ACTIVIDAD GUIADA 4
Componentes de texto

Etiquetas y campos de texto

Un componente **`JLabel`** muestra información que el usuario no puede modificar. Esta información puede ser texto, gráficos o ambos. Estos componentes suelen emplearse para etiquetar a otros componentes de una interfaz, de ahí el nombre de **etiqueta**, y suelen identificar campos de texto.

Un componente **`JTextField`** es una zona en la que el usuario puede introducir una sola línea de texto. Puede definir la anchura del cuadro al crear el campo de texto. Las siguientes instrucciones, crean un componente **`JLabel`** y un objeto **`JTextField`** y los añaden a un contenedor:

```
public class FormularioWeb extends JFrame {
public FormularioWeb() {
    super("TextField");
    setLookAndFeel();

    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JLabel lblaWeb = new JLabel("Página Web: ", JLabel.RIGHT);
    JTextField txtWeb = new JTextField(20);

    FlowLayout admin = new FlowLayout();
    setLayout(admin);

    add(lblWeb);
    add(txtWeb);

    pack();
    setVisible(true);
}
public static void main(String[] arguments) {
    FormularioWeb frFormularioWeb = new FormularioWeb();
}
}
```

En la figura siguiente puede ver esta etiqueta y el campo de texto. Ambas instrucciones usan un argumento para configurar el aspecto del componente.

La etiqueta **lblWeb** se establece en el texto "Página Web:" y un argumento **JLabel.RIGHT**. Este valor indica que la etiqueta debe aparecer desplazada a la derecha. **JLabel.LEFT** alinea el texto de la etiqueta a la izquierda y **JLabel.CENTER** lo centra. El argumento usado con **JTextField** indica que el campo de texto debe tener 20 caracteres de ancho. También puede especificar un texto predeterminado que aparezca en el campo por medio de la siguiente instrucción:

```
JTextField txtPais = new JTextField("Introduzca un país",20);
```

Esta instrucción crea un objeto **JTextField** de 20 caracteres de ancho y con el texto *Introduzca un país* dentro del campo.

También podemos establecer el texto con el método adecuado:

```
txtPais.setText("España");
```

Podemos recuperar el texto incluido dentro del objeto a través de método **getText()** que devuelve una cadena:

```
String paisElegido=txtPais.getText();
```

Este método lo usaremos más adelante para manipular los datos del formulario.

JFormattedTextField

Como ya hemos dicho JTextField permite introducir texto, pero no hace ningún tipo de comprobación. Un componente más evolucionado que es JFormattedTextField. Aunque el usuario mete un texto, el campo lo convierte a la clase que queramos (Integer, Float, etc.), encargándose de las conversiones.

Por ejemplo, si queremos admitir sólo números:

```
JFormattedTextField textField1 = new JFormattedTextField (new
Integer(3));
```

También es posible introducir máscaras más específicas con **MaskFormatter**.

JPasswordField

Por su uso intensivo en todo tipo de programas

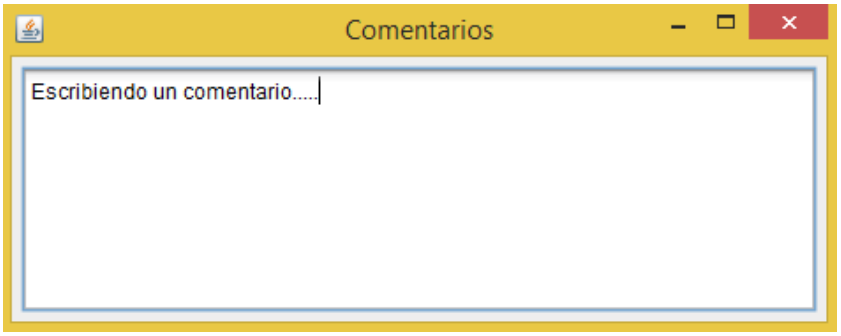
```
final JFrame frame = new JFrame("JPassword Usage Demo");
JLabel jlbPassword = new JLabel("Enter the password: ");
JPasswordField jpwName = new JPasswordField(10);
jpwName.setEchoChar('*');
```

JTextArea

```
JTextArea taComentarios = new JTextArea(8, 40);
FlowLayout admin = new FlowLayout();
setLayout(admin);
add(taComentarios);
```

Un componente **JTextArea** es un campo de texto que permite al usuario introducir más de una líneas de texto. Se puede especificar la anchura y la altura del componente. Las siguientes instrucciones crean un componente **JTextArea** con una anchura de 40 caracteres y una altura de 8 líneas, y lo añaden a un contenedor:

En la figura siguiente se muestra este ejemplo en un marco:



El código completo sería:

```
import java.awt.FlowLayout;
import javax.swing.*;

public class Comentarios extends JFrame {
    public Comentarios(){
    super("Comentarios");
    setLookAndFeel();

    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

   FlowLayout admin = new FlowLayout();
   setLayout(admin);

    JTextArea taComentarios = new JTextArea(8, 40);
    add(taComentarios);

    pack();
    setVisible(true);
    }

 public static void main(String[] args) {
       Comentarios frComentarios=new Comentarios();
    }
}
```

Se puede especificar una cadena en el constructor **`JTextArea()`** para mostrarla en el área de texto, usando el carácter de nueva línea (\n) para representar los saltos de línea:

```
JTextArea comentarios = new JTextArea("Esta es la primera línea de texto\n"
+ "…y esta es la segunda.", 8, 40);
```

Si queremos hacer cosas más avanzadas, como permitir colores, tipos de letras, o que el campo esté destinado a un tipo de texto concreto como HTML, existen opciones más específicas como **JEditorPane** o **JTextPane**.

***NOTA**: los componentes de área de texto pueden comportarse de forma inesperada, por lo que pueden aumentar de tamaño cuando el usuario llega a la parte inferior o no incluir barras de desplazamiento en los bordes. Para implementar el tipo de áreas de texto habituales de otras GUI, se debe incluir el área dentro de un contenedor panel de desplazamiento.*