8. Perform All Test Annotations

1. Creating a simple Java project

- Open Eclipse
- Go to the **File** menu. Choose **New->Maven Project**
- Enter the project name as **hello-selenium-testng**. Click on **Next**
- This will create the project files in the Project Explorer

2. Installing TestNG

- TestNG is installed in our eclipse

3. Adding TestNG dependencies in the pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>hello-selenium-testng</groupId>
  <artifactId>hello-selenium-testng</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <dependencies>
                <!--

https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
                <dependency>
                        <groupId>org.seleniumhq.selenium</groupId>
                        <artifactId>selenium-java</artifactId>
                        <version>4.10.0</version>
                </dependency>

                <!--

https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-chrome-driv
er -->
                <dependency>
                        <groupId>org.seleniumhq.selenium</groupId>
                        <artifactId>selenium-chrome-driver</artifactId>
                        <version>4.10.0</version>
                </dependency>

                <!--
```

```
https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-firefox-driver -->
        <dependency>
            <groupId>org.seleniumhq.selenium</groupId>
            <artifactId>selenium-firefox-driver</artifactId>
            <version>4.10.0</version>
        </dependency>


        <!--Testng-->
        <dependency>
            <groupId>org.testng</groupId>
            <artifactId>testng</artifactId>
            <version>7.8.0</version>
            <scope>compile</scope>
        </dependency>

    </dependencies>


    <build>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                </configuration>
            </plugin>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-surefire-plugin</artifactId>
                <configuration>
                    <suiteXmlFiles>

<suiteXmlFile>testng.xml</suiteXmlFile>
                    </suiteXmlFiles>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

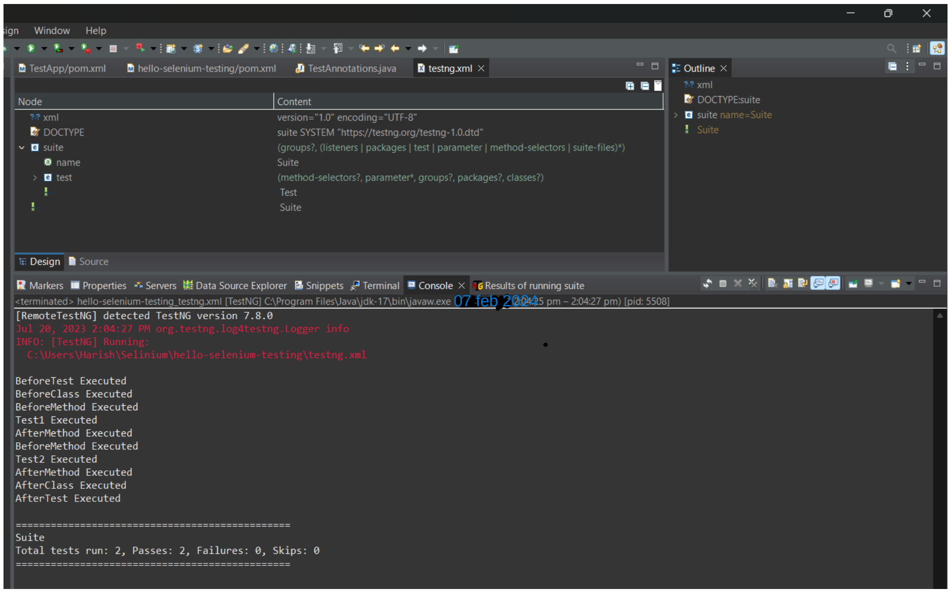
4. Creating a class file named TestAnnotations

- In **Class Name,** enter **TestAnnotations**. In **Package Name,** enter **com.ecommerce.test** and click on **Finish**
- Enter the following code:

```
package com.ecommerce.test;

import org.testng.annotations.AfterClass;
import org.testng.annotations.Test;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.AfterClass;
import org.testng.annotations.Test;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.*;

public class TestAnnotations {

        @Test
        public void Test1() {
                System.out.println("Test1 Executed");
        }
        @Test
        public void Test2() {
                System.out.println("Test2 Executed");
        }

        @BeforeTest
        public void beforeTest() {
                System.out.println("BeforeTest Executed");
        }
        @AfterTest
        public void AfterTest() {
                System.out.println("AfterTest Executed");
        }

        @BeforeMethod
        public void beforeMethod() {
                System.out.println("BeforeMethod Executed");
        }
        @AfterMethod
        public void afterMethod() {
                System.out.println("AfterMethod Executed");
        }

        @BeforeClass
        public void beforeClass() {
                System.out.println("BeforeClass Executed");
        }
        @AfterClass
        public void afterClass() {
                System.out.println("AfterClass Executed");
        }

}
```

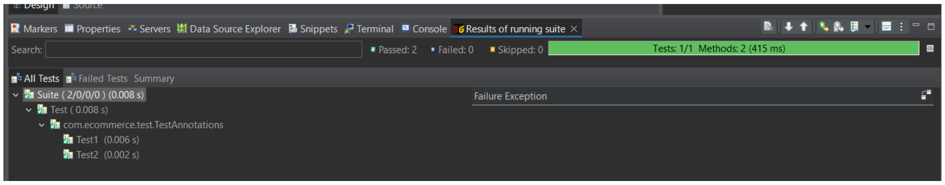
5. Running the project as TestNG

- Right click on **TestAnnotations** class. Click on **TestNG->Convert to TestNG**
- Click on **Finish.** It will create a **TestNG.xml** file. Open that file
- Right click. Select **Run As ->TestNG Suite**



Node	Content
xml	version="1.0" encoding="UTF-8"
DOCTYPE	suite SYSTEM "https://testng.org/testng-1.0.dtd"
suite	(groups?, (listeners | packages | test | parameter | method-selectors | suite-files)*)
name	Suite
test	(method-selectors?, parameter*, groups?, packages?, classes?)
	Test
	Suite

```
<terminated> hello-selenium-testing_testng.xml [TestNG] C:\Program Files\Java\jdk-17\bin\javaw.exe 07 feb 2024 2:04:25 pm – 2:04:27 pm) [pid: 5508]
[RemoteTestNG] detected TestNG version 7.8.0
Jul 20, 2023 2:04:27 PM org.testng.log4testng.Logger info
INFO: [TestNG] Running:
  C:\Users\Harish\Selinium\hello-selenium-testing\testng.xml

BeforeTest Executed
BeforeClass Executed
BeforeMethod Executed
Test1 Executed
AfterMethod Executed
BeforeMethod Executed
Test2 Executed
AfterMethod Executed
AfterClass Executed
AfterTest Executed

===============================================
Suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```



Passed: 2 Failed: 0 Skipped: 0 Tests: 1/1 Methods: 2 (415 ms)

```
Suite ( 2/0/0/0 ) (0.008 s)
  Test ( 0.008 s)
    com.ecommerce.test.TestAnnotations
      Test1  (0.006 s)
      Test2  (0.002 s)
```

Pushing the code to your GitHub repositories : -

- Open your folder where the Project . And then click the right button to open the git bash command prompt.
- Before that, open the github and create a new repository.
- Initialize your repository using the following command:

```
git init
```

- Add all the files to your git repository using the following command:

```
git add .
```

- To check the status of the repository use the below command:

```
git status
```

- Commit the changes using the following command:

```
git commit .  -m “Changes have been committed.”
```

- To add the files to the repository use the (URL) from the github and use the command;

```
git remote add origin <url>
```

- Push the files to the folder you initially created using the following command:

```
git push origin master.
```