

UT 1 - Confección de interfaces de usuario

INTERFACES DE USUARIO

INTERFACES DE USUARIO

1. **Elaboración de interfaces de usuario.**
2. **Componentes y bibliotecas de componentes.**
3. **Herramientas para la elaboración de interfaces.**
4. **Contenedores.**
5. **Componentes de la interfaz.**
6. **Asociación de acciones a eventos.**
7. **Diálogos modales y no modales.**
8. **Edición de código generado por herramientas de diseño.**
9. **Clases, propiedades, métodos.**
10. **Eventos, escuchadores.**

1.- Elaboración de interfaces de usuario.

Que son las interfaces de usuario?

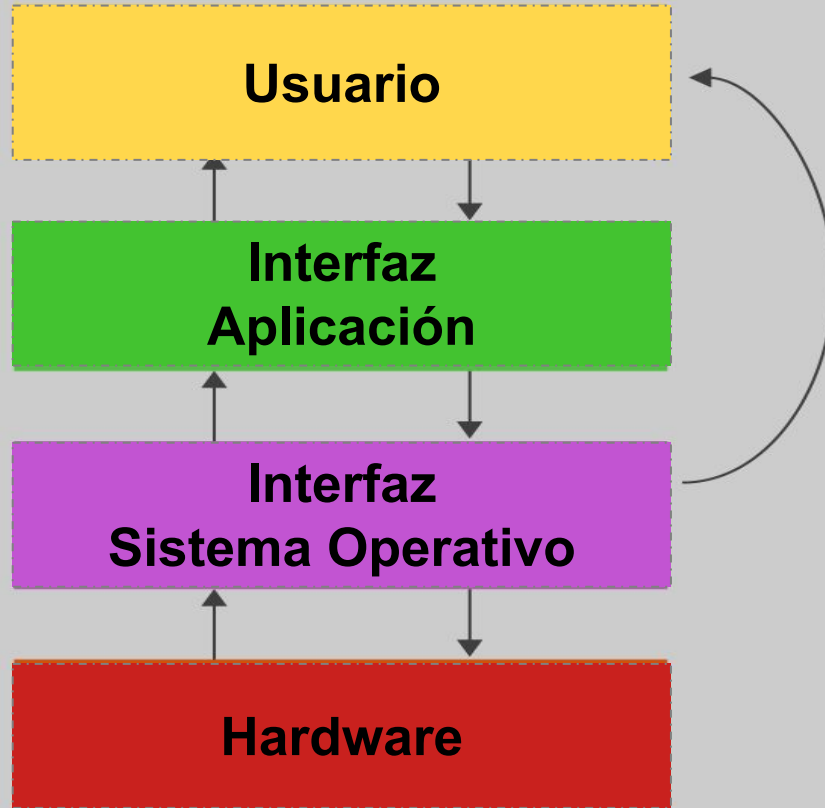
Es un conjunto de elementos (que pueden pertenecer al SW o al HW) que ofrecen una **información al usuario**, y permiten, a demás, la **interacción entre el usuario y el ordenador**, por medio de un dispositivo periférico o un enlace de comunicaciones.

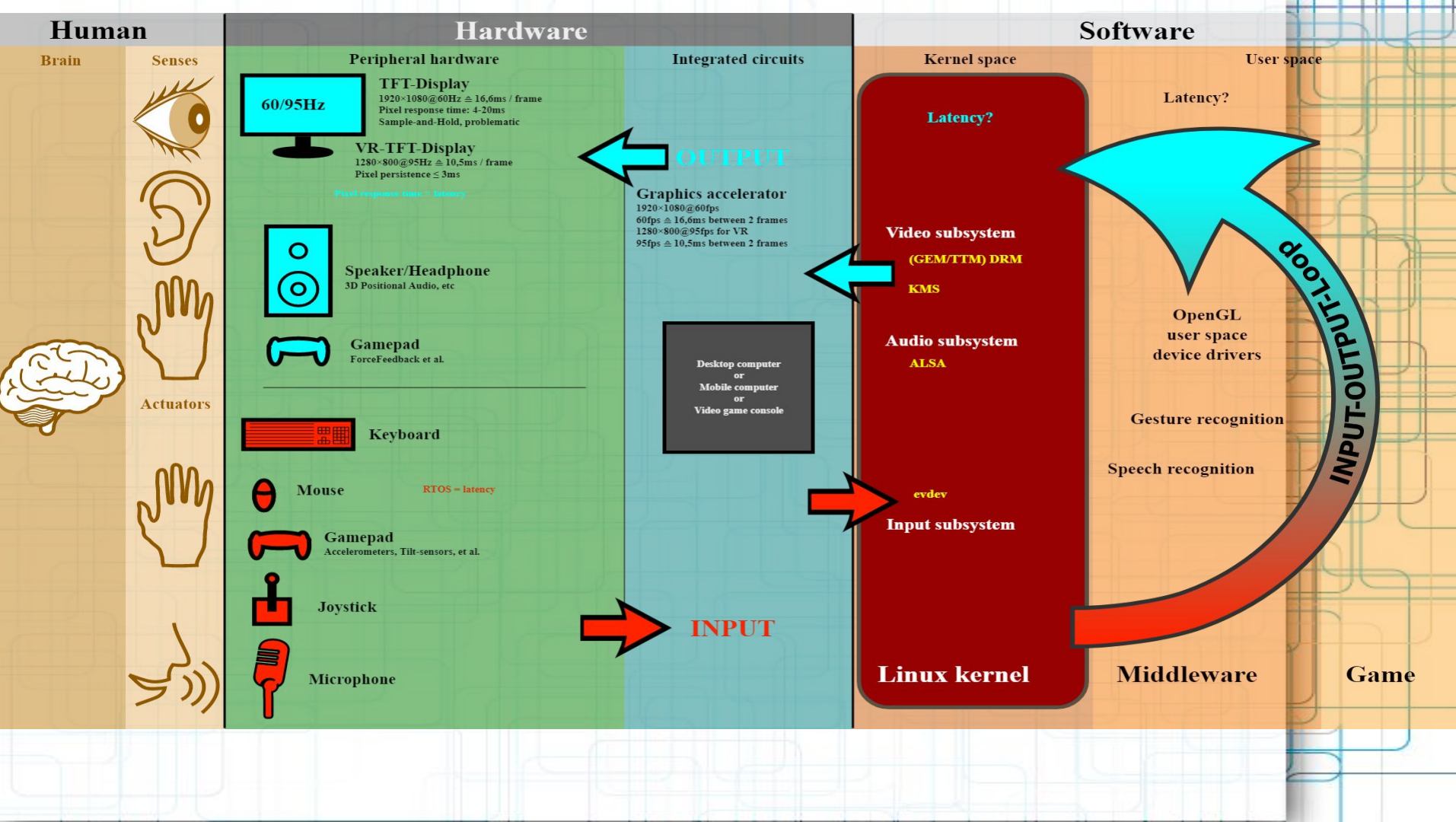
1.- Elaboración de interfaces de usuario.

Donde encontramos las interfaces?

Las interfaces forman parte de las aplicaciones. Son la parte de las aplicaciones con la que se relacionan los usuarios.

Una aplicación informática tendrá varias interfaces de usuario.





1.- Elaboración de interfaces de usuario.

Que es una aplicación?

Es un tipo de programa informático diseñado como herramienta para permitir a un usuario realizar uno o varios tipos de trabajos.

Esto los diferencian de otros tipos de programas como los sistemas operativos, las utilidades (que realizan tareas de mantenimiento o de uso general), y los lenguajes de programación (con los que se crean los programas informáticos).

1.- Elaboración de interfaces de usuario.

TIPOS DE INTERFACES DE USUARIO

Atendiendo a la forma que el usuario interacciona con la interfaz distinguimos 4 tipos de interfaces

a) Interfaces de _____ o _____

b) Interfaces _____ (_____)

c) Interfaces _____

d) Interfaces _____

TIPOS DE INTERFACES DE USUARIO

a) Interfaces de líneas de instrucciones o CLI:

El usuario introduce una orden que interpretará la interfaz, lo que le obliga a memorizar una serie de mandatos que le dan acceso a las funciones deseadas.

b) Interfaces gráficas de usuario (GUI): Son aquellas que utilizan elementos gráficos, como pueden ser menús, ventanas o diálogos, además del uso de otros recursos del sistema informático (periféricos como el teclado, el ratón o el sonido) para permitir al usuario interactuar con el ordenador de manera muy sencilla e intuitiva

TIPOS DE INTERFACES DE USUARIO

c) Interfaces desarrolladas para pantallas táctiles: Se deben adaptar a las necesidades de un dispositivo de entrada y de salida al mismo tiempo, que será una pantalla táctil, y a sus características específicas

d) Interfaces desarrolladas para dispositivos móviles:
Hoy en día coinciden este tipo de interfaces con las características descritas en c) además de condicionarnos el tamaño de la pantalla.

SIGLAS:

— — —

TIPOS DE INTERFACES DE USUARIO

c) Interfaces desarrolladas para pantallas táctiles: Se deben adaptar a las necesidades de un dispositivo de entrada y de salida al mismo tiempo, que será una pantalla táctil, y a sus características específicas

d) Interfaces desarrolladas para dispositivos móviles:
Hoy en día coinciden este tipo de interfaces con las características descritas en c) además de condicionarnos el tamaño de la pantalla.

Ejercicio: En grupos de 2 explica qué elementos tienen para la interacción con el usuario vuestros móviles

Atendiendo a cómo se ha desarrollado la interfaz y los elementos con la que se ha construido:

- a) **Interfaces de HW:** Permiten la interacción entre el usuario y el sistema informático mediante elementos de hardware. Ejemplos: pulsadores, lectores de barras, ...
- b) **Interfaces de SW:** Son las desarrolladas para ordenador.
- c) **Interfaces de HW-SW:** Son interfaces que compartirán los dos tipos explicados anteriormente. Ejemplo: un lector de código de barras.

Ejercicio: En grupos de 2 busca más ejemplos de interfaces de los apartados a) y c)

Atendiendo a la arquitectura de las aplicaciones y su manera de trabajar

a) Aplicaciones locales: interfaces Winforms: Se desarrollan para trabajar en una única máquina un único usuario a la vez.

b) Aplicaciones cliente-servidor: Este tipo de aplicaciones están basadas en el trabajo en más de una máquina, en la que una hará de servidor y el resto harán de clientes que acceden remotamente al servidor para acceder a los datos o en las funcionalidades. El tipo de interfaces que se utilizan habitualmente para este tipo de aplicaciones también serán las Winforms, aunque también se podrán utilizar interfaces de tipo Web.

c) Aplicaciones web: Son un tipo específico de las aplicaciones cliente / servidor.

Este tipo de aplicaciones permite acceder a datos y funcionalidad por medio de las redes telemáticas, y el acceso más habitual es lo que se hace por medio de un navegador de Internet, que accede a un servidor, donde se localizarán los datos y la implementación de las funcionalidades. Son interfaces que compartirán los dos tipos explicados anteriormente.

CARACTERÍSTICAS DE UNA INTERFAZ

a) Natural: el nuevo sistema automatizado debe tender a ser el más similar al antiguo.

b) Fácil de aprender y uso: para disfrutar de esta característica, la interfaz debe incorporar:

- Administración de perfiles de usuario.
- Mecanismos de realimentación que proporcione al usuario información sobre la ejecución actual del trabajo.
- Mecanismos de prevención de desastres.

c) Consistente: la interfaz debe mantener uniformidad en cuanto a estilo, vocabulario, etc.

d) Accesible e intuitiva

e) Legible

f) Autónoma: un usuario no debe necesitar más información o ayuda que la que una interfaz le ofrece.

g) Internacionalizada: utilizable por diferentes culturas e idiomas.

h) Optimizada a la ley de Fitts: esta ley, en ergonomía, moldeará el movimiento humano, haciendo una estimación del tiempo que puede necesitar un ser humano para mover un puntero desde una zona de la pantalla hasta otra teniendo en cuenta variables como los objetivos, la distancia hasta alcanzarlos y el tamaño que tendrán.

2.- Componentes y bibliotecas

ELEMENTOS BÁSICOS INTERFACES GUI

-
-
-
-
-
-
-

Profesión

Edad

Entre 18 y 30 años ▼

Nº Hermanos

Sexo

☐

HOMBRE

☐

MUJER

☐ ¿Práctica algún deporte ?

¿Cuál?

Fútbol

Tenis

Tenis de Mesa

Baloncesto

Marque de 1 a 10 su grado de afición a:

Compras

1 2 3 4 5 6 7 8 9 10

Ver la televisión

1 2 3 4 5 6 7 8 9 10

Ir al cine

1 2 3 4 5 6 7 8 9 10

ACEPTAR

CANCELAR

ES GUI

2.- Componentes y bibliotecas

ELEMENTOS BÁSICOS INTERFACES GUI

Etiquetas: Permiten situar un texto en la interfaz.

Campos de texto: cuadros de una sola línea en los que podemos escribir algún dato.

Áreas de texto: cuadros de varias líneas en los que podemos escribir párrafos.

Botones: áreas rectangulares que se pueden pulsar para llevar a cabo alguna acción.

Botones de radio: botones circulares que se presentan agrupados para realizar una selección de un único elemento.

2.- Componentes y bibliotecas

ELEMENTOS BÁSICOS INTERFACES GUI

Cuadros de verificación: botones en forma de rectángulo. Usados para marcar una opción. Podemos seleccionar varios.

Imágenes: se usan para añadir información gráfica a la interfaz.

Password: es un cuadro de texto en el que los caracteres aparecen ocultos.

Listas: conjunto de datos que se presentan en un cuadro entre los que es posible elegir uno o varios.

Listas desplegables: combinación de cuadro de texto y lista, permites escribir un dato o seleccionarlo de la lista que aparece oculta y puede ser desplegada.

2.- Componentes y bibliotecas

Ejercicio I: Abre la calculadora y describe cuáles de los elementos explicados en Elementos básicos interfaces GUI encuentras

2.- Componentes y bibliotecas

Ejercicio II: Diferencias entre

Diseño UI y Diseño UX

2.- Componentes y bibliotecas de componentes.

Los componentes se suelen presentar agrupados en **bibliotecas** con la posibilidad de que el usuario pueda generar sus propios componentes y añadirlos o crear sus propias bibliotecas.

Estas bibliotecas se componen de un conjunto de clases que se pueden incluir en proyectos software para crear interfaces gráficas.

2.- Componentes y bibliotecas de componentes.

JAVA Foundation Classes (JFC):

- **AWT**
- **Swing**

Además, existen bibliotecas para desarrollo grafico en 2D y 3D y para realizar tareas de arrastrar y soltar (drag and drop).

Bibliotecas MSDN de Microsoft (C#, ASP, ...):

.NET framework: Para el desarrollo de interfaces gráficas la biblioteca ADO.NET, ASP.NET, formularios Windows Forms y la WPF (Windows Presentation Foundation).

2.- Componentes y bibliotecas de componentes.

Bibliotecas basadas en XML:

También existen bibliotecas implementadas en lenguajes intermedios basados en tecnologías XML.

Otras API (Application Programming Interface, Interfaz de programación)

DirectX: => Direct3D, Direct Graphics, Direct sound, Direct Input, DirectPlay, DirectShow, DirectMusic, DirectSetuo y DirectCompute.)

GTK (GIMP tOOL KIT): ntorno gráfico de GNOME. Se puede utilizar en lenguajes C, C++, C#, Java, Python, Ruby.

QT: utilizada por el entorno gráfico KDE. Utiliza lenguaje de programación C++.

3.- Herramientas para la elaboración de interfaces.

- . Microsoft Visual Studio**
- . NetBeans**
- . Eclipse**
- . JDeveloper**
- . Aptana Studio**
- . Mono developer**
- . Dreamweaver**
- . Komodo Edit**

Datos Generales

Datos Opcionales

Datos del Usuario

Nombre: Liana Nataly

Apellido P: Arcila

Apellido M: Díaz

Sexo: ☐ Masculino

☒ Femenino

Nacionalidad: Perú

Datos del Perfil



Fecha Nacimiento:

23/02/1993



☒ Desea Mostrar estos datos

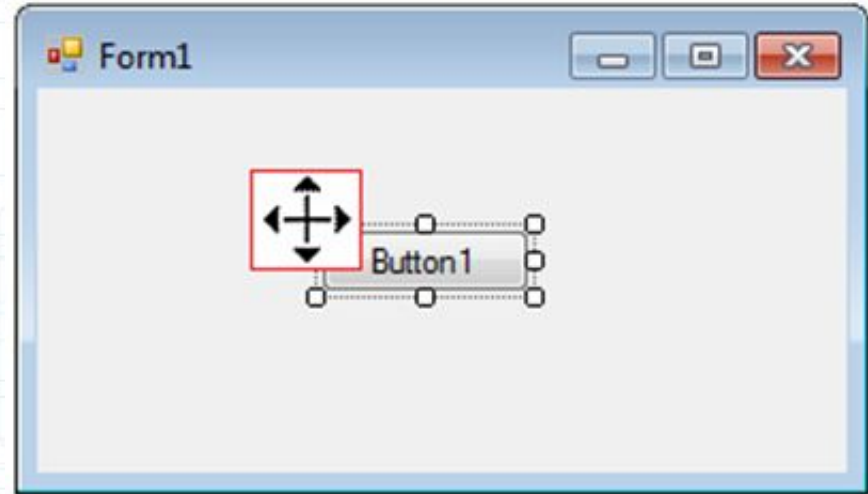
cción con

y

+

4.- Contenedores

El **formulario** es la base para crear una aplicación de escritorio. Es una ventana que dispone de tres botones para minimizarse, maximizarse o cerrarse, una barra de título y está delimitado por unos bordes. Sobre él se añadirán controles o componentes,...



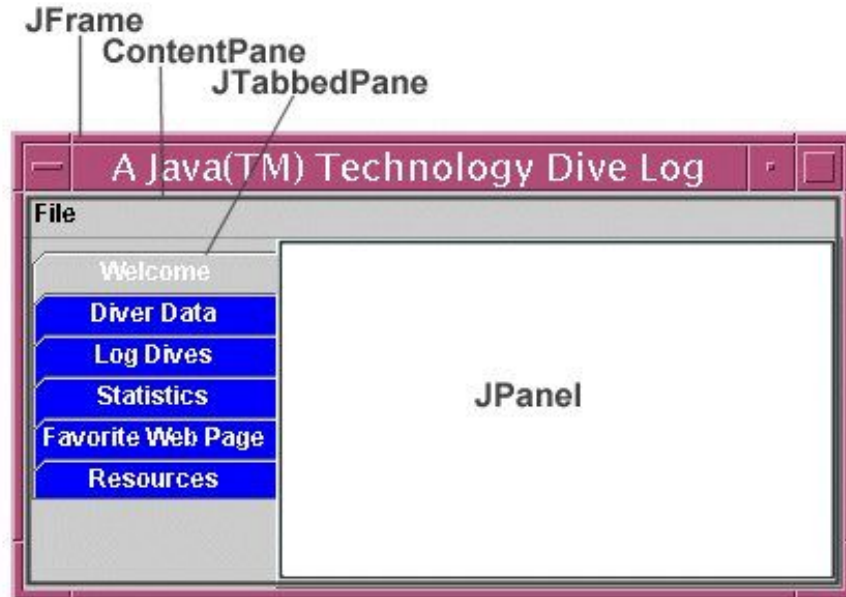
4.- Contenedores

El **formulario** esta formado por un contenedor especial que se llama contenedor de **nivel superior**.

- **Ventana (JFrame)**: es un formulario con título, los botones para maximizar, minimizar o cerrar y borde.
- **Dialogo (JDialog)**: formularios que se suelen usar para solicitar información al usuario.
- **Applet (JApplet)**: ventana que ejecuta una aplicación Java en el contexto de una página web.

4.- Contenedores

El **contenedor de nivel superior** incluye un panel de contenido (ContentPane) que permite añadir otros componentes



4.- Contenedores

También se interpretan como diálogos los siguientes componentes:

Panel de opciones (JOptionPane): responder cuestiones con respuestas del tipo si-no, aceptar-cancelar, aceptar, etc.

Selector de archivos (JFileChooser): permite seleccionar un archivo del sistema

Selector de colores (JColorChooser): permite seleccionar entre un conjunto de colores.

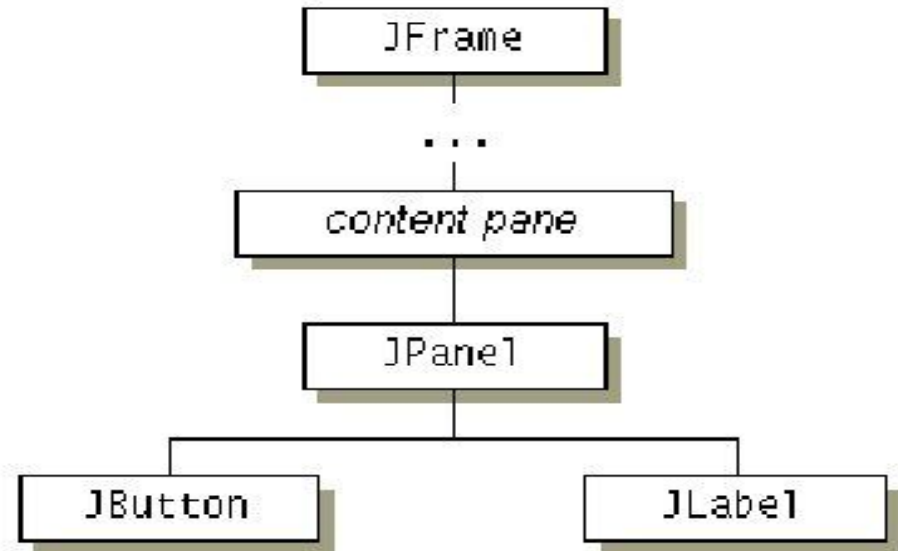
4.- Contenedores

Para distribuir el resto de los controles que se incluyen en la ventana principales:

- **Paneles (JPanel)**
- **Barra de menús (JMenu)**
- **Barra de herramientas (JToolBar)**
- **Pestañas (JTabbedPane)**
- **Paneles deslizables (JScrollPane)**
- **Ventanas internas (JInternalFrame)**
- **Paneles divididos (JSplitPane)**

4.- Contenedores

Jerarquía de contenedores



5.- Componentes de la interfaz.

[Java Swing Oracle](#)

[Añadir contenedores a una aplicación Java](#)

[Guía visual de los gestores de diseño.](#)

6.- Asociación de acciones a eventos

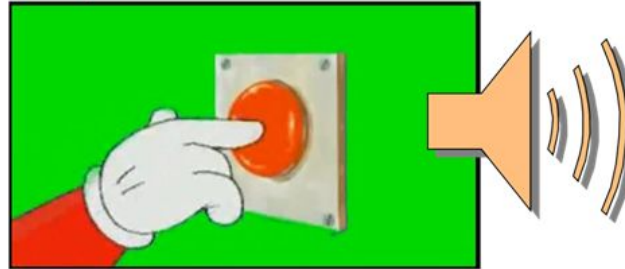
USUARIO:
Quién provoca el evento



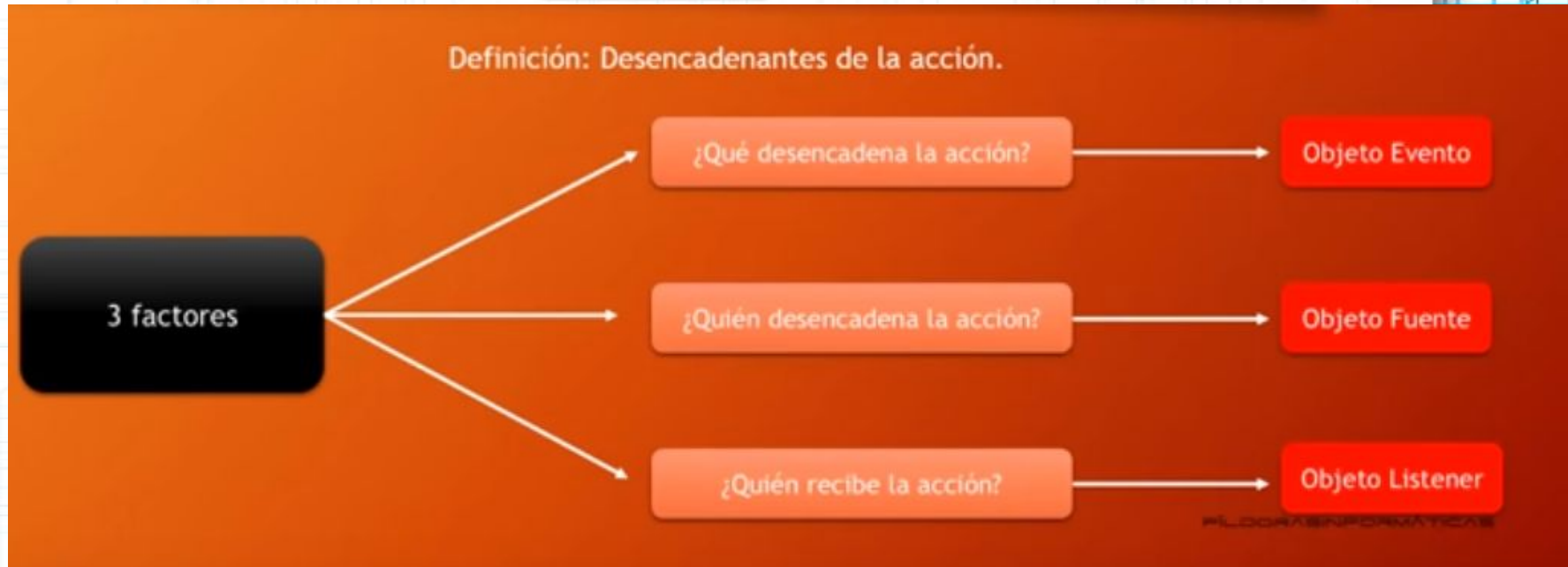
BOTÓN:
Objeto sobre el que se produce el evento



EVENTO:
Al presionar el botón..... RINGGG!!!!!!

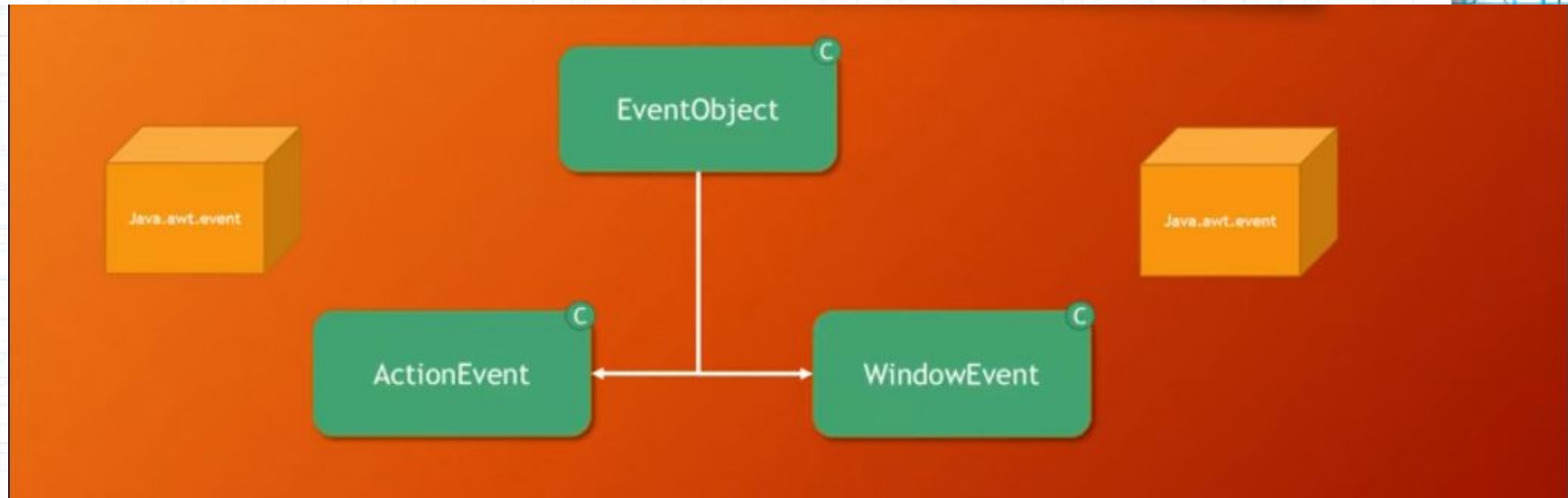


6.- Asociación de acciones a eventos

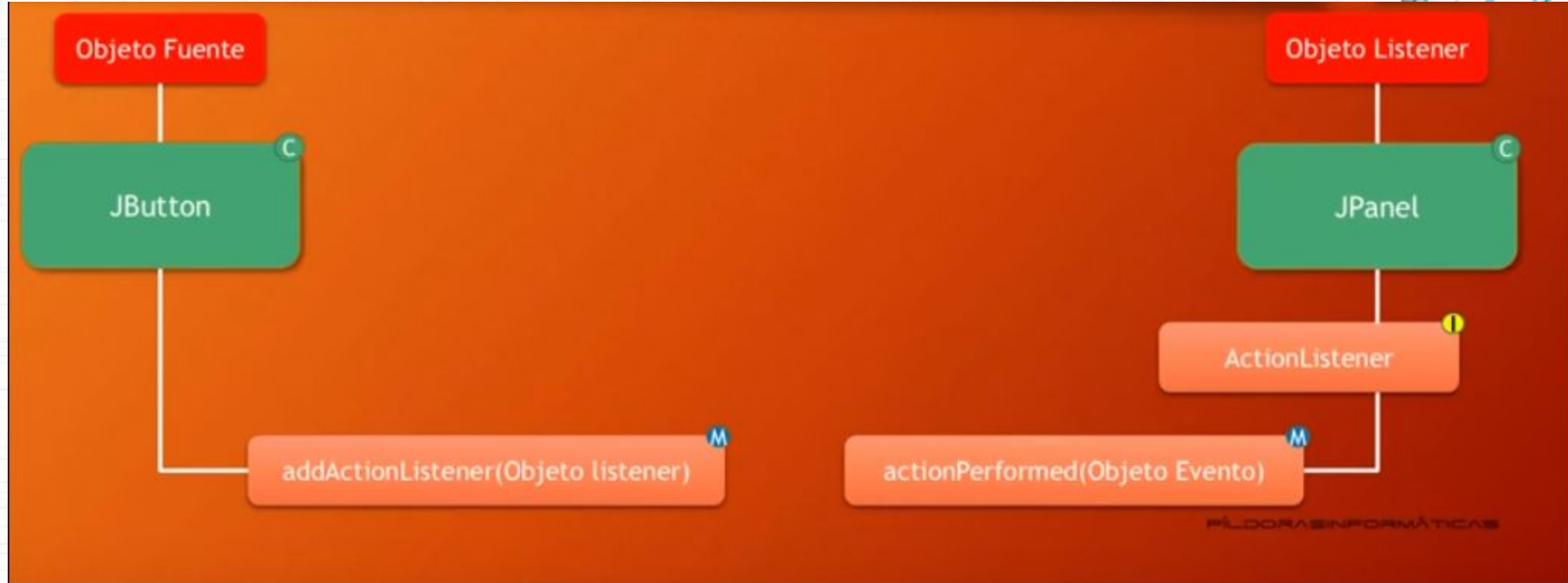


6.- Asociación de acciones a eventos

Objeto Evento_



6.- Asociación de acciones a eventos



7.- Diálogos modales y no modales.

Modales: diálogo que mantiene el foco, impidiendo que pueda ser tomado por cualquier otro, dentro de la aplicación, de forma que no podemos seguir ejecutando la aplicación hasta cerrarlo.

No modales: el Diálogo ,una vez que se encuentra activo, permite alternar el foco a cualquier otro diálogo que se encuentre abierto.

7.- Diálogos modales y no modales.

Modal: diálogo que mantiene el foco

Los modales **respecto a una aplicación** permiten alternar el foco a otros diálogos del sistema, pero no al diálogo que le da origen (diálogo padre).

Los que confirman una acción del usuario.

Los modales **respecto al sistema** no va ceder el foco a ninguna otra aplicación hasta que produzca la acción sobre él. Un ejemplo podría ser **un diálogo que permita apagar el equipo.**

7.- Diálogos modales y no modales.

codigos

Códigos:

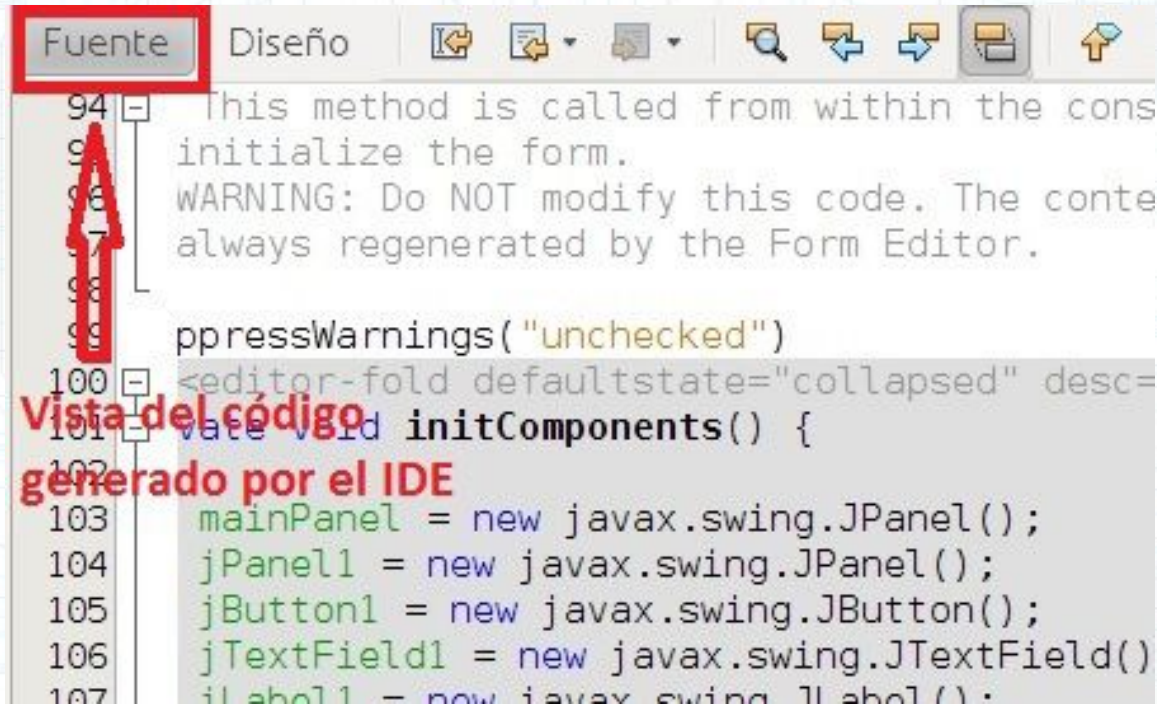
Modales:

https://www3.gobiernodecanarias.org/medusa/eforma/campus/pluginfile.php/5443764/mod_resource/content/2/71_dilogos_modales.html

No modales:

https://www3.gobiernodecanarias.org/medusa/eforma/campus/pluginfile.php/5443764/mod_resource/content/2/72_dilogos_no_modales.html

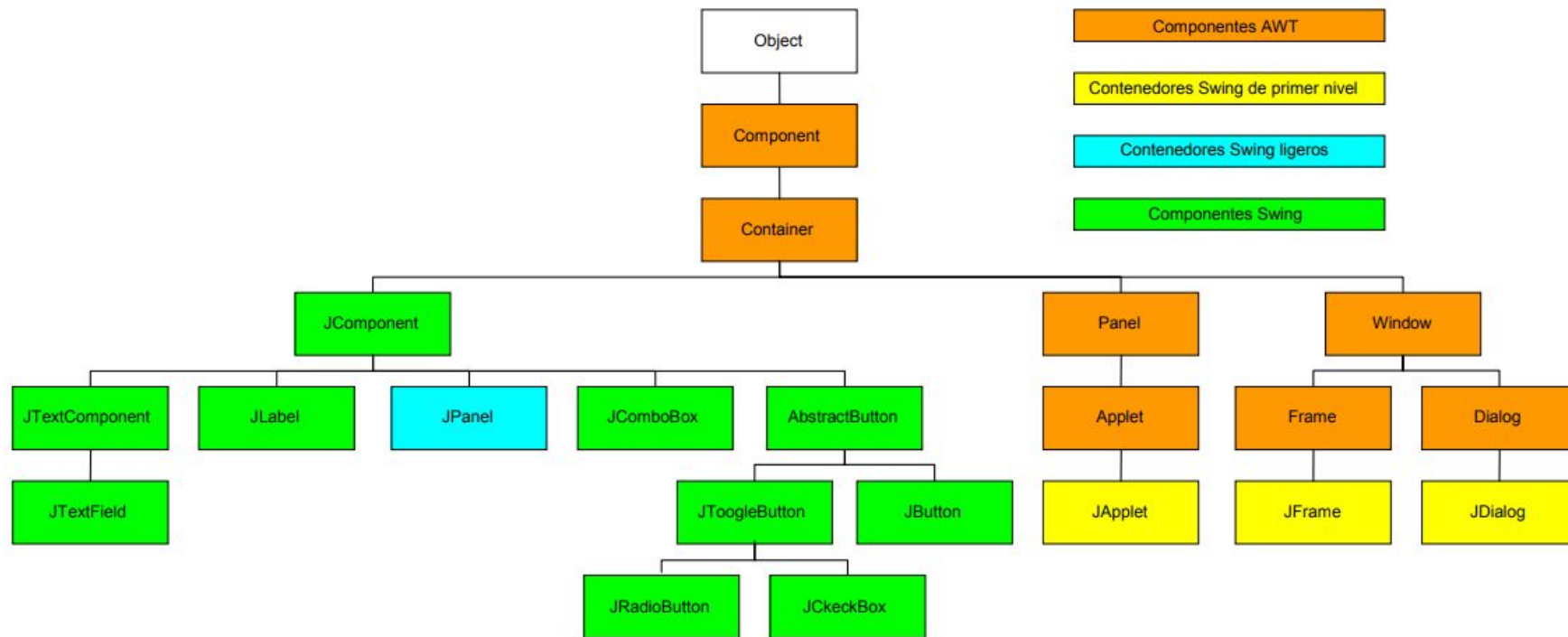
8.- Edición de código generado por herramientas de diseño.



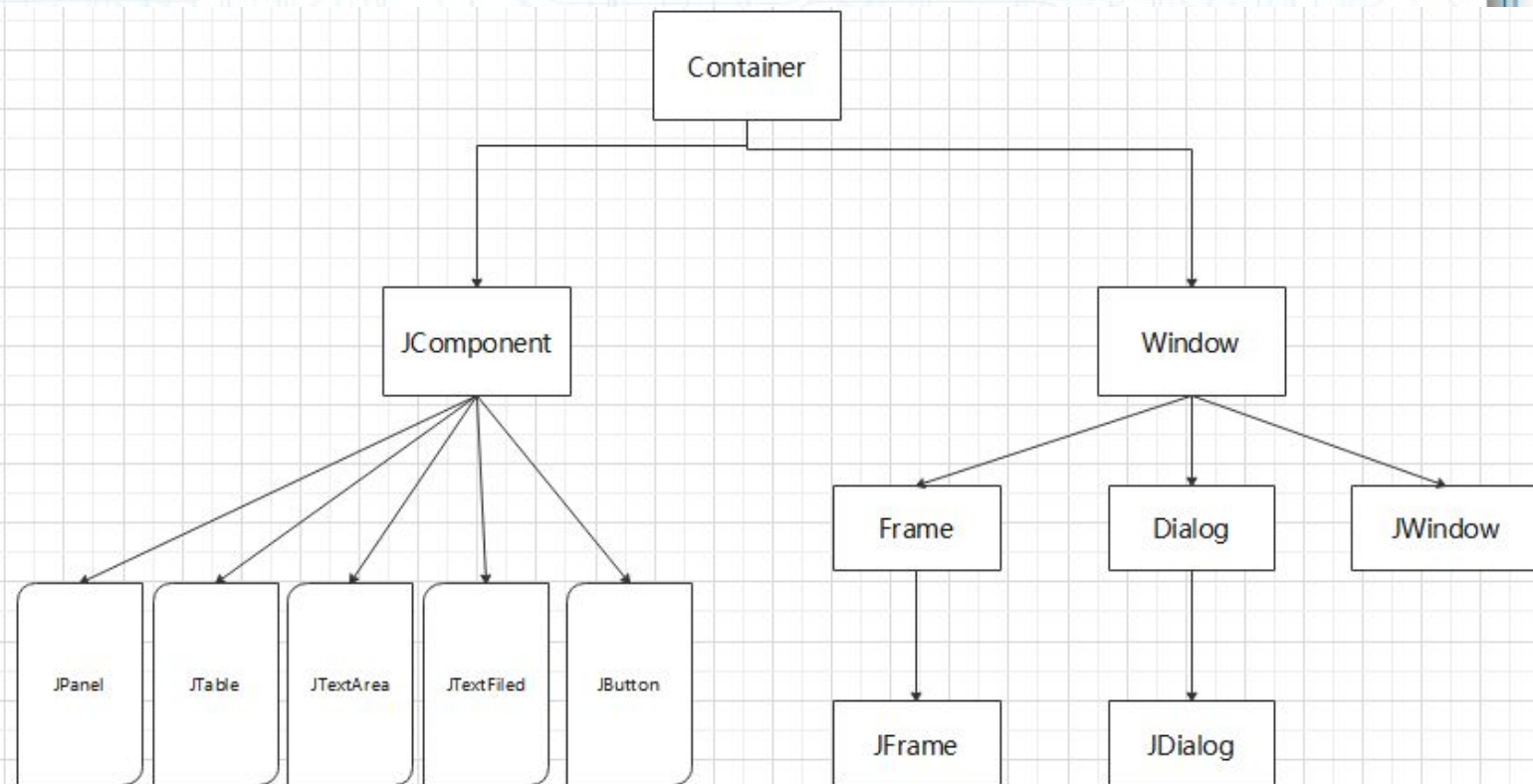
```
94 This method is called from within the cons
95 initialize the form.
96 WARNING: Do NOT modify this code. The conte
97 always regenerated by the Form Editor.
98
99 ppressWarnings("unchecked")
100 <editor-fold defaultstate="collapsed" desc=
101 >
102     initComponents() {
103         mainPanel = new javax.swing.JPanel();
104         jPanel1 = new javax.swing.JPanel();
105         jButton1 = new javax.swing.JButton();
106         jTextField1 = new javax.swing.JTextField()
107         jLabel1 = new javax.swing.JLabel();
```

Vista del código
generado por el IDE

9.- Clases, propiedades, métodos.



9.- Clases, propiedades, métodos.



9.- (

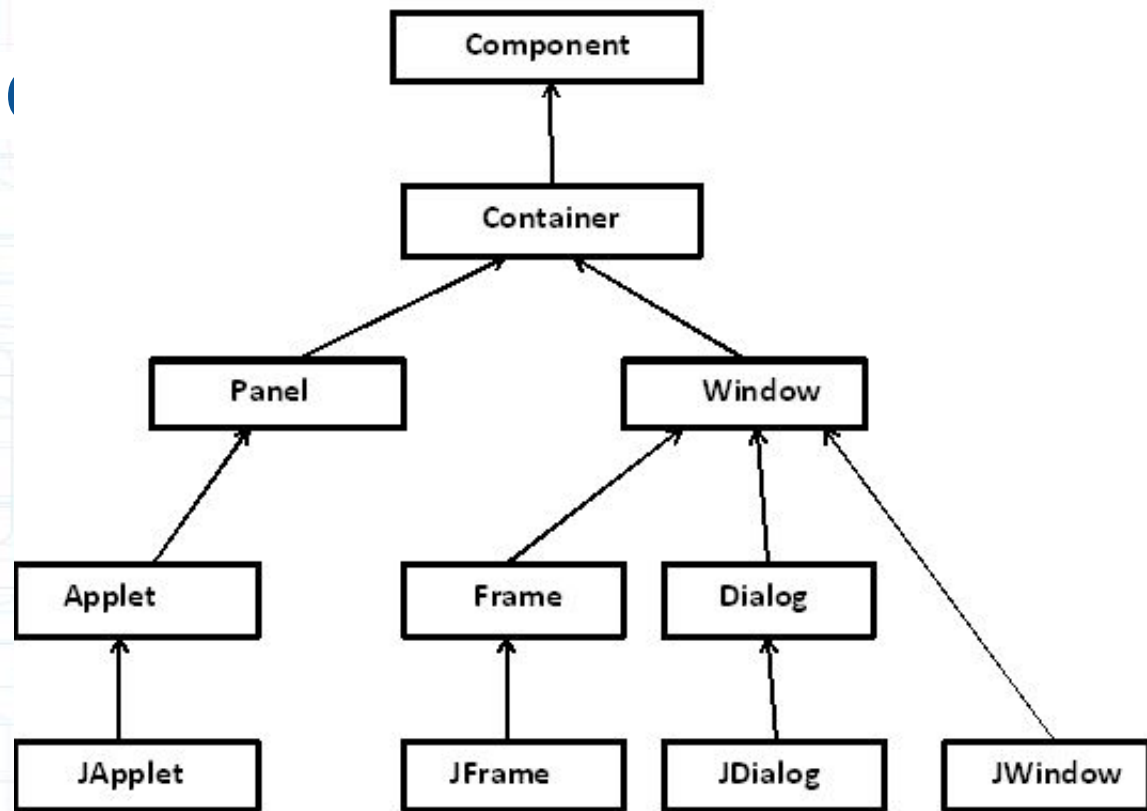


Figure 1.4 Top-Level Container Hierarchy

9.- Clases, propiedades, métodos.

- **JWindow** es una ventana sin barra de título y sin botones.
- **JFrame** es una ventana con barra de título y con los botones que permiten su manipulación.
- **JDialog** permite visualizar una cuadro de diálogo.
- **JApplet** permite crear un **applet swing**.

9.- Clase

- <https://docs.oracle.com/javase/7/docs/api/>

- <https://www.java.com/es/download/javafx/8/whatsnew/whatsnewfx8.html>

- ✓ **JComponent**: Clase base para los componentes swing.
 - ◆ **AbstractButton**: Define el comportamiento común de los botones y los menús.
 - **JButton**: Botón.
 - **JMenuItem**: Elemento de un menú.
 - ✖ **JCheckBoxMenuItem**: Elemento del menú que se puede seleccionar o deseleccionar.
 - ✖ **JMenu**: Menú de una barra de menús.
 - ✖ **JRadioButtonMenuItem**: Elemento que forma parte de un grupo de elementos de menú.
 - **JToggleButton**: Botón de estados.
 - ✖ **JCheckBox**: Casilla de verificación.
 - ✖ **JRadioButton**: Botón de opción.
 - ◆ **JColorChooser**: Diálogo para seleccionar colores.
 - ◆ **JComboBox**: Combinación de caja de texto y lista desplegable.
 - ◆ **JLabel**: Etiqueta.
 - ◆ **JList**: Lista desplegable.
 - ◆ **JMenuBar**: Barra de menús.
 - ◆ **JOptionPane**: Componente que facilita la visualización de un cuadro de diálogo.
 - ◆ **JPanel**: Contenedor genérico.
 - ◆ **JPopupMenu**: Menú que aparece cuando se selecciona un elemento de una barra de menús.
 - ◆ **JProgressBar**: Barra de progreso.
 - ◆ **JScrollBar**: Barra de desplazamiento.
 - ◆ **JScrollPane**: Área de trabajo con barras de desplazamiento.
 - ◆ **JSeparator**: Separador para colocar entre dos elementos del menú.
 - ◆ **JSlider**: Permite seleccionar un valor dentro de un intervalo que define.
 - ◆ **JTableHeader**: Se utiliza para manejar la cabecera de una tabla.
 - ◆ **JTextComponent**: Clase base para los componentes de texto.
 - **JEditorPane**: Edita diferentes tipos de contenido.
 - ✖ **JTextPane**: Edita texto con formato, incluyendo imágenes.
 - **JTextArea**: Caja de texto multilinea.
 - **TextField**: Caja de texto de una línea.
 - ✖ **JPasswordField**: Se usa para introducir contraseñas.
 - ◆ **JToolBar**: Barra de herramientas.

9.- Clases, propiedades, métodos.

- ❖ **JComponent**: Clase base para los componentes swing.
- **AbstractButton**: Define el comportamiento común de los botones y los menús.
 - **JButton**. Botón.
 - **JMenuItem**. Elemento de un menú.
 - **JCheckBoxMenuItem**: Elemento del menú que se puede seleccionar o deseleccionar.
 - **JMenu**: Menú de una barra de menús.
 - **JRadioButtonMenuItem**: Elemento que forma parte de un grupo de elementos de menú.
 - **JToggleButton**: Botón de estados.
 - **JCheckBox**. Casilla de verificación.
 - **JRadioButton**: Botón de opción.

9.- Clases, propiedades, métodos.

- ❖ **JComponent:** Clase base para los componentes swing.
- **JColorChooser:** Diálogo para seleccionar colores.
- **JComboBox:** Combinación de caja de texto y lista desplegable.
- **JLabel:** Etiqueta.
- **JList:** Lista desplegable.
- **JMenuBar:** Barra de menús.
- **JOptionPane:** Componente que facilita la visualización de un cuadro de diálogo.
- **JPanel:** Contenedor genérico.
- **JPopupMenu:** Menú que aparece cuando se selecciona un elemento de una barra de menús.
- **JProgressBar:** Barra de progreso.

9.- Clases, propiedades, métodos.

- ❖ **JComponent:** Clase base para los componentes swing.
- **JScrollBar:** Barra de desplazamiento.
- **JScrollPane:** Área de trabajo con barras de desplazamiento.
- **JSeparator:** Separador entre dos elementos del menú.
- **JSlider:** Permite seleccionar un valor dentro de un intervalo.
- **JTableHeader:** Se utiliza para manejar la cabecera de tabla.
- **JTextComponent:** Clase base para los componentes texto.
 - **JEditorPane:** Edita diferentes tipos de contenido.
 - **JTextPane:** Edita texto con formato, con imágenes.
 - **JTextArea:** Caja de texto multilínea.
 - **TextField:** Caja de texto de una línea.
 - **JPasswordField:** Se usa para introducir contraseñas.
- **JToolBar:** Barra de herramientas.

9.- Clases, propiedades, métodos.

- ❖ **background:** Determina el color de fondo del componente.
- ❖ **font:** Establece el tipo de fuente que va a mostrar el componente.
- ❖ **foreground:** Establece el color de la fuente que muestra el componente.
- ❖ **horizontalAlignment:** Establece la alineación del texto dentro del componente en el eje X.
- ❖ **icon:** Indica si el componente muestra o no un icono, y cual sería.
- ❖ **labelFor:** Indicaría si el componente es etiquetable.
- ❖ **text:** Muestra el texto que indica la propiedad en componentes como caja de texto o etiquetas.
- ❖ **toolTipText:** Con esta propiedad definimos el texto que aparece como tool tip.

9.- Clases, propiedades, métodos.

- ❖ **verticalAlignment**: Establece la alineación del texto dentro del componente en el eje Y.
- ❖ **alignmentX**: Alineamiento horizontal preestablecido para el componente.
- ❖ **alignmentY**: Alineamiento vertical preestablecido para el componente.
- ❖ **autoscrolls**: Determina si el componente puede realizar scroll de forma automática cuando se arrastra el ratón.
- ❖ **border**: Establece el tipo de borde que va presentar el componente.
- ❖ **componentPopupMenu**: Establece el menú contextual que se muestra en este componente.

9.- Clases, propiedades, métodos.

- ❖ **cursor**: Establece el tipo de cursor que se va a mostrar cuando el curso entre en el componente.
- ❖ **disableIcon**: el icono a mostrar si el componente no está activo.
- ❖ **enabled**: Nos indica si el componente está o no activo.
- ❖ **focusable**: Establece si el componente puede o no, recibir el foco.
- ❖ **horizontalTextPosition**: Establece la posición horizontal del texto del componente, en relación con su imagen.
- ❖ **iconTextGap**: Si el componente tiene activo el texto y el icono, con esta propiedad se establece la distancia entre ellos.
- ❖ **inheritsPopupMenu**: si el menú contextual se hereda o no.
- ❖ **inputVerifier**: el componente de verificación para este.

9.- Clases, propiedades, métodos.

- ❖ **maximunsize**: Establece el tamaño máximo del componente.
- ❖ **minimunsize**: Establece el tamaño mínimo del componente.
- ❖ **name**: Establece el nombre del componente.
- ❖ **opaque**: Modifica la opacidad del componente.
- ❖ **preferredSize**: Tamaño predefinido para este componente.
- ❖ **verifyInputWhenFocusTarget**: Si el componente es verificado antes de recibir el foco.
- ❖ **verticalTextPosition**: Establece la posición vertical del texto del componente, en relación con su imagen.

9.- Clases, propiedades, métodos.

❖ JFrame.

- getTitle(),
- setBounds(x,yx,w,h),
- setLocation,
- setMaximumSize(w,h),
- setMinimumSize(w,h),
- setPreferredSize(w,h),
- setResizable(bool),
- setSize(w,h),
- setTitle(str) y
- setVisible(bool).

❖ JPanel.

- add(), para añadir componentes al panel.

9.- Clases, propiedades, métodos.

❖ **JLabel:**

- setText() que permite modificar el texto,
- setVerticalAlignment() para modificar la alineación vertical, etc

❖ **JButton:**

- setEnabled(bool) que permite activar o desactivar el botón,
- isEnabled() que permite comprobar si está o no activo,
- setMnemonic(char) que permite asociar una tecla al botón,

❖ **JProgressBar, JScrollBar.**

- setExtent(),
- setMaximum(),
- setMinimum(),
- setValue(),
- getValuesAdjusting() y
- setOrientation().

9.- Clases, propiedades, métodos.

❖ **JSlider:**

- setPaintTicks(bool),
- setPaintLabels(bool).
- setSnapToTicks(true) y
- setInverted(bool).

❖ **JRadioButton.**

- isSelected() que devuelve el estado del mismo.

❖ **JList.**

- addElement(objName),
- elementAt(index),
- removeElement(objName),
- contains(objName) Para comprobar el estado de la lista,
- indexOf(name) y
- size().
- copyInto(array) Para convertir la lista en array se usa .

9.- Clases, propiedades, métodos.

❖ JComboBox.

- `setEditable(true)` que permite editar los items del combo box.
- `getSelectedItem()` Para recuperar los ítems seleccionados
- `getSelectedIndex()`.
- `getItemAt(idx)`, `getItemCount()`, `setSelectedItem(obj)` y `setMaximumRowCount(x)`.

❖ JMenuItem.

- `addSeparator()` o
- `insertSeparator(posn)`.
- `setEnabled(bool)` activar o desactivar
- `isEnabled()` comprobar si están activos .
- `setMnemonic(char)` añadir teclas de acceso rápido

9.- Clases, propiedades, métodos

❖ **JTextPane y JTextArea.**

- `getText()`,
- `setText()`,
- `append(str)`,
- `insert(str,pos)`,
- `replace(str,beg,end)`,
- `setEditable(bool)`,
- `setLineWrap(bool)` y
- `setWrapStyleWord(bool)`.

10.- Eventos, escuchadores.

- ❖ **La fuente del evento (event source):** Es el componente que origina el evento.
- ❖ **El escuchador (event listener):** es el encargado de atrapar o escuchar el evento.
- ❖ **El manejador del evento (event handler),** es el método que permite implementar la interfaz.

...

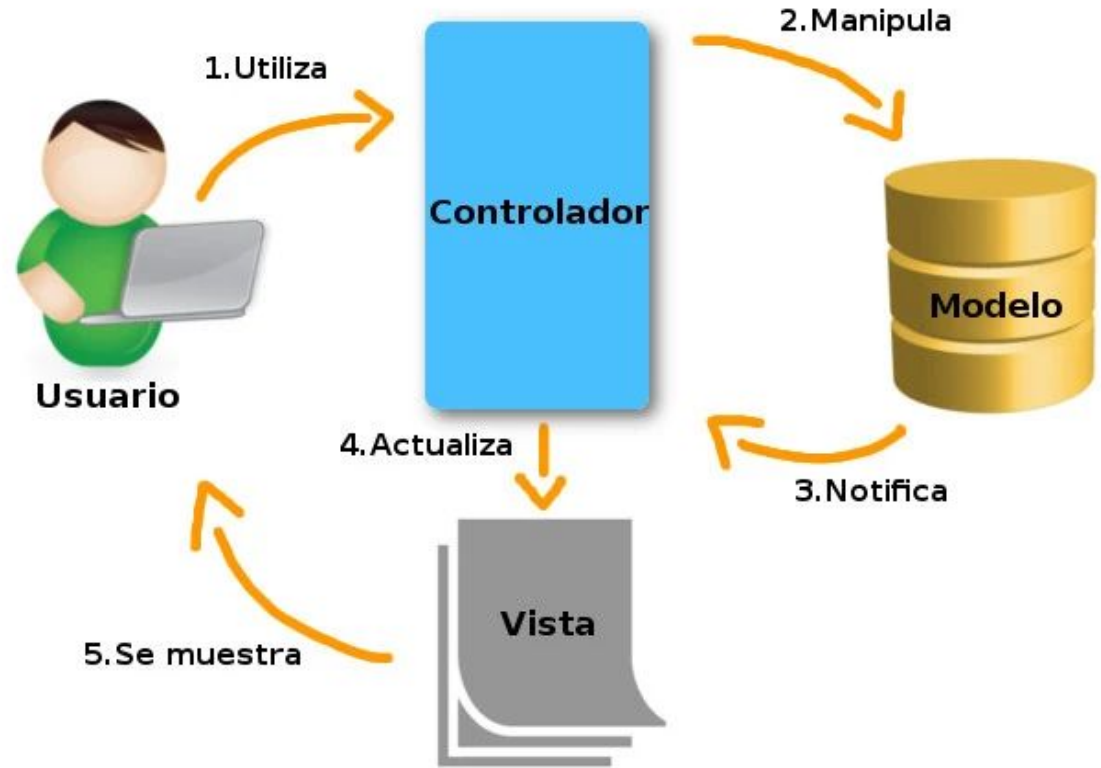
11.- Patrón de programación MVC

- ❖ **Modelo**
- ❖ **Vista**
- ❖ **Controlador**



11.- Patrón de prog

- ❖ **Modelo**
- ❖ **Vista**
- ❖ **Controlador**



12.- Estructura de datos “Array”

Diferencias entre

- ❖ **Array**
- ❖ **ArrayList**
- ❖ **List**

12.- Estructura de datos “Array”

Diferencias entre Array y ArrayList:

- ❖ **ArrayList -> tamaño dinámico,
Array -> se define en la creación**
- ❖ **Un ArrayList no puede ser de datos primitivos, sólo Objetos.**
- ❖ **El ArrayList permite comprobar que los datos que se añaden a la colección son del tipo correcto en tiempo de compilación.**
- ❖ **Array -> varias dimensiones,
ArrayList es unidimensional (ArrayList de ArrayLists).**

