JAVA

3. Diseño del Interfaz Usuario

```
import java.awt.*;
```

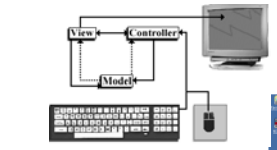
1. Diseño en Ventanas

1. Descripción de Ventanas (Contenedores)
2. Configuración (títulos, tamaño...)
3. Componentes (menús, botones, etc.)
4. Ubicación de elementos (layout)
5. Gestión eventos

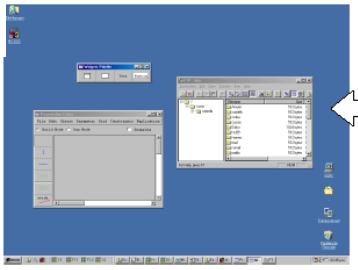
   import java.awt.event.*;

1.1. Diseño: Sistema de Ventanas

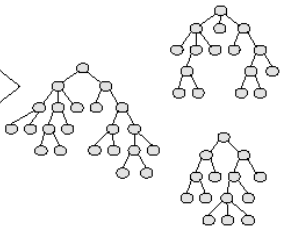
- Los programas se representan en una o varias ventanas
- Ventana: Área rectangular que permite la interacción con el usuario (E/S)
- Cada ventana (contenedor) se describe mediante componentes y contenedores jerárquicamente
- Iniciativa del usuario (no del programa) mediante acciones
- Cada acción de usuario es gestionada por la ventana activa



Acción de usuario



Diseño de Ventanas



1.2. Diseño: Paquetes IU de Java

AWT

```
import java.awt.*;
```

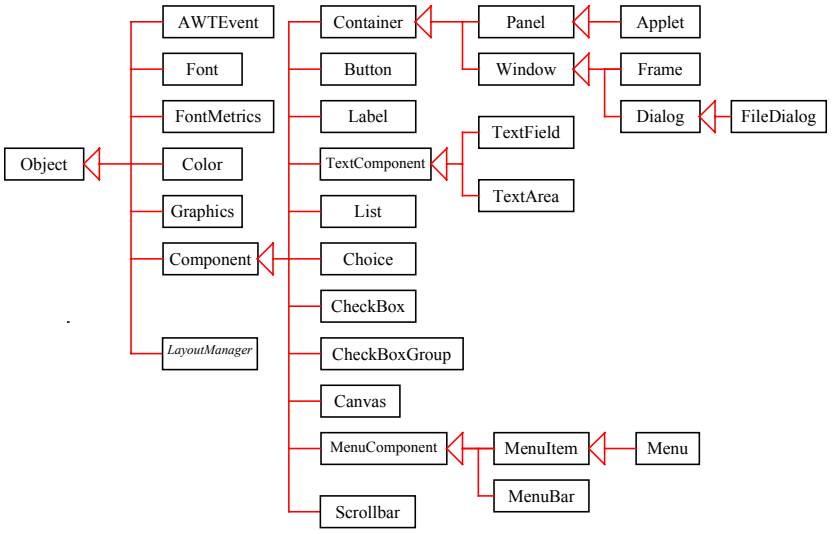
- Primera versión de Java (jdk.1.0)
- Componentes pesados (*heavyweight*) dependientes de plataforma
- Posee sobre 30 componentes
- Consume más recursos

SWING

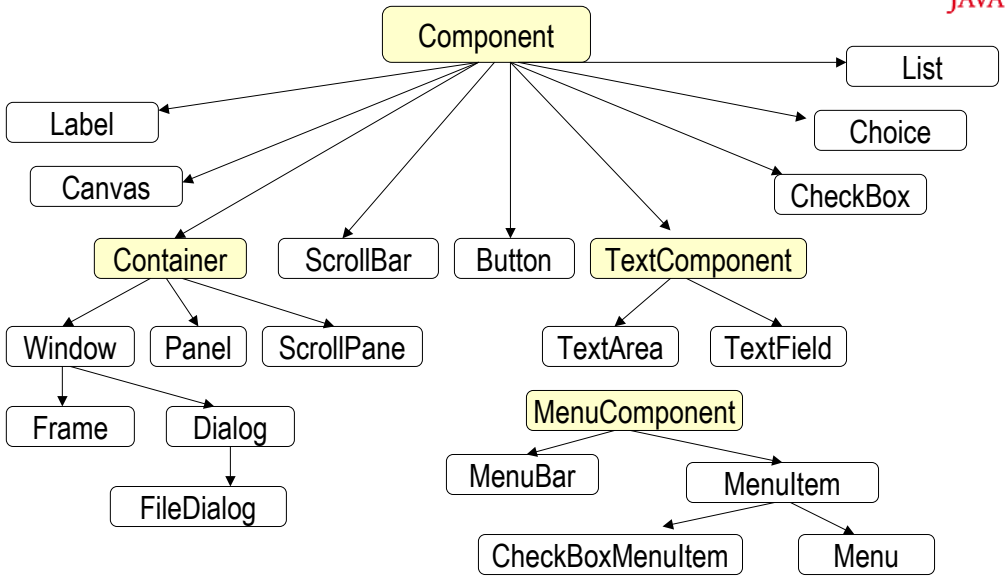
```
import javax.swing.*;
```

- Aparece en Java2 (jdk.1.2)
- Componenetes ligeros (*lightweight*) independientes de plataforma
- Diferentes *look&feel*
- Cerca de 250 componentes
- Sustituye y extiende los componentes AWT (JButton, JPanel, JTextField)
- No se pueden/deben mezclar

2. AWT: Jerarquía de objetos

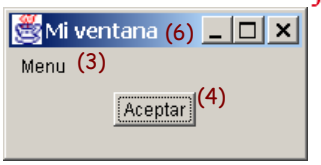


2. AWT: Jerarquía de componentes



2.1 AWT: Diseño de una Ventana

```
import java.awt.*;
public class Ventana extends Frame {                    (1)
  public Ventana () {    // Constructor
      setLayout (new FlowLayout ());                     (2)
      Menu m = new Menu ("Menu");
          m.add (new MenuItem ("Salir"));
          MenuBar mb = new MenuBar ();
          mb.add (m);                                    (3)
      setMenuBar (mb);
      add (new Button ("Aceptar"));                      (4)
  }
  public static void main (String[] args) {
    Ventana v = new Ventana ();                          (5)
    v.setTitle ("Mi ventana");                           (6)
    v.pack ();      v.setVisible (true);                 (7)
  }
} // Fin de la clase Ventana
```

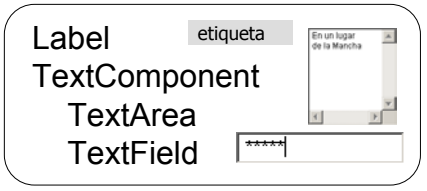


Constructor
- (1) Extiende clase Frame
- (2) Añade controlador geométrico
- (3) Añade componente (barra de menú)
- (4) Añande componente (botón)

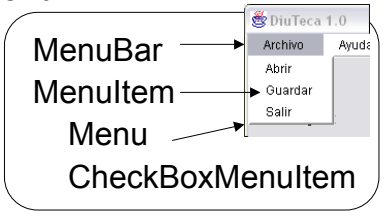
Main
- (5) Crea un objeto
- (6) Propiedades
- (7) Ubica en Display

3. AWT: Clasificación componentes

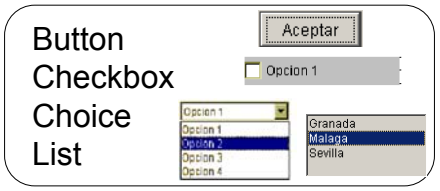
Texto



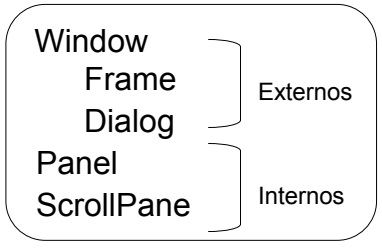
Menú



Selección



Contenedores



3.1. Java.awt.Label

❖ Descripción:

Etiqueta de texto estático

❖ Constructores:

```
Label();
Label (String);
Label (String, LEFT|RIGHT|CENTER);
```

❖ Propiedades y métodos:

- **Etiqueta**

  String getTextl();
  void setText(String);

- **Alineación**

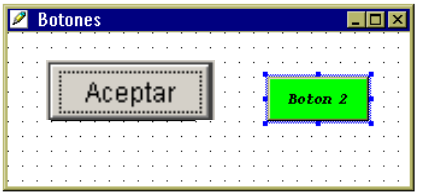
  getAlignment()
  void SetAlignment(Label.LEFT|RIGHT|CENTER);

```
Label e1,e2,e3;
e1=new Label("Etiq 1", Label. LEFT);
e2=new Label("Etiq 2", Label. CENTER);
e2=new Label("Etiq 3");
e3.setAlignment(Label.RIGHT);
add(e1);
add(e3);
```


3.2. Java.awt.Button

❖ Descripción:

Control de Pulsación



❖ Constructores:

Button()
Button (String)

❖ Propiedades y métodos:

- **Etiqueta del botón**

  String getLabel();
  void setLabel(String);

- **Activo/Inactivo**

  void setEnabled(Boolean);

- **Eventos**

  void setActionCommand (String); // nombre de la acción a realizar

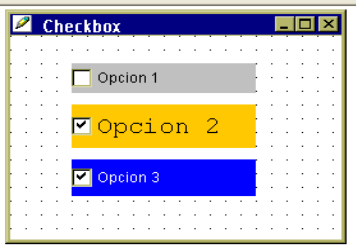
Emite un ActionEvent al ser pulsado (ActionPerformed)

```
Button boton;
boton=new Button("Aceptar");
boton.setEnabled(true);
Boton.setActionCommand("volver");
add(boton);
...
public void ActionPerformed(ActionEvent e) {
   String s = e.getActionCommand();
   if (s.equals("volver")) {...}
}
```

3.3. Java.awt.CheckBox

❖ Descripción:

Botones de Comprobación
Se pueden agrupar (CheckBoxGroup)



❖ Constructores:

CheckBox()
CheckBox(String, boolean); // estado

❖ Propiedades y métodos:

- **Etiqueta del botón**
  String getLabel();
  void setLabel(String);
- **Estado**
  void setState(boolean);
  boolean getState();

```
CheckBox opt1,opt2;
opt1=new CheckBox("Opcion 1");
opt2=new CheckBox("Opcion 2", true);
opt1.setState(false);
this.add(opt1);  // añadir a frame
this,add(opt2)
...
```

Emite un `ItemEvent` al cambiar de estado (`ItemStateChanged`)

3.4. Java.awt.CheckBoxGroup

❖ Descripción:

Agrupación de botones (CheckBox)
Selección excluyente
Sólo uno del grupo puede estar selecc.

❖ Constructores:

CheckBoxGroup()
CheckBox(String, boolean, CheckBoxGroup)

❖ Propiedades y métodos:

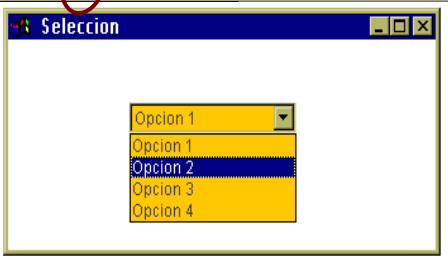
- **Selección**
  Checkbox getSelectedCheckbox();
  void setSelectedCheckbox(Checkbox);
  void setCheckboxGroup(CheckboxGroup g);

```
CheckBoxGroup g = new CheckBoxGroup();
CheckBox opA,opB, opC,opD;
opA=new CheckBox("A", true,  g );
opB=new CheckBox("B", false, g);
opC=new CheckBox("C", false, g);
opD = new CheckBox("D");
opD.setCheckboxGroup(g);
// add..
```

3.5. Java.awt.Choice

❖ Descripción:

Lista de selección desplegable



❖ Constructores:

Choice();

❖ Propiedades y métodos:

- **Añadir/recuperar items**

```
void    add (String);
String getItem(int);
int     getItemCount();
void    insert(String item, int  index);
```

```
Choice elecc = new Choice();
elecc.add( "Opción 1" );
elecc.add( "Opción 2" );
elecc.add( "Opción 3" );
elecc.select( "Opción 2" );
...
```

- **Eliminar**

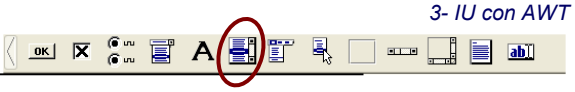
```
void remove(int posicion)
void remove(String item); // primera ocurrencia
void removeAll();  // todos
```

- **Seleccionar**

```
void select(int)
void select(String)
```

Emite un `ItemEvent` al cambiar de estado (`ItemStateChanged`)

3.6. Java.awt.List



❖ Descripción:

Lista de selección (con deslizadores automáticos si no cabe en área)

❖ Constructores:

List ();
List (int); // nº filas
List (int, boolean); // selec múltiple= true

❖ Propiedades y métodos:

- **Operaciones sobre items**

add (String); add(String, int indice)
remove(int); remove (String)
void select(int); void select(String);
void removeAll();
String getSelectedItem()

```
List lista; String s;
lista= new List();
lista.addItem("Granada");
lista.addItem("Malaga");
lista.addItem("Sevilla");
lista.select(1);
s = lista.getSelectedItem();
...
```

Emite un `ActionEvent` con doble click (`ActionPerformed`)
Emite un `ItemEvent` al cambiar la selección (`ItemStateChanged`)

3.7. Java.awt.TextField

❖ Descripción:

Texto editable (una línea)
Puede eliminar eco (passwords)

❖ Constructores:

TextField();
TextField(String, int); // nº columnas

❖ Propiedades y métodos:

• **Operaciones sobre texto**

```
boolean   echoCharIsSet()
char  getEchoChar()
void  setEchoChar(char c); // carácter de eco
void  setText(String);
void  setColumns(int)  // nº columnas
```

```
TextField t = new TextField(); String s;
t.setColumns(12);
t.setEchoChar('*');
...
```

Emite un TextEvent al cambiar un carácter del texto (keyTyped)
Emite un ActionEvent al pulsar **Enter** (ActionPerformed)

3.8. Java.awt.TextArea

❖ Descripción:

Área de texto editable

❖ Constructores:

SCROLLBARS_BOTH,
SCROLLBARS_HORIZONTAL_ONLY
SCROLLBARS_NONE

TextArea();
TextArea(String);
TextArea(String, int filas, int cols);
TextArea (String, int f, int c, int estilo)

❖ Propiedades y métodos:

• **Operaciones sobre texto**

append (String);
insert(String, int);
replaceRange(String, int, int)
getRows(); getColumns();
setRows(int); setColumns(int);

```
TextArea a = new TextArea("",15,25, SCROLLBARS_BOTH );

a.append("En un Lugar \n de la ...");
...
```

Emite un `TextEvent` al cambiar un carácter del texto (`keyTyped`)
No Emite `ActionEvent`

3.9. Java.awt.Scrollbar

❖ Descripción:

Barra de desplazamiento

❖ Constructores:

Scrollbar();
Scrollbar(int orientation);
Scrollbar(int orientation, int value, int visible, int minimum, int maximum)

Scrollbar.HORIZONTAL|VERTICAL

❖ Propiedades y métodos:

```
s = new Scrollbar(Scrollbar.HORIZONTAL, 0, 60, 0, 300);

...
```

• **Operaciones**

int getMaximum (); void setMaximum(int);
int getMinimum (); void setMinimum(int);
int getOrientation (); void setOrientation(int or);
int getValue(); void SetValue(int pos);

3.10. AWT: Contenedores

- Window
- Frame
- Dialog
- Panel
- MenuBar

Tamaño	setSize() minimumSize()
Redimensionar	setResizable(bool) minimumSize()
Visible	pack( ) dispose( ) setVisible(bool)
Modal	SetModal(bool)

SetLayout(new FlowLayout(..))
add(..)

jframe1.setMenuBar(menubar1)

3.11. Java.awt.ScrollPane

❖ Descripción:

Panel con scroll automático para el componente que se inserte

❖ Constructores:

```
ScrollPane();
ScrollPane(int policia);
 // ScrollPane.SCROLLBARS_ALWAYS|SCROLLBARS_NEVER | SCROLLBARS_AS_NEEDED
```

❖ Propiedades y métodos:

```
public int getScrollbarDisplayPolicy() ;
void setScrollbarDisplayPolicy(int p);
```

```
sp = new ScrollPane (ScrollPane.SCROLLBARS_ALWAYS);

...
```


© Miguel Gea y Fco. Luis Gutiérrez
Dpt. Lenguajes y Sistemas Informáticos