



[△ Home](#) » [Core Java](#) » [junit](#) » JUnit Keyboard Input Example

ABOUT VINOD KUMAR KASHYAP



Vinod is Sun Certified and love to work in Java and related technologies. Having more than 10 years of experience, he had developed software's including technologies like Java, Hibernate, Struts, Spring, HTML 5, jQuery, CSS, Web Services, MongoDB, AngularJS. He is also a JUG Leader of Chandigarh Java User Group.



JUnit Keyboard Input Example

□ Posted by: Vinod Kumar Kashyap □ in junit □ February 14th, 2017

In this post we shall show users the usage of JUnit keyboard input working. This example is very useful in case users want to enter data from keyboard for testing of their methods. Do not worry, we will the same in the post.

Users are advised to see the JUnit Hello World example where they can see the basic usage of JUnit. In addition, if users want to run their test cases in the order of their choice, they are recommended to visit JUnit FixMethodOrder example.

First of all, let's start with the small introduction of JUnit.

1. JUnit Introduction

JUnit is a testing framework for Java programmers. This is the testing framework where users can unit test their methods for working. Almost all Java programmers used this framework for basic testing. Basic example of the JUnit can be seen in JUnit Hello World example.

Want to be a JUnit Master ?

Subscribe to our newsletter and download the JUnit Programming Cookbook right now!

In order to help you master unit testing with JUnit, we have compiled a kick-ass guide with all the major JUnit features and use cases! Besides studying them online you may download the eBook in PDF format!

Email address:

Your email address

Sign up

2. Tools Required

Users have to have a basic understanding of the Java. These are the tools that are used in this example.

- Eclipse (users can use any tool of their choice. We are using STS, which is built on the top of the Eclipse)
- Java
- Maven (optional, users can also use this example without Maven, but we recommend to use it)

NEWSLETTER

180,180 insiders are already enjoying weekly updates and complimentary whitepapers!

Join them now to gain exclusive access to the latest news in the Java world as well as insights about Android, Scala, Groovy and other related technologies.

Email address:

Your email address

Sign up

JOIN US



With **1,240,6**
unique visitors
500 authors
placed among
related sites at
Constantly bei
lookout for par
encourage you
So if you have
unique and interesting content then yo

- JUnit 4.12 (this is the latest version we are using for this example)

3. Project Setup

Tip

You may skip project creation and jump directly to the **beginning of the example** below.

In this post we are using the Maven for initial setup of our application. Let's start with the creating of the Maven project. Open eclipse,



File -> New -> Maven Project

On the following screen, fill in the details as shown and click on Next button.

check out our **JCG** partners program. \n be a **guest writer** for Java Code Geek\n your writing skills!

Figure 1: Project Setup Screen 1

Clicking on Next button users will be taken to the below screen. Fill in the details as shown and click on finish.

Figure 2: Project Setup Screen 2

Now we are ready to start writing code for our application.

4. JUnit Keyboard Input

Open

pom.xml

and add the following lines to it. This is the main file which is used by the Maven to download dependencies on users local machine.

pom.xml

```
1 <dependencies>
2   <dependency>
3     <groupId>junit</groupId>
4     <artifactId>junit</artifactId>
5     <version>4.12</version>
6   </dependency>
7 </dependencies>
```

Create a class with the following code. This class will test the parameter passed for leap year. If the passed parameter is Leap year, it will return

true

otherwise it will return

false

.

LeapYear.java

```
1 package junitkeyboardinput;
2
3 public class LeapYear {
4
5     public boolean isLeapYear(int year) {
6         return ((year % 400 == 0) || ((year % 4 == 0) && (year % 100 != 0)));
7     }
8 }
```

```

7   }
8   }

```

Finally, we create a

```
LeapYearTest
```

class which is a JUnit keyboard input test class, that test our

```
isLeapYear()
```

method. We are using, Scanner class of Java.

LeapYearTest.java

```

01 package junitkeyboardinput;
02
03 import static org.junit.Assert.assertTrue;
04
05 import java.io.IOException;
06 import java.util.Scanner;
07
08 import org.junit.Test;
09
10 public class LeapYearTest {
11
12     @Test
13     public void isLeapYearTest() throws IOException {
14         LeapYear check = new LeapYear();
15         assertTrue("Leap Year", check.isLeapYear(2015));
16     }
17
18     @Test
19     public void isLeapYearKeyboardTest() throws IOException {
20         LeapYear leapYear = new LeapYear();
21
22         Scanner sc = new Scanner(System.in);
23         System.out.print("Enter year(yyyy):");
24         int year = sc.nextInt();
25         assertTrue("Leap Year", leapYear.isLeapYear(year));
26         sc.close();
27     }
28 }

```

Let's see what is happening in this class.

Line no 22: We are creating object of Scanner class.

Line no 24: We are taking the input as Integer

Line no 26: closing the Scanner.

In the console window, users will be prompted for entering year. Enter year and depend upon the input, test case will fail or pass.

```
1 Enter year(yyyy):
```

If user passes, 2016 as the year, result will be shown in JUnit window.

```
1 Enter year(yyyy): 2016
```