Create and run JUnit 5 Test Case

This lessons focuses on creating and running JUnit 5 test cases in Maven.

We'll cover the following

- Create a Java class
- Create method inside Calculator class
- Create a Java Test class
- Add a @Test method

Create a Java class

After adding the required dependencies, let's create a Java class. It will be our class under test.

Step 1 - Expand the ***junit5-maven-starter*** project. In order to create a new Java class, right-click on **src/main/java** traverse to New --> Class. Click on Class to add a new Java class.

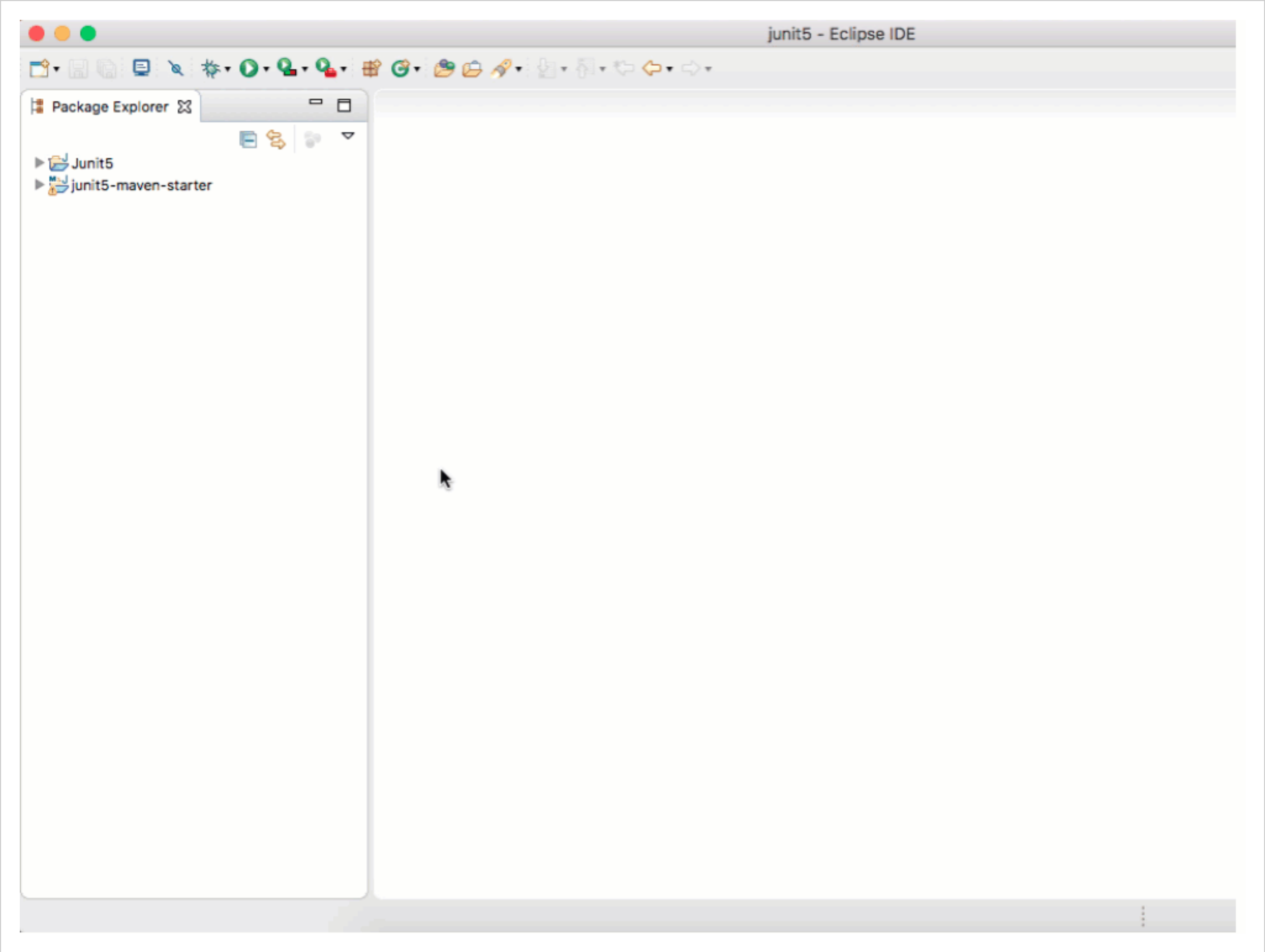
Step 2 - New Java Class popup window will be opened.

Step 3 - Provide **Package** name of your choice.

Step 4 - Enter Name of class as, **Calculator**.

Step 5 - Click Finish.

A Java class by name **Calculator.java** will be created in Eclipse IDE. See the demonstration of steps below.



Create method inside Calculator class

Step 1 - Create a method by name, **addition()** into **Calculator** class. For this method, we will write and execute JUnit 5 test cases in Maven.

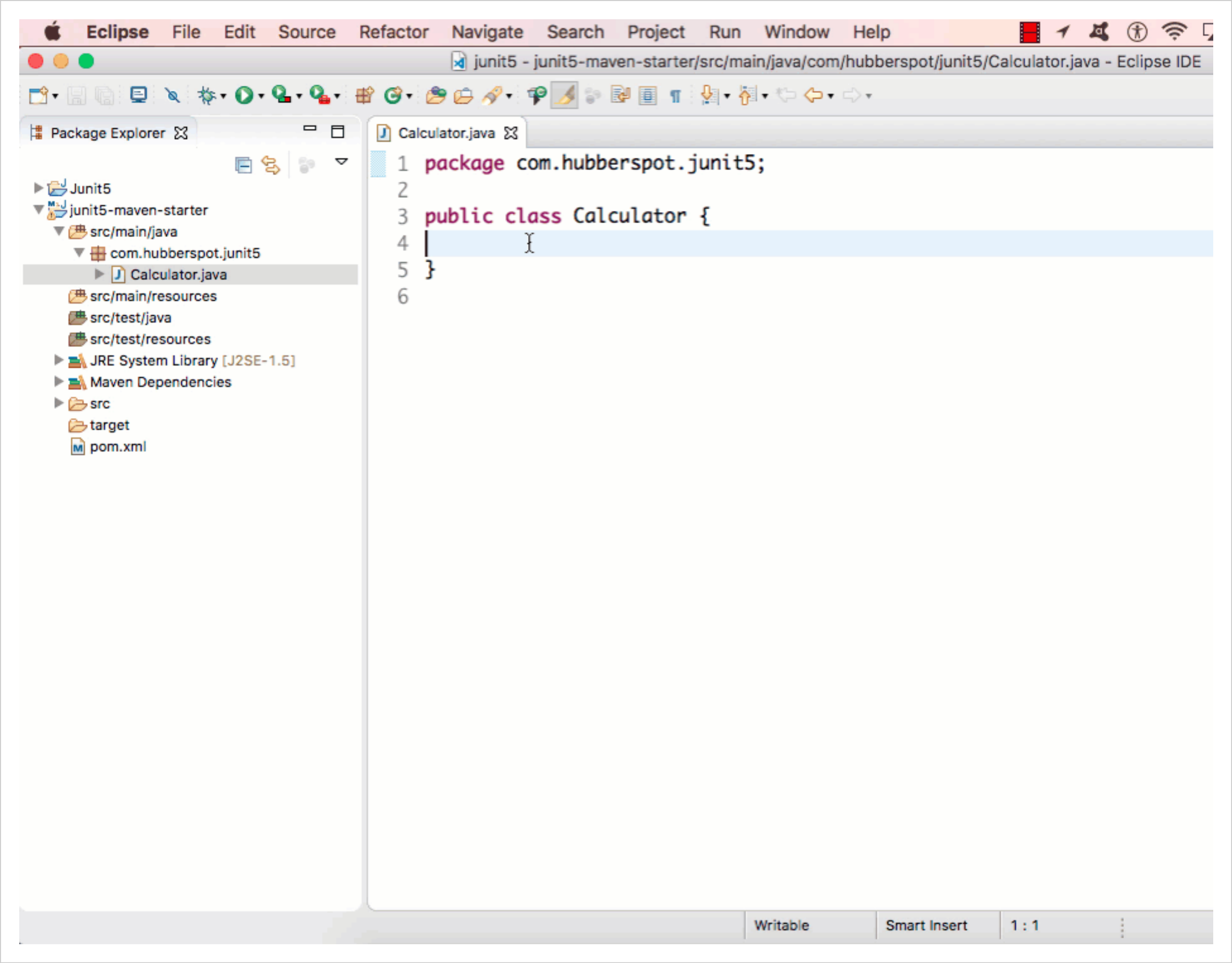
Step 2 - addition() takes in two integer arguments say, num1 and num2.

Step 3 - It will calculate the sum of num1 and num2 and will return it.

See the demonstration of steps below.

Calculator.java

```
package com.hubberspot.junit5;

public class Calculator {
        public int addition(int num1, int num2) {
                return num1 + num2;
        }
}
```



```java
package com.hubberspot.junit5;

public class Calculator {

}
```

Create a Java Test class

Let's create a Java Test class. It will test **Calculator.java** class.

Step 1 - Expand the ***junit5-maven-starter*** project. In order to create a new Java Test class, right-click on **src/test/java** traverse to New --> Class. Click on Class to add a new Java class.

Note -The test class will be created in src/test/java, as it will allow seperation of concerns, i.e. keeping source files seperate from the test files.

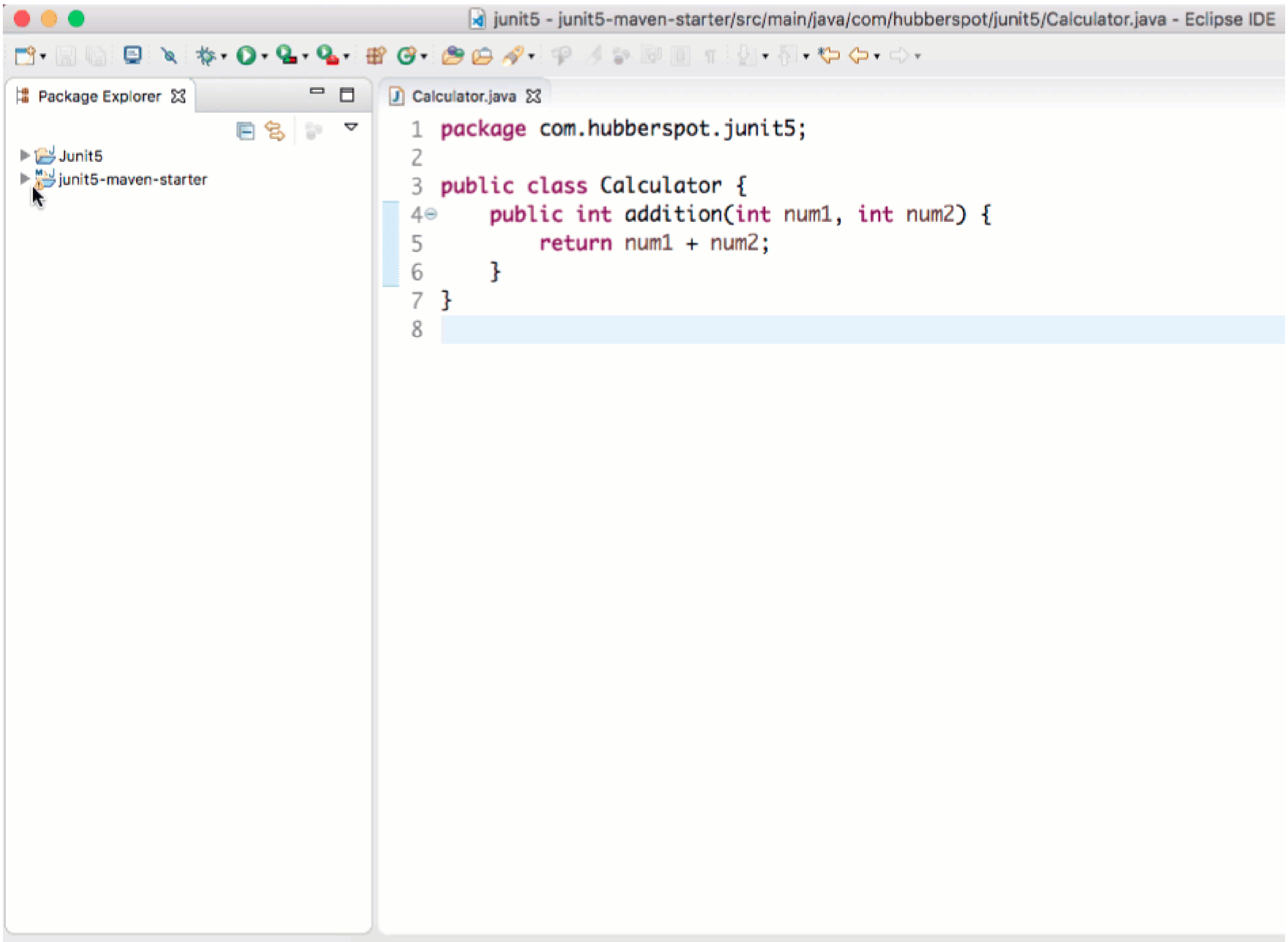
Step 2 - New Java Class popup window will be opened.

Step 3 - As a best practice, we keep the same **Package** name of the test class and src class. Thus, folder names are different but package names are same.

Step 4 - As it is a test class and it provides test methods for **Calculator** class therefore, we keep Name of class as, **CalculatorTest**.

Step 5 - Click Finish.

A Java class by name **CalculatorTest.java** will be created in Eclipse IDE. See demonstration of steps below.



junit5 - junit5-maven-starter/src/main/java/com/hubberspot/junit5/Calculator.java - Eclipse IDE
Package Explorer
Junit5
junit5-maven-starter
Calculator.java

```java
package com.hubberspot.junit5;

public class Calculator {
    public int addition(int num1, int num2) {
        return num1 + num2;
    }
}

```

Add a @Test method

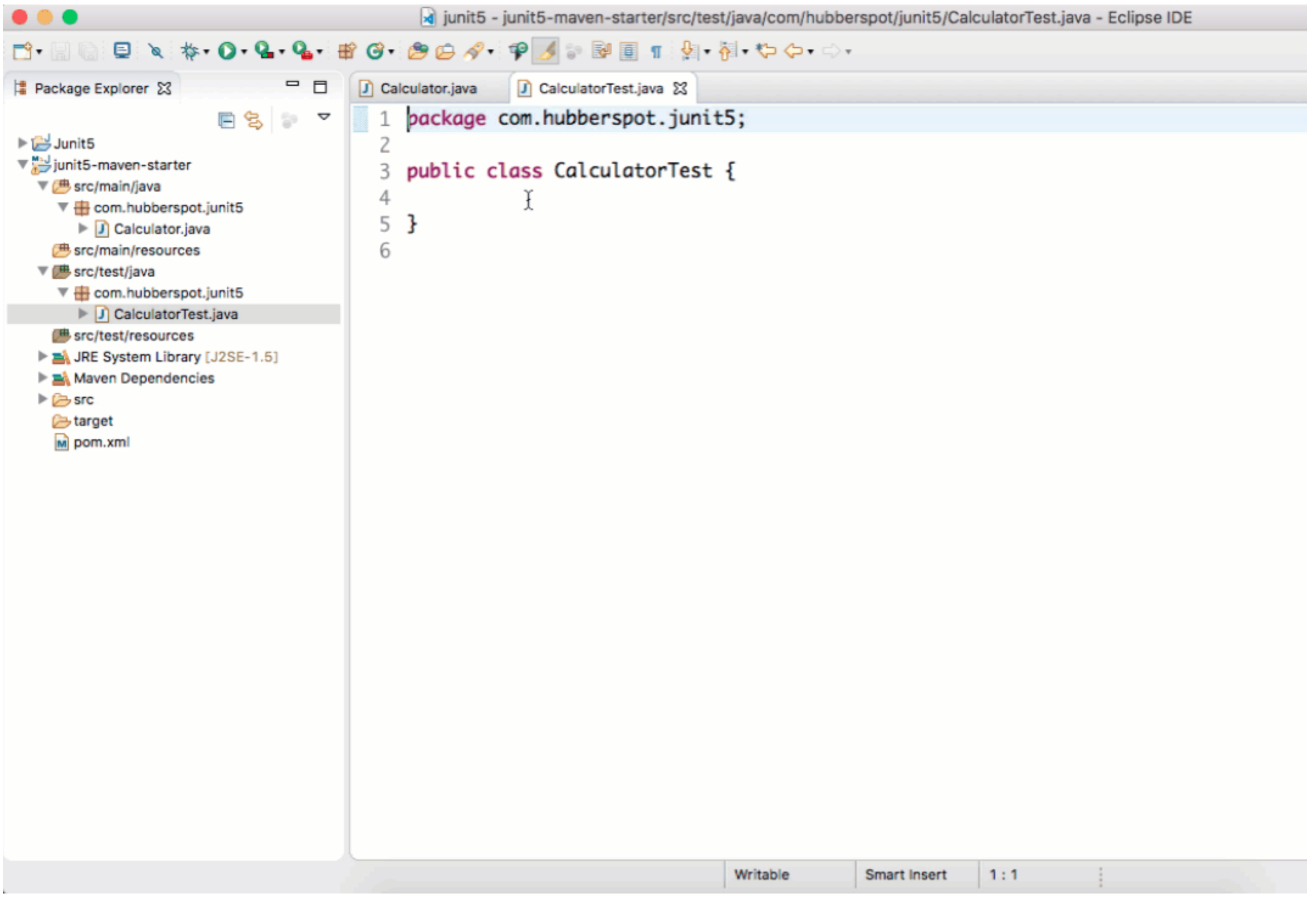
In **CalculatorTest.java** class create a test method **givenTwoNumbers3And4_whenAdditionIsCalled_then7IsReturned()**. This method is created using **given/when/then** format, which we will discuss more in upcoming lessons. This method is marked with **@Test** annotation, which signifies that it is a test method. This method tests addition of two numbers by calling **addition()** method of **Calculator** class.

It passes two numbers as, 3 and 4 and expects 7 to be returned from **addition()** method. It asserts return value using **assertEquals()** method present in Assertions API, which we will discuss more in upcoming lessons.

On running **mvn test** command. It executes **@Test** method using Maven.

CalculatorTest.java

```
package com.hubberspot.junit5;


import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.Test;

public class CalculatorTest {

        @Test
        public void givenTwoNumbers3And4_whenAdditionIsCalled_then7IsReturned() {
                Calculator calculator = new Calculator();
                assertEquals(7, calculator.addition(3, 4));
        }

}
```



```
package com.hubberspot.junit5;

public class CalculatorTest {

}

```