# 环境准备

## 安装myeclipse

在此不作详细描述

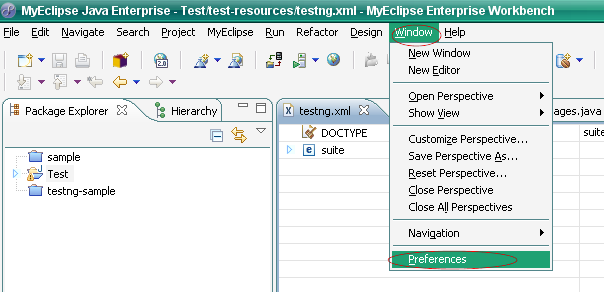
## 配置JDK

虽然myeclipse集成了JDK，但是一般我们都不使用它自带的，而是重新安装一个JDK。

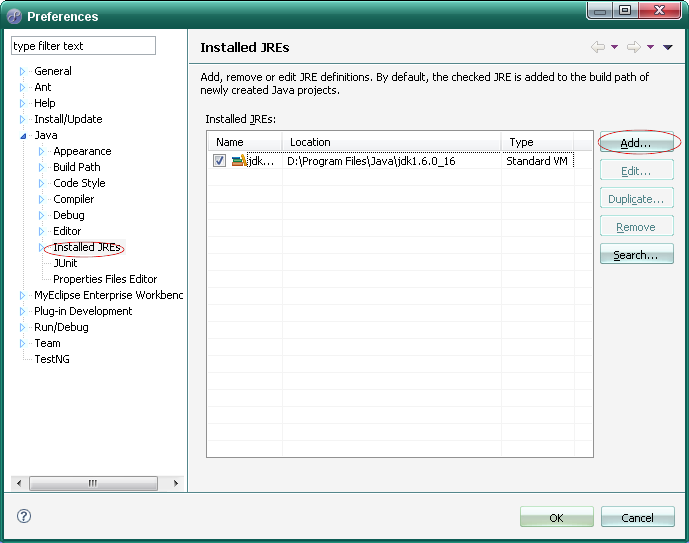
### 安装JDK1.6

### 配置JDK1.6

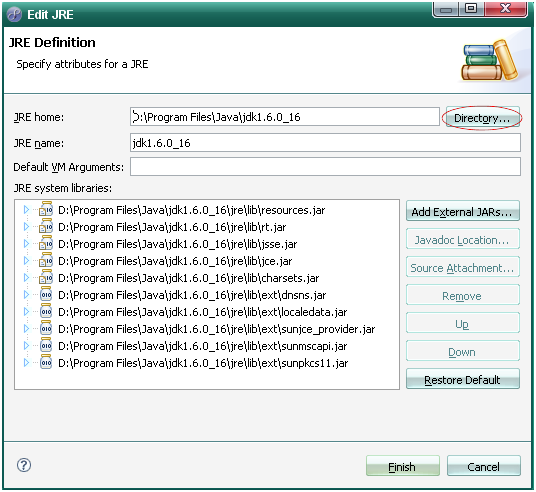
1. 打开myeclipse，选择“Window--Preferences”，如下图：



1. 在弹出的属性窗口中选择“Java--Installed JREs--Add”,如下图：



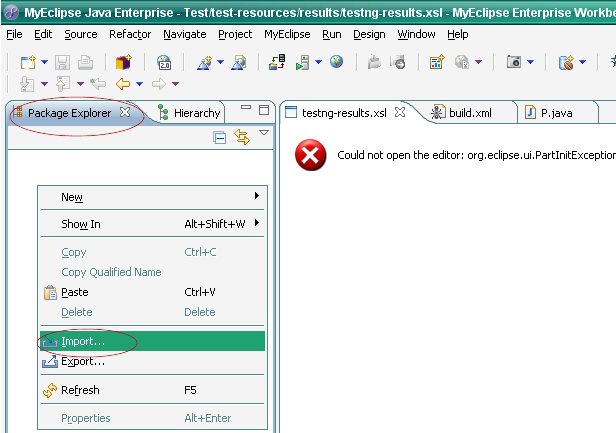
1. 在弹出的窗口的“JRE home”中选上自己的JDK的目录，如下图所示：



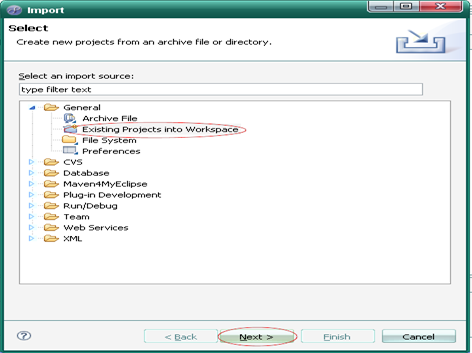
1. 点击“Finish”按钮，完成JDK的配置。

# 添加工程

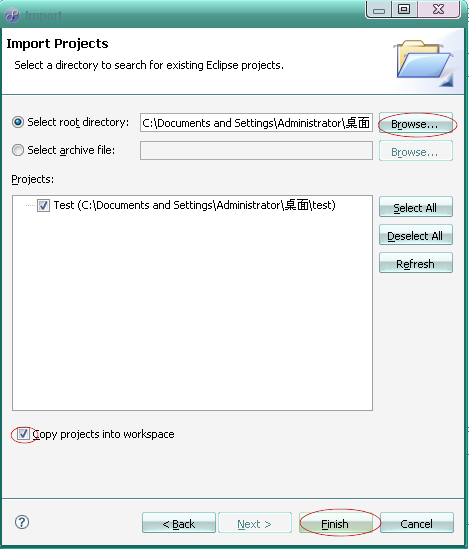
1. 将Test工程：  解压，到某个目录
2. 在myeclipse的Package Explorer的空白处右键选择“Import”，如下图：



1. 在弹出的窗口中选择如下图，点击“next”按钮

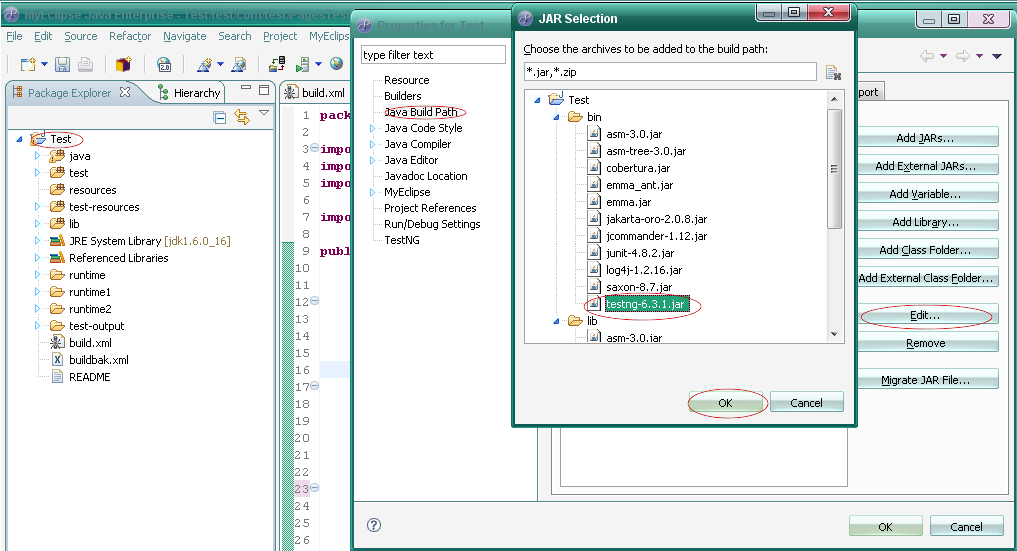


1. 选择test工程目录，如下图，点击“Finish”按钮，完成java工程导入。



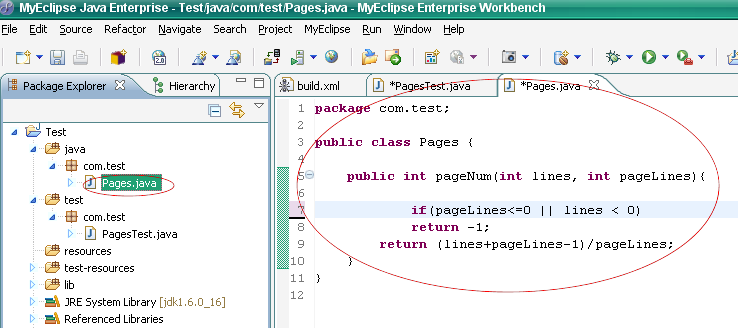
1. 工程添加完成后，可能需要重新加载testng-6.3.1.jar包（选择工程lib目录下的jar包），加载方法如下：

选择工程，右键—Properties--Java Built Path—Libraries-testng-6.3.1.jar-Test/bin—Edit

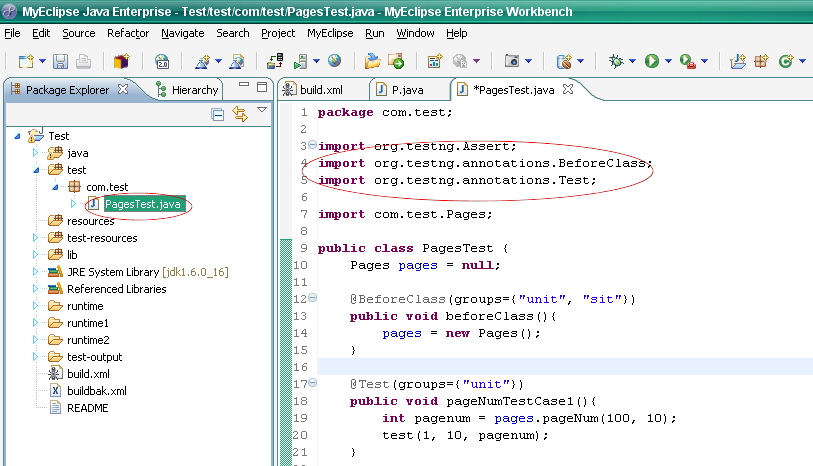


# 编写测试类

本工程的被测试文件是：Pages.java，如下图所示：



测试文件为：PagesTest.java，如需要对Pages类添加测试类，就直接在本文件中编辑添加即可。下图所示的引用的包，就是上节添加的testng-6.3.1.jar包中的。

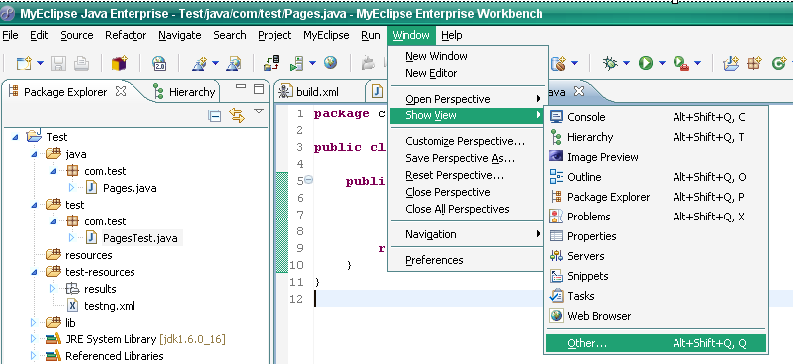


# 运行测试

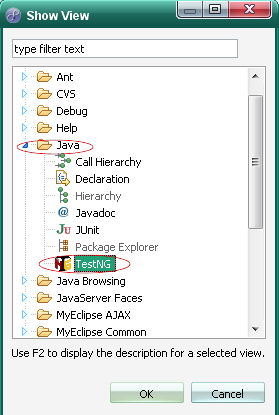
## 方式一（testng运行）

### 安装testng插件

1. 首先解压testng插件 ，并将解压后的文件org.testng.eclipse\_5.14.0.1.jar 拷贝到myeclipse的插件存放路径如“D:\Program Files\MyEclipse 7.0M1\eclipse\plugins”
2. 重新打开myeclipse工具，选择“Window—Show View—Other…”如下图：

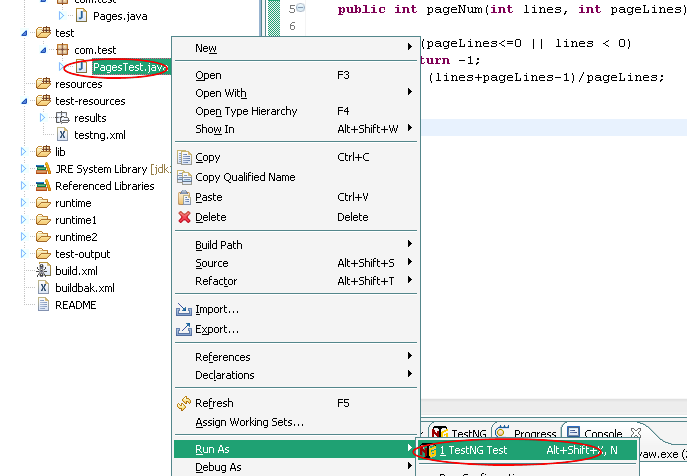


1. 可以查看到testng的图形界面如下图：

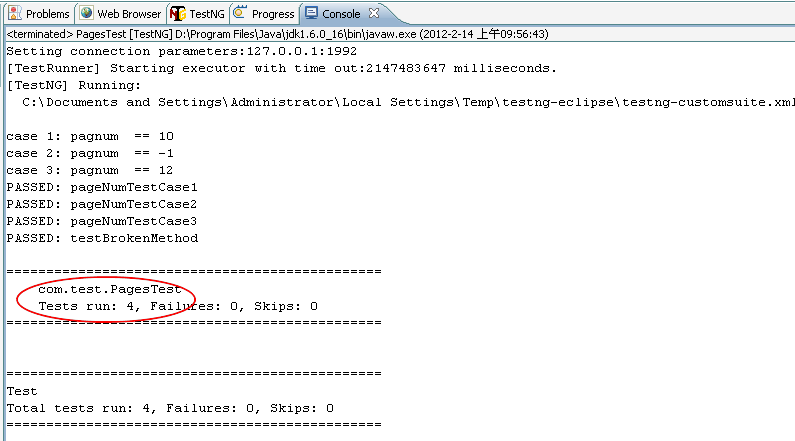
****

### 直接运行测试类

1. **选择PageTest.java右键—Run As—TestNG Test，如下图所示：**

****

1. **运行结果：**

****

### 运行testng配置文件testng.xml

1. **前面在编写测试类时我们已经对每个案例进行了分组，例如：**

@Test(groups={"unit"})

**public** **void** pageNumTestCase1(){

**int** pagenum = pages.pageNum(100, 10);

test(1, 10, pagenum);

}

**测试类**pageNumTestCase1()分组为“unit”（单元测试），那么在**配置文件testng.xml中就可实现测试类的运行策略，如下：**

**<!DOCTYPE suite SYSTEM "http://beust.com/testng/testng-1.0.dtd">**

**<suite name="com.test">**

**<test verbose="2" name="PagesTest">**

**<groups>**

**<run>**

**<include name="unit" />**

**<include name="sit" />**

**<exclude name="broken" />**

**</run>**

**</groups>**

**<classes>**

**<class name="com.test.PagesTest" />**

**</classes>**

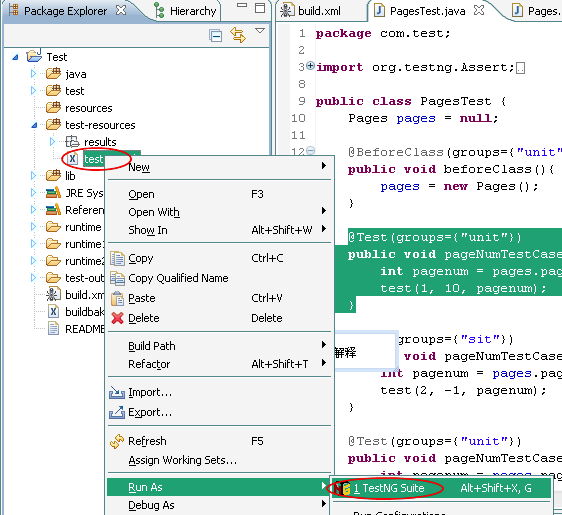
**</test>**

**</suite>**

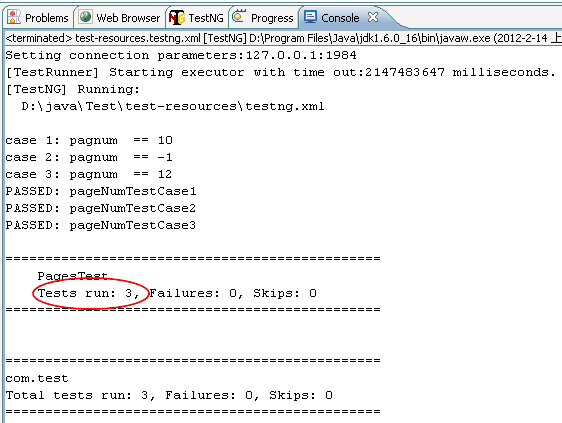
**<run><include name="unit" /><include name="sit" /><exclude name="broken" />/run>**

**此段代码的表示运行“unit”“sit”组的测试类，不运行“broken”组的测试类**

1. **选择testng.xml右键—Run As—TestNG Test，如下图所示：**

****

1. **运行结果：（与直接运行测试类的区别在于这里没有运行broken组的测试类**testBrokenMethod**）**

****

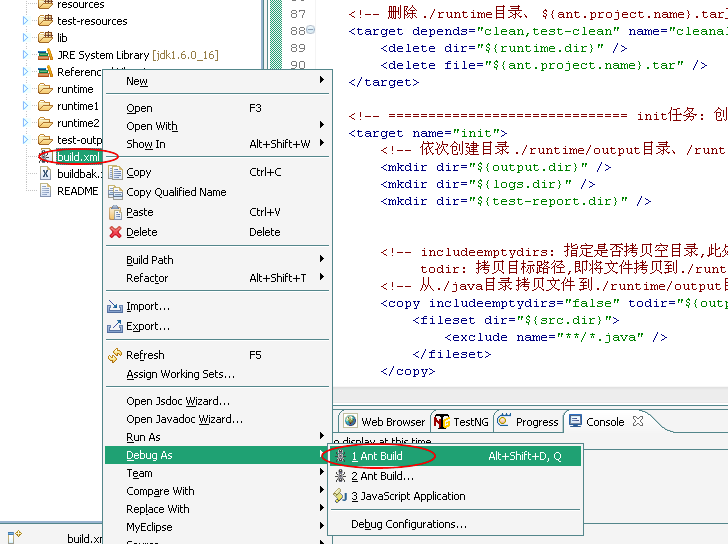
## 方式二（ant运行）

### 介绍

Ant编译文件build.xml,这个文件配置了xslt和emma，将testng原生的测试报告转换成更加清晰的报告，并且生成代码覆盖率报告，代码覆盖率是一个测量指标，它用于表示测试实际执行了多少代码。主要思想是，通过检查代码覆盖率，可以确定哪些代码是被执行了的，并确保我们为测试每个分支编写了测试。（具体实现请查看build.xml文件）

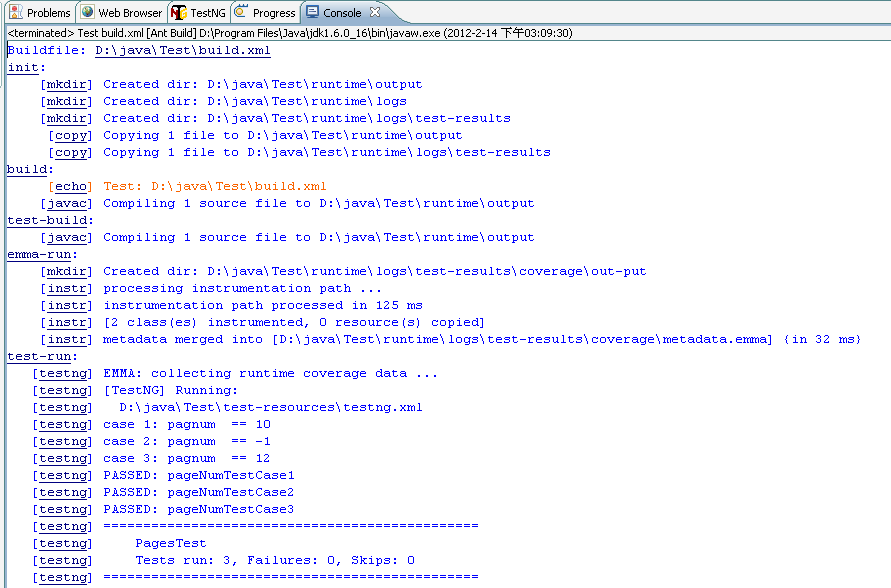
### 运行

1. **选择**build.xml**右键—Run As—Ant Build，如下图所示：**

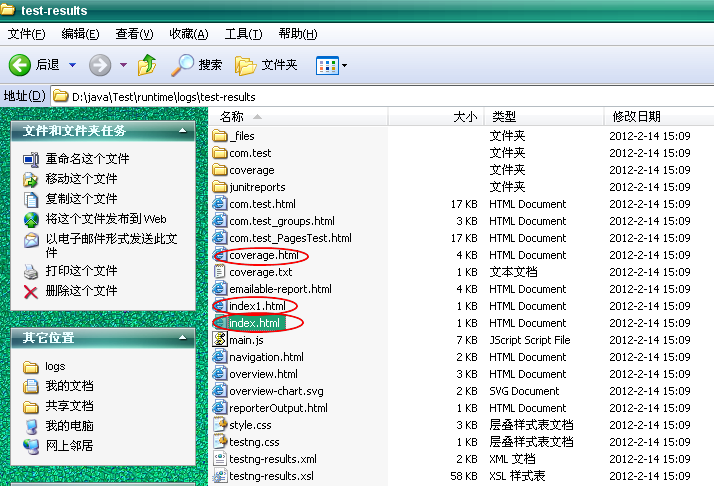


### 结果

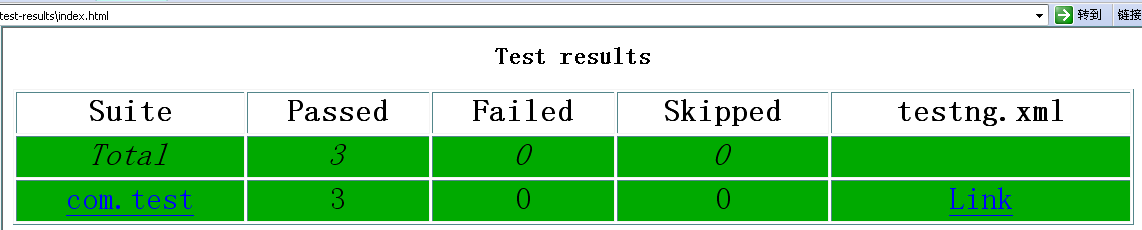
1. 控制台运行结果，如下图：



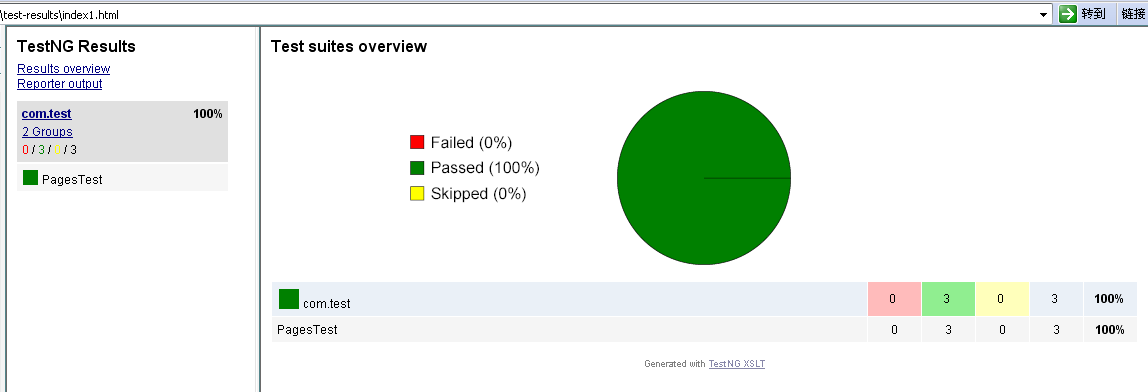
1. 在工程目录下会生成一个runtime的文件夹，该目录里存放着我们需要的报告，如下图所示：



index.html：是testng生成的原始报告文件，如下图：

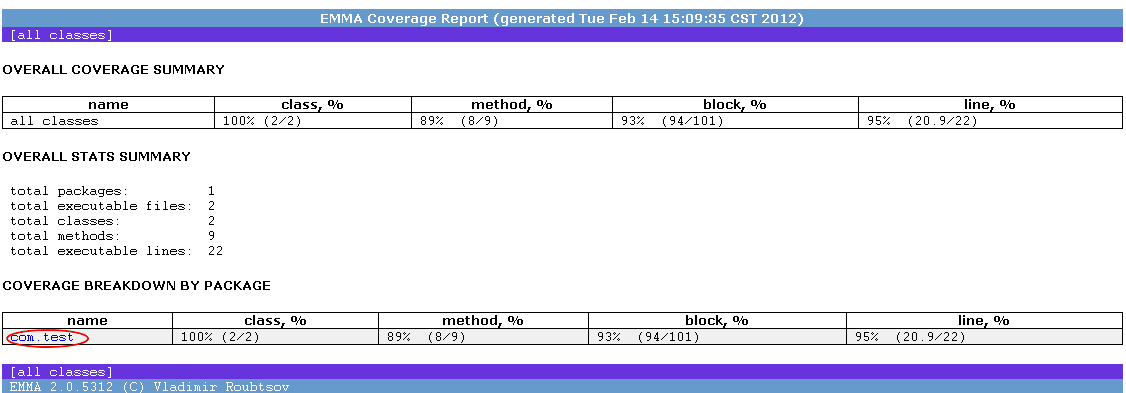


Index1.html：就是testngxslt重新生成的报告，如下图：

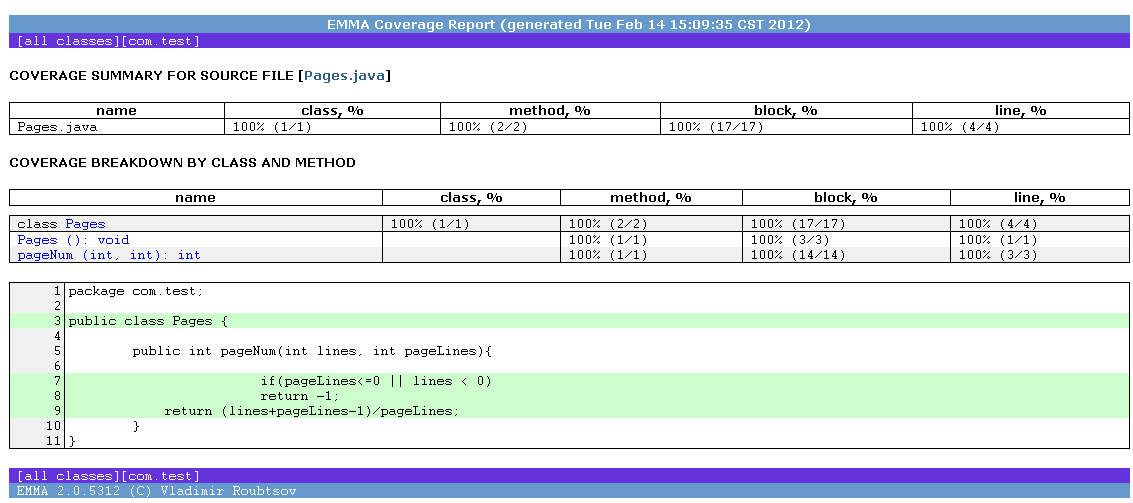


附：如果要生成上图的的饼图，需要安装一个插件，直接解压安装即可。

coverage.html：代码覆盖率报告，如下图所示：



点击“[com.test](file:///D:\java\Test\runtime\logs\test-results\_files\0.html)--[Pages.java](file:///D:\java\Test\runtime\logs\test-results\_files\2.html)”可以查看被测试类[Pages.java](file:///D:\java\Test\runtime\logs\test-results\_files\2.html)的测试执行情况如下图：



这里列出了测试的代码其中执行到的路径背景被标为绿色，如果未执行到，则是红色，只执行了一部分，如if(x>0 || x<11),当x>0时,测试将执行if(x>0)这一段，而x<11这一段可能不会执行，而这两段是在同一行的,则显示为黄色的。