Swing/JFC

- JFC es la abreviatura de Java Foundation Classes, que comprende un grupo de características para ayudar a construir interfaces gráficos de usuario (GUIs).
- Los componentes Swing
  - Incluye todo desde botones hasta splitpanes y tablas.

Swing/JFC

- Soporte de Aspecto y Comportamiento Conectable
  - Le ofrece a cualquier componente Swing una amplia selección de aspectos y comportamientos. Por ejemplo, el mismo programa puede usar el Aspecto y Comportamiento Java o el Aspecto y Comportamiento Windows.

Swing/JFC

- API de Accesibilidad
  - Permite tecnologías asistivas como lectores de pantalla y display Braille para obtener información desde la interfaz de usuario.
- Java 2D API
  - Permite a los desarrolladores incorporar fácilmente gráficos 2D de alta calidad, texto, e imágenes en aplicaciones y applets Java.

Swing/JFC

- Soporte de Drag and Drop
  - Proporciona la habilidad de arrastrar y soltar entre aplicaciones Java y aplicaciones nativas.
- Para utilizar JFC se importan las bibliotecas swing.
  - Import javax.swing.*;

Swing/JFC

- Se puede simular las interfaces gráficas en Java como un árbol
  - La raíz de este árbol están las clases contenedoras, son tres :
    - JFrame.
    - JDialog.
    - JApplet.

Swing/JFC

- El tallo de este árbol, lo componen las clases llamadas contenedoras intermedio, como:
  - JPanel.
  - JScrollPane.
  - JTabbedPane.

Swing/JFC

- Las hojas del árbol están compuestas por los componentes atómicos:
  - JButton.
  - JLabel.
  - JComboBox.
  - JTextField.
  - JTable.
  - ....

Swing/JFC

- Para crear un GUI, se deben cumplir los siguientes pasos:
  - Crear un Contenedor.
  - Asignar un contenedor intermedio a él.
  - Asignar uno o varios componentes atómicos al contenedor intermedio.

Swing/JFC

- Ejemplo:

```
import javax.swing.*;
public class VentanaSwing extends JFrame {
  public VentanaSwing(){
    super("Primera Ventana");
    setLocation(100,100);
    setSize(200,100);
    show();
  }
  public static void main(String[] arg) {
    new VentanaSwing();
    System.out.println("He creado la ventana");
  }
}
```

Swing/JFC

- Una aplicación que utiliza Swing, indirectamente está utilizando Threads.
- Una aplicación gráfica siempre debe estar a la espera de algún evento, apretar una tecla y/o mover el mouse (también presionarlo).
- Existen procesos que están permanentemente atentos a la espera de estos eventos.

Swing/JFC

- Estos escuchadores de eventos funcionan en paralelo con la aplicación swing.
- Por lo tanto la manera más seguro de ejecutar una aplicación swing es declarándola como un Thread.

Swing/JFC

- Forma segura de correr una aplicación swing

```
- public static void main(String[] args) {
-
  javax.swing.SwingUtilities.invokeLater(new
  Runnable() {
-          public void run() {
-             // Invocar la aplicación
-          }
-     });
- }
```

Swing/JFC

- Crear un botón
  - `JButton button = new JButton("Soy un boton");`
  - `button.setMnemonic(KeyEvent.VK_B);`
  - `button.addActionListener(new ActionListener()`
  - `{`
    - `public void actionPerformed(ActionEvent e)`
    - `{`
      - `numClicks++;`
      - `label.setText(labelPrefix + numClicks);`
    - `}`
  - `});`

Swing/JFC

- Manejo de eventos
  - ActionListener
    - El usuario pulsa un botón, presiona Return mientras teclea en un campo de texto, o elige un ítem de menú.
  - WindowListener
    - El usuario elige un frame (ventana principal).
  - MouseListener
    - El usuario pulsa un botón del ratón mientras el cursor está sobre un componente.
  - MouseMotionListener
    - El usuario mueve el cursor sobre un componente.

Swing/JFC

- ComponentListener
  - El componente se hace visible.
  - FocusListener
  - El componente obtiene obtiene el foco del teclado.
  - ListSelectionListener
  - Cambia la tabla o la selección de una lista.
- Para manejar los eventos es necesario importar las bibliotecas de java.awt.*;

Swing/JFC

- Crear una etiqueta
  - `JLabel label = new Jlabel("Una etiqueta");`
- Crear TextField
  - `JTextField texto = new JtextField("Texto");`
  - `texto.setToolTipText("Escriba un texto");`
  - `texto.addFocusListener(new java.awt.event.FocusAdapter() {`
  - `  public void focusGained(FocusEvent e) {`
  - `   texto_focusGained(e);`
  - `  }`
  - `  public void focusLost(FocusEvent e) {`
  - `   texto_focusLost(e);`
  - `  }`
  - `});`

Swing/JFC

- Crear ComboBox
  - `JComboBox comboBox = new JcomboBox(arry);`
  - `comboBox.setSelectedIndex(2);`
  - `comboBox.addActionListener(new ActionListener()`
    `{`
  - `public void actionPerformed(ActionEvent e) {`
  - `comboBox_actionPerformed(e,pane);`
  - `}`
  - `})`

Swing/JFC

- Ejemplo

Swing/JFC

- Ejercitar swing con los componentes presentados usando la informacion disponible en la API de JAVA.

  - ver: Swing*.java

JAR

- El formato de archivos JAR permite empaquetar varios archivos en un sólo archivo. Típicamente un archivo JAR contendrá los archivos de clases y los recursos auxiliares asociados con los applets y aplicaciones. Estos recursos auxiliares podrían incluir, por ejemplo, archivos de imagen y sonido que sean utilizados por una aplicacion.

JAR

- El formato de archivos JAR proporciona muchos beneficios:
  - Seguridad: Puede firmar digitalmente el contenido de un archivo JAR. Los usuarios que reconozcan su firma pueden permitir a su software privilegios de seguridad que de otro modo no tendría.

JAR

- Disminuir el tiempo de descarga: Si sus applets están empaquetados en un archivo JAR, los archivos de clases y los recursos asociados pueden ser descargados por el navegador en una sola transacción HTTP sin necesidad de abrir una nueva conexión para cada fichero.

JAR

- Compresión: El formato JAR permite comprimir los archivos para ahorrar espacio.
- Portabilidad: El mecanismos para manejar los ficheros JAR son una parte estándar del corazón del API de la plataforma Java.

JAR

- Empaquetado sellado : Los paquetes almacenados en archivos JAR pueden ser sellados opcionalmente para que el paquete puede reforzar su consistencia. El sellado de un paquete dentro de un archivo JAR significa que todas las clases definidas en ese paquete deben encontrarse dentro del mismo archivo JAR.

JAR

- Empaquetado versionado : Un archivo JAR puede contener datos sobre los archivos que contiene, como información sobre el vendedor o la versión.

JAR

- Uso:
  - Para crear un fichero JAR
    - `jar cf jar-file input-file(s)`
  - Para ver el contenido de un fichero JAR
    - `jar tf jar-file`
  - Para extraer el contenido de un fichero JAR
    - `jar xf jar-file`
  - Para ejecutar una aplicación empaquetada en un fichero JAR
    - `java -cp app.jar MainClass`
    - `java -jar app.jar`

JAR

- Firmar archivos JAR
  - Se utiliza la firma de archivos JAR para autentificar al proveedor del software.
  - Si una aplicación necesita ejecutar acciones sensibles al sistema, la plataforma impide dichas acciones a menos que el archivos JAR esté firmado y verificado por una fuente confiable.