Intro to JUnit

References:
Dr. Ghosh’s slides based on http://www.junit.org/
Paul Ammann & Jeff Offutt from http://www.cs.gmu.edu/~offutt/softwaretest/

What is JUnit?

• Open source Java testing framework used to write and run repeatable automated tests
• JUnit is open source (junit.org)
• A structure for writing test drivers
• JUnit features include:
  - Assertions for testing expected results
  - Test features for sharing common test data
  - Test suites for easily organizing and running tests
  - Graphical and textual test runners
• JUnit is widely used in industry
• JUnit can be used as stand alone Java programs (from the command line) or within an IDE such as Eclipse

JUnit Tests

• JUnit can be used to test ...
  - ... an entire object
  - ... part of an object – a method or some interacting methods
  - ... interaction between several objects
• It is primarily for unit and integration testing, not system testing
• Each test is embedded into one test method
• A test class contains one or more test methods
• Test classes include:
  - A test runner to run the tests (main())
  - A collection of test methods
  - Methods to set up the state before and update the state after each test and before and after all tests
• Get started at junit.org

Using JUnit

• Integrated with Eclipse – several advantages
  - Environment variables set up automatically
  - Execution tied with the Eclipse debugger
  - Test driver skeleton generated – just fill in test cases

Steps for using JUnit

• Simple framework to write repeatable tests.
• Write a test:
  - Import junit.framework *
  - Extends class junit.framework.TestCase
  - Modify the following methods:
    - protected void setUp() – set up the fixture of the test.
    - protected void tearDown() – release resources allocated in setUp().
• Write test methods for the test case (method name should begin with test):
  public void testMoney(){...
• Run the test using three different test runners:
  - java.junit.awtui.TestRunner ClassName
  - java.junit.swingui.TestRunner ClassName
  - java.junit.textui.TestRunner ClassName

Writing test cases in JUnit (Eclipse)

• Right-click on the ProjectWithJUnit title
• Select New -> Other
• Expand the "Java" selection, and choose JUnit.
• On the right column of the dialog, choose Test Case
• Click Next.
• Run the JUnit file as an application/debug mode.
public class Calc {
    public long add(int a, int b) {
        return a + b;
    }
}

import org.junit.Test;
import static org.junit.Assert.*;

public class calcTest {
    private Calc calc;
    @Test public void testAdd() {
        calc = new Calc();
        assertEquals((long) 5, calc.add(2, 3));
    }
}

import org.junit.runner.RunWith;
import org.junit.runners.Suite;
import junit.framework.JUnit4TestAdapter;

@Suite.Runner (Suite.class)
@Suite.SuiteClasses({StackTest.class}) // Add test classes here.
public class AllTests {
    public static void main(String[] args) {
        junit.textui.TestRunner.run(suite());
    }
    public static junit.framework.Test suite() {
        return new JUnit4TestAdapter(AllTests.class);
    }
}
Results

What to do and not to do

• Do's:
  ▪ Create new objects using constructors that are known to be correct.
  ▪ Use the equals() method if it is known to be correct.

• Don’ts:
  ▪ Suppose you want to test method foo(). **DO NOT** use methods bar() and
    foo() in the same test case if bar() is not known to be correct.
  ▪ If the test failed, you don’t know if it is because of foo() or bar().