# Cobertura

Cobertura is a free Java tool that calculates the percentage of code accessed by tests. It can be used to identify which parts of your Java program are lacking test coverage. It is based on jcoverage.

You can execute cobertura via command line, Ant, Maven or by plugins provides by different IDEs(eg. eCobertura plugin for eclipse).

Here we will discuss how to execute cobertura using eCobertura(an eclipse plugin) and using Maven.

eCobertura (An eclipse plugin)

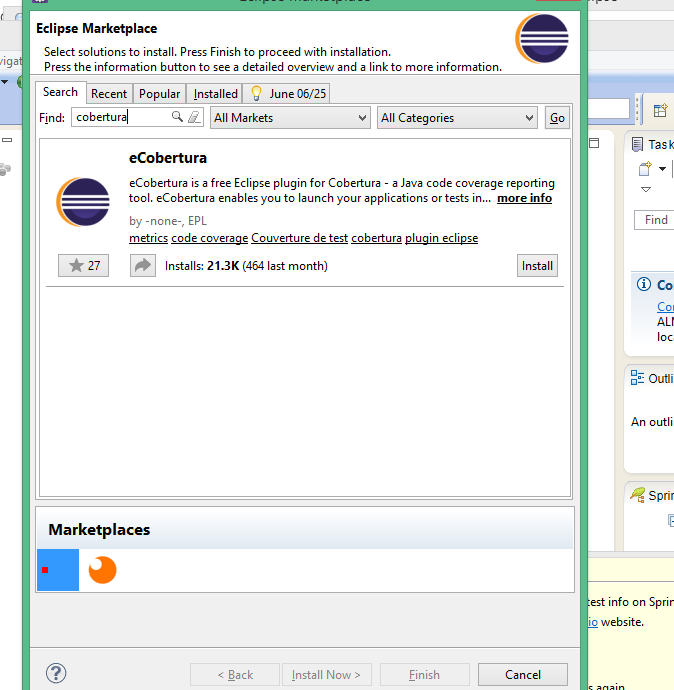
eCobertura is a free Eclipse plugin for Cobertura - a Java code coverage reporting tool.

eCobertura enables you to **launch your applications or tests** in Cobertura-covered mode directly from **within Eclipse**. View your source files **colored** according to the coverage results. **Browse** through the detailed coverage results in a tree view.

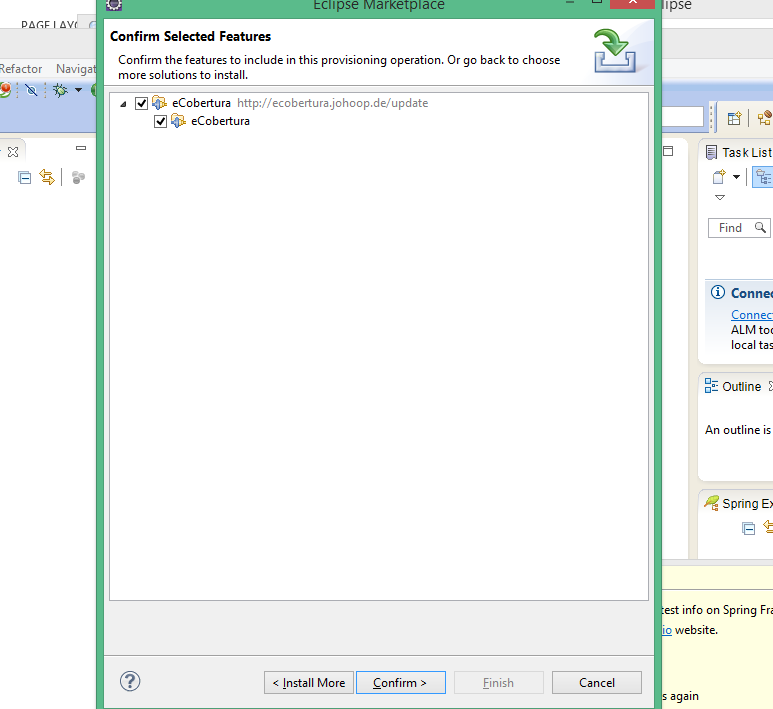
This plugin is based on Cobertura because Cobertura is able to measure **branch coverage**.

## Install

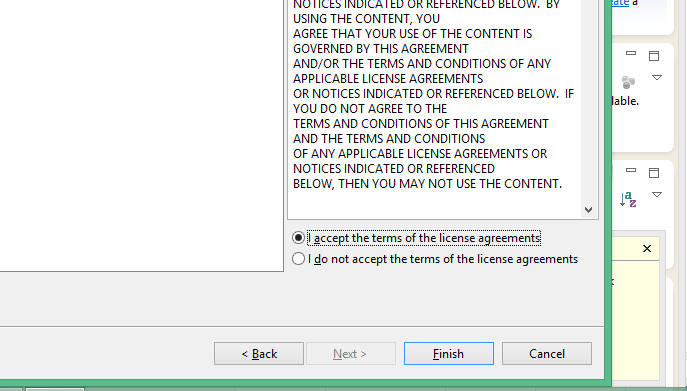
**Step 1:** goto **Eclipse Marketplace** and search for cobertura. Then select install button of eCobertura plugin as shown below.



**Step 2 :** select eCobertura checkbox and hit confirm button



**Step 3 :** accept the terms of the licesne agreements and click finish. Then eCobertura plugin will get installed in your eclipse



**Launching Steps**

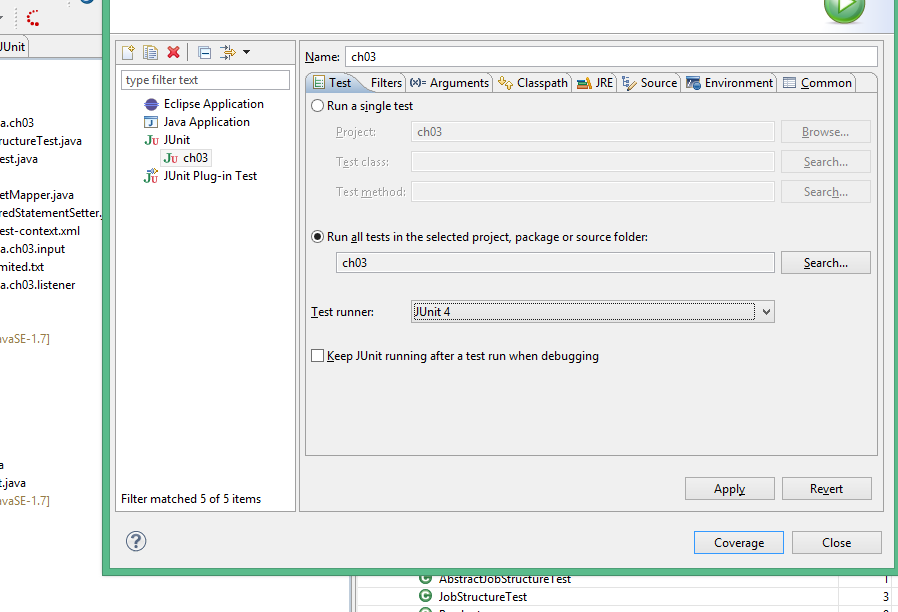
* Open the Coverage Session View (under "Window", "Show View", "Other...", "Cobertura".
* Select the application or test to run.
* Select the respective "Cover As..." command from the main menu, context menu or toolbar button.
* Optionally, you can define class and package coverage filters in the launch configuration (under the tab named "Filters").

**Demo execution:**

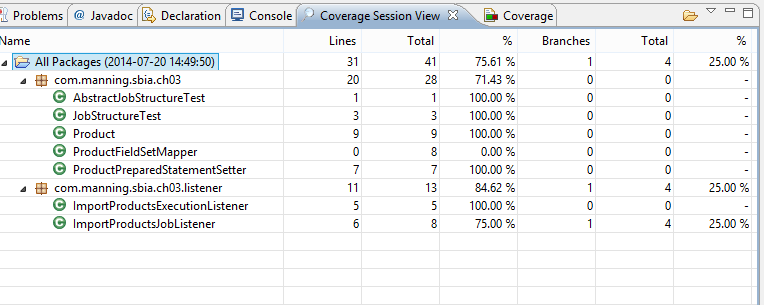
**Step 1:** Right click on your project > cover as > coverage configurations.

**Step 2:** Double click on junit, then you will find a run configuration gets created.

**Step 3:** In run configuration you can select to run a single test or all tests inside a project or package. Then click apply and click coverage button to begin coverage analysis using cobertura. As shown in the following figure



**Step 4:**Now you can view the coverage summary in **Coverage session view,** which shows all the details of line and branch coverage.



**Analyze**

* Source code lines in source editors are green or red depending on whether they were covered or not.
* Browse the coverage session tree view for detailed information.
* Click the column headers to sort the tree table.
* Jump from the classes in the tree view to their respective source editors.

Configure cobertura in maven project

In pom.xml add the following plugin

<plugin>

<groupId>org.codehaus.mojo</groupId>

<artifactId>cobertura-maven-plugin</artifactId>

<version>2.5.1</version>

<configuration>

<formats>

<format>html</format>

<format>xml</format>

</formats>

</configuration>

</plugin>

Now if you run : **mvn cobertura:cobertura**

It will generate two type of code coverage report one in the form of xml and the other one in html. You can find those report inside **target\site\cobertura in our project.**

By default cobertura will make coverage analysis on every java classes. We can configure cobertura to skip certain classes as follows

<plugin>

<groupId>org.codehaus.mojo</groupId>

<artifactId>cobertura-maven-plugin</artifactId>

<version>2.5.1</version>

<configuration>

<formats>

<format>html</format>

<format>xml</format>

</formats>

<instrumentation>

<excludes>

**<exclude>com/\*</exclude>**

</excludes>

</instrumentation>

</configuration>

</plugin>

Here  **<exclude>com/\*</exclude>** tells cobertura to skip all the clases inside com package and its sub packages