Programming Assignment: Checking Coverage with JaCoCo

Passed · 10/10 points

Deadline The assignment was due on Apr 30, 11:59 PM PKT
You can still pass this assignment before the course ends.

Instructions **My submissions** **Discussions**

Task

You will check and possibly extend your unit test suite to ensure a high level of coverage is achieved using JaCoCo.

An initial eclipse project is available in the .zip file below. This project contains a build.gradle file with all of the dependencies necessary to use JaCoCo. For directions in how to install gradle buildship support into Eclipse, see the "Test Doubles: Installing Gradle and Mockito" lecture.

CoffeeMaker_JaCoCo_Assign
TGZ File

CoffeeMaker_JaCoCo_Assign
ZIP File

Download and expand the .zip / .tgz file into a directory of your choice, then to import the project into Eclipse, follow the directions to import gradle projects into Eclipse:

Loading a Gradle project into Eclipse.pdf
PDF File

If you want to do things from the command line or from another IDE, you can use gradle directly starting from the build.gradle and settings.gradle file. To install gradle, follow the download and install directions from https://gradle.org/ that are appropriate for your platform. However, in this case you are on your own.

Deliverable

Your task is to measure the structural coverage achieved by your unit test suite against each of the domain classes in the CoffeeMaker example: CoffeeMaker, Inventory, Recipe, and RecipeBook using the JaCoCo tool. Follow the steps from the lecture on JaCoCo to create a gradle build with JaCoCo, and use JaCoCo to measure the coverage of your jUnit tests.

If you do not achieve full coverage of each class, add (useful!) tests until full coverage is achieved.

How will this be graded?

In order to measure the adequacy of your tests, we will be using JaCoCo and mutation analysis. We will check to ensure that your test suite achieves sufficient coverage using JaCoCo, and we will introduce five additional faulty versions of the software by adding additional mutations.

You will get 30% credit for submitting tests which accurately report a working correct implementation. You will achieve 5% additional credit for each class whose branch coverage is 90% or greater. The remaining 50% is incremental credit for each mutant your tests successfully identify as not correct.

How to submit

When you're ready to submit, you can upload files for each part of the assignment on the "My submissions" tab.

Like Dislike Report an issue

Acti