**Assertions in Selenium using TestNG:**

Assertions are used to perform various kinds of validations in the tests and help us to decide whether the test has passed or failed. This helps an automation tester to validate the actual result with expected result.

**Special Condition for Asserts**: Asserts must be inside @Test method only, if we write assert statement inside a normal method or inside the main method then TestNG will not consider the failure or pass.

There are two types of assertions in Selenium that we can place in our test scripts using TestNG:

* Hard Assert
* Soft Assert

**Packages to be imported:**

import org.testng.Assert;

import org.testng.asserts.Assertion;

import org.testng.asserts.SoftAssert;

Assert is a class provided by TestNG to work with Selenium. Asserts are used to validate the major checkpoints of the test case. Assert class provides certain methods which help in validating these checkpoints.

**Create object / instance of Hard / Soft assert:**

Assertion hardAssert = new Assertion();

SoftAssert softAssert = new SoftAssert();

**Hard Assertions:**

* A Hard Assertion throws an Assertion Error immediately when an assert statement fails and continues with the next @Test method in the test suite.
* Our test method script will stop executing when the assertion fails.
* The remaining statements inside @ test method will not get executed.
* To handle the above Assertion error which is thrown, we need to catch like the Java try catch block.
* Now, after the suit is completed with the execution, particular test will be marked as passed instead of FAIL.

**Common Hard Assert methods:**

1. Assert.assertEquals(actual,expected,Message);
2. Assert.assertNotEquals(actual,expected,Message);
3. Assert.assertTrue(condition);
4. Assert.assertFalse(condition);
5. Assert.assertNull(object);
6. Assert.assertNotNull(object);
7. Assert.fail(“Message”)
8. Assert.pass(“Pass”);

We can compare various items using assertEquals method in [TestNG with selenium](https://chercher.tech/java/install-associate-testng-junit-drawbacks-selenium-webdriver" \t "_blank). Below are few overloaded methods present in TestNG

* *byte[] : actual method is****assertEquals(byte[] actual, byte[] expected)***
* *short[] -> Compares two short arrays*
* *int[] -> Compares two in arrays*
* *boolean[] -> Compares two boolean arrays*
* *char[] -> Compares two char arrays*
* *float[] -> Compares two float arrays*
* *double[] -> Compares two double arrays*
* *long[] -> Compares two long arrays*
* *Object -> Compares two objects*
* *String -> Compares two Strings*
* *double Delta -> explained below.*
* *float Delta*
* *long ->Compares two long values*
* *boolean -> Compares two boolean values*
* *byte -> compares two byte values*
* *char -> Compares two char values*
* *short -> Compares two short values*
* *int -> Compares two int values*
* *Collection<?> -> Compares two Collections*
* *Iterator<?> -> Compares two iterators*
* *Iterable<?> -> Compares two Iterable*
* *Set<?> -> Compares two sets*
* *Map<?, ?> -> compares two maps*

@Test  
**public** **void** test1(){  
**try**{  
Assert.assertTrue(2<1);  
System.out.println(“Assertion Failed in Test 1”);  
Assert.assertFalse(1>0);  
System.out.println(“Assertion Failed in Test 1”);  
Assert.assertEquals(“Sample”, “Sample”);  
System.out.println(“Assertion Passed in Test 1”);  
} **catch**(AssertionError ae){  
ae.printStackTrace();  
}  
}

@Test  
**public** **void** test2(){  
**try**{  
Assert.assertTrue(2>1);  
System.out.println(“Assertion passed in Test 2”);  
Assert.assertFalse(1<0);  
System.out.println(“Assertion passed in Test 2”);  
Assert.assertEquals(“Sample”, “Sample”);  
System.out.println(“Assertion Passed in Test 2”);  
} **catch**(AssertionError ae){  
ae.printStackTrace();  
}  
}  
}

It is not mandatory to continue the execution of a test case after the assertion failure, but sometimes you might be checking a minor value. So just because minor verification failed you may not want to stop the execution of the test case.

**Soft Assertions:**

* If we need to execute the remaining test case statements even after an assertion fails, and we also want to report assertion and test case method failure in TestNG report, TestNG provides soft assertions for this.
* Soft Assertions does not throw an AssertionError when an assertion fails and would continue with the next statement after the assert statement.
* Soft Assert validates all assert and at the end it checks whether all asserts are passed or not.
* This is usually used when our test requires many assertions to execute and the user wants to execute all the assertions / code before failing / skipping the tests.

Anyways once the soft assertion is failed the TestNG throws AssertionError, instead of throwing an error immediately TestNG throws the error just before completing the test case.  
  
Example: Consider you have a test case which verifies the 100 CSS properties. Do you want to stop the Execution just because first CSS property is not matching? NO.  
  
What we would like is at least we want to execute the test case for all the check points/ verification so that we will get to know how many properties are matching and how many are not matching.

**The Architecture of SoftAssert :** All the assertion methods are present in Assertion class, the SoftAssert class extends the Assertion class. So through inheritance SoftAssert Class allows the user to access all the methods present in the Assertion class.

**Soft Assert methods:**

softAssert.assertTrue(false);

softAssert.assertEquals(Actual, Expected, “Message”);

softAssert.assertAll();

If any soft assert fails, script will continue unlike hard assert. Use assertAll() method as a last statement in the @Test, which will collate the result of all the soft assertions and if any of them is failed then that test method is marked as failed. But it will continue with next @Test as it is in test suite to ensure complete test execution.

If you forget to call assertAll() at the end of your test, the test method will be marked as passed even if any assert objects threw error, So don’t forget to add assertAll().

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

**import** org.testng.asserts.Assertion;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.testng.Assert;

**import** org.testng.asserts.SoftAssert;

**public** **class** TestinAssertions {

**public** **static** String *sSampleval* = "Soft\_Hard\_Assert";

@Test

**public** **void** hardAssertionTest(){

//Creating hardAssert object

Assertion hardAssert = **new** Assertion();

//Assertion failing

hardAssert.fail("Failing first assertion");

System.***out***.println("Failing 1");

//Assertion failing

hardAssert.fail("Failing second assertion");

System.***out***.println("Failing 2");

//Assertion passing

hardAssert.assertEquals(1, 1, "Passing third assertion");

System.***out***.println("Passing 3");

}

@Test

**public** **void** hardAssertMethod() {

Assertion hardAssert = **new** Assertion();

hardAssert.assertTrue(**true** == **true**,"First Hard assert passed");

hardAssert.assertEquals("Hard\_Assert", "Soft\_Hard\_Assert","second hard assert failed");

hardAssert.assertEquals(*sSampleval*, "Soft\_Hard\_Assert");

/\*

\* Assert.assertTrue(false);

\* Creating object for hardassert is optional

\*/

System.***out***.println("hardAssertMethod Successfully passed!");

}

@Test

**public** **void** softAssertionMethod() {

//Create an object of SoftAssert

SoftAssert softAssert = **new** SoftAssert();

softAssert.assertTrue(**true** == **true**);

softAssert.assertEquals("Soft\_Hard\_Assert", "Soft\_Hard\_Assert");

softAssert.assertEquals(*sSampleval*, "Soft\_Hard\_Assert");

System.***out***.println("softAssertionMethod Successfully passed!");

softAssert.assertAll();

}

@Test

**public** **void** test1(){

**try**{

Assert.*assertTrue*(2<1);

System.***out***.println("Assertion Failed in Test 1");

Assert.*assertFalse*(1>0);

System.***out***.println("Assertion Failed in Test 1");

Assert.*assertEquals*("Sample", "Sample");

System.***out***.println("Assertion Passed in Test 1");

} **catch**(AssertionError ae){

ae.printStackTrace();

}

}

@Test

**public** **void** test2(){

**try**{

Assert.*assertTrue*(2>1);

System.***out***.println("Assertion passed in Test 2");

Assert.*assertFalse*(1<0);

System.***out***.println("Assertion passed in Test 2");

Assert.*assertEquals*("Sample", "Sample");

System.***out***.println("Assertion Passed in Test 2");

} **catch**(AssertionError ae){

ae.printStackTrace();

}

}

@Test

**public** **void** softAssertionTest(){

//Creating softAssert object

SoftAssert softAssert = **new** SoftAssert();

//Assertion failing

softAssert.fail("Failing first assertion");

System.***out***.println("Failing 1");

//Assertion failing

softAssert.fail("Failing second assertion");

System.***out***.println("Failing 2");

//Assertion passing

softAssert.assertEquals(1, 1, "Passing third assertion");

System.***out***.println("Passing 3");

//Collates the assertion results and marks test as pass or fail

softAssert.assertAll();

}

@Test

**public** **void** verifyTitle() {

// To invoke browser

System.*setProperty*("webdriver.chrome.driver", "G:\\Selenium RS 2019\\chromedriver\_win32\\chromedriver.exe");

// Create object for browser driver

WebDriver driver = **new** ChromeDriver();

// To print the page title value in the console

driver.get("http://total-qa.com/");

String expected = "Total-QA - Future of Software Testing";

String actual = driver.getTitle();

Assert.*assertEquals*(actual, expected);

driver.close();

}

}

**Points to remember:**

1. We should instantiate a SoftAssert object within a @Test method. Scope of SoftAssert should only be within the Test method.

2. We should never use the same Soft Assertions objects with multiple test cases.

If the same object is used in different test casemethods, then when the control executes assertAll () method of the second test case, it will evaluate all the above assertions (even if they are in different test cases) at the same time. As a result, all the test cases using this same object of Soft Assert class will fail.

***We can use multiple hard assert and soft assert inside the same @Test method.***

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name=*"Suite"*>

<test name=*"Test"*>

<classes>

<class name=*"default package.TestinAssertions"*/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

**Reference links:**

Same Assert object problems:

<https://www.seleniumeasy.com/testng-tutorials/soft-asserts-in-testng-example>

<https://www.softwaretestingmaterial.com/soft-assert/>

<https://blogs.perficient.com/2016/01/13/hard-and-soft-assertions-in-selenium/>

<http://total-qa.com/soft-asserts-hard-asserts-testng/>

<http://www.appliedselenium.com/2019/03/soft-assert-in-selenium/>

<https://www.swtestacademy.com/soft-assertions-in-testng/>

[https://www.tothenew.com/blog/soft-assertions-in-selenium-using-testng/#](https://www.tothenew.com/blog/soft-assertions-in-selenium-using-testng/)

<https://www.softwaretestinghelp.com/assertions-in-selenium/>

<https://www.tothenew.com/blog/soft-assertions-in-selenium-using-testng/>

Custom Soft Assertions

<https://chercher.tech/java/testng-assertions-selenium-webdriver>

Custom collection assert– by Creating a collection object

<https://automatorsworld.com/2018/10/11/hard-soft-assert-custom-assert-selenium/>

<https://artoftesting.com/soft-assertion-in-testng-selenium-webdriver-java>

File download settings in browser and verify after download:

<https://www.seleniumeasy.com/selenium-tutorials/verify-file-after-downloading-using-webdriver-java>

<https://www.swtestacademy.com/download-file-in-selenium/>