


Maven – JaCoCo code coverage example

By [mkyong](#) | Last updated: November 15, 2018
Viewed: 0 | +0 pv/w



In this article, we will show you how to use a [JaCoCo Maven plugin](#) to generate a code coverage report for a Java project.

Tested with

- 1. Maven 3.5.3
- 2. JUnit 5.3.1
- 3. jacoco-maven-plugin 0.8.2

Note

JaCoCo is an actively developed line coverage tool, that is used to measure how many lines of our code are tested.

1. JaCoCo Maven Plugin

1.1 Declare the following JaCoCo plugin in the `pom.xml` file.

```
pom.xml

<plugin>
  <groupId>org.jacoco</groupId>
  <artifactId>jacoco-maven-plugin</artifactId>
  <version>0.8.2</version>
  <executions>
    <execution>
      <goals>
        <goal>prepare-agent</goal>
      </goals>
    </execution>
    <!-- attached to Maven test phase -->
    <execution>
      <id>report</id>
      <phase>test</phase>
      <goals>
        <goal>report</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

It will run the JaCoCo ‘report’ goal during the Maven test phase.

MessageBuilder.java

```
package com.mkyong.examples;

public class MessageBuilder {

    public String getMessage(String name) {

        StringBuilder result = new StringBuilder();

        if (name == null || name.trim().length() == 0) {

            result.append("Please provide a name!");

        } else {

            result.append("Hello " + name);

        }

        return result.toString();

    }

}
```

2.2 Unit test above class.

TestMessageBuilder.java

```
package com.mkyong.examples;

import org.junit.jupiter.api.Test;

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

public class TestMessageBuilder {

    @Test
    public void testNameMkyong() {

        MessageBuilder obj = new MessageBuilder();
        assertEquals("Hello mkyong", obj.getMessage("mkyong"));

    }

}
```

2.3 Run `mvn test`, the JaCoCo code coverage report will be generated at `target/site/jacoco/`*

Terminal

```
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.mkyong.examples.TestMessageBuilder
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.012 s - in com.mkyong.ex
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.2:report (report) @ maven-code-coverage ---
[INFO] Loading execution data file D:\maven-examples\maven-code-coverage\target\jacoco.exec
[INFO] Analyzed bundle 'maven-code-coverage' with 1 classes
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 4.164 s
[INFO] Finished at: 2018-11-14T16:48:39+08:00
[INFO] -----
```

2.4 Open the `target/site/jacoco/index.html` file, review the code coverage report :

maven-code-coverage

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.mkyong.examples	<div><div></div></div>	84%	<div><div></div></div>	50%	2	4	1	6	0	2	0	1
Total	5 of 32	84%	2 of 4	50%	2	4	1	6	0	2	0	1

MessageBuilder.java

```
1. package com.mkyong.examples;
2.
3. public class MessageBuilder {
4.
5.     public String getMessage(String name) {
6.
7.         StringBuilder result = new StringBuilder();
8.
9.         if (name == null || name.trim().length() == 0) {
10.
11.             result.append("Please provide a name!");
12.
13.         } else {
14.
15.             result.append("Hello " + name);
16.
17.         }
18.         return result.toString();
19.     }
20.
21. }
```

- 1. Green – Code is tested or covered.
- 2. Red – Code is not tested or covered.
- 3. Yellow – Code is partially tested or covered.

3. Improving the Unit Test

```
package com.mkyong.examples;

import org.junit.jupiter.api.Test;

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

public class TestMessageBuilder {

    @Test
    public void testNameMkyong() {

        MessageBuilder obj = new MessageBuilder();
        assertEquals("Hello mkyong", obj.getMessage("mkyong"));

    }

    @Test
    public void testNameEmpty() {

        MessageBuilder obj = new MessageBuilder();
        assertEquals("Please provide a name!", obj.getMessage(" "));

    }

}
```

Review the report again.


Terminal

\$ mvn clean test

target/site/jacoco/index.html

MessageBuilder.java

```
1. package com.mkyong.examples;
2.
3. public class MessageBuilder {
4.
5.     public String getMessage(String name) {
6.
7.         StringBuilder result = new StringBuilder();
8.
9.         if (name == null || name.trim().length() == 0) {
10.
11.             result.append("Please provide a name!");
12.
13.         } else {
14.
15.             result.append("Hello " + name);
16.
17.         }
18.         return result.toString();
19.     }
20.
21. }
```



3.2 Add one more test for the yellow line if condition.

TestMessageBuilder.java

```
import org.junit.jupiter.api.Test;

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

public class TestMessageBuilder {

    @Test
    public void testNameMkyong() {

        MessageBuilder obj = new MessageBuilder();
        assertEquals("Hello mkyong", obj.getMessage("mkyong"));

    }

    @Test
    public void testNameEmpty() {

        MessageBuilder obj = new MessageBuilder();
        assertEquals("Please provide a name!", obj.getMessage(" "));

    }

    @Test
    public void testNameNull() {

        MessageBuilder obj = new MessageBuilder();
        assertEquals("Please provide a name!", obj.getMessage(null));

    }

}
```


Review the report again.


Terminal

\$ mvn clean test

target/site/jacoco/index.html

maven-code-coverage

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
 com.mkyong.examples	<div></div>	100%	<div></div>	100%	0	4	0	6	0	2	0	1
Total	0 of 32	100%	0 of 4	100%	0	4	0	6	0	2	0	1

 WebAdvisor

Nous avons testé cette page et bloqué le contenu provenant de sites suspects ou potentiellement dangereux. Autorisez ce contenu uniquement si vous êtes certain qu'il provient de sites sécurisés.

Afficher tout le contenu bloqué

X

1. package com.mkyong.examples;
2.
3. public class MessageBuilder {
4.
5. public String getMessage(String name) {
6.
7. StringBuilder result = new StringBuilder();
8.
9. if (name == null || name.trim().length() == 0) {
10.
11. result.append("Please provide a name!");
12.
13. } else {
14.
15. result.append("Hello " + name);
16.
17. }
18. return result.toString();
19. }
20.
21. }




Finally, all lines are tested, 100% coverage.

4. FAQs

4.1 Make sure lines coverage must meet the minimum 90%.

pom.xml

 WebAdvisor

Nous avons testé cette page et bloqué le contenu provenant de sites suspects ou potentiellement dangereux. Autorisez ce contenu uniquement si vous êtes certain qu'il provient de sites sécurisés.

Afficher tout le contenu bloqué

X

```
<artifactId>jacoco-maven-plugin</artifactId>
<version>${jacoco.version}</version>
<executions>
  <execution>
    <goals>
      <goal>prepare-agent</goal>
    </goals>
  </execution>
  <execution>
    <id>jacoco-report</id>
    <phase>test</phase>
    <goals>
      <goal>report</goal>
    </goals>
    <!-- default target/jscoco/site/* -->
    <configuration>
      <outputDirectory>target/jacoco-report</outputDirectory>
    </configuration>
  </execution>
</executions>
</plugin>
```

Download Source Code

```
$ git clone https://github.com/mkyong/maven-examples.git
$ cd maven-code-coverage
$ mvn clean test
# view report at 'target/site/jacoco/index.html'
```

References

- 1. [Wikipedia : Java code coverage tools](#)
- 2. [JaCoCo Java Code Coverage Library](#)
- 3. [JaCoCo in Eclipse IDE](#)


Related Articles

- [How to install Maven on Mac OSX](#)
- [Maven - PITest mutation testing example](#)
- [Maven 2 + Hibernate 3.2 + MySQL Example \(Annotatio...](#)
- [Gradle JaCoCo - Incompatible version 1006](#)
- [Maven 3 + Hibernate 3.6 + Oracle 11g Example \(Anno...](#)
- [JaCoCo Java Code Coverage + Maven example](#)



mkyong

Founder of [Mkyong.com](#), love Java and open source stuff. Follow him on [Twitter](#). If you like my tutorials, consider make a donation to [these charities](#). Read all published posts by [mkyong](#)



Join the discussion...

newest oldest most voted

robert

I get the below error. I am using Maven 3.6.1, Java 8, Jacoco plugin 0.8.3 (but same error for 0.8.2)

INFO] — jacoco-maven-plugin:0.8.3:report (report) @ RRC —
[INFO] Loading execution data file /Users/acme/dev/src/gitlab/acme/abc/target/jacoco.exec
[INFO] —————
[INFO] BUILD FAILURE
[INFO] —————
[INFO] Total time: 7.519 s
[INFO] Finished at: 2019-05-03T11:48:26+01:00
[INFO] —————
[ERROR] Failed to execute goal org.jacoco:jacoco-maven-plugin:0.8.3:report (report) on project ABC: An error has occurred in JaCoCo report generation.: Error while creating report: Error while analyzing /Users/acme/dev/src/gitlab/acme/abc/target/classes/docs/ABC Release Statement.odt. unexpected EOF -> [Help 1]
[ERROR]

+ 12 -

Reply

1 year ago

EnriqueSF

it's a known problem, please take a look:
<https://github.com/jacoco/jacoco/issues/546#issuecomment-305495921>
<https://github.com/jacoco/jacoco/issues/394>

+ 0 -

Reply

18 days ago

Pat

Error : The POM for org.jacoco:jacoco-maven-plugin:jar:0.8.2 is missing,

+ 4 -

Reply

1 year ago

chris

Worked perfectly. This is the best documentation I've seen for jacoco around. You would think that the dolts who wrote the code would document how to use their coverage tool.

+ 3 -

Reply

9 months ago

Pat

not working i get:

[WARNING] The POM for org.jacoco:jacoco-maven-plugin:jar:0.8.2 is missing, no dependency information available
[WARNING] Error injecting: org.jacoco.maven.AgentMojo
java.lang.NoClassDefFoundError: org/jacoco/core/runtime/AgentOptions
[ERROR] Failed to execute goal org.jacoco:jacoco-maven-plugin:0.8.2:prepare-agent (default)
Execution default of goal org.jacoco:jacoco-maven-plugin:0.8.2:prepare-agent failed: A required class was missing while executing org.jacoco:jacoco-maven-plugin:0.8.2:prepare-agent: org/jacoco/core/runtime/AgentOptions

+ 0 -

Reply

1 year ago

- Guest

+ 0 -

Reply

1 year ago
- Adina
- Excellent tutorial! Works without any problems with the latest JaCoCo version, 0.8.5. Thank you!
- Guest

+ 0 -

Reply

2 months ago
- Rares Cristea
- Hello!
- I've followed your tutorial, but for some reason, my report doesn't include line by line coverage. I can see for each function the percentage of coverage, but I cannot click on it to see my java code as it was covered by the tests. Any help ?
- Thank you!
- Guest

+ 0 -

Reply

2 months ago
- omprasad
- Guest

[INFO] Scanning for projects... [INFO] [INFO] ----- [INFO] Building demo1 0.0.1-SNAPSHOT [INFO] -----[jar]----- [INFO] [INFO] - jacoco-maven-plugin:0.8.2:prepare-agent (default) @ demo1 - [INFO] argLine set to - javaagent:C:\\Users\\Admin\\.m2\\repository\\org\\jacoco\\org.jacoco.agent\\0.8.2\\org.jacoco.agent-0.8.2-runtime.jar=destfile=D:\\algoshack_development\\AlgoAfScripts_02042020_1226\\demo1\\target\\jacoco.exec [INFO] [INFO] - maven-resources-plugin:2.6:resources (default-resources) @ demo1 - [WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e. build is platform dependent! [INFO] Copying 0 resource [INFO] [INFO] - maven-compiler-plugin:3.1:compile (default-compile) @ demo1 - [INFO] Nothing to compile - all classes are up to date [INFO] [INFO] - maven-resources-plugin:2.6:testResources (default-testResources) @ demo1 - [WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e. build is platform dependent! [INFO] Copying 0... [Read more »](#)

+ 0 -

Reply

1 month ago