

Vi, Java, Ant 和 Junit 的自学报告

13331093 黄雄镖

今天是实训的第二天了，这两天来学到了很多新的东西，由于平时有使用 Linux，所以对于 Linux 的基本使用是没问题的，而之前也看过 Java 的教程，所以学起来并不是很困难，有不懂的地方请教 TA，能够获得足够的帮助，下面是我对于 Vi, Java, Ant 和 Junit 的学习报告。

Vi 学习报告：

早就听说过这个神级编辑器，之前也有接触过，感觉就是刚开始的时候起步比较难，就连退出都要费很大劲。感觉不会用的时候，想做一个操作还要去查查该怎么做，不过估计等熟悉了以后会非常快，对提升效率方面很有效。众多的快捷键，感觉就是在用 terminal 一样，减少鼠标的参与。毕竟鼠标提供的相关功能有限，然与快捷键如果能娴熟使用，速度能显著提高，何况快捷键可以提供众多的功能，使得一些复杂的操作只要一些按键操作就可以完成。此外，还有丰富的插件，虽然还没有使用，不过看介绍，感觉这些插件会使编辑器更加强大。据说还可以高度配置，经过配置可以拥有语法高亮和自动补全，甚至配置成 IDE，非常厉害。目前还在学习中，很多命令都还不熟悉，等以后熟悉了的话用起来会更爽吧。

Java 学习报告：

放假的时候看过 Java 的教程，快速学习了一下，所以基本上是没有问题的。在自己的电脑上配置好了环境，用 Eclipse 打起来还是很方便的（有提示和补全等功能）。不过在云桌面上一进去就已经配置好了 JAVA 等环境，也是很方便。

对于 Java 这门语言，用了之后感觉像是 C++ 和 Python 的结合体，在某些地方会相似，所以学起来比较快。据我了解 Java 既算是编译型的语言，也算是解释型的语言，它的运行过程是将 java 文件编译成 class 文件，然后再解释，个人感觉用起来它更像是编译型的吧，比较不像解释型的一样拥有 REPL。

其实之前有其他编程语言基础的话学起来真的方便不少，因为很多东西都是互通的，还有培养起来的编程的思维都是不变的。只是有时候不知道具体的方法名称而已，这个通过查询一下就可以了，还有遇到问题在网上也很容易找到解决的办法。第一天通过做计算器小程序的练习，感觉已经对 Java 有了基本的了解，不过还有很多 Java 语言的特性我还没有掌握，希望在接下来的学习中会学到更多。

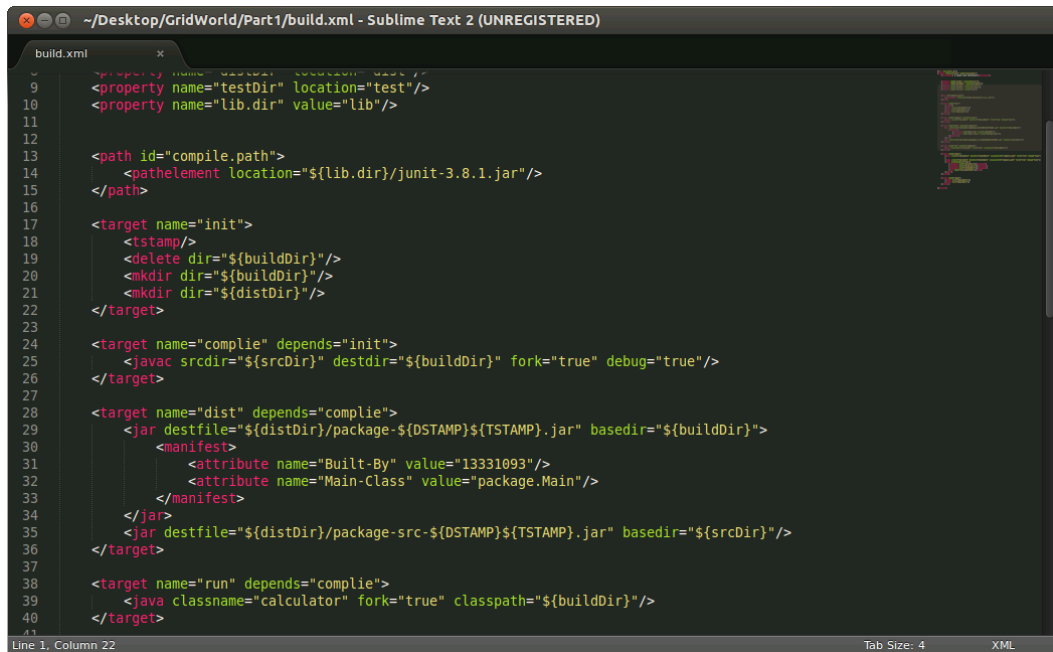
Ant 的学习报告：

ant 是一个将软件编译、测试、部署等步骤联系在一起加以自动化的一个工具，大多用于 Java 环境中的软件开发。在实际软件开发中，有很多地方可以用到 ant。由于在云桌面上已经配置好了，所以就不用配置了。

这是第一次接触到 Ant，感觉这是一个很好到 东西。觉得自动化的东西能方便很多，能够减少很多繁琐的重复的操作，如编译多个文件等，可以直接使用一个命令来实现，达到了编写一次，使用多次的作用。之前也使用过类似的自动化工具，如 C 语言里到 make，JS 里有 Gulp，Grunt，与我使用过的自动化工具比较，我觉得 ant 的其中一个特点是使用了 XML 这种标记语言，XML 具有容易理解的特点，每个标签都代表了它是有什么作用的。由于之前使用过 XML，所以障碍就更小了，通过这次 ant 的使用，加深了对 XML 的理解。这是在写法上的，我觉得是一个优点，不过这种易读易解析的优点牺牲的就是要打更多的字吧。在本质上，他与其他自动化工具做的差不多，拥有的特性也是差不多的。

Ant 在自动构建和部署 Java 程序方面方便易用，而且非常灵活，不失为 Java 开发者的绝佳帮手。

下面是我编写的 Ant 的文件的一部分：



```
build.xml
9  <property name="testDir" location="test"/>
10 <property name="lib.dir" value="lib"/>
11
12
13 <path id="compile.path">
14   <pathelement location="${lib.dir}/junit-3.8.1.jar"/>
15 </path>
16
17 <target name="init">
18   <tstamp/>
19   <delete dir="${buildDir}"/>
20   <mkdir dir="${buildDir}"/>
21   <mkdir dir="${distDir}"/>
22 </target>
23
24 <target name="compile" depends="init">
25   <javac srcdir="${srcDir}" destdir="${buildDir}" fork="true" debug="true"/>
26 </target>
27
28 <target name="dist" depends="compile">
29   <jar destfile="${distDir}/package-${DSTAMP}${TSTAMP}.jar" basedir="${buildDir}">
30     <manifest>
31       <attribute name="Built-By" value="13331093"/>
32       <attribute name="Main-Class" value="package.Main"/>
33     </manifest>
34   </jar>
35   <jar destfile="${distDir}/package-src-${DSTAMP}${TSTAMP}.jar" basedir="${srcDir}"/>
36 </target>
37
38 <target name="run" depends="compile">
39   <java classname="calculator" fork="true" classpath="${buildDir}"/>
40 </target>
41
```

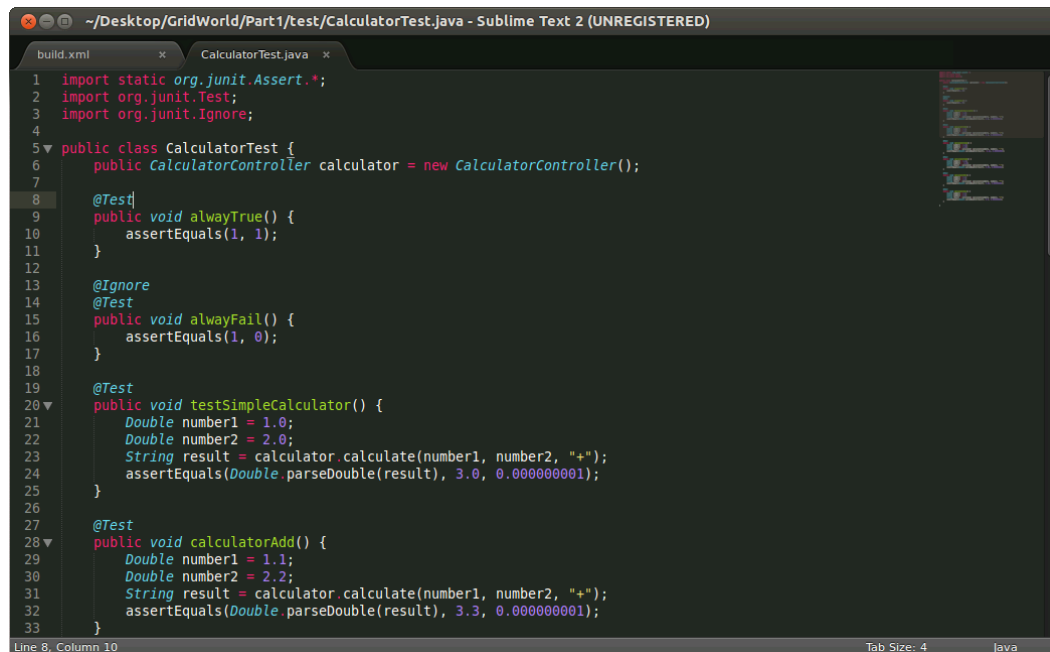
Junit 的学习报告：

这是我第一次接触单元测试，之前知道写测试对于写代码来说是很重要的事情，因为没有经过测试的代码是不能说明能够正常运行的，是不可靠的。在开发中也有测试驱动编程一说，它要求在编写某个功能的代码之前先编写测试代码，根据测试来一步一步验收代码，保证了代码的正确性。测试对于维护来说也是非常必要的，如果之前有写比较完备的测试，那么在后来的更新或者是重构代码之类会更加安全，只要修改后的代码还是通过测试就可以了。

据了解，Junit 是一个可编写重复测试的简单框架。看了网上的一些 Junit 的代码，之前的版本好像都要继承于 TestCase 类，显得有些繁琐。Junit4 使用注解（元数据），而且写 test case 的类不需要继承

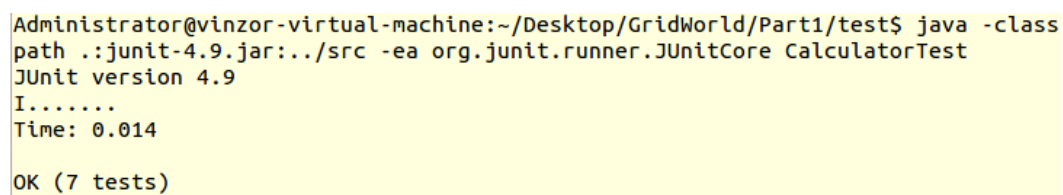
TestCase，只需要在所要做 test case 的方法前加@Test 注解即可。它提供了许多新的 API，而且现在还使用注解，所以使开发测试用例更为方便容易。不过我也没有编写到很复杂的测试样例，所以很多特性都没有使用到，以后将继续学习这个强大的测试框架。

下面是我为自己写的计算器小程序编写的 Junit 的文件的一部分：



```
1 import static org.junit.Assert.*;
2 import org.junit.Test;
3 import org.junit.Ignore;
4
5 public class CalculatorTest {
6     public CalculatorController calculator = new CalculatorController();
7
8     @Test
9     public void alwayTrue() {
10         assertEquals(1, 1);
11     }
12
13     @Ignore
14     @Test
15     public void alwayFail() {
16         assertEquals(1, 0);
17     }
18
19     @Test
20     public void testSimpleCalculator() {
21         Double number1 = 1.0;
22         Double number2 = 2.0;
23         String result = calculator.calculate(number1, number2, "+");
24         assertEquals(Double.parseDouble(result), 3.0, 0.000000001);
25     }
26
27     @Test
28     public void calculatorAdd() {
29         Double number1 = 1.1;
30         Double number2 = 2.2;
31         String result = calculator.calculate(number1, number2, "+");
32         assertEquals(Double.parseDouble(result), 3.3, 0.000000001);
33     }
34 }
```

测试结果：



```
Administrator@vinzor-virtual-machine:~/Desktop/GridWorld/Part1/test$ java -class
path ./junit-4.9.jar:../src -ea org.junit.runner.JUnitCore CalculatorTest
JUnit version 4.9
I.....
Time: 0.014

OK (7 tests)
```

以上就是我对于 Vi，Java，Ant 和 Junit 的学习报告，两天来学到很多有用的技术，实训马上就要进入第二阶段，希望能再第二阶段学到更多东西。