**TestNG Framework**

* Test NG – Unit test framework. Also called TDD (Test Driven development)
* Purpose – design the test cases in systematic way
* It is an open source. And it is available in jar files.
* It provides HTML reports
* Different types of annotations
* Priorities and sequences
* Dependency
* Grouping

Data provider

// This way the execution of test cases should be done

// BeforeSUit---Setup system properties

// BeforeTest---lunch chrome browser

// BeforeClass---Login Method

// BeforeMethod---Enter URL

// Test No-2.---Google Logo

// AfterMethod---logout from app

// BeforeMethod---Enter URL

// Test No-1.---Google Title Tesst

// AfterMethod---logout from app

// AfterClass---close browser

// AfterTest---Delete all Cookies

**The above flow will be executing as per the below commands**

* @BeforeSuite
* @BeforeTest
* @BeforeClass
* @BeforeMethod
* @Test
* @AfterMethod
* @AfterClass
* @AfterTest
* @AfterSuite

This are the basic annotations of TestNG

There are multiple annotations in the TestNG will look into next

While executing the test cases TestNG executes the test cases randomly by using **priority** parameter we can set the priorities to the test cases as per the requirement, hence test cases will execute as we have parameterized the @Test(priority = 2)

**Groups** is also a parameter for testNG annotations which is used to execute the test case in particular group as per the requirements

@Test(priority = 2, groups = "Group\_name")

**DependsOnMethod** will help to depend on the particular method. Suppose the login method is not working then homepage method will not works hence we can assign as homepage method is depends on login method.

@Test(dependsOnMethods = "login")

**InvocationCount** is the one more keyword in TestNG which will execute the particular test case multiple times by using below syntax. Below syntax executes current method 10 times equaly

@Test(invocationCount = 10)

@Test(invocationTimeOut = 2, expectedExceptions = NumberFormatException.class)

The invocationTimeOut keyword will stop the test execution and will give the TestNG report

And expectedException keyword gives the expected exception that means we already know that the particular test case will xyz exception. Hence

Testing is concept of actual vs expected and to perform this we have class called Assert and it used as below. There are multiple methods in this class.

Assert.assertEquals(title, "Google", "Title is not matching");

This method will get two args as actual result and expected result

While testing a project there are thousands of methods and classes hence, we cannot run this manually one by one hence we have one option to run this all frame work at the same time. To use this concept we have to .xml file and we have to mention the page name class name etc as shown in below example:

<?**xml** version=*"1.0"* encoding=*"UTF-8"*?>

<!**DOCTYPE** suite SYSTEM "http://testng.org/testng-1.0.dtd">

<**suite** name=*"TestNG framework suite test cases"*>

<**test** name=*"TestNG suite Example"*>

<**classes**>

<**class** name=*"maven.testNG.test.ExceptionTimeoutTestNG"*/>

<**class** name=*"maven.testNG.test.GoogleTest"*/>

<**class** name=*"maven.testNG.test.GoogleTitle"*/>

<**class** name=*"maven.testNG.test.InvocationCount"*/>

<**class** name=*"maven.testNG.test.TestNGBasics"*/>

<**class** name=*"maven.testNG.test.TestNGFeatures"*/>

</**classes**>

</**test**>

</**suite**>

As shown in the above example “maven.testNG.test” is a page name and followed by class name “GoogleTest”.

Suite name and test name we can provide whatever you want.

**What is POM.?**

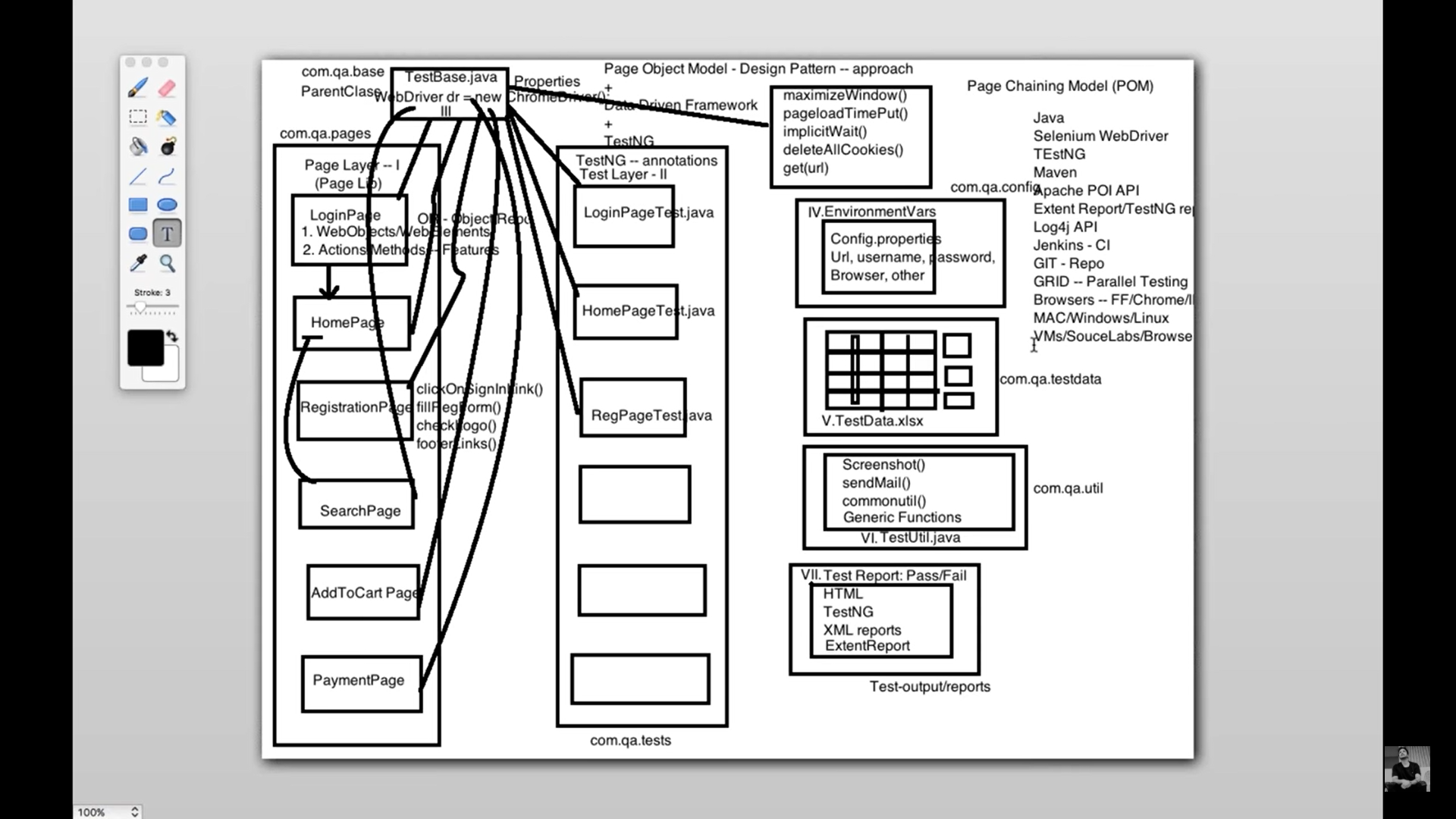
Basically, POM stands for Page Object model which is nothing but a Design pattern approach for a testing framework. There are five layers of this model.

* Page layer
* Test layer
* Test Base class (Parent class)
* Environment variables
* Test data
* Test reports

Page object model is also called as page chaining model.

In this model we will create the classes of each page of the application under test. And accordingly manage the methods of that classes as per the requirements.

Below diagram will give the brief explanation of POM



While creating a testing framework first create a baseclass and implement all the initialization of the selenium such as System properties and browsers and etc

So this is called as data driven framework. With TestNG and Selenium.

Index.HTML file path will show the html test report. Get the path of that file from eclipse and paste it on webbrowser. It will show the perfect report of all the test cases we have run. To get this file we have to refresh the project and then it will show in test output file.

There is one more file named emailable-report.html. using this file path we can also see the report of test cases.