*Moataz Nabil*

set up selenium webdriver project in eclipse?

1. Create webdriver project in eclipse
2. Download jar files
3. Include jar files in IDE

First WebDriver Script: -

Connect to Driver

1. We set property to connect to browser Driver ‘System.setproperty(key, value)

Key “driver name “name of the system property (name of web driver)

Value path of this driver

1. Then we take an object “instance” from webdriver class to deal with web driver.

Maven

Tool Manage my files and dependencies.

Very powerful and widely used java project management, build and dependency managing tool.

 It dynamically downloads Java libraries and Maven plug-ins from one or more repositories such as the Maven 2 Central Repository, and stores them in a local cache. We use maven in Selenium as a build tool or project management tool. It helps in managing all project dependencies and ensure an easy build process.

Features:

1)Manage Entire life of java project” documentation, developing, refactoring, testing, deployment, build,

2)Defining the project structure, dependencies, build and test management.

3)Maven as a build tool allow setting up the execution environment for the project code to run independently” bisa3dni eni a run project 3la l environment l m7tgaha”

4) enable a unified platform where you can check out the source code from git/svn ,Compile, package into a jar/war file.”bi2dr y connect to remote reporositry l remote w y2dr y compile l packages wl war file l mwgodes 3lih

5)you need project management tool ,maven had the project object model (POM)

File to manage project build “files kolha m3molha compilation”, dependencies”e3tmad l projects 3la files mo3ina zi selenium and testng” and documentation”

6)manage all projects related dependencies using POM.xml , it helps setting up all the configurations

**POM.xml :** A Project Object Model . It is an XML file that contains information about the project and configuration details used by Maven to build the project. It contains default values for most projects. Examples for this is the build directory, which is target; the source directory, which is src/main/java; the test source directory, which is src/test/java; and so on. When executing a task or goal, Maven looks for the POM in the current directory. It reads the POM, gets the needed configuration information, then executes the goal.

7)download all project dependency jar automatically from central repository.

Maven -🡪 has central repository

How does it handle updates of dependencies?

1)whenever you download the dependency version in pom file, maven first verifies the version of the jar file from the local repository.

2)If the version is available locally then no action will take place else it will download the upgrade from the central repository.

3)If the jars not available in the central repository the maven will look for them in the remote repository

4)we can configure the remote repository in the pom.xml to enable automatic download of the dependencies.

How to make path of Driver Dynamic ?

1. Add Drivers to new folder in project
2. Make value of system property “path “ Dynamic , 34an law et8ir yt8ir m3ah

And this is done using userdir”user directory”. + path of file System.*setProperty*("Webdriver.chrome.driver", System.*getProperty*("user.dir"+"\\resources\\chromedriver.exe"));

OR more Enhancment

String Chromepath=System.*getProperty*("user.dir")+"\\Drivers\\chromedriver.exe";

System.*setProperty*("webdriver.chrome.driver",Chromepath);

TestNG”Test Next Generation”

-Testing framework used to run test Script. Mainly used to execute unit test and now used to execute automated test cases

-TestNG provide lots of annotation to simple use and more flexible in test Execution.

Features of TestNG:

-Support for parameter” annotation bt support l parameters mmkn a8ir l parameters zi ma ana 3aiza

-Support dependent method using “ make prioritization of method a abl a, msln method bt3tmd 3la l login method “

-Test Configuration flexibility.” Deal with xml files f mmkn a8ir l configuration parameters zi ma ana 3aiza

-more powerful because application run as testNG and more powerfule reports.

-provide data driven test, p provides l test case with multiple and different data.

-Support different annotation: @beforetest, aftertest.

Why Do We Use TestNG in selenium?

1)TestNG produce readable report with selenium result as Web Driver has no native report mechanism so we use TestNG

2)Manage test execution in selenium webdriver

TestNG annotation: -

describe a piece of code that is inserted into the program or business logic used to control the flow of methods in the programming language Java

|  |  |
| --- | --- |
| @Test | Attach a class or method to become a part of the test , ai class 7ttha 3nd l annotation dh b2t goz2 mn l testing |
| @BeforeTest | Instruct the method to run before any  @test method |
| @AfterTest | Hold a method from execution till @test method finish execution |
| @BeforeMethod | Allow method to run before execution of any of the @test annotated methods. |
| @AfterMethod | Allow method to take off after all @test annotated method finish execution |
| @Parameters | Used to pass parameter to the test methods |
| @DataProvider | Marks a method as a data source for the test  Every @DataProvider annotated must always return the values Object [][] and can use it in any of the @test annotated method |
| @BeforeClass | Method executed befor any @test method in the current clas |
| @AfterClass | Run once after finishing a;; test methods in the current class |
| @BeforeGroups | Method run before the first method belonging to any of the groups involved in execution |
| @AfterGroups  @BeforeSuit | Method run after execution of all methods belonging yo any groups participated in test |
| @AfterSuit | Any such method will run after any test suit finish execution |
| @Factory | Marks a method as a factory that returns objects that will be used by TestNG as Test classes. The method must return Object[ ]. |
| @Listeners | Use them with the test classes for the logging functions |

Use of TestNG:

Add dependency of TestNG in POM.xml from TestNG dependency maven reprository.

Benefits of annotation:

* Easier to understand
* Can group the test cases using suitable annotation
* Do parallel testing
* Pass extra parameter to annotation

Paramaters: -

* IF you want method to be executed in order use the parameter priority
* Parameter require to assign value (parameter=”value”)

Prioritization:

* + Smaller number, higher priority
  + If no priority assigned, test cases will be executed in alphabetical order

Enable:

* If you have million lines of code and you want one test case not to run you don’t have to delete it you can disable it.
* Disable done by assign value false to Enable parameter (Enable=false)

Dependency:

* If you want a piece of code to run if condition satisfied or if specific method pass
* method will be executed depending on another method
* we use dependsOnMethod ().
* @parameter(dependsOnMethod={“method”})

**AlwaysRun :**

This is used when we want to make sure a method always runs even if the parameters on which the method depends, fails. If set to true, this test method will always run. E.g.: @Test (alwaysRun = true)

Grouping:

Imagine that we have 100 test cases and we want to execute 20 test case in our next test 🡪 so we can use group attribute.

We can assign a group name to this test cases and choose to execute this test cases instead of all the entire code.

And in testng.xml we add this group tags

<groups>

<run>

<include name=”groupname”></include>

</run>

</groups>

Generate Report:

Report will only generates through XML file in Test output.

Assertion:

Used in test methods to determine if the test pass or fail.

Deep in webdriver

How Web Drivers work :

Test Script ->invoke Driver -> And Driver automates browser

Dealing with Elements:

Exception thrown from finding element is nosuchelementfound

Web Element -> any seen object on page

BY- > class provide various locator strategies, find method take locator object as instance of

By class as an argument.

Locators:

1. Find Element By ID
2. Find Element By Name
3. Find Element By Class Name
4. Find Element By XPath
5. Find Element By CSS Selector
6. Find Element By Link Text
7. Find Element By Partial Link Text
8. Find Element By Tag Name
9. Find Element By ID :

Always your first solution, the fastest way to get element

As it’s unique on the page

2)Find Element by Name:

Good as ID but , it doesn’t always exist as you expected

3)Find Element by Class Name:

Not so reliable, since a Class Name is usually shared by multiple elements

4)By X-path based on xml document (relative, absolute):

Xpath is a query language for selecting nodes from xml document.

You simply can't avoid having to use XPath for at least some elements.  
It's not as bad as they say.An XPath is like a route to The element

Not stable ID , bring not stable xpath .

So we find the most stable attribute, then find xpath according this attribute.

5) Find Element By CSS Selector

The CSS Selector locator type is similar to XPath. Some people actually claim that it's faster.

### **6) Find Element By Link Text**

The **Link Text** locator type only works for links.

design patterns

Page Object Model(POM):

Record and playback pattern

Spaghetti pattern

Dry Testing pattern

Data driven testing

Testing the behavior

Page object Objects pattern

Creating page Object framework

Design pattern:

* Represents the best practices used by experienced object oriented sw developers
* Are solution to general problems the sw developer facing during sw developer
* These solution were obtained by triall and error by various sw developers

The Idea behind this approach is to allow user to record their normal testing activities and playback them through testing tool at later date such as : IDE FIREFOX PLUGIN

Advantages:

* Fast Test growth : user able to record new individual tests as fast as the tester click on links , larger test suits can be created in hours instead of weeks
* No previous Experience :it doesn’t required any experience with programming languages just click record and click around
* Element Lookup: there ‘s no need to look at page source to find an element by hand just click on desired element

Disadvantage: all these disadvantage must avoid it while writing tests using pom.

* Bad Locator :tool will record absolute path so if the elements shifts right or left , then test case will be failed
* Inflexible tests:these are the only output from the recording since the playback is identical to recording , but what if a test needs to register a unique user for each run , this need more time than automating this test
* HardCoded test data: what if you need to use flexible data and use different data depending on testing environment
* Doesn’t provide compatibility testing in various browser
* Poorly written tests: the maintenance is difficult as variable names and method named might be poorly named and strangely nested, so code will be complex
* Duplicated code : most of recording tool are not intelligent enough to detect duplicate steps and will not reuse existing code

2)The spaghetti pattern :

* Lack of architecture and design
* Impossible to understand anything without spending time finishing out and untangling each individual strand of spagetting “everything is overlapped files , drivers ,code”
* Depends on the execution order of all test but also tends to avershare the internal component with each other
* The run order is very imp. Because each test is not self sufficient and independent but needs previous shared tests,ex: login test reuires the registration test to pass successfully register a new user , instead of having already registerd user
* Variables in test suits are shared on a global level allowing individual test too much control over the whole suit

Advantage :

* Quick start:it is the easiest way and fastest way to get going , no need to sit down and plan a head ,just use a record and playback to record one long test session the spli it up into smaller chunks
* Smaller code base: since all tests depend on each other thers is no need to repeat test action within individual tests as a result each individual test is smaller in code size
* Smoke test : as smoke test need to be fast brief and leave as small test data are possible ,having several tests the include single registered user in production is good practice

Disadvantage:

* Anti pattern :building one test on top of another seems a great idea at first , that is what we have do manually but the context of automation leads to long term maintaibility problems
* Tight coupling:tightly integrated perevent code reusability
* No random order:depends on strict order of execution leads to inability to run them in random order
* No parallel test runs: having each test depends on the execution order of the whole suit prevent us from running multiple tests in parallel
* Cover up all failures: a failure at the beginning of the test suit can perevent the execution of the entire test suit
* No resilience:

Dry pattern “don’t repeat yourself “:

* The main principle is to reduce long term maintenance costs by removing all necessary duplication
* Not also duplicate code or implementation but also don’t duplicate test goals
* Ex: if the target of the current test is registration flow , this test shouldn’t fail if social media icon fail to load
* Social media icons should have a test of their own the is not related to registration test

Advantage:

* Modular tests: tests and test implementation are self sufficient . any test can run in any order , also test actions such as clicking or registering a new user
* Reduced duplication: All actions such as filling out a form are nearly kept in single place instead of having multiple copies all over the suit
* Fast Updates: having a unique actions in a single place makes it easy to update tests to mimic new growth of the application
* No Junk Code : constant upkeep of test suit , deletion of duplication prevent the test suite from having code that is no longer used

Disadvantages:

* Complicated project structure: come test action will be logically grouped with other similar action

Ex: login form and clicking login button will happen in the same implementation but some actions will end up in different file

* Lack of good IDE: They aren’t many good ide that will notify developer if a test action has already implemented so most of developer re-implement it instead of looking for them
* Constant upkeep: keeping test suit clean and applying the Dry test pattern will need dedication from the team . Duplication code nee to be deleted instead of being ignored
* Programming skills: this need to be improved by the whole team

Data Driven Testing: automation code run on any environment with dynamic data .

* Test Data is critical part of automated tests
* As Automated test developers , our goal is to make the tests fulfill their destiny of repeating identical steps
* The only way to accomplish this goal is to have as much control as possible over every single piece of data our application consumes
* Test data is not just the test out test will type into the purchase form , it is the complete state of the environment

Example of how hard coded data is a nightmare :

* URL Of WebSite:Like most of web project we have several testing environment : staging ,local, dev, so if the url is hard coded we can’t execute test in all of this phases
* HardCoded product: different test environment don’t share the same identical data most environment have subset of data available in production
* Private user data : our test environment should neve contain user data form production environment , as credit numbers and emails
* We hide data from tests in separate place like csv file or excel
* To make our test flexible enough to work on any test environment .we will need to provide them with data applicable to the said environment but the test itself doesn’t need to know what data we are using
* Test doesn’t care what username and password is used , the information fed into the test from outside in stored as variable

Behavior data driven pattern:

* Bit3aml m3 l application mn n7it l business wise, b test application hwa dh l behavior l mtlob wla la

Advantage:

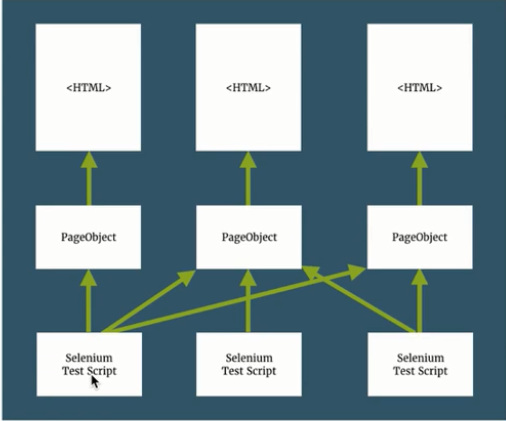
* Better test understanding: l2n enta btbni testing upon business scenario, if the test is written probably the it is possible to know exactly what the test plan do without being involved in code details
* Modular implementation: the methods that perform the actual implementation can be shared while testing

Disadvantage:

* Consistent specification language: every one describe sentence with his own view, do having all team agree on how registration flow should be will be a nightmare
* Which BDD tool to use: any team might have a long heated debate over which tool is prefect for a project
* Learning curve: each new framework will have a learning curve before everyone can use the tool

Page Object pattern:

* Describe any web page in terms of a hierarchal **Domain Specific Language (DSL)**
* DSL: helps to hide the implementation of the page, the test is no longer allow to directly interact with a given page but instead we use a framework oF classes and methods to accomplish the same goal



* This pattern abstracts the implementation details such as element IDs into framework specifically designed for the application being tested

Advantage:

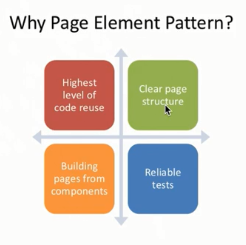
* DSL Framework: After Implementing the page Object pattern we end up with framework that describe the application point of view
  + Each action performed by a test should be easy to comprehend to anyone in given field
* Modular and reusable: each page Object is made from smaller objects such as header section or login form, the smaller objects can be shared between multiple page objects
  + Have reusable methods in the POM classes so that you can focus on the code optimization

Disadvantage:

* Complexity increased when using page Object framework. As the name implies we can’t just write test, we need to create Framework
* Programming design pattern should be followed to make code consistent and easy to Understand otherwise framework become muddled and Complex to use and maintain.

Why is Page Object Model: there are several problems in traditional automation method which are:

* Script maintenance take more time as the test suit grows
* When some locator changes, so we need to go over the whole source code to adjust locator
* Duplicate code
* Un-necessary code increase cost of maintenance for the entire project



POM Implementation:

1.Basic selenium set up

2.Analyze your application under test

3.Create page Object

4.Write test

Steps to Implement :

1. Put all driver functionality in one class( setup , cleanup, quit).
2. Analyze Your Application.

Page factory process all annotated web element and locates the element on page using annotated selection :

In order to support page Object pattern . web Driver support library contain factory class

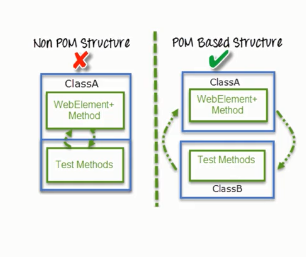
There are two different ways of implementing POM:

1) **Regular Java classes**: Please visit [**Page Object Model**](http://toolsqa.com/selenium-webdriver/page-object-model/).

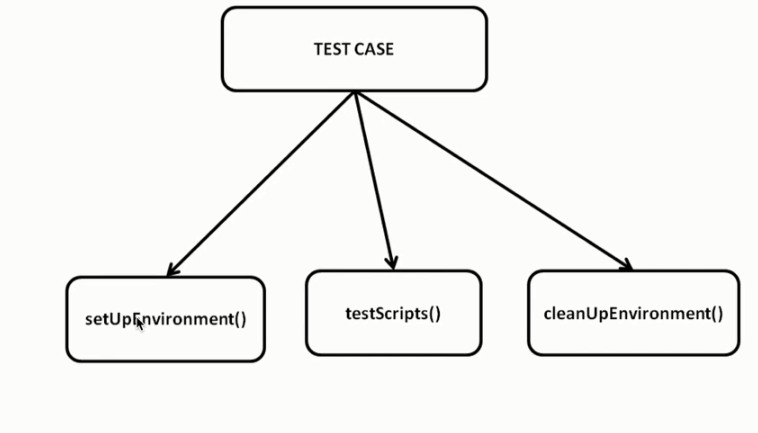
***Note****: If you are new with Selenium and Java programming language, I would suggest you to carry on with this strategy and follow Page Factory once you got good command over your test scripts.*

2) **Page Factory class:**The ***PageFactory*** Class in Selenium is an extension to the Page Object design pattern. It is used to initialize the elements of the Page Object or instantiate the Page Objects itself. Annotations for elements can also be created (and recommended) as the describing properties may not always be descriptive enough to tell one object from the other.

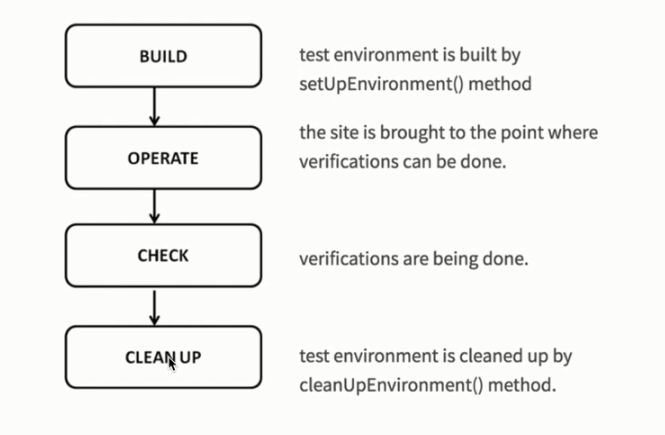
It is used to initialize elements of a Page class without having to use ‘FindElement’ or ‘FindElements’. Annotations can be used to supply descriptive names of target objects to improve code readability.



The Component of test class are :



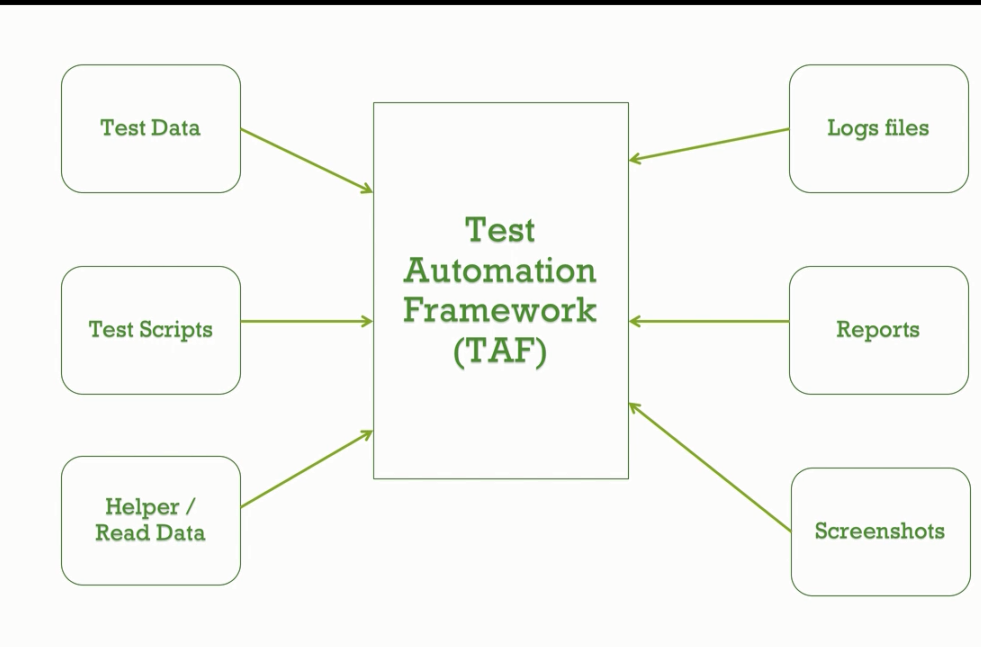
Test script when executes goes through few phases :

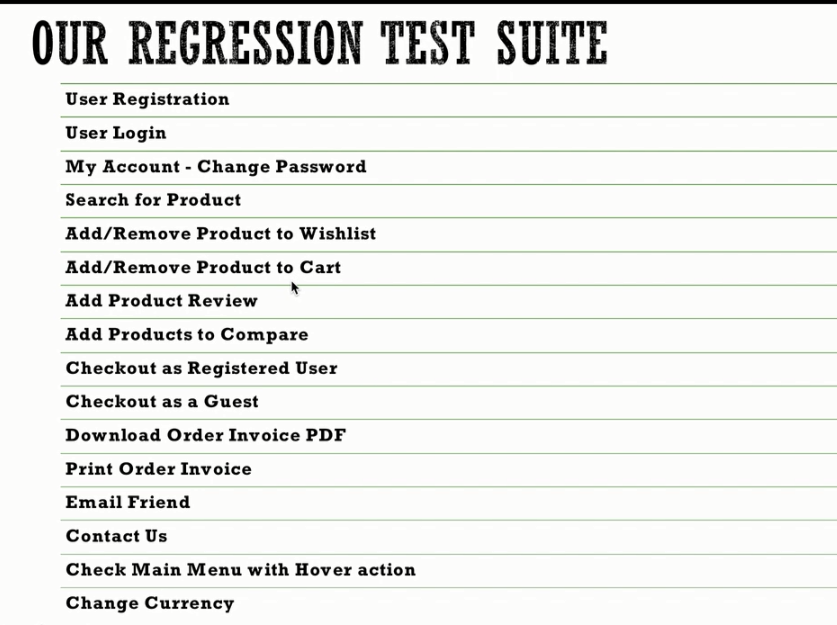


Operate 🡪 core code

Check 🡪 assertion

Component of framework:





Page Object 🡪 m3anaha en l kol page 3ndi mwazi liha class b7pt fiha l elements wl methods l ana hsht8l 3liha

Steps :

* 1. Create Files For : Drivers – screenshots –Download if exists – Upload if exists
  2. Src/main/java : Create Pages packages which contain class for each page that contain elements and methods of this page and contain PageBase “page set up that contain initialization for driver ”
  3. Src/test/java : create test java : which contain testBase
  4. H3ml Constructor in pageBase to initialize driver in it