**ABN Mobile automation assignment**

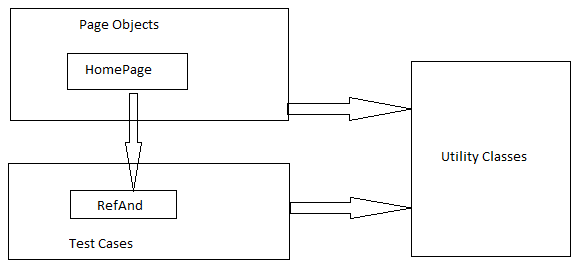
|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| Aug 9th, 2019 | Madhur\_Sharma03 (686931) | Workflow explanation for the Framework used in the ABN Mobile automation assignment |

**Pre-Requisites:**

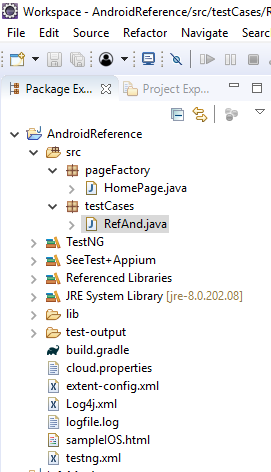
1. The **ReferenceAndroid.apk** file has been created using Android Studio from the shared project.
2. The ReferenceAndroid.apk file has been installed in the testing device.

**Page Object Model (POM):**

In designing an easily maintainable Appium project, we need to minimize writing a duplicated code. If some locator changes, we would have to go through the entire test code to modify the locators wherever necessary. By using a page object model, we could eliminate duplicate code and improve readability. This implementation can be achieved by separating the code design of test object and test scripts. So keeping all this in mind we have used POM for automating the **ReferenceAndroid** app.

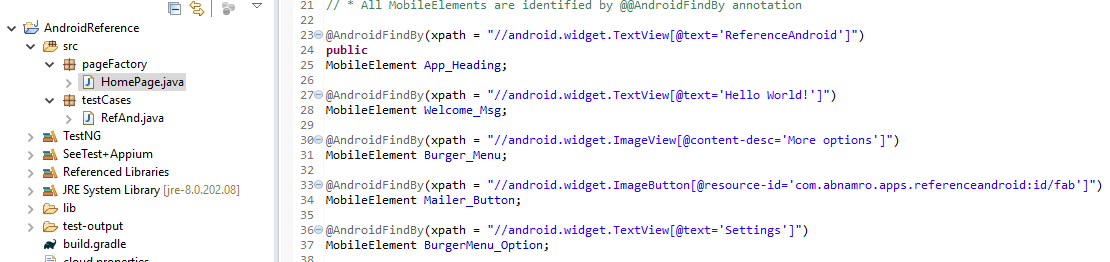
****

**Folder Structure details:**

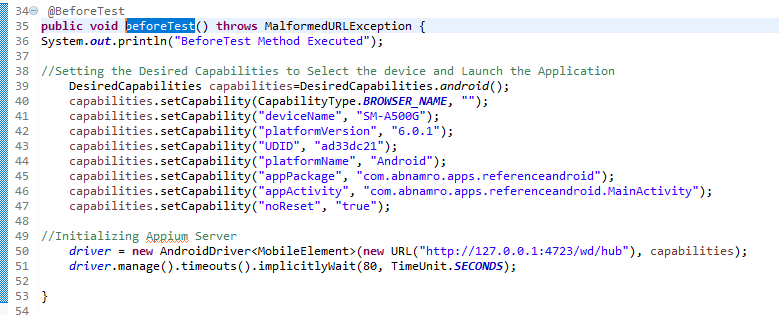


**Framework details:**

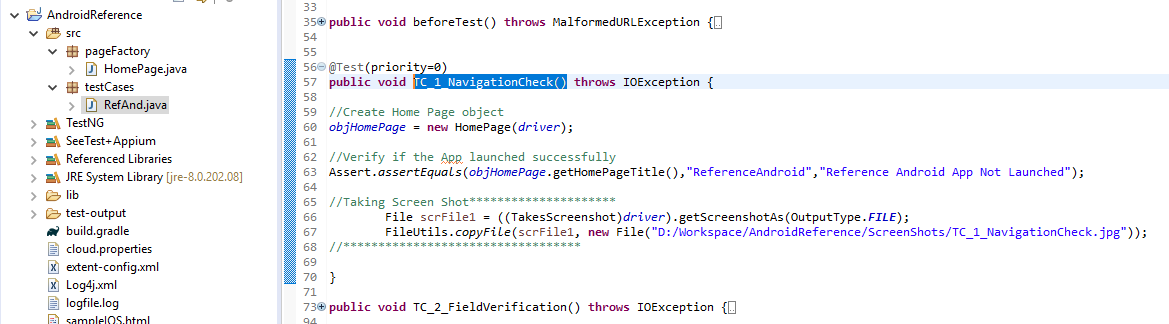
1. A Java Project **AndroidReference** has been created.
2. Under **AndroidReference** project we have two packages a) pageFactory and b) testCases
3. **pageFactory** – Under this package a **HomePage.java** class has been created which is having MobileElements of the application home page and also contains Page methods which performs operations on those MobileElements. This class will be invoked by the RefAnd.java class under testCases Package.



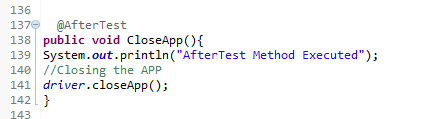
1. **testCases** - Under this package a **RefAnd.java** class has been created which is having below methods:
2. **@beforeTest –** Under thisannotation a method for setting the Desired capabilities for Android Mobile app and initiating Appium driver to connect devices is written.



1. **@test –** In this annotation we have written all the test case methods to test the different test scenarios. For every test step, a separate test method is to be written. Eg: TC\_1\_NavigationCheck ()



1. **@afterTest -** Under thisannotation a method for closing the app is written once all the tests under @test gets executed.



1. The Methods invoked using TestNG Framework.
2. Screenshots are getting stored in the respective folders, here we have used predefined directory to store the screen shots of each test method i.e. (D:/Workspace/AndroidReference/ScreenShots/)

**Execution Workflow**

**Test Results & Screenshots**

**TestNG**

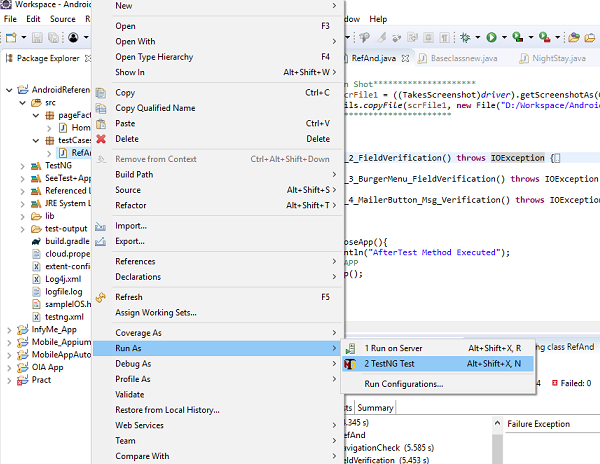
**Test Methods**

**Generic Method**

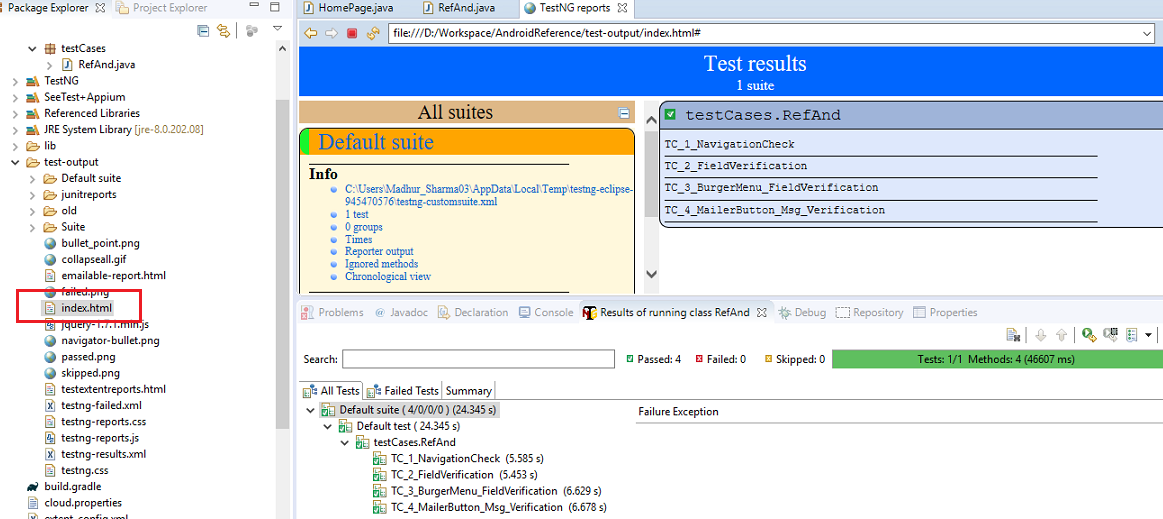
**Test Data**

**Steps to Run the Test Case: -**

1. Right click on RefAnd.java class under testCase Package and > Run as > TestNG Test



1. It will execute all the test methods of RefAnd.java class, after successful execution we can analyze the test results from test result window or we can open double click on index.html file under test-output folder.



1. For reviewing the screenshot, we can navigate to the directory defined in the code i.e. (D:/Workspace/AndroidReference/ScreenShots/)