CoSc 10403 - Selected Java Classes

The diagram above shows a subset of the Swing components and their inheritance hierarchy.

**ActionEvent**

**Constructor Summary**
- none generated from button clicks

**Some Useful Methods**
- public void addActionListener(ActionListener l)
  - Adds the specified action listener to receive action events from the specified component.
- public Object getSource()
  - Returns a reference to the object that generated the event.
- public void actionPerformed(ActionEvent e)
  - Invoked when an action occurs.
AudioClip

Constructor Summary
- none
call method getAudioClip on applet

Some Useful Methods
public void loop()
Plays the audio clip continuously
public void play()
Plays the audio clip through one time.
public void stop()
Stops playing the audio clip.

BorderLayout Class

Class Constants
BorderLayout.CENTER or Center The middle of container.
BorderLayout.EAST or East The right side of container.
BorderLayout.NORTH or North The top of container.
BorderLayout.SOUTH or South The bottom of container.
BorderLayout.WEST or West The left side of container.

Class Variables
hGap The size of the horizontal gap
vGap The size of the vertical gap

Constructor Summary
BorderLayout()
Constructs a new border layout with no gaps between components.
BorderLayout(int hgap, int vgap)
Constructs a border layout with the specified gaps between components.

Some Useful Methods
public int getAlignment()
Returns an integer corresponding to the current alignment of this BorderLayout object.
public int getHgap()
Returns the current horizontal gap for this BorderLayout.

public int getHgap()
    Returns the current vertical gap for this BorderLayout.

public void setVgap(int v)
    Sets the vertical gap of this BorderLayout to the specified value.

public void setHgap(int h)
    Sets the horizontal gap of this BorderLayout to the specified value.

**ButtonGroup class**

*Constructor Summary*

**ButtonGroup()**

Creates a new ButtonGroup

*Some Useful Methods*

**public void add(AbstractButton b)**

Adds the button b to the group.

**public int getButtonCount()**

Returns the number of buttons in the group.

**public Enumeration getElements()**

Returns all the buttons that are participating in this group.

**public boolean isSelected(ButtonModel m)**

Returns whether a ButtonModel is selected. True if selected, false otherwise.

**public void remove(AbstractButton b)**

Removes the button from the group.

**public void setSelected(ButtonModel m, boolean b)**

Sets the selected value for the ButtonModel.

**Color Class**

*Class Constants*

<table>
<thead>
<tr>
<th>Color.BLACK</th>
<th>Color.black</th>
<th>Color.RED</th>
<th>Color.red</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color.ORANGE</td>
<td>Color.orange</td>
<td>Color.GREEN</td>
<td>Color.green</td>
</tr>
<tr>
<td>Color.CYAN</td>
<td>Color.cyan</td>
<td>Color.BLUE</td>
<td>Color.blue</td>
</tr>
<tr>
<td>Color.GRAY</td>
<td>Color.gray</td>
<td>Color.WHITE</td>
<td>Color.white</td>
</tr>
<tr>
<td>Color.Dark_GRAY</td>
<td>Color.darkGray</td>
<td>Color.YELLOW</td>
<td>Color.yellow</td>
</tr>
<tr>
<td>Color.LIGHT_GRAY</td>
<td>Color.LightGray</td>
<td>Color.MAGENTA</td>
<td>Color.magenta</td>
</tr>
</tbody>
</table>
**Constructor Summary**

public Color(int r, int g, int b)

Creates a Color object with specified red, green, and blue

**Some Useful Methods**

public abstract void setColor(Color c) -- *from the Graphics class*

Sets this graphics context's current color to the specified color.

---

**DefaultListModel Class**

**Constructor Summary**

DefaultListModel()

**Some Useful Methods**

public void add (index, element)

Inserts the specified element at the specified position in this list.

public void addElement(obj)

Adds the specified component to the end of this list.

public int capacity()

Returns the current capacity of this list.

public void clear()

Removes all of the elements from this list.

public boolean contains(elem)

Tests whether the specified object is a component in this list.

public Object get(index)

Returns the element at the specified position in this list.

public Object getElementAt(index)

Returns the component at the specified index.

public int getSize()

Returns the number of components in this list.

public void insertElementAt(obj, index)

Inserts the specified object as a component in this list at the specified index.

public void remove(index)

Removes the element at the specified position in this list.

public void removeAllElements()

Removes all components from this list and sets its size to zero.
public void removeElement(obj)
    Removes the first (lowest-indexed) occurrence of the argument from this list.

Double Class
Constructor Summary
Double(number)
    Creates an instance of Double object with the specified number.

Some Useful Methods
public double Double.parseDouble(String)
    Returns a double type of the number stored in the String.

FlowLayout Class
Class Constants
FlowLayout.CENTER  FlowLayout.RIGHT  FlowLayout.LEFT

Class Variables
hGap  The size of the horizontal gap
vGap  The size of the vertical gap

Constructor Summary
FlowLayout()
    Constructs a new FlowLayout with center alignment and default 5-unit horizontal and vertical gap.
FlowLayout(int align)
    Constructs a new FlowLayout with specified alignment and default 5-unit horizontal and vertical gap.
    Alignment Constants may be:
    FlowLayout.CENTER
    FlowLayout.LEADING - each row of components is to be justified to leading edge of container
    FlowLayout.LEFT
    FlowLayout.RIGHT
    FlowLayout.TRAILING - each row of components is to be justified to trailing edge of container
FlowLayout(int align, int hgap, int vgap)
    Creates a new flow layout manager with the indicated alignment and the indicated
Some Useful Methods

```java
public int getAlignment()
    Gets the alignment for this layout
```
```java
public void setAlignment(int align)
    Sets the alignment for this layout.
```

Font Class

Class Constants

- Font.PLAIN
- Font.BOLD
- Font.ITALIC

Constructor Summary

```java
Font(String name, int style, int size)
    Creates a new Font where name is "Serif", "Sanserif"or Monospaced", style is one of the above class constants, and size is in integer points.
```

Some Useful Methods

```java
public void setFont(Font f)
    Sets the font for the object to the specified font.
```

GridLayout Class

Class Variables

- hGap: The size of the horizontal gap
- vGap: The size of the vertical gap
- columns: The number of columns for this GridLayout
- rows: The number of rows for this GridLayout

Constructor Summary

```java
GridLayout()
    Creates a grid layout with a default of one column per component, in a single row.
```
```java
GridLayout(int rows, int cols)
    Creates a grid layout with the specified number of rows and columns.
```
GridLayout(int rows, int cols, int hgap, int vgap)
    Creates a grid layout with the specified number of rows and columns.

Some Useful Methods
public int getColumns()
    Returns an integer corresponding to the number of columns in this GridLayout.
public int getHgap()
    Returns the current horizontal gap for this GridLayout.
public int getRows()
    Returns an integer corresponding to the number of rows in this GridLayout.
public int getVgap()
    Returns the current vertical gap for this GridLayout.
public void setColumns(int cols)
    Sets the number of columns in this GridLayout to the specified value.
public void setHgap(int gap)
    Sets the current horizontal gap for this GridLayout to the specified value.
public void setRows(int rows)
    Sets the current number of rows for this GridLayout to the specified value.
public void setVgap(int gap)
    Sets the current vertical gap for this GridLayout to the specified value.

ImageIcon Class
Constructor Summary
ImageIcon()
    Creates a new ImageIcon object with the specified image.

Some Useful Methods
public Image getImage()
    Return the image of the icon.
public void setImage(Image i)
    Changes the image of the icon.

Integer Class
Constructor Summary
Integer(number)
Creates an instance of Integer object with the specified number.

**Some Useful Methods**
public int Integer.parseInt(String)
Returns a int type of the number stored in the String.

**ItemEvent**

**Constructor Summary**
- none Generated from selecting/deselecting radio buttons/checkboxes/lists

**Some Useful Methods**
public void addItemListener(ItemListener l)
Adds the specified item listener to receive item events from the specified component.
public Object getSource()
Returns a reference to the object that generated the event.
public void itemStateChanged(ItemEvent e)
Invoked when an item event occurs.

**JApplet Class**

**Some Useful Methods**
public void add(component)
Adds the specified component to the container
public AudioClip getAudioClip(getCodeBase(), filename)
Gets an AudioClip in the file system
public Image getImage(getCodeBase(), filename)
Gets an Image in the file system
public void remove(component)
Removes the specified component from this container.
public void setSize(width, height)
set the size of this container
public void setLayout(LayoutManager)
sets the Layout Manager.
**JButton Class**

*Fires an ActionEvent when pressed - the event is sent after the button is released, and only if the cursor is still over the button.*

**Constructor Summary**

- `JButton()`  
  Creates a button with no set text or icon.
- `JButton(Action a)`  
  Creates a button where properties are taken from the Action supplied.
- `JButton(Icon icon)`  
  Creates a button with an icon.
- `JButton(String text)`  
  Creates a button with text.
- `JButton(String text, Icon icon)`  
  Creates a button with initial text and an icon.

**Some Useful Methods - (derived from AbstractButtonClass)**

- `public void addActionListener(ActionListener l)`  
  Adds an ActionListener to the button.
- `public void addMouseListener(MouseListener l)`  
  Adds a MouseListener to the button.
- `public String getActionCommand()`  
  Returns the action command for this button.
- `public Icon getIcon()`  
  Returns the default icon.
- `public String getName()`  
  Returns the name of the component.
- `public String getText()`  
  Returns the button's text.
- `public void remove(Component comp)`  
  Removes this component from the container.
- `public void remove(int index)`  
  Removes the component in the specified location from the container.
- `public void removeActionListener(ActionListener l)`  
  Removes an ActionListener from the button.
- `public void removeMouseListener()`  
  Removes the MouseListener from this button.
- `public void requestFocus()`  
  Requests that this component gets the input focus.
public void setActionCommand(String actionCommand)
    Sets the action command for this button.
public void setBackground(Color bg)
    Sets the background color of this component.
public void setDisabledIcon(Icon disabledIcon)
    Sets the disabled icon for the button.
public void setEnabled(boolean b)
    Enables (or disables) the button.
public void setFont(Font font)
    Sets the font for this component.
public void setForeground(Color fg)
    Sets the foreground color of this component.
public void setIcon(Icon defaultIcon)
    Sets the button's default icon. This icon is also used as the "pressed"
    and "disabled" icon if there is no explicitly set pressed icon.
public void setHorizontalAlignment(int alignment)
    Sets the horizontal alignment of the icon and text.
public void setHorizontalTextPosition(int textPosition)
    Sets the horizontal position of the text relative to the icon.
public void setName(String s)
    Sets the name of the component to the specified string.
public void setRolloverIcon(Icon rolloverIcon)
    Sets the rollover icon for the button.
public void setText(String text)
    Sets the button's text.
public void setVerticalAlignment(int alignment)
    Sets the vertical alignment of the icon and text.
public void setVerticalTextPosition(int textPosition)
    Sets the vertical position of the text relative to the icon.
public void setVisible(boolean aFlag)
    Makes the component visible or invisible.

**JCheckBox Class**

**Constructor Summary**

JCheckBox()
    Creates an initially unselected check box button with no text, no icon.
JCheckBox(Action a)
Creates a check box where properties are taken from the Action supplied.

`JCheckBox(Icon icon)`
Creates an initially unselected check box with an icon.

`JCheckBox(Icon icon, boolean selected)`
Creates a check box with an icon and specifies whether or not it is initially selected.

`JCheckBox(String text)`
Creates an initially unselected check box with text.

`JCheckBox(String text, boolean selected)`
Creates a check box with text and specifies whether or not it is initially selected.

`JCheckBox(String text, Icon icon)`
Creates an initially unselected check box with the specified text and icon.

`JCheckBox(String text, Icon icon, boolean selected)`
Creates a check box with text and icon, and specifies whether or not it is initially selected.

**Some Useful Methods**

`public void addItemListener(ItemListener l)`
Adds an ItemListener to the checkbox.

`public void addMouseListener(MouseListener l)`
Adds a MouseListener to the checkbox.

`public String getActionCommand()`
Returns the action command for this checkbox.

`public Icon getIcon()`
Returns the default icon.

`public String getName()`
Gets the name of the component.

`public Icon getSelectedIcon()`
Returns the selected icon for the checkbox

`public String getText()`
Returns the checkbox's text.

`public boolean isSelected()`
Returns the state of the checkbox. True if selected, false if it's not.

`public void removeItemListener(ItemListener l)`
Removes an ItemListener from the button.

`public void requestFocus()`
Requests that this component gets the input focus.

`public void setActionCommand(String actionCommand)`
Sets the action command for this checkbox.

`public void setBackground(Color bg)`
Sets the background color of this component.
public void setEnabled(boolean b)
    Enables (or disables) the checkbox

public void setFont(Font font)
    Sets the font for this component.

public void setForeground(Color fg)
    Sets the foreground color of this component.

public void setIcon(Icon defaultIcon)
    Sets the checkbox's default icon. This icon is also used as
    the "pressed" and "disabled" icon if there is no explicitly set pressed icon.

public void setName(String name)
    Sets the name of the component to the specified string.

public void setSelected(boolean b)
    Sets the state of the checkbox
    Note that this method does not trigger an Event.

public void setSelectedIcon(Icon selectedIcon)
    Sets the selected icon for the checkbox.

public void setText(String text)
    Sets the checkbox's text.

**JComboBox Class**

**Constructor Summary**

JComboBox()
    Creates a JComboBox with a default data model.

JComboBox(ComboBoxModel aModel)
    Creates a JComboBox that takes it's items from an existing ComboBoxModel.

JComboBox(Object[] items)
    Creates a JComboBox that contains the elements in the specified array.

JComboBox(Vector items)
    Creates a JComboBox that contains the elements in the specified Vector.

**Some Useful Methods**

public void addActionListener(ActionListener l)
    Adds an ActionListener to the combobox

public void addItem(Object anObject)
    Adds an item to the item list

public void addItemListener(ItemListener aListener)
    Adds an ItemListener.
public String getActionCommand()
    Returns the action command that is included in the event sent to action listeners.
public Object getItemAt(int index)
    Returns the list item at the specified index.
    If index is out of range (less than zero or greater than or equal to size) it will return null.
public int getItemCount()
    Returns the number of items in the list.
public int getSelectedIndex()
    Returns the position of the current selected item.
public Object getSelectedItem()
    Returns the current selected item.
public void insertItemAt(Object anObject, int index)
    Inserts an item into the item list at a given index.
public void removeActionListener(ActionListener l)
    Removes the ActionListener
public void removeAllItems()
    Removes all items from the item list.
public void removeItem(Object anObject)
    Removes an item from the item list
public void removeItemAt(int anIndex)
    Removes the item at anIndex.
public void removeItemListener(ItemListener aListener)
    Removes an ItemListener.
public void setActionCommand(String aCommand)
    Sets the action command that should be included in the event sent to action listeners.
public void setBackground(Color bg)
    Sets the background color of this component.
public void setEditable(boolean aFlag)
    Determines whether the JComboBox field is editable.
public void setEnabled(boolean b)
    Enables (or disables) the button.
public void setFont(Font font)
    Sets the font for this component.
public void setForeground(Color fg)
    Sets the foreground color of this component.
public void setSelectedIndex(int anIndex)
    Selects the item at index anIndex.
public void setSelectedItem(Object anObject)
    Sets the selected item in the combo box display area to the object in the argument
**JComponent Class**

*Some Useful Methods*

```java
public void requestFocus()
    Requests that this Component get the input focus.

public void setBackground(Color bg)
    Sets the background color of this component.

public void setEnabled(boolean b)
    Enables (or disables) the component

public void setFont(Font font)
    Sets the font for this component.

public void setForeground(Color fg)
    Sets the foreground color of this component.

public void setVisible(boolean aFlag)
    Makes the component visible or invisible
```

**JFrame Class**

*Constructor Summary*

```java
JFrame()
    Constructs a new frame that is initially invisible.

JFrame(String title)
    Creates a new, initially invisible Frame with the specified title.
```

*Some Useful Methods*

```java
public void add(Object)
    Adds the Object to the frame based on the layout manager specified.

public void pack()
    Sizes the JFrame to fit the components displayed within the fram.

public void remove(Component comp)
    Removes the specified component from this container.

public void setSize(int w, int h)
    Sets the width and height of the frame.

public void setLayout(LayoutManager manager)
    Sets the layout for the frame.

public void setLocation(int x, int y)
```
Sets the upper-left corner of the screen of where to display the frame.

```java
public void setBounds(int x, int y, int w, int h)
    Sets the location of the top-left corner as specified by x and y, and the
    size is specified by width and height.
```

Sets the location of the top-left corner as specified by x and y, and the
size is specified by width and height.

```java
public void setVisible(boolean)
    Shows the frame if the argument is true; otherwise hides it.
```

### JLabel Class

**Constructor Summary**

- `JLabel()`
  - Creates a JLabel instance with no image and with an empty string for the title.
- `JLabel(Icon image)`
  - Creates a JLabel instance with the specified image.
- `JLabel(Icon image, int horizontalAlignment)`
  - Creates a JLabel instance with the specified image and horizontal alignment.
- `JLabel(String text)`
  - Creates a JLabel instance with the specified text.
- `JLabel(String text, Icon icon, int horizontalAlignment)`
  - Creates a JLabel instance with the specified text, image, and horizontal alignment.
- `JLabel(String text, int horizontalAlignment)`
  - Creates a JLabel instance with the specified text and horizontal alignment.

### Some Useful Methods

- `public Icon getIcon()`
  - Returns the graphic image (glyph, icon) that the label displays.
- `public String getText()`
  - Returns the text string that the label displays.
- `public void setIcon(Icon icon)`
  - Defines the icon this component will display.
- `public void setText(String text)`
  - Defines the single line of text this component will display.

### JList Class

**Constructor Summary**

- `JList()`
Constructs a JList with an empty model.

JList(ListModel dataModel)
   Constructs a JList that displays the elements in the specified, non-null model.

JList(Object[] listData)
   Constructs a JList that displays the elements in the specified array.

JList(Vector listData)
   Constructs a JList that displays the elements in the specified Vector.

Some Useful Methods

public int getSelectedIndex()
   Returns the first selected index; returns -1 if there is no selected item.

public int[] getSelectedIndices()
   Returns an array of all of the selected indices in increasing order.

public Object getSelectedValue()
   Returns the first selected value, or null if the selection is empty.

public Object[] getSelectedValues()
   Returns an array of the values for the selected cells. The returned values are
   sorted in increasing index order.

public boolean isSelectedIndex(int index)
   Returns true if the specified index is selected.

public boolean isSelectionEmpty()
   Returns true if nothing is selected

public void setSelectedIndex(int index)
   Selects a single cell.

public void setSelectedIndices(int[] indices)
   Selects a set of cells.

public void setSelectedValue(Object anObject, boolean shouldScroll)
   Selects the specified object from the list

public void setSelectionBackground(Color selectionBackground)
   Sets the background color for selected cells.

public void setSelectionForeground(Color selectionForeground)
   Sets the foreground color for selected cell

public void setSelectionMode(int selectionMode)
   Determines whether single-item or multiple-item selections are allowed.
   The following selectionMode values are allowed:
   ListSelectionModel.SINGLE_SELECTION Only one list index can be selected at a time.
   In this mode the setSelectionInterval and
   addSelectionInterval methods are equivalent, and
   only the second index argument is used.
ListSelectionModel.SINGLE_INTERVAL_SELECTION One contiguous index interval can be selected at a time. In this mode setSelectionInterval no restriction on what can be selected and addSelectionInterval are equivalent.

ListSelectionModel.MULTIPLE_INTERVAL_SELECTION In this mode, there's This is the default.

public void setVisibleRowCount(int) Sets the number of Jlist rows to display.

**JPanel Class**

*Constructor Summary*

JPanel() Creates a new JPanel with a double buffer and a flow layout.

JPanel(LayoutManager layout) Creates a new JPanel with the specified layout manager

*Some Useful Methods*

public void add(component) Adds the component to the panel using the layout manager.

public void setBackground (Color) Changes the background color of the panel.

public void setLayout(layoutManager) Sets the layout manager to the one specified.

**JPasswordField Class**

*Constructor Summary*

JPasswordField() Constructs a new JPasswordField, with default document, null starting text string, and 0 col width.

JPasswordField(Document doc, String txt, int columns) Constructs a new JPasswordField that uses the given text storage model and the given number of cols.

JPasswordField(int columns) Constructs a new empty JPasswordField with the specified number of columns.

JPasswordField(String text)
Constructs a new JPasswordField initialized with the specified text.
JPasswordField(String text, int columns)
Constructs a new JPasswordField initialized with the specified text and columns

Some Useful Methods
public char[] getPassword()
Returns the text contained in this TextComponent

JRadioButton Class
Constructor Summary
JRadioButton()
Creates an initially unselected radio button with no set text.
JRadioButton(Icon icon)
Creates an initially unselected radio button with the specified image but no text.
JRadioButton(Icon icon, boolean selected)
Creates a radio button with the specified image and selection state, but no text.
JRadioButton(String text)
Creates an unselected radio button with the specified text.
JRadioButton(String text, boolean selected)
Creates a radio button with the specified text and selection state.
JRadioButton(String text, Icon icon)
Creates a radio button that has the specified text and image, and that is initially unselected.
JRadioButton(String text, Icon icon, boolean selected)
Creates a radio button that has the specified text, image, and selection state.

Some Useful Methods
public String getText()
Returns the text for the radio button.
public void isSelected()
Returns whether or not the radio button is selected.
public void setBackground(Color)
Sets the background color.
public void setEnabled(boolean)
Enables or disables the radio button.
public void setForeground(Color)
Sets the foreground color.
public void setSelectionIcon(ImageIcon)
Sets which image to display when the radio button is selected.

**JScrollPane Class**

**Constructor Summary**

- JScrollPane()
  - Creates empty JScrollPane - both horizontal and vertical scrollbars appear when needed.

- JScrollPane(Component view)
  - Creates a JScrollPane that displays the contents of the specified component - both horizontal & vertical scrollbars appear whenever the component's contents are larger than the view.

**JTextArea Class**

**Constructor Summary**

- JTextArea()
  - Constructs a new JTextArea.

- JTextArea(Document doc, String text, int rows, int columns)
  - Constructs a new JTextArea with the specified number of rows and columns, and the given model.

- JTextArea(int rows, int columns)
  - Constructs a new empty JTextArea with the specified number of rows and columns.

- JTextArea(String text)
  - Constructs a new JTextArea with the specified text displayed.

- JTextArea(String text, int rows, int columns)
  - Constructs a new JTextArea with the specified text and number of rows and columns.

**Some Useful Methods**

- public void append(String str)
  - Appends the given text to the end of the document.

- public void insert(String str, int pos)
  - Inserts the specified text at the specified position.

- public void replaceRange(String str, int start, int end)
  - Replaces text from the indicated start to end position with the new text specified.

- public void setFont(Font f)
  - Sets the current font.

- public void setLineWrap(boolean wrap)
  - Sets the line-wrapping policy of the TextArea.
public void setWrapStyleWord(boolean word)
    Sets the style of wrapping used if the text area is wrapping lines.

**JTextComponent Class**

**Constructor Summary**

None

**Some Useful Methods**

- **public void copy()**
  Transfers the currently selected range in the associated text model to the system clipboard, leaving the contents in the text model.

- **public void cut()**
  Transfers the currently selected range in the associated text model to the system clipboard, removing the contents from the model.

- **public String getSelectedText()**
  Returns the selected text contained in this TextComponent.

- **public String getText()**
  Returns the text contained in this TextComponent.

- **public boolean isEditable()**
  Returns the boolean indicating whether this TextComponent is editable or not.

- **public void selectAll()**
  Selects all the text in the TextComponent.

- **public void select(int selectionStart, int selectionEnd)**
  Selects the text between the specified start and end positions.

- **public void setEditable(boolean b)**
  Sets the state to indicate whether or not the component is editable.

- **public void setText(String t)**
  Sets the text of this TextComponent to the specified text. If the text is null or empty, has the effect of simply deleting the old text.

**JTextField Class**

**Constructor Summary**

**JTextField()**
Constructs a new JTextField.

`JTextField(int columns)`
Constructs a new empty JTextField with the specified number of columns.

`JTextField(String text)`
Constructs a new JTextField initialized with the specified text.

`JTextField(String text, int columns)`
Constructs a new JTextField initialized with the specified text and columns.

**Some Useful Methods**

`public void addActionListener(ActionListener l)`
Adds the specified action listener to receive action events from this textfield.

`public void removeActionListener(ActionListener l)`
Removes the specified action listener so that longer action events are no longer received from textfield.

`public voidsetFont(Font f)`
Sets the current font.

`public void setHorizontalAlignment(int alignment)`
Sets the horizontal alignment of the text.

---

**MouseEvent Class**

**Constructor Summary**

- none generated from mouse enter/exit/press/release/click

**Some Useful Methods**

`public void addMouseListener(MouseListener l)`
Adds the specified mouse listener to receive mouse events from the specified component.

`public void getSource()`
Returns a reference to the object that generated the event.

`public void mousePressed(MouseEvent e)`
Mouse button is pressed.

`public void mouseReleased(MouseEvent e)`
Mouse button is released.

`public void mouseEntered(MouseEvent e)`
Mouse has entered a component.

`public void mouseExited(MouseEvent e)`
Mouse has exited a component.

`public void mouseClicked(MouseEvent e)`
String Class

Constructor Summary

String(text)
    Creates a new String object based on the given text.

    - String s = text;
      Shortcut way to create a new String object.

Some Useful Methods

public char charAt(index)
    Returns the character at the specified index position in the string.

public int compareTo(String anotherString)
    Compares two strings lexicographically.

public String concat(String str)
    Concatenates the specified string to the end of this string.

public boolean contains(String)
    Returns true if the string contains the parameter string.

public void equals(String)
    Returns true if the two string are identical (case sensitive).

public int indexOf(charOrString)
    Returns the index of where the character or String sequence first appears.

public String replace(char oldChar, char newChar)
    Returns a new string resulting from replacing all occurrences of oldChar in this string
    with newChar.

public String substring(int beginIndex)
    Returns a new string that begins with the character at the specified index and
    extends to the end of this string.

public String substring(int beginIndex, int endIndex)
    The substring begins at the specified beginIndex and extends to the character at
    index endIndex - 1

public String toLowerCase()
    Converts all of the characters in this String to lower case.

public String toUpperCase()
    Converts all of the characters in this String to upper case.

public String trim()
    Returns a copy of the string, with leading and trailing whitespace omitted.

public String valueOf(number)
Returns a String object of the number.