# 测试环境搭建

## eclipse中普通工程

1. 导入jar包 -- eclipse中一般自带junit，可以自己导入jar包
2. 编写具体类
3. 右键新建 junit test case ，运行测试类。

## eclipse中 maven项目

1. 在pom.xml中加入依赖

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

</dependency>

1. 编写具体类，新建测试类测试就行。
2. maven会自动下载相关jar包。

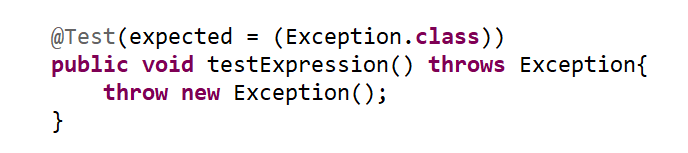
# Junit4 – framework

## 注解

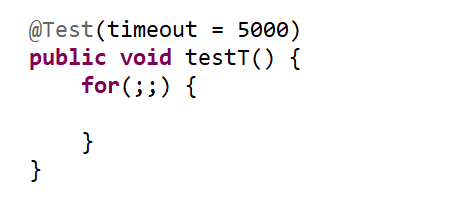
* 1. @Test

1. 作用：该注解的方法会被当做测试用例。测试方法必须是 public void，即公共、无返回数据。可以抛出异常。
2. 提供两个参数参数：

“expected”，定义测试方法应该抛出的异常，如果测试方法没有抛出异常或者抛出了一个不同的异常，测试失败。

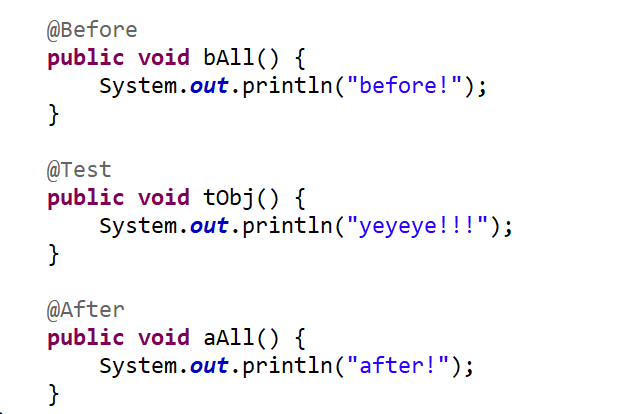


“timeout”，如果测试运行时间长于该定义时间，测试失败（单位为毫秒）



1.2 @Before 和 @After

1. 使用 @Before 注解一个 public void 方法会使该方法在 @Test 注解方法被执行前执行（那么就可以在该方法中创建相同的对象）
2. 如果在 @Before 注解方法中分配了额外的资源，那么在测试执行完后，需要释放分配的资源。
3. 使用 @After 注解一个 public void 方法会使该方法在 @Test 注解方法执行后被执行
4. 即使在 @Before 注解方法、@Test 注解方法中抛出了异常，所有的 @After 注解方法依然会被执行，见示例
5. 每个用例前都会执行一次@Before，每一个用例后都会执行一次@After。



* 1. @BeforeClass 和 @AfterClass

区别是：public static void

1. 使用 @BeforeClass 注解一个 public static void 方法，并且该方法不带任何参数，会使该方法在所有测试方法被执行前执行一次，并且只执行一次。
2. 如果在 @BeforeClass 注解方法中分配了代价高昂的额外的资源，那么在测试类中的所有测试方法执行完后，需要释放分配的资源。
3. 使用 @AfterClass 注解一个 public static void 方法会使该方法在测试类中的所有测试方法执行完后被执行
4. 即使在 @BeforeClass 注解方法中抛出了异常，所有的 @AfterClass 注解方法依然会被执行
   1. @Ignore
5. 对包含测试类的类或 @Test 注解方法使用 @Ignore 注解将使被注解的类或方法不会被当做测试执行
6. JUnit 执行结果中会报告被忽略的测试数

## 断言



