

# Guía de Trabajo Autónomo

## Interfaces Gráficas y Eventos en Java Swing

Curso: **ICC490 – Programación Orientada a Objetos**

Profesor: **Dr. Samuel Sepúlveda**

DCI – FICA – UFRO

**Objetivo General:** Desarrollar competencias iniciales en la construcción de interfaces gráficas (GUI) en Java, utilizando **Swing** y el modelo de **programación orientada a eventos**, integrando los principios de la programación orientada a objetos.

### 1 Tabla de Autoevaluación Iterativa del Estudiante

Antes de comenzar con la revisión del contenido y a desarrollar el código fuente, lee atentamente las siguientes instrucciones para realizar el paso a paso de la actividad, y cumplir tanto los aspectos técnicos, como los formativos y pedagógicos.

#### Instrucciones

- Marca con una **X** las tareas que lograste completar en cada iteración (It., It.2 e It.3).
- Puedes realizar hasta 3 intentos para perfeccionar tu aplicación.
- Usa la columna de observaciones para registrar sobre lo que mejoraste o te faltó en cada iteración (It., It.2 e It.3)
- Al final del proceso, plantea una reflexión respecto del trabajo realizado, los objetivos alcanzados y las tareas pendientes.
- Considera que el trabajo está diseñado para ser desarrollado en aprox. 60-75 mins.

**Esta guía deberá ser desarrollada durante la semana en curso, previo a la clase del viernes 5/9, en la cual se aplicarán los conceptos aquí revisados.**

<b>Criterio / Tarea</b>	<b>It.1</b>	<b>It.2</b>	<b>It.3</b>	<b>Observaciones / Mejoras realizadas</b>
Tiempo empleado (mins.)				
Ventana funcional con JFrame y visibilidad	5'			Código añadido, aún no pasa nada.
Tiempo empleado (mins.)				
Componentes añadidos: JTextField, JLabel, JButton	5'			Código añadido, aún no pasa nada.
Evento del botón funcionando correctamente	2'			El botón funciona y se muestra la ventana.
Validación de entrada implementada	10'			
Modularización (métodos o clases separadas)	20'			Me apoyé con ia, me hizo reestructurar el código.
Uso de clase auxiliar (ej. Usuario)	5'			Se crea nueva clase Usuario.java
Soporte para tecla ENTER	5'	5'		
Estilo del código claro y comentado	5'	5'		
Botón limpiar, nueva funcionalidad	8'	10'		
<b>Reflexión final del estudiante:</b>	Se logró con apoyo de IA, sin embargo se nota mucho, el código fue reestructurado casi por completo y se eliminaron cosas mencionadas en la guía, como imports no utilizados y se agregaron constructores.			

## 2 Introducción

A continuación se presenta un resumen de los principales conceptos necesarios para el desarrollo de aplicaciones Java en entorno de escritorio, usando los paquetes y clases de Swing. Para más detalles y resolver dudas técnicas usando las fuentes oficiales de documentación, consulte la Sección 8.

### 2.1 ¿Qué es Java Swing?

Swing es una biblioteca del lenguaje Java para crear interfaces gráficas de usuario. Permite construir componentes como botones, ventanas, campos de texto, entre otros.

## 2.2 ¿Qué es JFrame?

JFrame representa una ventana principal. Es el contenedor base donde se ubican todos los demás componentes.

## 2.3 ¿Qué es un evento?

Un evento es una acción del usuario, como presionar un botón. Java utiliza un modelo basado en listeners (oyentes) que reaccionan a estos eventos.

## 3 Modelo de Eventos en Java

- Usuario interactúa con un componente. ■

El componente genera un evento.

- Un objeto oyente (Listener) lo recibe. ■

Se ejecuta el código asociado.

La secuencia anterior del modelo de manejo de eventos en Java Swing, se puede observar de manera gráfica en la Figura 1.

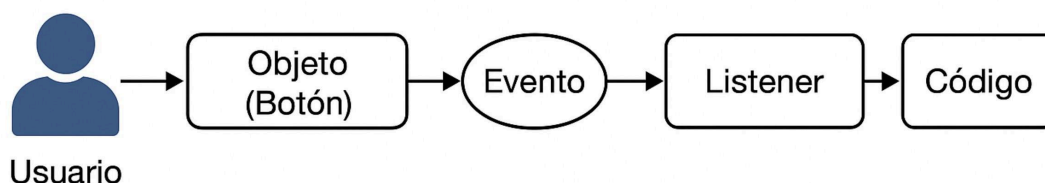


Figura 1: Flujo de interacción: el Usuario hace clic en el **Botón** (objeto), lo que genera un **Evento**, capturado por un **Listener**, que finalmente invoca el **código** asociado.

## 4 Creación del Proyecto en IntelliJ IDEA

### Paso 1: Crear nuevo proyecto

1. Abre IntelliJ IDEA.
2. Selecciona New Project.
3. Elige **Java** como tipo de proyecto.

4. Asegúrate de tener un SDK configurado (versión 8 o superior).
5. Haz clic en Next, luego en Finish.

## Paso 2: Crear la clase principal

1. Haz clic derecho en el paquete principal (por defecto src).
2. Selecciona New → Java Class.
3. Nómbrala VentanaSaludo.
4. Asegúrate de que sea public class VentanaSaludo con un método main.

### ¡OJO! Paquetes necesarios

Para que el proyecto compile correctamente, debes importar estos paquetes al comienzo del archivo:

```
import javax . swing . * ;           // Para JFrame , JButton , JLabel ,
    JTextField
import java . awt . event . * ; Key   // Para manejo de eventos como
    Listener
```

## Paso 3: Estructura mínima de la clase

```
public class Ventana Saludo {
    public static void main ( String [] args ) {
        ...
    }
}
```

## 5 Ejercicio Base: Aplicación de Saludo

A partir del proyecto creado en el paso anterior, siga las siguientes instrucciones.

### Descripción

Crear una aplicación que reciba un nombre en un campo de texto y muestre un saludo personalizado al presionar un botón.

### 5.1 Paso 1: Crear la ventana

```
JFrame ventana = new JFrame ( " App _ de _ Saludo _ ICC490 " );
ventana . setSize ( 400 , 200 );
ventana . setDefaultCloseOperation ( JFrame . EXIT_ON_CLOSE ); ventana .
setLayout( null );
```

Esta sección configura la ventana principal. Se utiliza 'setLayout(null)' para usar posicionamiento absoluto.

**OJO!** La ventana hasta aquí NO hace nada, pues no se han definido objetos que puedan “escuchar” y gestionar los eventos del usuario.

## Vista previa



## 5.2 Paso 2: Agregar componentes visuales

```
JTextField campo Texto = new JTextField (); campo Texto .  
setBounds (50 , 30 , 200 , 25);  
  
JButton boton Saludar = new JButton (" Saludar"); boton  
Saludar. setBounds (270 , 30 , 100 , 25);  
  
JLabel etiqueta Saludo = new JLabel(""); etiqueta Saludo .  
setBounds (50 , 80 , 300 , 25);
```

Aquí se agregan los componentes: un campo de entrada, un botón y una etiqueta vacía.

**OJO!** La ventana hasta aquí tampoco hace nada, pues no se han definido objetos que puedan “escuchar” y gestionar los eventos del usuario.

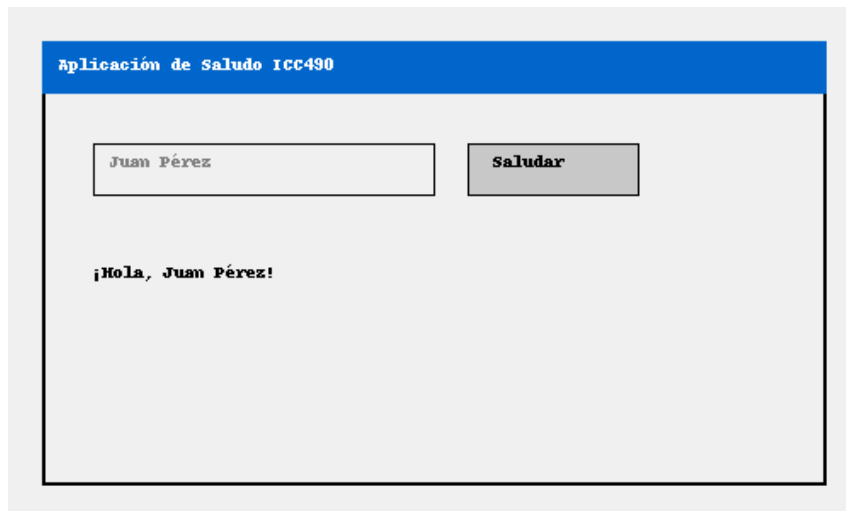
## 5.3 Paso 3: Evento del botón

```
boton Saludar. add ActionListener ( e -> { String  
    nombre = campo Texto . getText ();  
    etiqueta Saludo . setText(" Hola : " + nombre );  
});
```

El método 'addActionListener' permite definir una acción cuando el usuario presiona el botón.

**OJO!** Aquí “Sí pasa algo” en la ventana, pues se vincula el método `addActionListener()` con el objeto `botonSaludar`, y así gestionar el evento e para que “algo pase” cuando se detecta el evento de tipo “click” en el botón.

## Vista previa al ejecutar



## 5.4 Código Base

Una primera aproximación al código completo de la `VentanaSaludo` sería el que se muestra a continuación.

```
import javax . swing . * ;

public class Ventana Saludo {
    public static void main ( String [] args ) {
        JFrame ventana = new JFrame ( " App _de _Saludo _ICC490 " ); ventana .
        setSize ( 400 , 200 );
        ventana . setDefaultCloseOperation ( JFrame . EXIT_ON_CLOSE ); ventana .
        setLayout ( null );

        JTextField campo Texto = new JTextField (); campo
        Texto . setBounds ( 50 , 30 , 200 , 25 );

        JButton boton Saludar = new JButton ( " Saludar " ); boton
        Saludar . setBounds ( 270 , 30 , 100 , 25 );

        JLabel etiqueta Saludo = new JLabel ( "" ); etiqueta Saludo .
        setBounds ( 50 , 80 , 300 , 25 );

        boton Saludar . add Action Listener ( e -> { String
            nombre = campo Texto . getText ();
            etiqueta Saludo . setText ( " Hola , _ " + nombre );
        } );

        ventana . add ( campo Texto ); ventana .
        add ( boton Saludar ); ventana . add (
        etiqueta Saludo );
    }
}
```

```
        ventana . setLocation Relative To ( null ); ventana .  
        setVisible ( true );  
    }  
}
```

### Actividad 2: Reflexionar sobre el código generado

A partir del código fuente anterior del listado anterior, reflexione sobre el uso de las buenas prácticas de programación y la creación-uso de métodos.

- ¿El método main(...) de la clase VentanaSaludo realiza una sola tarea? Sí/No  
¿Por qué?
- Si la respuesta anterior fue No ¿Qué cambios propone en la estructura de su código fuente? Refactorice su código y ejecute nuevamente, evalúe sus resultados

## 6 Extensiones y Validaciones

Con el fin de mejorar tanto el código, como los aspectos visuales de su solución, incorpore paulatinamente las siguientes modificaciones en su código fuente.

### 6.1 Aspectos visuales

#### Actividad 3: Personalización Visual

Para experimentar con las diferentes opciones visuales de la ventana y demás objetos se le pide:

- modificar color de fondo
- modificar los tipos/tamaño de fuentes
- tamaño de ventana, etc.

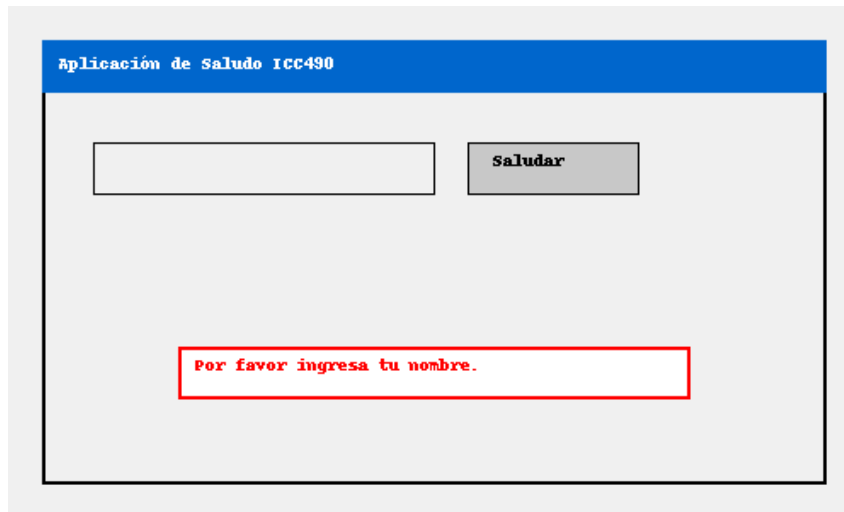
### 6.2 Agregar validación de entrada básica

#### Actividad 4: Validación de Entrada

Mostrar un mensaje si el usuario no escribe nada.

```
if ( nombre . trim (). is Empty () ) {  
    JOption Pane . show Message Dialog ( null , " Por_favor_ingresa _tu_nombre .");  
} else {  
    etiqueta Saludo . setText(" Hola : " + nombre );  
}
```

## Vista previa con mensaje de validación



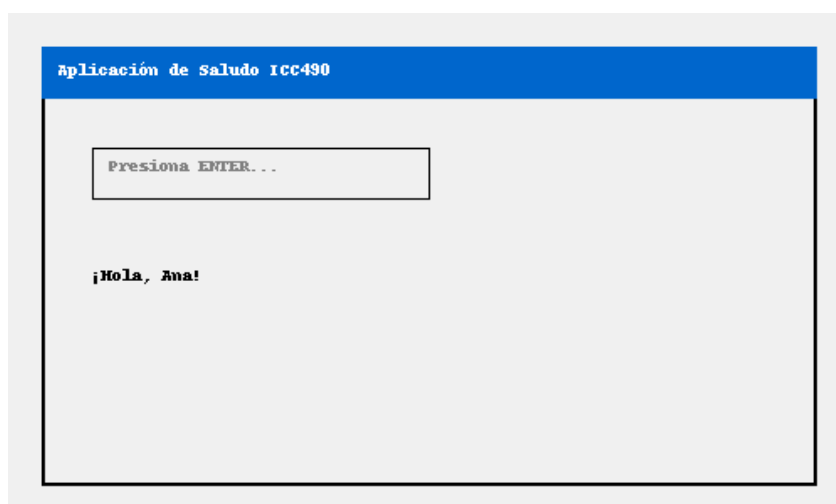
## 6.3 Manejo de tecla ENTER

### Actividad 5: Evento con Tecla ENTER

Permitir saludar presionando la tecla "Enter", así la Ventana tendría un comportamiento análogo al hacer click en el botón "Saludar".

```
campo Texto . add Key Listener ( new Key Adapter () { public  
    void key Pressed ( Key Event e ) {  
        if ( e. getKey Code () == Key Event. VK_ ENTER ) {  
            boton Saludar. do Click ();  
        }  
    }  
});
```

## Vista previa presionando ENTER





## 7 Actividades Sugeridas

### Actividad 6: Uso de Clases Auxiliares

Crear clase Usuario con método getSaludo().

### Actividad 7: Modularización

Separar lógica en métodos y clases que extiendan JFrame.

### Actividad 8: Nuevo objeto y funcionalidad

Agregar un nuevo objeto de tipo JButton que tenga el siguiente comportamiento: al recibir un click del usuario debe limpiar el contenido del JTextField en su JFrame.

## 8 Referencias y bibliografía

Java & IntelliJ IDEA <https://dev.java/learn/intellij-idea/>

Starting with Swing, Oracle Java Doc <https://docs.oracle.com/javase/tutorial/uiswing/start/index.html>

API Java, Swing package, Oracle <https://docs.oracle.com/en/java/javase/24/docs/api/java.desktop/javax/swing/package-summary.html>