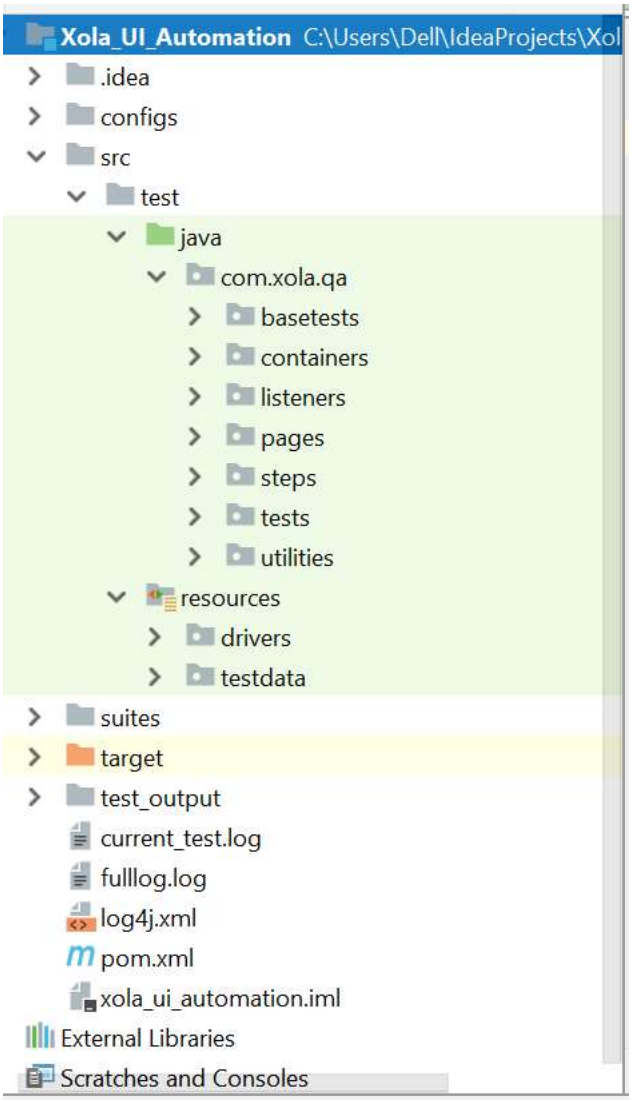
Contents

1. Framework structure



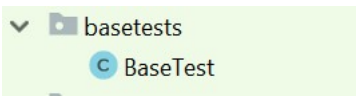
1.1 Xola_UI_Automation/configs: It contains application properties file "config.properties" which contains following attributes.

url, browser, timeout and testDataFile

```
url=https://sandbox-checkout.xola.com/index.html#seller/58e235b107876cdd1f8b45e2/experiences/58e2371107876cdd1f8b45e5
browser=chrome
timeout=30000
testDataFile=BookingDetails.xlsx
```

1.2 Xola_UI_Automation\src\test\java\com\xola\qa\basetests: It contains the "BaseTest.java" where the driver and reporting components are initialized .

***N.P: Every new test case class should extend the BaseTest class.**



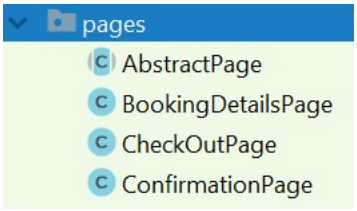
1.3 Xola_UI_Automation\src\test\java\com\xola\qa\containers: It contains the java classes for some specific part of some pages which we can/may use for different pages.

E.g;

For Xola ui automation I have used the containers for calendar which is very useful to separate the elements for calendar object.

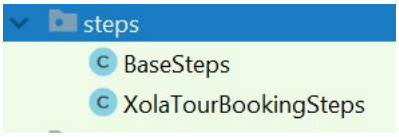
1.4 Xola_UI_Automation\src\test\java\com\xola\qa\listeners: It contains the Xola specific listener actions which is implemented the methods of ITestListner where some specific actions are defined to execute in different lifecycle stages of the test run.

1.5 Xola_UI_Automation\src\test\java\com\xola\qa\pages: It contains the java classes for pages with separate methods for each element along with "**AbstractPage.java**" which every new introduced page should extend.



1.6 Xola_UI_Automation\src\test\java\com\xola\qa\steps: It contains the java classes where all the action for the page objects are written as per required which inherits from" BaseSteps.java". " BaseSteps.java" contains some common actions which can be used for every step class

***N.P: Every step class should extend BaseSteps class.**



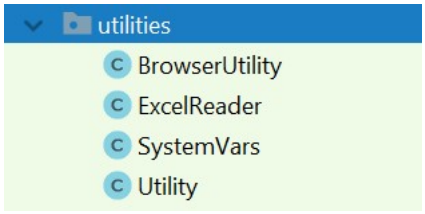
1.7 Xola_UI_Automation\src\test\java\com\xola\qa\tests: It contains all the test classes which use the testng annotations (@Test etc.) to run the tests.

Multiple test methods can be written in this class with an @Test for each test method.

1.8 Xola_UI_Automation\src\test\java\com\xola\qa\utilities: It contains java classes with reusable methods which can be used throughout the framework.

E.g;

BrowserUtility.java, ExcelReader.java etc.

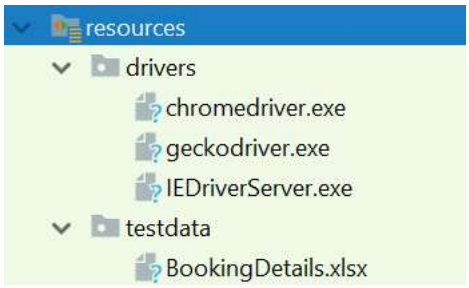


1.9 Xola_UI_Automation\src\test\resources: It contains the browser drivers and testdata file.

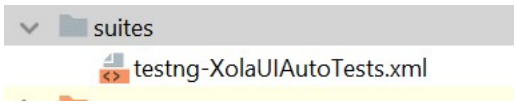
Drivers: It contains the useful drivers to open the browser and run the test

E.g; chromedriver.exe, geckodriver.exe etc.

Testdata: It contains the excel file with the test data.

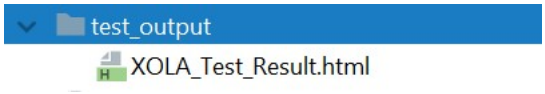


1.10 Xola_UI_Automation\suites: It contains the testng xml file which used to run the tests locally or through Jenkins.



1.11 Xola_UI_Automation\test_output: Reports generated on completion of the test run will be available here.

I have used extent reports which will generate html test reports. Test report will contain an screenshot in case test is failed.



2. Add new auto test

- To add new auto test create a method with @Test annotation under any test class or add a new java class file which extends the BaseTest class.
- In case of adding new tests with different pages add a new page classes and extend it from AbstractPage class and add the page object methods in the new page classes.
- Create new methods to do actions on the above added page objects in steps class.
- Create the object of the newly added steps class in test method and call the appropriate steps to verify the application.
- After adding the test method/class add the class path to the testng suite to run.

3. Add newly added fields to the page

- Add new method to return the WebElement for newly added fields in the appropriate page class.
- Call the WebElement method through class object in steps class in the appropriate action method or create a new method and do the necessary action as needed.
- In case new action method is created call the action method in tests under test class and run the tests.

4. Framework Properties

Framework properties file "config.properties" is available under **Xola_UI_Automation/configs** folder which contains following attributes.
Attribute: **url, browser, timeout and testDataFile**

In case new attribute is added, we can access the newly added attribute by using **"Utility.getProperties.getProperty(attribute name)"**

```
url=https://sandbox-checkout.xola.com/index.html#seller/58e235b107876cdd1f8b45e2/experiences/58e2371107876cdd1f8b45e5
browser=chrome
timeout=30000
testDataFile=BookingDetails.xlsx
```

5. Test data

- Test data file is available under Xola_UI_Automation\src\test\resources\testdata folder.
- Testdata file is in excel format, we can access the test data from the BaseTest class using "bookingDetails" and/or "creditCardDetails".
- In case new file is added for the test data, replace the "testDataFile" attribute value in properties file to use it.