

**TECNOLOGICO NACIONAL DE MEXICO
CAMPUS "OAXACA"**

INSTITUTO TECNOLÓGICO DE OAXACA

INGENIERIA EN SISTEMAS COMPUTACIONALES.

NOMBRE DE LA MATERIA: TOPICOS AVANZADOS DE PROGRAMACION.

UNIDAD 1: "INTERFAZ GRAFICA DEL USUARIO".

ACTIVIDAD 1: GUIAS DE INTERFAZ DE USUARIO.

GRUPO: ISU

HORA: 09:00-10:00 AM

DOCENTE: M.C. LUIS ALBERTO ALONSO HERNÁNDEZ

ALUMNO: GONZALEZ PASCUAL MELVIN PAUL

FECHA DE ENTREGA: 02 DE OCTUBRE DEL 2020

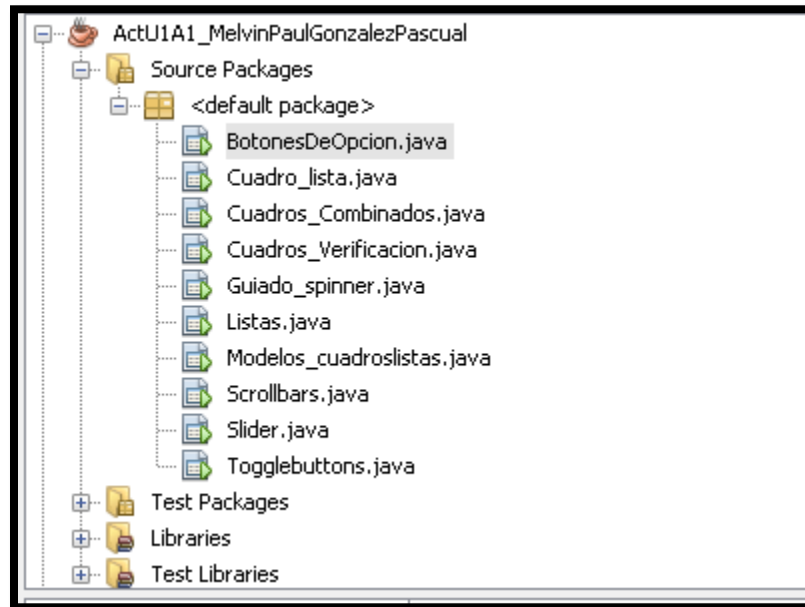
Contenido

Introducción:	3
Cuadro de verificación.	4
Botones de opción.	6
Listas.....	8
Combo.	10
Listas modelos.....	11
Combos modelos.....	13
Togglebuttons.	16
Sliders.	18
Spinner.	20
Barras de desplazamiento.....	22

Introducción:

En esta primera actividad, empezaremos a familiar con los componentes que estaremos usando para las actividades, también se desarrollara cada una de las actividades, usando las guías proporcionada anteriormente, en mi caso, todo fue realizado con el GUI NetBeans, para hacer un poco más fácil la manera de usar los componentes, a continuación, se explicara cada unos de los ejercicios elaborados.

Se creo un nuevo proyecto con el nombre de ActU1A1_MelvinPaulGonzalezPascual, y se fueron creado JFrame por cada actividad desarrollada.



Cuadro de verificación.

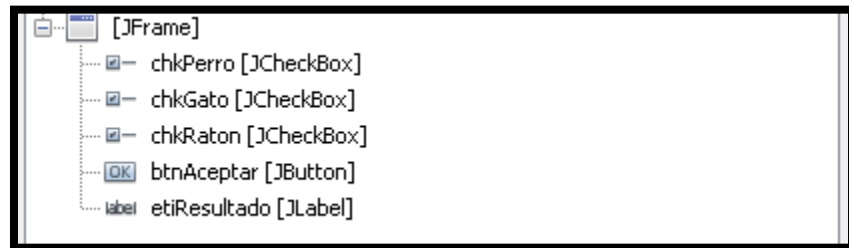
Para esta primera actividad se usarán tres componentes los cuales son los siguientes:

JBUTTON para realizar la acción al momento de ejecutar.

JCheckBox el cual es nuestro cuadro para seleccionar nuestra opción.

JLabel una etiqueta con borde, lo cual el borde de negro, se debe modificar en el apartado de model, y seleccionar la forma con borde.

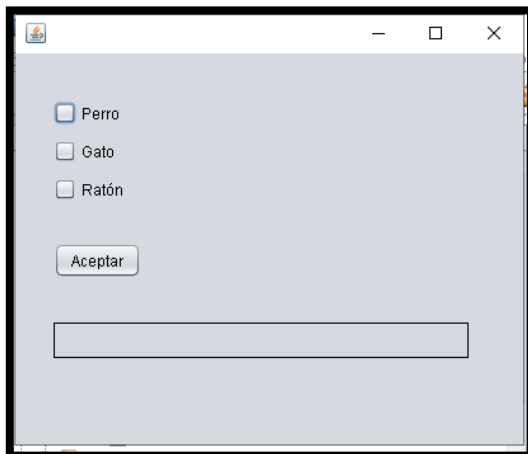
Se le cambiaran el nombre de las variables y quedan de las siguientes manera.



En el evento de acción de nuestro botón se debe agregar la siguiente codificación para que pueda funcionar de la forma que nos pide la guía.

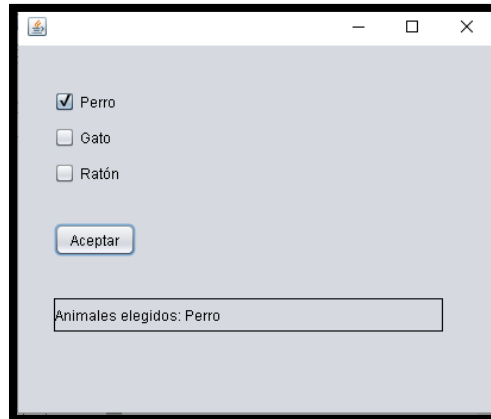
```
private void btnAceptarActionPerformed(java.awt.event.ActionEvent evt) {  
    String mensaje="Animales elegidos: ";  
    if (chkPerro.isSelected()) {  
        mensaje=mensaje+"Perro ";  
    }  
    if (chkGato.isSelected()) {  
        mensaje=mensaje+"Gato ";  
    }  
    if (chkRaton.isSelected()) {  
        mensaje=mensaje+"Raton ";  
    }  
    etiResultado.setText(mensaje);  
}
```

Al momento de ejecutar nuestro programa nos debe dar la siguiente ventana.



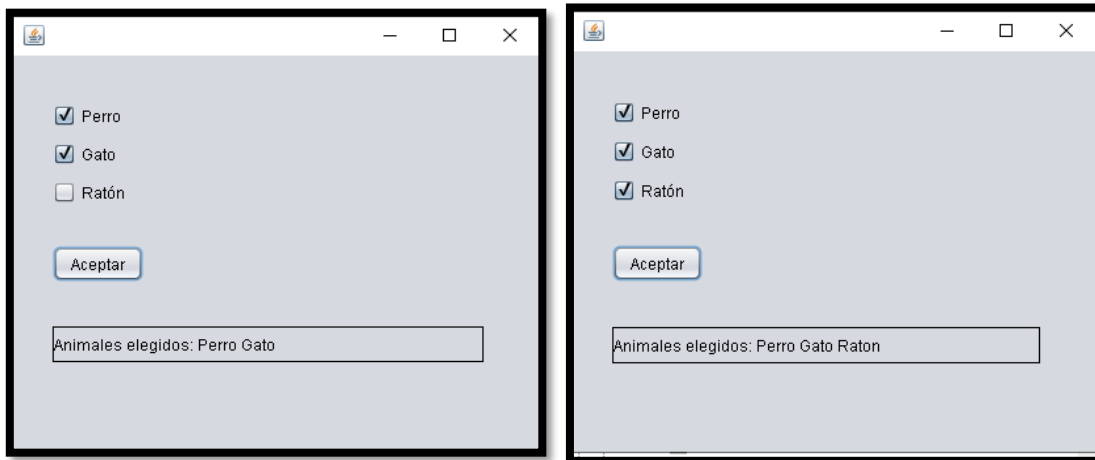
Como se puede apreciar nuestra venta nos debe salir sin ninguna casilla seleccionada y en nuestra etiqueta vacía.

Al momento de seleccionar una casilla el resultado nos sale en la etiqueta, de la siguiente manera.



A screenshot of a Java Swing dialog box. It contains three checkboxes: 'Perro' (checked), 'Gato' (unchecked), and 'Ratón' (unchecked). Below the checkboxes is an 'Aceptar' button. At the bottom, there is a text field labeled 'Animales elegidos: Perro'.

Si se selecciona más de una opción lo que hace es concatenar las opciones en un mismo diálogo.

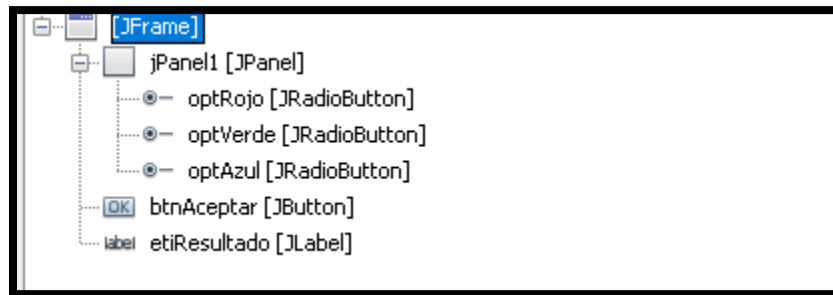


Two side-by-side screenshots of the same dialog box. The left screenshot shows 'Perro' and 'Gato' selected, with the text field displaying 'Animales elegidos: Perro Gato'. The right screenshot shows 'Perro', 'Gato', and 'Ratón' selected, with the text field displaying 'Animales elegidos: Perro Gato Raton'.

Aquí se ve demostrado el uso de las casillas de seleccion. Con esto se da concluido esta actividad 1.

Botones de opción.

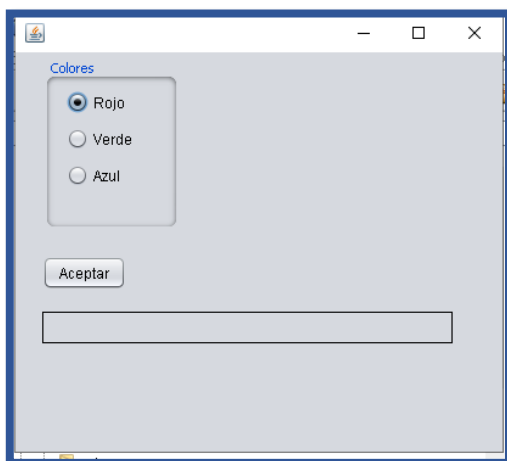
En esta actividad se usaron los siguiente componentes:



Los cuales son tres **JRadioButton** que se ingresan en un **Jpanel**, un **JButton**, y por un ultimo un **JLabel** para realizar la actividad.

Se le asigna la siguiente codificación en el evento de nuestro JButton, el cual al momento de seleccionar una opción y dar en aceptar nos salda el color en nuestra etiqueta .

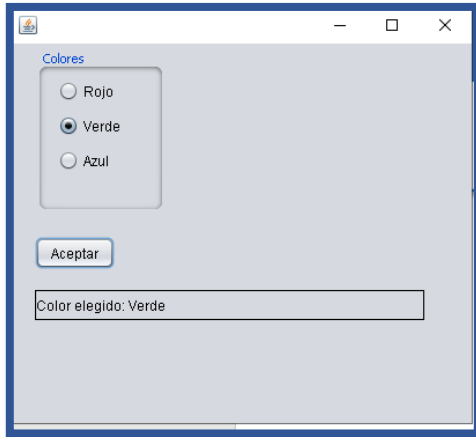
```
private void btnAceptarActionPerformed(java.awt.event.ActionEvent evt) {  
    String mensaje="Color elegido: ";  
    if (optRojo.isSelected()) {  
        mensaje=mensaje+"Rojo";  
    } else if (optVerde.isSelected()) {  
        mensaje=mensaje+"Verde";  
    } else if (optAzul.isSelected()) {  
        mensaje=mensaje+"Azul";  
    }  
    etiResultado.setText(mensaje);  
}
```



Como se puede observar al momento de ejecutar, ya nos trae marcado un botón, lo cual no tiene alguna acción en esa forma.

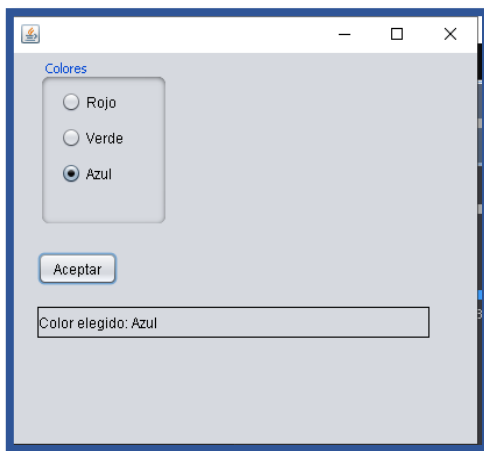
```
public BotonesDeOpcion() {  
    initComponents();  
    optRojo.setSelected(true);  
}
```

El cual se programa desde nuestro constructor de la clase, y se da como la opción activa de inicio.



Al momento de seleccionar una de las opciones y dar en aceptar, nos dirá el color seleccionado.

Ah diferencia de la anterior actividad, esta no se puede seleccionar mas de un opción, por lo cual siempre será un color.

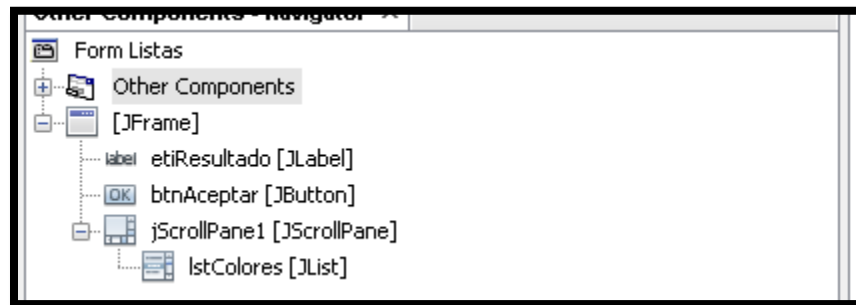


Como se puede observar siempre será un color.

Los **JToggleButton** son un componente muy útil debido que solo se puede para tener un solo resultado.

Listas.

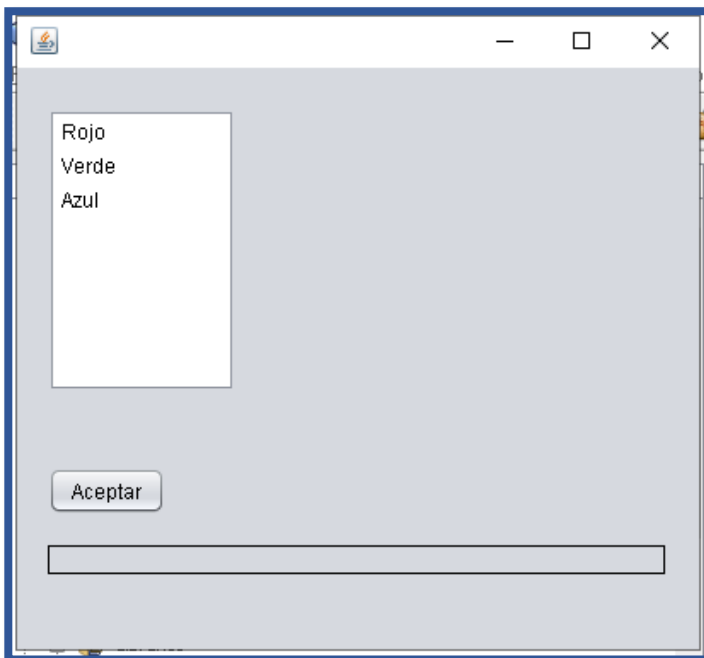
Para esta actividad nos enfocaremos en las **JList**, además JLabel y JButton los cuales se les cambiara el nombre de la variable para que quede de la siguiente manera.



En el evento de nuestro botón aceptar se debe programar lo siguiente para que nuestro programa funcione adecuadamente.

```
private void btnAceptarActionPerformed(java.awt.event.ActionEvent evt) {  
    String mensaje;  
    if (lstColores.getSelectedIndex() == -1) {  
        mensaje = "No hay un color seleccionado.";  
    } else {  
        mensaje = "El color seleccionado es: " + lstColores.getSelectedValue().toString();  
    }  
    etiResultado.setText(mensaje);  
}
```

Al momento de ejecutar nuestro programa nos deberá de salir de la siguiente manera.

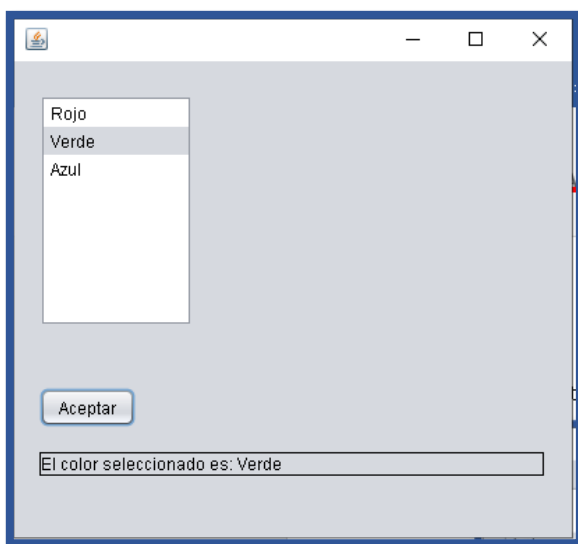


Lo cual siempre iniciará sin ningún valor en nuestra etiqueta, el cual tendrá hasta que seleccionemos una opción y demos en el botón aceptar.

Al momento de seleccionar una opción nos dará el siguiente resultado.



Como se puede observar nos da el color que se selecciono de las opciones de la lista.



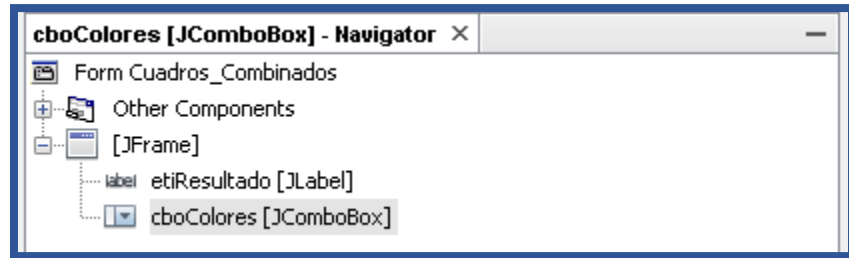
Como se puede apreciar siempre cambiara al color que se ha seleccionado.

Las listas son útiles para seleccionar un amplio catálogo de opciones y tener una mejor velocidad de poder escoger.

Combo.

Para esta actividad usaremos dos componentes, los cuales serán: JComboBox y JLabel.

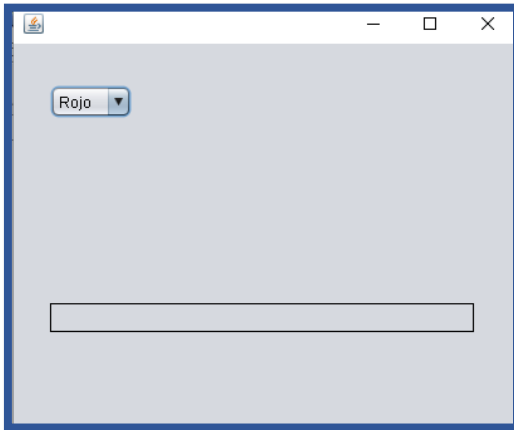
Las cuales se les cambia el nombre de sus variables, y quedaran de la siguiente manera.



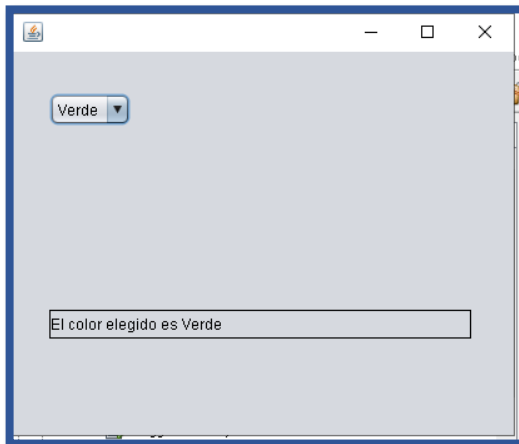
En el evento de nuestro JComboBox, se ingresará el siguiente código para que tenga su funcionamiento.

```
private void cboColoresActionPerformed(java.awt.event.ActionEvent evt) {  
    String mensaje="El color elegido es ";  
    mensaje=mensaje+cboColores.getSelectedItem().toString();  
    etiResultado.setText(mensaje);  
}
```

Cuando se ejecuta el programa nos debe salir de la siguiente manera.



Nos sale sin una opción, ya que nuestra etiqueta se encuentra vacía, para que tenga algún texto se deberá escoger una opción.

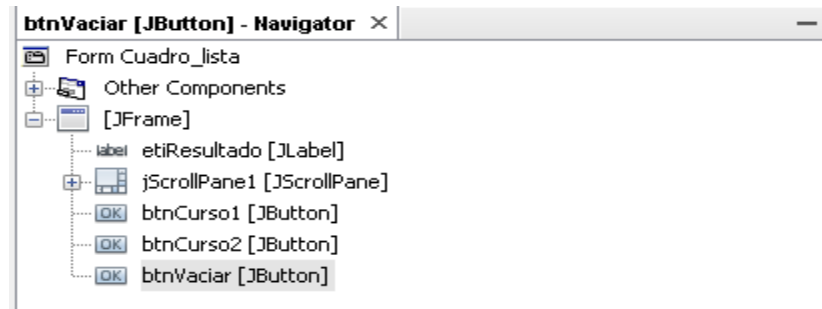


Al momento de elegir una opción en nuestra etiqueta, con el JComboBox es que no es necesario usar un botón para generar las acciones.

Por lo cual en esta actividad se realizó con solos dos componentes, y su funcionamiento fue bien.

Listas modelos.

Para esta actividad necesitaremos de lista, botones y etiqueta, las cuales serán renombradas de la siguiente manera.



Debemos importar la siguiente.

```
import javax.swing.DefaultListModel;
```

En los eventos de los botones se programarán lo deseado que es lo siguiente.

En nuestro botón curso uno será lo siguiente, los cuales son tres nombres que deberán aparecer en la lista.

```
private void btnCurso1ActionPerformed(java.awt.event.ActionEvent evt) {  
    DefaultListModel modelo = new DefaultListModel();  
    modelo.addElement("Juan");  
    modelo.addElement("María");  
    modelo.addElement("Luis");  
    lstNombres.setModel(modelo);  
}
```

De la misma forma en el botón de curso dos.

```
private void btnCurso2ActionPerformed(java.awt.event.ActionEvent evt) {  
    DefaultListModel modelo = new DefaultListModel();  
    modelo.addElement("Ana");  
    modelo.addElement("Marta");  
    modelo.addElement("Jose");  
    lstNombres.setModel(modelo);  
}
```

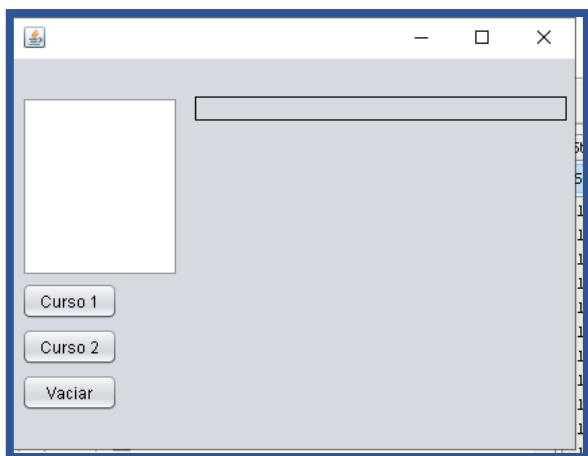
En nuestra etiqueta se hará para que nos aparezca lo seleccionado.

```
private void lstNombresMouseClicked(java.awt.event.MouseEvent evt) {  
    etiResultado.setText(lstNombres.getSelectedValue().toString());  
}
```

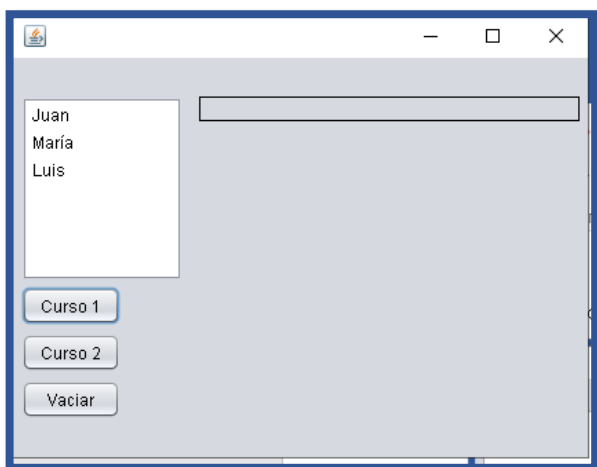
Se creará un botón para vaciar las lista de la siguiente manera.

```
private void btnVaciarActionPerformed(java.awt.event.ActionEvent evt) {  
    DefaultListModel modelo = new DefaultListModel();  
    lstNombres.setModel(modelo);  
}
```

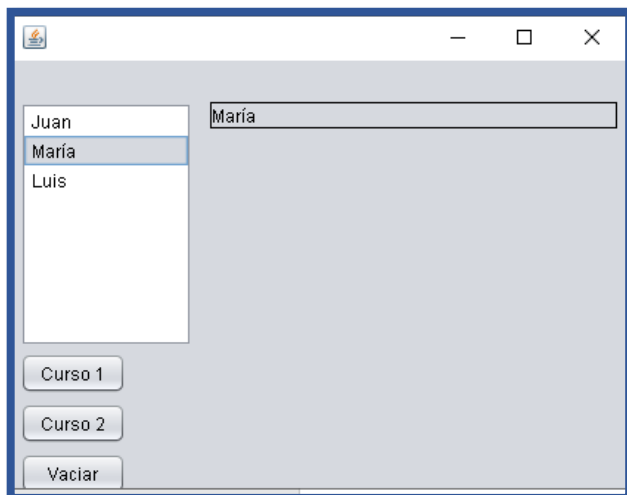
Al momento de ejecutar nuestro programa nos deberá de salir de la siguiente manera.



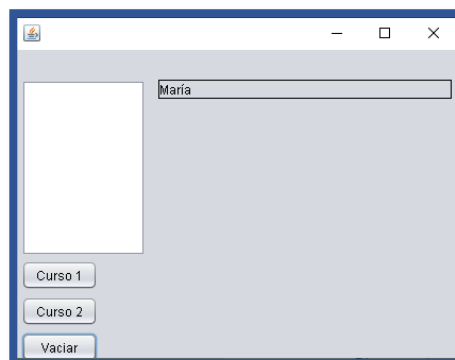
Nos aparecerá con los datos vacíos por lo cual al momento de seleccionar el botón curso1/curso2 nos saldrá lo siguiente.



Al momento de seleccionar un nombre nos aparecerá en la etiqueta.



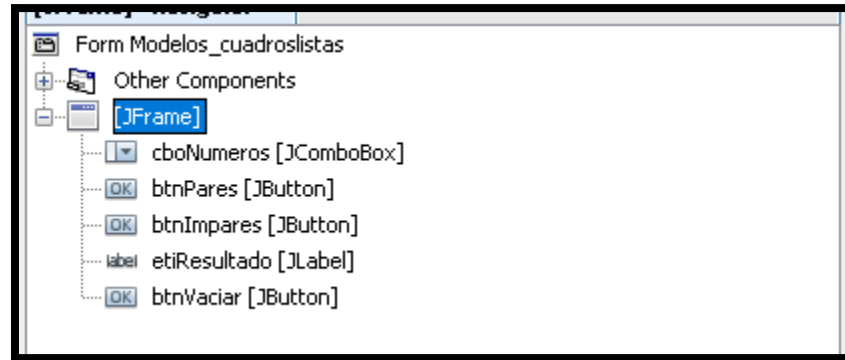
Lo cual nos da los siguiente, para poder vaciar los elementos de nuestra lista se creó el botón vaciar por lo cual al momentos de presionarlo nos limpia la lista como se muestra a continuación.



Combos modelos.

Para esta actividad será necesario más de un botón, también se usará un JComboBox y JLabel.

Por lo cual quedara de la siguiente manera:



Es importante saber que se tiene que exportar la siguiente librería.

```
import javax.swing.DefaultComboBoxModel;
```

En esta actividad se programarán varios eventos los cuales quedan de la siguiente manera.

En nuestro botón Pares, como su nombre dice, serán solo número pares que van del 0-2.

```
private void btnParesActionPerformed(java.awt.event.ActionEvent evt) {  
    int i;  
    DefaultComboBoxModel modelo = new DefaultComboBoxModel();  
    for (i=0;i<10;i+=2) {  
        modelo.addElement("N° "+i);  
    }  
    cboNumeros.setModel(modelo);  
}
```

En el botón impar será los números impares del 0-10.

```
private void btnImparesActionPerformed(java.awt.event.ActionEvent evt) {  
    int i;  
    DefaultComboBoxModel modelo = new DefaultComboBoxModel();  
    for (i=1;i<10;i+=2) {  
        modelo.addElement("N° "+i);  
    }  
    cboNumeros.setModel(modelo);  
}
```

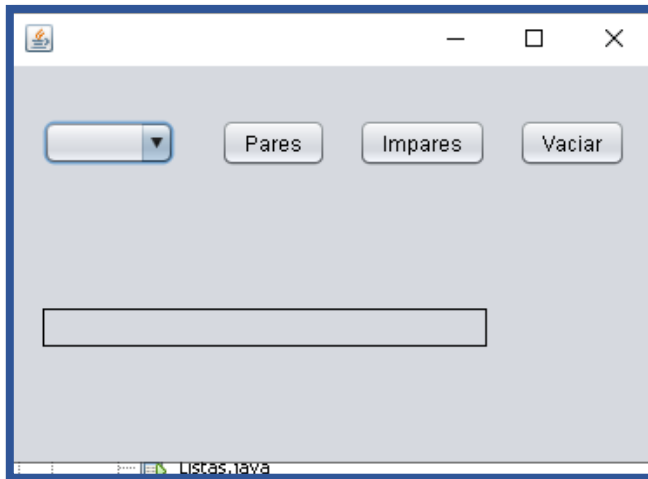
Ambos botones harán que en nuestra etiqueta tenga la información que se obtenga del JComboBox, por lo cual queda de la siguiente manera.

```
private void btnVaciarActionPerformed(java.awt.event.ActionEvent evt) {  
    DefaultComboBoxModel modelo = new DefaultComboBoxModel();  
    cboNumeros.setModel(modelo);  
}
```

También de manera opcional se crea un botón de Vaciar, lo que hace es eliminar los datos de nuestro JComboBox.

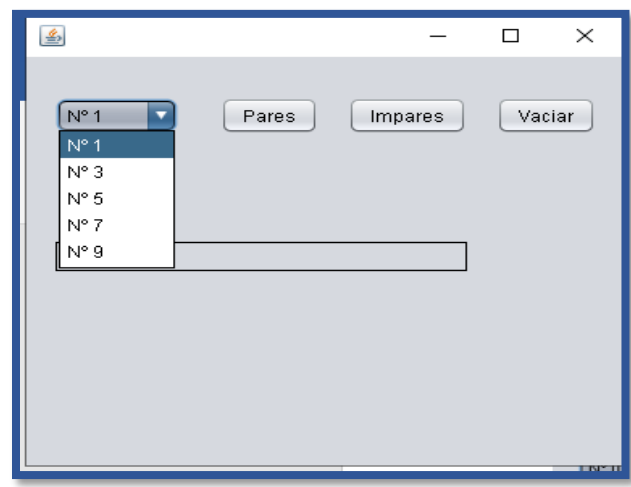
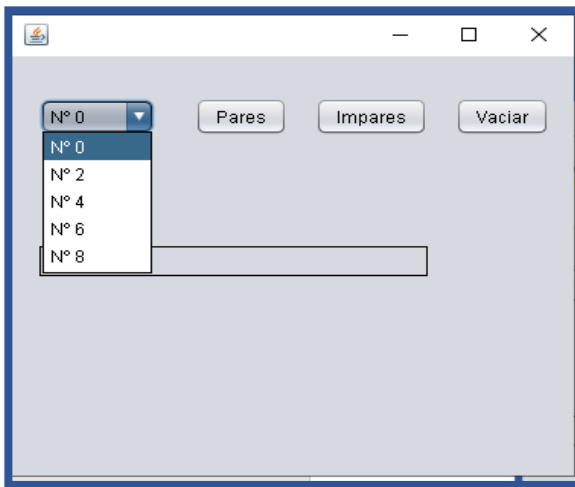
```
private void btnVaciarActionPerformed(java.awt.event.ActionEvent evt) {  
    DefaultComboBoxModel modelo = new DefaultComboBoxModel();  
    cboNumeros.setModel(modelo);  
}
```

Nuestra interfaz debe quedar de la siguiente manera.

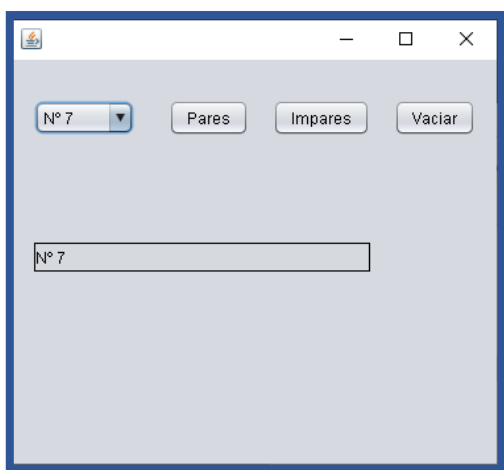


Lo cual se puede observar el combo, los tres botones y la etiqueta.

Al momento de presionar los botones de pares/impares nos saldrá lo siguiente.

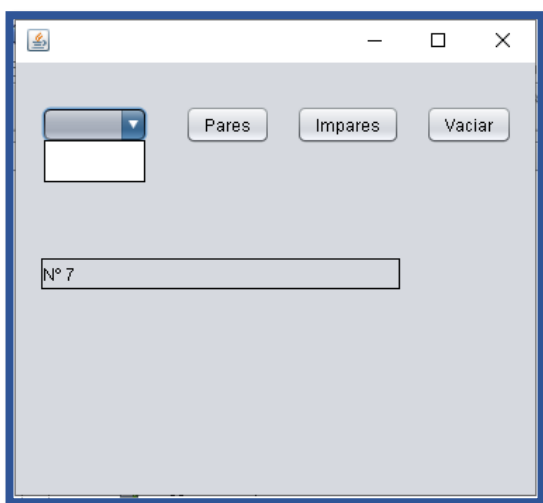


Al momento de seleccionar cualquiera de los números nos saldrá en nuestra etiqueta.



Nos sale la opción que le damos en nuestro combo.

Para poder vaciar el contenido usaremos el botón que se creó.

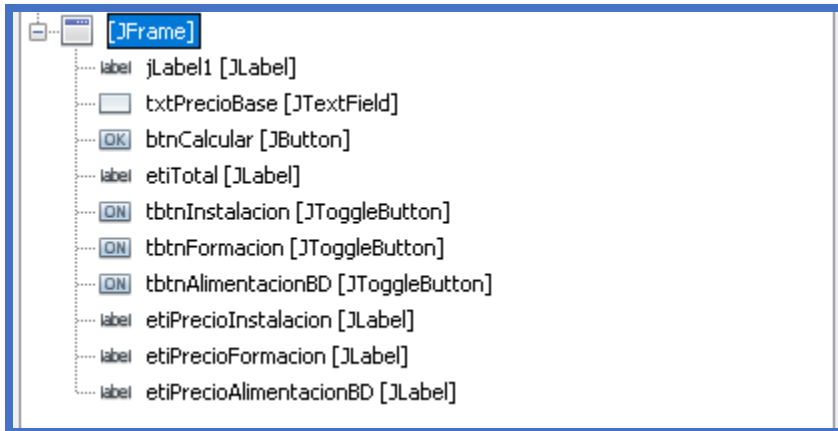


Como se observa el combo se quedó sin valores, por lo cual se puede concluir que esta actividad se hizo de la manera adecuada.

Togglebuttons.

En esta actividad se conocerán la función de los togglebuttons, lo cual tienen una característica especial, el cual se mantendrá pulsado hasta que nuevamente sea oprimido.

Se ocuparán los siguientes componentes y se le cambiara la variable para tener un mejor orden para la codificación.



Se puede observar que para esta actividad se ocupan varias etiquetas al igual que botones, también usaremos por primera vez el JTextField, el cual funciona para rellenar nuestro datos, por ejemplo, en esta actividad con una operación.

En nuestro botón aceptar se programará todo en el evento el cual queda de la siguiente manera.

```
private void btnCalcularActionPerformed(java.awt.event.ActionEvent evt) {  
    double precio_base;  
    double precio_instal; //precio instalación  
    double precio_for; //precio formacion  
    double precio_ali; //precio alimentacion  
    //Recojo datos desde la ventana:  
    precio_base = Double.parseDouble(txtPrecioBase.getText());  
    precio_instal = Double.parseDouble(etiPrecioInstalacion.getText());  
    precio_for = Double.parseDouble(etiPrecioFormacion.getText());  
    precio_ali = Double.parseDouble(etiPrecioAlimentacionBD.getText());  
    //Ahora que tengo los datos, puedo hacer cálculos.  
    //Al precio base se le van añadiendo precio de extras  
    //según estén o no activados los JToggleButtons  
    double precio_total;  
    precio_total = precio_base;  
    if (tbtnInstalacion.isSelected()) { //Si se seleccionó instalación  
        precio_total = precio_total+precio_instal;  
    }  
    if (tbtnFormacion.isSelected()) { //Si se seleccionó formación  
        precio_total = precio_total+precio_for;  
    }  
    if (tbtnAlimentacionBD.isSelected()) { //Si se seleccionó Alimentación BD  
        precio_total = precio_total+precio_ali;  
    }  
    //Finalmente pongo el resultado en la etiqueta  
    etiTotal.setText(precio_total+" €");  
    // TODO add your handling code here:  
}
```

En el constructor, se programa que el botón Instalación aparezca pulsado de la siguiente manera:


```

public Togglebuttons() {
    initComponents();
    tbtnInstalacion.setSelected(true);
}

```

Nuestra interfaz quedara de la siguiente manera:

Precio Base

Instalación 40

Formación 200

Alimentación 200

Calcular

Como se puede observar el botón instalación ya aparece marcado, debido que anteriormente fue activado la opción, en la cuadro de texto, se ingresara la cantidad y con los togglebuttons, se le aplicara la suma que se desea como a continuación:

Precio Base

400

Instalación 40

Formación 200

Alimentación 200

Calcular

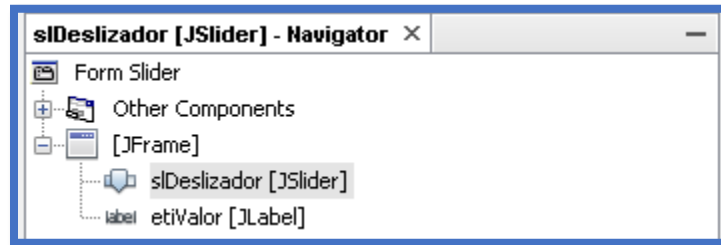
640.0 €

En este caso se puede presionar mas de una opción, como se ve a continuación, lo que hace el programa es sumar las cantidades, primero la que le damos y después con los botones le daremos los demás datos.

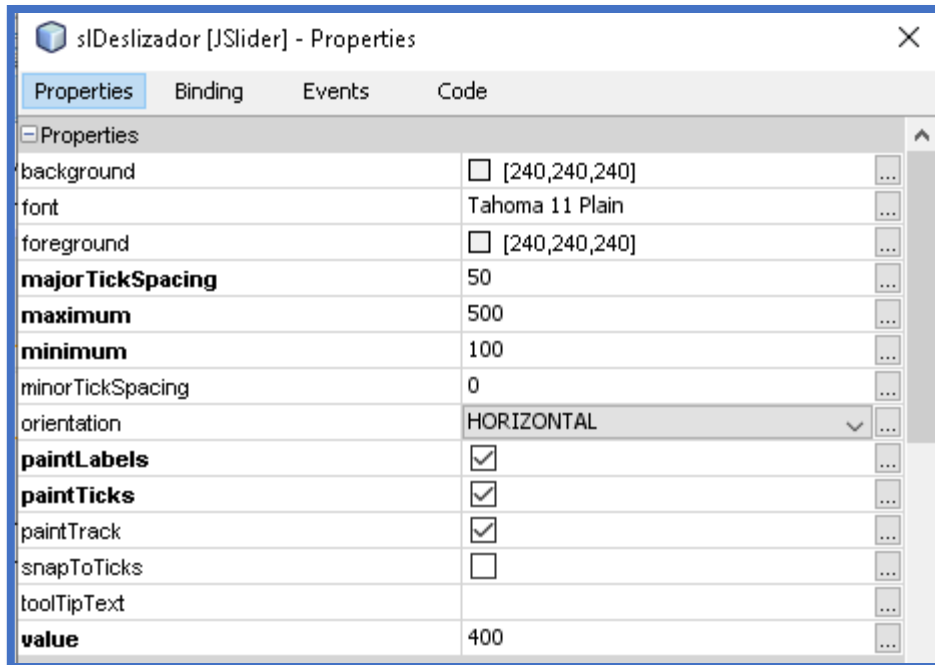
En esta actividad se aprendió el uso del componente, lo cual es muy útil, para diferentes programas.

Sliders.

Para esta actividad solo se usarán dos componentes los cuales son el JSlider y JLabel, los cuales se les cambiara el nombre de sus variables, para un mejor manejo.

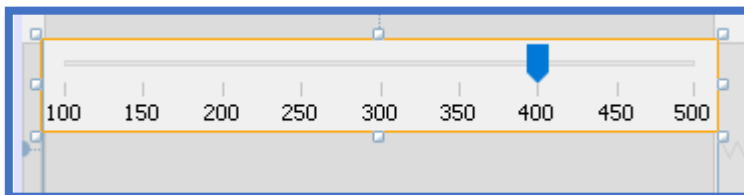


El componente JSlider tiene varias propiedades las cuales se modificaron y quedaron de la siguiente manera.

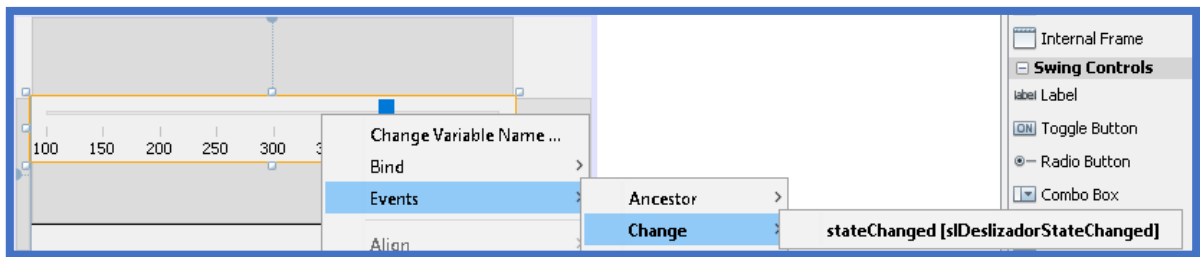


Donde el valor máximo será de 500 y el mínimo 100, donde el desplazamiento será de 50, con un valor inicial de 400, también activaron las opciones de se vea la división de los desplazamiento.

A continuación, se muestra la forma en la que queda dividido:



Antes de empezar a programar el evento, se deberá activar de la siguiente manera.

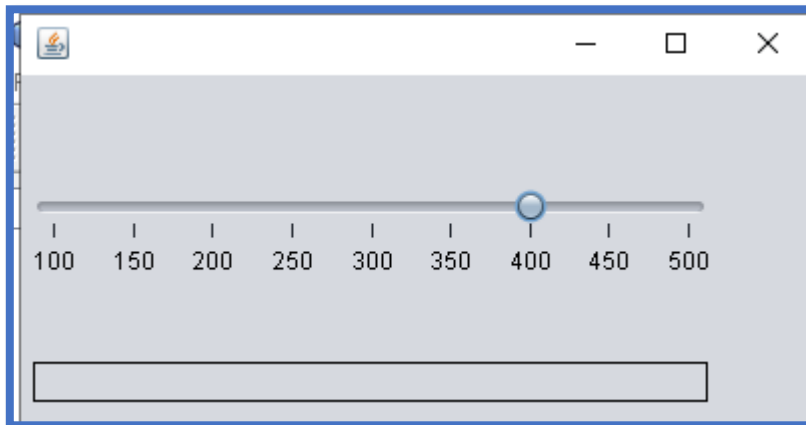


Lo siguiente es solo programar en el evento que se activo de la siguiente manera.

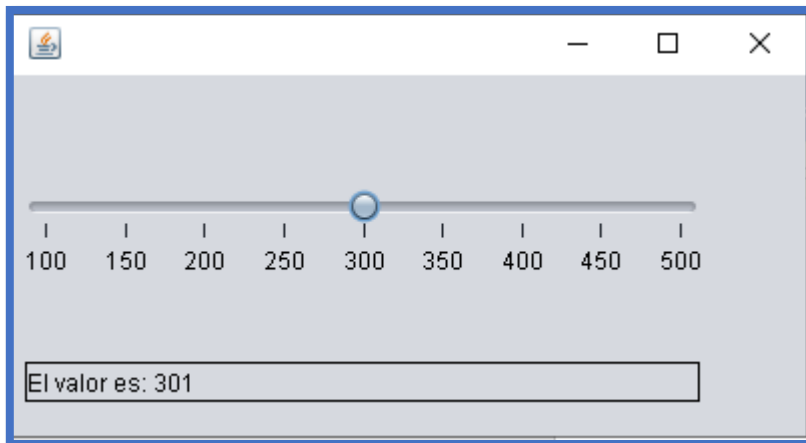
El cual toda la función lo hará el evento y será agregará a nuestra etiqueta de texto.

```
private void slDeslizadorStateChanged(javax.swing.event.ChangeEvent evt) {  
    etiValor.setText("El valor es: "+slDeslizador.getValue());  
    // TODO add your handling code here:  
}
```

Nuestra interfaz grafica debe quedar de la siguiente manera.



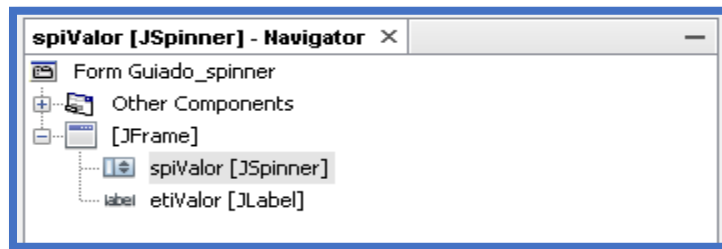
Para que nuestra se le agregue texto, se deberá mover el cursor y nos saldrá lo siguiente.



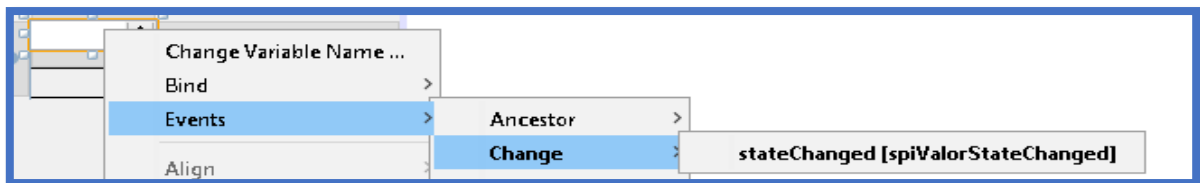
Al momento de dar click avanzara de poco, mientras que deslizamos con el mouse ira de uno en uno, al momento de para nos mandara una cadena con el valor en nuestra etiqueta.

Spinner.

En esta actividad usaremos el componente JSpinner y un JLabel, se le cambiaran el nombre a las variable para que queden de la siguiente manera.



Lo siguiente será activar el evento en nuestro JSpinner, de la siguiente manera.



Primero se debe agregar la siguiente librería a nuestra clase.

```
import javax.swing.SpinnerNumberModel;
```

Lo siguiente será programar en distintas partes de nuestra clase.

El siguiente código realiza la escritura en nuestra etiqueta.

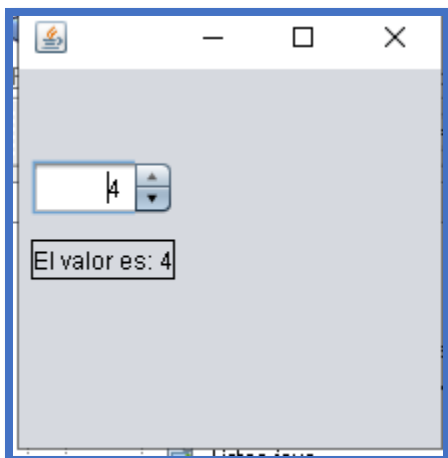
```
private void spiValorStateChanged(javax.swing.event.ChangeEvent evt) {  
    etiValor.setText("El valor es: "+spiValor.getValue().toString());  
}
```

Para que nuestro spinner tenga un límite, se debe programar en el constructor de nuestra clase, de la siguiente manera.

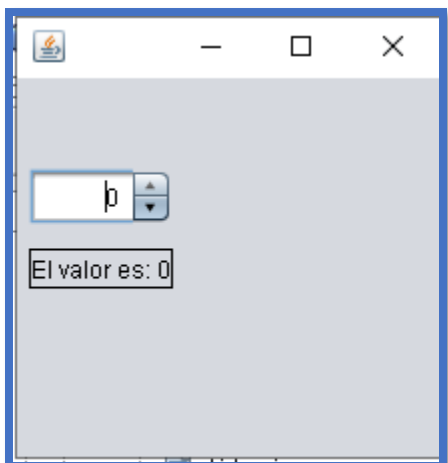
El límite se da desde el método set Maximum y set Minimum, el método setStepSize será a que medida va aumentando o disminuyendo y setValue, el valor que inicia por defecto.

```
public Guiado_spinner() {  
    initComponents();  
    SpinnerNumberModel nm = new SpinnerNumberModel();  
    nm.setMaximum(10);  
    nm.setMinimum(0);  
    nm.setStepSize(2);  
    spiValor.setModel(nm);  
    nm.setValue(4);  
}
```

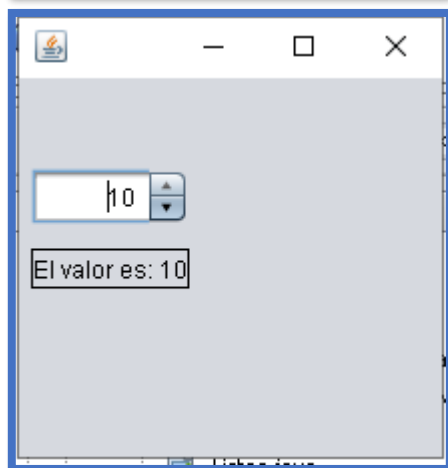
Al momento de ejecutar nuestro programa deberá aparecer de la siguiente manera.



Como se puede apreciar salió con el numero 4 como y en nuestra etiqueta nos aparecerá lo que tiene nuestro spinner.



Nuestro mínimo siempre será 0, ya que así se definió en el constructor de la clase.

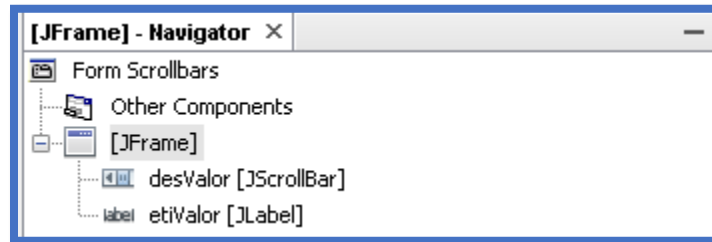


Y como un máximo 10, ya que también se definió en el constructor.

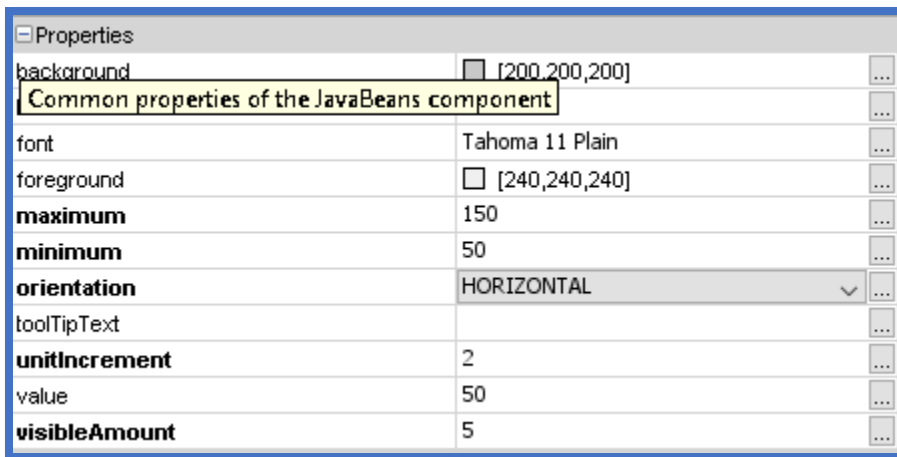
Este componente es muy útil cuando quieres agregar datos cortos, para poder una practica mas fácil, esta actividad fue un poco mas sencilla que las demás.

Barras de desplazamiento.

Para esta actividad también usaremos dos componentes el primero será el JScrollBar y JLabel, los cuales se les cambiará el nombre de la variable que quedará de la siguiente manera:

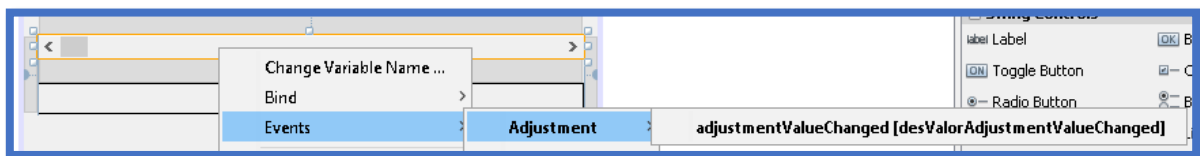


Nuestro JScrollBar tiene varias propiedades las cuales se editarán de la siguiente manera:



Como primera modificación es la orientación que pasa de vertical a horizontal el valor máximo será de 150 y el mínimo de 50, cuando damos clic aumentará de 20, y con la flecha del curso de 2.

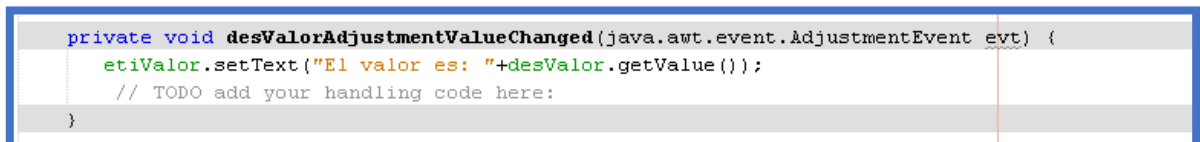
Lo siguiente es



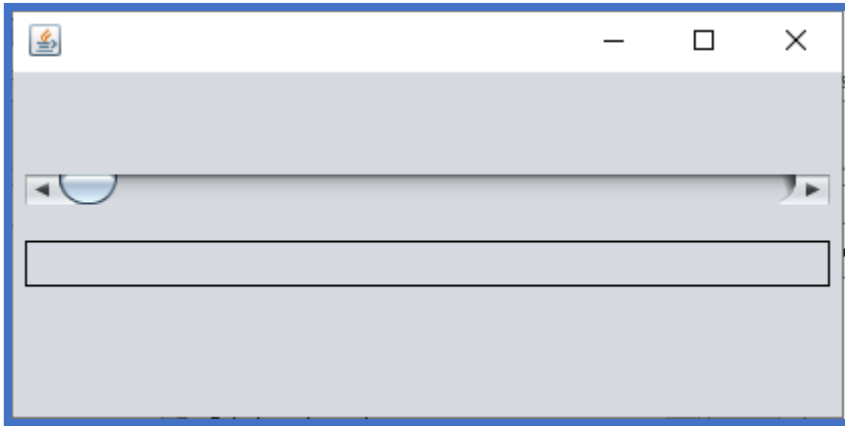
activar el evento en nuestro JScrollBar de la siguiente manera.

Lo cual el evento se programará de la siguiente manera.

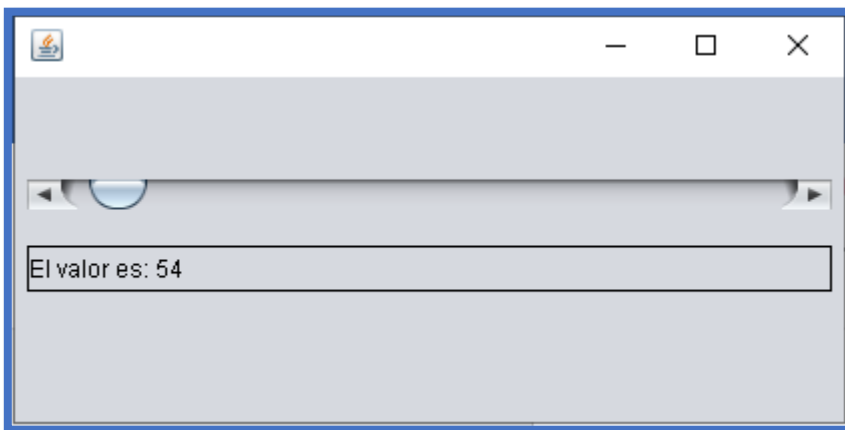
Lo que nos hará que el valor que este en nuestro JScrollBar aparezca en la etiqueta.



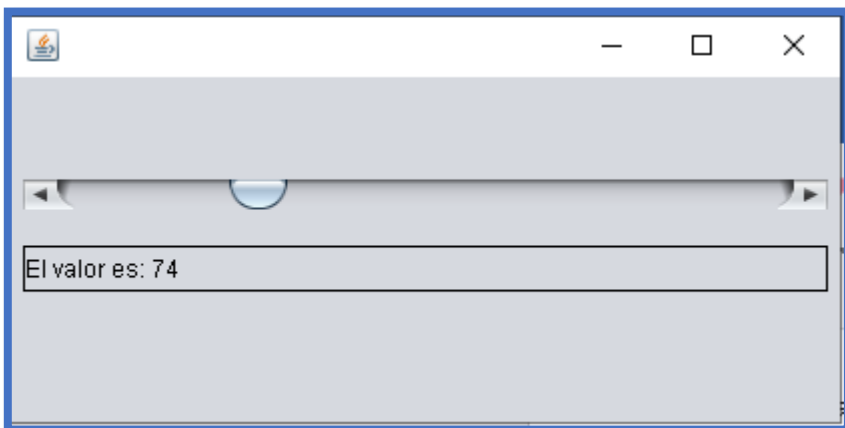
En la parte grafica se debe mostrar de la siguiente manera.



Como se puede apreciar al momento de ejecutar nos saldrá vacía la etiqueta, por lo siguiente será mover el cursor con nuestro mouse.



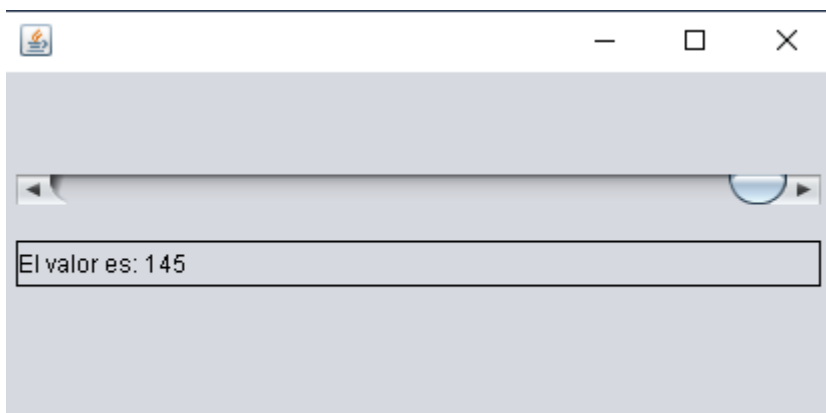
Si seleccionamos la flechita de los costados ira aumentando de 2, pero si presionamos en la barra ira de 30.



Como se ve paso de 54 a 74, debido a que se configuro de esa manera en nuestra parte de propiedades del JScrollBar

El máximo:

Debido a que todo el scrollbar es de un tamaño de 150, se le resta el tamaño de la parte donde se genera.



Mínimo:



Como se puede apreciar el Scrollbar es un componente muy diferente a los demás que hemos usados ya que se comparte de una manera distinta, por lo que se puede decir que dependiendo de la actividad que vayas a realizar debes saber que componente será para la mejor para tu actividad.