Junit & Ant

Suneesh VR

Importance of the test framework

“Any program feature without an automated test simply doesn’t exists “

- Software bugs have enormous cost :time,money and frustrations.
- Continuously executing test cases –a practical approach to address software bugs

Junit

- Java open source project which offers an extremely useful framework for unit testing
- You can easily run your unit test again and again (regression testing)
- You have a "framework" that facilitates the testing--it might actually run the test automatically whenever you build (i.e. compile) or deploy your application

Downloads

- download Junit from http://www.junit.org. You must put junit.jar into ANT_HOME/lib so that Ant can find it. Because of Ant class loader issues, you must have junit.jar in the system classpath or ANT_HOME/lib;

Example

- public class A{
- public A(){}
- public int sum(){
- return 2;
- }
- }

Testing- conventional method

- Conventional method for testing
- A a = new A();
- if(a.sum()!=2){
- System.out.println(“Error occurred ….”);
- }

Disadvantages of conventional method

- The user has to look at the console o/p and decide whether it is correct or not
- No method to collect the results in a structured fashion
- After each run a person has to examine and interpret the results .

Using Junit

- Testing by Junit
- assertsEquals(“Message”,2,a.sum());
- assertsEquals(“Message”,1,a.sum());
- testSomething(TestA) junit.framework.AssertionFailedError: expected:<1> but was:<2>

Junit –Writing Test classes

- Put your tests in a class that extends the JUnit-class "TestCase".
- If your test cases use some common data then set it up in a method called "setUp".
- Place the testcode (e.g. calls to"assertEquals") in one or more methods having names starting with "test".
- method tearDown for releasing resources allocated in setup.

Junit –Asserts methods

- assertEquals(“Message”,a,b)
- asserts that the two parameters are equal. a and b must be either the same primitive type or both Objects
- assertTrue(“Message”,boolean)
- asserts that a given condition is true
- assertNull(“Messsage”,Object)
- asserts that an object is null
- assertSame(“Message”,Object, Object)
- asserts that two objects references the same object

Junit –directory structure

- Well organized directory structure is essential
- Test code should be separate from production code.
- Use package hierarchy

Junit –suite method

- If you only want some of your tests to be run you should define a static method called "suite" and let it return a "TestSuite", which defines the tests to be run.
- public static Test suite() {
- TestSuite suite= new TestSuite(); suite.addTest(new TestA("testMethod1")); return suite;
- }

Junit-suite method

- public static Test suite() {
- TestSuite suite= new TestSuite();
  suite.addTest(new TestA("testMethod1"));
  suite.addTest(new TestA("testMethod2"));
- return suite;

}

- public static Test suite() {
- TestSuite suite= new TestSuite();
- suite.addTest(TestA.suite());
- suite.addTest(TestB.suite());
- return suite;

}

Junit –Best practices

- Do not use the test-case constructor to set up a test case
  Avoid writing test cases with side effects

1. They can affect data that other test cases rely upon
2. You cannot repeat tests without manual intervention

- Do not load data from hard-coded locations on a filesystem
  FileInputStream inp ("C:\\TestData\\dataSet1.dat");
  ……..

Junit-Best practices

1. A tester does not have room to store the test data on C:
2. The tests run on another platform, such as Unix

- When ever bugs comes add methods to test cases

Ant

- Cross platform ,extensible,simple and fast build tool
- Ant integrates with Junit to allow executing test suites as a part of the build process
- Capturing their output and generates color enhanced reports

Advantages –Junit & Ant

- Increase productivity
- Test case can also be used as a documentation

Extensions to Junit

- HttpUnit -A test framework that could be embedded in JUnit tests to perform automated web site testing.
- JUnitPerf JUnit -test decorators to perform scalability and performance testing.
- Mock Objects -Allows testing of code that accesses resources such as database connections and servlet containers without the need of the actual resources.
- Cactus In-container unit testing.
- DBUnit -Sets up databases in a known state for repeatable DB testing.

Resources

- http://developer.java.sun.com/developer/Books/javaprogramming/ant/ant_chap04.pdf
- http://javaboutique.internet.com/tutorials/UnitTesting/index.html
- http://www.javaworld.com/javaworld/jw-12-2000/jw-1221-junit.html