**Automation of VU Application Using Selenium Tool**

Online appointments of the VU application are considered to provide a sample of how selenium can be used for the automation.

Please find below the scenario that has been automated.

## Scenario:

## *Launch a new Firefox browser.*

## *Open* <https://vuqa.premex.com/vu/Login.aspx>

## *Login with the Premex user credential.*

## *Book one appointment.*

## *Search the Booked appointment.*

## *Rearrange the appointment.*

## *Cancel the appointment.*

## *Add appointment in Manage Expert diaries.*

## *Cancel appointment in Manage Expert diaries.*

## *Close the Browser.*

The process involved and the script created for the above scenario are described below.

## **Page Object Model**

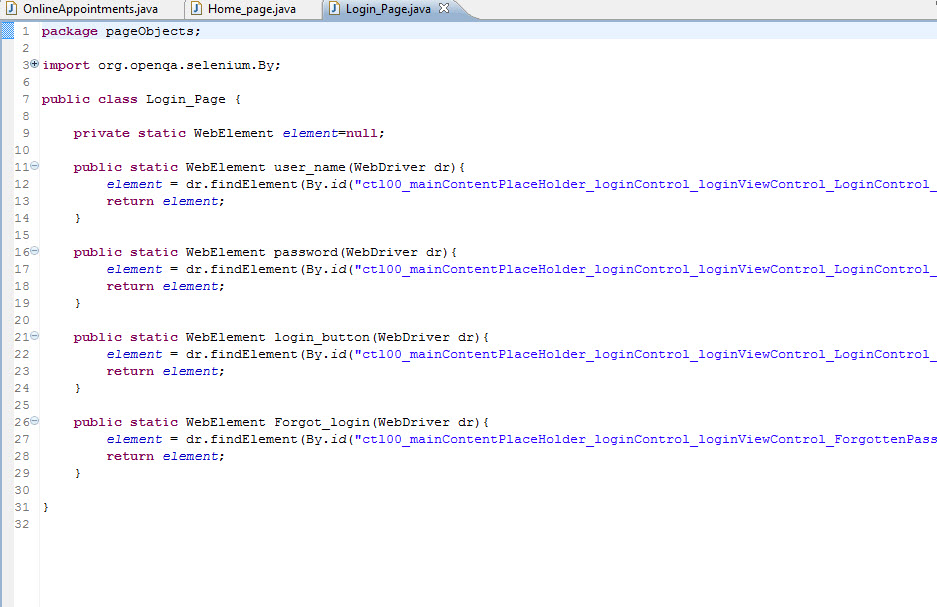
Creating Selenium test cases can result in an unmaintainable project. One of the reasons is that too many duplicated code is used. Duplicated code could be caused by duplicated functionality and this will result in duplicated usage of locators. The disadvantage of duplicated code is that the project 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 model 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.

## **Steps**

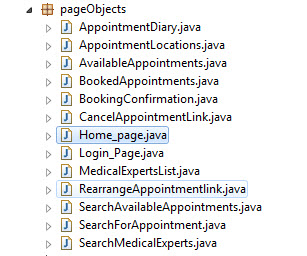
* Create a ‘[New Package‘](http://toolsqa.wpengine.com/selenium-webdriver/configure-eclipse-with-selenium-webdriver/) file and name it as ‘**pageObjects’**, by right click on the Project and select **New** > **Package**. It is always recommended to use this structure, as it is easy to understand, easy to use and easy to maintain.
* Create a ‘[New Class‘](http://toolsqa.wpengine.com/selenium-webdriver/configure-eclipse-with-selenium-webdriver/) file and refer the name to the actual page from the test object, by right click on the above created Package and select **New**>**Class**. In our case it is **Home\_page**and **Login\_Page**.
* Now create a **Static Method** for each **Element** (Object) in the Home Page. Each method will have an **Argument** (dr) and a **Return** value (element).



* Driver is being passed as an Argument so that Selenium is able to locate the element on the browser (dr).
* Element is returned, so that an Action can be performed on it.
* Method is declared as **Public Static**, so that it can be called in any other method without instantiate the class.
* Follow the same rule for creating **Login\_Page**class.



* Create page objects class for all the pages like **Home\_page** and **Login\_page** class.
* Your Project explorer window will look like this now.

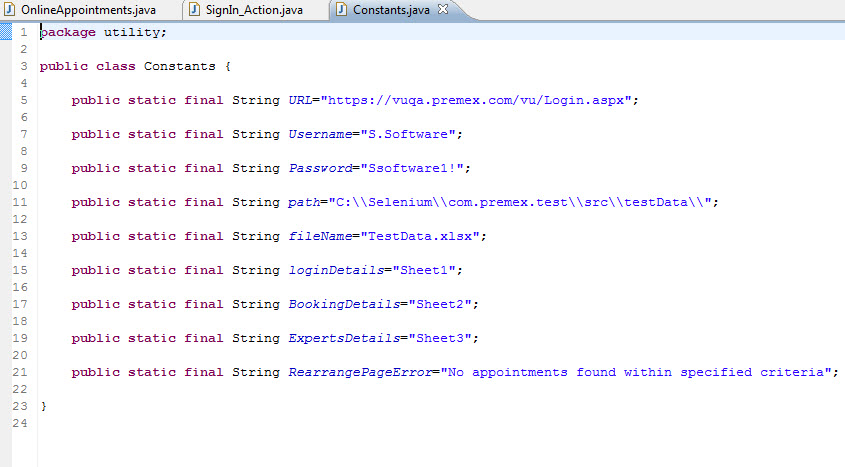


## **Constant Variables**

Test data can be of two types, fixed or variable. If it is fixed, we can easily hard code the test data in to our test scripts. But sometimes the fixed test data is also used in so many scripts and if it gets changed then it is a huge task to update all the effected test scripts for example the URL of your test application. It remains same but once you shifted to other environment, you need to change it in all of your test scripts. We can easily place the URL in Text file or Excel file outside our test scripts but Java gives us special feature of creating Constants variables.

## **Steps**

* Create a ‘[New Package](http://toolsqa.wpengine.com/selenium-webdriver/configure-eclipse-with-selenium-webdriver/#package)’ file and name it as “**utility**”, by right click on the Project and select **New** > **Package**.
* Create a ‘[New Class](http://toolsqa.wpengine.com/selenium-webdriver/configure-eclipse-with-selenium-webdriver/#Class)’ file, by right click on the above created Package and select **New**>**Class** and name it as **Constant**.
* Assign keywords in this class to your fixed data for e.g. URL, Username and Password.



* Constants Variables are declared as **public static**, so that they can be called in any other methods **without instantiate** the class.
* Constant Variables are declared a **final**, so that they cannot be changed during the execution.

## **Data Driven Framework with Apache POI – Excel**

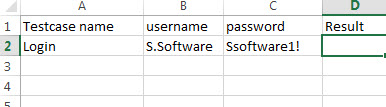
**Data-driven testing (DDT)** is a term used in the testing of [computer](https://en.wikipedia.org/wiki/Computer)[software](https://en.wikipedia.org/wiki/Software) to describe testing done using a table of conditions directly as test inputs and verifiable outputs as well as the process where test environment settings and control are not hard-coded. In the simplest form the tester supplies the inputs from a row in the table and expects the outputs which occur in the same row.

## **Reading data from the Excel**

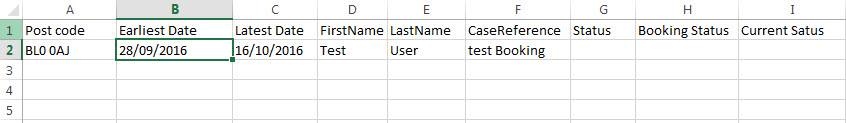
We need a way to open this Excel sheet and read data from it within our Selenium test script. For this purpose, I use the Apache POI library, which allows you to read, create and edit Microsoft Office-documents using Java.

## **Steps**

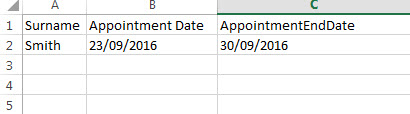
* [Download JAR files](http://toolsqa.wpengine.com/selenium-webdriver/download-apache-poi/) of Apache POI and [Add Jars](http://toolsqa.wpengine.com/selenium-webdriver/add-apache-poi-jars/) to your project library. That’s all about configuration of Apache POI with eclipse. Now you are ready to write your test.
* Create a ‘[New Package](http://toolsqa.wpengine.com/selenium-webdriver/configure-eclipse-with-selenium-webdriver/#package)‘ file and name it as ‘**testData’**, by right click on the Project and select **New** > **Package**. Place all of your test data in this folder (package) whether it is a sql file, excel file or anything.
* Place an**Excel** file in the above created package location and save it as **TestData.xlsx**.
* Fill the data in the excel like below image
* Login details



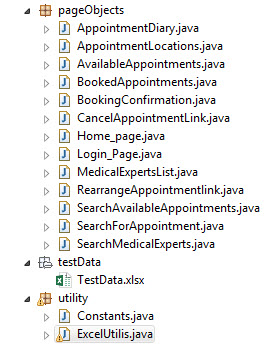
* Appointment Booking detail

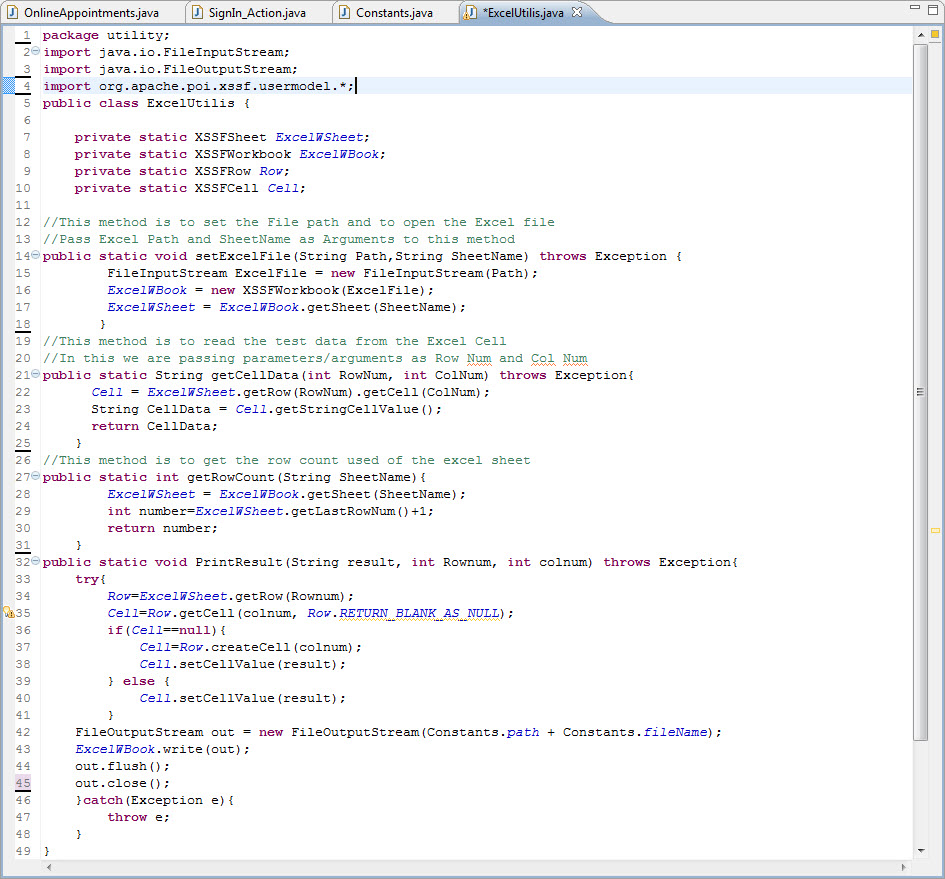


* Appointment date detail



* Create a ‘[New Class](http://toolsqa.wpengine.com/selenium-webdriver/configure-eclipse-with-selenium-webdriver/#Class)‘ file, by right click on the ‘**utility**‘ Package and select **New**>**Class** and name it as ‘**ExcelUtils**‘**.** First we will write basic read/write methods.
* Your Project explorer window will look like this now.





# TestNG

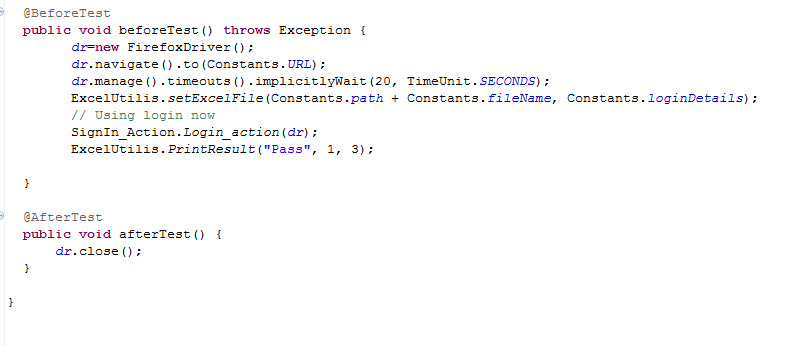
* TestNG is a testing framework inspired from JUnit and NUnit but introducing some new functionalities that make it more powerful and easier to use.
* In simple words TestNG is a tool that help us to organize the tests and help us to produce the test reports.
* TestNG framework can be used for automation testing with Selenium (web application automation testing tool).

# TestNG Advantages

* Multiple built in **Annotations** which are easier to use and understand.
* Test method can be **dependent** to other method.
* Test cases can be **Grouped** and can be execute separately by groups.
* TestNG has built in HTML report and XML report generation facility. It has also built in logging facility.

## **Steps**

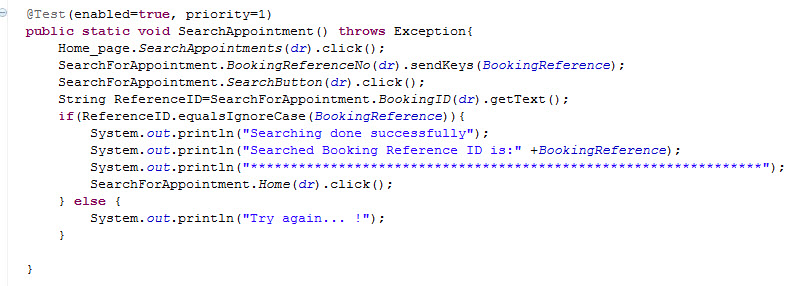
* 1) First step is to [Install TestNG](http://toolsqa.wpengine.com/selenium-webdriver/install-testng/). It is easy to install TestNG, as it comes as a plugin for Eclipse IDE.
* 2) Create a ‘[New Class](http://toolsqa.wpengine.com/selenium-webdriver/configure-eclipse-with-selenium-webdriver/)‘ by right click on the ‘**ExecutionEngine’** package then select **TestNG**>**Create a TestNG Class** and name it as **OnlineAppointments**.
* 3) Let’s divide the test case in to three parts.
* **@BeforeMethod**:Launch a new Firefox browser then open https://vuqa.premex.com/vu/Login.aspx and login with the Premex user credential.
* **@Test**: Execute BookAppointment, SearchAppointment, RearrangeAppointment, CancelAppointment and ManageExpertsDiariesactions and Log out.
* **@AfterMethod**: Close Firefox browser.



* BookAppointment method.



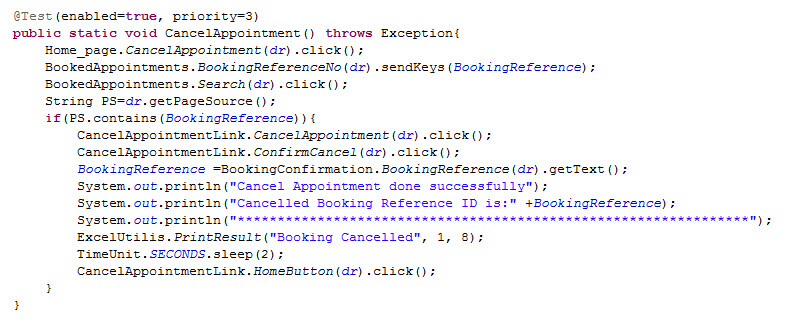
* SearchAppointment method.



* RearrangeAppointment method.



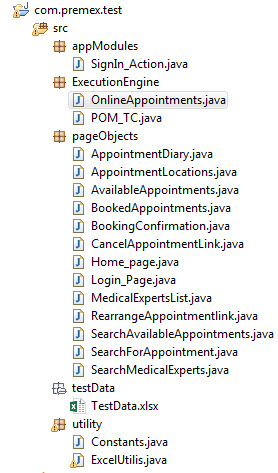
* CancelAppointment method.



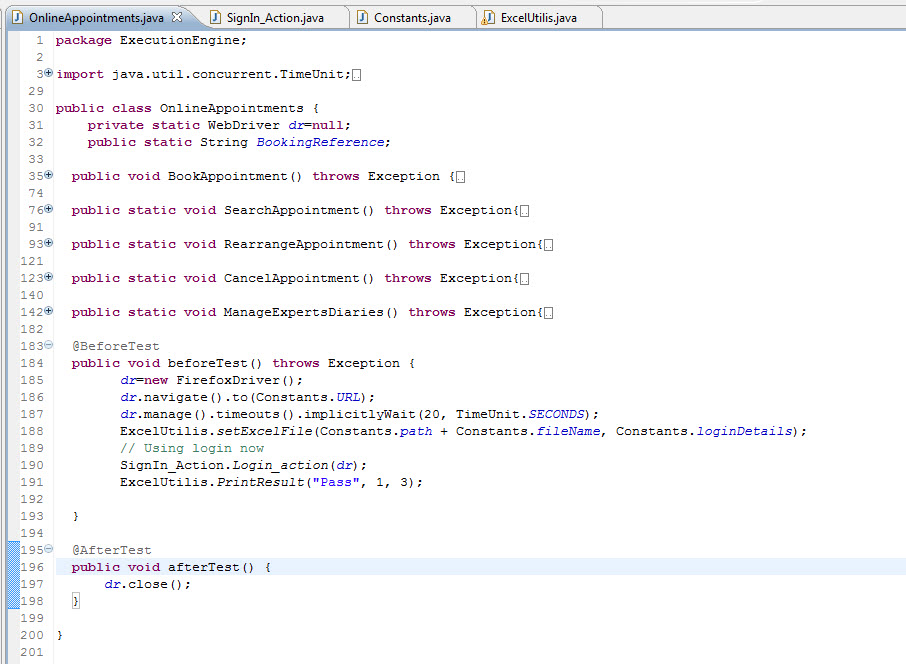
* ManageExpertsDiaries method.



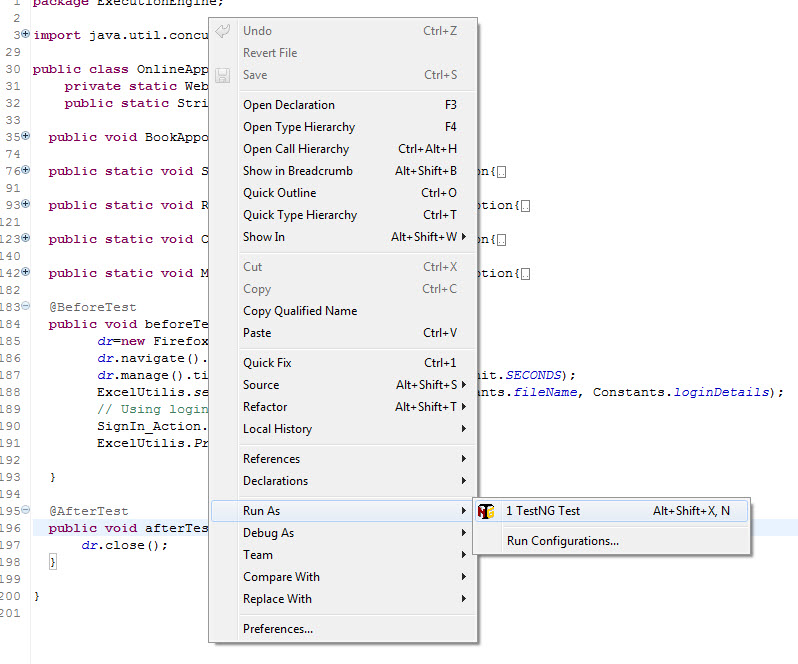
* Your Project explorer window will look like this now.



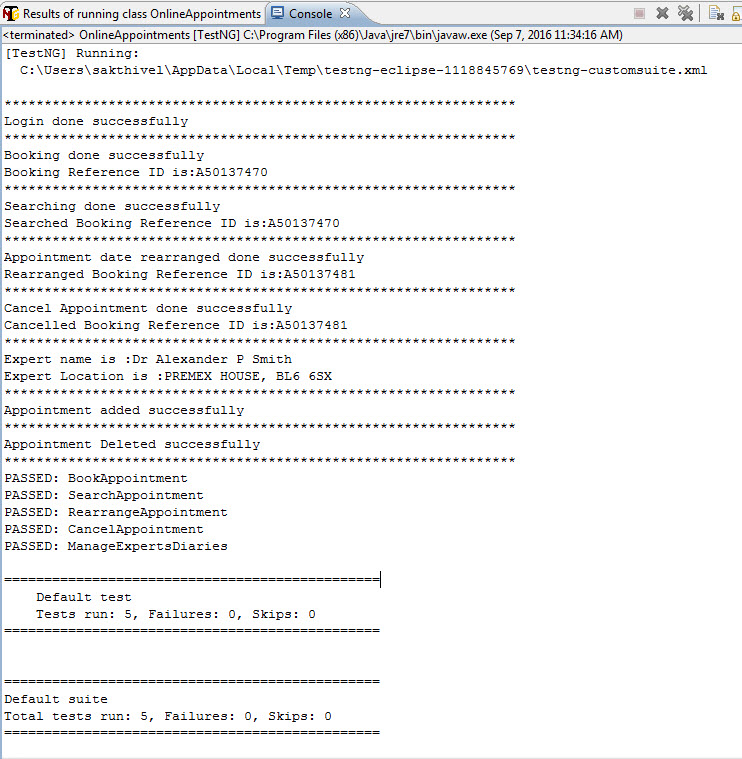
* **OnlineAppointments class file** will look like this now.



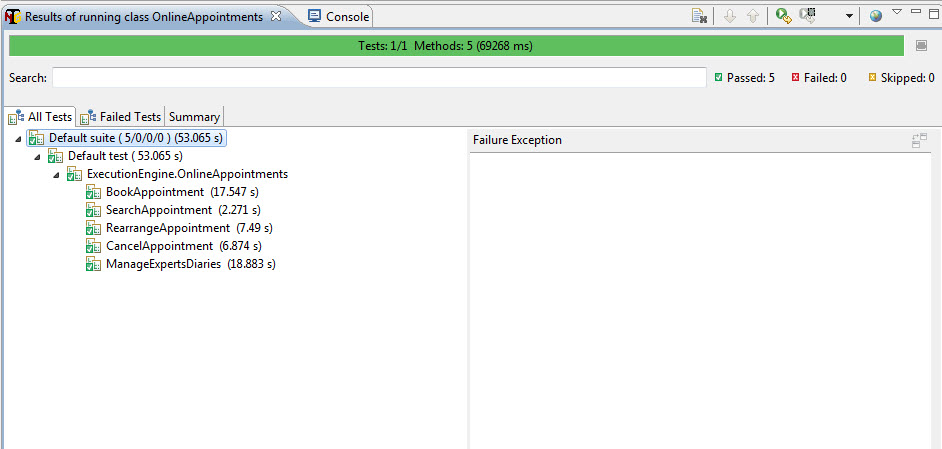
* Run the test by right click on the test case script and select **Run As**>**TestNG Test**.



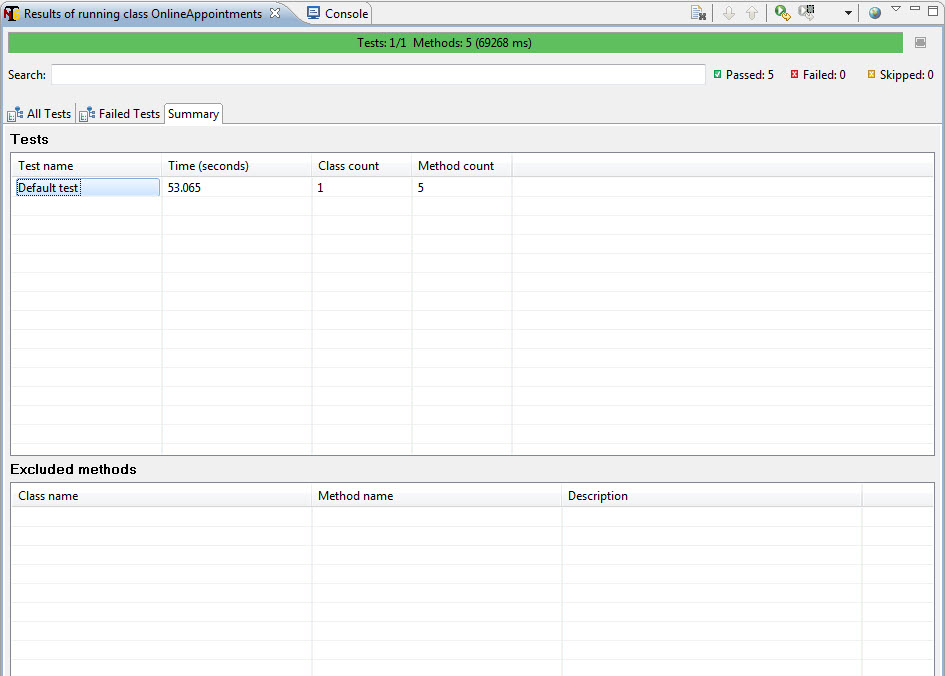
* Give it few minutes to complete the execution, once it is finished the results will look like this in the **Console** window.



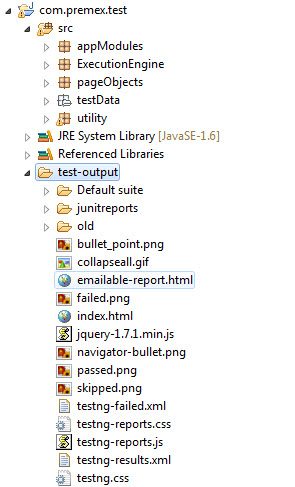
* Click on the **Results of TestNG** tab. It will display the total passed, failed and skipped test with time taken during the execution.

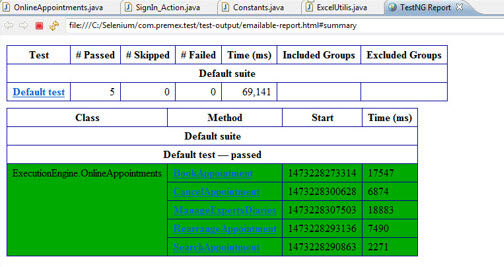


* It displayed ‘Passed: 5’. This means test is successful and Passed.
* There are 3 sub tabs. “All Tests”, “Failed Tests” and “Summary”. Just click “Summary” to see what is there.

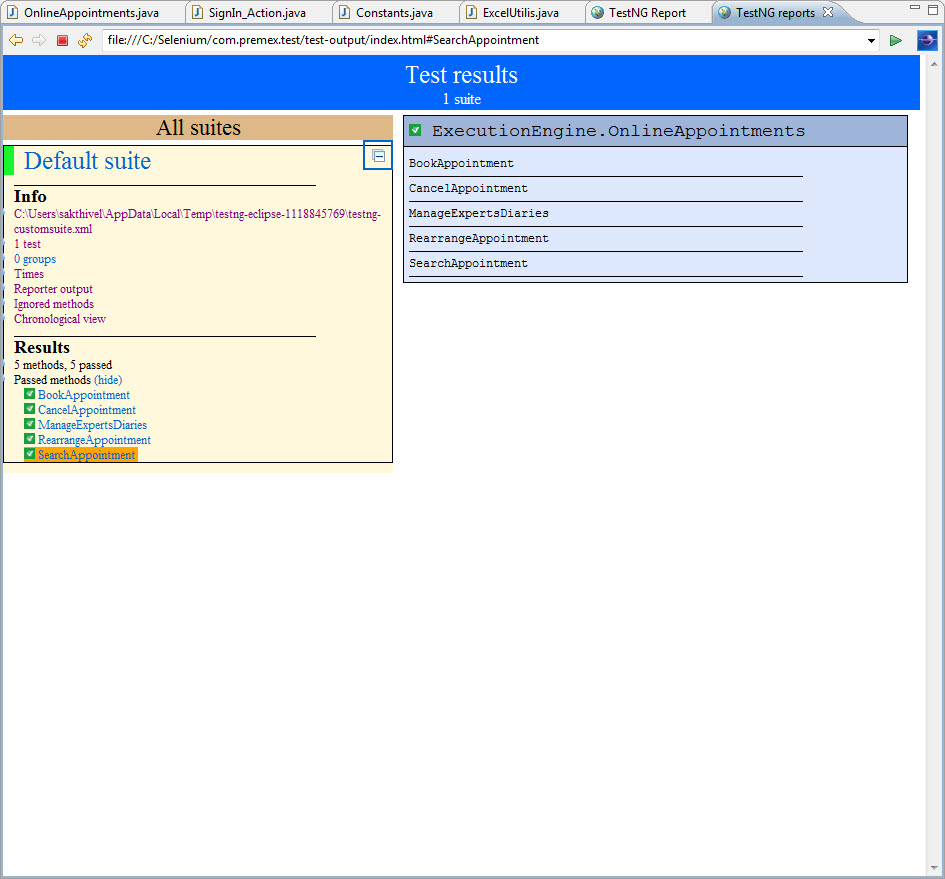


* TestNG also produce HTML reports. To access those reports go to your **Project** folder and open **test-output** folder.
* Open ‘**emailable-report.html’**, as this is a html report open it with browser.





* TestNG also produce ‘**index.html’** report and it resides in the same **test-output** folder.
* This reports gives the link to all the different component of the TestNG reports like **Groups**&**Reporter Output**.
* On clicking these will display detailed descriptions of execution. In the advance chapter of TestNG we will go through each of the TestNG topics.



**Time involved in Manual and Automation:**

In order to execute the mentioned scenario manually it took around 1 hour. While to create the same scenario as a script in selenium it took 3 days to complete the same. This script created can be reused to test the same scenario “n” number of times.

**Process time to change manual and automated script**

As as far manual testing is concerned we will not require any extra time. But in case of automation as we have used the framework model we would only require to change the page objects. Once this is done, the script created can be reused to execute the scenario. The time required to change the page objects will only be minimal considering the sample scenario.

**Code changes to the application and the Skills required**

In case of code change to the application, we will require the entire page objects to be modified. The one who created scripts will be able to do the required changes.

Developer/ Tester should require the knowledge on core JAVA and basic Oops concepts to make modifications to the script developed. It depends on the individual capabilities.

**Other tests we can do with this tool?**

Selenium is a functional testing tool. We can use the tool for regression and smoke testing as well. This is a web automation tool mainly used for browser automation.

**Advantage on VU by introducing this automation tool.**

As VU is a web based application, and the following are the advantages of selenium tool.

* Selenium is an open source web automation tool.
* Selenium have Cross browser compatibility (Firefox, IE, Chrome, and Safari).
* Selenium WebDriver helps in creating automated test scripts and reduces the execution time, increase the efficiency and removes manual intervention.
* Selenium WebDriver makes direct calls to the browser using each browser’s native support for automation.

New point has to be introduced if this is not mentioned in this document.

1. Is there a way to store the test script on version control?

We can use GIT and BIT Buckets to store the test scripts based on versions.

1. Any database involved? Does Selenium come with this Database?

No, it is a JAR file configured with IDE like Eclipse for web application automation.

1. What are the system specifications? If there is nothing then we need to mention it.

Selenium has great Platform compatibility (Windows, Mac OS, and Linux).

1. Multiple test users allowed?

No, selenium does not support multiple test users for load testing as it is a functional testing tool.

1. Any restrictions and negative about this tool?

Selenium supports testing of only web based applications.

Reports can be generated using a third party tool like TestNG.