JUnitMe : JUnit tests generation with Alloy and CodeModel

Auteurs :
Salla DIAGNE
Anis TELLO

16^{th} December 2015

Table of contents

Introduction

"Every program is guilty until proven innocent"

During the development phase, developers spend lots of time to write tests corresponding to code they have written in the program.

A program with high code coverage has been more thoroughly tested and has a lower chance of containing software bugs than a program with low code coverage.[1]

But what if developers didn't have to spends hours writing tests to have a good code coverage? what if there was a magic stick that can generate a bunch of unit tests?

We have decided to find an answer to previously asked questions. And since Java is one of the most used programming languages in the world, we have decided to find a solution that would generate automatically Java unit tests. Thats would save the time of developers and would give the time and the energy to focus on working on business layer.

To realise this project we have used Spoon, Alloy , Alloy Analyzer and CodeModel.

This project is an extended version of an application developed by Valentin Lefils and Quentin Marrecau. The first version of the application has treated the same problems we are facing but only on small example of Java programs.

The rest of this report is organized as follows. Section 1.1 provides motivating examples and the goals of this work. Section 1.2 describes an overview of our tool: JUnitMe. Section 1.3 describes the algorithms. Section 1.4 explains the implementation and the architecture, section 1.5 provides an example of a use case and expected results. Section 2.1 evaluates our tool from a complexity point of view. Section 2.2 evaluates our tool from a performance point of view. Section 2.3 evaluates the ease of use of our tool. Finally, Section 3. concludes this report.

Chapter 1

Technical work

1.1 Goal

Project goal is to generate a big amount of tests for a Java program using Alloy analyser.

1.2 Overview

1.3 Algorithm

1.4 Implementation & architecture

1.5 Utilisation

Chapter 2

Evaluation

2.1 Complexity

2.2 Performance

2.3 Ease of use

[2]

Conclusion

Bibliography

[1] Wikipedia. Code coverage. `https://en.wikipedia.org/wiki/Code_coverage`.

[2] Albert Einstein. Zur Elektrodynamik bewegter Körper. (German) [On the electrodynamics of moving bodies]. *Annalen der Physik*, 322(10):891–921, 1905.