ROBOT FRAMEWORK

What is Framework?

A framework is a collection of functions that can be used to achieve particular task.

Python Frameworks: Django, Flask and Robot Framework

Java Frameworks: Spring, Hibernate and Struts

Java Script Frameworks: React, and Angular

➔ Robot Framework Features

1. Robot Framework is a Python and Java based keyword-driven test automation framework
2. Robot Framework is an open source and platform independent framework
3. Robot Framework is a generic test automation framework
4. By default, Robot Framework creates an XML output file and a log and a report in HTML format.
5. Official sit is for more information http://robotframework.org/
6. Provides ability to create reusable higher-level keywords from the existing keywords.
7. Provides tagging to categorize and select test cases to be executed.
8. Provides test-case and test-suite -level setup and teardown.

ROBOT FRAMEWORK

IMP Features in robot Framework

[Setup]

setup is something that is executed before a test case to do required setup for test case execution.

Example: open a file, start process and create a connection

[Teardown]

Tear down will execute after executing the test case to do cleanup process.

Example: close a file, stop process and close a connection

Suite Setup: --

It will execute before executing the all test cases to do required setup for execute all test cases.

Example: open a file, start process and create a connection

Suite Teardown: -

It will execute after executing the all test cases to do cleanup process.

Example: close a file, stop process and close a connection

ROBOT FRAMEWORK

[Tags]

Using TAGS, we can give alias name to the test cases to categorize test cases like (sanity, smoke, regression, Critical).

With tags, we can include or exclude test cases to be executed.

[Documentation]

Allows you to set a free documentation for a test case. That text is shown in the command line output, as well as the resulting test logs and test reports.

Robot framework execution options

```
# To know robot version
  robot --version

# paasing values to variable through command line =>
   robot -v ONE:10 Loop.txt
```

ROBOT FRAMEWORK

 # create log file ==>

```
robot -b debug.log test2_list. robot
```

 # validate the test data ==>

```
robot --dryrun test2_list. robot
```

 # To run particular test case ==>

```
robot -t "Add Two Values" first_test.txt
robot -t Addition -t "Test Case 3" first_test.txt
robot -t  Add* first_test.txt
```

 # TO run a test case based on tag =>

```
robot -i test2 first_test.txt
```

 # TO run all test cases except given tag =>

```
robot -e test2 first_test.txt
```

 # Change logs path ==>

```
robot -d C:\Users\ROBOT_FRAMWEORK\Logs   test2_list. robot
```