



# Tema 13: Creación de interfaces de usuario

## ¿Qué aprenderás?

---

- Crear aplicaciones con interfaz gráfica: ventanas, botones, cajas de texto...
- Programar eventos a determinadas acciones sobre componentes gráficos.

## ¿Sabías que...?

---

- La primera interfaz gráfica de usuario (GUI) fue presentada con el ordenador personal XEROX Alto en el año 1973.
- La librería Swing nació en 1997 y se incluyó en la versión 1.2 de Java.
- Inicialmente, Java contaba con el kit AWT para implementar la interfaz gráfica. Swing surgió más adelante como respuesta a las deficiencias de AWT.



## 1. Introducción

---

En los capítulos anteriores hemos ido haciendo programas basados en la consola. Esto quiere decir que no usan interfaz gráfica de usuario (GUI). Pese a que los programas hechos en la consola son completamente operativos, la mayoría de aplicaciones tienen una interfaz gráfica. En este capítulo, veremos un kit de herramientas GUI que ofrece Java, llamado Swing.

Cabe destacar que Swing ofrece una amplia variedad de componentes, por lo que en este capítulo se tratará el tema de forma superficial. Será un punto de partida para entender Swing y saber utilizarlo, siendo capaces de crear aplicaciones con una interfaz gráfica completa y funcional.

## 2. Componentes

---

Botón

<b>JButton</b>	Botón
<b>JLabel</b>	Etiqueta de texto
<b>TextField</b>	Cuadro de texto
<b>JCheckBox</b>	Casilla de verificación
<b>JRadioButton</b>	Botón de opción
<b>JComboBox</b>	Lista desplegable

Todas las clases de componentes empiezan por la letra J. Para incorporar alguno de estos componentes en nuestro programa simplemente se debe instanciar un objeto de uno de estas clases. Su uso no difiere del resto de objetos ya vistos en Java. Más adelante veremos un ejemplo completo de creación de una aplicación que usará interfaz gráfica y se mostrará el código para crear diversos componentes y cómo editarlos a través de sus propiedades y métodos.

Hay un grupo de componentes especiales que sirven para agrupar otros componentes, y poder así organizarlos dentro de nuestra ventana: los contenedores.



## 2.1. Contenedores

Un contenedor es un tipo especial de componente cuyo propósito es agrupar un conjunto de componentes. Todas las GUI de Swing deben tener como mínimo un contenedor que almacenará el resto de componentes de nuestra interfaz gráfica. Sin un contenedor, el resto de componentes no se pueden mostrar.

Swing ofrece varios tipos de contenedores. Podemos clasificarlos en dos tipos: los de nivel superior (o pesados) y los de nivel inferior (o ligeros).

Los contenedores de nivel superior ocupan el primer lugar en la jerarquía, es decir, no se incluyen en ningún otro contenedor. Es más, en nuestra GUI siempre debemos empezar por un contenedor de nivel superior que incluirá el resto de componentes.

Entre los contenedores de nivel superior podemos destacar:

- **JFrame:** implementa una ventana. Sería la ventana principal de nuestra aplicación.
- **JDialog:** implementa una ventana de tipo diálogo. Serían las ventanas secundarias de nuestra aplicación, que se llamarían desde la ventana principal (JFrame).
- **JApplet:** implementa una zona dentro de un Applet donde poder incluir componentes de Swing.

El que más usaremos en este curso será JFrame. Podemos destacar varios métodos, de los muchos que tiene la clase JFrame, por el uso que les daremos:

<b>setDefaultCloseOperation(int)</b>	<p>Especifica la acción a realizar cuando cerramos la ventana (pulsando sobre el icono 'X'). Las opciones son:</p> <ul style="list-style-type: none"><li>• <b>DO_NOTHING_ON_CLOSE:</b> no hace nada.</li><li>• <b>HIDE_ON_CLOSE:</b> esconde el JFrame.</li><li>• <b>DISPOSE_ON_CLOSE:</b> borra el JFrame de memoria. La aplicación puede seguir en funcionamiento si tiene más operaciones ejecutándose.</li><li>• <b>EXIT_ON_CLOSE:</b> equivale a un <code>System.exit</code>. Es decir, cierra la aplicación entera.</li></ul>
<b>setVisible(boolean)</b>	<p>Hace que la ventana se muestre. Por defecto no lo hace (el booleano es false). Es un método heredado de la clase <code>Window</code>.</p>
<b>setSize(int, int)</b>	<p>Establece el tamaño de la ventana. El primer número especifica el ancho y el segundo el alto. Se puede usar el método <code>pack</code> para ajustar el tamaño al contenido de la ventana. Es un método heredado de la clase <code>Window</code>.</p>



### **setLocationRelativeTo(Component)**

Establece la posición de la ventana relativa al componente especificado. Si se pone “null”, la ventana se centra en la pantalla. Es un método heredado de la clase Window.

Los contenedores de nivel inferior, o contenedores ligeros, se usan para organizar grupos de componentes relacionados entre sí. Algunos ejemplos de contenedores ligeros son JPanel, JScrollPane o JRootPane.

A modo de resumen, para tener una idea más clara de cómo funcionan los diferentes tipos de contenedores y cómo se relacionan entre sí, podemos tener una aplicación con una ventana principal, JFrame. Dentro de este contenedor, podríamos tener un JRootPane, que sirve para gestionar otros paneles y además se encarga de gestionar la barra de menús opcional de nuestra aplicación. Luego podríamos tener otros contenedores ligeros para agrupar diferentes componentes de nuestra aplicación, que se podrían ir mostrando u ocultando según ciertos eventos, como podrían ser seleccionar una opción concreta de la barra de menús.

Para terminar con el concepto de contenedor, debemos conocer los administradores de diseño, que sirven para ubicar los componentes gráficos de nuestra aplicación dentro del contenedor.

En la siguiente tabla vemos los administradores de diseño que nos ofrece Swing.

<b>FlowLayout</b>	Los componentes se ubican de izquierda a derecha y de arriba abajo.
<b>BorderLayout</b>	Los componentes se ubican en el centro o los bordes del contenedor. Divide el contenedor en cinco partes (norte, sur, este, oeste y centro).
<b>GridLayout</b>	Los componentes se organizan en una cuadrícula (filas y columnas).
<b>GridBagLayout</b>	Los componentes se organizan en una cuadrícula flexible (un componente puede ocupar más de una fila o columna).
<b>BoxLayout</b>	Los componentes se organizan horizontal o verticalmente en un cuadro.
<b>SpringLayout</b>	Los componentes se organizan con respecto a una serie de restricciones.

No siempre tendremos que usar uno de los layouts anteriores en nuestra aplicación. Una práctica bastante habitual es la de no usar ningún layout, es decir, usar el layout NULL. Con esto tenemos completa libertad para ubicar los componentes en la posición que queramos y con el tamaño deseado.



## 2.2. Controladores de eventos

En todas las aplicaciones hay componentes que reaccionan a algún evento. Por ejemplo, cuando pulsamos un botón, cuando cerramos una ventana, cuando presionamos la tecla Intro sobre un campo de texto, etc.

Java ofrece una serie de controladores de eventos, llamados listeners, especializados en un evento concreto sobre un componente. En estos listeners podemos implementar las acciones que se llevarán a cabo después de dicho evento.

En la siguiente tabla se pueden ver algunos listeners y la acción a la que responden.

<b>ActionListener</b>	Hace referencia a la acción más típica sobre un componente. Por ejemplo, sobre un JButton será presionarlo, sobre un JTextField será pulsar Intro, sobre un JComboBox será seleccionar una opción, etc.
<b>FocusListener</b>	Ejecuta acciones cuando un componente obtiene o pierde el foco (colocarnos sobre el componente o irnos cuando éste estaba activo).
<b>KeyListener</b>	Responde a la acción de pulsar una tecla cuando un componente tiene el foco.
<b>ItemListener</b>	Hace referencia a la acción de seleccionar o deseleccionar una opción (en un JCheckBox, por ejemplo).
<b>MouseListener</b>	Responde al click sobre el ratón.
<b>MouseMotionListener</b>	Responde a acciones como arrastrar (drag) un elemento o pasar por encima.
<b>WindowListener</b>	Responde a acciones sobre una ventana como, por ejemplo, cerrarla.

Para incorporar listeners a nuestro programa, debemos crear primero el componente. Podemos añadir varios listeners a un mismo componente. Por ejemplo, podemos querer que un botón realice una acción al pulsarlo con el ratón, pero también queremos que reaccione al pulsar Intro cuando el botón tenga el foco.



## 2.3. Ejemplo de uso de Swing

Ahora que hemos visto qué componentes podemos usar en nuestras aplicaciones, y cómo programar eventos sobre éstos, sólo nos falta ver código de ejemplo. Vamos a crear una aplicación con una ventana que contendrá varios componentes. La apariencia de la aplicación será la siguiente:

Introduce tu nombre:

Edad: 20-24

Sexo: ☐ Hombre ☐ Mujer

Aficiones: ☐ Programar ☐ Jugar ☐ Leer

Aceptar

En el siguiente vídeo puedes ver el proceso de creación de la aplicación:





## 2.4. Window Builder en Eclipse

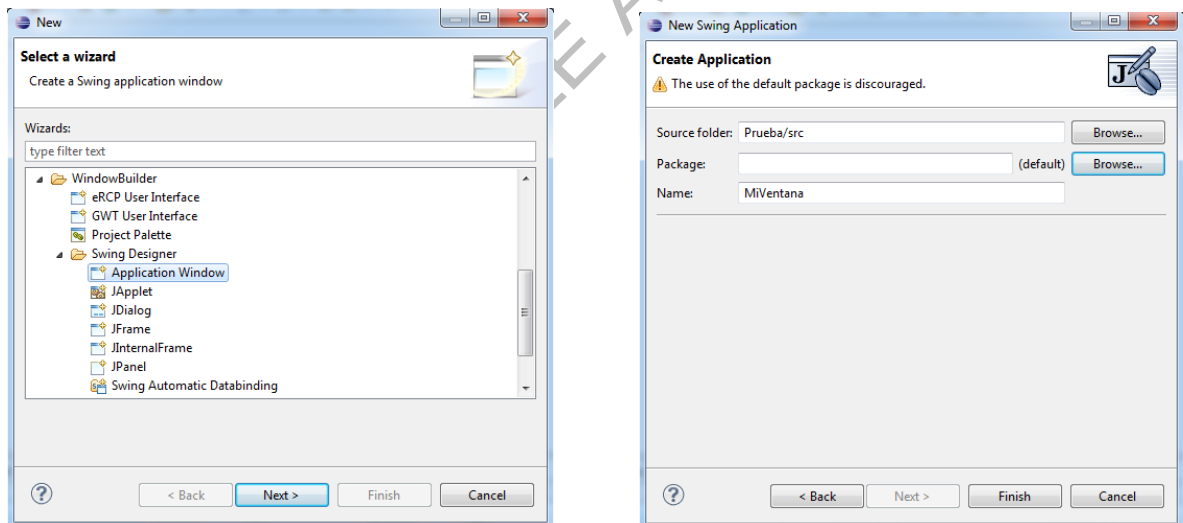
Window Builder es un plugin de Eclipse que nos permite desarrollar de forma rápida y cómoda la GUI (interfaz gráfica de usuario) de nuestras aplicaciones Java.

El proceso de instalación de Window Builder tiene dos posibilidades. Ambas las encontraremos en la página de descarga del plugin (ver apartado Recursos y enlaces).

Una posibilidad es descargarnos un ZIP que contendrá el plugin. Si abrimos el archivo veremos que tiene varias carpetas y archivos. Lo que nos interesan son las carpetas “features” y “plugins”. Descomprimiremos el contenido de ambas dentro de las carpetas con el mismo nombre en el directorio donde hagamos instalado Eclipse.

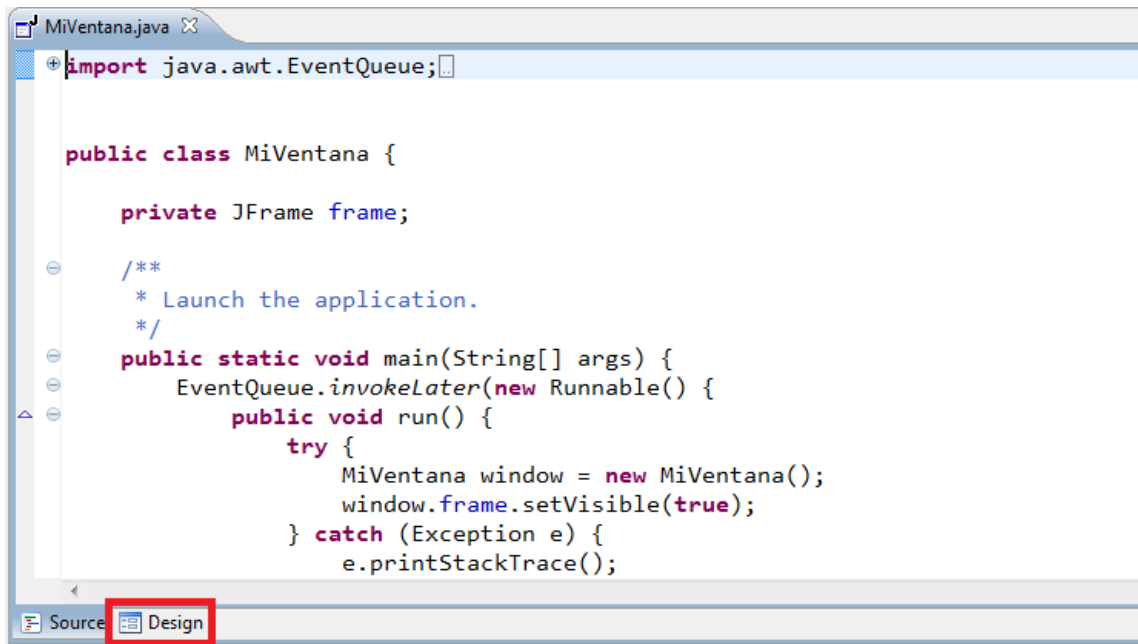
La otra forma de instalar Window Builder es desde el propio Eclipse, con su herramienta de instalación de software que encontrarás en el menú “Help > Install new software...”. Los pasos a seguir están explicados en la página de descarga de Window Builder.

Vamos a crear una sencilla aplicación Java con una ventana. Para ello ejecutamos Eclipse y creamos un proyecto Java al que llamaremos “Prueba”. A continuación, vamos al menú “File > New > Other...” y seleccionamos “WindowBuilder > Swing designer > Application Window”. Crearemos una interfaz gráfica para el proyecto “Prueba” creado anteriormente, tal y como se muestra en las siguientes imágenes:





Obtendremos un archivo de código autogenerated como el que se muestra a continuación. Pero para trabajar más cómodamente, usaremos la propiedad de WindowBuider “Design”:



```
import java.awt.EventQueue;

public class MiVentana {

    private JFrame frame;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    MiVentana window = new MiVentana();
                    window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

Vemos que de esta forma se abre un editor gráfico que nos permitirá retocar las propiedades de nuestras ventanas y añadir nuevos elementos (menús, cuadros de texto, botones, etc.). Cada vez que cambiemos algo desde esta interfaz, se generará automáticamente el código fuente asociado a las acciones que realicemos.





## Recursos y enlaces

---

- Descarga plugin WindowBuilder para Eclipse:  
<http://www.eclipse.org/windowbuilder/download.php>



- Documentación oficial sobre Swing:  
<https://docs.oracle.com/javase/tutorial/uiswing/index.html>



## Conceptos clave

---

- **Swing:** kit de herramientas GUI que ofrece Java.
- **Componente:** un componente es una clase que representa un elemento visual independiente: un botón, un campo de texto, etc.
- **Contenedor:** un contenedor es un tipo especial de componente cuyo propósito es agrupar un conjunto de componentes.
- **Controlador de eventos:** es una clase predefinida de Java que se encarga de controlar y administrar que interacciones que realice el usuario con un elemento de la interfaz gráfica.



## Test de autoevaluación

---

¿Cuál de los siguientes NO es uno de los contenedores superiores de la librería Swing?

- a) JPanel
- b) JFrame
- c) JDialog
- d) JApplet

¿Qué objeto de la librería Swing permite crear listas desplegables?

- e) JList
- f) JSelect
- g) JComboBox
- h) JScrollBar

Si quiero que un JButton realice una acción cuando el usuario presiona el botón Intro, ¿qué tipo de listener tengo que programar?

- a) ActionListener
- b) ItemListener
- c) FocusListener
- d) KeyListener



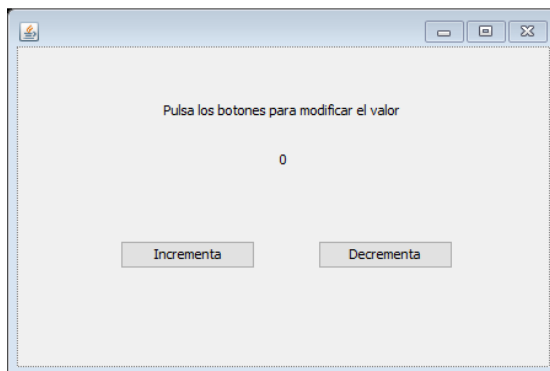
## Ponlo en práctica

---

### Actividad 1

---

Crea una aplicación que conste de una ventana con el siguiente contenido:



El funcionamiento de la aplicación será el siguiente: cuando pulses el botón “Incrementa” se debe sumar uno al valor que contiene la etiqueta con el valor 0 en la imagen. Al pulsar el botón “Decrementa”, se debe restar uno al valor de la etiqueta.

Hay que hacer que, al pulsar sobre el botón de cerrar de la ventana, el programa se cierre.

### Actividad 2

---

Crea una aplicación con una ventana en la que el usuario introduzca su nombre en una caja de texto y seleccione su edad de un desplegable con varias opciones.

Si el usuario ha introducido su nombre, y ha seleccionado una edad igual o mayor a 18, al darle a un botón se cerrará la ventana actual y se abrirá otra en la que se mostrará un mensaje de bienvenida que contendrá el nombre introducido por el usuario en la ventana anterior.

Si la edad seleccionada es menor a 18, en la segunda ventana se mostrará el mensaje “Eres menor de edad”.

Si el usuario dejara vacío el campo de texto, se debería mostrar un mensaje de error al pulsar el botón para ir a la siguiente ventana.



## SOLUCIONARIOS

### Test de autoevaluación

---

¿Cuál de los siguientes NO es uno de los contenedores superiores de la librería Swing?

- e) JPanel**
- f) JFrame
- g) JDialog
- h) JApplet

¿Qué objeto de la librería Swing permite crear listas desplegables?

- i) JList
- j) JSelect
- k) JComboBox**
- l) JScrollBar

Si quiero que un JButton realice una acción cuando el usuario presiona el botón Intro, ¿qué tipo de listener tengo que programar?

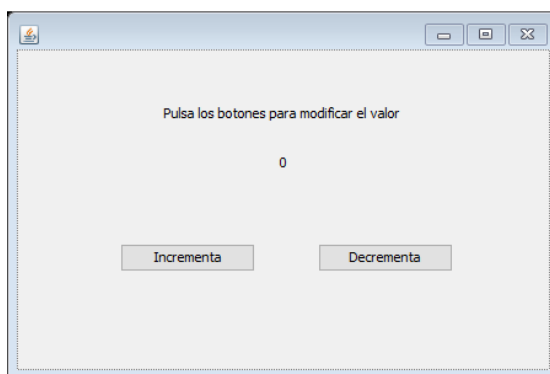
- m) ActionListener
- n) ItemListener
- o) FocusListener
- p) KeyListener**



## Ponlo en práctica

### Actividad 1

Crea una aplicación que conste de una ventana con el siguiente contenido:



El funcionamiento de la aplicación será el siguiente: cuando pulses el botón “Incrementa” se debe sumar uno al valor que contiene la etiqueta con el valor 0 en la imagen. Al pulsar el botón “Decrementa”, se debe restar uno al valor de la etiqueta.

Hay que hacer que, al pulsar sobre el botón de cerrar de la ventana, el programa se cierre.

**El solucionario está disponible en la versión interactiva del aula.**

### Actividad 2

Crea una aplicación con una ventana en la que el usuario introduzca su nombre en una caja de texto y seleccione su edad de un desplegable con varias opciones.

Si el usuario ha introducido su nombre, y ha seleccionado una edad igual o mayor a 18, al darle a un botón se cerrará la ventana actual y se abrirá otra en la que se mostrará un mensaje de bienvenida que contendrá el nombre introducido por el usuario en la ventana anterior.

Si la edad seleccionada es menor a 18, en la segunda ventana se mostrará el mensaje “Eres menor de edad”.

Si el usuario dejara vacío el campo de texto, se debería mostrar un mensaje de error al pulsar el botón para ir a la siguiente ventana.

**El solucionario está disponible en la versión interactiva del aula.**