Robot Framework

Stage 1 – Python

Stage 2 – Robot Framework – standard libraries

Stage 3 – Robot Framework – External libraries – Selenium, API, Database, Datadriver...

Complete the Pre-test – Link in chat window

Web Automation – Selenium

* Pre-requisite (anyone – java, C#, Python, Ruby, Javascript, python- Robot Framework )

Python- Robot Framework

* Selenium (Web Automation)
* AutoIT (Windows GUI automation)
* Request Lib (API automation)
* Database Lib (Database )
* Appium (mobile automation – not included in this session)

1. Installation:-
   1. Python Install – version – 3.9.4

<https://www.python.org/downloads/release/python-394/>

Check version in cmd – python 3.9.4

* 1. PyCharm IDE – Community

<https://www.jetbrains.com/pycharm/download/download-thanks.html?platform=windows&code=PCC>

1. Architecture:-

Source code (.py) 🡪 Interpreter (use bultin lib) 🡪 O/P

Interpreter 🡪 python.exe

1. Naming convention

lowercase\_with\_underscore – my\_first\_project

UpperCamelCase – MyFirstProject

Project (lowercase\_with\_underscore)

Package (lowercase\_with\_underscore)

Python file (.py) (lowercase\_with\_underscore)

Scripts, Class (UpperCamelCase)

Methods/ variable - (lowercase\_with\_underscore)

1. Create Project
2. Create package
3. Create python file
4. Datatypes
   1. Numeric types – int, float, complex
   2. Boolean
   3. Sequence types – String, List, tuples (index starts at 0)
   4. Dictionary
5. Convert to String

str(res)

1. List vs tuples
   1. Tuples – will be faser/ immutable
2. Methods – building block of the program
   1. Reuse
   2. Maintenance

Can create method/call methods at

* Module level
* Class level
  + Static methods
  + Non-Static methods

1. User defined datatypes – Employee, Student
2. Class & Object

Robot framework – <https://robotframework.org/>

1. Installation
   1. Python – 3.9.4
   2. Pycharm
   3. Install robot framework

**pip install robotframework==4.0.3**

* 1. Verify in cmd

**pip list**

**robot --version**

* 1. Install plugin in pycharm **– robot framework language server**

1. Create project
2. Create suite folder and suite file (.robot)
3. Work on standard libraries – Builtin, String, Collection, Operating System, Process

Bulitin - <https://robotframework.org/robotframework/latest/libraries/BuiltIn.html#library-documentation-top>

Datatime - <https://robotframework.org/robotframework/latest/libraries/DateTime.html>

Operating System - <https://robotframework.org/robotframework/latest/libraries/OperatingSystem.html>

1. To trigger the robot file - **robot .\robot\_suite\demo1\_suite.robot**
2. Variable
   1. Scalar variable ($)
   2. List (@)
   3. Dictionary (&)
3. To trigger the particular testcase 🡪

<https://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html#command-line-options-for-test-execution>

**robot -t "TC4" .\robot\_suite\demo1\_suite.robot**

**robot --test "TC4" .\robot\_suite\demo1\_suite.robot**

1. Selenium WebDriver – Web automation

Architecture

**Source code (Robot framework+selenium lib) 🡪 Browser**

**Install**

**pip install --upgrade robotframework-seleniumlibrary**

1. Selenium library keyword - <https://robotframework.org/SeleniumLibrary/SeleniumLibrary.html>
2. Launch browser
3. WebDriverException: Message: 'chromedriver' executable needs to be in PATH. Please see <https://chromedriver>

TO fix the driver

* Using executable\_path argument of open browser keyword
* Add the driver path to environment variable
* Add using the driver path in robot framework

Append To Environment Variable Path D:\\Software\\

1. Inspect – tagname, attribute, text or not
2. Click, type, select
3. Basic locators
   1. Id
   2. Name
   3. Classname
   4. Tagname
   5. Linktext
   6. Partial link text

When there are duplicate locator then it will pick the first element.

1. Advance locators
   1. Xpath
   2. Css
2. Findelement/getwebelement 🡪 check for presence of element in 0.5s /500ms
3. Synchronization
   1. Unconditional wait (from buitlin)

Sleep 5s 🡪 not recommended

* 1. Conditional wait (from selenium lib)
     1. Implicit wait
        1. Default implicit wait – 0s
        2. Applicable for all findelement(getwebelement) and findelements(getwebelements) keywords
        3. Example: Implicit wait – 30s
           1. If element is not present then it will check for 30s and then throw error.
           2. If element is present then it will do the operation immediately.
           3. Polling time – 0.5s (how frequently it checks)
     2. Explicit wait
        1. Exact condition
        2. Polling time – 0.5s

1. Dropdown:-

Reference:-

|  |  |  |
| --- | --- | --- |
| **Strategy** | **Match based on** | **Example** |
| id | Element id. | id:example |
| name | name attribute. | name:example |
| identifier | Either id or name. | identifier:example |
| class | Element class. | class:example |
| tag | Tag name. | tag:div |
| xpath | XPath expression. | xpath://div[@id="example"] |
| css | CSS selector. | css:div#example |
| dom | DOM expression. | dom:document.images[5] |
| link | Exact text a link has. | link:The example |
| partial link | Partial link text. | partial link:he ex |
| sizzle | Sizzle selector deprecated. | sizzle:div.example |
| data | Element data-\* attribute | data:id:my\_id |
| jquery | jQuery expression. | jquery:div.example |
| default | Keyword specific default behavior. | default:example |

Reference:-

Cick button