Developing a plug-in connector for web service endpoints using WebSphere Cast Iron: Part 2: Implementing the TestConnection orchestration

Raghvendra Neelekani (raneelek@in.ibm.com)
Software Engineer
IBM

04 March 2015

This five-part tutorial series shows you how to use the WebSphere Cast Iron Connector Development Kit (CDK) to quickly and easily develop plug-in connectors for web service and ReST based applications. Part 2 covers design considerations when developing a plug-in connector, and shows you how to create a plug-in connector for a Salesforce.com application.

View more content in this series

Introduction

Part 1 of this tutorial series showed you how to use the Cast Iron CDK wizard to create and configure a new plug-in connector for Salesforce.com application, configure the connector properties, and generate activity orchestrations. Part 2 shows you how to implement the TestConnection orchestration and develop the TestConnection feature for the Connector Endpoint panel.

Implementing the TestConnection orchestration

The TestConnection orchestration is called when you click Test Connection on the Connector Endpoint panel. Therefore you need to implement the TestConnection logic in this orchestration to extract the connection fields, connect to the target application, and authenticate based on your design.

When you open the generated TestConnection orchestration in Cast Iron studio, you will see two preconfigured activities, as shown in Figure 1:
Developing a plug-in connector for web service endpoints using WebSphere Cast Iron: Part 2: Implementing the TestConnection orchestration

Figure 1. Generated TestConnection template orchestration

![Generated TestConnection template orchestration diagram]

- **Provide Service (TestConnection)** – The Provide Service activity is the entry point for TestConnection operation, and is called when you click **Test Connection** on the Connector Endpoint panel. From this activity, you can access all of the connection fields in the Map Outputs section, as shown in Figure 1. These connection fields are the ones you configured in the CDK wizard when you generated the orchestration.

- **Send Reply (Send Reply)** – The Send Reply activity sends the TestConnection result back to the connector.

To test a connection to Salesforce.com and authenticate, Salesforce.com provides a SOAP API login operation in partner.wsdl. Therefore, the TestConnection orchestration implementation for Salesforce.com involves using the Cast Iron WebService Invoke Service activity to invoke the Salesforce.com login operation and send the result back to the connector. Here are the steps to implement the TestConnection orchestration for Salesforce.com:

**Configure Provide Service activity**

1. Click on the **Provide Service** activity in the orchestration and click **Map Outputs**.
2. Click **Copy**, copy the variable header in the list of output parameters, and then click **Create**, as shown in Figure 2. The connectionFields are copied from the header parameter to an orchestration variable that can be used by other activities.
3. Double-click on the **header** variable in the To Orchestration side and rename it to **SalesforceConnectionFields**, as shown in Figure 3, and then press **Enter**.

Figure 3. Renaming the variable header to SalesforceConnectionFields

**Configure Salesforce Login activity**

1. Next you need to create a WebService endpoint to make calls to Salesforce.com. In the right-hand Projects tab, right-click on **Endpoints => Create Endpoint** and select **WebService**.
2. In the WebService Endpoint panel, click **Browse** for the WSDL Name field and select **partner.wsdl** to load the Salesforce.com partner.wsdl into this WebService endpoint. This endpoint represents a Salesforce.com endpoint that you can use to make calls to Salesforce.com using web services defined in the partner.wsdl.

3. To rename this newly created WebService endpoint to **SalesforceEndpoint**, right-click on it and select **Rename**.

4. Drag and drop **SalesforceEndpoint** to the orchestration and select the **Invoke Service** activity, as shown in Figure 5. You can use this activity to invoke a web service to the Salesforce.com application.
Figure 5. Creating a WebService Invoke Service activity

5. Click on the newly added **Invoke Service** activity in the orchestration. An activity checklist opens at the bottom.

6. Go to Summary section, rename the activity to **Salesforce Login**, and press **Enter**.

7. Go to the Configure section and select the **login** operation from the list of WSDL operations. You are configuring this activity to make a Salesforce.com login web service call.

8. Go to Map Inputs section. You will see two fields named **username** and **password**, which are required for the Salesforce.com login operation.

9. The URL field is an optional parameter for this activity. Hence you need to enable it by adding optional parameters as shown in Figure 6. Right-click on the blank space in the To Activity side and select **Show Optional Parameters**. All of the optional parameters of this activity will be displayed, and you will see a field location to which you need to map the URL from connection fields.

**Figure 6. Enabling optional parameters in WebService Invoke Service activity Map Inputs**

10. Map the Salesforce connection fields to this activity input. Click on **Select Inputs**, select the **SalesforceConnectionFields** variable, and click **OK**. Map the connection fields to Salesforce Login activity as shown in Figure 7:
Figure 7. Mapping from connection fields to WebService Invoke Service activity input

11. Go to the Map Outputs section, copy the body to a new variable on the right side, and rename the variable to loginResponse.

Configure Send Reply activity

This section shows you how to configure the Send Reply activity to send the Salesforce TestConnection result back to the connector. Depending on whether the login response is successful, you need to send a success response message or a failure response message back to the connector. To check this condition, use Cast Iron If..Then activity and configure the Send Reply activity as shown in Figure 8:

Figure 8. Configure If... Then activity with passwordExpired Boolean field
1. From the Logic section under the Activities tab, drag and drop the **If...Then** activity to the orchestration.
2. Click on the **If** activity in the orchestration.
3. Click on the **...** button at the bottom. The Select XPath window opens.
4. Select the **loginResponse** variable in the Variables combo box.
5. Select the **passwordExpired** field and click **OK**.
6. Select the operator **=** in the Op column and enter **false()** in the Right Hand Expression column.
7. Drag and drop the **Send Reply** activity inside the **If** condition. Rename this activity to **Send Success Reply**.
8. Click on the **Send Reply** activity.
9. Provide the default value **true** to the Success field.
10. Provide the default value **TestConnection Successful** to the Message field.
11. Drag and drop another **Send Reply** activity from **SalesforceEndpoint** to the **Else** condition. Rename this activity to **Send Failure Reply**.
12. To the Send Failure Reply activity, provide **false** as the default value to the Success field and **Password expired. TestConnection Failed** to the Message field.

You have now configured a valid TestConnection orchestration. It is a best practice to add a Try-Catch block to the entire orchestration to handle error conditions.

**Configure Try-Catch block**

1. Right-click the initial orchestration image and select **Add Catch Branch**. This will add a catch block for the entire orchestration.
2. Drag and drop another **Send Reply** activity from **SalesforceEndpoint** to the **CatchAll** block. Rename this activity as **Send Fault Reply**.
3. Go to the Map Inputs section of this Send Fault Reply activity and enter **false** as the default value for the Success field.
4. Click **Select Inputs**, select **faultInfo xml** variable from the list, and click **OK**.
5. Wire the **message** field from **faultInfo** variable to the **message** field on the right hand side. At the end, you will see the complete orchestration as shown in Figure 9:

   **Figure 9. Complete TestConnection orchestration**

**Verify TestConnection orchestration**

You can verify this TestConnection orchestration using the utility Invoke Service:
1. From the Orchestration menu, click **Invoke Service**, as shown in Figure 10. The orchestration starts and opens the Invoke Service window.

**Figure 10. Selecting Invoke Service utility from Orchestration menu**

2. Provide your Salesforce.com credentials in the UserName, Password, and URL fields and click **Execute**. The TestConnection orchestration will be called and will fetch the result.
3. Click **Show Response** on the right side. You will see the response from this orchestration in the right side window: Executing Invoke Service utility for TestConnection orchestration.

To verify a negative case, provide invalid credentials and note the response from the orchestration.

**Conclusion**

Part 2 of this tutorial series showed you how to implement the plug-in connector TestConnection orchestration using the Salesforce.com login operation. In Part 3, you will learn how to build a ListObjects template orchestration to implement the browsing of objects feature for the connector.

**Acknowledgments**

The author would like to thank Sarath K. Ambadas and Srihari Na Kumar of IBM India for reviewing and providing valuable suggestions for this tutorial.
Resources

- **WebSphere Cast Iron resources**
  - **WebSphere Cast Iron Cloud Integration Knowledge Center**
    A single portal to all WebSphere Cast Iron Cloud Integration documentation, with conceptual, task, and reference information on installing, configuring, and using WebSphere Cast Iron.
  - **WebSphere Cast Iron Cloud Integration product library**
    Product announcements, case studies, white papers, and more.
  - **WebSphere Cast Iron Cloud Integration product page**
    Product descriptions, product news, training information, support information, and more.
  - **WebSphere Cast Iron Cloud Integration support**
    A portal for support problems and their solutions, plus downloads, fixes, problem tracking, and more.
  - **WebSphere Cast Iron Cloud Integration Community forums**
    Get answers to your technical questions and share your expertise with other WebSphere Cast Iron users.
  - **IBM Redbook: Getting started with WebSphere Cast Iron Cloud Integration**
    Detailed introduction to the development and administrative interfaces for WebSphere Cast Iron.
  - **Salesforce.com: SOAP API Developer's Guide**
    Salesforce provides programmatic access to your organization's information using simple, powerful, and secure APIs.
  - **WebSphere Cast Iron Connector Development Kit (CDK) Developer's Guide**
    For developers who want to use the WebSphere Cast Iron Studio development platform to develop and deploy plug-in connectors that extract data from application systems or other data sources.
  - **Developing a starter activity for plug-in connectors using the WebSphere Cast Iron Studio CDK**
    This developerWorks article shows you how to use the WebSphere Cast Iron CDK to develop a plug-in connector that has a starter activity, deploy the plug-in, and test it locally on WebSphere Cast Iron Studio.
  - **Hints and pointers for connector development in WebSphere Cast Iron Studio**
    This developerWorks article shows you how to easily configure the base orchestrations for connectors developed with the WebSphere Cast Iron Studio CDK wizard. Includes tips on password and fault handling.

- **WebSphere resources**
  - **developerWorks WebSphere**
    Technical resources for developers who use WebSphere and related middleware products. Includes downloads, how-to information, support resources, and a free technical library of tutorials, best practices, IBM Redbooks, and product manuals.
  - **Most popular WebSphere trial downloads**
    No-charge trial downloads for key WebSphere products.
  - **WebSphere forums and community**
Product-specific forums where you can get answers to your technical questions and share your expertise with other WebSphere users.

- **WebSphere demos**
  Download and watch these self-running demos, and learn how WebSphere products can provide business advantage for your company.

- **WebSphere-related tutorials on developerWorks**
  Nearly 3000 tutorials on WebSphere and related technologies by top practitioners and consultants inside and outside IBM. Search for what you need.

- **WebSphere-related books from IBM Press**
  Convenient online ordering through Barnes & Noble.

- **WebSphere-related events**
  Conferences, trade shows, Webcasts, and other events around the world.

- **developerWorks resources**
  - **Trial downloads for IBM software products**
    No-charge trial downloads for selected IBM® DB2®, Lotus®, Rational®, Tivoli®, and WebSphere® products.
  - **developerWorks blogs**
    Join a conversation with developerWorks users and authors, and IBM developers.
  - **Technical events for developers**
    Conferences, seminars, and technical events to deepen your technical expertise and help you succeed in your software projects.
  - **developerWorks podcasts**
    Listen to interesting and offbeat interviews and discussions with software innovators.
  - **developerWorks on Twitter**
    Check out recent Twitter messages and URLs.
  - **IBM Education Assistant**
    A collection of multimedia educational modules that will help you better understand IBM software products and use them more effectively to meet your business requirements.
About the author

Raghvendra Neelekani

Raghavendra Neelekani is a Staff Software Engineer for IBM in India. He has been working for IBM since 2009, and has extensive experience with Java and Web 2.0 technologies. He works on the WebSphere Cast Iron Development Team and has developed many Cast Iron connectors. You can contact Raghavendra at raneelek@in.ibm.com.

© Copyright IBM Corporation 2015
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)