**Using Python Scripts in the Robot Framework**

Calling Python in the Robot framework.



Since the Robot framework is built on top of Python, importing Python modules inside the Robot framework is a simple process. It allows tremendous flexibility to your code and helps you create your own custom keywords that are better suited for your task than its robot counterparts.

Learning Robot framework’s keywords is not worth your cognitive power unless the company you’re working for has specially told you to use pure Robot keywords, in which case, may God bless you. For most of our test cases, we would prefer to write the logic in Python and import the keyword in Robot. It makes the automation process faster and cleaner.

In this short article, we will learn to use Python libraries inside your test suite in the Robot framework.

Let’s start by creating and starting the virtual environment.

$ mkdir learning-robot  
$ cd learning-robot  
$ virtualenv env  
$ source venv/bin/activate

This is the folder structure:

.  
├── func.py  
├── ClassA.py  
├── test.robot  
└── venv

**Importing python function**

Let’s start by creating a basic Python script that adds 1 to the integer.

***func.py***

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

We have a simple function that adds 1 to the input number.

***test.robot***

To import the Python script inside Robot, we use the keyword Library in the Robot file under \*\*\*settings\*\*\*.

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

To call the function, we use <file\_name> <dot> <function name>.

We can assign keywords for Python functions. Generally, it’s recommended that for Python files that are imported into Robot, keep the following import in the Python file:

try:  
 from robot.libraries.BuiltIn import BuiltIn  
 from robot.libraries.BuiltIn import \_Misc  
 import robot.api.logger as logger  
 from robot.api.deco import keyword  
 ROBOT = False  
except Exception:  
 ROBOT = False

To add keywords inside the function, we use the keyword decorator.

***func.py***

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

***test.robot***

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

Here, BuildIn().log\_to\_console does printing in the Robot terminal. If we run $ robot tests.robot, we see the following output.

***output***

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

terminal report for robot

**Import Python classes**

***classA.py***

It is important that the class name matches the filename. Here, both class name and filename is classA.

Kuva, joka sisältää kohteen teksti, näyttökuva, valkokangas

Kuvaus luotu automaattisesti

***test.robot***

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

***output***

For calling the class function, we just have to call <function\_name>. We don’t need to specify the file name.

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

output without ROBOT\_LIBRARY\_SUITE

If we look at classA.py , the log to console is called only when the class is created. In the output, we can see that the object was created 3 times. That’s a lot of overhead and it keeps piling as the number of test cases increase. If we want the object to be created only when the object is created, then we will use ROBOT\_LIBRARY\_SCOPE = ‘TEST SUITE’ in the class.

***classA.py***

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

***output***

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

output with ROBOT\_LIBRARY\_SUITE