


Métodos Avanzados de Programación Científica y Computación

M^a Luisa Díez Platas

Tema 6. Interfaces de usuario(II)

¿Cómo estudiar este tema?

IDEAS CLAVE	LO + RECOMENDADO	+ INFORMACIÓN	TEST
<p>¿Cómo estudiar este tema?</p> <p>AWT</p> <p>Gestores de posicionamiento</p> <p>Gestión de eventos</p> <p>Adaptadores</p>	<p>No dejes de leer...</p> <p>Clase <code>Box</code></p> <p>No dejes de ver...</p> <p>TV Crear interfaces de usuario</p> <p>TV Gestionar eventos en Java</p> <p>TV Crear eventos en Java</p>	<p>A fondo</p> <p>Desarrollo y generación de interfaces de usuario</p> <p>Verificar interfaces de usuario</p> <p>Uso de componentes en SWING</p> <p>Gestores de posicionamiento</p> <p>Recursos externos</p> <p>Eclipse</p>	

Interfaz gráfica de usuario

Interfaz formada por un conjunto de componentes que permiten al usuario interactuar con la aplicación.

AWT (*Abstract Window Toolkit*). Caja de herramientas abstractas para la creación de ventanas mediante jerarquía de clases de Java

- Herramientas originales de Java para la creación de interfaces de usuario.
- Soportado por JDK 1.0 y 1.1.
- Utiliza código nativo de la plataforma en la que se ejecuta.
- No incluye componentes complejos.
- Utiliza un modelo de manejo de eventos robusto.
- Los componentes no se comportan igual en todas las plataformas.

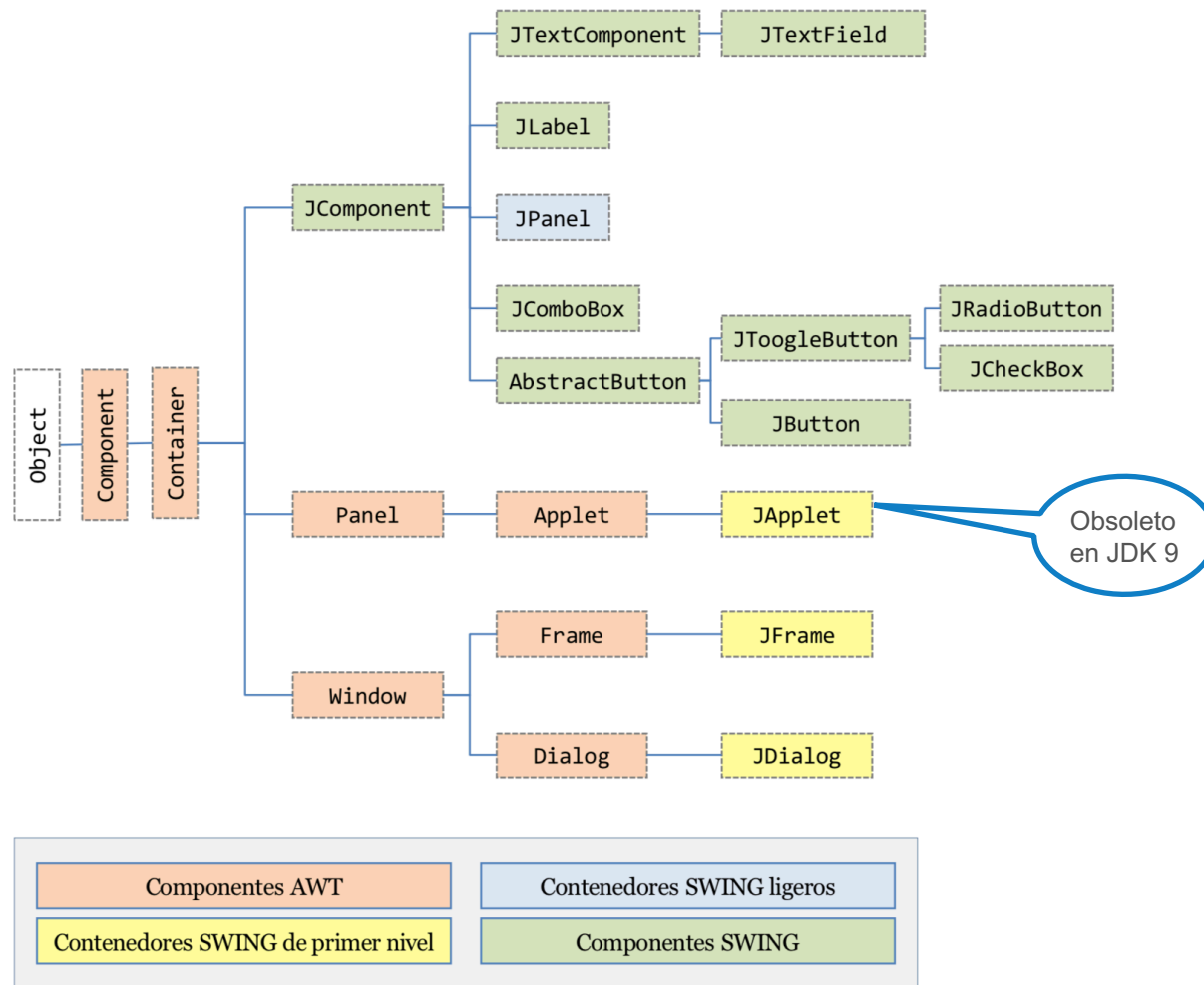
SWING

- Sus componentes están contruidos sobre AWT.
- Soportado por JDK 1.2
- No utiliza código nativo.
- Amplio conjunto de componentes
- Los componentes se comportan igual en todas las plataformas.

Swing

Usa el comportamiento de AWT

- Eventos
- Administradores de disposición



AWT

- **Administrador de disposición**, gestor de posicionamiento o administrador de diseño

- Se implementa la interfaz `LayoutManager` de `java.awt`.
- Tipos de administradores

`FlowLayout`

`BorderLayout`

`GridLayout`

`GridBagLayout`

`BoxLayout`

`SpringLayout`



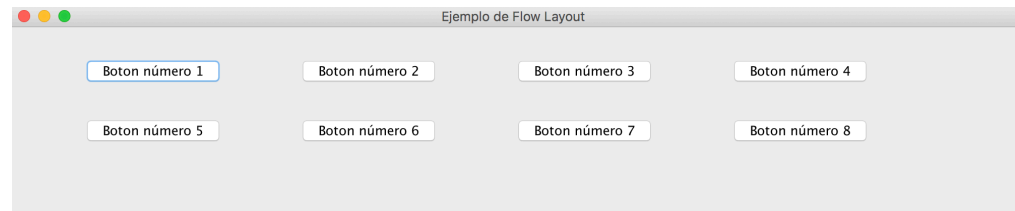
`setLayout(LayoutManager m)`
establece el gestor de
posicionamiento.

Para distribuciones más complejas usar `JPanel` en los contenedores.

FlowLayout

Diseño básico, los componentes se encadenan en forma de lista, de izquierda a derecha y de abajo a arriba.

Selección de espacio vertical y horizontal entre componentes.

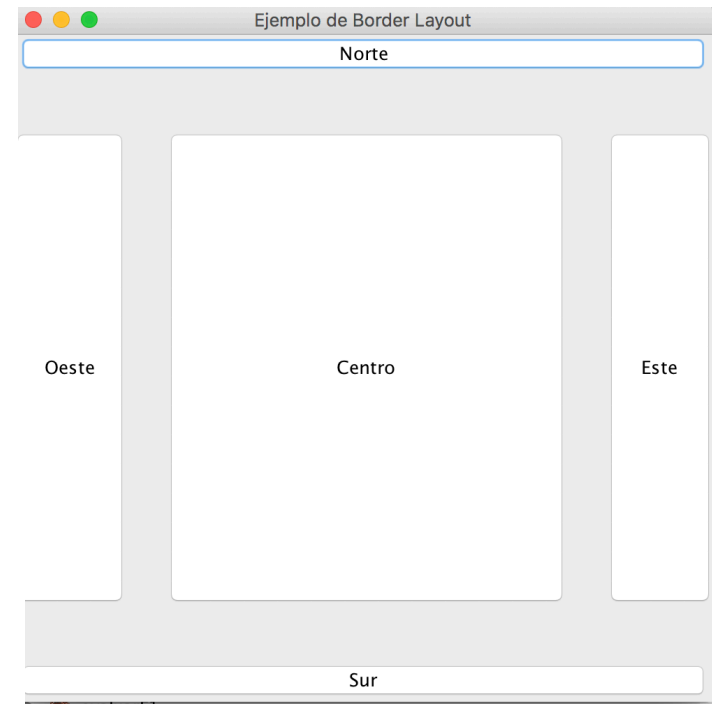


```
setLayout(new FlowLayout(FlowLayout.LEFT,70,30));  
for (int i=1; i<=8; i++){  
    add(new JButton("Boton número "+i));  
}
```

BorderLayout

Cinco zonas para colocar los componentes.
Layout por defecto en JFrame y JDialog.

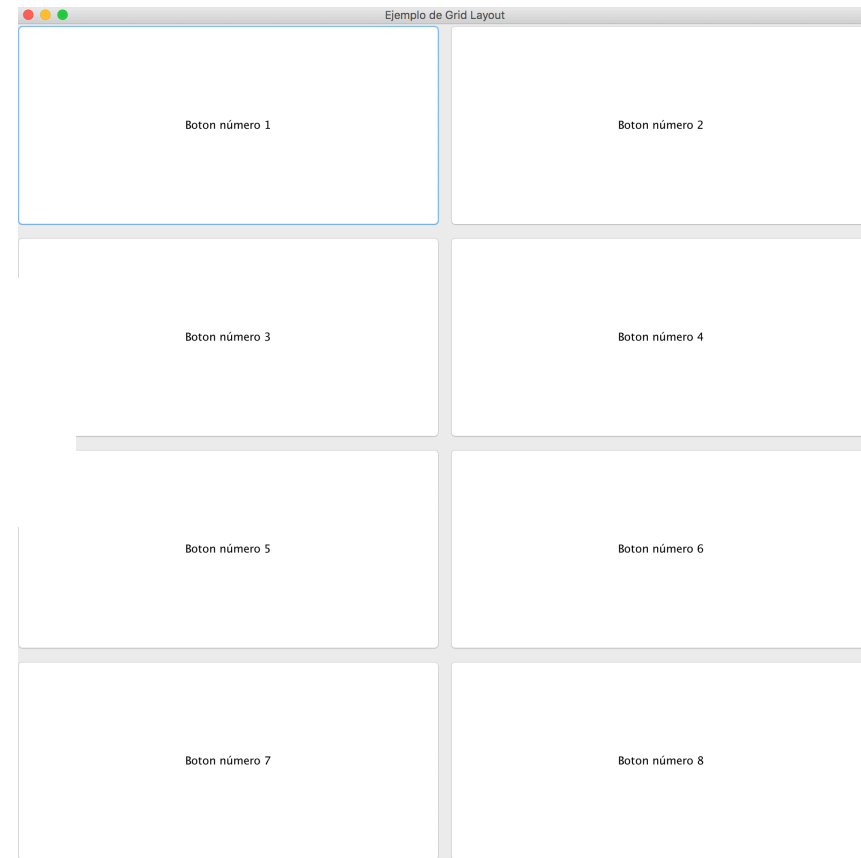
```
setLayout(new BorderLayout(30, 40));  
add(new JButton("Este"), BorderLayout.EAST);  
add(new JButton("Sur"), BorderLayout.SOUTH);  
add(new JButton("Oeste"), BorderLayout.WEST);  
add(new JButton("Norte"), BorderLayout.NORTH);  
add(new JButton("Centro"), BorderLayout.CENTER);
```



GridLayout

Zona a modo de cuadrícula (**filas x columnas**) de componentes colocados de izquierda a derecha y de arriba a abajo.

```
setLayout(new GridLayout(4,3,10,10));  
for (int i=1; i<=8; i++){  
    add(new JButton("Boton número "+i));  
}
```



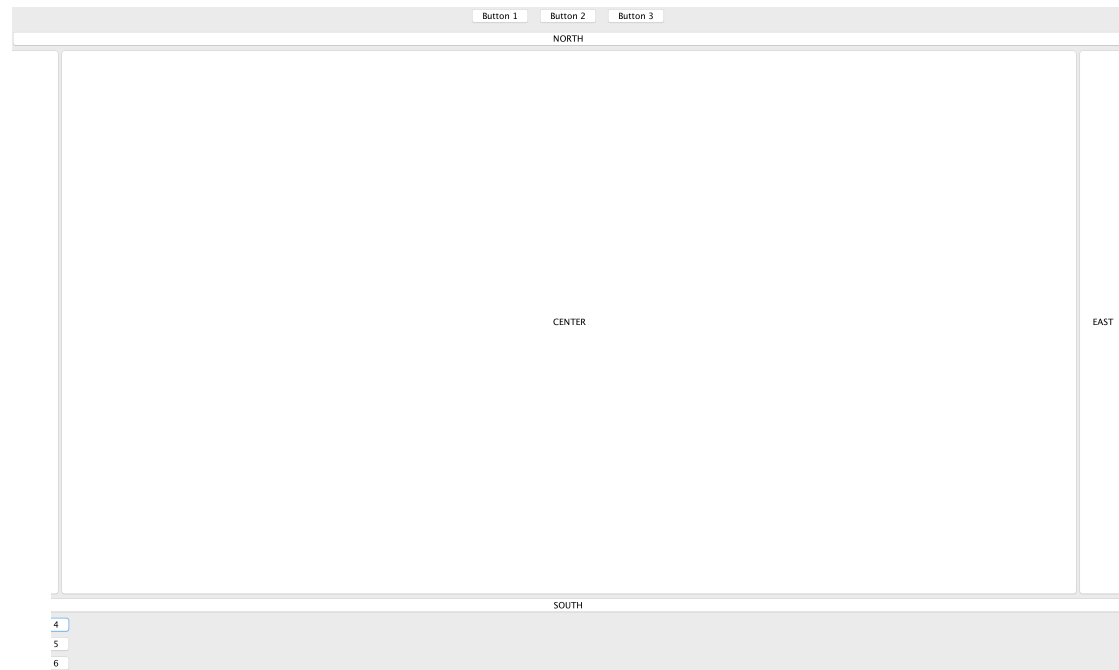
JPanel

Contenedor al que se le puede asignar un gestor de posicionamiento propio.

Muy útil para agrupar componentes.

```
// new GUI application logic here
JPanel panel1 = new JPanel(new FlowLayout());
JPanel panel2 = new JPanel(new BorderLayout());
JPanel panel3 = new JPanel();
panel1.add(new JButton("Button 1"));
panel1.add(new JButton("Button 2"));
panel1.add(new JButton("Button 3"));
panel2.add(new JButton("CENTER"), BorderLayout.CENTER);
panel2.add(new JButton("NORTH"), BorderLayout.NORTH);
panel2.add(new JButton("SOUTH"), BorderLayout.SOUTH);
panel2.add(new JButton("EAST"), BorderLayout.EAST);
panel2.add(new JButton("WEST"), BorderLayout.WEST);
panel3.setLayout(new BoxLayout(panel3, BoxLayout.Y_AXIS));
panel3.add(new JButton("Button 4"));
panel3.add(new JButton("Button 5"));
panel3.add(new JButton("Button 6"));
```

```
this.getContentPane().setLayout(new BorderLayout());
getContentPane().add(panel1, BorderLayout.NORTH);
getContentPane().add(panel2, BorderLayout.CENTER);
getContentPane().add(panel3, BorderLayout.SOUTH);
```



Gestión de Eventos

- Se implementa la interfaz `ActionListener` de `java.awt.event.*`.

```
public class PruebaFlowLayout implements ActionListener{
```

```
    panelFlow= new JPanel(new FlowLayout());
```

Panel que va a verse afectado por el evento

```
    botonPanelFlow= new JButton("Botón para mostrar u ocultar un panel");
```

Botón que captura el evento (oyente)

Existen otras interfaces como `ItemListener`, `FocusListener`, `MouseListener`...

Gestión de Eventos

- Se redefine la función `actionPerformed` (`ActionEvent e`) en la clase que implementa la interfaz.

```
@Override
public void actionPerformed(ActionEvent e) { //se sobrescribe el metodo del listener

    if(e.getSource() == botonPanelFlow){
        if(panelFlow.isVisible()){
            panelFlow.setVisible(false);
        }else{
            panelFlow.setVisible(true);
        }
    }else{
        JOptionPane.showMessageDialog(null, e.getActionCommand());
    }
}
```

El evento capturado por el `botonPanelFlow` mostrará u ocultará el panel dependiendo de su estado

Para otros oyentes muestra ventana emergente con su contenido

- Se asignan eventos a los componentes seleccionados mediante la función `addActionListener(ActionListener objeto)`.

```
botonPanelFlow = new JButton("Botón para mostrar u ocultar un panel");
botonPanelFlow.addActionListener(this);
```

this es un objeto de la clase que ha implementado la interfaz

Algunos tipos de eventos

Tipo de evento	Objeto origen	Acción
ActionEvent	JButton	Pulsar un botón
ActionEvent	TextField	Pulsar en un campo de texto
ActionEvent	JCheckBox	Marcar casilla de verificación
ActionEvent	JRadioButton	Marcar un botón de radio
AdjustmentEvent	JScrollBar	Mover la barra de desplazamiento

UNIVERSIDAD
INTERNACIONAL
DE LA RIOJA

unir

www.unir.net