36 - Swing - JComboBox

Listado completo de tutoriales

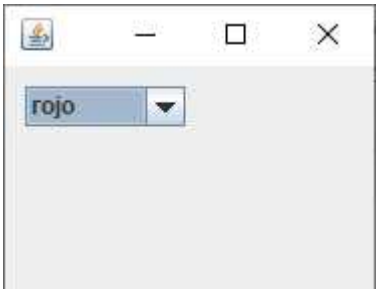
El control JComboBox permite seleccionar un String de una lista.

Para inicializar los String que contendrá el JComboBox debemos llamar al método addItem tantas veces como elementos queremos cargar.

Un evento muy útil con este control es cuando el operador selecciona un Item de la lista. Para capturar la selección de un item debemos implementar la interface ItemListener que contiene un método llamada itemStateChanged.

Problema 1:

Cargar en un JComboBox los nombres de varios colores. Al seleccionar alguno mostrar en la barra de título del JFrame el String seleccionado.



Programa:

Ver video

```
import javax.swing.*;
import java.awt.event.*;
public class Formulario extends JFrame implements Ite
    private JComboBox<String> combo1;
    public Formulario() {
        setLayout(null);
        combo1=new JComboBox<String>();
        combo1.setBounds(10,10,80,20);
        add(combo1);
        combo1.addItem("rojo");
        combo1.addItem("vede");
        combo1.addItem("azul");
        combo1.addItem("amarillo");
        combo1.addItem("negro");
        combo1.addItemListener(this);
    }
```

```
    public void itemStateChanged(ItemEvent e) {
        if (e.getSource()==combo1) {
            String seleccionado=(String)combo1.getSel
            setTitle(seleccionado);
        }
    }

    public static void main(String[] ar) {
        Formulario formulario1=new Formulario();
        formulario1.setBounds(0,0,200,150);
        formulario1.setVisible(true);
        formulario1.setDefaultCloseOperation(JFrame.E
    }
}
```

Indicamos a la clase que implementaremos la interface ItemListener:

```
public class Formulario extends JFrame implements ItemListener{
```

Declaramos un objeto de la clase JComboBox:

```
    private JComboBox<String> combo1;
```

Debemos anteceder el tipo de datos que administra el JComboBox <String>, luego veremos el concepto de genéricos para comprender completamente el porque anteceder entre los símbolos menor y mayor el tipo de dato String.

En el constructor creamos el objeto de la clase JComboBox:

```
        combo1=new JComboBox<String>();
```

Posicionamos el control:

```
        combo1.setBounds(10,10,80,20);
```

Añadimos el control al JFrame:

```
        add(combo1);
```

Añadimos los String al JComboBox:

```
        combo1.addItem("rojo");
        combo1.addItem("vede");
        combo1.addItem("azul");
        combo1.addItem("amarillo");
        combo1.addItem("negro");
```

Asociamos la clase que capturará el evento de cambio de item (con this indicamos que esta misma clase capturará el evento):

```
        combo1.addItemListener(this);
```

El método itemStateChanged que debemos implementar de la interface ItemListener tiene la siguiente sintaxis:

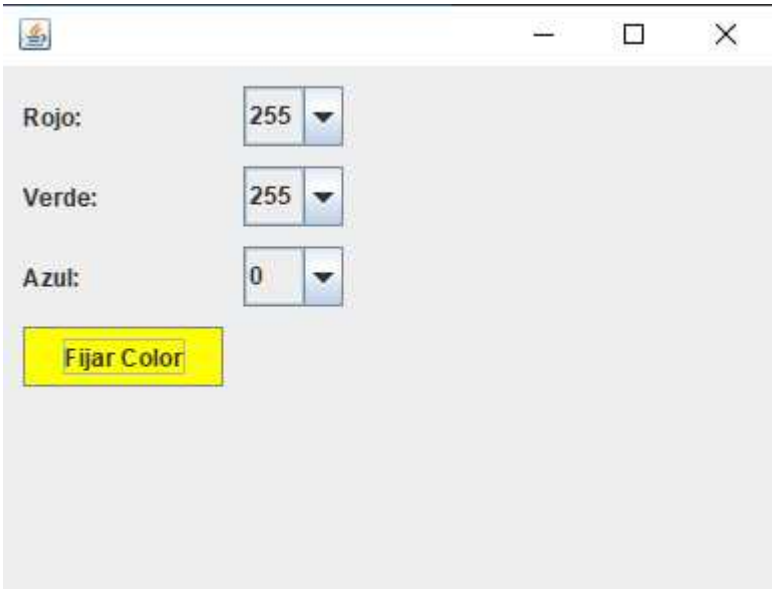
```
    public void itemStateChanged(ItemEvent e) {
        if (e.getSource()==combo1) {
            String seleccionado=(String)combo1.getSelectedItem();
            setTitle(seleccionado);
        }
    }
```

Para extraer el contenido del item seleccionado llamamos al método getSelectemItem() el cual retorna un objeto de la clase Object por lo que debemos indicarle que lo transforme en String:

```
            String seleccionado=(String)combo1.getSelectedItem();
```

Problema 2:

Disponer tres controles de tipo JComboBox con valores entre 0 y 255 (cada uno representa la cantidad de rojo, verde y azul). Luego al presionar un botón pintar el mismo con el color que se genera combinando los valores de los JComboBox.



Programa:

Ver video

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class Formulario extends JFrame implements Act
    private JLabel label1,label2,label3;
    private JComboBox<String> combo1,combo2,combo3;
    private JButton boton1;
    public Formulario() {
        setLayout(null);
        label1=new JLabel("Rojo:");
        label1.setBounds(10,10,100,30);
        add(label1);
        combo1=new JComboBox<String>();
        combo1.setBounds(120,10,50,30);
        for(int f=0;f<=255;f++) {
            combo1.addItem(String.valueOf(f));
        }
        add(combo1);
        label2=new JLabel("Verde:");
        label2.setBounds(10,50,100,30);
        add(label2);
        combo2=new JComboBox<String>();
        combo2.setBounds(120,50,50,30);
        for(int f=0;f<=255;f++) {
            combo2.addItem(String.valueOf(f));
        }
        add(combo2);
        label3=new JLabel("Azul:");
```

```
        label3.setBounds(10,90,100,30);
        add(label3);
        combo3=new JComboBox<String>();
        combo3.setBounds(120,90,50,30);
        for(int f=0;f<=255;f++) {
            combo3.addItem(String.valueOf(f));
        }
        add(combo3);
        boton1=new JButton("Fijar Color");
        boton1.setBounds(10,130,100,30);
        add(boton1);
        boton1.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource()==boton1) {
            String cad1=(String)combo1.getSelectedIte
            String cad2=(String)combo2.getSelectedIte
            String cad3=(String)combo3.getSelectedIte
            int rojo=Integer.parseInt(cad1);
            int verde=Integer.parseInt(cad2);
            int azul=Integer.parseInt(cad3);
            Color color1=new Color(rojo,verde,azul);
            boton1.setBackground(color1);
        }
    }

    public static void main(String[] ar) {
        Formulario formulario1=new Formulario();
        formulario1.setBounds(0,0,400,300);
        formulario1.setVisible(true);
        formulario1.setDefaultCloseOperation(JFrame.E
    }
}
```

Importamos el paquete java.awt ya que el mismo contiene la clase Color:

```
import java.awt.*;
```

Implementaremos la interface ActionListener ya que tenemos que cambiar el color del botón cuando se lo presione y no haremos actividades cuando cambiemos items de los controles JComboBox:

```
public class Formulario extends JFrame implements ActionListener{
```

Definimos los siete objetos requeridos en esta aplicación:

```
private JLabel label1,label2,label3;
private JComboBox<String> combo1,combo2,combo3;
private JButton boton1;
```

En el constructor creamos los objetos, primero el control label1 de la clase JLabel:

```
label1=new JLabel("Rojo:");
label1.setBounds(10,10,100,30);
add(label1);
```

Lo mismo hacemos con el objeto combo1:

```
combo1=new JComboBox<String>();
combo1.setBounds(120,10,50,30);
```

Para añadir los 256 elementos del JComboBox disponemos un for y previa a llamar al método addItem convertimos el entero a String:

```
for(int f=0;f<=255;f++) {
    combo1.addItem(String.valueOf(f));
}
add(combo1);
```

En el método actionPerformed cuando detectamos que se presionó el botón procedemos a extraer los tres item seleccionados:

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource()==boton1) {
        String cad1=(String)combo1.getSelectedItem();
        String cad2=(String)combo2.getSelectedItem();
        String cad3=(String)combo3.getSelectedItem();
```

Los convertimos a entero:

```
        int rojo=Integer.parseInt(cad1);
        int verde=Integer.parseInt(cad2);
        int azul=Integer.parseInt(cad3);
```

y creamos finalmente un objeto de la clase Color, el constructor de la clase Color requiere que le pasemos tres valores de tipo int:

```
        Color color1=new Color(rojo,verde,azul);
```

Para cambiar el color de fondo del control JButton debemos llamar al método setBackground y pasarle el objeto de la clase Color:

```
boton1.setBackground(color1);
```

Problema propuesto

1. Solicitar el ingreso del nombre de una persona y seleccionar de un control JComboBox un país. Al presionar un botón mostrar en la barra del título del JFrame el nombre ingresado y el país seleccionado.
   Ver video

Solución

Retornar