

《Java程序设计》上机实验报告

第 6 次上机

学号： 201611130126

姓名： 袁宇昊

学院： 信息科学学院

专业： 计算机科学

教师： 栾华

日期： 2019.3.28

1. 实验要求

1.上机之前应做好充分准备，认真思考所需的上机题目，提高上机效率。

2.独立上机输入和调试自己所编的程序，切忌抄袭、拷贝他人程序。

3.上机结束后，整理出实验报告。书写报告时，重点放在实验的方法、思路以及总结反思上，以达到巩固课堂学习、提高动手能力的目的。

1. 实验过程

对于每个问题，可以包括:

1.实验步骤，如对某个问题的分析、算法思路；

2.实验过程，如关键代码。

3.运行结果，如相关问题运行结果的截图，结果说明等等。

1. 实验目的

1．掌握 Java 应用程序可视化设计开发过程。

2．应用 Java 组件在 Java 应用程序中可视化设计 GUI。

3．可视化设计与编写代码结合设计事件处理。

四、实验内容（每个程序需要建立一个独立的Java包）

1．应用 Eclipse 向导创建工程文件（建立一个 Java 包）。

new-->java project 设置名字为test6

2．应用 Eclipse 向导创建包含 Frame 窗口框架的 Java 应用程序。

在工程中 File-->New-->Other ...-->WindowBuilder-->Swing Designer-->JFrame 改名字为MainFrame

3．在 Frame 窗口中设计一个按钮，按钮显示为退出程序，单击按钮时，退出程序， 编写“退出程序”的事件处理代码。

最开始写的是System.exit(1),但是看到后来有实验6-10创建一个新窗口，如果是System.exit(1)的话，关闭一个窗口会关闭所有的窗口，故改为dispose()关闭当前窗体。

4．在 Frame 窗口中设计一个文本框和一个文本区域，文本框内容改变时，将文本框 中的内容显示在文本区域中；在文本框中按回车键时，清空文本区域的内容。

在对文本框1中进行type事件处理：去更新文本框2。

5．在 Frame 窗口中设计一个复选按钮和一个普通按钮，复选按钮选中时，普通按钮 的背景色为青色，未选中时为灰色。

对复选按钮进行事件监听：按条件更新普通按钮的背景色。

6．在 Frame 窗口中设计一个单选按钮组和一个普通按钮，单选按钮组中包含三个单 选，文本说明分别为“普通”、“黑体”和“斜体”选择文本标签为“普通”的单选按 钮时，普通按钮中的文字为普通字体，选择文本标签为“黑体”的单选按钮时，普通按 钮中的文字的字体为黑体，选择文本标签为“斜体”的单选按钮时，普通按钮中的文字 的字体为斜体。

对三个单选按钮进行监听：按条件改字体。先要关联三个单选按钮。

\* 7．在 Frame 窗口中设计一个下拉列表和一个按钮，下拉列表中有 10、14、18 三个选 项，选择 10 时，按钮中文字的字号为 10，选择 14 时，按钮中文字的字号为 14，选择 18 时，按钮中文字的字号为 18。

监听下拉列表：更新按钮文字大小

\* 8．在 Frame 窗口中设计一个列表和两个标签，在第一个标签中显示列表中被双击的 选项的内容，在第二个标签中显示列表中被选中的所有选项的内容。

先说选中标签：监听list的选中改变情况，每改变一次，就遍历list中选中的项，然后更新标签的Text。

再说双击：监听list的鼠标运动情况，因为鼠标双击后，鼠标所在位置的list项总是被选中的，所以在鼠标双击后，就按选中的list项更新标签内容就可以了。

9．在 Frame 窗口中设计一个标签，编程确定当前鼠标的位置坐标代码，以（X1，Y1）、 （X2，Y2）的形式显示在标签中。

在窗口层面监听鼠标运动情况：移动鼠标就把当前鼠标所在XY更新到标签的Text中就行了。

\* 10．在 Frame 窗口中设计一按钮，另外创建另一个 Frame，当单击按钮时，弹出另一个 Frame。

按钮触发事件：在原标题的基础上，实现了新建的每一个窗口名字是不一样的。



这幅程序运行截图中有两个窗体：“实验6大合集”和“新实验6大合集”。其中“新实验6大合集窗体”是由另一个窗口点击新窗口触发的。

左边与右边有以下的不同点：

1．左复选按钮勾上使得单选按钮背景色为青色，右复选按钮未勾上使得单选按钮背景色为灰色。

2．左字体选择为黑体，右字体选择为斜体。

3．左字号大小为18，右字号大小为10。

4．左双击选中的是item2，右 配合使用Ctrl键选中的是item1、item3和item4。

11．思考与分析 \* （1）总结 Java 的事件处理机制。 （2）描述事件监听器和事件适配器的关系。

（1）java中事件处理机制简单说是事件监听者event listener去解决事件源event source产生的各种事件event （object）。

event source可以是各个图形组成对象、或者窗体本身等。比如一个文本框JTextField()就是一个事件源。

event listener是被封装在java.awt.event.里的一种特殊的接口，都继承EventListener这个空接口，这些接口里面全是预定义的方法（虚函数）。比如KeyListener ()就是java.awt.event类中自带的一种监听键盘输入的监听者。而keyPressed()就是这个监听者的一个虚方法，如果实现了它，那么这个方法相当于被“激活”了（一般来说一个类要实现它接口中的所有的方法才能使用，但通过事件适配器EventAdapter()就可以实现只实现了部分方法，就可以使用这个监听者类了（其他方法默认为空方法）。特别注意在监听者中不能写新的方法，只能实现已经声明过的虚方法，否则将永远不会调用这个新方法，如果要看一个监听者有哪些方法，可以查看它的定义。与监听者方法不能自己写相反的是，咱们可以自己写一个继承某个Listener的监听者并且实现它，或者调用库里自带的监听者并且实现其接口的全部方法，或者直接借用库里的适配器并重载（其实不是借用本身，而是借用Adapter的子类）其部分方法。

event(object)是event source产生的、带有事件信息的一种类，一般它是由jvm虚拟机自动产生并抛给那些对此事件源进行监听的监听者们的“对应”方法（对应的方法是死的，所以自己写的方法没用，而对应的监听者是活的，可以是自己写的关联上的监听者），在这时它相当于是一个listener方法的参数，或者说listener的方法靠它来触发。比如说KeyAdapter适配器的方法keyPressed(KeyEvent e){}，此方法函数的参数就是KeyEvent，它就是事件。

三者的关系是：

Event Source关联Event Listener (Event Listener 监听Event Source)

Event Source产生Event Object

Event Listener收到Event Object

一般事件处理机制顺序：

1 事件源添加监听者，同时监听者实现部分方法。

2 事件源产生事件并抛给监听者（不管该事件对应的方法监听者是否实现，未实现就是空方法）

3 事件监听者调用方法。

值得注意的是：事件、事件监听者都在Frame类下，所以监听者方法可以自由调用各个事件的方法。由于java支持在参数中定义参数，所以有监听者匿名的写法，这样使得代码更为简略。

（2）事件适配器就是事件监听器的空实现者，即每个方法都是空方法。这样的好处是当程序员借用适配器时，只需要实现部分方法就可以了，其他方法不用管都行。

用适配器来写textfield1的监听者的匿名写法很简洁：

textfield1.addKeyListener(**new** KeyAdapter() {//KeyAdapter

**public** **void** keyPressed(KeyEvent e) {

textfield2.setText("打字了");

});

而用Listener的话就必须还要补全两个方法的实现。

textfield1.addKeyListener(**new** KeyListener() {//KeyListener

**public** **void** keyPressed(KeyEvent e) {

textfield2.setText("打字了");

}

**public** **void** keyTyped(KeyEvent e) {}//必须写

**public** **void** keyReleased(KeyEvent e) {} //必须写

});

适配器的原理在于类方法的重写（覆盖）。

提交说明：  
提交一个rar 或zip压缩文件，其中包括：实验报告、源程序等，rar 或zip文件名为学生学号和姓名。

例如，20192255225\_姓名\_实验x

文件夹内包括所有源程序、工程文件等也用此命名。  
发送到：yueer0538@126.com