CSE 331
Section 6 cheat sheet

Swing

Parts of the GUI

Window

A first-class citizen of the graphical desktop

- Also called a *top-level container*
- In both AWT and Swing: a class that extends `Window`

`JFrame`

- `new JFrame(String title)` *make a new frame with optional title*
- `setVisible(true)` *make a frame appear on the screen*
- `add(Component comp)` *place the given component or container inside the frame*
- `setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)` *make it so that the program exits when the frame is closed*
- `setSize(int width, int height)` *gives the frame a fixed size in pixels*
- `pack()` *resize the frame to fit the components inside it snugly*

`JDialog`

- `new JDialog(Frame parent, String title, boolean modal)` *make a new JDialog with given parent and title. If* `modal` *is set, the parent will be locked until the dialog is closed*
- `JOptionPane.showMessageDialog(parent, message)` *static method to pop up a dialog with just a message and OK button*
- `JOptionPane.showConfirmDialog(parent, message)` *static method to pop up a dialog with a message and Yes and No buttons*
- `JOptionPane.showInputDialog(parent, message)` *static method to pop a dialog with a message and a text field for entering information*

and many more…

Component

A GUI widget that resides in a window

- Also called a *control* in many other languages and graphical toolkits
- In Swing: a class that extends `JComponent` (which extends `Component` and `Container` from AWT)

JComponent Properties:

name	type	description
background	`Color`	background color behind component
foreground	`Color`	foreground color of component
border	`Border`	border line around component
enabled	`boolean`	whether it can be interacted with
focusable	`boolean`	whether key text can be typed on it
font	`Font`	font used for text in component
height, width	`int`	component's current size in pixels

visible	`boolean`	whether component can be seen
tooltip text	`String`	text show when hovering mouse
size, minimum / maximum / preferred size	`Dimension`	various sizes, size limits, or desired sizes that the component may take

Each property has a `get` (or `is`) accessor method, and a `set` modifier method.

`JLabel`

- `new JLabel(String text)` *creates a new label with the given text*
- `getText()` *returns the text showing on the label*
- `setText()` *sets label's text*

`JButton`

- `new JButton(String text)` *creates a new button with text*
- `getText()` *returns the text showing on the button*
- `setText(String text)` *sets button's text*
- `setMnemonic(char)` *assigns the given character as a hotkey for this button, activated when keyboard focus is on the appropriate component and Alt+char is pressed*

`JTextArea`

- `new JTextArea(int lines, int columns)` *create a new text area with preferred size for the given number of lines and columns*

`JTextField`

(Also useful: `JFormattedTextField`, which can enforce formatting of entered text)

- `new JTextField(int columns)` *create a new field, the given number of columns wide*

and many more…

Container

A logical grouping for storing components

- In Swing: containers are just another type of component
- Used to create a hierarchy of components inside components inside components… all inside a top-level window
- `JFrame` can also be used as a container (thought it doesn't extend `JComponent`)

Useful container methods:

- `add(Component comp)` *places the given component inside this container*
- `add(Component comp, Object info)` *places the given component inside this container*
- `remove(Component comp)` *removes the given component from this container*
- `setLayout(LayoutManager mgr)` *use a particular layout to position components*
- `validate()` *refreshes the layout if changes have been made after the container is onscreen*

`JPanel`

A general container

`JScrollPane`

Used to add scroll bars to a component when it becomes too large

- `new JScrollPane(Component comp)`

and many more…

Layout Manager
Used to position components within a container

`FlowLayout`
- Treats container as a left-to-right, top-to-bottom "paragraph"
- Components are given preferred height and width
- Components are positioned in the order added
- If too long, components wrap around to the next line
- The default layout for containers other than `JFrame`

`BorderLayout`
- Divides container into five regions
- NORTH and SOUTH regions expand to fill container horizontally, and use the components' preferred heights
- WEST and EAST regions expand to fill container vertically, and use preferred widths
- CENTER uses all space not occupied by other regions
- The default layout for `JFrame`

`GridLayout`
- Treats container as a grid of equally-sized rows and columns
- Preferred sizes are ignored to make grid cells the same width and height

`BoxLayout`
- Align components in container in a single row or column
- Components use preferred size and preferred alignment

and many more…

The Event System
Responses to user interactions with the GUI are handled using AWT's event system. A user action generates an `EventObject`, which is passed to any `EventListener` objects attached to that component. `ActionEvent` and `ActionListener` are commonly used implementations. Button clicks, menu clicks, check box selections, and many other actions generate `ActionEvent` objects.

Event
Object that contains detailed information about the event
- `getSource()` *returns a reference to the object to which the event occurred*
- other getter methods that describe state of the component when the event occurred

`ActionEvent`
- `getActionCommand()` *returns the command string associated with this action*
- `getWhen()` *returns a timestamp of when this event occurred*

and many more…

Listener
Object that can be attached to a component to listen for events. Contains a method that is automatically called when an event occurs

`ActionListener`

```
public class name implements ActionListener {
      public void actionPerformed(ActionEvent event) {
            code to handle the event;
      }
}
```

- Often implemented as a nested class of some GUI class
- `addActionListener(new` **name** `())` *attaches an instance of this listener to a component, using the component's* `addActionListener` *method*

and many more…

Section Exercise

Identify the components, containers and layouts used in the following frames: