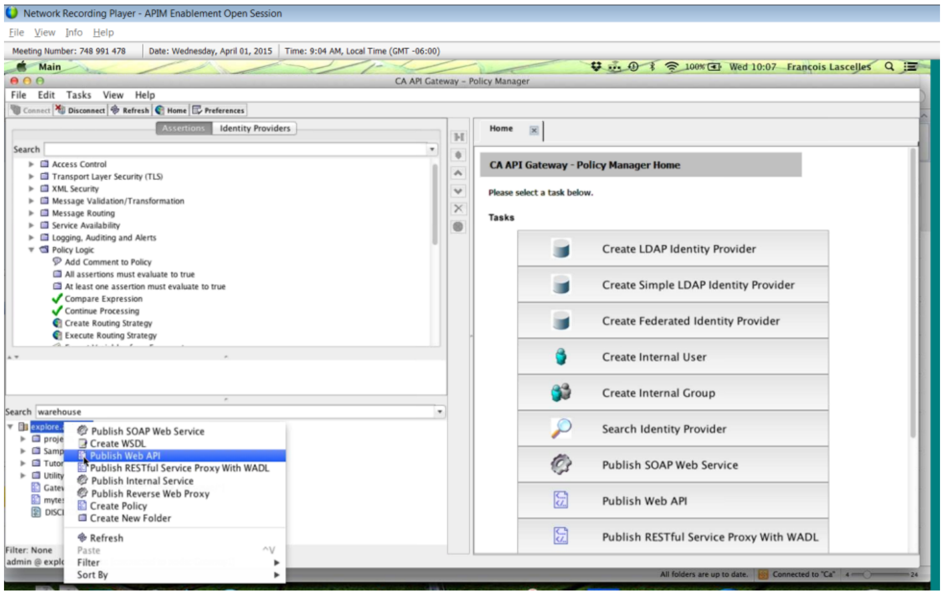
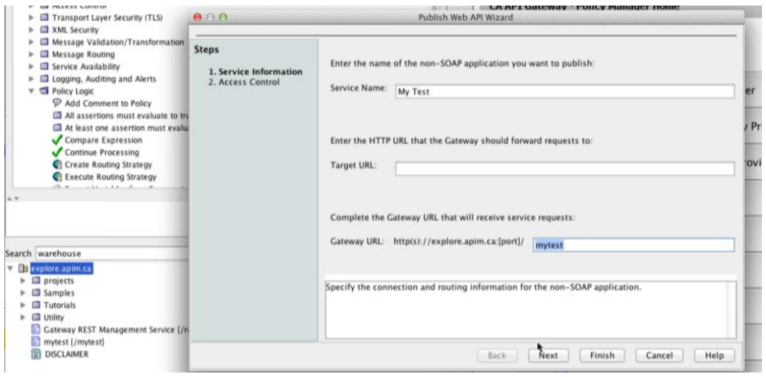
Publish a web API, validate, create a new policy, and tailor the response/error messages.

Step 1, right click server name, select Publish Web API.


Network Recording Player - APIM Enablement Open Session
Meeting Number: 748 991 478
Date: Wednesday, April 01, 2015
Time: 9:04 AM, Local Time (GMT -06:00)
CA API Gateway – Policy Manager
File Edit Tasks View Help
Connect Disconnect Refresh Home Preferences
Assertions Identity Providers
Access Control
Transport Layer Security (TLS)
XML Security
Message Validation/Transformation
Message Routing
Service Availability
Logging, Auditing and Alerts
Policy Logic
Add Comment to Policy
All assertions must evaluate to true
At least one assertion must evaluate to true
Compare Expression
Continue Processing
Create Routing Strategy
Execute Routing Strategy
Search warehouse
Publish SOAP Web Service
Create WSDL
Publish Web API
Publish RESTful Service Proxy With WADL
Publish Internal Service
Publish Reverse Web Proxy
Create Policy
Create New Folder
Refresh
Paste
Filter
Sort By
Home
CA API Gateway - Policy Manager Home
Please select a task below.
Tasks
Create LDAP Identity Provider
Create Simple LDAP Identity Provider
Create Federated Identity Provider
Create Internal User
Create Internal Group
Search Identity Provider
Publish SOAP Web Service
Publish Web API
Publish RESTful Service Proxy With WADL
All folders are up to date.
Connected to "Ca"

Step 2, create an endpoint on the gateway that listens at a particular spot. Note there was already one called this so he received a resolution conflict, stopped, deleted the old one, and continued with this command.



Enter the name of the non-SOAP application you want to publish:

Service Name: My Test

Enter the HTTP URL that the Gateway should forward requests to:

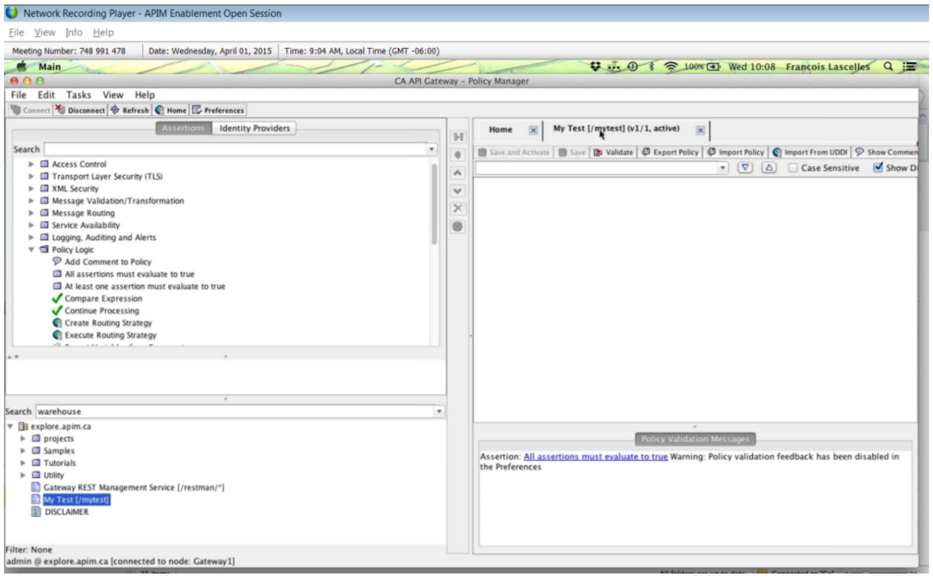
Target URL:

Complete the Gateway URL that will receive service requests:

Gateway URL: http(s)://explore.apim.ca:[port]/ mytest

Specify the connection and routing information for the non-SOAP application.

Successfully created empty policy below. Note service name on screen below matches "My Test" which is the new one we just created. The duplicate (prior to deletion) is shown in the above screen shot noted as "mytest" in the list, bottom left corner above.

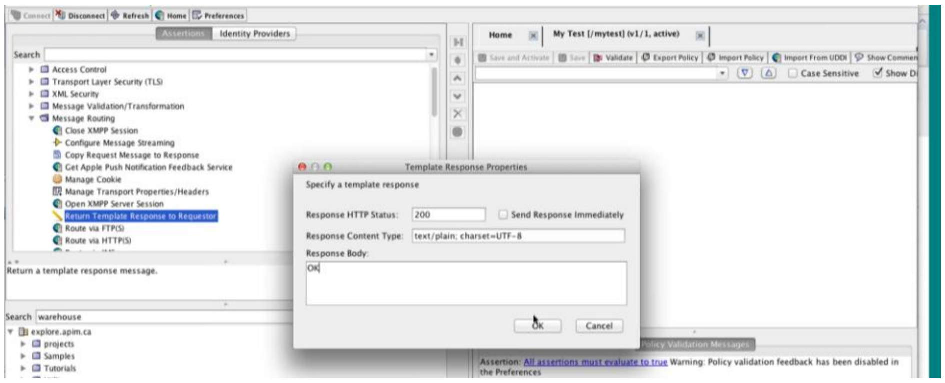

Network Recording Player - APIM Enablement Open Session
CA API Gateway – Policy Manager
Assertions
Identity Providers
Home
My Test [/mytest] (v1/1, active)
Save and Activate
Save
Validate
Export Policy
Import Policy
Import From UDDI
Policy Validation Messages
explore.apim.ca
projects
Samples
Tutorials
Utility
Gateway REST Management Service [/restman/*]
My Test [/mytest]
DISCLAIMER

Meeting Number: 748 991 478 Date: Wednesday, April 01, 2015 Time: 9:04 AM, Local Time (GMT -06:00)

Assertion: All assertions must evaluate to true Warning: Policy validation feedback has been disabled in the Preferences

Filter: None
admin @ explore.apim.ca [connected to node: Gateway1]

Next, insert a return response with just "ok". Be sure to change type to text/plain.


Template Response Properties
Message Routing
Return Template Response to Requestor
OK
Cancel

Return a template response message.

Specify a template response

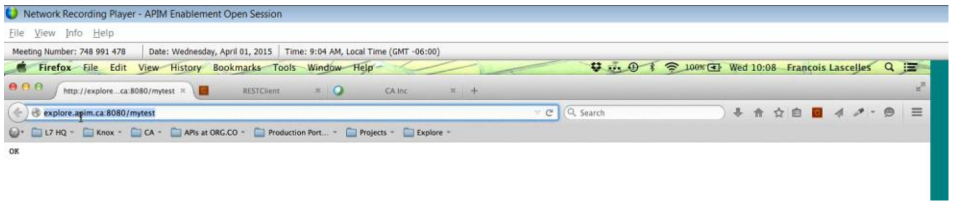
Response HTTP Status: 200 ☐ Send Response Immediately

Response Content Type: text/plain; charset=UTF-8

Response Body: OK

Open a browser and test this much.

Note that this URL will create a HTTP Get at "/mytest" which resolves to our policy.

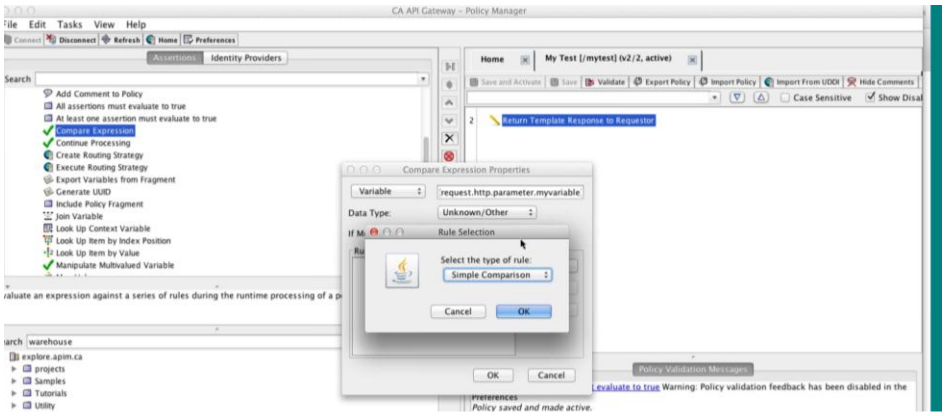

Network Recording Player - APIM Enablement Open Session
File View Info Help
Meeting Number: 748 991 478
Date: Wednesday, April 01, 2015
Time: 9:04 AM, Local Time (GMT -06:00)
Firefox File Edit View History Bookmarks Tools Window Help
Wed 10:08 Francois Lascelles
http://explore...ca:8080/mytest
RESTClient
CA Inc
Search
L7 HQ
Knox
CA
APIs at ORG.CO
Production Port...
Projects
Explore

explore.apim.ca:8080/mytest

OK

So far so good. We know it is working. Now let's add some logic.

Insert a comparison and examine the contents of a context variable. Somewhat like an "IF" using compare assertion found in the policy logic section of assertions. Drag and drop. Fill it in.



2 Return Template Response to Requestor

valuate an expression against a series of rules during the runtime processing of a p

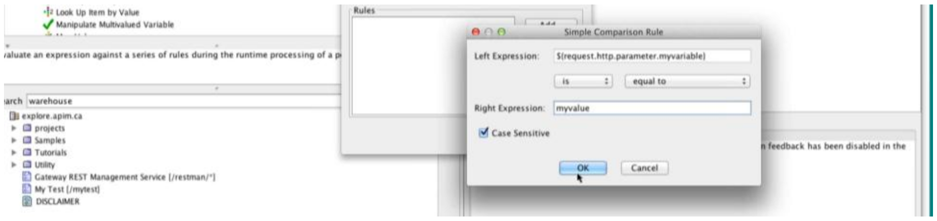
Variable: request.http.parameter.myvariable

Data Type: Unknown/Other

Select the type of rule: Simple Comparison

evaluate to true Warning: Policy validation feedback has been disabled in the

Policy saved and made active.

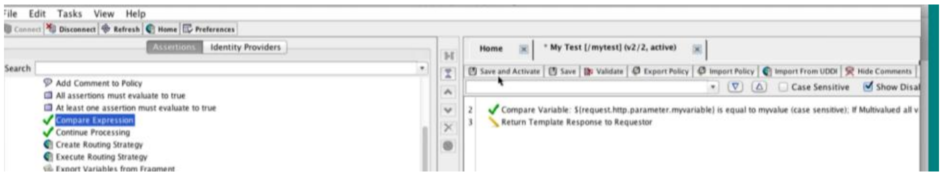


Left Expression: ${request.http.parameter.myvariable}

is equal to

Right Expression: myvalue

✓ Case Sensitive

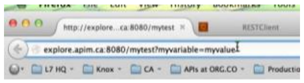


2 Compare Variable: ${request.http.parameter.myvariable} is equal to myvalue (case sensitive); If Multivalued all v

3 Return Template Response to Requestor

Save and activate. Go to Firefox.

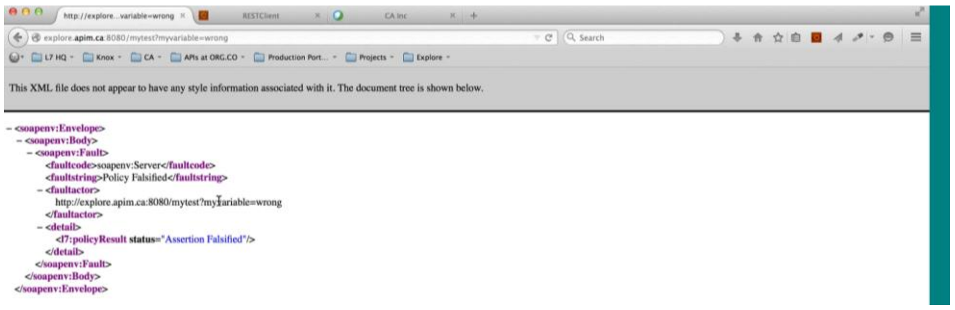
The way to add the variable into the test is by adding ?myvariable=myvalue to the end of the URL



explore.apim.ca:8080/mytest?myvariable=myvalue

And it works. All good, it returns "ok".

If we submit a bad value as the parameter value, we see the following canned error message.



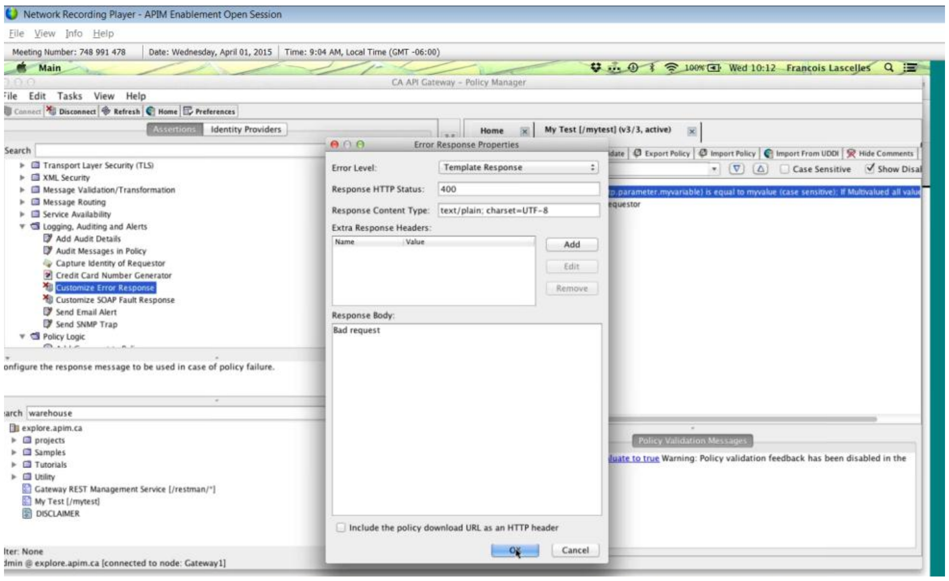
explore.apim.ca:8080/mytest?myvariable=wrong

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
- <soapenv:Envelope>
  - <soapenv:Body>
    - <soapenv:Fault>
        <faultcode>soapenv:Server</faultcode>
        <faultstring>Policy Falsified</faultstring>
      - <faultactor>
          http://explore.apim.ca:8080/mytest?myvariable=wrong
        </faultactor>
      - <detail>
          <l7:policyResult status="Assertion Falsified"/>
        </detail>
      </soapenv:Fault>
    </soapenv:Body>
  </soapenv:Envelope>
```

In this case above, the policy failed on the compare statement, **exited without executing the return response**. (Remember the hidden "and" like envelope around all statements as line 1.)

Now let's pretty up our error messages. Select an assertion from the logging, auditing, alerts grouping. Place it at the top of the policy and change the HTTP status to 400 and the body of the response to "bad request."



Error Response Properties

Error Level: Template Response

Response HTTP Status: 400

Response Content Type: text/plain; charset=UTF-8

Extra Response Headers:

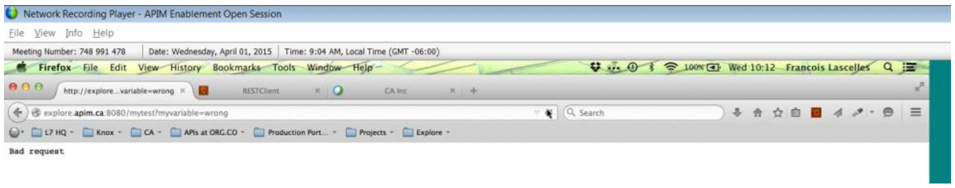
Name	Value

Response Body:

Bad request

☐ Include the policy download URL as an HTTP header

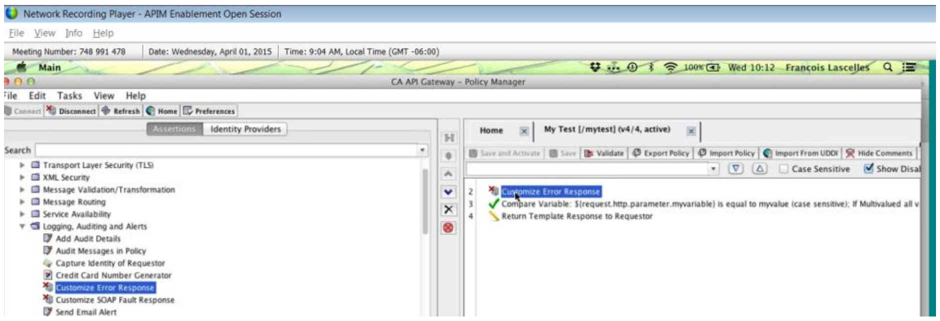
Save and activate, return to Firefox and submit the request again. Voila! Friendly error message.



explore.apim.ca:8080/mytest?myvariable=wrong

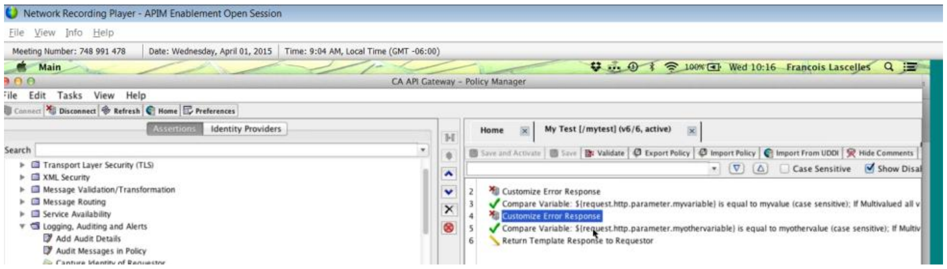
```
Bad request
```

From line 2 of our policy onwards, if there is an error, we will get this message above returned to the user.

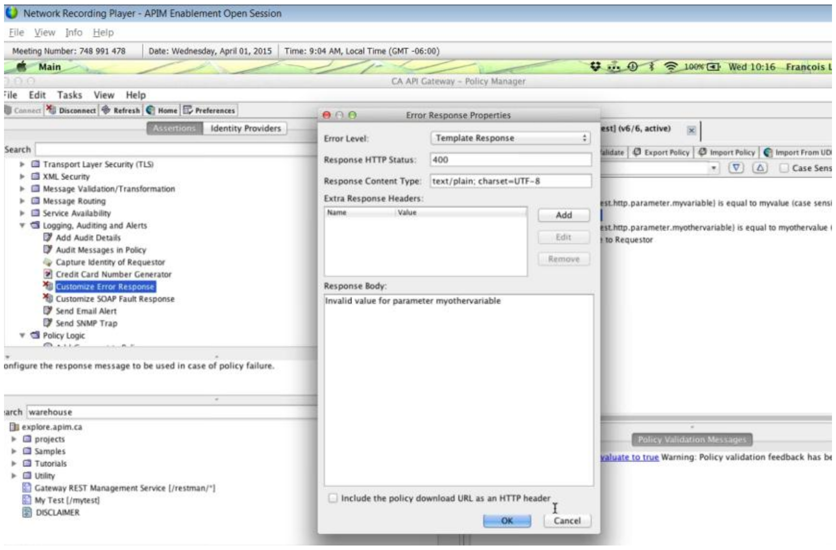


```
2  Customize Error Response
3  Compare Variable: ${request.http.parameter.myvariable} is equal to myvalue (case sensitive); If Multivalued all v
4  Return Template Response to Requestor
```

Since you might want different error messages at different points in the policy, you need to insert additional customize error responses along the way. Remember to put this assertion immediately before what you are testing because if that line fails, processing stops. The format of the error message will be the style of the last set instruction (customize error response assertion). Line 4 in the example below will change the message so you can see that the problem is with “myothervalue.”



```
2  Customize Error Response
3  Compare Variable: ${request.http.parameter.myvariable} is equal to myvalue (case sensitive); If Multivalued all v
4  Customize Error Response
5  Compare Variable: ${request.http.parameter.myothervariable} is equal to myothervalue (case sensitive); If Multiv
6  Return Template Response to Requestor
```



Error Response Properties

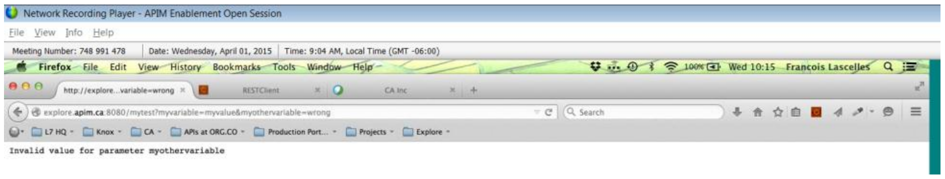
Error Level: Template Response

Response HTTP Status: 400

Response Content Type: text/plain; charset=UTF-8

Response Body: Invalid value for parameter myothervariable

Test results: (Note format with the “&” to submit request with more than one parameter.)



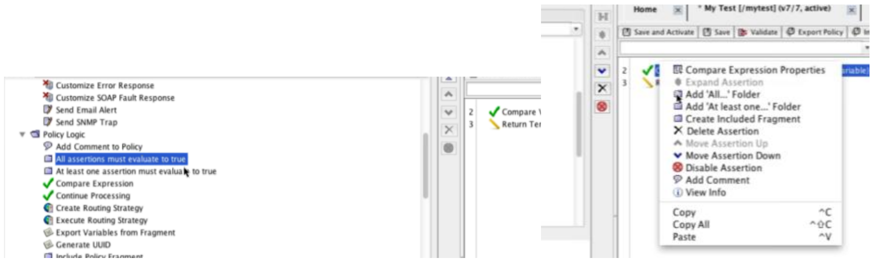
```
explore.apim.ca:8080/mytest?myvariable=myvalue&myothervariable=wrong

Invalid value for parameter myothervariable
```

***********************************END of TOPIC******************

Next topic, conditional statements and branching.

ALL must evaluate to true = AND   AT LEAST ONE must evaluate to true = OR

Can be chosen from the assertions tree on the left or from the right click context menu in the policy on the right because they are used frequently. Both examples are shown here:



Use this construct to return one of two different messages. The first is ok confirmation, the second is an error message.

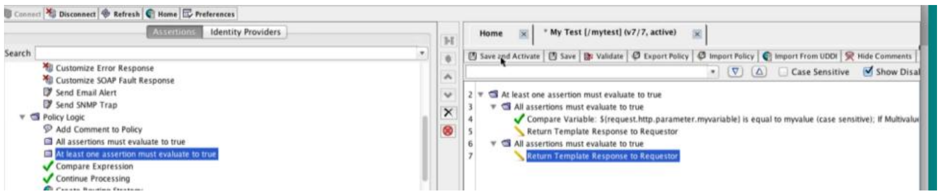
OR (at least one…)

AND (all must…) (child 1)

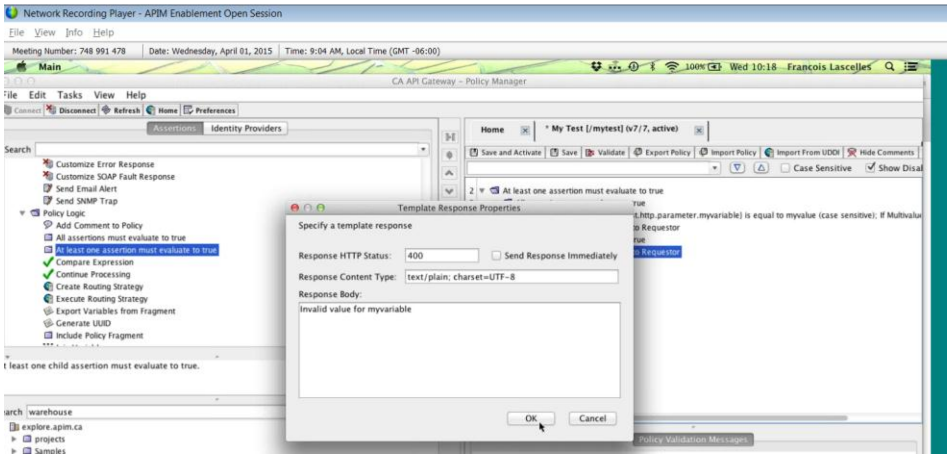
AND (all must…) (child 2)

If the first AND works it will NOT process the second AND. So, if child 1 is true then child 2 is skipped. Since child 1 is an AND, it will execute the return response in that child group.

If line 4 fails, it stops, returns a false to line 3, which then goes to the parent on line 2 and says what now? The parent on line 2 attempts the second child group, line 6. The return response on line 7 executes and returns a true value for the child 2 grouping to the parent on line 2.



```
2 At least one assertion must evaluate to true
3   All assertions must evaluate to true
4     Compare Variable: ${request.http.parameter.myvariable} is equal to myvalue (case sensitive); If Multivalu
5     Return Template Response to Requestor
6   All assertions must evaluate to true
7     Return Template Response to Requestor
```



Meeting Number: 748 991 478 | Date: Wednesday, April 01, 2015 | Time: 9:04 AM, Local Time (GMT -06:00)

Template Response Properties

Specify a template response

Response HTTP Status: 400   [ ] Send Response Immediately

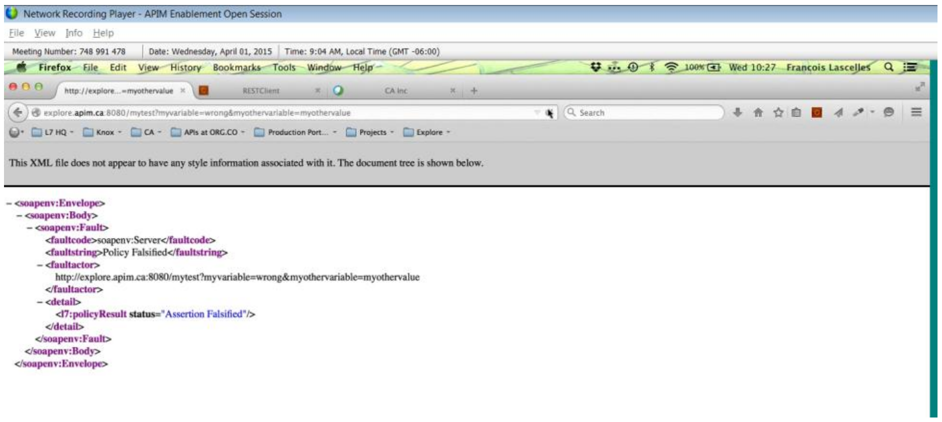
Response Content Type: text/plain; charset=UTF-8

Response Body:

Invalid value for myvariable

Occasionally you might need to stop or continue with the logic. Stop Processing assertion = false, Continue processing assertion = true.

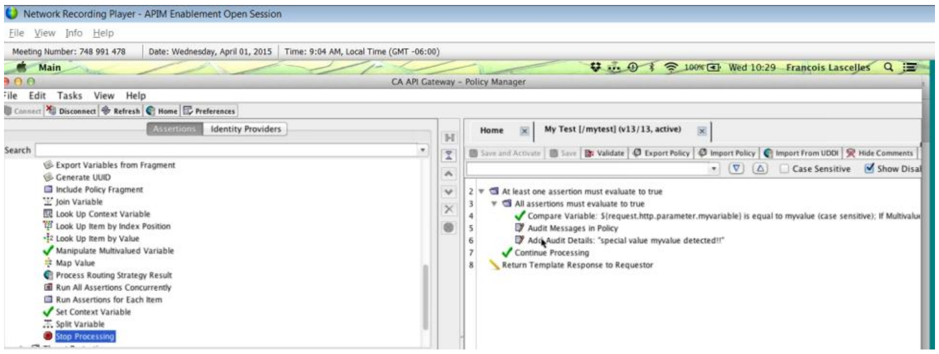
If you do use a stop processing assertion, BE SURE to check the Send Response Immediately box so the user sees the custom message. Otherwise since the policy fails, the user will get the default/standard ugly xml message below.


Network Recording Player - APIM Enablement Open Session
File View Info Help
Meeting Number: 748 991 478
Date: Wednesday, April 01, 2015
Time: 9:04 AM, Local Time (GMT -06:00)
Firefox File Edit View History Bookmarks Tools Window Help
Wed 10:27 Francois Lascelles
http://explore...=myothervalue
RESTClient
CA Inc
explore.apim.ca:8080/mytest?myvariable=wrong&myothervariable=myothervalue
Search
L7 HQ
Knox
CA
APIs at ORG.CO
Production Port...
Projects
Explore

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
- <soapenv:Envelope>
  - <soapenv:Body>
    - <soapenv:Fault>
        <faultcode>soapenv:Server</faultcode>
        <faultstring>Policy Falsified</faultstring>
      - <faultactor>
          http://explore.apim.ca:8080/mytest?myvariable=wrong&myothervariable=myothervalue
        </faultactor>
      - <detail>
          <l7:policyResult status="Assertion Falsified"/>
        </detail>
      </soapenv:Fault>
    </soapenv:Body>
  </soapenv:Envelope>
```

Note in the example below you will NOT get messages in your audit log unless the compare condition is true and the processing continues with the audit assertions. If the compare on line 4 fails (is false), processing will stop on line 4 and bubble up to line 3 which now is false (processing stops with the first false in an ALL group) and returns to line 2 for advice. Line 2 (which is an OR) then checks for another child instruction to execute which is line 7. (Just like home, if the first kid won't feed the dogs, ask the second one.) Line 7 returns a true to line 2 and processing continues sequentially with line 8.


Network Recording Player - APIM Enablement Open Session
File View Info Help
Meeting Number: 748 991 478
Date: Wednesday, April 01, 2015
Time: 9:04 AM, Local Time (GMT -06:00)
Main
Wed 10:29 Francois Lascelles
CA API Gateway - Policy Manager
File Edit Tasks View Help
Connect
Disconnect
Refresh
Home
Preferences
Assertions
Identity Providers
Search
Export Variables from Fragment
Generate UUID
Include Policy Fragment
Join Variable
Look Up Context Variable
Look Up Item by Index Position
Look Up Item by Value
Manipulate Multivalued Variable
Map Value
Process Routing Strategy Result
Run All Assertions Concurrently
Run Assertions for Each Item
Set Context Variable
Split Variable
Stop Processing
Home
My Test [/mytest] (v13/13, active)
Save and Activate
Save
Validate
Export Policy
Import Policy
Import From UDDI
Hide Comments
Case Sensitive
Show Disal

```
2 ▼ At least one assertion must evaluate to true
3   ▼ All assertions must evaluate to true
4       ✓ Compare Variable: ${request.http.parameter.myvariable} is equal to myvalue (case sensitive); If Multivalu
5       Audit Messages in Policy
6       Add Audit Details: "special value myvalue detected!!"
7     ✓ Continue Processing
8   Return Template Response to Requestor
```

Example of when you might use this construct is for exceptions. You want to throw out a conditional alert but you don't want to stop the processing.

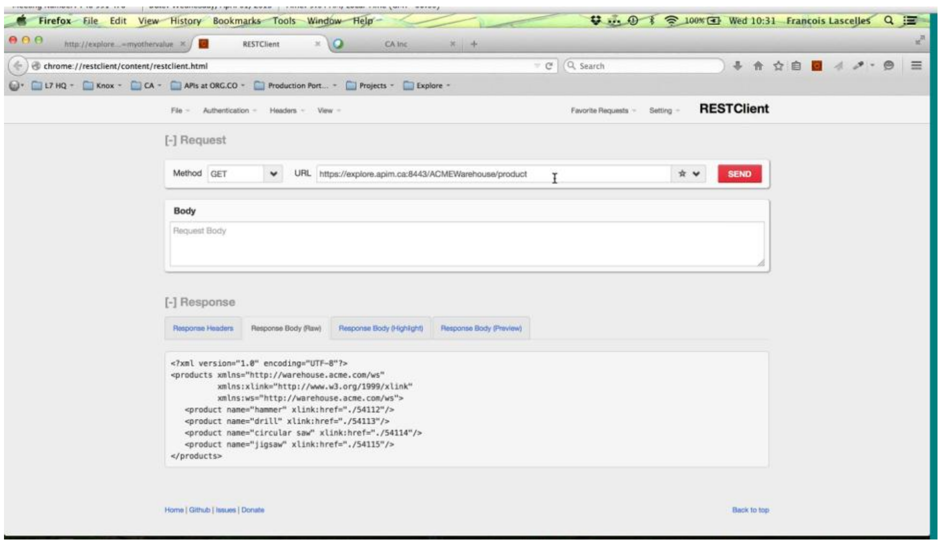
************************* End of Topic ***************************

Next topic is Looping, aka "For Each"

Run Assertions for Each item

Had another endpoint at the ready as per the test/validation below. Added a routing assertion which routes to this endpoint.

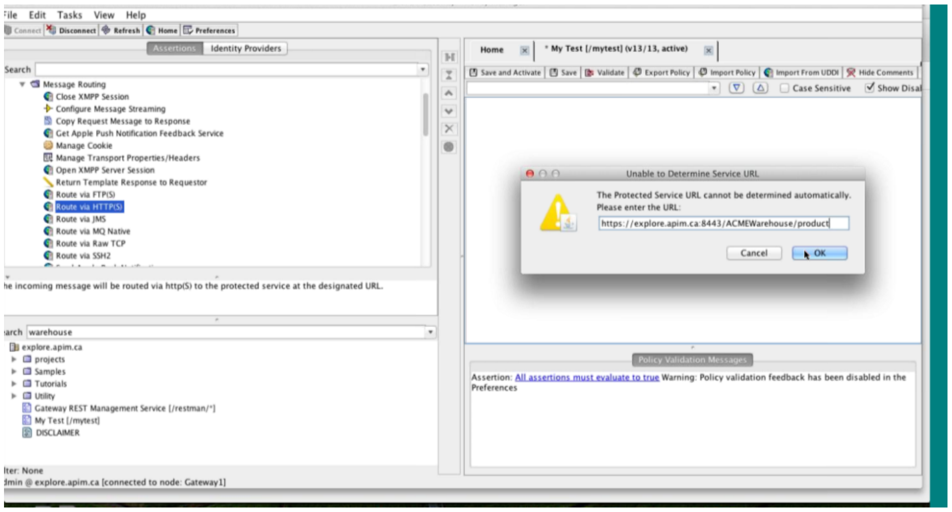
Results of the validation test:



Method GET URL https://explore.apim.ca:8443/ACMEWarehouse/product

```
<?xml version="1.0" encoding="UTF-8"?>
<products xmlns="http://warehouse.acme.com/ws"
          xmlns:xlink="http://www.w3.org/1999/xlink"
          xmlns:ws="http://warehouse.acme.com/ws">
   <product name="hammer" xlink:href="./54112"/>
   <product name="drill" xlink:href="./54113"/>
   <product name="circular saw" xlink:href="./54114"/>
   <product name="jigsaw" xlink:href="./54115"/>
</products>
```

Copy the URL above into the URL below for the route via HTTP(S) assertion.

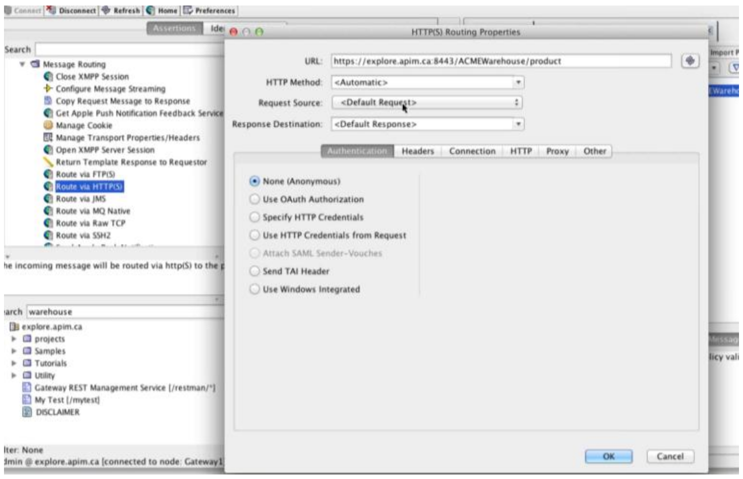


Unable to Determine Service URL

The Protected Service URL cannot be determined automatically.
Please enter the URL:
https://explore.apim.ca:8443/ACMEWarehouse/product

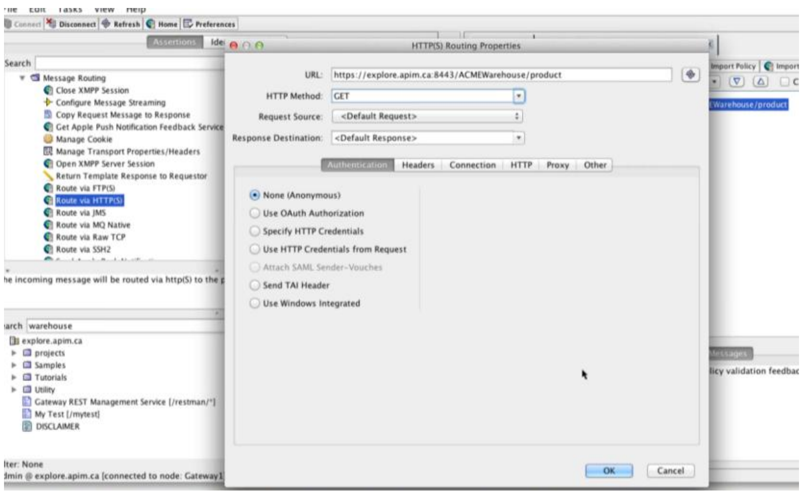
Cancel OK

Say ok and assertion is added to the policy. Click it to change some of the additional properties.



```
HTTP(S) Routing Properties
URL: https://explore.apim.ca:8443/ACMEWarehouse/product
HTTP Method: <Automatic>
Request Source: <Default Request>
Response Destination: <Default Response>
Authentication: (•) None (Anonymous)
```

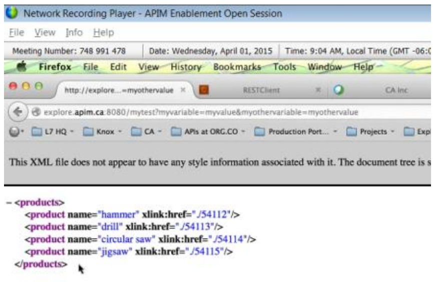
Change it (the routing assertion) to an http GET by changing the method:



```
HTTP(S) Routing Properties
URL: https://explore.apim.ca:8443/ACMEWarehouse/product
HTTP Method: GET
Request Source: <Default Request>
Response Destination: <Default Response>
Authentication: (•) None (Anonymous)
```

Save and activate. Test the policy by sending a request to the URL mytest. Remember mytest is routing to another destination (as per the http get above).

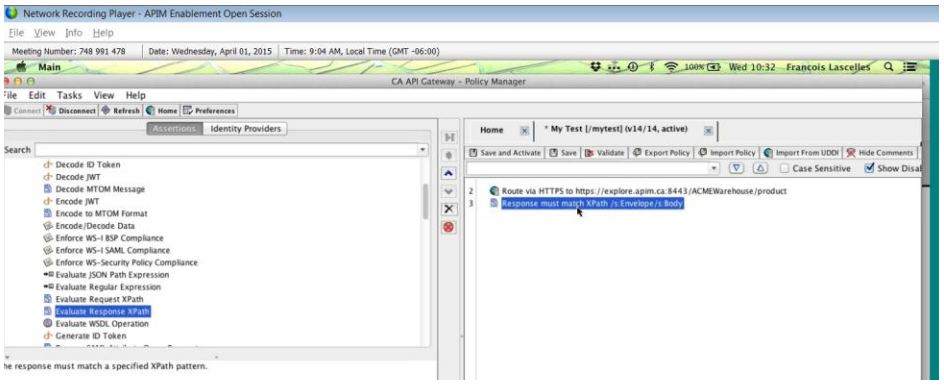
Results from submitting the mytest in firefox, note the parameters are ignored for now.



```
explore.apim.ca:8080/mytest?myvariable=myvalue&myothervariable=myothervalue

This XML file does not appear to have any style information associated with it. The document tree is s

- <products>
    <product name="hammer" xlink:href="./54112"/>
    <product name="drill" xlink:href="./54113"/>
    <product name="circular saw" xlink:href="./54114"/>
    <product name="jigsaw" xlink:href="./54115"/>
  </products>
```

Now that we know the proxy works and is retrieving data, let’s set up the loop.

Step 1 to setting up the loop is to evaluate the xpath expression returned from the backend service/routing step.


Network Recording Player - APIM Enablement Open Session

Meeting Number: 748 991 478 | Date: Wednesday, April 01, 2015 | Time: 9:04 AM, Local Time (GMT -06:00)

CA API Gateway - Policy Manager

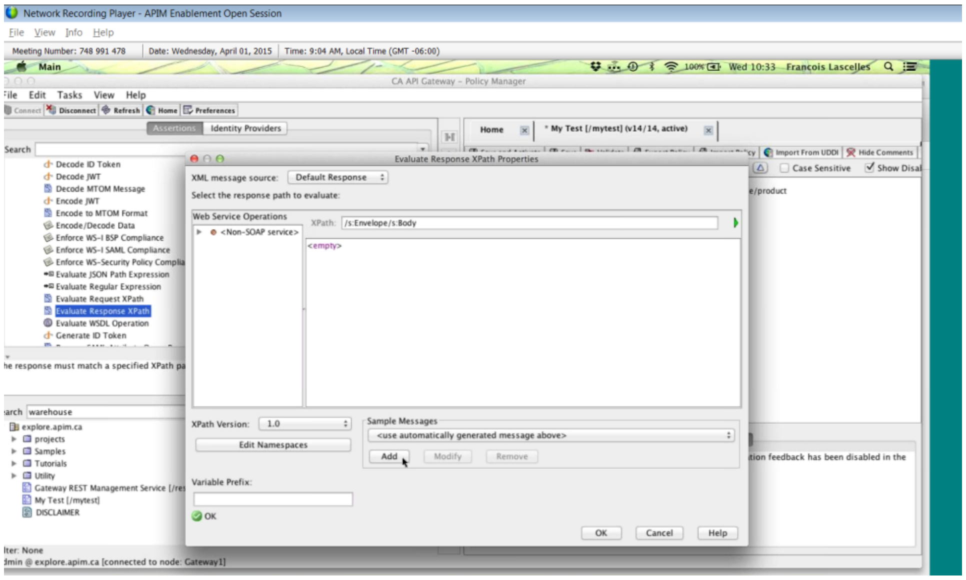
* My Test [/mytest] (v14/14, active)

```
2  Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
3  Response must match XPath /s:Envelope/s:Body
```

he response must match a specified XPath pattern.

Open the "must match Xpath" assertion and copy the XML result returned from our initial tests into the sample message. See below.

Click Add on sample message box after opening the assertion. (It is handy to have your XML result already copied at the ready to paste later.)


Evaluate Response XPath Properties

XML message source: Default Response

Select the response path to evaluate:

Web Service Operations: <Non-SOAP service>

XPath: /s:Envelope/s:Body

<empty>

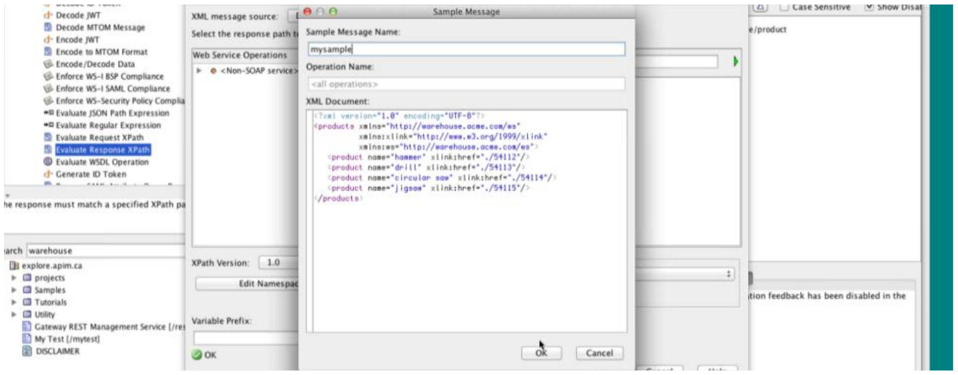
XPath Version: 1.0

Sample Messages: <use automatically generated message above>

Variable Prefix:

OK

Give the sample a name and paste the result into the document field. Press ok.


Sample Message

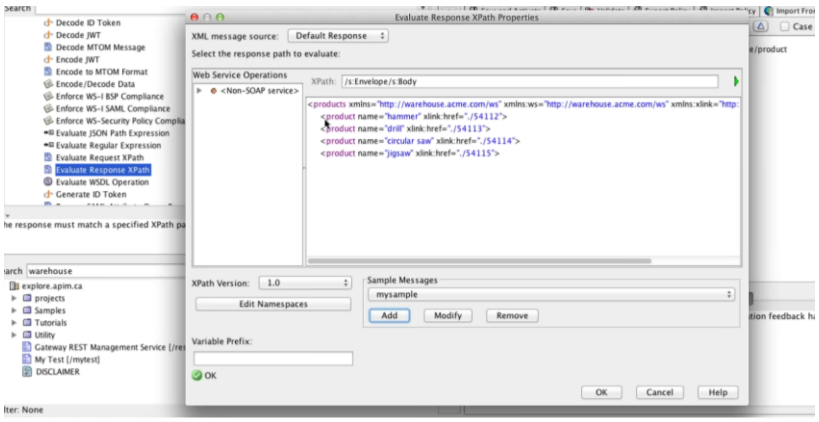
Sample Message Name: mysample

Operation Name: <all operations>

XML Document:

```
<?xml version="1.0" encoding="UTF-8"?>
<products xmlns="http://warehouse.acme.com/ws"
          xmlns:xlink="http://www.w3.org/1999/xlink"
          xmlns:ws="http://warehouse.acme.com/ws">
   <product name="hammer" xlink:href="./54112"/>
   <product name="drill" xlink:href="./54113"/>
   <product name="circular saw" xlink:href="./54114"/>
   <product name="jigsaw" xlink:href="./54115"/>
</products>
```

Using a sample makes it easier to create your XPath expression:

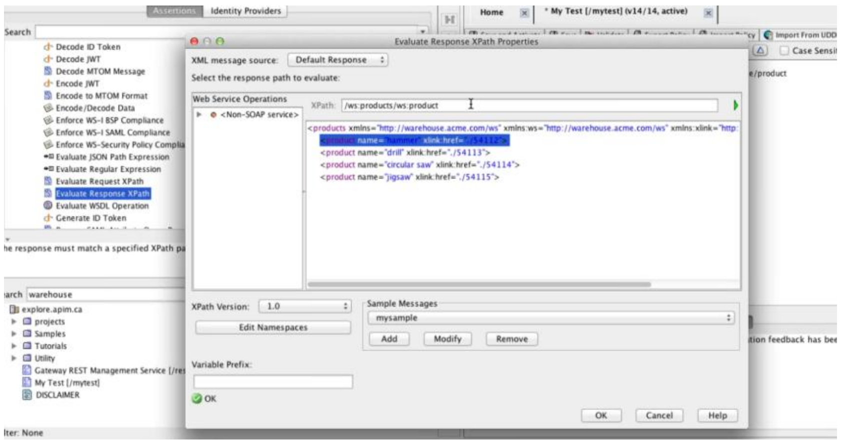


XPath: /s:Envelope/s:Body

```
<products xmlns="http://warehouse.acme.com/ws" xmlns:ws="http://warehouse.acme.com/ws" xmlns:xlink="http:
  <product name="hammer" xlink:href="./54112">
  <product name="drill" xlink:href="./54113">
  <product name="circular saw" xlink:href="./54114">
  <product name="jigsaw" xlink:href="./54115">
```

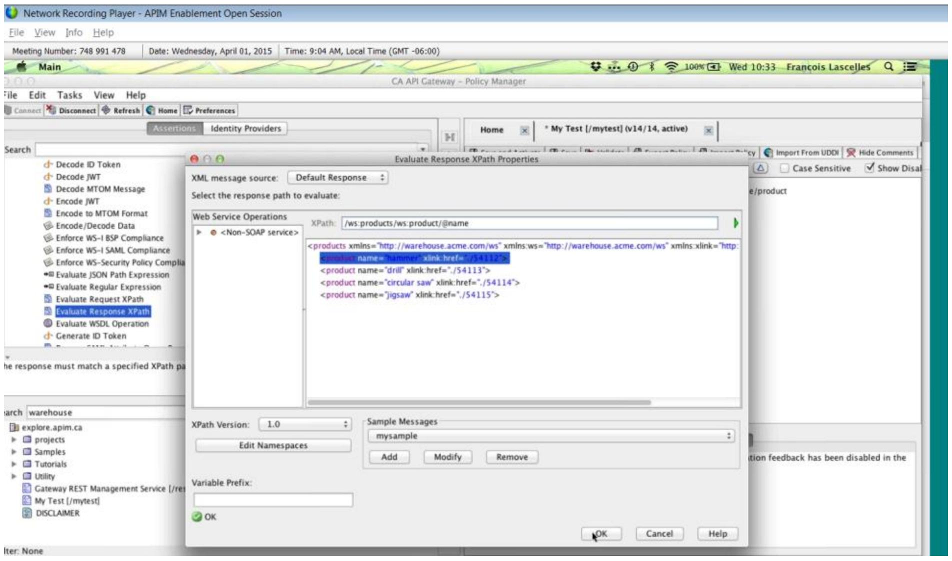
Sample Messages: mysample

Click on the line with the desired element and the wizard puts it into the XPath line.



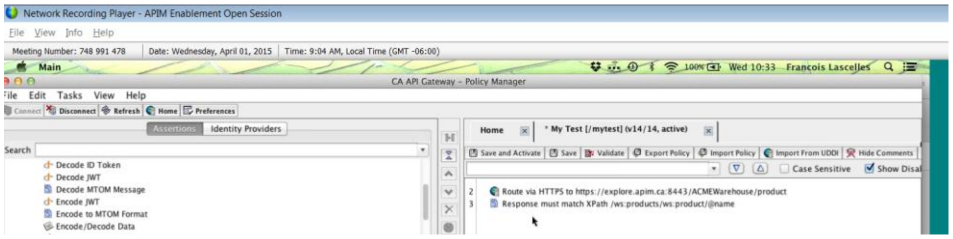
XPath: /ws:products/ws:product

Type @name, press ok.


Network Recording Player - APIM Enablement Open Session
File View Info Help
Meeting Number: 748 991 478
Date: Wednesday, April 01, 2015
Time: 9:04 AM, Local Time (GMT -06:00)
CA API Gateway – Policy Manager
Evaluate Response XPath Properties
XML message source: Default Response
Select the response path to evaluate:
OK
Cancel
Help

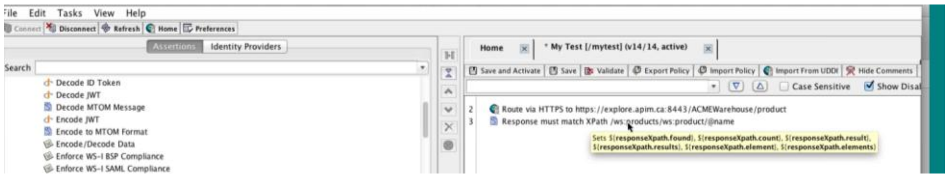
XPath: /ws:products/ws:product/@name

Line 3 is complete as per below:



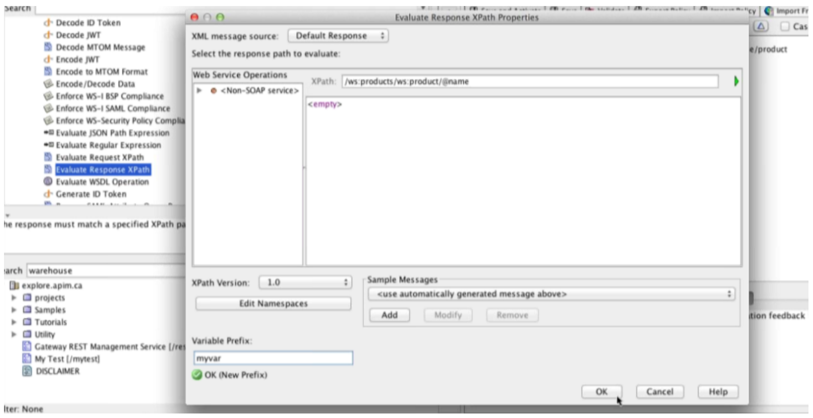
* My Test [/mytest] (v14/14, active)

```
2  Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
3  Response must match XPath /ws:products/ws:product/@name
```



```
2  Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
3  Response must match XPath /ws:products/ws:product/@name
     Sets ${responseXpath.found}, ${responseXpath.count}, ${responseXpath.result},
     ${responseXpath.results}, ${responseXpath.element}, ${responseXpath.elements}
```

Click on the assertion again and add a prefix:



Evaluate Response XPath Properties

XML message source: Default Response

Select the response path to evaluate:

XPath: /ws:products/ws:product/@name

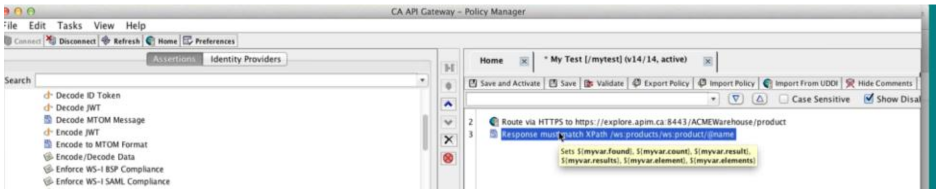
XPath Version: 1.0

Sample Messages: <use automatically generated message above>

Variable Prefix: myvar

OK (New Prefix)

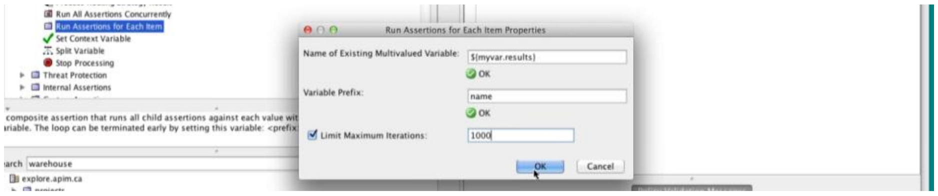
Note the difference in the hover help now that the prefix was changed:



```
2  Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
3  Response must match XPath /ws:products/ws:product/@name
     Sets ${myvar.found}, ${myvar.count}, ${myvar.result},
     ${myvar.results}, ${myvar.element}, ${myvar.elements}
```

The value of the name variable will be in the myvar.results. It is a multi-value context variable. Loop on a set of variables inside a multi-value context variable.

Drag in the “for each” assertion to create the loop. Point to myvar.results to drive the loop. Specify the prefix and max iterations.



Run Assertions for Each Item Properties

Name of Existing Multivalued Variable: ${myvar.results}

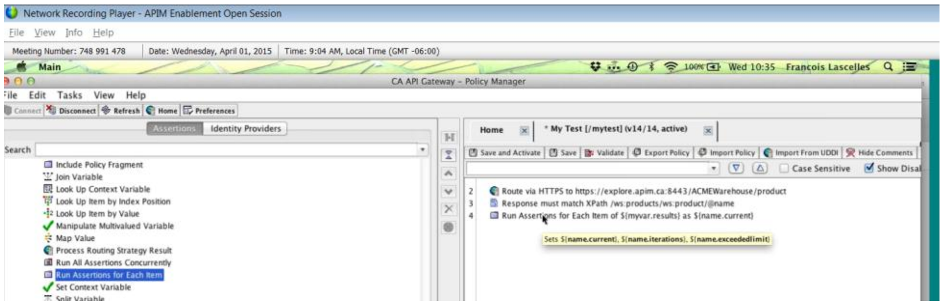
OK

Variable Prefix: name

OK

[✓] Limit Maximum Iterations: 1000

Note the variable names in the hover help:

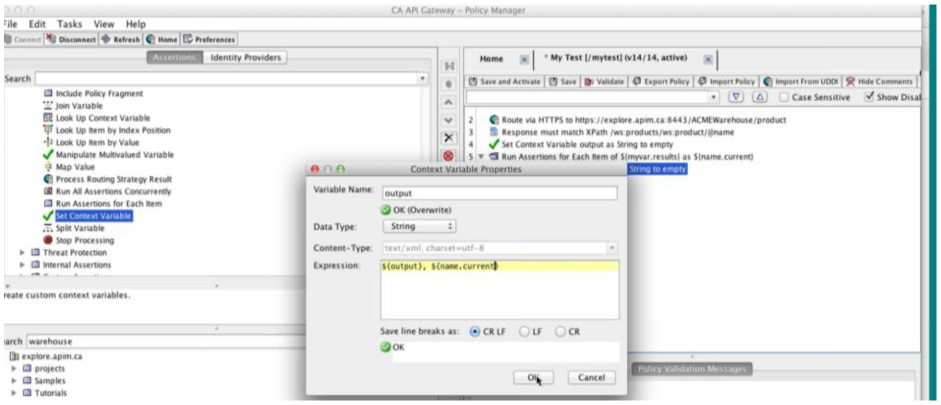

Network Recording Player - APIM Enablement Open Session
CA API Gateway - Policy Manager
Home
* My Test [/mytest] (v14/14, active)

```
2  Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
3  Response must match XPath /ws:products/ws:product/@name
4  Run Assertions for Each Item of ${myvar.results} as ${name.current}
```

Sets ${name.current}, ${name.iterations}, ${name.exceededlimit}

Now let's set the output variables.

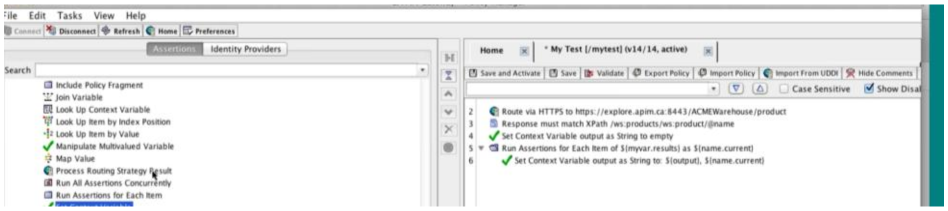
First set the output context variable to empty to clear it. Then set it to store the value within the loop by adding two set context variable assertions. Leave the expression field blank on the assertion in line 4.


CA API Gateway - Policy Manager
Context Variable Properties
Policy Validation Messages

```
2  Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
3  Response must match XPath /ws:products/ws:product/@name
4  Set Context Variable output as String to empty
5  Run Assertions for Each Item of ${myvar.results} as ${name.current}
```

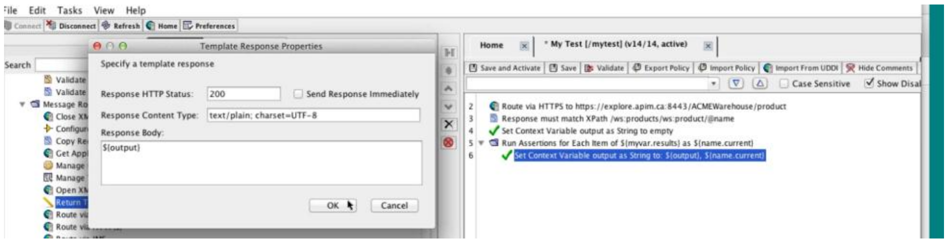
Variable Name: output
OK (Overwrite)
Data Type: String
Content-Type: text/xml; charset=utf-8
Expression: ${output}, ${name.current}
Save line breaks as: [✓ CR LF] LF CR
OK

Policy so far:



```
2  Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
3  Response must match XPath /ws:products/ws:product/@name
4  Set Context Variable output as String to empty
5  Run Assertions for Each Item of ${myvar.results} as ${name.current}
6     Set Context Variable output as String to: ${output}, ${name.current}
```

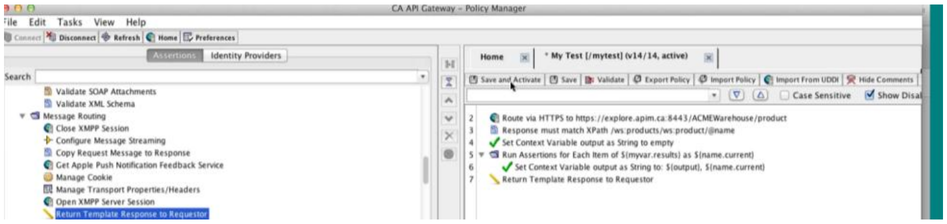
Add final template response:



Specify a template response
Response HTTP Status: 200
Send Response Immediately: [ ]
Response Content Type: text/plain; charset=UTF-8
Response Body:
${output}

```
2  Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
3  Response must match XPath /ws:products/ws:product/@name
4  Set Context Variable output as String to empty
5  Run Assertions for Each Item of ${myvar.results} as ${name.current}
6     Set Context Variable output as String to: ${output}, ${name.current}
```

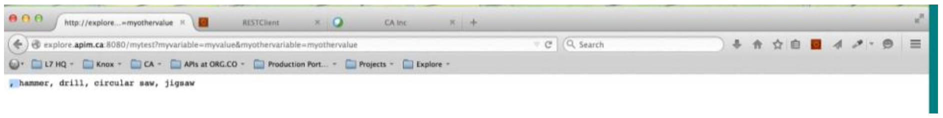
It will add the template response as a child of the loop in line 5. To cure this, drag line 6 above line 5, close/collapse line 5, then use the up/down arrows to reposition it below line 5. Expand line 5 and line 7 will remain properly out-dented, equal with line 5.


CA API Gateway – Policy Manager
File Edit Tasks View Help
Connect Disconnect Refresh Home Preferences
Assertions Identity Providers
Search
Validate SOAP Attachments
Validate XML Schema
Message Routing
Close XMPP Session
Configure Message Streaming
Copy Request Message to Response
Get Apple Push Notification Feedback Service
Manage Cookie
Manage Transport Properties/Headers
Open XMPP Server Session
Return Template Response to Requestor
Home
* My Test [/mytest] (v14/14, active)
Save and Activate Save Validate Export Policy Import Policy Import From UDDI Hide Comments
Case Sensitive Show Disal

```
2    Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
3    Response must match XPath /ws:products/ws:product/@name
4    Set Context Variable output as String to empty
5 ▼  Run Assertions for Each Item of ${myvar.results} as ${name.current}
6      Set Context Variable output as String to: ${output}, ${name.current}
7    Return Template Response to Requestor
```

Save and activate.

Test the policy by submitting the following "/mytest" request in Firefox:

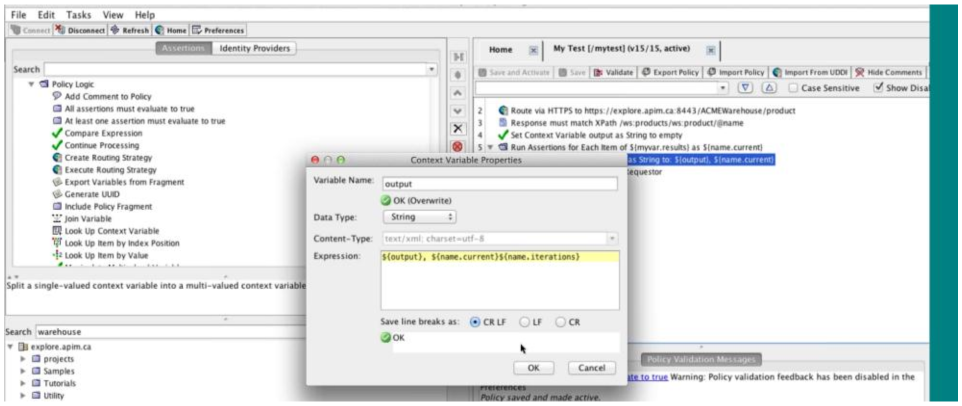


```
explore.apim.ca:8080/mytest?myvariable=myvalue&myothervariable=myothervalue
```

```
, hammer, drill, circular saw, jigsaw
```

Note the results above. Some additional work can be done to remove the leading comma.

Add named iteration to the context variable so we can test which execution of the loop we are on.

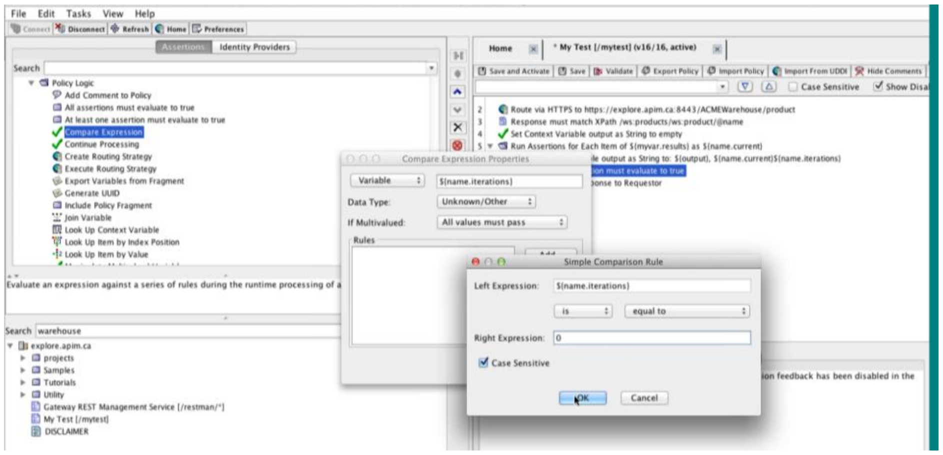

File Edit Tasks View Help
Connect Disconnect Refresh Home Preferences
Assertions Identity Providers
Search
Policy Logic
Add Comment to Policy
All assertions must evaluate to true
At least one assertion must evaluate to true
Compare Expression
Continue Processing
Create Routing Strategy
Execute Routing Strategy
Export Variables from Fragment
Generate UUID
Include Policy Fragment
Join Variable
Look Up Context Variable
Look Up Item by Index Position
Look Up Item by Value
Split a single-valued context variable into a multi-valued context variable
Search warehouse
explore.apim.ca
projects
Samples
Tutorials
Utility
Home
My Test [/mytest] (v15/15, active)
Save and Activate Save Validate Export Policy Import Policy Import From UDDI Hide Comments
Case Sensitive Show Disa
Context Variable Properties
OK
Cancel
Policy Validation Messages

```
2    Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
3    Response must match XPath /ws:products/ws:product/@name
4    Set Context Variable output as String to empty
5 ▼  Run Assertions for Each Item of ${myvar.results} as ${name.current}
       as String to: ${output}, ${name.current}
       equestor
```

Variable Name: output
OK (Overwrite)
Data Type: String
Content-Type: text/xml; charset=utf-8
Expression: `${output}, ${name.current}${name.iterations}`
Save line breaks as: [✓ CR LF] LF CR
OK

ate to true Warning: Policy validation feedback has been disabled in the Preferences
Policy saved and made active.

Run the test, look at the output to discover that the first iteration is number "0". Next, setup an OR condition and a compare.



Compare Expression Properties

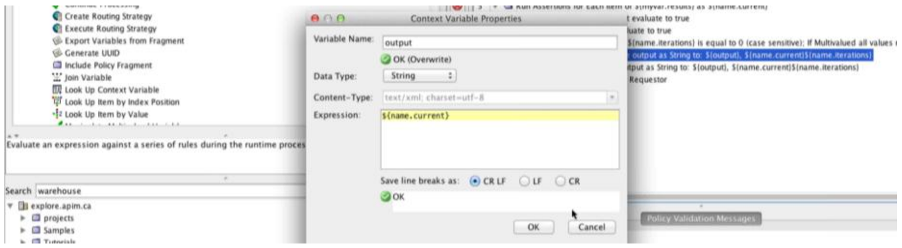
Variable: ${name.iterations}
Data Type: Unknown/Other
If Multivalued: All values must pass

Simple Comparison Rule

Left Expression: ${name.iterations}
is equal to
Right Expression: 0
[✓] Case Sensitive

```
Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
Response must match XPath /ws:products/ws:product/@name
Set Context Variable output as String to empty
Run Assertions for Each Item of ${myvar.results} as ${name.current}
```

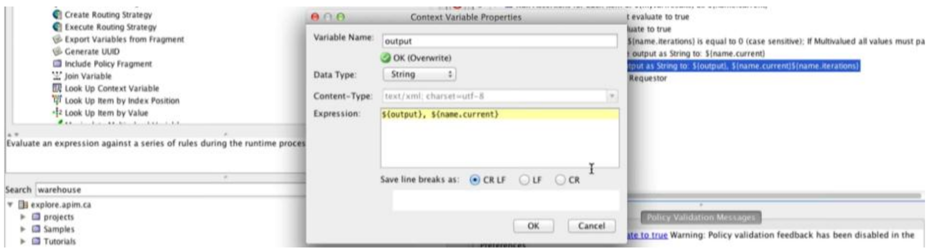
If zero, don’t append.



Context Variable Properties

Variable Name: output
OK (Overwrite)
Data Type: String
Content-Type: text/xml; charset=utf-8
Expression: ${name.current}
Save line breaks as: [✓ CR LF] LF CR

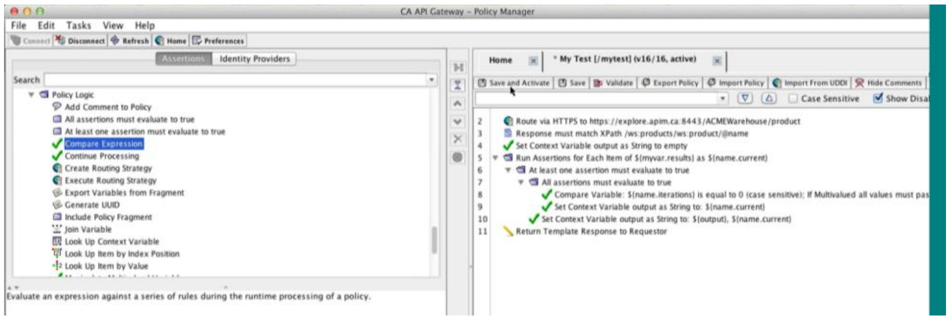
If not zero, then append it, go ahead and remove the counter as well.



Context Variable Properties

Variable Name: output
OK (Overwrite)
Data Type: String
Content-Type: text/xml; charset=utf-8
Expression: ${output}, ${name.current}
Save line breaks as: [✓ CR LF] LF CR

Save and activate the following:



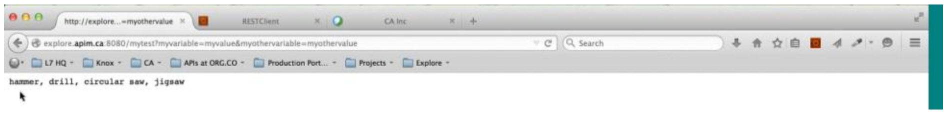
CA API Gateway – Policy Manager

* My Test [/mytest] (v16/16, active)

```
Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
Response must match XPath /ws:products/ws:product/@name
Set Context Variable output as String to empty
Run Assertions for Each Item of ${myvar.results} as ${name.current}
    At least one assertion must evaluate to true
        All assertions must evaluate to true
            Compare Variable: ${name.iterations} is equal to 0 (case sensitive); If Multivalued all values must pas
            Set Context Variable output as String to: ${name.current}
        Set Context Variable output as String to: ${output}, ${name.current}
Return Template Response to Requestor
```

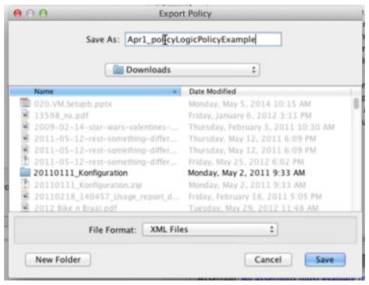
Evaluate an expression against a series of rules during the runtime processing of a policy.

Here are the results:



```
hammer, drill, circular saw, jigsaw
```

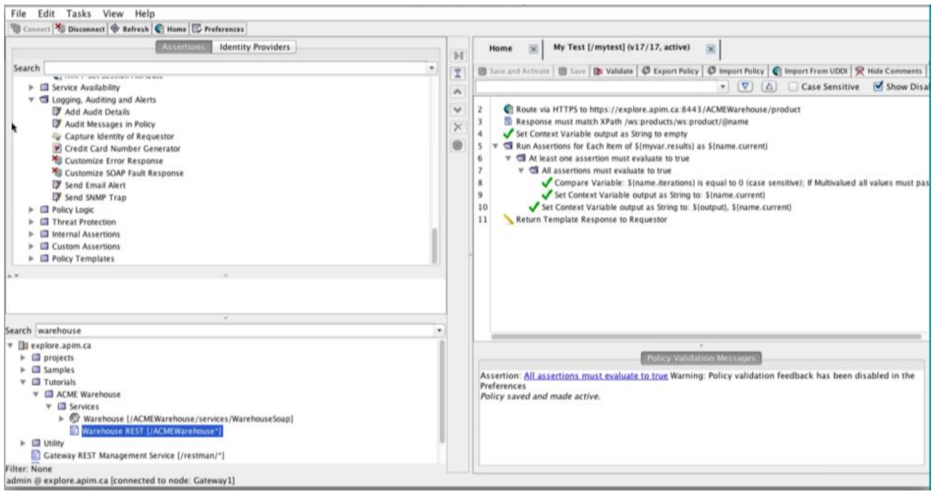
Policy exported as:



Save As: Apr1_policyLogicPolicyExample

File Format: XML Files

Depends on the ACME Warehouse REST below:



My Test [/mytest] (v17/17, active)

```
Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
Response must match XPath /ws:products/ws:product/@name
Set Context Variable output as String to empty
Run Assertions for Each Item of ${myvar.results} as ${name.current}
    At least one assertion must evaluate to true
        All assertions must evaluate to true
            Compare Variable: ${name.iterations} is equal to 0 (case sensitive); If Multivalued all values must pas
            Set Context Variable output as String to: ${name.current}
        Set Context Variable output as String to: ${output}, ${name.current}
Return Template Response to Requestor
```

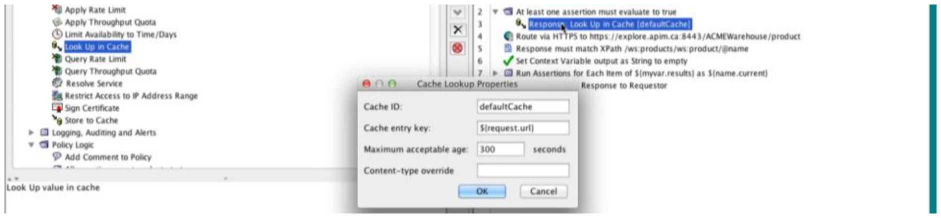
Assertion: All assertions must evaluate to true Warning: Policy validation feedback has been disabled in the Preferences
Policy saved and made active.

Use split and join assertions to go from multi value context variables to single value context variables.

*************************End of topic**

Next topic: Using local cache via assertions Look up in Cache and Store to Cache

Each cache entry has its own key.



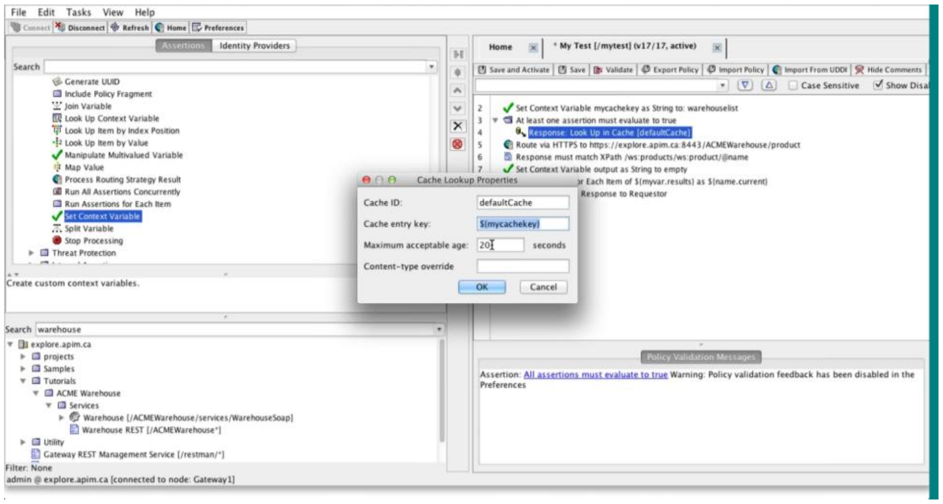
```
At least one assertion must evaluate to true
    Response: Look Up in Cache [defaultCache]
Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
Response must match XPath /ws:products/ws:product/@name
Set Context Variable output as String to empty
Run Assertions for Each Item of ${myvar.results} as ${name.current}
```

Cache Lookup Properties

Cache ID: defaultCache
Cache entry key: ${request.url}
Maximum acceptable age: 300 seconds
Content-type override:

Look Up value in cache

Or you can set a context variable and use that as your key. Note the new assertion on line 2.



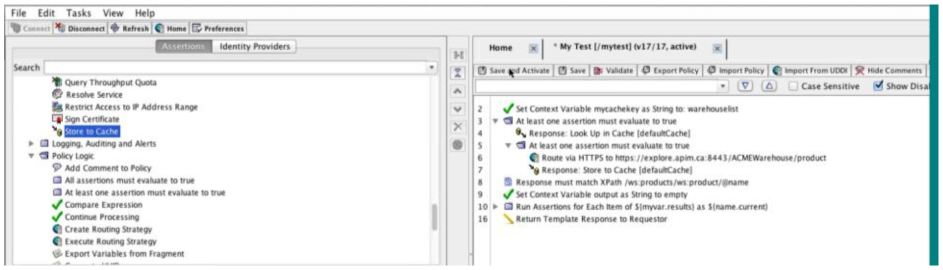
```
* My Test [/mytest] (v17/17, active)

2      Set Context Variable mycachekey as String to: warehouselist
3    ▼ At least one assertion must evaluate to true
4          Response: Look Up in Cache [defaultCache]
5      Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
6      Response must match XPath /ws:products/ws:product/@name
7      Set Context Variable output as String to empty
       ...r Each Item of ${myvar.results} as ${name.current}
       ... Response to Requestor
```

Cache Lookup Properties

Cache ID:	defaultCache
Cache entry key:	${mycachekey}
Maximum acceptable age:	20 seconds
Content-type override	

Assertion: All assertions must evaluate to true Warning: Policy validation feedback has been disabled in the Preferences

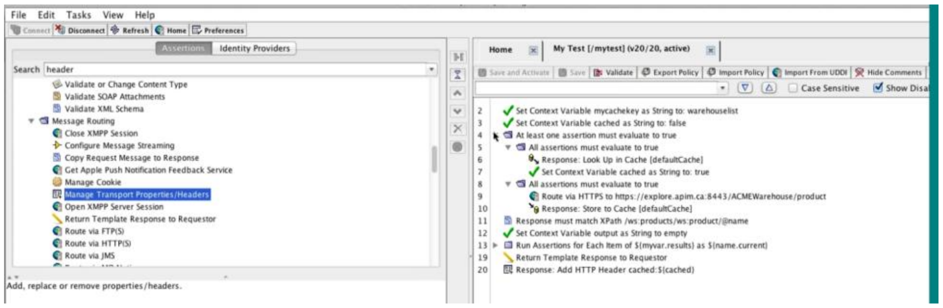
Add logic to check the cache first, if not there, look up on the backend service and store it to cache limiting the life of the cache to 20 seconds.



```
* My Test [/mytest] (v17/17, active)

2      Set Context Variable mycachekey as String to: warehouselist
3    ▼ At least one assertion must evaluate to true
4          Response: Look Up in Cache [defaultCache]
5        ▼ At least one assertion must evaluate to true
6              Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
7              Response: Store to Cache [defaultCache]
8      Response must match XPath /ws:products/ws:product/@name
9      Set Context Variable output as String to empty
10   ▶ Run Assertions for Each Item of ${myvar.results} as ${name.current}
16     Return Template Response to Requestor
```

You can also change a value to true or false and then output this into the http header to determine if it is using the cache values. See line 20. Note you will need to test this using the rest client because you cannot see header values in Firefox.

Policy sample exported as “looping plus cache.”



```
My Test [/mytest] (v20/20, active)

2      Set Context Variable mycachekey as String to: warehouselist
3      Set Context Variable cached as String to: false
4      At least one assertion must evaluate to true
5        ▼ All assertions must evaluate to true
6              Response: Look Up in Cache [defaultCache]
7              Set Context Variable cached as String to: true
8        ▼ All assertions must evaluate to true
9              Route via HTTPS to https://explore.apim.ca:8443/ACMEWarehouse/product
10             Response: Store to Cache [defaultCache]
11     Response must match XPath /ws:products/ws:product/@name
12     Set Context Variable output as String to empty
13   ▶ Run Assertions for Each Item of ${myvar.results} as ${name.current}
19     Return Template Response to Requestor
20     Response: Add HTTP Header cached:${cached}
```

Another way to test the branching is to turn on debug for the service policy (left side, right click). Then add break points and start, use step into function and look at the data in the window in debug.