Robot-Framework as Test Automation tool

Uday Kumar Reddy, Kaipa
Department of Computer Science and Automation
Technische Universitat Ilmenau, Germany
uday-kumar-reddy.kaipa@tu-ilmenau.de

***Abstract*—Testing is essential for producing high-quality software, with Test Automation being the most widely employed method for quality control. In addition to highlight the growing trend of test automation, this paper goes into great depth on the advantages of robot-framework, open-source general purpose automation framework.**

***Index Terms*—Robot-Framework, Test Automation, Automation Testing, Robotic Process Automation.**

I. Introduction

Test automation [1] refers to the automated verification of software, which includes developing code to verify flows to test the software. Thus, it can be considered software to test software. Automation of testing will be particularly helpful when complex software is updated frequently. Test automation of scenarios is not efficient in cases where there are large data dependencies that require manual efforts or interconnection with other systems that can not be easily automated. It is also critical to select the appropriate framework and technology based on AUT [3] in terms of cost, reusability, and component portability. The advantages of test automation are obvious; it reduces the resources (manual effort and time) required for testing. As a result, money is saved and testing time is reduced, allowing more features to be available and keeping up with the competition.

II. Literature Survey

Robot-framework is an open-source, modular, highly reusable, general automation framework that is suitable for RPA [Robotic Process Automation] [2] and test automation. Recent advancements in Test Automation include using the principle of RPA [2], [4] by automating the system applications, thereby simulating user testing [5]. The relationship between RPA (Robotic Process Automation) and Test Automation is that RPA is used to automate repetitive, rule-based tasks in business processes, while Test Automation is used to automate the testing of software applications to ensure that they are functioning properly. Both RPA and Test Automation aim to improve efficiency and reduce the amount of manual work required to perform certain tasks. Although there are many similarities in the implementation of Test Automation and RPA, RPA is primarily used in Automation of business workflows [2]. One of the main reasons for using test automation in acceptance testing is to improve the efficiency and effectiveness of the testing process. Test automation allows for the validation of a large number of tests with very less manual efforts in a short amount of time, which can be useful for ensuring that the system is thoroughly tested. According to the authors [2] the trend of using RPA in Test Automation increased in recent times, which can be attributed to the ease of setup and features offered by the tool. Several RPA tools are used in Test Automation, QTP/UFT, Robot-framework, UiPath, WorkFusion etc., are some.

III. Methodology

Robot-framework is available as *Python* library and can be installed via PiP, the Python package manager. Robot-framework is a keyword-driven, open-source framework that emphasizes reuse. The source files are extremely readable. It uses a keyword-driven architecture with plain-text-like syntax that non-specialists in the organization may easily understand. Test Automation teams typically run into issues with *regression testing*, when doing end-to-end tests that call for interacting with external systems or third-party applications. The robot-framework is capable of supporting custom libraries. The fact that it is built on Python, which has a plethora of libraries and support to make the most of current capabilities, is an added benefit.

A. Robot-Framework Libraries

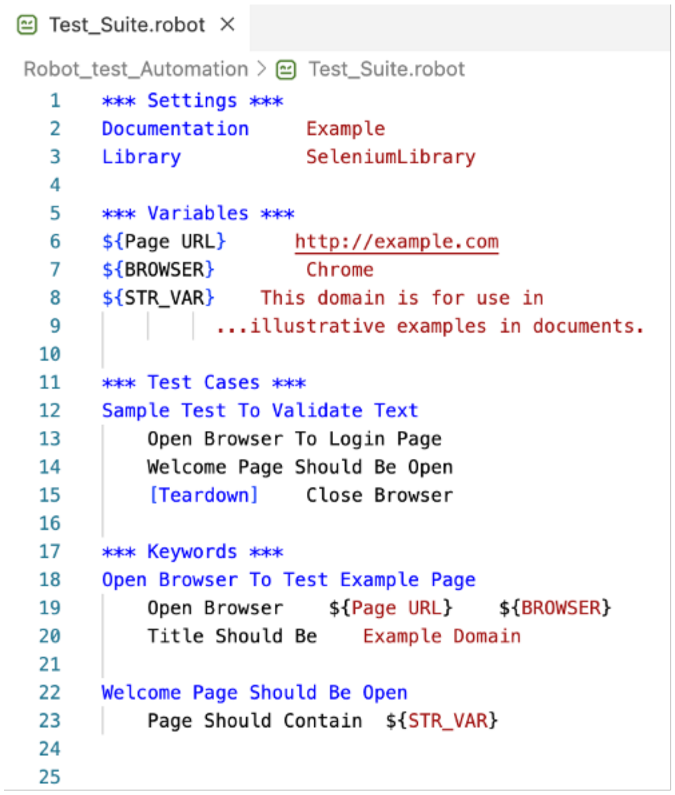
robotframework-seleniumlibrary and *robotframework-appiumlibrary* are two widely used open-source python libraries for test automation of web applications and mobile applications (iOS and Android) respectively. These libraries have all of the necessary keywords to perform the basic interactions built on top of *selenium* and *appium* respectively. In this paper *Selenium* refers to selenium webdriver. *Appium* and *Selenium* are popular open-source automation tools, which contains the implementation to interact with the AUT.

B. Test Implementation

A straightforward web test case is used in the example below to validate a text on a web-page using norms for variables and keywords as provided by the framework.

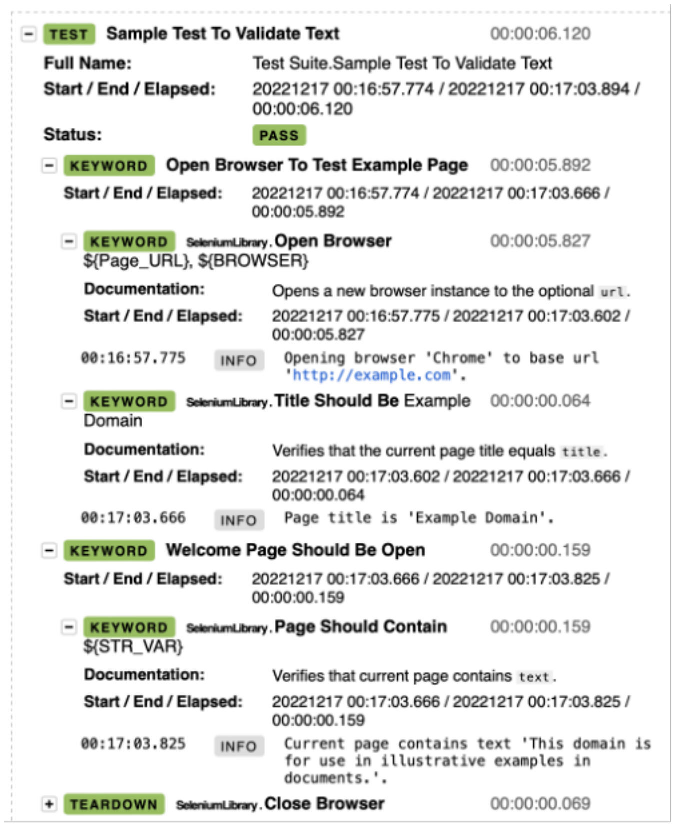
With modest library (appium) and element(Xpath) changes, the simple, keyword-driven tests may be used to test both native mobile apps and online apps. Reusing test suites and higher level keywords can save a substantial amount of time.

Custom or existing python libraries can be easily imported in the *Settings* section of script and their implementation can used in the *Test Cases* [2], [3]. Test data can be changed easily from the variables section or it is also possible to script custom library to read data from data file in Data-driven testing [3].



```
Test_Suite.robot ×
Robot_test_Automation > Test_Suite.robot
*** Settings ***
Documentation       Example
Library             SeleniumLibrary

*** Variables ***
${Page URL}        http://example.com
${BROWSER}          Chrome
${STR_VAR}      This domain is for use in
...             ...illustrative examples in documents.

*** Test Cases ***
Sample Test To Validate Text
    Open Browser To Login Page
    Welcome Page Should Be Open
    [Teardown]      Close Browser

*** Keywords ***
Open Browser To Test Example Page
    Open Browser      ${Page URL}     ${BROWSER}
    Title Should Be     Example Domain

Welcome Page Should Be Open
    Page Should Contain   ${STR_VAR}
```

Fig. 1. Robot-Framework test case with Keywords



```
⊟ TEST  Sample Test To Validate Text                                00:00:06.120
  Full Name:            Test Suite.Sample Test To Validate Text
  Start / End / Elapsed: 20221217 00:16:57.774 / 20221217 00:17:03.894 /
                        00:00:06.120
  Status:               PASS
  ⊟ KEYWORD  Open Browser To Test Example Page                      00:00:05.892
    Start / End / Elapsed: 20221217 00:16:57.774 / 20221217 00:17:03.666 /
                          00:00:05.892
    ⊟ KEYWORD  SeleniumLibrary.Open Browser                         00:00:05.827
      ${Page_URL}, ${BROWSER}
      Documentation:        Opens a new browser instance to the optional url.
      Start / End / Elapsed: 20221217 00:16:57.775 / 20221217 00:17:03.602 /
                            00:00:05.827
      00:16:57.775   INFO   Opening browser 'Chrome' to base url
                            'http://example.com'.
    ⊟ KEYWORD  SeleniumLibrary.Title Should Be Example              00:00:00.064
      Domain
      Documentation:        Verifies that the current page title equals title.
      Start / End / Elapsed: 20221217 00:17:03.602 / 20221217 00:17:03.666 /
                            00:00:00.064
      00:17:03.666   INFO   Page title is 'Example Domain'.
  ⊟ KEYWORD  Welcome Page Should Be Open                            00:00:00.159
    Start / End / Elapsed: 20221217 00:17:03.666 / 20221217 00:17:03.825 /
                          00:00:00.159
    ⊟ KEYWORD  SeleniumLibrary.Page Should Contain                  00:00:00.159
      ${STR_VAR}
      Documentation:        Verifies that current page contains text.
      Start / End / Elapsed: 20221217 00:17:03.666 / 20221217 00:17:03.825 /
                            00:00:00.159
      00:17:03.825   INFO   Current page contains text 'This domain is
                            for use in illustrative examples in
                            documents.'.
  ⊞ TEARDOWN  SeleniumLibrary.Close Browser                         00:00:00.069
```

Fig. 2. Robot-Framework generated test report.

The enormous volume of generic and custom libraries along with the framework's ease of setup and extension make it the perfect tool for test automation.

C. Test Execution

Robot framework, like many other popular test automation tools, has a feature for segregating tests by tagging [3], which can be selectively executed using special flags. As robot-framework is modular, individual components of a Test can written and executed separately, which can also be used as part of other test.

D. Reporting

By default, test execution generates reports and logs of tests that contain details of data used in the TC as well as screenshots of failure step, which can be very useful for testers for debugging and reporting issues. Reports that can be used for analysis are also included in RPA tools. It contains a variety of specific details regarding the various steps and variable values as seen in Fig. 2. When there is a problem, a screenshot of the results is automatically taken and can be used for debugging and failure causes.

IV. Challenges

- Unlike other frameworks that support multiple languages, such as Cucumber, robot-framework is supported only in Python.
- The framework is still being actively improved and changed; some keywords in older versions have been deprecated, so newer versions are not always backward compatible.

V. Conclusion

This paper evaluates the various approaches to software test automation and suggests a new open-source solution based on RPA usage, which is crucial in an era where apps are created for several platforms.

References

[1] Y. Labiche, "Test Automation - Automation of What?," 2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), 2018, pp. 116-117, doi: 10.1109/ICSTW.2018.00037.

[2] S. Yatskiv, I. Voytyuk, N. Yatskiv, O. Kushnir, Y. Trufanova and V. Panasyuk, "Improved Method of Software Automation Testing Based on the Robotic Process Automation Technology," 2019 9th International Conference on Advanced Computer Information Technologies (ACIT), 2019, pp. 293-296, doi: 10.1109/ACITT.2019.8780038.

[3] Dustin, Elfriede, et al. Implementing Automated Software Testing: How to Save Time and Lower Costs While Raising Quality. 1. printing. Addison-Wesley, 2009.

[4] N. Falih, S. H. Supangkat and F. F. Lubis, "Robotic Process Automation in Smart System Platform: A Review," 2022 International Conference on ICT for Smart Society (ICISS), 2022, pp. 01-05, doi: 10.1109/ICISS55894.2022.9915043.

[5] S. Berner, R. Weber and R. K. Keller, "Observations and lessons learned from automated testing", Proceedings - 27th International Conference on Software Engineering ICSE05, pp. 571-579, 2005.