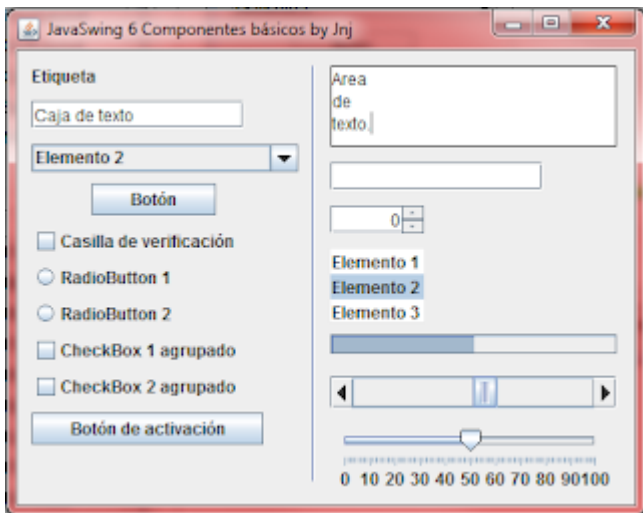


# Java Swing 6: Los componentes básicos

2013-01-01 - Categorías: [Java](#) / [Java Swing](#)

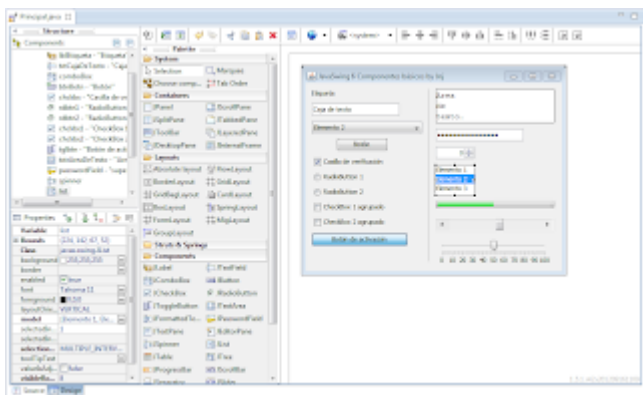


¡¡Feliz Año Nuevo!!

Hace ya un tiempo que no escribo, así que voy a tratar de que éste año sea mejor que el anterior y comienzo por avanzar en mis investigaciones con Java, así que os dejo éste post por si a alguien le sirve...

Sigo escudriñando los componentes de Swing, a ver cómo son **los más simples**, que no faltan en cualquier programa. Si lo que buscas es aprender rápido lo necesario para usarlos, éste es tu post xD He utilizado la **versión 7 del JDK y Eclipse Juno**.

Si vas a seguir éste tutorial desde cero, lo suyo sería que hicieras un nuevo proyecto, crearas un nuevo JFrame con el generador de código de **Eclipse** y fueras al **Swing Designer**. Es decir, entras al modo de diseño para editarlo intentando hacer algo como la imagen de arriba, usando las herramientas que ves aquí abajo:



Queda cada vez menos componentes de los presentados en el post 'Java Swing 1'. Así que vamos al grano.

## JLabel

Es una etiqueta. Con el diseñador ponemos el texto en el valor text, se puede cambiar en el código con la función `nombreEtiqueta.setText(«Cadena»);` aunque en el código fuente del ejemplo se ha puesto el texto en el constructor al usar `JLabel lblEtiqueta = new JLabel(«Etiqueta»);`

## JTextField

**Es una caja de texto.** Se usa de igual manera que una etiqueta a la hora de programar o diseñar la interfaz, pero su diferencia con la etiqueta es que el usuario puede cambiar el contenido de la caja escribiendo en ella, con la etiqueta no puede hacerlo. Tendremos también la función `nombre.setText(«Cadena»);` anterior. Con `variable = txtCajaDeTexto.getText();` podemos tener el contenido de la caja de texto en la variable. Éste componente sólo admite una línea.

## JComboBox

**Significa cuadro combinado**, como se puede ver en el ejemplo, es una lista de elementos desplegable donde el usuario puede elegir entre las opciones que le demos. La elección la

## Entradas recientes

- [Java: cómo usar las enumeraciones y los EnumMap](#)
- [Lynix: auditoria de seguridad para sistemas basados en UNIX](#)
- [El ABC con Angular](#)
- [Cómo mejorar la calidad del software con PHPSTAN](#)
- [Java Spring Boot: simplificando el código Java con Lombok](#)
- [Symfony: cómo identificar con UUIDs las entidades](#)
- [Java Spring Boot & Angular: cómo automatizar la compilación de un proyecto para desplegar en un servidor](#)
- [Cómo organizar los fuentes PHP con espacios de nombres](#)
- [Java: sincronización con señales entre procesos concurrentes](#)
- [Controlando el acceso exclusivo a memoria compartida en Java](#)

## Lo + visto esta semana

- [Datatables: listando y filtrando tablas de datos con JavaScript](#)
- [Los patrones del diseño software GRASP](#)
- [VHDL: decodificador BCD de 4 bits para display de 7 segmentos](#)
- [VHDL: diseñando una unidad aritmético lógica](#)
- [Datatables con botones para CSV, Excel, PDF e imprimir](#)
- [Sencillo mensaje de aceptar cookies para tu web](#)
- [VHDL: los ladrillos básicos; and, or, not, nand, nor, xor y xnor](#)
- [VHDL: multiplexor de 4 a 1 con selección de 2 bits](#)
- [VHDL: uniendo circuitos, un sumador comple](#)

podemos tener con a función `variable = comboBox.getSelectedIndex();`

## JButton

Es un simple botón, con el que al hacer click programamos lo que necesitemos. En el ejemplo se muestra un diálogo simple con:

```
btnBotn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        JOptionPane.showMessageDialog(frame, "Ésto es un botón simple.");
    }
});
```

Tambien se puede cambiar el texto con el método `.setText(cadena);`

## JCheckBox

Las casillas de verificación, que en ejemplo hay varias. Las podemos usar de una en una, o en grupo, poniéndolas dentro de un grupo, de manera que cuando se selecciona una las demás se des-seleccionan automáticamente.

Para saber si un JCheckBox está 'checkeado' debemos usar la función `nombreCheckBox.isSelected();` que devolverá `true` o `false`.

## JRadioButton

Los botones de radio, JRadioButton para los amigos, se usan de igual manera que los JCheckBox. La diferencia es que tradicionalmente se usan para elegir una entre varias opciones, mientras que los JCheckBox normalmente se usan para elegir en varias opciones si se desean o no, sin que unas excluyan a las otras. Pero en realidad se pueden usar igual dependiendo de si los agrupamos o no. En el ejemplo se han agrupado los botones de radio con el código:

```
ButtonGroup radioGroup = new ButtonGroup();
radioGroup.add(rdbtn1);
radioGroup.add(rdbtn2);
```

Ésto de aquí arriba lo que hace es que si elegimos el rdbtn1 entonces el rdbtn2 se des-selecciona y viceversa. Así de simple ocurre si también agrupamos los JCheckBox, es decir, en los elementos de un grupo sólo estará seleccionado uno.

## JToggleButton

Llamado botón de activación, se usa de nuevo igual que un JCheckBox o un JRadioButton, creo que sobran las explicaciones aquí porque tenemos las mismas funciones y comportamientos, sólo cambia su apariencia por la forma de un botón que se mantiene pulsado cuando está activado.

## JTextArea

Es un área de texto de varias líneas, se usa igual que el JTextField pero con la diferencia de que admite varias líneas. Podemos establecer el número de líneas con `nombre.setRows(n);` y tenemos también las funciones `nombre.getText()` y `nombre.setText()`. En el ejemplo se establece el texto de la forma:

```
txtrAreaDeTexto.setText("Arearnderntexto.");
```

Podemos ver que con `rn` se produce un salto de línea. Ésto nos puede servir en muchos otros sitios para hacer un salto de línea, así que de memoria a aprendérselo.

• [Java: serializando objetos para guardar y recuperar en ficheros](#) [\(editado\)](#).

## Utilidades para WordPress

- [What's going on \(WAF\)](#).
- [The SEO Machine](#)
- [The SEO Workspace](#)
- [Miscellaneous Scripts](#)
- [Looking for the 100 points \(theme\)](#).

### ¡Suscríbete a las novedades!

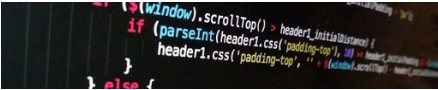
Sólo te enviaré un email cuando haya posts nuevos.

Nombre \*

Email \*

☐ Acepto la [política de privacidad](#).

Suscribeme



# JPasswordField

Es un **área de contraseña**, es decir, un área de texto pero con la diferencia de que los caracteres no se muestran para que el usuario del ordenador de al lado no pueda leer las contraseñas que pones en tu pantalla. La diferencia con un JTextField es que para conseguir la cadena de caracteres que ha puesto el usuario hay que usar la función `nombre.getPassword();`

# JSpinner

**Ésto es una caja donde con un par de botones cambiamos los valores.** Traducido se llama 'hilandero', otra palabra nueva para mi vocabulario, nunca la habia oido ni leído. Por defecto se configura para que recorra los enteros, si le das a la flecha arriba suma uno al valor que haya 1, 2, 3, 4, 5... Si le das abajo resta de la forma 3, 2, 1, 0, -1, -2, -3...

Con `spinner.getValue();` obtenemos el valor elegido. Va a devolver un objeto del tipo correspondiente, puede ser una fecha o una lista de elementos.

Para configurarlo hay que acceder en el modo Design a la propiedad **model**. Internamente utiliza la función `setModel()` para establecer una lista de elementos que utiliza el componente. Éstos vectores son habituales en varios componentes Swing y nos van a facilitar la vida. Por resumirlos, los **ListModel** son **vectores**, y dependiendo del tipo de componente los elementos del vector van a ser de un tipo o de otro. En el JList se ha establecido uno para verlo claro.

# JList

**Es una lista seleccionable.** Estableciendo la propiedad model pondremos los elementos fácilmente con el Swing Designer. Si vamos al código veremos que se genera un **ListModel** especial para el tipo JList, el **AbstractListModel**, y entonces podremos acceder a los elementos seleccionados con las funciones siguientes:

```
entero = list.getSelectedIndex();
vectorDeEnteros = list.getSelectedIndices();
elemento = list.getSelectedValue();
vectorDeElementos = list.getSelectedValues();
```

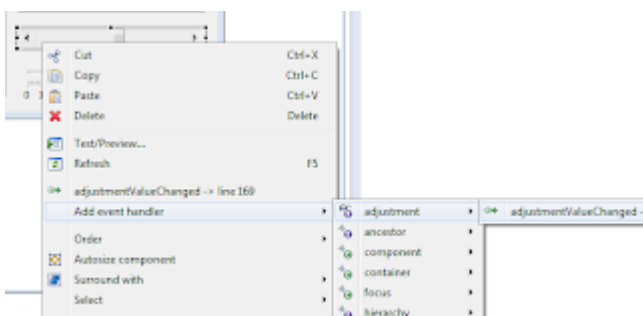
La lista permite selección múltiple, por eso que tenemos las funciones que devuelven vectores.

# JProgressBar

**Es una barra de progreso.** Con `nombre.setValue(nuevoValor)` y `nombre.getValue()` podemos cambiar o saber el valor que tiene en cada momento. Su manejo es sencillo y pienso que con esas dos funciones en bastante para empezar con ello.

# JScrollbar

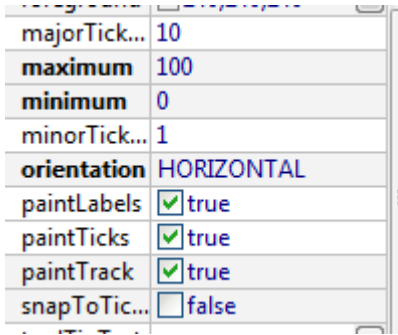
**Es una barra de desplazamiento.** Se usa igual que la barra de progreso a la hora de programar, pero añade que el usuario puede moverla, de manera que puede cambiar el valor de su posición. Con ésto que añade es interesante capturar el evento para hacer algo cuando el usuario cambia su posición, para ello con el boton derecho en el componente la damos al evento de cambio de ajuste:



En el código del ejemplo se puede ver que cuando se mueve la barra de desplazamiento se van a ajustar tanto la barra la progreso como el deslizador que viene a continuación.

## JSlider

En castellano **se llama deslizador**. Funciona igual que la barra de desplazamiento, pero nos permite un mayor ajuste, pudiendo mostrar una regla con los números. Para que se vea la regla debemos poner los valores que se ven en la imagen siguiente:



majorTick...	10
maximum	100
minimum	0
minorTick...	1
orientation	HORIZONTAL
paintLabels	<input checked="" type="checkbox"/> true
paintTicks	<input checked="" type="checkbox"/> true
paintTrack	<input checked="" type="checkbox"/> true
snapToTic...	<input type="checkbox"/> false

Si no configuramos éstos valores en el editor de diseño no se verán los números. De igual manera que los anteriores, con **getValue()** tendremos el valor actual.

En la barra de desplazamiento se ha capturado el evento que se dispara al **cambiar el ajuste**. Para el deslizador el evento que se dispara cuando se mueve el deslizador se llama cambio de estado, como se ve en el código fuente:

```
slider.addChangeListener(new ChangeListener() {
    public void stateChanged(ChangeEvent arg0) {
        progressBar.setValue(slider.getValue());
        scrollbar.setValue(slider.getValue());
    }
});
```

## JSeparator

**Simplemente es una línea de separación**, una raya, que ya puse en el post anterior sobre los menús. Se puede poner también en el JFrame, concretamente en el panel, para dibujar una separación.

## Terminando

Todo lo anterior se puede ver al detalle escudriñando un poco el código fuente generado con el Swing Designer y un poco programado manualmente:



```

import java.awt.Color;
import java.awt.EventQueue;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.AdjustmentEvent;
import java.awt.event.AdjustmentListener;

import javax.swing.AbstractListModel;
import javax.swing.ButtonGroup;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
import javax.swing.JCheckBox;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JList;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JProgressBar;
import javax.swing.JRadioButton;
import javax.swing.JScrollBar;
import javax.swing.JSeparator;
import javax.swing.JSlider;
import javax.swing.JSpinner;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.JToggleButton;
import javax.swing.SwingConstants;
import javax.swing.border.EmptyBorder;
import javax.swing.border.LineBorder;
import javax.swing.event.ChangeEvent;
import javax.swing.event.ChangeListener;

// ÉSTO ES LA CLASE PRINCIPAL QUE EXTIENDE DE JFRAME...
public class Principal extends JFrame {

    private JPanel contentPane;
    private JTextField txtCajaDeTexto;
    static Principal frame;
    private JPasswordField passwordField;

    // la función que primero se ejecuta que crea
    // el frame en memoria...
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    frame = new Principal();
                    // ... y lo visualiza
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // aquí está el constructor de la clase que todo
    // lo programa...
    public Principal() {
        // se contruye la ventana
        setTitle("JavaSwing 6 Componentes bu00Elsicos by Jnj");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 458, 363);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);
    }

```

```

// la etiqueta
JLabel lblEtiqueta = new JLabel("Etiqueta");
lblEtiqueta.setBounds(10, 11, 67, 14);
contentPane.add(lblEtiqueta);

// la caja de texto
txtCajaDeTexto = new JTextField();
txtCajaDeTexto.setText("Caja de texto");
txtCajaDeTexto.setBounds(10, 36, 152, 20);
contentPane.add(txtCajaDeTexto);
txtCajaDeTexto.setColumns(10);

// el cuadro combinado
JComboBox comboBox = new JComboBox();
// los elementos
comboBox.setModel(new DefaultComboBoxModel(new String[] { "Elemento 1",
"Elemento 2", "Elemento 3" }));
// comienza seleccionado el elemento 1
comboBox.setSelectedIndex(1);
comboBox.setBounds(10, 67, 191, 20);
contentPane.add(comboBox);

// el botón
JButton btnBotn = new JButton("Botu00F3n");
// cuando se pulsa hace ésto
btnBotn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        JOptionPane.showMessageDialog(frame, "Ésto es un botón simple.");
    }
});
btnBotn.setBounds(53, 95, 89, 23);
contentPane.add(btnBotn);

// la casilla de verificación
JCheckBox chckbx = new JCheckBox("Casilla de verificaciu00F3n");
chckbx.setSelected(true);
chckbx.setBounds(10, 125, 191, 23);
contentPane.add(chckbx);

// los botones de radio
JRadioButton rdbtn1 = new JRadioButton("RadioButton 1");
rdbtn1.setBounds(10, 151, 109, 23);
contentPane.add(rdbtn1);

JRadioButton rdbtn2 = new JRadioButton("RadioButton 2");
rdbtn2.setBounds(10, 177, 109, 23);
contentPane.add(rdbtn2);

// el grupo para los botones
ButtonGroup radioGroup = new ButtonGroup();
radioGroup.add(rdbtn1);
radioGroup.add(rdbtn2);

// otras casillas de verificación que se van a agrupar
// para ver el comportamiento
JCheckBox chckbx1 = new JCheckBox("CheckBox 1 agrupado");
chckbx1.setBounds(10, 203, 166, 23);
contentPane.add(chckbx1);

JCheckBox chckbx2 = new JCheckBox("CheckBox 2 agrupado");
chckbx2.setBounds(10, 229, 152, 23);
contentPane.add(chckbx2);

// el grupo para las casillas de verificación
ButtonGroup chkGroup = new ButtonGroup();
chkGroup.add(chckbx1);
chkGroup.add(chckbx2);

// el botón de activación
JToggleButton tglbtn = new JToggleButton("Botu00F3n de activaciu00F3n");
// que está seleccionado inicialmente
tglbtn.setSelected(true);

```

```

tglbtn.setBounds(10, 259, 166, 23);
contentPane.add(tglbtn);

// el área de texto
JTextArea txtrAreaDeTexto = new JTextArea();
// con una linea en el borde
txtrAreaDeTexto.setBorder(new LineBorder(new Color(0, 0, 0)));
// tres lineas de texto permitidas
txtrAreaDeTexto.setRows(3);
// el texto
txtrAreaDeTexto.setText("Arearnderntexto.");
txtrAreaDeTexto.setBounds(224, 11, 205, 58);
contentPane.add(txtrAreaDeTexto);

// el texto de contrasela
passwordField = new JPasswordField();
passwordField.setBounds(224, 80, 152, 20);
// contraseña inicial
passwordField.setText("supercontraseña");
contentPane.add(passwordField);

// el llamado 'hilandero'
JSpinner spinner = new JSpinner();
spinner.setBounds(224, 111, 101, 20);
contentPane.add(spinner);

// la lista
JList list = new JList();
// la declaración de los elementos
list.setModel(new AbstractListModel() {
    String[] values = new String[] {"Elemento 1", "Elemento 2", "Elemento
3"};
    // función que devuelve el número de elementos
    public int getSize() {
        return values.length;
    }
    // función para acceder a cada elemento
    // por ejemplo usando
    //
    // list.getModel().getElementAt(1)
    //
    // nos devolverá el elemento "Elemento 2"
    public Object getElementAt(int index) {
        return values[index];
    }
});
list.setSelectedIndex(1);
list.setBounds(224, 142, 67, 52);
contentPane.add(list);

// el separador
JSeparator separator = new JSeparator();
separator.setOrientation(SwingConstants.VERTICAL);
separator.setBounds(211, 11, 2, 297);
contentPane.add(separator);

// inicializacion de la barra de progreso,
// la barra de desplazamiento y el deslizador.
// los he puesto aquí juntos porque los uso
// después y era necesario.
final JProgressBar progressBar = new JProgressBar();
final JScrollBar scrollBar = new JScrollBar();
final JSlider slider = new JSlider();

// configuraciones de la barra de progreso
progressBar.setValue(50);
progressBar.setBounds(224, 203, 205, 14);
contentPane.add(progressBar);

// la barra de desplazamiento
scrollBar.addAdjustmentListener(new AdjustmentListener() {
    public void adjustmentValueChanged(AdjustmentEvent arg0) {

```

```
// mueve a la vez la barra de progreso y el deslizador
progressBar.setValue(scrollBar.getValue());
slider.setValue(scrollBar.getValue());
}
});
// valor inicial
scrollBar.setValue(50);
scrollBar.setOrientation(JScrollBar.HORIZONTAL);
scrollBar.setBounds(224, 233, 205, 23);
contentPane.add(scrollBar);

// el deslizador, capturando evento
slider.addChangeListener(new ChangeListener() {
    public void stateChanged(ChangeEvent arg0) {
        // mueve la barra de progreso y la de desplazamiento
        // cada vez que se mueve el deslizador.
        progressBar.setValue(slider.getValue());
        scrollBar.setValue(slider.getValue());
    }
});
// valores para que salga la regla numerada
slider.setMinorTickSpacing(1);
slider.setMajorTickSpacing(10);
slider.setToolTipText("");
slider.setPaintTicks(true);
slider.setPaintLabels(true);
slider.setBounds(224, 267, 200, 52);
contentPane.add(slider);

} // termina el constructor de la ventana
} // termina la clase
```

Un fichero .zip para descargar con todo el código fuente y un ejecutable .jar para verlo en acción lo tienen aquí: [descargar ejemplo](#).

Para ejecutarlo es necesario por lo menos tener el JRE, recomendando la última versión a ser posible, la 7.

Como viene siendo habitual, para más información, me remito a la documentación oficial:

<http://docs.oracle.com/javase/tutorial/uiswing/>

Espero que haya servido.

Un saludo.

## Deja una respuesta

Tu dirección de correo electrónico no será publicada. Los campos obligatorios están marcados con \*

Comentario \*

Nombre \*

Correo electrónico \*

Web



☐ I'm not a robot

reCAPTCHA  
[Privacy](#) - [Terms](#)



Notifícame cuando se añadan nuevos comentarios.

Publicar el comentario

[⏪ Java Swing 5: Preparando el menú para estas Navidades...](#)

[Mapas conceptuales y construcción del conocimiento con Freeplane ⏩](#)



## Lo + visto

- [VHDL: decodificador BCD de 4 bits para display de 7 segmentos](#)
- [Looking for the 100 points](#)
- [Datatables con botones para CSV, Excel, PDF e imprimir](#)
- [VHDL: uniendo circuitos, un sumador completo de 4 bits](#)
- [VHDL: multiplexor de 4 a 1 con selección de 2 bits](#)
- [VHDL: los ladrillos básicos; and, or, not, nand, nor, xor y xnor](#)
- [Datatables: listando y filtrando tablas de datos con JavaScript](#)
- [Los patrones del diseño software GRASP](#)
- [Sencillo mensaje de aceptar cookies para tu web](#)
- [Nueva versión del sencillo mensaje de aceptar cookies para tu web](#)
- [Java: serializando objetos para guardar y recuperar en ficheros \(editado\)](#)
- [VHDL: sumador completo de 1 bit](#)
- [VHDL: diseñando una unidad aritmético lógica](#)

## Etiquetas


[bases de datos](#) [Bower](#) [CMS](#) [Composer](#)  
[copias de seguridad](#) [CSS](#) [Drupal](#)  
[Electrónica](#) [general](#) [GNU/Linux](#)  
[HTML](#) [Inteligencia Artificial](#)  
[Java](#) [JavaScript](#) [Joomla](#) [Joomla](#)  
[Platform](#) [Mac](#) [Magento](#)  
[mantenimiento](#) [nube](#) [Open](#)  
[Source](#) [ordenadores](#) [Packagist](#)  
[PHP](#) [Prestashop](#)  
[programación](#)  
[programas](#) [Prolog](#) [PWA](#)  
[Raspberry Pi](#) [seguridad](#) [SEO](#)  
[servidores](#) [shell script](#) [Sistemas](#)  
[SQL](#) [Swing](#) [Symfony](#) [Ubuntu](#)  
[utilidades](#) [Vagrant](#) [VHDL](#)  
[webs](#) [Windows](#)  
[WordPress](#)  
[Legales y otros enlaces](#)  
[Aviso legal](#)  
[Política de privacidad](#)  
[Declaración de accesibilidad](#)

## Categorías

[Amazon Web Services](#)  
[Angular](#)  
[C/C++](#)  
[CSS](#)  
[Drupal](#)  
[General](#)  
[GNU/Linux](#)  
[HTML](#)  
[Inteligencia Artificial](#)  
[Java](#)  
[Java Spring Boot](#)  
[Java Swing](#)  
[JavaScript](#)  
[Magento](#)  
[PHP](#)  
[PrEDA](#)  
[Prestashop](#)  
[Principios y patrones](#)  
[Prolog](#)  
[Python](#)  
[Seguridad](#)  
[SEO](#)  
[Symfony](#)  
[VHDL](#)  
[WordPress](#)

- [Prolog: cómo hacer condicionales](#)
- [VHDL: registro sencillo de 8 bits](#)
- [GNU/Linux: cómo configurar el antivirus ClamAV con auto-escaneo de ficheros](#)
- [¿Qué es un Bean de Java?](#)
- [IA: cómo montar un Sistema Basado en Reglas con PHP](#)
- [Cómo implementar una Arquitectura Hexagonal con Symfony Flex \(editado\)](#)
- [Java Swing 6: Los componentes básicos](#)

© 2023 JnjSite.com - MIT license

Sitio hecho con  WordPress, diseño y programación del [tema](#) por Jnj.

