UNIVERSIDAD SANTO TOMÁS

PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA

SECCIONAL TUNJA

VIGILADA MINEDUCACIÓN - SNIES 1732

Acreditación Institucional Internacional

OTORGADA POR EL IAC CINDA ACUERDO 55 DEL 8 DE MAYO-VIGENCIA 5 AÑOS

ACREDITACIÓN INSTITUCIONAL DE ALTA CALIDAD MULTICAMPUS

Res. MEN No. 03455 del 23 de enero de 2020

Vigencia por seis años

QS STARS RATED FOR EXCELLENCE

ISO 9001 Icontec SC4259-1

CERTIFIED IQNet MANAGEMENT SYSTEM

UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
SECCIONAL TUNJA
VIGILADA MINEDUCACIÓN - SNIES 1732

Faculty: Systems engineer
Course: Introduction of Programming
Topic: **Interfaz gráfica en JAVA -GUI (Graphical User Interface)**

Socializer: Luis Fernando Castellanos Guarín
Email: Luis.castellanosg@usantoto.edu.co
Phone: 321-4582098

Topics

Interfaz gráfica en JAVA.

- Acerca de /About
- Ejemplos / examples
- Ejercicios / Exercises

¡Siempre hacia lo alto!

Interfaz Gráfica GUI

Hasta ahora hemos desarrollado programas que usan **la consola** para interactuar con el usuario.

Esa forma de interfaz de usuario **es muy simple** y nos ha permitido centrarnos en todo aquello que tiene que ver tan sólo con la programación del lenguaje Java, sin tener que tratar al mismo tiempo con ventanas, botones y otros elementos similares.

Las interfaces gráficas de usuario (GUI) ofrecen al usuario **ventanas**, **cuadros de diálogo**, **barras de herramientas**, **botones**, **listas desplegables** y muchos otros elementos con los que ya estamos muy acostumbrados a tratar.

Las aplicaciones son conducidas por eventos y se desarrollan haciendo uso de las clases que para ello nos ofrece la API de Java.

Interfaz Gráfica GUI

Llamamos Interfaz Gráfica **GUI** (Graphical User Interface) al conjunto de componentes gráficos que posibilitan la interacción entre el usuario y la aplicación. Es decir ventnas, botones, combos, listas, cajas de diálogo, campos de texto, etc.

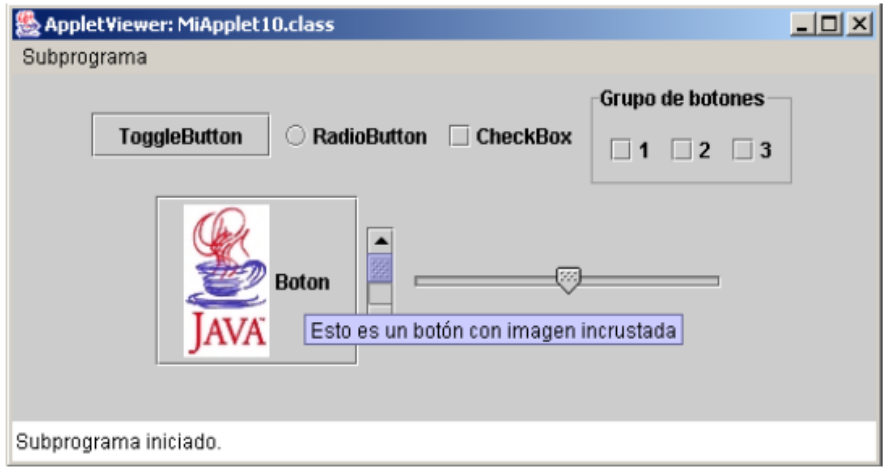
Primero tenemos que diseñar la aplicación, programarla y por último los eventos que se generan a medida que el usuario interactúa con la Interfaz.

Los componentes son objetos de las clases que heredan de la clase base componente como **Button, List, TextField, TextArea, Label,** etc.

En una **GUI** los componentes son contenidos en Contenedores o **containers**. Un **Container** es un objeto cuya clase hereda de Container(clase que a su vez es subclase de **Component**) y tiene la responsabilidad de contener Componentes.

Generalmente una GUI se monta sobre un **Frame**. Esté será el Container principal que contendrá a los componentes de la Interfaz Gráfica, un Container podría contener a otros containers.

Interfaz gráfica (GUI) VS Console User Interface (CUI)



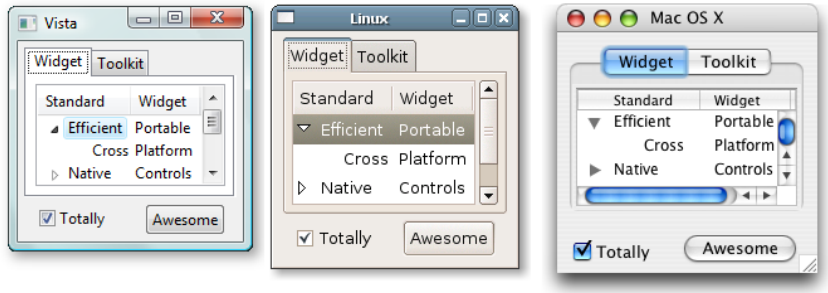
- GUI, es más lenta y compleja de realizar.
- Requiere una base de formación y un conocimiento para cada tipo de interfaz gráfica.
- Aporta una experiencia más visual al usuario.

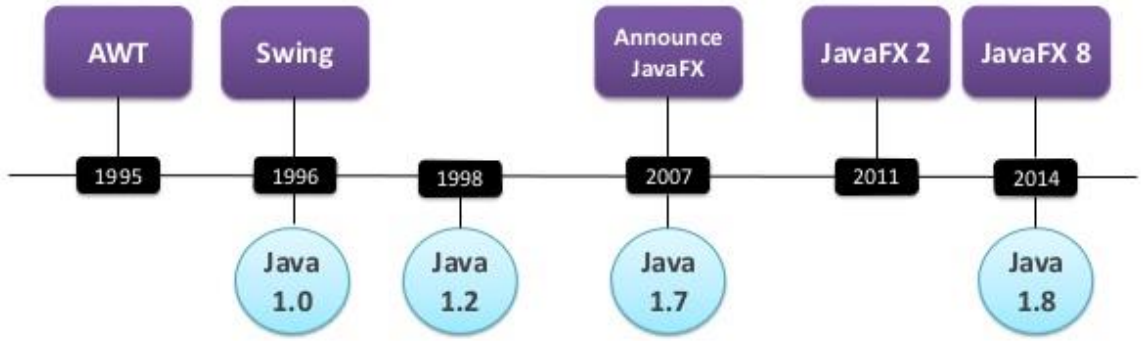
- CUI, es rápida y sencilla de realizar.
- No requiere grandes conocimientos.
- El principal inconveniente de esta interfaz es la mala experiencia de usuario.

Clasificación de las principales interfaces gráficas (GUI) de Java

Las principales librerías para crear interfaces gráficas en JAVA son:

AWT: significa **Abstract Window Toolkit** y la podríamos bautizar como la «**old school**», la vieja escuela de las interfaces gráficas. Pese a ser la más antigua, también es la más completa. Y se centra en el sistema operativo (SO) para dibujar gráficos, por lo que cada SO o plataforma tendrá su propia GUI . Esto, significa que se verá diferente o tendrá un aspecto distinto en cada una de las plataformas. Como por ejemplo:



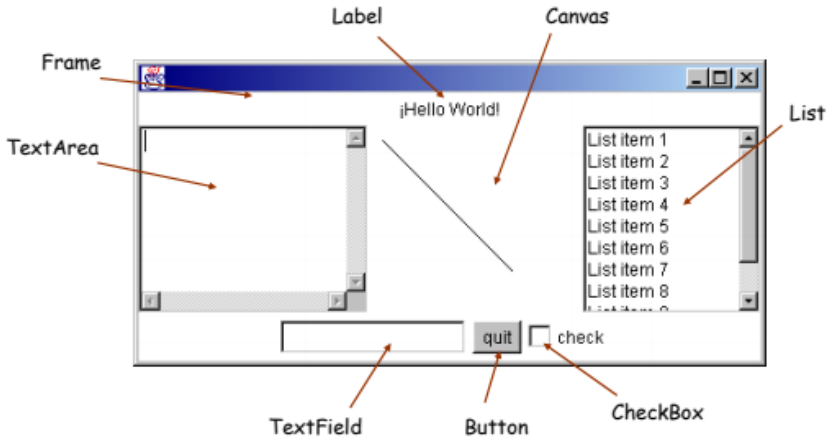


Swing: extiende de la librería gráfica AWT y proporciona un conjunto de componentes bastante «ligero». Y que trata de tener el mismo comportamiento independientemente del SO o plataforma en el que se ejecuta.

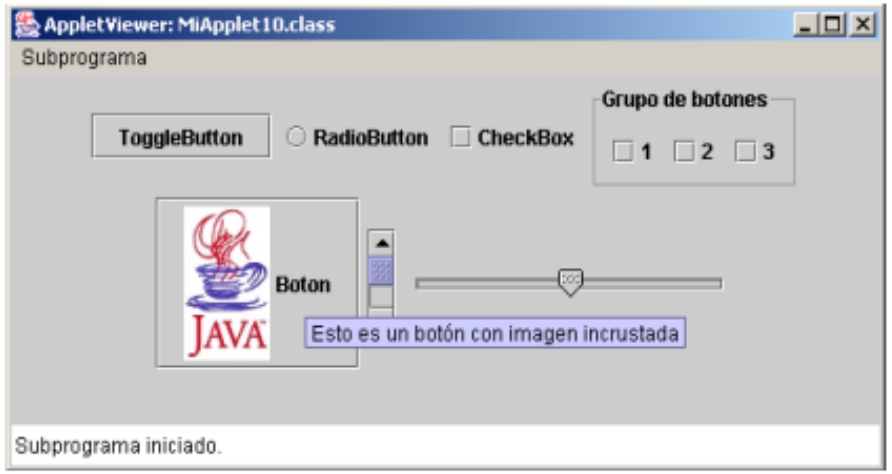
JavaFX: se lanzó bajo el concepto de **RIA** (Rich Internet Applications) que tiene como finalidad crear aplicaciones para webs, tablets, tvs… Muy «**parecidas**» a las de escritorio

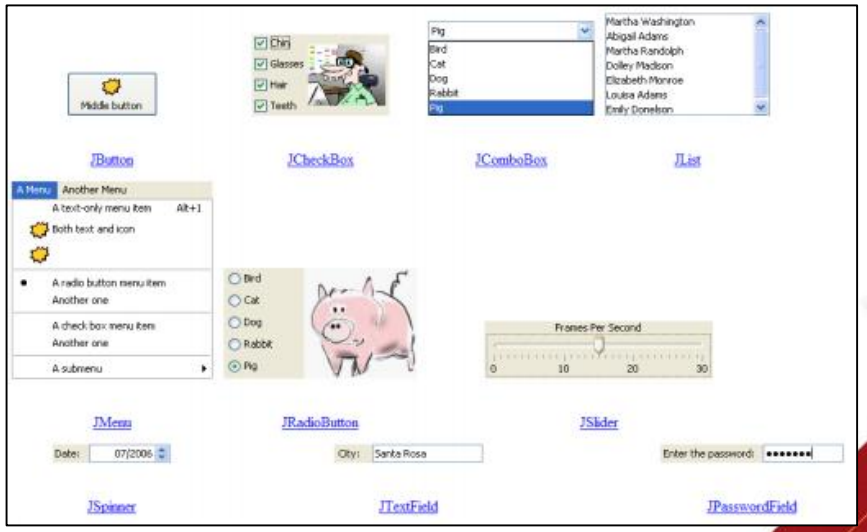
Algunos componentes por GUI

Componentes de AWT



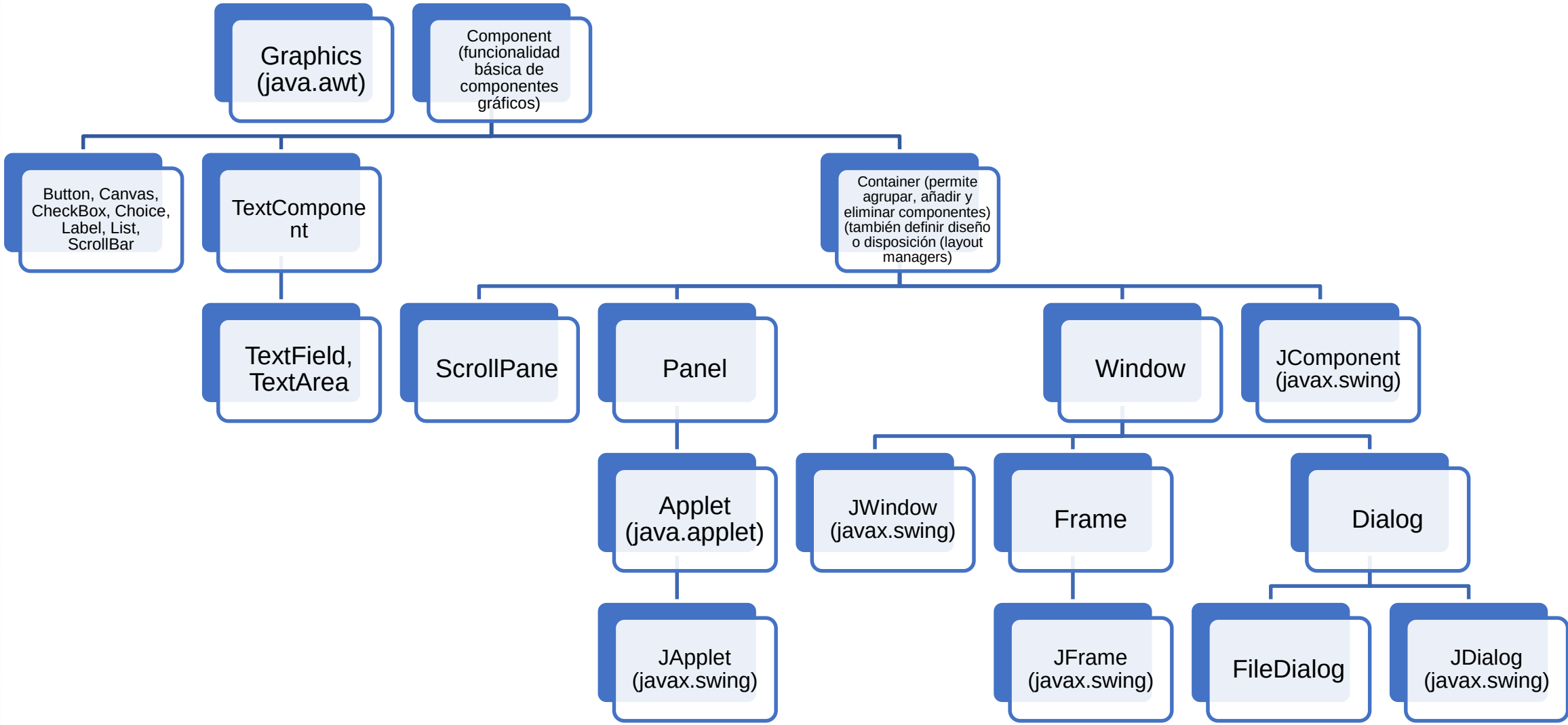
Componentes de Swing





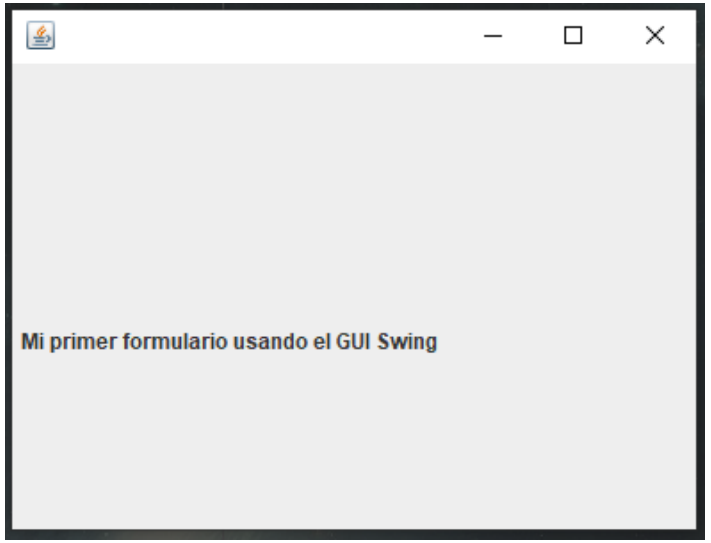
¡Siempre hacia lo alto!

Jerarquías de componentes GUI



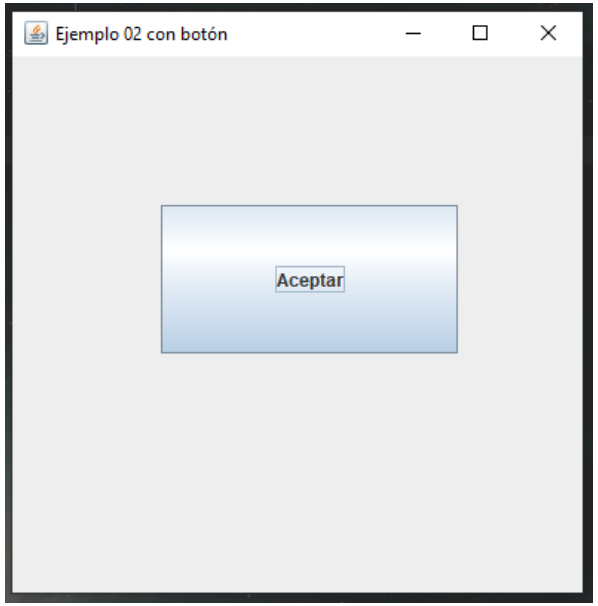
Sintaxis

```java
package usta.sistemas;

import javax.swing.*;

public class Formulario01 extends JFrame{
    // Constantes y componentes (objetos)
    private JLabel label1;

    public Formulario01(){
        // Configurar Componentes ;
        label1 = new JLabel("Mi primer formulario usando el GUI Swing");
        label1.setBounds(5,5,300,300);
        add(label1);
        setLayout(null);            //para ubicar cosas en el formulario (x,y)
        setVisible(true);           //para que se haga visible
        setBounds(0,0,400,300);     //para que tenga 400px de ancho y 300px de alto
        setLocationRelativeTo(null);
        // Configurar Manejadores Eventos ;
        // Terminar la aplicación al cerrar la ventana
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public static void main(String args[]){
        Formulario01 formulario = new Formulario01();
    }
}
```



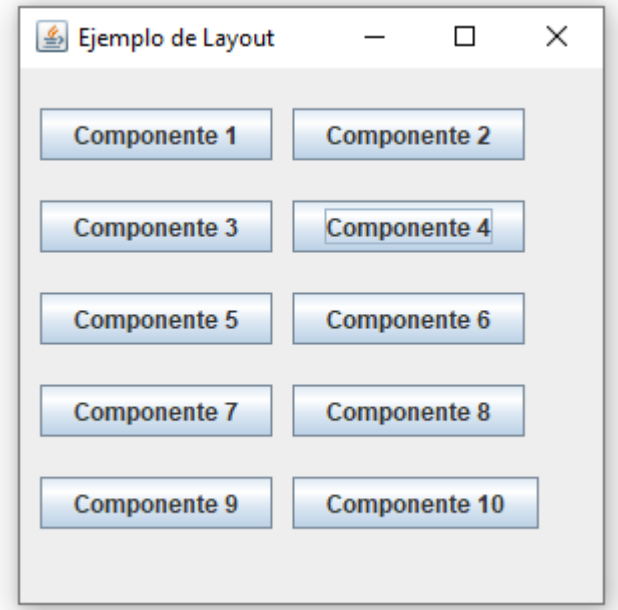
Sintaxis – use of button

```java
package usta.sistemas;

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

public class formulario02 extends JFrame {

    private Container panel;
    private JButton miboton;

    public formulario02() {
        super("Ejemplo 02 con botón");
        // Configurar componentes ;
        miboton = new JButton("Aceptar");
        miboton.setBounds(100,100,200,100);
        panel = getContentPane();
        panel.add(miboton);
        setLayout(null);
        setVisible(true);
        setSize(250,150);
        setBounds(0,0,400,400);      //para que tenga 400px de ancho y 300px de alto
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public static void main(String args[]) {
        formulario02 aplicacion = new formulario02();
    }
}
```



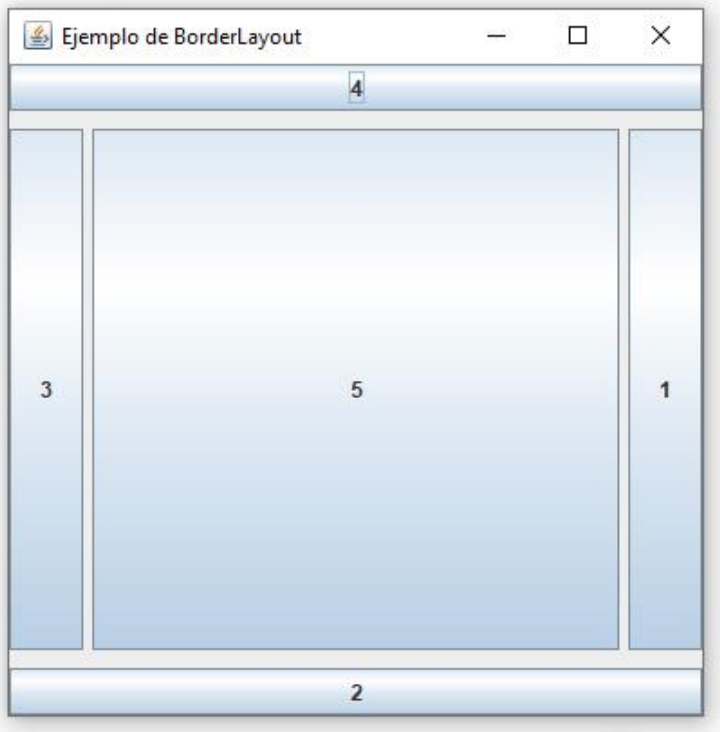
Sintaxis – use layouts

```
package usta.sistemas;

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

public class Formulario03 extends JFrame {
    public Formulario03() {
        super("Ejemplo de uso de Layout");
        // Configurar componentes ;
        // Configurar layout ;
        setLayout(new FlowLayout(FlowLayout.LEFT, 10, 20));
        //agregar 10 nuevos botones al layout
        for (int i = 1; i <= 10; i++)
            add(new JButton("Componente " + i));

        setSize(400, 400);//pack();
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public static void main(String args[]) {
        Formulario03 aplicacion = new Formulario03();
    }
}
```



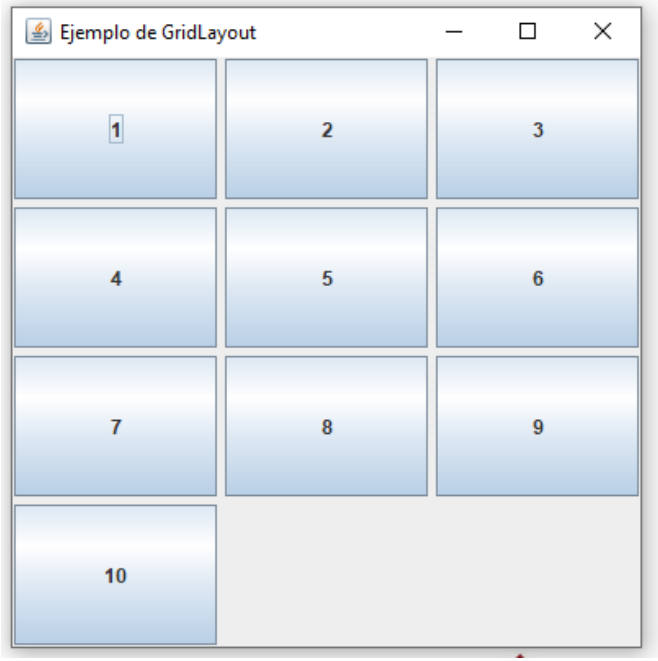
Sintaxis – use BorderLayout

```
package usta.sistemas;
import javax.swing.*;
import java.awt.*;

public class Formulario04 extends JFrame {

    public Formulario04() {
        super("Ejemplo de BorderLayout");
        // BorderLayout
        setLayout(new BorderLayout(5, 10));
        add(new JButton("1"), BorderLayout.EAST);
        add(new JButton("2"), BorderLayout.SOUTH);
        add(new JButton("3"), BorderLayout.WEST);
        add(new JButton("4"), BorderLayout.NORTH);
        add(new JButton("5"), BorderLayout.CENTER);
        setSize(400, 400);
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        Formulario04 aplicacion = new Formulario04();
    }
}
```



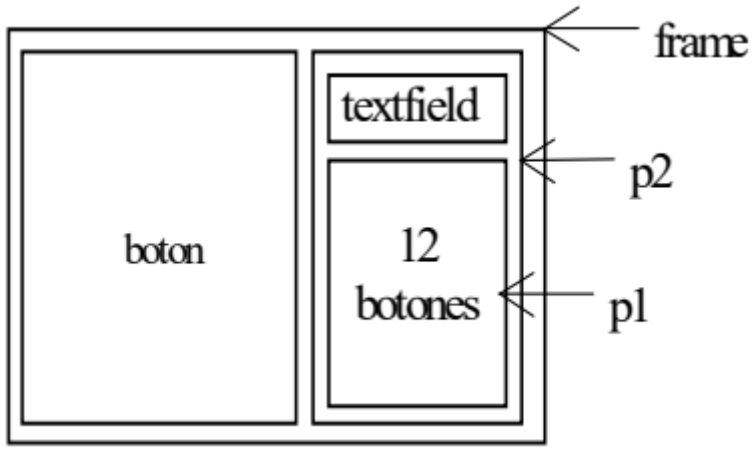
Sintaxis – use GridLayout

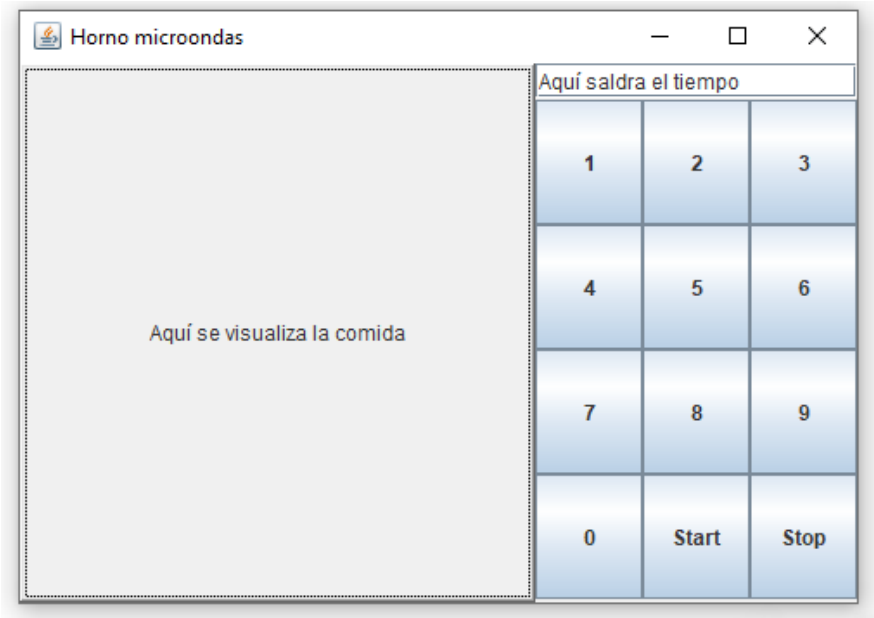
```
package usta.sistemas;

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

public class Formulario05 extends JFrame {

    public Formulario05() {
        super("Ejemplo de GridLayout");
        setLayout(new GridLayout(4, 3, 5, 5));

        for(int i = 1; i <= 10; i++)
            add(new JButton(Integer.toString(i)));
        setSize(400,400);
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        Formulario05 aplicacion = new Formulario05();
    }
}
```



Usando paneles y layout

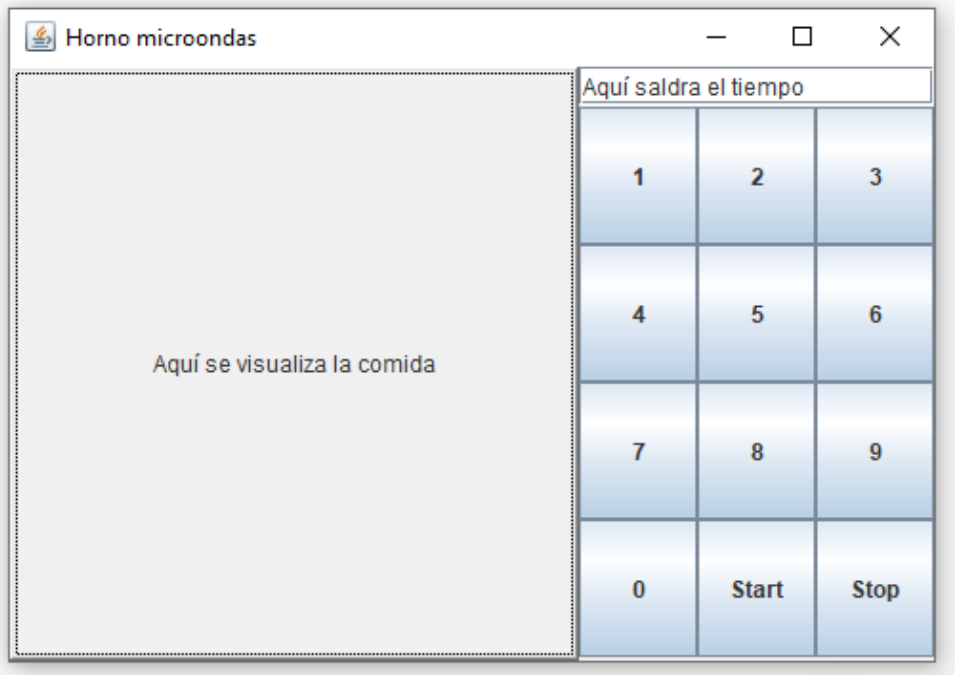
Los paneles actúan como pequeños contenedores para agrupar componentes.
Colocamos los componentes en paneles y los paneles en el **frame** o incluso en otros paneles.





Usando paneles y layout

```
package usta.sistemas;
import javax.swing.*;
import java.awt.*;

public class Formulario06 extends JFrame {

    public Formulario06()  {
        setTitle("Horno microondas");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
        // Creamos el panel p1 para los 10 botones de numeros + star y stop
        JPanel p1 = new JPanel();
        p1.setLayout(new GridLayout(4, 3));
        // Add buttons to the panel
        for (int i = 1; i <= 9; i++) {
            p1.add(new JButton("" + i));
        }
        p1.add(new JButton("" + 0)); //agregamos el boton de 0 al final
        p1.add(new JButton("Start"));
        p1.add(new JButton("Stop"));
        //creamos el segun panel P2 para P1 y el textfield
        JPanel p2 = new JPanel();
        p2.setLayout(new BorderLayout());
        p2.add(new JTextField("Aquí saldra el tiempo"),BorderLayout.NORTH);
        //creamos un borderlayout  donde pondremos un boton y los demas paneles
        p2.add(p1, BorderLayout.CENTER);
        // Add p2 and a button to the frame
        add(p2, BorderLayout.EAST);
        add(new Button("Aquí se visualiza la comida"),BorderLayout.CENTER);
        setSize(500, 350);
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        Formulario06 aplicacion = new Formulario06();
    }
}
```



Interacción con el usuario

Para que el usuario pueda tener interacción con la aplicación (dar clic, escribir, etc), se deben dar acciones a los componentes (**ActionEvent**):

- El usuario pulsa un botón.
- El usuario termina de introducir un texto en un campo y presiona Intro(enter).
- El usuario selecciona un elemento de una lista pulsando el preferido (o de un menú).
- Pulsa o suelta botones del ratón (**MouseEvent**).
- Minimiza, cierra o manipula una ventana (**WindowEvent**).
- Escribe con el teclado (**KeyEvent**).
- Descubre porciones de ventanas (**PaintEvent**).

Interacción con el usuario

Cuando el usuario de un programa o applet (JAVA) mueve el ratón, presiona un pulsador o pulsa una tecla, genera un evento (**actionEvent**).

Los eventos son objetos de ciertas clases. Normalmente un objeto de alguna subclase de **EventObject** que indica:

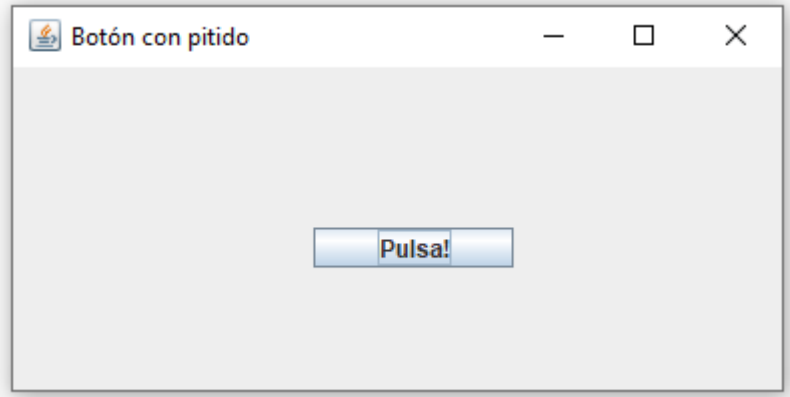
- El elemento que accionó el usuario.
- La identificación del evento que indica la naturaleza del evento.
- La posición del ratón en el momento de la interacción.
- Teclas adicionales pulsadas por el usuario, como la tecla Control, la tecla de Cambio a mayúsculas, etcétera.

Acciones del usuario

Acción	Objeto de origen	Tipo de evento
Pulsar un botón	JButton	ActionEvent
Cambio del texto	JTextComponent	TextEvent
Pulsar Intro en un campo de texto	JTextField	ActionEvent
Selección de un nuevo elemento	JCombobox	ItemEvent ActionEvent
Selección de elemento(s)	JList	ListSelection-Event
Pulsar una casilla de verificación	JCheckBox	ItemEvent ActionEvent
Pulsar un botón de radio	JRadioButton	ItemEvent ActionEvent
Selección de una opción de menú	JMenuItem	ActionEvent
Mover la barra de desplazamiento	JScrollBar	AdjustmentEvent
Abrir, cerrar, minimizar, maximizar o cerrar la ventana	JWindow	WindowEvent

Acciones del usuario - ejemplo

```java
package usta.sistemas;

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

public class Formulario07 extends JFrame {
    JButton boton;
    public Formulario07() {
        setTitle("Botón con pitido");
        boton = new JButton("Pulsa!");
        boton.setBounds(150,80,100,20);
        add(boton);
        boton.addActionListener(new OyenteBoton());
        setLayout(null);
        setBounds(0,0,400,200);      //400px de ancho y 300px de alto
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
    public static void main(String[] args) {
        Formulario07 aplicacion = new Formulario07();
    }
}
class OyenteBoton implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        Toolkit.getDefaultToolkit().beep();
    }
}
```



To practice!

¡Siempre hacia lo alto!

P2Txx: Gui_principal

create a JAVA software that:

Create a menu at the top of the window with the following options:

- Students.
- Subjects
- Exit

In the center of the advantage, the name of the software "Sac JAVA version 1.0" should appear

In the lower part of the window, the creator of the software should appear: name and email.

Construir un programa en JAVA que,

Cree un menú en la parte superior de la ventana con las siguientes opciones:

- Estudiantes.
- Asignaturas
- Salir

En el centro de la ventaja debe aparece el nombre del software “Sac JAVA versión 1.0”

En la parte inferior de la ventana debe salir quien es en creador del software: nombre y correo electrónico.

P2Txx: contacts_txt

create a JAVA software that:

View a form with the following fields:

- Full name
- Profession
- Address
- Email

And a save button (the button will only work if you have all the fields with information), the data will be saved in a text file called contacts.txt.

Construir un programa en JAVA que,

Visualice un formulario con los siguientes campos:

- Nombre completo
- Profesión
- Dirección
- Correo electrónico

Y un botón de guardar (el botón solo funcionara si tiene todos los campos con información), los datos serán guardados en un archivo de texto denominado contactos.txt.

P2Txx: even_numbers_text

Create a JAVA software that:

Show the user a menu with the following options:

1. Create an array with the even numbers existing between two numbers (the user must indicate what those numbers are), the result of the array must be saved in a file whose name will be chosen by the user.
2. Display the contents of the created text file on the screen.
3. Exit the program.

Construir un programa en JAVA que,

Muestre al usuario un menú con las siguientes opciones:

1. Crear un array con los números pares existentes entre dos números (el usuario debe indicar cuales son esos números), el resultado del array debe ser guardado en un archivo cuyo nombre lo elegirá el usuario.
2. Mostrar por pantalla el contenido del archivo de texto creado.
3. Salir del Programa.


PROYECTO
CRUD
PERSISTENCIA
MATRICES
OPERACIONES ARREGLOS
ARREGLOS
MÉTODOS
CICLOS
CONCEPTOS BÁSICOS


GOOD JOB

UNIVERSIDAD SANTO TOMÁS

PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA

SECCIONAL TUNJA

VIGILADA MINEDUCACIÓN - SNIES 1732

¡Siempre hacia lo alto!

USTATUNJA.EDU.CO

@santotomastunja