Programación en Java

Tema 6. Interfaces gráficas de usuario (Parte 1)

Luis Rodríguez Baena

Universidad Pontificia de Salamanca (campus Madrid)
Facultad de Informática

Swing y AWT (I)

- JDK 1.0 introdujo la creación de interfaces gráficas de usuario (GUI, *Graphics User Interfaces*).
  - AWT (*Abstract Windows Toolkit*).
- Java 2: JFC (*Java Foundation Classes*).
  - AWT.
  - Swing.
  - Aspecto configurable (*Pluggable Look and Feel*).
  - Interfaz de accesibilidad.
  - API para dibujo 2D.
  - Soporte *drag and drop.*

Swing y AWT (II)

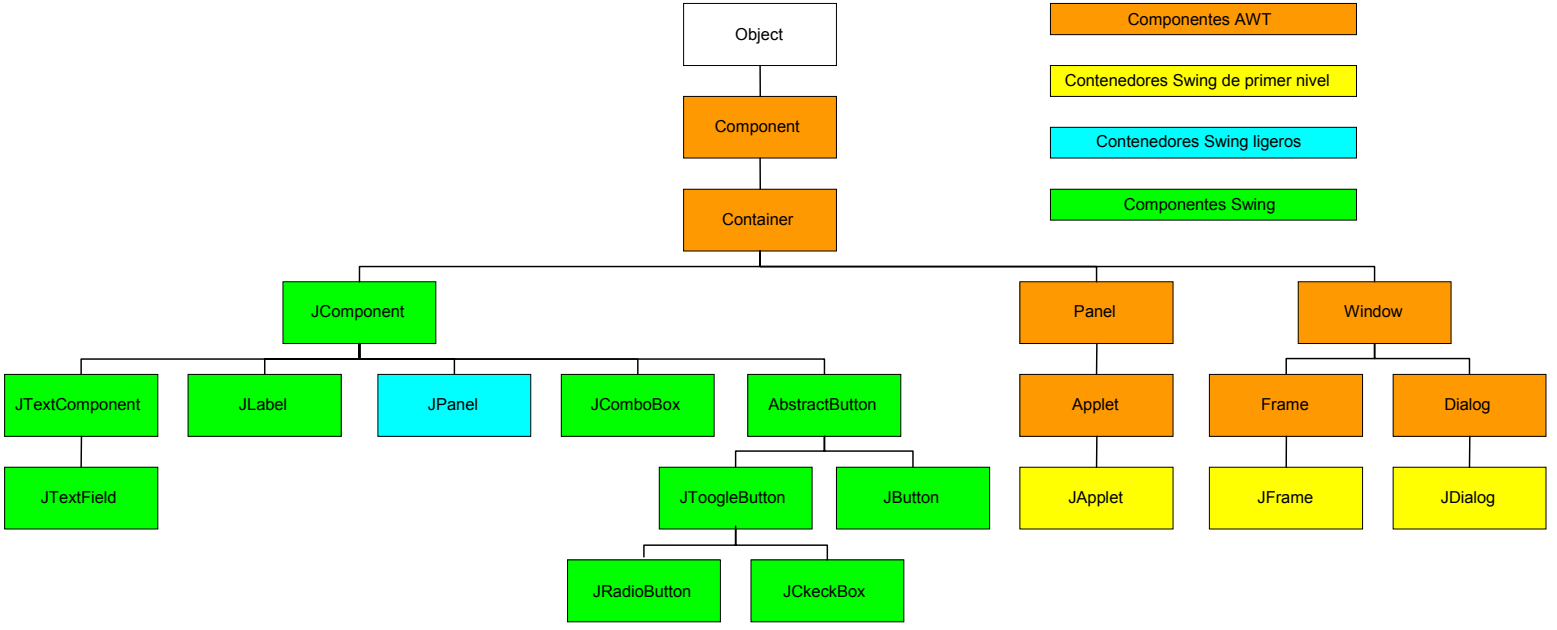
- Diferencias entre Swing y AWT.
  - AWT.
    - ✓ Soportado por JDK 1.0 y 1.1.
    - ✓ Utiliza código nativo de la plataforma en la que se ejecuta el programa.
    - ✓ Resta compatibilidad: no todos los componentes GUI de todas las plataformas se comportan de la misma forma.
  - Swing.
    - ✓ Soportado por JDK 1.2.
    - ✓ No utiliza código nativo.
    - ✓ Todos los componentes se comportan igual en todas las plataformas.
    - ✓ Aspecto distinto según la plataforma.
    - ✓ Conjunto de componentes más extenso y con más características.
    - ✓ Precisa de algunas clases de AWT.

Swing y AWT (III)

- Una interfaz gráfica común va a tener tres elementos.
  - Un contenedor de primer nivel (JFrame, JDialog, JApplet)
  - Componentes de la interfaz gráfica (botones, etiquetas, campos de texto, etc.).
  - Elementos para la gestión de eventos.

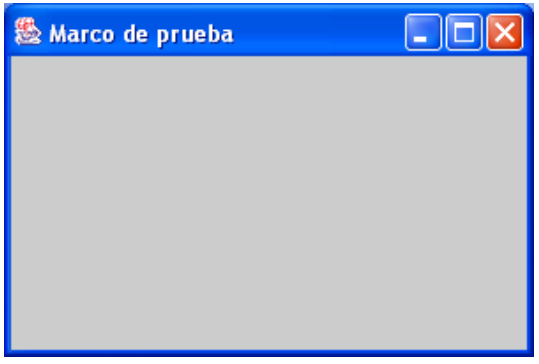
Jerarquía de Swing

(Sólo se incluyen los componentes utilizados)



Ventana principal (I)

- Una aplicación GUI se desarrolla sobre un marco.
  - Se hereda de la clase `JFrame`.
  - Sobre el marco se colocarán los distintos componentes de la interfaz.




```
import javax.swing.*;
public class MarcoPrueba{
  public static void main(String args[]){
    MiMarco marco = new MiMarco("Marco de prueba");
    marco.setDefaultCloseOperation(
         JFrame.EXIT_ON_CLOSE);
    marco.show();
  }
}
class MiMarco extends JFrame{
  final int ALTO = 200;
  final int ANCHO = 300;
  MiMarco(String titulo){
    setTitle(titulo);
    setSize(ANCHO,ALTO);
  }
}
```

Ventana principal (II)

- El método `main` crea una instancia de la clase `MiMarco`.
- El método `showDefaultOperation` indica una acción predeterminada al cerrar la ventana.
  - Disponible a partir de la versión 1.3.
  - Su argumento puede tomar los valores `DO_NOTIHG_ON_CLOSE`, `HIDE_ON_CLOSE` o `EXIT_ON_CLOSE`.
- El método `show()` permite mostrar el marco.

Ventana principal (III)

- `MiMarco` hereda de la clase `JFrame`.
- Su constructor por omisión crea una ventana sin título de un tamaño de 0 por 0 pixels.
- El método `setTitle()` permite establecer el título de la ventana.
- El método `setSize()` crea el marco con el tamaño deseado.
  - El marco también se puede crear `con setVisible(true).`
  - O con `pack()`.

Ventana principal (IV)

- Algunos métodos de JFrame y sus superclases.

Constructores	
JFrame()	Constructor que crea un marco sin título
JFrame(String título)	Constructor que crea un marco con el título indicado
Métodos	
setTitle(String título)	Establece el título de la ventana.
setSize(int alto, int ancho)	Establece el ancho y el alto de la ventana.
setLocation(int x, int y)	Sitúa el marco en la posición x, y.
setBounds(int x,int y,int ancho,int alto)	Sitúa en la posición x, y con un ancho y un alto determinado.
setResizable(boolean opc)	Establece si el marco se puede redimensionar. Por omisión es true.
show()	Muestra el marco y sus componentes
hide()	Esconde el marco y sus componentes
dispose()	Descarga todos los recursos del sistema necesarios para mostrar el marco
pack()	Muestra la ventana y coloca sus componentes. Necesario se se realiza una redimensión de la ventana o se modifican sus componentes
setVisible(boolean opc)	Establece si el marco es visible. setVisible(true) es equivalente a show()

Otros contenedores de primer nivel

- Existen otros contenedores: `JDialog` y `JApplet`.
- `JAppet` se utiliza para la realización de applets (siguiente capítulo).
- `JDialog`.
  - Contenedor de primer nivel dependiente de una ventana principal.
  - Su constructor precisa indicar la ventana propietaria (por ejemplo, una referencia a un objeto JFrame)

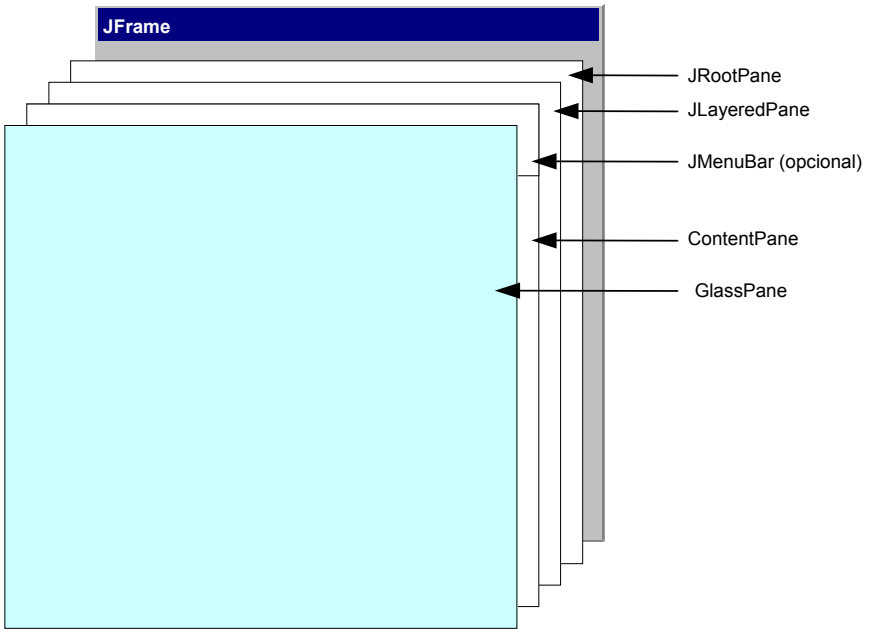
```
JDialog diag = new JDialog(MiMarco,"Diálogo de prueba");
```

  - Diálogos modales.

```
JDialog diag = new JDialog(MiMarco,"Diálogo de prueba",true);
```

Estructura de un JFrame

- Varios paneles dispuestos en capas



- JRootPane.
  - Sobre él residen los demás.
- JLayeredPane.
  - Eje Z.
- GlassPane.
  - Panel transparente que está por encima de los demás.
- JMenuBar.
- ContentPane.
  - En él se suelen situar los componentes.
  - Es sobre el que se trabaja habitualmente.

Añadir componentes

- `JFrame` es un contenedor donde colocar componentes.
- Los componentes se sitúan sobre un panel.
  - Puede ser un objeto de la clase `JPanel` o directamente sobre el ContentPane.
- Para obtener el panel de contenido se utiliza el método `getContentPane()`.

```
Container panelContenido = getContentPane();
```

- Es posible establecer un componente como panel de contenido con el método `setContentPane()`.

```
setContentPane(new JLabel("Etiqueta de prueba"));
```

- Pero normalmente los componentes se añaden con el método `add()`.

```
panelContenido.add(new JLabel("Etiqueta de prueba"));
```

Gestores de posicionamiento

- `add()` en un objeto `JFrame` añade el componente al final del panel de contenido, sobreescribiendo los demás.
- Para añadir más componentes se utiliza un gestor de posicionamiento mediante la interfaz `LayoutManager` del paquete `java.awt`.
  - `FlowLayout`.
  - `BorderLayout`.
  - `GridLayout`.
  - `BoxLayout`.
  - `GridBagLayout`.
- El posicionamiento se realiza de forma relativa, dependiendo del tamaño del marco y de los componentes.
- Para establecer el gestor de posicionamiento en un contenedor se utiliza el método `setLayout(LayoutManager mgr)` donde `mgr` es un objeto de alguna de las subclases anteriores.

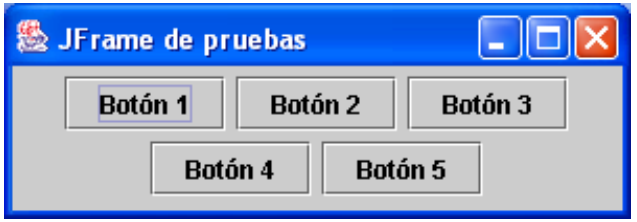
```
getContentPane().setLayout(new FlowLayout())
```

FlowLayout (I)

- Los componentes "fluyen" de izquierda a derecha, dejando, por omisión, un espacio vertical y horizontal de 5 pixels entre sus componentes.
- Constructores

FlowLayout(int alineación)	Modifica la disposición de los componentes. Los valores de alineación pueden ser FlowLayout.CENTER, FlowLayout.RIGHT y FlowLayout.LEFT
FlowLayout(int alineación, int seph, int sepv)	Permite indicar la separación horizontal y vertical en pixels

FlowLayout(II)




```
PruebasFrame(){
    setTitle("JFrame de pruebas");
    //Para un FlowLayout
    getContentPane().setLayout(new FlowLayout());
    getContentPane().add(new JButton("Botón 1"));
    getContentPane().add(new JButton("Botón 2"));
    getContentPane().add(new JButton("Botón 3"));
    getContentPane().add(new JButton("Botón 4"));
    getContentPane().add(new JButton("Botón 5"));
    setSize(300,200);
}
```

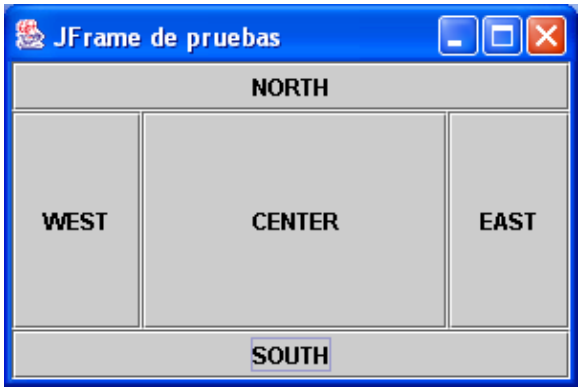
BorderLayout (I)

- Divide el contenedor en 5 zonas (`NORTH`, `SOUTH`, `EAST`, `WEST` y `CENTER`) donde se añaden los componentes.
  - Por omisión se colocan en el centro.
- Para añadir los componentes se utiliza el método

  `add(Component compt, int zona)`
  - `zona` puede tomar los valores `BorderLayout.CENTER`, `BorderLayout.NORTH`, `BorderLayout.SOUTH`, `BorderLayout.WEST` y `BorderLayout.EAST`.
- Constructores

BorderLayout()	Los componentes se colocan sin separación.
BorderLayout(int seph, int sepv)	Los componentes se colocoan cun una separación horizontal de seph pixel y vertical de sepv pixels

BorderLayout (II)



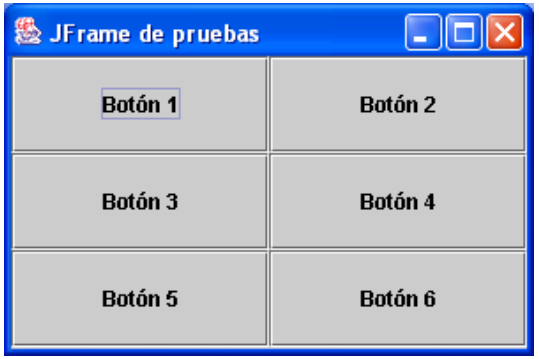

```
PruebasFrame(){
    setTitle("JFrame de pruebas");
    //Para un BorderLayout
    getContentPane().setLayout(new BorderLayout());
    getContentPane().add(new JButton("CENTER"), BorderLayout.CENTER);
    getContentPane().add(new JButton("NORTH"), BorderLayout.NORTH);
    getContentPane().add(new JButton("SOUTH"), BorderLayout.SOUTH);
    getContentPane().add(new JButton("EAST"), BorderLayout.EAST);
    getContentPane().add(new JButton("WEST"), BorderLayout.WEST);
    setSize(300,200);
}
```

GridLayout (I)

- Los componentes se colocan en una rejilla de celdas iguales.
  - Se colocan de arriba hacia abajo y de izquierda a derecha.
- Constructores.

GridLayout()	Coloca los componentes en una única fila y una única columna
GridLayout(int f, int c)	Coloca los componentes en una rejilla de f filas y c columnas.
GridLayout(int f, int c, int seph, int sepv)	Coloca los componentes en una rejilla de f filas y c columnas con una separación horizontal y vertical determinada

GridLayout (II)




```
PruebasFrame(){
    setTitle("JFrame de pruebas");
    //Para un GridLayout
    getContentPane().setLayout(new GridLayout(3,2,5,5));
    getContentPane().add(new JButton("Botón 1"));
    getContentPane().add(new JButton("Botón 2"));
    getContentPane().add(new JButton("Botón 3"));
    getContentPane().add(new JButton("Botón 4"));
    getContentPane().add(new JButton("Botón 5"));
    getContentPane().add(new JButton("Botón 6"));
    setSize(300,200);
}
```

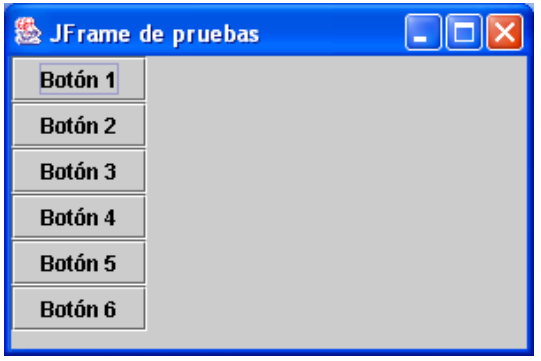
BoxLayout (I)

- Muestra los componentes en una única fila o columna.
- Su constructor necesita un argumento con el contenedor a utilizar y la orientación determinada por las constantes `BoxLayout.X_AXIS` o `BoxLayout.Y_AXIS`.

```
BoxLayout(Container destino, int orientación)
```

- Precisa la creación de un objeto de la clase `Container` para su utilización como contenedor.
  - Normalmente será un objeto de la clase `JPanel` (ver más adelante).

BoxLayout (II)




```
PruebasFrame(){
    setTitle("JFrame de pruebas");
    JPanel panel = new JPanel();
    panel.setLayout(new BoxLayout(panel,BoxLayout.Y_AXIS));
    panel.add(new JButton("Botón 1"));
    panel.add(new JButton("Botón 2"));
    panel.add(new JButton("Botón 3"));
    panel.add(new JButton("Botón 4"));
    panel.add(new JButton("Botón 5"));
    panel.add(new JButton("Botón 6"));
    setContentPane(panel);
    setSize(300,200);
}
```

BoxLayout (III)

- Existe un contenedor que tiene como gestor de posicionamiento un `BoxLayout`: la clase `Box`.
- Box permite añadir separaciones entre los componentes añadiendo "puntales" (Strut), zonas rígidas (RigidArea) y pegamento (Glue).
- Creación de un objeto Box.

```
//Crea una caja horizontal
Box cajah = Box.createHorizontalBox();
//Crea una caja vertical
Box cajav = Box.createVerticalBox();
```

BoxLayout (IV)

- Puntal (Strut): zona invisible de un alto o ancho fijo para separar dos componentes.

```
Box.createHorizontalStrut(int ancho)
Box.createVerticalStrut(int alto)
```

- Zona rígida (RigidArea): proporciona un área invisible de separación horizontal y vertical.
  - En una caja horizontal, la separación vertical afectará a todos sus componentes y viceversa.

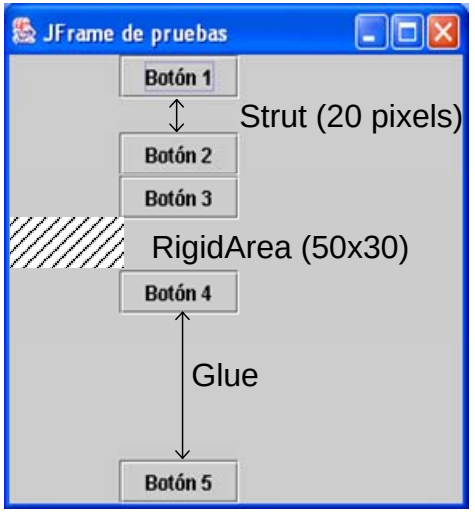
```
Box.createRigidArea(Dimension dim)
```

    - ✓ Precisa de la creación de un objeto `Dimension` (`new Dimension(alto,ancho)`).

- Glue: zona invisible que alinea los siguientes controles con el borde derecho o inferior.

```
Box.createGlue()
```

BoxLayout (V)



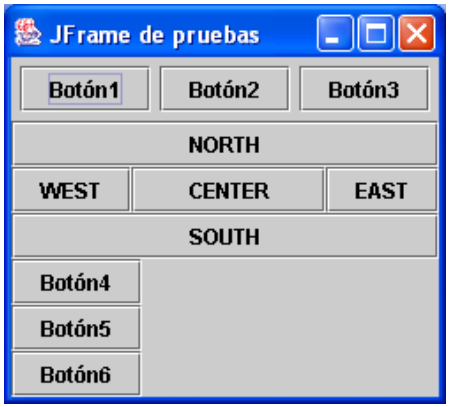

```
PruebasFrame(){
  setTitle("JFrame de pruebas");
  //Para un BoxLayout con un objeto Box
  Box caja = Box.createVerticalBox();
  caja.add(new JButton("Botón 1"));
  caja.add(Box.createVerticalStrut(20));
  caja.add(new JButton("Botón 2"));
  caja.add(new JButton("Botón 3"));
  caja.add(Box.createRigidArea(new Dimension(50,30)));
  caja.add(new JButton("Botón 4"));
  caja.add(Box.createGlue());
  caja.add(new JButton("Botón 5"));
  setContentPane(caja);
  setSize(300,200);
}
```

Combinar varios gestores (I)

- Cada contenedor sólo puede tener gestores de un tipo.
- Pero un contenedor de primer nivel puede tener otros contenedores anidados.
  - Se utiliza como contenedor para anidar un objeto `JPanel`.
  - Constructores de JPanel

JPanel()	Crea un JPanel con un gestor de posicionamiento de tipo FlowLayout.
JPanel(LayoutManager mnr)	Crea un JPanel con el gestor de posicionamiento indicado.

Combinar varios gestores (II)




```
PruebasFrame(){
  setTitle("JFrame de pruebas");
  JPanel panel1 = new JPanel(new FlowLayout());
  JPanel panel2 = new JPanel(new BorderLayout());
  JPanel panel3 = new JPanel();
  panel1.add(new JButton("Botón1"));
  panel1.add(new JButton("Botón2"));
  panel1.add(new JButton("Botón3"));
  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("Botón4"));
  panel3.add(new JButton("Botón5"));
  panel3.add(new JButton("Botón6"));
  getContentPane().setLayout(new BorderLayout());
  getContentPane().add(panel1, BorderLayout.NORTH);
  getContentPane().add(panel2, BorderLayout.CENTER);
  getContentPane().add(panel3, BorderLayout.SOUTH);
  pack();
}
```