Selenium with TestNG
Lecture 1

What is TestNG:

TestNG is a powerful, open-source testing framework for Java, inspired by JUnit and NUnit, that allows you to automate tests for various scenarios, including unit, integration, and end-to-end testing, with features like annotations, grouping, and data-driven testing.

Key Features:

- **Annotations**: TestNG uses annotations (like @Test, @BeforeSuite, @AfterSuite, etc.) to define test methods, set up preconditions, and manage test dependencies.
- **Grouping**: Allows you to group test methods into logical groups, enabling selective execution of tests based on these groups.
- **Data-Driven Testing**: Enables you to run the same test with different data inputs, making it easier to test various scenarios.
- **Parallel Execution**: Supports running tests in parallel, significantly reducing overall test execution time.
- **Reporting**: Provides detailed HTML-based test reports, making it easier to analyze test results.

How to install TestNG in Eclipse:

1. Open Eclipse and Access the Marketplace:
   - Launch Eclipse IDE and Go to Help > Eclipse Marketplace.

2. Search for and Install TestNG:
   - In the Marketplace window, search for "TestNG" in the search bar.
   - Click on "Install" for the "TestNG for Eclipse" result.

3. Accept the License and Finish the Installation:
   - Make sure all the checkboxes for TestNG are checked.
   - Click on the "Confirm" button.
   - Accept the license agreement and click "Finish".

4. Restart Eclipse:
   - Restart Eclipse for the TestNG plugin to take effect.

What are the TestNG annotations:

TestNG annotations are special markers in Java code that provide instructions to the TestNG framework on how to execute test methods, defining the life cycle of tests and their execution sequence, such as before or after a test, class, or suite.

Core Annotations:

- **@Test**: Marks a method as a test method, indicating that TestNG should execute it as part of the test suite.
- **@BeforeSuite**: Executes the annotated method before any tests in the suite are run.
- **@AfterSuite**: Executes the annotated method after all tests in the suite have been executed.
- **@BeforeTest:** Executes the annotated method before any tests in the current test are run.
- **@AfterTest:** Executes the annotated method after all tests in the current test have been executed.
- **@BeforeClass:** Executes the annotated method before the first test method in the current class is invoked.
- **@AfterClass:** Executes the annotated method after all test methods in the current class have been run.
- **@BeforeMethod:** Executes the annotated method before each test method is invoked.
- **@AfterMethod:** Executes the annotated method after each test method has been executed.

How to write testcases in TestNG:

```
package UI;
import org.testng.annotations.Test;

public class LoginTest {
	@Test (priority = 1, description = "This is login Test")

	public void bloginTest() {
		System.out.println("Login is Sucessful");
	}

	@Test(priority = 2, description = "This is logout Test")

	public void alogoutTest() {
		System.out.println("Logout is Sucessful");
	}
}
```

BeforeMethod Vs BeforeTest Vs BeforeClass Vs BeforeSuite

Before Test:

```
package UI;
import org.testng.annotations.AfterTest;
import org.testng.annotations.BeforeTest;
import org.testng.annotations.Test;

public class BeforeMethodBeforeTest {

	@BeforeTest
	public void loginToApplication() {
		System.out.println("Login to Application");
	}

	@AfterTest
	public void logoutFromApplication() {
		System.out.println("Logout from Application");
	}

	@Test (priority = 1)

	public void bloginTest() {
		System.out.println("Test1");
	}

	@Test(priority = 2)

	public void alogoutTest() {
		System.out.println("Test2");
	}
}
```

Before Method:

```
package UI;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.AfterTest;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.BeforeTest;
import org.testng.annotations.Test;

public class BeforeTestDemo {

        @BeforeTest
        public void loginToApplication() {
                System.out.println("Login to Application");
        }

        @AfterTest
        public void logoutFromApplication() {
                System.out.println("Logout from Application");
        }

        @BeforeMethod
        public void connectToDB() {
                System.out.println("DB Connected");
        }

        @AfterMethod
        public void disConnectFromDB() {
                System.out.println("DB DisConnected");
        }


        @Test (priority = 1)

        public void bloginTest() {
                System.out.println("Test1");
        }

        @Test(priority = 2)

        public void alogoutTest() {
                System.out.println("Test2");
        }

}
```

BeforeClass:

```
package UI;
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;

public class GroupsDemoTest {

        @BeforeClass
        public void beforeClass() {
                System.out.println("Run This before class");
        }
        @AfterClass
        public void afterClass() {
                System.out.println("Run This after class");
        }

        @Test (priority =1,groups="regression")


        public void Test1() {
                System.out.println("Test1");
        }

        @Test (priority =2,groups="regression")
        public void Test2() {
                System.out.println("Test2");
        }

        @Test (groups={"regression","sanity"})
        public void Test3() {
                System.out.println("Test3");
        }

        @Test (groups="sanity")
        public void Test4() {
                System.out.println("Test4");
        }
}
```

BeforeSuite:

```
package Common;


import org.testng.annotations.AfterSuite;

import org.testng.annotations.BeforeSuite;
```

```
public class CommonDataSetup {

        @BeforeSuite
        public void dataSetup() {
                System.out.println("Common data setup");
        }

        @AfterSuite
        public void dataTeardown() {
                System.out.println("Common data cleanup");
        }

}

package UI;
import org.testng.annotations.Test;
import Common.CommonDataSetup;

public class GroupsDemoTest2 extends CommonDataSetup{

        @Test (priority =1,groups="regression")

        public void Test1() {
                System.out.println("Test1");
        }

        @Test (priority =2,groups="regression")

        public void Test2() {
                System.out.println("Test2");
        }

        @Test (groups={"regression","sanity"})
        public void Test3() {
                System.out.println("Test3");
        }

        @Test (groups="sanity")
        public void Test4() {
                System.out.println("Test4");
        }
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite Setup and Teardown">
 <test name="Test Before Suite After Suite">
   <classes>
     <class name="UI.GroupsDemoTest2"/>
   </classes>
 </test> <!-- Test -->
</suite> <!-- Suite -->
```

How to use assertion in Selenium

```
package UI;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.Assert;
import org.testng.annotations.Test;

public class VerifyTitleTest {
        @Test

        public void titleTest()
        {
                String expectedTitle = "Google";

                WebDriver driver = new ChromeDriver();
                driver.get("https://www.google.com/");
                driver.manage().window().maximize();

                String actualTitle = driver.getTitle();
                System.out.println(actualTitle);
                Assert.assertEquals(actualTitle,expectedTitle);

        }
}
```

What are TestNG Groups

TestNG provides a compelling feature known as TestNG groups. It is a way to organize and categorize test methods, which allows running a specific set of test cases based on the tester’s needs.

Typically, a large automation suite contains all kinds of test cases such as smoke, sanity and regression. Testers can assign a group to all these test cases at the time of creation and using the TestNG xml file they can execute the required group of test cases using TestNG group feature.

```
package UI;
import org.testng.annotations.Test;

public class GroupsDemoTest {
        @Test (priority =1,groups="regression")

        public void Test1() {
                System.out.println("Test1");
        }

        @Test (priority =2,groups="regression")

        public void Test2() {
                System.out.println("Test2");
        }

        @Test (groups={"regression","sanity"})

        public void Test3() {
                System.out.println("Test3");
        }
```

```
    @Test (groups="sanity")

    public void Test4() {
        System.out.println("Test4");
    }
}
```