INF43: Introduction to Software Engineering

Instructor: Alex Thornton
TA: Hye Jung Choi
05/15/2009

Announcement

- Phase 3 Assignment:
  - May 27th (Wednesday) 9:00 pm

Testing

- What is testing?
  - The process consisting of all life cycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects. [ISTQB Glossary of Testing Terms]

Definitions

- Error – a human action that produces an in correct result
- ***Fault*** – the manifestation of an error
- Failure – a fault may result in a failure

Types of testing

- Black-box testing
  - Specification-based testing, functional testing
- White-box testing
  - Code-based testing, program-based testing, structural testing

- ***Unit testing***
  - based on source code
- Integration testing
  - either source code or specification
- System testing
  - specification

JUnit

- Framework for performing unit testing on Java pr ograms
- Available as a stand-alone application and built into Eclipse
  - Cppunit available for C++ (http://apps.sourceforge.net/mediawiki/cppunit/index.php?title=Main_Page)
  - httpunit for web pages (http://httpunit.sourceforge.net/)
- Framework executes the test cases and records the results
  - Displays results in a GUI

<From prof. Susan Sim's lecture notes>

Unit Testing

- A unit test typically tests one class in the system
  - A unit test suite contains many test cases
- Each test case typically tests one method in the system
- There can be many test cases for each method in the system
- Each test case either succeeds or fails, there is no gray area
- If a test case has an error, that is also a failure
- A test or test suite can be said to succeed to a certain percentage

<From prof. Susan Sim's lecture notes>

JUnit in Eclipse

1. Install Eclipse Classic 3.4.2
   - http://www.eclipse.org/downloads/
2. Change workspace
   - http://www.ics.uci.edu/~thornton/inf43/CourseProject/Implementation/43Workspace.zip
3. Create new project
4. Import source files
5. Add junit.jar to a project's build path
   - http://sourceforge.net/project/showfiles.php?group_id=15278&package_id=12472
6. Create a new JUnit TestCase wizard
7. Running a JUnit test case

JUnit in Eclipse

- Each test method consists of a sequence of steps, and some checks of the results.

- Once you have the unit tests written, you run them. You could run them directly from main(), but it is easier to use a test running utility
  - We use JUnit TestRunners.

JUnit Methods

http://junit.org/junit/javadoc/4.5/

- **assertEquals(x, y)**
  - Test passes if x and y are equal
  - x and y can be primitives or any type with an appropriate equals method
  - Three argument versions exist for floating point numbers
- **assertFalse(b)**
  - Test passes if boolean value b is false
- **assertTrue(b)**
  - Test passes if boolean value b is true
- **assertNull(o)**
  - Test passes if object o is null
- **assertNotNull(o)**
  - Test passes if object o is not null
- **assertSame(ox, oy)**
  - Test passes if ox and oy refer to the same object
- **assertNotSame(ox, oy)**
  - Test passes if ox and oy do not refer to the same object

Help

- Eclipse help
  - Help -> Java development user guide -> Getting Started -> Basic tutorial -> Writing and running JUnit tests
- JUnit cookbook
  - http://junit.sourceforge.net/doc/cookbook/cookbook.htm
- JUnit homepage
  - http://www.junit.org