WAY2LEARN

LAB- Transformers

In this lab, you will be working on **01-transformers-basics-start** project under **05-transformers** section

In this lab you will understand

a) how to write data weave expressions

b) How to use transform message component

STEP 1

We want to make a Http POST request with json content. We want to transform it to XML and Java

1)create a mule configuration file with name "transformerdemo"

2)Drag a Http Endpoint and configure it to listen for requests at URL http://localhost:8081/transform

3) Drag "transform message" component and a logger after it. Configure logger to log the payload.

Double click on transform message. In the Transform Message Properties view, look at the Input, Transform, and Output sections.

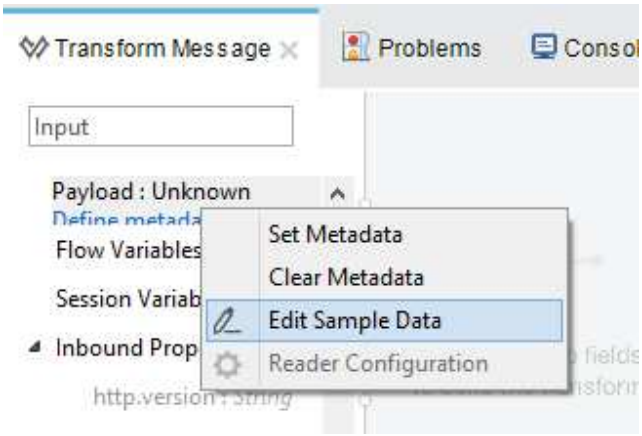
In the Transform section, locate the drop-down menu at the top of the view that sets the output
type; it should be set to Payload.

Beneath it, locate the directive that sets the output to application/java.

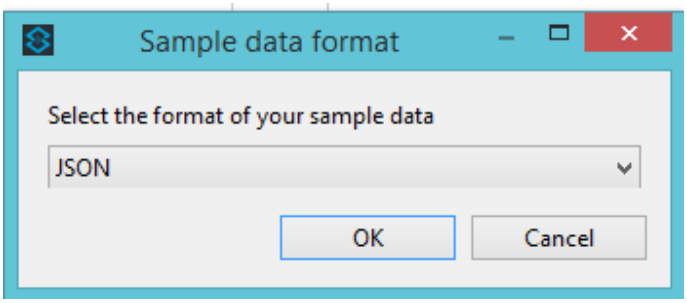
Delete the existing DataWeave expression (the curly braces) and set it to payload.

In the Input section of the Transform Message Properties view, right click Payload: Unknown
Define metadata.

Click the Edit Sample Data button

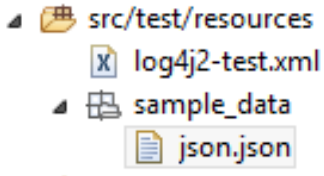


In the Sample data format dialog box, select JSON and click OK

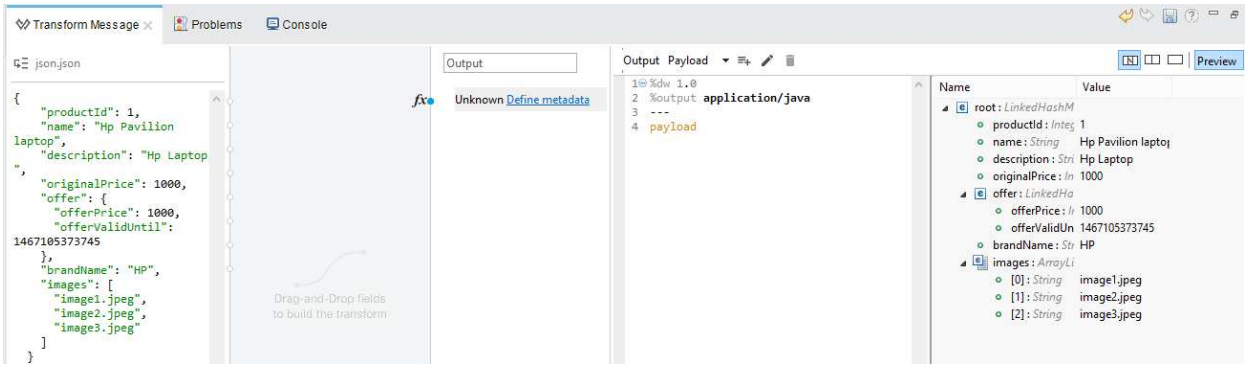


In the new payload window that is created in the Input section, replace the sample JSON data with the contents of file src/main/resources/product.json

Observe that there is a file with name json.json created under src/test/resources/sample_data folder



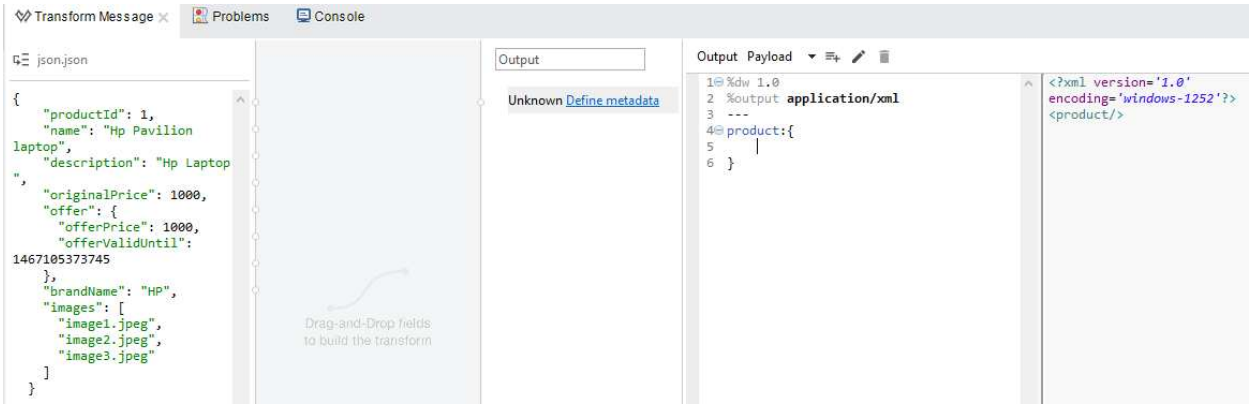
Now click on the preview button in the output section. you should see the sample output for application/java as below :



4) In the Transform section, change the output type from application/java to application/json. You should observe the same json in preview

5) Now change the output type to xml. It shows error as root tag is required for xml.

Now change the expression as product :{} . it looks like below:



Change the expression as below and observe the preview

```
product:{
       pid:payload.productId,
       productName:payload.name,
       offerPrice:payload.offer.offerPrice,
       image1 :payload.images[0]
}
```

Change the expression as below and observe the preview

```
product:{
       pid:payload.productId,
```

```
        productName:payload.name,
        offer:{
                offerPrice:payload.offer.offerPrice,
                offerValidUntil:payload.offer.offerValidUntil
        },
        image1 :payload.images[0]
}
```

we want "pid" to be attribute of product tag. So, change the expression as shown below :

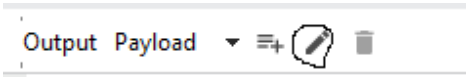
```
product @(pid:payload.productId):{

        productName:payload.name,
        offer:{
                offerPrice:payload.offer.offerPrice,
                offerValidUntil:payload.offer.offerValidUntil
        },
        image1 :payload.images[0]
}
```

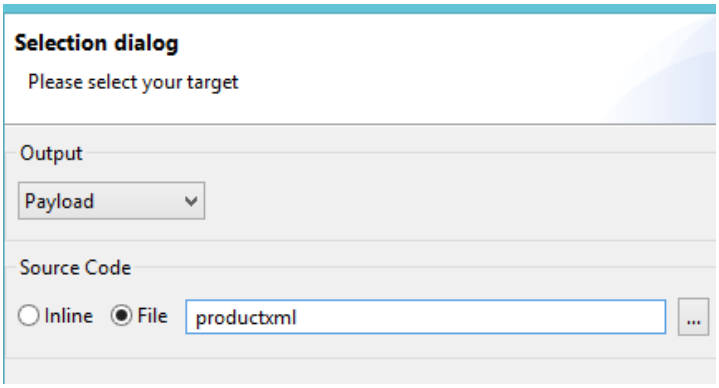
Now switch to xml view and observe that all dwl expressions are inline in xml which makes xml difficult to read.

So, we want to externalize the dwl into separate files.

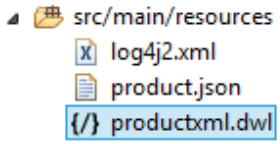
Now click on edit button highlighted below to edit the current target for dwl .



Select target as a file with name productxml as shown below and press ok :



You should observe that a file with name productxml.dwl is created as shown below

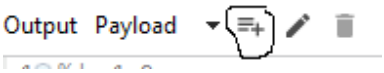


Now switch to xml view and observe that the transform-message configuration looks like below :

```
<dw:transform-message doc:name="Transform Message">
     <dw:input-payload doc:sample="sample_data\json.json"/>
     <dw:set-payload resource="classpath:productxml.dwl"/>
 </dw:transform-message>
```

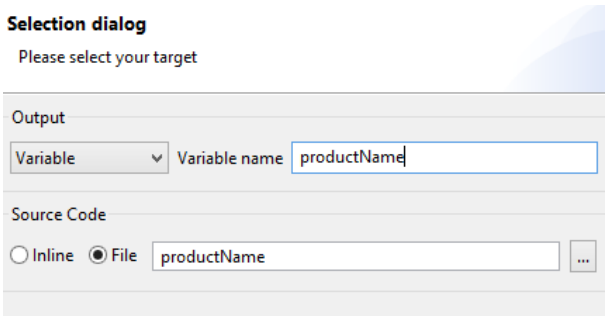
We want to write an expression which will evaluate to productName and store it in a flow variable with name productName.

So, Create a new target by clicking on "+" highlighted below :



Then Select Variable from Dropdown and give the variable name as productName.

Select the radio button "File" and give the filename as productname as shown below and press OK:



WAY2LEARN

Write the expression as payload.name

Similarly create session variable with name brandName and write the corresponding expression. Make sure that you store that expression in a new file "brand.dwl"

Keep a break point on Logger run the application in debug mode.

Open POSTMAN and give Http POST request to http://localhost:8081/transform and send the content of src/main/resources/product.json in body.

Dont forget to add a request header "Content-Type" with value "application/json".

Once you switch to debug perspective after making request, observe that flowvariable productname and session variable brandname will be created.

This is the end of the Exercise