**Page Object Pattern using PageFactory**

**Why page object model :**

Creating Selenium test cases can result in an un-maintainable project. One of the reasons is that too many duplicated code is used. Duplicated code could be caused by duplicated functionality and this will result in duplicated usage of locators. The disadvantage of duplicated code is that the project is less maintainable. If some locator will change, you have to walk through the whole test code to adjust locators where necessary. By using the ***Page Object Pattern using PageFactory*** we can make non-brittle test code and reduce or eliminate duplicate test code. Beside of that it improves the readability and allows us to create interactive documentation. Last but not least, we can create tests with less keystroke. An implementation of the page object model can be achieved by separating the abstraction of the test object and the test scripts.

### Advantages of using Page Object Pattern:

* Easy to Maintain
* Easy Readability of scripts
* Reduce or Eliminate duplicacy
* Re-usability of code
* Reliability
* Coming to  **Page Object Model**, we should keep our tests and element locators separately, this will keep code clean and easy to understand and maintain.
* The Page Object approach makes test automation framework programmer friendly, more durable and comprehensive.
* Another important advantage is our ***Page Object Repository is Independent of Automation Tests***. Keeping separate repository for page objects helps us to use this repository for different purposes with different frameworks like, we are able to integrate this repository with other tools like [**JUnit**](https://junit.org/)/[**NUnit**](http://nunit.org/)/[**PhpUnit**](https://phpunit.de/) as well as with [**TestNG**](http://testng.org/)/[**Cucumber**](https://cucumber.io/)/etc.
* Test cases become short and optimized as we are able to reuse page object methods in the **POM** classes.
* Any change in UI can easily be implemented, updated and maintained into the Page Objects and Classes.

constantsPackage/Constant class:

In this we maintain the all reusable file paths

APPLICATION\_UNDER\_TEST

CONFIG\_FILE\_PATH

XLS\_FILE\_PATH

EXTENT\_REPORTS\_PATH

UPLOAD\_FILE\_PATH

SCREENSHOT\_PATH

*TestBase(helper)*

*It should be TestNG class*

*@before suite*

initializing the log4J and properties class

Browser selection , maximize

@AfterSuite

Closing driver

TestBase also have frequently used reusable methods like text field and text area entry ,Dropdown,radio button and check box selection, Browse document, click on button , links , mouse hoverand take screenshot etc ,In this class all possible exceptions are handled

Also have extent report pass and fail methods

In page factory design pattern we are using below packages in src/main/java

1)constants

2)pages

3)utility packagh

In page factory design pattern we are using testcases package in src/test/java

Constants :

* Under this model, for each web page in the application, there should be corresponding page class, where we maintain locators of web elements of that web page , and each webelement should declare as public static final String and the reference variable should be capital , variable should be public becoz should be accessible to within the project , static becoz without creating the object we can access the locator , final becoz this is the final value for that variable

Utility package:

We are maintaining separate classes to differentiate reusable methods

Those classes we are maintaining in utility package for easy maintenance .

We have following classes

1) **Assesrtion util:**

For every test case we need to check whether are there any deviation from the business requirements or not like is it navigating to a valid page or alert presence or verify title etc ,for this we have developed some reusable methods along with all possible exception handling

Here we can get all the assertion methods which are reusable like

1)verify title

2)verify text(element with text), verify Text (with expected and actual STrings) , verify alert message (based on alert text) , verify alert present , verify bdrop down values , is element present

2)**Windows :** here **we developed all the reusable methods for Windows handling like** getCurrentWindowTitle, waitForNewWindowAndSwitchToIt , closeAllOtherWindows , switchToFrame (with id and Name) , switch to default name.

switchToIframe, switchToPreviewFrame

3)WaitUtil : Here we developed all kinds of wait(synchronization) methods

Wait for element visible , wait for list of elements to be visible, wait for element to be clickable , waitForEelementToBeSelected , waitForPagetoload , , waitForElementVisible(By by) , WebElement waitForAndReturnElement(By by), waitForAndReturnElement(WebElement element)

4)Error util

5)Function util:

moveMouseToElement(WebElement element),

MouseClick(WebElement element),

browseDocument(WebElement element,String browseDocument ), doubleClick(WebElement element),

rightClick(WebElement element),

dragAndDrop(WebElement sourceElement, WebElement destinationElement), scrollToBottom(WebDriver driver), scrollToelement(WebDriver driver, WebElement element),

doClear(WebElement element),

mouseHoverJScript(WebElement HoverElement),

safeJavaScriptClick(WebElement element),

calander(String xpath,String fromdate),

tocalander(String ByID,String todata), verify\_Values\_In\_Dropdown(List<WebElement> listOfElements, String[] strValues),

Select\_The\_CheckBox\_from\_List(WebElement element,String TagName, String valueToSelect)

6)Test util

7)Extent manager util

8)XlsReader

Pages :

The page class will find the webelements of that web page and also contains page methods which perform operations on those webelements

Name of these methods should be given as per the task they are performing

Ex: if user wwants to enter text in the citizen name the method name should be like enterCitizenName()

we follow the concept of separation of Page Object Repository and Test Methods. Additionally, with the help of PageFactory class, we use annotations **@FindBy** to find WebElement. We use initElements method to initialize web elements , for each webelement in the page t

**@FindBy** can accept **tagName, partialLinkText, name, linkText, id, css, className, xpath**as attributes.

For each functionality ,we verify whether the application meets business requirements or not like success messages , validation alerts , values in the drop down , for example when user logged in to the applications we verify the login functionality based on logout button



Test Cases :

Parameterization : It run a test case multiple times with different input and validation value

Dataprovider : We are passing parameters through excel using Dataprovider annotation

Data Provider is a method annotated with *@DataProvider*. A Data Provider returns an array of objects.

In this excel we are maintaining two sheets

Testcases(sheet 1) : consists of three columns headers

1. TCID
2. Description
3. Run mode

|  |  |  |
| --- | --- | --- |
| TCID | Description | Runmode |
| LoginTest | Verifying Login Functionality | Y |
| LoggedInPageTest | Add stock functionality | N |

Test data (sheet 2)

1)Run mode(Y 🡪run the test case , N 🡪 skip the test case )

2)Data flag(for positive test case : P , Negative test case N)

3)TC (test case ID with serial number)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| LoginTest |  |  |  | |  |
| Runmode | DataFlag | TC | | LoginId | Password |
| N | P | TC-001 | | 9866752267 | 123 |
| N | N | TC-003 | | 33 | guest |
| N | N | TC-004 | | AMCD | euest |
| N | N | TC-005 | | Demo | emo |

The class name of test case should be same as in the excel of test case (TC ID ) and in test data

For ex if user want to execute a test case “LoginTest”, then tool will check whether the run mode for the corresponding test case flag is Y or not , if it is N it will skip the test case otherwise (Y) it will go to the test data sheet and check for the corresponding testcase class name , and also check for the run mode of the particular class test data , if the the test data run mode is N it will skip other wise it will take the data and provide to the dataprovider method as test data .

Test report will generate based on two conditions

1)if the test case type(data flag) is positive (P) and assertion false occurs then the test case will fail

if the test case type(data flag )is negative(N) and assertion true occurs then the test case will fail

for each failure of test case , tool will generate screenshot in screenshot directory

finally tool will generate extent report(html report) in the reports directory