POM

Page Object Model

\*it is an automation approach or design pattern

\*it is not a framework

\*we will integrate with Data driven framework and TestNG framework

What is POM/also called as Page chaining model.?

Packages:

Each layer will be available in a separate package.

For.eg. com.qa\_pages\_package🡪all the page layers/classes will be here.

Com.qa.test\_package🡪all test layers will be here.

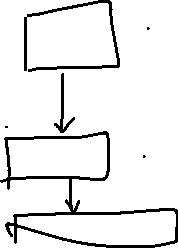
Com.qa.testdata\_package

Test-output/reports folder🡪reports will be saved here.

**1.Page layers**(or page libraries)🡪we create resp no.of classes for no.of pages and define webelements and methods in page layers/classes.

Login page –🡪1.webelements /webobjects(or Object repository)

2. Actions/Method🡪clickonsignin(),Chkfooter(),checklogo()



Home page \*all the pages are interconnected to each other.for.eg.

If we register button in login page,it will be redirected to

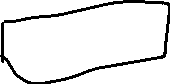
Registration page redirected to registration page,so registration page

Is lending page of login page.

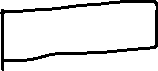
Search page



Addto cart page



Payment page



Note:

Object Repository(OR)🡪collection of all the webelements or web objects.

**2.Test layer:**

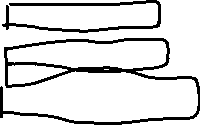


Here,we create test cases for the respective pages,and we don’t write any test cases in page layers.

These test layers will be written with the help of TestNG framework,where we use TestNG annotations

Loginpagetest.java

Homepagetest.java ++TestNG framework



Registartionpagetest.java

**3.Baseclass:** This is the parent class of all the classes or all the page layers

Testbase.java ---🡪parent class



What can be done in bae class?

1. All the pre-requisities/initialization can be done here.

2.. e.g.WEbdriver initialization

webdriver driver=new chromedriver();

3. Properties file can be de defined here

4.maximize \_window() ,pageloadtimeout() ,implicit wait method(),dleteallcookies(),driver.get(“URL”).

**4.Config.Properties/env.variables layer:**

Here,we can define all the env variables such as

URL,username,Password,browser,

**5.Excel file:**

Tedtdata.xls🡪we can maintain test data here and we can read and write the data in excel file using APACHE POI API.

**6.Utilities file:**

**TestUTIl.java🡪here we can store some generic function here such as**

Screenshot(),sendmail(),commonutil(),readdatafromexcel(),

writedatafromexcel().

**7.Reporting:**

Test Reports🡪

It can be any types of reports like XML report,HTML report,TestNG reports.

We can use different tools for reporting like:

1.TestNG report

2.Extent reports🡪very important and it will create very good HTML dashboard and HTML reports.

What are the most commonly used technologies in a selenium project ?

1.Java

2.selenium

3.TestNG

4.Apache POI API

5.Maven

6.Extentreports/TestNG reports

7.Log4j API🡪used instead of system.out.println

8.Jenkins🡪For CI(continuous integration)

9.GIT repositories(to push the code for check in and check out point of view)

10.GRID-parallel testing point of view

11.Different browsers🡪FF,IE,chrome

12.Different platforms:windows,linux,Virtual machines,MAC,Soucelabs/Browserstacks

POM design with selenium:

IT is a hybrid approach sine we are using POM with data driver framework.

1.Step1:

🡪 Create a maven proj

\*In group id,we have to give our package name.and artifact id,we can give our proj name ..

\* Delete all dummy packages .  
 \*in POM.xml file🡪delete junit dependencies,since we are not using.

\*add all dependencies(selenium API TestNG,APCHE POI API,Extent report),

2.Folder structure:

These both are the source folders and not normal folders.

\*Src/main/java🡪utilities,page classes,libraries and entire thing within main/java.

\*Src/test/java-🡪only test cases which are written under TESTNG

3.Packages:

i) Create a package under Src/main/java

All the page classes will be here

🡪com.facebook.qa.pages🡪Page layer will be created

ii) Create a package under src/test/java

Test NG Layer will be here.

Note:do not create TestNG layer in src/main/java.

🡪com.facebook.qa.testcases🡪test classes will be created.

iii) create a package under Src/main/java for base layer

base layer will be here

🡪com.facebook.qa.base

iv) create a package for under Src/main/java for config file

🡪com.facebook.qa.config

v) create a package under under Src/main/java for testdata

🡪com.facebook.qa.testdata

vi) create a package under under Src/main/java for util

🡪com.facebook.qa.util

vii)Reports:

no need to create a report package,it will be created automatically later.

4.config file:

Create a config.properties file under com.facebook.qa.config package.

Package: com.facebook.qa.config 🡪new🡪other🡪file🡪config.properties🡪finish.

Define all the variables here like username,password,URL

Note:don’t write any test data inside properties file.it is for storing all global env variables.

5.Page classes:

Create page classes under package com.facebook.qa.pages

Create a class for a particular page.its one -to-one mapping.

If we have 100 pages in our proj,we need to create 100 classes.

6.Base class:

Create a base class under package com.facebook.qa.base

Using inheritance concept.we can define some generic variables and methods in base class and access these in child classes(both page classes and test classes ),since base class is the parent class here.

\*note:donot select main method here.

1. All the pre-requisities/initialization can be done here.

2.. e.g.WEbdriver initialization

webdriver driver=new chromedriver();

3. Properties file can be de defined here

4.maximize \_window() ,pageloadtimeout() ,implicit wait method(),dleteallcookies(),driver.get(“URL”).

Check for code im PO<\_structure project🡪baselayer🡪testbase

6.util class

Create a util class testutil.java under util layer com.facebook.qa.util

Here, **public** **static** **long** *Pageload\_time\_out*=30;

**public** **static** **long** *implicitly\_wait*=40;

we can define the pageload timeout etc.here and we need not hardcode the values in base class,so if we need o change the time later,we can just change here alone.

7.define relationship between child classes and parent class:

All page layer classes and test layer classes are child classes and base layer class is a parent class.

For.eg.loginpage is a child class and test base is parent class,them use

**public** **class** Loginpage **extends** TestBase

so that it inherits all prop of base class .

8.Page class:

We have to create a method and define

1. object repository/page factory
2. create all the findby elements related to login page

//instead of driver.findelement,we need to use like this

@FindBy(name="email")

WebElement username;

@FindBy(name="password")

WebElement password;

@FindBy(xpath="//\*[@class='ui fluid large blue submit button']")

WebElement loginbtn;

//pagefactory initialization with constructor

**public** Loginpage()

{

1. PageFactory.*initElements*(*driver*, **this**);

//this gets all the elements of this particular class

Note:

Testcases shd be independent or separated.

Each time,when browser opens,

initialization occurs,login occurs and driver shd be closed for each test case

foe.eg.if we have 50 test cases,

50 times,initialization happens,

50 times login occurs

50 times browser will be closed

9.creating runner class

Instead of executing each class separately ,we can create a runner class and execute all the classes at a single shot.

Testing.xml

How to create?

Project🡪src🡪create a src folder(src/resources)🡪new🡪file🡪other🡪file🡪nest🡪testng.xml

Order:

<suite name=’’>

<test name=’’>

<classes>

<class=”package.classname”></class>

</test>

</suite>

!-- function f2-->shortcut for rename functionality

to comment-->selevt text-->right click-->source-->add block comment

-->

Hybrid approach:

Data driven +POM is recommended and easy to maintain.

Keyword drivern framework is not recommended since it is difficult to maintain

How to implement data driven approach in POM?

🡪using dataprovider(in the form of excel sheet)

🡪using APACHE POI API ,to read data from excel

Add @cachelookup concept:

🡪new TestNG annotation

It will store the webelement in its cache memory.

Instead of getting the webelement from the page each time,it will get it from cache,so that speed or performance will be improved.

What happens if we don’t use @ cachelookup ?

Consider e.g. @FindBy(name="first\_name")

WebElement firstname;

When we need to call this webelement 10 times,it will interact 10 times with td which contains firstname and it will try to get xpath 10 times from HTML DOM.

But when we use @cachelookup,

It will create a small memory and the webelement will be stored in that cache memory

Instead of going to the browser again and again,it will get directly from cache memory.

Disadvantages:

When webpage is updated with new version or HTML DOM structure is changed or page is refreshed,then webelements in @cachelookup will become stale or inactive.

So,if we know that there wont be any change with DOM structure,only then it shd be used.

\*\*interview question:

How will u improve the performance of script?

🡪we can use @cachelookup to improve the speed

Add webdriver fire event:

🡪to generate selenium action logs.

What are action logs:

It is used to trace what is happening when code is executed stepwise.it diplays all the events one by one

Steps:

1.add a class webdriver listener class in utility package.

2.add some standard code in webdriver listener class

3.it shd extend base class and it shd also implement webdrivereventlistener

4.add four lines of code in base class after driver initialization.

//create object for Webdriverlistener to register with eventfiringwebdriver

Webdriverlistener elistener=**new** Webdriverlistener();

//create object for eventfiringwebdriver

EventFiringWebDriver e\_driver=**new** EventFiringWebDriver(*driver*);

//now register elistener with e\_driver

e\_driver.register(elistener);

*driver*=e\_driver;

Generate reports in POM:

Developed by relevant code company

Used frequentky since it gives good dashboards.

How to generate extent report?

Steps:

Method 1:

1.add library/dependency of extent report

2.create a package extentreportlistener like com.freecrm.qa.Extentreportlistener

3.create a class Extentreportlistener in this package.

Purpose of listener:

It will listen each and every activity of the execution and generates report .

🡪no need to remember the class,it is a listener standard template provided.

But we need to import all.

🡪extentreportlistener class implements IListener interface.it also used TestNG Listener class ,so we need TestNG to use this extentreportlistener class.

4.Add a code in testing.xml

We have to add a listener entry in this.

Where to add in testing.xml?

After <suite> tag and before <test>tag.

<listeners>

<listener>

</listener class-name=”packagename.listenerclassname”>

</listeners>

What happens now when we run testing.xml>

Extent report wil be created in test-output folder.

🡪test-output/extent.html

One more easy way to generate Extent report :

Method 2:

Step 1:

Add dependency:

<dependency>

<groupId>com.relevantcodes</groupId>

<artifactId>extentreports</artifactId>

<version>2.41.2</version>

</dependency>

Step2:

Create reference variable for extent report and ExtentTest as class variables.

WebDriver driver;

ExtentReports extent;

ExtentTest extenttest

Step 3:

Function to generate report in specific location and close it after generated

@BeforeTest

**public** **void** extentreport()

{

//create object ref for extentreport and pass 2 parameters:

1.path to generate report

2.boolean exp:true🡪is given to replace new report with old one each time

extent=**new** ExtentReports(System.*getProperty*("user.dir")+"/test-output/Extentreports.html", **true**);

}

@AfterTest

**public** **void** endreport()

{

extent.flush();

extent.close();

//to close the connection with extent report

}

Step 4:

To get the result of extent report:

We need to import ITestResult Testng listener class

@AfterMethod

**public** **void** teardown(ITestResult result) **throws** IOException

{

//here,we need to import ITestResult Testng listener class

//purpose of ITestResult-it stores the results of all the test cases whether pass or fail

**if**(result.getStatus()==ITestResult.***FAILURE***)

{

//getName-->to get test case name

//getThrowable-->to throw all the error and exception

extenttest.log(LogStatus.***FAIL***, "Name of test case failed is "+result.getName());

extenttest.log(LogStatus.***FAIL***, "Name of test case failed is "+result.getThrowable());

}

**else** **if**(result.getStatus()==ITestResult.***SKIP***)

{

extenttest.log(LogStatus.***SKIP***, "Name of test case skipped is"+result.getName());

}

**else** **if**(result.getStatus()==ITestResult.***SUCCESS***)

{

extenttest.log(LogStatus.***PASS***, "Name of test case passed is"+result.getName());

}

extent.endTest(extenttest);//ending test

driver.quit();

}

**Step 5:**

Add this code in the beginning of every test case

@Test

**public** **void** login()

{

extenttest=extent.startTest("login"); **}//Login is the name of test case**

How to add screenshot in extent report for failed test cases in selenium?

With above script,with logic to generate extent report,add script for adding screenshot also.

//function to take screenshot

**public** **static** String getscreenshot(WebDriver driver,String sreenshotname) **throws** IOException

{

String date=**new** SimpleDateFormat("yyyyMMddhhmmss").format(**new** Date());

//here,new Date() is the object of date class

//now add logic for screenshot

File src=((TakesScreenshot) driver).getScreenshotAs(OutputType.***FILE***);

String destination=System.*getProperty*("user.dir")+"/Failedtestcasescreenshots/"+sreenshotname+date;

FileUtils.*copyFile*(src, **new** File(destination));

**return** destination;

}

String screenshotpath=Logintest.*getscreenshot*(driver, result.getName());

extenttest.log(LogStatus.***FAIL***, extenttest.addScreenCapture(screenshotpath));//to add screenshot in extentreport