LAB- Munit

STEP1

In this lab, you will be working on **01-munit-basic-start**

Open muletrainingdb.sql under src/main/resource and execute it on the database so that data is populated into product table

1) Open munit-basic.xml and observe the code.

Run the application and give a request to

http:// localhost:8081/products?id=2

You should get below json as response.


```
{
   "id": 1,
   "name": "Hp Pavilion laptop",
   "brand": "HP",
   "offerPrice": 1000.0,
   "price": 1000.0
}
```


Now we want to write a Functional Test case for this scenario.

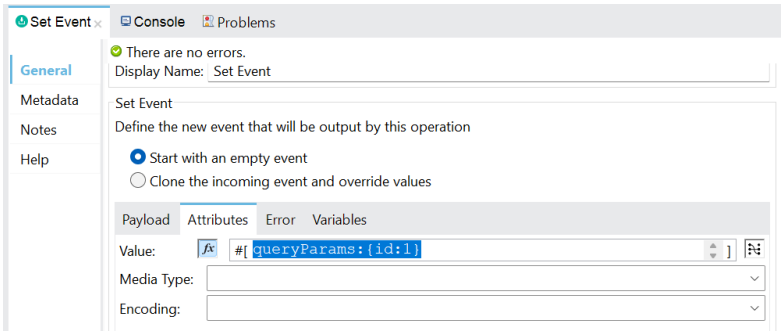
Right Click on flow with name **mainflow** and select "MUNIT->create new bank test for this flow"

You should see that munit-basic-test-suite.xml is created under src/test/munit folder

Select Munit module in the Mule palette and drag "SetEvent" before flow reference.

We want to create a event with empty payload and attributes should contain a query parameter "id" with value 1

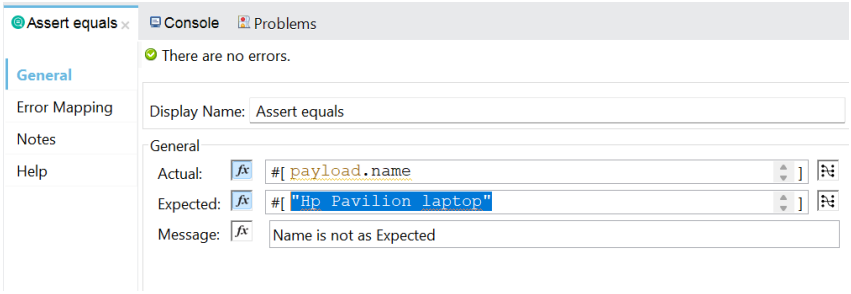
On SetEvent component , Configure queryParams:{id:1} as value for attributes as shown below



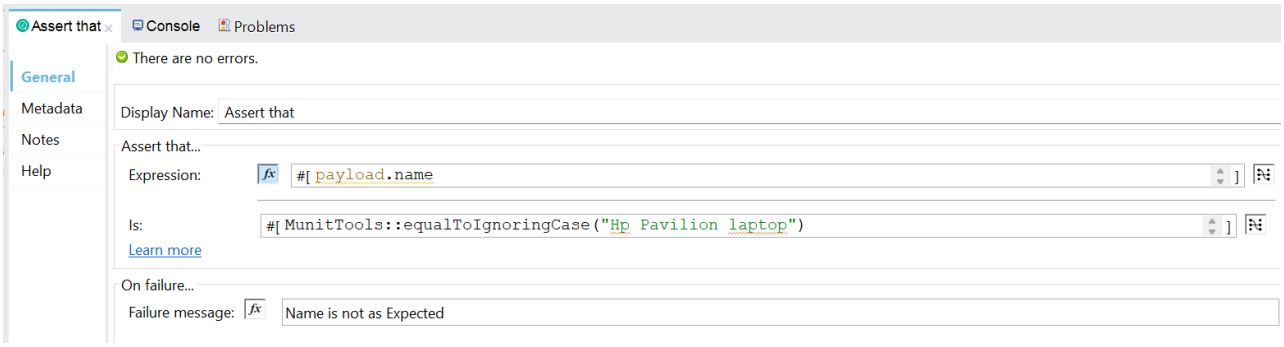
We want to Assert that the payload.name is "Hp Pavilion laptop" after invoking the mainflow.

There are 3 ways of doing the same

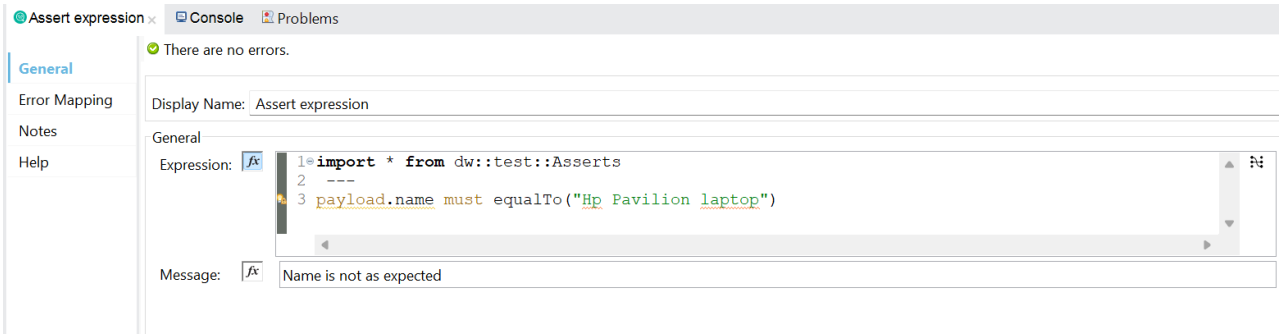
Drag "Assert Equals" in Validation part of test case and configure as shown below:



Drag "AssertThat" " in Validation part of test case and configure as shown below:



Drag "Assert Expression " in Validation part of test case and configure as shown below:



Right click on the TestSuite file and select "Run Munit Suite"

You should observe that test case is successful.

On SetEvent component , change attributes value to queryParams:{id:2}

Now run the test suite and observe that it fails.

Again change the attributes value to queryParams:{id:1}

You can visit the mulesoft documentation page for various Munit matchers that can be used in "Assert That "

https://docs.mulesoft.com/munit/3.1/munit-matchers

Can u Modify Assert that to assert payload not null?

Can you modify "Assert that " to assert payload.name as "Hp Pavilion laptop" **and** not null?

Can you modify "Assert that " to assert payload.name as "Hp Pavilion laptop" **or** "Apple"

Can you modify "Assert that " to assert payload.name as "Hp Pavilion laptop" **or Not Null or** contains string "Hp" or starts with "Hp"

Can you modify "Assert that " to assert payload.offerPrice is less than or equal to 20000

Read the documentation at below URL to understand how to use "Assert Expression"

https://docs.mulesoft.com/munit/2.3/assertion-expression-processor

2) We want to write another test case which will verify that "productsbyIdFlow" is called

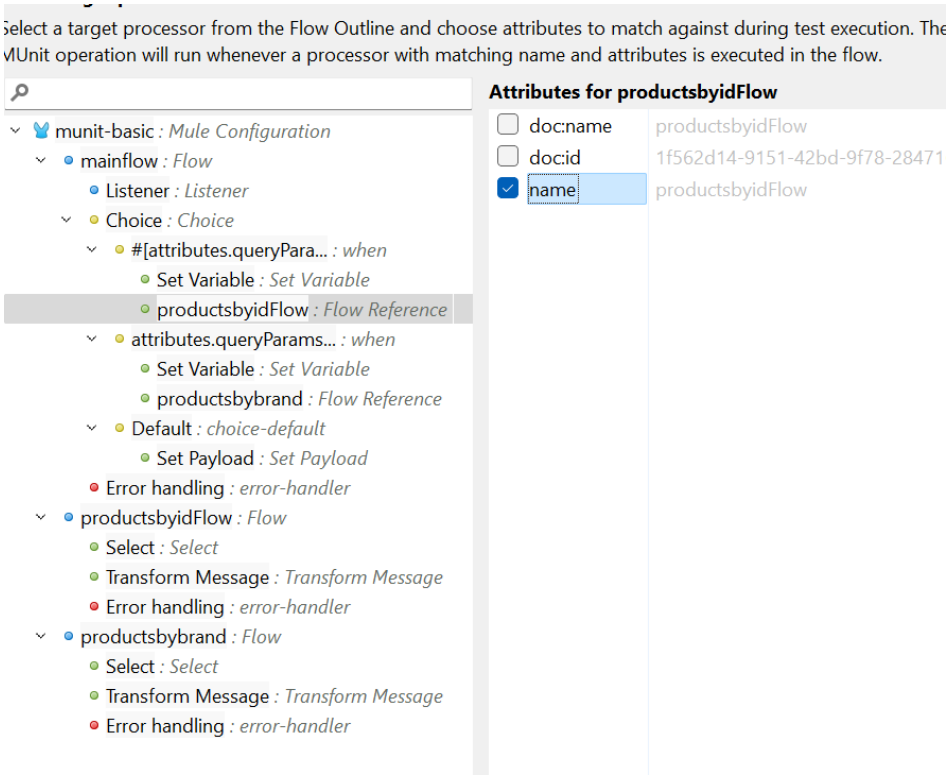
Right Click on flow with name **mainflow** and select "MUNIT->create new bank test for this flow"

One more Test case should be added to same suite xml file.

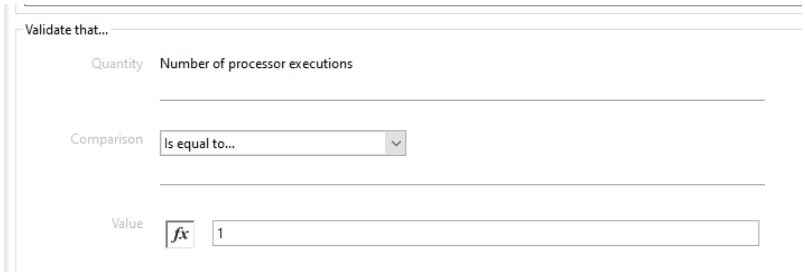
Copy "SetEvent" in previous test case and paste it before flow reference of the newly generated test case

Now Drag "VerifyCall " in the Validation part

Click on Pick Processor button and select "productsbyIdFlow" flow reference and select name attribute as shown below



Configure it to validate the number of processor executions is equal to 1 as shown below:



Run the suite and make sure that both the test cases pass

3)If you have observed, both the test cases have "SetEvent" as same.

We want this "Set Event " to be called before every Test case.

So, drag "Before Test" from the Mule palette and drag "set Event" into it.

Make sure that both the flow don't contain Set event now.

Run the suite and make sure that both the cases are passed.

4) Till now, we have written Integration Tests. Now, we want to write Unit Test case

When we write Unit test cases, all the external calls to other flows have to be mocked.

Open munit-basic.xml .

Right Click on productsByIdFlow and select "MUNIT->create blank test for this flow"

One more Test case should be added to same suite xml file.

We want to Assert that the payload after invoking productsByIdFlow as


```
{
        "original_price": 1000.0,
        "product_id": 1,
        "name": "Hp Pavilion laptop",
        "offer_valid_until": "2016-06-27T10:45:56",
        "description": "Hp Laptop ",
        "brand_name": "HP",
        "offer_price": 1000.0
    }
```

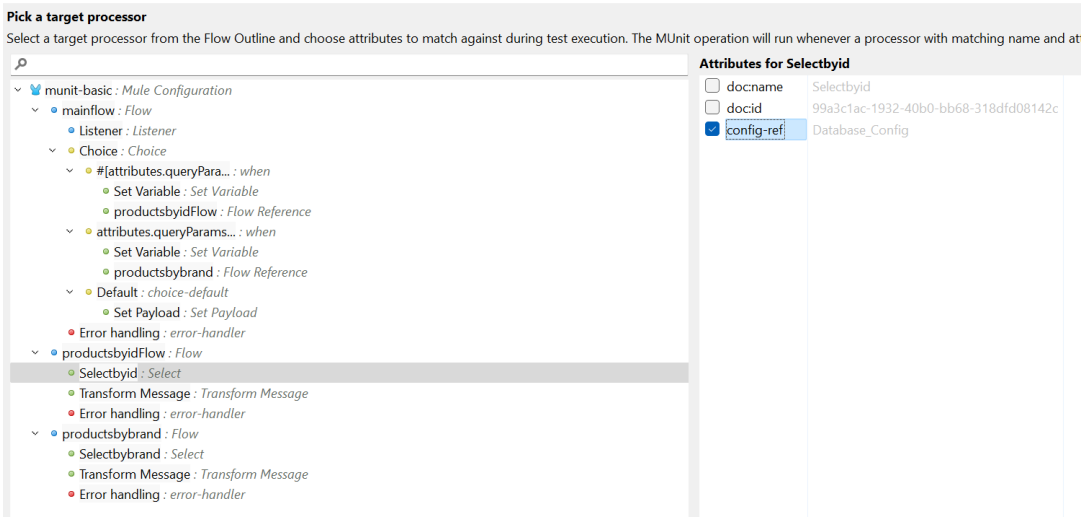
So, drag "Assert Expression" in the Validation block of test case and configure the expression as shown below :

```
import * from dw::test::Asserts
 ---
payload must equalTo{
        "original_price": 1000.0,
        "product_id": 1,
        "name": "Hp Pavilion laptop",
        "offer_valid_until": "2016-06-27T10:45:56",
        "description": "Hp Laptop ",
        "brand_name": "HP",
        "offer_price": 1000.0
    }
)
```

We want to mock the call to 'Database Select'.

Now drag "Mock when" in to Behaviour part of Test case.

Click on Pick Processor and select "SelectById" as shown below



Inside "Then return " section, configure payload with below expression

```
[
    {
        "original_price": 1000.0,
        "product_id": 1,
        "name": "Hp Pavilion laptop",
        "offer_valid_until": "2016-06-27T10:45:56",
        "description": "Hp Laptop ",
        "brand_name": "HP",
        "offer_price": 1000.0
    }
]
```

Thas all. Now you mocked the call to Database Select.

Run the suite and make sure that all the test cases pass.

STEP2

In this step, you will understand how to externalize the input data and assert expressions to external dwl files

In this step, you will be working on 02-munit-externalizing-input-assert-expressions-start. So, import it

Open externalizing-in-munit.xml and observe that the flows are exactly same as the ones in step1

We have already written a test case for getptoductby id with an existing id.

Open externalizing-in-munit-test-suite.xml under src/test/munit and observe the test case.

Click on "Set Event" processor and select attributes tab. Observe that we have written an expression already. We want to externalize this to a dwl file.

Create a folder with name getproductbyid under src/test/resources

Under that folder, create a file with name getproductbyid-withexisting-id-setevent-attributes.dwl

Copy the expression from attributes tab of Set Event in to this file

Now write the below expression in the value for attributes

readUrl("classpath://getproductbyid/getproductbyid-withexisting-id-setevent-attributes.dwl")

Now We want to externalize the assert expression also .

So, create a file with name getproductbyid_withexisting_id_assert_response.dwl under getproductbyid folder under src/test/resources

Copy the expression inside "Assert Expression " component into this file

Now configure the following expression inside Assert Expression

```
import   getproductbyid::getproductbyid_withexisting_id_assert_response
 ---
getproductbyid_withexisting_id_assert_response::main({payload:payload,attributes:attributes,vars:vars})
```

We are done with externalizing .

Run the test case and make sure that it succeds

STEP3

NOTE: Documenting this was difficult. Better if u follow along with my video and do this exercise

In this step, you will understand how to create parameterized tests

In this step, you will be working on 03-munit-parameterised-start project. So, import it

Open Parameterization-in-munit.xml and observe that it contains same flows which we used in earlier steps.

We want to test this flow by passing brandName as query parameter.

Open parameterization-in-munit-test-suite.xml under src/test/munit folder and observe that we have already created on test case for you

Click on "set event" processor and click on attributes tab. Observe that we have given below expression for value. It is expecting a parameter called as brand.

```
{queryParams:{brand: Mule::p('brand')}}
```

Click on "Assert Expression" and observe that it is configured with parameter as shown below:

```
import * from dw::test::Asserts
 ---
payload must haveSize(Mule::p('size') as Number)
```

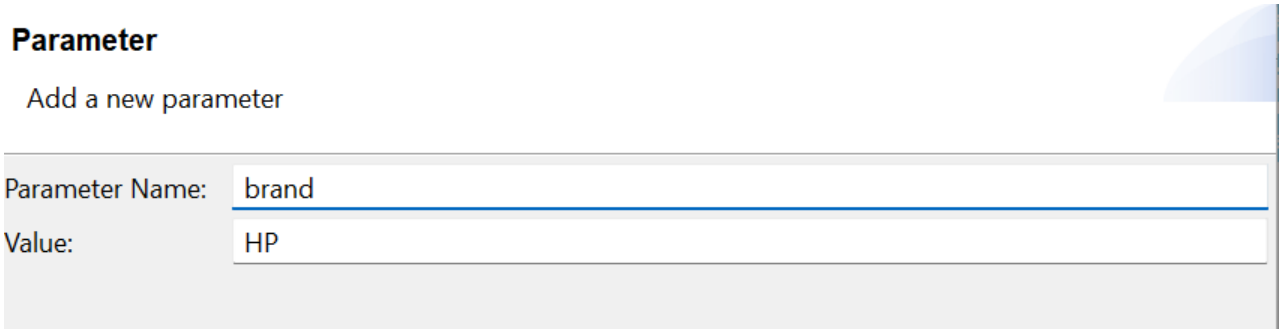
How to pass parameters?

Click on Global Elements and edit "Munit Configuration"

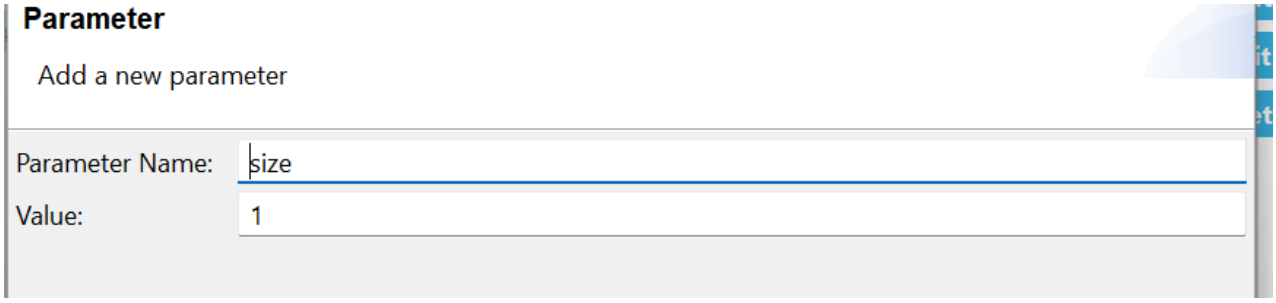
In the Parameterizations section, Click on + icon and add a parameter with name "brandhp".

Now select brandhp and click on + on the right sideto add a paramatername

and value as shown

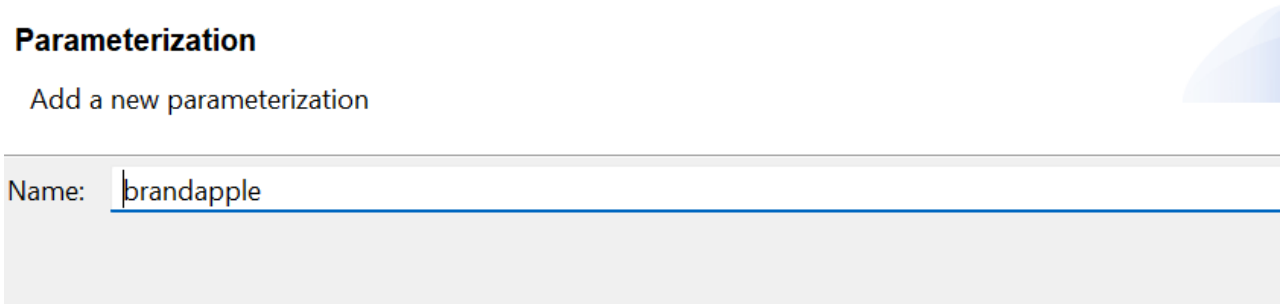


Add one more parameter as shown below

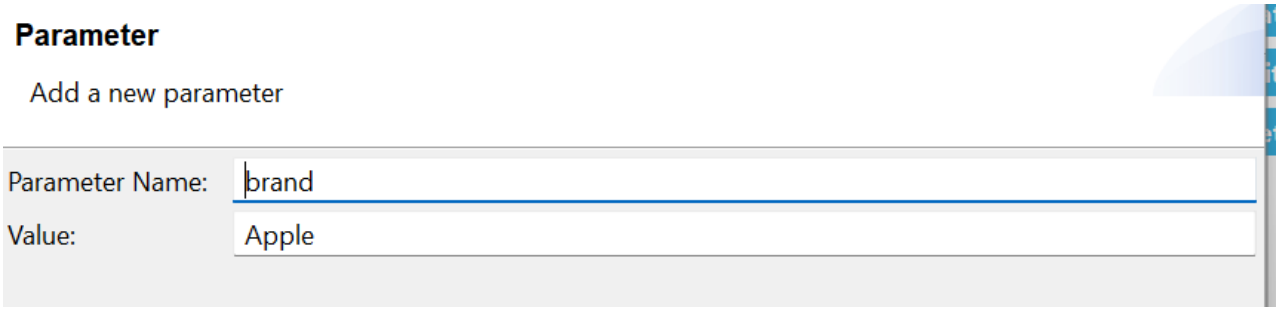


So, u have added 2 parameters brand and size for parameterization with name brandhp

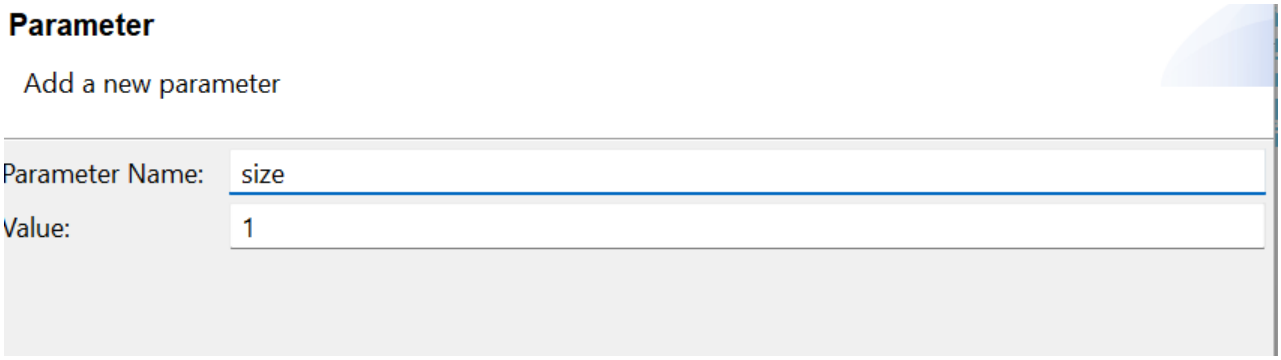
Similarly, click on + on the left side to add a new parameterization with name brandapple



Now select brandapple and click on + on the right side to add a paramatername and value as shown



Add one more parameter as shown below



That's all. U have added to parameters for this test

Run the test case and observe that 2 cases will be executed for 2 parameters.

NOTE: Documenting this was very difficult. Better if u follow along with my video and do this exercise

This is the end of the Exercise