Cheat Sheet based on:

Swing/GUI Cheat Sheet at Williams College
http://eventfuljava.cs.williams.edu/s04/handouts/SwingGUICheatSheet.html

Displaying a Swing component

Construct and initialize the component.

```
button = new JButton ("Bu tto nLa bel ");
```

Add it to the content pane of the window or to a JPanel that is added to the display.

```
getCon ten tPa ne( ).add (button);
```

Import javax.swing. *and sometimes also java.awt.* at the beginning of the class creating the components.

```
import javax.s wing.; import java.awt.
```

Getting events from GUI component

Declare that the class handling the event implements the appropriate listener interface.

```
implements Action Lis tener
```

Define the method that the listener interface requires.

```
public void action Per formed (Actio nEvent event)
```

Add a listener appropriate for the component to the component.

```
button.ad dAc tio nLi stener (this);
```

Import java.awt.event. *(and occasionally javax.swing.event.)* at the beginning of the class that is the listener.

```
import javax.s wing.; import java.awt.
```

Finding out which component sent the event

When the listener method is called, you can find out which component sent the event by calling getSource() on the event:

```
public void action Per formed (Actio nEvent event)
 {
  Object theButton = event.getSource();
  if (theButton == framed Cir cle Button) {
    // Create a framed circle
  }
}
```

If a method returns a String, remember to compare the result using the equals method, not ==:

```
aMenu.g et Sel ect edI tem ().e quals ("A value");
```

Containers

JPanel constructor:	`new JPanel ()`
Define the type of layout:	`void setLayout (Layou tMa nager lm)`
Add an object to a container: (FlowLayout or GridLayout)	`void add (Component c)`
Add an object to a container: (BorderLayout)	`void add (Component c, int position)`

Both JPanel and the object obtained by sending getContentPane() to a WindowController object are containers (and have type Container). These methods are available for all containers.
For BorderLayouts, position may be either `Border Lay out.NORTH`, `Border Lay out.SOUTH`, `Border Lay out.EAST`, `Border Lay out.WEST`, or `Border Lay out.CENTER`.

Layout Managers

BorderLayout constructor:	`new Border Layout ()`
FlowLayout constructor:	`new FlowLayout ()`
GridLayout constructor:	`new GridLayout (int rows, int cols)` `new GridLayout (int rows, int cols, int colSpa cing, int rowSpa cing)`

BorderLayout is the default layout for WindowController, whereas FlowLayout is default for JPanel.

GUI Components - General

The following methods can be applied to any Component:

```
void setFont (Font f)
void setFor eground (Color c)
void setBac kground (Color c)
```

To construct a font use:

```
new Font (String name, int style, int size)
```

Style can be one of the following:

```
Font.BOLD
Font.ITALIC
Font.PLAIN
Font.BOLD+Font.ITALIC
```

GUI Components - JButton

Constr-uctor:	`new JButton (String s)`
General Methods:	`String getText ( )` `void setText (String s)`
Listener Interface:	`Action Lis tener`
Adding the listener:	`void addAct ion Lis tener (Actio nLi stene r al)`
Listening Method:	`void action Per formed (Actio nEvent e)`

GUI Components - JComboBox

Constr-uctor and Initializ-ation:	`new JComboBox ( )` `void addItem (Object item)`

General Methods:	`Object getSel ect edItem ( )`
	`String text=(String)menu.getSelectedItem();`
	`int getSel ect edIndex ( )`

Listener Interface:	`ItemLi stener` `Action Lis tener`
Adding the listener:	`void addIte mLi stener (ItemL istener il)` `void addAct ion Lis tener (Actio nLi stener al)`

GUI Components - JComboBox (cont)

Listening Method:	`void itemSt ate Changed (ItemEvent e )` `void action Per formed (Actio nEvent e)`

About methods:

`getSel ect edItem ( )` returns the selected item

`(String) menu.g etS ele cte dItem ( );` is a typecast which treats the above returned value as a String

`int getSel ect edIndex ( )` returns the index of the selected item.

About the listeners:

This component can hear the user making a menu selection dependong on the chosen interface. Be consistent in your choice of listener interface, adding method, and listening method.

GUI Components - JLabel

Constructors:	`new JLabel (String s)` `new JLabel (String s, int align)`
General Methods:	`void setText (String s)` `String getText ( )`
Listener Interface:	No listeners available.

align can be either `JLabel.RIGHT`, `JLabel.LEFT` or `JLabel.CEN TER`.

GUI Components - JSlider

Constr-uctor:	`new JSlider (int orient ation, int minimum ,` `int maximum, int initia lValue)`
General Methods:	`void setValue (int newVal)` `int getValue ( )`
Listener Interface:	`Change Lis tener`
Adding the Listener:	`addCha nge Lis tener (Chang eLi stener al)`
Listening Method:	`void stateC hanged (Chang eEvent e)`

orientation can be either `JSlide r.H ORI ZONTAL` or `JSlide r. V ERTICAL`.

GUI Components - JTextField

Constructors:	`new JTextField (String s)`
General Methods:	`void setText (String s)` `String getText ( )`
Listener Interface:	`Action Lis tener`
Adding the Listener:	`addAct ion Lis tener (Actio nLi stene r al)`
Listening Method:	`void action Per formed (Actio nEvent e)`