Unit Tests

Bill Harlan

September, 2011

Some Assertions

- Unit tests are "tests written by developers for the benefit of developers"
- Tests are useful at all scales — for classes, packages, and entire modules
- A developer who believes tests are not useful will not write useful tests
- Make it easy. Accept all styles
- Test frameworks/harnesses should not get in the way
- Main obstacle: overuse of inheritance (tomorrow)

How to write a Unit Test

- Write a block of code that throws Exceptions or Errors when the unexpected happens
- Do not catch anything; assert everything
- Drop it in a test suite to be run frequently

Level one: Trying to Care

- Go for coverage with smoke tests
- Exercise the code and see if it blows up. Few assertions
- Avoids the embarrassment of crashing in a routine workflow
- Useful, but will not catch enough bugs, even with 100% coverage
- Motivation: Just doing your job

Level two: Fixing Bugs

- You have seen this bug before
- Reproduce the bug in a test, then fix it
- Motivation: Stay fixed already!

Level three: Protecting Your Code

- Make it hard for others to break your code
- Someone may misunderstand this and change it
- Motivation: This was not easy. Once was enough

Level four: Corner Cases

- I wonder what happens if I do this?
- You can only fix bugs you have seen
- Look for weaknesses. Do not play fair
- Now you are thinking like tester
- Motivation: Save time later (and have fun)

Level five: Example Code

- This is the right way to use this code
- Assertions show the behavior to expect
- Good coverage and free documentation
- Motivation: Helps you remember and saves explanations

Level six: Test Driven

- How do I want to be able to use this class?
- Like example code, but written before the implementation
- Try writing javadocs first too
- Motivation: This is going to be a masterpiece of design

Level seven: Reusable Tests

- How should every implementation of this interface behave?
- Reuse tricky test code – such as multi-threaded stress tests
- Upgrade many tests with each enhancement
- Motivation: Any new code better follow these same rules, or I'm in trouble

What are Test Frameworks for?

- Group tests into suites to be run at appropriate times
- Useful suites:
  - Fast and slow suites
  - All tests for one package
  - Tests requiring no other modules
  - Tests dependent on a database
  - Tests that can run under OSGi, or not
- Tests should outlive your testing framework. Minimize dependencies