Curso de Java a distancia

Clase 12: Swing - JTextField

Así como podríamos decir que el control JLabel remplaza a la salida estándar System.out.print, el control JTextField cumple la función de la clase Scanner para la entrada de datos.

El control JTextField permite al operador del programa ingresar una cadena de caracteres por teclado.

PROBLEMA 1:

Confeccionar un programa que permita ingresar el nombre de usuario y cuando se presione un botón mostrar el valor ingresado en la barra de títulos del JFrame.

PROGRAMA:

```
import javax.swing.*;

import java.awt.event.*;

public class Formulario extends JFrame implements ActionListener{

   private JTextField textfield1;
```

```
private JLabel label1;
private JButton boton1;
public Formulario() {
   setLayout(null);
   label1=new JLabel("Usuario:");
   label1.setBounds(10,10,100,30);
   add(label1);
   textfield1=new JTextField();
   textfield1.setBounds(120,10,150,20);
   add(textfield1);
   boton1=new JButton("Aceptar");
   boton1.setBounds(10,80,100,30);
   add(boton1);
   boton1.addActionListener(this);
}

public void actionPerformed(ActionEvent e) {
   if (e.getSource()==boton1) {
      String cad=textfield1.getText();
      setTitle(cad);
   }
}
```

```
    public static void main(String[] ar) {
        Formulario formulario1=new Formulario();
        formulario1.setBounds(0,0,350,170);
        formulario1.setVisible(true);
    }
}
```

Definimos los tres objetos que colaboran con nuestra aplicación:

```
public class Formulario extends JFrame implements ActionListener{
    private JTextField textfield1;
    private JLabel label1;
    private JButton boton1;
```

En el constructor creamos los tres objetos y los ubicamos:

```
    public Formulario() {
        setLayout(null);
        label1=new JLabel("Usuario:");
        label1.setBounds(10,10,100,30);
        add(label1);
        textfield1=new JTextField();
        textfield1.setBounds(120,10,150,20);
        add(textfield1);
        boton1=new JButton("Aceptar");
```

```
        boton1.setBounds(10,80,100,30);
        add(boton1);
        boton1.addActionListener(this);
    }
```

En el método actionPerformed se verifica si se presionó el objeto JButton, en caso afirmativo extraemos el contenido del control JTextField mediante el método getText:

```
    public void actionPerformed(ActionEvent e) {
        if (e.getSource()==boton1) {
            String cad=textfield1.getText();
            setTitle(cad);
        }
    }
```

En la variable auxiliar cad almacenamos temporalmente el contenido del JTextField y seguidamente actualizamos el título del control JFrame.

Problema 2:

Confeccionar un programa que permita ingresar dos números en controles de tipo JTextField, luego sumar los dos valores ingresados y mostrar la suma en la barra del título del control JFrame.

Programa:

```
import javax.swing.*;
```

```
import java.awt.event.*;
public class Formulario extends JFrame implements ActionListener{
   private JTextField textfield1,textfield2;
   private JButton boton1;
   public Formulario() {
      setLayout(null);
      textfield1=new JTextField();
      textfield1.setBounds(10,10,100,30);
      add(textfield1);
      textfield2=new JTextField();
      textfield2.setBounds(10,50,100,30);
      add(textfield2);
      boton1=new JButton("Sumar");
      boton1.setBounds(10,90,100,30);
      add(boton1);
      boton1.addActionListener(this);
   }

   public void actionPerformed(ActionEvent e) {
      if (e.getSource()==boton1) {
         String cad1=textfield1.getText();
         String cad2=textfield2.getText();
         int x1=Integer.parseInt(cad1);
```

```
            int x2=Integer.parseInt(cad2);
            int suma=x1+x2;
            String total=String.valueOf(suma);
            setTitle(total);
        }
    }

    public static void main(String[] ar) {
        Formulario formulario1=new Formulario();
        formulario1.setBounds(0,0,240,170);
        formulario1.setVisible(true);
    }
}
```

Definimos los tres objetos:

```
public class Formulario extends JFrame implements ActionListener{
    private JTextField textfield1,textfield2;
    private JButton boton1;
```

En el método actionPerformed es donde debemos sumar los valores ingresados en los controles de tipo JTextField. Para extraer el contenido de los controles debemos extraerlos con el método getText:

```
            String cad1=textfield1.getText();
            String cad2=textfield2.getText();
```

Como debemos sumar numéricamente los valores ingresados debemos convertir el contenido de las variables cad2 y cad2 a tipo de dato int:

```
int x1=Integer.parseInt(cad1);

int x2=Integer.parseInt(cad2);
```

El método parseInt de la clase Integer retorna el contenido de cad1 convertido a int (provoca un error si ingresamos caracteres en el control JTextField en lugar de números)

Una vez que tenemos los dos valores en formato numérico procedemos a sumarlos y almacenar el resultado en otra variable auxiliar:

```
int suma=x1+x2;
```

Ahora tenemos que mostrar el valor almacenado en suma en la barra de títulos del control JFrame, pero como el método setTitle requiere un String como parámetro debemos convertirlo a tipo String:

```
String total=String.valueOf(suma);

setTitle(total);
```

Como veremos de acá en adelante es muy común la necesidad de convertir String a enteros y enteros a String:

De String a int:

```
int x1=Integer.parseInt(cad1);
```

De int a String:

```
String total=String.valueOf(suma);
```

Problemas propuestos

1. Ingresar el nombre de usuario y clave en controles de tipo JTextField. Si se ingresa las cadena (usuario: "juan", clave: "abc123") luego mostrar en el título del JFrame el mensaje "Correcto" en caso contrario mostrar el mensaje "Incorrecto".

Solución

```
import javax.swing.*;
import java.awt.event.*;
public class Formulario extends JFrame implements ActionListener {
   private JLabel label1,label2;
   private JTextField textfield1,textfield2;
   private JButton boton1;
   public Formulario() {
      setLayout(null);
      label1=new JLabel("Nombre:");
      label1.setBounds(10,10,100,30);
      add(label1);
      label2=new JLabel("Clave:");
      label2.setBounds(10,50,100,30);
      add(label2);
      textfield1=new JTextField();
      textfield1.setBounds(130,10,100,30);
      add(textfield1);
      textfield2=new JTextField();
      textfield2.setBounds(130,50,100,30);
      add(textfield2);
      boton1=new JButton("Confirmar");
      boton1.setBounds(10,100,100,30);
```

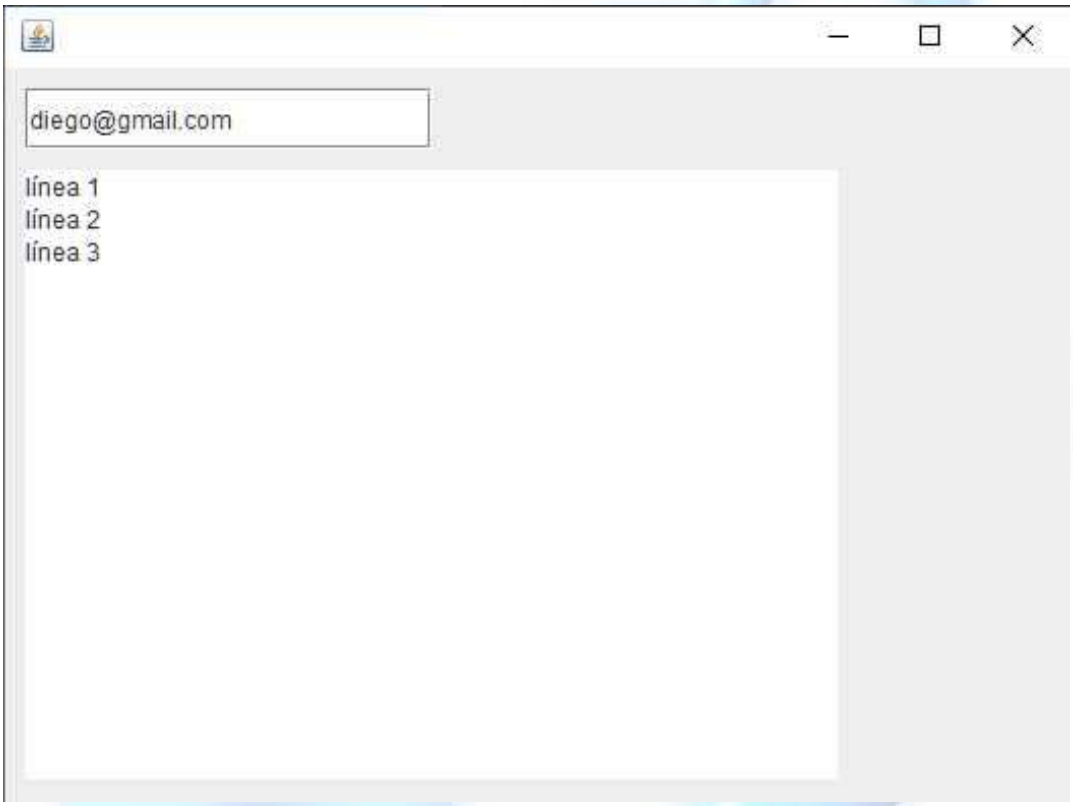
```
        add(boton1);
        boton1.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource()==boton1) {
            String cad1=textfield1.getText();
            String cad2=textfield2.getText();
            if (cad1.equals("juan")==true && cad2.equals("abc123")==true) {
                setTitle("Correcto");
            } else {
                setTitle("Incorrecto");
            }
        }
    }

    public static void main(String[] ar) {
        Formulario formulario1=new Formulario();
        formulario1.setBounds(0,0,260,200);
        formulario1.setVisible(true);
    }
}
```

Swing - JTextArea

El control de tipo JTextArea permite ingresar múltiples líneas, a diferencia del control de tipo JTextField.

PROBLEMA 1:

Confeccionar un programa que permita ingresar un mail en un control de tipo JTextField y el cuerpo del mail en un control de tipo JTextArea.



PROGRAMA:

```
import javax.swing.*;

public class Formulario extends JFrame{

   private JTextField textfield1;

   private JTextArea textarea1;

   public Formulario() {

      setLayout(null);
```

```
        textfield1=new JTextField();
        textfield1.setBounds(10,10,200,30);
        add(textfield1);
        textarea1=new JTextArea();
        textarea1.setBounds(10,50,400,300);
        add(textarea1);
    }

    public static void main(String[] ar) {
        Formulario formulario1=new Formulario();
        formulario1.setBounds(0,0,540,400);
        formulario1.setVisible(true);
    }
}
```

Como vemos crear un control de tipo JTextArea es similar a la creación de controles de tipo JTextField:

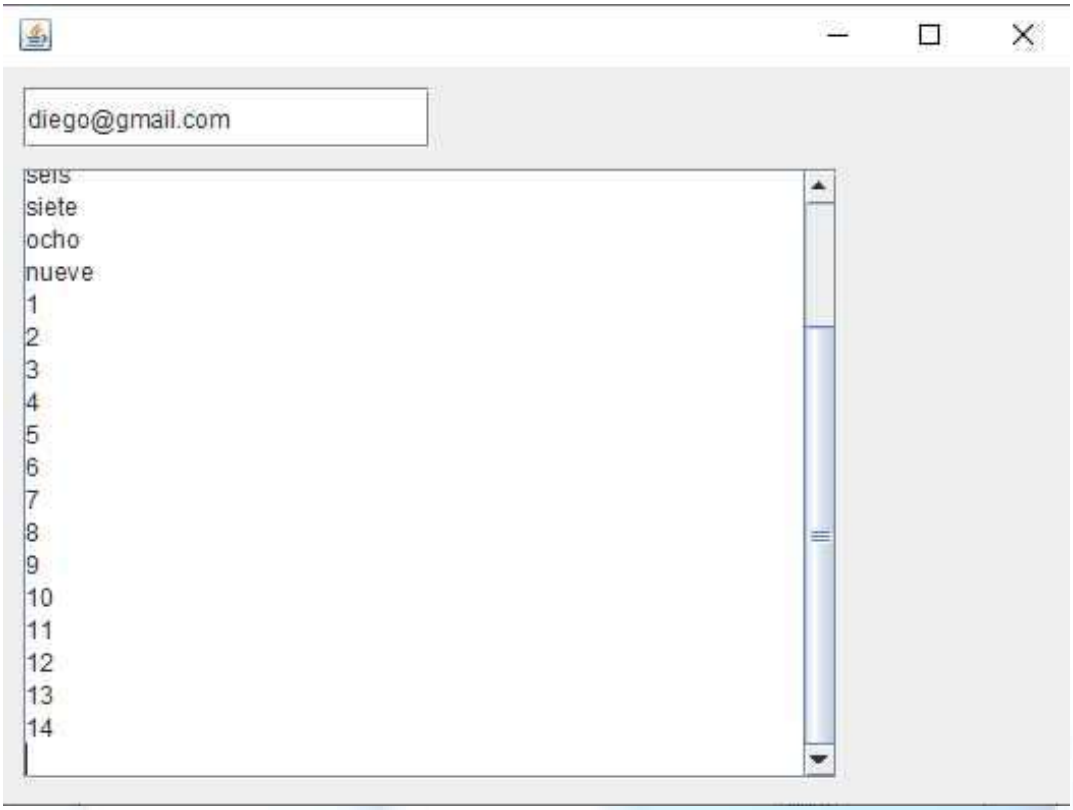
```
        textarea1=new JTextArea();
        textarea1.setBounds(10,50,400,300);
        add(textarea1);
```

El inconveniente que tiene este control es que si ingresamos más texto que el que puede visualizar no aparecen las barras de scroll y no podemos ver los caracteres tipeados.

Para salvar el problema anterior debemos crear un objeto de la clase JScrollPane y añadir en su interior el objeto de la clase JTextArea, luego el programa definitivo es el siguiente:

```
import javax.swing.*;

public class Formulario extends JFrame{
```

```
    private JTextField textfield1;
    private JScrollPane scrollpane1;
    private JTextArea textarea1;
    public Formulario() {
        setLayout(null);
        textfield1=new JTextField();
        textfield1.setBounds(10,10,200,30);
        add(textfield1);
        textarea1=new JTextArea();
        scrollpane1=new JScrollPane(textarea1);
        scrollpane1.setBounds(10,50,400,300);
        add(scrollpane1);
    }

    public static void main(String[] ar) {
        Formulario formulario1=new Formulario();
        formulario1.setBounds(0,0,540,400);
        formulario1.setVisible(true);
    }
}
```



Declaramos los dos objetos:

```
private JScrollPane scrollpane1;

private JTextArea textarea1;
```

Primero creamos el objeto de la clase JTextArea:

```
textarea1=new JTextArea();
```

Seguidamente creamos el objeto de la clase JScrollPane y le pasamos como parámetro el objeto de la clase JTextArea:

```
scrollpane1=new JScrollPane(textarea1);
```

Definimos la posición y tamaño del control de tipo JScrollPane (y no del control JTextArea):

```
scrollpane1.setBounds(10,50,400,300);
```

Finalmente añadimos el control de tipo JScrollPane al JFrame:

```
add(scrollpane1);
```

Problema 2:

Confeccionar un programa que permita ingresar en un control de tipo JTextArea una carta. Luego al presionar un botón mostrar un mensaje si la carta contiene el String "argentina".

Programa:

```
import javax.swing.*;

import java.awt.event.*;

public class Formulario extends JFrame implements ActionListener{

   private JScrollPane scrollpane1;

   private JTextArea textarea1;

   private JButton boton1;

   public Formulario() {

      setLayout(null);

      textarea1=new JTextArea();

      scrollpane1=new JScrollPane(textarea1);

      scrollpane1.setBounds(10,10,300,200);

      add(scrollpane1);

      boton1=new JButton("Verificar");

      boton1.setBounds(10,260,100,30);

      add(boton1);

      boton1.addActionListener(this);

   }


   public void actionPerformed(ActionEvent e) {

      if (e.getSource()==boton1) {

         String texto=textarea1.getText();
```

```
        if (texto.indexOf("argentina")!=-1) {
           setTitle("Si contiene el texto \"argentina\"");
        } else {
           setTitle("No contiene el texto \"argentina\"");
        }
      }
   }

   public static void main(String[] ar) {
      Formulario formulario1=new Formulario();
      formulario1.setBounds(0,0,400,380);
      formulario1.setVisible(true);
   }
}
```

Cuando se presiona el botón se ejecuta el método actionPerformed y procedemos a extraer el contenido del control TextArea a través del método getText:

```
        String texto=textarea1.getText();
```

Luego mediante el método indexOf de la clase String verificamos si el String "argentina" está contenido en la variable texto:

```
        if (texto.indexOf("argentina")!=-1) {
           setTitle("Si contiene el texto \"argentina\"");
        } else {
           setTitle("No contiene el texto \"argentina\"");
        }
```

Si queremos introducir una comilla doble dentro de un String de Java debemos antecederle la barra invertida (luego dicho caracter no se lo considera parte del String):

```
setTitle("Si contiene el texto \"argentina\"");
```

Problemas propuestos

1. Disponer dos controles de tipo JTextArea, luego al presionar un botón verificar si tienen exactamente el mismo contenido.

Solución

```
import javax.swing.*;
import java.awt.event.*;
public class Formulario extends JFrame implements ActionListener{
   private JScrollPane scrollpane1,scrollpane2;
   private JTextArea textarea1,textarea2;
   private JButton boton1;
   public Formulario() {
      setLayout(null);
      textarea1=new JTextArea();
      scrollpane1=new JScrollPane(textarea1);
      scrollpane1.setBounds(10,10,200,140);
      add(scrollpane1);
      textarea2=new JTextArea();
      scrollpane2=new JScrollPane(textarea2);
      scrollpane2.setBounds(220,10,200,140);
      add(scrollpane2);
      boton1=new JButton("Verificar contenidos");
      boton1.setBounds(10,170,150,30);
      add(boton1);
      boton1.addActionListener(this);
   }
```

```
    public void actionPerformed(ActionEvent e) {
       if (e.getSource()==boton1) {
          String texto1=textarea1.getText();
          String texto2=textarea2.getText();
          if (texto1.equals(texto2)==true) {
             setTitle("Los dos controles tiene el mismo texto.");
          } else {
           setTitle("Los dos controles no tiene el mismo texto.");
          }
       }
    }

    public static void main(String[] ar) {
       Formulario formulario1=new Formulario();
       formulario1.setBounds(0,0,500,350);
       formulario1.setVisible(true);
    }
}
```

Swing - JComboBox

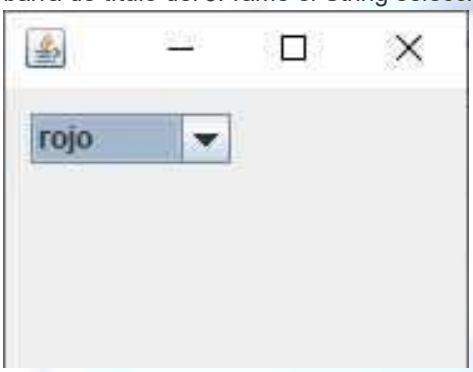
El control JComboBox permite seleccionar un String de una lista.

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:

```
import javax.swing.*;

import java.awt.event.*;

public class Formulario extends JFrame implements ItemListener{

   private JComboBox combo1;

   public Formulario() {

      setLayout(null);

      combo1=new JComboBox();

      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.getSelectedItem();
            setTitle(seleccionado);
        }
    }

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

Indicamos a la clase que implementaremos la interface ItemListener:

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

Declaramos un objeto de la clase ComboBox:

```
private JComboBox combo1;
```

En el constructor creamos el objeto de la clase JComboBox:

```
combo1=new JComboBox();
```

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:

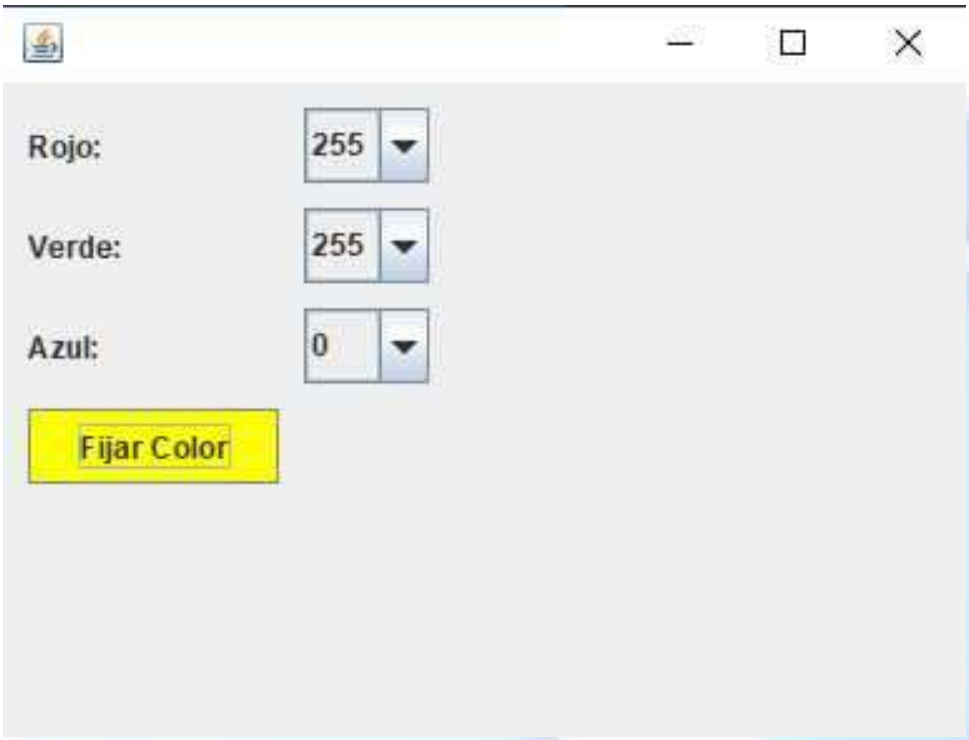
```
public void itemStateChanged(ItemEvent e) {
   if (e.getSource()==combo1) {
      String seleccionado=(String)combo1.getSelectedItem();
      setTitle(seleccionado);
   }
}
```

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:

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.*;

public class Formulario extends JFrame implements ActionListener{

   private JLabel label1,label2,label3;

   private JComboBox 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();
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();
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();
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.getSelectedItem();
        String cad2=(String)combo2.getSelectedItem();
        String cad3=(String)combo3.getSelectedItem();
        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);
   }
}
```

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 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();
      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);
```

Problemas propuestos

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.

Solución

```
import javax.swing.*;
import java.awt.event.*;
public class Formulario extends JFrame implements ActionListener{
   private JLabel label1,label2;
   private JTextField textfield1;
   private JComboBox combo1;
   private JButton boton1;
   public Formulario() {
      setLayout(null);
      label1=new JLabel("Usuario:");
      label1.setBounds(10,10,100,30);
      add(label1);
      textfield1=new JTextField();
      textfield1.setBounds(120,10,120,30);
      add(textfield1);
      label2=new JLabel();
      label2.setBounds(10,50,100,30);
      add(label2);
      combo1=new JComboBox();
      combo1.setBounds(120,50,100,30);
      combo1.addItem("Argentina");
      combo1.addItem("Chile");
```

```
        combo1.addItem("España");
        combo1.addItem("Brasil");
        add(combo1);
        boton1=new JButton("Confirmar");
        boton1.setBounds(10,100,100,30);
        boton1.addActionListener(this);
        add(boton1);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource()==boton1) {
            String nombre=textfield1.getText();
            String pais=(String)combo1.getSelectedItem();
            setTitle(nombre+" - "+pais);
        }
    }

    public static void main(String[] ar) {
        Formulario formulario1=new Formulario();
        formulario1.setBounds(10,20,300,200);
        formulario1.setVisible(true);
    }
}
```

Muchas gracias hasta la próxima clase.

Alsina 16 [B1642FNB] San Isidro | Pcia. De Buenos Aires |Argentina |

TEL.: [011] 4742-1532 o [011] 4742-1665 |

www.institutosanisidro.com.ar info@institutosanisidro.com.ar