

Componentes visuales II

Listas, combos y cuadros de diálogo

Tecnología de la Programación

Curso 2015-2016

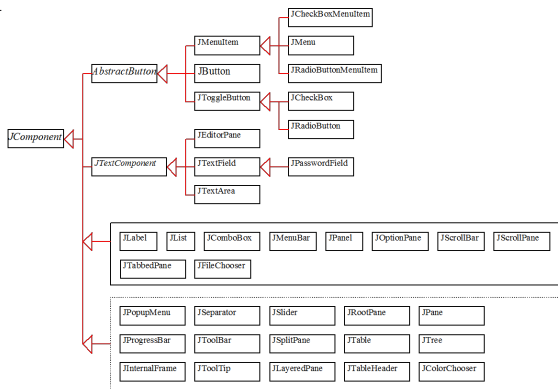
Departamento de Sistemas Informáticos y Computación
Universidad Complutense de Madrid

(Basado en material creado por Yolanda García)



Componentes visuales de Swing

- Excepto los contenedores de primer nivel, todos los demás componentes visuales heredan de `JComponent`.



- Veremos algunos de ellos.

Listas y combo boxes

- Hay dos componentes básicos para representar listas de elementos:
 - ▶ **JList**: muestra una lista de elementos para que el usuario seleccione uno o varios de ellos.
 - ▶ **JComboBox**: muestra una lista desplegable. El usuario puede seleccionar un solo elemento, que es el que aparecerá en la lista sin desplegar.
 - ▶ En un `JComboBox` la lista puede ser **editable**, de forma que se puede utilizar como un `JTextField`.
- Vamos a ver cómo se utilizan estos componentes.

Listas y combo boxes

- En Java 1.7 `JList` y `JComboBox` son clases genéricas. El tipo genérico es el de los elementos que aparecerán en la lista.
- Los listener en cada tipo de lista son distintos:
 - ▶ En el caso de un `JList` se debe añadir un listener de eventos `ListSelectionEvent`.
 - ▶ Para un `JComboBox` se utilizan manejadores de `ActionEvent`.
- El método para obtener el elemento seleccionado también es distinto:
 - ▶ Para `JList` se utiliza `getSelectedValue()`.
 - ▶ Para `JComboBox` es `getSelectedItem()`.
- En un `JList` se pueden seleccionar varios elementos, para obtenerlos se utiliza `getSelectedValuesList()`.
- Además tienen otros métodos para obtener el elemento en una posición determinada, la lista de elementos, etc.

Listas y combo boxes

- Hay dos formas de dar valores a los elementos de una lista o combo box:
 - ▶ Mediante un array de elementos que se pasa a la constructora.
 - ▶ Mediante un **modelo de lista**.
- En el primer caso, **se crea una lista o combo con los elementos de un array.**
- Se puede utilizar para mostrar una lista de elementos en la que se seleccionan uno o varios de ellos.
- Pero los elementos de la lista no se pueden modificar ni añadir.
- Ejemplos: **EjemploListas1.java, EjemploCombo1.java.**

Listas y combo boxes

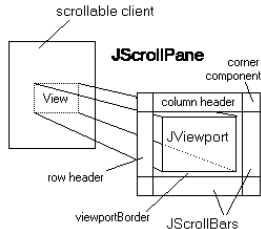
- También se pueden dar valores a las listas (`JList` y `JComboBox`) utilizando un **modelo de lista**.
- Para utilizar un modelo de lista, se debe:
 - ❶ Crear un modelo de lista.
 - ❷ Añadir elementos al modelo.
 - ❸ Asociar el modelo a la lista (`JList` o `JComboBox`).
- Un modelo de lista es **una clase que mantiene la información de los elementos** que aparecen en un componente `JList` o `JComboBox` y **notifica a la lista los cambios que se produzcan en el modelo**.
- Se pueden crear modelos de lista extendiendo la clase `AbstractListModel`.

Listas y combo boxes

- En el API de Swing existen **modelos por defecto** para ambos tipos de lista: `DefaultListModel` y `DefaultComboBoxModel`.
- Ambos modelos implementan una estructura de datos lista, similar a las de `java.util`, por lo que se permiten las operaciones habituales:
 - ▶ `addElement(E)` añade un elemento al modelo de lista.
 - ▶ `removeElement(Object)` elimina un elemento del modelo de lista.
 - ▶ `elementAt(int)` obtiene el elemento en una posición de la lista.
 - ▶ `contains(Object)` devuelve true si el elemento ya está en la lista.
 - ▶ etc.
- Además, incluyen los métodos necesarios para notificar a la lista asociada que se han producido cambios en el modelo.
- Ejemplo: [EjemploLista2.java](#), [EjemploListaYCombo3.java](#).

Barras de desplazamiento: JScrollPane

- El componente `JList` es de bajo nivel:
 - ▶ Se puede indicar el número de elementos visibles con el método `setVisibleRowCount(int)`.
 - ▶ Si los elementos de una lista no caben en el espacio que tienen en la ventana, simplemente no se muestran.
- Por defecto no aparecen **barras de desplazamiento** (*scroll*).
- Para mostrar barras de desplazamiento, se debe utilizar el componente **JScrollPane**.
- Este componente **muestra una vista parcial de un componente visual** que no puede verse en su totalidad.



(Origen de la imagen: <http://docs.oracle.com/javase/7/docs/api/javax/swing/JScrollPane.html>)

Barras de desplazamiento: JScrollPane

- Para añadir barras de desplazamiento a una lista, se debe crear un objeto `JScrollPane`, asociar la lista a este objeto y añadirlo al contenedor que corresponda:

```
lstLista = new JList<String>(modelo);  
JScrollPane sclLista = new JScrollPane(lstLista);  
...  
JPanel pnlCentro = new JPanel();  
pnlCentro.setLayout(new BorderLayout());  
pnlCentro.add(sclLista, BorderLayout.CENTER);
```

- Se puede fijar cuándo se muestran las barras de desplazamiento:

```
setHorizontalScrollBarPolicy(int policy)  
setVerticalScrollBarPolicy(int policy)
```

- donde la política puede ser (de forma análoga para las verticales):

```
HORIZONTAL_SCROLLBAR_AS_NEEDED  
HORIZONTAL_SCROLLBAR_NEVER  
HORIZONTAL_SCROLLBAR_ALWAYS
```

- Ejemplo: `EjemploListaScroll.java`.

Cuadros de diálogo sencillos

- **Cuadros de diálogo:** Ventanas prefijadas con las que se tiene más control sobre las acciones del usuario.
- Estas ventanas suelen **bloquear el resto de la aplicación hasta que la ventana se cierra**. Cuando una ventana tiene este comportamiento se denomina **modal**.
- Heredan de **JDialog**.
- La clase **JOptionPane** contiene métodos estáticos para mostrar cuadros de diálogo sencillos:
- Método **showMessageDialog**:
 - ▶ Muestra una ventana informativa con un texto y un botón para cerrar la ventana.
 - ▶ Además, se puede mostrar un icono en la parte izquierda.
 - ▶ El icono puede ser un icono prefijado o un `ImageIcon`.
 - ▶ Ejemplo: **EjemploJOptionPane_MessageDialog.java**

Cuadros de diálogo sencillos

- Método `showConfirmDialog`:
 - ▶ Similar al anterior, pero permite mostrar varios botones prefijados para obtener una respuesta básica del usuario (sí/no, sí/no/cancelar, aceptar/cancelar).
 - ▶ También se puede mostrar un icono en la parte izquierda, que puede ser uno de los iconos prefijados.
 - ▶ Como resultado de la llamada al método se obtiene el botón pulsado (un valor `int`).
 - ▶ Ejemplo: `EjemploJOptionPane_ConfirmDialog.java`
- Método `showOptionDialog`:
 - ▶ Similar al anterior, pero permite mostrar varios botones definidos por el programador.
 - ▶ También se puede mostrar un icono en la parte izquierda.
 - ▶ Como resultado de la llamada al método se obtiene el botón pulsado (un valor `int`).
 - ▶ Ejemplo: `EjemploJOptionPane_OptionDialog.java`
- Más información: <http://java.sun.com/docs/books/tutorial/uiswing/components/dialog.html>

Cuadros de diálogo sencillos

- Método `showInputDialog`:

- ▶ Este método muestra una ventana que permite introducir un dato al usuario (con un `JTextField`).
- ▶ También se puede mostrar un icono en la parte izquierda.
- ▶ Como resultado de la llamada al método se obtiene el texto introducido (un `String`).
- ▶ Ejemplo: `EjemploJOptionPane_InputDialog.java`
- ▶ Se puede configurar para que se muestre una lista de opciones, entre otras cosas.

- Más información: <http://java.sun.com/docs/books/tutorial/uiswing/components/dialog.html>