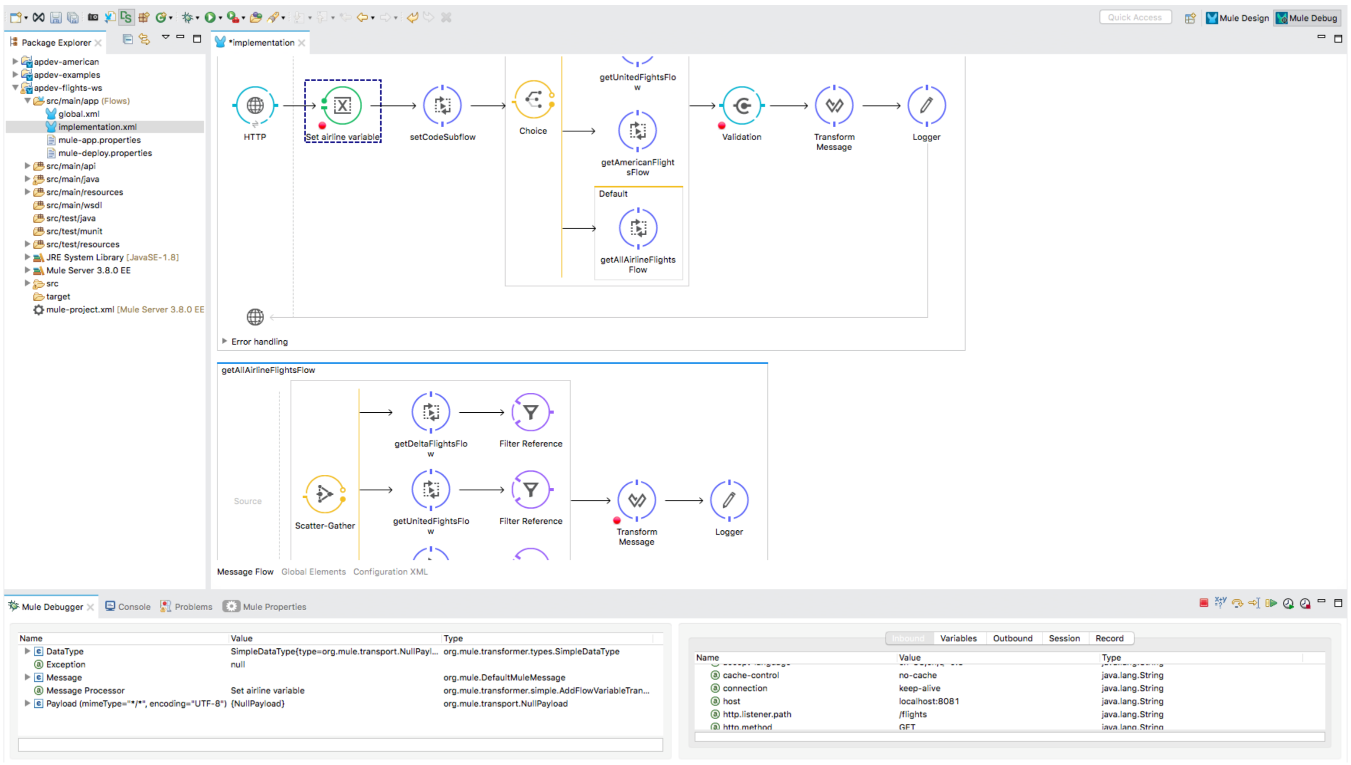
PART 2: Building Applications with Anypoint Studio



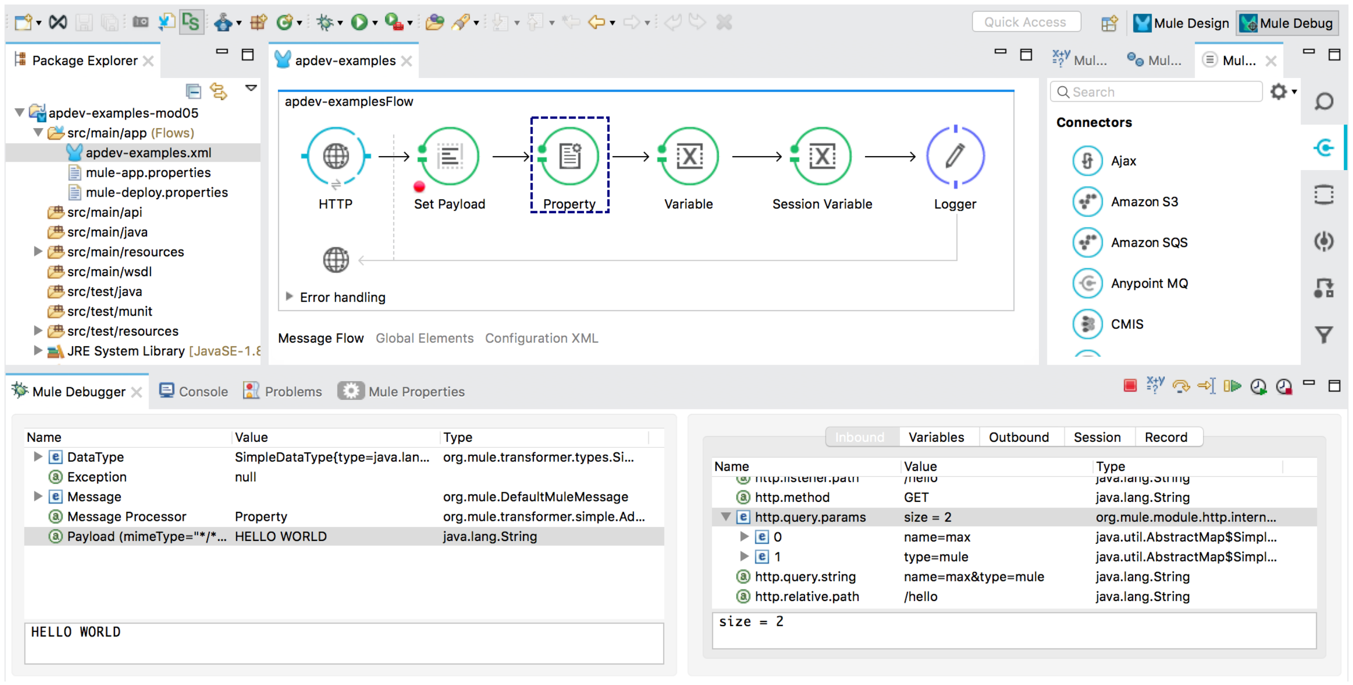
Name	Value	Type
DataType	SimpleDataType{type=org.mule.transport.NullPayl...	org.mule.transformer.types.SimpleDataType
Exception	null	
Message		org.mule.DefaultMuleMessage
Message Processor	Set airline variable	org.mule.transformer.simple.AddFlowVariableTran...
Payload (mimeType="*/*", encoding="UTF-8")	{NullPayload}	org.mule.transport.NullPayload

Name	Value	Type
cache-control	no-cache	java.lang.String
connection	keep-alive	java.lang.String
host	localhost:8081	java.lang.String
http.listener.path	/flights	java.lang.String
http.method	GET	java.lang.String

Objectives:

- Debug Mule applications.
- Read and write message payloads, properties, and variables using the Mule Expression Language.
- Structure Mule applications using flows, subflows, in-memory message queues, properties files, and configuration files.
- Connect to web services, SaaS applications, files, polled resources, JMS queues, and more.
- Route, filter, and validate messages and handle message exceptions.
- Write DataWeave expressions for more complicated transformations.
- Process individual records in a collection and synchronize data in databases to SaaS applications.

Module 5: Accessing and Modifying Mule Messages



Name	Value	Type
DataType	SimpleDataType{type=java.lan...	org.mule.transformer.types.Si...
Exception	null	
Message		org.mule.DefaultMuleMessage
Message Processor	Property	org.mule.transformer.simple.Ad...
Payload (mimeType="*/*...	HELLO WORLD	java.lang.String

```
HELLO WORLD
```

Name	Value	Type
http.listener.path	/hello	java.lang.String
http.method	GET	java.lang.String
http.query.params	size = 2	org.mule.module.http.intern...
0	name=max	java.util.AbstractMap$Simpl...
1	type=mule	java.util.AbstractMap$Simpl...
http.query.string	name=max&type=mule	java.lang.String
http.relative.path	/hello	java.lang.String

```
size = 2
```

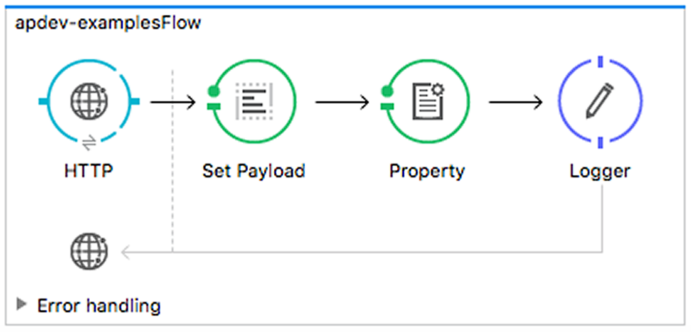
Objectives:

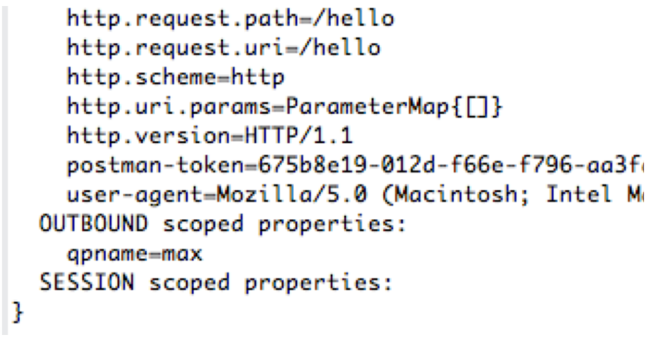
- Log message data.
- Debug Mule applications.
- Read and write message properties.
- Write expressions with Mule Expression Language (MEL).
- Create variables.

Walkthrough 5-1: Set and log message data

In this walkthrough, you create a new project to use in the next two modules for learning about Mule messages and Mule applications. You will:

- Create a new Mule project with an HTTP Listener endpoint.
- Set the message payload and an outbound property.
- Use a Logger to view message data in the Anypoint Studio console.
- Review message data in Postman.

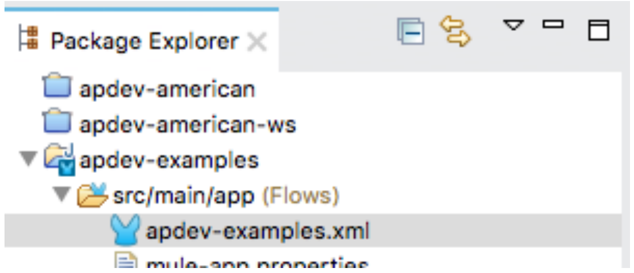




```
    http.request.path=/hello
    http.request.uri=/hello
    http.scheme=http
    http.uri.params=ParameterMap{[]}
    http.version=HTTP/1.1
    postman-token=675b8e19-012d-f66e-f796-aa3f
    user-agent=Mozilla/5.0 (Macintosh; Intel M
  OUTBOUND scoped properties:
    qpname=max
  SESSION scoped properties:
}
```

Create a new Mule project

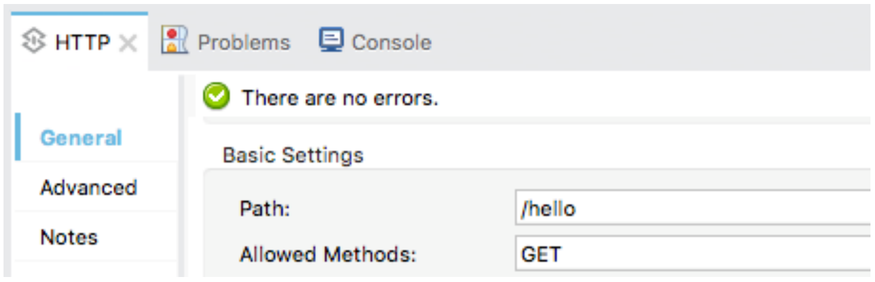
1. Return to Anypoint Studio.
2. Right-click apdev-american-ws and select Close Project.
3. Select File > New > Mule Project.
4. Set the Project Name to apdev-examples and click Finish.



Create an HTTP connector endpoint to receive requests

5. Drag an HTTP connector from the Mule Palette to the canvas.
6. In the HTTP properties view, click the Add button next to connector configuration.
7. In the Global Element Properties dialog box, look at the default values and click OK.
8. In the HTTP properties view, set the path to /hello.

9. Set the allowed methods to GET.



There are no errors.

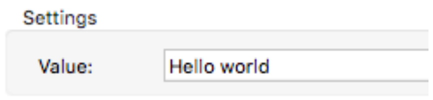
Basic Settings

Path: /hello

Allowed Methods: GET

Set the message payload

10. Drag a Set Payload transformer from the Mule Palette into the process section of the flow.
11. In the Set Payload properties view, set the value field to Hello world.

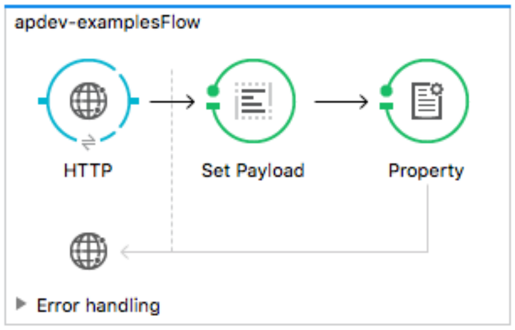


Settings

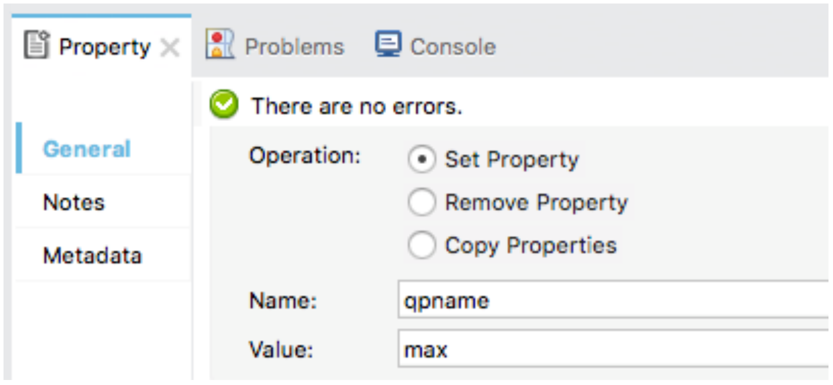
Value: Hello world

Set an outbound message property

12. Drag a Property transformer from the Mule Palette into the flow.



13. In the Properties view for the Property transformer, select Set Property.
14. Set the name to qpname and the value to max.



There are no errors.

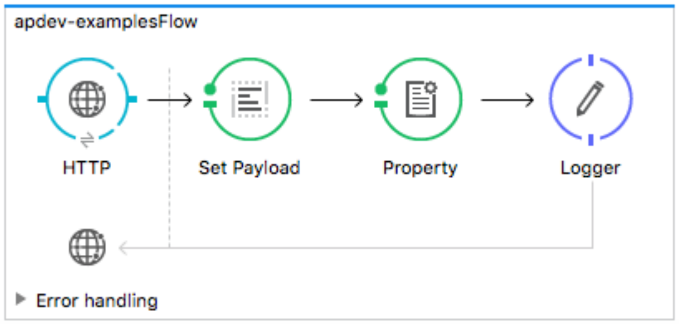
Operation: [✓ Set Property] Remove Property Copy Properties

Name: qpname

Value: max

Add a Logger

15. Drag a Logger component from the Mule Palette and drop it at the end of the flow.

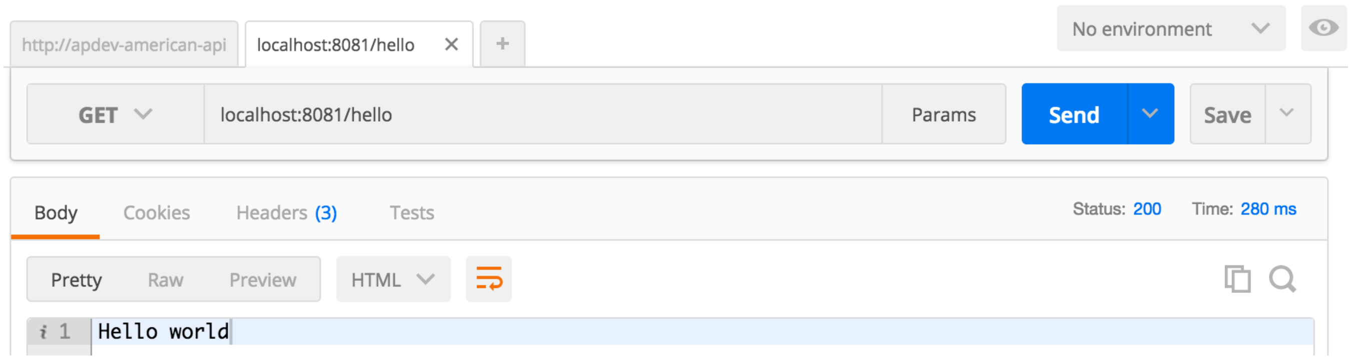


Run the application and review message data

16. Run the project.
17. Return to Postman and click the button to create a new tab.

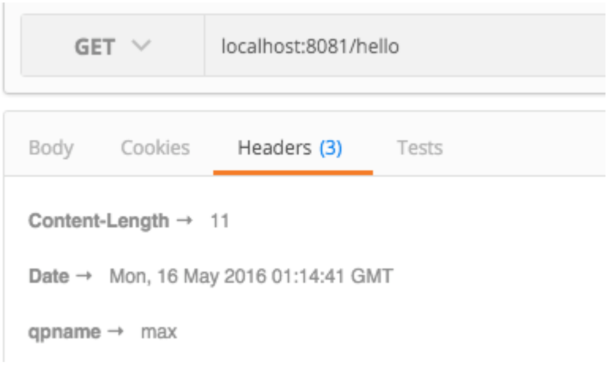
    Note: You are adding a new tab so that you can keep the request to your American API saved in the another tab for later use. Optionally, you could also save it to a Postman collection.

18. In the new tab, make a GET request to http://localhost:8081/hello; you should see Hello world displayed.



```
Hello world
```

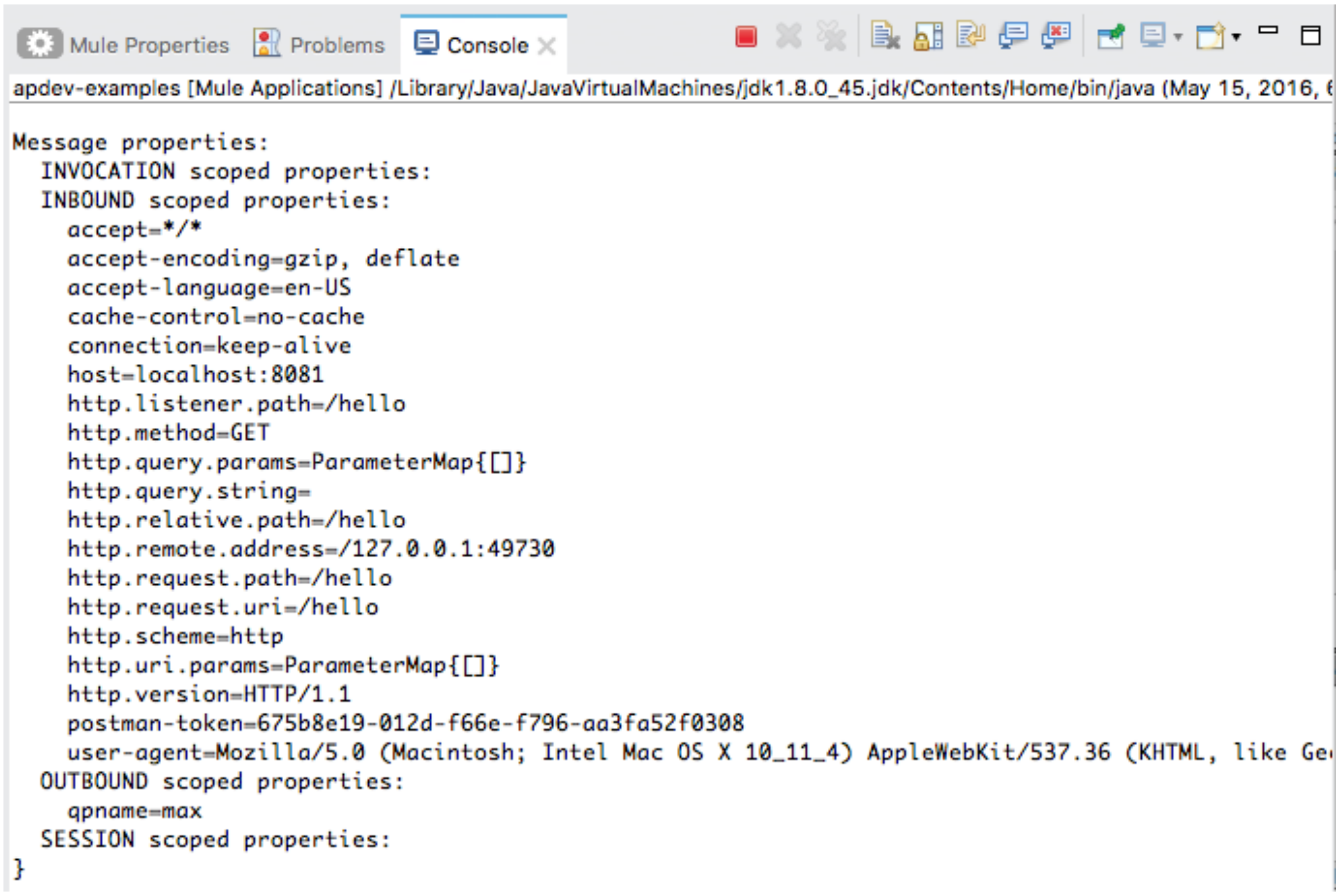
19. Click the Headers link in the response; you should see the qpname property.



```
Content-Length → 11
Date → Mon, 16 May 2016 01:14:41 GMT
qpname → max
```

View message data in the Anypoint Studio console

20. Return to Anypoint Studio and look at the console.
21. Locate the data displayed by using the Logger.
22. Find where the data type of the payload is specified.
23. Review the message inbound properties.
24. Review the message outbound properties.
25. Locate the other two scopes of properties that have no values set.



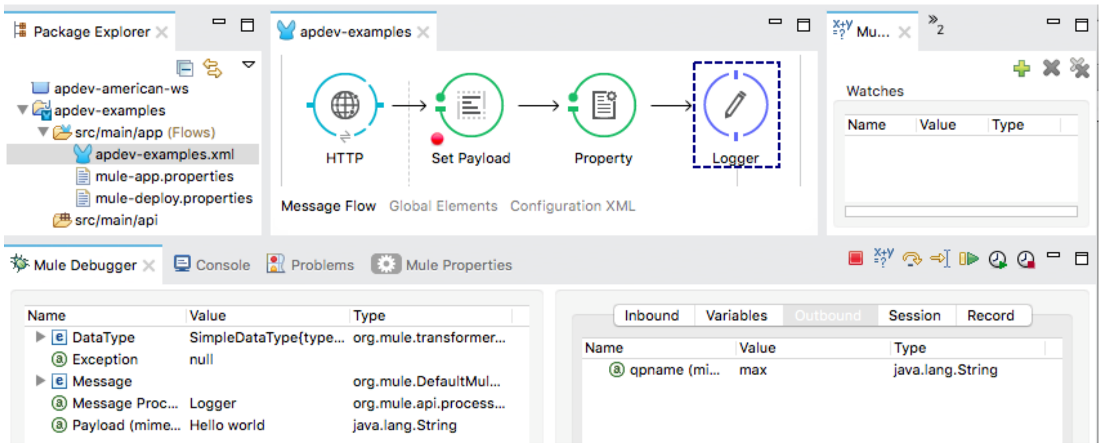
```
apdev-examples [Mule Applications] /Library/Java/JavaVirtualMachines/jdk1.8.0_45.jdk/Contents/Home/bin/java (May 15, 2016, 6

Message properties:
  INVOCATION scoped properties:
  INBOUND scoped properties:
    accept=*/*
    accept-encoding=gzip, deflate
    accept-language=en-US
    cache-control=no-cache
    connection=keep-alive
    host=localhost:8081
    http.listener.path=/hello
    http.method=GET
    http.query.params=ParameterMap{[]}
    http.query.string=
    http.relative.path=/hello
    http.remote.address=/127.0.0.1:49730
    http.request.path=/hello
    http.request.uri=/hello
    http.scheme=http
    http.uri.params=ParameterMap{[]}
    http.version=HTTP/1.1
    postman-token=675b8e19-012d-f66e-f796-aa3fa52f0308
    user-agent=Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_4) AppleWebKit/537.36 (KHTML, like Ge
  OUTBOUND scoped properties:
    qpname=max
  SESSION scoped properties:
}
```

26. Stop the project.

Walkthrough 5-2: Debug a Mule application

In this walkthrough, you debug and step through the code in a Mule application. You will:

- Locate the port used by the Mule Debugger.
- Add a breakpoint, debug an application, and step through the code.
- Use the Mule Debugger to view message properties.
- Pass query parameters to a request and locate them in the Mule Debugger.

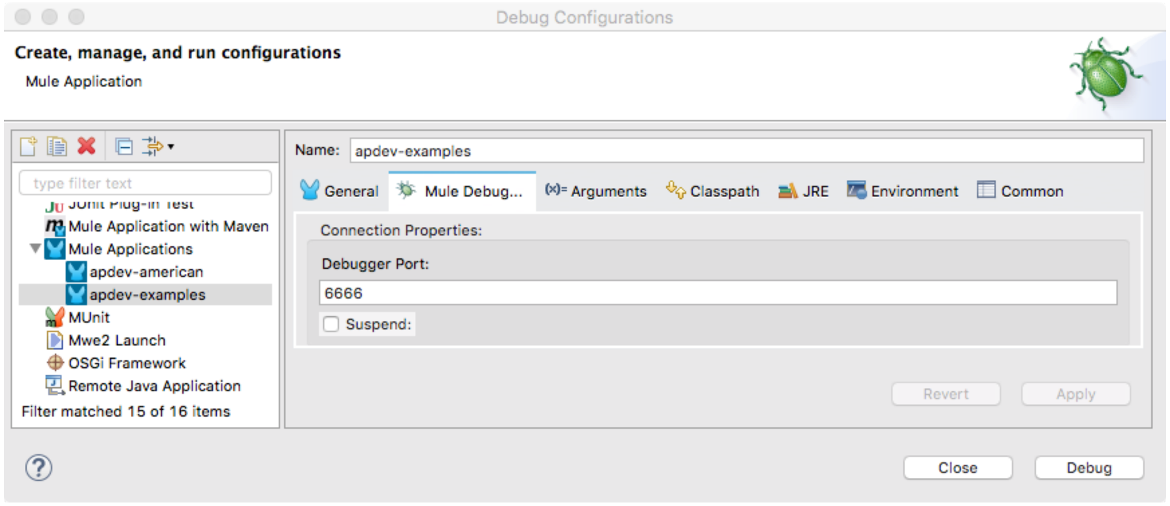


Name	Value	Type
DataType	SimpleDataType{type...	org.mule.transformer...
Exception	null	
Message		org.mule.DefaultMul...
Message Proc...	Logger	org.mule.api.process...
Payload (mime...	Hello world	java.lang.String

Name	Value	Type
qpname (mi...	max	java.lang.String

Locate the port used by the Mule Debugger

1. Return to the apdev-examples project in Anypoint Studio.
2. In the main menu bar, select Run > Debug Configurations.
3. Click apdev-examples in the left menu bar under Mule Applications; you should see that the apdev-examples project is selected to launch.
4. Click the Mule Debug tab; you should see the debugger port is set by default to 6666 for the project.



Name: apdev-examples

Connection Properties:

Debugger Port: 6666

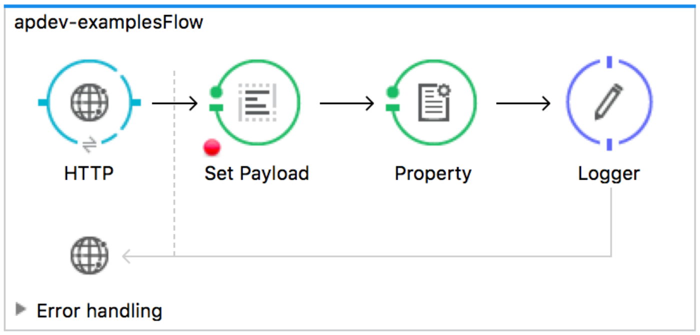
Suspend: [ ]

Note: If you know you have another program using port 6666 like McAfee on Windows, change this to a different value. Otherwise, you can test the debugger first and come back and change the value here later if there is a conflict.

5. Click Close.

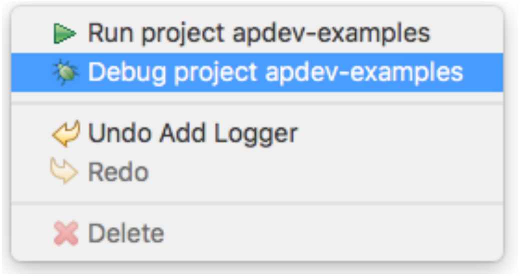
Debug the application

6. Right-click the Set Payload component and select Toggle breakpoint.

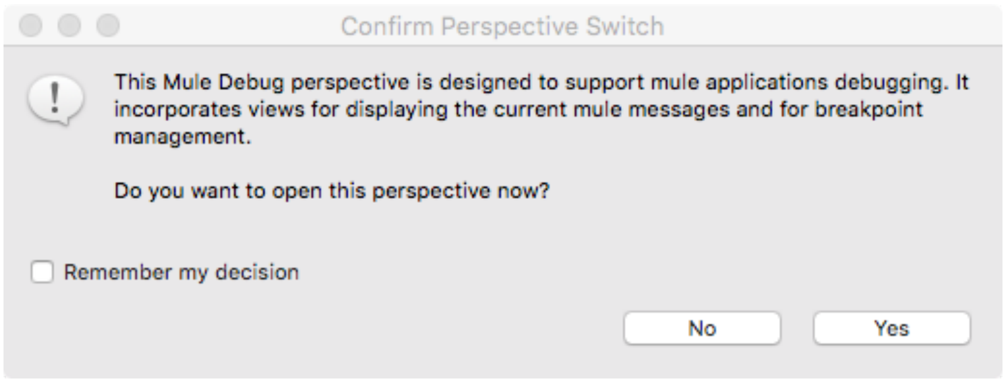


7. Right-click in the canvas and select Debug project apdev-examples.

Note: You can also select Run > Debug or click the Debug button in the main menu bar.



8. If you get an Accept incoming network connections dialog box, click Allow.
9. If you get a Confirm Perspective Switch dialog box, select Remember my decision and click Yes.


Confirm Perspective Switch

This Mule Debug perspective is designed to support mule applications debugging. It incorporates views for displaying the current mule messages and for breakpoint management.

Do you want to open this perspective now?

☐ Remember my decision

No Yes

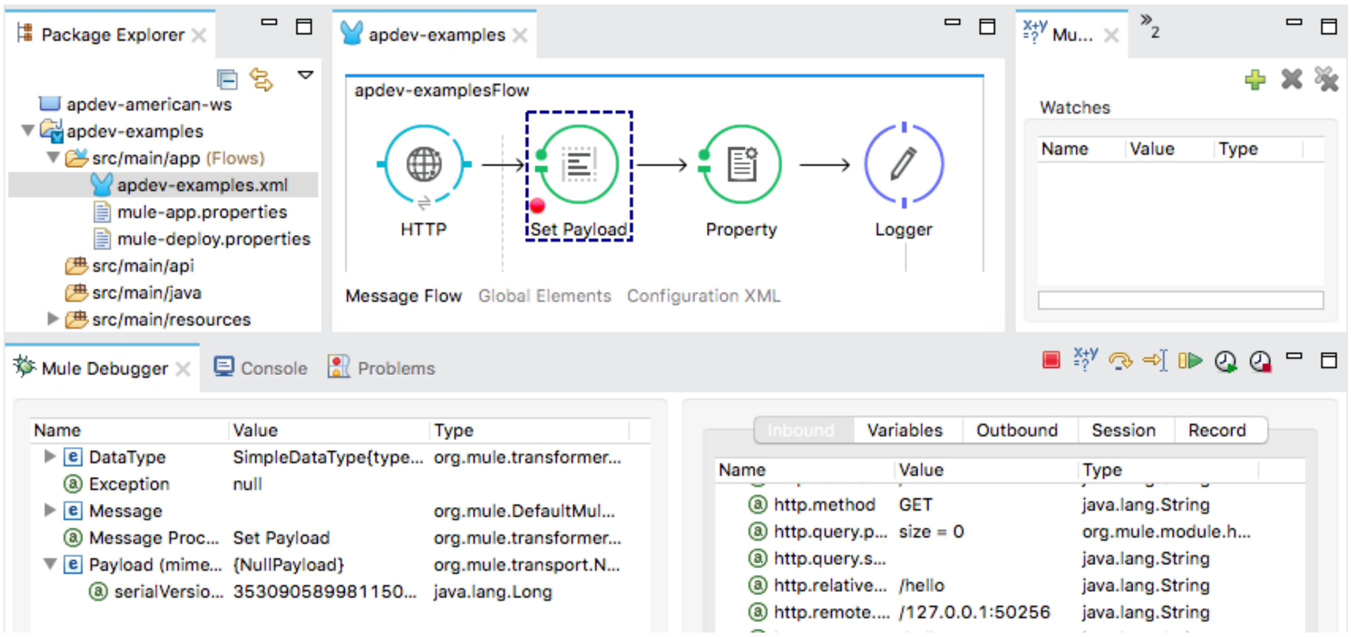
10. In the Debug perspective, close the MUnit view.

    Note: You will not use MUnit in this course. MUnit is covered in the Anypoint Platform Development: Advanced course.

11. Look at the console and wait until the application starts.
12. In Postman, make another request to http://localhost:8081/hello.

View message properties and variables in the Mule Debugger

13. Return to Anypoint Studio and locate the Mule Debugger view.
14. Drill-down and explore the variables on the left side of the Mule Debugger.
15. Look at the values of the inbound properties on the right side of the Mule Debugger.
16. Select each of the other tabs: Variables, Outbound, Session, Record; you should not see values for any other properties or variables.

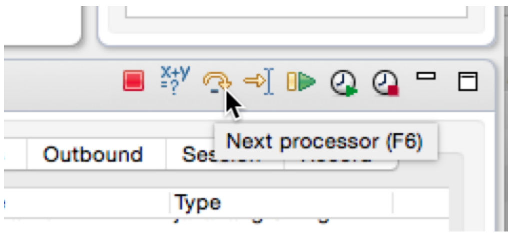


Name	Value	Type
DataType	SimpleDataType{type...	org.mule.transformer...
Exception	null	
Message		org.mule.DefaultMul...
Message Proc...	Set Payload	org.mule.transformer...
Payload (mime...	{NullPayload}	org.mule.transport.N...
serialVersio...	353090589981150...	java.lang.Long

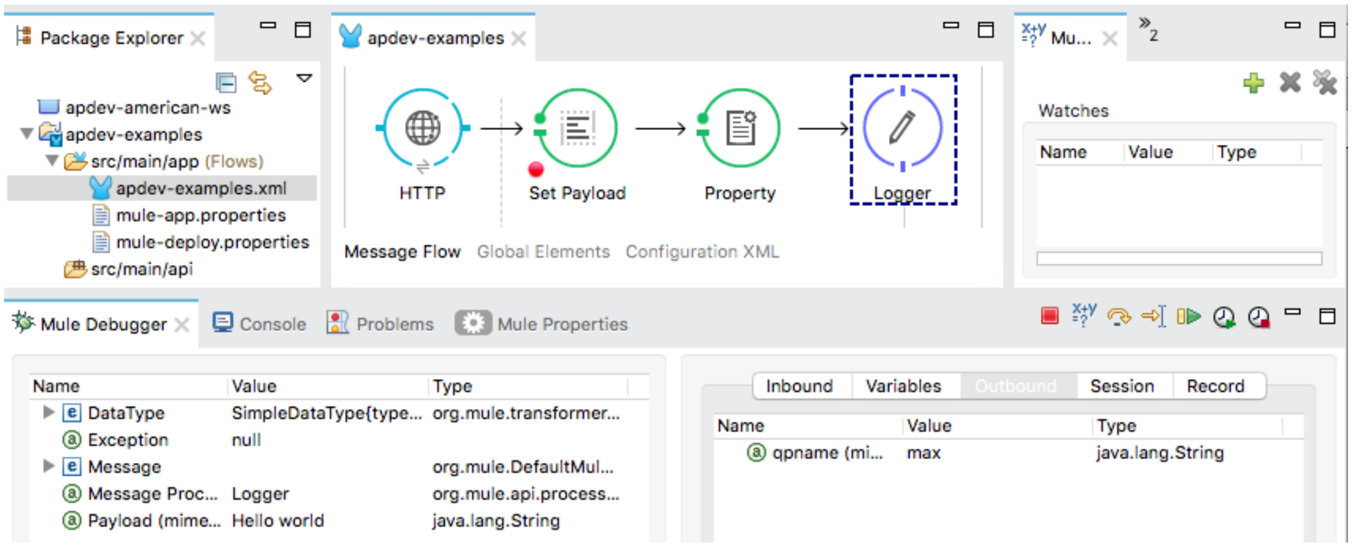
Name	Value	Type
http.method	GET	java.lang.String
http.query.p...	size = 0	org.mule.module.h...
http.query.s...		java.lang.String
http.relative...	/hello	java.lang.String
http.remote....	/127.0.0.1:50256	java.lang.String

Step through the application

17. Click the Next processor button.



18. Look at the new value of the payload; it should be Hello world.
19. Look at the inbound and outbound properties; you should see they have not changed.
20. Click the Next processor button; you should now see the outbound property qpname.



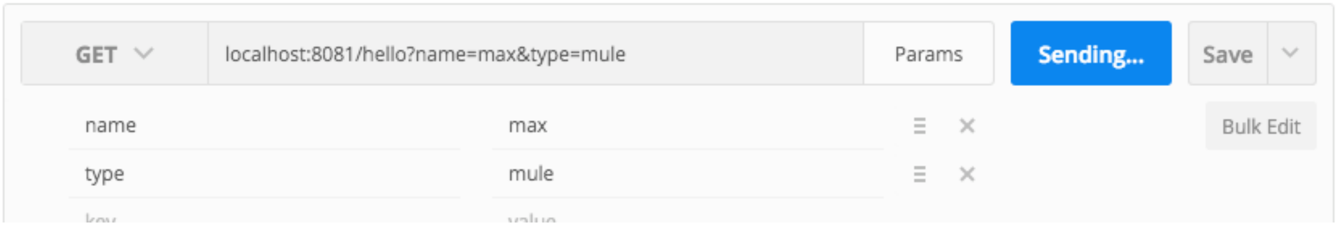
Name	Value	Type
DataType	SimpleDataType{type...	org.mule.transformer...
Exception	null	
Message		org.mule.DefaultMul...
Message Proc...	Logger	org.mule.api.process...
Payload (mime...	Hello world	java.lang.String

Name	Value	Type
qpname (mi...	max	java.lang.String

21. Step through the rest of the application.

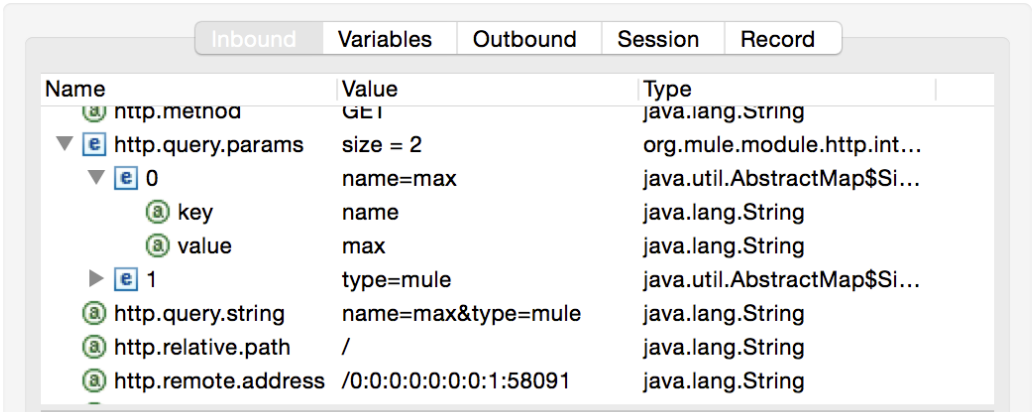
Send query parameters with a request

22. Return to Postman and add a parameter with a key of name and a value of max.
23. Add a second key / value pair of type and mule.
24. Click Send.



name	max
type	mule

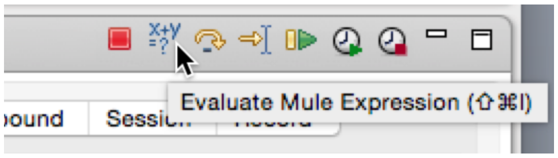
25. Return to the Mule Debugger view and locate your query parameters.



Name	Value	Type
http.method	GET	java.lang.String
http.query.params	size = 2	org.mule.module.http.int…
0	name=max	java.util.AbstractMap$Si…
key	name	java.lang.String
value	max	java.lang.String
1	type=mule	java.util.AbstractMap$Si…
http.query.string	name=max&type=mule	java.lang.String
http.relative.path	/	java.lang.String
http.remote.address	/0:0:0:0:0:0:0:1:58091	java.lang.String

Use the Mule Expression evaluator

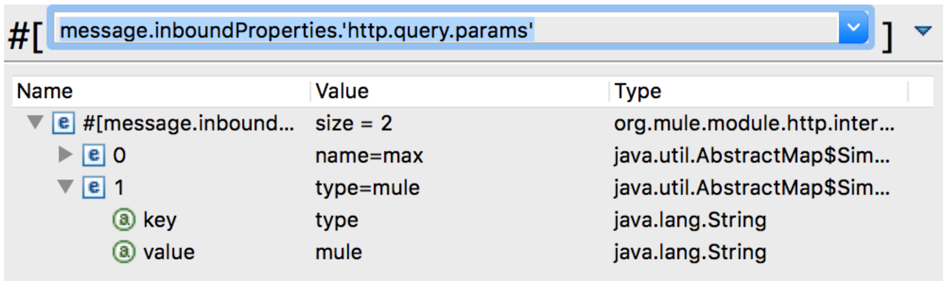
26. Click the Evaluate Mule Expression button.



27. Enter the following expression and press the Enter key.

```
#[message.inboundProperties.'http.query.params']
```

28. Expand the results; you should see the query parameters.



#[message.inboundProperties.'http.query.params']

Name	Value	Type
#[message.inbound…	size = 2	org.mule.module.http.inter…
0	name=max	java.util.AbstractMap$Sim…
1	type=mule	java.util.AbstractMap$Sim…
key	type	java.lang.String
value	mule	java.lang.String

29. Click anywhere outside the expression window to close it.

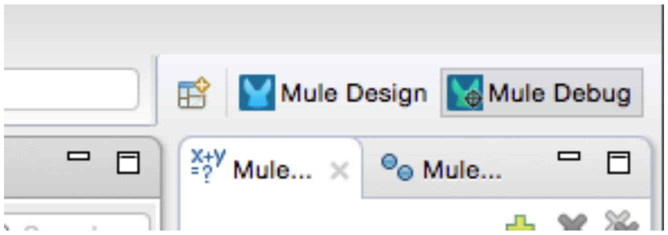
30. Click the Resume button to execute the rest of the flow.

31. Stop the project.

MuleSoft

Switch perspectives

32. Click the Mule Design tab in the upper-right corner of Anypoint Studio to switch perspectives.

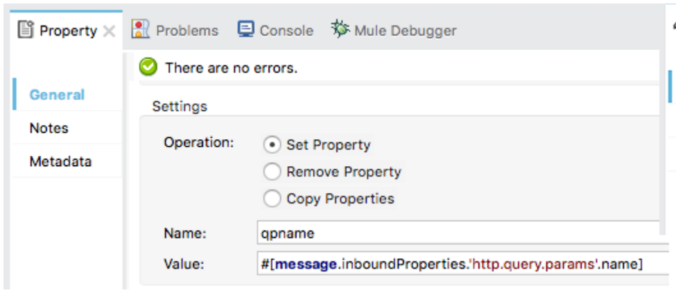


Note: In Eclipse, a perspective is a specific arrangement of views in specific locations. You can rearrange the perspective by dragging and dropping tabs and views to different locations. Use the Window menu in the main menu bar to save and reset perspectives.

Walkthrough 5-3: Read and write message properties using MEL expressions

In this walkthrough, you manipulate message properties. You will:

- Use an expression to set the payload.
- Use an expression to display specific info to the console.
- Use an expression to set an outbound property.
- Use an expression to read an outbound property.



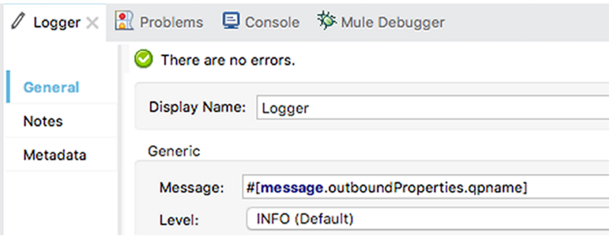
There are no errors.

Settings

Operation: [✓ Set Property] Remove Property Copy Properties

Name: qpname

Value: #[message.inboundProperties.'http.query.params'.name]



There are no errors.

Display Name: Logger

Generic

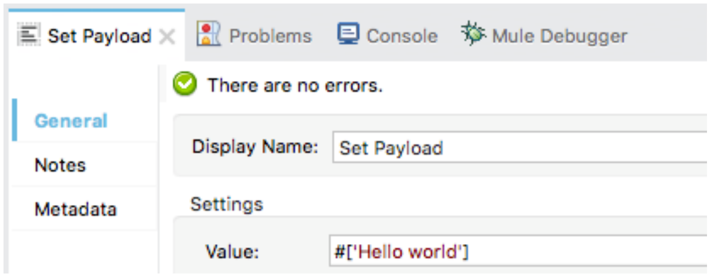
Message: #[message.outboundProperties.qpname]

Level: INFO (Default)

Use an expression to set the payload

1. Return to apdev-examples.xml.
2. Navigate to the Properties view for the Set Payload transformer.
3. Change the value to an expression.

```
#['Hello world']
```



There are no errors.

Display Name: Set Payload

Settings

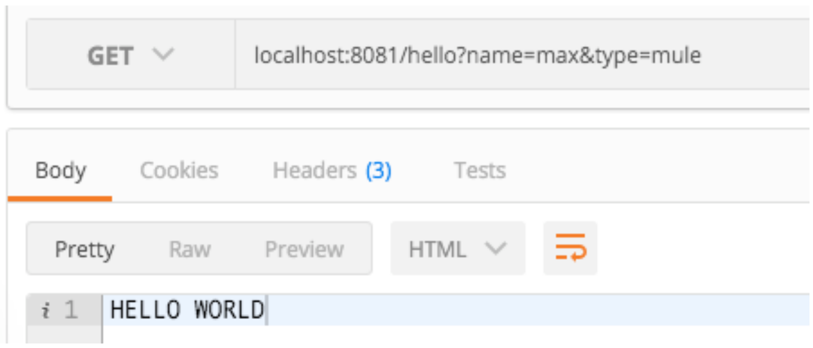
Value: #['Hello world']

4. Run the project.
5. In Postman, make a request to http://localhost:8081/hello with the query parameters; the application should work as before.
6. In Anypoint Studio, return to the Set Payload expression and use autocomplete to use the toUpperCase() method to return the value in upper case.

```
#['Hello world'.toUpperCase()]
```

Note: Press Ctrl+Space to trigger autocomplete if it does not appear.

7. Click the Apply Changes button in the upper-right corner of the properties view to redeploy the application.
8. In Postman, send the same request; the return string should now be in upper case.

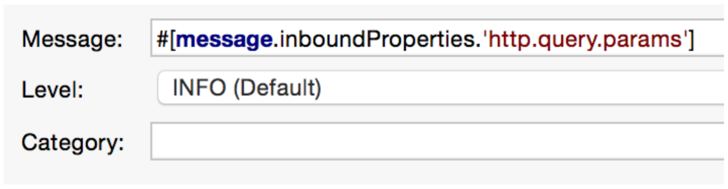


```
HELLO WORLD
```

Use an expression to display specific info to the console

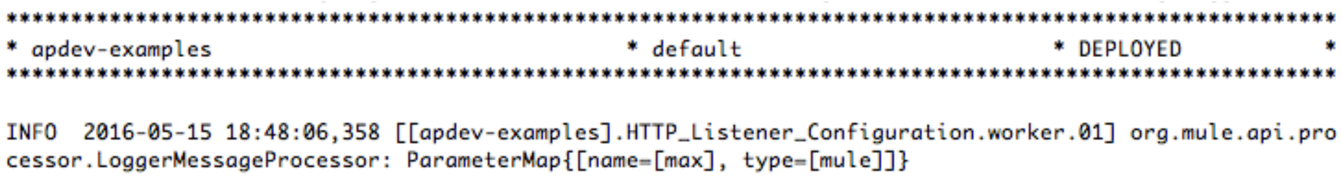
9. In Anypoint Studio, navigate to the Properties view for the Logger component.
10. Set the message to display the http.query.params property of the message inbound properties.

```
#[message.inboundProperties.'http.query.params']
```



Message: #[message.inboundProperties.'http.query.params']
Level: INFO (Default)
Category:

11. Click Apply Changes to redeploy the application.
12. In Postman, send the same request.
13. Return to the Anypoint Studio console; you should see a ParameterMap object listed.



```
**********************************************************************************
* apdev-examples                         * default                    * DEPLOYED        *
**********************************************************************************

INFO  2016-05-15 18:48:06,358 [[apdev-examples].HTTP_Listener_Configuration.worker.01] org.mule.api.pro
cessor.LoggerMessageProcessor: ParameterMap{[name=[max], type=[mule]]}
```

14. Modify the Logger to display one of the query parameters.

```
#[message.inboundProperties.'http.query.params'.name]
```

15. Click Apply Changes to redeploy the application.
16. In Postman, send the same request.

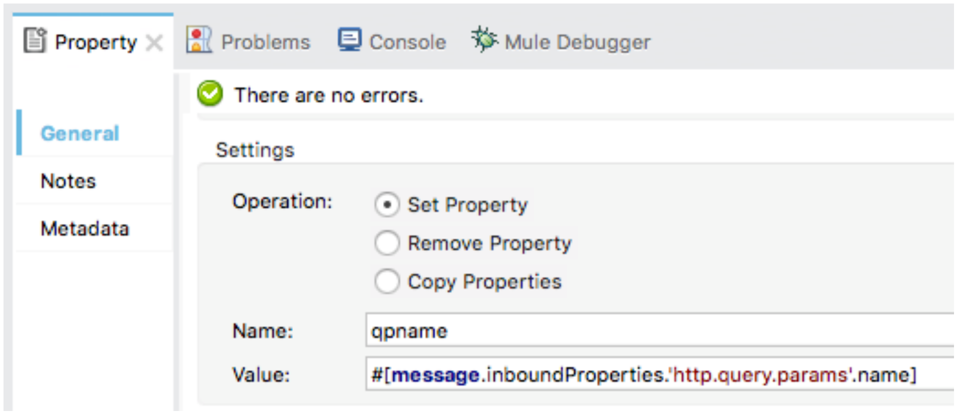
17. Return to the console; you should now see the value of the parameter displayed.

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
INFO  2016-05-15 18:48:41,451 [[apdev-examples].HTTP_Listener_Configuration.worker.01] org.mule.api.pro
cessor.LoggerMessageProcessor: max
```

Use an expression to set an outbound property

18. In the Properties view for the Property transformer, change the value to an expression that evaluates the name query parameter.

    `#[message.inboundProperties.'http.query.params'.name]`



There are no errors.

Settings

Operation: [✓ Set Property] Remove Property Copy Properties

Name: qpname

Value: #[message.inboundProperties.'http.query.params'.name]

19. Modify the Logger to display the value of this outbound property.

    `#[message.outboundProperties.qpname]`

20. Click Apply Changes to redeploy the application.
21. In Postman, send the same request.
22. Return to the console; you should see the value of your variable displayed.

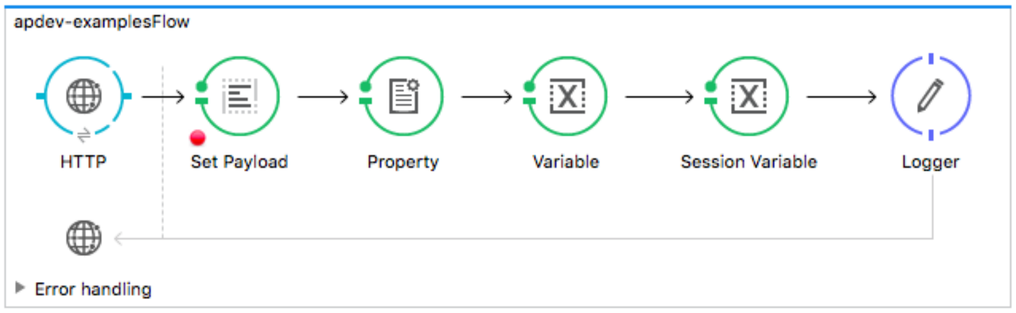
```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
INFO  2016-05-15 18:50:12,742 [[apdev-examples].HTTP_Listener_Configuration.worker.01] org.mule.api.pro
cessor.LoggerMessageProcessor: max
```

23. Stop the project.

Walkthrough 5-4: Read and write variables

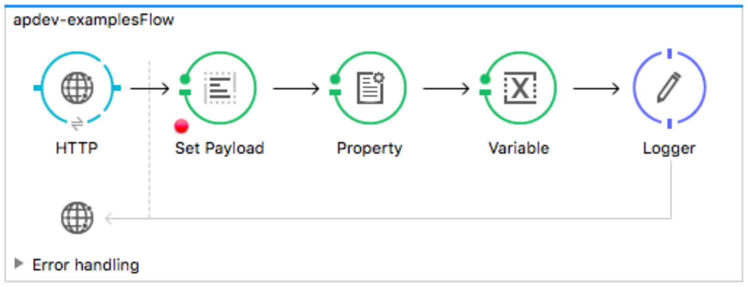
In this walkthrough, you create flow and session variables. You will:

- Use the Variable transformer to create a flow variable.
- Use the Session transformer to create a session variable.
- Use the Mule Debugger to see their values.



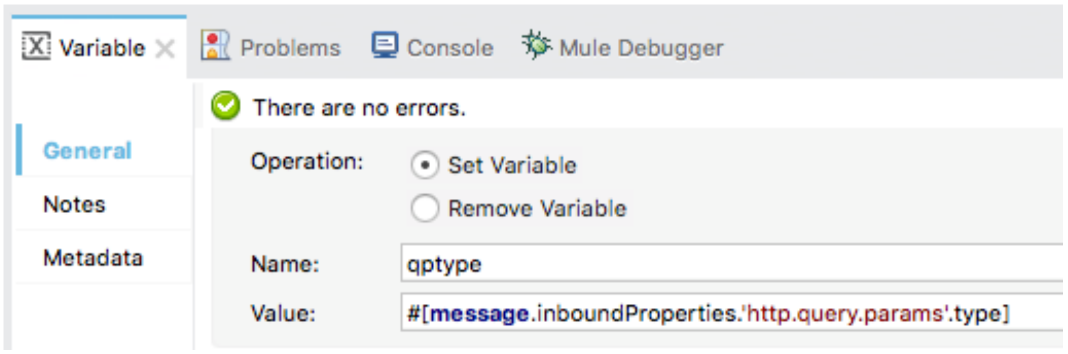
Create a flow variable

1. Return to apdev-examples.xml.
2. Add a Variable transformer between the Property and Logger processors.



3. In the Variable properties view, select Set Variable.
4. Set the name to qptype and the value to your second query parameter, type.

```
#[message.inboundProperties.'http.query.params'.type]
```



There are no errors.

Operation: [✓ Set Variable] Remove Variable

Name: qptype

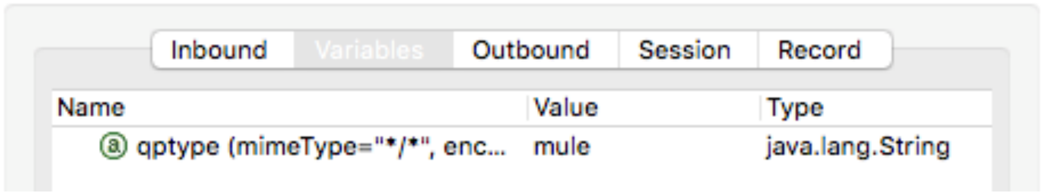
Value: #[message.inboundProperties.'http.query.params'.type]

5. Modify the Logger to also display the value of this new variable.

```
Name: #[message.outboundProperties.qpname] Type: #[flowVars.qptype]
```

Debug the application

6. Debug the project.
7. In Postman, send the same request with the parameters.
8. In the Mule Debugger, select the Variables tab.
9. Step to the Logger; you should see your new flow variable.



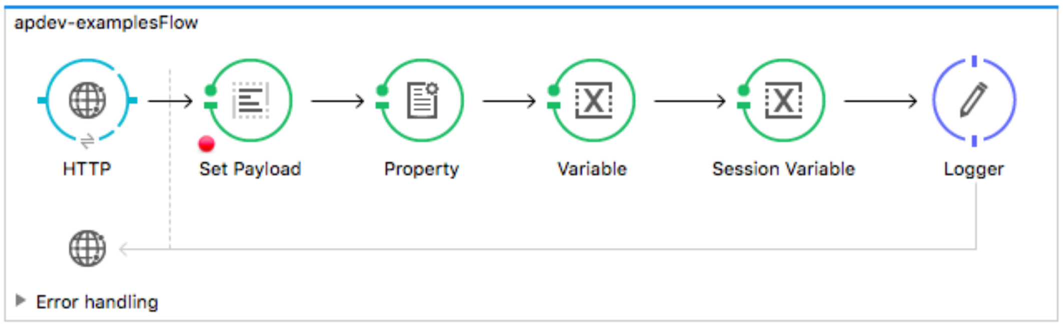
Name	Value	Type
qptype (mimeType="*/*", enc...	mule	java.lang.String

10. Click the Resume button.
11. Look at the console; you should see the value of your outbound property and the value of your new flow variable displayed.

```
INFO  2016-05-15 18:57:51,141 [[apdev-examples].HTTP_Listener_Configuration.worker.01] org.mule.api.pro
cessor.LoggerMessageProcessor: Name: max Type: mule
```

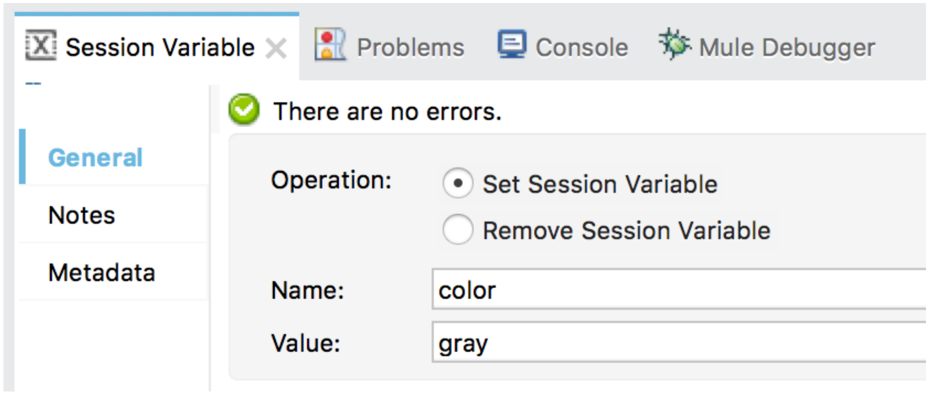
Create a session variable

12. Add a Session Variable transformer between the Variable and Logger processors.



13. In the Session Variable Properties view, select Set Session Variable.

14. Set the name to color and give it any value, static or dynamic.



There are no errors.

Operation: (● Set Session Variable) ○ Remove Session Variable

Name: color

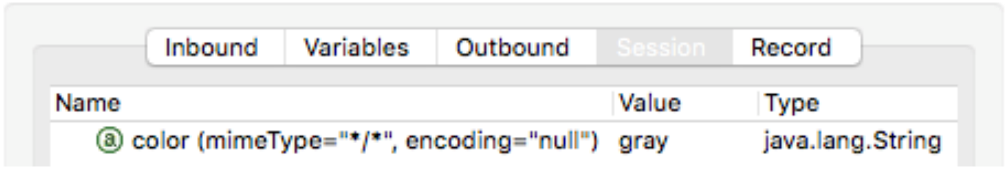
Value: gray

15. Modify the Logger to also display the session variable.

```
Name: #[message.outboundProperties.qpname] Type: #[flowVars.qptype]
Color: #[sessionVars.color]
```

Debug the application

16. Click Apply Changes to redeploy the application in debug mode.
17. In Postman, send the same request.
18. In the Mule Debugger, select the Session tab.
19. Step to the Logger; you should see your new session variable.



Name	Value	Type
color (mimeType="*/*", encoding="null")	gray	java.lang.String

20. Click the Resume button.
21. Look at the console; you should see the value of your session variable displayed.

```
INFO  2016-05-15 19:00:03,872 [[apdev-examples].HTTP_Listener_Configuration.worker.01] org.mule.api.pro
cessor.LoggerMessageProcessor: Name: max Type: mule Color: gray
```

22. Stop the project.

Note: You will explore the persistence of these variables in the next module.