Java Swing – An introduction to User Interface

Sources  
<https://docs.oracle.com/javase/tutorial/uiswing/>

<http://www.tutorialspoint.com/swing/swing_quick_guide.htm>

<http://cs.nyu.edu/~yap/classes/visual/03s/lect/l7/>

# Organization

Visual programming related classes can be categorized into 4 groups:

1. Component Classes – parts that can be placed on the screen
2. Container Classes – a subclass of Component; help hold things
3. Helper Classes – classes that we do not see directly, but help us organize and make pretty the visual designs that we do
4. Event Classes – responsible for handling our interactions with the computer

This lab, we’ll focus more on the designing of the interface first.

All of what we do is in the javax.swing library and java.awt library

# Frame

To create a user interface, you need to create either a frame or an applet to hold the user- interface components. This section introduces frames.

The **JFrame** class is responsible for creating a place for our user interface. To use it, we need to create one, and also have it be set as visible.

**Take a look at DemoFrame.java.**

This creates a new Frame which is visible. Line 12 of this program simple means that when you close the frame, the program ends as well.

Like any other window, you can move it, make it bigger/smaller, or minimize it.

**Take a look at DemoFrame2.java.**

**Again, this is creating a Frame which is visible. The difference is, now there is a title for the Frame.**

This is pretty useless for us. Who wants to see a small, useless screen?

The JFrame class has functions to set the initial size and the initial location.

**Take a look at DemoFrame3.java.**

You can also set position relative to another component (if we say that component is null, the computer centers the window). The title can be changed as well.

**Take a look at DemoFrame4.java.**

## **Practise**

1. Create a new frame, with the title as your name. Set its size as 500 by 500, and have it centered on the screen.

# Components

Components are parts of the screen we can see.

Some components we will see are:

* JLabel
* JButton
* JTextField
  + JPasswordField
  + JFormattedTextField
* JTextArea
* JRadioButton
* JCheckBox
* JComboBox
* JList

Adding a component to the frame is done exactly like that: adding to the frame.

**Take a look at DemoJComponent.java.**

A button is created, and the frame adds the button onto itself. This button is useless at this moment.

## Practise

1. Create a frame with a label saying your name.
2. Create a frame with a passwordfield.

# Helper #1 – Layout

Look again at DemoJComponent. We only added one button. Trying to add the other button means that the first button is no longer visible. If this were the case with all programs, we would not really want to use the computer because of lack of inefficiency.

The frame has no organization as to know how to place things on the screen.

Layouts are objects responsible for organizing the components in a container/frame (we’ll discuss container in a short while.

Some layouts we might use:

* FlowLayout
* Gridlayout
* BorderLayout
* GridBagLayout

Again, this has to do with the frame/container.

The frame must set its layout as a certain layout BEFORE adding components.

**Take a look at DemoFlowLayout.java.**

**FlowLayout when used arranges swing components from left to right until there’s no more space available. Then it begins a new row below it and moves from left to right again. Each component in a FlowLayout gets as much space as it needs and no more.**

**BorderLayout places swing components in the North, South, East, West and center of a container. You can add horizontal and vertical gaps between the areas.**

**GridLayout is a layout manager that lays out a container’s components in a rectangular grid. The container is divided into equal-sized rectangles, and one component is placed in each rectangle.**

**GridBagLayout is a layout manager that lays out a container’s components in a grid of cells with each component occupying one or more cells, called its display area. The display area aligns components vertically and horizontally, without requiring that the components be of the same size.**

**Try out the other layouts. Their constructors are given in the** [Java Docs](https://docs.oracle.com/javase/8/docs/api/)**, under java.awt.**

## Practise

1. Create a frame with 6 JRadioButton in with FlowLayout
2. Create a frame with 6 JCheckBox in with Gridlayout
3. Create a frame with 5 labels (each with a classmate’s name) with BorderLayout
4. Create a frame with 2 labels and 2 textfields (like a login user interface)

# Containers

Suppose that you want to place ten buttons and a text field in a frame. The buttons are placed in grid formation, but the text field is placed on a separate row. It is difficult to achieve the desired look by placing all the components in a single container. With Java GUI programming, you can divide a window into panels. Panels act as containers to group user interface components. You add the buttons in one panel, then add the panel to the frame.

Containers are subclasses of Component class, so we could put containers in containers.

**Take a look at DemoPanels.java.**

## Practise

1. Create a Calculator Interface (scientific).

# Inheritance

We can see how other classes are superclasses/subclasses of others. We could technically make our own.

**Take a look at DemoInheritanceFrame.java.**

## **Practise**

1. Make a CalculatorFrame class.