Interfaces gráficas de usuario

La mayor parte de las aplicaciones que se utilizan hoy en día incluyen interfaces de usuario más sofisticadas que las de las aplicaciones que se ejecutan en modo consola.

Java incluye, como parte de su biblioteca de clases estándar, un conjunto de componentes para crear interfaces gráficas de usuario.

Siguiendo la filosofía de su lema ("write once, run anywhere"), estos componentes permiten crear interfaces portables.

AWT y Swing

Los componentes que se utilizan en Java para crear interfaces gráficas de usuario se agrupan en dos paquetes (y sus correspondientes subpaquetes):

- `java.awt` (Abstract Window Toolkit)

  Los componentes AWT dependen de las facilidades gráficas ofrecidas por cada sistema operativo: los programas escritos con AWT tendrán un "look and feel" distinto en Windows y en UNIX.

- `javax.swing`

  SWING es 100% Java y, por tanto, completamente independiente de la plataforma: los componentes gráficos se pintan en tiempo de ejecución (por lo que las aplicaciones SWING suelen ser algo más lentas que las AWT).

En la práctica, las aplicaciones Java con interfaces gráficas de usuario suelen mezclar AWT y SWING (porque AWT se encarga de toda la gestión de eventos y SWING ofrece componentes más sofisticados).

Frames

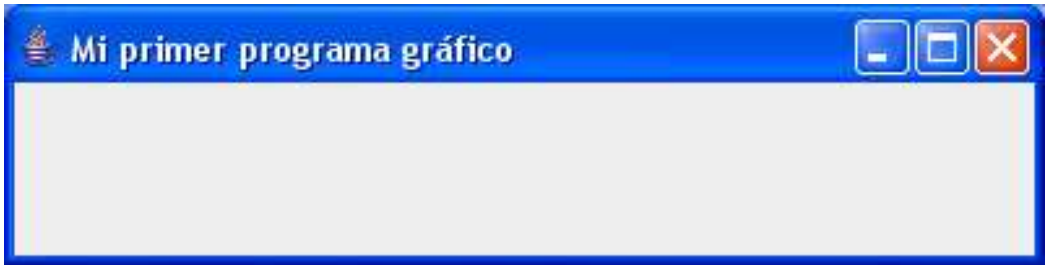
Los *frames* (marcos) son ventanas en las que se pueden colocar otros controles de los que se utilizan para crear las interfaces gráficas de usuario (botones, etiquetas, cajas de texto, listas desplegables, menús, etcétera).

AWT	SWING
`java.awt.Frame`	`javax.swing.JFrame`

```
import javax.swing.*;

class PrimerFrame extends JFrame
{
  public PrimerFrame()
  {
    setTitle("Mi primer programa gráfico");
    setSize(400,100);
  }
}

public class FrameTest
{
  public static void main(String[] args)
  {
    JFrame frame = new PrimerFrame();
    frame.setVisible(true);
  }
}
```



NOTA: Por defecto, un frame es de tamaño 0 x 0 píxeles.

El programa anterior tiene un problema:

Cuando se cierra la ventana,
dejamos de verla pero el programa no finaliza su ejecución.

Para que el programa funcione correctamente, hemos de interceptar el ***evento*** que se produce cuando cerramos la ventana y hacer que el programa termine su ejecución en ese momento.

Para ello, hay que definir un ***manejador de eventos*** en el cuál especificamos qué es lo que queremos que haga el programa.

WindowListener & WindowAdapter

En este caso, hemos de definir un `WindowListener` (clase encargada de escuchar eventos asociados a ventanas) y asociárselo al Júrame de nuestro ejemplo.

La interfaz `java.awt.event.indowListener` incluye 7 métodos que corresponden a las distintas acciones que podemos realizar sobre una ventana:

- traerla a primer plano (`windowActivated`),
- dejarla en segundo plano (`windowDeactivated`),
- minimizarla (`windowIconified`),
- restaurarla (`windowDeiconified`),
- abrirla (`windowOpened`)
- o cerrarla (`windowClosed` y `windowClosing`)

Para que no tengamos que definir los 7 métodos, AWT incluye una clase auxiliar, llamada `WindowAdapter`, de la que sólo tenemos que redefinir aquéllos métodos que verdaderamente nos interesen.

```
import javax.swing.*;
import java.awt.event.*;

class PrimerFrame extends JFrame
{
  public PrimerFrame()
  {
    setTitle("Mi primer programa gráfico");
    setSize(400,100);
    addWindowListener(new PrimerWindowListener());
  }
}

class PrimerWindowListener extends WindowAdapter
{
  public void windowClosing(WindowEvent e)
  {
    System.exit(0);
  }
}


public class FrameTestExit
{
  public static void main(String[] args)
  {
    JFrame frame = new PrimerFrame();
    frame.setVisible(true);
  }
}
```