

UNIVERSIDAD DE CORDOBA

FACULTAD DE INGENIERIAS

PROGRAMA INGENIERIA DE SISTEMAS

CURSO: Programación I

TEMA: Ejemplo relaciones entre clases y aplicaciones de ventana.

DESCRIPCION: En el presente documento se aborda el tema de las relaciones entre clases dentro de un diagrama de clases UML, considerando para ello un problema planteado y la respectiva aplicación de ventana (interfaz gráfica de usuario) desarrollada en el IDE NetBeans mediante controles de la paleta Swing, e implementada en el lenguaje de programación Java, siguiendo un modelo de orientación a objetos. De esta manera, se presenta un enunciado al que se le hace el diagrama lógico de la aplicación con clases, relaciones y el estereotipo de la multiplicidad para algunas de ellas, siguiendo la simbología y semántica definida en UML para tales efectos. Igualmente se presentan las clases y relaciones relativas a los componentes gráficos que constituyen la interfaz gráfica de usuario, conjuntamente con las respectivas relaciones de ventana y las clases de la lógica de aplicación. Así mismo, se presenta paso a paso la construcción del proyecto de ventana en *NetBeans*, la adición de la ventana y sus componentes gráficos con su correspondiente configuración, como también la implementación de la clase del diseño lógico y la codificación de eventos en la ventana.

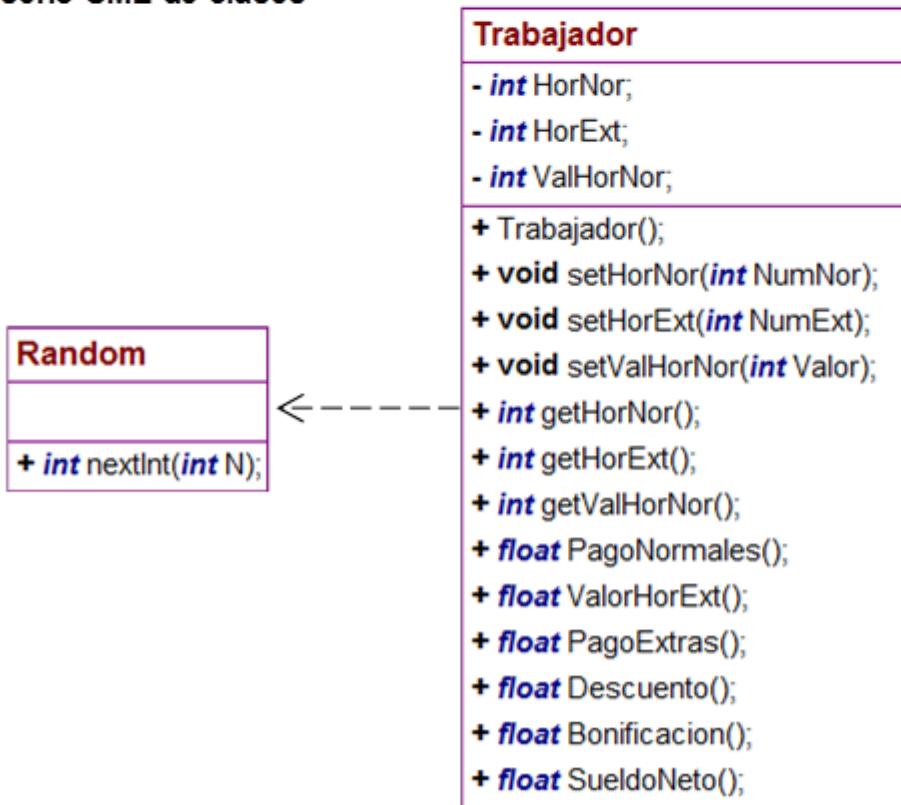
OBJETIVO: Diseñar en UML e implementar en Java una aplicación de ventana hecha en el IDE a *NetBeans*, que ilustre el uso de relaciones entre clases, uso componentes de la paleta swing, implementación de clases y programación de eventos, a partir del planteamiento textual de un problema.

PALABRAS CLAVES: Relaciones entre clases UML, relaciones de composición, asociación y dependencia, multiplicidad de una relación, aplicaciones de ventana en *NetBeans*, controles de la paleta swing e implementación de clases y eventos en Java.

1. Presentación del ejemplo

Para este ejemplo, consideraremos el diseño en UML de una clase que permita calcular el salario neto mensual a pagarle a un trabajador, tomando en cuenta que éste labora al mes un número de horas normales, las cuales se pagan a un precio determinado. El empleado también puede tener acumulado un número de horas extras, las que se le paga con un incremento del 55% con respecto al valor de la hora normal. Así mismo, al trabajador se le hace un descuento por afiliación al sindicato, el cual se efectúa solo sobre el pago recibido por las horas normales, de manera que si dicho pago esta entre 1.000.000 y 1.500.000 el descuento es del 1%, si es mayor a 1.500.000 y no supera los 2.000.000, el descuento será del 1.5%; para el caso que sea mayor que 2.000.000 y no superior a 2.500.000 se hace un descuento del 2%, pero si es superior a 2.500.000 se aplica un descuento del 2.3%, y si gana menos de 1.500.000 se descuenta el 0.6%. Adicionalmente al empleado se le da una bonificación, siempre que el número de horas extras laboradas superen los dos tercios de las horas normales trabajadas; esta bonificación consiste en un valor comprendido entre 250.000 y 500.000 que el sistema debe generar al azar.

2. Diseño UML de clases



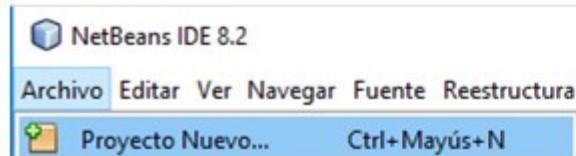
En el diseño UML anterior, para el caso de los atributos como tienen nombres abreviados, tome en cuenta que **HorNor** es el número de horas normales trabajadas, **HorExt** es el número de horas extras acumuladas y **ValHorNor** es el valor o precio de la hora normal. En cuanto a los métodos con nombres abreviados, tenemos que **ValorHorExt** calcula el valor de la hora extra individual, **PagoExtras** obtiene el valor a pagar por todas las horas extras laboradas por el empleado y **PagoNormales** por su parte, es el valor a pagar por todas las horas normales.

3. Creación del proyecto en NetBeans

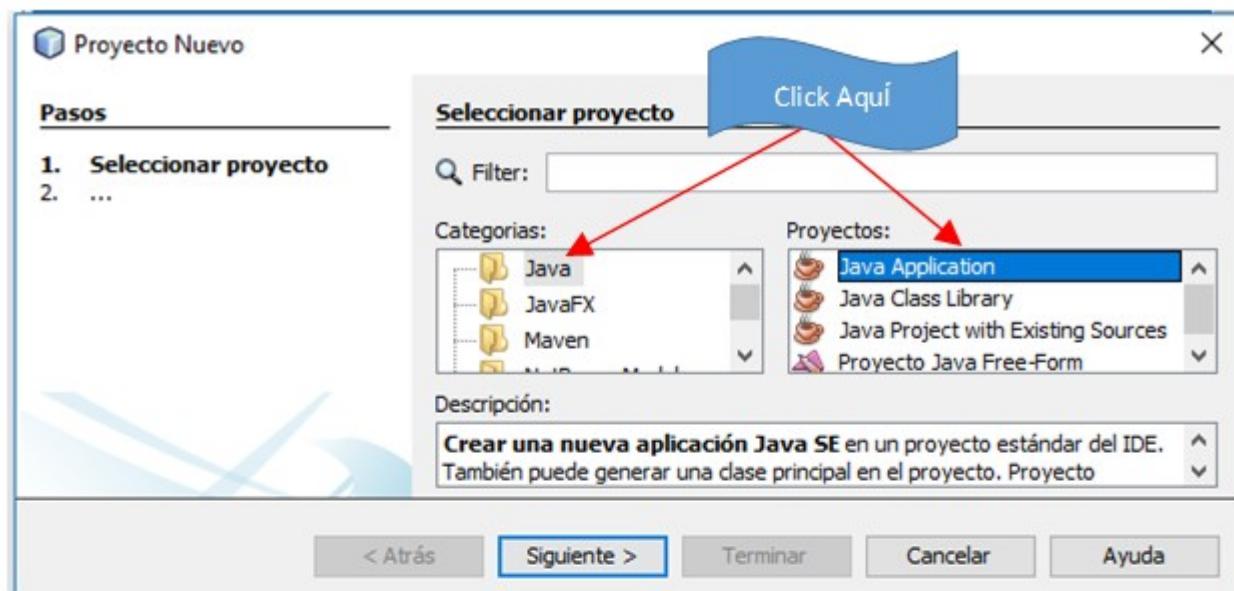
- Inicie o ejecute el programa *NetBeans* y observará una pantalla como la siguiente:



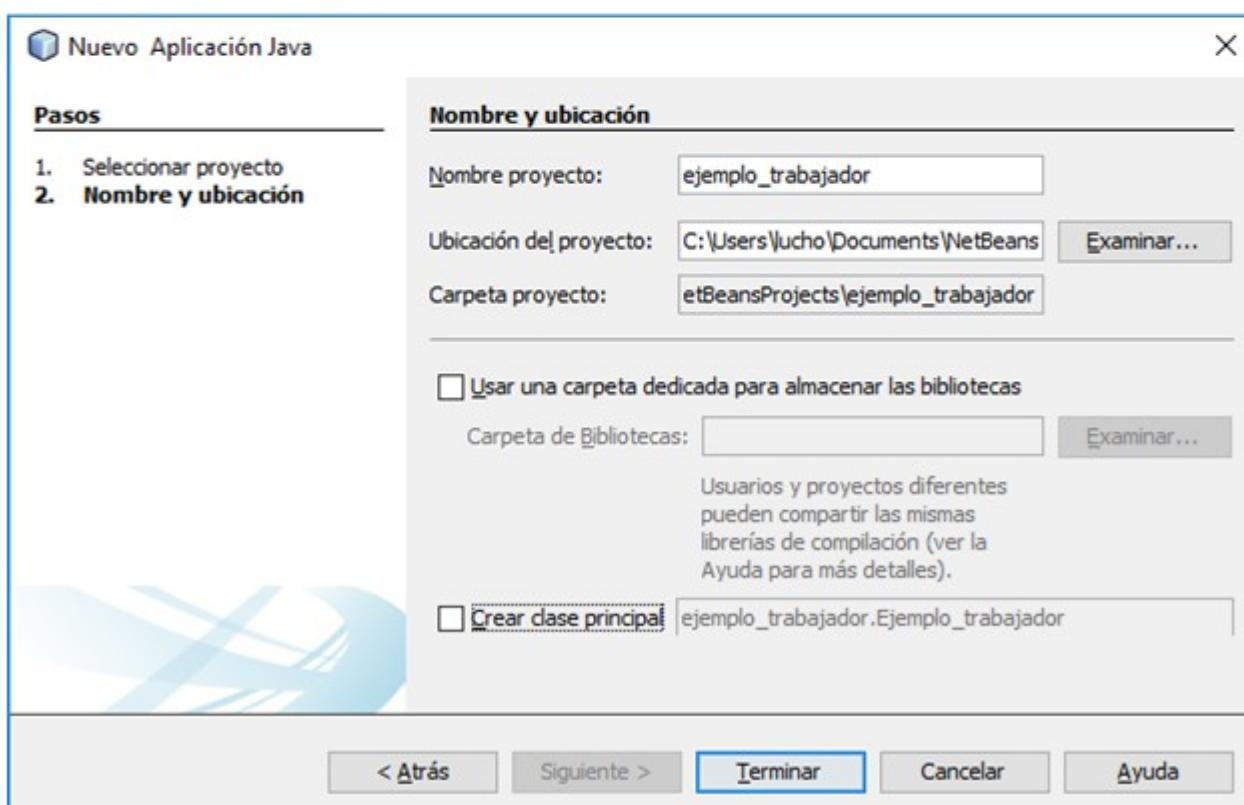
- Cree un nuevo proyecto haciendo click en el menú **Archivo** y luego en la opción **Proyecto nuevo**, tal como se muestra en la siguiente imagen:



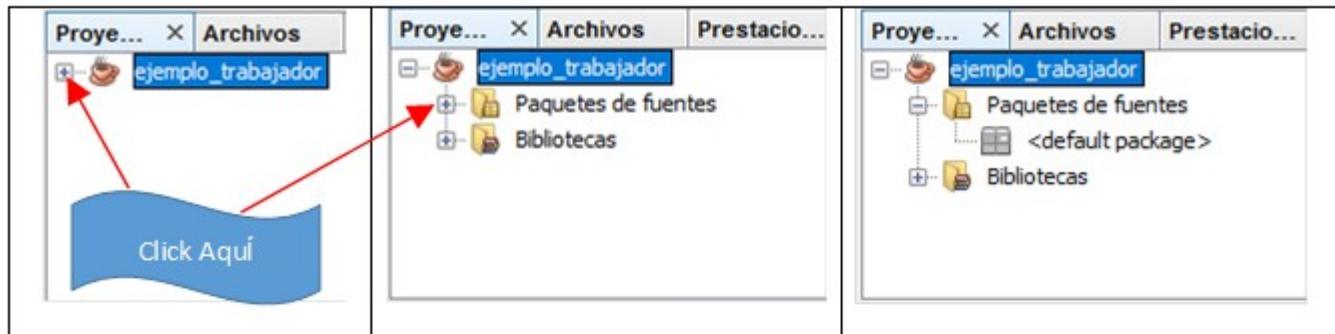
- En la ventana desplegada, seleccione el nodo **java** del panel de **categorías** y el nodo **Java Application** del panel **proyectos**; a continuación haga click en el botón **siguiente**:



- En la siguiente ventana, ingrese el nombre del proyecto (**ejemplo_trabajador**), desmarque la casilla de verificación titulada como "**Crear clase principal**" (**Crear clase principal**) y haga click en el botón **Terminar**.



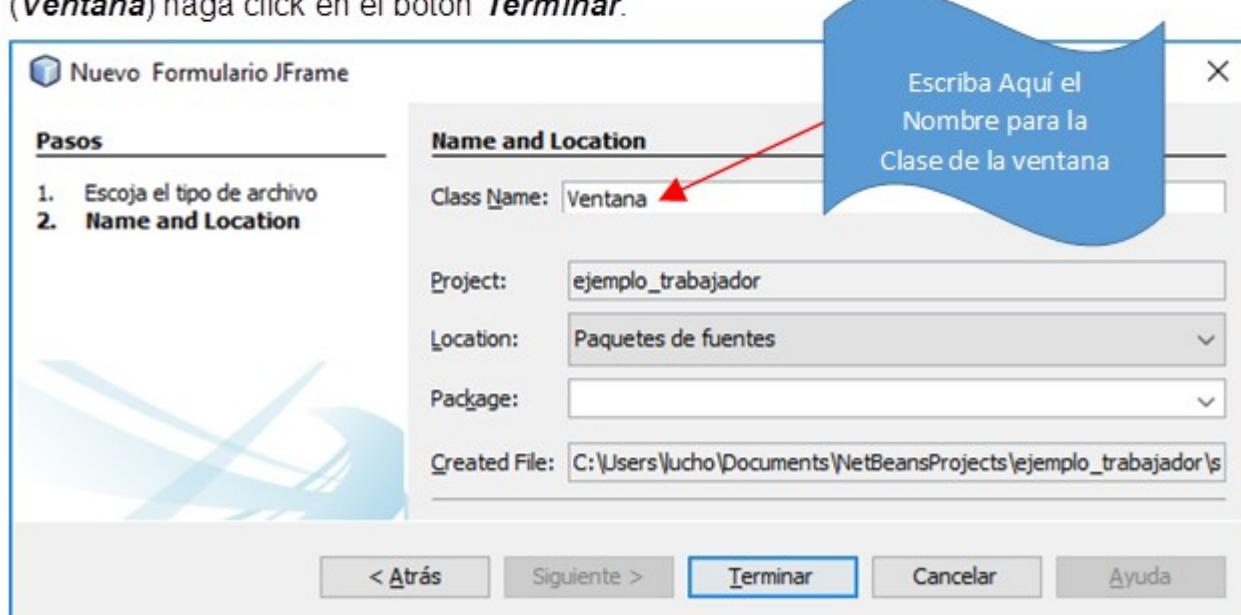
- ➁ Una vez creado el proyecto en la parte superior ventana. Si el proyecto aparece contraído, vera a su derecha un signo +, haga click en este signo para expandirlo; luego haga click en el signo + de Paquetes fuentes para expandirlo.



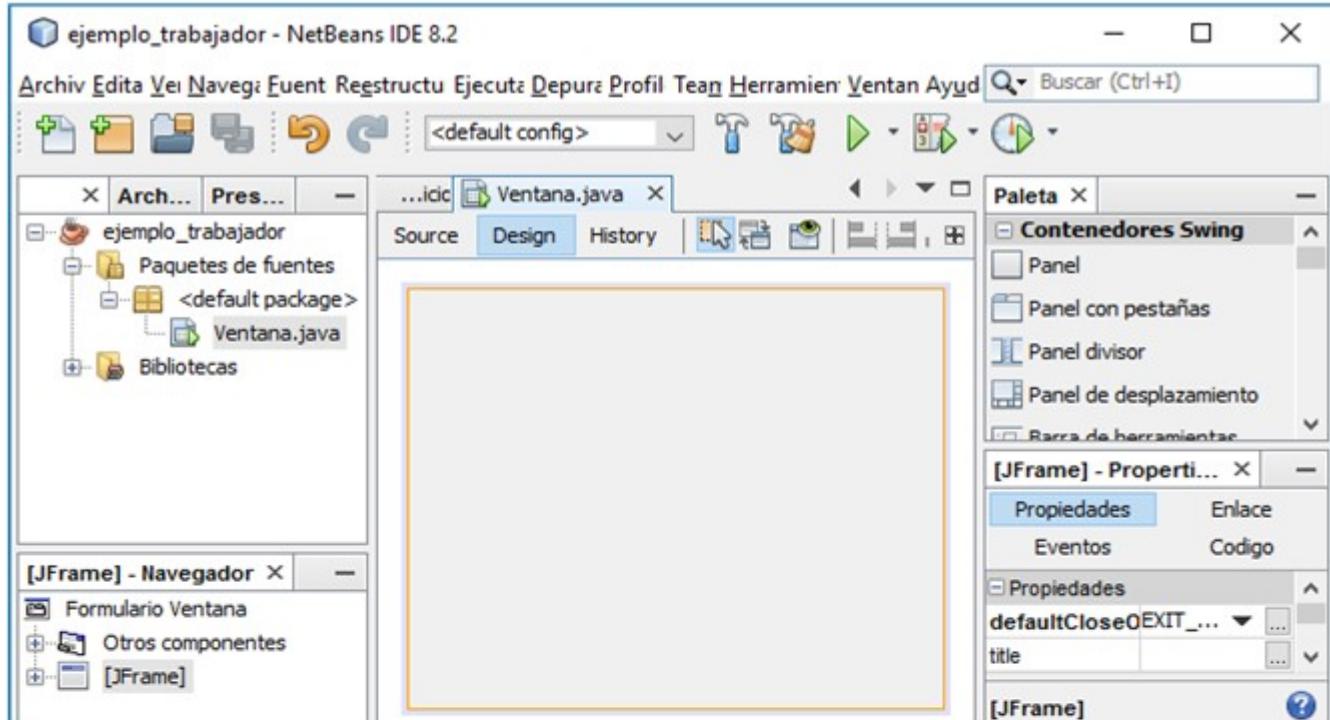
- ➂ Ahora vamos a crear la ventana para la interfaz gráfica de usuario; para lo cual haga click derecho en nodo **Paquetes fuentes** del proyecto y escoja la opción **Nuevo** (o **New** si el IDE esta en inglés) y luego **Formulario JFrame**, tal como se muestra en la siguiente imagen:



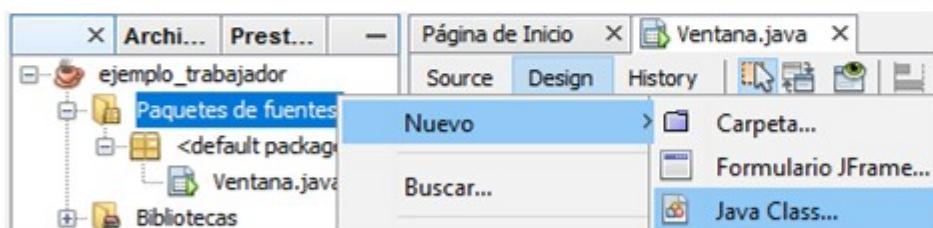
- ➃ En la ventana desplegada, en la entrada **Class Name**, ingrese el nombre para la clase (**Ventana**) haga click en el botón **Terminar**.



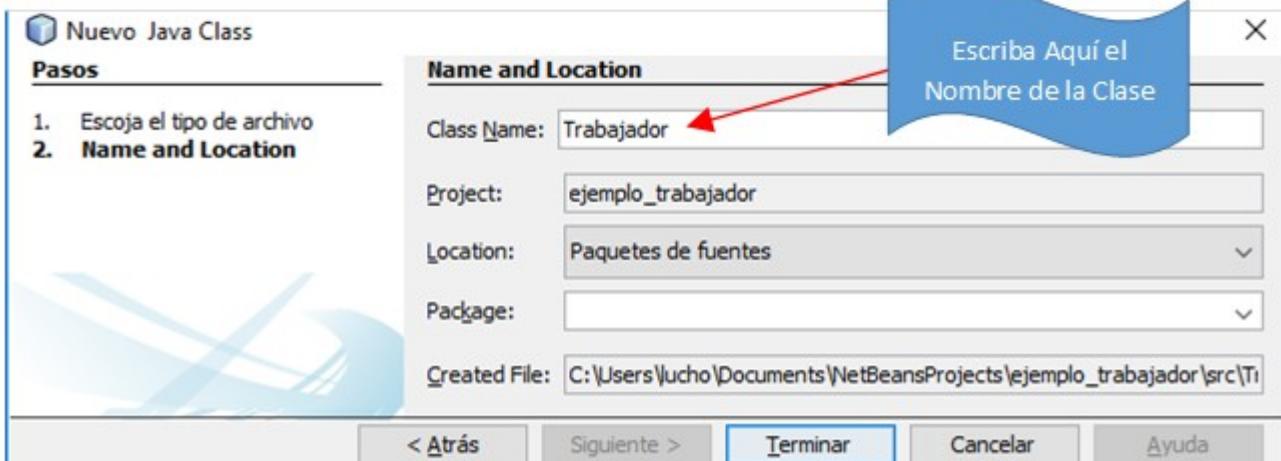
Ahora el IDE tendrá apariencia similar a la siguiente:



- Para crear la clase **Trabajador**, haga click derecho en nodo **Paquetes fuentes** del proyecto y escoja la opción **Nuevo** y luego **Java Class**, tal como se muestra en la siguiente imagen:



- En la entrada **Class Name** de la ventana presentada, ingrese el nombre para la clase (**Trabajador**) y después haga click en el botón **Terminar**.



● Implemente la clase **Trabajador** de la siguiente manera:

The screenshot shows a Java code editor window with the title "Trabajador.java". The menu bar includes "Source" and "History". Below the menu is a toolbar with various icons. The code itself is a Java class named "Trabajador" with methods for setting and getting hours and values.

```
11  import java.util.Random;
12
13  public class Trabajador {
14
15      private int HorNor;
16      private int HorExt;
17      private int ValHorNor;
18
19      public Trabajador() {
20          HorNor=0;
21          HorExt=0;
22          ValHorNor=0;
23      }
24
25      public void setHorNor(int NumNor) {
26          HorNor=NumNor;
27      }
28
29      public void setHorExt(int NumExt) {
30          HorExt=NumExt;
31      }
32
33      public void setValHorNor(int Valor) {
34          ValHorNor=Valor;
35      }
36
37      public int getHorNor() {
38          return HorNor;
39      }
40
41      public int getHorExt() {
42          return HorExt;
43      }
44
45      public int getValHorNor() {
46          return ValHorNor;
47      }
48  }
```

```
49  [ ] public float PagoNormales(){
50    [ ]     return HorNor*ValHorNor;
51  }
52
53  [ ] public float ValorHorExt(){
54    [ ]     return ValHorNor*1.55f;
55  }
56
57  [ ] public float PagoExtras(){
58    [ ]     return HorExt*ValorHorExt();
59  }
60
61  [ ] public float Descuento(){
62    float Porcen;
63    float PagNor;
64    PagNor=PagoNormales();
65    if(PagNor>=1000000 && PagNor<=1500000){
66        Porcen=0.01f;
67    }
68    else
69        if(PagNor>1500000 && PagNor<=2000000){
70            Porcen=0.015f;
71        }
72    else
73        if(PagNor>2000000 && PagNor<=2500000){
74            Porcen=0.02f;
75        }
76    else
77        if(PagNor>2500000){
78            Porcen=0.023f;
79        }
80    else{
81        Porcen=0;
82    }
83    return PagNor*Porcen;
84}
85
86  [ ] public float Bonificacion(){
87    Random R;
88    if(HorExt>2*HorNor/3){
89        R=new Random();
```

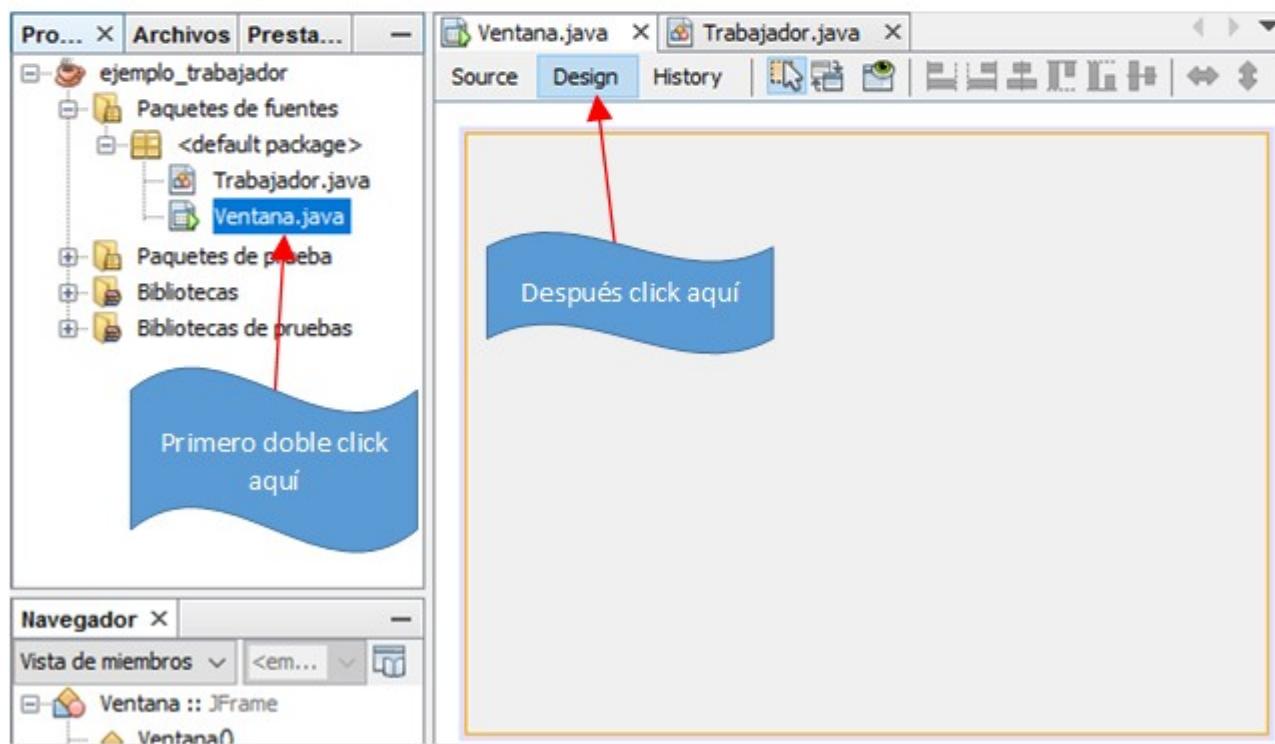
```

90     return (R.nextInt(251) + 250)*1000;
91 }
92 else{
93     return 0;
94 }
95 }

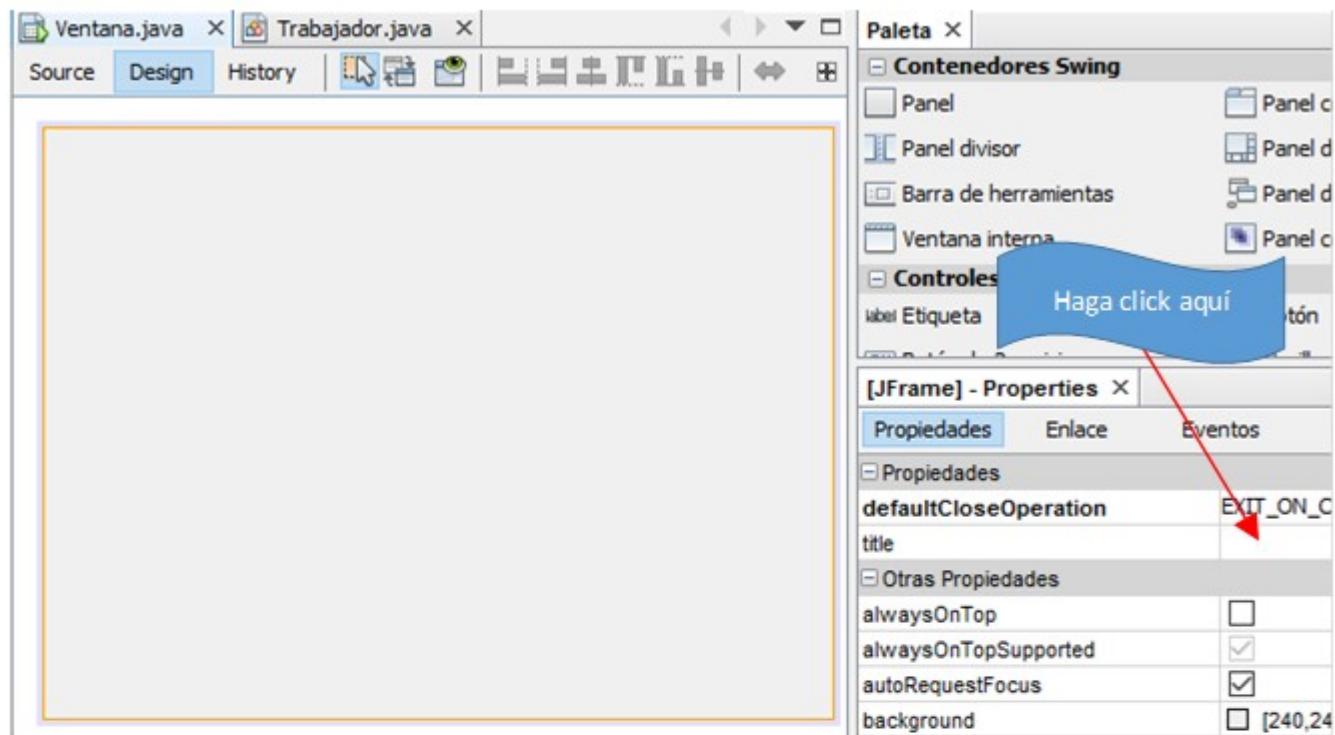
96
97 public float SueldoNeto(){
98     return PagoNormales()+PagoExtras()-Descuento()+Bonificacion();
99 }
100
101 }
```

4. Diseño gráfico de la ventana

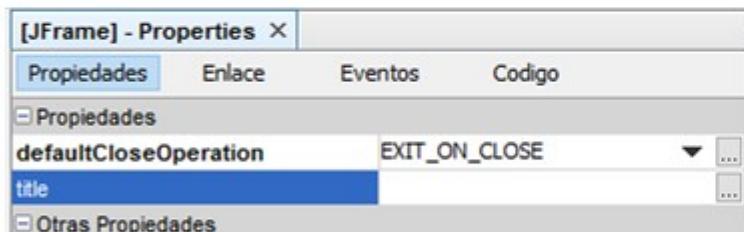
- Si la ventana (**JFrame**) no está visible, actívelo en modo diseño: para ello primero haga doble click en el archivo **Ventana.java** y después click en la vista **diseño (Design)**. Así la vista de diseño de la ventana, es el rectángulo gris de borde naranja que vemos en la siguiente imagen:



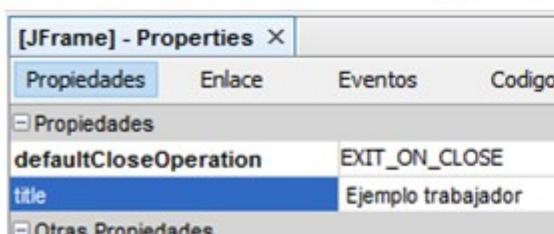
- Haga click dentro de la ventana y luego vaya al inspector de propiedades; ubique la propiedad **title** y haga click en el recuadro de su derecha:



- Hecho esto, se activa el modo de edición de la propiedad como vemos abajo:



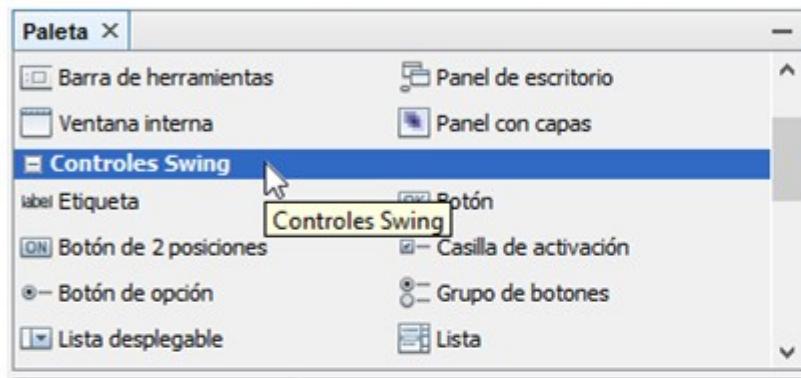
- Escriba el título de la ventana y pulse enter; para el caso: **Ejemplo trabajador**



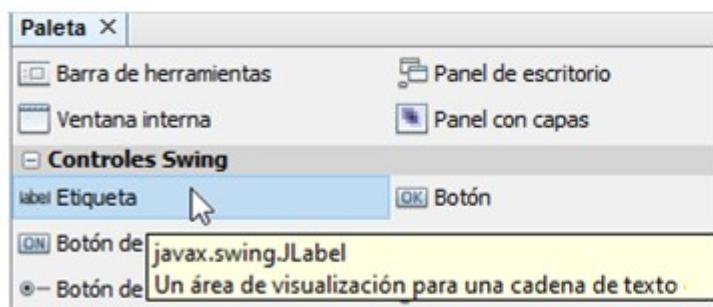
- Cuando modificamos el diseño de la ventana o el código fuente de algún archivo, debemos guardar los cambios; para ello podemos hacer click en el botón de los dos discos (mostrado más abajo), o también pulsamos la combinación de teclas **Shift + Ctrl + S**. Tome en cuenta que este botón está habilitado siempre que exista un cambio pendiente de ser guardado; en caso contrario lo verá deshabilitado.

<i>Botón habilitado para guardar.</i>	<i>Botón deshabilitado después de guardar.</i>

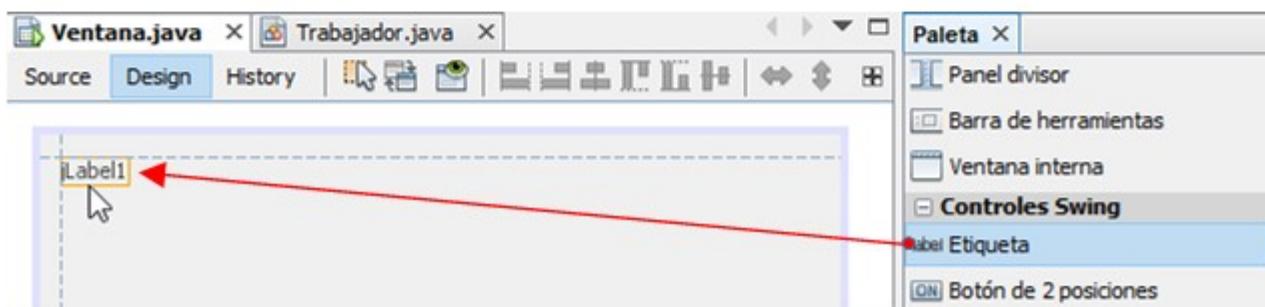
- Para arrastrar controles a la ventana usamos la paleta de controles de **Swing**, para lo cual asegúrese que el nodo **Controles Swing** este expandido (con el signo meno a su izquierda), para haga click en el signo + si está presente:



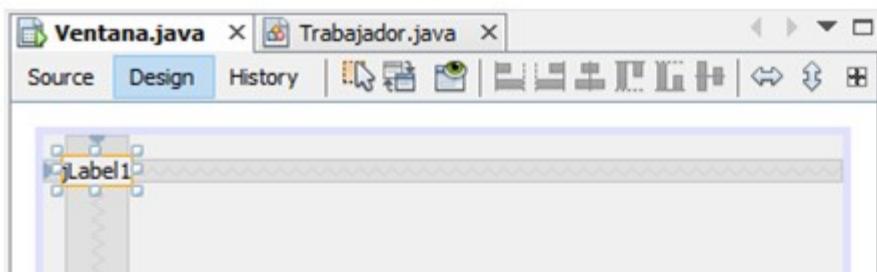
- Para poner texto en la ventana usamos el control **JLabel** (*Etiqueta*), por lo tanto haga click en el para seleccionarlo.



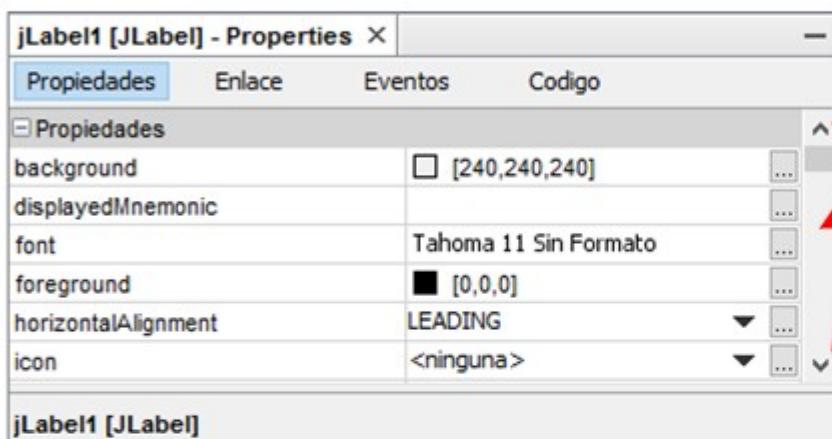
- Arrastre la etiqueta sobre la ventana hasta el punto indicado:



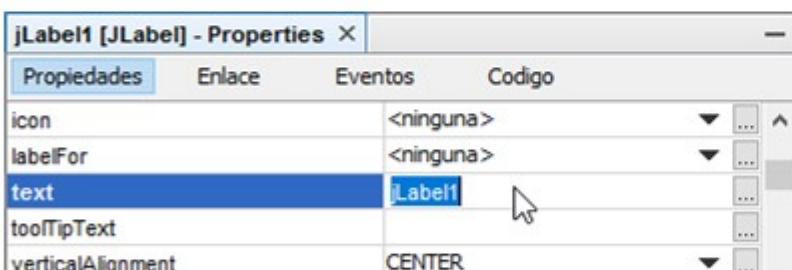
- Con ello la ventana toma la siguiente apariencia:



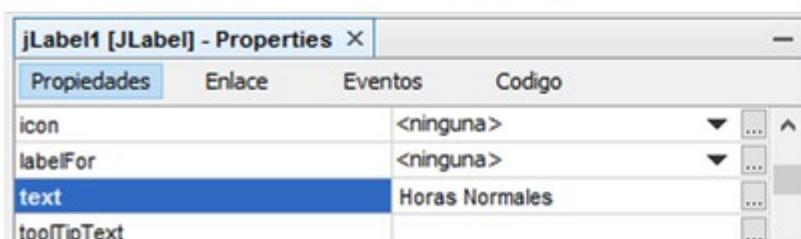
- Para localizar alguna propiedad del control seleccionado o activo en la ventana, debemos desplazarnos hacia arriba o hacia abajo con las barras laterales del inspector de propiedades:



- De esta manera, localice la propiedad **text** y haga click a su derecha (para el caso en el texto jLabel1, activando así la edición de la propiedad).



- Borre este texto y escriba: **Horas Normales**



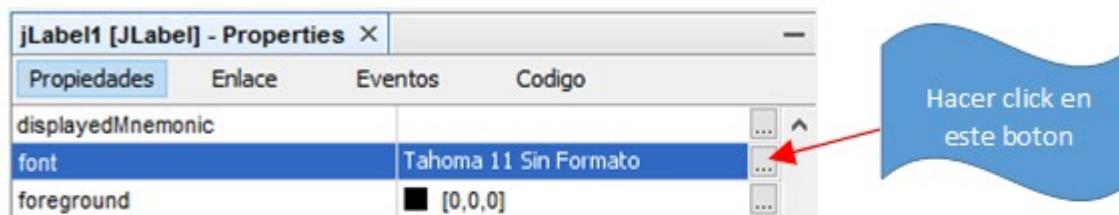
- Al pulsar la tecla enter, la propiedad se resalta en azul así:



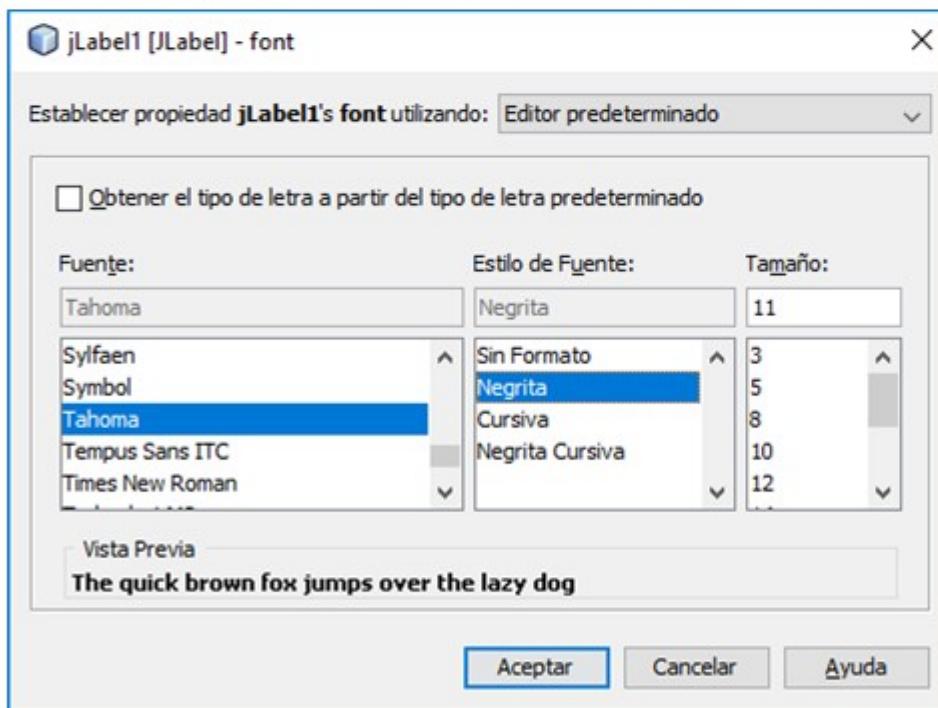
- Por su parte el aspecto de la ventana será como sigue:



- Para poner en negrita un texto, hacemos click en el botón de tres puntos, que se encuentra a la derecha de la propiedad **font**.



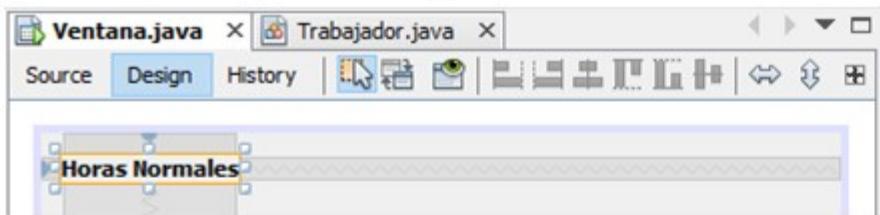
- Con ello se despliega el siguiente cuadro de dialogo, en el que puede escoger el tipo de letra, tamaño y estilos. Una vez hecho esto haga click en el botón **Aceptar**.



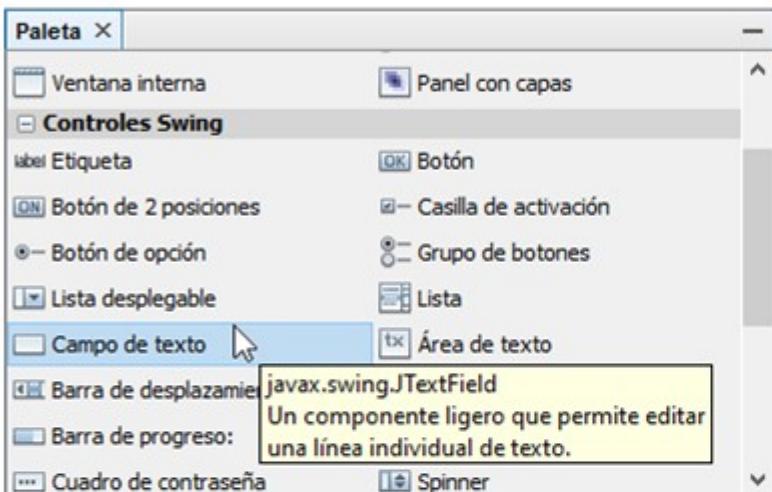
- Con ello observara el cambio en la propiedad, que ya muestra el estilo *Negrita*:



- Así la etiqueta se verá como sigue:



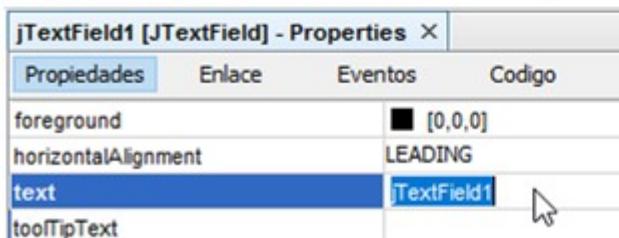
- Para entrada o captura de texto, usamos el control **JTextField** (Campo de texto) haga click en este para seleccionarlo:



- Arrastre el campo de texto hasta la ventana en la posición indicada:



- Estando seleccionado el campo de texto, busque la propiedad **text** en el inspector de propiedades; haga click a su derecha para activar la edición del texto



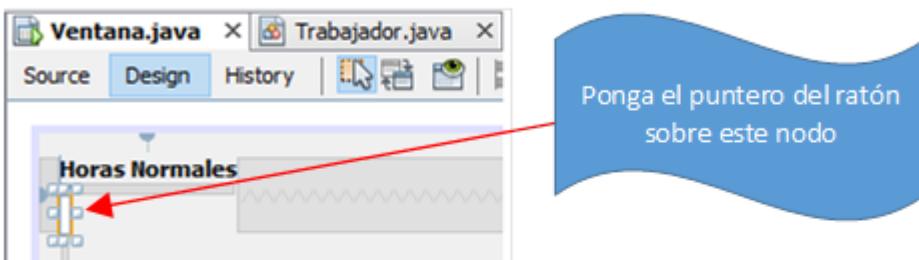
- Seguidamente borre el texto y pulse la tecla enter:

horizontalAlignment	LEADING
text	TextField1
toolTipText	

- Tras esto, la propiedad se vera de esta manera:

horizontalAlignment	LEADING
text	
toolTipText	

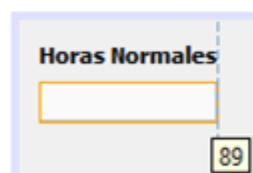
- Además esto causa que el campo de texto se "encoja", tal como se muestra abajo; por lo tanto, debemos redimensionarlo y para ello acerque el puntero del ratón a uno de los nodos (uno de los ocho cuadrillos que lo rodean) del control.



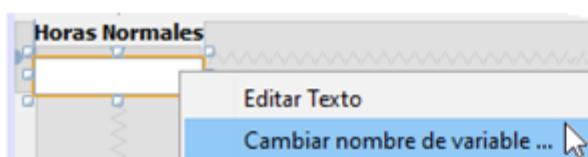
- Acerque el puntero del ratón hasta el nodo indicado, hasta que se convierta en una flecha horizontal doble:



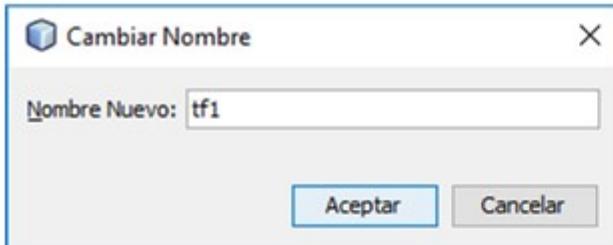
- Mantenga click izquierdo pulsado y mueva el ratón hacia la derecha para cambiar el ancho del control; digamos hasta el punto indicado con un valor de 89 (pixeles) o el valor que usted prefiera.



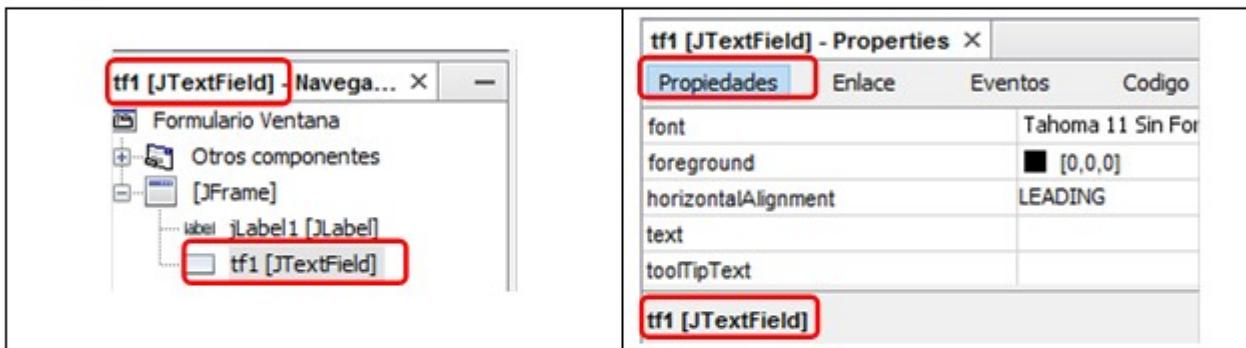
- La propiedad más importante de un control es el nombre de variable (nombre de la instancia), pues la usamos para identificar, diferenciar y referenciar a cada control en código de java relativo a la programación de la ventana; es decir, de sus eventos y de su funcionalidad. Para asignar un nombre a un control haga click derecho sobre él; para el caso sobre el campo de texto. Seguidamente haga click izquierdo sobre la opción "Cambiar nombre de variable ..."



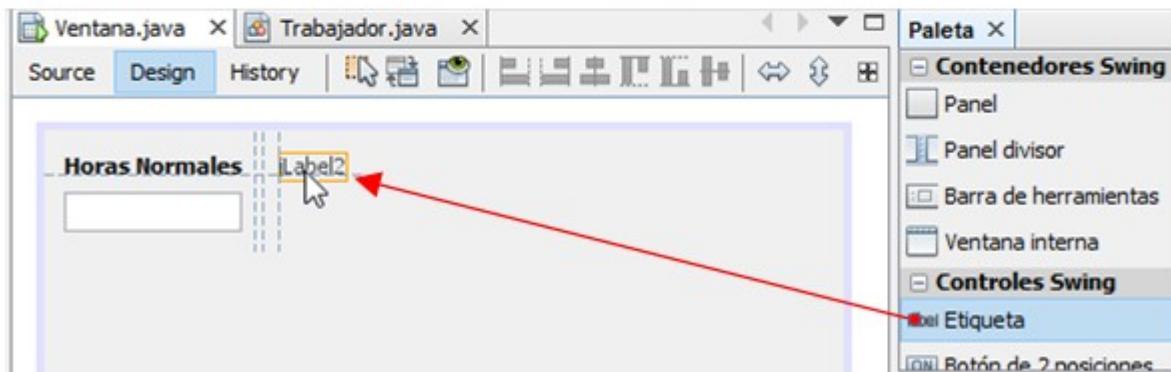
- Así se muestra este cuadro de dialogo, en el que ponemos el nombre para el control; en el ejemplo es **tf1** (tf es abreviatura de Text Field) y haga click en el botón **Aceptar**.



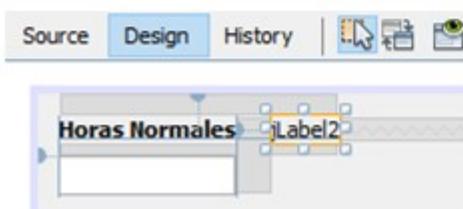
- Hecho esto evidenciamos el cambio, tanto en el navegador de componentes (imagen de la izquierda) como en el inspector de propiedades (imagen de la derecha), que encerramos en rojo aquí:



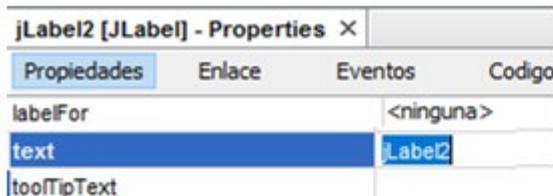
- Ahora arrastre hasta la ventana otra etiqueta (**JLabel**): para ello en el nodo **Controles Swing** de la paleta de controles, haga click sostenido sobre la etiqueta, arrástrela hasta el punto indicado abajo por la flecha roja y déjela en esa posición.



- Hecho esto la ventana tendrá un aspecto como el siguiente:



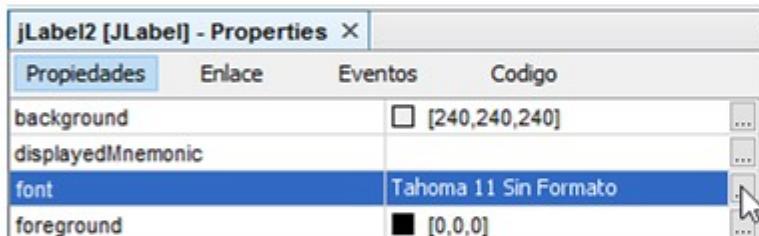
- En el inspector de propiedades busque la propiedad **text** y haga click a su derecha para editarla.



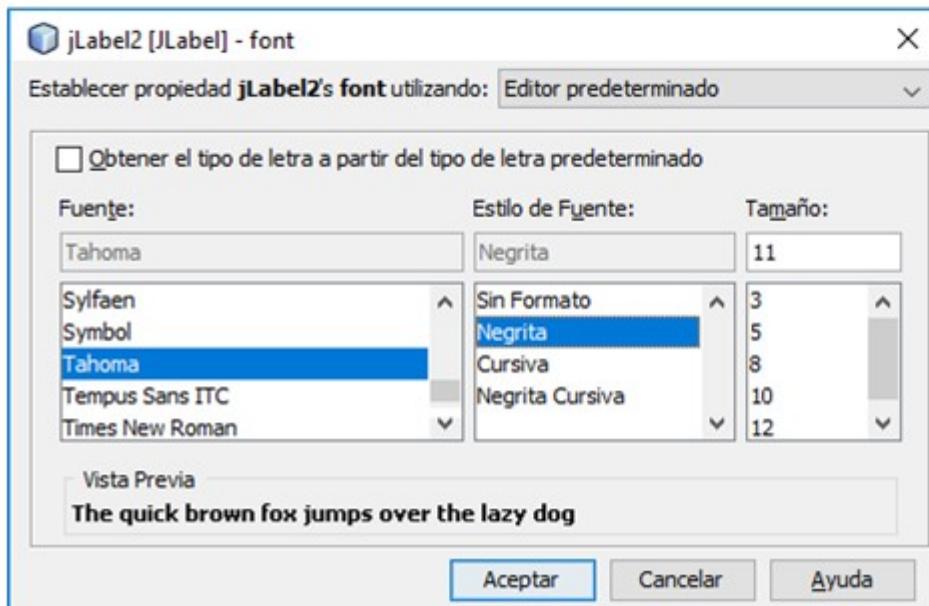
- Borre el texto *jLabel2* y escriba el texto: **Valor hora Nor.**, y luego pulse la tecla enter, con lo cual la ventana y el inspector de propiedades toman el siguiente aspecto.



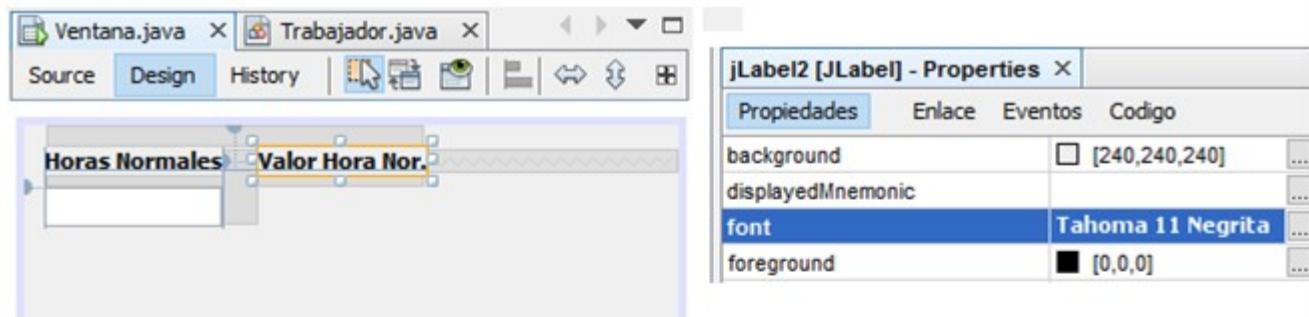
- Busque la propiedad **font**, haga click en el botón de los tres puntos que está a la derecha de la propiedad:



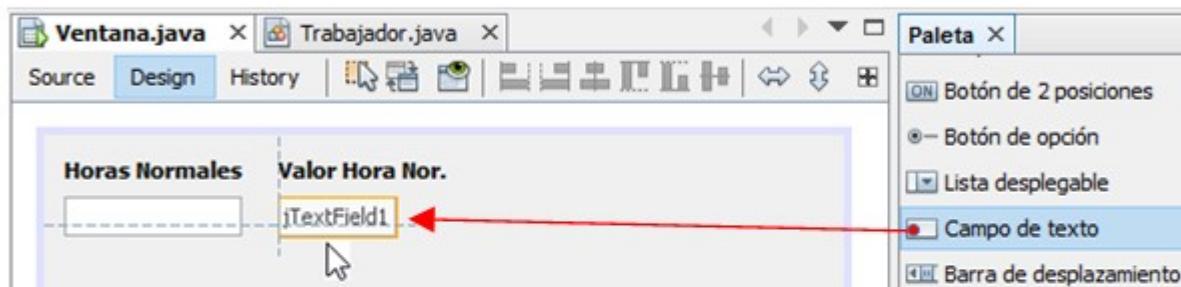
- Asigne el estilo de negrita al texto



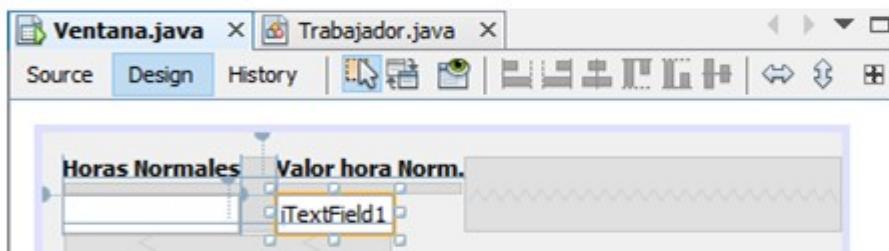
- Hecho esto la ventana y la propiedad tendrán el siguiente aspecto



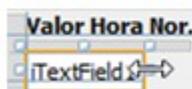
- Vaya al nodo **Controles Swing** de la paleta de controles, arrastre un campo de texto (**JTextField**) y suéltelo hasta la posición señalada por la fecha roja.



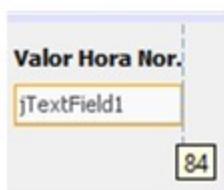
- La ventana ahora tiene el siguiente aspecto:



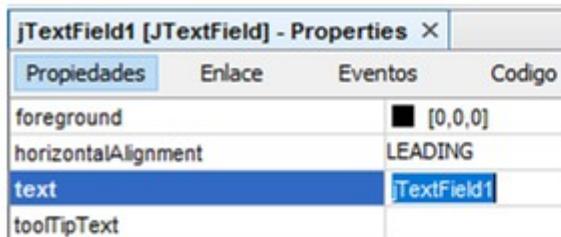
- Ajuste el ancho del campo de texto ampliéndolo, para lo cual acerque el puntero del ratón al nodo indicado abajo, hasta que el cursor se convierta en una flecha horizontal de línea doble.



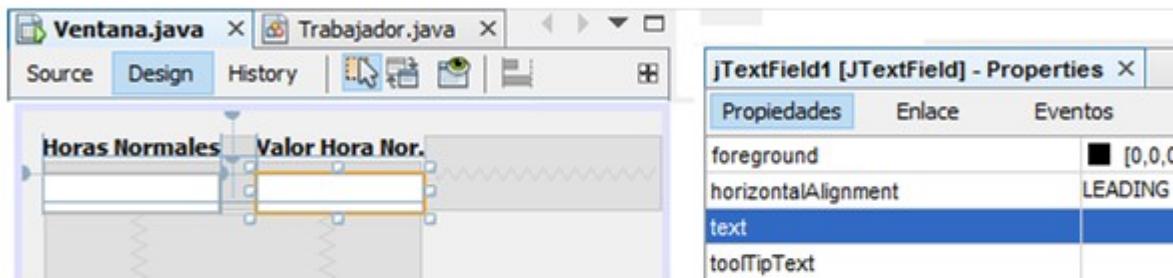
- Con el botón izquierdo del mouse pulsado, muévase hacia la derecha hasta que el ancho del control tenga un valor de 84 (pixeles) o el valor que usted prefiera.



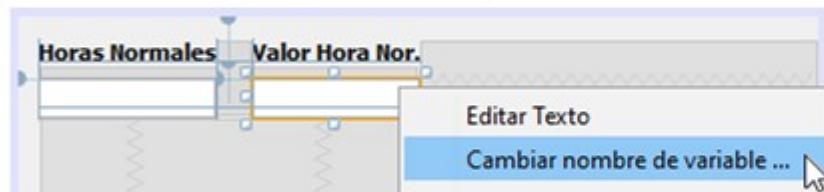
- Ahora en el inspector de propiedades, diríjase a la propiedad **text** y haga click en el texto que está a su derecha para editarlo:



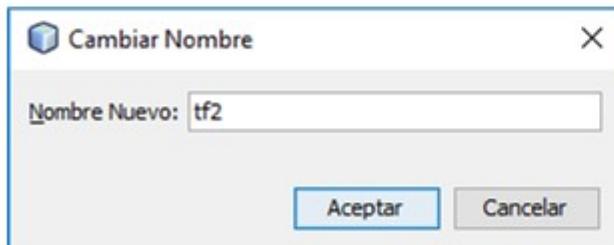
- Borre el texto (*jTextField1*) de la propiedad y pulse la tecla enter, con lo cual la ventana y el inspector de propiedades se verán así:



- Asigne ahora un nombre de variable (instancia) para el campo de texto, haciendo click derecho sobre este y click en la opción “Cambiar nombre de variable ...”

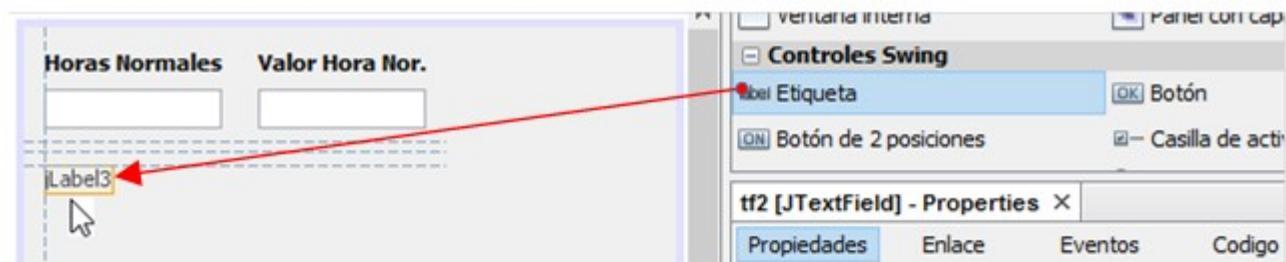


- Ahora se despliega el siguiente cuadro de dialogo, en el que ingresemos el nombre que deseamos darle al control; para el caso es **tf2** y enseguida haga click en el botón **Aceptar**.

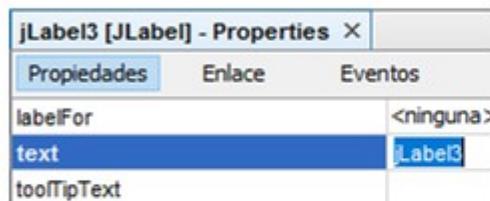


Nota: Hasta este punto es importante señalar que los nombres de variables usados para los controles, siempre deben ser distintos entre sí; además que deben cumplir las mismas reglas aplicadas, por ejemplo, a los nombres que le damos a los atributos de una clase; igualmente son nombres sensibles a mayúsculas y minúsculas, pues a fin de cuentas son usados desde el código de Java.

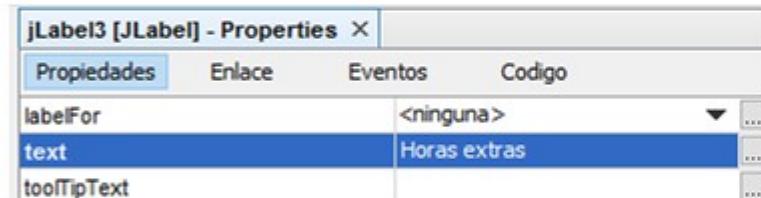
- Desde el nodo **Controles Swing** de la paleta de controles, arrastre hasta la ventana otra etiqueta (**JLabel**), poniéndola en la posición señalada abajo por la flecha roja.



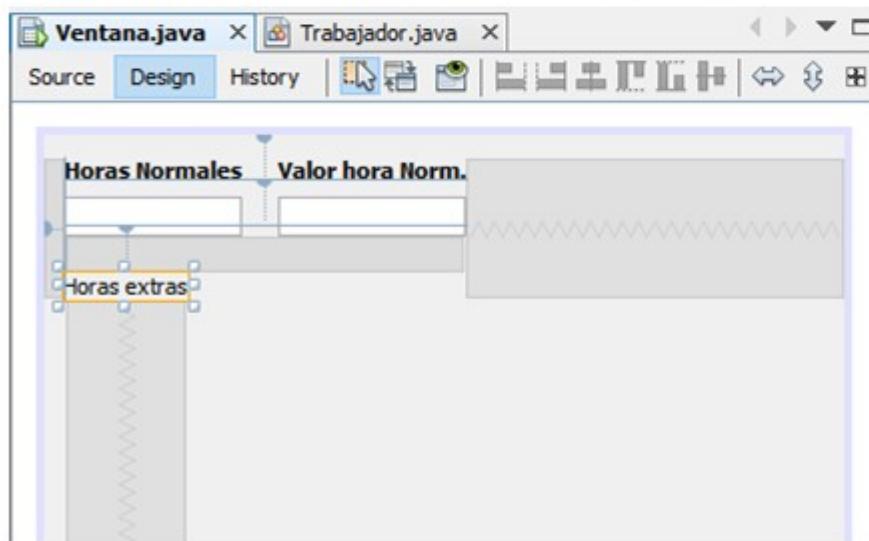
- Ahora en el inspector de propiedades, haga click en texto (jLabel3) de la derecha de la propiedad **text**.



- Borre ese texto y en cambio escriba: **Horas extras** y después pulse enter, quedando la propiedad como se muestra en esta imagen:



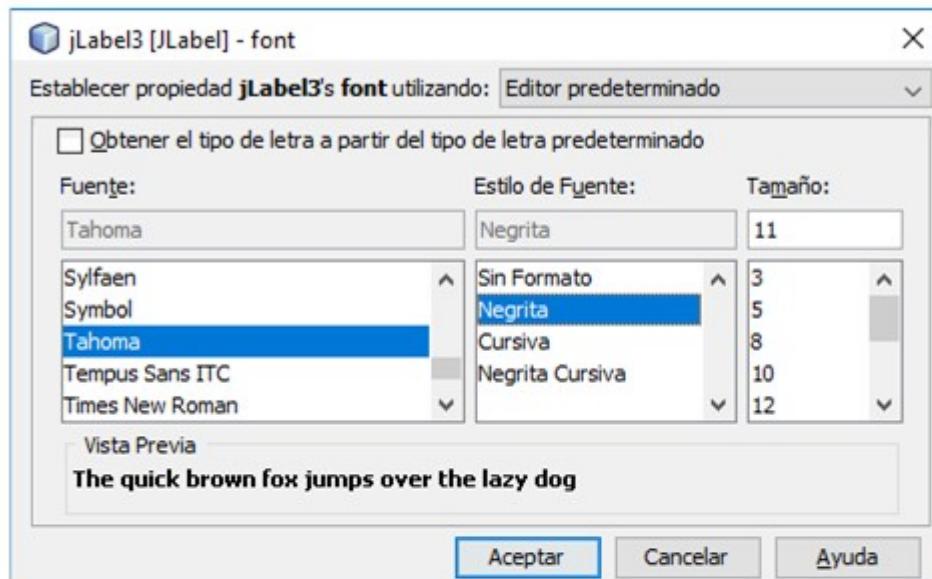
- La ventana por su parte, tomara el siguiente aspecto:



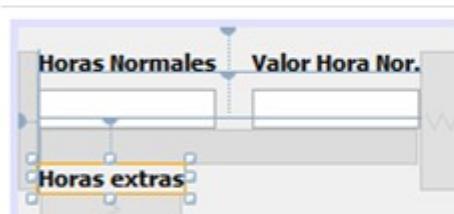
- Ahora busque la propiedad font en el inspector de propiedades, haga click en botón de los tres puntos de la derecha.



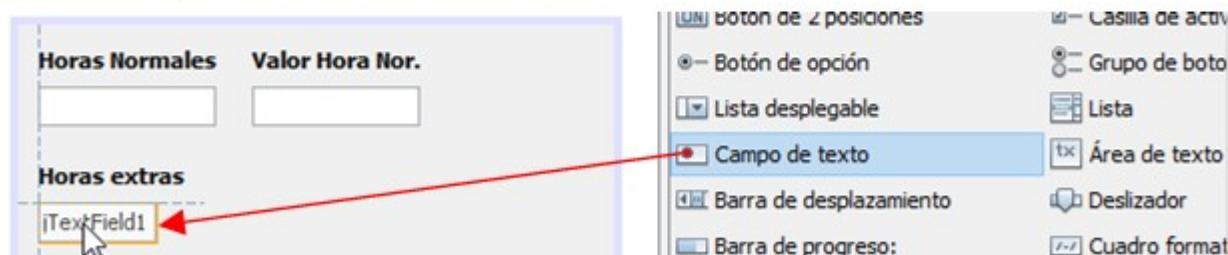
- Aplique el estilo de negrita al texto de la etiqueta y haga click en el botón **Aceptar**



- Quedando entonces los controles de la ventana en la siguiente forma:



- Del nodo **Controles Swing** de la paleta de controles, arrastre un campo de texto (**JTextField**) hasta la posición señalada por la fecha roja.



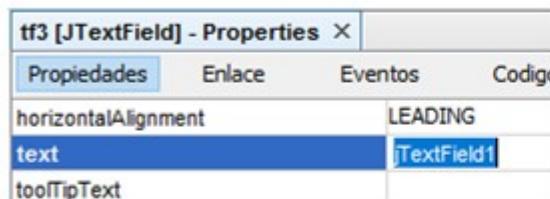
- Acerque el puntero del ratón al nodo indicado abajo, hasta que el cursor se convierta en una flecha horizontal de línea doble, con el fin de ajustar el ancho del campo de texto.



- Con el botón izquierdo pulsado, muévase hacia la derecha hasta que el ancho del control tenga un valor adecuado, para el ejemplo 73 (pixeles) o el que usted decida.



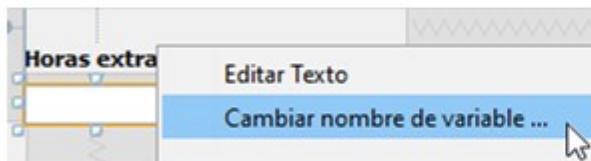
- En el inspector de propiedades seleccione la propiedad **text** para editarla.



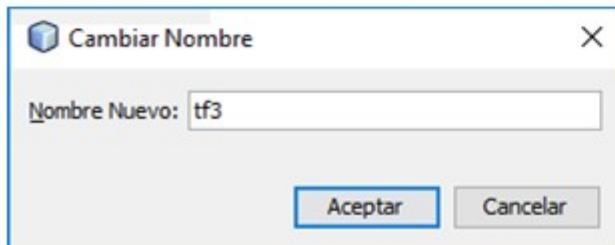
- Borre el contenido de esta propiedad y pulse enter, con lo cual la ventana y el inspector de propiedades se verán así:



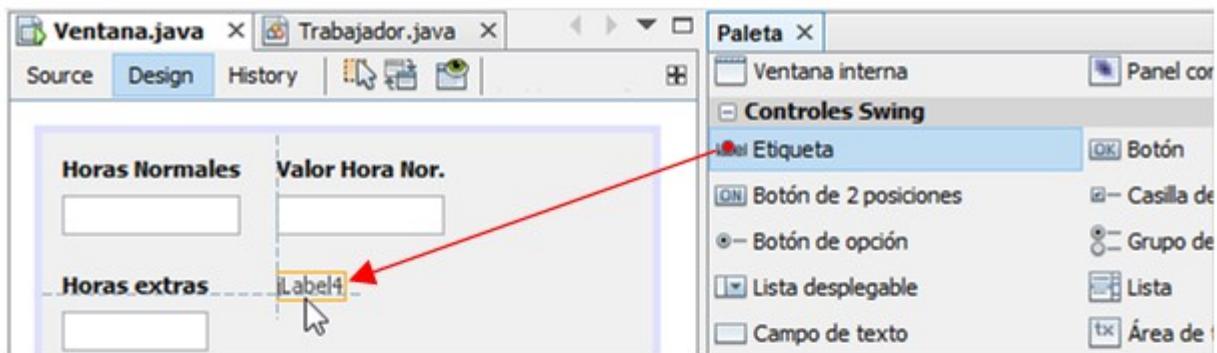
- Ahora cambie el nombre de variable al campo de texto, haciendo click derecho en este y seleccionando la opción “*Cambiar nombre de variable ...*”



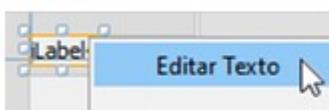
- En el cuadro de dialogo desplegado ingrese **tf3** y haga click en el botón **Aceptar**.



- Nuevamente desde la paleta de **Controles Swing** arrastre y suelte una etiqueta (**JLabel**) hasta donde lo indica la flecha roja.



- Otra manera de cambiar el texto de una etiqueta, es haciendo click derecho en ella y seleccionando la opción “*Editar Texto*”



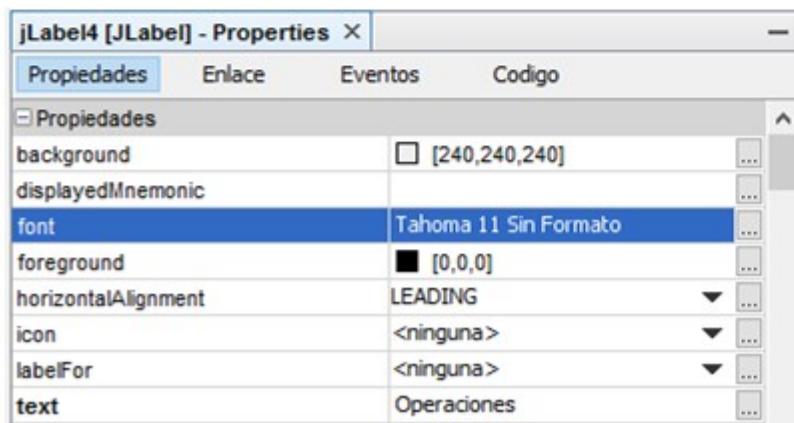
- Con ello la etiqueta tomara el siguiente aspecto:



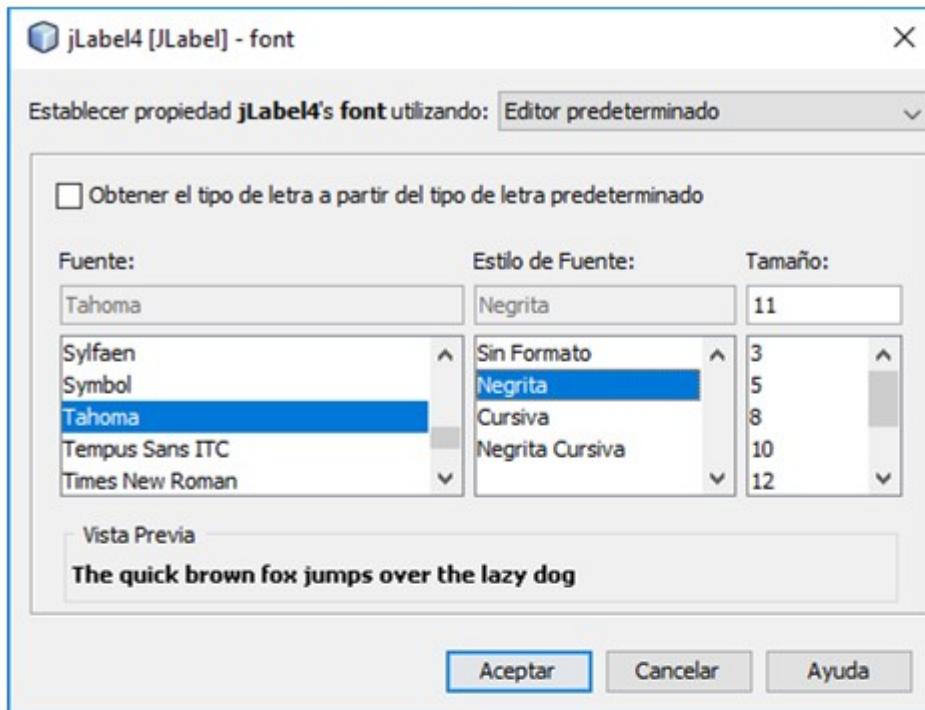
- Borre el contenido y escriba el texto: **Operaciones** y pulse enter, con lo cual la ventana se muestra así:



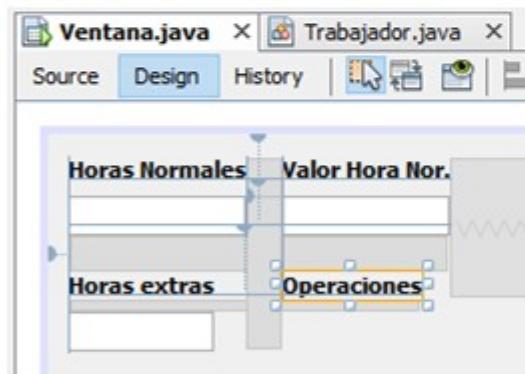
- En el inspector de propiedades haga click en el botón de los tres puntos, que está a la derecha de la propiedad **font**.



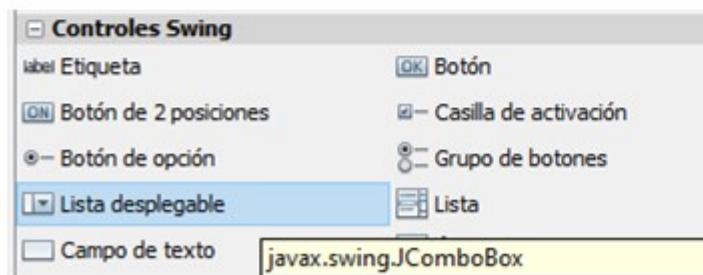
- Seleccione el estilo de Negrita y haga click en el botón **Aceptar**.



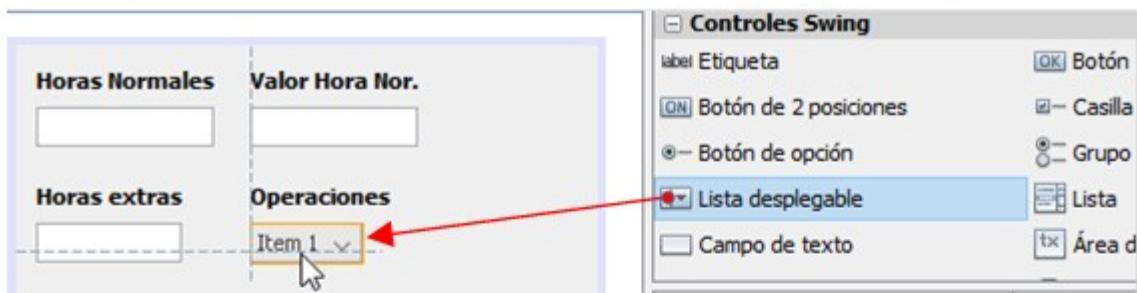
- Después de esto la ventana tendrá el siguiente aspecto:



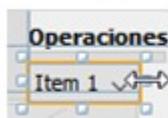
- Ahora en el nodo de **Controles Swing** de la paleta, seleccione una *Lista desplegable* (**JComboBox**), que se muestra en la siguiente imagen:



- Arrastre y suelte al **JComboBox** hasta donde lo indica la flecha roja



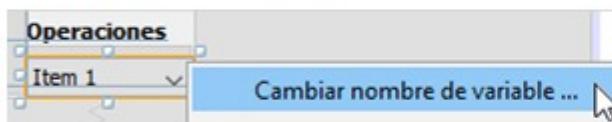
- Redimensione el **JComboBox**, para lo cual acerque el puntero del mouse al nodo indicado, hasta que el cursor se convierta en una flecha de doble horizontal:



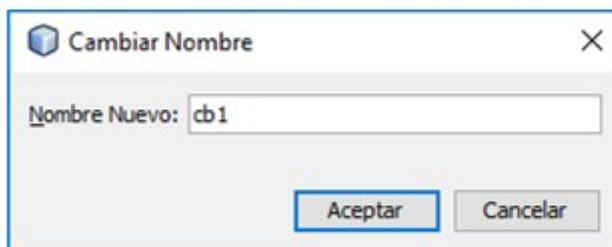
- Con el botón izquierdo del mouse presionado, muévalo hacia la derecha hasta que el control alcance un ancho apropiado (para el ejemplo 84 pixeles).



- Cambie el nombre al **JComboBox**, haciendo click derecho en él y escogiendo la opción "Cambiar nombre de variable ..."



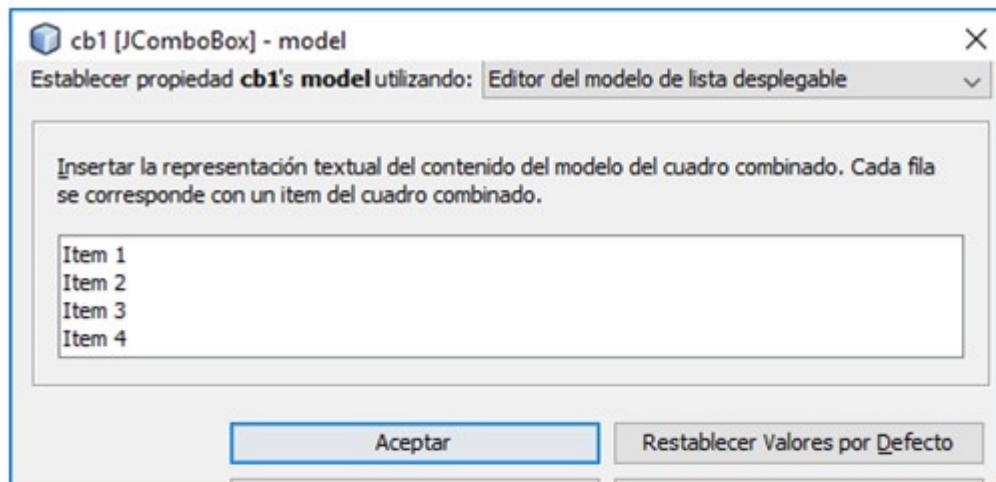
- En el cuadro de dialogo desplegado ingrese **cb1** y haga click en el botón **Aceptar**.



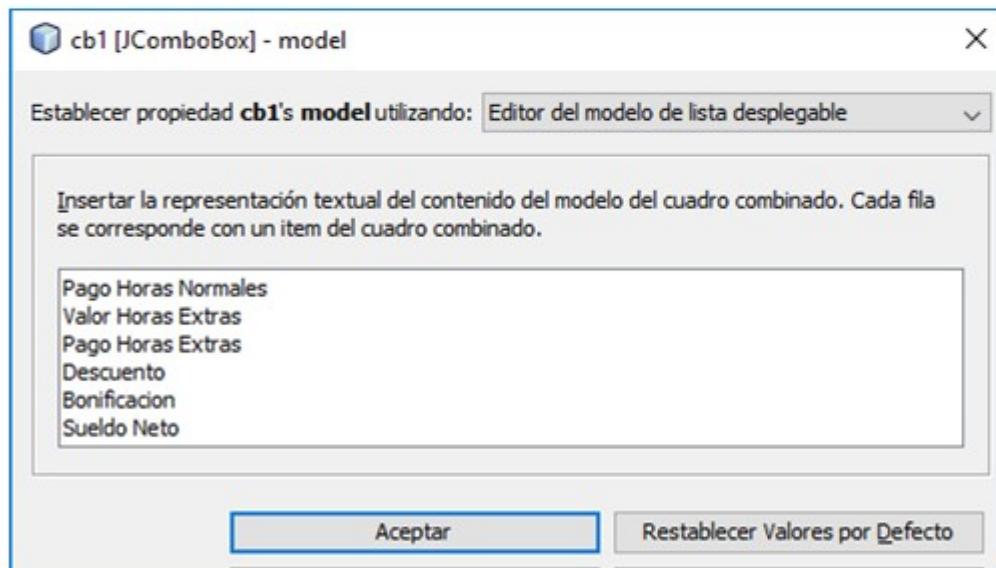
- Para asignar las opciones del **JComboBox**, en el inspector de propiedades localice la propiedad **model** y haga click en el botón de tres puntos que está a su derecha.



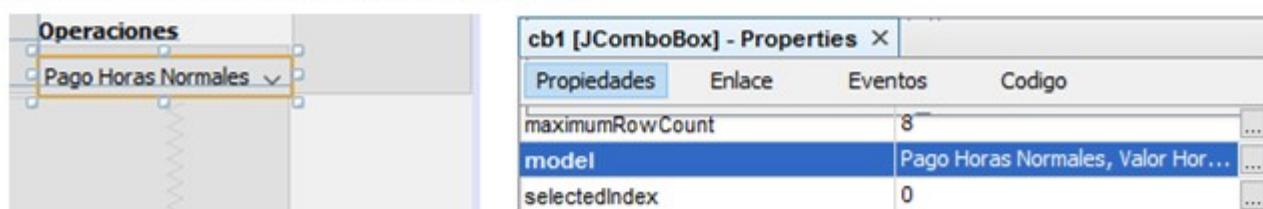
- Con ello se despliega el siguiente cuadro de dialogo, que muestra unas opciones por defecto (de Item1 a Item4).



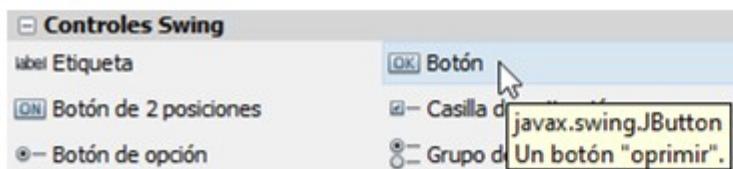
- Borre estas opciones y en cambio escriba las siguientes una debajo de otra: **Pago Horas Normales, Valor Horas Extras, Pago Horas Extras, Descuento, Bonificación y Sueldo Neto**; que se corresponden con los métodos de la clase Trabajador acorde al problema planteado.



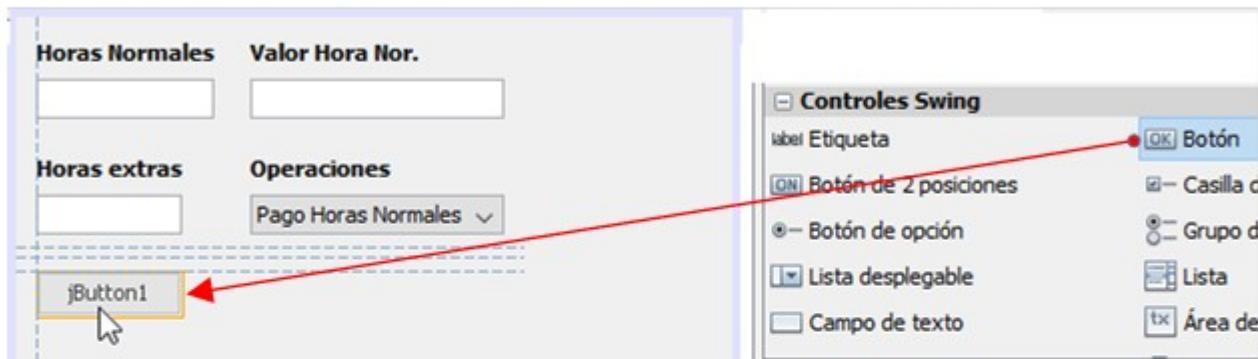
- Hecho esto, haga click en el botón **Aceptar**, con lo cual el control y el inspector de propiedades muestra el siguiente aspecto:



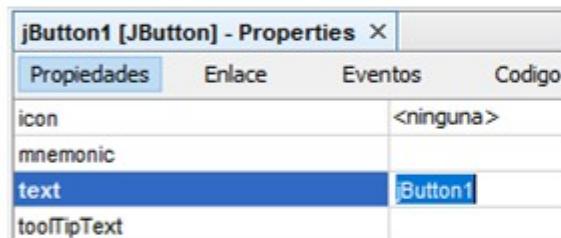
- Desde el nodo de **Controles Swing** de la paleta, seleccione un botón (**JButton**), el cual se muestra en la siguiente imagen:



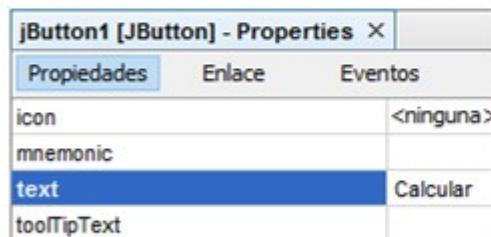
- Seleccione, arrastre y suelte al **JButton** hasta donde lo indica la flecha roja



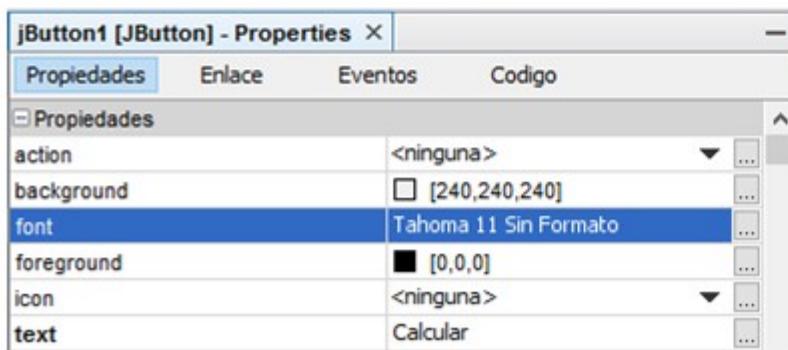
- En el inspector de propiedades seleccione la propiedad **text** para editarla.



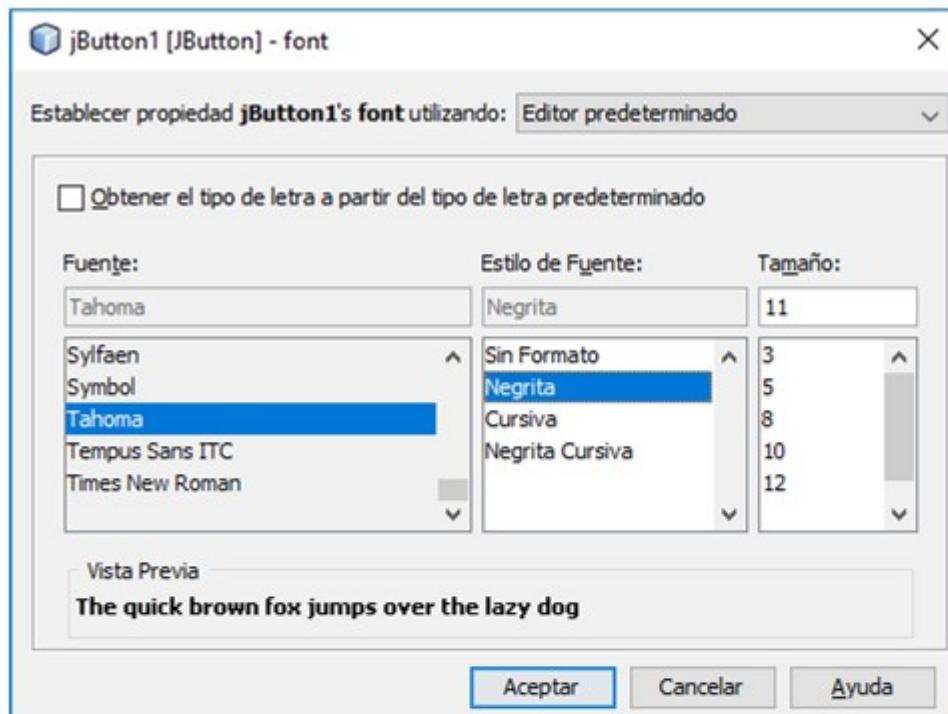
- Ahora borre el texto (*jButton1*) y escriba: **Calcular**, Luego pulse enter.



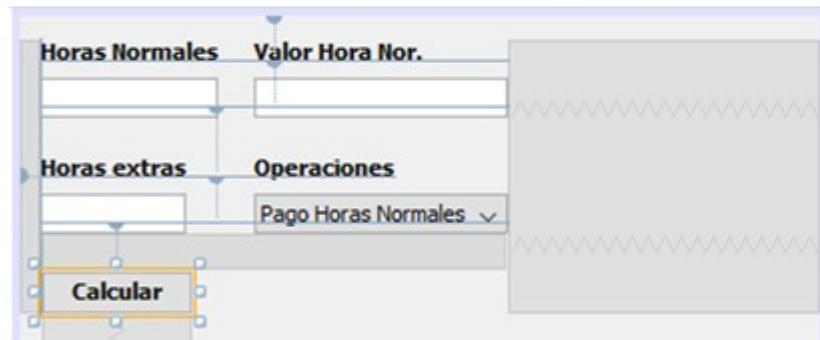
- Seleccione la propiedad **font** y haga click en el botón de tres puntos de la derecha



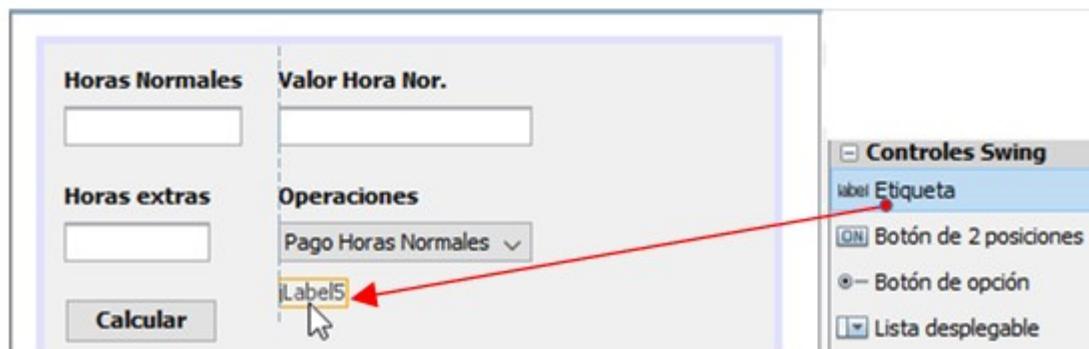
- Enseguida se muestra el siguiente cuadro de dialogo, escoja Negrita por estilo y haga click en el botón **Aceptar**.



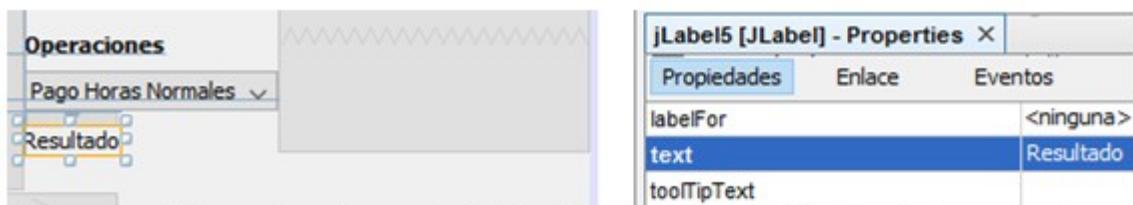
- Tras esto la ventana mostrará el siguiente aspecto:



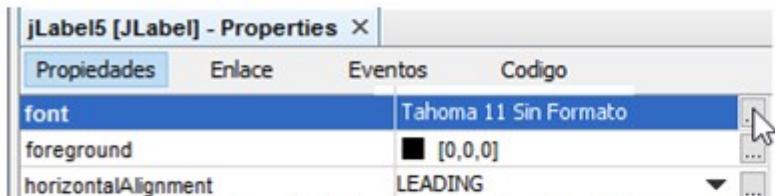
- Desde el nodo de **Controles Swing** de la paleta, seleccione, arrastre y suelte un **JLabel** hasta la posición indicada por la flecha roja



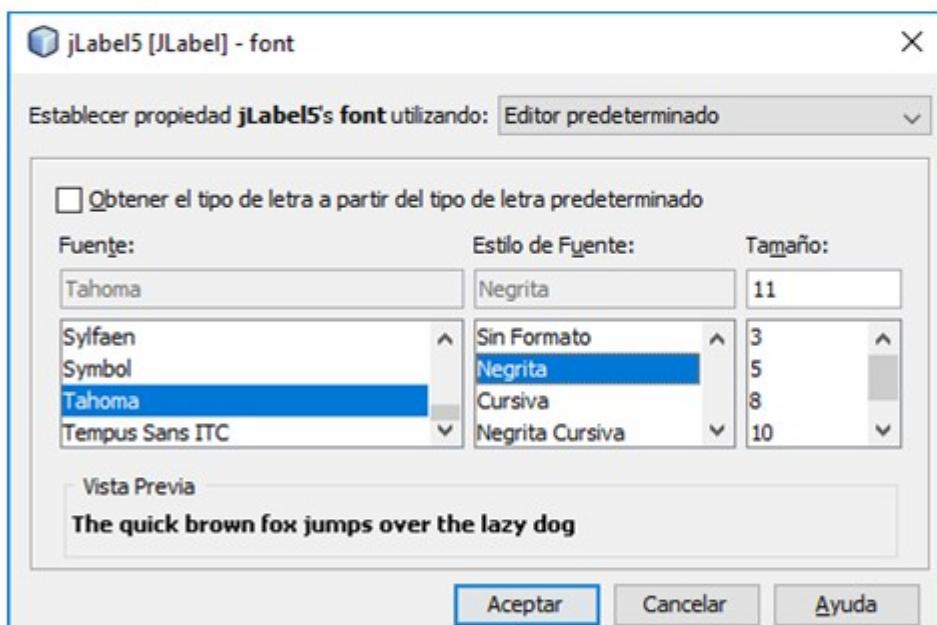
- En el inspector de propiedades, seleccione la propiedad **text**, borre su contenido, escriba el texto **Resultado** y pulse enter; quedando la ventana y la propiedad como se muestra abajo:



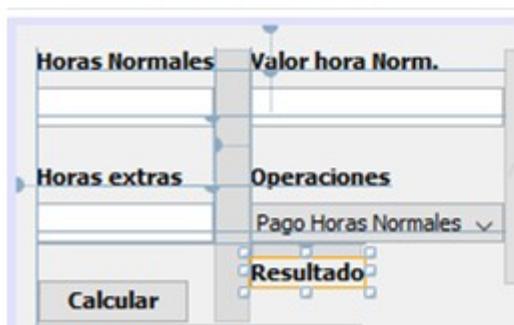
- Ahora en el inspector de propiedades haga click en el botón de tres puntos que está a la derecha de la propiedad **font**.



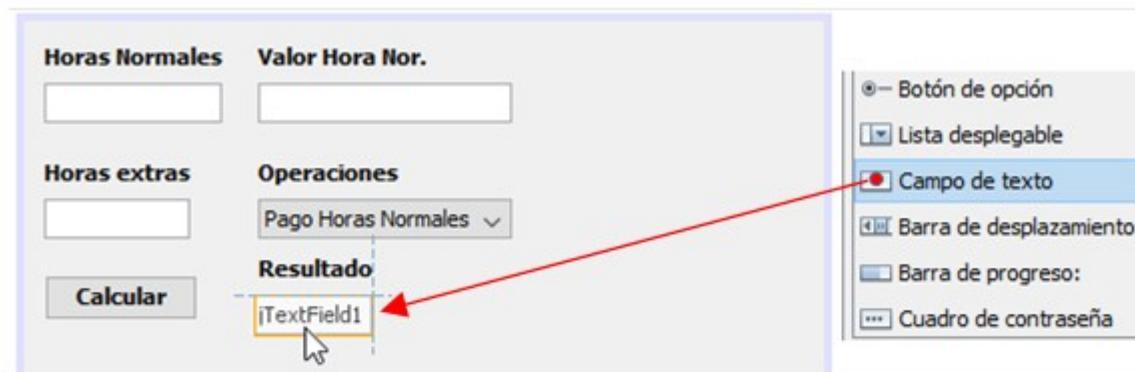
- Luego se muestra el siguiente cuadro de dialogo, ponga Negrita por estilo y haga click en el botón **Aceptar**.



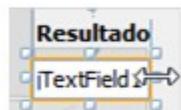
- Hasta este punto, la ventana se verá como sigue:



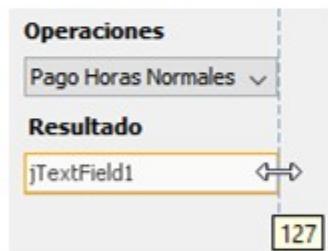
- Del nodo **Controles Swing** de la paleta, arrastre un campo de texto (**JTextField**) hasta la posición señalada por la fecha roja.



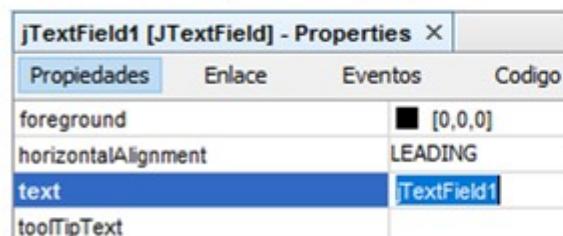
- Mueva el puntero del mouse hasta al nodo de redimensionamiento lateral, hasta que el cursor se convierta en una flecha de doble horizontal, de modo que podamos cambiarle el ancho al campo de texto:



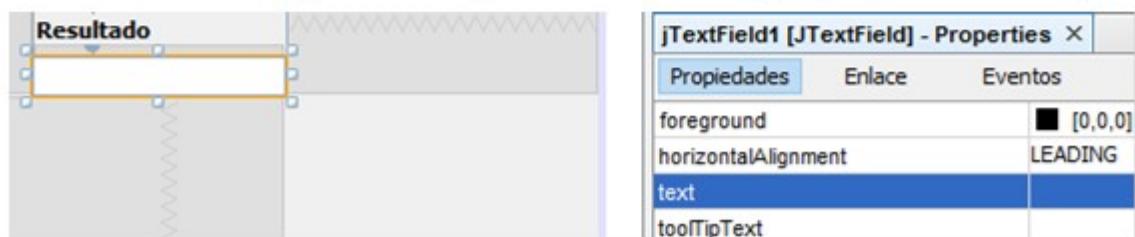
- Con el botón izquierdo pulsado del ratón, muévase hacia la derecha hasta que el ancho sea un valor adecuado, para el ejemplo 127 (pixeles) o lo que usted prefiera.



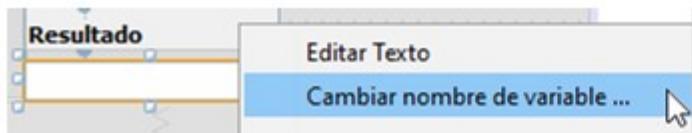
- Ahora en el inspector de propiedades seleccione la propiedad **text**.



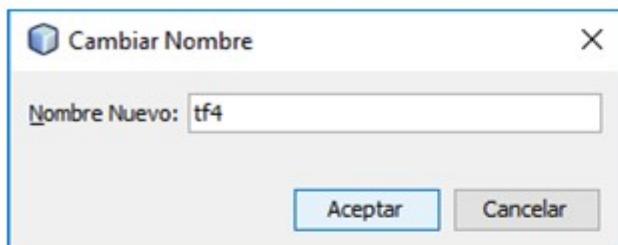
- Borre su contenido y pulse enter, quedando el control y la propiedad de la siguiente manera:



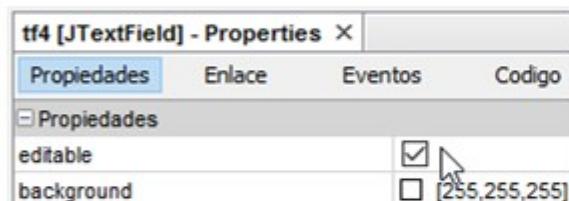
- Luego cambie el nombre de variable al campo de texto, haciendo click derecho en este y seleccionando la opción “Cambiar nombre de variable ...”



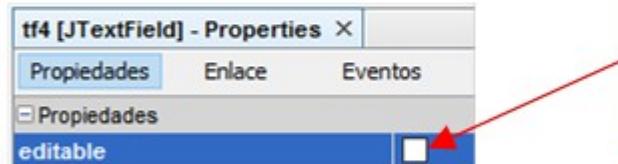
- En el cuadro de dialogo desplegado ingrese **tf3** y haga click en el botón **Aceptar**.



- En el inspector de propiedades busque y haga click en la propiedad **editable**.

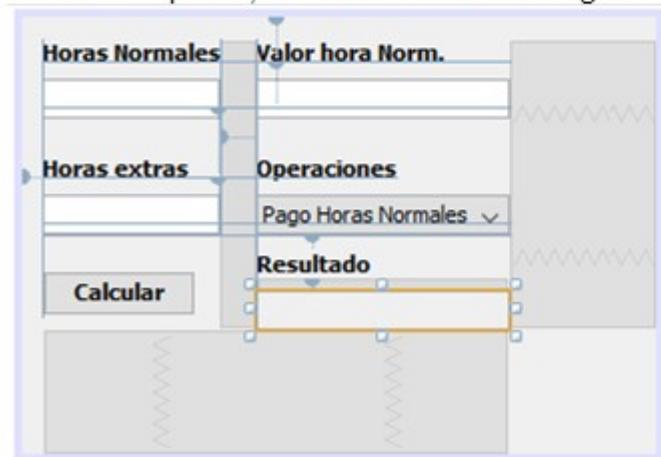


- Desmarca la casilla que está a la derecha de esta propiedad, que hace que la caja de texto sea de solo lectura; esto es, que el usuario no podrá cambiar ni barrar el texto de su contenido; no obstante, nosotros desde código Java si lo podemos hacer.

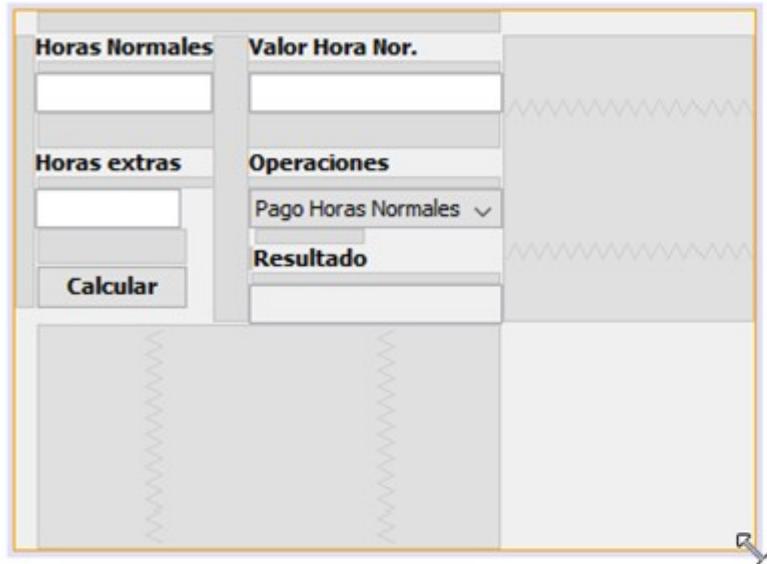


Asegúrese que esta casilla quede desmarcada (con el cuadro en blanco como se muestra aquí)

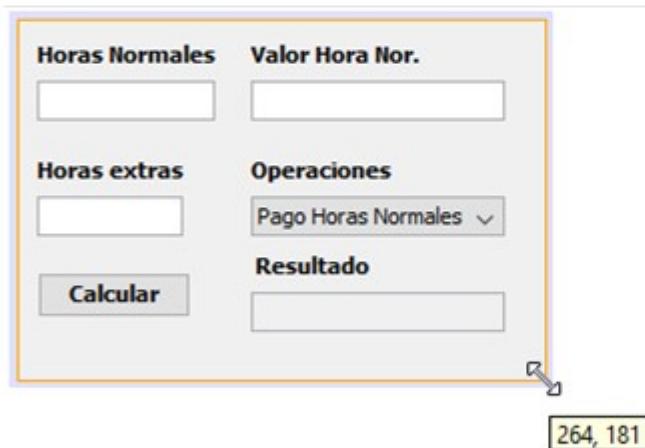
- Hasta este punto, la ventana tendrá la siguiente forma:



- De manera opcional podemos ajustar el tamaño de la ventana, para ello acerque el ratón hasta la esquina inferior derecha. Cuando el cursor se convierte en la flecha doble diagonal mostrada abajo, entonces haga click sostenido y mueva el mouse para cambiar el tamaño de la ventana.



- Recorte la ventana hasta un tamaño proporcional adecuado, para el ejemplo vemos en el recuadro amarillo 264,181 donde 264 es el ancho y 181 es el alto; ambos en pixeles.



- Para programar el evento click del botón **Calcular**, puede hacer doble click sobre el botón, pero también click derecho, seleccione **Eventos**, luego **Action** y finalmente **actionPerformed**



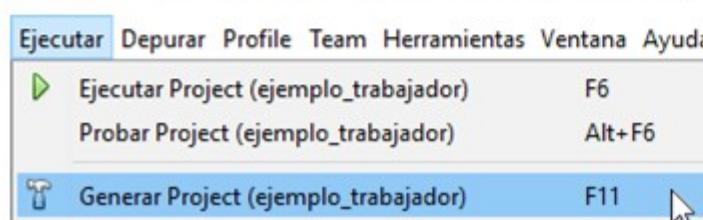
- En ambos casos el IDE activa la vista de código fuente de la ventana, de modo que se añade un método privado tipo **void** a la ventana, cuyo nombre está compuesto por el nombre del botón (**jButton1**) seguido por el tipo de acción del evento (**ActionPerformed**)

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}
```

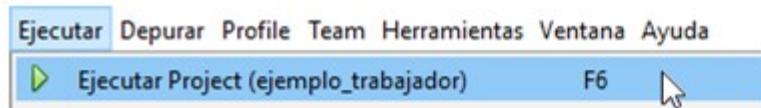
- La implementación la hacemos entre las dos llaves {}, es decir, borrando el comentario que dice: **//TODO add your handling code here**: y escribiendo nuestro propio código, que para el caso es el siguiente:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    float Res;
    Trabajador Tra;
    Res=0;
    Tra=new Trabajador();
    Tra.setHorNor(Integer.parseInt(tf1.getText()));
    Tra.setValHorNor(Integer.parseInt(tf2.getText()));
    Tra.setHorExt(Integer.parseInt(tf3.getText()));
    switch(cb1.getSelectedIndex()){
        case 0:Res=Tra.PagoNormales();break;
        case 1:Res=Tra.ValorHorExt();break;
        case 2:Res=Tra.PagoExtras();break;
        case 3:Res=Tra.Descuento();break;
        case 4:Res=Tra.Bonificacion();break;
        case 5:Res=Tra.SueldoNeto();break;
    }
    tf4.setText(String.format("% .2f",Res));
}
```

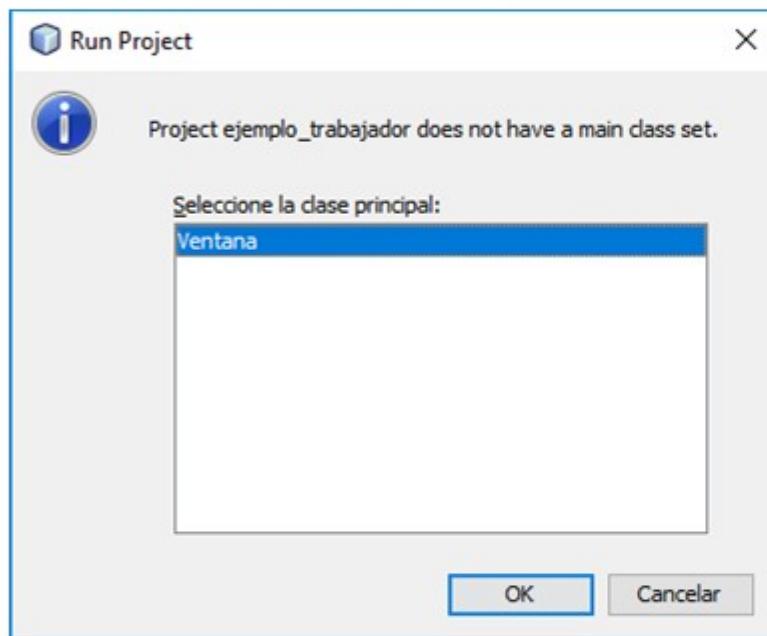
- Una vez hecho esto puede compilar el programa con la opción **Generar Project** del ítem **Ejecutar** del menú principal o pulsando la tecla F11.



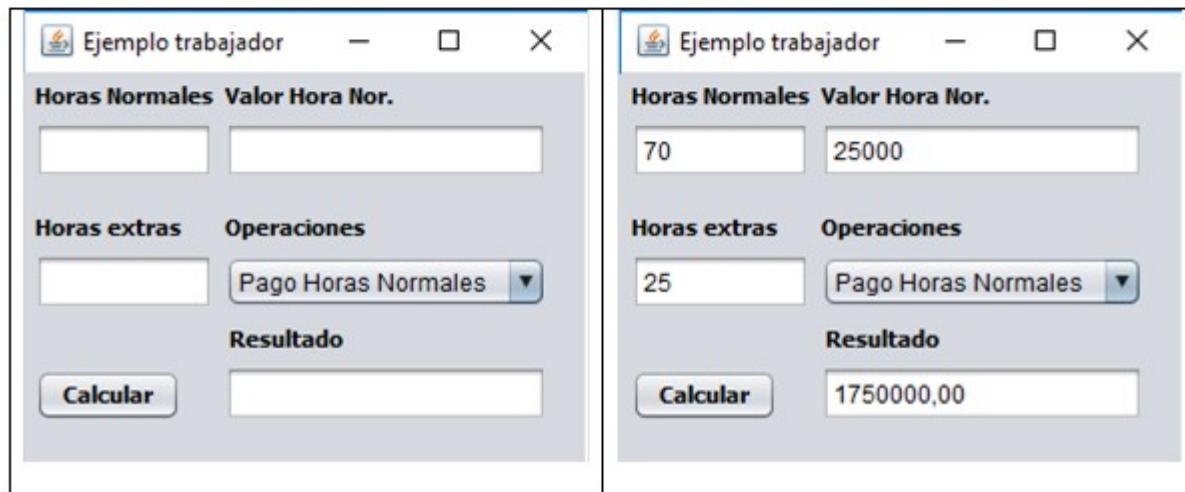
- ➁ Una vez pasada la compilación puede ejecutar el programa, con la opción **Ejecutar Project** del ítem **Ejecutar** del menú principal o pulsando la tecla F6.



- ➂ Cuando se ejecuta el programa por primera vez (y solo solo por la primera vez), *NetBeans* le mostrará el siguiente cuadro de dialogo, en el cual sugiere usar como clase principal la clase de la ventana (el **JFrame**), pues no incluimos la creación de una clase principal cuando creamos el proyecto; por lo tanto, hacemos click en el botón **OK**, para que nuestra ventana sea la clase principal del programa. :



- ➃ Las siguientes son algunas capturas de pantalla de la aplicación en ejecución:



Ejemplo trabajador

Horas Normales Valor Hora Nor.

70	25000
----	-------

Horas extras Operaciones

25	Valor Horas Extras
----	--------------------

Resultado

Calcular	38750,00
----------	----------

Ejemplo trabajador

Horas Normales Valor Hora Nor.

70	25000
----	-------

Horas extras Operaciones

25	Pago Horas Extras
----	-------------------

Resultado

Calcular	968750,00
----------	-----------

Ejemplo trabajador

Horas Normales Valor Hora Nor.

70	25000
----	-------

Horas extras Operaciones

25	Descuento
----	-----------

Resultado

Calcular	26250,00
----------	----------

Ejemplo trabajador

Horas Normales Valor Hora Nor.

70	25000
----	-------

Horas extras Operaciones

25	Bonificacion
----	--------------

Resultado

Calcular	0,00
----------	------

Ejemplo trabajador

Horas Normales Valor Hora Nor.

70	25000
----	-------

Horas extras Operaciones

50	Bonificacion
----	--------------

Resultado

Calcular	275000,00
----------	-----------

Ejemplo trabajador

Horas Normales Valor Hora Nor.

70	25000
----	-------

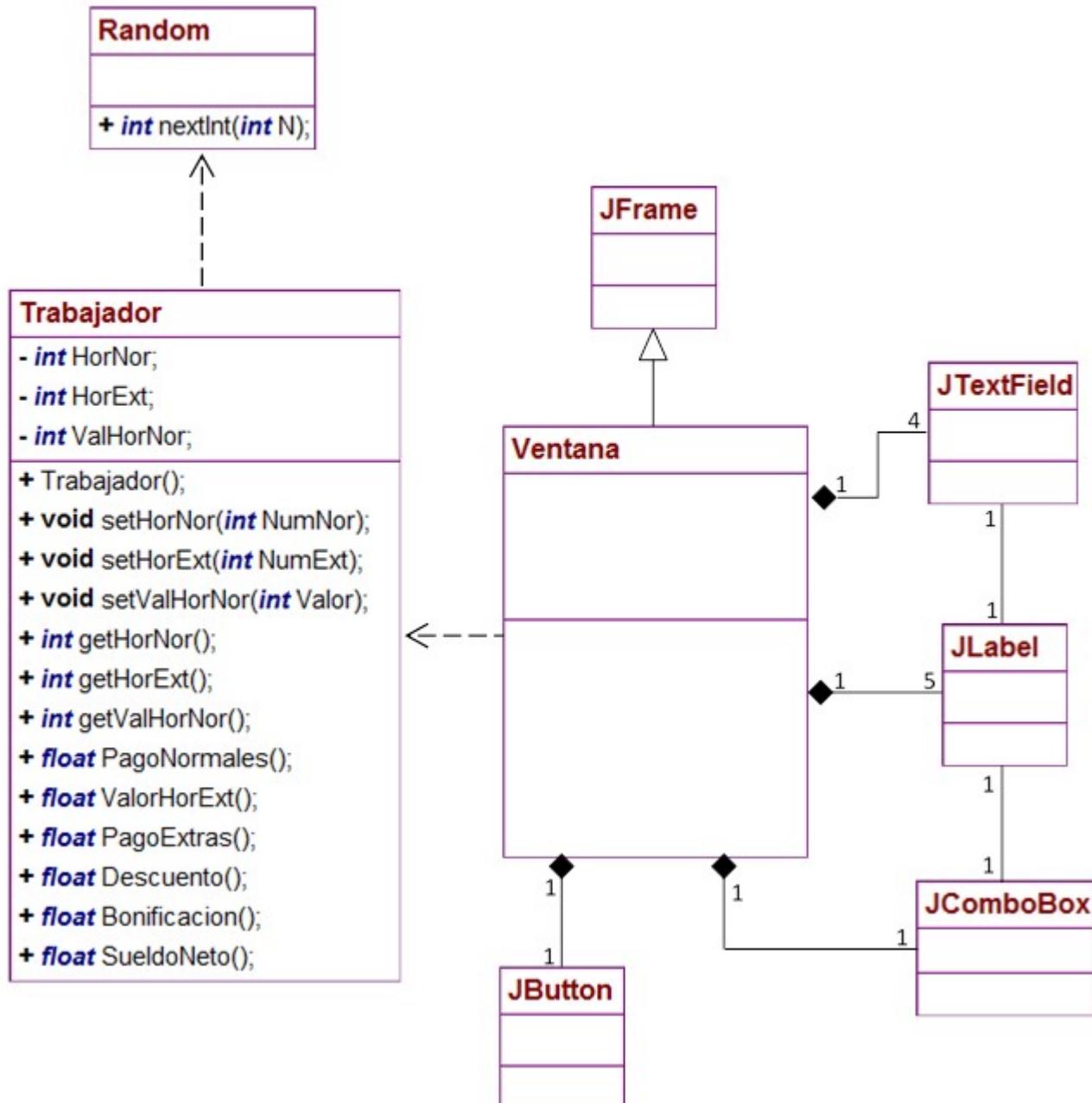
Horas extras Operaciones

25	Sueldo Neto
----	-------------

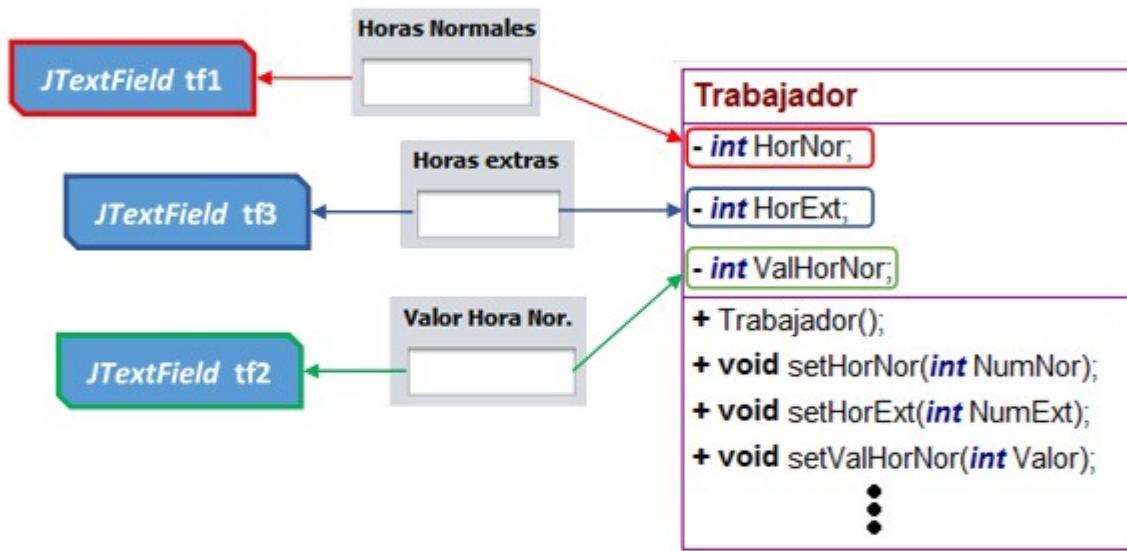
Resultado

Calcular	2692500,00
----------	------------

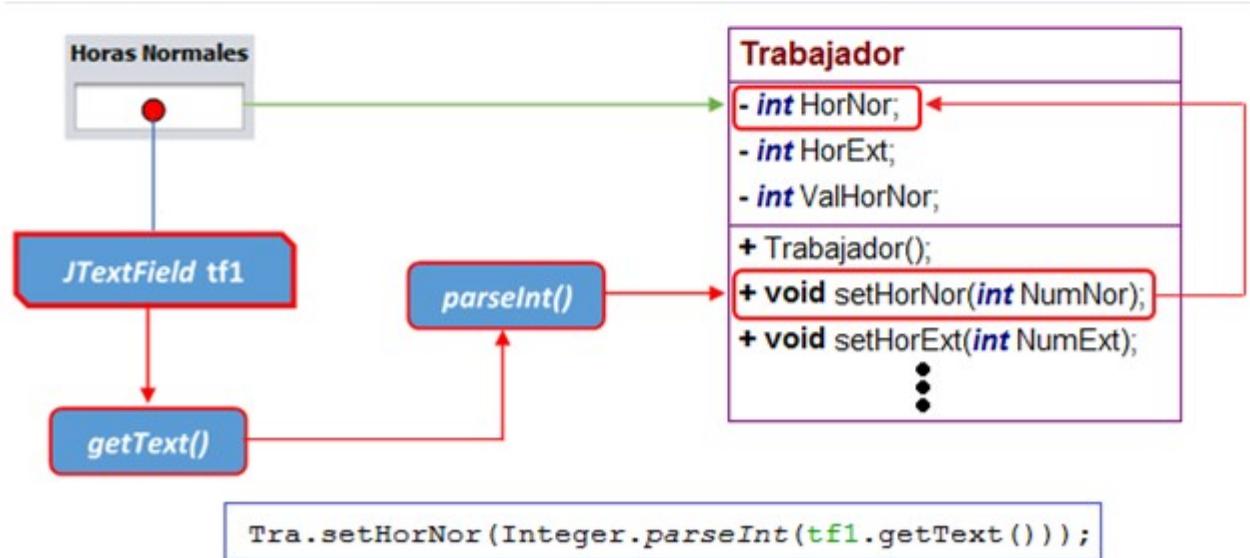
- El diseño UML de la lógica de la aplicación, con el de los controles de la ventana es el siguiente:



- En el siguiente grafico (que NO es para representar relaciones entre clases con UML), ilustramos la correspondencia de los controles **JTextField** de la ventana, con respecto a los atributos de la clase **Trabajador**: recordando que los atributos representan valores o datos de entrada para las instancias de la clase; razón por la cual en la ventana pusimos tres campos de texto para ingreso de datos, que se corresponden con los atributos **HorNor** (número de horas normales) con el campo de texto **tf1**, **HorExt** (número de horas extras) con el campo de texto **tf3** y **ValHorNor** (precio o valor de la hora normal) con el campo de texto **tf2**.

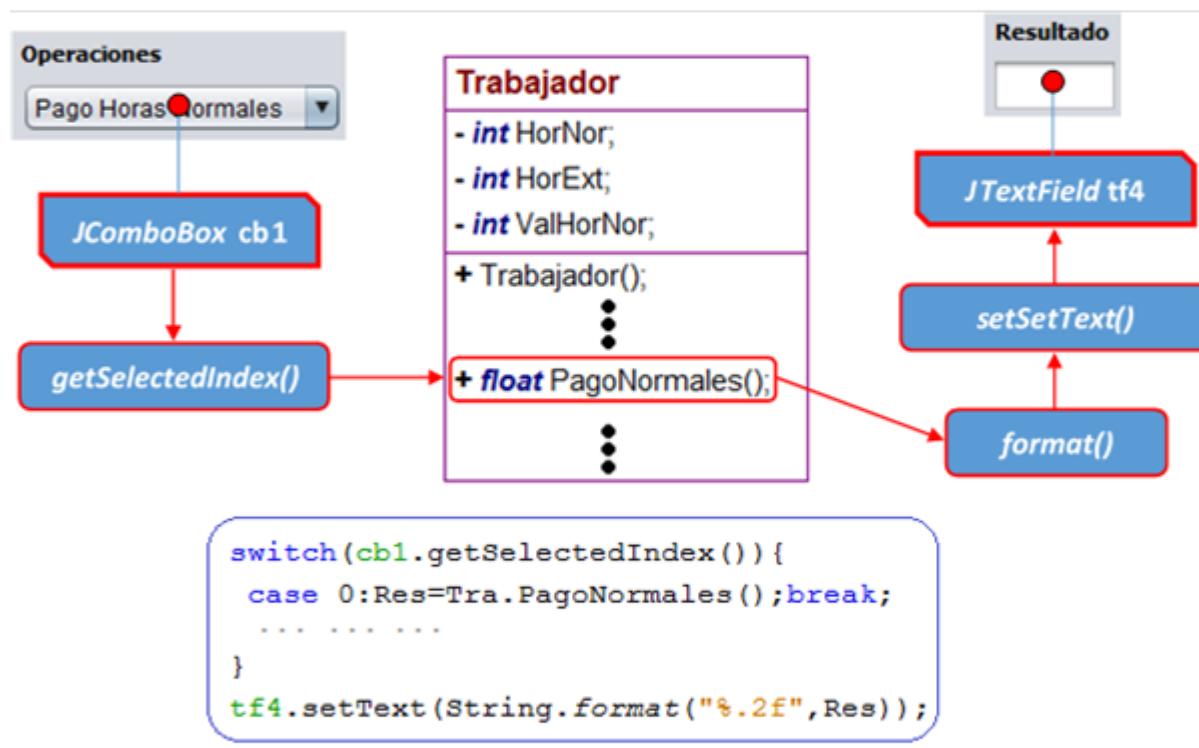


- El grafico mostrado abajo (que NO es para representar relaciones entre clases con UML), representa el flujo del valor ingresado por el usuario en el campo de texto **tf1**, hasta que se envía al atributo **HorNor** de la instancia (**Tra**) de la clase **Trabajador**.



Este esquema corresponde a la línea de código resaltada en el marco azul, donde podemos ver que el valor ingresado por el usuario en el **JTextField** para horas normales (**tf1**), es obtenido mediante el método **getText()**, el cual lo retorna como una cadena (**String**), la que a su vez se convierte a un número entero (**int**) con el método estático **parseInt()** de la clase **Integer**, después el valor ya convertido a entero es pasado como parámetro (entrada) al método modificador **setHorNor()**, el cual se llama desde instancia **Tra** de la clase **Trabajador**, que finalmente guarda dicho valor hasta el atributo **HorNor**.

- La siguiente imagen por su parte, muestra el flujo de ejecución del evento click o acción del botón **Calcular**, para el caso particular del cálculo del pago de las horas normales, de modo que la gráfica es una representación de las tres líneas de código enmarcadas en azul en la parte inferior de la misma.



De esta manera, vemos que la opción escogida por el usuario en el **JComboBox** (llamado **cb1**), la obtenemos mediante el método **getSelectedIndex()**, que retorna la posición o índice de la opción como un numero entero; así se selecciona la primera opción (*Pago Horas Normales*), el valor returnedo por este método será 0. Este valor se evalúa con la sentencia selectiva (**switch**), que para el caso 0 el valor del resultado (variable local **Res**) será el resultado devuelto por el método **PagoNormales()**, que es ejecutado usando la instancia **Tra** de la clase **Trabajador**. Después este resultado se pasa al método estático **format()** de

la clase **String**, para que lo convierta a una cadena de caracteres con dos dígitos decimales de precisión; finalmente el valor ya convertido a cadena se pasa como parámetro al método **setText()** del **JTextField** llamado **tf4**, para que muestre el resultado al usuario.

Ejercicios Propuestos

Para cada uno de los siguientes enunciados, se pide diseñar en UML el diagrama de clases e implementarlo en Java con una aplicación de ventana y empleando el IDE de *NetBeans*; tome en cuenta que es suficiente usar los mismos componentes de Swing, que empleados en ejemplo presentado anteriormente; es decir, solo usando controles **JLabel**, **JButton**, **JTextField** y **JComboBox**.

1. Realizar las cuatro operaciones básicas entre dos números reales cualesquiera.
2. Realizar las cuatro operaciones básicas entre dos números racionales.
3. Se requiere hallar la definitiva de un estudiante sabiendo que se toman tres exámenes y que se totalizan de la siguiente manera:
 - ✓ 30%, 30% y 40% si dos exámenes fueron reprobados.
 - ✓ 35%, 35% y 30% si dos exámenes fueron aprobados.
 - ✓ 25%, 40% y 35% si ningún examen fue aprobado.
 - ✓ 20%, 45% y 35% en cualquier otro caso.
4. Una tienda ha puesto en oferta la venta al por mayor de cierto producto, ofreciendo un descuento del 15% por la compra de más de 3 docenas y 10% en caso contrario. Además por la compra de más de 3 docenas se obsequia una unidad del producto por cada docena en exceso sobre 3. Se pide que determine el monto de la compra, el monto del descuento, el monto a pagar y el número de unidades de obsequio por la compra de cierta cantidad de docenas del producto.
5. Dada la siguiente serie $100/3, 103/2, 106/3, 109/2, 112/3 \dots$ se requiere: hallar la suma de sus N primeros términos, obtener la suma de los N primeros términos mayores que 100, calcular la suma de los términos que sean menores a 5000 y determinar cuántos términos es necesario sumar para superar un valor X dado por el usuario.