Robot Framework in Python

Difficulty Level : Medium • Last Updated : 23 Sep, 2021

In this article, we will see the robot framework and we will cover standard libraries and underlying functions, and going forward we will also cover test cases, keywords, variables, and finally organizing the test cases.

Robot Framework is a common open-source automation framework for Acceptance Testing, Acceptance Test-Driven Development (ATTD), and Robotic Process Automation (RPA). It uses a keyword-driven testing technology approach and the capabilities can be extended by testing libraries that can be implemented in Python or Java.

Let's talk about the **Acceptance Technique** in which the capability of the system is tested and the purpose of the Acceptance Technique is to evaluate the capabilities of the system in accordance with the business requirement.

Acceptance Test-Driven Development (ATDD)

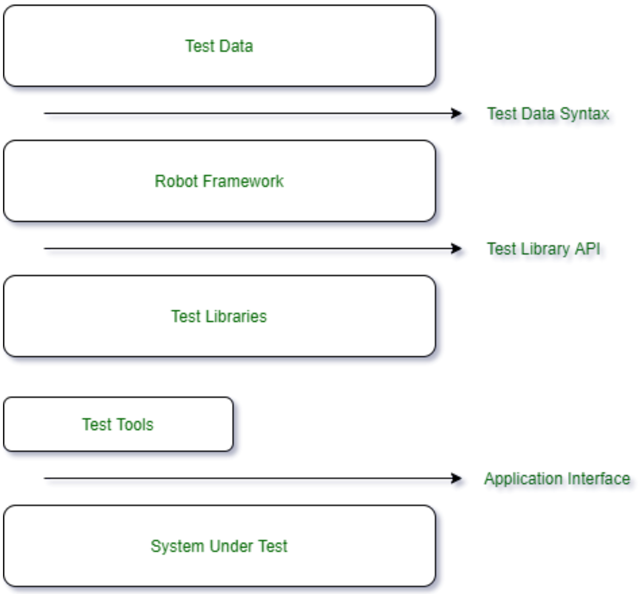
This is an advanced method rooted in business communication between customers, developers, and testers. They work in coordination and do acceptance testing before implementing the functionality.

Robotic Process Automation (RPA)

It is a process of conveniently reduces human efforts in resonance with software that uses machine learning and artificial intelligence capabilities. RPA handles high-level repeatable tasks.

The architecture of the robot framework

The test data is simple to edit the table format when the robot framework is started. It executes the process data and generates logs and reports. The core framework does not know anything about the target under test and the intersection with it is handled be bundled and expanded with self-declared libraries. When execution begins, the framework parses the data before it. It then uses the keywords provided by the libraries to interact with the destination system. Libraries can intercommunicate orderly with the system or use other tools as drivers. Test or task execution starts from the command line. As an outcome, we get a report and log in HTML format as well as an XML output.



Installation

Step 1: Python Installation

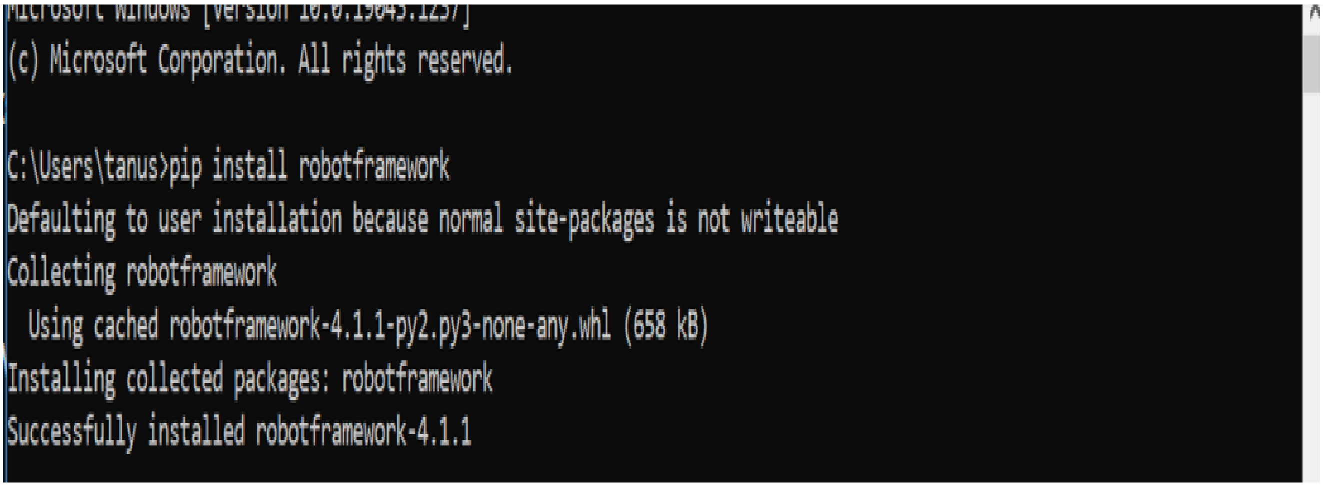
Check if Python is installed on your system.

```
python --version
pip --version
```

Step 2: Install Robot Framework

The suggested route to install the robot framework on Python is to use pip. We can use the undermentioned command to install the framework.

```
pip install robotframework
```

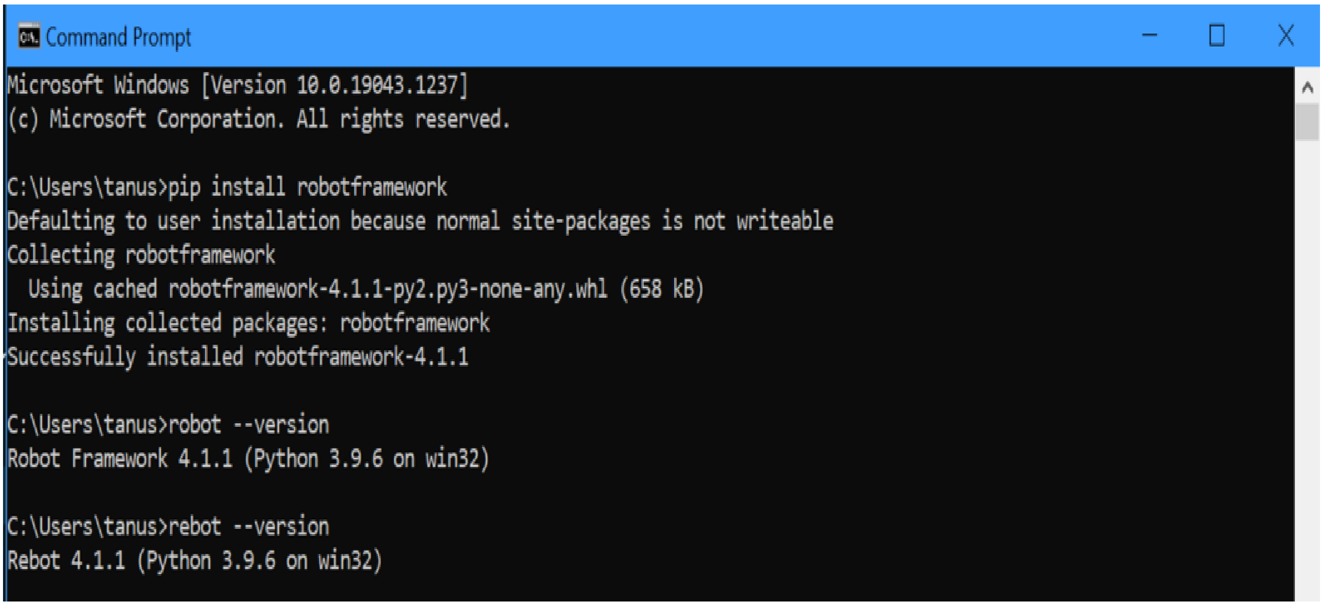


```
Microsoft Windows [Version 10.0.19043.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\tanus>pip install robotframework
Defaulting to user installation because normal site-packages is not writeable
Collecting robotframework
  Using cached robotframework-4.1.1-py2.py3-none-any.whl (658 kB)
Installing collected packages: robotframework
Successfully installed robotframework-4.1.1
```

Step 3: Verifying Installation

After the well-turned installation, we should be able to see both interpreter and robot framework versions using the –version option.

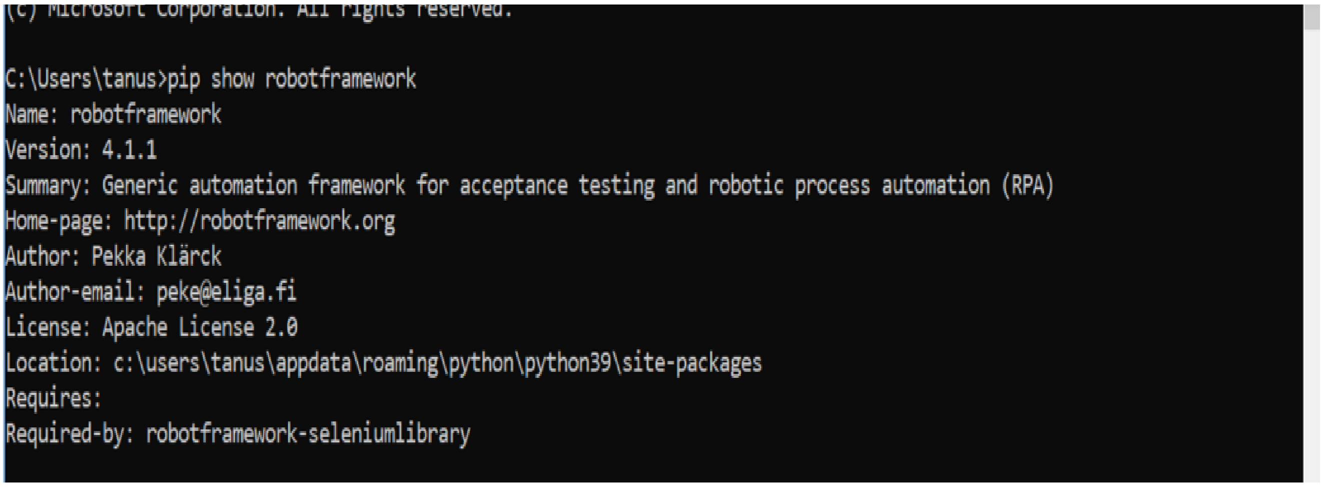
```
robot --version
rebot --version
```



```
Microsoft Windows [Version 10.0.19043.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\tanus>pip install robotframework
Defaulting to user installation because normal site-packages is not writeable
Collecting robotframework
  Using cached robotframework-4.1.1-py2.py3-none-any.whl (658 kB)
Installing collected packages: robotframework
Successfully installed robotframework-4.1.1

C:\Users\tanus>robot --version
Robot Framework 4.1.1 (Python 3.9.6 on win32)

C:\Users\tanus>rebot --version
Rebot 4.1.1 (Python 3.9.6 on win32)
```

Step 4: Check robot framework is installed properly

```
pip show robotframework
```



```
(c) Microsoft Corporation. All rights reserved.

C:\Users\tanus>pip show robotframework
Name: robotframework
Version: 4.1.1
Summary: Generic automation framework for acceptance testing and robotic process automation (RPA)
Home-page: http://robotframework.org
Author: Pekka Klärck
Author-email: peke@eliga.fi
License: Apache License 2.0
Location: c:\users\tanus\appdata\roaming\python\python39\site-packages
Requires:
Required-by: robotframework-seleniumlibrary
```

Standard Libraries in robot framework

There are a bunch of standard libraries in the robot framework. Let's discuss one by one

- **Builtin** is a Robot Framework standard library that provides a set of common keywords that are needed very often so are automatically imported and always available.
- **Collection** is a standard library that provides a set of keywords to handle python lists and dictionaries. The library has keywords to modify and derive values from lists and dictionaries.
- **DateTime** is a robot framework that supports the creation and transformation of date and time values.
- **Dialogs** is a Robot Framework standard library that provides a means to pause execution and receive input from users. The dialogs are a bit inconsistent depending on whether the tests are running or not.
- **OperatingSystem** enables the various operating system-related tasks which are going to perform in the system when the robot framework is running. It can among other things executing commands, can also create and remove files directly as well.
- **Process** is used for running processes, this library utilizes python submodules and its open class. The main use of the library is to wait for the running process in the system and compile using the run process keyword. we have start process on the background using start process and then we start the process or end process or terminate all process using wait for the process.
- **Screenshot** is a test library for taking screenshots on the machine where the tests are being run.

it possible to connect to a telnet server and execute commands over an open connection.
- **XML** is also a robot framework text library for verifying and modifying XML documents.

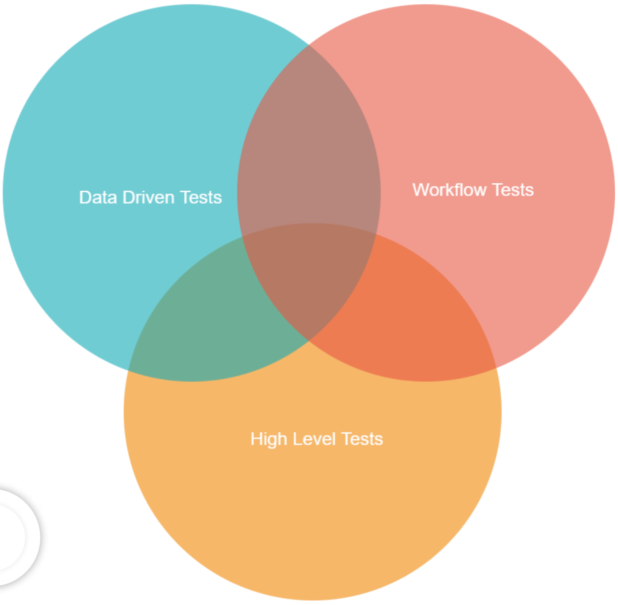
Built-in Tools

There are mainly four built-in tools that are used in robot framework that are:

- **Testdoc** – Robot Framework generates high-level HTML documentation based on test cases.
- **Rebot** – Tools to generate logs and reports based on XML output and merge multiple outputs simultaneously.
- **Tidy** – Robot Framework Tools for cleaning and changing the format of data files.
- **Libdoc** – Tools for generating keyword documentation for libraries and resource files.

Test-Cases

The test cases in the robot framework are allowed to write in simple English language rather than automated test cases. It follows a Keyword Test driven approach that resonates with natural language in terms of action rather than expectations. Test cases are created from available keywords in test case table and keywords can be imported from test library or resource file or we can also create a keywords table for test case file.



The first column in a test case table has the test names and the test case starts with something in the column and is continuous with the next test case name or the end of the table.

Higher-level Tests

The second column usually contains a keyword name and the exception to this rule is to set variables with keyword written values.

Data-Driven Tests

Data-driven testing allows test data to be isolated without mimicking workflows. The [template] setting in the robot framework sets the test case for a data-driven test.

Keywords

Test cases in Robot Framework are created with keywords coming from two sources.

- **Library Keywords:** All the lowest level keywords are defined in the standard library which can be implemented using programming languages like Java, Python, etc. Robot Framework comes with test libraries that can be divided into standard and custom libraries, so the standard libraries are in the main framework such as built-in screenshots and external libraries are installed separately like the Selenium library.
- **User Keywords:** One of the powerful features of robot framework is that we can create custom high-level keywords using other keywords.

Organizing Test Cases

Robot test cases are created in test case files, but we can organize them in directories that make up the test suite hierarchy. A collection of test cases is called a test suite. Each file containing test cases also forms a test suite. It is possible to organize test cases into a hierarchy by using directories, all of which form a higher-level test suite that derives their names from directory names.

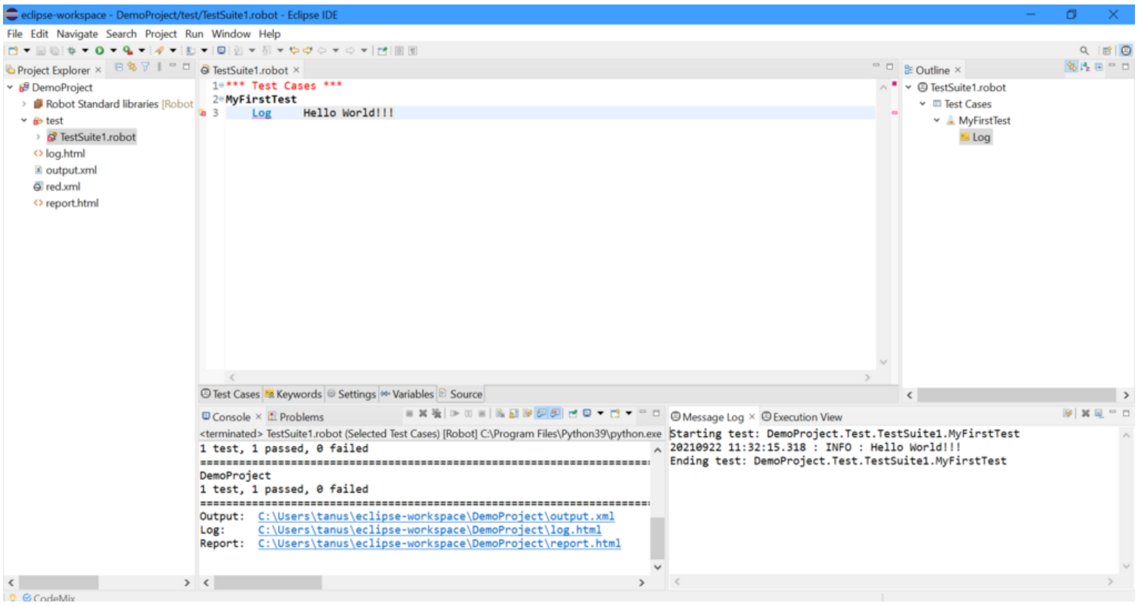
Creating First Robot Test

Step 1: Check robot framework is installed

Step 2: Go to the project Eclipse > Robot ct >New Test Suite

A simple test of the robot:

```
*** Test Cases ***
MyFirstTest
    Log     Hello World!!!
```

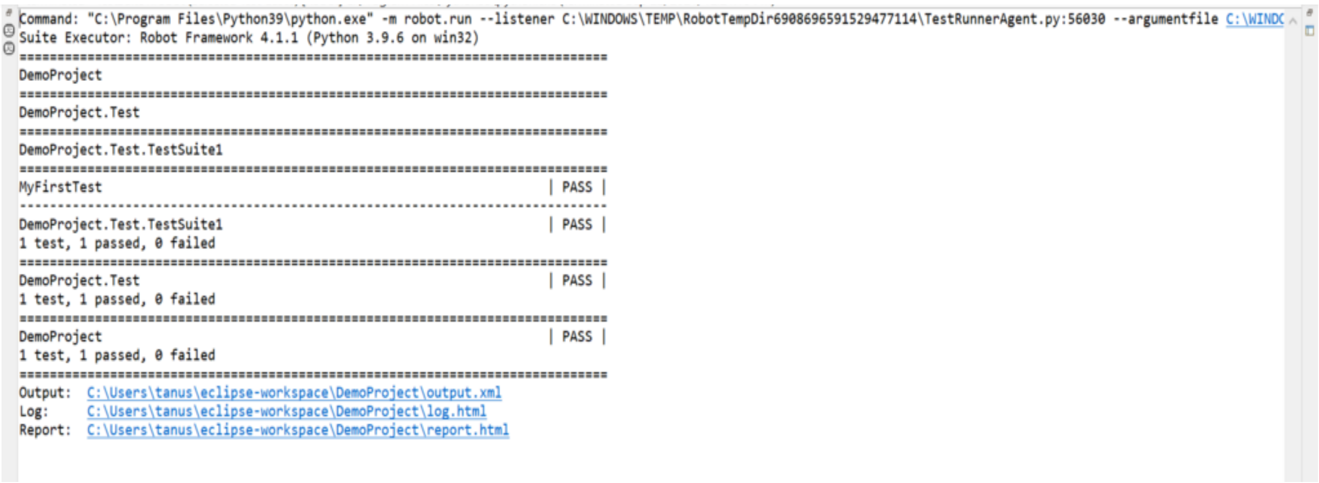


```
*** Test Cases ***
MyFirstTest
    Log     Hello World!!!
```

```
<terminated> TestSuite1.robot (Selected Test Cases) [Robot] C:\Program Files\Python39\python.exe
1 test, 1 passed, 0 failed
==============================================================================
DemoProject
1 test, 1 passed, 0 failed
==============================================================================
Output:  C:\Users\tanus\eclipse-workspace\DemoProject\output.xml
Log:     C:\Users\tanus\eclipse-workspace\DemoProject\log.html
Report:  C:\Users\tanus\eclipse-workspace\DemoProject\report.html
```

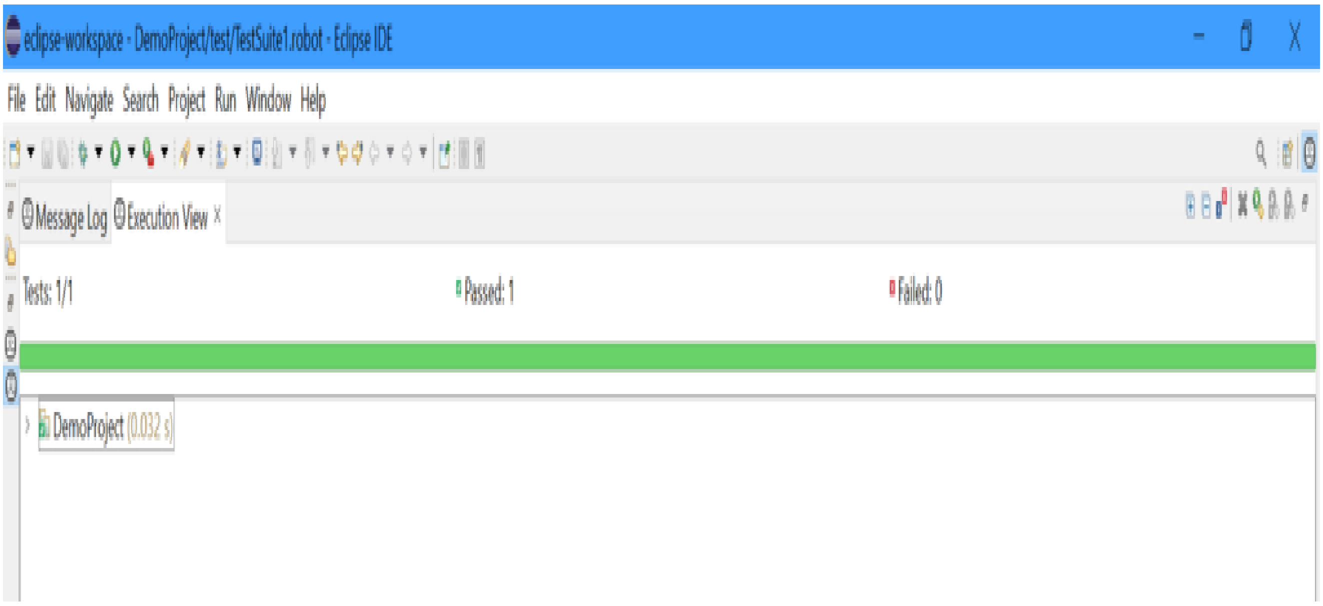
```
Starting test: DemoProject.Test.TestSuite1.MyFirstTest
20210922 11:32:15.318 : INFO : Hello World!!!
Ending test: DemoProject.Test.TestSuite1.MyFirstTest
```

To run our Test Case, simply right click on the screen and select **Run tests: 'MyFirstTest'**, and the console will open



```
Command: "C:\Program Files\Python39\python.exe" -m robot.run --listener C:\WINDOWS\TEMP\RobotTempDir6908696591529477114\TestRunnerAgent.py:56030 --argumentfile C:\WINDC
Suite Executor: Robot Framework 4.1.1 (Python 3.9.6 on win32)
==============================================================================
DemoProject
==============================================================================
DemoProject.Test
==============================================================================
DemoProject.Test.TestSuite1
==============================================================================
MyFirstTest                                                           | PASS |
------------------------------------------------------------------------------
DemoProject.Test.TestSuite1                                           | PASS |
1 test, 1 passed, 0 failed
==============================================================================
DemoProject.Test                                                      | PASS |
1 test, 1 passed, 0 failed
==============================================================================
DemoProject                                                           | PASS |
1 test, 1 passed, 0 failed
==============================================================================
Output:  C:\Users\tanus\eclipse-workspace\DemoProject\output.xml
Log:     C:\Users\tanus\eclipse-workspace\DemoProject\log.html
Report:  C:\Users\tanus\eclipse-workspace\DemoProject\report.html
```

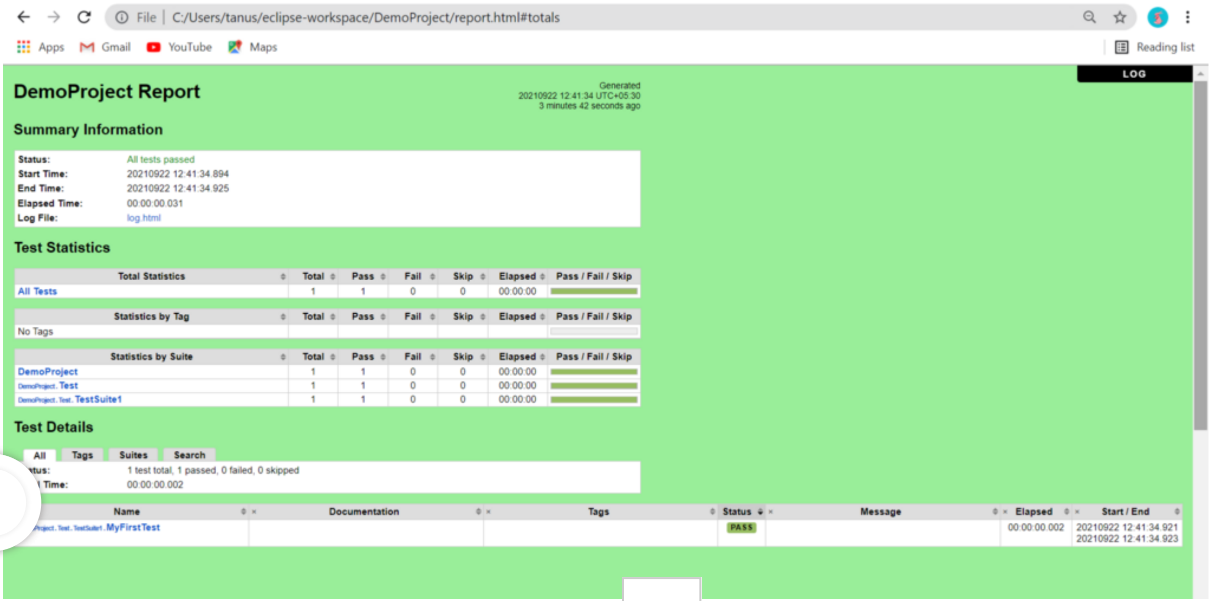
As we can see all the details about the test and we can see the output, log, and report by following the given address link



eclipse-workspace - DemoProject/test/TestSuite1.robot - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Message Log
Execution View

Tests: 1/1 Passed: 1 Failed: 0

DemoProject (0.032 s)



File | C:/Users/tanus/eclipse-workspace/DemoProject/report.html#totals
Apps
Gmail
YouTube
Maps
Reading list
LOG

DemoProject Report

Generated
20210922 12:41:34 UTC+05:30
3 minutes 42 seconds ago

Summary Information

Status:	All tests passed
Start Time:	20210922 12:41:34.894
End Time:	20210922 12:41:34.925
Elapsed Time:	00:00:00.031
Log File:	log.html

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	1	1	0	0	00:00:00	🟩

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
No Tags						

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
DemoProject	1	1	0	0	00:00:00	🟩
DemoProject . Test	1	1	0	0	00:00:00	🟩
DemoProject . Test . TestSuite1	1	1	0	0	00:00:00	🟩

Test Details

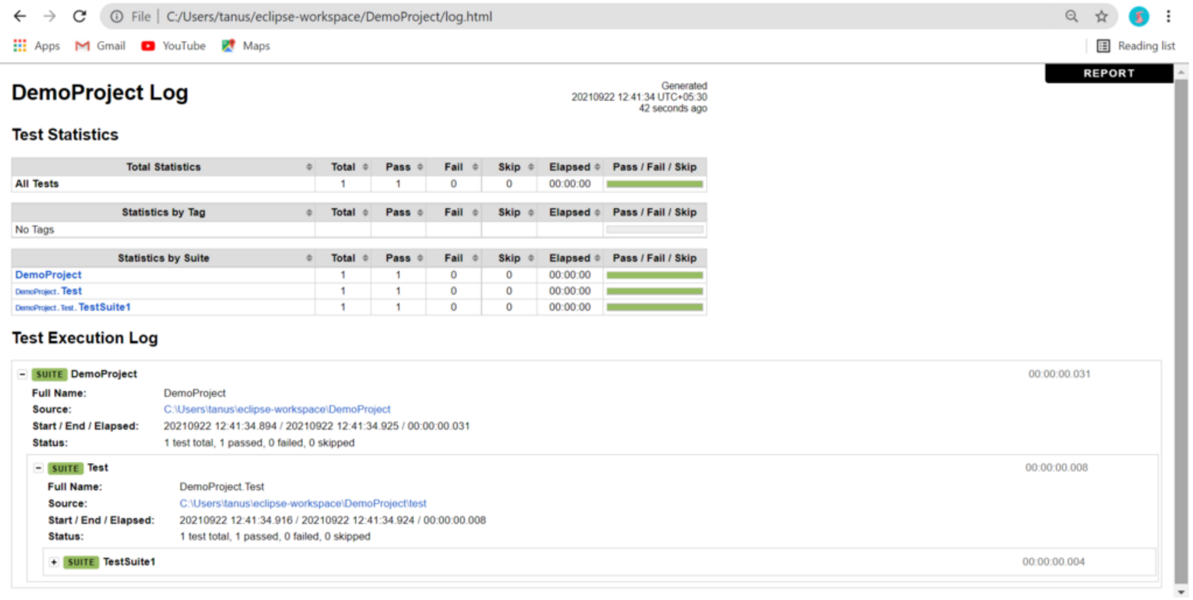
All Tags Suites Search

tus:	1 test total, 1 passed, 0 failed, 0 skipped
Time:	00:00:00.002

Name	Documentation	Tags	Status	Message	Elapsed	Start / End
Project . Test . TestSuite1 . MyFirstTest			PASS		00:00:00.002	20210922 12:41:34.921 20210922 12:41:34.923

Elapsed Time, and Log File.

And we visit the Log File by clicking on the link given:


File | C:/Users/tanus/eclipse-workspace/DemoProject/log.html
Apps
Gmail
YouTube
Maps
Reading list
REPORT

DemoProject Log

Generated
20210922 12:41:34 UTC+05:30
42 seconds ago

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	1	1	0	0	00:00:00	🟩

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
No Tags						

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
DemoProject	1	1	0	0	00:00:00	🟩
DemoProject . Test	1	1	0	0	00:00:00	🟩
DemoProject . Test . TestSuite1	1	1	0	0	00:00:00	🟩

Test Execution Log

```
− SUITE DemoProject                                                        00:00:00.031
  Full Name:           DemoProject
  Source:              C:\Users\tanus\eclipse-workspace\DemoProject
  Start / End / Elapsed:  20210922 12:41:34.894 / 20210922 12:41:34.925 / 00:00:00.031
  Status:              1 test total, 1 passed, 0 failed, 0 skipped

  − SUITE Test                                                             00:00:00.008
    Full Name:           DemoProject.Test
    Source:              C:\Users\tanus\eclipse-workspace\DemoProject\test
    Start / End / Elapsed:  20210922 12:41:34.916 / 20210922 12:41:34.924 / 00:00:00.008
    Status:              1 test total, 1 passed, 0 failed, 0 skipped

    + SUITE TestSuite1                                                     00:00:00.004
```

Here, we can see the statistics and also check the pass/fail/skip number of test cases. We can check the test execution log and test suite status.

Advantages of Robot Framework

1. It is convenient to use and install.
2. It is an open-source framework, anyone with or without programming experience can drive.
3. It is suited to various operating systems and external libraries such as the Selenium library.
4. The Robot framework supports behavior-driven, keyword-driven, and data-driven styles of building test cases.
5. We can use any Robot Integrated Development Environment (RIDE) or any text editor to write the test cases.

Disadvantages of Robot Framework

1. Scarcity of parallel test execution.
2. Customizing HTML reports is not straightforward.

Limitations of Robot Framework

Conclusion

The Robot Framework is an open-source test automation framework for acceptance testing and acceptance test-driven development. The test cases in the robot framework are based on keywords written in a tabular format, which makes it clear and readable, and reveals true information about the intent of the test case. For example, to open a browser, the keyword used is "open browser".

Like 220

Previous Next

RECOMMENDED ARTICLES Page : 1 2 3

01 Introduction to Kivy ; A Cross-platform Python Framework
26, Feb 18

02 Python | Sessions framework using django
18, Jun 19

03 Kivy - Python Framework for Mobile App development

05 Creating first web application using Bottle Framework - Python
03, Jan 21

06 Wikipedia search app using Flask Framework - Python
03, Jan 21

07 Sending Email using FastAPI Framework in Python

04 Joke App using Bottle Framework - Python
15, Feb 21

08 Introduction to Bottle Web Framework - Python
16, Mar 21

Article Contributed By :

amitverma2d
@amitverma2d

Vote for difficulty

Current difficulty : Medium

Easy Normal Medium Hard Expert

Article Tags : Blogathon-2021, Python Framework, Blogathon, Python

Improve Article Report Issue

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

GeeksforGeeks
A-143, 9th Floor, Sovereign Corporate Tower,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Company

About Us

Careers

In Media

Contact Us

Privacy Policy

Copyright Policy

Learn

Algorithms

Data Structures

SDE Cheat Sheet

Machine learning

CS Subjects

Video Tutorials

Courses

News

Top News

Technology

Work & Career

Business

Finance

Lifestyle

Knowledge

Languages

Python

Java

CPP

Golang

C#

SQL

Kotlin

Web Development

Web Tutorials

Django Tutorial

HTML

JavaScript

Bootstrap

ReactJS

NodeJS

Contribute

Write an Article

Improve an Article

Pick Topics to Write

Write Interview Experience

Internships

Video Internship

▲