Backend Training Steps

Localhost :9999/hawtio/camel/routes

PROTECT SERVICES

1. Edit the portalserver/src/main/resources/META-INF/spring/backbase-portal-presentation-security.xml file.
2. And add the following interceptor in the default <http> Spring Security element:

<intercept-url pattern="/services/\*\*" access="hasAnyRole('role\_admin', 'role\_manager', 'role\_user')"/>

EXERCISE PROPERTIES

configuration/src/main/resources/backbase.properties

#Training Server Host  
training.server.host=${training.server.host}  
training.server.http.port=${training.server.http.port}  
training.server.mq.port=${training.server.mq.port}

configuration/src/main/filters/local.properties

#Training Server Host  
training.server.host=localhost  
training.server.http.port=9999  
training.server.mq.port=61616

REBUILD CXP

Go to the root of your CXP project and execute:

mvn clean install

## INSTALLING TRAINING SERVER

<https://github.com/Backbase/training-server>

## INSTALL JOLOKIA

1. Jolokia is a JMX-HTTP bridge
2. Download the WAR-Agent from:
   1. <https://jolokia.org/download.html>
3. Copy the war file to the webapps folder
4. Edit the webapps/pom.xml file, including the following content (in red). Make sure you are using the correct filename (in this example, jolokia-war-1.3.5.war):

|  |
| --- |
| Sample webapp/pom.xml  ...     </webApp>     <!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->     <!-- NOTE: this must be outside the <webApp> tag -->     <!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->     <contextHandlers>          <contextHandler implementation="org.eclipse.jetty.maven.plugin.JettyWebAppContext">              <war>${project.basedir}/../jolokia-war-1.3.5.war</war>              <contextPath>/jolokia</contextPath>          </contextHandler>     </contextHandlers>  </configuration>  <dependencies>  ... |

1. Start the portalserver, contentservices and orchestrator web servers
2. Use Hawtio
   1. To use the hawtio just access the URL:
      1. <http://localhost:9999/hawtio/>
   2. Click on Connect tab
      1. Connecting to the Portal Server
         1. Name: Portal Server
         2. Scheme: http
         3. Host: localhost
         4. Port: 7777
         5. Path: jolokia
         6. User name: <leave it blank>
         7. Password: <leave it blank>
         8. Use proxy: <does not matter - using localhost>
         9. Click on Save
            1. It will become available in the Saved Connections for future use
         10. Click on Connect to remote server
      2. Connecting to the Content Services or Orchestrator
         1. Name: <an appropriate name here>
         2. Port: 8082 (Content Services) or 8083 (Orchestrator)
         3. The remaining fields are exactly the same
   3. Now you can just start using the hawtio connected to the selected server

## INSTALLING THE TRAINING WIDGETS

<https://github.com/Backbase/training-be-module-01/tree/master/training-collection>

## STOCK PRICE QUERY

### CREATE A SERVICE USING ARCHETYPE

1. We will create a simple service to return the stock price
   1. Service to consume. Some samples:
      1. Yahoo: <https://www.google.com/finance/info?q=YHOO>
      2. Apple: <https://www.google.com/finance/info?q=AAPL>
      3. Google: <https://www.google.com/finance/info?q=GOOGL>
      4. Amazon: <https://www.google.com/finance/info?q=AMZN>
      5. Microsoft: <https://www.google.com/finance/info?q=MSFT>
2. Reference:
   1. Visit:
      1. <http://my.backbase.com>
   2. Select:
      1. Docs -> Product Documentation -> CXP Documentation
   3. And then:
      1. Development Guide -> Integration Services -> How To Create an Integration Service With Camel Spring XML DSL -> Generate Project
3. Steps:
   1. Go to the services directory:
      1. cd services
   2. Generate the service from the archetype:
      1. Execute the command:
         1. mvn archetype:generate -DarchetypeArtifactId=integration-service-spring-archetype -DarchetypeGroupId=com.backbase.portal.archetypes -DarchetypeVersion=5.6.2.0
      2. And answer the following:
         1. groupId = com.backbase.training
         2. artifactId = stock-price-service
         3. version = (leave the default value)
         4. package = (leave the default value)
      3. Open the services/pom.xml file
         1. Check if stock-price-service is included as a <module>
   3. Open the generated Camel XML file:
      1. Location:
         1. services/stock-price-service/src/main/resources/META-INF/spring/backbase-integration-service.xml
      2. Add the following route:

   <camel:routeContext id="com.backbase.portal.integration.service.StockPriceService"

       xmlns="http://camel.apache.org/schema/spring">

       <route id="com.backbase.training.StockPriceRoute">

           <from uri="restlet:/stock/{code}/price?restletMethods=GET" />

           <log message="Code received: ${in.header.code}" loggingLevel="DEBUG" />

           <removeHeaders pattern="CamelHttp\*" excludePattern="CamelHttpMethod"/>

           <setHeader headerName="CamelHttpQuery">

               <simple>q=${in.header.code}</simple>

           </setHeader>

           <to uri="https://www.google.com/finance/info?throwExceptionOnFailure=false"/>

           <choice>

               <when>

                   <simple>${header.CamelHttpResponseCode} == 200</simple>

                   <convertBodyTo type="String"/>

                   <transform>

                       <simple>${body.replaceAll("//", "")}</simple>

                   </transform>

                   <unmarshal>

                       <json library="Jackson" unmarshalTypeName="java.util.List"/>

                   </unmarshal>

                   <log message="body: ${body}" loggingLevel="DEBUG"/>

                   <setHeader headerName="price">

                       <simple>${body.get(0)[l]}</simple>

                   </setHeader>

               </when>

               <when>

                   <simple>${header.CamelHttpResponseCode} == 400</simple>

                   <setHeader headerName="CamelHttpResponseCode">

                       <constant>200</constant>

                   </setHeader>

                   <setHeader headerName="price">

                       <constant>null</constant>

                   </setHeader>

               </when>

           </choice>

           <setBody>

               <simple>{"code" : "${header.code}", "price" : ${header.price}}</simple>

           </setBody>

           <log id="com.backbase.training.content.log.body" message="body: ${body}" loggingLevel="DEBUG"/>

       </route>

   </camel:routeContext>

1. Add camel-jackson dependency
   1. Open services/stock-price-service/pom.xml
   2. Add the following <dependency>

<dependency>  
   <groupId>org.apache.camel</groupId>  
   <artifactId>camel-jackson</artifactId>  
   <version>2.12.4</version>  
</dependency>

1. Include dependency in webapps/portalserver/pom.xml

       <dependency>

           <groupId>com.backbase.training</groupId>

           <artifactId>stock-price-service</artifactId>

           <version>1.0-SNAPSHOT</version>

       </dependency>

1. Restart Portal
   1. Build configuration, services, & portal
   2. Run portal
   3. Test the service:
      1. <http://localhost:7777/portalserver/services/rest/stock/YHOO/price>

### INSTALL THE STOCK PRICE QUERY WIDGET

1. Download the Stock Price Query Widget:
   1. <https://drive.google.com/file/d/0B7dzciN5c-MwbVJnc1hVVWpIR3c/view?usp=sharing>
2. Go to the Enterprise Catalog
3. Import the stock-price-widget-v2.zip file
4. Import it to the Portal Catalog
5. Use it in some page based on Launchpad
6. Test the widget