IES de Teis
Avda. de Galicia, 101
36216 – Vigo
Tfno: 886 12 04 64
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

PRÁCTICA PRUEBA: INTERFAZ GRÁFICA SWING – LENGUAJE JAVA

Objetivos:

- *Iniciar al alumno en el desarrollo y manejo de interfaces gráficas*

Introducción

1. ***Swing***

Swing es una biblioteca de interfaces gráficas de usuario (GUI) para Java.

- Viene incluida con el entorno de desarrollo de Java (JDK).
- Extiende a otra librería gráfica más antigua llamada AWT.

- Paquetes:
  - javax.swing
  - java.awt
  - java.awt.event

2. ***Ventana:*** La clase JFrame proporciona operaciones para manipular ventanas.

- Constructores:
  - JFrame()
  - JFrame(String titulo)
- Una vez creado el objeto de ventana, hay que:
  - Establecer su tamaño.
  - Establecer la acción de cierre.
  - Hacerla visible.

3. ***Creación de ventanas***

Para construir una interface gráfica de usuario hace falta

- Un "contenedor" o *container*, que es la ventana o parte de la ventana donde se situarán los componentes (botones, barras de desplazamiento, etc.) y donde se realizarán los dibujos.
- Los *componentes*: menús, botones de comando, barras de desplazamiento, cajas y áreas de texto, botones de opción y selección, etc.
- *El modelo de eventos. El usuario controla la aplicación actuando sobre los componentes, de ordinario con el ratón o con el teclado. Cada vez que el usuario realiza una determinada acción, se produce el evento correspondiente, que el sistema operativo transmite al AWT. El AWT crea un objeto de una determinada clase de evento, derivada de AWTEvent. Este evento es transmitido a un determinado método para que lo gestione.*

```
import javax.swing.*;
public class VentanaTest {
        public static void main(String[] args) {
                JFrame f = new JFrame("Titulo de ventana");
                f.setSize(400, 300);
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setVisible(true);
        }
}
```

4. ***Acciones de cierre:***

- JFrame.EXIT_ON_CLOSE: Abandona aplicación.

- JFrame.DISPOSE_ON_CLOSE: Libera los recursos asociados a la ventana.
- JFrame.DO_NOTHING_ON_CLOSE: No hace nada.
- JFrame.HIDE_ON_CLOSE: Cierra la ventana, sin liberar sus recursos.

5. *Creación de clase ventana que hereda de Jframe:*

Es usual extender la clase JFrame, y realizar las operaciones de inicialización en su constructor.

```
public class MiVentana extends JFrame {
        public MiVentana() {
                super("Titulo de ventana");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        }
}
public class VentanaTest {
        public static void main(String[] args) {
                MiVentana v = new MiVentana();
                v.setVisible(true);
        }
}
```

6. *Componentes de una ventana:*

Para añadir componentes a una ventana, como los que se muestran en esta imagen es necesario crearlos con **new** y a continuación añadirse al contentPane de la ventana correspondiente mediante su método add.

- **JFrame:** Representa una ventana básica, capaz de contener otros componentes. Casi todas las aplicaciones construyen al menos un Jframe
- **Jbutton, JCheckBox, JRadioButton:** Distintos tipos de botones. Un check box sirve para marcar una opción. Un radio button permite seleccionar una opción entre varias disponibles.
- **JcheckBox:** La versión Swing soporta botones checkbox con la clase JCheckBox. Swing también soporta checkboxes en menús, utilizando la clase JCheckBoxMenuItem. Como JcheckBoxMenuItem y JcheckBox descienden de AbstractButton, los checkboxes de Swing tienen todas las características de un botón normal.
- Los checkboxes son similares a los botones de radio, pero no su modelo de selección:ninguno, alguno o todos, pueden ser seleccionados. Sin embargo en un grupo de botones deradio, solo

puede haber uno seleccionado.Los métodos de AbstractButton que son mas utilizados son setMnemonic, addItemListener, setSelected y isSelected

Constructores y métodos

Constructor	Propósito
JCheckBox(String) **JCheckBox(String,boolean)** **JCheckBox(Icon)** **JCheckBox(Icon,boolean)** **JCheckBox(String,Icon)** **JCheckBox(String,Icon,boolean)** **JCheckBox()**	Crea un ejemplar de **JCheckBox**. El argumento string especifica el texto, si existe, que el checkbox debería mostrar. De forma similar, el argumento **Icon** especifica la imagen que debería utilizarse en vez de la imagen por defecto del aspecto y comportamiento. Especificando el argumento booleano como **true** se inicializa el checkbox como seleccionado. Si el argumento booleano no existe o es **false**, el checkbox estará inicialmente desactivado.
JCheckBoxMenuItem(String) **JCheckBoxMenuItem(String,boolean)** **JCheckBoxMenuItem(Icon)** **JCheckBoxMenuItem(String,Icon)** **JCheckBoxMenuItem(String,Icon, boolean)** **JCheckBoxMenuItem()**	Crea un ejemplar de **JCheckBoxMenuItem**. Los argumentos se interpretan de la misma forma que en los constructores de **JCheckBox**.

- ***JRadioButton:***

  Métodos y constructores:

-ButtonGroups

Método	Propósito
ButtonGroup()	Crea un ejemplar de **ButtonGroup**.
void add(AbstractButton) **void remove(AbstractButton)**	Añade un botón a un grupo, o elimina un botón de un grupo

-RadioButton

Constructor	Propósito
JRadioButton(String) **JRadioButton(String,boolean)** **JRadioButton(Icon)** **JRadioButton(Icon, boolean)** **JRadioButton(String, Icon)** **JRadioButton(String, Icon, boolean)** **JRadioButton()**	Crea un ejemplar de **JRadioButton**. El argumento string especifica el texto, si existe, que debe mostrar el botón de radio. Similarmente, el argumento, **Icon** especifica la imagen que debe usar en vez la imagen por defecto de un botón de radio para el aspecto y comportamiento. Si se especifica **true** en el argumento booleano, inicializa el botón de radio como seleccionado, sujeto a la aprovación del objeto **ButtonGroup**. Si el argumento booleano esta ausente o es **false**, el botón de radio está inicialmente deseleccionado.
JRadioButtonMenuItem(String) **JRadioButtonMenuItem(Icon)** **JRadioButtonMenuItem(String, Icon)** **JRadioButtonMenuItem()**	Crea un ejemplar de **JRadioButtonMenuItem**. Los argumentos se interpretan de la misma forma que los de los constructores de **JRadioButton**.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

Secuencia y desarrollo:

1. Implementa el siguiente ejemplo:

```
import javax.swing.*;

public class Ejemplo1 extends JFrame{

public static void main(String args[]){

        Ejemplo1 miVentana = new Ejemplo1();
        JPanel miPanel = new JPanel();
        JButton boton1 = new JButton("Boton 1");
        JButton boton2 = new JButton("Este es el boton 2");
        JButton boton3 = new JButton("Boton 3");
        miPanel.add(boton1);
        miPanel.add(boton2);
        miPanel.add(boton3);

        miVentana.setContentPane(panel);
        miVentana.setSize(350,150);
        miVentana.setTitle("Prueba de FlowLayout");
        miVentana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        miVentana.setVisible(true);
}
 }
```

2. Añade los componentes que aparecen en la introducción de la práctica, comprueba como se van colocando en la ventana creada.
3. Modifica el ejemplo1 pero en vez de declarar todo en la función main, debes crear una clase ventana, y en el main crear un objeto de tipo ventana, donde le añades los componentes.

4. Implementa la siguiente interface