

Lab 8 – JComponents – posted 10/26, due 11/7

The program for this lab will *extend* JFrame, but will have no *paint(Graphics g)* method. Instead it will place various components on our frame in the program's constructor, and set and get values for the components.

Carefully follow the steps ahead:

1. Create the program class framework. Name your class **JFrameComponents**.

Import java.awt.*; and javax.swing.*;

Extend JFrame in your class declaration

Include a public variables section, a constructor and a main.

2. In the public variables section, create objects for
a JButton – this is a pushbutton
a JProgressBar – this is like a thermometer
a JSlider – looks like a radio volume control
a JLabel – a way of displaying text
and a JSpinner – a dial

only the JButton and JLabel need constructor parameters:

```
public JButton myButton = new JButton("Push");  
public JLabel greeting = new JLabel("Have a nice day");
```

everything else just needs a simple instantiation:

```
public JSpinner spinner1 = new JSpinner();
```

3. In the constructor of your program, set up the JFrame: setSize(..., setLocation(..., setDefaultCloseOperation(..., BUT NOT YET setVisible(true); *setVisible* will be last in the constructor.

4. In the constructor, something new (but very important) – create an object which helps manage the layout of components on the frame:

```
// create a layout manager  
FlowLayout layout = new FlowLayout();  
// attach it to frame  
setLayout( layout );
```

Layout managers are helper classes that work behind the scenes. *setLayout* is a JFrame method that uses a layout object to help place components on a frame. Once *setLayout* is used, you never have to worry about how the various components will look.

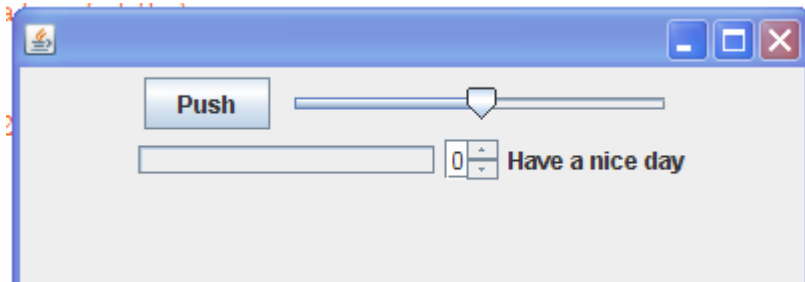
5. In the constructor, add all the component objects, like:

```
add( componentObject );
```

do that for each of the JComponent objects.

6. setVisible(true); as the last statement of your constructor.

7. Add a main that instantiates the JFrameComponents class, compile the program, and run to see if all of your components appear.



8. Look up the java pages for each control and check out the methods. Experiment with various methods for each object:

for example, if your slider object is named slider1, you might...

```
slider1.setMajorTickSpacing( 10 );
slider1.setPaintTicks( true );
slider1.setPaintLabels( true );
slider1.setValue( 99 );
add( slider1 );
```

note that the component's "look-and-feel" is set before it is added.

You can find these methods on the JSlider java page:

	Used to specify what label will be drawn at any given value.
void	setMajorTickSpacing (int n) This method sets the major tick spacing.
void	setMaximum (int maximum) Sets the models maximum property.
void	setMinimum (int minimum) Sets the models minimum property.
void	setMinorTickSpacing (int n) This method sets the minor tick spacing.
void	setModel (BoundedRangeModel newModel) Sets the model that handles the sliders three fundamental properties: minimum, maximum, value.
void	setOrientation (int orientation) Set the scrollbars orientation to either VERTICAL or HORIZONTAL.
void	setPaintLabels (boolean b) Determines whether labels are painted on the slider.
void	setPaintTicks (boolean b) Determines whether tick marks are painted on the slider.
void	setSnapToTicks (boolean b)

9. Each component has a *setValue* method. Use it.

10. As the last task in your constructor, create a while(true) loop. Since the constructor simply runs from start to finish and then exits, this is a way to keep the program going so that we can adjust the various components.

In the while(true) loop, call the getValue() method for the sliders and the spinner. Each one returns an integer. Print the values to the screen.

You can do something clever with the slider and progress bar (if your objects are named progressBar1 and slider1):

```
while( true )
{
    progressBar1.setValue( slider1.getValue() );
}
```

11. In that same while (true) loop, try this little mix of three methods (if your JLabel object is named label1, and your JSpinner object is named spinner1):

```
label1.setText( String.valueOf( spinner1.getValue() ) );
```

what does this do?

12. Experiment with the various components and their methods. Get used to setting and getting their values.

13. For every component that has a getValue method, print the components value to the screen, so that if you adjust the component while the program is running, the changing value is shown.

Make sure you submit your lab.