Topics in Unit Testing Tools

Java Testing Tools

- junit is a testing harness for unit testing.
- emma is a code coverage tool.
- The tools can be used in concert to provide statement and branch coverage reports during the unit testing process.

Drexel UNIVERSITY

The junit Unit Testing Tool for Java

- Home page:
  - http://www.junit.org
- Documentation and tutorial:
  - http://junit.org/junit/javadoc/4.5/
- Download page for version 4.7:
  - http://sourceforge.net/projects/junit/files/junit/4.7/junit-4.7.jar/download
- junit FAQ page (has answers to most user questions):
  - http://junit.sourceforge.net/doc/faq/faq.htm

Drexel UNIVERSITY

How to install junit

(from the www.junit.org website)

- Download junit
  - *i.e.,* a file called **junit.zip**
- Unzip the **junit.zip** to a directory (junit home):
  - Windows users:
    - %JUNIT_HOME%
    - Add junit to the class path:
      - set CLASSPATH=%CLASSPATH%;%JUNIT_HOME%\junit-4.7.jar
  - Unix users (bash shell):
    - $JUNIT_HOME.
    - Add junit to the class path:
      - export CLASSPATH=$CLASSPATH:$JUNIT_HOME/junit-4.7.jar

Drexel UNIVERSITY

How to test the junit installation

- Test the installation by running the sample tests distributed with junit.
- The sample tests are located in the installation directory, not the **junit.jar** file.
- Then type:
  - java org.junit.runner.JUnitCore org.junit.tests.AllTestsAll
  - Later, just replace *org.junit.tests.AllTestsAll* with your test code name.
- The tests should pass with an "OK" message.
  - If the tests don't pass, verify that junit-4.7.jar is in the CLASSPATH.

Drexel UNIVERSITY

junit Examples

- In your junit distibution go to:
  - junit-4.7/junit/samples
- Examine the code of:
  - SimpleTest.java
  - ListTest.java
  - AllTests.java
  - money/MoneyTest.java
- Run the tests and use them as a template for your own test cases.

Drexel UNIVERSITY

Simple Test Suite

```
package junit.samples;

import junit.framework.Test;
import junit.framework.TestCase;
import junit.framework.TestSuite;

/* Some simple tests. */
public class SimpleTest extends TestCase {
        protected int fValue1;
        protected int fValue2;

        @Override
        protected void setUp() {
                fValue1= 2;
                fValue2= 3;
        }
        public static Test suite() {
                        /* the dynamic way */
            return
                new TestSuite(SimpleTest.class);
```

```
public void testAdd() {
        double result= fValue1 + fValue2;
        // forced failure result == 5
        assertTrue(result == 6);
 }
 public void testDivideByZero() {
        int zero= 0;
        int result= 8/zero;
        result++;
}
 public void testEquals() {
        assertEquals(12, 12);
        assertEquals(12L, 12L);
        assertEquals(new Long(12), new Long(12));
        assertEquals("Size", 12, 13);
        assertEquals("Capacity", 12.0, 11.99, 0.0);
}
 public static void main (String[] args) {
        junit.textui.TestRunner.run(suite());
 }
} /* SimpleTest */
```

List Test Suite

```
import java.util.ArrayList;
import java.util.List;
import junit.framework.Test;
import junit.framework.TestCase;
import junit.framework.TestSuite;

public class ListTest extends TestCase {
        protected List<Integer> fEmpty;
        protected List<Integer> fFull;

        public static void main (String[] args) {
                junit.textui.TestRunner.run (suite());
        }
        @Override
        protected void setUp() {
                fEmpty= new ArrayList<Integer>();
                fFull= new ArrayList<Integer>();
                fFull.add(1);
                fFull.add(2);
                fFull.add(3);
        }
```

```
public static Test suite() {
    return new TestSuite(ListTest.class);
}

public void testCapacity() {
    int size= fFull.size();
    for (int i= 0; i < 100; i++)
        fFull.add(new Integer(i));
    assertTrue(fFull.size() == 100+size);
}

public void testContains() {
    assertTrue(fFull.contains(1));
    assertTrue(!fEmpty.contains(1));
}

// continued on next slide …
```

List Test Suite (Cont'd)

```
    public void testElementAt() {
        int i= fFull.get(0);
        assertTrue(i == 1);

        try {
            fFull.get(fFull.size());
        } catch (IndexOutOfBoundsException e) {
            return;
        }
        fail("Should raise an ArrayIndexOutOfBoundsException");
    }
    public void testRemoveAll() {
        fFull.removeAll(fFull);
        fEmpty.removeAll(fEmpty);
        assertTrue(fFull.isEmpty());
        assertTrue(fEmpty.isEmpty());
    }
    public void testRemoveElement() {
        fFull.remove(new Integer(3));
        assertTrue(!fFull.contains(3));
    }
}
```

Setting up Composite Test Suites

```
package junit.samples;

import junit.framework.Test;
import junit.framework.TestSuite;

/* TestSuite that runs all the sample tests
 */
public class AllTests {
	public static void main (String[] args) {
		junit.textui.TestRunner.run (suite());
	}
	public static Test suite ( ) {
		TestSuite suite= new TestSuite("All JUnit Tests");
		suite.addTest(ListTest.suite());
		suite.addTest(new TestSuite(junit.samples.money.MoneyTest.class));
		suite.addTest(junit.tests.AllTests.suite());
		return suite;
	}
}
```

The Emma Code Coverage Tool

- Home page:
  - http://emma.sourceforge.net/
- Quick start introduction page:
  - http://emma.sourceforge.net/intro.html
- Download page:
  - http://sourceforge.net/projects/emma/files/
- Emma sample reports page:
  - http://emma.sourceforge.net/samples.html
- Emma FAQ page:
  - http://emma.sourceforge.net/faq.html

Drexel UNIVERSITY

Downloading and Installing Emma

1. Go to the download page:
   - http://sourceforge.net/projects/emma/files/
2. Get the file **emma-2.0.5312-lib.zip** in the **emma-release** directory.
   - The download starts automatically after a short sourceforge advertisement.
3. Unzip the **emma-2.0.5312-lib.zip** file and put the file **emma.jar** in the directory with the code.
   - You can put the file elsewhere as long as you include the directory in the java class path using the "–cp" option.

Drexel UNIVERSITY

How to instrument Java byte code using Emma

- Assume that **emma.jar** and **junit-4.7.jar** are both in the same directory that contains the MyCode byte code (.class files).
- Assume your CLASSPATH contains:
  - emma.jar
  - junit-4.7.jar
- To instrument the byte code run:
  - java emma instr –m overwrite -cp . MyCode
  - java -cp . MyCode
- To execute the instrumented byte code run:
  - java MyCode
- Files **coverage.em** and **coverage.ec** store code coverage data.

Drexel UNIVERSITY

How to generate an HTML code coverage report

- To generate an HTML code coverage report run:
  - java emma report -r html -sp . –in coverage.em,coverage.ec
- The **-sp** option is used to specify the source code directory.
  - Use **-sp .** if the source code is in the same directory as the byte code.
  - Relative paths may be used.
  - Without the **-sp** option, the report will not show the source code view.

Drexel UNIVERSITY

Running code without Emma

- Once you are done running instrumented code you must re-compile the java source into plain (non-instrumented) byte code.
  - javac *.java

Drexel UNIVERSITY

Other tools

- CHECK is a tool for unit testing of C code
  - http://check.sourceforge.net/doc/check.html/index.html
- GCC's GCOV can be used for C code coverage
- Valgrind can be used for memory analysis for C code
  - http://valgrind.org/
- Jconsole can be used for memory analysis of Java code
  - http://freshmeat.net/projects/jconsole/
- For bug tracking you might want to use Trac
  - http://trac.edgewall.org

Drexel UNIVERSITY