



Unidad 4

IGU (Interfaz Gráfica de Usuario) y MVC (Modelo Vista Controlador)

Prof: Lic. Rosales Pablo (prosales@unpata.edu.ar)
Materia: Programación Orientada a Objetos
Año: 2024





Temario

1. Interfaz gráfica del usuario (IGU)
2. Swing / AWT
3. JFrame
4. JLabel
5. AbstractButton
6. Button
7. JRadioButton
8. JCheckBox
9. JComboBox
10. JTextComponent
11. JTextField y JPasswordField
12. JTextArea
13. Evento
14. Modelo Vista Controlador



Interfaces Gráficas de Usuario (IGU)

Hasta la versión Java SE 1-2 `java.awt` -> Abstract Window Toolkit (AWT)

Sustituida por Swing, proporcionando más componentes visuales. Contiene tres API una para 2D, otra para arrastrar y soltar y el último para facilitar el acceso.

AWT depende del sistema operativo donde se esté ejecutando para mostrar un componente visual.

Swing no depende del SO, los elementos visuales son todos iguales.

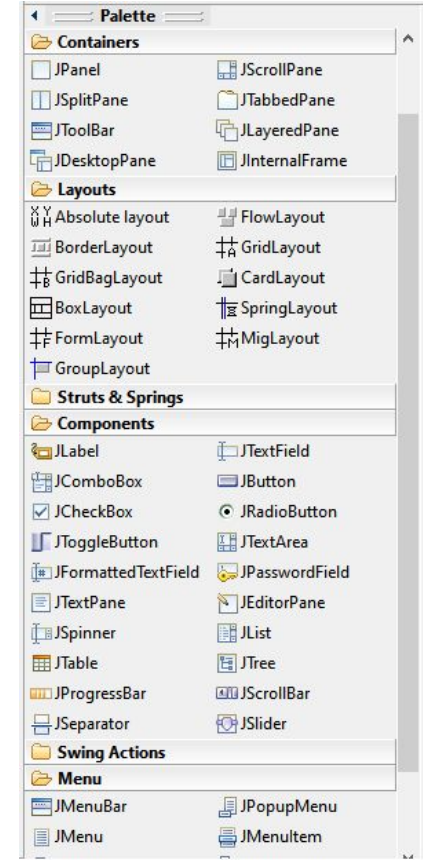
Swing

Incorpora

- Frames
- Containers
- Botones
- Labels (etiquetas)
- Campos y áreas de texto
- Listas desplegables

Todos las clases de Swing comienzan con J ejemplo, JButton, JFrame, etc...

La mezcla con componentes de AWT puede llegar a causar inconsistencias visuales y de comportamiento.





JFrame

```
public Vista() {
    setTitle("TITULO");

    /* JFrame.DO_NOTHING_ON_CLOSE
    JFrame.HIDE_ON_CLOSE
    JFrame.DISPOSE_ON_CLOSE
    JFrame.EXIT_ON_CLOSE
    */
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    setResizable(false);

    URL iconURL = getClass().getResource("/img/32.png");
    ImageIcon icon = new ImageIcon(iconURL);
    setIconImage(icon.getImage());
    //setIconImage(Toolkit.getDefaultToolkit().getImage(Vista.class.getResource("/img/32.png")));

    setBounds(100, 100, 326, 244);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    setContentPane(contentPane);
    contentPane.setLayout(null);

    setLocationRelativeTo(null);
}
```



Desactivar Layout

En ocasiones puede interesar desactivar el layout, permitiendo trabajar con posiciones absolutas para eso se usa `setLayout(null)` del `JPanel`. `contentPane.setLayout(null);`

```
JLabel lblNewLabel = new JLabel("Mi label");  
lblNewLabel.setBounds(100, 100, 50, 20);  
//equivalente a:  
lblNewLabel.setLocation(100,100);  
lblNewLabel.setSize(50,20);
```



JLabel

Simple etiqueta de texto que además puede contener una imagen.

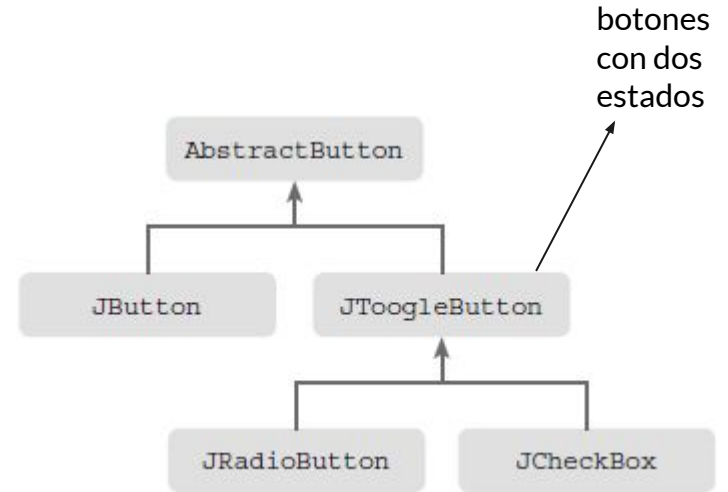
```
JLabel lbl = new JLabel("Mi label");  
lbl.setBounds(0, 0, 490, 301);  
String texto = lbl.getText();  
lbl.setText("");  
lbl.setIcon(new ImageIcon(getClass().getResource("/img/ungif.gif")));
```

AbstractButton

Superclase abstracta de botones que define comportamiento:

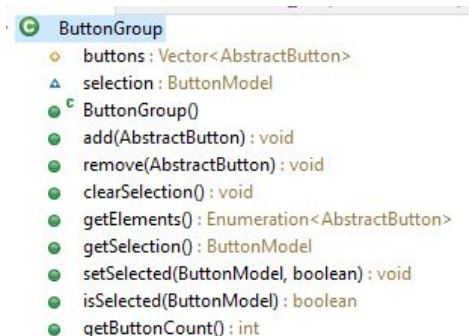
- `setText(String str) / getText()`
- `isSelected() / setSelected(Boolean bool)`
- `doClick(int tiempo)` hace click programáticamente
- `setIcon(Icon icon)`
- `setMnemonic(int mnem)` ALT+tecla
- `addActionListener(ActionListener listener)`

La clase `JButton` es la más usada, no agrega mucho más comportamiento que el especificado por la superclase



JRadioButton

- JRadioButton que son agrupados por un ButtonGroup.
- Para seleccionar uno por defecto se usa el `unRdbtn.setSelected(true)`
- para obtener el seleccionado `group.getSelection()`



```
ButtonGroup group = new ButtonGroup();
JRadioButton rdbtnA = new JRadioButton("A");
rdbtnA.setBounds(65, 308, 33, 23);
rdbtnA.setSelected(true);
contentPane.add(rdbtnA);
```

```
JRadioButton rdbtnB = new JRadioButton("B");
rdbtnB.setBounds(110, 308, 39, 23);
contentPane.add(rdbtnB);
```

```
JRadioButton rdbtnC = new JRadioButton("C");
rdbtnC.setBounds(151, 308, 49, 23);
contentPane.add(rdbtnC);
```

```
group.add(rdbtnA);
group.add(rdbtnB);
group.add(rdbtnC);
```

JCheckBox

Cambia solo en el aspecto visual, por lo general se usa para múltiple selección, aunque también permite ser agregado a un ButtonGroup



```
JCheckBox chckbxD = new JCheckBox("D");  
chckbxD.setSelected(true);  
chckbxD.setBounds(255, 308, 39, 23);  
contentPane.add(chckbxD);
```

```
JCheckBox chckbxE = new JCheckBox("E");  
chckbxE.setSelected(true);  
chckbxE.setBounds(296, 308, 39, 23);  
contentPane.add(chckbxE);
```

```
JCheckBox chckbxF = new JCheckBox("F");  
chckbxF.setBounds(337, 308, 39, 23);  
contentPane.add(chckbxF);
```

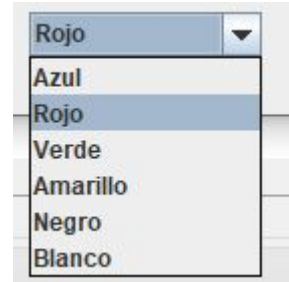
JComboBox

```
JComboBox<Dias> comboBox = new JComboBox<Dias>(Dias.values());
```

```
String[] items = {"Azul", "Rojo", "Verde", "Amarillo", "Negro", "Blanco"};  
JComboBox<String> comboBox = new JComboBox<String>(items);  
comboBox.setBounds(460, 312, 116, 22);  
contentPane.add(comboBox);
```

```
comboBox.getSelectedIndex();  
comboBox.getSelectedItem();
```

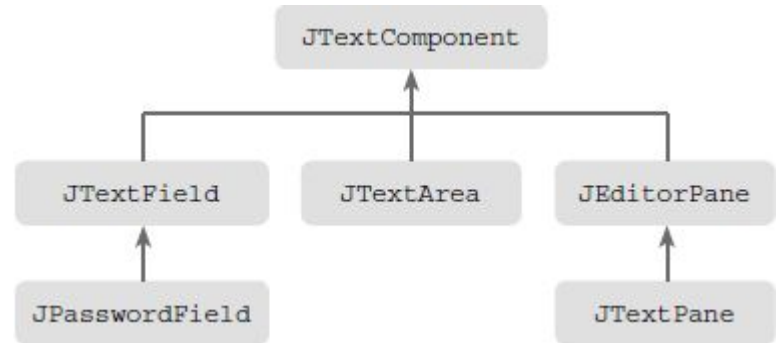
```
comboBox.addItem("Violeta");
```



JTextComponent

Clase abstracta que agrupa comportamiento de todos los input de texto, por ejemplo:

- `getText()`
- `getText(int dspl, int lon)`
- `setText(String txt)`
- `setEditable(boolean bool)`



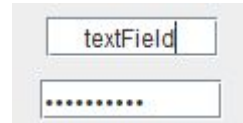
JTextField y JPasswordField

```
textField = new JTextField();  
textField.setHorizontalAlignment(SwingConstants.CENTER);  
textField.setBounds(422, 309, 86, 20);  
contentPane.add(textField);  
textField.setColumns(10);
```

```
passwordField = new JPasswordField();  
passwordField.setBounds(419, 340, 91, 20);  
passwordField.setEchoChar('x');
```

```
contentPane.add(passwordField);
```

```
char[] passText = passwordField.getPassword();
```



JTextArea

Para mostrar muchas líneas de texto.

Se tiene que asociar una barra de Scroll

El constructor puede especificar el texto, la cantidad de columnas y filas.



```
JTextArea textArea = new JTextArea();  
textArea.setBounds(182, 381, 238, 124);  
contentPane.add(textArea);
```

```
textArea.append("Ehh hola!");  
textArea.insert(" alumnos", 8);  
textArea.replaceRange("¡", 0, 4);
```

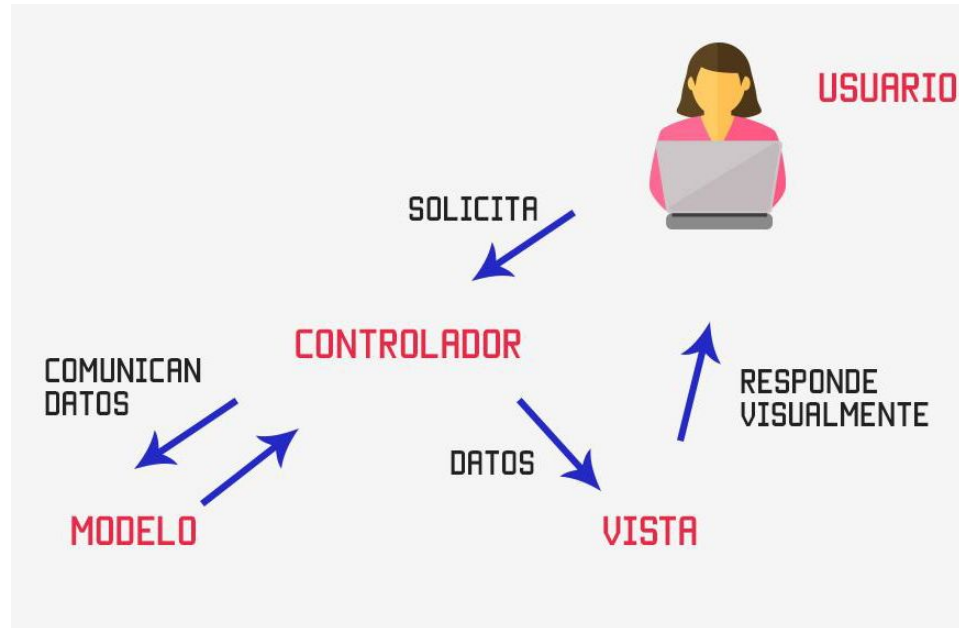
```
JScrollPane scrollPane = new JScrollPane(textArea);  
scrollPane.setBounds(91, 403, 353, 102);  
contentPane.add(scrollPane);
```

mas componentes: <https://docs.oracle.com/javase/tutorial/uiswing/TOC.html>

- cuando se hace click sobre un botón
- cuando se pasa el mouse sobre un campo de texto
- cuando se cierra una ventana
- etc.



Modelo Vista Controlador



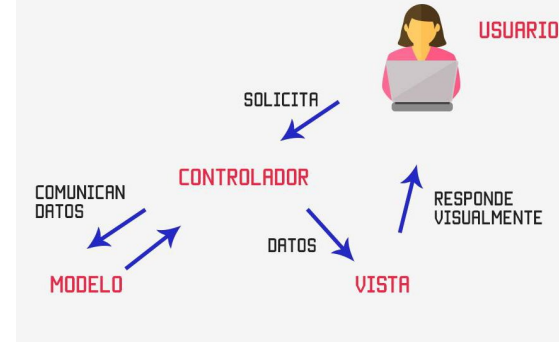
Modelo

Tiene lógica de la aplicación (estados y funcionalidad)

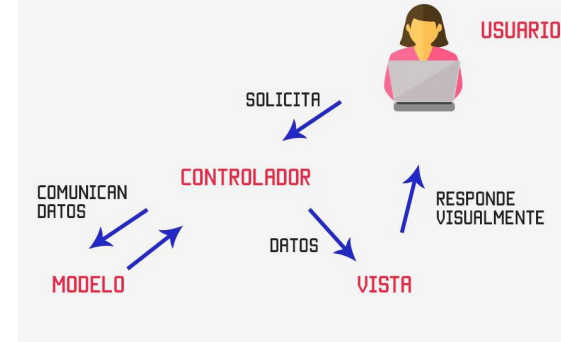
Tiene muy bajo acoplamiento con la vista y el controlador

Toda interacción con el modelo debe hacerse mediante:

- Métodos de consulta
- Métodos de modificación
- Métodos de notificación



Vista



Parte del sistema que administra la visualización y presentación de la información.

El conjunto de clases y objetos son altamente dependiente del dispositivo y de la tecnología de la visualización

Debe conocer bien al modelo.

Pueden existir múltiples vistas para un mismo modelo.

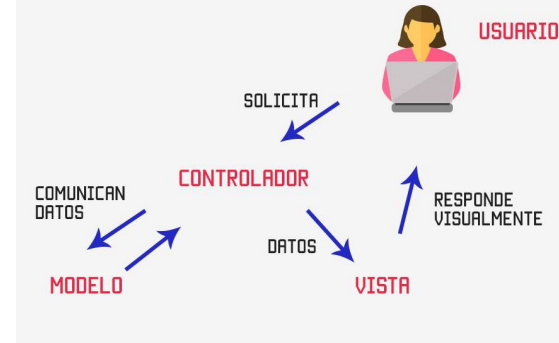
Controlador

Responsable de manejar el comportamiento global de la aplicación.

Su principal tarea consiste en recibir los eventos generados por el usuario y reaccionar frente a ello.

Depende de los dispositivos y mecanismos de interacción con el usuario.

También debe conocer al modelo.





Formas de detectar el evento

Implementado la interfaz adecuada, en este caso ActionListener

```
@Override
public void actionPerformed(ActionEvent e) {
    switch (e.getActionCommand()) {
        case "A":
            System.out.println("accion evento A");
            break;
        case "B":
            System.out.println("accion evento B");
            break;
        default:
            System.out.println("No se que desencadeno el evento");
            break;
    }
}
```



Formas de detectar el evento

Con el uso de lambda

```
private void inicializarVista() {  
    this.getVista().getBtnA().addActionListener(e->this.accionBotonA());  
    getVista().setVisible(true);  
}  
private void accionBotonA() {  
    System.out.println("accion boton A");  
}
```



Formas de detectar el evento

Comparando el componente desencadenante

```
@Override
public void mouseClicked(MouseEvent e) {
    if(e.getComponent().equals(getVista().getLbl1())){
        System.out.println("click en gif desde controlador ");
    }
}
```

Parte 2 - Layouts en JAVA

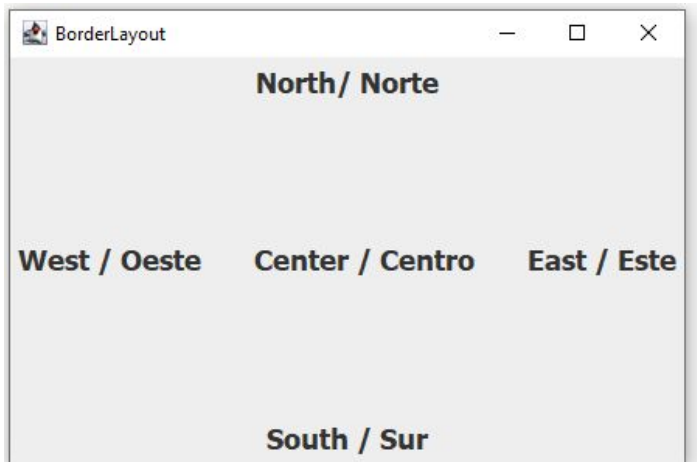


Temario

1. Layout
2. BorderLayout
3. FlowLayout
4. GridLayout
5. BoxLayout
6. CardLayout
7. JTable
8. Menu (JMenuBar, JMenu, JMenuItem, JPopupMenu, JCheckBoxMenuItem, JRadioButtonMenuItem)
9. JFileChooser

BorderLayout

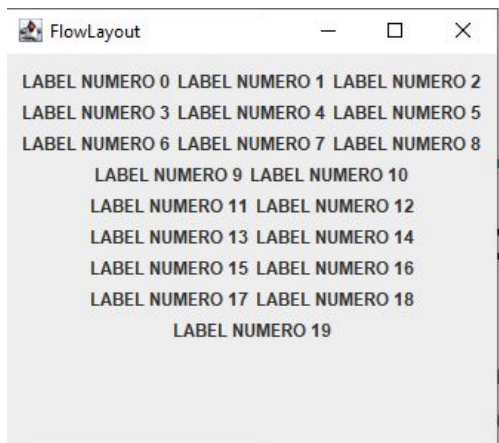
Divide al container en 5 zonas, North, South, East, West y Center



```
public Vista() {  
    setTitle("BorderLayout");  
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    setBounds(100, 100, 450, 300);  
    contentPane = new JPanel();  
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));  
    contentPane.setLayout(new BorderLayout(0, 0));  
    setContentPane(contentPane);  
  
    JLabel lblNewLabel = new JLabel("North/ Norte");  
    lblNewLabel.setFont(new Font("Tahoma", Font.BOLD, 18));  
    lblNewLabel.setHorizontalAlignment(SwingConstants.CENTER);  
    contentPane.add(lblNewLabel, BorderLayout.NORTH);  
  
    JLabel lblNewLabel_1 = new JLabel("West / Oeste");  
    lblNewLabel_1.setFont(new Font("Tahoma", Font.BOLD, 18));  
    lblNewLabel_1.setHorizontalAlignment(SwingConstants.CENTER);  
    contentPane.add(lblNewLabel_1, BorderLayout.WEST);  
  
    JLabel lblNewLabel_2 = new JLabel("Center / Centro");  
    lblNewLabel_2.setFont(new Font("Tahoma", Font.BOLD, 18));  
    lblNewLabel_2.setHorizontalAlignment(SwingConstants.CENTER);  
    contentPane.add(lblNewLabel_2, BorderLayout.CENTER);  
  
    JLabel lblNewLabel_3 = new JLabel("East / Este");  
    lblNewLabel_3.setFont(new Font("Tahoma", Font.BOLD, 18));  
    lblNewLabel_3.setHorizontalAlignment(SwingConstants.CENTER);  
    contentPane.add(lblNewLabel_3, BorderLayout.EAST);  
  
    JLabel lblNewLabel_4 = new JLabel("South / Sur");  
    lblNewLabel_4.setFont(new Font("Tahoma", Font.BOLD, 18));  
    lblNewLabel_4.setHorizontalAlignment(SwingConstants.CENTER);  
    contentPane.add(lblNewLabel_4, BorderLayout.SOUTH);  
}
```

FlowLayout

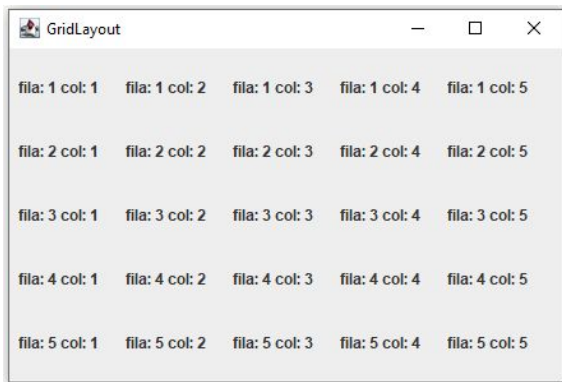
Sitúa los componentes de izquierda a derecha y cuando completa la fila comienza otra.



```
public FlowLayout() {  
    setTitle("FlowLayout");  
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    setBounds(100, 100, 450, 300);  
    contentPane = new JPanel();  
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));  
    setContentPane(contentPane);  
    contentPane.setLayout(new java.awt.FlowLayout(java.awt.FlowLayout.CENTER, 5, 5));  
  
    for (int i = 0; i < 20; i++) {  
        JLabel label = new JLabel("LABEL NUMERO " + i);  
        contentPane.add(label);  
    }  
}
```

GridLayout

Distribuye los elementos en una grilla con filas y columnas. Al agregar componentes agrega primero por columna.



```
public VistaGridLayout() {  
    setTitle("GridLayout");  
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    setBounds(100, 100, 450, 300);  
    contentPane = new JPanel();  
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));  
    setContentPane(contentPane);  
    contentPane.setLayout(new GridLayout(5, 5, 0, 0));  
  
    for (int i = 1; i < 6; i++) {  
        for (int j = 1; j < 6; j++) {  
            contentPane.add(new JLabel("fila: " + i + " col: " + j));  
        }  
    }  
}
```

BoxLayout

Acomoda los elementos en una columna o en una fila, según el parámetro al crear el objeto.

```
public VistaBoxLayout() {  
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    setBounds(100, 100, 450, 300);  
    contentPane = new JPanel();  
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));  
    setContentPane(contentPane);  
    contentPane.setLayout(new BoxLayout(contentPane, BoxLayout.X_AXIS));  
    //contentPane.setLayout(new BoxLayout(contentPane, BoxLayout.Y_AXIS));  
  
    for (int i = 1; i < 25; i++) {  
        contentPane.add(new JLabel(" LABEL N: " + i ));  
    }  
}
```



CardLayout

Se muestra un solo elemento al mismo tiempo de los agregados al panel que contenga este Layout

```
private CardLayout crd = new CardLayout(20, 20);  
panelCard.setLayout(crd);  
panelCard.add(card1);  
panelCard.add(card2);  
crd.previous(panelCard);  
crd.next(panelCard);
```



JTable

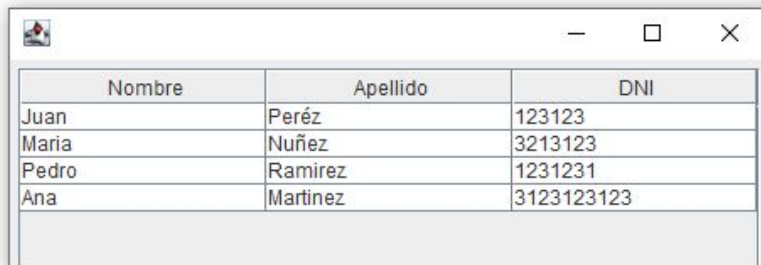
Un componente visual para mostrar datos de manera tabular.

Los datos y encabezado se manejan por su modelo asociado.

Es necesario asociar un JScrollPane.

```
table = new JTable();
table.setModel(new DefaultTableModel(
    new Object[][] {
        {"Juan", "Peréz", "123123"},
        {"Maria", "Nuñez", "3213123"},
        {"Pedro", "Ramirez", "1231231"},
        {"Ana", "Martinez", "3123123123"},
    },
    new String[] {
        "Nombre", "Apellido", "DNI"
    }
));
contentPane.add(table, BorderLayout.CENTER);

JScrollPane scrollPane = new JScrollPane(table);
contentPane.add(scrollPane, BorderLayout.NORTH);
```



Nombre	Apellido	DNI
Juan	Peréz	123123
Maria	Nuñez	3213123
Pedro	Ramirez	1231231
Ana	Martinez	3123123123

JMenu

```
JMenuBar menuBar = new JMenuBar();  
setJMenuBar(menuBar);
```

```
JMenu mnNewMenu = new JMenu("A");  
menuBar.add(mnNewMenu);
```

```
JMenuItem mntmNewMenuItem = new JMenuItem("A-1");  
mnNewMenu.add(mntmNewMenuItem);
```

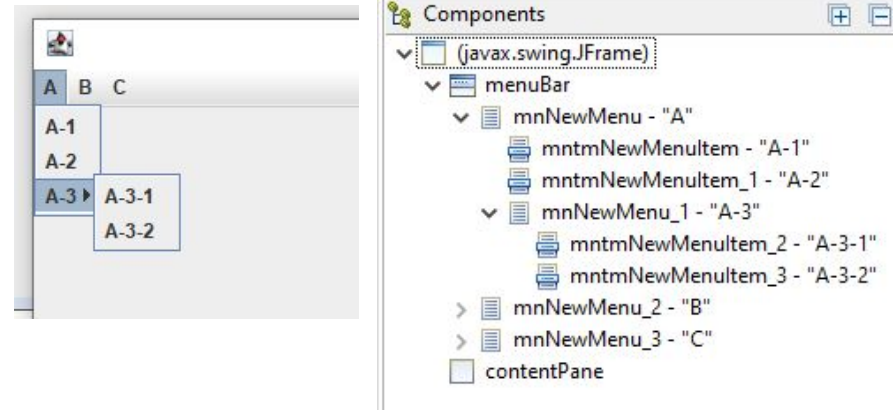
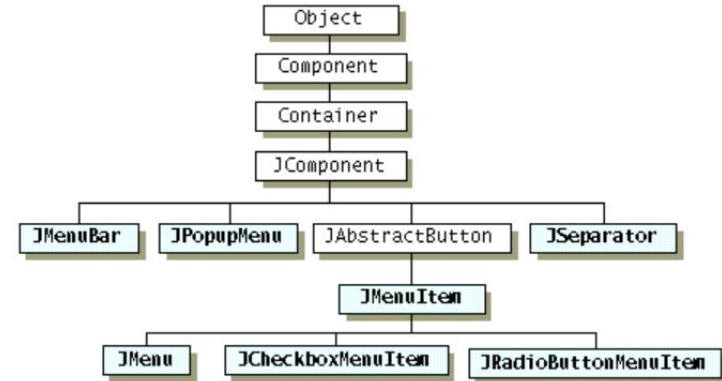
```
JMenuItem mntmNewMenuItem_1 = new JMenuItem("A-2");  
mnNewMenu.add(mntmNewMenuItem_1);
```

```
JMenu mnNewMenu_1 = new JMenu("A-3");  
mnNewMenu.add(mnNewMenu_1);
```

```
JMenuItem mntmNewMenuItem_2 = new JMenuItem("A-3-1");  
mnNewMenu_1.add(mntmNewMenuItem_2);
```

```
JMenuItem mntmNewMenuItem_3 = new JMenuItem("A-3-2");  
mnNewMenu_1.add(mntmNewMenuItem_3);
```

```
JMenu mnNewMenu_2 = new JMenu("B");
```



JFileChooser

Permite abrir un Dialog para seleccionar archivos.

```
JFileChooser fileChooser = new JFileChooser();  
fileChooser.showOpenDialog(contentPane);  
File file = fileChooser.getSelectedFile();  
System.out.println("Archivo seleccionado: " + file);  
System.out.println("Path: " + fileChooser.getCurrentDirectory());
```

