Trabajando con GUI en Java Componente java.swing

Autor: Wilder López Meléndez[1]
wlopezm@yahoo.com
Fecha: 26 de junio de 2007

Es muy común a la hora de crear aplicaciones de escritorios querer utilizar formularios que interactúen con los usuarios. Claro esto hace que tu aplicación sea amigable y querida por los mismos. Java provee un paquete llamado .**swing** que agrupa componentes ya definidos y que serán de mucha utilidad para el programador a la hora de diseñar nuestras aplicaciones; en la Figura 1 se observa un formulario creado con componentes swing.

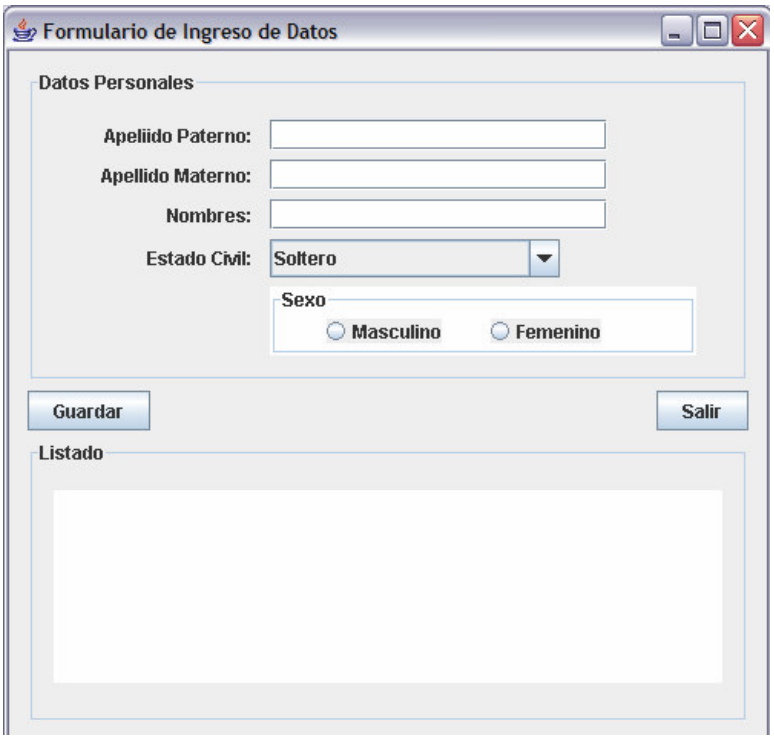

Figura 1: Aplicación de Ingreso de Datos utilizando java.swing

Empezare a guiarle pasa a paso para lo construcción de este formulario:
Lo vamos a construir en dos pasos primeros nos preocuparemos por el diseño de la GUI y luego empezaremos a programar nuestra aplicación:

PARTE I: DISEÑO LA GUI

Primero debemos ingresar al NetBeans 5.5, (sino lo tienes puedes bajarlo desde este link http://www.netbeans.org), vamos a crear un nuevo proyecto, ingresando al menú File – New Project, tal como se observa en la Figura 2,

Figura 2: Creando un nuevo proyecto

Aquí nos mostrará el cuadro de dialogo New Project, seleccionamos la categoría General y como proyecto Java Aplication, tal como se observa en la Figura 3.

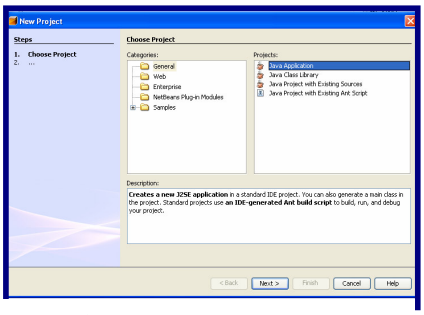

Figura 3: Nuevo Proyecto

Dentro de este nuevo proyecto vamos a insertar un JFrame del componente Swing, pulsamos clic derecho en el paquete luego seleccionamos la opción New, y finalmente seleccionamos del submenú la opción JFrame Form, tal como se observa en la Figura 4.

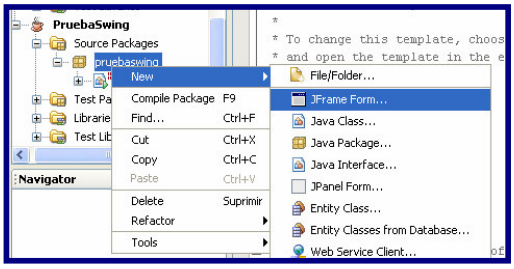

Figura 4: Creando un JFrame Form

Ahora nos encontramos dentro del entorno del paquete java.swing, un ambiente muy amigable, que no tiene nada que envidiar a otros leguajes visuales. La figura 5 muestra las partes del entorno

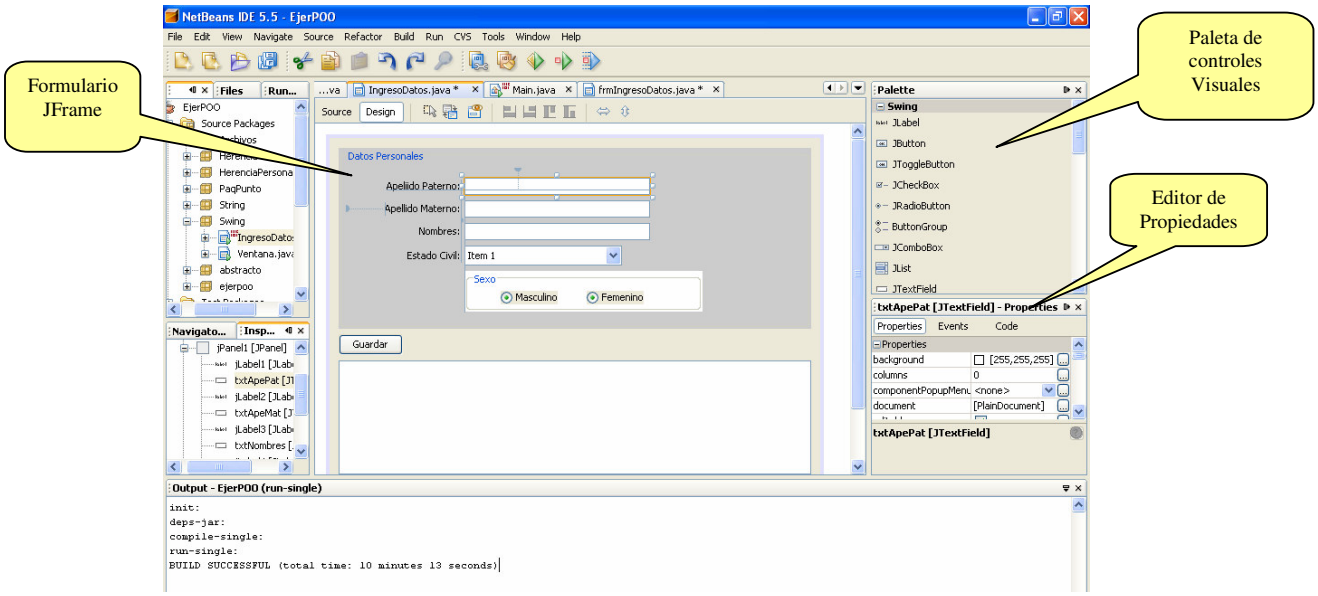

Figura 5: Entorno de diseño visual java.swing

El formulario JFrame es el lugar donde se irán colocando los controles de diseño para ir formando nuestra Interfaz Gráfica de Usuario, esto queda a criterio del diseñador de la interfaz, pero debo mencionar que existen normas de diseño de GUI que rigen en la Ingeniería de software que se debería tener en cuenta.

La Paleta de Controles, agrupa todos los controles (componentes Beans) que el paquete Swing trae, ejemplo JButton, JPanel, JLabel, JTextField, entre otros, podemos cambiar sus propiedades en cualquier momento utilizando el editor de propiedades.

Empezaremos a diseñar el formulario que se observa en la figura 5; para insertar un control de la paleta, debemos pulsar un clic en el control a insertar, luego ubicar con el puntero del mouse la posición en el JFrame a insertar (Ud. observará que el icono del puntero del mouse contendrá la imagen del control seleccionado) y pulse clic para colocar el control en el lugar seleccionado, luego Ud. podrá mover los controles hacia otro lugar más adecuado arrastrando con el Mouse, igual como si estuviera en cualquier ventana de una hoja de texto.

Resumen de los controles utilizados con sus propiedades.

Control	Propiedad	Valor
JPanel	Variable Name	jpanel1
JLabel	Variable Name	jLabell
	Text	Apellido Paterno:
JTextField	Variable Name	txtApePat
	Text	""
JLabel	Variable Name	jLabel2
	Text	Apellido Materno:
JTextField	Variable Name	txtApeMat
	Text	""
JLabel	Variable Name	jLabel3
	Text	Nombres:
JTextField	Variable Name	txtNombres
	Text	""
JLabel	Variable Name	jLabel4
	Text	Estado Civil:
JComboBox	Variable Name	cboEstCiv
JPanel	Variable Name	jPanel2
	Border	TitleBorder (ver figura 6)
ButtonGroup	Variable Name	grupSexo
JRadioButton	Variable Name	optSexoM
	Text	Masculino
	ButtonGroup	grupSexo
JRadioButton	Variable Name	optSexoF
	Text	Femenino
	ButtonGroup	grupSexo
Button	Variable Name	btGuardar
	Text	Guardar
Button	Variable Name	btSalir
	Text	Salir
JPanel	Variable Name	jPanel1
JTextArea	Variable Name	txtLista
	Text	""

Debes colocar estos controles en tu JFrame, teniendo en cuenta que dentro del JPanel se agrupan otros controles, esto nos permite dar una mejor apariencia a nuestra GUI, debes cambiar sus propiedades por las que aquí aparecen, tú puedes aumentar otras propiedades de acuerdo a tus necesidades.

El JPanel cuenta con una propiedad Border, para cambiar debes pulsar clic en el botón de examinar (…), tal como se observa en la figura 6, en ella estamos seleccionando TitleBorder para que nuestro panel tenga una etiqueta que lo identifique.

El control JComboBox permite llenar una lista para que el usuario pueda seleccionar de ella solo uno, por defecto te pone datos como Item1, Item2, Item3 e Item4, hay una propiedad llamada Model en la cual podemos colocar los datos manualmente pero no es recomendable utilizar esta opción ya que los datos de estado civil deberían venir de una tabla de Base de Datos y son ellos los que deberían de aparecer en mi lista, ¿como lograría esto?, en modo diseño es imposible, tiene que ser por código Java.

Otro control que también necesita una explicación es el ButtonGroup, este control nos permite agrupar varios JRadioButton, para poder seleccionar solo uno de ellos, es decir el usuario debe poder elegir entre Masculino o Femenino, nunca las dos opciones, bastará con ponerle un nombre al ButtonGroup en este caso lo pusimos grupSexo, y colocar en la propiedad ButtonGroup de los controles optSexoM y optSexoF al grupSexo, con esto estamos diciendo que estos dos controles pertenecen a un solo grupo, por lo tanto el usuario solo podrá seleccionar uno de ello.

Hay muchos diseñadores de GUI que no acostumbran poner nombres a los controles, pero debo decirles que esto es una mala práctica de Desarrollo de Software, yo les recomiendo que solo coloquen nombres a los controles que van a ser utilizados en la ventana de código, esto nos hará más fácil identificar nuestros controles. En nuestro caso hemos colocado nombres a los controles JTextField, JComboBox, JRadiobutton y a los JButton's, pero no hicimos eso con los JLabel's, la razón fundamental es que lo JLabel's solo nos servirán para mostrar etiquetas, no procesaremos la información que en ella lleva desde la ventana de código.

Ahora ya hemos logrado crear nuestra Interfaz Gráfica de Usuario (GUI), tan solo utilizando los componentes del paquete SWING, como habrá notado hasta ahora no hemos ingresado ninguna línea de código, este es la ventaja del .Swing con respecto al AWT en ella hubiéramos tenido que programar todo. Muchos de mis alumnos se quedan maravillados al pasar de .AWT a .Swing, después de programar hasta una simple etiqueta y tener que lidiar con el BorderLayout.

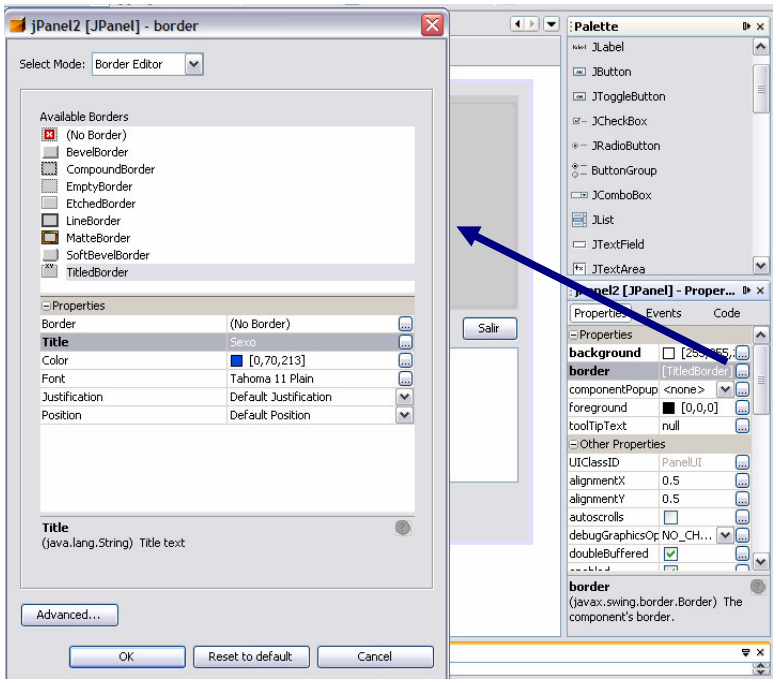

Figura 6: cambiar la propiedad Border al JPanel

PARTE II: INGRESANDO CÓDIGO JAVA A NUESTRA GUI

Ahora hemos llegado a la parte donde tenemos que hacer que funcione nuestra aplicación y para eso necesitamos programar ciertas cosas un poco de código de Java.

Voy a explicar cual es la lógica de la aplicación, como se habrán dado cuenta estamos trabajando un formulario de captura de datos, el usuario deberá ingresar los datos que en el formulario se solicita, cuando se pulse clic en el botón guardar debemos insertarlo en el cuadro de texto llamado txtLista, pero antes debemos validar los datos ingresados.

En la página anterior mencionamos que la lista que se debe mostrar en el cboEstCivil debe hacer por código, entonces manos a la obra:

Vamos a utilizar el evento windowOpened que ocurre cuando una ventana es abierta, allí llenaremos nuestro cboEstcivil, este lugar es ideal para inicializar variables. En algún espacio vacío del JFrame pulse clic derecho para mostrar un menú de opciones seleccione la opción Events – Window – windowOpened, tal como se observa en la figura 7,

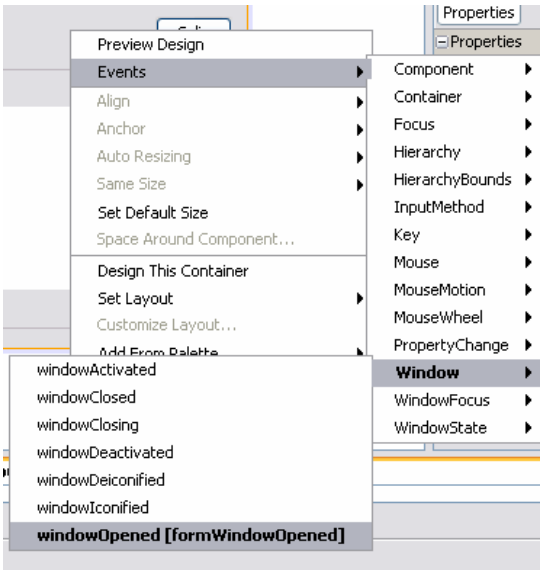

Figura 7: Menú de opciones del JPanel

Aquí ingresamos los estados civiles actuales con la propiedad .addItem(“”), pero primero debemos borrar los datos que nos puso por defecto el control con la propiedad .removeAllItem();

```
private void formWindowOpened(java.awt.event.WindowEvent evt) {
    cboEstCivil.removeAllItems();
    cboEstCivil.addItem("Soltero");
    cboEstCivil.addItem("Casado");
    cboEstCivil.addItem("Divorciado");
    cboEstCivil.addItem("Viudo");
}
```

Pulse clic en la pestaña Diseño que se encuentra en la parte superior de la ventana si desea seguir modificando el JFrame. Tal como se observa en la figura 8.

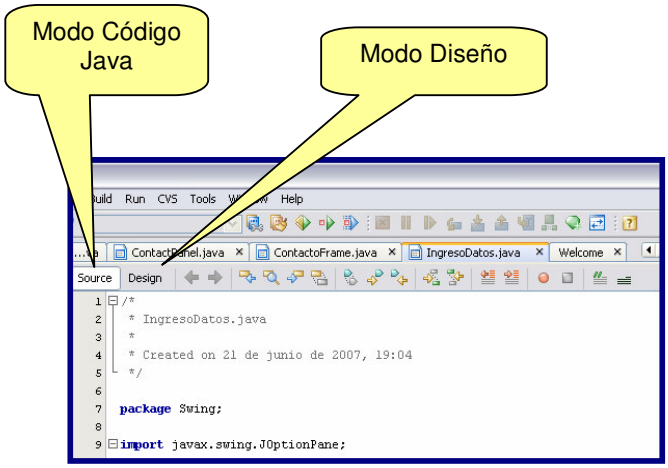

Figura 8: Cambiar de Modo Diseño a Modo Código de Java

Ahora nos toca programar el evento ActionPerformed del JBoton btGuardar, para ello pulse clic derecho en el botón Guardar, luego selecciones Event – Action – ActionPermormed, tal como se observa en la figura 9.

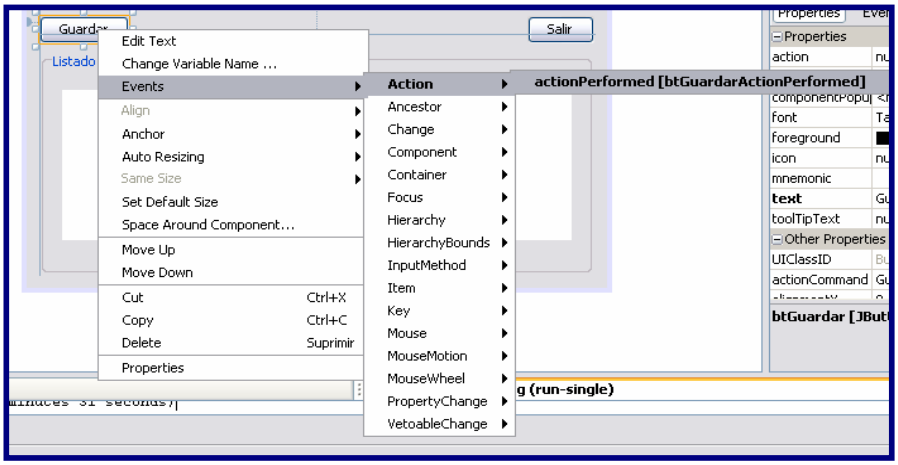

Figura 9: Menú de opciones del JButon

Aquí programamos la lógica del negocio del botón, este evento ocurrirá al pulsar clic

```
private void btGuardarActionPerformed(java.awt.event.ActionEvent evt) {
    //validando entradas
    String data="";
    if (txtApePat.getText().equals("")){
        JOptionPane.showMessageDialog(null,"Ingrese Apellido Paterno");
        return;
    }
    if (txtApeMat.getText().equals("")){
        JOptionPane.showMessageDialog(null,"Ingrese Apellido Materno");
        return;
    }
    if (txtNombres.getText().equals("")){
        JOptionPane.showMessageDialog(null,"Ingrese Nombres");
        return;
    }
    //grabamos
    data = txtLista.getText() + txtApePat.getText() + "  " + txtApeMat.getText() +
           "  "+txtNombres.getText() + "  " + cboEstCivil.getSelectedItem();
    if ( optSexM.isSelected() )
        data += "  Masculino";
    else
        data += "  Femenino";
    txtLista.setText(data+"\n");
}
```

Primero empezamos validando el ingreso de datos, preguntando si la propiedad Text del txtApePat contiene datos vacíos, para ello utilizamos el método .getText() que me devuelve el contenido de la propiedad Text. Esto es así en Java por la Programación Orientada a Objeto POO, Text es un atributo de la clase JTextField, y sabemos que la teoría de encapsulamiento prohíbe el acceso a ello fuera de la clase.

Regresando a lo nuestro, tenemos que si el Apellido Paterno esta vacío mandamos un mensaje por medio del JOptionPane solicitando el ingreso del mismo, y finalmente hacemos un return abortando el evento actionPerformed. Lo mismo hacemos con el Apellido Materno, y el Nombre.

Una vez validado el ingreso de los datos, concatenamos a una variable data el contenido de los controles, note que para el cboEstcivil utilizamos el método .getSelectedItem(), que nos devuelve el item seleccionado de la lista.

Finalmente para saber que selecciono el usuario como sexo, utilizamos el método .isSelected() del control optSexoM, y concatenamos la palabra “Masculino” o “Femenino”, y por último asignamos la variable data a la propiedad Text del control txtLista por medio del método .setText(data+”\n”).

Listo ahora ya podemos echar a andar nuestra aplicación pulsando las teclas Shift+F6, obtendremos un resultado con en la figura 10

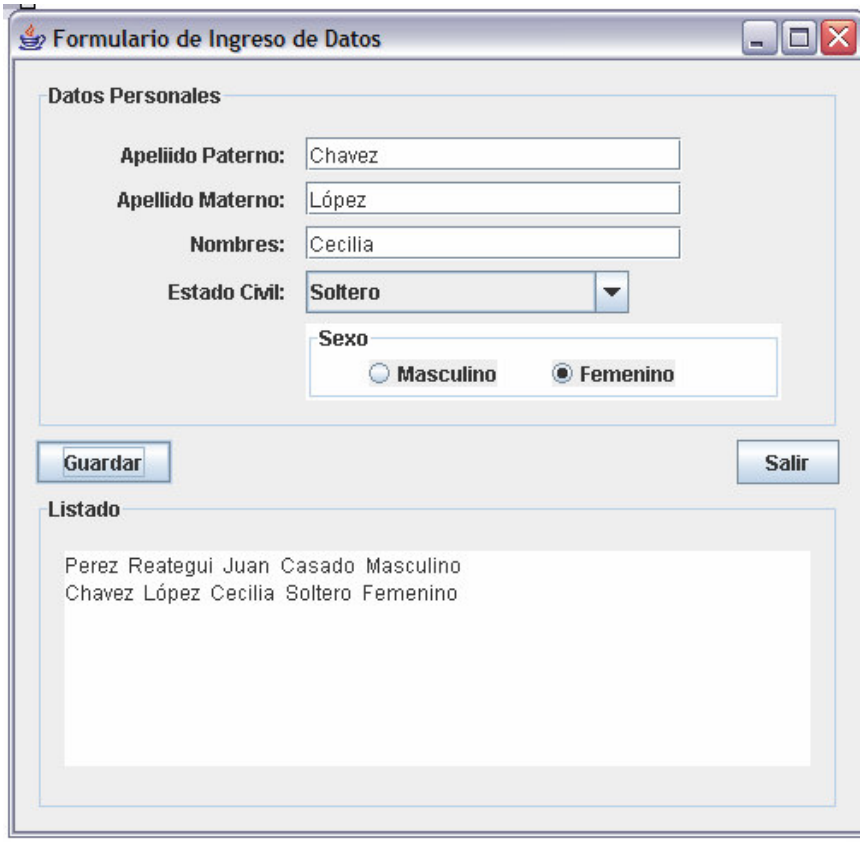

Figura 10: Resultado final de la aplicación de ingreso a Datos

---

[1] **Acerca del Autor**

- Director de la Oficina General de Sistemas e Informática Universidad Nacional de Ucayali Docente Universitario en J2EE y J2SE.
- Actualmente dirige varios proyectos de desarrollo de software WEB en J2EE
- Cuenta con 10 años de experiencia en el área de Desarrollo de Software, dedicando los últimos 04 años al uso exclusivo de la tecnología JAVA
- Propulsor del uso de la tecnología Java para el desarrollo de software WEB
- El Sr. López es Ingeniero de Sistemas titulado por la Universidad Privada del Norte – Trujillo - Perú