


[ANDROID](#)
[CORE JAVA](#)
[DESKTOP JAVA](#)
[ENTERPRISE JAVA](#)
[JAVA BASICS](#)
[JVM LANGUAGES](#)
[SOFTWARE DEVELOPMENT](#)
[Home](#) » [Core Java](#) » [junit](#) » JUnit Test Timeout 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 Test Timeout Example

Posted by: Vinod Kumar Kashyap | in junit | March 30th, 2017

In this article, we shall show you the working of JUnit timeout. JUnit provides a very good way of testing your methods against the timeout. In JUnit Test Timeout example, we will show how we can test out methods for timeouts.

There are times when we want our methods to execute in a specific time. For example, we want a method to be completed in 1 second. This can be achieved easily by using different types in JUnit.

1. Introduction

As a regular reader, you already know that JUnit is a powerful way of unit testing our programs. It provides various techniques through which we can test our methods.

Let's take a case where our method is taking a lot of time to execute. It causes a lot of performance issue and you want to test in a way so that it completes in a specific time. What will you do?

To answer this, JUnit provides a way through which this can be achieved. We will see in this example how we can do that.

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:

[Sign up](#)

2. Ways to Test Timeout

So, JUnit provides 3 different ways of testing methods against the time.

- First is the

NEWSLETTER

180,180 insiders are already receiving 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:

[Sign up](#)

JOIN US



With **1,240,600** unique visitors and **500** authors placed among related sites and Constantly being a lookout for new and interesting content then you

5/10/2018

JUnit Test Timeout Example | Examples Java Code Geeks - 2017

timeout

parameter to the

@Test

annotation

• Second, using global

@Rule

annotation

• Lastly using the

@ClassRule

annotation

We will see these 3 methods in details in below example.

3. Project Setup

Tip

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

We are using Eclipse for this project. We are also using Maven as a dependency build tool. Open Eclipse, Click

File -> New -> Maven Project

.

Fill in the details as shown and click **Next** button.

check out our **JCG** partners program. 1
be a **guest writer** for Java Code Geek
your writing skills!

Figure 1: JUnit Test Timeout Example Setup 1

Fill all details as shown and click on **Finish** button.

Figure 2: JUnit Test Timeout Example Setup 2

We are done with the project setup.

4. JUnit Test Timeout Example

First of all copy below code and paste in your

pom.xml

.

pom.xml

```
01 <dependencies>
02   <dependency>
03     <groupId>junit</groupId>
04     <artifactId>junit</artifactId>
05     <version>4.12</version>
06   </dependency>
07 </dependencies>
08
09 <properties>
10   <maven.compiler.source>1.8</maven.compiler.source>
11   <maven.compiler.target>1.8</maven.compiler.target>
12 </properties>
```

Now we create a simple bean class for testing.

Bank.java

```
01 package junittesttimeout;
02
03 import java.util.concurrent.TimeUnit;
04
05 public class Bank {
06
07   private double totalCash;
08   private int totalAccounts;
09
10   public Bank(double cash, int accounts) {
11     this.totalCash = cash;
12     this.totalAccounts = accounts;
13   }
```

```

14
15 public double getTotalCash() throws InterruptedException {
16     double cash = 0.0;
17     for (int index = 0; index < 5; index++) {
18         cash += index;
19         TimeUnit.SECONDS.sleep(1);
20     }
21     return cash;
22 }
23
24 public int getTotalAccounts() throws InterruptedException {
25     TimeUnit.MILLISECONDS.sleep(500);
26     return totalAccounts;
27 }
28
29 @Override
30 public String toString() {
31     return "Bank [cash=" + totalCash + ", accounts=" + totalAccounts + "]";
32 }
33 }

```

This class is used for testing purpose only. As you can see, we are having time sleep statements(highlighted) in the program. That are used for testing timeout scenarios.

5. Test Class

Let's explore the various strategies.

5.1 Using timeout parameter

We can use

```
timeout
```

parameter with

```
@Test
```

[TestClass.java](#)

```

01 package junittesttimeout;
02
03 import static org.hamcrest.CoreMatchers.is;
04 import static org.junit.Assert.assertThat;
05
06 import org.junit.BeforeClass;
07 import org.junit.Test;
08
09 public class TestTimeout {
10
11     private static Bank bank;
12
13     @BeforeClass
14     public static void init() {
15         bank = new Bank(500000, 100);
16     }
17
18     @Test(timeout = 2000)
19     public void totalCashTest() throws InterruptedException {
20         assertThat(10.0, is(bank.getTotalCash()));
21     }
22
23     @Test(timeout = 1000)
24     public void totalAccountsTest() throws InterruptedException {
25         assertThat(100, is(bank.getTotalAccounts()));
26     }
27 }

```

In this example, at line 18 and 23, we have set the timeout on tests. Now when the method starts execution and it takes more time than the timeout specified it will not pass and eventually fail. If test the case executes within a specified time then it will pass.

5.2 Using @Rule annotation

We can use

```
@Rule
```

annotation. This is helpful when we want all our test cases to be passed within a specific time. For example, we want our each test case to be executed in 2 seconds. See example below for more details.

[TestTimeoutGlobal.java](#)

```

01 package junittesttimeout;
02
03 import static org.hamcrest.CoreMatchers.is;
04 import static org.junit.Assert.assertThat;
05
06 import org.junit.BeforeClass;
07 import org.junit.Rule;
08 import org.junit.Test;

```

```

9  import org.junit.rules.Timeout;
10
11  public class TestTimeoutGlobal {
12
13      private static Bank bank;
14
15      @Rule
16      public Timeout globalTimeout = Timeout.seconds(2);
17
18      @BeforeClass
19      public static void init() {
20          bank = new Bank(500000,100);
21      }
22
23      @Test
24      public void totalCashTest() throws InterruptedException {
25          assertThat(10.0, is(bank.getTotalCash()));
26      }
27
28      @Test
29      public void totalAccountsTest() throws InterruptedException {
30          assertThat(100, is(bank.getTotalAccounts()));
31      }
32  }

```

Here we simply, create

```
@Rule
```

at starting of the class. It applies to each and every test case in a class including

```
@BeforeClass
```

and

```
@Before
```

annotations.

5.3 Using @ClassRule annotation

We can use the

```
@ClassRule
```

annotation on class. It will see that all methods in a class execute in a specific time. So, here we want all the test cases collectively to be passed within a specific time of 10 seconds.

[TestTimeoutGlobalClass.java](#)

```

01  package junittesttimeout;
02
03  import static org.hamcrest.CoreMatchers.is;
04  import static org.junit.Assert.assertThat;
05
06  import org.junit.BeforeClass;
07  import org.junit.ClassRule;
08  import org.junit.Test;
09  import org.junit.rules.Timeout;
10
11  public class TestTimeoutGlobalClass {
12
13      private static Bank bank;
14
15      @ClassRule
16      public static Timeout globalTimeout = Timeout.seconds(10);
17
18      @BeforeClass
19      public static void init() {
20          bank = new Bank(500000, 100);
21      }
22
23      @Test
24      public void totalCashTest() throws InterruptedException {
25          assertThat(10.0, is(bank.getTotalCash()));
26      }
27
28      @Test
29      public void totalAccountsTest() throws InterruptedException {
30          assertThat(100, is(bank.getTotalAccounts()));
31      }
32  }

```

Tip

Difference between

```
@Rule
```

and

```
@ClassRule
```

is that, former is used for testing each method for a specific time, whereas latter is used to test all methods to be executed in a specific time. Above example will simply executes the methods of the class and sees that all test cases passed in a specific time of i.e. 10 seconds.

6. Conclusion

Through this example, you have learned how we can test our methods against time. We have used 3 strategies for this.

1. using timeout parameter with

```
@Test
```

2. using

```
@Rule
```

annotation

3. using

```
@ClassRule
```

annotation

7. Download the Eclipse Project

This is an example of JUnit Test Timeout.

Download

You can download the full source code of this example here: **JUnitTestTimeoutExample.zip**

Tagged with: TIMEOUT

Do you want to know how to develop your skillset to become a **Java Rockstar?**

Subscribe to our newsletter to start Rocking right now!

To get you started we give you our best selling eBooks for **FREE!**

1. JPA Mini Book
2. JVM Troubleshooting Guide
3. JUnit Tutorial for Unit Testing
4. Java Annotations Tutorial
5. Java Interview Questions
6. Spring Interview Questions
7. Android UI Design

and many more

Email address:

[Sign up](#)

KNOWLEDGE BASE

[Courses](#)

[News](#)

[Resources](#)

[Tutorials](#)

HALL OF FAME

[Android Alert Dialog Example](#)

[Android OnClickListener Example](#)

[How to convert Character to String and a String to Character Array in Java](#)

ABOUT JAVA CODE GEEKS

JCGs (Java Code Geeks) is an independent online community focused on creating ultimate Java to Java developers resource center; targeted at the technical and technical team lead (senior developer), project manager and junior developer. JCGs serve the Java, SOA, Agile and Telecom communities with daily news via domain experts, articles, tutorials, reviews, announcements, code snippets and source projects.