User Guide

for

Vanilla Python Framework

Afour Technologies

209/1B/1A, 1st Floor, Range Hills Road, Opposite Symphony Hotel,

Shivaji Nagar, Pune, Maharashtra 411020

INDEX

1) Install Python

First, download the latest version of Python 2.7 from the official Website.

Add the directories for your default Python version to the PATH. Assuming that your Python installation is in C:\Python27\, add this to your Path :

```
C:\Python27\;C:\Python27\Scripts\
```

2) Install an IDE

Install an IDE for writing scripts efficiently.

For installing PyCharm refer link : https://www.jetbrains.com/pycharm/

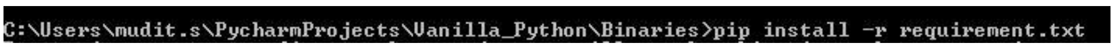
For downloading and setting up Eclipse for Python refer link :

https://eclipse.org/downloads/ and

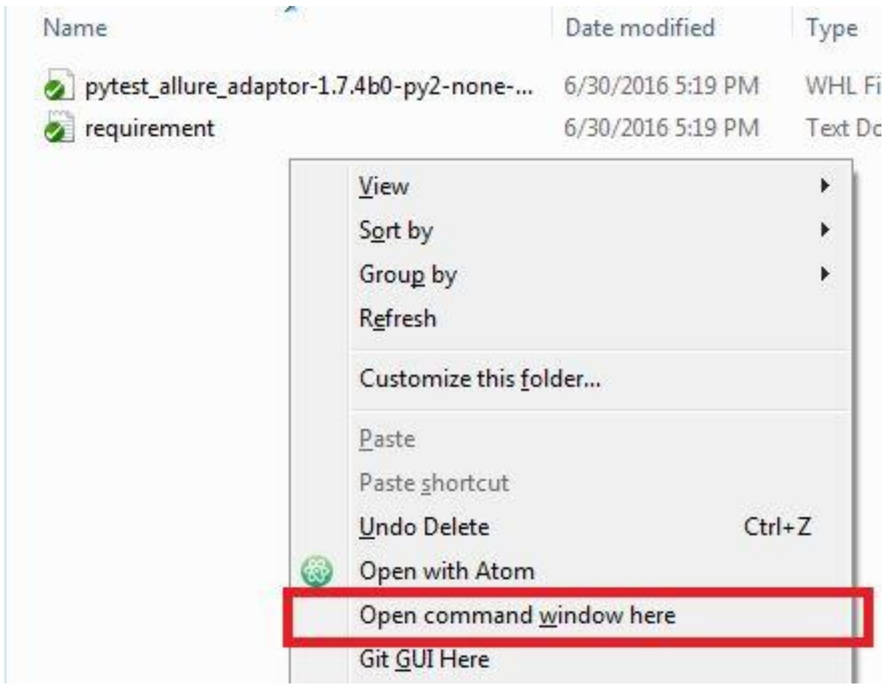
https://seleniumpythontutorial.wordpress.com/how-to-configure-eclipse-for-python/

3) Install required dependencies of the framework

Navigate to Binaries folder in project structure and open command prompt by pressing shift+right click and select Open command window here or open command prompt and navigate to Binaries folder. In command prompt enter "**pip install -r requirement.txt**". It will install all the required binaries/dependencies.



```
C:\Users\mudit.s\PycharmProjects\Vanilla_Python\Binaries>pip install -r requirement.txt
```



4) Configuration File Details

[Application]

URL=http://afourtech.com/ -> This URL parameter is used when user does not pass URL argument to OpenURL method of Base class.

[Timeout]

MinTimeOut=10

MidTimeOut=20

MaxTimeOut=30

TimeoutType=Mid -> This parameter is used for implicit wait settings.

[Environment]

BrowserType=Chrome -> This parameter is used to select Browser type for automation.

[Execution]

FullSuite = False -> If this parameter is set to True all test methods in TestScript folder will be executed else the test scripts name mentioned in Batch file will be executed.

[OperatingSystem]

OS = Windows -> Write the OS name in which you are working.

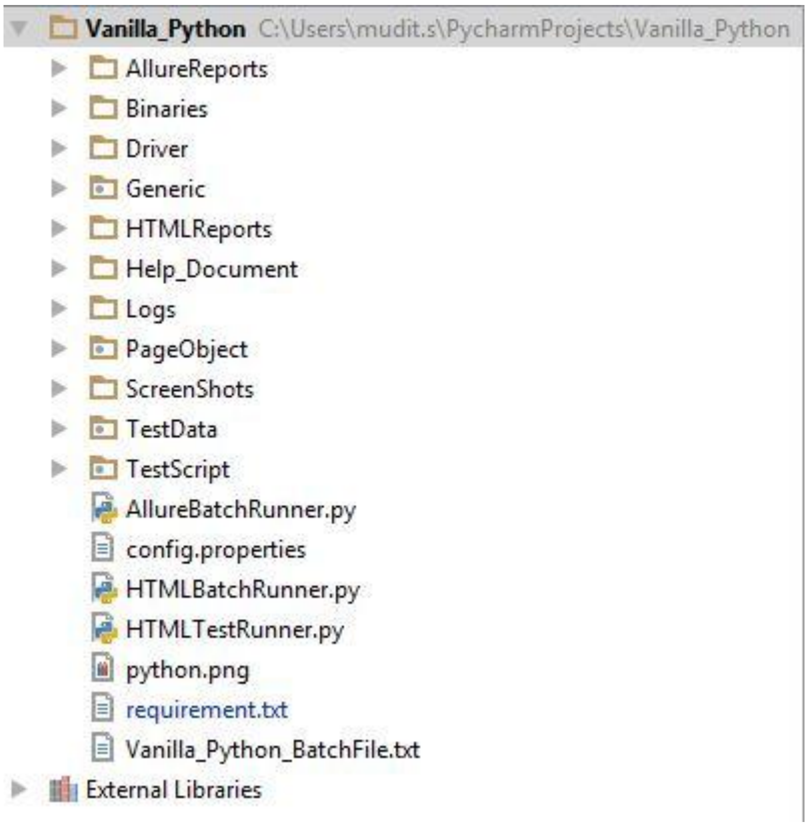
[BatchFileName]

FileName=Vanilla_Python_BatchFile.txt -> Mention the Batch file name which you want to execute.

[LoggingLevel]

levelName = info -> Mention the logging level debug,info,warning,error or critical as per requirement.

5) Framework Structure



```
Vanilla_Python C:\Users\mudit.s\PycharmProjects\Vanilla_Python
    AllureReports
    Binaries
    Driver
    Generic
    HTMLReports
    Help_Document
    Logs
    PageObject
    ScreenShots
    TestData
    TestScript
    AllureBatchRunner.py
    config.properties
    HTMLBatchRunner.py
    HTMLTestRunner.py
    python.png
    requirement.txt
    Vanilla_Python_BatchFile.txt
External Libraries
```

Brief Explanation of all folders in the Framework:

Allure Reports : This folder contains XML type report and screen-shots for generating Allure reports.

Driver : This folder contains browser drivers like Chromedriver.exe and IEDriverServer.exe. New browser drivers can be added as per requirement.

Generic : This Python package contains generic classes having generic methods which will be reused throughout automation. The brief description of the Classes and their respective methods :-

I) Base

a) **openURL** : This method sets up the driver according the settings in configuration properties and opens the application url.

II) CSVDriver

a) **get_csv_data :** This method reads the csv type test data file and return the all the rows of the file as a list.

III) ReadConfig

a) **setConfigValues :** This method reads the config.properties file and set the respective variable values.

IV) SkipIfTrue

a) **skipIfTrue :** This method is called when skipIfTrue decorator is used above any test method. This method is used to skip a test method if its dependent test method has failed.

V) Wrappers

This Class contains the reusable methods :-

a) clickButton : Click the given button if it exists and displayed on the page.

b) clickLink : Click the given link if it exists and is displayed on the page.

c) enterText : Enter the given text in the input box if it exists and is displayed on the page.

d) is_visible : Wait for the object to become visible. If it becomes visible in the given time, return true else return false.

e) getCheckboxSelection : This function will return check-box selected state.

f) clickCheckbox : This function will click on checkbox in order to select or deselect.

g) objectExists : Wait for object to get rendered. If it is rendered in the given time , return True else return False.

h) is_clickable : Wait for the object to be rendered and become clickable. If it becomes clickable in the given time return true else return false.

i) isEnabled : Check if the given object is enabled.

j) textExistsOnPage : Check if the given string exists on the entire page. Return true if found. If object is provided wait till the object becomes visible and then check for the string in the page source.

k) **textExistsInControl :** Check if the given string exists in the text of the given control. Check if the control is present on the page, get it's text and then check if the text contains the given string. Return true if found.

l) **getText :** Return the text of the control if it exists. objObject parameter is the sequence, not the web element.

m) **getTextForWebElement :** Return the text of the control if it exists. objObject parameter is the web element.

n) **getValue :** Return the value of the control if it exists. objObject parameter is the sequence, not the web element.

o) **getAttribute :** Returns object attribute specified as a parameter . objObject parameter is the sequence, not the web element.

p) **getAttributeForWebElement :** Return the value of the given attribute for the element. The element is passed as web element and not as the sequence.

q) setRadioButtonByValue : Set the value of the radio button.

r) **elementsInContainer :** Return all the elements of the given type in the given container/parent. E.g. all links in a section or buttons in a div, etc.

s) **getChildElementsByTag :** Return all the child elements of the given tag for the given parent element. The parent element is passed as web-element rather than the sequence.

t) **isNotVisible :** An Expectation for checking that an element is either invisible or not present on the DOM.

u) **selectValueFromDropdown :** Select the given value from the drop-down.

v) **switchToFrame :** Switch to the given iFrame

w) **switchToDefault :** Switch back from the iFrame to default content

x) refreshPage : This function will refresh page.

y) getElements : This function will return array of elements of the same type.

z) **dragAndDrop :** This function will drag and drop an element from source location to target location.

aa) **mouseHverAndClick :** This function will mouse-hover and click on the given element.

bb) **mouseHover :** Hover the mouse over the given object. Typically used where there's a menu dropdown to be accessed which opens up after mouse hover.

cc) **moveBrowserHistory :** Move back or forward in the browser's history.

dd) **closeBrowserAlert :** Close any browser alert that is displayed. It may not be possible to check the text that is displayed in the alert.We are just pressing the Ok button and closing the alert. We are only checking that alert was displayed.

ee) **getCurrentURL :** Get the URL of the current page.

ff) **connectToMailServer :** Connect to the gmail server using given login/password and select the specified folder.

gg) **findEmailBySubject :** Find the first unread mail in the email folder for the given subject. IMPORTANT: It is assumed that the mailbox is sorted on unread mails. If there are multiple emails found for the given filter, it'll take the first one found.

hh) **getEmailContent :** Get contents of the email as string. Currently, html emails are considered. Conditions for other content type to be added as required.

ii) **getURLfromEmail :** Extract the URL from the email. It'll return the first <a> tag found in the contents of the email so be careful if there are multiple <a> tags in the email message.This method can be directly called as it'll connect to the server, find the required email, extract the contents and return the URL.

jj) **get_base64_encode_screenshot :** Method to capture screenshot.

kk) **get_project_path :** Get the prject path for the current project.

HTML Reports : This folder contains the HTML reports.

Logs : This folder contains the execution logs.

Page Object : This python package contains the page object classes.

Screen-Shots : This folder contains the screen-shots of the failures occurred while execution.

Test Data : This python package contains the CSV or JSON type test data files.

Test Script : This python package contains the Test script classes.

AllureBatchRunner.py : This file is executed when you do batch execution and want to generate Allure report for the execution.

Config.properties : This file contains all configuration properties.

HTMLBatchRunner.py : This file is executed when you do batch execution and want to generate HTML report for the execution. This file uses HTMLTestRunner.py to generate HTML reports.

HTMLTestRunner.py : This file is used by HTMLBatchRunner.py to generate HTML reports of execution.

Python.png : This is the sample image   or logo which will appear on HTML reports. Replace it with respective client logo.

6) How to add a Test script :

Step 1 : Add respective page object class in Page Object python package. The class will contain the the respective page elements locators and respective methods to perform action on those locators using the generic methods of Wrapper class.

Step 2 : Add respective CSV or JSON test data file in Test Data python package.

Step 3 : Write test script class according to the test case or test scenario using respective Test data file from Test Data python package in Test Script python package and call respective methods from respective page object classes in Page object python package.

7) How to do batch execution and generate Allure/HTML Report:

Open Vanilla_Python_BatchFile.txt and write the name of the test scripts to be executed in batch one below the other. Run AllureBatchRunner.py if you want to generate xml reports for Allure or run HTMLBatchRunner.py if you want to generate HTML reports. The reports will be generated in their respective folders.

8) Useful Links :

1) Unit Test

2) PyTest

3) Allure Reports

4) Install Python

5) Install required dependencies mentioned in requirement.txt using PIP.

6) IDE PyCharm

7) IDE Eclipse - How to Configure for Python.

8) **You can find project related videos and documents on** https://drive.google.com/a/afourtech.com/folderview?id=0B8itzcqJsp40SGdtR0NpNlA2Wk0&usp=sharing