TESTNG - EXECUTING TESTS

The test cases are executed using **TestNG** class. This class is the main entry point for running tests in the TestNG framework. Users can create their own TestNG object and invoke it in many different ways such as:

- On an existing testng.xml.
- On a synthetic testng.xml, created entirely from Java.
- By directly setting the test classes.

You can also define which groups to include or exclude, assign parameters, etc. The command line parameters are:

- -d outputdir: specify the output directory.
- -testclass class_name: specifies one or several class names.
- -testjar jar_name: specifies the jar containing the tests.
- -sourcedir src1;src2: ; separated list of source directories *usedonlywhenjavadocannotationsareused*.
- -target
- -groups
- -testrunfactory
- -listener

We will create the TestNG object an existing testng.xml in our example below.

Create a Class

- Create a java class to be tested, say, MessageUtil.java in **C:\>TestNG_WORKSPACE**.

```
/*
* This class prints the given message on console.
*/

public class MessageUtil {

   private String message;

   //Constructor
   //@param message to be printed
   public MessageUtil(String message){
      this.message = message;
   }

   // prints the message
   public String printMessage(){
      System.out.println(message);
      return message;
   }
}
```

Create Test Case Class

- Create a java test class, say, SampleTest.java.
- Add a test method testPrintMessage to your test class.

- Add an Annotation @Test to method testPrintMessage.
- Implement the test condition and check the condition using assertEquals API of TestNG.

Create a java class file called SampleTest.java in **C:\>TestNG_WORKSPACE**.

```
import org.testng.Assert;
import org.testng.annotations.Test;

public class SampleTest {

   String message = "Hello World";
   MessageUtil messageUtil = new MessageUtil(message);

   @Test
   public void testPrintMessage() {
      Assert.assertEquals(message, messageUtil.printMessage());
   }
}
```

Create testng.xml

Next, let's create testng.xml file in **C:\>TestNG_WORKSPACE**, to execute test case*s*. This file captures your entire testing in XML. This file makes it easy to describe all your test suites and their parameters in one file, which you can check in your code repository or e-mail to coworkers. It also makes it easy to extract subsets of your tests or split several runtime configurations *e. g. , testng − database. xmlwouldrunonlyteststhatexerciseyourdatabase*.

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

<suite name="Sample test Suite">
   <test name="Sample test">
    <classes>
      <class name="SampleTest" />
    </classes>
  </test>
</suite>
```

Compile the test case using javac.

```
C:\TestNG_WORKSPACE>javac MessageUtil.java SampleTest.java
```

Now, run the testng.xml, which will run the test case defined in <test> tag.

```
C:\TestNG_WORKSPACE>java -cp "C:\TestNG_WORKSPACE" org.testng.TestNG testng.xml
```

Verify the output.

```
Hello World

===============================================
Sample test Suite
Total tests run: 1, Failures: 0, Skips: 0
===============================================
```