CS 580 Client-Server Programming
Spring Semester, 2009
Doc 2 Source Control & Testing
Jan 27, 2009

Testing & Subversion References

JUnit Cookbook http://junit.sourceforge.net/doc/cookbook/cookbook.htm

JUnit Test Infected: Programmers Love Writing Tests http://junit.sourceforge.net/doc/testinfected/testing.htm

JUnit Javadoc: http://www.junit.org/junit/javadoc/3.8/index.htm, http://junit.org/junit/javadoc/4.5/

JUnit FAQ, http://junit.sourceforge.net/doc/faq/faq.htm

Brian Marick's Testing Web Site: http://www.exampler.com/testing-com/

Testing for Programmers, Brian Marick, Available at: http://www.exampler.com/testing-com/writings.html

Main Subversion Website, http://subversion.tigris.org/

Version Control with Subversion, Collins-Sussman, Fitzpatrick, Pilato, http://svnbook.red-bean.com/

Source Control

Why Use Source Control?

Three Free & Common Source Control Systems

CVS

Concurrent Versions System

Command line interface in Unix

Various interfaces in Window

Subversion

Claims to be a better CVS

Many commands are same as CVS

Git

Created by Linus Torvald

Distributed Version control

Claims to be better model of development than Subversion

Subversion

http://subversion.tigris.org/

Runs on:

- Mac OS X
- Unix
- Linux
- Windows

GUI interface

Eclipse plugins

Can use locally with no server

Basic Source Control Operations

Starting a new project
Adding code to a project
Modifying existing code
Retrieving past versions of code
Handling conflicts in code
Creating code branches
Merging code branches

Command Line Subversion

Creating a Repository

svnadmin create pathToRepository

Example

svnadmin create /Users/whitney/test/repository

Note this has already been done for you on Bismarck

Starting a New Project

svn import existingDirectory urlToRepository

Local Repository

svn import test file:////Users/whitney/test/repository/assignment1

Remote Repository

svn import test svn://bismarck.sdsu.edu:8009/09/580/whitney/assignment1

Contents of directory test are placed in the repository under assignment1. Bismarck is not using the standard port for svn. You will be prompted for an import comment unless you use the -m flag. The first time you connect to a remote server you will be asked for your username and password. This information is cached locally so you will be be asked the second time.

Looking at the Repository

Explicit Repository

svn ls svn://bismarck.sdsu.edu:8009/09/580/instructor

assignment1/

please

testAssignment/

svn ls svn://bismarck.sdsu.edu:8009/09/580/instructor/assignment1

assignment1/

foo.java

Implicit Repository

svn ls

Need to be in directory of working svn project

ls is short for list

Checking out

svn co svn://bismarck.sdsu.edu:8009/09/580/whitney/assignment1

Warning

After importing files checkout the files

Work on the files you checkout

co is the short cut for checkout

Adding Files

Create a file called client.java using an editor

```
svn add client.java
     A         client.java

svn commit -m "added client.java"
     Adding         client.java
Transmitting file data .
Committed revision 4.
```

Accessing the log data

svn log

svn -r 4 log

```
------------------------------------------------------------------------
r4 | whitney | 2009-01-26 20:43:48 -0800 (Mon, 26 Jan 2009) | 1 line

added client.java
------------------------------------------------------------------------
```

svn log svn://bismarck.sdsu.edu:8009/09/580/whitney

You get the log for all projects in the repository

svn log svn://bismarck.sdsu.edu:8009/09/580/whitney/assignment1

You get the log for all files in assignment1

Comparing Files

```
svn diff client.java
    Index: client.java
    ===================================================================
    --- client.java  (revision 5)
    +++ client.java      (working copy)
    @@ -1,2 +1,3 @@
     this is the client
     add a change
    +another change

svn diff -r 4:5 client.java
    Index: client.java
    ===================================================================
    --- client.java  (revision 4)
    +++ client.java      (revision 5)
    @@ -1 +1,2 @@
     this is the client
    +add a change
```

Local Copy Verse Repository Copy

Why your copy of project files may differ from repository

You make changes to local copy
Someone else updates repository

Commands for Your Local Changes

svn status

Print the status of working copy files and directories

svn diff

This displays the differences between two revisions or paths

svn revert

Undo all local edits

Command for Syncing with Repository

svn update

Update your working copy

svn resolve

Resolve conflicts on working copy files or directories

Unit Testing

Testing

Johnson's Law

If it is not tested it does not work

The more time between coding and testing

More effort is needed to write tests
More effort is needed to find bugs
Fewer bugs are found
Time is wasted working with buggy code
Development time increases
Quality decreases

Unit Testing

Tests individual code segments

Automated tests

What wrong with:

Using print statements

Writing driver program in main

Writing small sample programs to run code

Running program and testing it be using it

We have a QA Team, so why should I write tests?

When to Write Tests

First write the tests

Then write the code to be tested

Writing tests first saves time

- Makes you clear of the interface & functionality of the code

- Removes temptation to skip tests

What to Test

Everything that could possibly break

Test values

- Inside valid range
- Outside valid range
- On the boundary between valid/invalid

GUIs are very hard to test

- Keep GUI layer very thin
- Unit test program behind the GUI, not the GUI

Common Things Programs Handle Incorrectly

Adapted with permission from “A Short Catalog of Test Ideas” by Brian Marick,

http://www.testing.com/writings.html

Strings

Empty String

Collections

Empty Collection

Collection with one element

Collection with duplicate elements

Collections with maximum possible size

Numbers

Zero

The smallest number

Just below the smallest number

The largest number

Just above the largest number

XUnit

Free frameworks for Unit testing

SUnit originally written by Kent Beck 1994

JUnit written by Kent Beck & Erich Gamma

Available at: http://www.junit.org/

Ports to many languages at:

http://www.xprogramming.com/software.htm

Sample JUnit 4.x Example

```
import static org.junit.Assert.*;
import java.util.ArrayList;
import org.junit.Before;
import org.junit.Test;

public class HelloWorldTest {
	int testValue;
	@Test
	public void testMe() {
		assertEquals(1, testValue);
	}

	@Test
	public void foo() {
		assertTrue(2 == testValue);
	}
```

```
@Test(expected=IndexOutOfBoundsException.class)
   public void testIndexOutOfBoundsException() {
      ArrayList emptyList = new ArrayList();
      Object notValid = emptyList.get(0);
   }

      @Before
      public void initialize(){
            testValue = 1;
      }
}
```

JUnit Example - JUnit 3.x

Goal: Implement a Stack containing integers.

Tests:

Subclass junit.framework.TestCase

Methods starting with 'test" are run by TestRunner

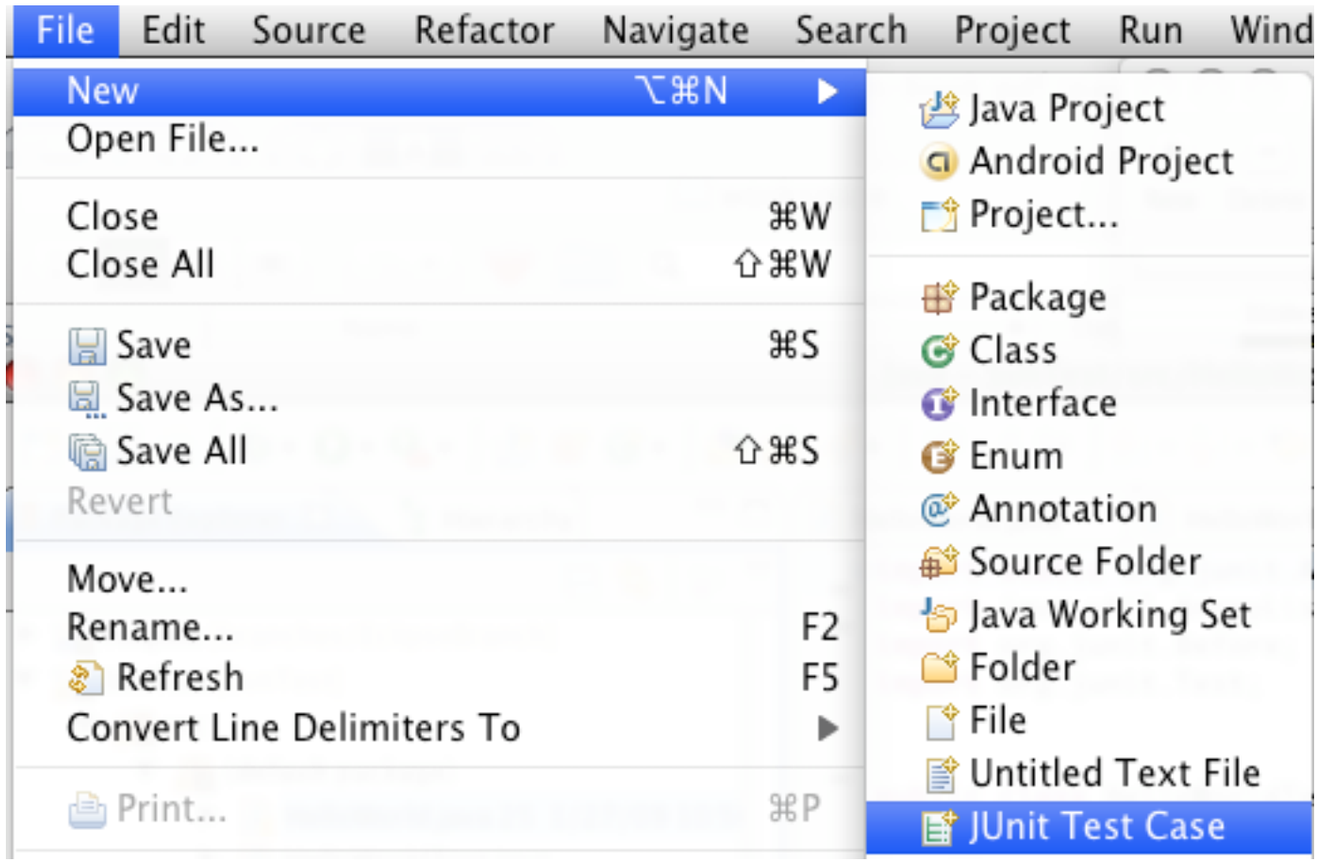
```
import junit.framework.*;
public class  TestStack extends TestCase {

        public void testDefaultConstructor() {
                Stack test = new Stack();
                assertTrue("Default constructor", test.isEmpty() );
        }

        public void testSizeConstructor() {
                Stack test = new Stack(5);
                assertTrue( test.isEmpty() );
        }
}
```

Start of Stack Class

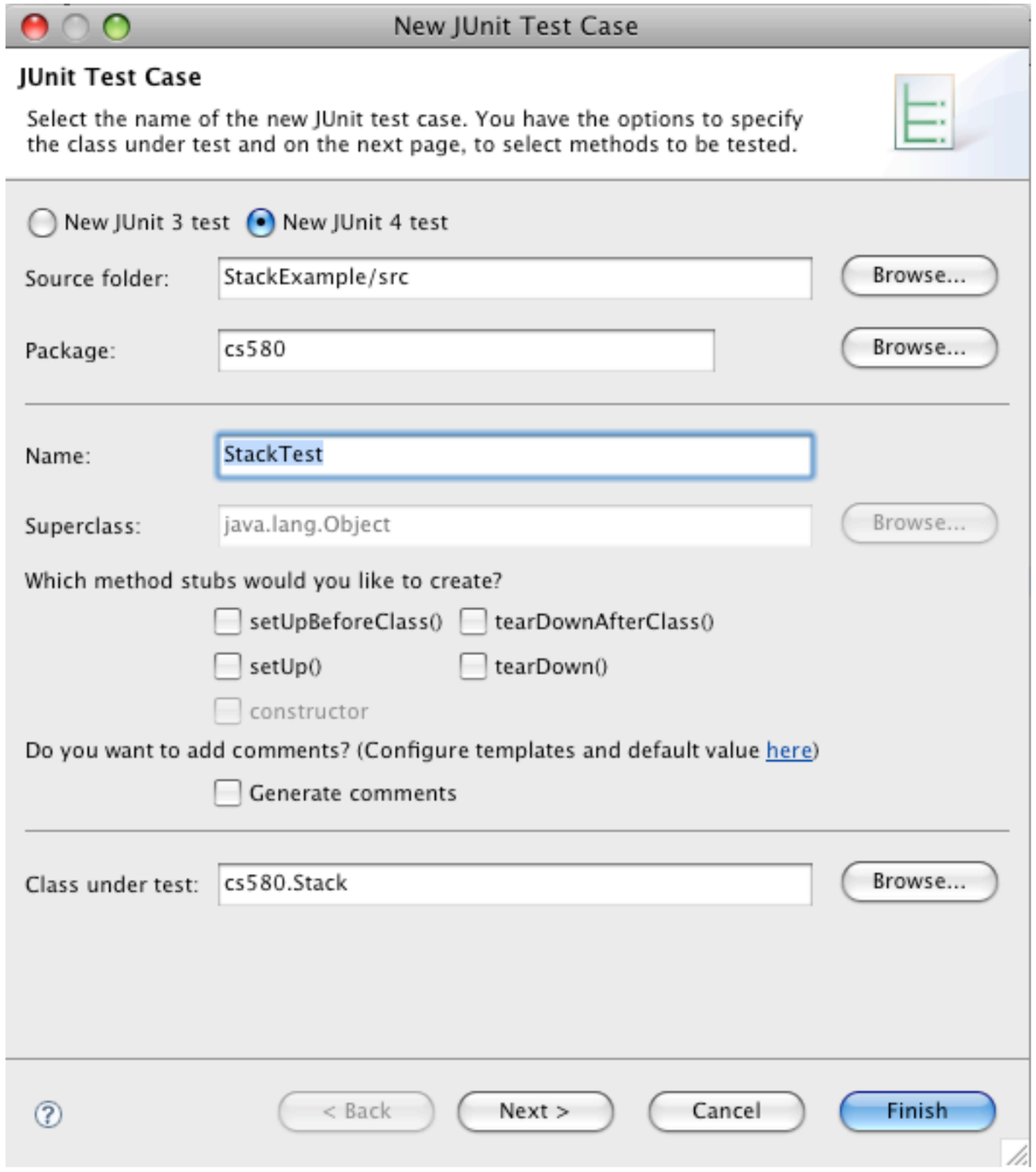
```
public class Stack  {
        int[] elements;
        int topElement = -1;

        public Stack() {
         this(10);
        }

        public Stack(int size) {
                elements = new int[size];
        }

        public boolean isEmpty() {
                return topElement == -1;
        }
}
```

Running JUnit Using Eclipse

Creating Test Case



Running JUnit Using Eclipse



Fill in dialog window & create the test cases

Select Junit test case from the "Run as..." menu

Assert Methods

assertTrue()
assertFalse()
assertEquals()
assertNotEquals()
assertSame()
assertNotSame()
assertNull()
assertNotNull()
fail()

For a complete list see

http://www.junit.org/junit/javadoc/3.8/index.htm/allclasses-frame.html/junit/junit/framework/Assert.html/Assert.html

Testing the Tests

If can be useful to modify the code to break the tests

```
package example;

public class Stack  {
	int[] elements;
	int topElement = -1;

	etc.

	public boolean isEmpty() {
		return topElement == 1;
	}
}
```

Test Fixtures - JUnit 3.x

Before each test setUp() is run

After each test tearDown() is run

```
package example;
import junit.framework.TestCase;

public class  StackTest extends TestCase {
        Stack test;

        public void setUp() {
                test = new Stack(5);
                for (int k = 1; k <=5;k++)
                        test.push( k);
        }

        public void testPushPop() {
                for (int k = 5; k >= 1; k--)
                        assertEquals( "Pop fail on element " + k,  test.pop() , k);
        }
}
```

Testing Exceptions - JUnit 3.x

```
public void testIndexOutOfBoundsException() {

    ArrayList list = new ArrayList(10);
    try {
        Object o = list.get(11);
        fail("Should raise an IndexOutOfBoundsException");
    } catch (IndexOutOfBoundsException success) {}
}
```

Example is from the JUnit FAQ