

Practice Testing Services

Web services are becoming more numerous in applications and therefore it is necessary to have alternatives to verify them quickly and efficiently.

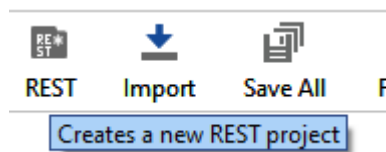
The SOAPUI tool allows you to automatically perform some of the checks on web services. The idea is that you can automate these checks.

For this workshop two possibilities of the tool will be used:

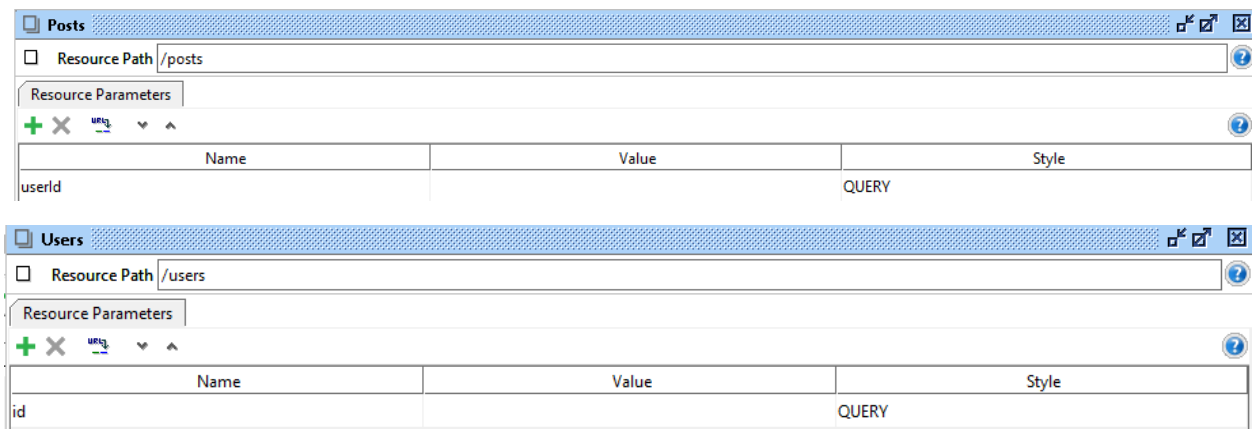
- 1 – For the scenario in which we require to expose a web service and want to verify its correct deployment.
- 2 – To isolate a test unit that consumes a third-party web service.

For the first one will use the possibility of creating test suites that provides SOAPUI

- A. The test case will be generated based on REST technology. For that it is necessary to address to SOAPUI and follow the following steps:
- B. Creates a new REST project

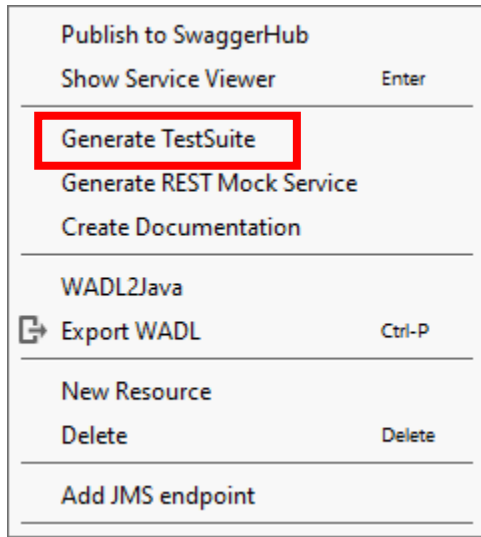


- a.
- b. Add the following web service: jsonplaceholder.typicode.com
- c. Add the resources Posts, User, Photos

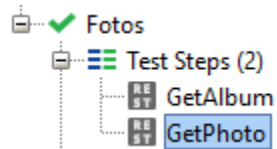


Photos		
Resource Path /photos		
Resource Parameters		
Name	Value	Style
albumId		QUERY
id		QUERY

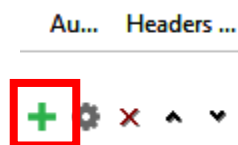
- C. Generate a new test suite
- D. Right click on the added service and select "Generate Test Suite"



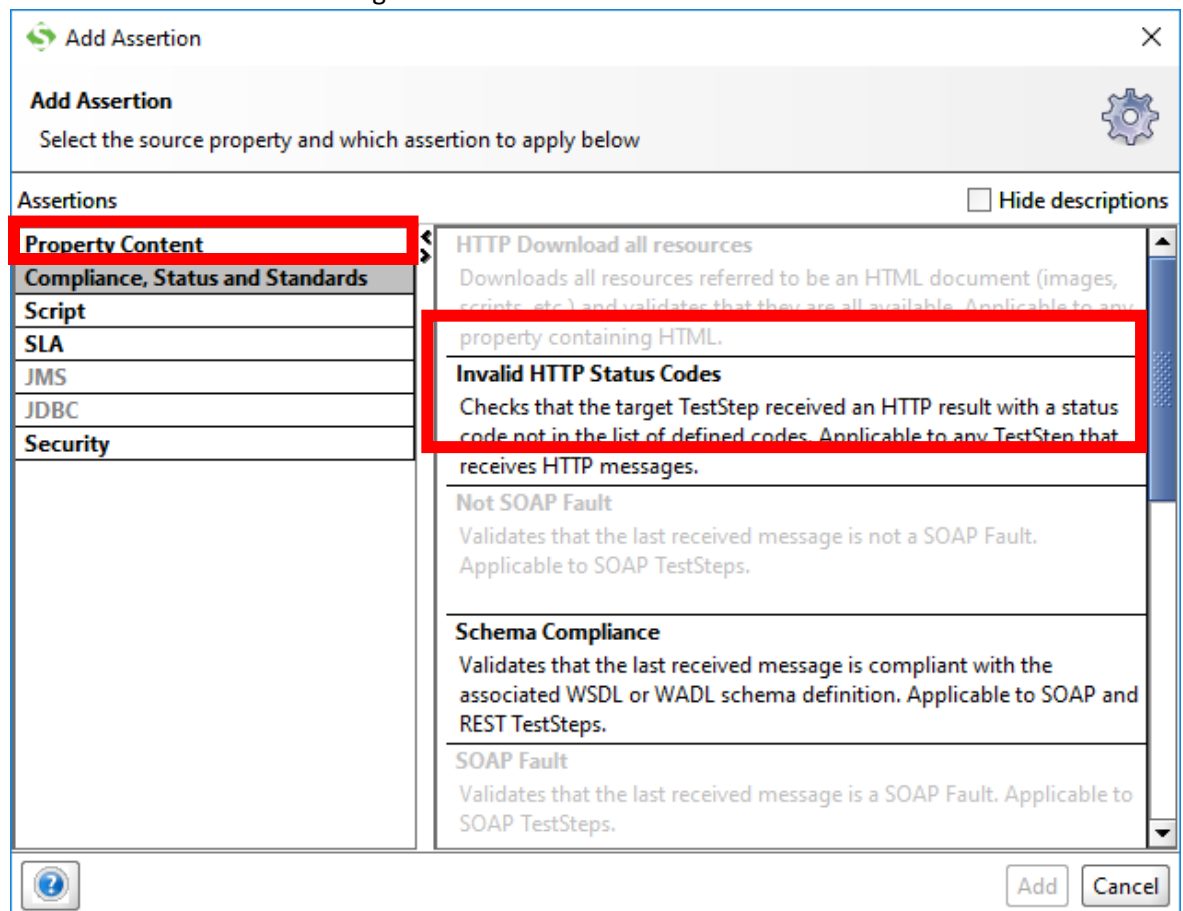
- a. This will create a test suite with a set of test tests for each request that the service has
- b. Select the test case "Fotos" and the step "GetPhoto"



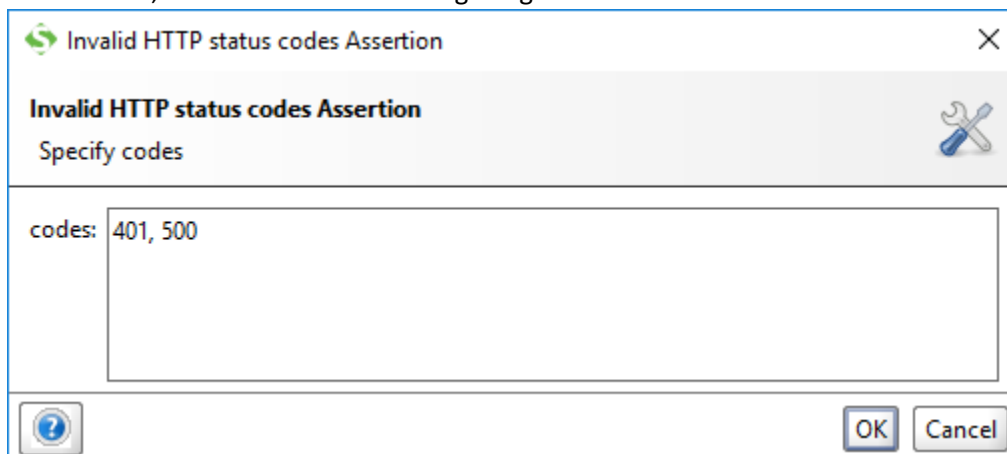
- c. Open the step and make click on the button "Assertions"
- d. Press the button "Add Assertion"



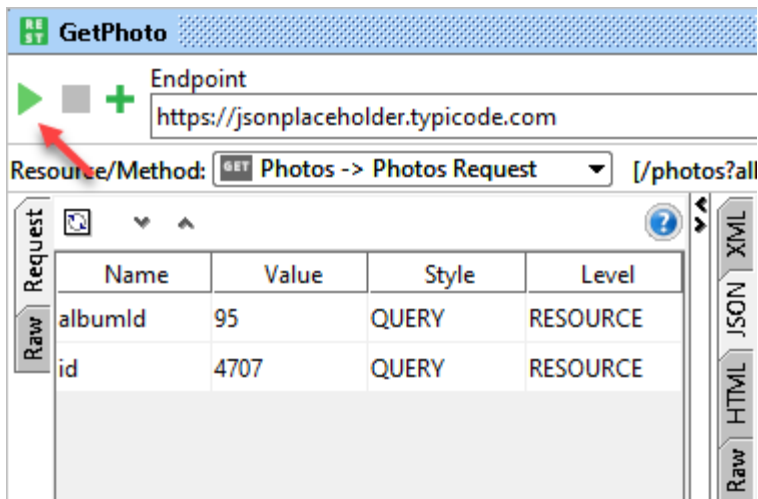
- e. Select the assertion of the image shown below



- E. Press the button "Add"
- F. Add the values to be checked that are not received as a result of the execution of the request "Get Photo", as shown in the following image:

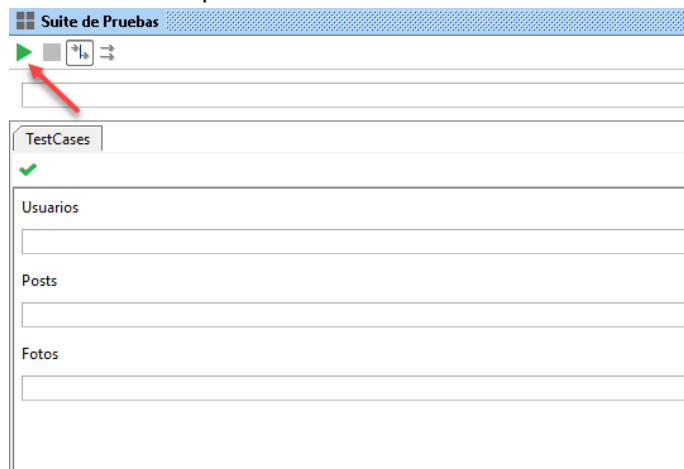


- G. Execute Test



Exercise:

1. Generate an assertion for the "GetAllUsers" method of the test case "Usuarios" After generated execute the complete test suite.



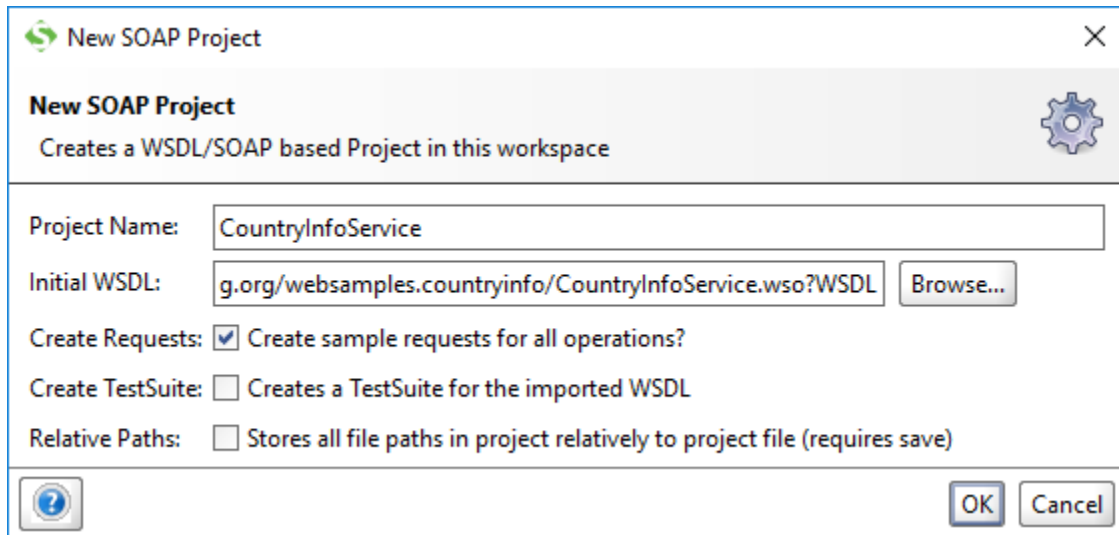
- 2.

For the second case, the goal is to isolate a functionality dependent on a web service of a third party to achieve, for example, a unit test on functionality.

In order to achieve this goal it is necessary to create an automatic response of the web service through a mock of the service. To do this, it is necessary to perform the following steps:

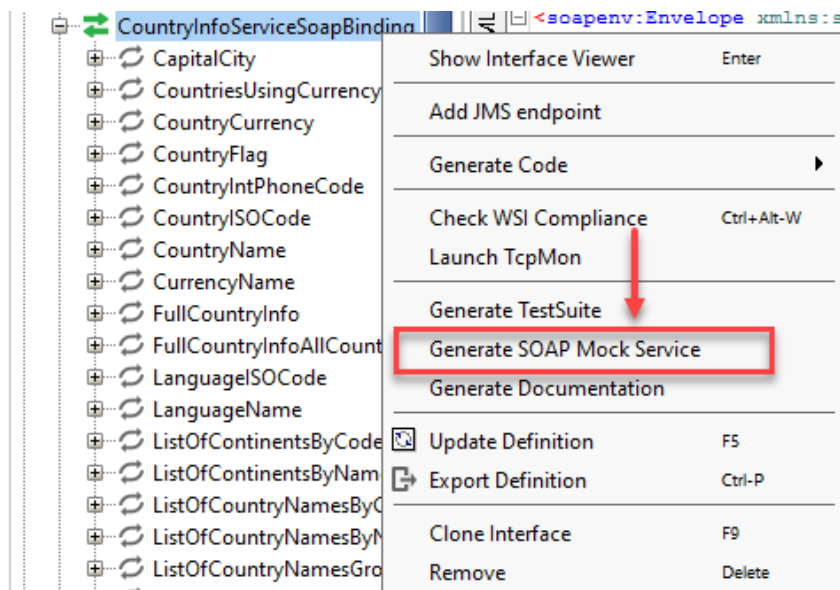
A. Creating a new Project

- a. Start a new SOAP project using the following WSDL
<http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL>

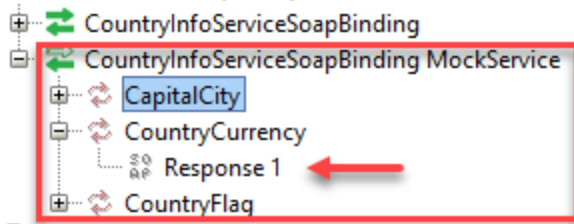


B. Creating a MockService

- a. Right-click on the Web Service created and select Generate MockService.

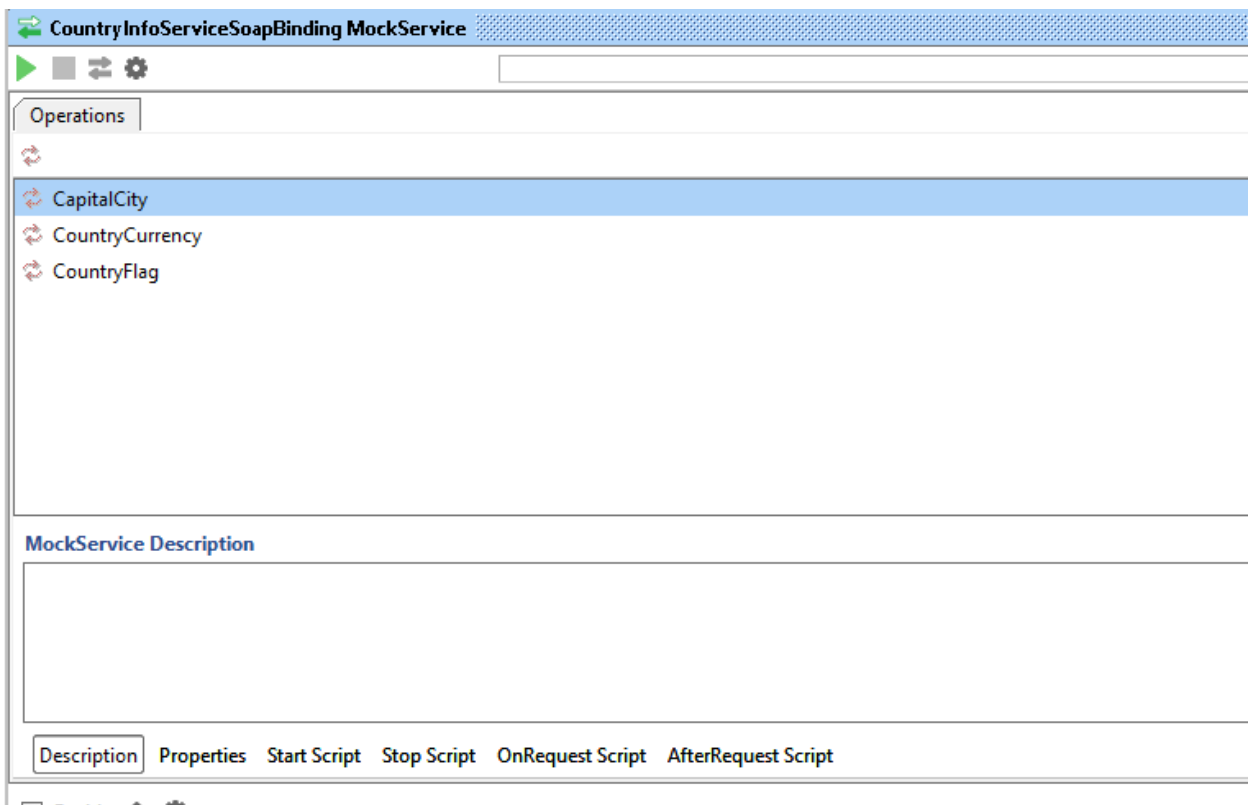


- b. In the dialog *Generate Mock Service* you can specify the local port/path for the service. Click OK.
- c. Enter the name of your MockService in the Name dialog and click OK.
- d. After creating the MockService, you should get a MockService with one operation and one request for each operation.

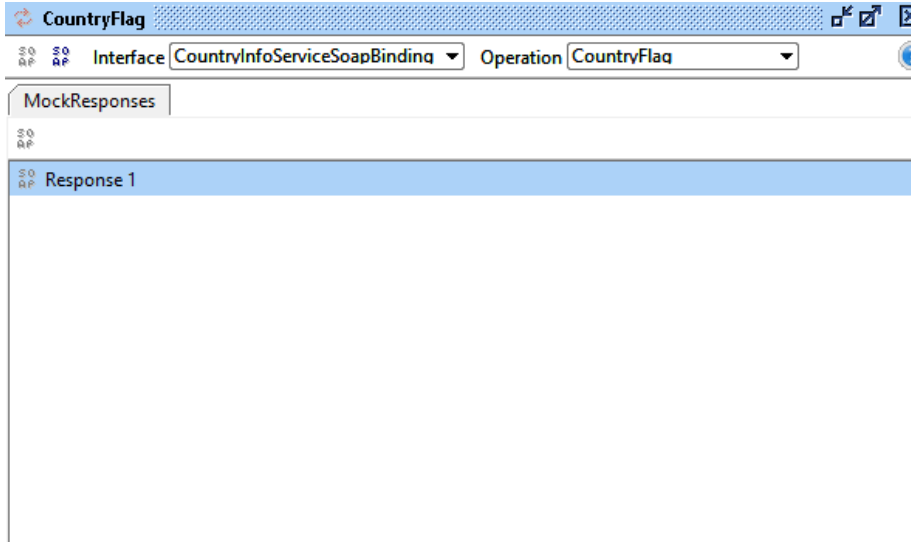


C. Editing a MockService

- a. Now, double click on the MockService to see the MockService editor. In the editor we can see a list of all operations for the service, as well as a request and response log which logs all incoming requests that have been handled by the MockService.



- b. Double Click on the CountryCity to see the Response that we have in the MockService:




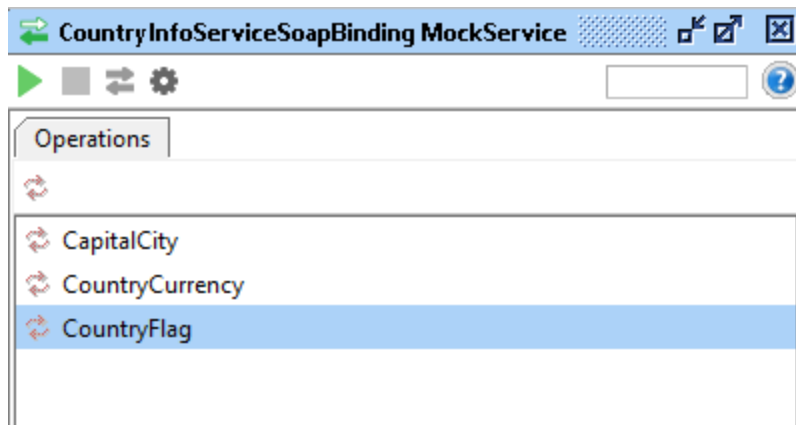
- c. As you can see we do only have one. Let's edit it. Double click on it to open the Response Editor.



- d. Edit the CountryFlagResult to:
<http://www.oorsprong.org/WebSamples.CountryInfo/Images/Uruguay.jpg>.

D. Invoking a MockService

- a. First we must start the MockService. Click  in the MockService editor. This will start the service on the configured port and path.



E. Used from GeneXus the Mock created

- From GeneXus open the kb (name of the KB: **KBUpToDateGeneXus15**) -> Ctrl + Shift + O
- In order to use the web service consumed we first need to make some modifications to the "Country" transaction

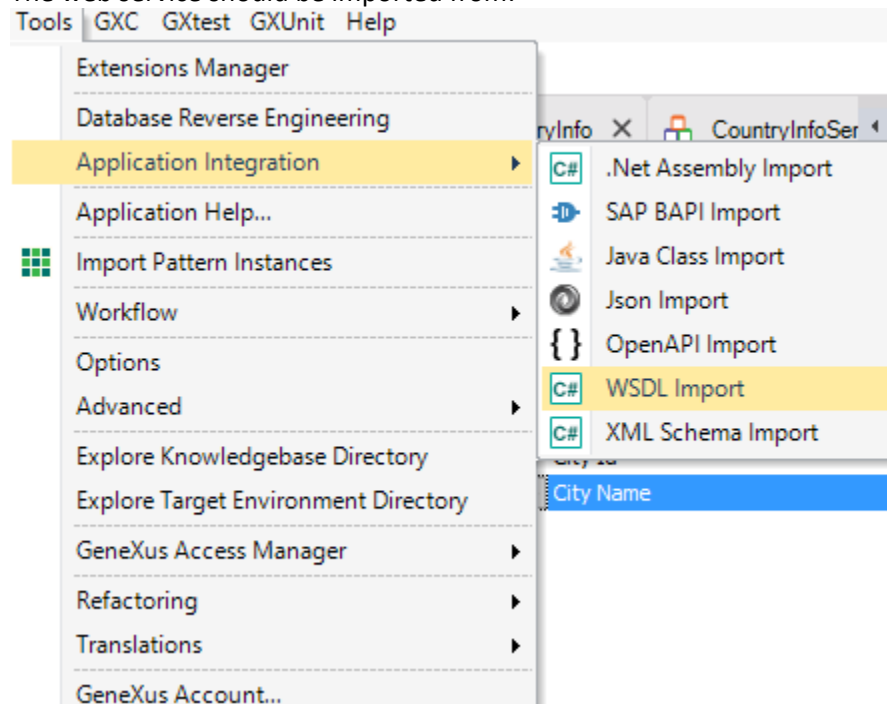
Add attribute "CountryISO" as Character(20)

And a new sub-level "City" to which it will be added the attributes "CityId" type Id y "CityName" type Name

Name	Type	Description	Formula	N
Country	Country	Country		
CountryId	Id	Country Id		No
CountryName	Name	Country Name		No
CountryISO	Character(20)	Country ISO		No
City	City	City		
CityId	Id	City Id		No
CityName	Name	City Name		No

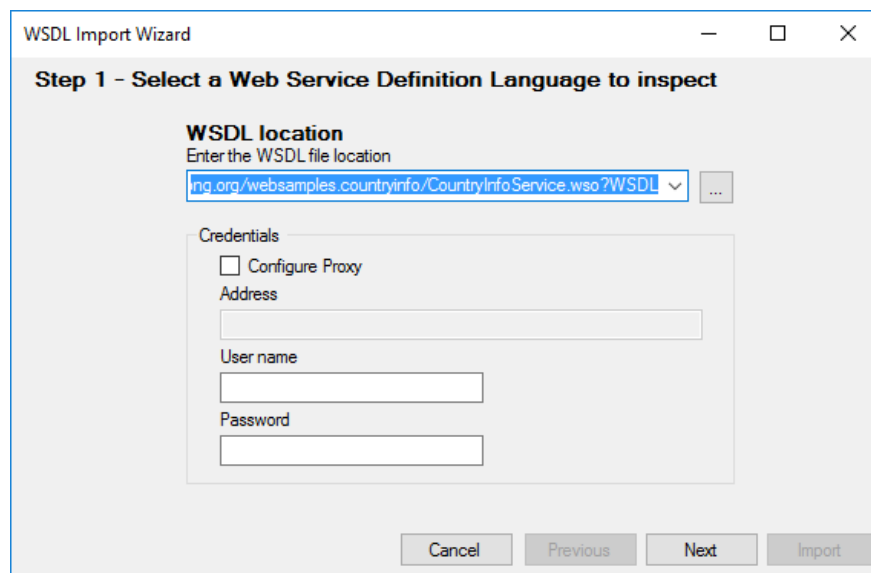
Then the application should be executed and the corresponding ISO should be added to the existing countries

The web service should be imported from:



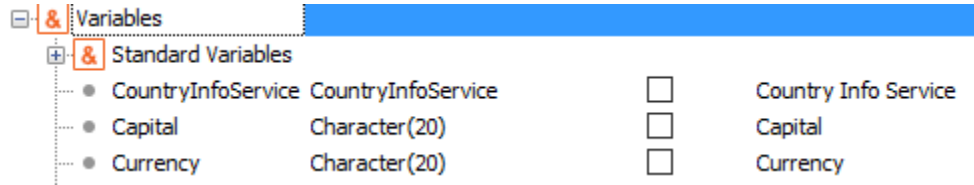
- c. Import via the WSDL Import Wizard the web service: <http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso>

This will create an "External Object" of type WSDL (Web Service) to be invoked as another type of data



- d. Then create a new GeneXus procedure called "CountryInfo" which will print the name, capital, currency and flag of each country in a PDF. OBS: Only name will not be provided by the web service.

- e. Create the following variables:



In the Source of the procedure define the For Each that will cross the Country table to provide the data to load in the layout of the procedure. The structure should be as follows:

```
print Title
for each Country
    &Capital = &CountryInfoService.CapitalCity(CountryISO)
    &Currency = &CountryInfoService.CountryCurrency(CountryISO).sName
```

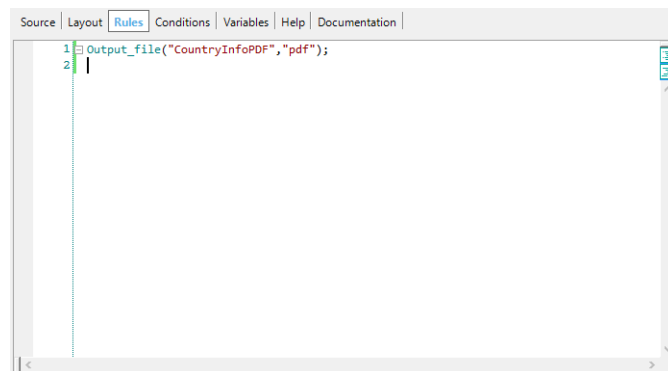
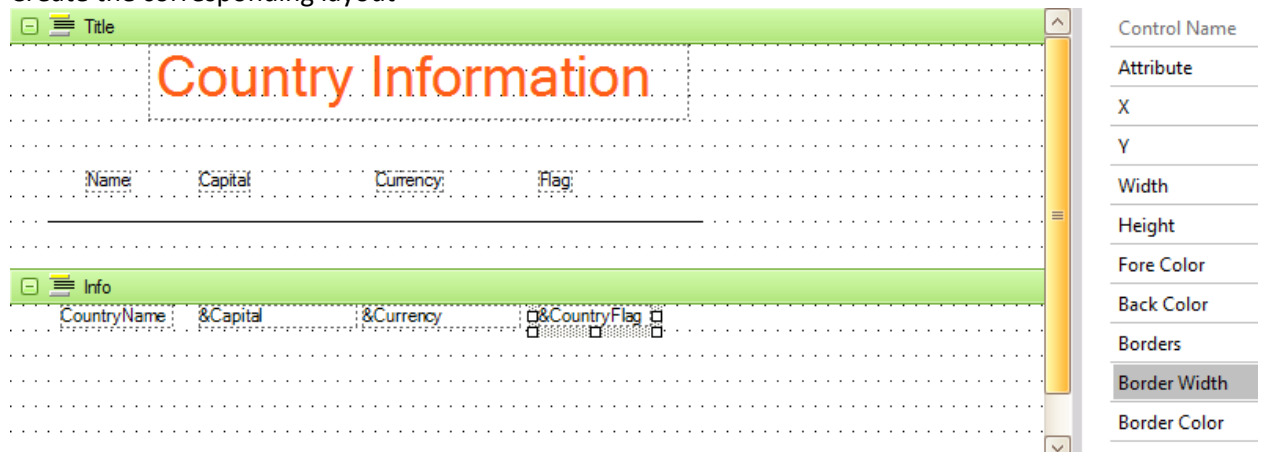
- f. The variables are almost ready. However, it is necessary to add the variable that allows to obtain the image of the flags. It is observed that the web service returns the url of the image of the flag (<http://www.oorsprong.org/WebSamples/CountryInfo/Images/Uruguay.jpg>) so a variable must be created & Flag to store the url and another & CountryFalg of the image type to store the image of the flag itself.



- g. Add in the procedure source the information that will be loaded from the For Each path



Create the corresponding layout



- h. Perform the following settings and then execute the procedure (or the application if you need to create countries)

▼ Procedure: CountryInfo	
Name	CountryInfo
Description	Country Info
Module/Folder	Root Module
Main program	True
Call protocol	HTTP
Execute in new LL	False
Test Case	False
Object To Test	
Qualified Name	CountryInfo
Object Visibility	Public
▼ Web information	
Auto compress	Use Environment property...

- i. Execution should fail. What failed? If you can't find the error, go to the following [Attachment](#)
- j. After correcting the error, it is re-run and the response from the web service list should be returned.
- k. But this is not using the Mock response created. To achieve this it is necessary to change the "Address" property in the properties of each service, by the Mock address.

Protocol	SOAP 1.1
Style	Document
Use	Literal
Enable MTOM	False
Address	http://gmanceboNTB:8088/mockCountryInfoServiceSoapBi
Port Type Name	
Port Type Namespace	http://www.oorsprong.org/websamples.countryinfo
Action	
XML Name	CountryFlag
Request Namespace	http://www.oorsprong.org/websamples.countryinfo
Response XML Name	CountryFlagResponse
Response Namespace	http://www.oorsprong.org/websamples.countryinfo
v XML Return Element Info	

- l. Ask for teaching support to carry out this activity to set up a mock sequential response.
- m. After the settings have been made, run the program again and you can display the contents this time consuming Mock information.

Attachment:

The Country transaction has set the Data Provider property to "Yes" and changed the transaction settings by adding new attributes. This data load is no longer valid. The populate Data property must be changed to "False"

