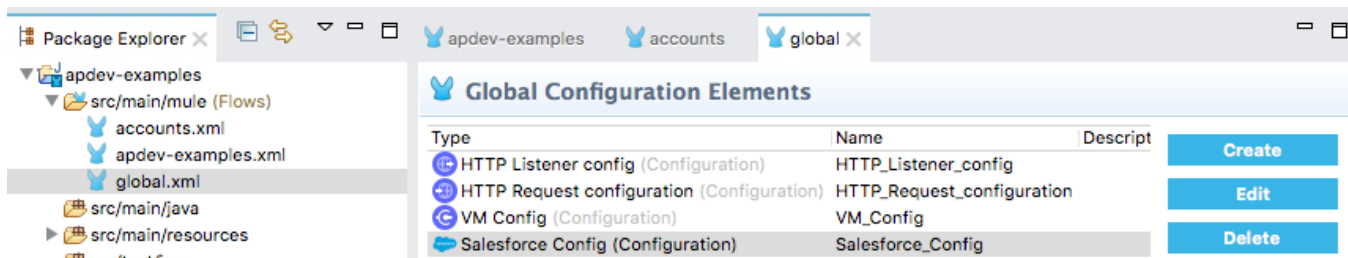


Walkthrough 7-3: Encapsulate global elements in a separate configuration file

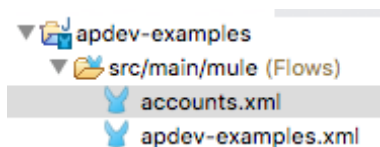
In this walkthrough, you refactor your apdev-examples project. You will:

- Create a new configuration file with an endpoint that uses an existing global element.
- Create a configuration file global.xml for just global elements.
- Move the existing global elements to global.xml.
- Create a new global element in global.xml and configure a new connector to use it.

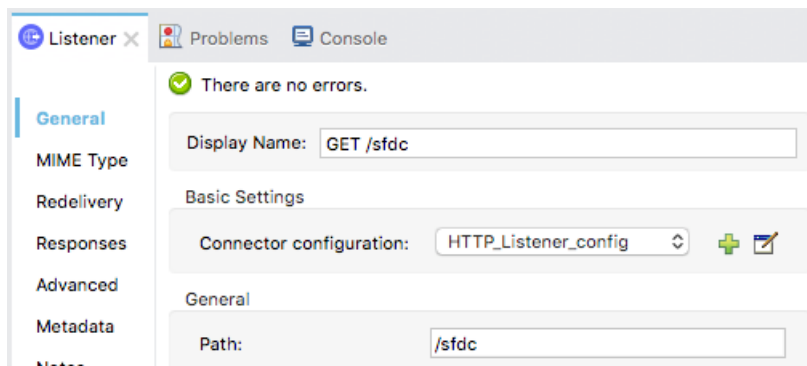


Create a new configuration file

1. Return to the apdev-examples project.
2. In the Package Explorer, right-click the project and select New > Mule Configuration File.
3. In the dialog box, set the name to accounts and click Finish.



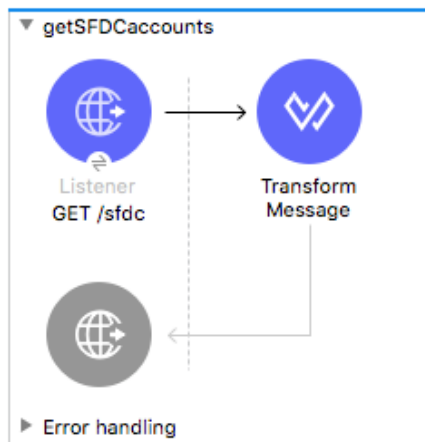
4. Drag an HTTP Listener to the canvas from the Mule Palette.
5. In the Listener properties view, set the display name to GET /sfdc.
6. Set the connector configuration to the existing HTTP_Listener_config.
7. Set the path to /sfdc and the allowed methods to GET.



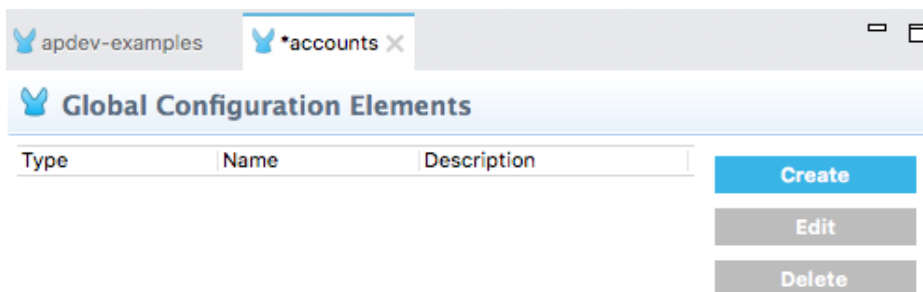
8. Add a Transform Message component to the flow.
9. In the Transform Message properties view, change the output type to application json and set the expression to payload.

```
1 %dw 2.0
2 output application/json
3 ---
4 payload
```

10. Change the name of the flow to getSFDCaccounts.

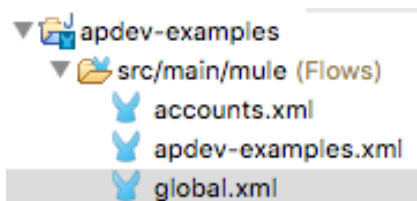


11. Switch to the Global Elements view; you should not see any global elements.



Create a global configuration file

12. Create a new Mule configuration file called global.xml.



13. In global.xml, switch to the Configuration XML view.

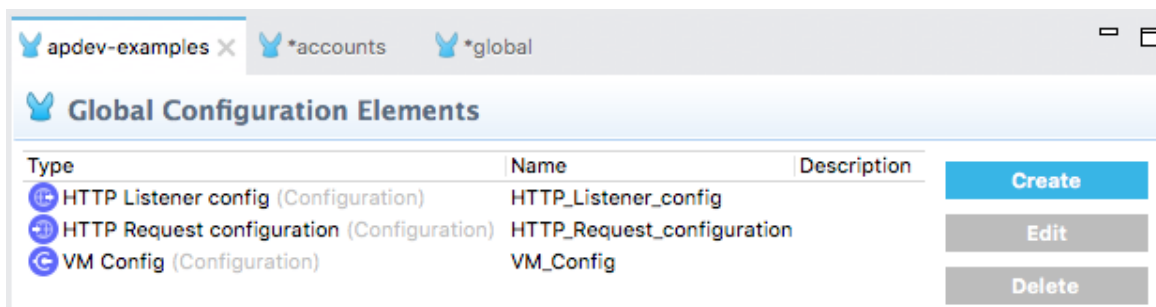
14. Place some empty lines between the start and end mule tags.



```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <mule xmlns="http://www.mulesoft.org/schema/mule/core" xmlns:doc="http://www.mulesoft.org/schema/mule/documentation"
4     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5     xsi:schemaLocation="http://www.mulesoft.org/schema/mule/core http://www.mulesoft.org/schema/mule/core
6                         http://www.mulesoft.org/schema/mule/documentation http://www.mulesoft.org/schema/mule/documentation" />
7
8 </mule>
```

Move the existing global elements to the new global configuration file

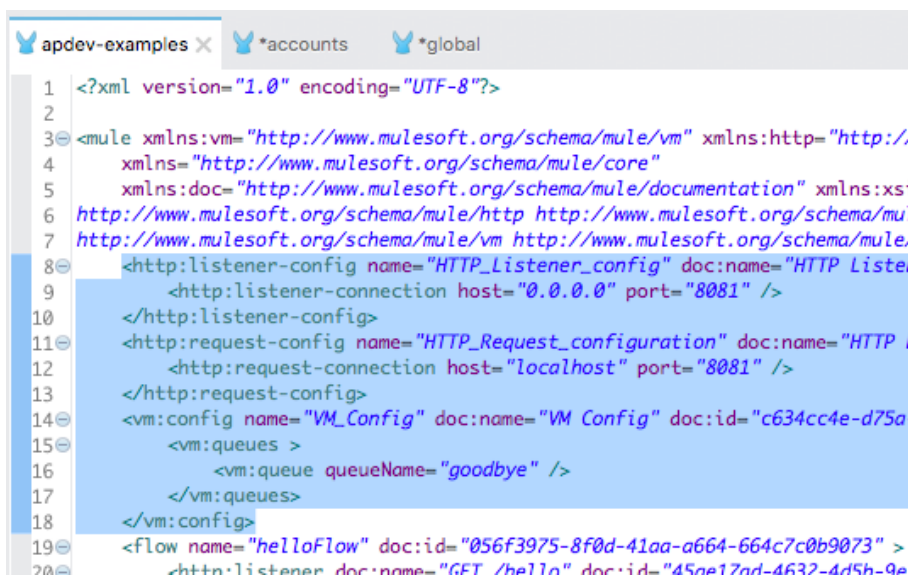
15. Return to apdev-examples.xml.
16. Switch to the Global Elements view and see there are three configurations.



Type	Name	Description
HTTP Listener config (Configuration)	HTTP_Listener_config	
HTTP Request configuration (Configuration)	HTTP_Request_configuration	
VM Config (Configuration)	VM_Config	

Create Edit Delete

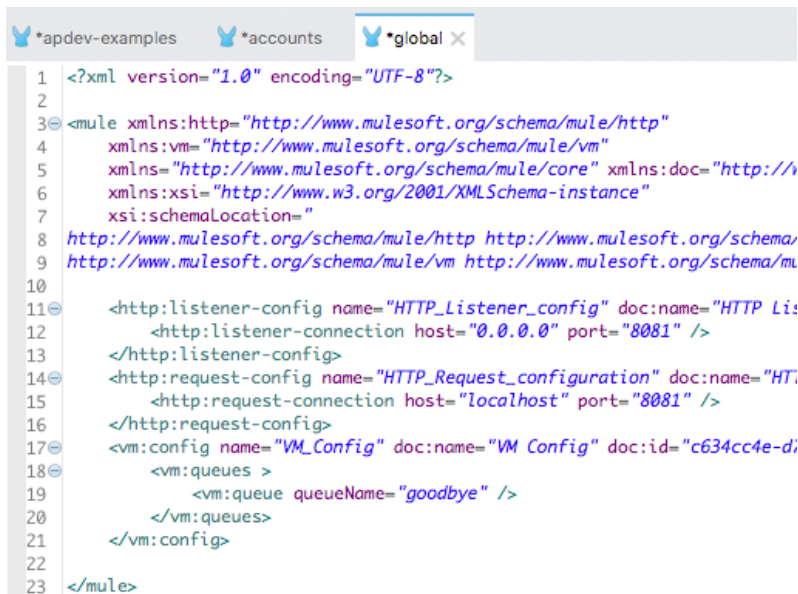
17. Switch to the Configuration XML view.
18. Select and cut the three configuration elements defined before the flows.



```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <mule xmlns:vm="http://www.mulesoft.org/schema/mule/vm" xmlns:http="http://www.mulesoft.org/schema/mule/http"
4     xmlns="http://www.mulesoft.org/schema/mule/core"
5     xmlns:doc="http://www.mulesoft.org/schema/mule/documentation" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6     xsi:schemaLocation="http://www.mulesoft.org/schema/mule/http http://www.mulesoft.org/schema/mule/http
7                         http://www.mulesoft.org/schema/mule/vm http://www.mulesoft.org/schema/mule/vm
8                         http://www.mulesoft.org/schema/mule/core http://www.mulesoft.org/schema/mule/core
9                         http://www.mulesoft.org/schema/mule/documentation http://www.mulesoft.org/schema/mule/documentation" />
10
11 <http:listener-config name="HTTP_Listener_config" doc:name="HTTP Listener config" />
12 <http:listener-connection host="0.0.0.0" port="8081" />
13 </http:listener-config>
14 <http:request-config name="HTTP_Request_configuration" doc:name="HTTP Request configuration" />
15 <http:request-connection host="localhost" port="8081" />
16 </http:request-config>
17 <vm:config name="VM_Config" doc:name="VM Config" doc:id="c634cc4e-d75a-4b4a-8d5b-9a..." />
18 <vm:queues>
19 <vm:queue queueName="goodbye" />
20 </vm:queues>
21 </vm:config>
22 <flow name="helloFlow" doc:id="056f3975-8f0d-41aa-a664-664c7c0b9073">
23 <http:listener doc:name="GET /hello" doc:id="45ae17ad-d632-4d5b-9a..." />
```

Note: If you delete the global elements from the Global Elements view instead, the config-ref values are also removed from the connector operations and you need to re-add them.

19. Return to the Message Flow view.
20. Return to global.xml.
21. Paste the global elements you cut to the clipboard between the start and end mule tags.

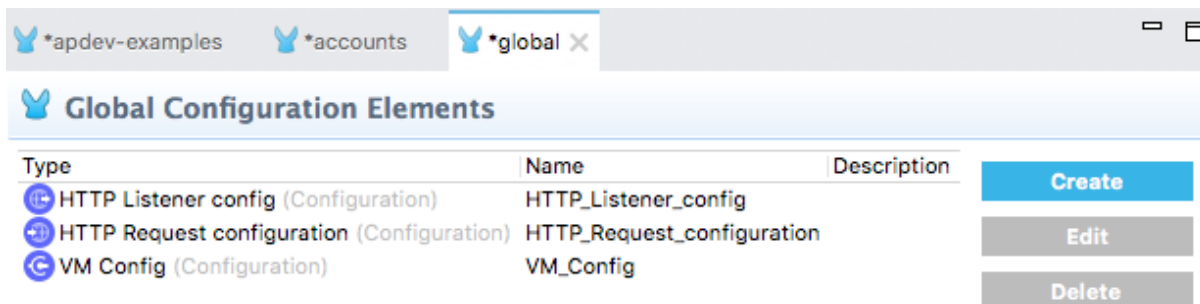


```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <mule xmlns:http="http://www.mulesoft.org/schema/mule/http"
4     xmlns:vm="http://www.mulesoft.org/schema/mule/vm"
5     xmlns="http://www.mulesoft.org/schema/mule/core" xmlns:doc="http://www.mulesoft.org/schema/mule/doc"
6     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
7     xsi:schemaLocation="
8         http://www.mulesoft.org/schema/mule/http http://www.mulesoft.org/schema/mule/http
9         http://www.mulesoft.org/schema/mule/vm http://www.mulesoft.org/schema/mule/vm
10
11 <http:listener-config name="HTTP_Listener_config" doc:name="HTTP Listener" >
12     <http:listener-connection host="0.0.0.0" port="8081" />
13 </http:listener-config>
14 <http:request-config name="HTTP_Request_configuration" doc:name="HTTP Request" >
15     <http:request-connection host="localhost" port="8081" />
16 </http:request-config>
17 <vm:config name="VM_Config" doc:name="VM Config" doc:id="c634cc4e-d1" >
18     <vm:queues >
19         <vm:queue queueName="goodbye" />
20     </vm:queues>
21 </vm:config>
22
23 </mule>

```

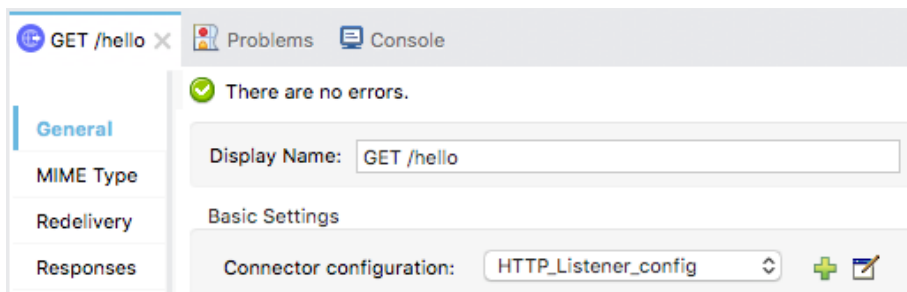
22. Switch to the Global Elements view; you should see the three configurations.



Type	Name	Description	
HTTP Listener config (Configuration)	HTTP_Listener_config		Create
HTTP Request configuration (Configuration)	HTTP_Request_configuration		Edit
VM Config (Configuration)	VM_Config		Delete

Test the application

23. Return to apdev-examples.xml.
24. Select the GET /hello HTTP Listener in helloFlow; the connector configuration should still be set to HTTP_Listener_config, which is now defined in global.xml.



GET /hello

Problems Console

General

MIME Type

Redelivery

Responses

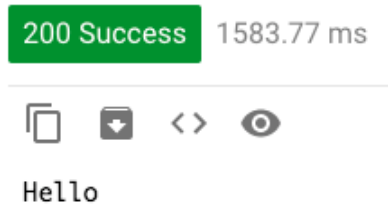
There are no errors.

Display Name: GET /hello

Basic Settings

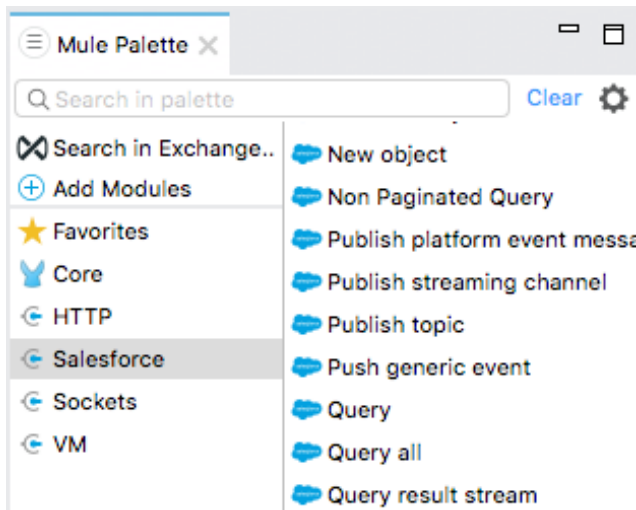
Connector configuration: HTTP_Listener_config

25. Run the project.
26. In Advanced REST Client, send the same request; you should still get a response of Hello.



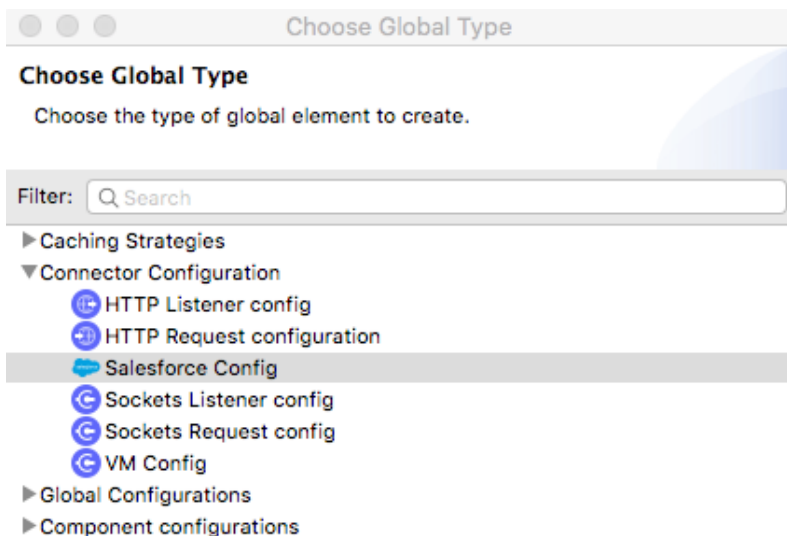
Create a new global element for the Salesforce component in global.xml

27. Return to apdev-examples.xml.
28. In the Mule Palette, select Add Modules.
29. Select the Salesforce connector in the right side of the Mule Palette and drag and drop it into the left side.

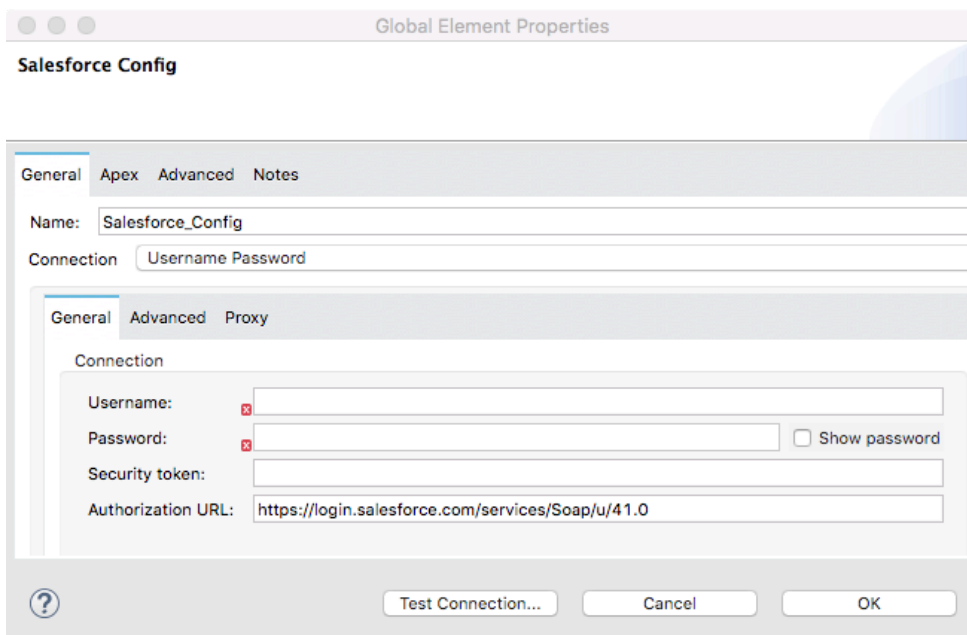


30. Locate the new Salesforce JAR in the project.
31. Return to global.xml.
32. Click Create.

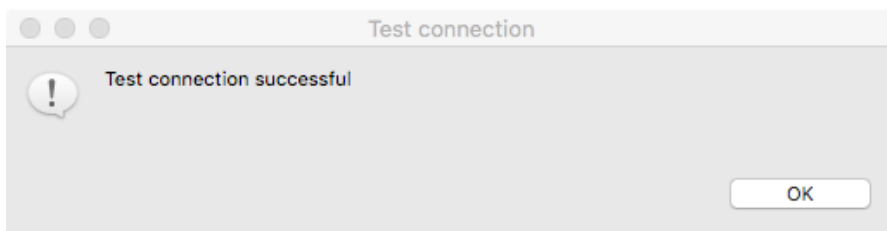
33. In the Choose Global Type dialog box, select Connector Configuration > Salesforce Config and click OK.



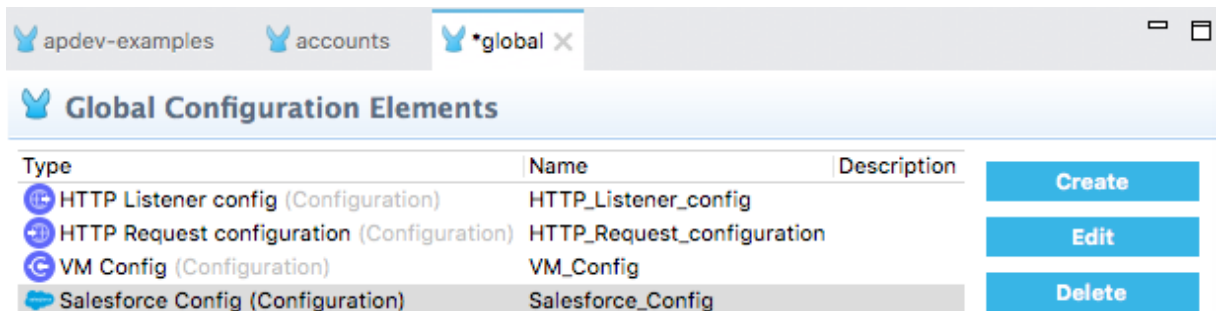
34. In the Global Element Properties dialog box enter your Salesforce username, password, and security token.



35. Click Test Connection; your connection should be successful.



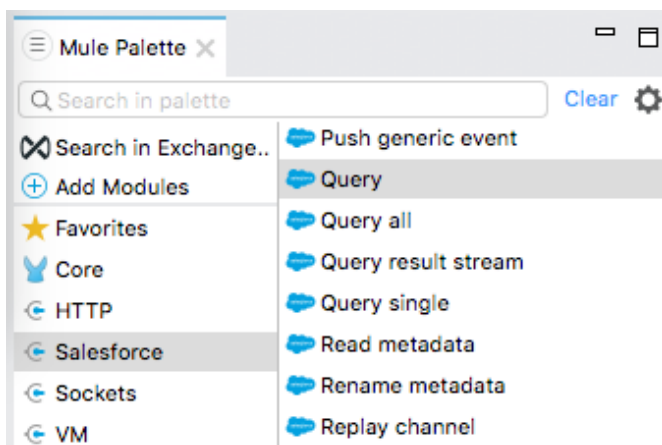
36. In the Test connection dialog box, click OK.
37. In the Global Element Properties dialog box, click OK; you should now see the new configuration in global.



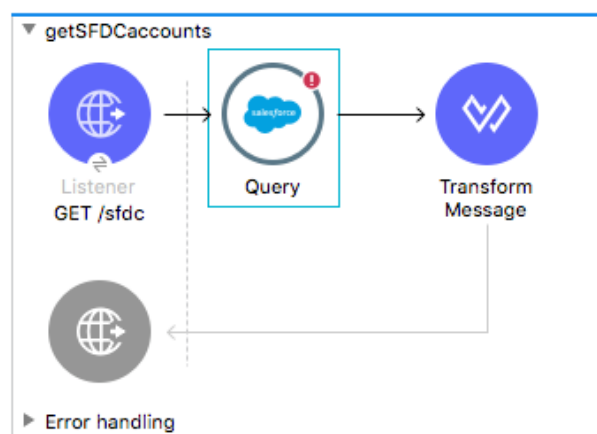
Type	Name	Description	
HTTP Listener config (Configuration)	HTTP_Listener_config		Create
HTTP Request configuration (Configuration)	HTTP_Request_configuration		Edit
VM Config (Configuration)	VM_Config		
Salesforce Config (Configuration)	Salesforce_Config		Delete

Add a new Salesforce operation that uses the global configuration

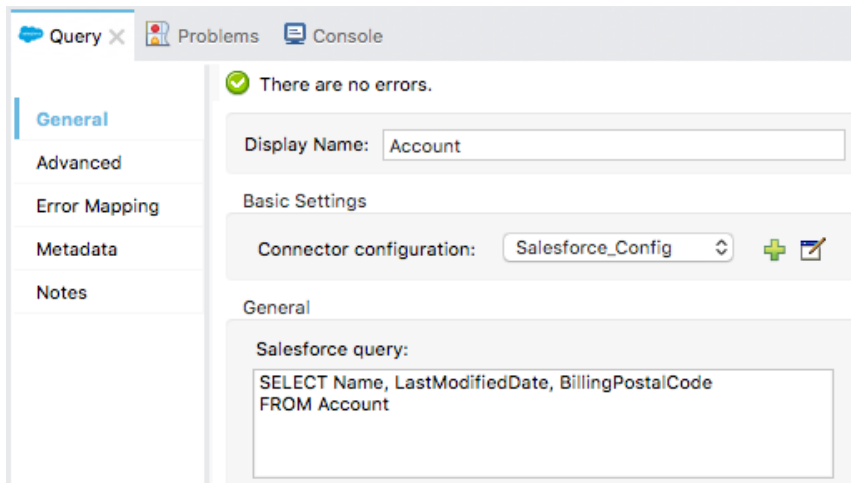
38. Return to the Message Flow view in accounts.xml.
39. In the Mule Palette, select Salesforce.



40. Locate the Query operation in the right side of the Mule Palette and drag and drop it before the Logger in getSFDCaccounts.

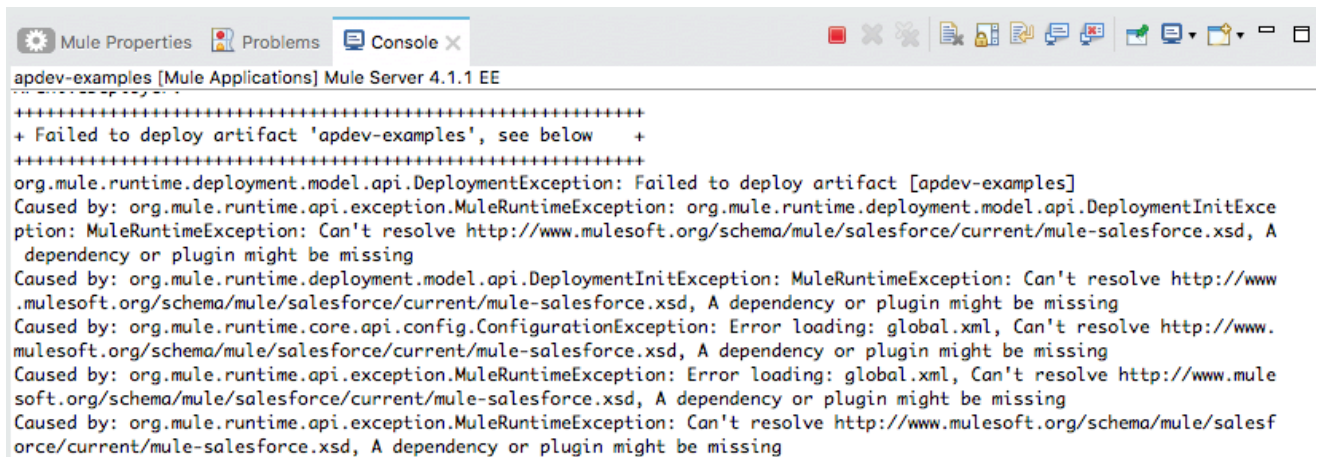


41. In the Query properties view, set the display name to Account.
42. Set the connector configuration to the existing Salesforce_Config.
43. Return to the course snippets.txt file and copy the Salesforce query.
44. Return to Anypoint Studio and paste the query in the Salesforce query section of the Query properties view.



Test the application

45. Save all the files; you should get a missing dependency error in the console.



46. Stop the project.
47. Run the project.

48. In Advanced REST Client, send a request to <http://localhost:8081/sfdc>; you should get a list of the accounts in your Salesforce account.

The screenshot shows the Advanced REST Client interface. At the top, the 'Method' is set to 'GET' and the 'Request URL' is 'http://localhost:8081/sfdc'. A blue 'SEND' button is visible. Below this, the 'Parameters' section is collapsed. The response status is '200 OK' with a response time of '1757.84 ms'. A 'DETAILS' link is also present. The response body is displayed in a code editor, showing a JSON array of two account objects. The first object has 'LastModifiedDate': '2015-07-12T21:00:51.000Z', 'BillingPostalCode': null, 'Id': null, 'type': 'Account', and 'Name': 'GenePoint'. The second object is partially visible.

```
[
  {
    "LastModifiedDate": "2015-07-12T21:00:51.000Z",
    "BillingPostalCode": null,
    "Id": null,
    "type": "Account",
    "Name": "GenePoint"
  },
  {
    "LastModifiedDate": "2015-07-12T21:00:51.000Z",
    "BillingPostalCode": null
  }
]
```