DEPARTMENT OF COMPUTER SCIENCE

UCL

COMP2007 Coursework 2

Purpose: To familiarise yourself with using Swing.

Submission: Complete this coursework by Friday 28th October and submit online via Moodle (instructions will be sent by email).

Marking: This is a *collaborative* coursework that will be binary marked and is worth 1% of the overall module mark. Collaborative means that it is all right to work together on doing this coursework. The essential thing is that you individually fully understand the programming ideas and concepts involved when using Swing.

Preliminary Activities

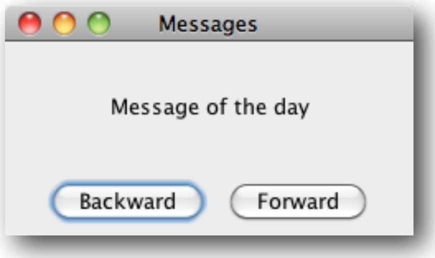
1. Read through resources such as the Swing tutorial at the http://download.oracle.com/javase/tutorial/uiswing/ web site and learn more about constructing user interfaces. Also see the COMP2007 pages on Moodle for the Swing examples mentioned in lectures.

2. To check that your Java installation is working, copy and paste the HelloGoodbye program from the Moodle page, save it to a file, compile and run it.

A common question: Can you use a GUI builder, like the one in NetBeans, or an Eclipse plugin, etc.? Well, the point of these exercises is to learn about Swing by writing the code by hand, so you will be missing out if you do. My experience is that it is easier to code simpler GUIs by hand in the long run. GUI builders tend to produce messy and difficult to read code. They are useful, however, for planning the structure and layout of a GUI.

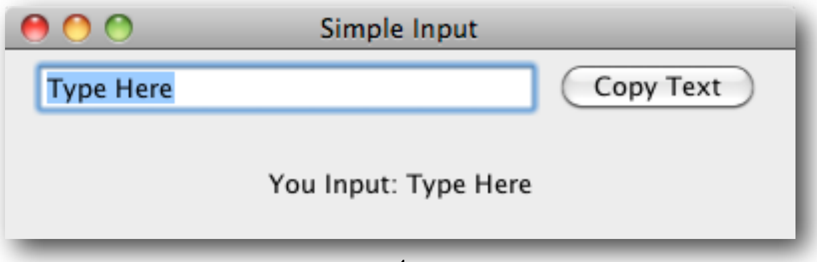
Questions (must complete these)

Q1. Modify HelloGoodBye so that two buttons are displayed, one to move forward in the message sequence, the other backward. The window should look like this:



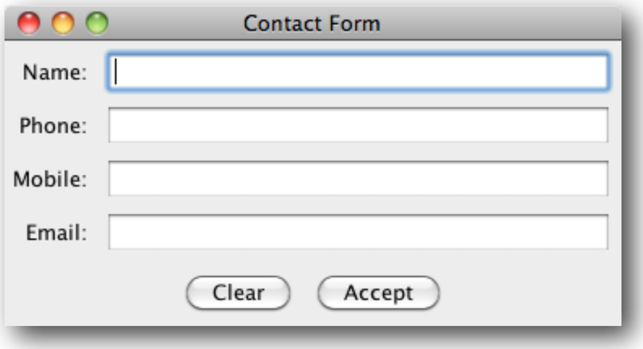
Hint: How do you layout the buttons and label? Use two panels and FlowLayout.

Q2. Write a Swing program to display this window:

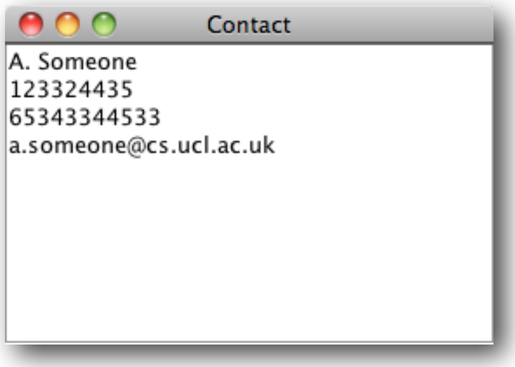


When the button is clicked, the text typed into the JTextField at the top of the window is copied into the label in the middle of the window. Note the position and size of the button.

Q3. Write a Swing program to display this window:



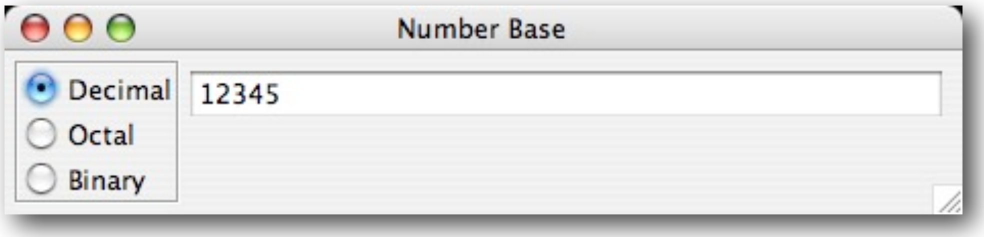
Clicking the accept button should open a second JFrame to display the information typed in:



```
A. Someone
123324435
65343344533
a.someone@cs.ucl.ac.uk
```

The input form should use JTextField's to accept the input. Use a combination of panels and the GridLayout to position the labels and text fields. The clear button should clear the text from the input fields.

Q4. Implement a Swing program with this interface:

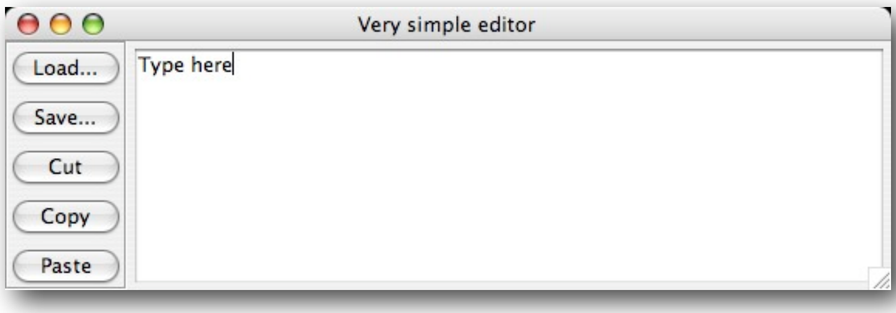


The buttons on the left are Radio Buttons (JRadioButton objects). When a number is entered in the JTextField on the right, and a radio button clicked, the number is converted to the new number base and displayed in place of the original. Note, only digits valid in the current number base can be typed into the JTextField.

Hints: See the Swing tutorial sections on radio buttons and text fields. Use an EtchedBorder on the panel holding the radio buttons, the BorderFactory class will create one for you. Set the insets on the border to move the radio buttons away from the edge. The radio buttons need to be in a ButtonGroup to function correctly. A JTextField can generate a KeyEvent when a character is typed and accepts KeyListeners. Class Integer will do all the work of number base conversion.

Optional Challenge

Q5. Look at the source code for the Very Simple Editor program (see the Moodle web page). When run the program looks like this:



This is a functional editor (compile and run the program to try it out). Spend some time studying the code, and find out how it works. Note how much is done by the Swing components. The JTextArea handles all the work of displaying text, doing input, using the mouse to select text, and even provides ready made cut, copy and paste methods. Also note how load and save work.

Rewrite this program, so that it includes the following changes:

- The buttons are replaced by a menu bar and menu(s). See the JMenuBar, JMenu and JMenuItem classes.
- New files can be opened for editing in a new window. This means creating a new JFrame for each window. In the case of the editor, the editor class (VerySimpleEditor) is a JFrame subclass so a new VerySimpleEditor object is needed for each window.
- A status bar at the bottom of the window that displays information such as the file name, the current line number and a symbol indicating if the file has been altered or not (a simple version of
- An undo command.

As an optional challenge, replace the JTextArea with the more powerful JTextPane or JEditorPane components and provide support for text formatting and different fonts. Use the Javadoc to find out more about these components.