# JUnit - Parameterized Test



Previous Page Next Page

JUnit 4 has introduced a new feature called **parameterized tests**. Parameterized tests allow a developer to run the same test over and over again using different values. There are five steps that you need to follow to create a parameterized test.

Annotate test class with @RunWith(Parameterized.class).

Create a public static method annotated with @Parameters that returns a Collection of Objects (as Array) as test data set.

Create a public constructor that takes in what is equivalent to one "row" of test data.

Create an instance variable for each "column" of test data.

Create your test case(s) using the instance variables as the source of the test data.

The test case will be invoked once for each row of data. Let us see parameterized tests in action.

## Create a Class

Create a java class to be tested, say, **PrimeNumberChecker.java** in C:\>JUNIT_WORKSPACE.

```
public class PrimeNumberChecker {
   public Boolean validate(final Integer primeNumber) {
      for (int i = 2; i < (primeNumber / 2); i++) {
         if (primeNumber % i == 0) {
            return false;
         }
      }
      return true;
   }
}
```

## Create Parameterized Test Case Class

Create a java test class, say, **PrimeNumberCheckerTest.java**. Create a java class file named **PrimeNumberCheckerTest.java** in C:\>JUNIT_WORKSPACE.

```
import java.util.Arrays;
import java.util.Collection;

import org.junit.Test;
import org.junit.Before;

import org.junit.runners.Parameterized;
import org.junit.runners.Parameterized.Parameters;
import org.junit.runner.RunWith;
import static org.junit.Assert.assertEquals;

@RunWith(Parameterized.class)
public class PrimeNumberCheckerTest {
   private Integer inputNumber;
   private Boolean expectedResult;
   private PrimeNumberChecker primeNumberChecker;

   @Before
   public void initialize() {
      primeNumberChecker = new PrimeNumberChecker();
   }

   // Each parameter should be placed as an argument here
   // Every time runner triggers, it will pass the arguments
   // from parameters we defined in primeNumbers() method

   public PrimeNumberCheckerTest(Integer inputNumber, Boolean expectedResult) {
      this.inputNumber = inputNumber;
      this.expectedResult = expectedResult;
   }

   @Parameterized.Parameters
   public static Collection primeNumbers() {
      return Arrays.asList(new Object[][] {
         { 2, true },
         { 6, false },
         { 19, true },
         { 22, false },
         { 23, true }
      });
   }

   // This test will run 4 times since we have 5 parameters defined
   @Test
   public void testPrimeNumberChecker() {
      System.out.println("Parameterized Number is : " + inputNumber);
      assertEquals(expectedResult,
      primeNumberChecker.validate(inputNumber));
   }
}
```

# 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(PrimeNumberCheckerTest.class);

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

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

Compile the PrimeNumberChecker, PrimeNumberCheckerTest and Test Runner classes using javac.

```
C:\JUNIT_WORKSPACE>javac PrimeNumberChecker.java PrimeNumberCheckerTest.java
TestRunner.java
```

Now run the Test Runner, which will run the test cases defined in the provided Test Case class.

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

Verify the output.

```
Parameterized Number is : 2
Parameterized Number is : 6
Parameterized Number is : 19
Parameterized Number is : 22
Parameterized Number is : 23
true
```

☐ Previous Page Next Page ☐