

Testing Services Over JMS

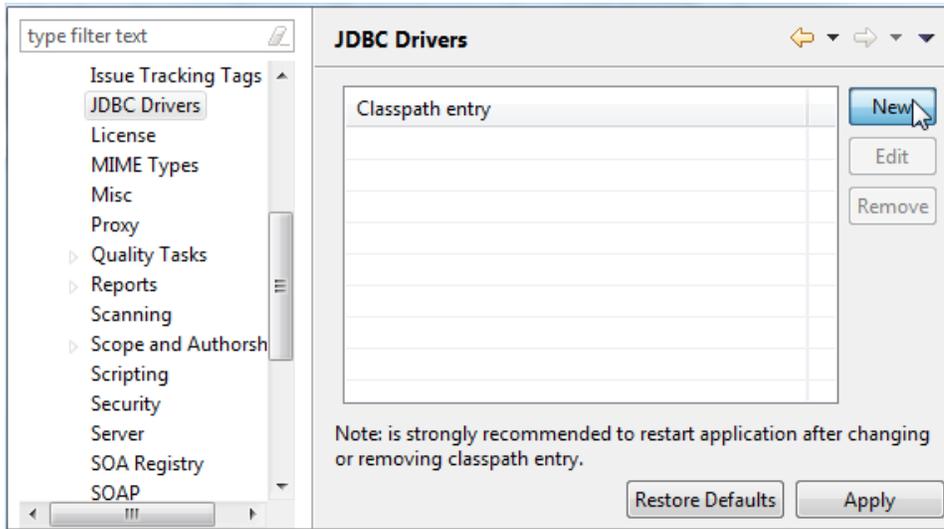
Parasoft's solution provides extensive support for generating and consuming JMS messages, and simulating various patterns-including point-to-point and publish-and-subscribe patterns. This allows for end-to-end testing and validation of messaging systems. These same scenarios can be scaled into load tests.

The exercises in this lesson will use the ParaBank application introduced in [Setting Up ParaBank](#).

Preparation: Running ParaBank

To configure SOAtest for the ParaBank tests:

1. Add the JDBC driver to the SOAtest JDBC Driver preference page as follows:
 - a. Choose **Parasoft > Preferences**, then open **Parasoft > JDBC Drivers**.
 - b. Click **New**.

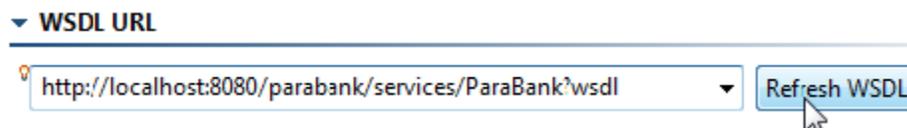


- c. Navigate to your ParaBank workspace, select `{PARABANK}/WebContent/WEB-INF/lib/hsqldb-<version>.jar` and click **Open**.
 - d. Click **OK** and apply the changes.
2. Add the ActiveMQ driver to the SOAtest System Properties preference page as follows:
 - a. Choose **Parasoft > System Properties** from the main menu
 - b. Click **Add JARs**.
 - c. Navigate to your ParaBank workspace, select `{PARABANK}/WebContent/WEB-INF/lib/activemq-client-<version>.jar`, then click **Open**.
 3. Ensure that the ParaBank Tomcat 8.5 Server is Started and Synchronized.

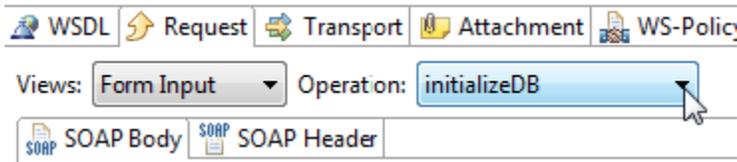
Testing JMS Services

To test JMS services provided by the ParaBank application:

1. Right click the main project you have been using in this tutorial, then choose **Add New > Test (.tst) File**.
2. Enter `LoanProcessor` in the File Name field, then click **Next**.
3. Select **Empty**, then click **Finish**.
4. Right click the **Loan Processor > Test Suite: Test Suite** node, then choose **Add New > Test**.
5. Add a SOAP Client tool as follows:
 - a. Right click the **Loan Processor > Test Suite: Test Suite** node, then choose **Add New > Test**.
 - b. Select **SOAPClient**, then click **Finish**.
 - c. In the SOAP Client editor that opens, rename the SOAP Client to `initializedDB`.
 - d. To to the **WSDL** tab and enter `http://localhost:8080/parabank/services/ParaBank?wsdl` into the **WSDLURL** field.
 - e. Click **RefreshWSDL** to propagate the WSDL to the other sections of the SOAP Client.



f. Go to the **Request** tab and ensure that **Operation** is set to **initializeDB**. This will reset the ParaBank database for this example.

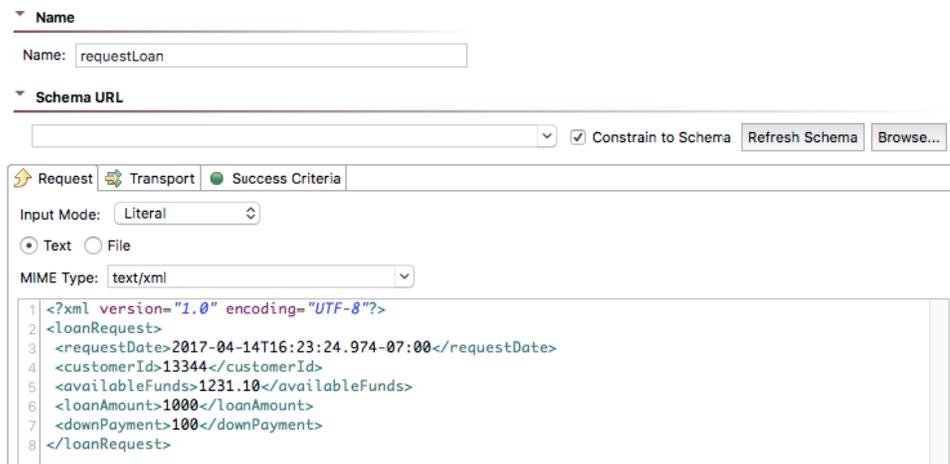


g. Save the initializeDB SOAP Client.

6. Add a Messaging Client tool as follows:

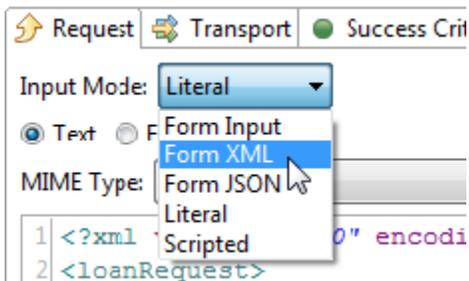
- a. Right click the **Loan Processor > Test Suite: Test Suite** node, then choose **Add New > Test**.
- b. Select **MessagingClient**, then click **Finish**.
- c. In the Messaging Client editor that opens, rename the Messaging Client to `requestLoan`.
- d. Click the **Request** tab and set **InputMode** to **Literal**.
- e. Replace the `<Placeholder/>` element with the following XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<loanRequest>
  <requestDate>2017-04-14T16:23:24.974-07:00</requestDate>
  <customerId>13344</customerId>
  <availableFunds>1231.10</availableFunds>
  <loanAmount>1000</loanAmount>
  <downPayment>100</downPayment>
</loanRequest>
```

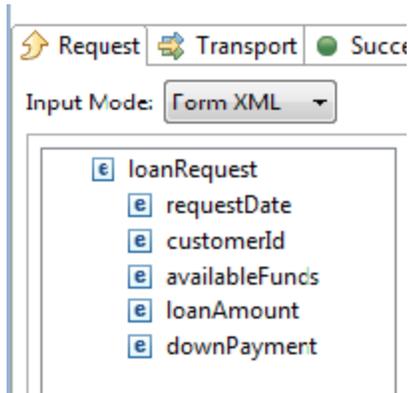


f. Save the Messaging Client.

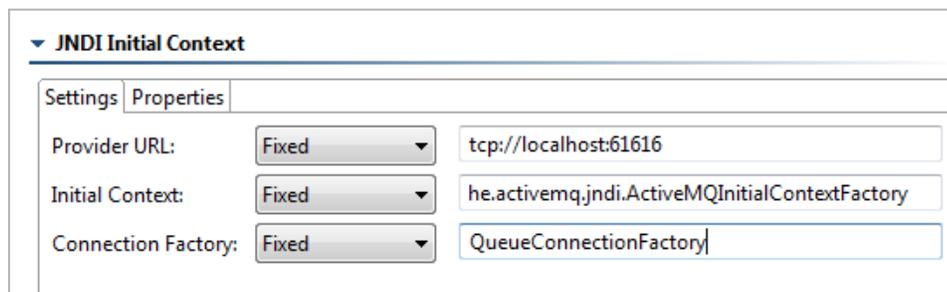
g. Switch the **InputMode** to **FormXML**.



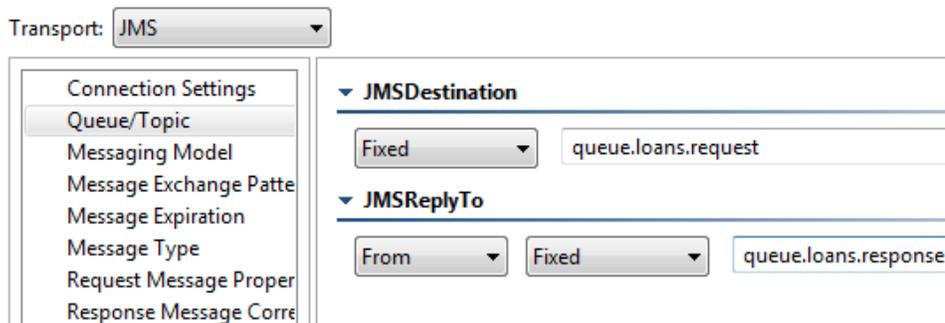
If prompted, click **Yes** to propagate the values. You will see the tree structure of the `loanRequest` element.



- h. Go to the **Transport** tab, set **Transport** to **JMS**, then enter the following:
- **ProviderURL:** tcp://localhost:61616
 - **InitialContext:** org.apache.activemq.jndi.ActiveMQInitialContextFactory
 - **Connection Factory:** QueueConnectionFactory

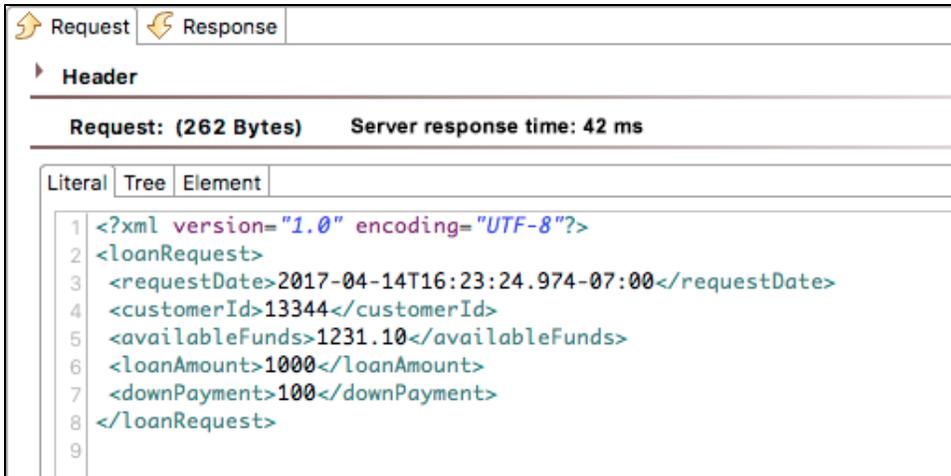


- i. Select **Queue/Topic** from the left panel, then enter `queue.loans.request` for the **JMSDestination** and `queue.loans.response` for the **JMSReplyTo**.



- j. Save the requestLoan Messaging Client.
7. Run the test suite.
 8. Expand the **requestLoan** Messaging Client node, double click the Traffic Viewer.

9. Open the Traffic Viewer's **Response** tab and note that the <approved> element returns true.



The screenshot shows a web traffic viewer interface. At the top, there are two tabs: "Request" and "Response", with "Response" being the active tab. Below the tabs, there is a "Header" section. Underneath the header, it displays "Request: (262 Bytes)" and "Server response time: 42 ms". The main content area is divided into three columns: "Literal", "Tree", and "Element". The "Literal" column is selected, showing the following XML code:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <loanRequest>
3   <requestDate>2017-04-14T16:23:24.974-07:00</requestDate>
4   <customerId>13344</customerId>
5   <availableFunds>1231.10</availableFunds>
6   <loanAmount>1000</loanAmount>
7   <downPayment>100</downPayment>
8 </loanRequest>
9
```