Framework

Definition:

Framework is a well-organized structure of reusable components where one driver (.xml) file will take care of the execution without any manual intervention.

(or)

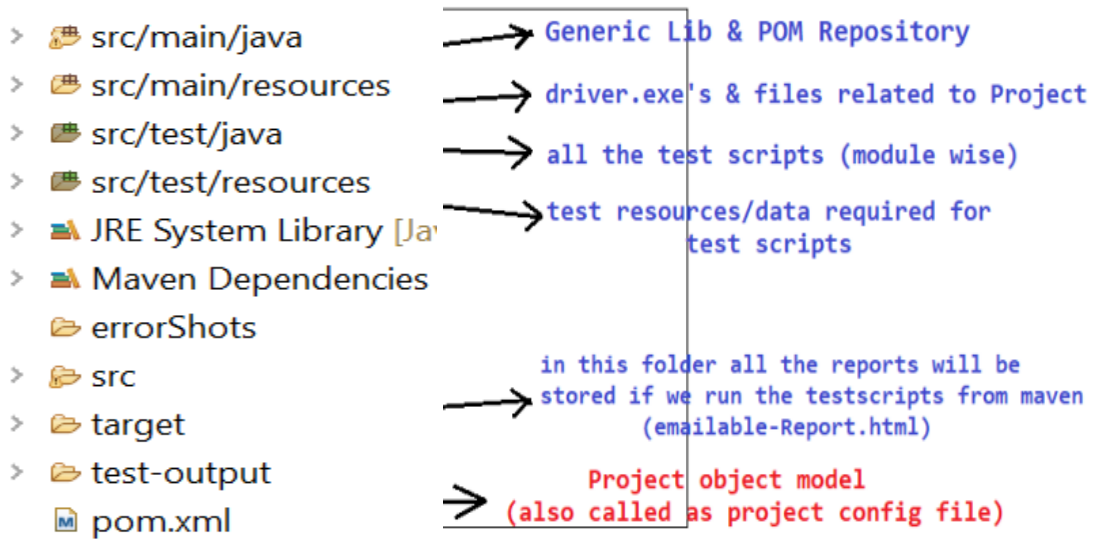
Framework is collection of reusable components that makes automation development execution, modification and maintenance is easier and faster.

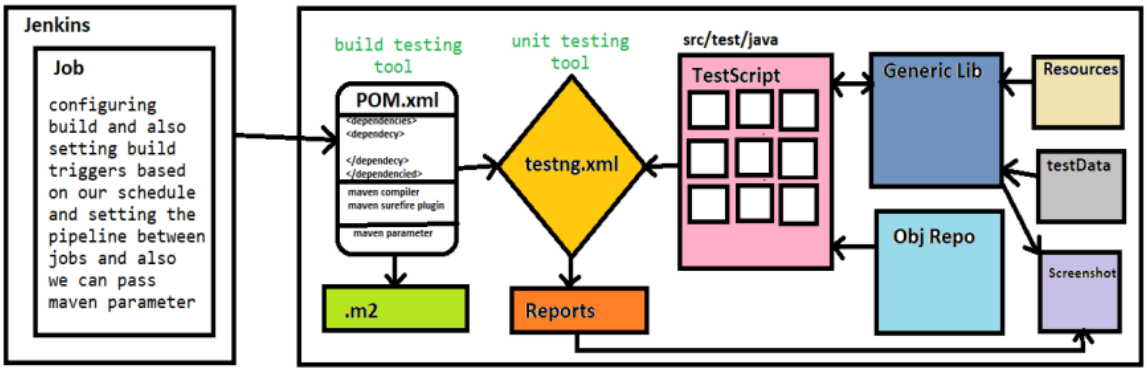
(or)

Framework is a set of instruction followed by every organization that makes automation test engineer work easy

- The Framework used in our previous project is Hybrid driven framework
- The framework was the combination of data driven, modular driven, method driven framework and TestNG components, based on advantages among the above frameworks, our framework has been customized to enhance and provide a mechanism to write test scripts in efficient, organized manner

Project Structure



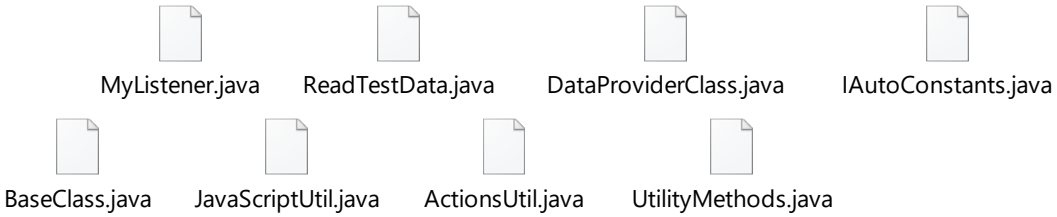


Components of framework:

1. generic_utilities
2. element_repository
3. resources ->main/resources
4. test resources ->test/resources
5. test scripts -> test/java
6. reports
7. errorshots
8. pom.xml
9. .m2

1. generic_utilities

the first component of our framework which contains utility classes which provides code reusability which contains some important components(classes) like



a. baseclass:

it is a class contains basic testNg config annotations which is needed for all the test scripts so,as per automation standards for every test script we will extend base class where our config annotations will be used by each and every scripts

In our project we have used mainly @BeforeClass →BrowserSetUp, @AfterClass -> Browser TearDown, @BeforeMethod -> Login @AfterMethod -> Logout

b. ReadTestData:

it is a util class which contains some methods which are used to retrieve and return the data which are present in the excel file and properties file

as per automation standards data should not be hardcoded in the scripts and this particular util class plays a major role in interating and retrieve the data need for data driven testing

c. UtilityMethods:

This util class conatins many important methods to switch to a particular component and handle it, method to captureScreenshot, method to generate random number , which helps us in reusability

d. DataProviderClass:

This util class has @DataProvider methods which are responsible for retrieving data from excel sheet row by row and execute same script with different set of data for multiple times

e. MyListener:

This util class has implementation for the ITestListener so we can track our testscripts assign what work has to be done in a respective stage of test script execution

Ex: if test script fail , ITestListener should cpature screenshot and store in error shots.

f. IAutoConstants:

This is an Interface we we store all the constants needed for the framework that is all the test data file paths, storing the timeouts , so that we can use these constants through out the project.

g. ActionUtils:

This util class contains customized the methods related all the major actions need to be used in our project by using actions class methods so it will easy to automate.

h. JavaScriptUtils:

This util class contains customized the methods related all the major javascript code operations need to be used in our project so it will easy to automate.

2. element_repository


Base_Page.java


Login_Page.java


Register_Page.java

it is collection of Reusable webElement and getter methods of each and every page of the given module or project in the form of POM design pattern, by using findBy, findBys and findAll annotations for reusability of the locators and to avoid staleness of the stored

ex: Base_Page, Login_Page, Register_Page etc..

3. resources:

it is important content in our framework where we will have folders to store :

- ➔ Driver.exe
- ➔ project related documents userguide documents, guidelines etc..
- ➔ project related knowledge transfer videos.

4. test resources:

it is an important content in our framework which contains all the data required for testing the application and run the framework in effient manner in a folder called src/test/resourcescommonly we store our test data in excel and property files as per automation standard we will store all the test data in files like excel because data should be externalized because modification and maintainance is easier.

5. test scripts:

a major content in our framework which contains collection of TestNG classes(testscripts) which contains @Test annotated methods(test case) in a folder called src/test/javawhile developing test scripts we have to make sure that we are using generic_utilities and element_repositories whenever testcases are given to automate them we should create packages for each and every module anddevelop the scripts, here normally we will extend base class which contains all before and after config annotations and develop scripts

6. Reports:

these are the files which is going to analyzed by our senior authorities, by reports we can find our loop holes i.e where the mistake has been occured in our scripts

There will 2 different report folder based on our execution, if we execute our scripts using testNg it will be present in test-output folder, if we have used maven to execute our scripts we will find our report in target/surefire-reports

Report plays a major role in enhancing our testScripts based on report generated and it acts a proof document

7. Errorshots:

it is one of the important component in the framework, when ever a script is getting failed during the execution of the script, we will capture the screenshot in errorshot folder, this screenshot can be used for tracing and raising defect in JIRA, they act as proof for failing of the scripts, these screenshot will be attached to our defect report .

8. pom.xml:

pom.xml stands for project object model which is responsible to execute all the testscripts which are present in src/test/java which are suffixed with Test if we run pom.xml file

this pom.xml is responsible to add dependencies in maven , it gives a flexiblity to manage dependencies, it also supports maven profiling where we will decide which xml suite to be executed at the runtime we can decide in cmd prompt using mvn test -P command we can add 'n' number of dependencies and profile using profiles and suiteXmlFile tag.

9 .m2:

it is a folder create to store the downloaded dependency files and folders in our system as local repository when dependencies are added in pom.xml file, this folder will be create at first time when we add dependency in pom.xml at first time, later other dependencies which is been added in pom.xml, control will check whether that dependency is present in this .m2 folder if it is present it will use those files, if not present it will download from the mvn global repository and store them in .m2 itself for further uses.

Advantages of Framework:

1. Test script development is faster and easier because of reusability

2. Modification and maintenance of data is easy because data is stored in external resource

3. Modification and maintenance of element is easy because we have used POM design pattern to maintain then elements in well-organized way

4. Framework provide automatic screenshot for failed script

5. Framework provides flexibility to achieve cross browser testing

6. Framework provide accurate execution report for every execution

7. Framework provides generic reusable utility for all actions like ActionUtils, JavaScriptUtils etc..

8. Test script is optimized

Disadvantages of Framework:

1. Should be good in programming