposing-several-services-as-a-single-service) tutorial:

**Note**

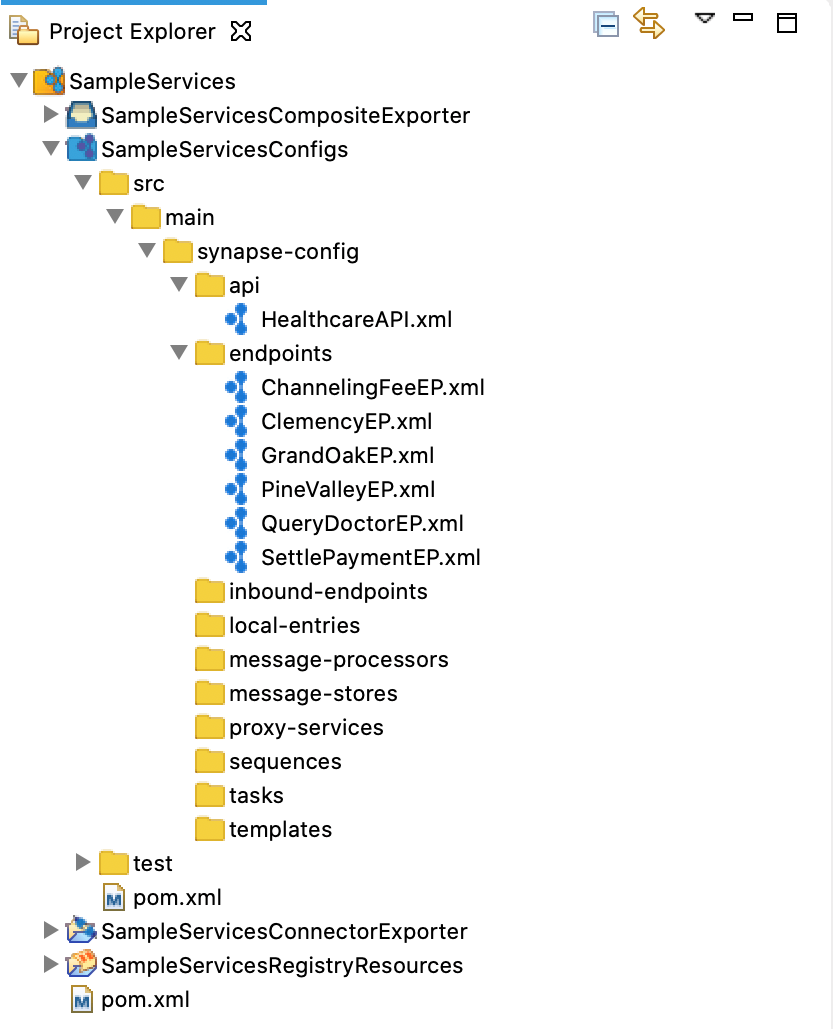
This tutorial is a continuation of the [Service Orchestration](https://ei.docs.wso2.com/en/latest/micro-integrator/use-cases/tutorials/exposing-several-services-as-a-single-service) tutorial.

* 1. Download the [pre-packaged project](https://github.com/wso2-docs/WSO2_EI/blob/master/Integration-Tutorial-Artifacts/Integration-Tutorial-Artifacts-EI7.1.0/service-orchestration-tutorial.zip).
  2. Open WSO2 Integration Studio and go to **File -> Import**.
  3. Select **Existing WSO2 Projects into workspace** under the **WSO2** category, click **Next**, and then upload the **prepackaged project**.

### **Step 2: Develop the integration artifacts**

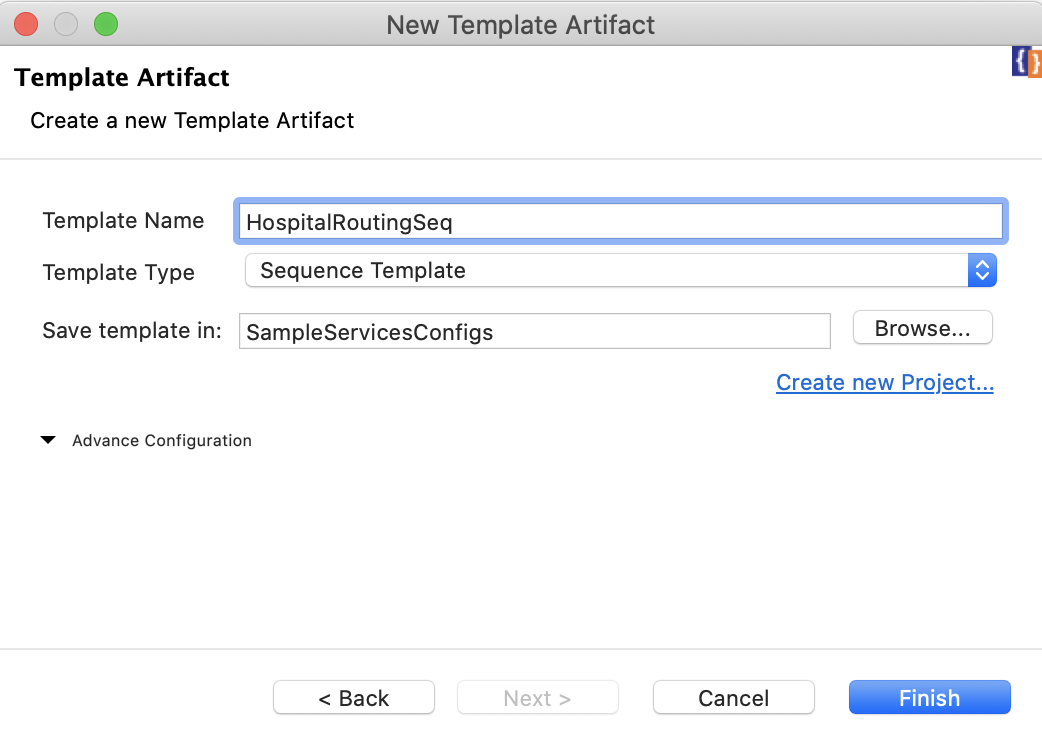
#### Create a Sequence Template

1. Once you have exported the integration project as described in above, the project directory will appear with the artifacts as shown below. Note the 'HealthcareAPI' that is already included.

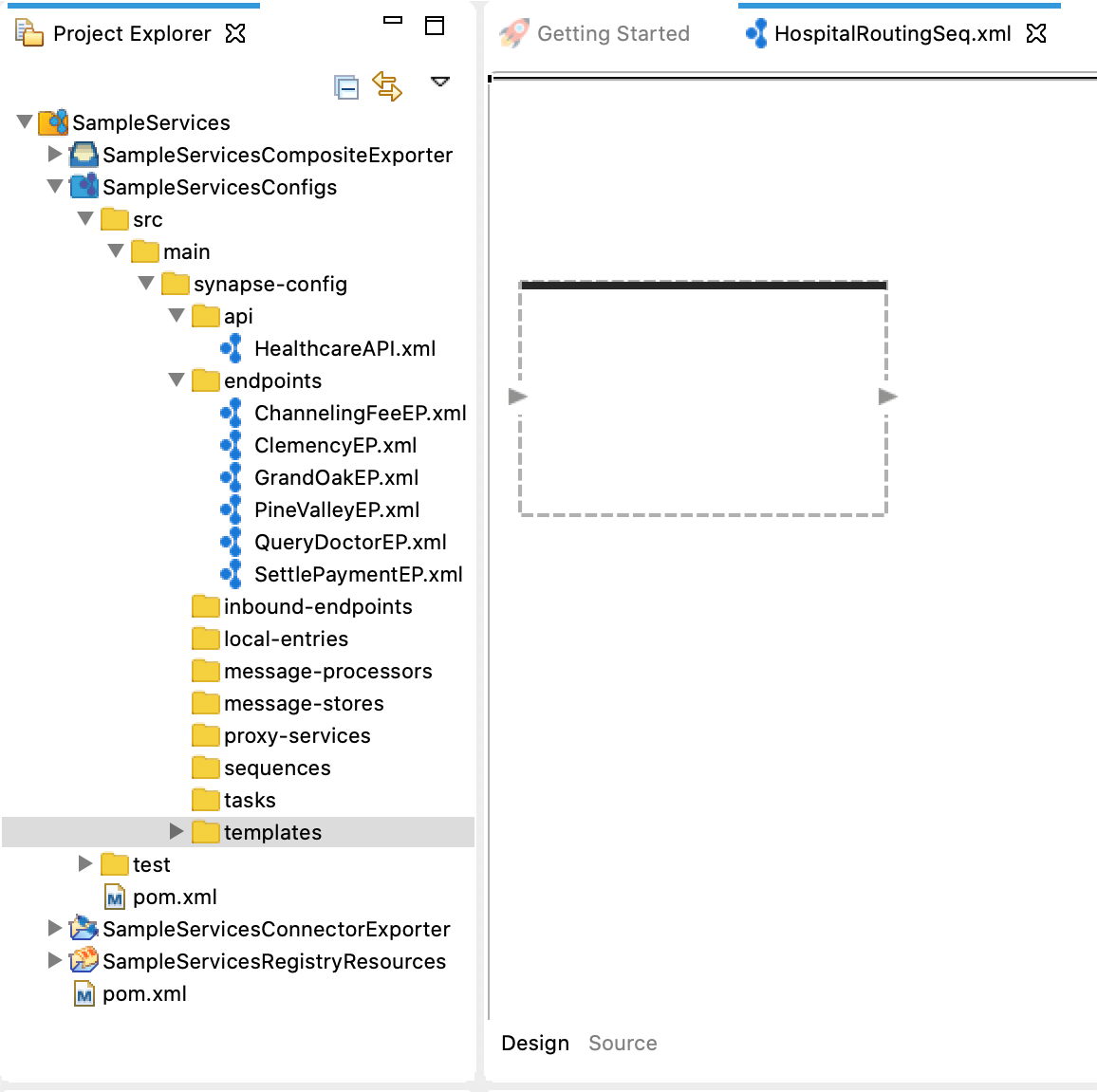


1. Right-click on **SampleServicesConfigs** and navigate to **New -> Template** . The **New Template Artifact** dialog box will open.
2. Select the **Create a New Template** and click **Next**.
3. Enter the following details and click **Finish**.

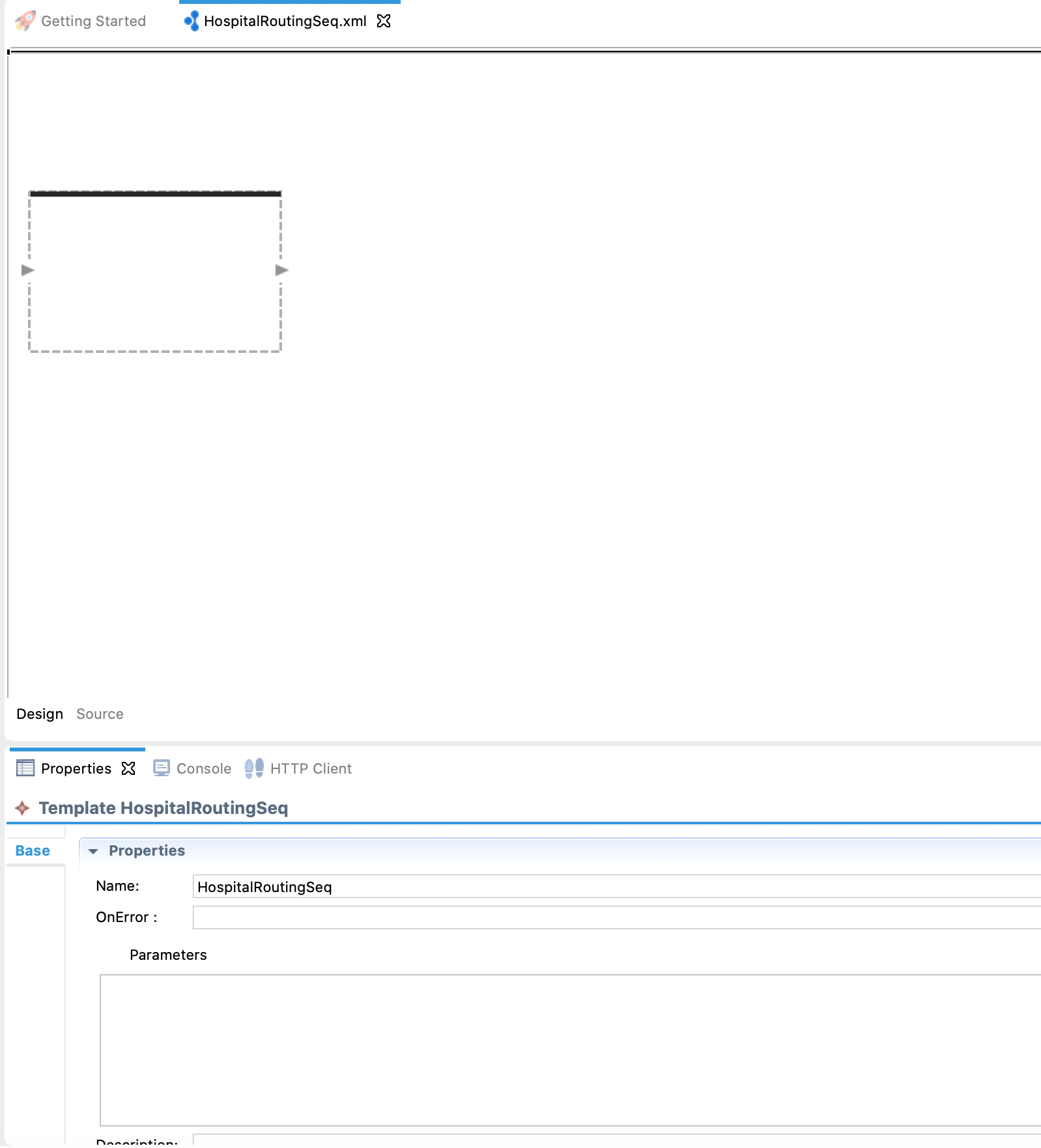
|  |  |
| --- | --- |
| Parameter | Description |
| Template Name | HospitalRoutingSeq |
| Template Type | Sequence Template |



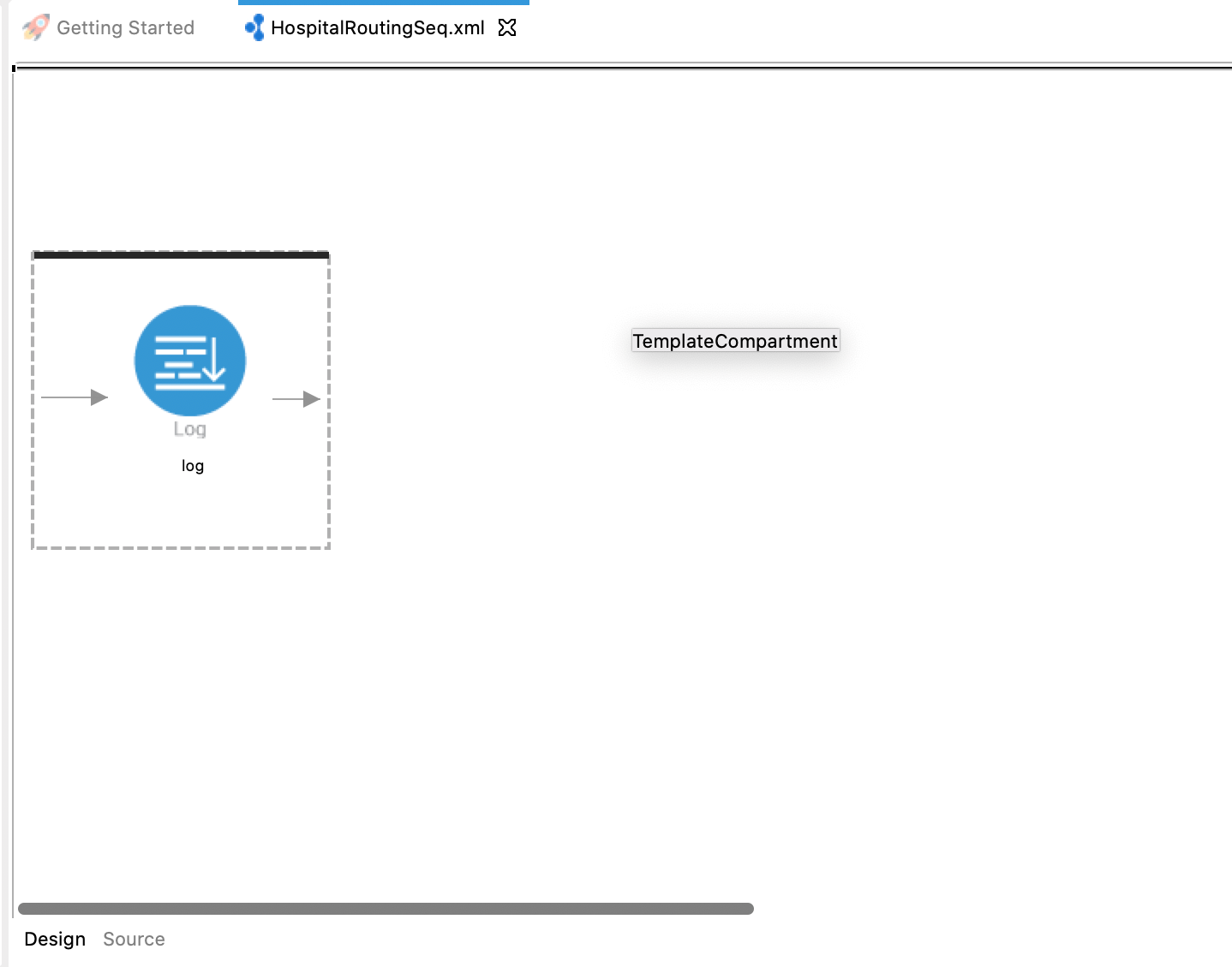
1. The template artifact will open in the canvas as shown below.



1. Open the **Properties** tab of the sequence template by clicking on the canvas (outside the sequence box).
2. Click the https://ei.docs.wso2.com/en/latest/micro-integrator/assets/img/tutorials/plus-icon.png icon to start adding parameters .



1. In the **Template Parameter** dialog box that opens, enter 'sethospital' as the parameter name and click **Finish**.
2. Add a **Log** mediator to the sequence template as shown below. This will print a message indicating to which hospital a requested message is routed.



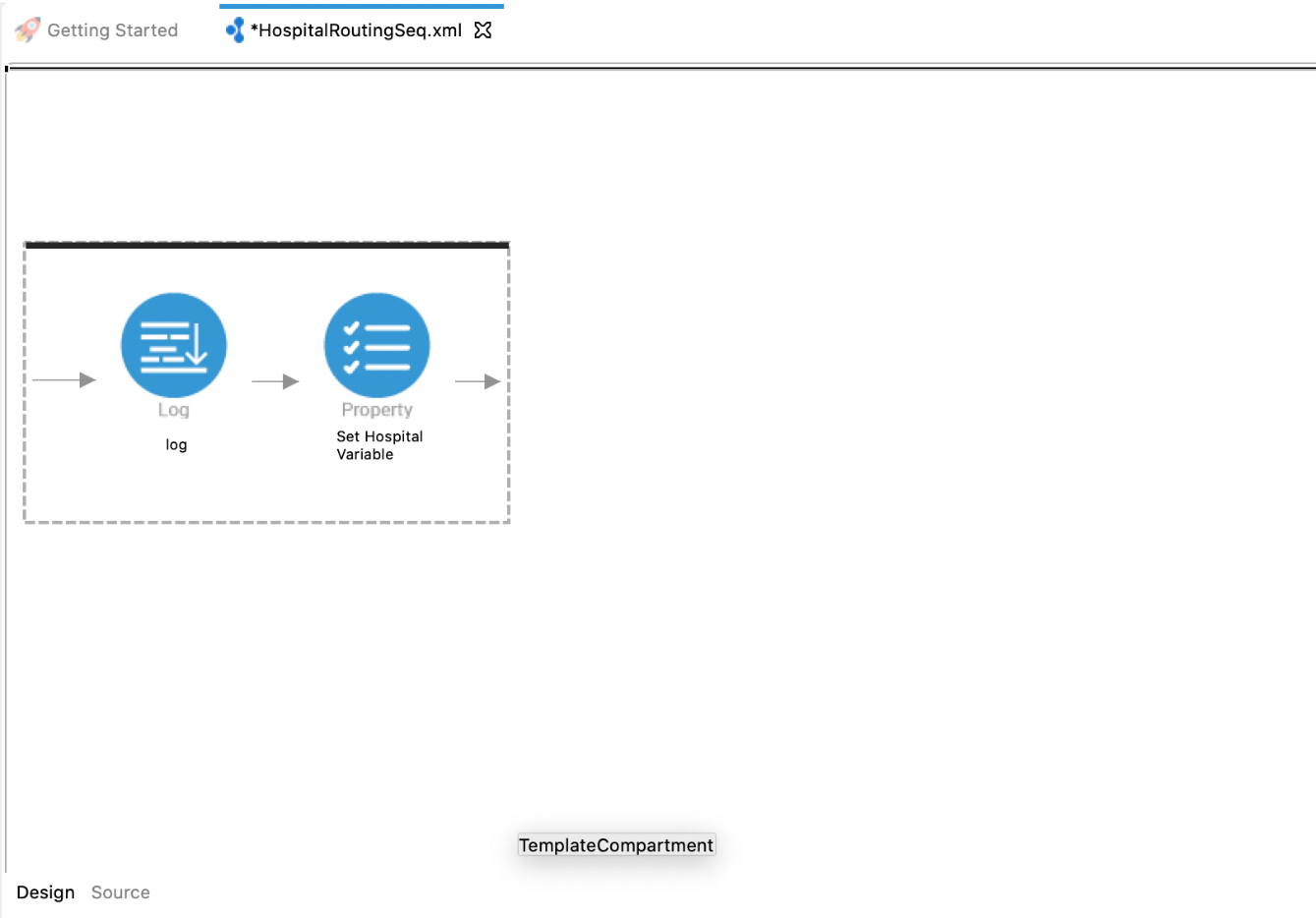
1. Open the **Properties** tab of the log mediator and specify the following:

|  |  |
| --- | --- |
| Property | Description |
| Log Category | INFO |
| Log Level | CUSTOM |

1. Click the https://ei.docs.wso2.com/en/latest/micro-integrator/assets/img/tutorials/plus-icon.png icon to start defining a property. Then add the following details for the property:

|  |  |
| --- | --- |
| Property Name | Description |
| Name | message |
| Type | EXPRESSION |
| Property Expression | fn:concat('Routing to ', get-property('Hospital')) |

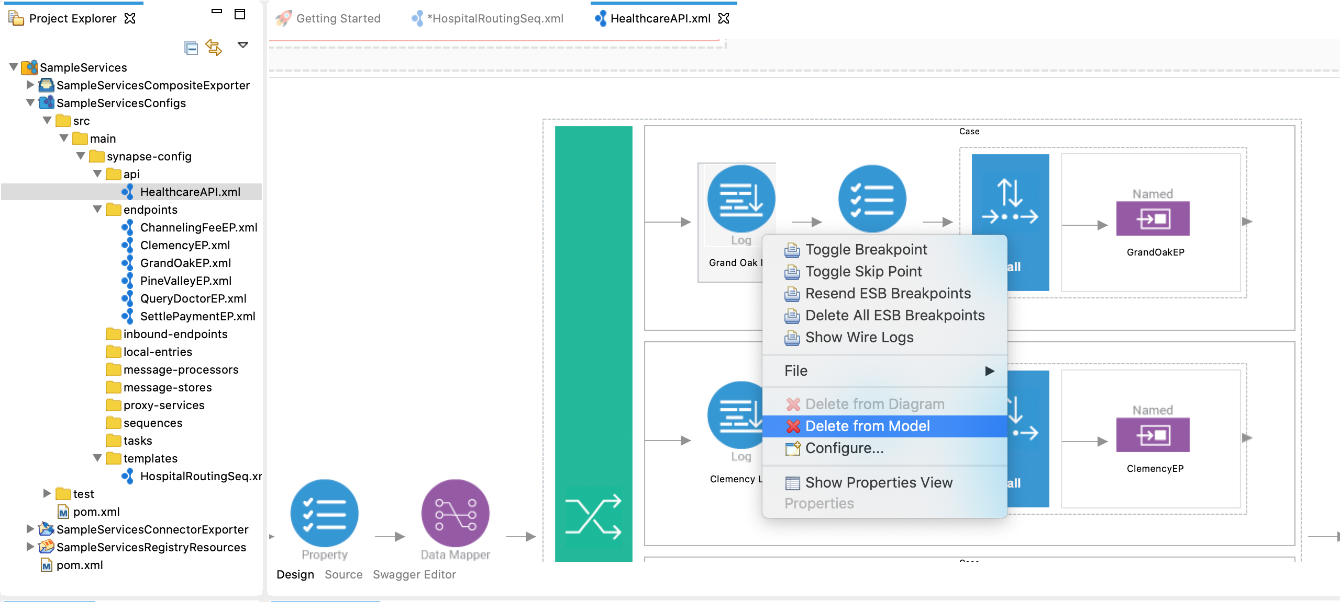
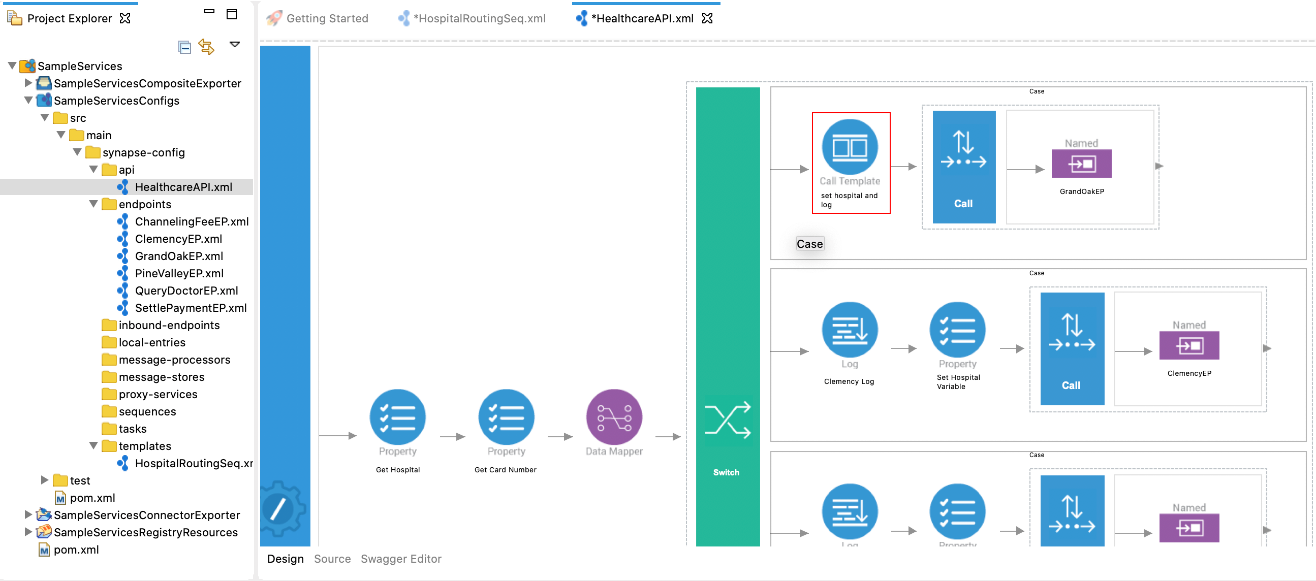
1. We select EXPRESSION because the required properties for the log message must be extracted from the request, which we can do using an XPath expression.
2. Add a **Property** mediator just after the **Log** mediator to store the value for uri.var.hospital.



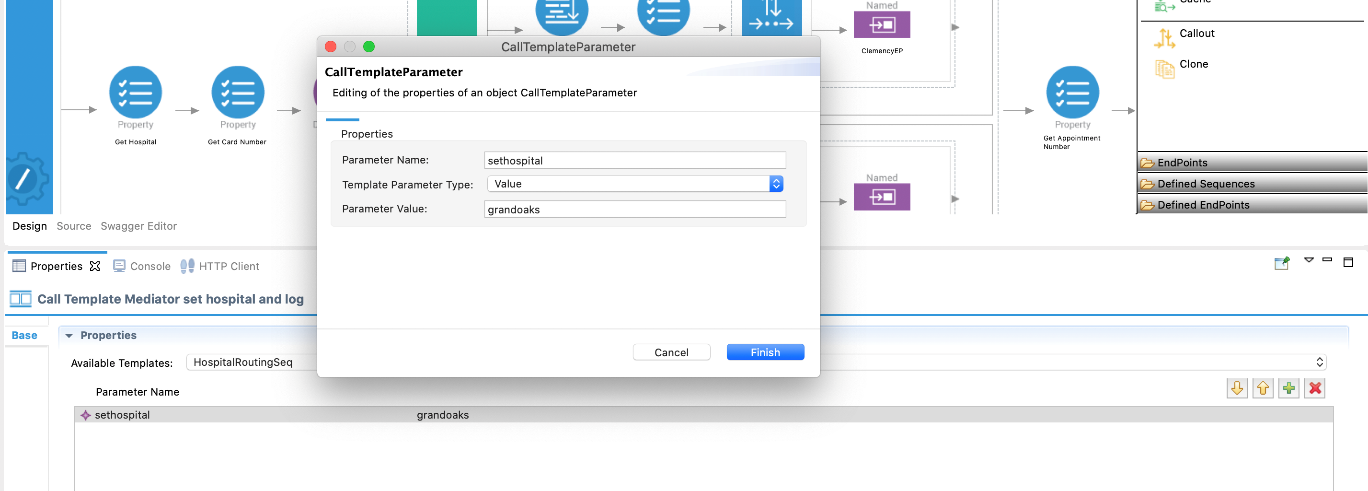
1. With the **Property** mediator selected, access the **Properties** tab and enter the information given below:

|  |  |
| --- | --- |
| Property | Description |
| Property Name | Select New Property |
| New Property Name | uri.var.hospital |
| URI Template | Select set |
| Property Data Type | Select STRING |
| Value | Click on the **Ex** button in front of the label value and add this $func:sethospital as the expression. |
| Description | Set Hospital Variable |

#### Update the mediation flow

1. Open the design view of the HealthcareAPI.xml and delete 'GrandOak' **Log** mediator by right clicking the mediator and selecting **Delete from Model** .  
   
2. Delete the 'Set Hospital Variable' **Property** mediator.
3. Add a **Call Template** mediator to the sequence as shown below.  
   
4. Open the **Properties** tab of the **Call Template** mediator and select ' HospitalRoutingSeq' from the list of available templates.
5. Click the https://ei.docs.wso2.com/en/latest/micro-integrator/assets/img/tutorials/plus-icon.png icon to start adding parameters. Enter the following parameter details and click **Finish** .

|  |  |
| --- | --- |
| Parameter | Description |
| Parameter Name | sethospital |
| Parameter Type | value |
| Value/Expression | grandoaks |



1. Repeat the above steps to add **Call Templates** for 'Clemency' and 'Pine Valley' hospitals. Add **clemency** and **pinevalley** as the respective parameter values.
2. Save the configuration.

After completion, your API will be similar to this.

<?xml version="1.0" encoding="UTF-8"?>

<**api** context="/healthcare" name="HealthcareAPI" xmlns="http://ws.apache.org/ns/synapse">

<**resource** methods="GET" uri-template="/querydoctor/{category}">

<**inSequence**>

<**log** description="Request Log" level="custom">

<**property** name="Log Property message" value="&quot;Welcome to HealthcareService&quot;"/>

</**log**>

<**send**>

<**endpoint** key="QueryDoctorEP"/>

</**send**>

</**inSequence**>

<**outSequence**>

<**send**/>

</**outSequence**>

<**faultSequence**/>

</**resource**>

<**resource** methods="POST" uri-template="/categories/{category}/reserve">

<**inSequence**>

<**property** description="Get Hospital" expression="json-eval($.hospital)" name="Hospital" scope="default" type="STRING"/>

<**property** description="Get Card Number" expression="json-eval($.cardNo)" name="card\_number" scope="default" type="STRING"/>

<**datamapper** config="gov:datamapper/RequestMapping.dmc" inputSchema="gov:datamapper/RequestMapping\_inputSchema.json" inputType="JSON" outputSchema="gov:datamapper/RequestMapping\_outputSchema.json" outputType="JSON" xsltStyleSheet="gov:datamapper/RequestMapping\_xsltStyleSheet.xml"/>

<**switch** source="get-property('Hospital')">

<**case** regex="grand oak community hospital">

<**call-template** target="HospitalRoutingSeq">

<**with-param** name="sethospital" value="grandoaks"/>

</**call-template**>

<**call**>

<**endpoint** key="GrandOakEP"/>

</**call**>

</**case**>

<**case** regex="clemency medical center">

<**call-template** target="HospitalRoutingSeq">

<**with-param** name="sethospital" value="Clemency"/>

</**call-template**>

<**call**>

<**endpoint** key="ClemencyEP"/>

</**call**>

</**case**>

<**case** regex="pine valley community hospital">

<**call-template** target="HospitalRoutingSeq">

<**with-param** name="sethospital" value="Pine Valley"/>

</**call-template**>

<**call**>

<**endpoint** key="PineValleyEP"/>

</**call**>

</**case**>

<**default**>

<**log** description="Fault Log" level="custom">

<**property** expression="fn:concat('Invalid hospital - ', get-property('Hospital'))" name="message"/>

</**log**>

<**respond**/>

</**default**>

</**switch**>

<**property** description="Get Appointment Number" expression="json-eval($.appointmentNumber)" name="uri.var.appointment\_id" scope="default" type="STRING"/>

<**property** description="Get Doctor Details" expression="json-eval($.doctor)" name="doctor\_details" scope="default" type="STRING"/>

<**property** description="Get Patient Details" expression="json-eval($.patient)" name="patient\_details" scope="default" type="STRING"/>

<**call**>

<**endpoint** key="ChannelingFeeEP"/>

</**call**>

<**property** description="Get Actual Fee" expression="json-eval($.actualFee)" name="actual\_fee" scope="default" type="STRING"/>

<**payloadFactory** media-type="json">

<**format**>{"appointmentNumber":$1, "doctor":$2, "patient":$3, "fee":$4, "confirmed":"false", "card\_number":"$5"}</**format**>

<**args**>

<**arg** evaluator="xml" expression="$ctx:uri.var.appointment\_id"/>

<**arg** evaluator="xml" expression="$ctx:doctor\_details"/>

<**arg** evaluator="xml" expression="$ctx:patient\_details"/>

<**arg** evaluator="xml" expression="$ctx:actual\_fee"/>

<**arg** evaluator="xml" expression="$ctx:card\_number"/>

</**args**>

</**payloadFactory**>

<**call**>

<**endpoint** key="SettlePaymentEP"/>

</**call**>

<**respond**/>

</**inSequence**>

<**outSequence**/>

<**faultSequence**/>

</**resource**>

</**api**>

### **Step 3: Package the artifacts**

Package the artifacts in your composite exporter module (SampleServicesCompositeExporter) to be able to deploy the artifacts in the server.

1. Open the pom.xml file in the composite exporter module.
2. Ensure that the following artifacts are selected in the POM file.
   * HealthcareAPI
   * HospitalRoutingSeq
3. Save the changes.

### **Step 4: Build and run the artifacts**

To test the artifacts, deploy the [packaged artifacts](https://ei.docs.wso2.com/en/latest/micro-integrator/use-cases/tutorials/using-templates/#step-3-package-the-artifacts) in the embedded Micro Integrator:

1. Right-click the composite exporter module and click **Export Project Artifacts and Run**.
2. In the dialog box that opens, confirm that the required artifacts from the composite exporter module are selected.
3. Click **Finish**.

The artifacts will be deployed in the embedded Micro Integrator and the server will start.

* See the startup log in the **Console** tab.
* See the URLs of the deployed services and APIs in the **Runtime Services** tab.

### **Step 5: Testing the use case**

Let's test the use case by sending a simple client request that invokes the service.

#### Start the back-end service

1. Download the JAR file of the back-end service from [here](https://github.com/wso2-docs/WSO2_EI/blob/master/Back-End-Service/Hospital-Service-JDK11-2.0.0.jar).
2. Open a terminal, navigate to the location where your saved the back-end service.
3. Execute the following command to start the service:

java -jar Hospital-Service-JDK11-2.0.0.jar

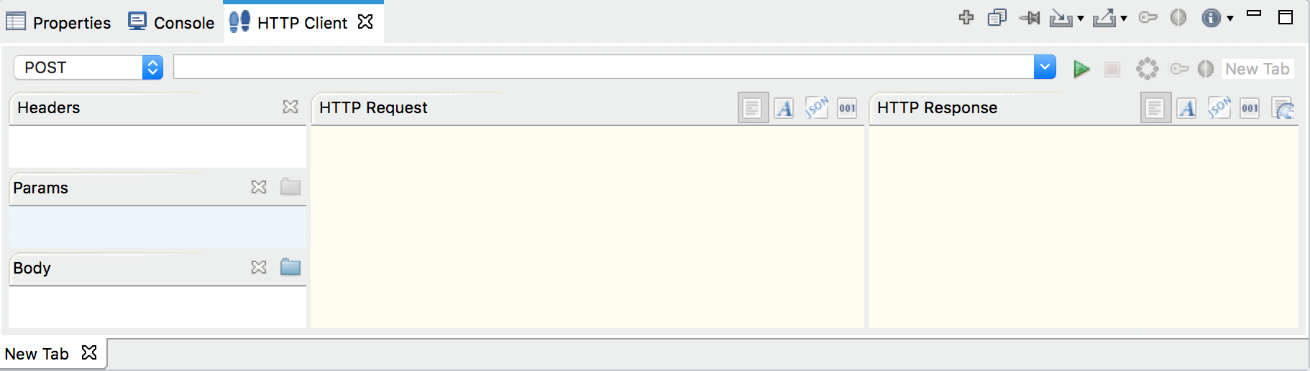
#### Send the client request

Let's send a simple request to invoke the service. You can use the embedded **HTTP Client** of WSO2 Integration Studio as follows:

1. Open the **HTTP Client** of WSO2 Integration Studio.

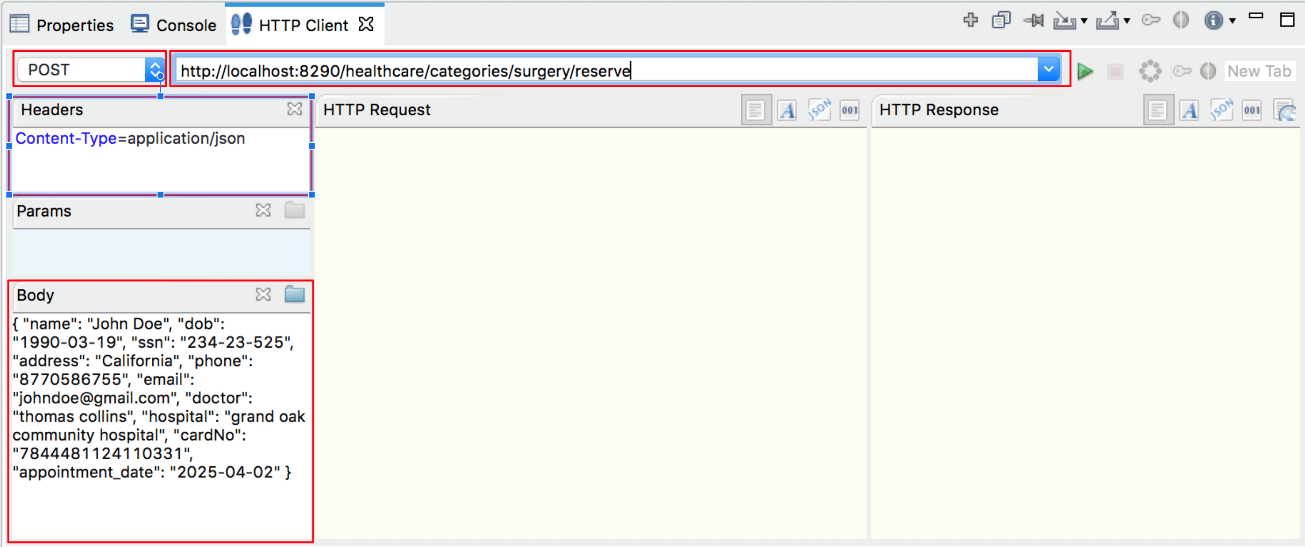
**Tip**

If you don't see the **HTTP Client** pane, go to **Window -> Show View - Other** and select **HTTP Client** to enable the client pane.



1. Enter the request information as given below and click the **Send** icon (https://ei.docs.wso2.com/en/latest/micro-integrator/assets/img/tutorials/common/play-head-icon.png).

|  |  |
| --- | --- |
| Method | POST |
| Headers | Content-Type=application/json |
| URL | http://localhost:8290/healthcare/categories/surgery/reserve |
| Body | { "name": "John Doe", "dob": "1940-03-19", "ssn": "234-23-525", "address": "California", "phone": "8770586755", "email": "johndoe@gmail.com", "doctor": "thomas collins", "hospital": "grand oak community hospital", "cardNo": "7844481124110331", "appointment\_date": "2025-04-02" }     * + This JSON payload contains details of the appointment reservation, which includes patient details, doctor, hospital, and data of appointment. |



If you want to send the client request from your terminal:

1. Install and set up [cURL](https://curl.haxx.se/) as your REST client.
2. Create a JSON file names request.json with the following request payload.

{

"name": "John Doe",

"dob": "1940-03-19",

"ssn": "234-23-525",

"address": "California",

"phone": "8770586755",

"email": "johndoe@gmail.com",

"doctor": "thomas collins",

"hospital": "grand oak community hospital",

"cardNo": "7844481124110331",

"appointment\_date": "2025-04-02"

}

1. Open a command line terminal and execute the following command from the location where the request.json file you created is saved:

curl -v -X POST --data @request.json http://localhost:8290/healthcare/categories/surgery/reserve --header "Content-Type:application/json"

#### Analyze the response

You will see the response received to your **HTTP Client**:

{"patient":"John Doe","actualFee":7000.0, "discount":20, "discounted":5600.0,"paymentID":”08c784c8-db8b-4539-9e1c-892356b739a9”, "status":”Settled”}