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

```java
// Create a framed circle
if (theButton == framedCircleButton) {
    // Create a framed circle
}
```

Add an object to a container:
- void add (Component c)
- void add (Component c, int position)
- void add (Component c, int position, int colSpacing, int rowSpacing)

Both JPanel and the object obtained by sending getContentPane() to a WindowController are containers (and have type Container). These methods are available for all containers.

For BorderLayouts, position may be either BorderLayout.NORTH, BorderLayout.SOUTH, BorderLayout.EAST, BorderLayout.WEST, or BorderLayout.CENTER.

GUI Components - General

The following methods can be applied to any Component:

- void setFont (Font f)
- void setForeground (Color c)
- void setBackground (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

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.

```java
button = new JButton ("ButtonLabel");
```

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

```java
getContentPane().add (button);
```

Import java.awt, and sometimes also java.awt.event at the beginning of the class creating the components.

```java
import java.awt; import java.awt.event;
```

Getting events from GUI component

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

```java
implements ActionListener
```

Define the method that the listener interface requires.

```java
public void actionPerformed (ActionEvent event) {
    // Create an action
}
```

Add a listener appropriate for the component to the component.

```java
button.addActionListener (this);
```

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

```java
import java.awt.event;
```

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:

```java
public void actionPerformed (ActionEvent event) {
    Object theButton = event.getSource();
    if (theButton == framedCircleButton) {
        // Create a framed circle
    }
}
```

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

```java
aMenu.getSelectedItem().equals ("A value");
```
## GUI Components - JButton

**Constructor:**
```
new JButton (String s)
```

**General Methods:**
```
String getText ( )
void setText (String s)
```

**Listener Interface:**
```
ActionListener
```

**Adding the listener:**
```
void addActionListener (ActionListener al)
```

**Listening Method:**
```
void actionPerformed (ActionEvent e)
```

**About methods:**
- `getText ( )` returns the text of the button.
- `setText (String s)` sets the text of the button.

**About the listeners:**
This component can hear the user clicking the button in response to an event. Be consistent in your choice of listener interface, adding method, and listening method.

## GUI Components - JComboBox

**Constructor and Initialization:**
```
new JComboBox ( )
void addItem (Object item)
```

**General Methods:**
```
Object getSelectedItem ( )
String text=(String)menu.getSelectedItem();
int getSelectedIndex ( )
```

**Listener Interface:**
```
ItemListener
```

**Adding the listener:**
```
void addItemListener (ItemListener il)
void addActionListener (ActionListener al)
```

**Listening Method:**
```
void itemStateChanged (ItemEvent e)
void actionPerformed (ActionEvent e)
```

**About methods:**
- `getText ( )` returns the selected item
- `(String) menu.getSelectedItem ();` is a typecast which treats the above returned value as a String
- `getSelectedIndex ( )` returns the index of the selected item.

**About the listeners:**
This component can hear the user making a menu selection depending 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.CENTER.
```

## GUI Components - JSlider

**Constructor:**
```
new JSlider (int orientation, int minimum, int maximum, int initialValue)
```

**General Methods:**
```
void setValue (int newVal)
int getValue ( )
```

**Listener Interface:**
```
ChangeListener
```

**Adding the Listener:**
```
addChangeListener (ChangeListener cl)
```

**Listening Method:**
```
void stateChanged (ChangeEvent e)
```

**orientation can be either:**
```
JSlider.HORIZONTAL or JSlider.VERTICAL.
```

## GUI Components - JTextField

**Constructors:**
```
new JTextField (String s)
```

**General Methods:**
```
void setText (String s)
String getText ( )
```

**Listener Interface:**
```
ActionListener
```

**Adding the Listener:**
```
addActionListener (ActionListener al)
```

**Listening Method:**
```
void actionPerformed (ActionEvent e)
```

**By NeonKnightOA**

cheatography.com/neonknightoa/

Published 11th November, 2015.
Last updated 11th November, 2015.

Sponsored by CrosswordCheats.com
Learn to solve cryptic crosswords!
http://crosswordcheats.com