

**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: EJERCICIOS DE LENGUAJE JAVA 1 Y 2.

GRUPO: ISU

HORA: 09:00-10:00 AM

DOCENTE: M.C. LUIS ALBERTO ALONSO HERNÁNDEZ

ALUMNO: GONZALEZ PASCUAL MELVIN PAUL

FECHA DE ENTREGA: 14 DE OCTUBRE DEL 2020

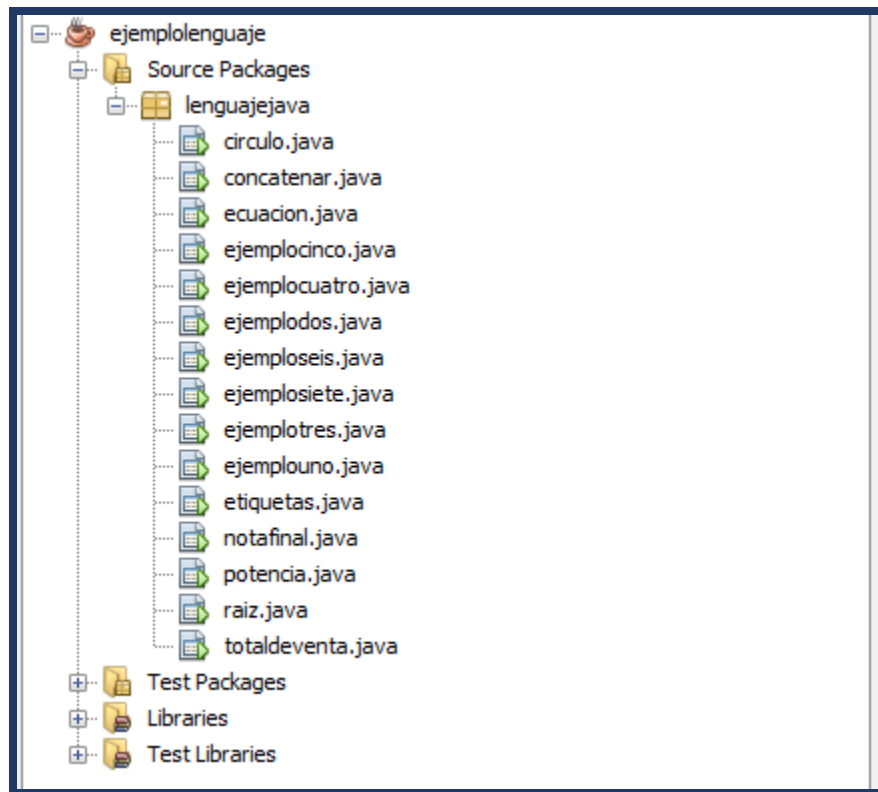
Contenido

Introducción:	3
Ejercicios de Lenguaje Java 1.	4
Ejercicio1.	4
Ejercicio 2.	6
Ejercicio 3.	8
Ejercicio 4.	10
Ejercicio 5.	13
Ejercicio 6.	15
Ejercicio 7.	18
Ejercicios de Lenguaje Java 2.	21
Ejercicio 1.	21
Ejercicio 2.	24
Ejercicio 3.	26
Ejercicio 4.	28
Ejercicio 5.	30
Ejercicio 6.	32
Ejercicio 7.	34
Ejercicio 8.	37

Introducción:

Se elaborará cada de uno del ejercicio proporcionado de los PDF, lo cual se harán mediante la GUI NetBeans, los cuales se crearon mediante los componentes, se validarán los datos necesarios, así mismo se hará un diseño amigable con el usuario.

Se creará un nuevo proyecto el cual tendrá su propio paquete llamado “lenguaje java”, donde se encontrara los diversos ejercicios realizados.

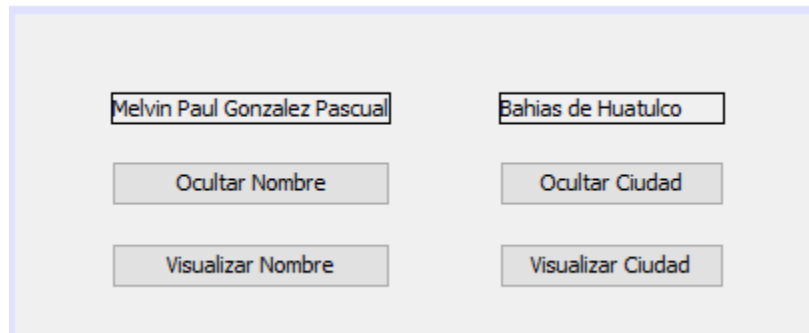


Ejercicios de Lenguaje Java 1.

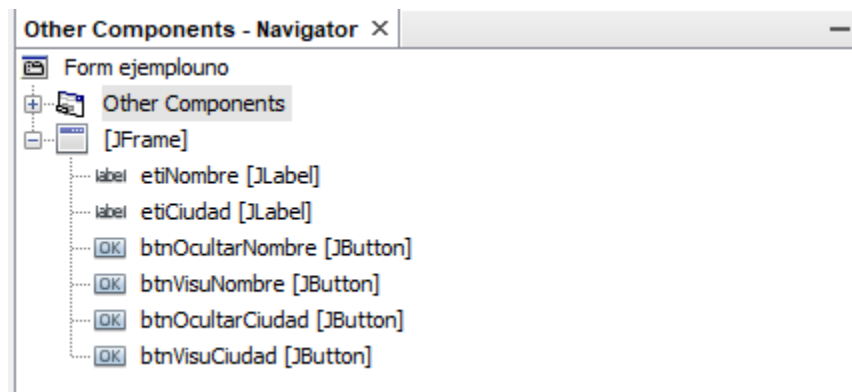
Ejercicio1.

Para nuestro primer ejercicio nos enfocaremos en el metodo **setVisible** el cual nos visualiza el componente que deseemos. En este programa usaremos dos JLabel, al igual que cuatro JButtons, los cuales fueron acomodados de la siguiente manera.

Las etiquetas contendrán nuestro nombre y ciudad, mientras que los botones las acciones a realizar.



Los componentes serán renombrados de la siguiente manera, ya que es una forma más profesional de usarlas.



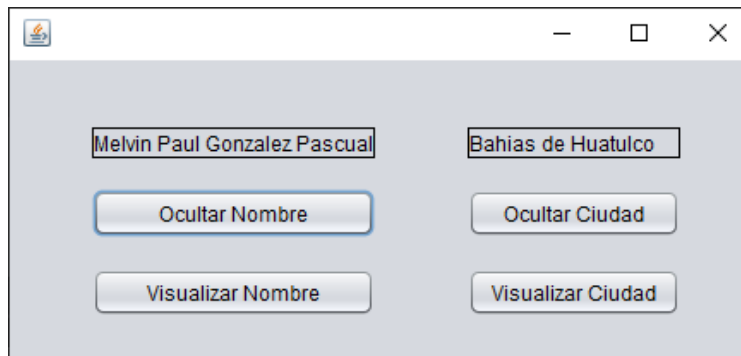
En la parte de la codificación, se implementó el evento de **ActionPerformed** en cada uno de los botones, además se usó el método de **setVisible**.

```
1 //Acción del Botón para visualizar la etiqueta de texto
2 private void btnVisuNombreActionPerformed(java.awt.event.ActionEvent evt) {
3     etiNombre.setVisible(true);
4 }
5 //Acción del Botón para ocultar la etiqueta de texto
6 private void btnOcultarNombreActionPerformed(java.awt.event.ActionEvent evt) {
7     etiNombre.setVisible(false);
8 }
9 //Acción del Botón para visualizar la etiqueta de texto
10 private void btnOcultarCiudadActionPerformed(java.awt.event.ActionEvent evt) {
11     etiCiudad.setVisible(false);
12 }
13 //Acción del Botón para ocultar la etiqueta de texto
14 private void btnVisuCiudadActionPerformed(java.awt.event.ActionEvent evt) {
15     etiCiudad.setVisible(true);
16 }
```

Ejecución de nuestro programa.

Al momento de presionar los botones de ocultar ciudad y nombre, la etiquetas con el nombre y la ciudad desaparecen, dejando vacío la parte de arriba, a lo que, si presionamos los botones de visualizar ciudad y nombre, vuelven a aparecer las etiquetas de texto, como se muestra a continuación.

Nuestro programa al momento de iniciar, como se puede apreciar nos sale con todos los componentes integrados.



Ocultando nombre.



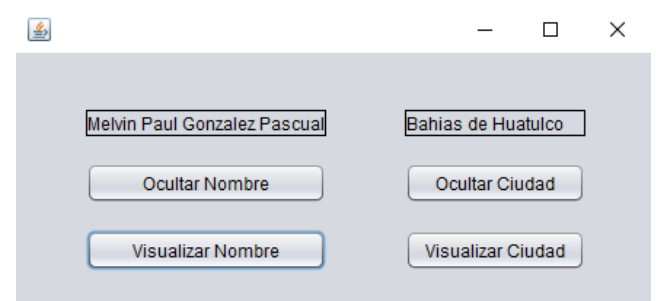
Visualizando ciudad.



Ocultando ciudad.



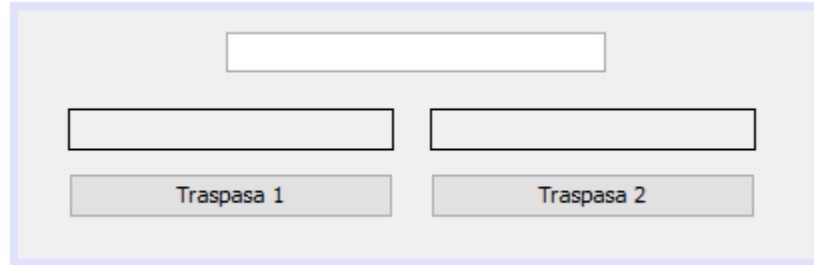
Visualizando nombre.



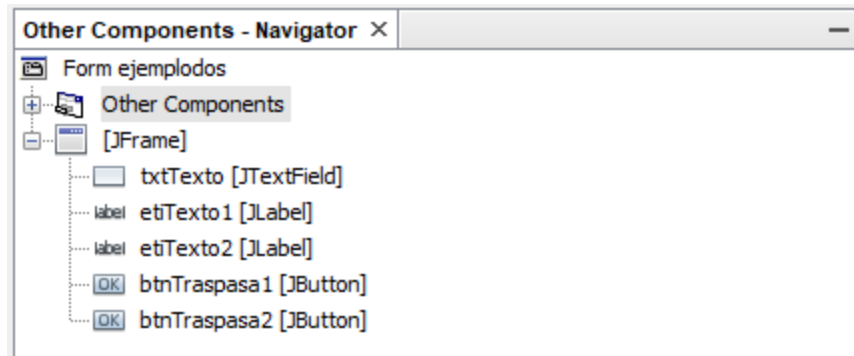
Como se puede apreciar cada uno de los botones realiza una acción diferente lo cual es mostrar/ocultar los textos.

Ejercicio 2.

En el ejercicio a continuación rellenaremos las etiquetas con el campo de texto, en cual tenemos dos opciones, para cada etiqueta que tenemos. Se ocuparán dos JLabel, un JTextField y dos JButtons, los cuales serán colocados de la siguiente manera.



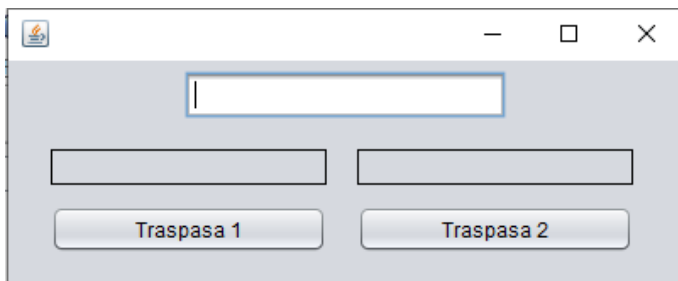
Mientras que sus variables tendrán el siguiente nombre.



En nuestra parte de la codificación usaremos el método **setText** el cual nos recoge el texto, pero antes debemos usar el método de **getText()**, el cual nos trae el texto que tenga la variable asignada, para esto, debemos generar el evento de **ActionPerformed**, todo esto quedara de la siguiente manera.

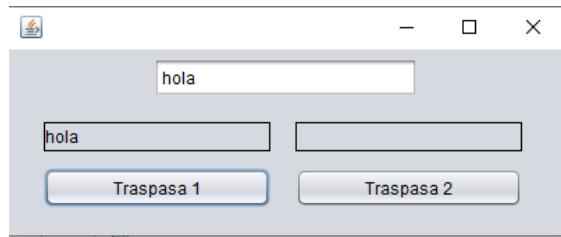
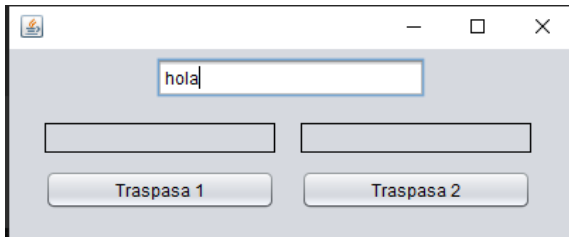
```
//Accción del Botón para generar la cadena en en el TextField.  
private void btnTraspasa1ActionPerformed(java.awt.event.ActionEvent evt) {  
    etiTexto1.setText(txtTexto.getText());  
}  
//Accción del Botón para generar la cadena en en el TextField.  
private void btnTraspasa2ActionPerformed(java.awt.event.ActionEvent evt) {  
    etiTexto2.setText(txtTexto.getText());  
}
```

La siguiente parte será la ejecución de nuestro programa.

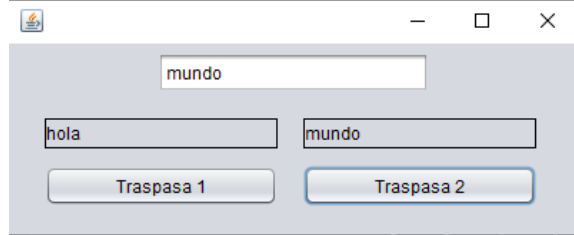
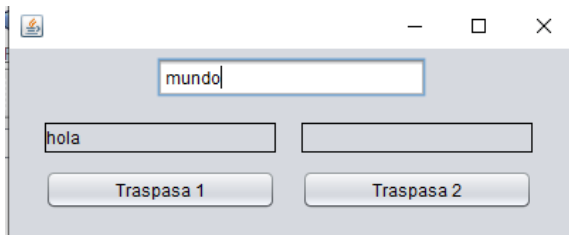


Al momento de ejecutar nos saldrá lo siguiente.

Ahora traspasaremos un texto a nuestra primera etiqueta.



Como siguiente a la etiqueta dos.

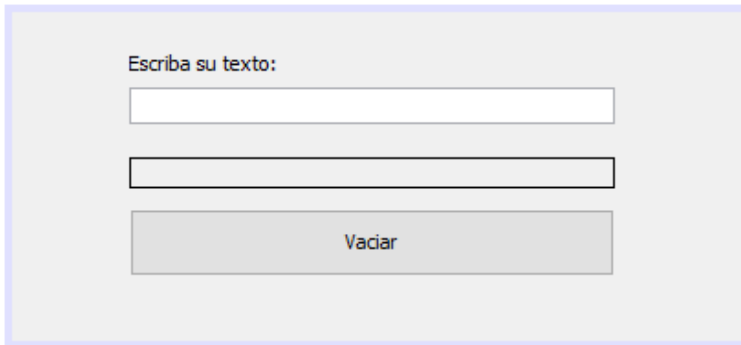


Como se puede apreciar la funcionalidad del programa es escribir en nuestro campo de texto y poder escoger en que lado de la etiqueta traspasar nuestro texto.

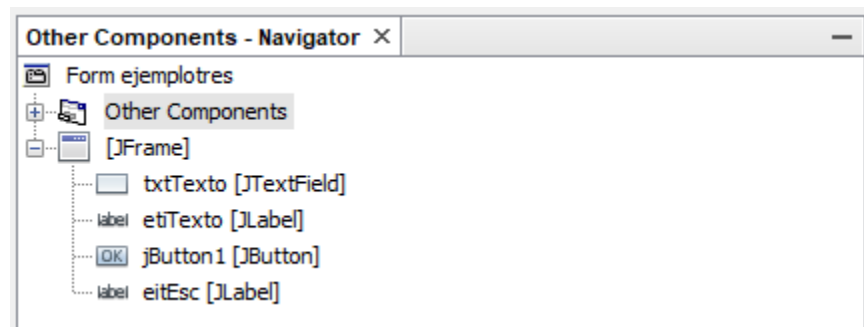
Ejercicio 3.

Nuestro siguiente ejercicio consiste en escribir en nuestro campo de texto y que te forma automática se escriba en nuestra etiqueta de texto, lo cual usaremos el evento de **keyPressed** , se ocuparan los siguientes componentes.

Nuestro diseño es el siguiente.



El nombre de nuestras variables será las siguiente.



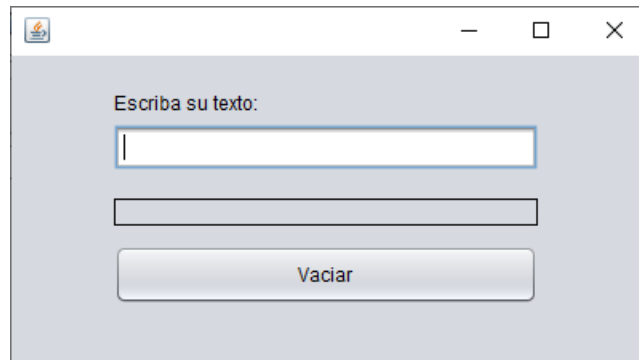
Para que nuestro programa haga lo que se indique deberemos agregar el evento de **keyPressed** de la siguiente manera. Además de utilizar los métodos de setText y getText ().

```
private void txtTextoKeyPressed(java.awt.event.KeyEvent evt) {  
    etiTexto.setText(txtTexto.getText());  
    // TODO add your handling code here:  
}
```

También un botón eliminar de la siguiente manera, con el evento de **ActionPerformed**. Utilizando el método de setText, el cual ira con una cadena vacía.

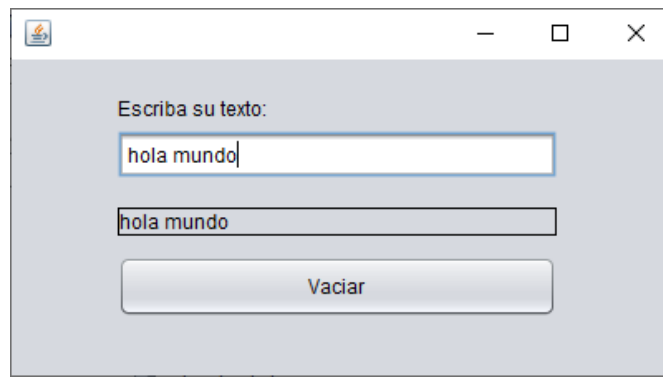
```
//Boton vaciar que nos borra el contenido de nuestro campo de texto y etiqueta.  
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    txtTexto.setText("");  
    etiTexto.setText("");  
    // TODO add your handling code here:  
}
```


Al momento de ejecutar el programa nos saldrá lo siguiente.



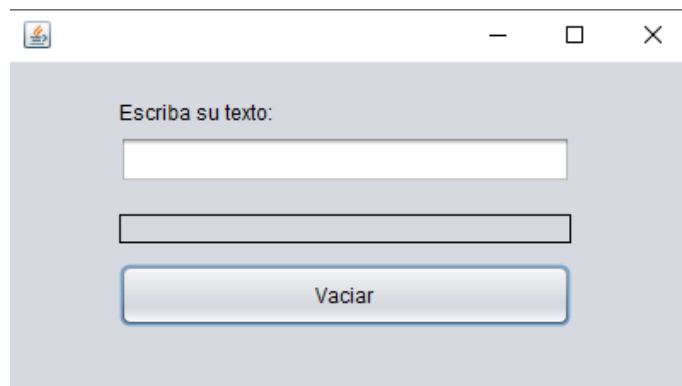
A screenshot of a Windows application window. The window has a title bar with a small icon on the left and standard minimize, maximize, and close buttons on the right. The main area of the window has a light gray background. At the top, the text "Escriba su texto:" is displayed. Below this text is a white text input field with a blue border. Underneath the input field is a disabled text field, which is gray and contains no text. At the bottom of the window is a button with a gradient background and the text "Vaciar".

Al momento de ingresar alguna tecla, deberá aparecer en nuestra etiqueta, de la siguiente manera.



A screenshot of the same application window. The text input field now contains the text "hola mundo". The disabled text field below it also contains the text "hola mundo". The "Vaciar" button remains at the bottom.

Como siguiente paso usaremos nuestro botón borrar como su nombre lo indica, va a borrar el campo de texto al igual que el contenido de la etiqueta.



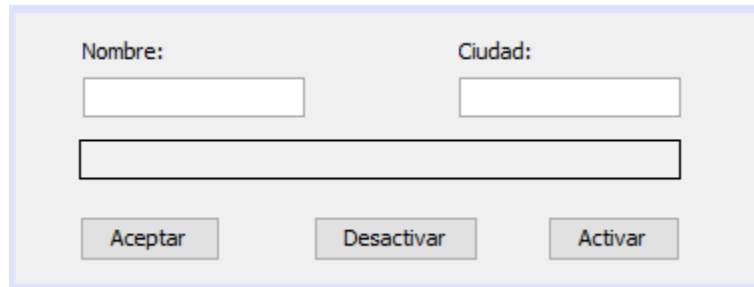
A screenshot of the application window after the "Vaciar" button has been clicked. The text input field is now empty. The disabled text field below it is also empty. The "Vaciar" button is still present at the bottom.

Como se pudo observar el evento de KeyPressed nos ayuda en muchas acciones, ya que su funcionamiento es en todos los lados.

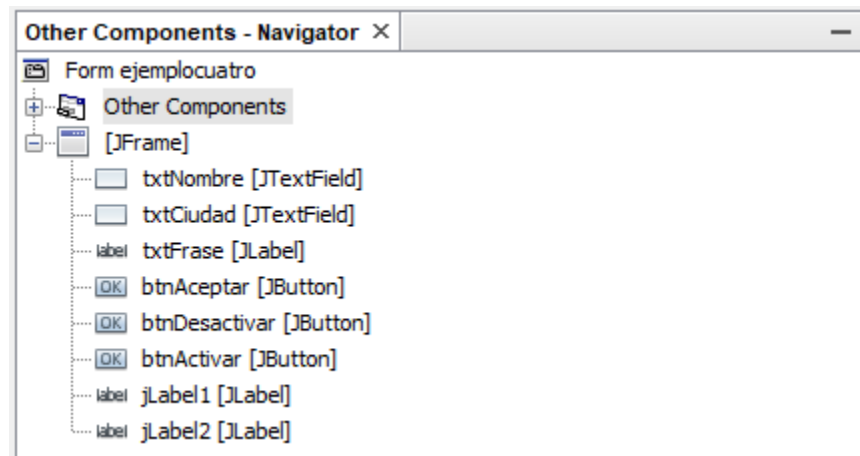
Ejercicio 4.

El siguiente ejercicio trata de agregar dos campos de textos en los cuales insertaremos nuestro nombre y nuestra ciudad, y al momento de presionar de aceptar, nos juntara ambos datos y creara una nueva cadena de texto en la cual estará en nuestra etiqueta de texto, además se crearán dos botones los cuales activaran y desactivar que nuestro sea editable con el siguiente método **setEnabled**, además se validaran los campos de texto, en los cuales solo se podrán ingresar letras.

Nuestro diseño de la interfaz ser la de la siguiente manera.



La cual sus variables tendrán los siguientes nombres.



Para la validación se usará el evento de **KeyTyped** en nuestros campos de texto, pero antes de eso, crearemos un método, ya que nos ahorraremos líneas de códigos.

El cual al momento de ingresar algo distinto a una letra, nos lanzara un sonido y un mensaje de alerta indicando que esta mal la forma de ingresar.

```
13 public void verif(KeyEvent evt){
14     char validar = evt.getKeyChar();
15     if(Character.isDigit(validar)){
16         getToolkit().beep();
17         evt.consume();
18         JOptionPane.showMessageDialog(rootPane, "Ingrese solo letras");
19     }
20 }
```

Como siguiente parte, mandaremos el método de validación en los campos de texto de la siguiente manera.

```
//Validación de datos.  
private void txtNombreKeyTyped(java.awt.event.KeyEvent evt) {  
    verif(evt);  
    // TODO add your handling code here:  
}  
  
//Validación de datos.  
private void txtCiudadKeyTyped(java.awt.event.KeyEvent evt) {  
    verif(evt);  
    // TODO add your handling code here:  
}
```

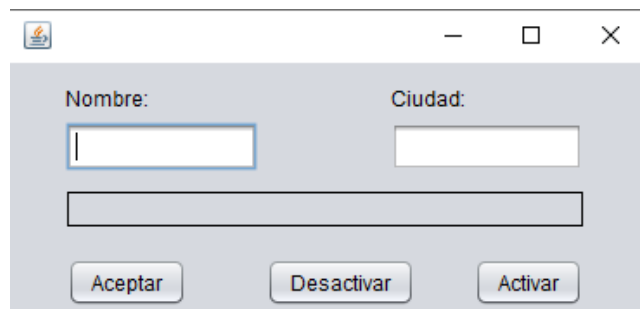
También agregaremos los siguientes métodos a los botones de activar y desactivar.

```
//Acción de bloquear nuestros campos de texto  
private void btnDesactivarActionPerformed(java.awt.event.ActionEvent evt) {  
    txtNombre.setEnabled(false);  
    txtCiudad.setEnabled(false);  
    // TODO add your handling code here:  
}  
  
//Acción de hacer editable nuestros campos de texto  
private void btnActivarActionPerformed(java.awt.event.ActionEvent evt) {  
    txtNombre.setEnabled(true);  
    txtCiudad.setEnabled(true);  
    // TODO add your handling code here:  
}
```

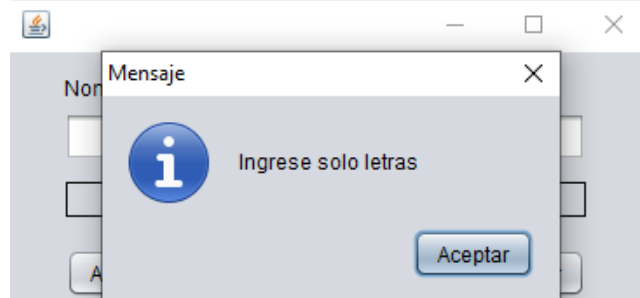
Lo que hace el método `setEnabled` es poder hacer que nuestro método sea editable.

Como último en nuestra etiqueta los métodos para hacer que los datos de nuestros campos se unan.

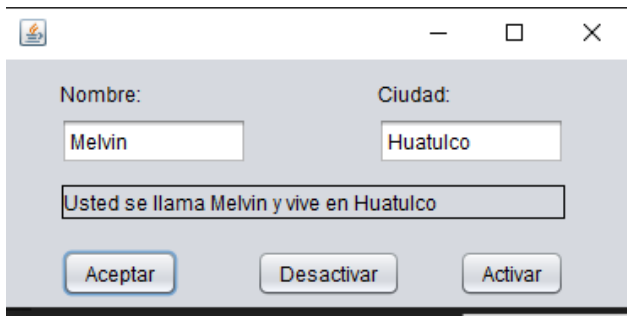
Cuando ejecutamos nos saldrá de la siguiente manera.



Al momento que ingresamos un carácter diferente a una letra nos saldrá lo siguiente.



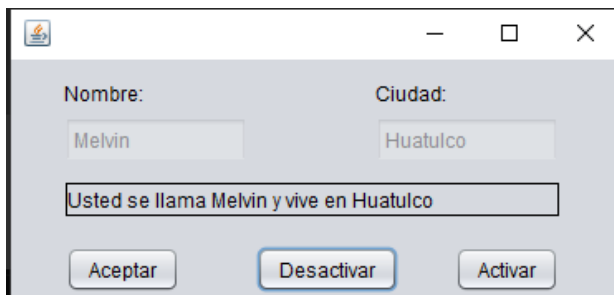
Por lo cual agregaremos texto en ambos campos de texto.



A screenshot of a Java Swing window titled 'Formulario'. It contains two text input fields labeled 'Nombre:' and 'Ciudad:'. The 'Nombre:' field contains the text 'Melvin' and the 'Ciudad:' field contains 'Huatulco'. Below these fields is a text area containing the text 'Usted se llama Melvin y vive en Huatulco'. At the bottom of the window are three buttons: 'Aceptar', 'Desactivar', and 'Activar'.

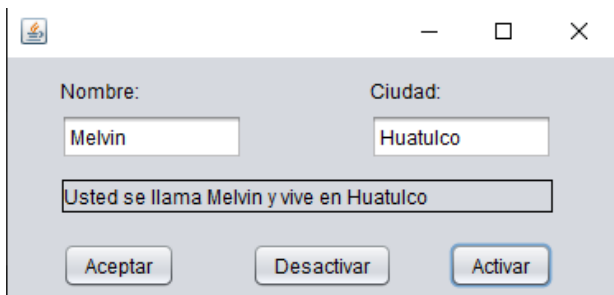
Como se puede observar se crea una cadena con ambos datos seleccionados.

Al momento de presionar los botones de activar y desactivar nos saldrá lo siguiente.



A screenshot of the same Java Swing window. The 'Desactivar' button is highlighted with a blue border, indicating it is the active element. The text fields and text area remain unchanged.

Al momento de presionar el botón de desactivar nos deshabilita la opción de escribir debido al método de SetEnable.



A screenshot of the same Java Swing window. The 'Activar' button is highlighted with a blue border, indicating it is the active element. The text fields and text area remain unchanged.

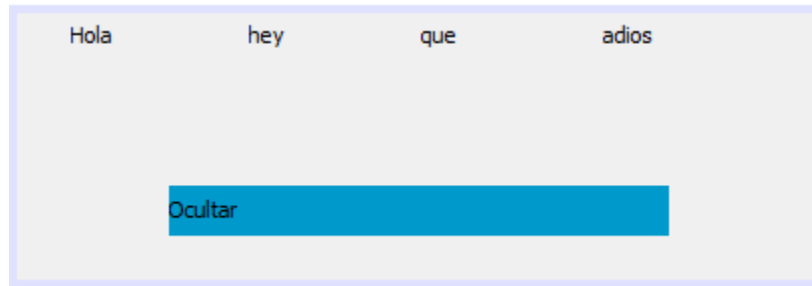
Al momento de presionar el botón de activar nos dejara nuevamente escribir.

Por lo cual se da concluido este ejercicio.

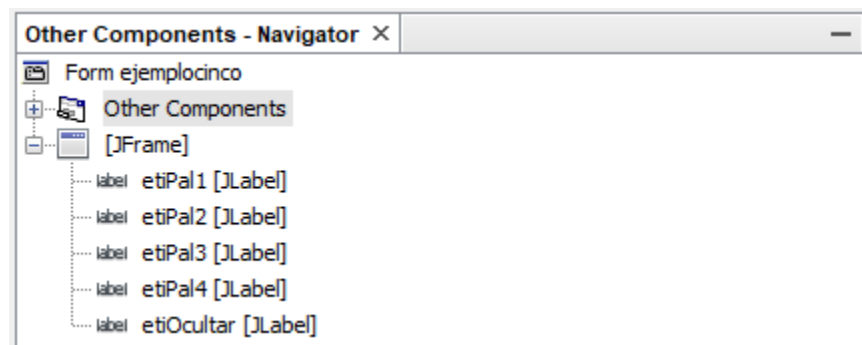
Ejercicio 5.

El siguiente ejercicio es muy sencillo, pero no por eso no se le debe menospreciar, lo que haremos es que cuando nuestro mouse pase la etiqueta con color, estas se ocultarán, pero al momento de que el mouse salga de la etiqueta de color, serán visibles de nueva, esto se hará con los eventos de mouse los cuales son los siguiente.

Se creará el siguiente diseño el cual es una etiqueta con un fondo de color, al igual que etiquetas con algún texto, como se muestra a continuación.



Con las variables con el siguiente nombre.



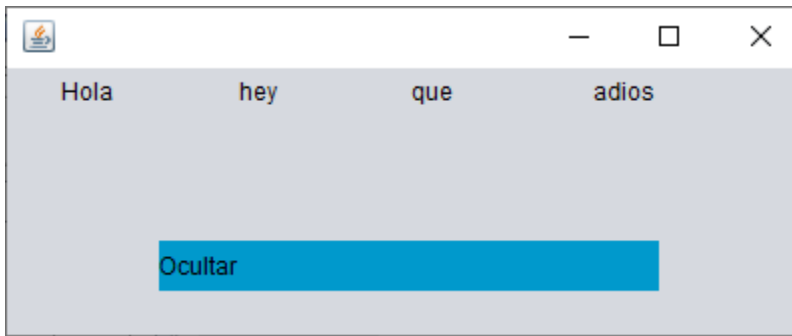
El evento de **MouseEntered** se programará de la siguiente manera, utilizando el método de setVisible de la siguiente manera.

```
//Evento que oculta las etiquetas.  
private void etiOcultarMouseEntered(java.awt.event.MouseEvent evt) {  
    etiPal1.setVisible(false);  
    etiPal2.setVisible(false);  
    etiPal3.setVisible(false);  
    etiPal4.setVisible(false);  
    // TODO add your handling code here:  
}
```

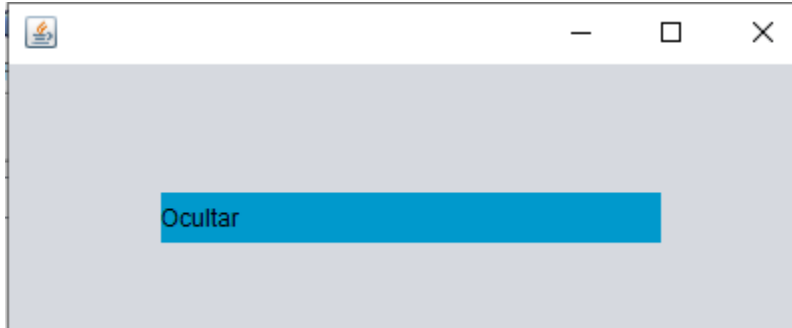
El evento de **MouseExited** hará que al momento que se salga el mouse de la etiqueta haga las siguientes acciones.

```
//Evento que hace visible las etiquetas.  
private void etiOcultarMouseExited(java.awt.event.MouseEvent evt) {  
    etiPal1.setVisible(true);  
    etiPal2.setVisible(true);  
    etiPal3.setVisible(true);  
    etiPal4.setVisible(true);  
    // TODO add your handling code here:  
}
```

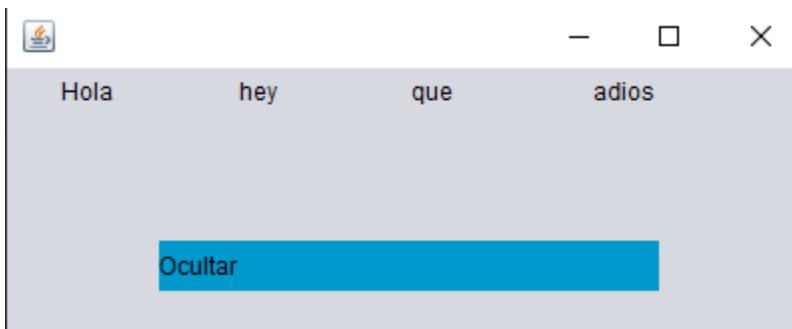
Al momento de ejecutar nuestro programa nos saldrá de la siguiente manera.



Nuestra pantalla al momento de iniciar.



Cuando nuestro mouse esta dentro de la etiqueta azul, las otras etiquetas desaparecen.



Cuando nos salimos de la etiqueta se vuelve visible.

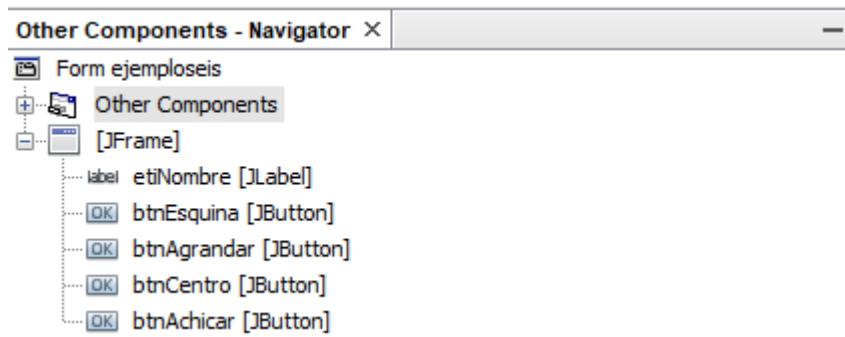
Con esto damos concluido este ejercicio.

Ejercicio 6.

El siguiente programa se harán varios botones con cada uno con su propia acción, las cuales serán, mandar el texto a la esquina, mandar el texto al centro, hacer grande el texto, hacer pequeño el texto, se creará de la siguiente manera. Además, se usarán los eventos de MouseEntered y MouseExited, los cuales nos ayudara para que cada vez que nuestro mouse pase por uno de los botones esto crezca de tamaño al igual que cuando salen, se harán más pequeño.



Y las variables de se renombran de la siguiente manera.



Lo siguiente será programar las acciones de los botones de agrandar y hacer pequeño el texto, los cuales se harán con el método de setLocation, además del evento de ActionPerformed, como se muestra a continuación.

```
//Metodo que nos manda la etiqueta a la esquina.  
private void btnEsquinaActionPerformed(java.awt.event.ActionEvent evt) {  
    etiNombre.setLocation(1, 1);  
    // TODO add your handling code here:  
}  
  
//Metodo que nos centra la etiqueta.  
private void btnCentroActionPerformed(java.awt.event.ActionEvent evt) {  
    etiNombre.setLocation(110,145);  
    // TODO add your handling code here:  
}
```

Lo siguiente son los botones de agrandar y enchicar por lo cual usaremos el evento de ActionPerformed además del método setSize, el cual nos hace poder modificar el tamaño de nuestro componente.

```
//Metodo que nos hace mas grande la etiqueta
private void btnAgrandarActionPerformed(java.awt.event.ActionEvent evt) {
    etiNombre.setSize(200,50);
    // TODO add your handling code here:
}
//Metodo que nos hace mas paqueña la etiqueta
private void btnAchicarActionPerformed(java.awt.event.ActionEvent evt) {
    etiNombre.setSize(80,20);
    // TODO add your handling code here:
}
```

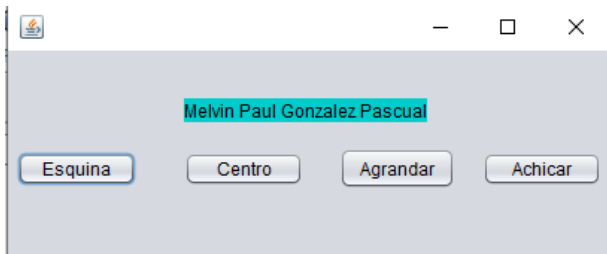
A continuación, se programará la acción de cuando el mouse salga de los botones regresen a su tamaño real.

```
//Metodo que nos regresa al tamaño de nuestro boton.
private void btnEsquinaMouseEntered(java.awt.event.MouseEvent evt) {
    btnEsquina.setSize(90, 30);
    // TODO add your handling code here:
}
//Metodo que nos regresa al tamaño de nuestro boton.
private void btnCentroMouseEntered(java.awt.event.MouseEvent evt) {
    btnCentro.setSize(90, 30);
    // TODO add your handling code here:
}
//Metodo que nos regresa al tamaño de nuestro boton.
private void btnAgrandarMouseEntered(java.awt.event.MouseEvent evt) {
    btnAgrandar.setSize(90, 30);
    // TODO add your handling code here:
}
//Metodo que nos regresa al tamaño de nuestro boton.
private void btnAchicarMouseEntered(java.awt.event.MouseEvent evt) {
    btnAchicar.setSize(90, 30);
    // TODO add your handling code here:
}
```

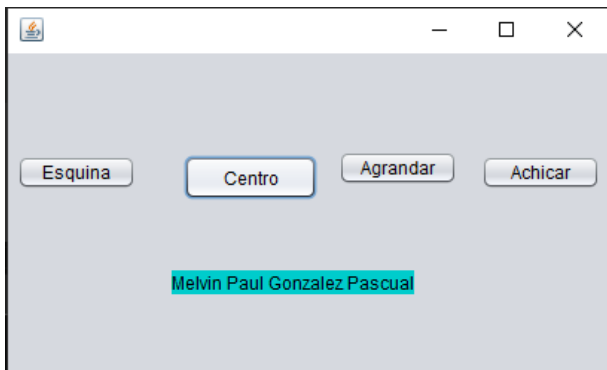
Lo siguiente es hacer que cuando el mouse este por los botones estos sean de mayor tamaño, por lo cual será de la siguiente manera.

```
//Metodo que nos hace incrementar el tamaño del boton.
private void btnEsquinaMouseExited(java.awt.event.MouseEvent evt) {
    btnEsquina.setSize(80,23);
    // TODO add your handling code here:
}
//Metodo que nos hace incrementar el tamaño del boton.
private void btnCentroMouseExited(java.awt.event.MouseEvent evt) {
    btnCentro.setSize(80,23);
    // TODO add your handling code here:
}
//Metodo que nos hace incrementar el tamaño del boton.
private void btnAgrandarMouseExited(java.awt.event.MouseEvent evt) {
    btnAgrandar.setSize(80,23);
    // TODO add your handling code here:
}
//Metodo que nos hace incrementar el tamaño del boton.
private void btnAchicarMouseExited(java.awt.event.MouseEvent evt) {
    btnAchicar.setSize(80,23);
    // TODO add your handling code here:
}
```

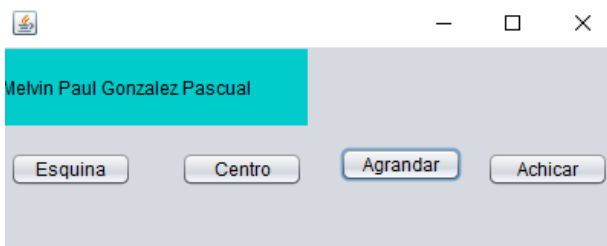

Al momento de ejecutar nuestro programa nos saldrá de la siguiente manera.



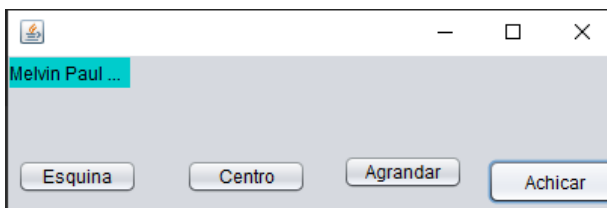
Como se puede apreciar no tiene ningún cambio por lo cual se harán los siguientes.



Cuando presionamos el botón de centra este no los manda al centro, además el botón crece de tamaño, pero al momento de salir regresa a su normalidad.



Al momento de presionar el botón de agrandar nuestro componente crecer de tamaño.



Cuando presionamos el botón de achicar este se hace de un tamaño muy pequeño, tanto que no nuestro texto no cabe en la etiqueta.

Como se puede ver los métodos de setSize y setLocation, nos ayudan para poder realizar cada uno de los ejercicios.

Ejercicio 7.

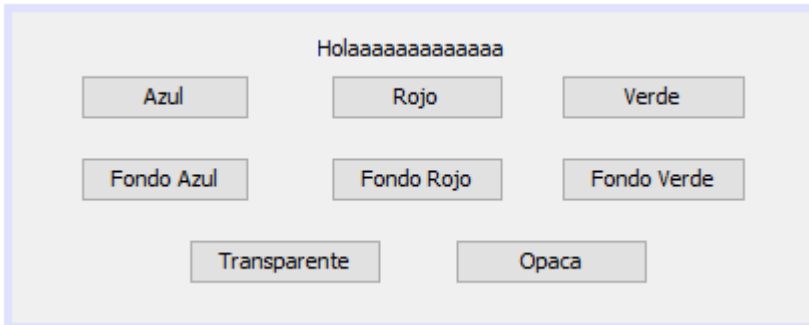
El siguiente ejercicio usaremos los métodos de `setForeground`, `setBackground` y `setOpaque`.

`setForeground`: Nos permite cambiar el color del contenido de nuestra etiqueta.

`setBackground`: Nos permite cambiar el fondo de la etiqueta, pero para que pueda funcionar el método de `setOpaque` debe estar activado.

`setOpaque`: Nos permite activar o desactivar lo opaco de una etiqueta.

En el siguiente programa cambiaremos el color de letra y fondo con los respectivos botones, además de poder manipular si es opaco o no, el diseño será el siguiente.



La variable de los componentes quedara de la siguiente forma.



Se programarán los siguientes métodos en los botones, utilizando el evento de `ActionPerformed`.

El método para que nuestros colores cambian dependiendo el botón que sea seleccionado.

```
//Metodo para cambiar el color
private void btnAzulActionPerformed(java.awt.event.ActionEvent evt) {
    etiTexto.setForeground(Color.BLUE);
    // TODO add your handling code here:
}

private void btnRojoActionPerformed(java.awt.event.ActionEvent evt) {
    etiTexto.setForeground(Color.RED);
    // TODO add your handling code here:
}

private void btnVerdeActionPerformed(java.awt.event.ActionEvent evt) {
    etiTexto.setForeground(Color.GREEN);
    // TODO add your handling code here:
}
```

Métodos para cambiar el color de fondo nuestras etiquetas.

```
//Cambia el color del fondo de las etiquetas.  
private void btnFondoAzulActionPerformed(java.awt.event.ActionEvent evt) {  
    etiTexto.setBackground(Color.BLUE);  
    // TODO add your handling code here:  
}  
  
private void btnFondoRojoActionPerformed(java.awt.event.ActionEvent evt) {  
    etiTexto.setBackground(Color.RED);  
    // TODO add your handling code here:  
}  
  
private void btnFondoVerdeActionPerformed(java.awt.event.ActionEvent evt) {  
    etiTexto.setBackground(Color.GREEN);  
    // TODO add your handling code here:  
}
```

También la opción que las etiquetas sean opacas las cuales quedan de la siguiente manera.

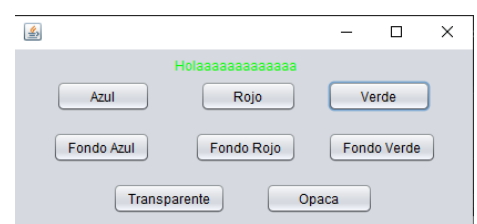
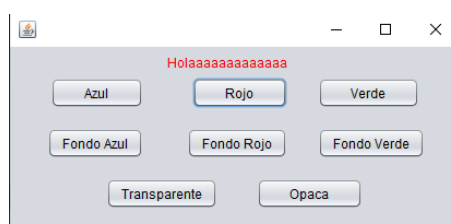
```
//Metodo para cambiar si son opaca o no.  
private void btnTransparenteActionPerformed(java.awt.event.ActionEvent evt) {  
    etiTexto.setOpaque(false);  
    // TODO add your handling code here:  
}  
  
private void btnOpacaActionPerformed(java.awt.event.ActionEvent evt) {  
    etiTexto.setOpaque(true);  
    // TODO add your handling code here:  
}
```

Al momento de ejecutar nuestro programa nos saldrá de la siguiente manera.

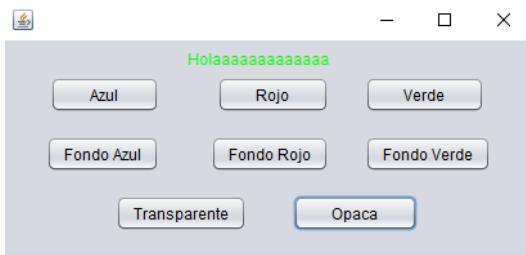


Como se puede observar nuestra etiqueta nos sale de color de negro ya que es color que se trae por defecto.

Cuando presionamos los botones con el color correspondiente nos los cambiara a ese color, como se muestra a continuación.

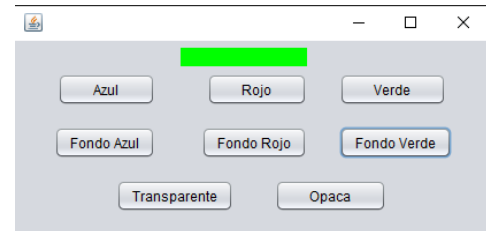
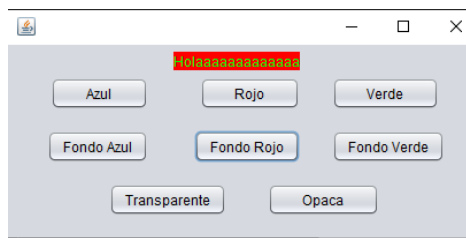


Si queremos cambiar el fondo de nuestro texto no se podrá, debido que se encuentra desactivado lo opaco, por lo cual primero debemos activar.



Se activa la opción para realizar lo siguiente.

Ya podemos cambiar el fondo de los colores a escoger.



Como se puede observar cada método tiene su funcionalidad, además que cada uno depende de otro, por lo cual se da como concluido la práctica.

Ejercicios de Lenguaje Java 2.

Ejercicio 1.

Se realizar un programa que calcules el total de ventas de los cuales se calcular el total, el iva y el total más IVA.

Los campos de texto de unidades y calcular serán los que desarrollen todo el proceso, por lo cual crearemos el siguiente diseño.

Unidades:

Precio:

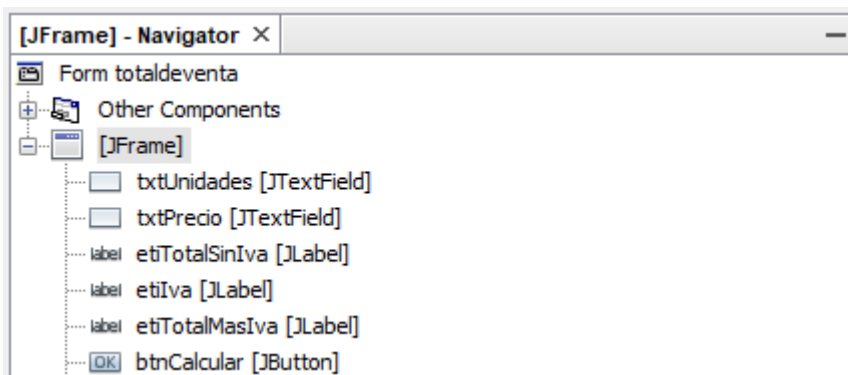
Total sin Iva

IVA:

Total mas Iva

Calcular

Con el nombre de las variables siguientes.



La parte de los cálculos será la siguiente.

Crearemos un método para calcular todo nuestras opciones, la cual es el siguiente.

```
public void generar() {  
    String cad1, cad2;  
    int a, b, s, iva;  
    cad1 = txtPrecio.getText();  
    cad2 = txtUnidades.getText();  
    a = Integer.parseInt(cad1);  
    b = Integer.parseInt(cad2); //Conversion de string a numero.  
    s = a * b; //Calcular el total sin iva  
    iva = (int) (s * 0.16); //Calcular el iva  
    etiTotalSinIva.setText("" + s); //Manda el total de venta.  
    etiIva.setText("" + iva); //Manda a la etiqueta el iva de la venta  
    etiTotalMasIva.setText("" + (s + iva)); //Manda a la etiqueta el total mas iva  
}
```

Además de un método para validar que solo acepte enteros.

```
//Metodo para verificar que sean solos numeros y mayores a 0;
public void verif(KeyEvent evt){
    char validar = evt.getKeyChar();
    if(validar<'0' || validar>'9'){
        evt.consume();
        getToolkit().beep();
    }
    if(Character.isLetter(validar)){
        getToolkit().beep();
        evt.consume();

        JOptionPane.showMessageDialog(rootPane, "Ingrese solo numeros");
    }
}
```

A continuación mandaremos a llamar los métodos creado en los eventos de los campos de texto y en nuestro botón calcular.

Para el botón.

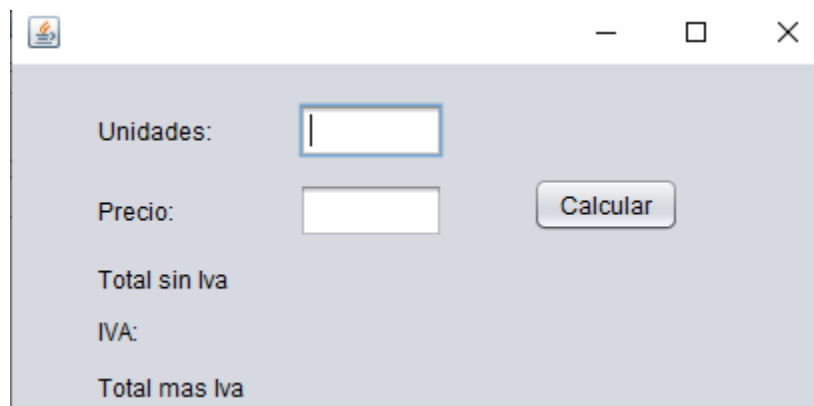
```
//Evento en cual se manda a llamar el metodo.
private void btnCalcularActionPerformed(java.awt.event.ActionEvent evt) {
    generar();
    // TODO add your handling code here:
}
```

Para la validación.

```
//Evento para verificar;
private void txtUnidadesKeyTyped(java.awt.event.KeyEvent evt) {
    verif(evt);
    // TODO add your handling code here:
}

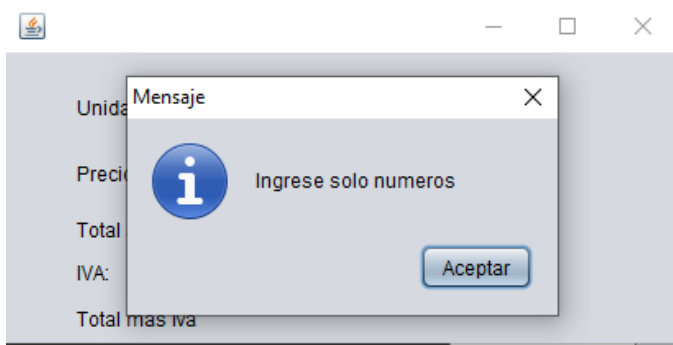
//Evento para verificar la entrada.
private void txtPrecioKeyTyped(java.awt.event.KeyEvent evt) {
    verif(evt);
}
```

Compilaremos nuestro programa y nos deberá arrojar lo siguiente.



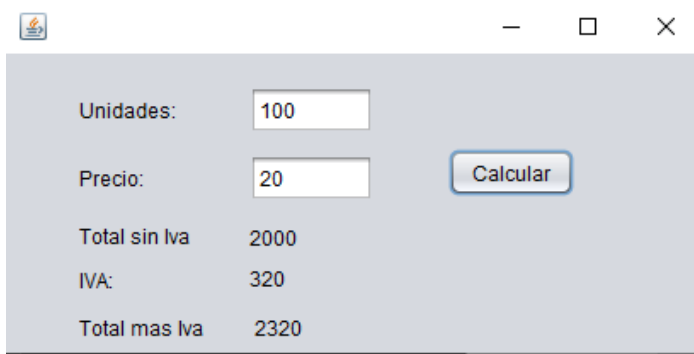
Nos aparecerá el diseño creado anteriormente.

Al momento de ingresar una letra o un numero negativo nos mostrara lo siguiente.



Lo cual nos indica que solo puede número positivos.

Haremos el siguiente calculo.

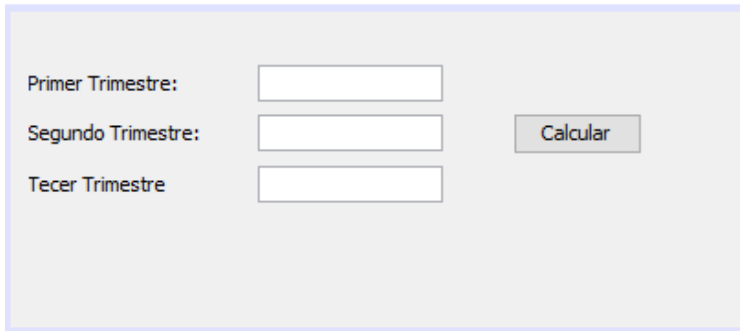


El siguiente calculo fue realizado con perfección por lo cual podemos concluir que nuestro programa fue muy bien realizado.

Ejercicio 2.

El siguiente ejercicio es para calcular el promedio de una nota, el cual nos pedirá 3 calificaciones, cuando el promedio sea menor a 5, nos mandara una alerta que ha sido suspendido, pero si es mayor, entonces nos mandara el promedio.

Por lo cual se crea el siguiente diseño.



Primer Trimestre:

Segundo Trimestre:

Tercer Trimestre:

Calcular

Se crearon un método para generar el cálculo y otros eventos para la validación de los datos.

Para el cálculo es el siguiente.

```
public void promedio() {
    String cad1, cad2, cad3;
    int a, b, c;
    cad1 = txtPrimerTrimestre.getText();
    cad2 = txtSegundoTrimestre.getText();
    cad3 = txtTercerTrimestre.getText();
    a = Integer.parseInt(cad1);
    b = Integer.parseInt(cad2);
    c = Integer.parseInt(cad3);
    int total = (a + b + c) / 3; // Saca el promedio de las calificaciones
    // Si es menor a 5 nos manda una alerta, donde se vea que fue reprobado.
    if (total < 5) {
        etiResultado.setText("SUSPENSO");
        etiNotaFinal.setText(" " + total);
        etiResultado.setForeground(Color.RED);
        etiNotaFinal.setForeground(Color.RED);
    }
    // Si es mayor a 5, nos mandara que si lo estamos.
    if (total >= 5) {
        etiResultado.setText("APROBADO");
        etiNotaFinal.setText(" " + total);
        etiResultado.setForeground(Color.BLACK);
        etiNotaFinal.setForeground(Color.BLACK);
    }
}
```


Para la validación de los datos será la siguiente con el evento de KeyTyped.

```
//Validacion de enteros
] private void txtSegundoTrimestreKeyTyped(java.awt.event.KeyEvent evt) {
    char c= evt.getKeyChar();
    if(c<'0' || c>'9') evt.consume();
    if (txtSegundoTrimestre.getText().length()>1){evt.consume();}
- }

] private void txtPrimerTrimestreKeyTyped(java.awt.event.KeyEvent evt) {
    char c= evt.getKeyChar();
    if(c<'0' || c>'9') evt.consume();
    if (txtPrimerTrimestre.getText().length()>1)evt.consume();
- }

] private void txtTercerTrimestreKeyTyped(java.awt.event.KeyEvent evt) {
    char c= evt.getKeyChar();
    if(c<'0' || c>'9') evt.consume();
    if (txtTercerTrimestre.getText().length()>1)evt.consume();
- }
```

Se creará el evento de ActionPerformed para el botón calcular.

```
//Evento para generar las calificaciones.
private void btnCalcularActionPerformed(java.awt.event.ActionEvent evt) {
    promedio();
    // TODO add your handling code here:
}
}
```

Al momento de ejecutar nuestro programa nos deberá aparecer el siguiente diseño.

Haremos de un calculo de un alumno aprobado.

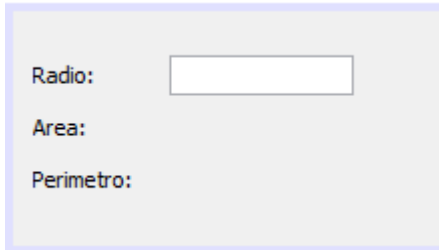
Cálculo de no aprobado.

Como se puede apreciar nuestro calculo fue realizado correctamente.

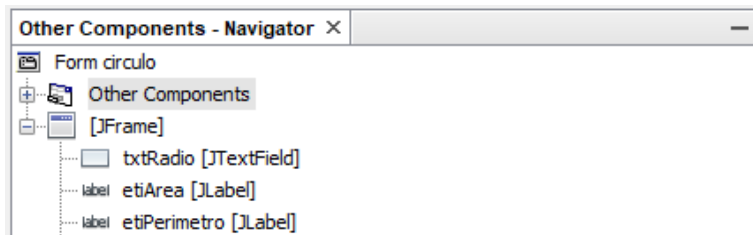
Ejercicio 3.

En el siguiente ejercicio se calculará el área y perímetro de un círculo mediante el radio que sea proporcionado, además se usará el evento `KeyPressed`, el cual hace que los cálculos sean automáticamente. Si el usuario ingresa número negativos, nos mostrará un error.

El diseño será el siguiente.



Y las variables tendrán el siguiente nombre.



Se creará un método para hacer los cálculos, el cual será el siguiente.

```
//Metodo para realizar los calculos
public void calcular() {
    int p, a;
    String s = txtRadio.getText();
    int r = Integer.parseInt(s);
    //Calculos realizados
    if (r > 0) {
        p = (int) (2 * Math.PI * r); //Calcula el perimetro.
        a = (int) (Math.PI * Math.pow(r, 2)); //Calcula el area.
        etiArea.setText("El Área es: " + a); //Envia el area.
        etiPerimetro.setText("El perimetro es: " + p); //Envia el perimetro
        etiArea.setForeground(Color.BLACK);
        etiPerimetro.setForeground(Color.BLACK);
    }
    //Si se ingresa numero negativos nos arroja lo siguiente.
    else {
        etiArea.setText("ERROR");
        etiPerimetro.setText("ERROR");
        etiArea.setForeground(Color.RED);
        etiPerimetro.setForeground(Color.RED);
        txtRadio.setText("");
    }
}
```

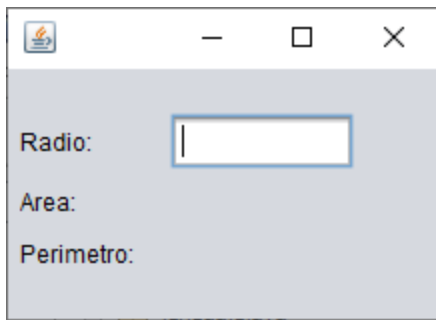
Además de un método para la validación de los datos.

```
//Validacion para solo numeros.  
public void verif(KeyEvent evt){  
    char validar = evt.getKeyChar();  
    if(Character.isLetter(validar)){  
        getToolkit().beep();  
        evt.consume();  
        JOptionPane.showMessageDialog(rootPane, "Ingrese solo numeros");  
    }  
}
```

Lo siguiente es agregar los métodos en los eventos siguiente.

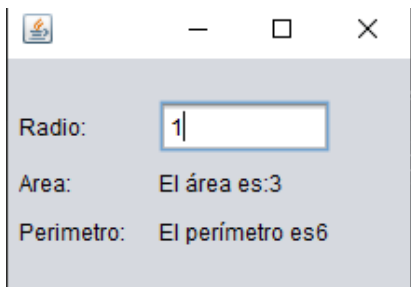
```
//Evento de KeyPressed que automaticamente calcula  
private void txtRadioKeyPressed(java.awt.event.KeyEvent evt) {  
    calcular();  
    // TODO add your handling code here:  
}  
  
//Evento para la validacion de los datos.  
private void txtRadioKeyTyped(java.awt.event.KeyEvent evt) {  
    verif(evt);  
    // TODO add your handling code here:  
}
```

Ahora ejecutaremos nuestro programa el cual quedara de la siguiente manera.

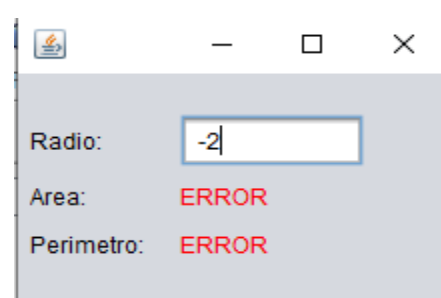


Como se puede observar no se cuenta con un botón porque cálculo se hará el con keyPressed.

Haciendo el calculo positivo.



Haciendo un calculo negativo.

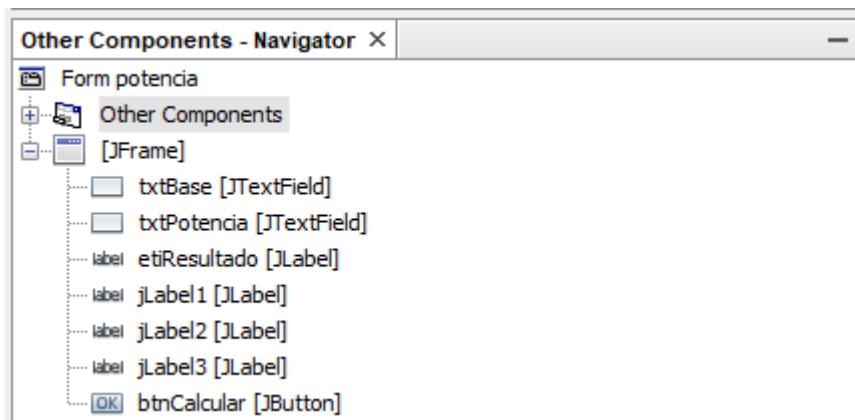


Como se puede observar se hizo correctamente lo que se pidió.

Ejercicio 4.

Se creará un programa que calcule la potencia de número, el cual usará de parámetros la base y la potencia, el diseño quedará de la siguiente manera.

Y el nombre de las variables de la siguiente manera.



Se crearán dos métodos, uno para calcular y otra para la validación, los cuales son los siguientes.

```
//Metodo para calcular la potencia de los numeros.
public void calcularpot(){
    String cad1,cad2;
    int p,b,r;
    cad1=txtBase.getText();
    cad2=txtPotencia.getText();
    b=Integer.parseInt(cad1);
    p=Integer.parseInt(cad2);
    r=(int) Math.pow(b, p); //Calcula la potencia con el metodo pow de la Clase Math
    etiResultado.setText(" "+r);
}
```

Y el método para la validación es el siguiente.

```
//Metodo para la validacion de los datos.
public void verif(KeyEvent evt){
    char validar = evt.getKeyChar();
    if(validar<'1' || validar>'9'){
        evt.consume();
        getToolkit().beep();
    }
    if(Character.isLetter(validar)){
        getToolkit().beep();
        evt.consume();
        JOptionPane.showMessageDialog(rootPane, "Ingrese solo numeros");
    }
}
```

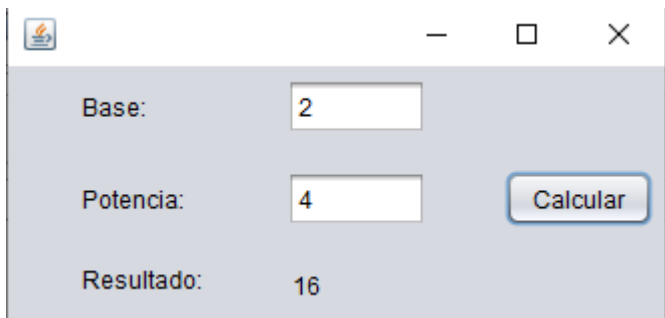
Lo siguiente es crear los eventos y agregarles los métodos creados que son los siguiente.

```
//Evento para calcular la potencia
private void btnCalcularActionPerformed(java.awt.event.ActionEvent evt) {
    calcularpot();
    // TODO add your handling code here:
}

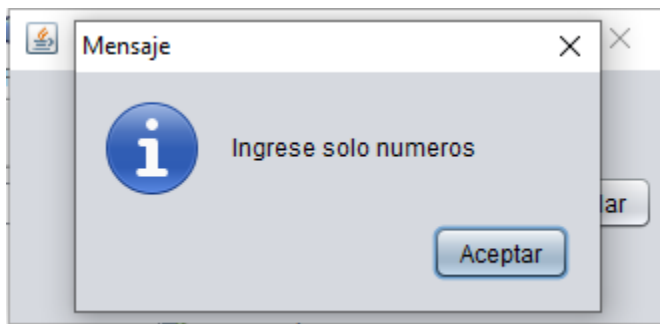
//Metodo para validar los enteros
private void txtBaseKeyTyped(java.awt.event.KeyEvent evt) {
    verif(evt);
    // TODO add your handling code here:
}

//Metodo para validar los enteros
private void txtPotenciaKeyTyped(java.awt.event.KeyEvent evt) {
    verif(evt);
    // TODO add your handling code here:
}
```

Ejecutaremos el programa y haremos unos cálculos.



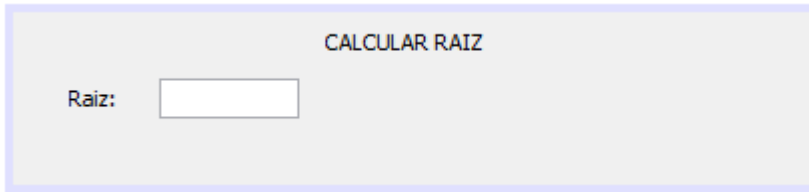
Como se puede observar los cálculos son correctos, si se llegara agregar un numero negativo o una letra la validación no dejará que se pueda ingresar, aparecerá lo siguiente.



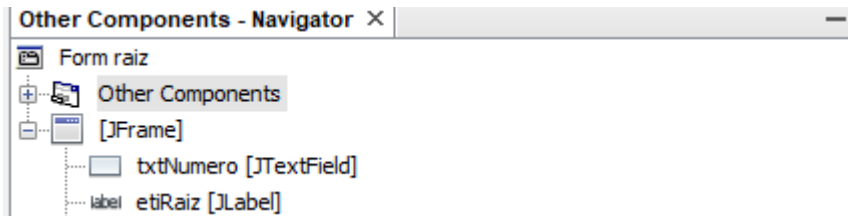
Por lo cual concluimos el ejercicio planteado.

Ejercicio 5.

El siguiente programa será calcular la raíz mediante un campo de texto, el cual se calculará en automático con el evento KeyPressed, el cual tendrá el siguiente diseño.



Y las siguientes variables.



Lo siguiente será calcular la raíz con método y además de crear otro para la validación de los datos.

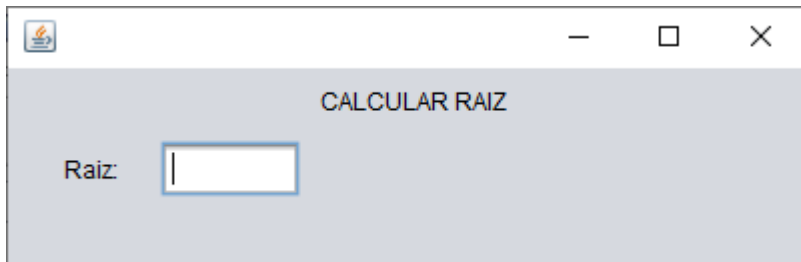
```
//Metodo para calcular la raiz con el metodo sqrt de la clase Math
public void calcular() {
    String cad1;
    double b,r;
    cad1=txtNumero.getText();
    b=Integer.parseInt(cad1);
    r=Math.sqrt(b); //Calcula l raiz
    etiRaiz.setText(""+r); //Imprime el resultado de la raiz
}
```

Método para la validación.

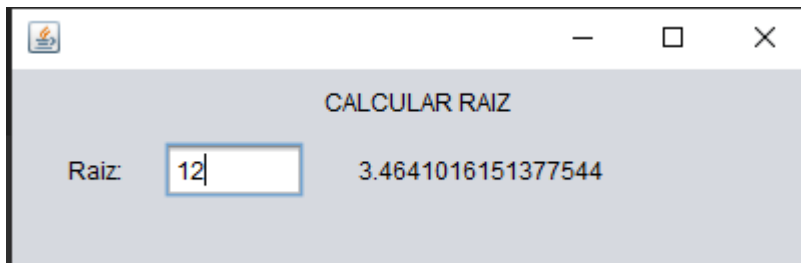
```
//Hace la validacion de los datos.
public void verif(KeyEvent evt){
    char validar = evt.getKeyChar();
    if(validar<'1' || validar>'9'){
        evt.consume();
        getToolkit().beep();
    }
    if(Character.isLetter(validar)){
        getToolkit().beep();
        evt.consume();

        JOptionPane.showMessageDialog(rootPane, "Ingrese solo numeros");
    }
}
```

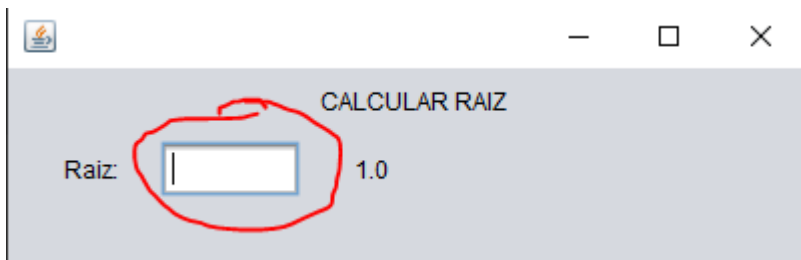
Al momento de ejecutar nuestro programa nos saldrá de la siguiente manera.



Ahora calcularemos una raíz, que sea positiva.



Si se ingresa un numero negativo, la validación no lo dejará, saldrá lo siguiente.

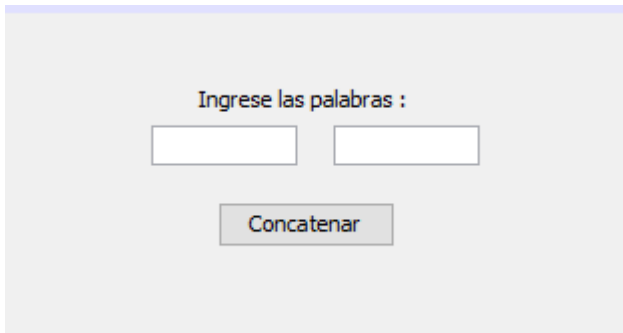


No dejará ingresar el signo.

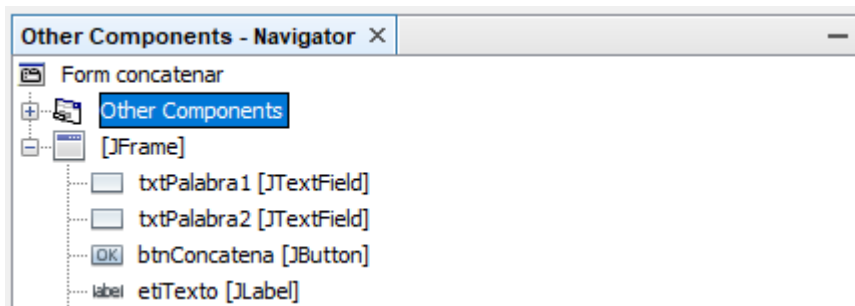
Por lo cual se da concluido el ejercicio.

Ejercicio 6.

El siguiente ejercicio es unir dos palabras que sean ingresadas en nuestros campos de texto, por lo cual el diseño es siguiente.



Y el nombre de las variables serán las siguientes.



En nuestro evento de ActionPerformed de nuestro botón concatenar será lo siguiente.

```
//Metodo para unir las dos palabras y mandarlo a la etiqueta de texto.  
private void btnConcatenaActionPerformed(java.awt.event.ActionEvent evt) {  
    etiTexto.setText(txtPalabra1.getText()+txtPalabra2.getText());  
    // TODO add your handling code here:  
}
```

Se hará un método de validación el cual solo nos dejara entrar letras que es el siguiente.

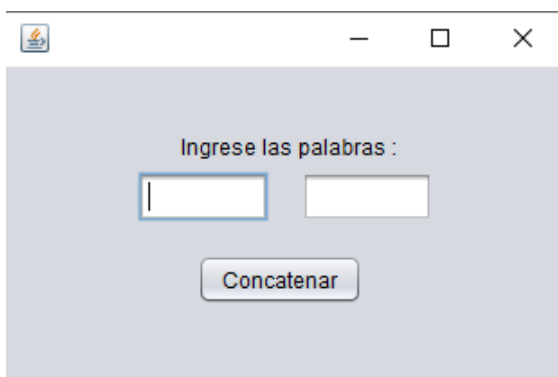
```
//Metodo para la validacion de los datos.  
public void verif(KeyEvent evt){  
    char validar = evt.getKeyChar();  
    if(Character.isDigit(validar)){  
        getToolkit().beep();  
        evt.consume();  
        JOptionPane.showMessageDialog(rootPane, "Ingrese solo letras");  
    }  
}
```


Como siguiente punto será agregar el método de validación en los eventos de keyPressed.

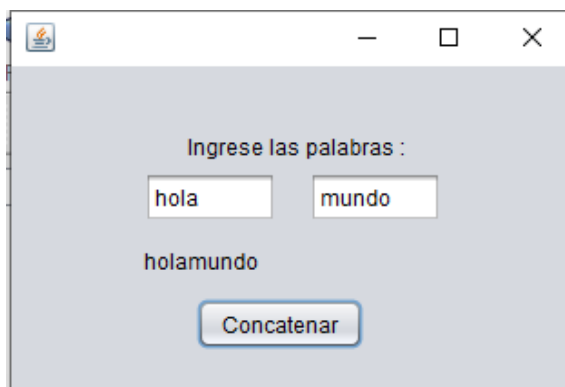
```
//Evento para la validacion de los datos
private void txtPalabra1KeyTyped(java.awt.event.KeyEvent evt) {
    verif(evt);
    // TODO add your handling code here:
}

private void txtPalabra2KeyTyped(java.awt.event.KeyEvent evt) {
    verif(evt);
    // TODO add your handling code here:
}
```

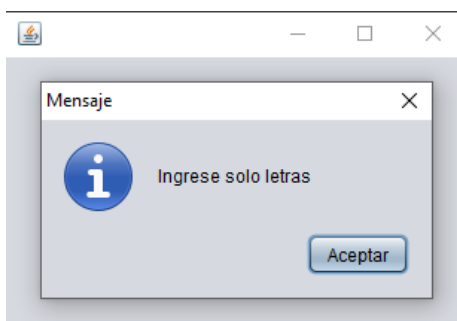
Al momento de ejecutar el programa nos saldrá de la siguiente manera.



Ahora ingresaremos dos palabras para unirlos.



Si llegamos a agregar un número algo diferente a una letra, nos saldrá lo siguiente.



Como se puede observar la validación impide agregar números.

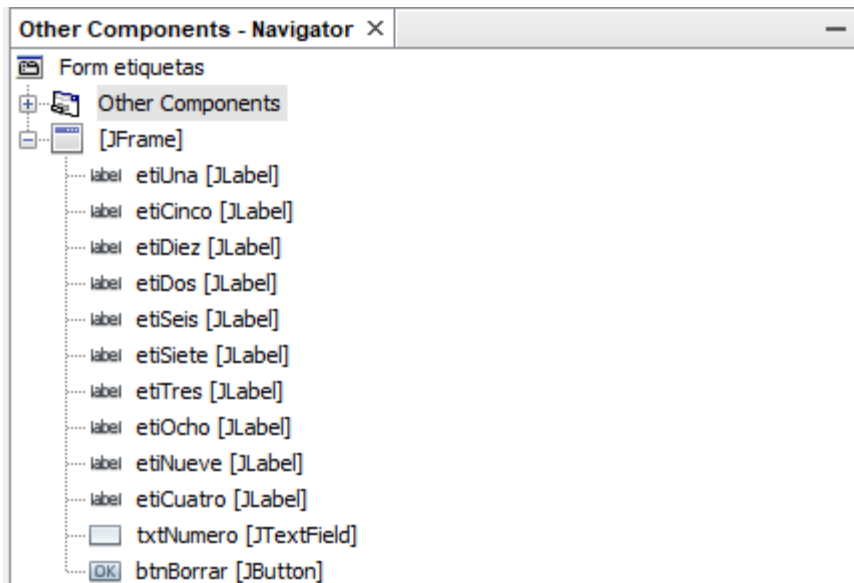
Por lo cual damos como acabada el ejercicio.

Ejercicio 7.

El siguiente ejercicio consiste en tener varias etiquetas y cuando el mouse pase por ellas se irán agregando al campo del texto, por lo cual se implemento el siguiente diseño, además de un botón de borrar.



Los nombres de las variables fueron los siguiente.



Se creara un botón borrar con el evento de ActionPerformed , cual será el siguiente.

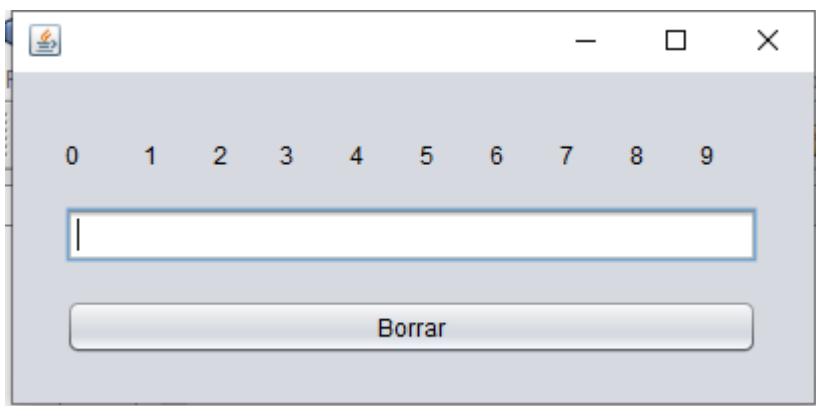
```
//Evento el cual borra el contenido del campo de texto.  
private void btnBorrarActionPerformed(java.awt.event.ActionEvent evt) {  
    txtNumero.setText("");  
    // TODO add your handling code here:  
}
```

Lo siguiente es crear cada uno de los eventos de MouseEntered, en cada etiqueta con número, lo cual es que se ira agregando a nuestro campo de texto.

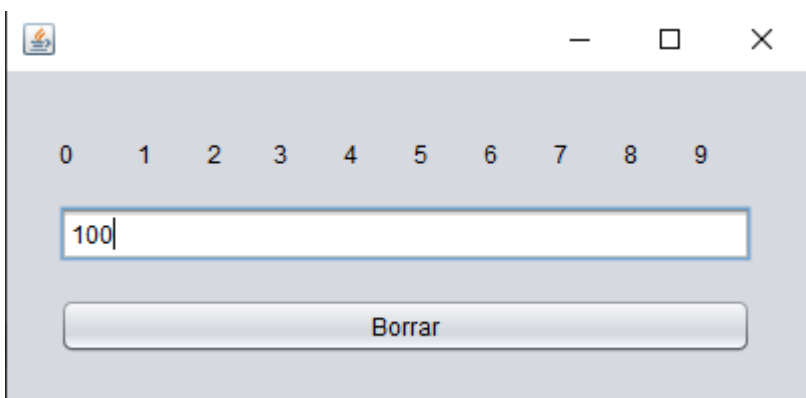
```
//Metodos para agregar los numeros a la etiqueta.  
private void etiUnaMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiUna.getText());  
    // TODO add your handling code here:  
}  
  
private void etiDosMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiDos.getText());  
    // TODO add your handling code here:  
}  
  
private void etiTresMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiTres.getText());  
    // TODO add your handling code here:  
}  
  
private void etiCuatroMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiCuatro.getText());  
}  
  
private void etiCincoMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiCinco.getText());  
}  
  
private void etiSeisMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiSeis.getText());  
}  
  
private void etiSeisMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiSeis.getText());  
}  
  
private void etiSieteMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiSiete.getText());  
}  
  
private void etiOchoMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiOcho.getText());  
}  
  
private void etiNueveMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiNueve.getText());  
}  
  
private void etiDiezMouseEntered(java.awt.event.MouseEvent evt) {  
    txtNumero.setText(txtNumero.getText()+etiDiez.getText());  
}
```

Lo que hace método es que el texto que este en la etiqueta, pasara al campo de texto, y se juntara con lo que tenga, para hacer poder hacer una cadena nueva.

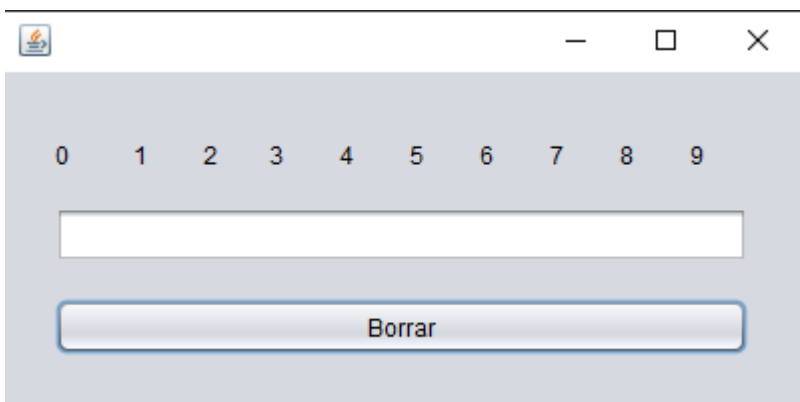
Nuestro programa queda de la siguiente manera.



Lo siguiente es ir agregando numero mientras el mouse se va colocando en las etiquetas, como se ve a continuación.



Como son capturas de pantalla no se puede apreciar, una vez que se ejecute hará lo que se pida.



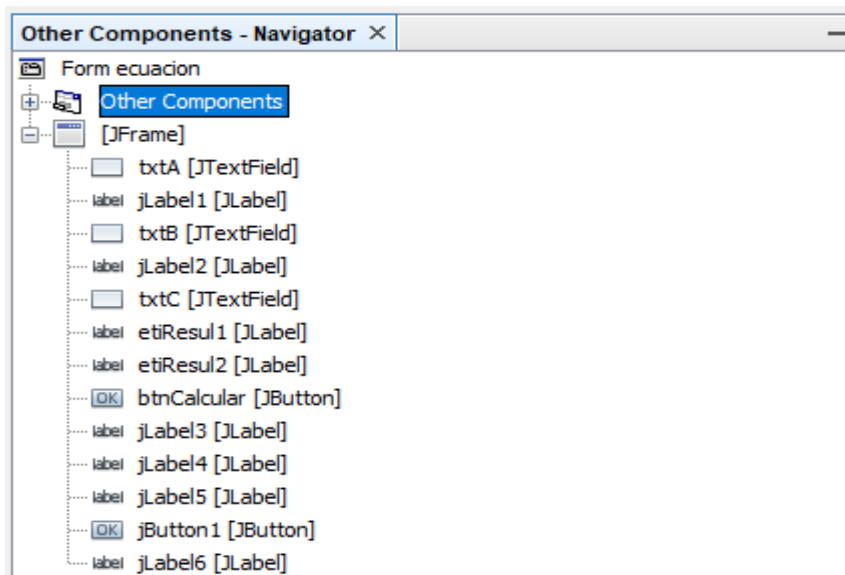
Cuando presionamos el botón borrar se elimina todo el contenido del campo, por lo cual se da concluido la actividad.

Ejercicio 8.

El siguiente programa es calcular las soluciones de una ecuación de 2° grado, el diseño de nuestra interfaz será la siguiente.

The screenshot shows a Java Swing window titled "Form ecuacion". At the top, the equation $Ax^{(2)}+Bx+C=0$ is displayed. Below the equation, there are five input fields arranged in two columns. The first column contains labels "Valor de A:", "Valor de B:", "Valor de C:", "Primera Solución", and "Segunda Solución". The second column contains five empty text input boxes. At the bottom of the window, there are two buttons: "Calcular" and "Borrar".

El nombre de las variables el siguiente.



Se crearan dos métodos, los cuales serán para el cálculo y para la validación de los datos.

El método para la validación es el siguiente.

```
//Metodo de validacion
public void verif(KeyEvent evt) {
    char validar = evt.getKeyChar();
    if(Character.isLetter(validar)) {
        getToolkit().beep();
        evt.consume();
        JOptionPane.showMessageDialog(rootPane, "Ingrese solo numeros");
    }
}
```

Y el método para los cálculos será el siguiente.

```
//Metodo para hacer los calculos
public void calcular() {
    String cad1,cad2,cad3;
    int a,b,c,R;
    double x1,x2;
    cad1=txtA.getText();
    cad2=txtB.getText();
    cad3=txtC.getText();
    a=Integer.parseInt(cad1);
    b=Integer.parseInt(cad2);
    c=Integer.parseInt(cad3);
    etiResul1.setForeground(Color.BLACK); //Se pone el color de letra negra como por defecto
    etiResul2.setForeground(Color.BLACK); //
    R=(int) (Math.pow(b, 2)-(4*a*c)); //Se realiza la primera operacion para saber cuantos resultados tiene
    //Si es menor a 0 nos mandara lo siguiente
    if (R<0) {
        etiResul1.setText("NO HAY SOLUCION:");
        etiResul2.setText("NO HAY SOLUCION:");
        etiResul1.setForeground(Color.RED);
        etiResul2.setForeground(Color.RED);
    }
    //Si es igual a 0 nos mandara una solucion.
    if (R==0) {
        x1=-b/(2*a);
        etiResul1.setText(""+x1);
    }

    //Si es mayor a 0 nos mandara las dos soluciones.
    if (R>0) {
        x1=((-b)+Math.sqrt(R))/(2*a);
        x2=((-b)-Math.sqrt(R))/(2*a);
        etiResul1.setText(""+x1);
        etiResul2.setText(""+x2);
    }
}
```

Lo siguiente será crear los eventos para los botones y los campos de texto.

El botón borrar tendrá los siguiente.

```
//Evento para boorrar todo
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    etiResul1.setText("");
    etiResul2.setText("");
    txtA.setText("");
    txtB.setText("");
    txtC.setText("");
    // TODO add your handling code here:
}

//Evento para la validacion de los campos
```

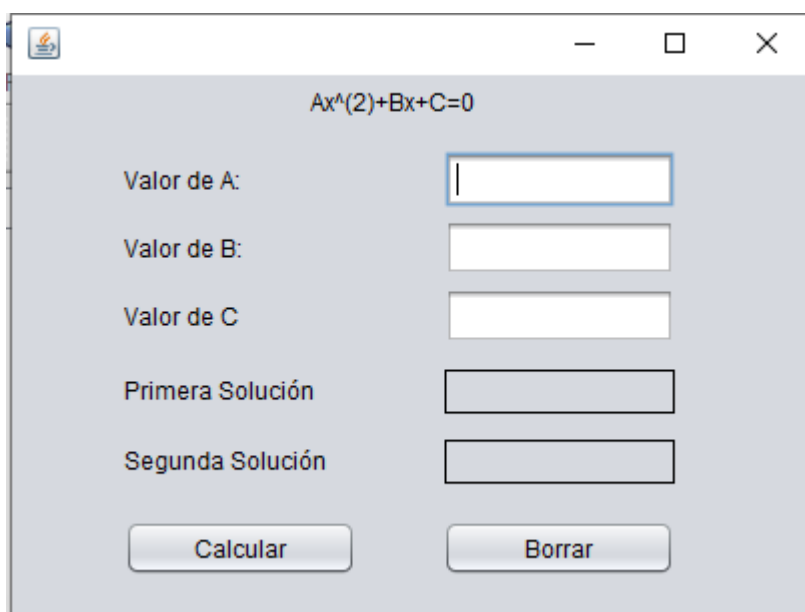
Nuestro botón calcular el siguiente.

```
//Evento para hacer el calculo
private void btnCalcularActionPerformed(java.awt.event.ActionEvent evt) {
    calcular();
}
```

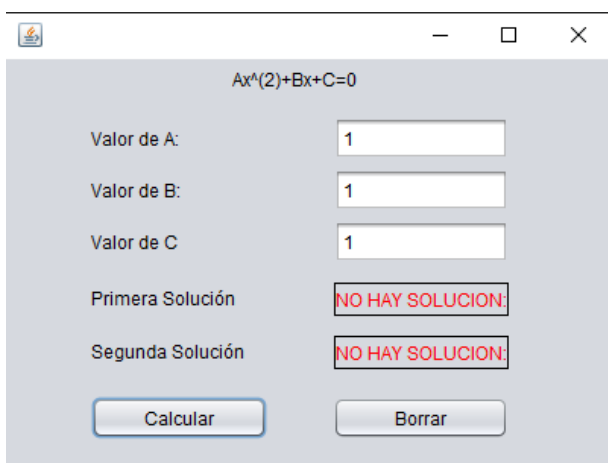
Y para la validación de los datos el siguiente.

```
//Evento para la validacion de los campos.  
private void txtAKeyTyped(java.awt.event.KeyEvent evt) {  
    verif(evt);  
}  
  
private void txtBKeyTyped(java.awt.event.KeyEvent evt) {  
    verif(evt);  
}  
  
private void txtCKeyTyped(java.awt.event.KeyEvent evt) {  
    verif(evt);  
}
```

Al momento de ejecutar nuestro programa será el siguiente.



Ingresaremos datos para poder tener las tres opciones de condición.



Como se puede apreciar esta ecuación no tiene solución.

Ax²+Bx+C=0

Valor de A: 1

Valor de B: 0

Valor de C: -18

Primera Solución: 4.2426406871192...

Segunda Solución: -4.242640687119...

Calcular Borrar

Mientras que este tiene dos soluciones.

Ax²+Bx+C=0

Valor de A: 1

Valor de B: -4

Valor de C: 4

Primera Solución: 2.0

Segunda Solución:

Calcular Borrar

A comparación de esta que solo tiene una solución.

Ax²+Bx+C=0

Valor de A:

Valor de B:

Valor de C:

Primera Solución:

Segunda Solución:

Calcular Borrar

El botón nos hace lo siguiente.

Nos limpia todos los datos.

Ax²+Bx+C=0

Mensaje

Ingrese solo numeros

Aceptar

Si deseamos ingresar algo que no sea un numero nos marcara lo siguiente.

Por lo cual podemos concluir con este ejercicio.