软件测试方法和技术实践

第1章 单元测试

赵钦佩

qinpeizhao@tongji.edu.cn

http://sse.tongji.edu.cn/zhaoqinpei/

1.8 JUnit是单元测试神器吗？

JUnit是一个开源的Java编程语言的单元测试框架，在代码驱动单元测试框架家族中无疑是最为成功的一例。

好处：

- **提高开发速度：**测试是以自动化方式执行的，提升了测试代码的执行效率
- **提高软件代码质量：**使用小版本发布至集成，便于除错。同时引入重构概念，让代码更干净和富有弹性
- **提升系统的可信赖度：**是回归测试的一种。支持修复或更正后的“再测试”，可确保代码的准确性。

参考: http://junit.sourceforge.net/doc/cookbook/cookbook.htm

1.8 JUnit–测试框架

JUnit 是针对JAVA语言进行单元测试的一个简单、灵活、易于使用的开源框架。提供了：

- 写测试用例和测试套件的类(class)
- 设置(setUp)和清除(tearDown)测试数据的**方法**（测试固件-test fixture）
- 设置断言(assertion)的**方法**
- 运行测试中的文字及图形显示**工具**

The test fixture is everything we need to have in place to exercise the SUT.

一个***测试用例*(test case)**可以是某个方法的单个测试

一个***测试套件*(test suite)**是多个测试用例的集合

1.8 JUnit-测试框架

JUnit 能区分**失效(failure)**和**错误(error)**

- **失效：**一般由单元测试使用的断言方法判断失败引起的，表示在测试点发现了问题。
- **错误：**是由代码异常引起，是测试目的之外的发现，表示可能产生于测试代码本身的错误（测试代码也是代码，无法保证完全没有问题），也可能是被测试代码中隐藏的bug。

1.8 断言的方法

method	description
assertEquals()	进行等值比较
assertFalse()	进行boolean值比较
assertTrue()	进行boolean值比较
assertNull()	比较对象是否为空
assertNotNull()	比较对象是否不为空
assertSame()	对2个对象应用的内存地址进行比较
assertNotSame()	对2个对象应用的内存地址进行比较
fail()	引发当前测试失败，通常用于异常处理

1.8 JUnit 3.x v.s. JUnit 4.x

JUnit 3.x 和JUnit 4.x 的区别

- JDK: JUnit 4 需要Java 5以上；JUnit 3能在JDK 1.2+ 版本上工作
- 包(package)：JUnit 4 的包为org.junit； JUnit 3 的包为junit.framework
- 测试用例类：JUnit 4 的测试用例的方法可以是任何命名方式，只需在方法前面标注*@Test*；JUnit 3 需要将测试用例的方法命名以test开头
- 设置和清除数据： JUnit 4 用*@Before*和*@After*标识； JUnit 3用*setUp()*和*tearDown()*方法
- ……

最大的改进
是
annotation

1.8 JUnit 3.x 代码样例

```
import junit.framework.*;
public class MoneyTest extends TestCase {
    private Money f12CHF;              // fixtures
    private Money f14CHF;

    protected void setUp() {       // create the test data
        f12CHF = new Money(12, "CHF");
        f14CHF = new Money(14, "CHF");
    }
  void testAdd() {                 // create the test data
        Money expected = new Money(26, “CHF”);
        assertEquals(“amount not equal”,
                          expected,f12CHF.add(f14CHF);
    }
    ...
}
```

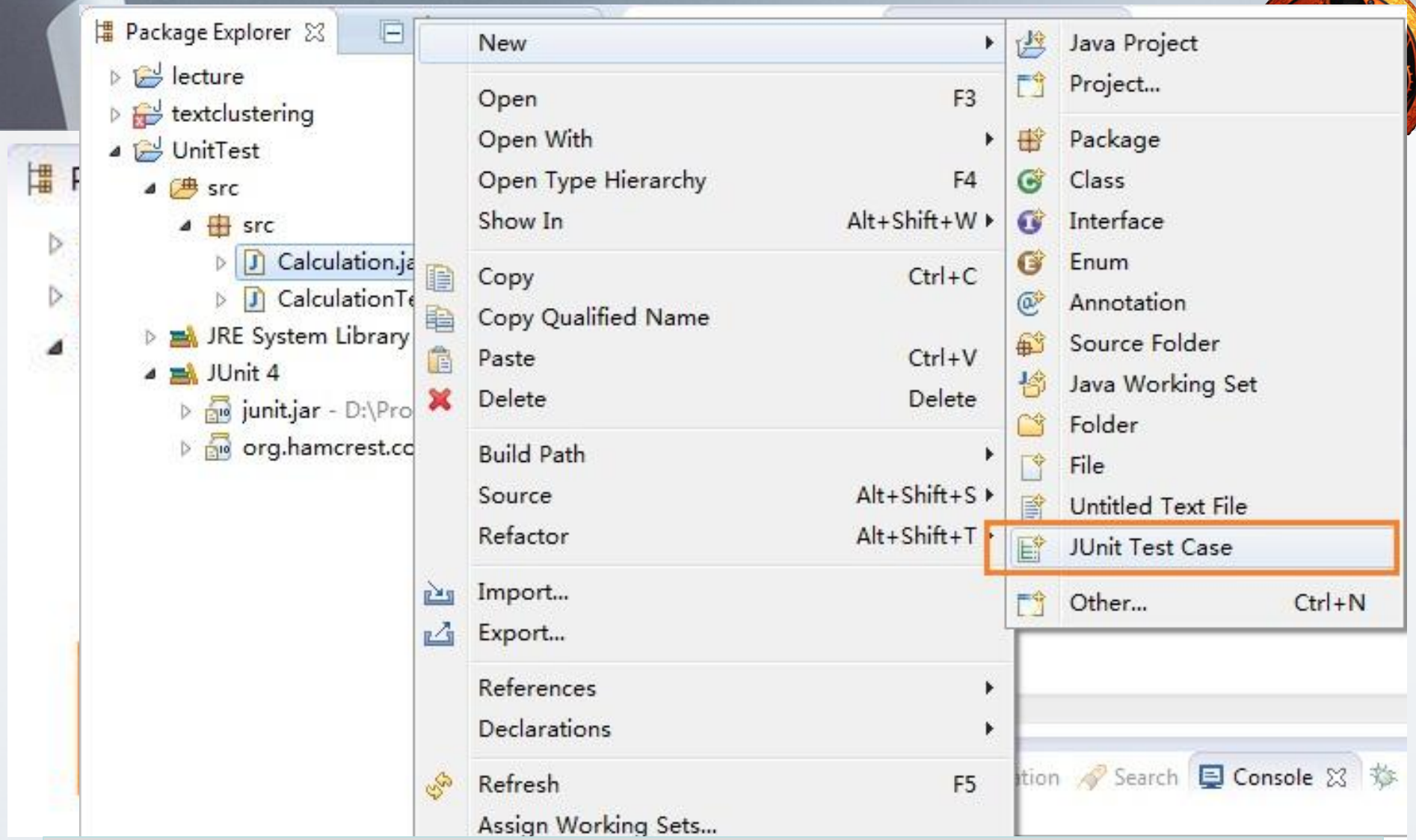
1.8 JUnit 4.x 代码样例

```
import junit.framework.*;
import org.junit.*;
import static org.junit.Assert.*;
public class MoneyTest extends TestCase {
    private Money f12CHF;
    private Money f14CHF;

    @Before
     public void setUp() {            // create the test data
        f12CHF = new Money(12, "CHF");    // - the fixture
        f14CHF = new Money(14, "CHF");
    }
    @Test
    public void add() {              // create the test data
        Money expected = new Money(26, “CHF”);
        assertEquals(“amount not equal”,
                        expected,f12CHF.add(f14CHF));
    }
    ...
}
```

1.8 基于JUnit的单元测试

```
public class Calculation {
	public int add (int a, int b)
	{
		return a+b;
	}
	public int substract ( int a, int b)
	{
		return a-b;
	}
	public int divide (int a, int b) throws Exception
	{
		if(b == 0)
		    throw new Exception();
		 return a/b;
	}
}
```



- *Create JUnit Tests in Eclipse*
  - *Right click on the Calculation.java file, and select Tools -> Create JUnit tests, following the steps by the wizard*

1.8 基于JUnit的单元测试

```
public class CalculationTest {
    public CalculationTest() { }
    @BeforeClass
    public static void setUpClass()  throws Exception {
        System.out.println();
    }
    @AfterClass
    public static void tearDownClass()  throws Exception {
    }
    @Before
    public static void setUp() {
        System.out.println("before a case");
    }
    @After
    public static void tearDown() {
        System.out.println("after a case");
    }
……
}
```

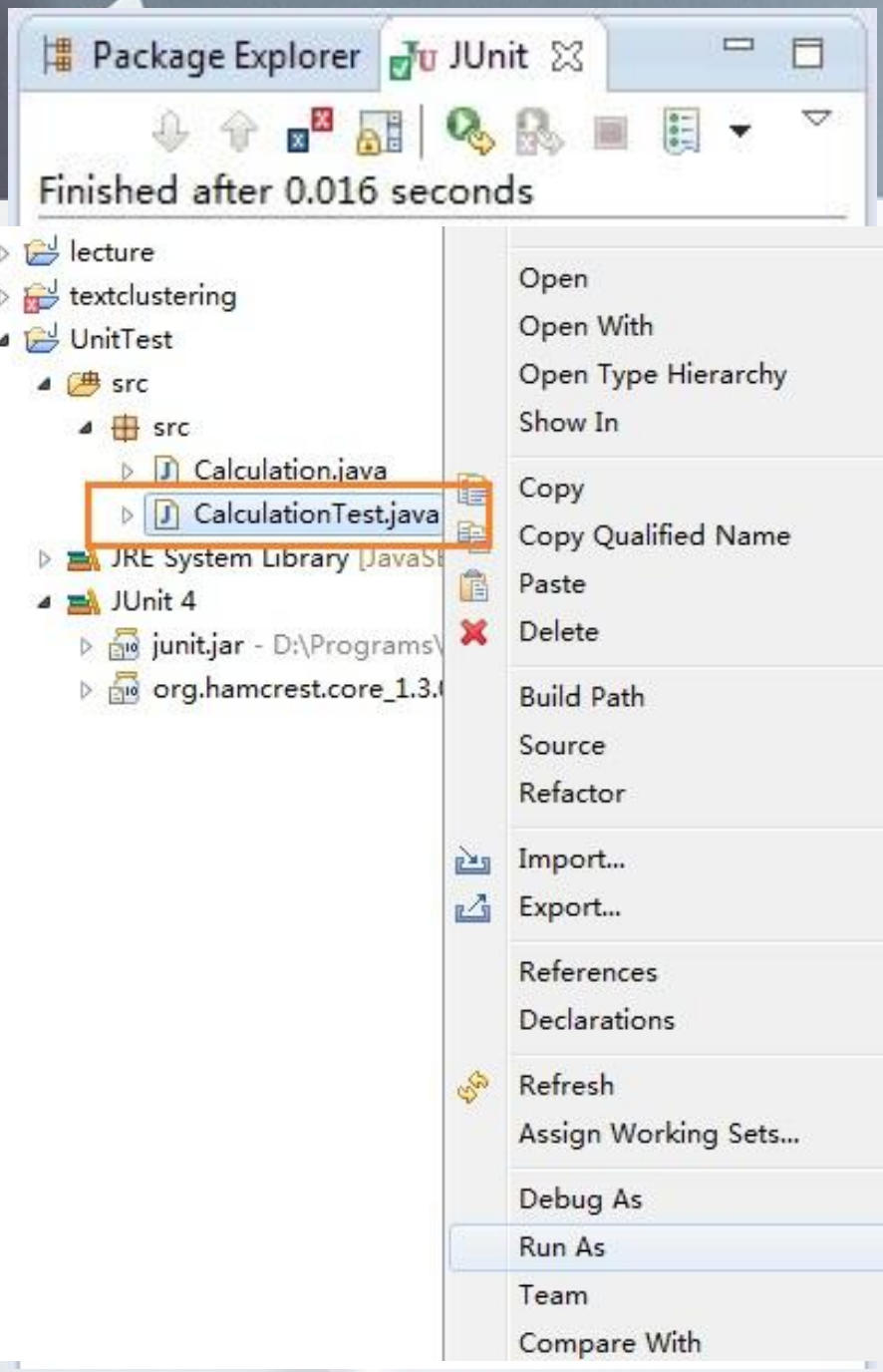
1.8 测试用例

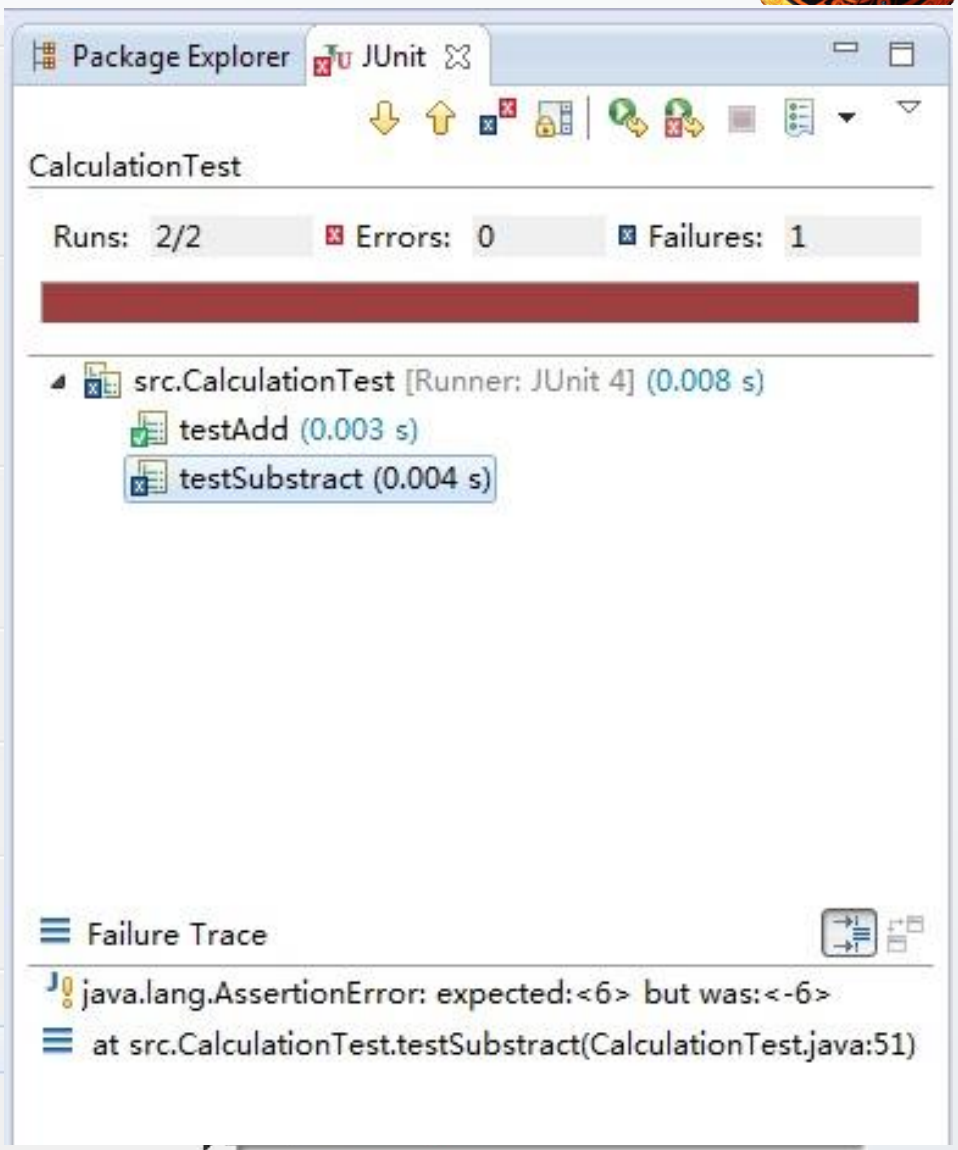
```
public class CalculationTest {
    @Test (expected = Exception.class)
    public void Divide() throws Exception
    {
        int a = 1;
        int b = 0;
        Calculation instance = new Calculation();
        instance.divide(a, b);
    }
}
```

```
public class CalculationTest {
  @Test (timeout = 1000)
    public void testAdd{
    System.out.println(“add”);
    int a = 2;
    int b = 3;
                                  new Calculation();


                                .add(a, b);
                                Result, result);


                                act(){
                                n(“substract”);
            int b = 2;
            Calculation instance = new Calculation();
            int expResult = 6;
            int result = instance.substract(a, b);
            assertEquals(expResult, result);
    }
}
```


Package Explorer
JUnit
Finished after 0.016 seconds
lecture
textclustering
UnitTest
src
src
Calculation.java
CalculationTest.java
JRE System Library [JavaSt
JUnit 4
junit.jar - D:\Programs\
org.hamcrest.core_1.3.
Open
Open With
Open Type Hierarchy
Show In
Copy
Copy Qualified Name
Paste
Delete
Build Path
Source
Refactor
Import...
Export...
References
Declarations
Refresh
Assign Working Sets...
Debug As
Run As
Team
Compare With


Package Explorer
JUnit
CalculationTest
Runs: 2/2
Errors: 0
Failures: 1
src.CalculationTest [Runner: JUnit 4] (0.008 s)
testAdd (0.003 s)
testSubstract (0.004 s)
Failure Trace
java.lang.AssertionError: expected:<6> but was:<-6>
at src.CalculationTest.testSubstract(CalculationTest.java:51)

1.9 还有哪些单元测试工具？

xUnit是为各种不同编程语言和平台服务的单元测试框架系列，x代表编程语言和平台。

- xUnit家族著名成员包括JUnit、CppUnit 、Nunit、 DBUnit、HttpUnit 和JSUnit
- 常用单元测试框架：

  http://c2.com/cgi/wiki?TestingFramework （2015.02.03）

1.9 还有哪些单元测试工具？

商业的单元测试工具

- 代码扫描工具：parasoft C++
- 内存资源泄漏检查工具：CompuWare BounceChecker，IBM Rational PurifyPlus等
- 代码覆盖率检查工具：CompuWare TrueCoverage，IBM Rational PureCoverage， TeleLogic Logiscope等
- 代码性能检查工具：Logiscope和Macabe等
- 可视化单元测试工具：Visual Unit
- ……

1.9 HttpUnit

测试web应用的工具

- 链接是否能连到正确的网页
- 一个网页是否能正确地呈现
- Javascript脚本是否能使按钮起作用

模拟浏览器行为

- 不需要运行浏览器
- 不需要网页服务器（*i.e. Apache/Tomcat*）
- 为Javascript提供一些支持

网页交互的JAVA API

- WebConversation, WebWindow, WebForm, WebLink

1.9 HttpUnit

功能包括

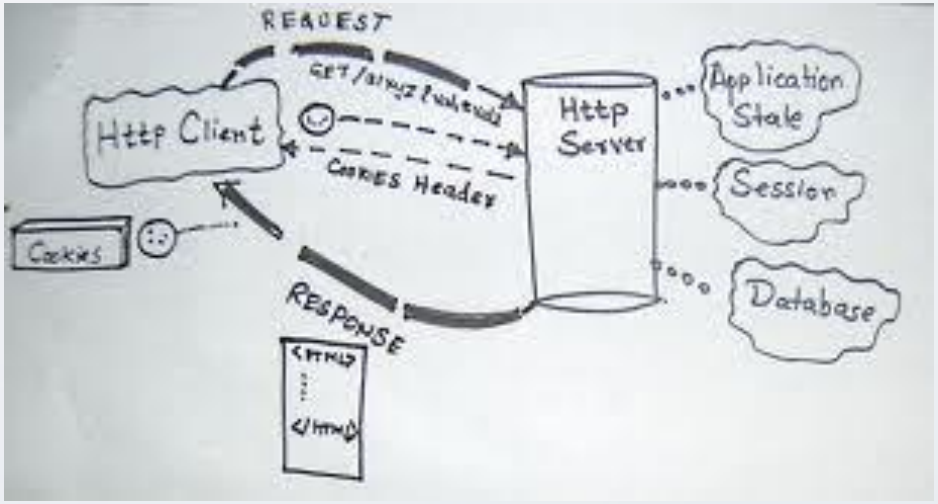
- 表单(Form)，处理页面框架(frames)，基本http验证，cookies及页面跳转处理等
- 与服务器端进行信息的交互，使用JUnit框架进行测试

*与其它工具比较（**Selenium**等）*

- ✘ 使用记录、回放功能
- ✓ 不需要重复录制

1.9 HttpUnit

- 免费
- 不完美但是比较实用
- 用户与网页的交互测试的自动化
- 脱离容器进行Servlet测试
- 不完全支持JavaScript



1.9 Eclipse中的使用

- 安装
  - 下载HttpUnit（http://httpunit.sourceforge.net/）
  - 启动Eclipse，new->java project
  - 在项目的Java build Path中将HttpUnit文件夹下lib和jars文件夹中的.jar文件加入到项目
- 使用
  - 使用HttpUnit模拟浏览器行为
  - 使用JUnit写测试用例进行网页测试

Q & A

Thank you