GUI Components

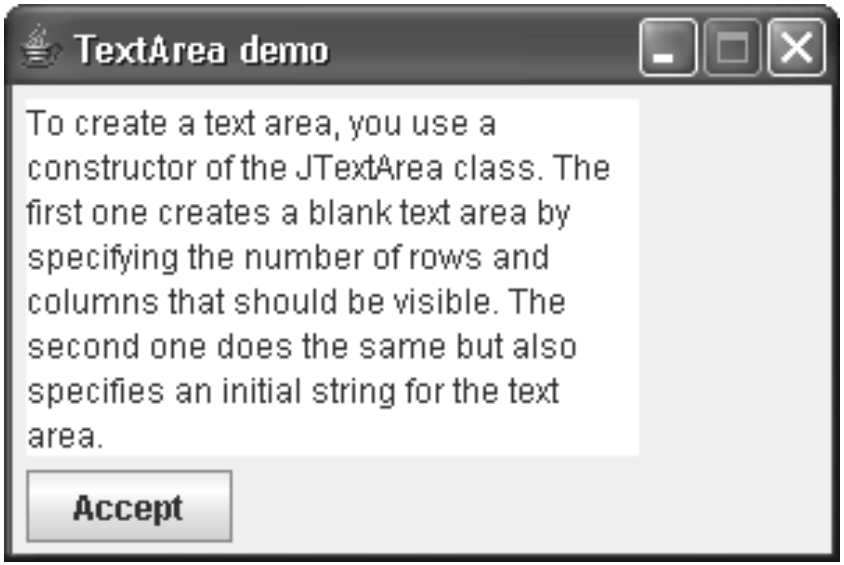
Swing controls presented in this chapter

Control	Class	Description
Text area	JTextArea	Lets the user enter more than one line of text.
Check box	JCheckBox	Lets the user select or deselect an option.
Radio button	JRadioButton	Lets the user select an option from a group of options.
List	JList	Lets the user select one or more items from a list of items.
Combo box	JComboBox	Lets the user select a single item from a drop-down list of items. A combo box can also let the user enter text into the text portion of the combo box.

Components that enhance controls

Component	Class	Description
Border	JBorder	Can be used to visually group components or to enhance the appearance of an individual component.
Scroll pane	JScrollPane	Contains scroll bars that can be used to scroll through the contents of other controls. Scroll panes are typically used with text area and list controls.

A frame with a text area



Common constructors of the JTextArea class

Constructor	Description
`JTextArea(intRows, intCols)`	Creates an empty text area with the specified number of rows and columns.
`JTextArea(String, intRows, intCols)`	Creates a text area with the specified number of rows and columns starting with the specified text.

Some methods that work with text areas

Method	Description
`setLineWrap(boolean)`	If the boolean value is true, the lines will wrap if they don't fit.
`setWrapStyleWord(boolean)`	If the boolean value is true and line wrapping is turned on, wrapped lines will be separated between words.
`append(String)`	Appends the specified string to the text in the text area.
`getText()`	Returns the text in the text area as a String.
`setText(String)`	Sets the text in the text area to the specified string.

Code that creates a text area

```
private JTextArea commentTextArea;
commentTextArea = new JTextArea(7, 20);
commentTextArea.setLineWrap(true);
commentTextArea.setWrapStyleWord(true);
add(commentTextArea);
```

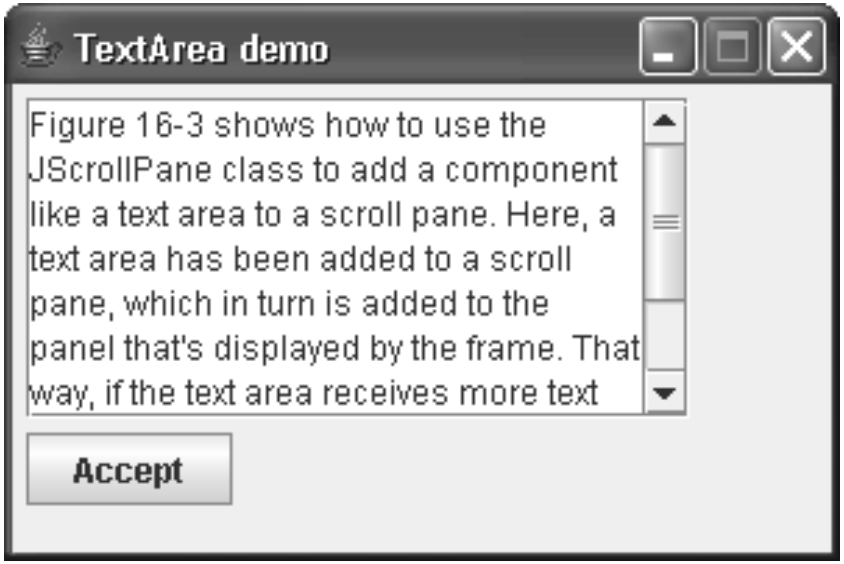
Code that gets the text stored in the text area

```
String comments = commentTextArea.getText();
```

Note

- If the text area is going to receive more text than can be viewed at one time, you should add the text area to a scroll pane.

A frame that displays a text area in a scroll pane



Common constructors of the JScrollPane class

Constructor	Description
`JScrollPane(Component)`	Creates a scroll pane that displays the specified component, along with vertical and horizontal scrollbars as needed.
`JScrollPane(Component, vertical, horizontal)`	Creates a scroll pane that displays the specified component and uses the specified vertical and horizontal policies.

Fields of the ScrollPaneConstants interface that set scrollbar policies

Field	Description
`VERTICAL_SCROLLBAR_ALWAYS`	Always display a vertical scrollbar.
`VERTICAL_SCROLLBAR_AS_NEEDED`	Display a vertical scrollbar only when needed.
`VERTICAL_SCROLLBAR_NEVER`	Never display a vertical scrollbar.
`HORIZONTAL_SCROLLBAR_ALWAYS`	Always display a horizontal scrollbar.
`HORIZONTAL_SCROLLBAR_AS_NEEDED`	Display a horizontal scrollbar only when needed.
`HORIZONTAL_SCROLLBAR_NEVER`	Never display a horizontal scrollbar.

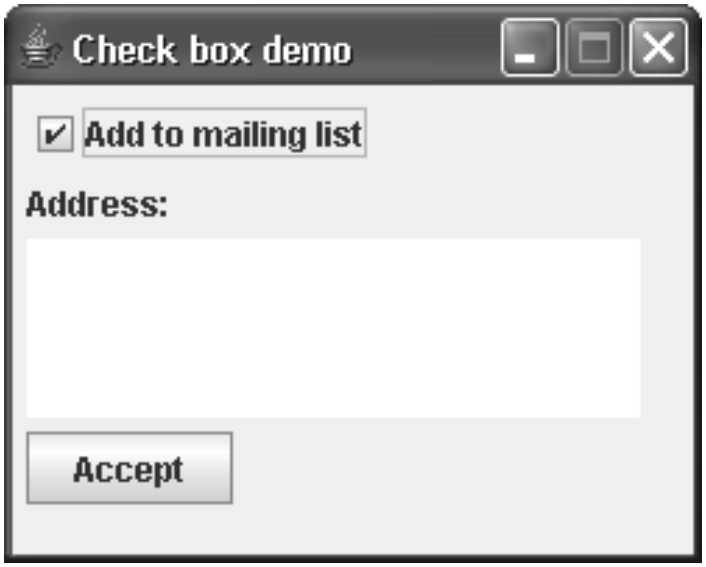
Code that uses a scroll pane with a text area

```
private JTextArea commentTextArea = new JTextArea(7, 20);
commentTextArea.setLineWrap(true);
commentTextArea.setWrapStyleWord(true);
JScrollPane commentScroll = new
JScrollPane(commentTextArea);
add(commentScroll);
```

Code that creates a scroll pane and sets the scroll bar policies

```
JScrollPane commentScroll =
    new JScrollPane(commentTextArea,
        ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
        ScrollPaneConstants.HORIZONTAL_SCROLLBAR_NEVER);
```

A frame with a check box



Common constructors of the JCheckBox class

Constructor	Description
`JCheckBox(String)`	Creates an unselected check box with a label that contains the specified string.
`JCheckBox(String, boolean)`	Creates a check box with a label that contains the specified string. If the boolean value is true, the check box is selected.

Some methods that work with check boxes

Method	Description
`isSelected()`	Returns a true value if the check box is selected.
`setSelected(boolean)`	Checks or unchecks the check box depending on the boolean value.
`addActionListener(ActionListener)`	Adds an action listener to the check box.

Code that creates the check box

```
private JCheckBox mailingCheckBox;
mailingCheckBox =
    new JCheckBox("Add to mailing list", true);
mailingCheckBox.addActionListener(this);
add(mailingCheckBox);
```

Code that checks the status of the check box

```
boolean addToList = mailingCheckBox.isSelected();
```

An actionPerformed method for the check box

```
public void actionPerformed(ActionEvent e)
{
    Object source = e.getSource();
    if (source == mailingCheckBox)
    {
        if (mailingCheckBox.isSelected())
            addressTextArea.setEnabled(true);
        else
            addressTextArea.setEnabled(false);
    }
}
```

A frame with three radio buttons

How to work with radio buttons

- You must add each radio button in a set of options to a ButtonGroup object.
- Selecting a radio button automatically deselects all other radio buttons in the same button group.

Common constructors and methods of the JRadioButton class

Constructor	Description
`JRadioButton(String)`	Creates an unselected radio button with the specified text.
`JRadioButton(String, boolean)`	Creates a radio button with the specified text. If the boolean value is true, the radio button is selected.
Method	**Description**
`isSelected()`	Returns a true value if the radio button is selected.
`addActionListener(ActionListener)`	Adds an action listener to the radio button.

Common constructor and method of the ButtonGroup class

Constructor	Description
`ButtonGroup()`	Creates a button group used to hold a group of buttons.
Method	**Description**
`add(AbstractButton)`	Adds the specified button to the group.

Code that creates three radio buttons and adds them to a panel

```
private JRadioButton uspsRadioButton, upsRadioButton,
    fedexRadioButton;
uspsRadioButton = new JRadioButton("USPS", true);
upsRadioButton = new JRadioButton("UPS");
fedexRadioButton = new JRadioButton("Fedex");
add(uspsRadioButton);
add(upsRadioButton);
add(fedexRadioButton);
ButtonGroup shipViaGroup = new ButtonGroup();
shipViaGroup.add(uspsRadioButton);
shipViaGroup.add(upsRadioButton);
shipViaGroup.add(fedexRadioButton);
```

Code that determines which radio button is selected

```
if (uspsRadioButton.isSelected())
    shipVia = "USPS";
else if (upsRadioButton.isSelected())
    shipVia = "UPS";
else if (fedexRadioButton.isSelected())
    shipVia = "Federal Express";
```

Radio buttons with an etched and titled border

How to work with borders

- To place controls in a border, you must create a panel, create a border and apply it the panel, and add the controls to the panel.
- Because a border only groups controls visually, you must still use a ButtonGroup object to group radio buttons logically.
- To set borders, you must import the javax.swing.border package.

Static methods of the BorderFactory class

Method	Description
`createLineBorder()`	Creates a line border.
`createEtchedBorder()`	Creates an etched border.
`createLoweredBevelBorder()`	Creates a lowered bevel border.
`createRaisedBevelBorder()`	Creates a raised bevel border.
`createTitledBorder(String)`	Creates a line border with the specified title.
`createTitledBorder(Border, String)`	Adds the specified title to the specified border.

Method of the JComponent class used to set borders

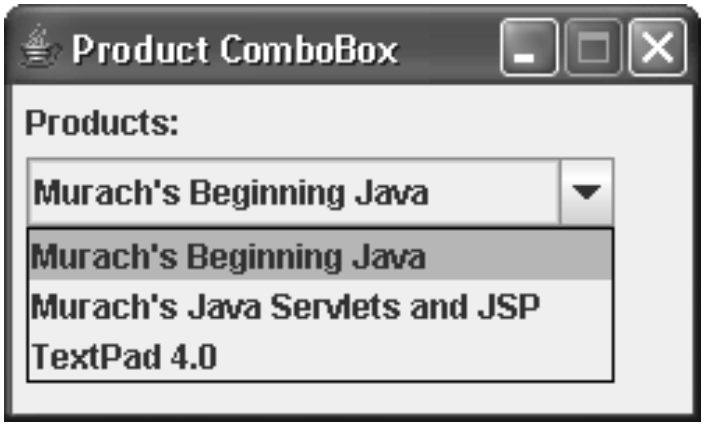
Method	Description
`setBorder(Border)`	Sets the border style for a component.

Code that creates bordered radio buttons

```
uspsRadioButton = new JRadioButton("USPS", true);
upsRadioButton = new JRadioButton("UPS");
fedexRadioButton = new JRadioButton("Fedex");
ButtonGroup shipViaGroup = new ButtonGroup();
shipViaGroup.add(uspsRadioButton);
shipViaGroup.add(upsRadioButton);
shipViaGroup.add(fedexRadioButton);

JPanel shipViaPanel = new JPanel();
Border shipViaBorder =
    BorderFactory.createEtchedBorder();
shipViaBorder =
    BorderFactory.createTitledBorder(shipViaBorder,
    "Carrier");
shipViaPanel.setBorder(shipViaBorder);
shipViaPanel.add(uspsRadioButton);
shipViaPanel.add(upsRadioButton);
shipViaPanel.add(fedexRadioButton);
add(shipViaPanel);
```

A frame with a combo box



Common constructors of the JComboBox class

Constructor	Description
`JComboBox()`	Creates an empty combo box.
`JComboBox(Object[])`	Creates a combo box with the objects stored in the specified array.

Some methods that work with combo boxes

Method	Description
`getSelectedItem()`	Returns an Object type for the selected item.
`getSelectedIndex()`	Returns an int value for the index of the selected item.
`setSelectedIndex(``intIndex`**`)`**	Selects the item at the specified index.
`setEditable(``boolean`**`)`**	If the boolean value is true, the combo box can be edited.
`getItemCount()`	Returns the number of items stored in the combo box.
`addItem(``Object`**`)`**	Adds an item to the combo box.
`removeItemAt(``int`**`)`**	Removes the item at the specified index from the combo box.

Some methods that work with combo boxes (continued)

Method	Description
`removeItem(Object)`	Removes the specified item from the combo box.
`addActionListener(ActionListener)`	Adds an action listener to the combo box.
`addItemListener(ItemListener)`	Adds an item listener to the combo box.

Code that creates the combo box

```
private ArrayList<Product> products;
products = getProducts();
                    // returns an ArrayList of products
productComboBox = new JComboBox();
for (Product p : products)
    productComboBox.addItem(p.getDescription());
add(productComboBox);
```

Code that determines which item was selected

```
int i = productComboBox.getSelectedIndex();
Product p = products.get(i);
```

A frame that uses an action event listener to update the display based on the user's selection

The actionPerformed method of the ActionListener interface

Method	Description
`void actionPerformed(ActionEvent e)`	Invoked when an item is selected.

The itemStateChanged method of the ItemListener interface

Method	Description
`void itemStateChanged(ItemEvent e)`	Invoked when an item is selected or deselected.

Common methods of the ItemEvent class

Method	Description
`getSource()`	Returns the source of the event.
`getItem()`	Returns the selected item.
`getStateChanged()`	Returns an int value that indicates whether an item was selected or deselected. The field names for these values are SELECTED and DESELECTED.

Code that creates a combo box

```
products = getProducts();
productComboBox = new JComboBox();
for (Product p : products)
    productComboBox.addItem(p.getDescription());
productComboBox.setSelectedIndex(0);
productComboBox.addActionListener(this);
add(productComboBox);
```

Code that implements the ActionListener interface for the combo box

```
public void actionPerformed(ActionEvent e)
{
    Object source = e.getSource();
    if (source == productComboBox)
    {
        int i = productComboBox.getSelectedIndex();
        Product p = products.get(i);
        priceTextField.setText(p.getFormattedPrice());
    }
}
```

A frame that includes a list

Common constructors of the JList class

Constructor	Description
`JList(Object[])`	Creates a list that contains the objects stored in the specified array of objects.
`JList(ListModel)`	Creates a list using the specified list model.

Some methods of the JList class

Method	Description
`getSelectedValue()`	Returns the selected item as an Object type.
`getSelectedIndex()`	Returns an int value for the index of the selected item.
`isSelectedIndex(intIndex)`	Returns a true value if the item at the specified index is selected.
`setFixedCellWidth(intPixels)`	Sets the cell width to the specified number of pixels. Otherwise, the width of the list is slightly wider than the widest item in the array that populates the list.
`setVisibleRowCount(intRows)`	Sets the visible row count to the specified int value. This only works when the list is displayed within a scroll pane.

Some methods of the JList class (continued)

Method	Description
`setSelectionMode(mode)`	Sets the selection mode. To allow single selections, specify ListSelectionModel.SINGLE_SELECTION.
`setSelectedIndex(intIndex)`	Selects the item at the specified index.

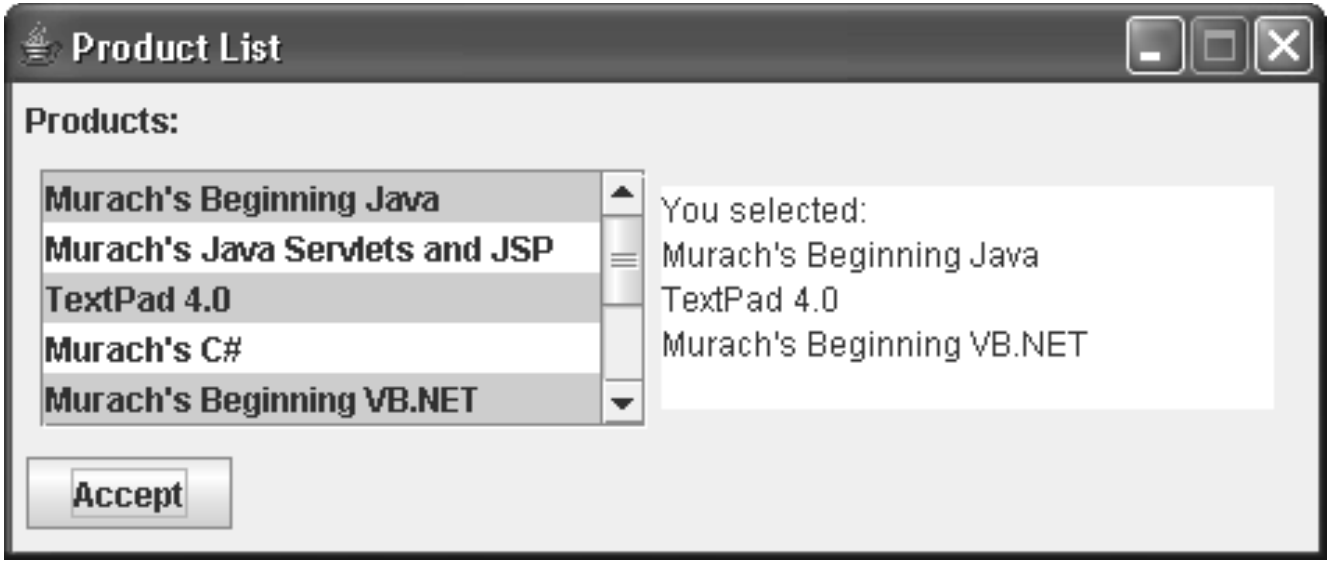
Code that creates a list

```
descriptions = getProductDescriptions();
                    // returns an array of descriptions
productList = new JList(descriptions);
productList.setFixedCellWidth(200);
productList.setVisibleRowCount(5);
productList.setSelectedIndex(0);
productList.setSelectionMode(
    ListSelectionModel.SINGLE_SELECTION);
add(new JScrollPane(productList));
```

Code that gets the selected item

```
String s = (String)productList.getSelectedValue();
```

A list that allows multiple selections



Fields of the ListSelectionModel interface used to set the selection mode

Field	Description
`SINGLE_SELECTION`	Allows just one selection.
`SINGLE_INTERVAL_SELECTION`	Allows a single range of selections.
`MULTIPLE_INTERVAL_SELECTION`	Allows multiple ranges of selections. This is the default.

Methods of the JList class used to process multiple selections

Method	Description
`getSelectedValues()`	Returns an array of Object types for the selected items.
`getSelectedIndices()`	Returns an array of ints corresponding to the indices of the selected items.

Code that creates a list

```
descriptions = getProductDescriptions();
                    // returns an array of descriptions
productList = new JList(descriptions);
productList.setFixedCellWidth(200);
productList.setVisibleRowCount(5);
productList.setSelectedIndex(0);
productList.setSelectionMode(
    ListSelectionModel.MULTIPLE_INTERVAL_SELECTION);
add(new JScrollPane(productList));
```

Code that displays the list selections

```
public void actionPerformed(ActionEvent e)
{
    Object source = e.getSource();
    if (source == acceptButton)
    {
        Object[] selections =
            productList.getSelectedValues();
        String s = "";
        for (Object o : selections)
            s += (String)o + "\n";
        productTextArea.setText("You selected:\n" + s);
    }
}
```

A frame that lets you add elements to a list