Code Coverage

Angelo Gargantini
Informatica III

Tool per la valutazione di copertura

Esistono tantissimi tool per la valutazione di copertura

- Gratis/open source
- Commerciali

Gratis:

- Google CodePro AnalytiX
- Eclemma
- CodeCover

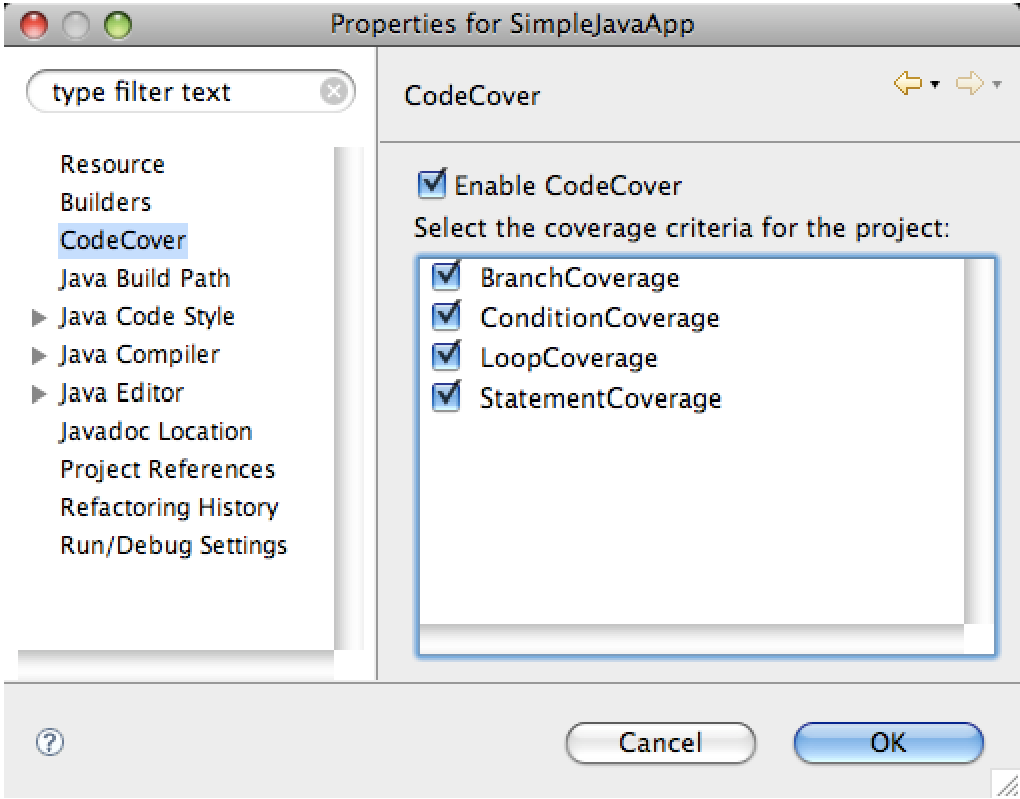
CodeCover

Come usarlo- enable

You need to enable CodeCover-

Open the project properties dialog of the project and navigate to the CodeCover category.

Select the checkbox as shown in the screenshot below. You have now activated CodeCover for the "SimpleJavaApp" project. You also need to select the coverage criteria to be used in the instrumentation. In this case all available criteria were selected.



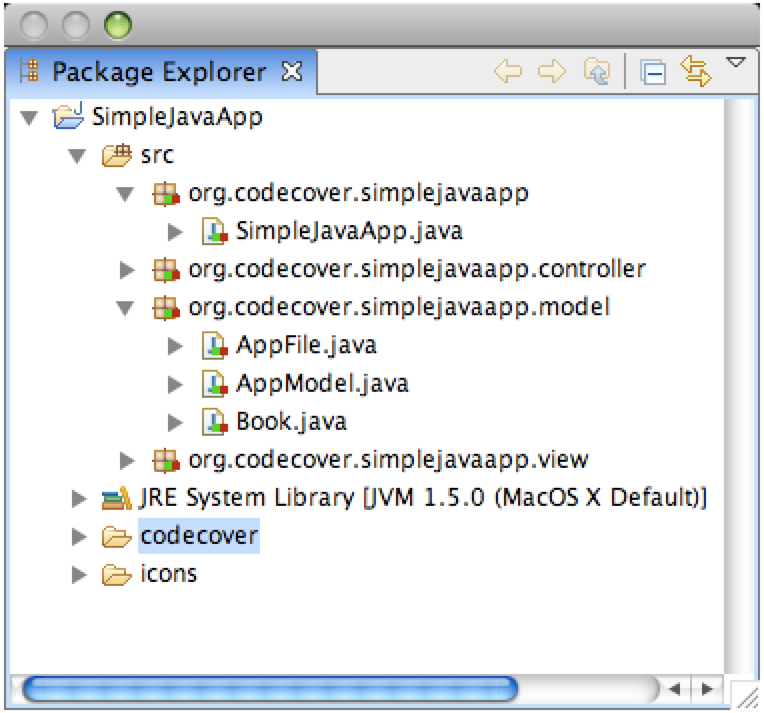
☑ Enable CodeCover

Select the coverage criteria for the project:

☑ BranchCoverage
☑ ConditionCoverage
☑ LoopCoverage
☑ StatementCoverage

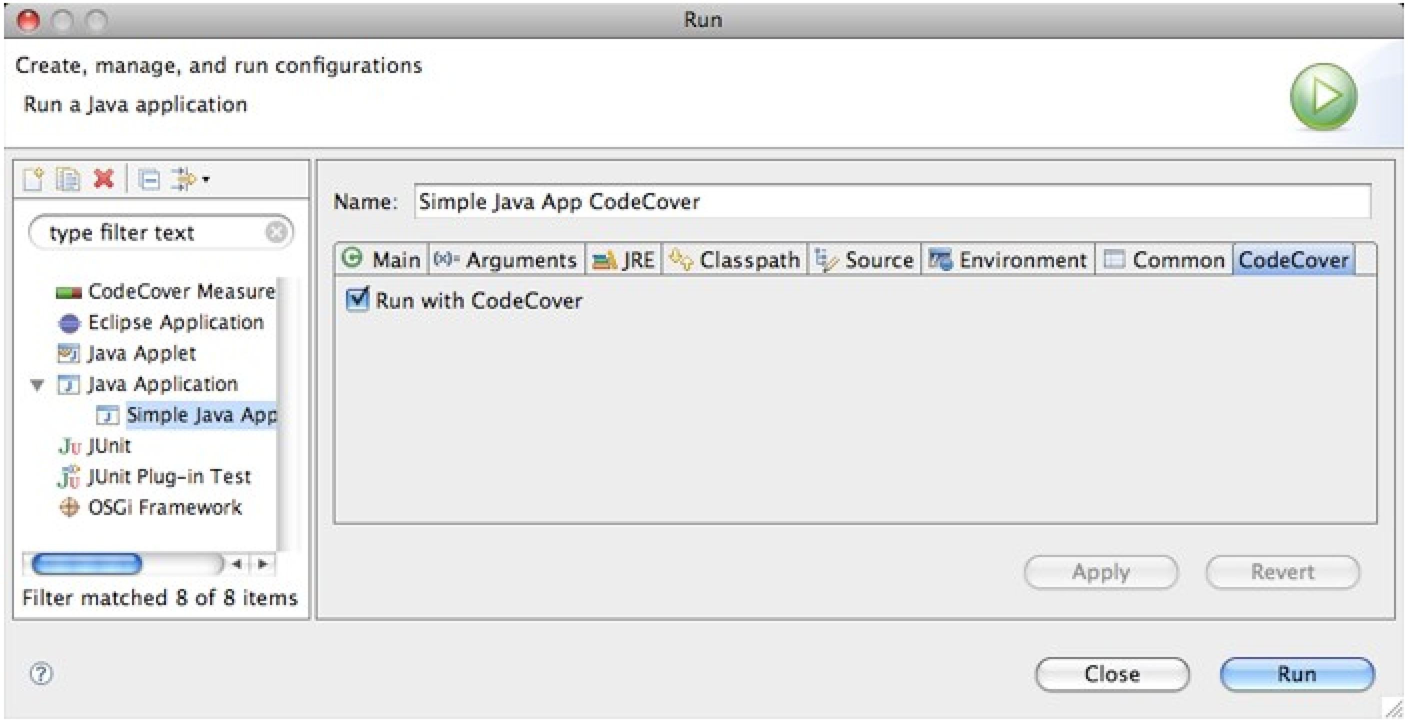
Use for Coverage Measurement

Select the classes you want to instrument. Open the package explorer view, navigate to the source folder of the "SimpleJavaApp" project and select those classes you want to be instrumented. Open the context menu of this selection and select the "Use For Coverage Measurement" entry. This will mark the selected items with an icon.


Package Explorer
SimpleJavaApp
src
org.codecover.simplejavaapp
SimpleJavaApp.java
org.codecover.simplejavaapp.controller
org.codecover.simplejavaapp.model
AppFile.java
AppModel.java
Book.java
org.codecover.simplejavaapp.view
JRE System Library [JVM 1.5.0 (MacOS X Default)]
codecover
icons

Run with CodeCover

You need to tell Eclipse to use CodeCover. This is done in the "run configuration" dialog. You can reuse existing configurations or create new ones. Navigate to the CodeCover tab in the configuration and select the "Run with CodeCover" checkbox.



Run

Create, manage, and run configurations

Run a Java application

type filter text

- CodeCover Measure
- Eclipse Application
- Java Applet
- Java Application
  - Simple Java App
- JUnit
- JUnit Plug-in Test
- OSGi Framework

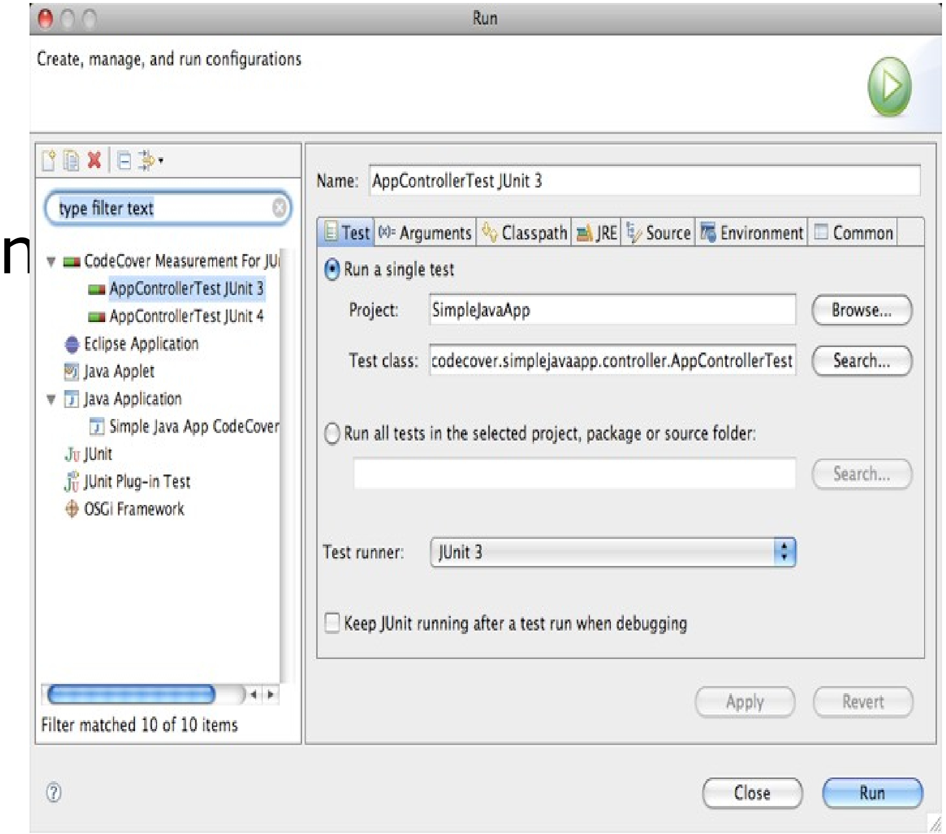
Filter matched 8 of 8 items

Name: Simple Java App CodeCover

☑ Run with CodeCover

Run with JUnit

To use your existing test suite you need to create a new "CodeCover Measurement For JUnit" run configuration.Select the class which contains your JUnit test cases or test suite. Start the SUT with the run configuration to start the measurement. After terminating, the measurements are automatically stored in a test session, which holds all the test cases your test suite defined.



Run

Create, manage, and run configurations

Name: AppControllerTest JUnit 3

● Run a single test

Project: SimpleJavaApp

Test class: codecover.simplejavaapp.controller.AppControllerTest

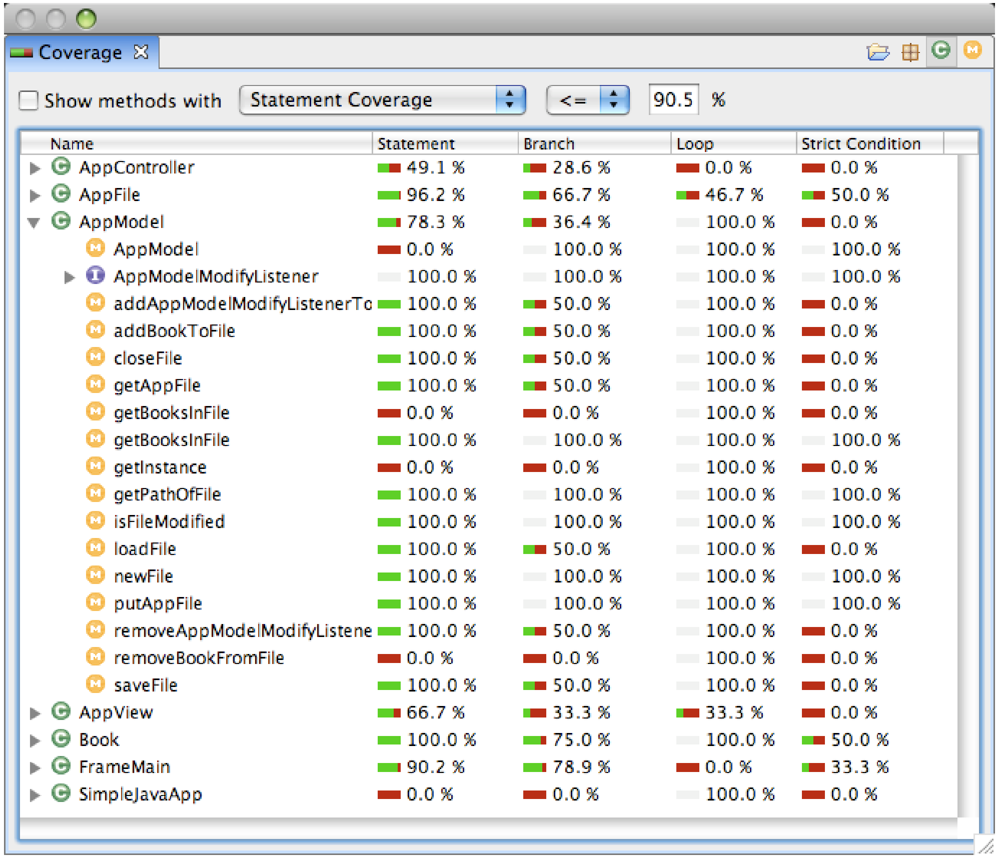
○ Run all tests in the selected project, package or source folder:

Test runner: JUnit 3

□ Keep JUnit running after a test run when debugging

Filter matched 10 of 10 items

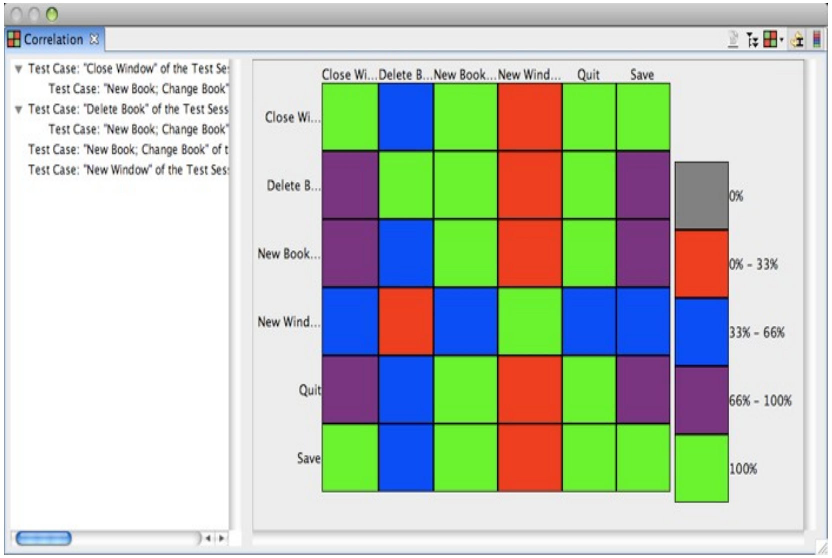
View measured data in Eclipse



☐ Show methods with Statement Coverage <= 90.5 %

Name	Statement	Branch	Loop	Strict Condition
▶ AppController	49.1 %	28.6 %	0.0 %	0.0 %
▶ AppFile	96.2 %	66.7 %	46.7 %	50.0 %
▼ AppModel	78.3 %	36.4 %	100.0 %	0.0 %
AppModel	0.0 %	100.0 %	100.0 %	100.0 %
▶ AppModelModifyListener	100.0 %	100.0 %	100.0 %	100.0 %
addAppModelModifyListenerTo	100.0 %	50.0 %	100.0 %	0.0 %
addBookToFile	100.0 %	50.0 %	100.0 %	0.0 %
closeFile	100.0 %	50.0 %	100.0 %	0.0 %
getAppFile	100.0 %	50.0 %	100.0 %	0.0 %
getBooksInFile	0.0 %	0.0 %	100.0 %	0.0 %
getBooksInFile	100.0 %	100.0 %	100.0 %	100.0 %
getInstance	0.0 %	0.0 %	100.0 %	0.0 %
getPathOfFile	100.0 %	100.0 %	100.0 %	100.0 %
isFileModified	100.0 %	100.0 %	100.0 %	100.0 %
loadFile	100.0 %	50.0 %	100.0 %	0.0 %
newFile	100.0 %	100.0 %	100.0 %	100.0 %
putAppFile	100.0 %	100.0 %	100.0 %	100.0 %
removeAppModelModifyListene	100.0 %	50.0 %	100.0 %	0.0 %
removeBookFromFile	0.0 %	0.0 %	100.0 %	0.0 %
saveFile	100.0 %	50.0 %	100.0 %	0.0 %
▶ AppView	66.7 %	33.3 %	33.3 %	0.0 %
▶ Book	100.0 %	75.0 %	100.0 %	50.0 %
▶ FrameMain	90.2 %	78.9 %	0.0 %	33.3 %
▶ SimpleJavaApp	0.0 %	0.0 %	100.0 %	0.0 %

Altre viste

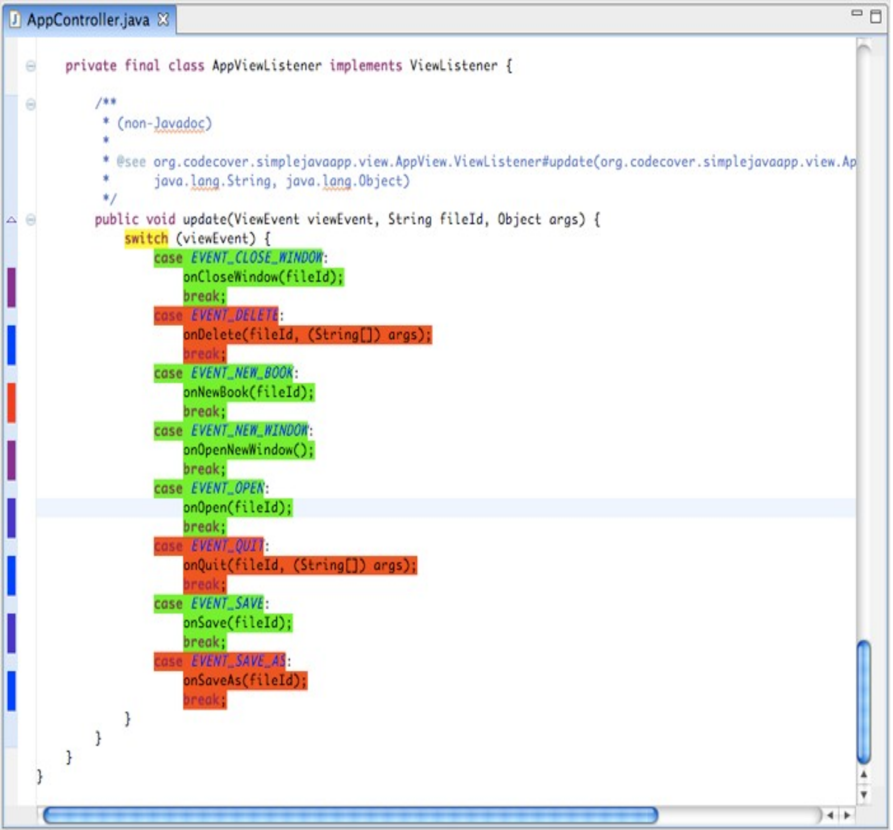


Correlation

- Test Case: "Close Window" of the Test Se
  - Test Case: "New Book; Change Book"
- Test Case: "Delete Book" of the Test Sess
  - Test Case: "New Book; Change Book"
- Test Case: "New Book; Change Book" of t
- Test Case: "New Window" of the Test Ses

	Close Wi...	Delete B...	New Book...	New Wind...	Quit	Save
Close Wi...	🟩	🟦	🟩	🟥	🟩	🟩
Delete B...	🟪	🟩	🟩	🟥	🟩	🟪
New Book...	🟪	🟦	🟩	🟥	🟩	🟪
New Wind...	🟦	🟥	🟦	🟩	🟦	🟦
Quit	🟪	🟦	🟩	🟥	🟩	🟪
Save	🟩	🟦	🟩	🟥	🟩	🟩

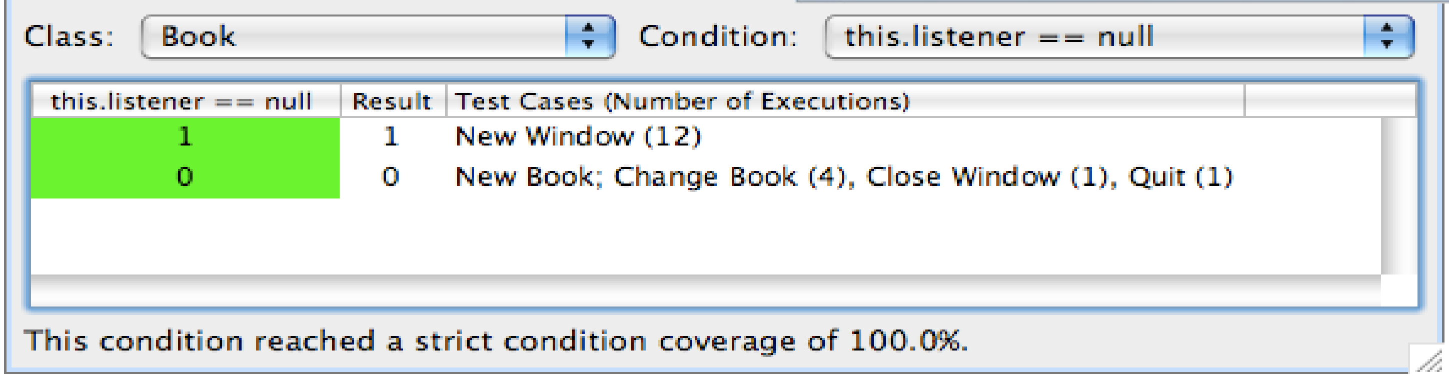
⬛ 0%
🟥 0% - 33%
🟦 33% - 66%
🟪 66% - 100%
🟩 100%



AppController.java

```
private final class AppViewListener implements ViewListener {

    /**
     * (non-Javadoc)
     *
     * @see org.codecover.simplejavaapp.view.AppView.ViewListener#update(org.codecover.simplejavaapp.view.Ap
     *      java.lang.String, java.lang.Object)
     */
    public void update(ViewEvent viewEvent, String fileId, Object args) {
        switch (viewEvent) {
            case EVENT_CLOSE_WINDOW:
                onCloseWindow(fileId);
                break;
            case EVENT_DELETE:
                onDelete(fileId, (String[]) args);
                break;
            case EVENT_NEW_BOOK:
                onNewBook(fileId);
                break;
            case EVENT_NEW_WINDOW:
                onOpenNewWindow();
                break;
            case EVENT_OPEN:
                onOpen(fileId);
                break;
            case EVENT_QUIT:
                onQuit(fileId, (String[]) args);
                break;
            case EVENT_SAVE:
                onSave(fileId);
                break;
            case EVENT_SAVE_AS:
                onSaveAs(fileId);
                break;
        }
    }
}
}
```



Boolean Analyzer

Class: Book

Condition: this.listener == null

this.listener == null	Result	Test Cases (Number of Executions)
1	1	New Window (12)
0	0	New Book; Change Book (4), Close Window (1), Quit (1)

This condition reached a strict condition coverage of 100.0%.