

CodeCover

Michel Meyer und Manuel Schwarz

7. Januar 2013

Inhalt

- 1 Einleitung
 - Motivation
 - Überblick
- 2 CodeCover allgemein
 - Einsatzzweck
 - Allgemeine Informationen
 - Installation (Eclipse)
- 3 Funktionsweise
 - Tests
 - Technische Integration
- 4 CodeCover Demo
 - Kommandozeile
 - Eclipse
- 5 Fazit

Motivation

-
-
-
-

Überblick



White- bzw. Glass-Box Testing

-
-
-
-
-
-

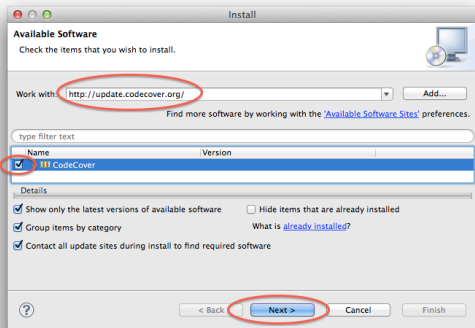
Download und Toolinfos

- Quelle: kostenlos unter codecover.org
- letzte Version (Stand: März 2011): CodeCover 1.0.1.2 (knapp 3 MB)
- Lizenz: Eclipse Public Licence (EPL)
- Plattformen: Kommandozeile (Linux, Windows, Mac OS) sowie Eclipse- und Ant-Integration
- Programmiersprachen: Java und COBOL

Installation (Eclipse)

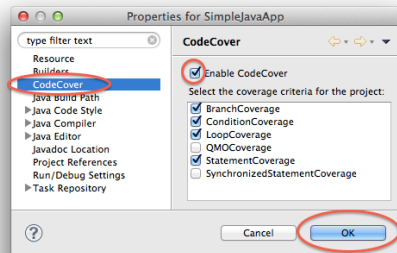
Installation

- Eclipse starten
- Help → Install New Software...
- URL:
`http://update.codecover.org/`
eingeben
- CodeCover auswählen



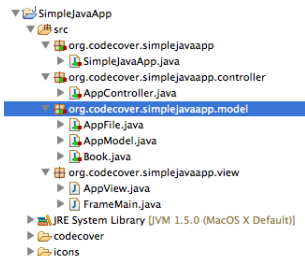
CodeCover aktivieren

- Rechtsklick auf gewünschtes Projekt
- CodeCover auswählen und aktivieren
- die gewünschten Kriterien auswählen



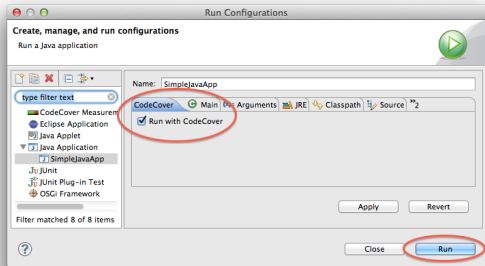
Zu prüfende Klassen auswählen

- zu testende Klassen auswählen
- Rechtsklick → Use For Coverage Measurement



Ausführen

- Run → Run Configurations...
- Run with CodeCover auswählen
- Ausführen



Testarten

CodeCover deckt folgende Tests ab:

- Bedingungsüberdeckung
- Zweigüberdeckung
- Schleifenüberdeckung
- Anweisungsüberdeckung
- Ternärer Operator Überdeckung
- Synchronisationsüberdeckung

Benutzung

- Kommandozeile (Report erstellen)
- Eclipse (verschiedene Views + Report)
- Ant
- JUnit

Farbkodierung

- grün: komplette Abdeckung
- gelb: Teilabdeckung
- rot: wird nicht evaluiert

```

AppModel.java  FrameMain.java  HelloHighlighting.java  Main.java
1  package main;
2
3  public class Main
4  {
5      public static void main(String[] args)
6      {
7          boolean a, b, c, d;
8
9          a = true;
10         b = true;
11         c = false;
12         d = false;
13
14         if (a || b)
15         {
16             System.out.println("a == true");
17         }
18
19         while ((a && b) || (c && d))
20         {
21             System.out.println("(a && b) || (c && d)");
22             a = false;
23         }
24
25         for (int t = 0; t < 100; t++)
26         {
27             if (t > 10)
28             {
29                 System.out.println("t > 10");
30             }
31
32             if (t == 100)
33             {
34                 System.out.println("t == 100");
35             }
36         }
37     }
38
39

```

einfaches Beispiel

DEMO

komplexes Beispiel

DEMO

SimpleJavaApp

Zusammenfassung und Fazit

- gute Eclipse-Integration
- einfaches Generieren und Zusammenfassen von Testfällen
- verschiedene nützliche Views in Eclipse
- keine Weiterentwicklung seit fast 2 Jahren
- nur Java und COBOL werden unterstützt
- evtl. Alternativen suchen