自学报告

13331231

孙圣

Vim：

由于习惯了最新的图形化界面编辑工具例如 Sublime-text，刚开始接触 Vim 的时候觉得这个编辑器很奇怪，但还是逐渐了解了其中的一些指令。

一开始打开 Vim 的时候进入的是命令行模式，在该模式中并不能对文本进行输入的操作，但可以利用 x 对单个字符进行删除，也可以用 dd 删除整一行。有很多快捷键可以帮助快速的移动光标，比较常用的应该是$和^符号，使光标分别移动到行尾和行的开头。u 可以帮助撤销之前的操作。

按下 i 之后会进入插入模式，这时候可以更加方便的对文件进行修改。如果想要回到命令行模式，按下 Esc 键便可。对所修改的文档进行保存需要先回到命令行模式，之后按：键进入 Last line mode。在：之后输入 w 可以保存文件；wq 可以保存并退出；q!可以强制退出。

对于常用的搜索字符的功能：在命令行模式中先输入/，之后输入想要查找的字符，按 n 可以找到下一个。

Swing & Awt：

Swing 和 Awt 是用于制作 Java GUI 的工具。想要制作 GUI，类需要 extends Jframe，这时候就能得到一个空白的框。我们可以通过 *setBounds* 函数来指定框的起始位置(x, y)坐标及其长和宽。之后，我们一般会建立一个大的 JPanel 来存放所有的元件，通过 *setLayout* 函数来改变它的布局。然后，我们可以根据布局建立多个 Panel。我们可以利用 JTextField, Jlabel, Jbutton 来分别创建文本框，标签和按钮，并把它们加入到所需要的 JPanel 中。我们可以通过丰富的函数例如 *setPreferredSize* 来修改各个元件的样式。

我们可以对元件进行监听，例如：如果按钮被按下，某些相应的函数会被触发。这时候就要用到 *addActionListener*。

同时，我们也要通过 *setDefaultCloseOperation* 来保证程序的正常关闭。通过 s*etVisible* 来控制界面的显示。

Java 语法:

Java 本身的语法与 C++类似，所以并不需要专门把教程的从头看到尾，只需要在不懂的地方上网查找资料解决问题即可。

这次完成简单计算器的过程中运用到了数字与字符串的相互转化。例如要将字符串转为 double，要用 *Double.valueOf(numberString).doubleValue()* 将 double 转为字符串，要用 *Double.toString(number)*。这次也将所计算的结果保留了 4 位小数，但如果结果为 1.2 时，就显示 1.2 而不会显示多余的 0。这是通过以下代码段实现的。

BigDecimal bd = new BigDecimal(answer);

double roundAns = bd.setScale(4, BigDecimal.ROUND_HALF_UP).doubleValue();

Ant:

Ant 是类似 C/C++中 Makefile 的工具，我们可以通过写一个 XML 文件，来简化编译运行过程中所需要键入的指令，节约的时间。

<project>标签内为语言的主体内容，我们可以通过<property>来给文件路径一个变量名，之后便可以通过这个变量名来代表文件路径。

<target>可以理解为一个一个的任务，其中 depends 定义了依赖，即必须先执行了依赖的任务才能执行本项目。

一般的架构为 clean, init(即，先删除所有之前存在的文件，再创建新的空文件夹)之后进行 compile，利用<javac>标签实现，只需要指出对应的文件夹而不需要指出具体的.java 文件。然后是生成 jar 文件，最后是执行生成好的文件。

还可以通过 Ant 与 Junit 整合，大大简化使用 Junit 过程中需要输入的繁琐命令。

Junit:

利用 Junit 可以对已经写好的类进行测试，从而保证类没有 bug，便于进一步的开发。对于需要测试的类，每一个都要有一个相对应的测试类。在测试类中，我们要先创建一个被测试类的对象。@Test 代表测试的起始。之后我们要定义测试函数，测试函数不能够有参数，返回值也一定要为 void。测试的一般步骤是，调用所要测试的那个函数，利用 *assertEquals* 方法将理想的结果与测试的函数所返回的结果进行比较。如果结果相同，则测试通过，此测试样例中该函数没有 bug。如果结果不相同，则测试失败，需要修改原函数，找出 bug 所在。@Igonre 可以避免对未完成的类进行测试。

调用 Junit 的指令较为复杂，首先用 javac 进行编译，通过-classpath(-cp)来指定要寻找的

类的路径(即 -cp .:junit-4.X.jar)，再写出测试类的 java 代码名称。之后便开始执行，同样需指出 classpath，后面要加入特定的一串字符(-ea org.junit.runner.JUnitCore)。