Components and Widgets

Object-Oriented Programming

Components

- Components: abstract class
  - car
    - 2003 Ford Escort;
    - 2003 Buick Century;
    - 2003 Dodge Caravan;

- Component
  - Button
  - Checkbox
  - Choice
  - Label
  - List
  - TextComponent
    - TextArea
    - TextField
Some Methods of Component

- **Dimension getSize()**
  - Returns the size of this component in the form of a Dimension object.

- **void print(Graphics g)**
  - Prints this component.

- **void repaint()**
  - Repaints this component.

- **void repaint(int x, int y, int width, int height)**
  - Repaints the specified rectangle of this component.

- **void setBackground(Color c)**
  - Sets the background color of this component.

- **void setBounds(int x, int y, int width, int height)**
  - Moves and resizes this component.

- **void setBounds(Rectangle r)**
  - Moves and resizes this component to conform to the new bounding rectangle r.

More Methods for Component

- **void setFont(Font f)**
  - Sets the font of this component.

- **void setForeground(Color c)**
  - Sets the foreground color of this component.

- **void setLocation(int x, int y)**
  - Moves this component to a new location.

- **void setLocation(Point p)**
  - Moves this component to a new location.

- **void setSize(int width, int height)**
  - Resizes this component so that it has width width and height height.

- **void setSize(Dimension d)**
  - Resizes this component so that it has width d.width and height d.height.

- **void update(Graphics g)**
  - Updates this component.
Textual widgets - Label

- Label:
  - a Component that displays a line of text on the screen, can not be changed by user, but can be modified by the program
- Label()
  Constructs an empty label.
- Label(String text)
  Constructs a new label with the specified string of text, left justified.
- Label(String text, int alignment)
  Constructs a new label that presents the specified string of text with the specified alignment.
- String getText()
  - Gets the text of this label.
- void setText(String text)
  - Sets the text for this label to the specified text.

TextComponent

- TextComponent includes methods that are inherited by TextArea and TextField.
- int getCursorPosition()
  Gets the position of the text insertion caret for this text component.
- String getSelectedText()
  Returns the selected text from the text that is presented by this text component.
- int getSelectionEnd()
  Gets the end position of the selected text in this text component.
- int getSelectionStart()
  Gets the start position of the selected text in this text component.
- String getText()
  Returns the text that is presented by this text component.
More about TextComponent

- void **select**(int selectionStart, int selectionEnd)
  Selects the text between the specified start and end positions.
- void **selectAll**()
  Selects all the text in this text component.
- void **setCaretPosition**(int position)
  Sets the position of the text insertion caret for this text component.
- void **setSelectionEnd**(int selectionEnd)
  Sets the selection end for this text component to the specified position.
- Void **setSelectionStart**(int selectionStart)
  Sets the selection start for this text component to the specified position.
- void **setText**(String t)
  Sets the text that is presented by this text component to be the specified text.

TextField

- **TextField**: a box which holds a single line of text, the text can be modified either by the user or the program

  - **TextField()**
    Constructs a new text field.
  - **TextField**(int columns)
    Constructs a new empty text field with the specified number of columns.
  - **TextField**(String text)
    Constructs a new text field initialized with the specified text.
  - **TextField**(String text, int columns)
    Constructs a new text field initialized with the specified text to be displayed, and wide enough to hold the specified number of columns.
**TextArea**

- **TextArea**: A TextArea object is a multi-line region that displays text. It can be set to allow editing or to be read-only.
  - **TextArea()**
    Constructs a new text area with the empty string as text.
  - **TextArea(int rows, int columns)**
    Constructs a new text area with the specified number of rows and columns and the empty string as text.
  - **TextArea(String text)**
    Constructs a new text area with the specified text.
  - **TextArea(String text, int rows, int columns)**
    Constructs a new text area with the specified text, and with the specified number of rows and columns.

**More about TextArea**

- **TextArea(String text, int rows, int columns, int scrollbars)**
  Constructs a new text area with the specified text, and with the rows, columns, and scroll bar visibility as specified. scrollbars - a constant that determines what scrollbars are created to view the text area
  1) SCROLLBARS_BOTH, 2) SCROLLBARS_VERTICAL_ONLY,
  3) SCROLLBARS_HORIZONTAL_ONLY, 4) SCROLLBARS_NONE.
- **void append(String str)**
  Appends the given text to the text area’s current text.
- **int getColumns()**
  Returns the number of columns in this text area.
- **int getRows()**
  Returns the number of rows in the text area.
- **void insert(String str, int pos)**
  Inserts the specified text at the specified position in this text area.
- **void replaceRange(String str, int start, int end)**
  Replaces text between the indicated start and end positions with the specified replacement text.
Active Widgets - Button

- This class creates a (labeled) button. The application can cause some action to happen when the button is pushed.
- **Button()**
  Constructs a Button with no label.
- **Button(String label)**
  Constructs a Button with the specified label.
- **String getLabel()**
  Gets the label of this button.
- **voidsetLabel(String label)**
  Sets the button’s label to be the specified string.

Checkbox

- A check box is a graphical component that can be in either an "on" (true) or "off" (false) state. Clicking on a check box changes its state from "on" to "off," or from "off" to "on."
- **Checkbox()**
  Creates a check box with no label.
- **Checkbox(String label)**
  Creates a check box with the specified label.
- **Checkbox(String label, boolean state)**
  Creates a check box with the specified label and sets the specified state.
- **Checkbox(String label, boolean state, CheckboxGroup group)**
  Constructs a Checkbox with the specified label, set to the specified state, and in the specified check box group.
More about Checkbox

- **CheckboxGroup** `getCheckboxGroup()`
  Determines this check box's group.

- **String** `getLabel()`
  Gets the label of this check box.

- **boolean** `getState()`
  Determines whether this check box is in the "on" or "off" state.

- **void** `setCheckboxGroup(CheckboxGroup g)`
  Sets this check box’s group to be the specified check box group.

- **void** `setLabel(String label)`
  Sets this check box’s label to be the string argument.

- **Void** `setState(boolean state)`
  Sets the state of this check box to the specified state.

CheckboxGroup

- The CheckboxGroup class is used to group together a set of Checkbox buttons.

- Exactly one check box button in a CheckboxGroup can be in the "on" state at any given time. Pushing any button sets its state to "on" and forces any other button that is in the "on" state into the "off" state.

- **Checkbox** `getSelectedCheckbox()`
  Gets the current choice from this check box group.

- **void** `setSelectedCheckbox(Checkbox box)`
  Sets the currently selected check box in this group to be the specified check box.
Choice

- The Choice class presents a pop-up menu of choices. The current choice is displayed as the title of the menu.
- String getSelectedItem()
  Gets a representation of the current choice as a string.
- void insert(String item, int index)
  Inserts the item into this choice at the specified position.
- void remove(int position)
  Removes an item from the choice menu at the specified position.
- void removeAll()
  Removes all items from the choice menu.
- void select(int pos)
  Sets the selected item in this Choice menu to be the item at the specified position.
- Void select(String str)
  Sets the selected item in this Choice menu to be the item whose name is equal to the specified string.

List

- The List component presents the user with a scrolling list of text items. The list can be set up so that the user can choose either one item or multiple items.
- List()
  Creates a new scrolling list. By default, there are four visible lines and multiple selections are not allowed.
- List(int rows)
  Creates a new scrolling list initialized with the specified number of visible lines. By default, multiple selections are not allowed.
- List(int rows, boolean multipleMode)
  Creates a new scrolling list initialized to display the specified number of rows.
- void add(String item)
  Adds the specified item to the end of scrolling list.
- void add(String item, int index)
  Adds the specified item to the the scrolling list at the position indicated by the index.
More about List

- **void deselect**(int index)
  Deselects the item at the specified index.

- **int getItemCount()**
  Gets the number of items in the list.

- **int getRows()**
  Gets the number of visible lines in this list.

- **int getSelectedIndex()**
  Gets the index of the selected item on the list.

- **boolean isIndexSelected**(int index)
  Determines if the specified item in this scrolling list is selected.

- **void remove**(int position)
  Remove the item at the specified position from this scrolling list.

More about List

- **void remove**(String item)
  Removes the first occurrence of an item from the list.

- **void removeAll()**
  Removes all items from this list.

- **void replaceItem**(String newValue, int index)
  Replaces the item at the specified index in the scrolling list with the new string.

- **void select**(int index)
  Selects the item at the specified index in the scrolling list.

- **void setMultipleMode**(boolean b)
  Sets the flag that determines whether this list allows multiple selections.