Junit4 使用纲要

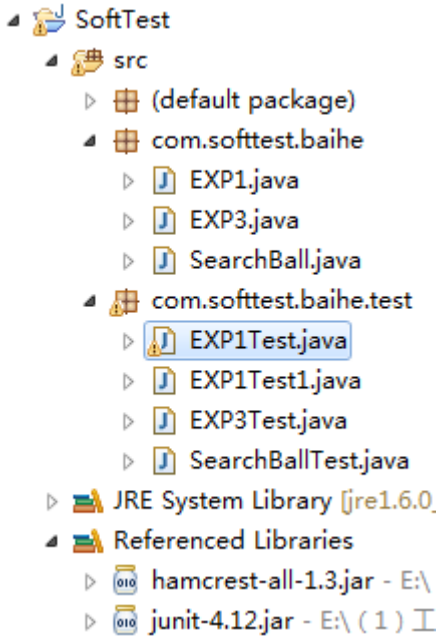
JUnit4 HelloWorld

新建

1. new project
2. 建立类
3. 建立 testcase

注意命名

1. 遵守约定，比如：
   a) 被测试类放在 com.softtest.baihe 包中
   b) 测试类放在 com.softtest.baihe.test
   c) 类名用 XXXTest 结尾，XXX 即为被测对象
   d) 方法用 testMethod 命名



放弃旧的断言,使用 hamcrest 断言

1. 新增：assertThat
   import org.hamcrest.Matchers.*;
   assertThat(z,Matchers.*anyOf*(Matchers.*greaterThan*(-1), Matchers.*lessThan*(2)));
   静态引入，更便捷：
   import static org.hamcrest.Matchers.*;
   assertThat(z,*anyOf*(*greaterThan*(-1), *lessThan*(2)));
2. 使用 hamcrest 的匹配方法 Matchers 规则匹配器
   a) 更自然、更符合自然语法
   b) 因为和原来 Junit 默认的 Junit 包不同的 classloder，因此需要删除原来默认，自己再引入 Junit 包
3. 示例
   a) assertThat(n, allOf(greaterThan(1), lessThan(15)));
      assertThat(n, anyOf(greaterThan(16), lessThan(8)));
      assertThat(n, anything());
      assertThat(str, is("bjsxt"));
      assertThat(str, not("bjxxt"));

   b) assertThat(str, containsString("bjsxt"));
      assertThat(str, endsWith("bjsxt"));
      assertThat(str, startsWith("bjsxt"));
      assertThat(n, equalTo(nExpected));
      assertThat(str, equalToIgnoringCase("bjsxt"));
      assertThat(str, equalToIgnoringWhiteSpace("bjsxt"));

   c) assertThat(d, closeTo(3.0, 0.3));
      assertThat(d, greaterThan(3.0));
      assertThat(d, lessThan (10.0));
      assertThat(d, greaterThanOrEqualTo (5.0));
      assertThat(d, lessThanOrEqualTo (16.0));

   d) assertThat(map, hasEntry("bjsxt", "bjsxt"));
      assertThat(iterable, hasItem ("bjsxt"));
      assertThat(map, hasKey ("bjsxt"));
      assertThat(map, hasValue ("bjsxt"));

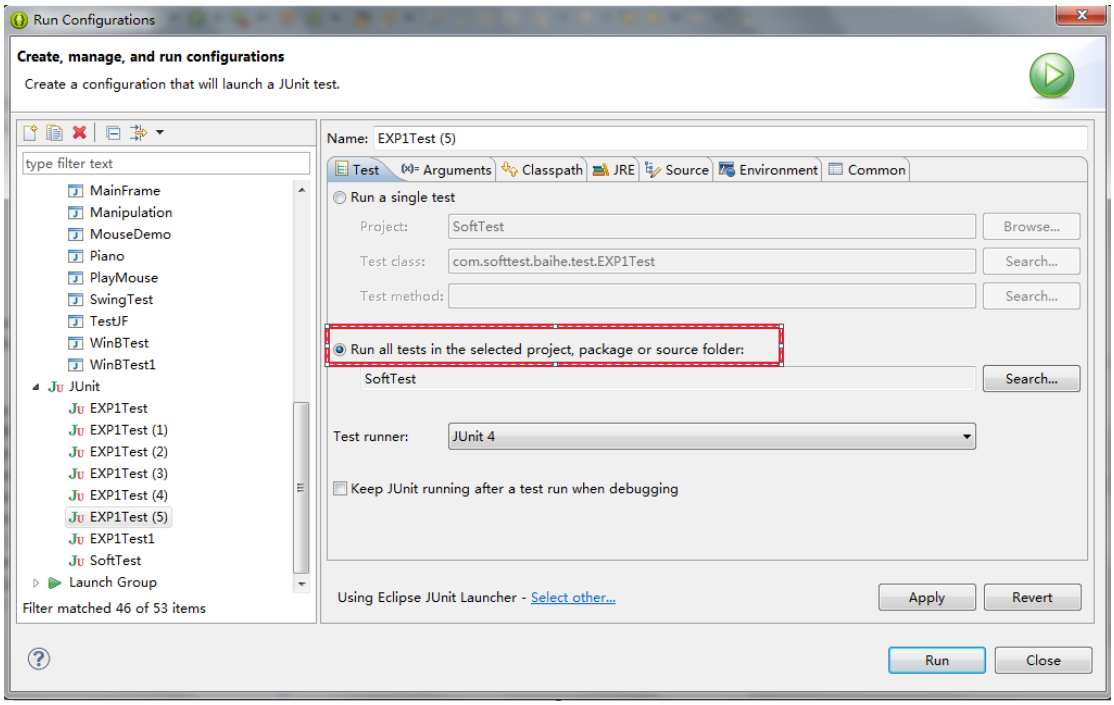
Failure 和 Error

1. Failure 是指测试失败
2. Error 是指测试程序本身出错 如：测试方法中增加 int a = 8/0;

JUnit4 Annotation 标记

1. @Test: 标记为一个测试方法
   a) (expected=XXException.class) **int** z = exp1.div(8, 0);抛出异常
   b) (timeout=xxx) 测试方法在 xxx 毫秒时间内结束
   如：
   @Test(expected = java.lang.ArithmeticException.**class**, timeout = 1)
2. @Ignore: 被忽略的测试方法，当项目还没有完成，这部分测试不执行
3. @Before: 每一个测试方法之前运行
4. @After: 每一个测试方法之后运行
5. @BeforeClass: 所有测试开始之前运行、测试程序初始化前就运行 如：建立数据库的连接，测试前需要进入的比较耗费时间的资源，搭建环境 static 引入
6. @AfterClass: 所有测试结束之后运行 如：关闭数据库的链接
   5、6 必须 static

运行多个测试



最佳实践

1. 一次只测试一个对象
2. 在 assert 调用中解释失败原因
3. 选择有意义的测试方法名

4. 一个单元测试等于一个测试方法
5. 同一个包，分离的目录，使用平行目录结构

其他框架

TestNG