


# Lenguaje de programación Java SE 8



Módulo 2 - Clase #06  
Carrera Java Programmer SE 8



# Java Swing

---

## Características:

- 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 (Abstract Window Toolkit).
- Paquetes:
  - javax.swing
  - java.awt
  - java.awt.event

# Ventanas (JFrame)

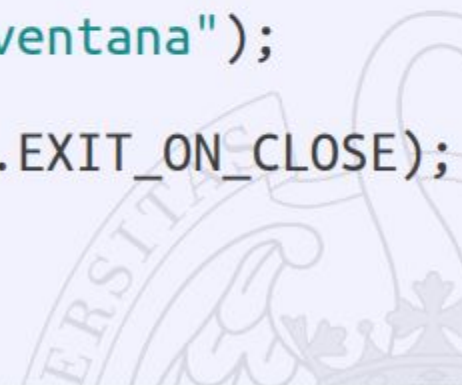
---

- La clase JFrame proporciona operaciones para manipular ventanas.
- Constructores:
  - JFrame()
  - JFrame(String titulo)
- Una vez creado el objeto de la ventana, hay que:
  - Establecer su tamaño.
  - Establecer la acción de cierre.
  - Hacerla visible.
- 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.

# Ventanas (JFrame)

```
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);
    }
}
```



# Ventanas (JFrame)

```
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);  
    }  
}
```

# Ejemplo



# Componentes de una Ventana



**JButton**

**JLabel**

**TextField**

**CheckBox**

**RadioButton**

Tras crear uno de estos componentes con `new`, ha de añadirse al `ContentPanel` de la ventana correspondiente mediante su método **`add`**.

# Componentes de una Ventana

```
public class MiVentana extends JFrame {  
    public MiVentana() {  
        super("Titulo de ventana");  
        setSize(400, 300);  
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
        Container cp = getContentPane();  
        cp.setLayout(new FlowLayout());  
        JLabel etiqueta = new JLabel("Nombre: ");  
        JTextField texto = new JTextField(20);  
        JButton boton = new JButton("Saludar");  
        cp.add(etiqueta);  
        cp.add(texto);  
        cp.add(boton);  
    }  
}
```



# Ejemplo



# Layout Managers

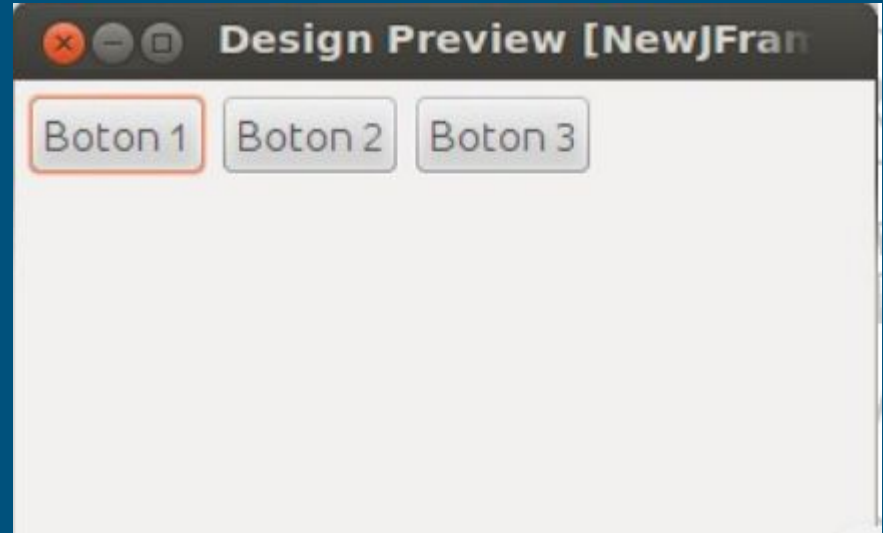
---

- En Java no es habitual indicar explícitamente la posición de los componentes de la interfaz dentro de la ventana.
- Los layout managers se encargan de colocar los componentes de la interfaz de usuario en la ventana contenedora.
- Especifican la posición y el tamaño de dichos componentes:
  - FlowLayout.
  - GridLayout.
  - BorderLayout.

# FlowLayout

## Características:

- Coloca los elementos uno a continuación de otro, de manera similar a la colocación de palabras en un procesador de textos.
- Métodos:
  - `setAligment(int alineacion)`
  - `setHgap(int separacion)`
  - `setVgap(int separacion)`



# Ejemplo



# GridLayout

## Características:

- Coloca los componentes de la interfaz en forma de rejilla.
- El orden en que se añaden los componentes determina su posición en la rejilla.
- Constructor:
  - `GridLayout(int filas, int columnas)`
- Métodos:
  - `setHgap(int separacion)`
  - `setVgap(int separacion)`



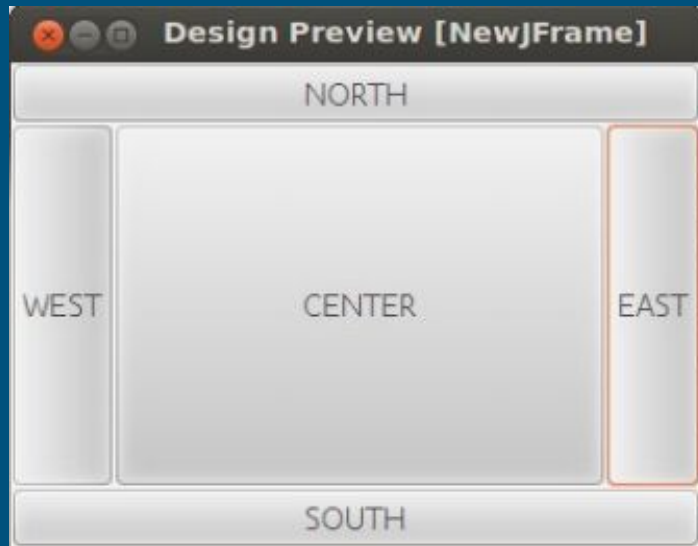
# Ejemplo



# BorderLayout

## Características:

- Coloca y cambia de tamaño sus componentes para que se ajusten a los bordes y parte central de la ventana.
- Métodos:
  - `setHgap(int separacion)`
  - `setVgap(int separacion)`
- Al añadir un elemento a la ventana, hay que especificar su colocación:



```
JButton b = new JButton(...);  
getContentPane().add(b, BorderLayout.EAST)
```

# Ejemplo

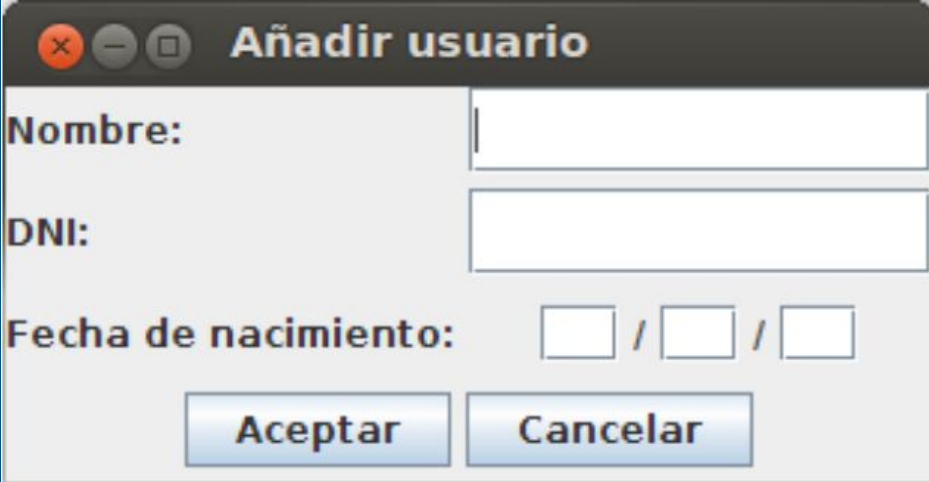




# Interfaces Complejas: JPanel

## Características:

- Un panel es un componente con un layout manager propio, y que puede contener varios componentes en su interior.
- Constructor:
  - `JPanel()`
- Métodos:
  - `void setLayout(LayoutManager lm)`
  - `void add(JComponent componente)`



A screenshot of a Java Swing window titled "Añadir usuario". The window has a standard title bar with a close button (red circle with an 'x'), a minimize button (grey circle with a dash), and a maximize button (grey square). The main content area is light grey and contains three labels with corresponding input fields: "Nombre:" followed by a single-line text field, "DNI:" followed by a single-line text field, and "Fecha de nacimiento:" followed by three single-digit text fields separated by forward slashes. At the bottom of the window, there are two buttons: "Aceptar" and "Cancelar".

# Interfaces Complejas: JPanel

```
public MiVentana3() {  
    super("Añadir usuario");  
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    // Panel de fecha  
    JPanel panelFecha = new JPanel();  
    panelFecha.setLayout(new FlowLayout());  
    panelFecha.add(new JTextField(2));  
    panelFecha.add(new JLabel("/"));  
    panelFecha.add(new JTextField(2));  
    panelFecha.add(new JLabel("/"));  
    panelFecha.add(new JTextField(2));  
    // Panel de datos  
    JPanel panelDatos = new JPanel();  
    GridLayout gl = new GridLayout(3,2,0,5);  
    panelDatos.setLayout(gl);  
    panelDatos.add(new JLabel("Nombre:"));  
    panelDatos.add(new JTextField(10));  
    panelDatos.add(new JLabel("DNI:"));  
    panelDatos.add(new JTextField(10));  
    panelDatos.add(new JLabel("Fecha de nacimiento: "));  
    panelDatos.add(panelFecha);  
    ...  
}
```

```
    // Panel de botones  
    JPanel panelBotones = new JPanel();  
    panelBotones.setLayout(new FlowLayout());  
    panelBotones.add(new JButton("Aceptar"));  
    panelBotones.add(new JButton("Cancelar"));  
  
    Container cp = getContentPane();  
    cp.add(panelDatos, BorderLayout.CENTER);  
    cp.add(panelBotones, BorderLayout.SOUTH);  
}
```

# Manejo de Eventos

## Características:


- Un evento es un suceso que ocurre como consecuencia de la interacción del usuario con la interfaz gráfica:
  - Pulsación de un botón.
  - Cambio del contenido en un cuadro de texto.
  - Deslizamiento de una barra.
  - Activación de un JCheckBox.
  - Movimiento de la ventana.
- La clase **JButton** tiene un método:
  - `void addActionListener(ActionListener l)`
- Que especifica el objeto (manejador de evento) que se encargará de tratar el evento de pulsación del botón.
- Este objeto ha de interpretar la interfaz `ActionListener` (paquete `java.awt.event`).

```
public interface ActionListener {  
    void actionPerformed(ActionEvent e)  
}
```

# Manejo de Eventos

- Cuando el usuario pulse el botón, se llamará al método `actionPerformed` de todos los manejadores de eventos que se hayan registrado.
- Métodos de `ActionEvent`:
  - `public Object getSource()`
  - `public int getModifiers()`

```
public class Manejador implements ActionListener {  
    public void actionPerformed(ActionEvent e) {  
        ...  
    }  
}
```



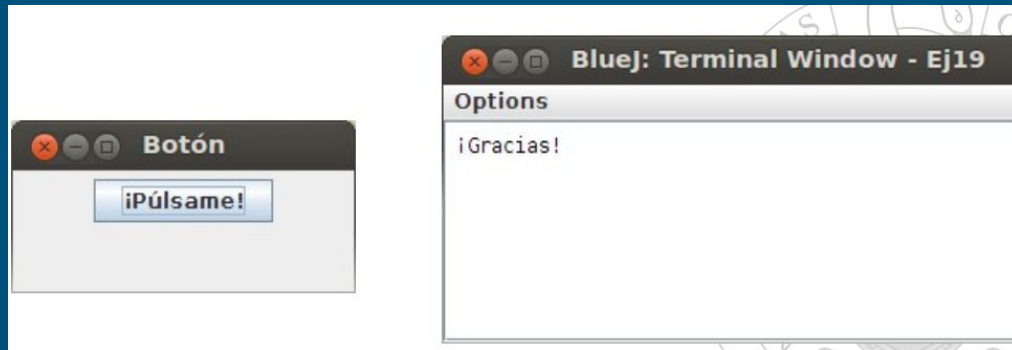
**Información sobre el evento**

# Manejo de Eventos

```
public class BotonVentana extends JFrame {  
    public BotonVentana() {  
        super("Botón");  
        setSize(200,100);  
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
        Container cp = getContentPane();  
        cp.setLayout(new FlowLayout());  
        JButton boton = new JButton("¡Púlsame!");  
        boton.addActionListener(new EventoBotonPulsado());  
        cp.add(boton);  
    }  
}
```

# Manejo de Eventos

```
public class EventoBotonPulsado implements ActionListener {  
    public void actionPerformed(ActionEvent e) {  
        System.out.println("¡Gracias!");  
    }  
}
```



# Ejemplo



# Uso de la Herramienta Gráfica de NetBeans

---



# Ejemplo

