JUnit - Ignore Test

Sometimes it so happens that our code is not completely ready while running a test case. As a result, the test case fails. The **@Ignore** annotation helps in this scenario.

> A test method annotated with @Ignore will not be executed.
>
> If a test class is annotated with @Ignore, then none of its test methods will be executed.

Now let's see @Ignore in action.

Create a Class

Create a java class to be tested, say, **MessageUtil.java** in C:\>JUNIT_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;
   }

   // add "Hi!" to the message
   public String salutationMessage(){
      message = "Hi!" + message;
      System.out.println(message);
      return message;
   }

}
```

Create Test Case Class

- Create a java test class, say, TestJunit.java.
- Add a test method testPrintMessage() or testSalutationMessage() to your test class.
- Add an Annotaion @Ignore to method testPrintMessage().

Create a java class file named **TestJunit.java** in C:\ JUNIT_WORKSPACE.

```
import org.junit.Test;
import org.junit.Ignore;
import static org.junit.Assert.assertEquals;

public class TestJunit {

   String message = "Robert";
   MessageUtil messageUtil = new MessageUtil(message);

   @Ignore
   @Test
   public void testPrintMessage() {
      System.out.println("Inside testPrintMessage()");
      message = "Robert";
      assertEquals(message,messageUtil.printMessage());
   }

   @Test
   public void testSalutationMessage() {
      System.out.println("Inside testSalutationMessage()");
      message = "Hi!" + "Robert";
      assertEquals(message,messageUtil.salutationMessage());
   }

}
```

Create Test Runner Class

Create a java class file named **TestRunner.java** in C:\>JUNIT_WORKSPACE to execute test case(s).

```
import org.junit.runner.JUnitCore;
import org.junit.runner.Result;
import org.junit.runner.notification.Failure;

public class TestRunner {
   public static void main(String[] args) {
      Result result = JUnitCore.runClasses(TestJunit.class);

      for (Failure failure : result.getFailures()) {
         System.out.println(failure.toString());
      }

      System.out.println(result.wasSuccessful());
   }
}
```

Compile the MessageUtil, Test case and Test Runner classes using javac.

```
C:\JUNIT_WORKSPACE>javac MessageUtil.java TestJunit.java TestRunner.java
```

Now run the Test Runner, which will not run the testPrintMessage() test case defined in the provided Test Case class.

```
C:\JUNIT_WORKSPACE>java TestRunner
```

Verify the output. testPrintMessage() test case is not tested.

```
Inside testSalutationMessage()
Hi!Robert
true
```

Now, update TestJunit in C:\>JUNIT_WORKSPACE to ignore all test cases. Add @Ignore at class level.

```
import org.junit.Test;
import org.junit.Ignore;
import static org.junit.Assert.assertEquals;

@Ignore
public class TestJunit {

   String message = "Robert";
   MessageUtil messageUtil = new MessageUtil(message);

   @Test
   public void testPrintMessage() {
      System.out.println("Inside testPrintMessage()");
      message = "Robert";
      assertEquals(message,messageUtil.printMessage());
   }

   @Test
   public void testSalutationMessage() {
      System.out.println("Inside testSalutationMessage()");
      message = "Hi!" + "Robert";
      assertEquals(message,messageUtil.salutationMessage());
   }

}
```

Compile the test case using javac.

```
C:\JUNIT_WORKSPACE>javac TestJunit.java
```

Keep your Test Runner unchanged as follows −

```
import org.junit.runner.JUnitCore;
import org.junit.runner.Result;
import org.junit.runner.notification.Failure;

public class TestRunner {
   public static void main(String[] args) {
      Result result = JUnitCore.runClasses(TestJunit.class);

      for (Failure failure : result.getFailures()) {
         System.out.println(failure.toString());
      }
      System.out.println(result.wasSuccessful());
   }
}
```

Now run the Test Runner, which will not run any test case defined in the provided Test Case class.

```
C:\JUNIT_WORKSPACE>java TestRunner
```

Verify the output. No test case is tested.

```
true
```