Freie Universität Berlin

Practice sheet 2 – due on 2010-11-01

Learning aims: The aim of this practice sheet is to gain first experiences with test driven development (TDD) and pair programming. It serves as preparation for the next practice sheet which asks you to actually implement test cases in a real open source project.

Task 2 – 1 : familiarize with TDD

The aim of this task is to familiarize with the process of test driven development.
You can work in pairs so that each person has to read only one of the texts.

1. Read the following sources at least to the extent that you gain a good understanding of TDD and are able to answer the questions in part b).
   If necessary, look for other sources and also read those.
   a. Scott W. Ambler: Introduction to Test Driven Development:
      http://www.agiledata.org/essays/tdd.html
      (You may skip sections 5, 9, 10, 11, 12)
   b. Frank Westphal: Testgetriebene Entwicklung:
      http://www.frankwestphal.de/TestgetriebeneEntwicklung.html

      The first source is fairly general, the second (in German) is much more precise and provides examples and is therefore longer.

2. Answer the following questions:
   a. What is the ideal TDD cycle?
      Which time interval should be aimed for?
   b. Test cases should ideally be designed according to *which* criterion?
   c. How many test cases should ideally be rewritten at the same time?
   d. Why is it not recommended to write more code as the test case demands?
   e. Why is it possible to write tests which run (rather than fail) at first go?
      Which question should then be asked?
   f. What can you assume when your test cases contain a lot of program logic of their own?
   g. What is *refactoring*?
   h. When and how does refactoring ideally play together with TDD?
   i. What is the relation between test cases produced via TDD and a specification?
   j. Ward Cunningham argues that test driven development is no test technique. What does he mean?

Practice sheet 2 – due on 2010-11-01

Task 2 – 2: first use of TDD

The aim of this task is to develop a Java program which converts natural numbers (such as 21) into spelled-out number words (such as »twenty one«). The focus is on two points:

- **Pair Programming**: Program the entire code as a pair on one computer. Look for a partner and agree on a time to develop the program. Take turns in using mouse and keyboard.
- **TDD**: Develop the *entire* code in a test-driven manner.

Now to the main task. You can find it at http://www.inf.fu-berlin.de/inst/ag-se/teaching/V-SWT2-2010/Problem.pdf (»Why Johnny Can't count«).

For you, however, the **task** will be somewhat **easier**: The input merely consists of one single positive integer.

Do not forget: **Develop all code test-driven**.

1. **Bring the executable program, source code, and all test cases to the tutorial! In case you do not possess a laptop, contact the tutor as early as possible. Prepare to present your solution and solution process.**
2. Let us now evaluate your programming session or rather prepare the tutorial. Think about the following questions.
   a. How did the development start (analysis/requirement elicitation, planning, decision for first test case, etc.)?
   b. When did you find it difficult to define new tests? Why?
   c. In which situations did you and your partner have different ideas concerning the next tests? Why?
   d. When did you change all tests again? Why?
   e. Were there situations in which (implicit) requirements were recognized too late? Why?
   f. When did you refactor? Why?