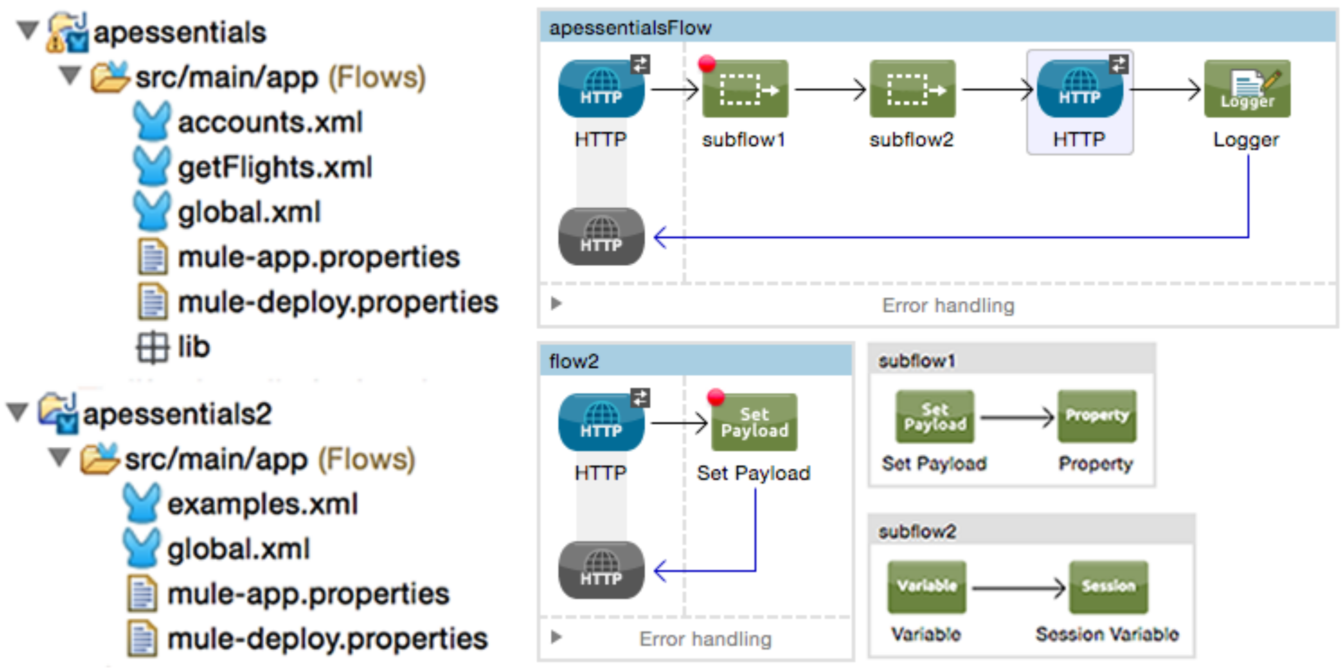
Module 6: Refactoring Mule Applications



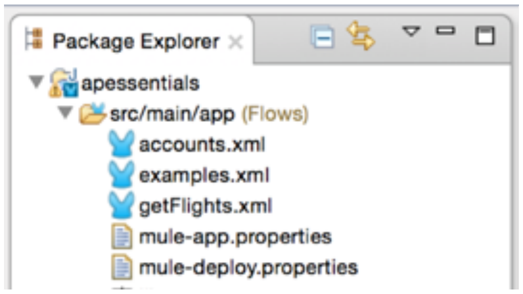
In this module, you will learn:

- To separate applications into multiple configuration files.
- To encapsulate global elements in a separate configuration file.
- To create and run multiple applications.
- To create and reference flows and subflows.
- About variable persistence through subflows and flows and across transport barriers.

Walkthrough 6-1: Separate applications into multiple configuration files

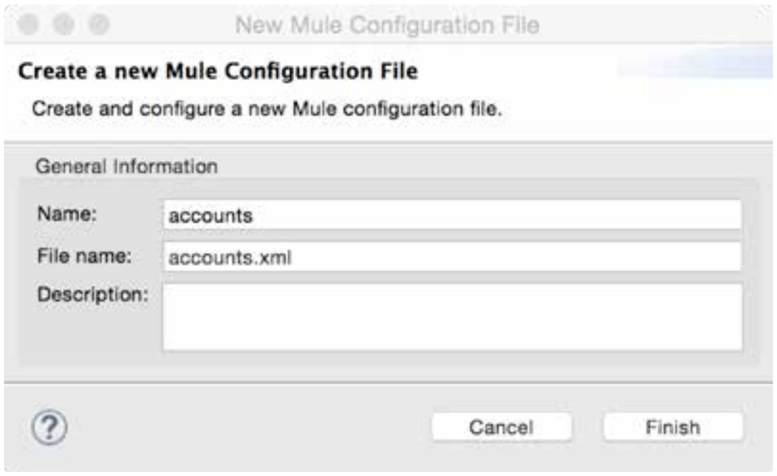
In this walkthrough, you will refactor your monolithic apessentials application. You will:

- Separate the account flows into accounts.xml.
- Move the rest of the example flows into examples.xml.
- Rename apessentials.xml to getFlights.xml.



Move account flows to a new configuration file

1. Right-click the apessentials project in the Package Explorer and select New > Mule Configuration File.
2. Set the name to accounts and click Finish.



New Mule Configuration File

Create a new Mule Configuration File

Create and configure a new Mule configuration file.

General Information

Name: accounts

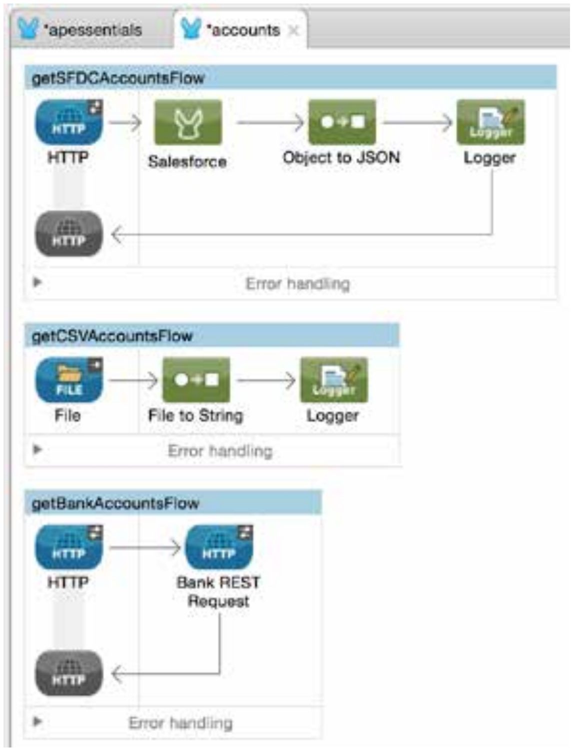
File name: accounts.xml

Description:

3. In accounts.xml, switch to the Configuration XML view.
4. Place some empty lines between the start and end mule tags.
5. Return to apessentials.xml and switch to the Configuration XML view.
6. Select and cut getSFDCAccountsFlow, getCSVAccountsFlow, and getBankAccountsFlow.
7. Return to accounts.xml and paste the flows inside the mule tags.

Note: If these flows are not adjacent, you will need to repeat the process several times.

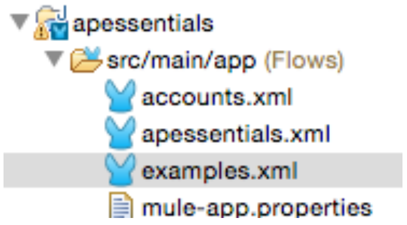
8. Switch to the Message Flow view.



9. Save all the files by selecting File > Save All, clicking the Save All button, or pressing Shift+Cmd+S.

Move non-flight flows to a new configuration file

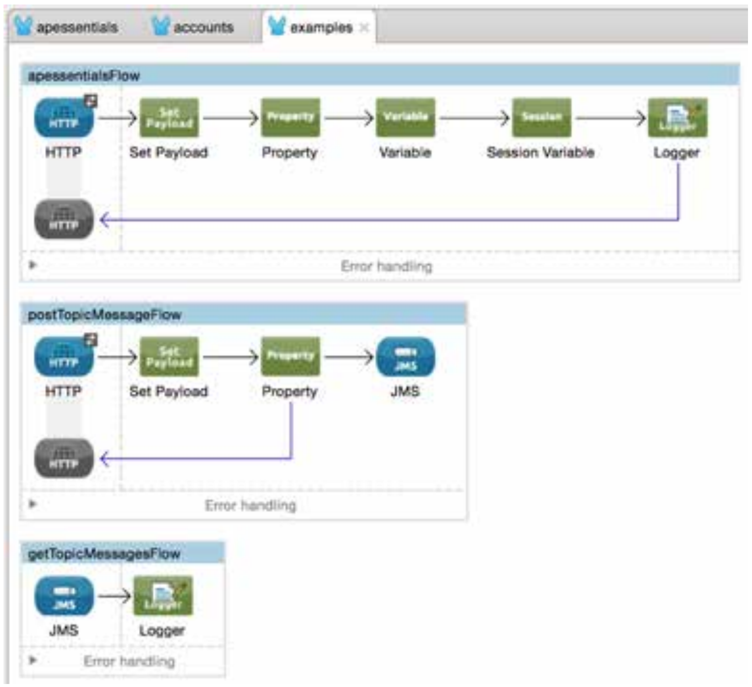
10. Right-click the apessentials project in the Package Explorer and select New > Mule Configuration File.
11. Set the name to examples and click Finish.



12. In examples.xml, switch to the Configuration XML view.
13. Place some empty lines between the start and end mule tags.
14. Return to apessentials.xml and select and cut apessentialsFlow, getTopicMessagesFlow, and postTopicMessageFlow.
15. Return to examples.xml and paste the flows inside the mule tags.

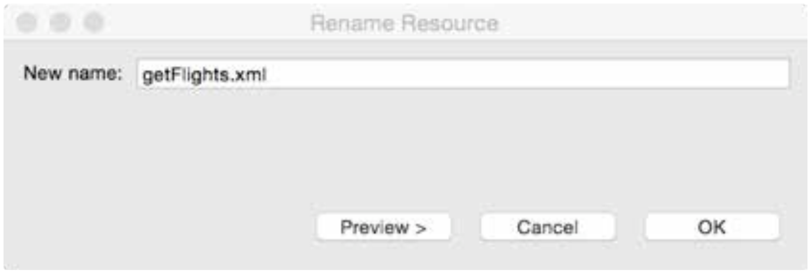
Note: If these flows are not adjacent, you will need to repeat the process several times.

16. Switch to the Message Flow view.
17. Save all the files.



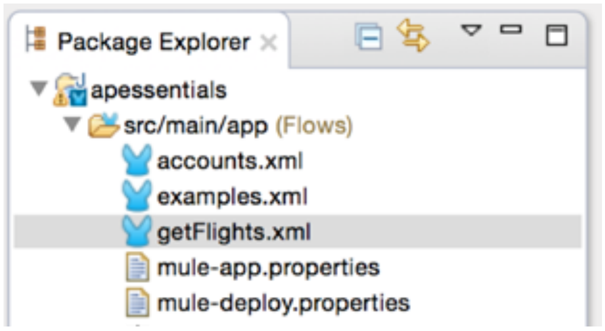
Rename the apessentials configuration file

18. Right-click apessentials.xml in the Package Explorer and select Refactor > Rename.
19. In the Rename Resource dialog box, enter a new name of getFlights.xml and click OK.


Rename Resource

New name: getFlights.xml

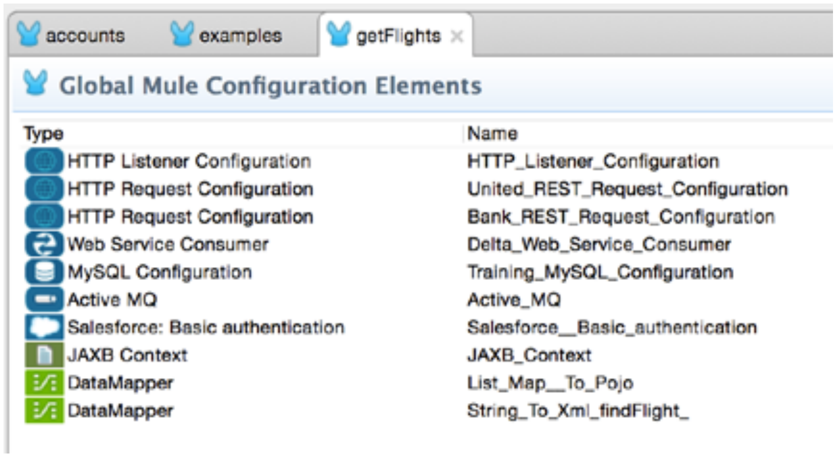
20. Double-click getFlights.xml in the Package Explorer to reopen it.



MuleSoft

Examine the global elements

21. Click the Global Elements tab at the bottom of the canvas.
22. Notice that all of the global elements are still defined in getFlights.xml.



Type	Name
HTTP Listener Configuration	HTTP_Listener_Configuration
HTTP Request Configuration	United_REST_Request_Configuration
HTTP Request Configuration	Bank_REST_Request_Configuration
Web Service Consumer	Delta_Web_Service_Consumer
MySQL Configuration	Training_MySQL_Configuration
Active MQ	Active_MQ
Salesforce: Basic authentication	Salesforce__Basic_authentication
JAXB Context	JAXB_Context
DataMapper	List_Map__To_Pojo
DataMapper	String_To_Xml_findFlight_

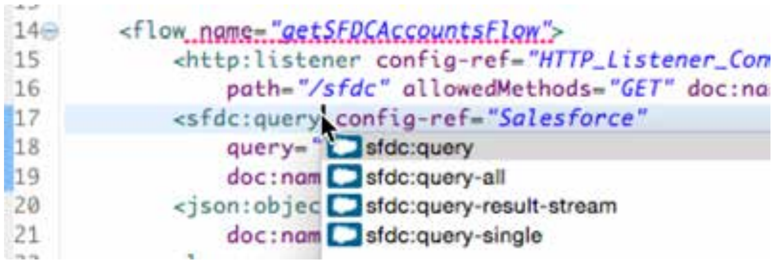
Test the application

23. Run the application; you should get an error that the prefix for sfdc:query is not bound.

```
ERROR 2015-08-20 14:11:19,395 [main] org.mule.module.launcher.application.DefaultMuleApplication: null
org.xml.sax.SAXParseException: The prefix "sfdc" for element "sfdc:query" is not bound.
	at org.apache.xerces.util.ErrorHandlerWrapper.createSAXParseException(Unknown Source) ~[?:?]
	at org.apache.xerces.util.ErrorHandlerWrapper.fatalError(Unknown Source) ~[?:?]
```

Add namespaces and schema locations

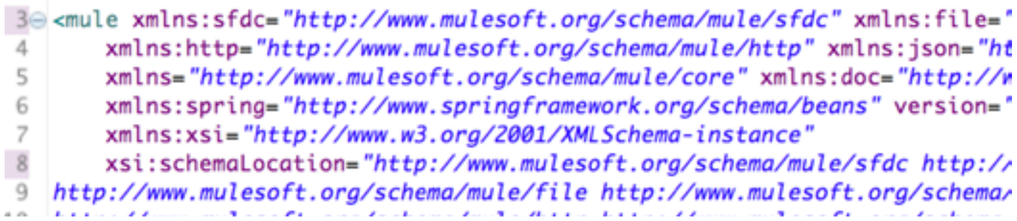
24. Return to the Configuration XML view for accounts.xml.
25. Look for the sfdc prefix in the beginning of the code; you should see that there is not a sfdc namespace definition or schema location.
26. Place the cursor after <sfdc:query and press Ctrl+Space.



```
    <flow name="getSFDCAccountsFlow">
        <http:listener config-ref="HTTP_Listener_Con
            path="/sfdc" allowedMethods="GET" doc:na
        <sfdc:query config-ref="Salesforce"
            query="
            doc:nam
        <json:objec
            doc:nam
```

27. In the autocomplete pop-up, select sfdc:query and press Enter.
28. Delete the new config-ref attribute that was added.

29. Locate the new xmlns and schemaLocation for sfdc.

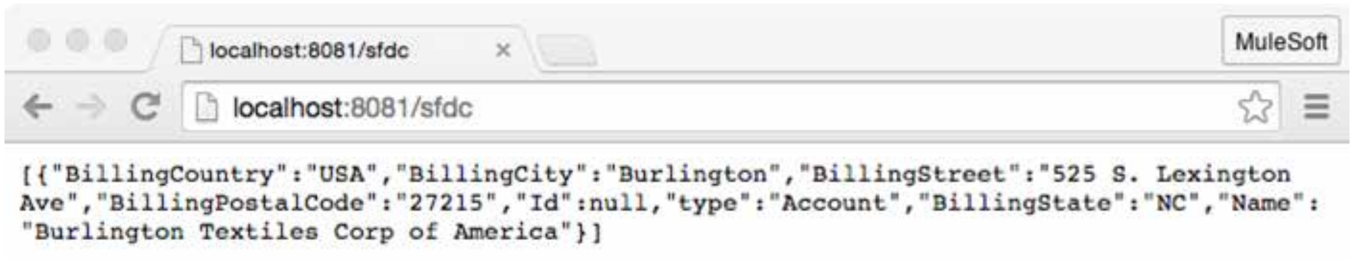


```
<mule xmlns:sfdc="http://www.mulesoft.org/schema/mule/sfdc" xmlns:file="
    xmlns:http="http://www.mulesoft.org/schema/mule/http" xmlns:json="ht
    xmlns="http://www.mulesoft.org/schema/mule/core" xmlns:doc="http://w
    xmlns:spring="http://www.springframework.org/schema/beans" version="
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.mulesoft.org/schema/mule/sfdc http://
http://www.mulesoft.org/schema/mule/file http://www.mulesoft.org/schema/
```

Test the application

30. Save all the files and run the application.

31. Make a request to http://localhost:8081/sfdc; you should now get account results.

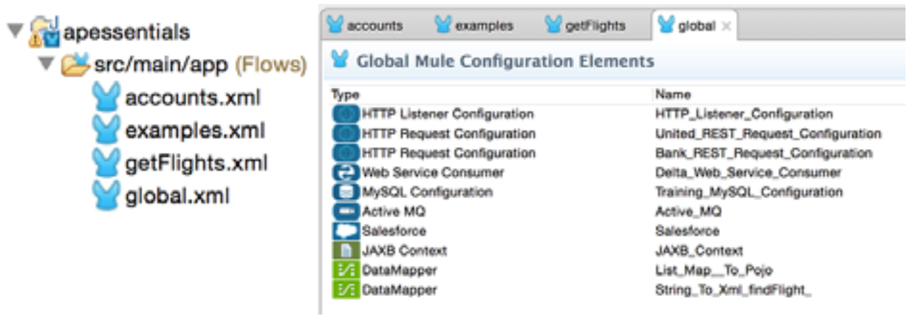


```
[{"BillingCountry":"USA","BillingCity":"Burlington","BillingStreet":"525 S. Lexington
Ave","BillingPostalCode":"27215","Id":null,"type":"Account","BillingState":"NC","Name":
"Burlington Textiles Corp of America"}]
```

Walkthrough 6-2: Encapsulate global elements in a separate configuration file

In this walkthrough, you will separate global elements into a new configuration file. You will:

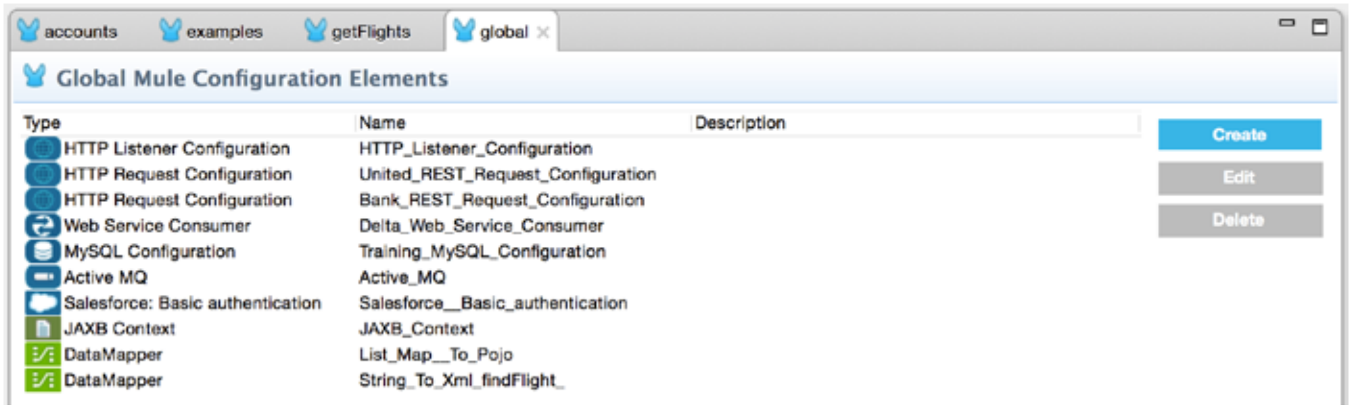
- Create a configuration file for just global elements.
- Move all the existing global elements to this file.



Type	Name
HTTP Listener Configuration	HTTP_Listener_Configuration
HTTP Request Configuration	United_REST_Request_Configuration
HTTP Request Configuration	Bank_REST_Request_Configuration
Web Service Consumer	Delta_Web_Service_Consumer
MySQL Configuration	Training_MySQL_Configuration
Active MQ	Active_MQ
Salesforce	Salesforce
JAXB Context	JAXB_Context
DataMapper	List_Map__To_Pojo
DataMapper	String_To_Xml_findFlight_

Create a new configuration file for the global elements

1. Right-click the getFlights.xml file in the Package Explorer and select Copy.
2. Right-click the src/main/app folder and select > Paste.
3. In the Name Conflict dialog box, enter a name of global.xml and click OK.
4. Open global.xml.
5. Shift-click all the flows to select them all and then right-click and select Delete.
6. Click the Global Elements tab; you should see all the existing global elements.
7. Save the file.

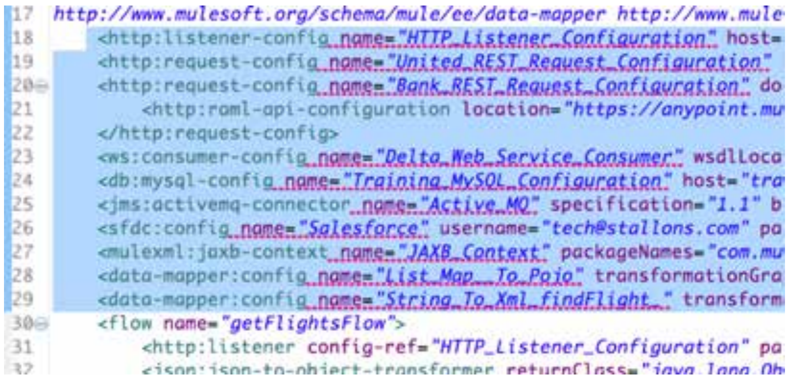


Type	Name	Description
HTTP Listener Configuration	HTTP_Listener_Configuration	
HTTP Request Configuration	United_REST_Request_Configuration	
HTTP Request Configuration	Bank_REST_Request_Configuration	
Web Service Consumer	Delta_Web_Service_Consumer	
MySQL Configuration	Training_MySQL_Configuration	
Active MQ	Active_MQ	
Salesforce: Basic authentication	Salesforce__Basic_authentication	
JAXB Context	JAXB_Context	
DataMapper	List_Map__To_Pojo	
DataMapper	String_To_Xml_findFlight_	

Delete the global elements in getFlights.xml

8. Return to getFlights.xml.

9. Switch the editor to the Configuration XML view.



```
http://www.mulesoft.org/schema/mule/ee/data-mapper http://www.mule
    <http:listener-config name="HTTP_Listener_Configuration" host=
    <http:request-config name="United_REST_Request_Configuration"
    <http:request-config name="Bank_REST_Request_Configuration" do
        <http:raml-api-configuration location="https://anypoint.mu
    </http:request-config>
    <ws:consumer-config name="Delta_Web_Service_Consumer" wsdlLoca
    <db:mysql-config name="Training_MySQL_Configuration" host="tra
    <jms:activemq-connector name="Active_MQ" specification="1.1" b
    <sfdc:config name="Salesforce" username="tech@stallons.com" pa
    <mulexml:jaxb-context name="JAXB_Context" packageNames="com.mu
    <data-mapper:config name="List_Map__To_Pojo" transformationGra
    <data-mapper:config name="String_To_Xml_findFlight_" transform
    <flow name="getFlightsFlow">
        <http:listener config-ref="HTTP_Listener_Configuration" pa
        <json:json-to-object-transformer returnClass="java.lang.Ob
```

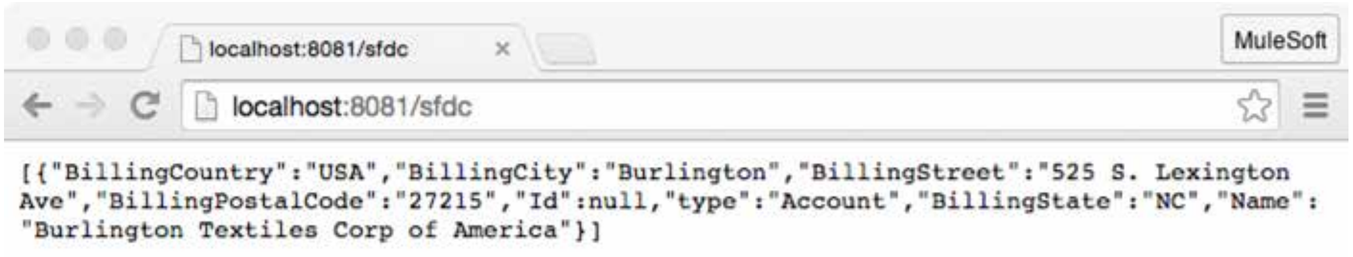
10. Select and delete the ten global elements defined before the flows.

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

11. Save the file.
12. Return to the Message Flow view.
13. Double-click the HTTP Listener connector of any flow; the connector configuration should still be set to HTTP_Listener_Configuration, which is now defined in global.xml.

Test the application

14. Run the application.
15. Make a request to http://localhost:8081/sfdc; you should still get account results.

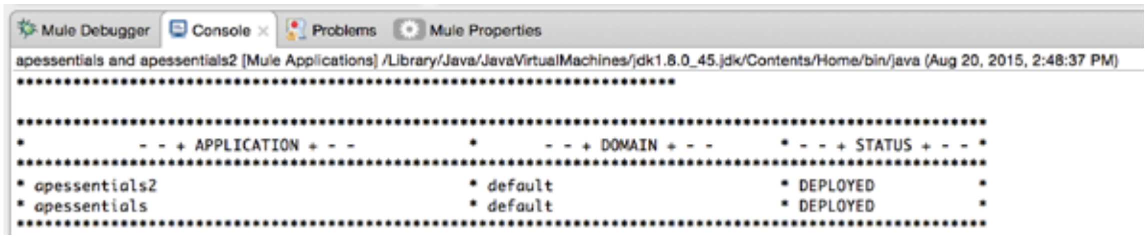


```
[{"BillingCountry":"USA","BillingCity":"Burlington","BillingStreet":"525 S. Lexington
Ave","BillingPostalCode":"27215","Id":null,"type":"Account","BillingState":"NC","Name":
"Burlington Textiles Corp of America"}]
```

Walkthrough 6-3: Create and run multiple applications

In this walkthrough, you will separate your project into two projects. You will:

- Create a new project and application called apessentials2.
- Move the examples configuration file into the apessentials2 application.
- Create new global connector configurations in the new project.
- Create a new run configuration to run multiple applications simultaneously.



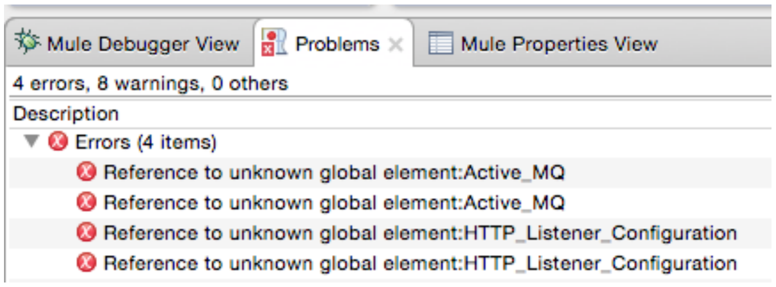
```
apessentials and apessentials2 [Mule Applications] /Library/Java/JavaVirtualMachines/jdk1.8.0_45.jdk/Contents/Home/bin/java (Aug 20, 2015, 2:48:37 PM)
**********************************************************************

**********************************************************************************************************
*            - - + APPLICATION + - -            *        - - + DOMAIN + - -        * - - + STATUS + - - *
**********************************************************************************************************
* apessentials2                                 * default                          * DEPLOYED           *
* apessentials                                  * default                          * DEPLOYED           *
**********************************************************************************************************
```

Create a new project and application

1. Select File > New > Mule Project.
2. Set the name to apessentials2 and click Finish.
3. Right-click the apessentials2.xml file in the Package Explorer and select Delete.
4. In the Delete dialog box, click OK.

Move the examples flow into the new project

5. Drag examples.xml from the apessentials project to the src/main/app folder of the apessentials2 project; you should get unknown global element errors.



4 errors, 8 warnings, 0 others

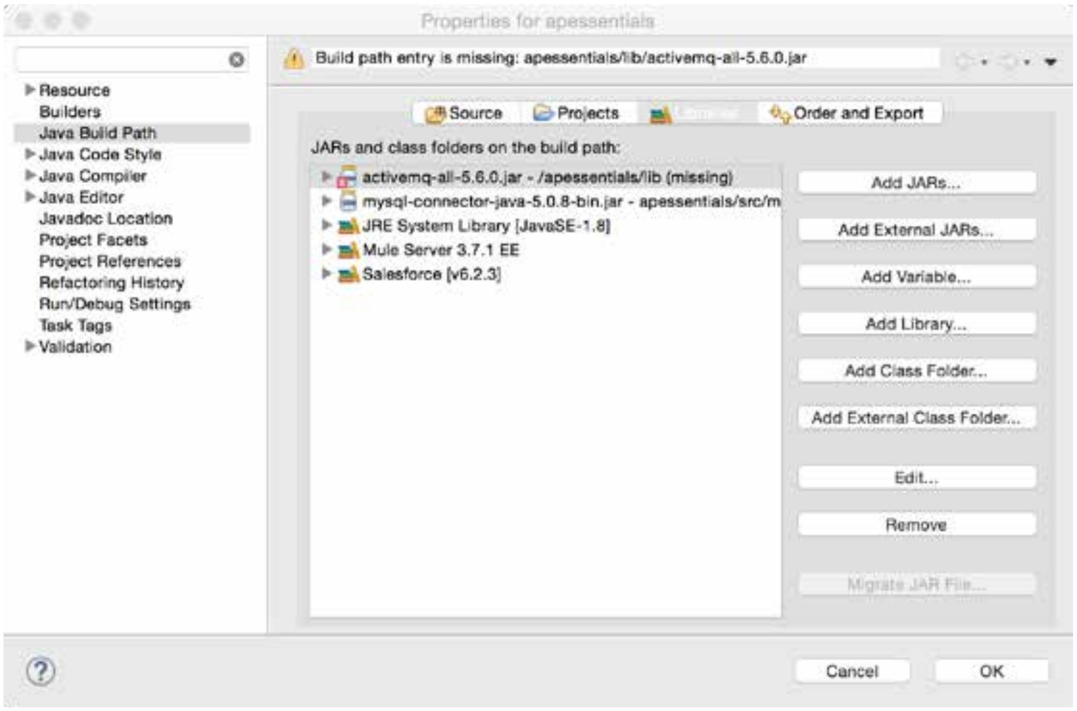
Description

- Errors (4 items)
  - Reference to unknown global element:Active_MQ
  - Reference to unknown global element:Active_MQ
  - Reference to unknown global element:HTTP_Listener_Configuration
  - Reference to unknown global element:HTTP_Listener_Configuration

Move activemq JAR into the new project

6. Right-click apessentials2 and select New > Folder.
7. In the New Folder dialog box, set the folder name to lib and click Finish.
8. Drag activemq-all-{version}.jar.xml from the apessentials/lib folder to the apessentials2/lib folder.

9. Right-click activemq-all-{version}.jar and select Build Path > Add to Build Path; you should see the JAR file added to Referenced Libraries.
10. In global.xml, switch to the Global Elements view.
11. Select the Active MQ configuration element and click Delete.
12. Save the file.
13. Right-click apessentials in the Package Explorer and select Properties.
14. Select Java Build Path and click the Libraries tab.
15. Select activemq-all-{version}.jar and click Remove.


Properties for apessentials

Build path entry is missing: apessentials/lib/activemq-all-5.6.0.jar

JARs and class folders on the build path:

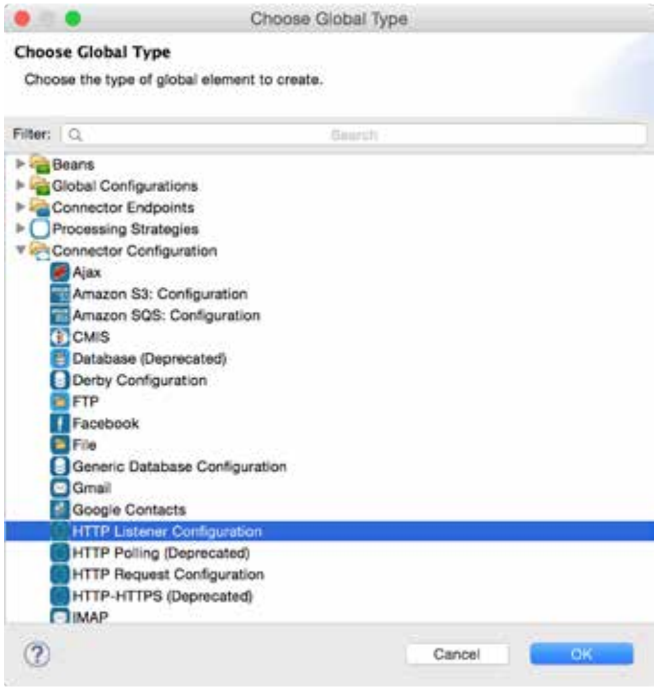
- activemq-all-5.6.0.jar - /apessentials/lib (missing)
- mysql-connector-java-5.0.8-bin.jar - apessentials/src/m
- JRE System Library [JavaSE-1.8]
- Mule Server 3.7.1 EE
- Salesforce [v6.2.3]

16. Click OK.

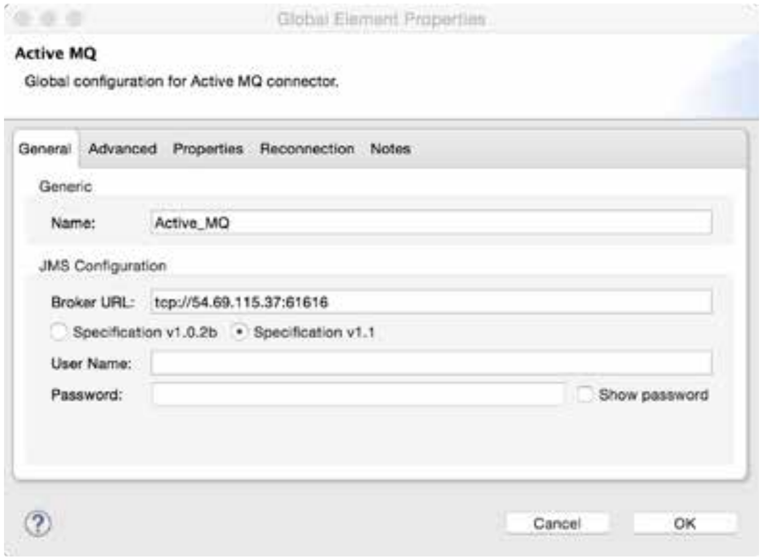
Create new global elements

17. Right-click the apessentials2 project and select New > Mule Configuration File.
18. In the New Mule Configuration File dialog box, set the name to global and click Finish.
19. In global.xml, switch to the Global Elements view.
20. Click the Create button.

21. Select Connector Configuration > HTTP Listener Configuration and click OK.



22. In the Global Elements dialog box, click OK.
23. Click the Create button again and select Connector Configuration > JMS > Active MQ and click OK.
24. In the Global Elements dialog box, set the Broker URL to the value in the course snippets.txt file.
25. Select Specification v1.1 and click OK.



Name: Active_MQ

Broker URL: tcp://54.69.115.37:61616

Specification v1.0.2b [✓ Specification v1.1]

User Name:

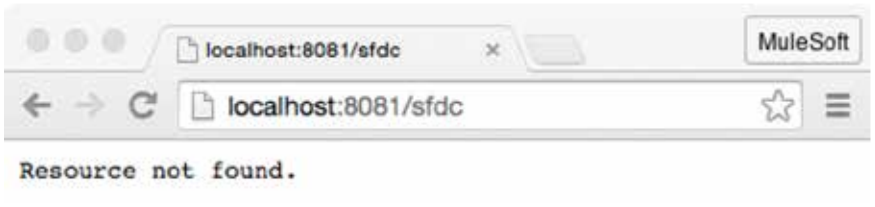
Password:

Note: You could also copy and paste these definitions from the apessentials global.xml file.

26. Save all the files.

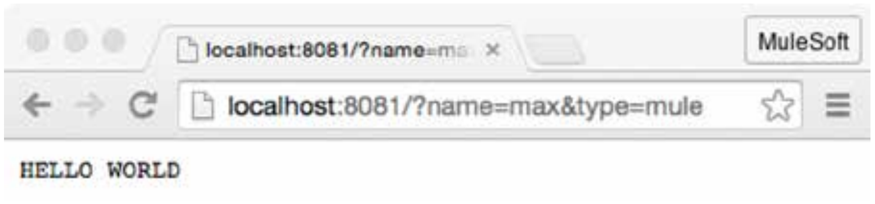
Run the new application

27. Right-click apessentials2 and select Run As > Mule Application.
28. Make a request to http://localhost:8081/sfdc; you should get Resource not found – that application is not running.



```
Resource not found.
```

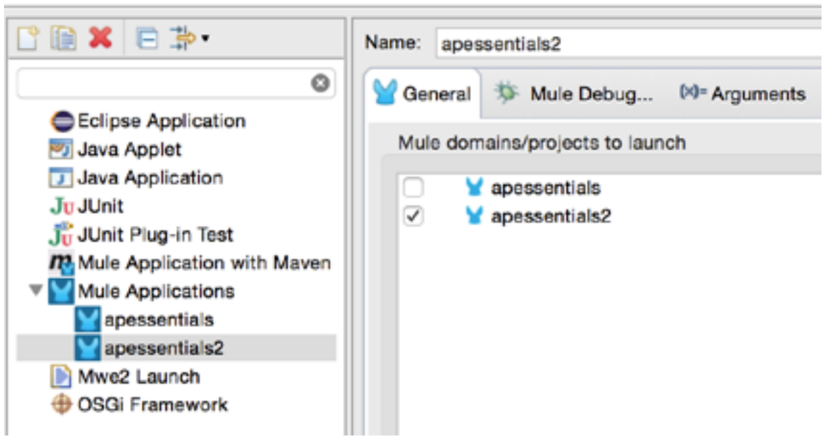
29. Make a request to http://localhost:8081?name=pliny&type=cat using your own query parameter values; you should get HELLO WORLD.



```
HELLO WORLD
```

Create a run configuration to run multiple applications

30. Return to Anypoint Studio.
31. On the main menu bar, select Run > Run Configurations.
32. Click apessentials in the left menu bar under Mule Applications; you should see that the apessentials project is selected to launch.
33. Click apessentials2 in the left menu bar; you should see that the apessentials2 project is selected to launch.

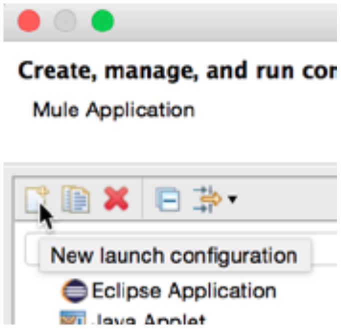


Name: apessentials2

Mule domains/projects to launch

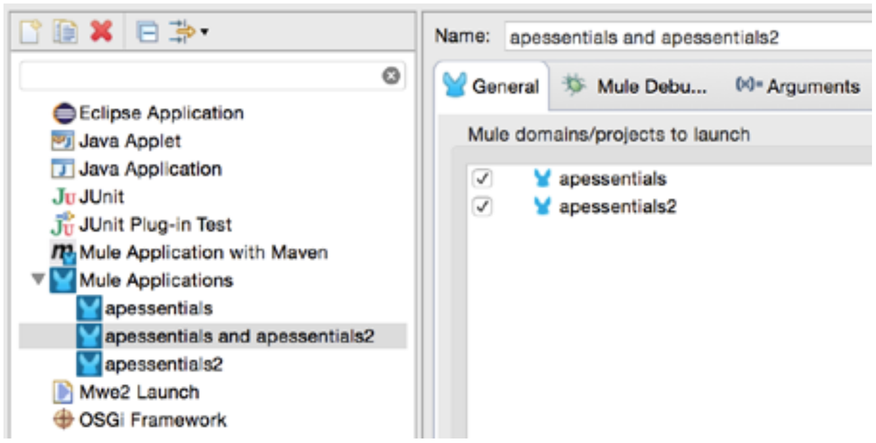
- [ ] apessentials
- [✓] apessentials2

34. Click the New Launch Configuration button.



35. Set the name to apessentials and apessentials2.

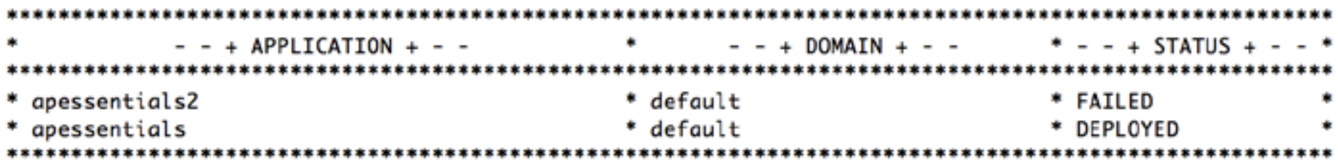
36. On the General tab, select both the apessentials and apessentials2 projects.



Run multiple applications

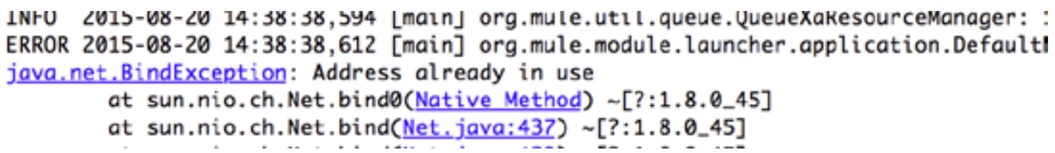
37. Click Run.

38. Look at the console; apessentials should have deployed and apessentials2 failed.



```
**********************************************************************************************
*              - - + APPLICATION + - -              *        - - + DOMAIN + - -        * - - + STATUS + - - *
**********************************************************************************************
* apessentials2                                     * default                          * FAILED             *
* apessentials                                      * default                          * DEPLOYED           *
**********************************************************************************************
```

39. Scroll up the console; you should see an address already in use error.

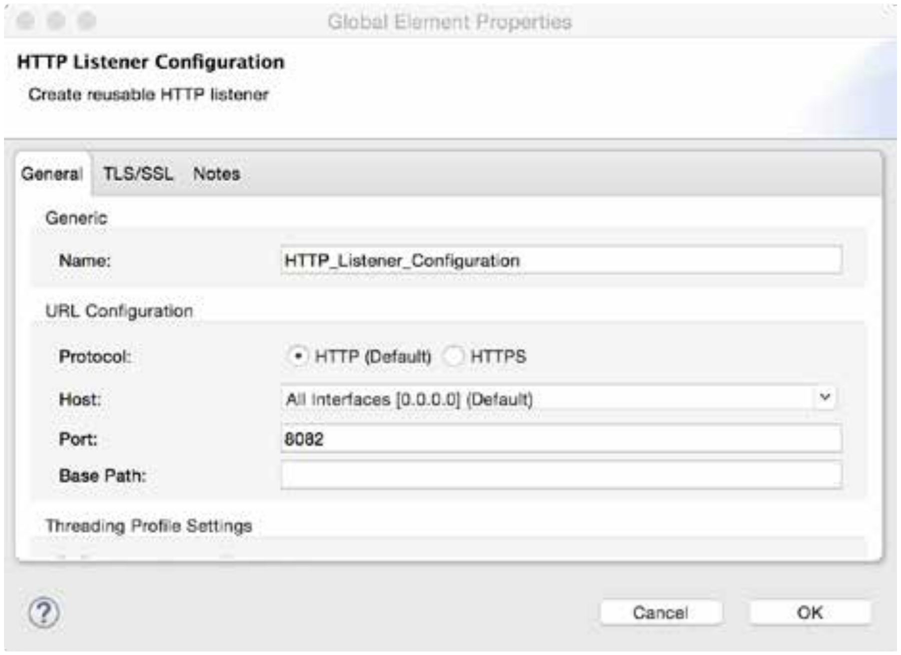


```
INFO  2015-08-20 14:38:38,594 [main] org.mule.util.queue.QueueXaResourceManager: :
ERROR 2015-08-20 14:38:38,612 [main] org.mule.module.launcher.application.Defaultl
java.net.BindException: Address already in use
        at sun.nio.ch.Net.bind0(Native Method) ~[?:1.8.0_45]
        at sun.nio.ch.Net.bind(Net.java:437) ~[?:1.8.0_45]
```

40. Navigate to the Global Elements view in global.xml in apessentials2.

41. Double-click the HTTP Listener Configuration (or select it and click the Edit button).

42. In the Global Elements Properties dialog box, change the port to 8082 and click OK.



Global Element Properties

HTTP Listener Configuration

Create reusable HTTP listener

General | TLS/SSL | Notes

Generic

Name: HTTP_Listener_Configuration

URL Configuration

Protocol: [✓ HTTP (Default)] HTTPS

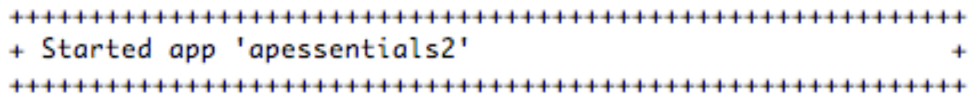
Host: All Interfaces [0.0.0.0] (Default)

Port: 8082

Base Path:

Threading Profile Settings

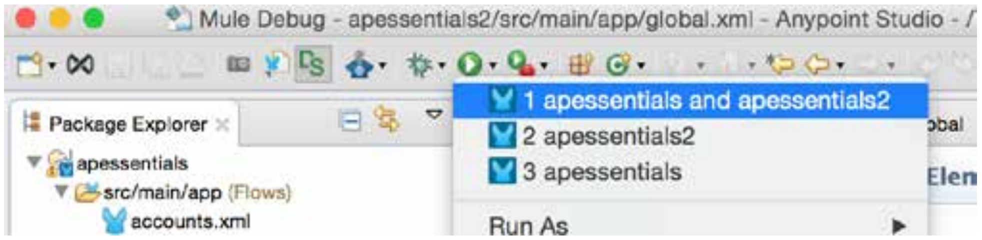
43. Save the file; both applications should now be successfully deployed – apessentials was already deployed.



```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ Started app 'apessentials2'                              +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

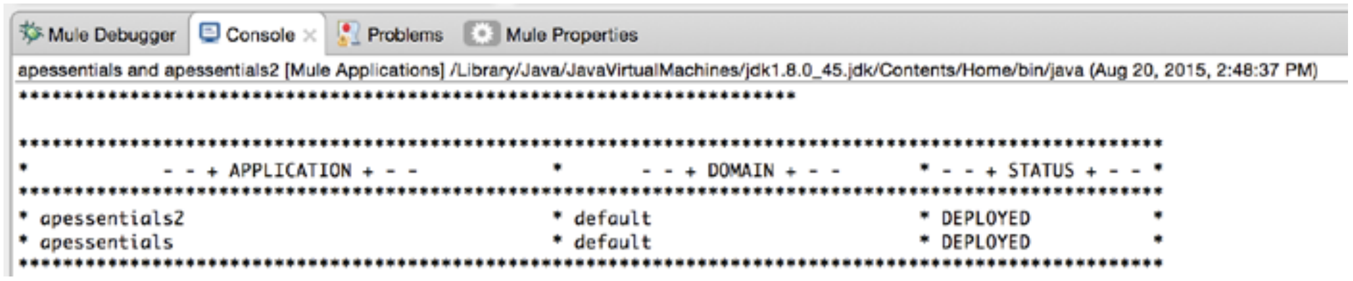
Test the application

44. Make a request to http://localhost:8081/sfdc; you should see Salesforce data.
45. Make a request to http://localhost:8082/?name=max&type=mule using your own query parameter values; you should get HELLO WORLD.
46. Stop the Mule runtime.
47. Click the arrow next to the Run button in the Anypoint Studio toolbar and select apessentials and apessentials2.



Mule Debug - apessentials2/src/main/app/global.xml - Anypoint Studio
Package Explorer
apessentials
src/main/app (Flows)
accounts.xml
1 apessentials and apessentials2
2 apessentials2
3 apessentials
Run As

48. Look at the console; you should both applications deployed.



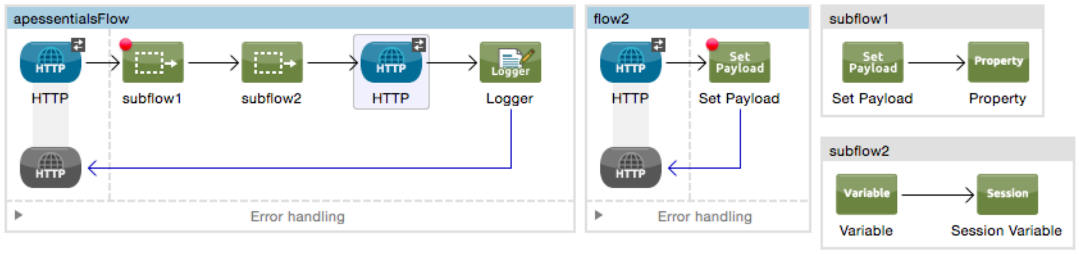
```
apessentials and apessentials2 [Mule Applications] /Library/Java/JavaVirtualMachines/jdk1.8.0_45.jdk/Contents/Home/bin/java (Aug 20, 2015, 2:48:37 PM)
**********************************************************************

**********************************************************************************************************
*            - - + APPLICATION + - -            *       - - + DOMAIN + - -       * - - + STATUS + - - *
**********************************************************************************************************
* apessentials2                                 * default                        * DEPLOYED           *
* apessentials                                  * default                        * DEPLOYED           *
**********************************************************************************************************
```

49. Stop the Mule runtime.

Walkthrough 6-4: Create and reference flows and subflows

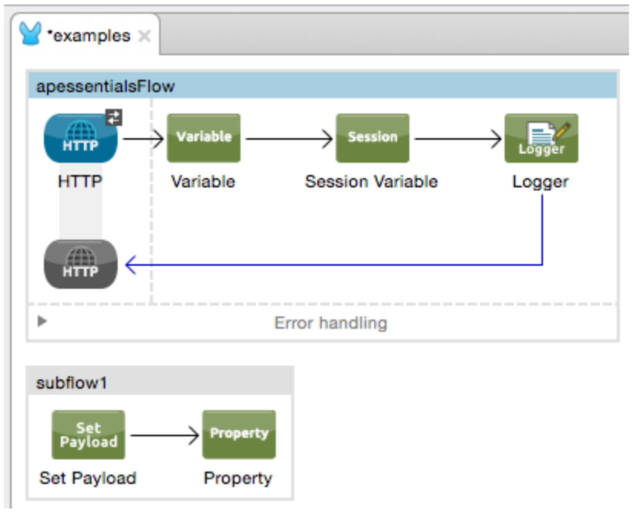
In this walkthrough, you will work with the apessentialsFlow in examples.xml. You will:

- Extract processors into separate subflows and flows.
- Use the Flow Reference component to reference other flows.
- Create and reference synchronous and asynchronous flows.
- Explore variable persistence through flows, subflows, and across transport barriers.



Create a subflow

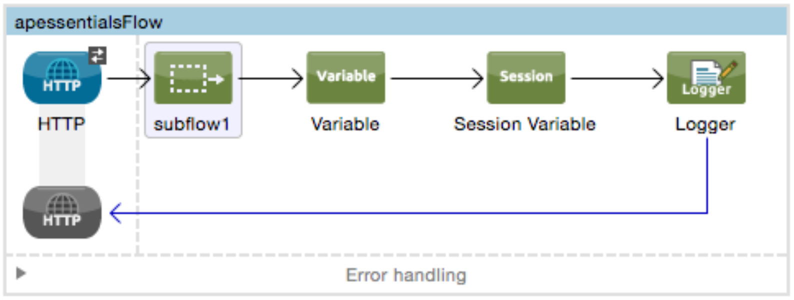
1. Open examples.xml in apessentials2.
2. Drag a Sub Flow scope element to the canvas.
3. Select the Set Payload and Property transformers in the apessentialsFlow and drag them into the subflow.
4. Change the name of the subflow to subflow1.



Reference a subflow

5. Drag a Flow Reference component from the palette and drop it into the apessentialsFlow between the HTTP Listener connector endpoint and the Variable transformer.

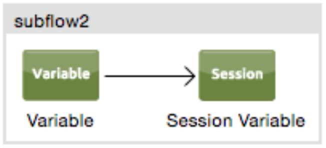
6. In the Flow Reference Properties view, set the flow name to subflow1.

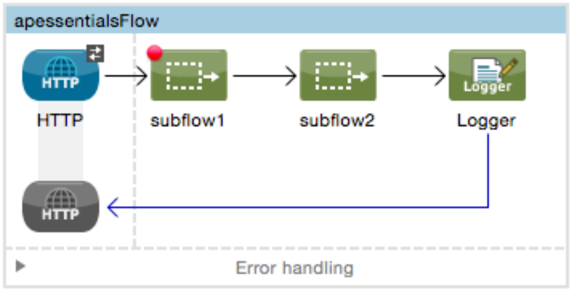


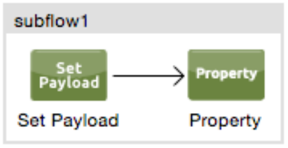
7. Add a breakpoint to the subflow1 Flow Reference.

Extract processors into a subflow

8. Right-click the Variable and Session Variable transformers and select Extract to > Sub Flow.
9. In the Extract Flow dialog box, set the flow name to subflow2.
10. Leave the target Mule configuration set to current and click OK.
11. Look at the new Flow Reference Properties view; the flow name should already be set to subflow2.



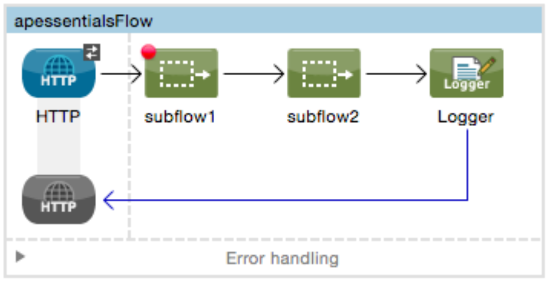


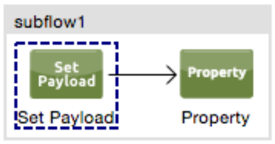


12. Drag subflow2 below subflow1.
13. Save the file.

Debug the application

14. Click the Debug button drop-down and select apessentials2 to just debug apessentials2.
15. Make a request to http://localhost:8082/?name=max&type=mule using your own query parameter values.
16. Step through the application, watching messages move into and out of the subflows.

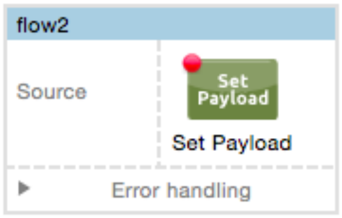




17. Make another request to http://localhost:8082/?name=max&type=mule using your own query parameter values.
18. Step through the application again, this time watching the values of the inbound properties, variables, outbound properties, and session variables.

Create a second flow

19. Drag a Flow scope element to the canvas.
20. Change the flow name to flow2.
21. Add a Set Payload transformer to the process section of the flow.
22. Add a breakpoint to it.

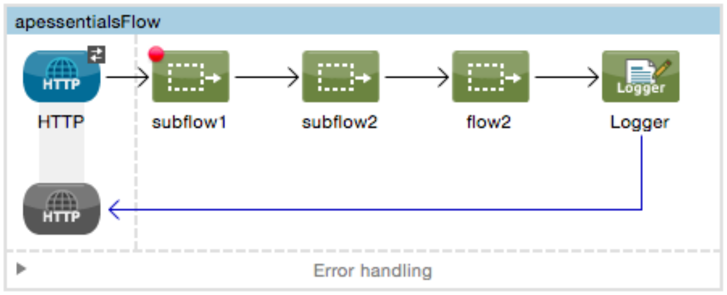


23. In the Set Payload Properties view, set the value to a string, like flow2.

Reference a flow

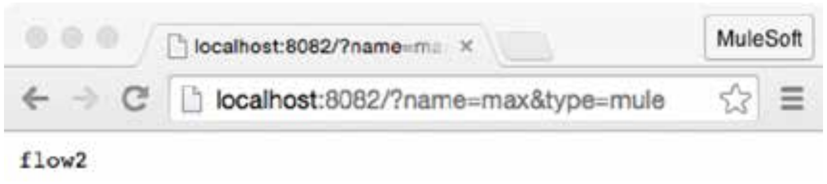
24. Drag a Flow Reference component from the palette and drop it into the apessentialsFlow between the subflow2 Flow Reference and the Logger.

25. In the Flow Reference Properties view, set the flow name to flow2.



Debug the application

26. Save the file to redeploy the application in debug mode.
27. Make a request to http://localhost:8082/?name=max&type=mule using your own query parameter values.
28. Step through the application, watching the flow move into and out of the second flow; at the end, you should see the new payload value set in the second flow returned.



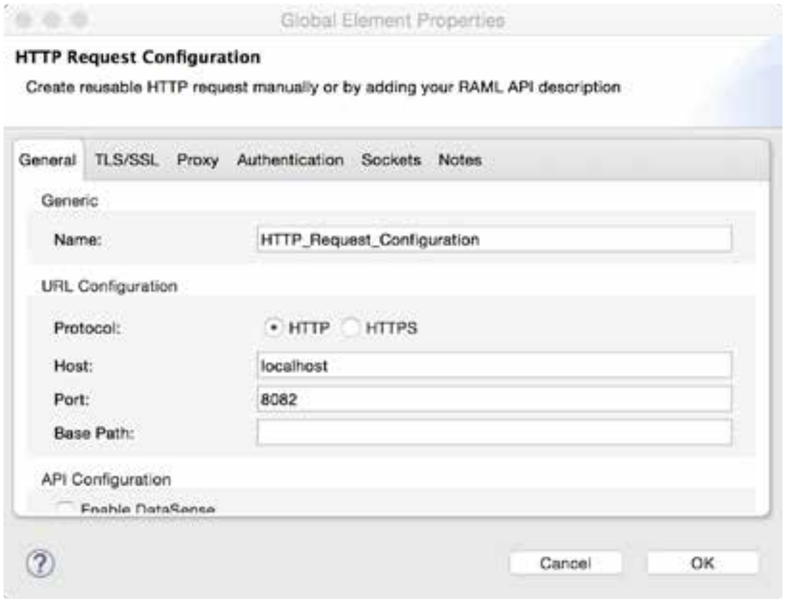
```
flow2
```

29. Make another request to http://localhost:8082/?name=max&type=mule using your own query parameter values.
30. Step through the application again, this time watching the values of the inbound properties, variables, outbound properties, and session variables.

Create a transport barrier for the second flow

31. In global.xml for apessentials2, switch to the Global Elements view.
32. Click Create.
33. In the Choose Global Type dialog box, select Connector Configuration > HTTP Request Configuration and click OK.

34. In the Global Elements dialog box, set the host to localhost, the port to 8082, and click OK.



HTTP Request Configuration

Create reusable HTTP request manually or by adding your RAML API description

Generic

Name: HTTP_Request_Configuration

URL Configuration

Protocol: (● HTTP) (○ HTTPS)

Host: localhost

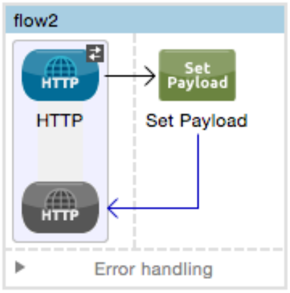
Port: 8082

Base Path:

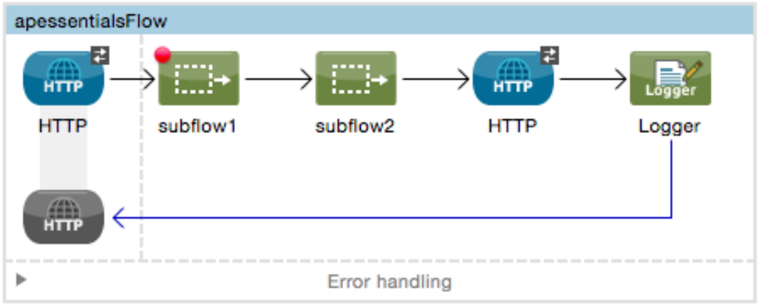
API Configuration

Enable DataSense

35. In examples.xml, drag an HTTP connector into the Source section of flow2.
36. In the HTTP Properties view, set the connector configuration to the existing HTTP_Listener_Configuration.
37. Set the path to /flow2 and the allowed methods to GET.



38. Add a second HTTP connector after the flow2 reference in apessentialsFlow.
39. Delete the flow2 reference.
40. In the HTTP Properties view, set the connector configuration to HTTP_Request_Configuration.
41. Set the path to /flow2 and the method to GET.



Debug the application

42. Add a breakpoint to the Logger.
43. Save all the files to redeploy the application.
44. Make another request to http://localhost:8082/?name=max&type=mule using your own query parameter values.
45. Step through the application again, pressing the Resume button to move into flow2.
46. Watch the values of the inbound properties, outbound properties, flow variables, and session variables as you move into flow2 and then back to apessentialsFlow.

    Note: Ignore any TimeoutException you may get.

    Note: Session variables are persisted across some but not all transport barriers. As you see here, they are not propagated across the HTTP transport barrier.