

Aplicaciones con Interfaz Gráfica de Usuario con NetBeans

Creación de un Proyecto

Para crear un programa con una interfaz de usuario gráfica en Java utilizando **NetBeans** lo primero que hay que hacer es crear un proyecto. Un proyecto nos permite administrar los archivos con el código fuente y compilado de una aplicación. Para crear un proyecto se sigue el siguiente procedimiento:

1. Ejecute el programa **NetBeans**. Al hacerlo aparecerá la ventana principal del programa como se ilustra en la figura 1.

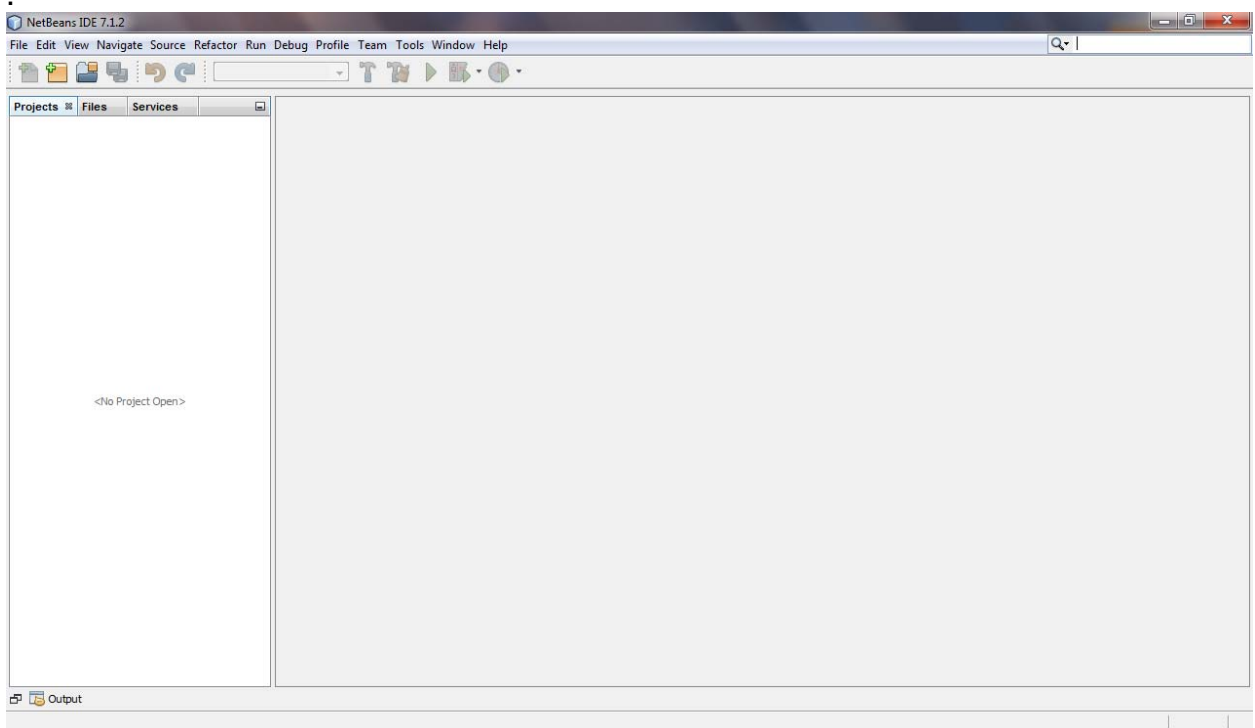


Figura 1

2. Del menú principal de NetBeans seleccione la opción **File/New Project ...** , presione las teclas **Ctrl+Mayúsculas+N** o haga clic en el icono **New Project** mostrado en la figura 2.

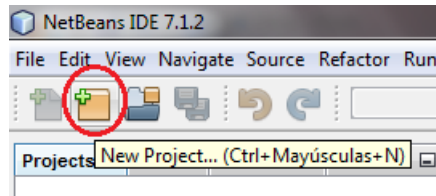


Figura 2

3. Aparecerá el primer cuadro de diálogo del asistente para crear un nuevo proyecto, figura 3. Seleccionaremos el tipo de proyecto que deseamos crear. Del recuadro **Categories:**, seleccione la opción **Java** y del recuadro **Projects:**, la opción **Java Application**. Luego presione el botón **Next>**.

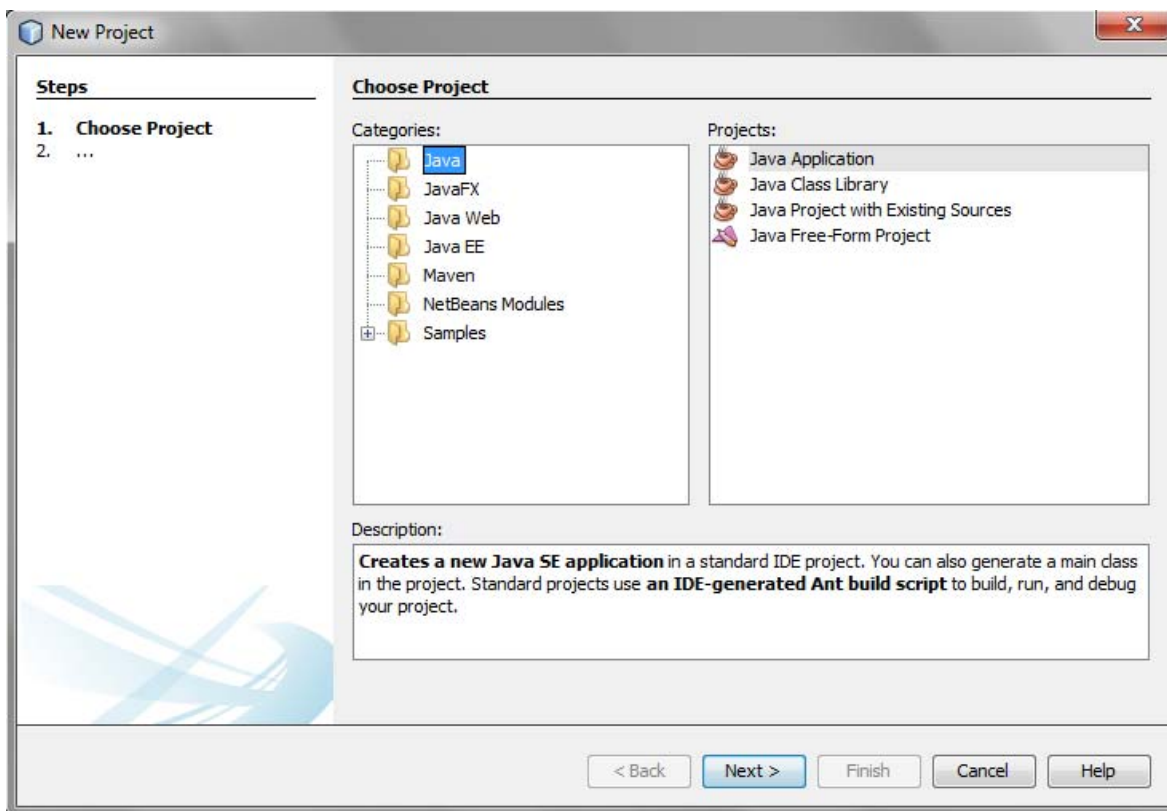


Figura 3

4. Aparecerá el segundo cuadro de diálogo del asistente para crear proyectos, figura 4. Aquí seleccionaremos el nombre y la ubicación del proyecto.
 - a) En el campo de texto **Project Name:** establezca el nombre del proyecto: Por ejemplo, “**amanteMusica**”.
 - b) En el campo de texto **Project Location:** establezca la carpeta donde se almacenará el proyecto. Aquí dejaremos el valor por ausencia. En el siguiente campo de texto **Project Location:**, aparece la ubicación de la carpeta en el que se almacenarán los archivos del proyecto.

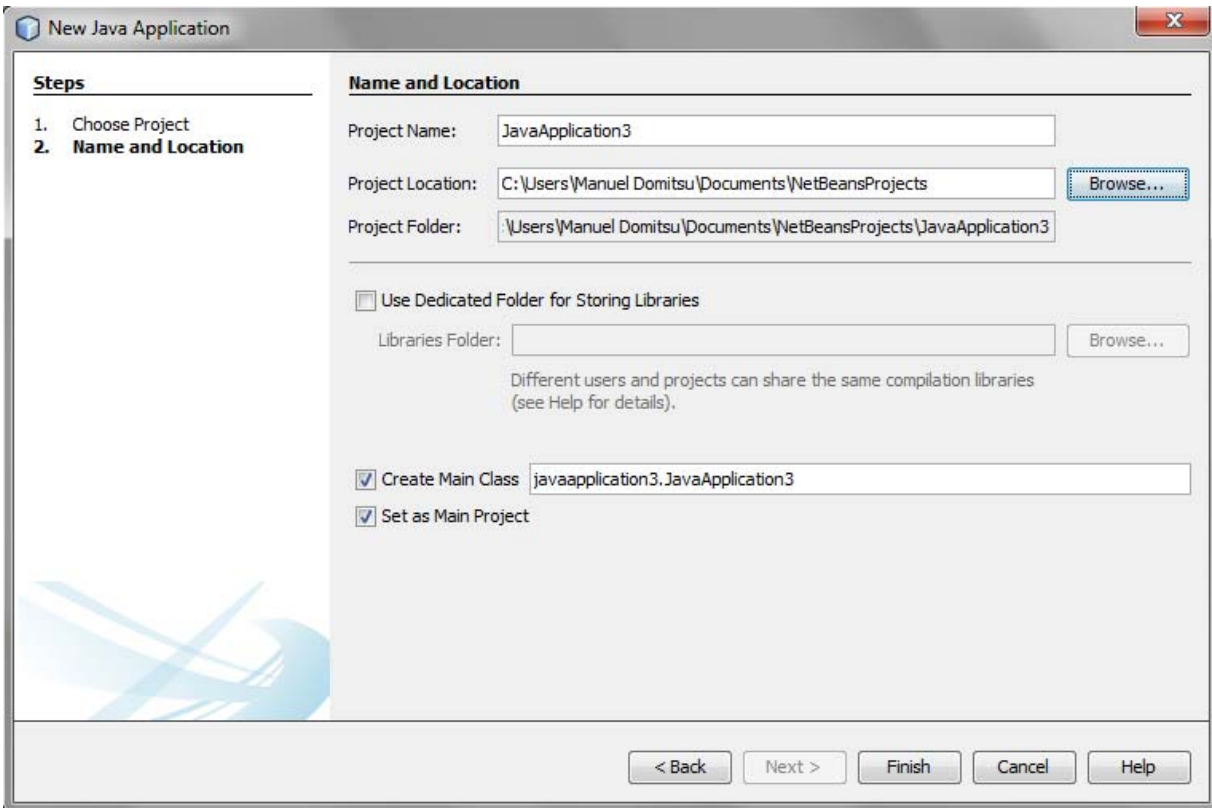


Figura 4

- c) Asegúrese que la casilla de verificación **Create Main Class** (Cree la clase principal, la clase con el método `main()`) esté deseleccionada. No se quiere que se genere automáticamente la clase principal. En lugar de ello, la clase con la primer ventana de la aplicación contendrá el método `main()`.
 - d) Asegúrese que la casilla de verificación: **Set as Main Project** (Haga que este proyecto sea el proyecto principal) esté seleccionada.
 - e) Presione el botón **Finish**.
5. Desaparecerá el asistente para crear un nuevo proyecto y aparecerá lo mostrado en la figura 5. Del lado izquierdo aparece el árbol de los proyectos, que en este momento sólo tiene el proyecto **amanteMusica**.
 6. Siguiendo el Tutorial: Programas de Consola en Java con NetBeans, agréguele al proyecto los archivos JAR de las bibliotecas que requiera el proyecto. Por ejemplo los archivos JAR `objetosServicio.jar`, `amanteMusicaObjNeg.jar`, `amanteMusicaInterfaces.jar` y `amanteMusicaPersistenciaListas.jar`.

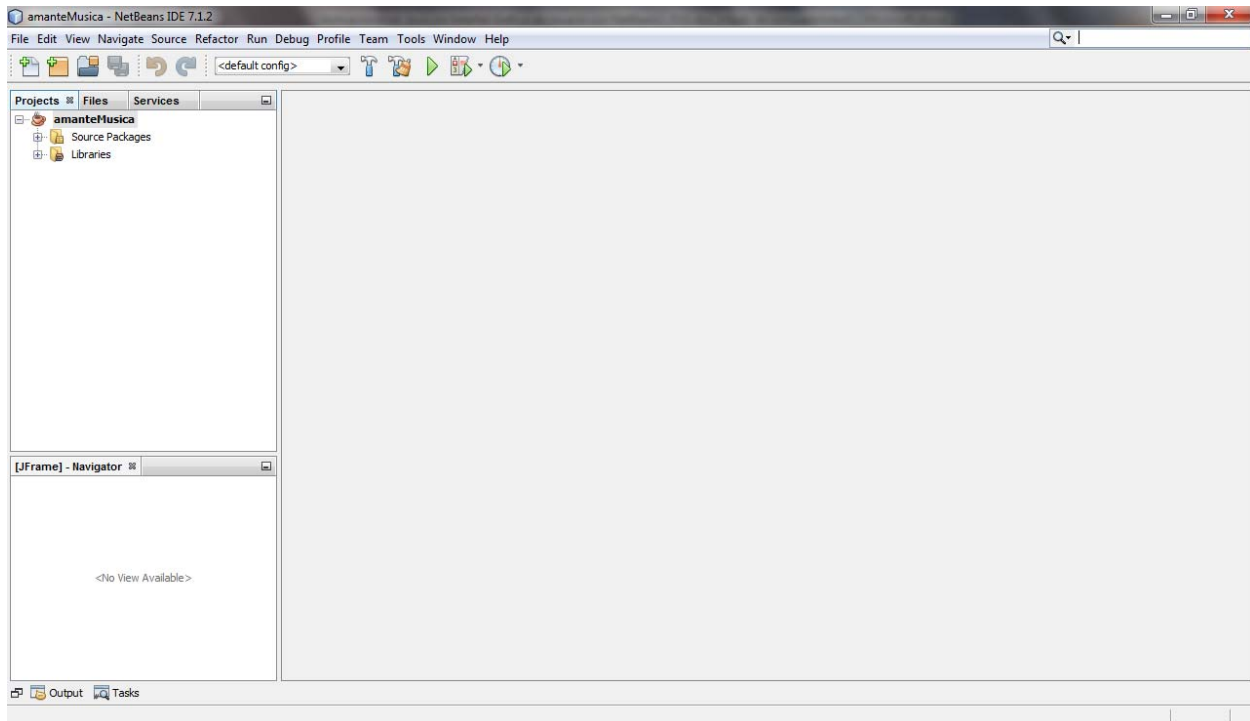


Figura 5

Creación de la Clase con la Ventana Principal de la Aplicación

Para crear la clase con la ventana principal de la aplicación se sigue el siguiente procedimiento:

1. De la barra de menú de NetBeans, seleccione la opción **Files/New File**, presione las teclas **Ctrl+ N** o haga clic en el icono **New File**, como se muestra en la figura 6:

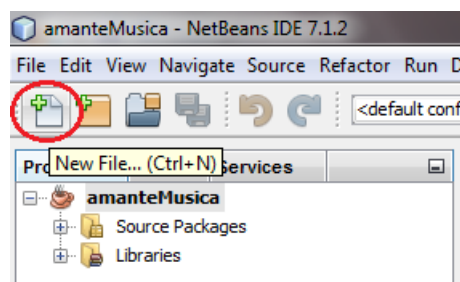


Figura 6

2. Aparecerá el primer cuadro de diálogo del asistente para crear una clase, figura 7. En el recuadro **Categories:** seleccione el nodo **Swing GUI Forms** y del recuadro **File Types** seleccionaremos el tipo **Application Sample Form** que nos creará el esqueleto de la ventana principal de la aplicación, incluyendo una

barra de menú de muestra, que posteriormente modificaremos para ajustarlo a nuestra aplicación. Presione el botón **Next**.

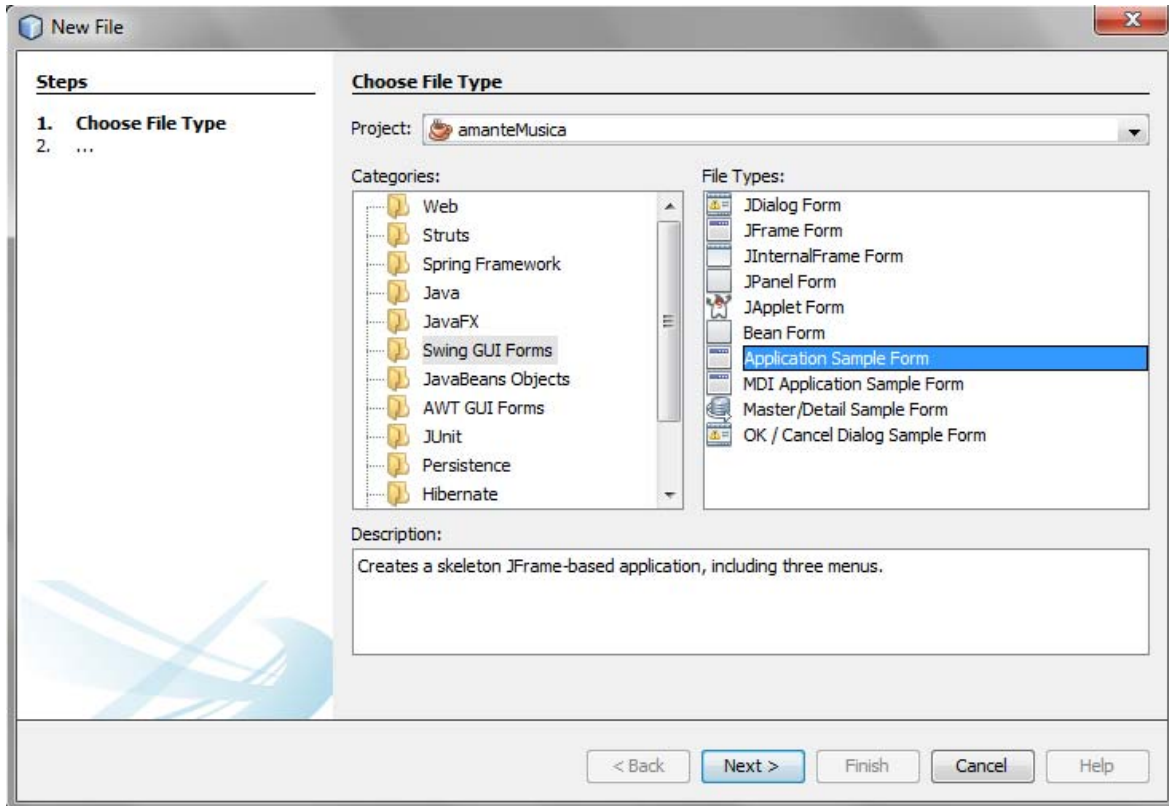


Figura 7

3. Aparecerá el segundo cuadro de diálogo del asistente para crear clases, mostrada en la figura 8. En esta ventana seleccionaremos el nombre y la ubicación de la clase de la ventana principal de una aplicación.
 - a) En el cuadro de texto **Class Name**: establezca el nombre de la clase. Por ejemplo, "**FrmAmanteMusica**".
 - b) En el cuadro de texto **Package**: establezca el paquete donde estará la clase. Por ejemplo, "**interfazUsuario**".
 - c) Presione el botón **Finish**.
4. Desaparecerá el asistente para crear una nueva clase y aparecerá lo mostrado en la Figura 9.

En el lado superior izquierdo, en el panel **Projects** aparece el árbol de los proyectos, con el proyecto **amanteMusica** y la clase **FrmAmanteMusica**. En el lado inferior izquierdo, en el panel Navegador se muestra un árbol con los elementos de la clase activa en el panel de edición, **FrmAmanteMusica**, en este caso.

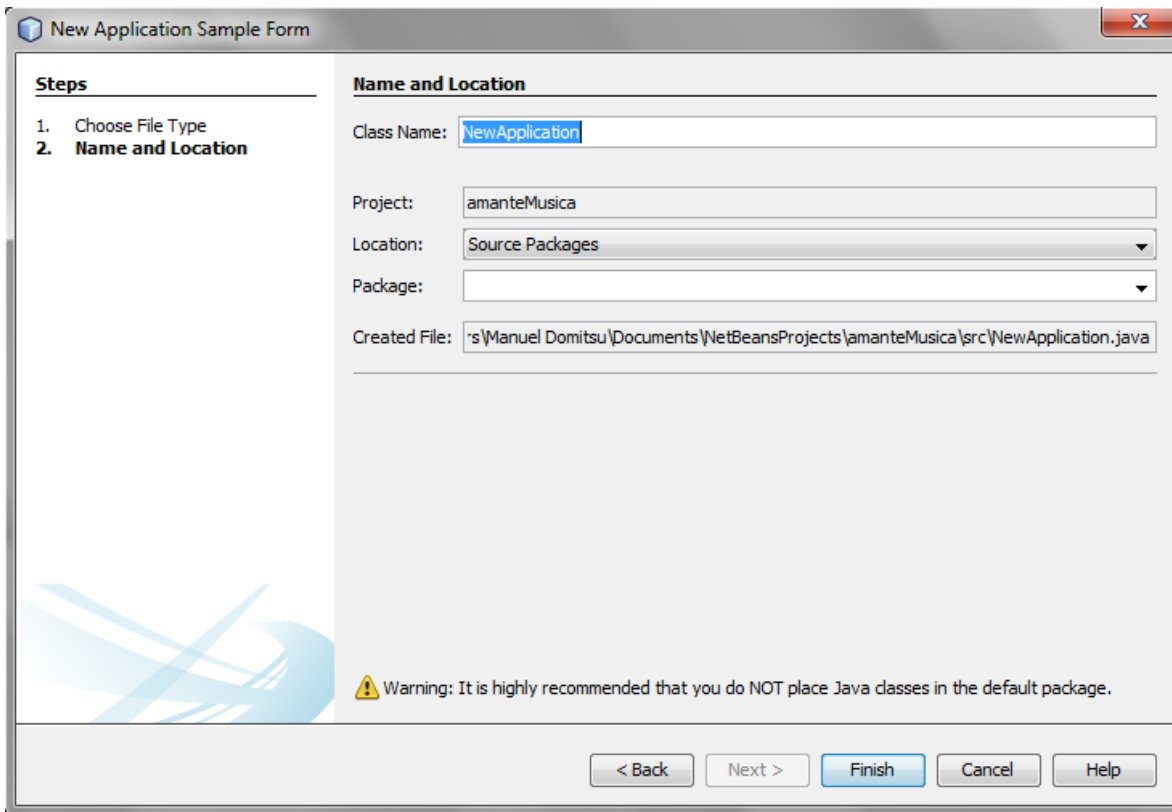


Figura 8

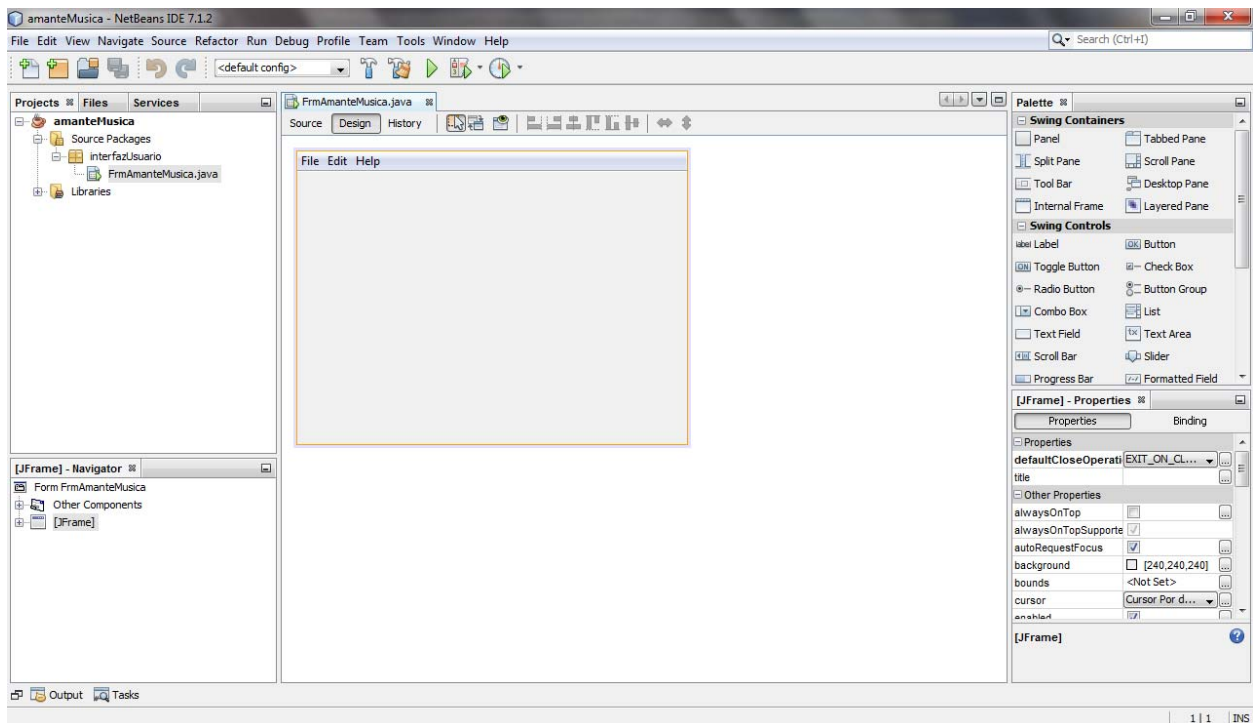


Figura 9

En el lado superior derecho, se encuentra el panel **Palette** con la paleta de componentes que se pueden agregar a las ventanas, cuadros de diálogo o paneles de una aplicación. En el lado inferior derecho aparece el panel **Properties** con el editor de propiedades en la que podemos editar los valores de las propiedades de la componente seleccionada en el panel de edición o en el panel de inspección.

En el centro se encuentra el panel de edición. En este momento se encuentran en él la clase de la ventana principal, **FrmAmanteMusica**. Cuando la clase desplegada en el panel de edición corresponde a una ventana o a un cuadro de diálogo, NetBeans nos presenta dos vistas de la clase: La **Vista de Diseño** que nos permite en forma gráfica agregar componentes a la ventana o cuadro de diálogo y editar sus propiedades y la **Vista de Código Fuente** que nos permite hacer lo mismo editando el código fuente de la clase. Ambas vistas están sincronizadas. Los cambios hechos en una se ven reflejadas en la otra. Podemos cambiar de una vista a otra mediante los selectores que se encuentran en la parte superior del **panel de edición**, figura 10.

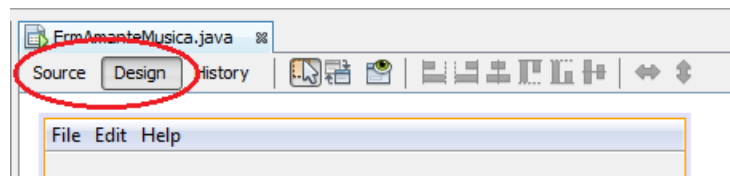


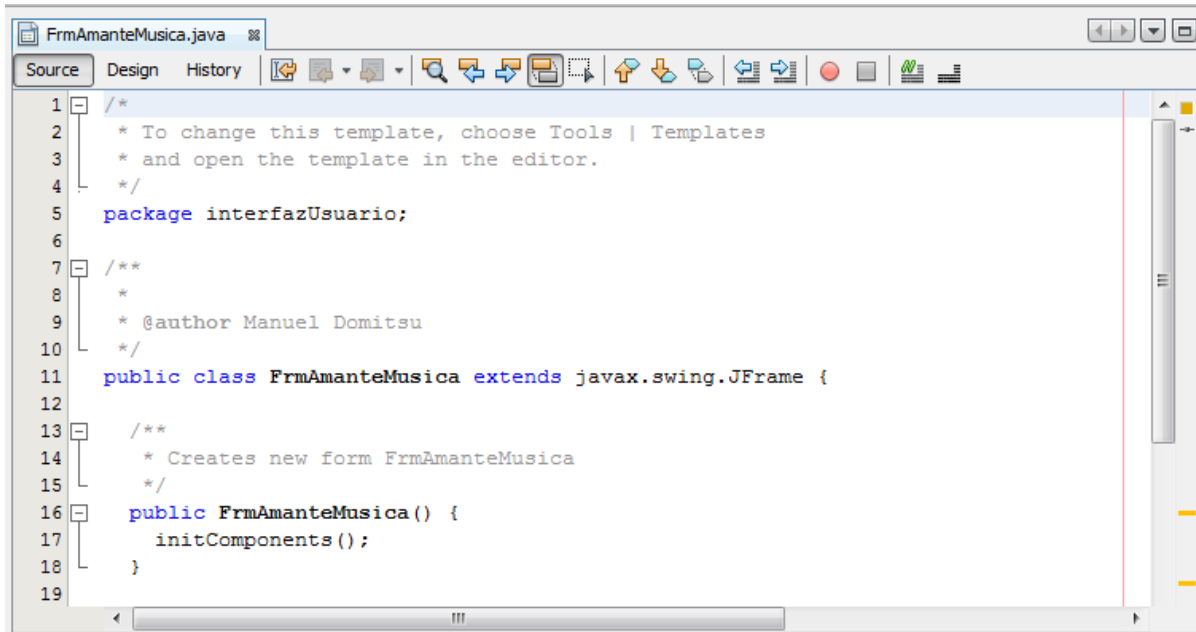
Figura 10

En la figura 11 se muestra un fragmento de la **Vista de Código Fuente** para la clase `FrmAmanteMusica.java`. En esta imagen podemos ver el encabezado de la clase y un constructor que invoca al método `initComponents()` que se encarga de inicializar los componentes de la interfaz gráfica de la ventana.

Al final del código de la clase se encuentran las declaraciones de los atributos de la clase, figura 12. Las declaraciones que aparecen en fondo azul fueron generadas por NetBeans para las componentes de la interfaz gráfica y no pueden ser modificadas. Sin embargo podemos agregar declaraciones en el renglón en blanco al final de las declaraciones.

Otro de los fragmentos de código generados por NetBeans es el método `initComponents()` que se encarga de inicializar los componentes de la interfaz gráfica de la ventana. Inicialmente el código de este método está oculto y sólo se muestra un botón con el símbolo **[+]** en el margen izquierdo de la ventana de edición para hacerlo visible, figura 13.

Al hacer clic en el botón **[+]** el código se hará visible y el botón cambiará a **[-]**, figura 14. Podemos ocultar el código de nuevo haciendo clic en el botón **[-]**. Note que ese aparece con fondo azul y por lo tanto no puede ser modificado.

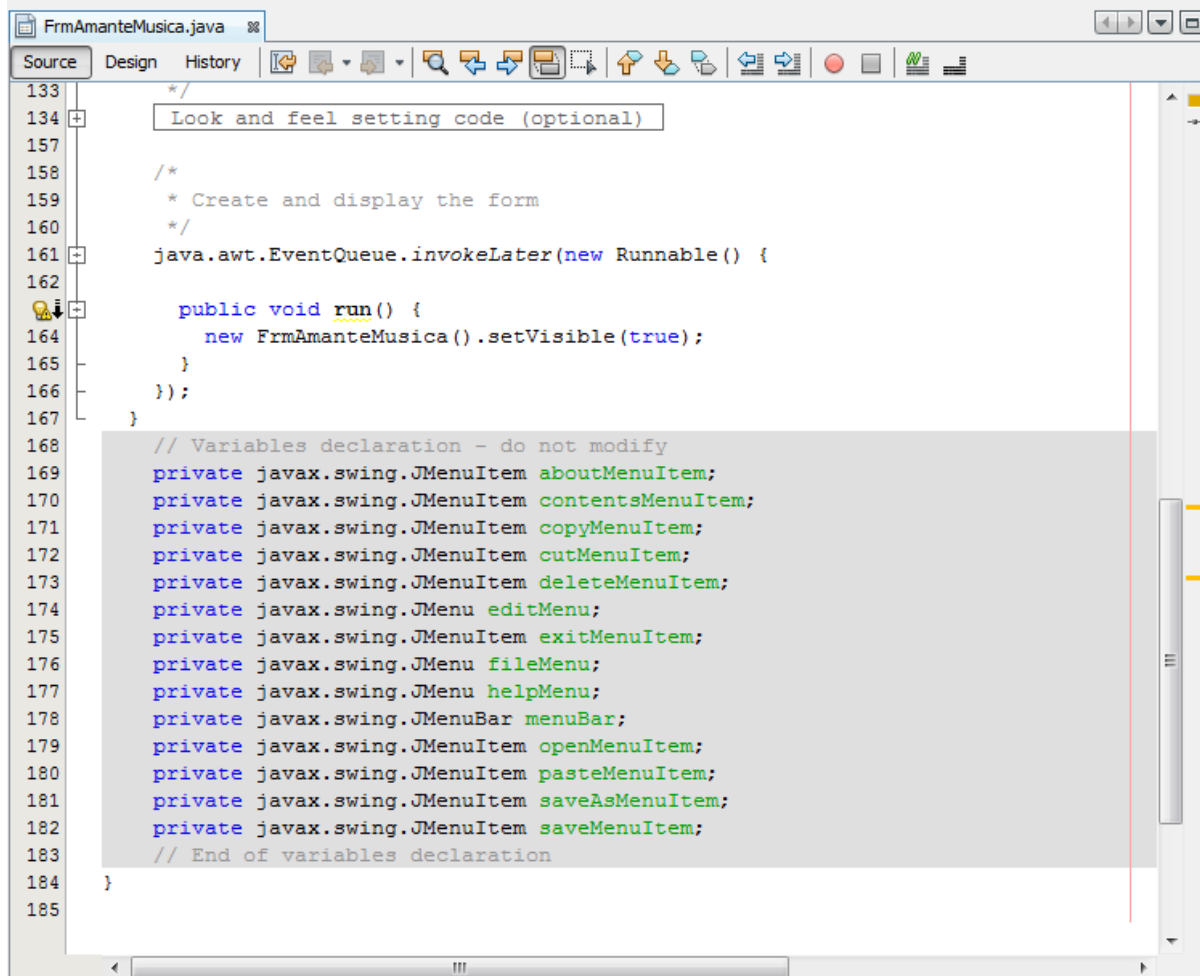


```

1  /*
2  * To change this template, choose Tools | Templates
3  * and open the template in the editor.
4  */
5  package interfazUsuario;
6
7  /**
8   *
9   * @author Manuel Domitsu
10  */
11  public class FrmAmanteMusica extends javax.swing.JFrame {
12
13  /**
14   * Creates new form FrmAmanteMusica
15   */
16  public FrmAmanteMusica() {
17      initComponents();
18  }
19

```

Figura 11



```

133  /*
134  Look and feel setting code (optional)
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149  /*
150  * Create and display the form
151  */
152  java.awt.EventQueue.invokeLater(new Runnable() {
153
154  public void run() {
155      new FrmAmanteMusica().setVisible(true);
156  }
157  });
158
159  }
160
161  // Variables declaration - do not modify
162  private javax.swing.JMenuItem aboutMenuItem;
163  private javax.swing.JMenuItem contentsMenuItem;
164  private javax.swing.JMenuItem copyMenuItem;
165  private javax.swing.JMenuItem cutMenuItem;
166  private javax.swing.JMenuItem deleteMenuItem;
167  private javax.swing.JMenu editMenu;
168  private javax.swing.JMenuItem exitMenuItem;
169  private javax.swing.JMenu fileMenu;
170  private javax.swing.JMenu helpMenu;
171  private javax.swing.JMenuBar menuBar;
172  private javax.swing.JMenuItem openMenuItem;
173  private javax.swing.JMenuItem pasteMenuItem;
174  private javax.swing.JMenuItem saveAsMenuItem;
175  private javax.swing.JMenuItem saveMenuItem;
176
177  // End of variables declaration
178
179  }
180
181
182
183
184
185

```

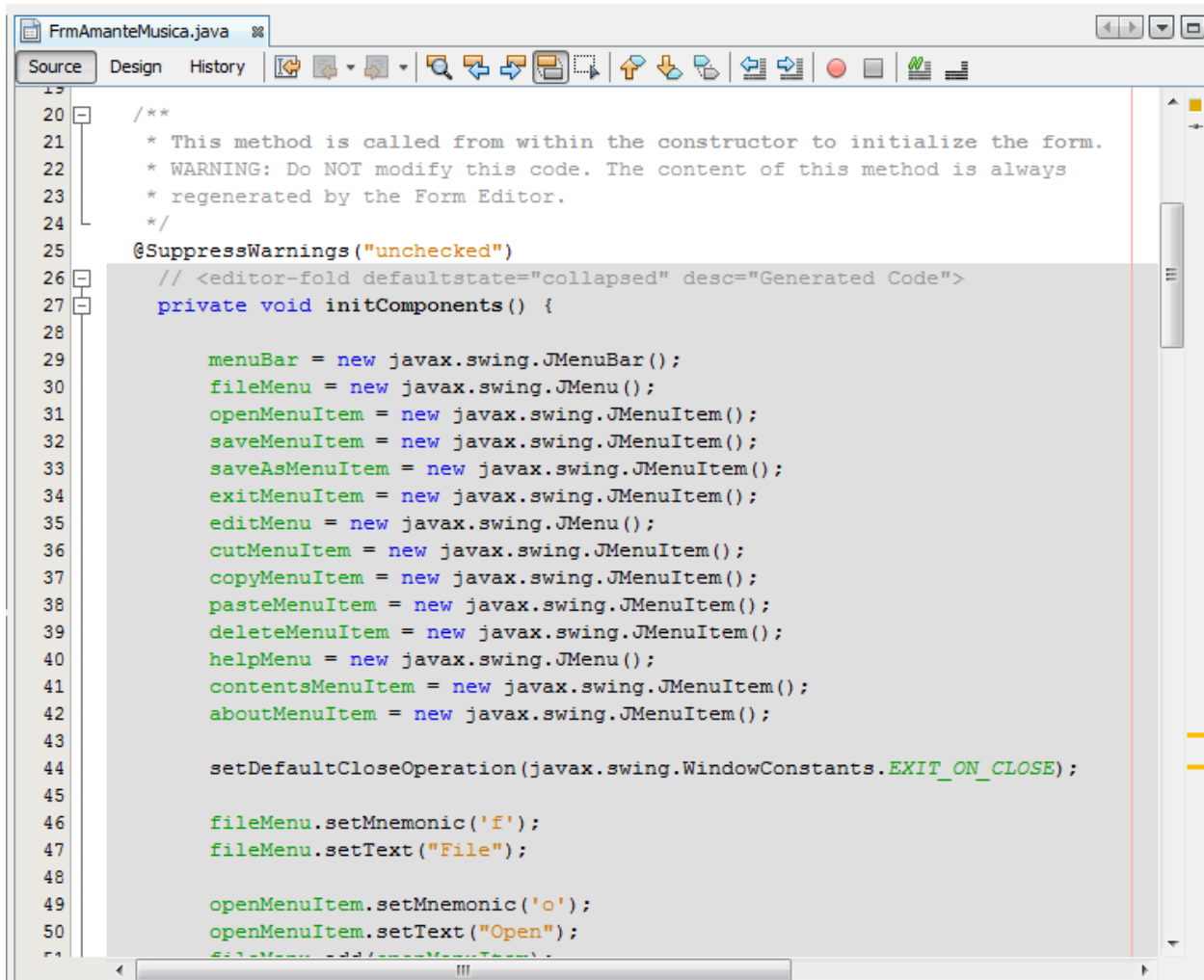
Figura 12


```

19
20 /**
21  * This method is called from within the constructor to initialize the form.
22  * WARNING: Do NOT modify this code. The content of this method is always
23  * regenerated by the Form Editor.
24  */
25  @SuppressWarnings("unchecked")
26  // Generated Code
122

```

Figura 13



```

FrmAmanteMusica.java
Source Design History
19
20 /**
21  * This method is called from within the constructor to initialize the form.
22  * WARNING: Do NOT modify this code. The content of this method is always
23  * regenerated by the Form Editor.
24  */
25  @SuppressWarnings("unchecked")
26  // <editor-fold defaultstate="collapsed" desc="Generated Code">
27  private void initComponents() {
28
29      menuBar = new javax.swing.JMenuBar();
30      fileMenu = new javax.swing.JMenu();
31      openMenuItem = new javax.swing.JMenuItem();
32      saveMenuItem = new javax.swing.JMenuItem();
33      saveAsMenuItem = new javax.swing.JMenuItem();
34      exitMenuItem = new javax.swing.JMenuItem();
35      editMenu = new javax.swing.JMenu();
36      cutMenuItem = new javax.swing.JMenuItem();
37      copyMenuItem = new javax.swing.JMenuItem();
38      pasteMenuItem = new javax.swing.JMenuItem();
39      deleteMenuItem = new javax.swing.JMenuItem();
40      helpMenu = new javax.swing.JMenu();
41      contentsMenuItem = new javax.swing.JMenuItem();
42      aboutMenuItem = new javax.swing.JMenuItem();
43
44      setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
45
46      fileMenu.setMnemonic('f');
47      fileMenu.setText("File");
48
49      openMenuItem.setMnemonic('o');
50      openMenuItem.setText("Open");
51      fileMenu.add(openMenuItem);

```

Figura 14

Por último, la figura 15 muestra otro fragmento de código de la clase en la que se muestra el método oyente `exitMenuItemActionPerformed()` que contiene el código que se ejecutará cuando se haga clic en la opción **Exit** del menú descendente **File** de la barra de menú. También contiene el código del método `main()`.

Guarde la clase seleccionando del menú principal la opción **File/Save**, presione las teclas **Ctrl+S** o haga clic en el icono **Save All**, mostrado en la figura 16.



Figura 15

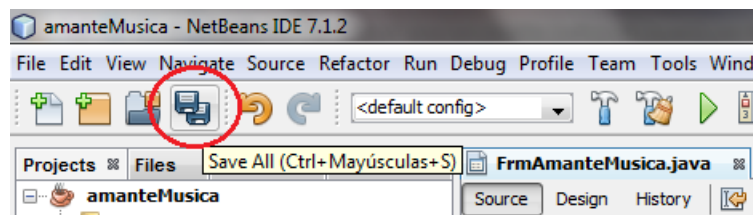


Figura 16

Ubicación de los Archivos de un Proyecto

La figura 17, muestra las carpetas y archivos generados al crear el proyecto. Todos los archivos del proyecto se encuentran en una carpeta con el nombre del proyecto, **amanteMusica** en este caso. El código fuente del proyecto se almacena en la carpeta **src** dentro de la carpeta del proyecto. El código de cada clase se guarda dentro de una carpeta con el mismo nombre del paquete establecido para la clase.

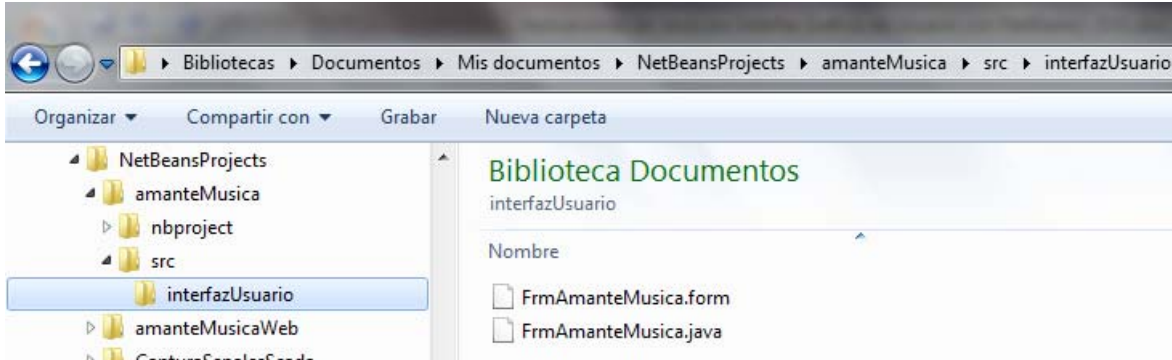


Figura 17

Compilación del Proyecto

Para compilar todas las clases de un proyecto seleccione del menú principal la opción **Run/Build Main Project**, presione la tecla **F11** o presione el icono **Build Main Project**, mostrado en la figura 18.

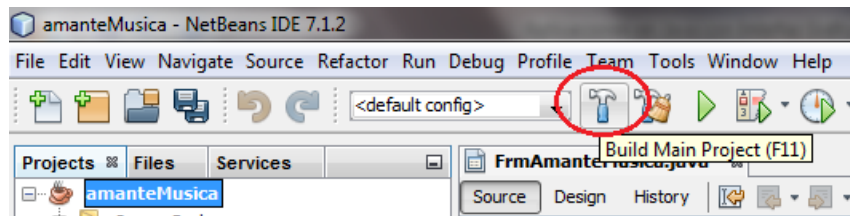


Figura 18

Durante la compilación, NetBeans muestra los mensajes resultantes del proceso, como se muestra en la figura 19.

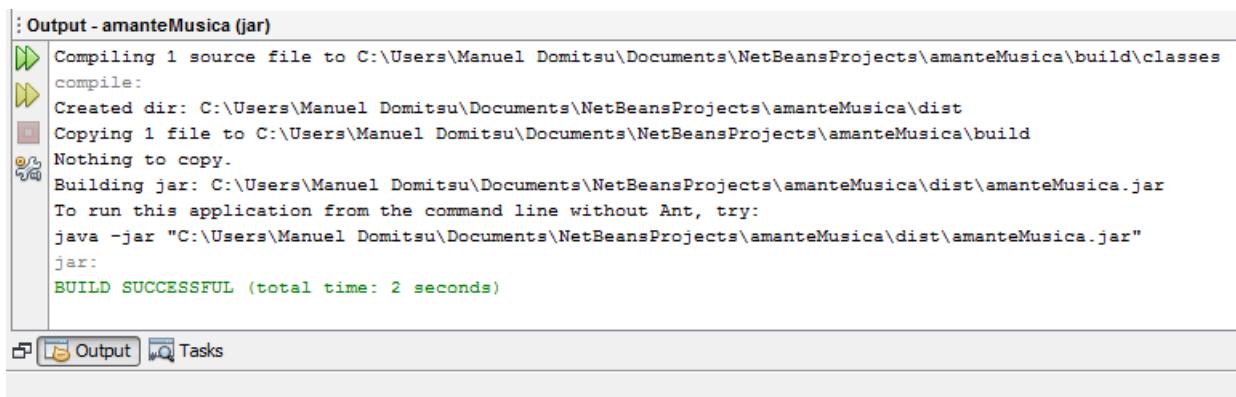


Figura 19

Ubicación de los Archivos con el Código “byteCode” del Proyecto

La figura 20, muestra los directorios y archivos generados al compilar el proyecto. Todos los archivos con el código bytecode resultado de la compilación del proyecto se encuentran en la carpeta `build/classes` dentro de la carpeta del proyecto, `amanteMusica` en este caso. El código bytecode de una clase se guarda en un archivo que tiene el mismo nombre de la clase y con la extensión `.class` dentro de una carpeta con el mismo nombre del paquete establecido para la clase. En este caso, el código fuente de la clase con la ventana principal de la aplicación se almacena en el archivo `FrmAmanteMusica.class` dentro de la carpeta `interfazUsuario`.

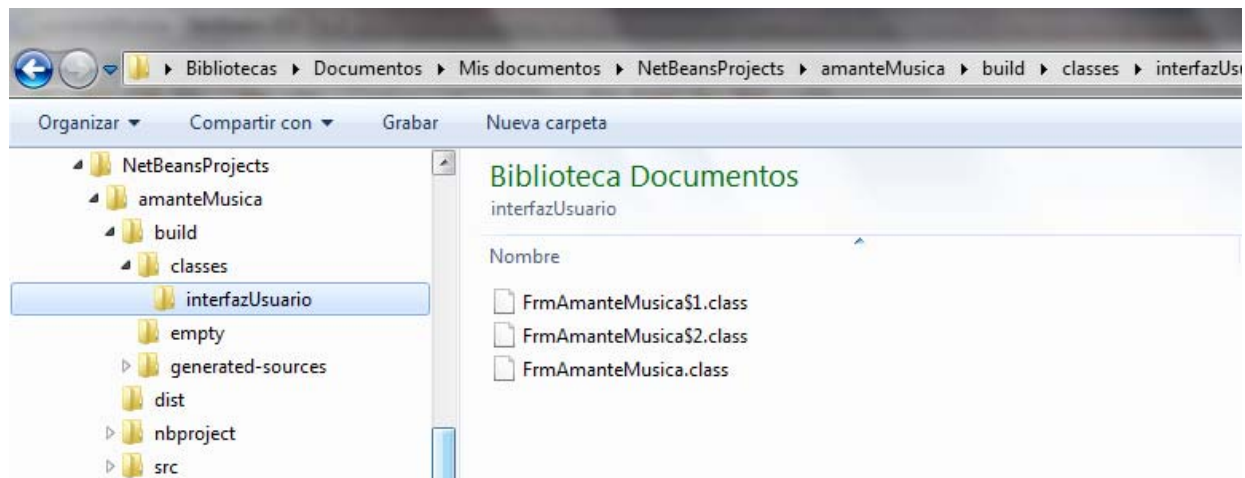


Figura 20

El archivo JAR con los archivos con el código bytecode empacados se encuentran en un archivo con el nombre del proyecto y la extensión `.jar` dentro del directorio `dist` dentro del directorio del proyecto, figura 21.

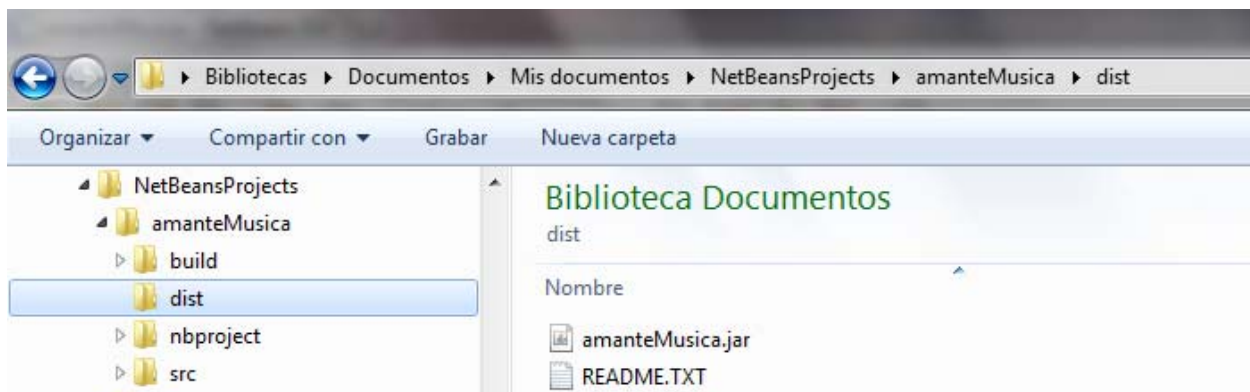


Figura 21

Ejecución de una Aplicación

1. Para ejecutar la aplicación dentro de NetBeans, seleccione del menú principal la opción **Run/Run Main Project**, presione la tecla **F6** o haga clic en el icono **Run Main Project**, mostrado en la figura 22.

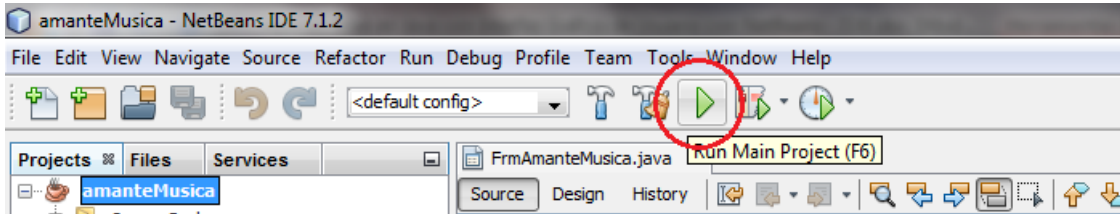


Figura 22

2. NetBeans nos presenta el cuadro de diálogo mostrado en la figura 23 en el que nos indica que no hemos establecido el nombre de la clase principal (la que contiene el método `main()`), y nos muestra una lista de las clases del proyecto para que seleccionemos la indicada. En este caso sólo hay una, la seleccionamos y hacemos clic en el botón **OK**.

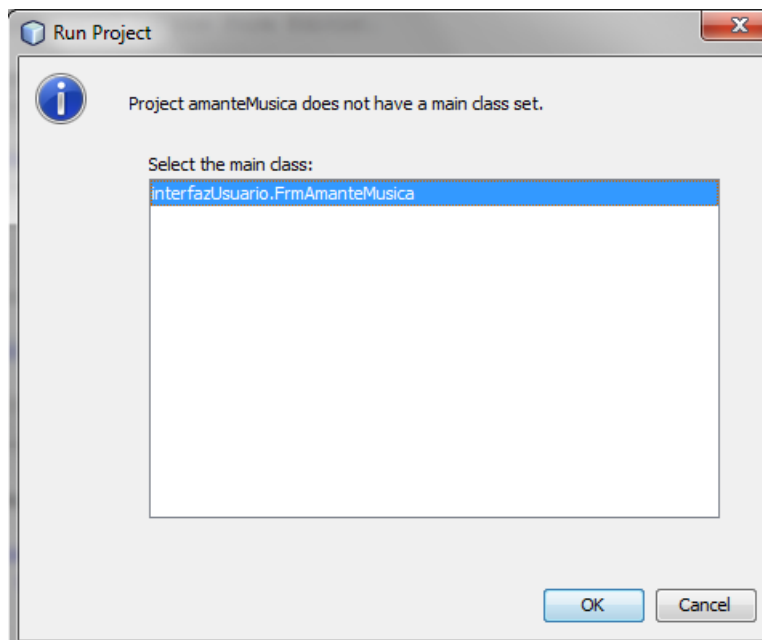


Figura 23

3. NetBeans ejecutará la aplicación desplegando su ventana principal como se muestra en la figura 24.

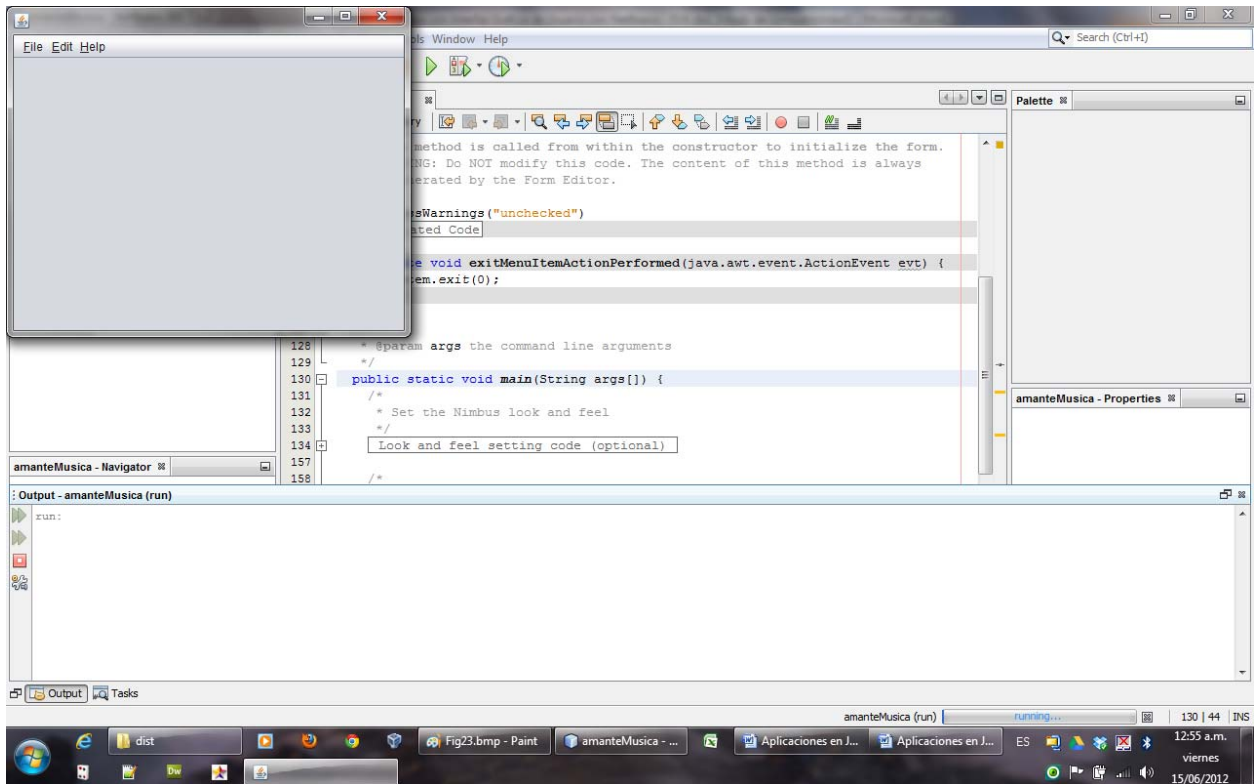


Figura 24

4. Para terminar la ejecución de la aplicación seleccione la opción **File/Exit** de la aplicación o presione el botón **Cerrar**.

Establecer el Título de la Aplicación en la Barra de Título de la Ventana de la Aplicación

Note en la figura 24, que la barra de título de la ventana de la aplicación (la barra donde están los iconos para minimizar, maximizar y cerrar la aplicación) no contiene el título de la aplicación. Para establecer el título de la aplicación en la barra de título en la ventana de la aplicación seguimos el siguiente procedimiento.

1. En la Ventana del Editor de Código seleccione la Vista de Diseño, figura 10.
2. En el Editor de Propiedades de las Componentes aparecen las propiedades de la clase que representa la ventana de la aplicación, **JFrame**, figura 25. Establezca el valor de la propiedad **title** al valor de título deseado, **Amante Música**, en este caso. La propiedad **title** es el texto que aparecerá en la barra de título en la ventana de la aplicación.
3. Vuelva a ejecutar la aplicación. En la barra de título en la ventana de la aplicación aparecerá el título como se muestra en la figura 26.

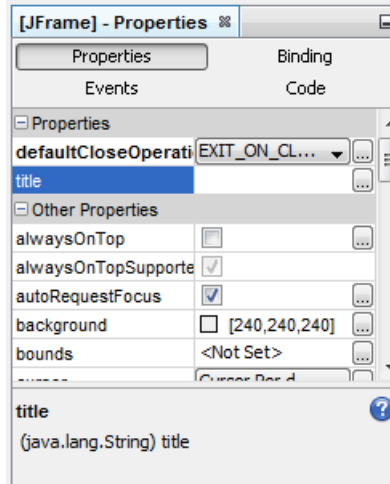


Figura 25

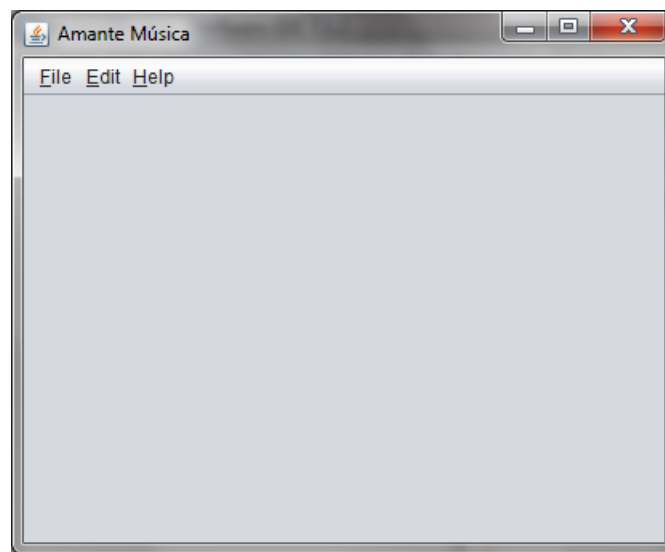


Figura 26

Centrado de la Ventana de la Aplicación en la Pantalla

Note en la figura 24, que la ventana de la aplicación aparece en la esquina superior izquierda de la pantalla. Para centrar la ventana de la aplicación en la pantalla seguimos el siguiente procedimiento.

1. En la Ventana del Editor de Código seleccione la Vista de Código Fuente, figura 11.
2. Localice el constructor de la clase: `public FrmAmanteMusica()` e inmediatamente después de su código agregue el siguiente método para centrar la ventana de la aplicación en la pantalla:


```
/**
 * Este método centra la ventana de la aplicación sobre la pantalla
 */
private void centraVentana() {
    //Obtiene el tamaño de la pantalla
    Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();

    // Obtiene el tamaño de la ventana de la aplicación
    Dimension frameSize = getSize();

    // Se asegura que el tamaño de la ventana de la aplicación
    // no exceda el tamaño de la pantalla
    if (frameSize.height > screenSize.height) {
        frameSize.height = screenSize.height;
    }
    if (frameSize.width > screenSize.width) {
        frameSize.width = screenSize.width;
    }

    // Centra la ventana de la aplicación sobre la pantalla
    setLocation((screenSize.width - frameSize.width) / 2,
                (screenSize.height - frameSize.height) / 2);
}
```

3. El código del método requiere agregarle a la clase las siguientes directivas import:

```
import java.awt.Dimension;
import java.awt.Toolkit;
```

4. Agréguele al constructor de la clase: `public void FrmAmanteMusica()` la llamada al método `centraVentana()`, como se muestra en el siguiente fragmento de código:

```
public FrmAmanteMusica() {
    initComponents();
    centraVentana();
}
```

5. Vuelva a ejecutar la aplicación. La ventana de la aplicación aparecerá centrada en la pantalla como se muestra en la figura 27

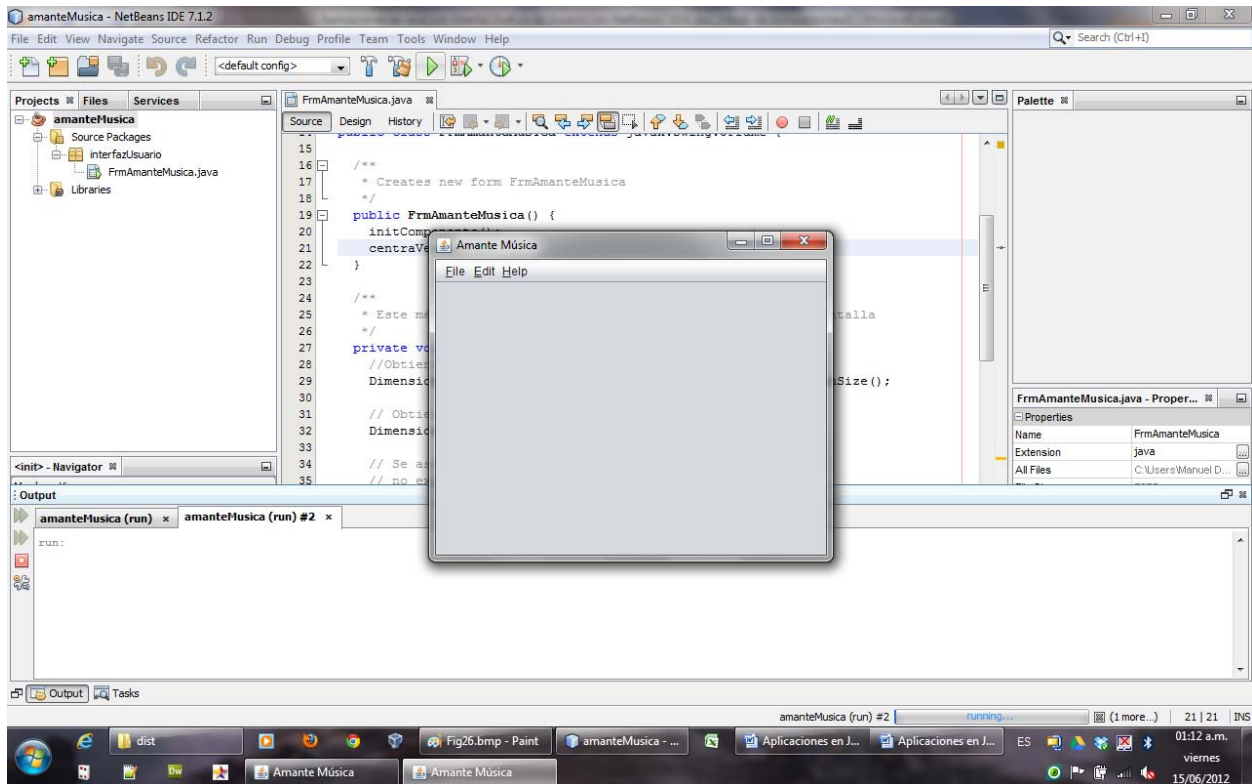


Figura 27

Edición de la Barra de Menú, Menús y Opciones de Menú

Para ajustar la ventana principal generada por NetBeans para la aplicación **amanteMusica**, modificaremos la barra de menú de muestra, sus menús y opciones de menú.

Edición de las Opciones de la Barra de Menú

Primero se modificarán las opciones de la barra de menús. Para cambiar el título de una opción de la barra de menú y el nombre de la variable del componente que lo representa, seguiremos el siguiente procedimiento:

1. La mayor parte de las componentes de la barra de menú no pueden ser manipulados directamente en la Vista de Diseño. Para acceder a una componente de la barra de menú debemos seleccionar el nodo que representa a esa componente en el árbol de componentes en la panel Navegador, figura 28. En ese árbol de componentes, el nodo **JFrame** representa a la ventana de la aplicación.

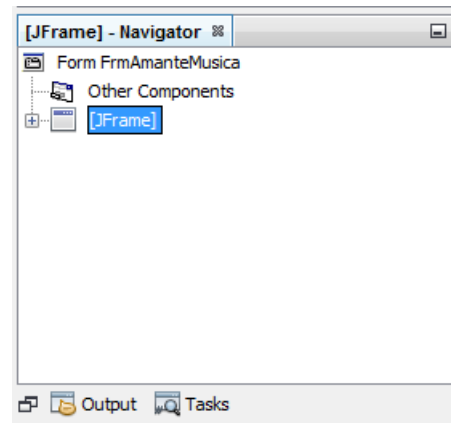


Figura 28

- Si expandimos el nodo **JFrame**, haciendo clic sobre el icono con el signo **+** del nodo, aparecerán los nodos que representan las componentes que contiene la ventana. En este caso sólo contiene la barra de menú, el nodo **menuBar** del tipo **JMenuBar**, figura 29.

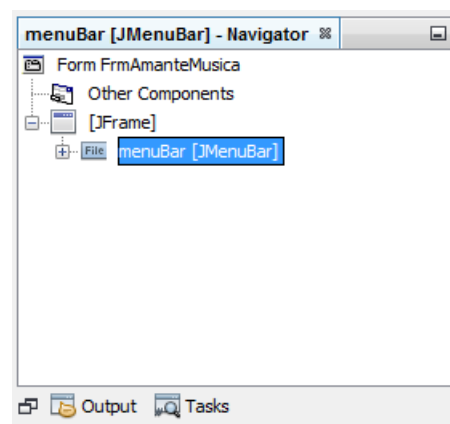


Figura 29

- Si expandimos el nodo **menuBar**, tendremos los nodos que representan los elementos de la barra de menú que pueden ser opciones de menú (del tipo **JMenuItem**) o menús descendentes (del tipo **JMenu**). En este caso sólo contiene menús descendentes, figura 30.
- Para modificar un elemento de la barra de menú, haga clic en el nodo que representa ese elemento, en este caso en el menú descendente **fileMenu**, del tipo **Jmenu**. Al hacer esto, en el Editor de Propiedades aparecen las propiedades del elemento de la barra de menú **File**, figura 31. Cambie el valor de la propiedad **text** de **File** a **Catálogos**. La propiedad **text** es el título de un elemento de la barra de menú, en este caso un menú descendente.

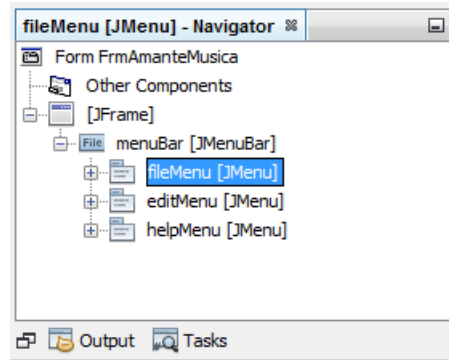


Figura 30

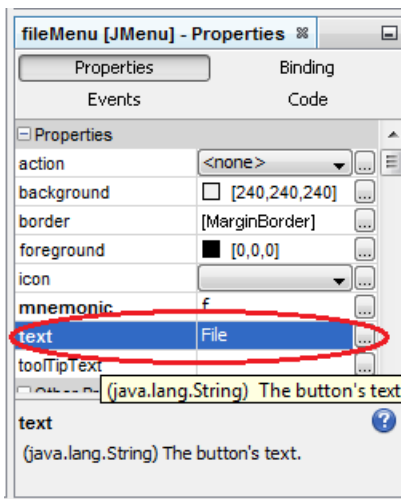


Figura 31

- Para cambiar el nombre de la variable del tipo `JMenu` que representa al elemento **File** hacemos clic en el selector **Code** del Editor de Propiedades, figura 32.

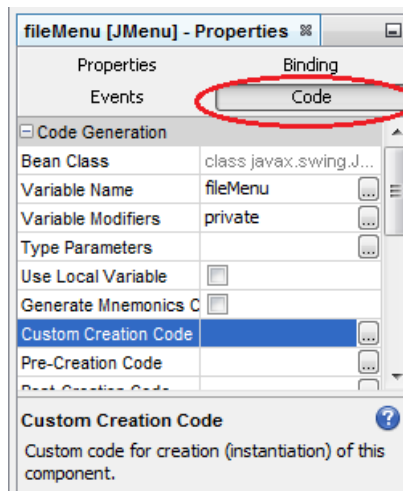


Figura 32

6. Cambie el nombre de la variable del tipo `JMenu` de `fileMenu` a `menuCatalogos`, como se muestra en la figura 33.

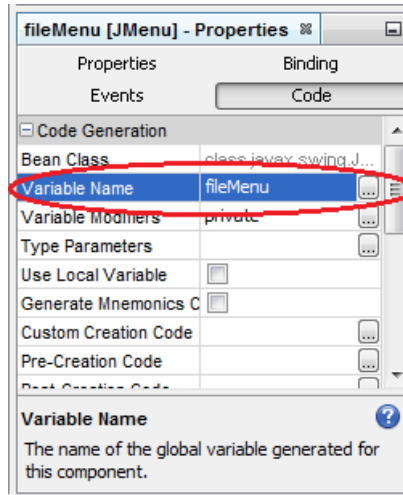


Figura 33

7. El cambio del nombre de la variable del componente se ve reflejada en el nodo que representa ese elemento, figura 34.

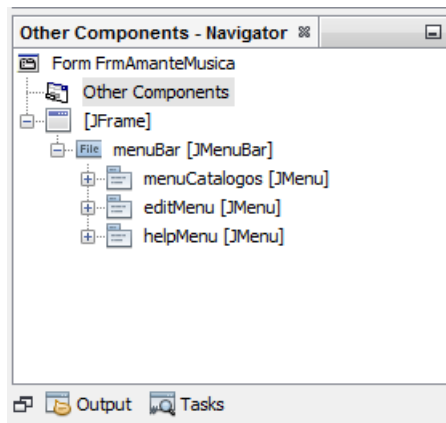


Figura 34

1. Repita los pasos 4 a 6 para cambiar el título de la opción **Edit** de la barra de menús a **Consultas** y el nombre de la variable del componente de `editMenu` a `menuConsultas`.
2. Repita los pasos 4 a 6 para cambiar el título de la opción **Help** de la barra de menús a **Ayuda** y el nombre de la variable del componente de `helpMenu` a `menuAyuda`.
3. Podemos visualizar los cambios realizados a la barra de menús de la aplicación, haciendo clic en el icono **Preview Design** que se encuentra encima de la Ventana de Edición del Código, figura 35.

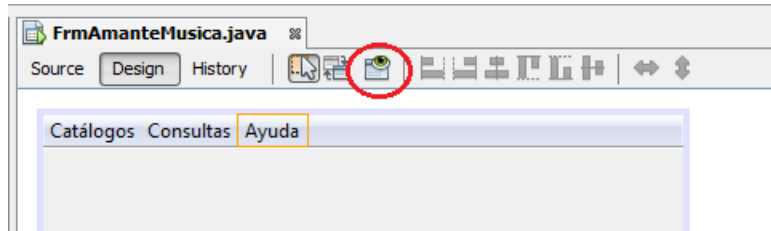


Figura 35

- Al hacerlo NetBeans desplegará una vista previa de la ventana de la aplicación en la que podemos ver los cambios realizados a la barra de menús, figura 36.

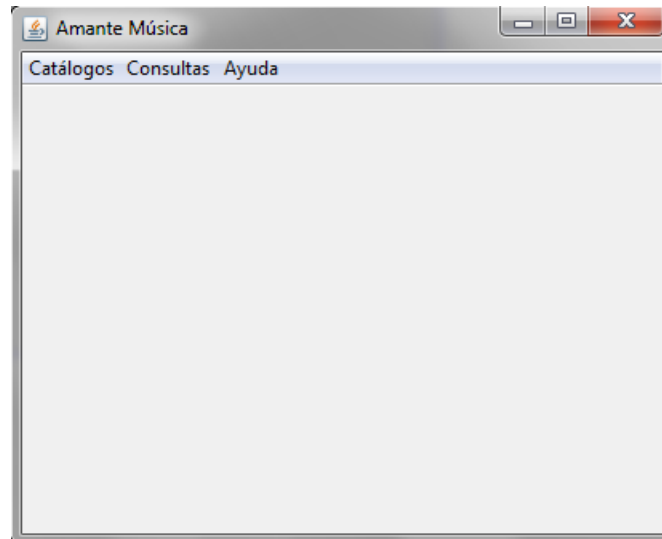


Figura 36

Borrado de Opciones de Menú

Ahora se borrarán algunas de las opciones de menú de los menús descendentes de la barra de menú. Para ello se sigue el siguiente procedimiento:

- Para acceder a un elemento de un menú debemos expandir el nodo que representa el menú y poder visualizar sus elementos. Estos elementos pueden ser opciones de menú (del tipo **JMenuItem**) o menús anidados (del tipo **JMenu**). En este caso si expandimos el nodo **menuCatalogos**, tendremos los nodos que representan los elementos de ese menú descendente, que sólo son opciones de menú, figura 37.
- Para eliminar una opción de un menú se hace clic sobre el nodo que representa a esa opción y luego presionamos la tecla **Suprimir**. Una alternativa a ese procedimiento es hacer clic con el botón derecho en el nodo y seleccionar la opción **Delete**, como se muestra la figura 38.

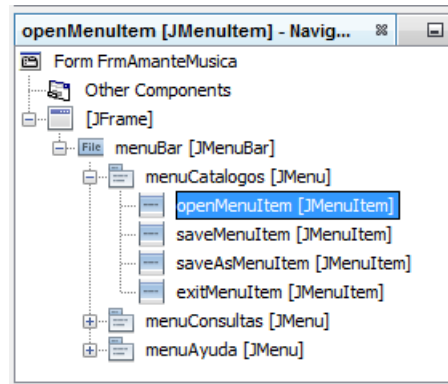


Figura 37

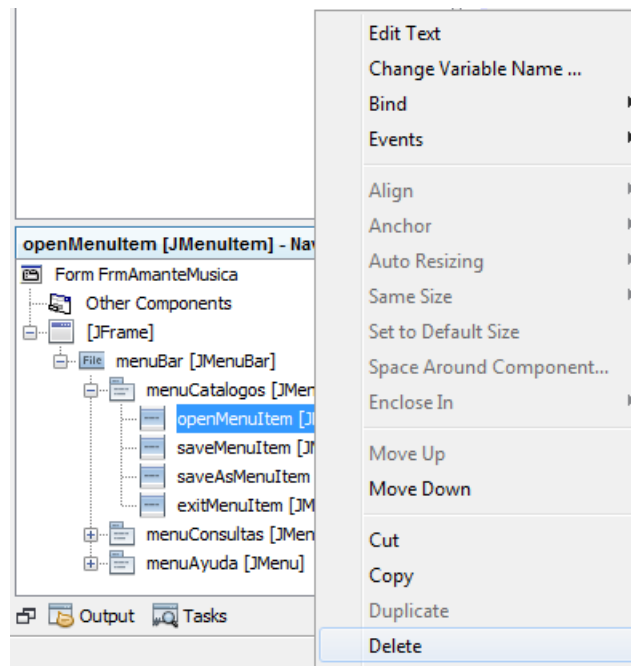


Figura 38

3. Ejecute el paso 2 para eliminar las opciones de menú del menú **menuCatalogos: openMenuItem, saveMenuItem, saveAsMenuItem**, dejando la opción **exitMenuItem**.
4. Ejecute los pasos 1 y 2 para eliminar las opciones de menú del menú **menuConsultas: cutMenuItem, copyMenuItem, pasteAsMenuItem y deleteAsMenuItem**.

Edición de las Opciones de un Menú Descendente

A continuación se modificarán las opciones de menú de los menús descendentes de la barra de menús. Para cambiar el título de una opción de un menú y el nombre de la variable del componente que la representa, seguiremos el siguiente procedimiento:

1. En el panel Navegador, haga clic en el nodo que representa la opción del menú que desee modificar, en este caso en la opción **exitMenuItem**, como se muestra en la figura 39.

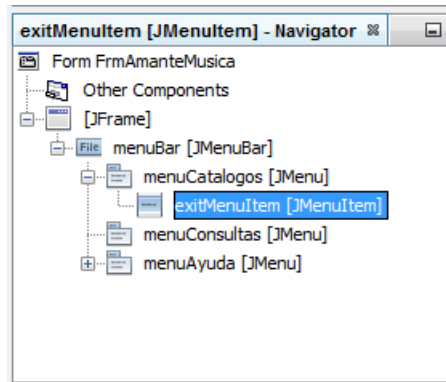


Figura 39

2. Usando el Editor de Propiedades, cambie el título de la opción del menú descendente **exitMenuItem** de **Exit** a **Salir** y el nombre de la variable del componente de **exitMenuItem** de **exitMenuItem** a **opcionMenuSalir**.
3. Repita los pasos 1 a 2 para cambiar el título de la opción **contentsMenuItem** del menú **menuAyuda** de **Contents** a **Contenido** y el nombre de la variable del componente de **contentsMenuItem** a **opcionMenuContenido**.
4. Repita los pasos 1 a 2 para cambiar el título de la opción **aboutMenuItem** del menú **menuAyuda** de **About** a **Acerca de** y el nombre de la variable del componente de **aboutMenuItem** a **opcionMenuAcercaDe**.

Creación de Menús Anidados

Ahora se le agregarán menús anidados a los menús descendentes de la barra de menú. Estos menús anidados aparecen cuando pasamos el cursor sobre su título. Para agregar un menú anidado a un menú descendente seguiremos el siguiente procedimiento:

1. En la ventana de inspección, haga clic con el botón derecho en el nodo que representa el menú descendente al que se desee agregar el menú anidado, en este caso el menú **menuCatalogos**. Aparecerá el menú emergente mostrado en la figura 40.
2. Haga clic en la opción **Add From Palette** del menú emergente. Aparecerá el menú emergente mostrado en la figura 41.

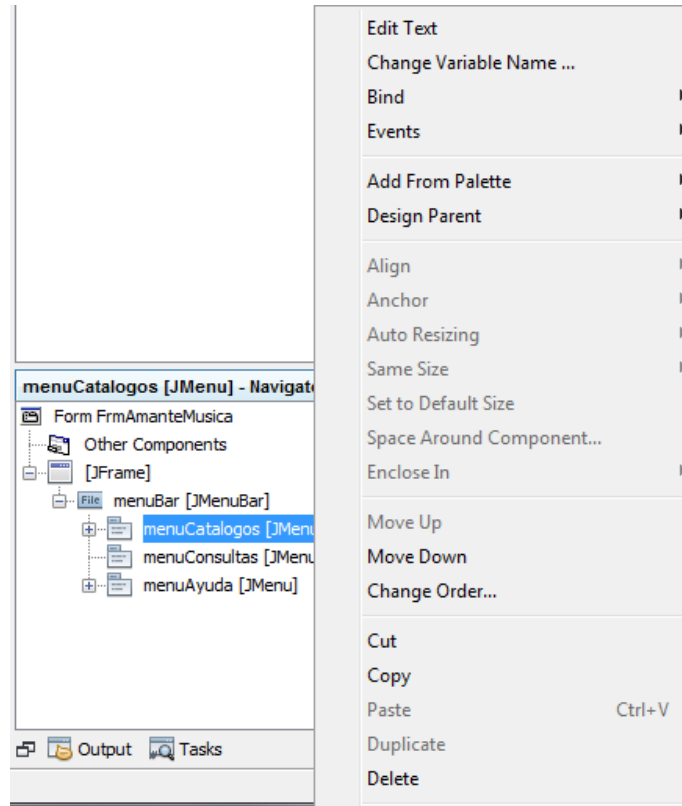


Figura 40

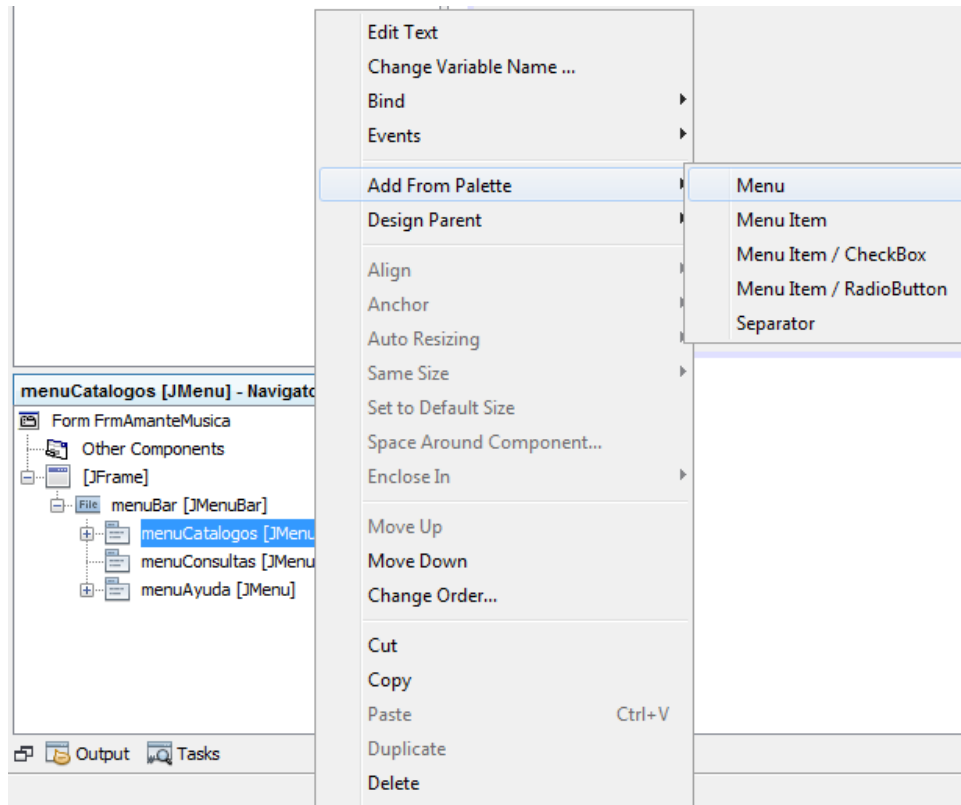


Figura 41

3. Haga clic en la opción **JMenu** del menú emergente. Se agregará un menú anidado al menú descendente como se muestra en la figura 42.

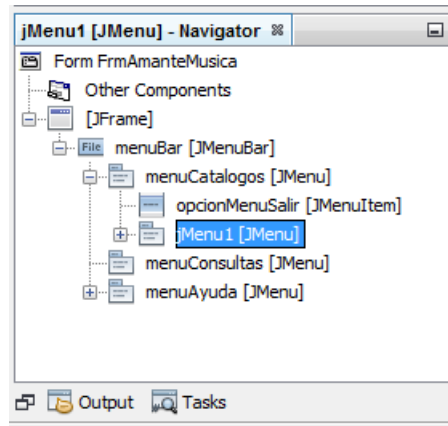


Figura 42

4. El menú anidado se creó debajo de la opción de menú **opcionMenuSalir**. Para subir el menú anidado para que quede encima de la opción de menú **opcionMenuSalir** haremos clic con el botón derecho y en el menú emergente haremos clic en la opción **Move Up**, como se muestra en la figura 43.

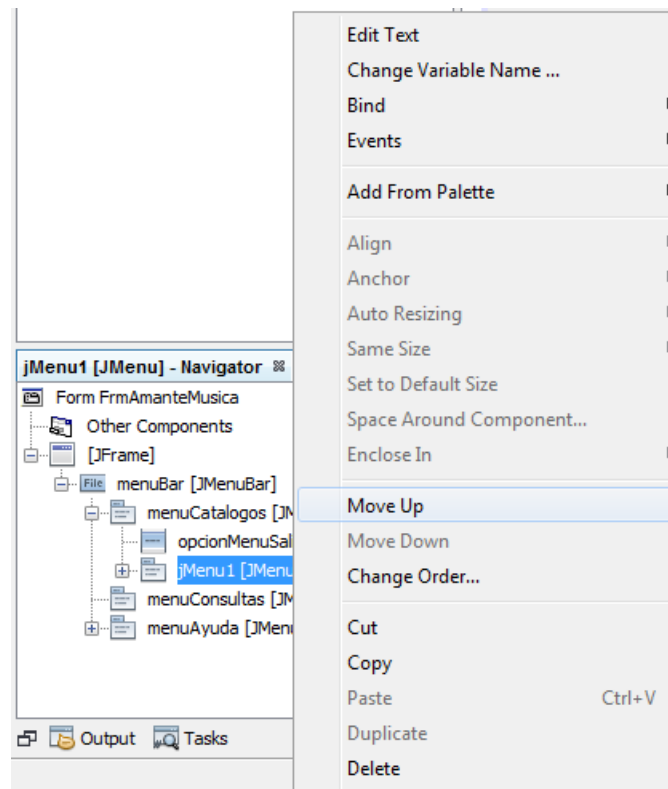


Figura 43

5. NetBeans subirá el menú anidado una posición como se muestra en la figura 44.

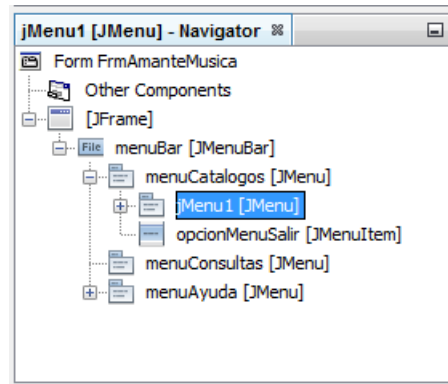


Figura 44

6. Usando el Editor de Propiedades, cambie el título del menú descendente **jMenu1** de **Menu** a **Catálogo de Canciones** y el nombre de la variable del componente de **jMenu1** a **menuCatalogoCanciones**.
7. Repita los pasos 1 a 6 para agregar un menú anidado a la opción **menuCatalogos** de la barra de menús y colóquelo debajo del menú anidado **menuCatalogoCanciones**. El título del menú y el nombre de la variable del componente serán **Catálogo de Películas** y **menuCatalogoPelículas**.
8. Repita los pasos 1 a 6 para agregar un menú anidado a la opción **menuCatalogos** de la barra de menús y colóquelo debajo del menú anidado **menuCatalogoPelículas**. El título del menú y el nombre de la variable del componente serán **Catálogo de Géneros** y **menuCatalogoGeneros**.
9. Repita los pasos 1 a 6 para agregar un menú anidado a la opción **menuConsultas** de la barra de menús. El título del menú y el nombre de la variable del componente serán **Consulta a Canciones** y **menuConsultasCanciones**.
10. Repita los pasos 1 a 6 para agregar un menú anidado a la opción **menuConsultas** de la barra de menús y colóquelo debajo del menú anidado **menuConsultasCanciones**. El título del menú y el nombre de la variable del componente serán **Consulta a Películas** y **menuConsultasPelículas**.
11. Repita los pasos 1 a 6 para agregar un menú anidado a la opción **menuConsultas** de la barra de menús y colóquelo debajo del menú anidado **menuConsultasPelículas**. El título del menú y el nombre de la variable del componente serán **Consulta a Géneros** y **menuConsultasGeneros**.

Agregado de Opciones de Menú

Ahora se le agregarán opciones de menú a los menús anidados. Para agregar una opción de menú a un menú anidado seguiremos el siguiente procedimiento:

1. En la ventana de inspección, haga clic con el botón derecho en el nodo que representa el menú anidado al que se desee agregar la opción de menú, en este caso el menú **menuCatalogoCanciones**. Aparecerá el menú emergente mostrado en la figura 45.

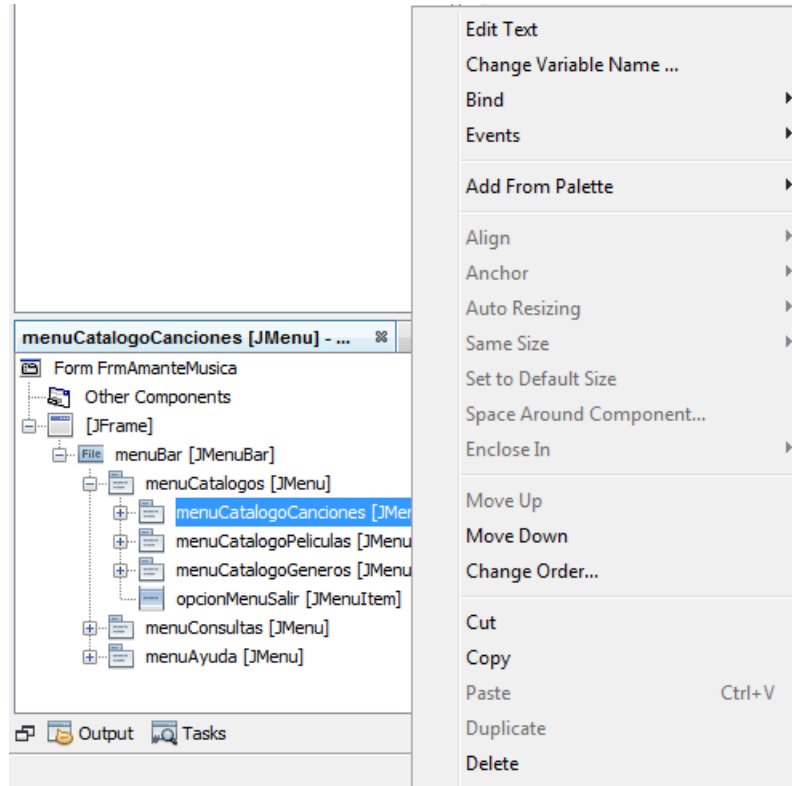


Figura 45

2. Haga clic en la opción **Add** del menú emergente. Aparecerá el menú emergente mostrado en la figura 46.
3. Haga clic en la opción **JMenuItem** del menú emergente. Se agregará un menú anidado al menú descendente como se muestra en la figura 47.
4. Usando el Editor de Propiedades, cambie el título de la opción de menú **jMenuItem1** de **Item** a **Agregar Canción** y el nombre de la variable del componente de **jMenuItem1** a **opcionMenuAgregarCancion**.
5. Repita los pasos 1 a 4 para agregar dos opciones de menú al menú anidado **menuCatalogoCanciones**. Los títulos de las opciones serán **Actualizar Canción** y **Eliminar Canción** y los nombres de las variables de las componentes serán **opcionMenuActualizarCancion** y **opcionMenuEliminarCancion**.

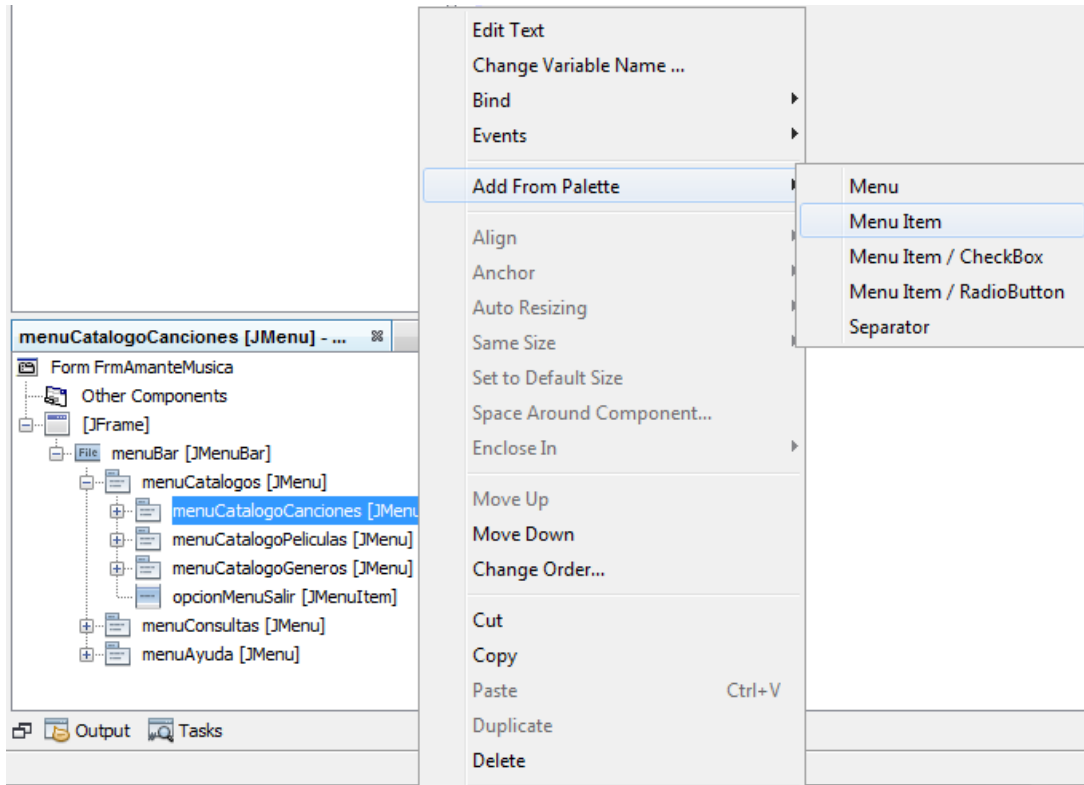


Figura 46

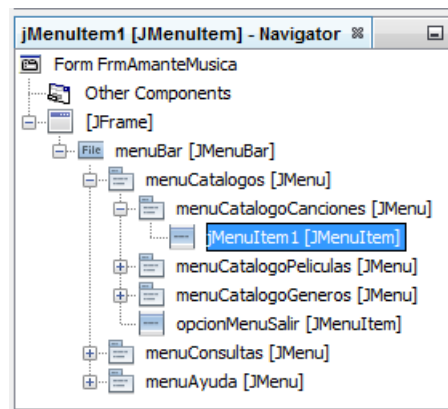


Figura 47

6. Repita los pasos 1 a 4 para agregar tres opciones de menú al menú anidado **menuCatalogoPeliculas**. Los títulos de las opciones serán **Agregar Película**, **Actualizar Película** y **Eliminar Película**. Los nombres de las variables de las componentes serán **opcionMenuAgregarPelicula**, **opcionMenuActualizarPelicula** y **opcionMenuEliminarPelicula**.
7. Repita los pasos 1 a 4 para agregar tres opciones de menú al menú anidado **menuCatalogoGeneros**. Los títulos de las opciones serán **Agregar Género**, **Actualizar Género** y **Eliminar Género**. Los nombres de las variables de las

componentes serán **opcionMenuAgregarGenero**, **opcionMenuActualizarGenero** y **opcionMenuEliminarGenero**.

8. Repita los pasos 1 a 4 para agregar ocho opciones de menú al menú anidado **menuConsultasCanciones**. Los títulos de las opciones serán **Todas**, **Por Título**, **Por Intérprete**, **Por Autor**, **Por Género**, **Por Álbum** y **Por Periodo**. Los nombres de las variables de las componentes serán **opcionMenuConsultasCancionesTodas**, **opcionMenuConsultasCancionesTitulo**, **opcionMenuConsultasCancionesInterprete**, **opcionMenuConsultasCancionesAutor**, **opcionMenuConsultasCancionesGenero**, **opcionMenuConsultasCancionesAlbum** y **opcionMenuConsultasCancionesPeriodo**.
9. Repita los pasos 1 a 4 para agregar seis opciones de menú al menú anidado **menuConsultas Peliculas**. Los títulos de las opciones serán **Todas**, **Por Título**, **Por Actor**, **Por Director**, **Por Género** y **Por Periodo**. Los nombres de las variables de las componentes serán **opcionMenuConsultasPeliculasTodas**, **opcionMenuConsultasPeliculasTitulo**, **opcionMenuConsultasPeliculasActor**, **opcionMenuConsultasPeliculasDirector**, **opcionMenuConsultasPeliculasGenero** y **opcionMenuConsultasPeliculasPeriodo**.
10. Repita los pasos 1 a 4 para agregar dos opciones de menú al menú anidado **menuConsultasGeneros**. Los títulos de los generos serán **Todos**, **De Canción** y **De Película**. Los nombres de las variables de las componentes serán **opcionMenuConsultasGenerosTodos**, **opcionMenuConsultasGenerosCancion** y **opcionMenuConsultasGenerosPelicula**.

Agregado de Separadores de Opciones de Menú

Por último se insertará un separador (una línea horizontal) entre el título del menú anidado **menuCatalogoGeneros** y la opción de menú **opcionMenuSalir** del menú descendente **menuCatalogos**. Para insertar un separador en un menú se sigue el siguiente procedimiento:

1. Haga clic con el botón derecho en el nodo que representa el menú en el que se desea insertar el separador, en este caso el menú **menuCatalogos**. Aparecerá lo mostrado en la figura 40.
2. Haga clic en la opción **Add From Palette** del menú emergente. Aparecerá el menú emergente mostrado en la figura 48

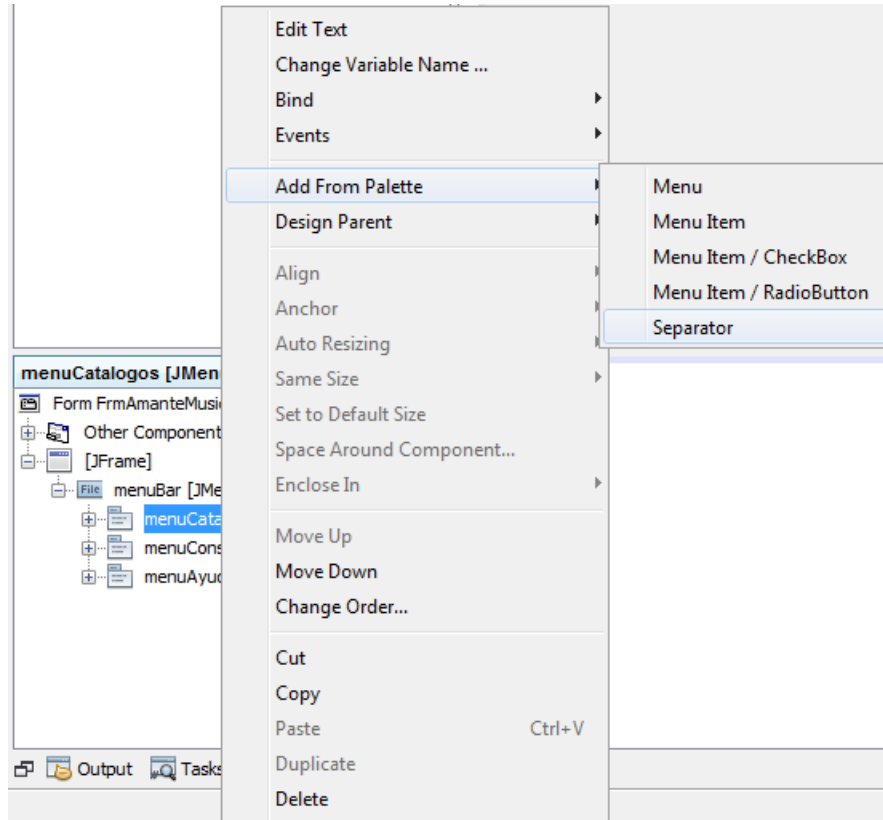


Figura 48

3. Haga clic en la opción **JSeparator** del menú emergente. Se agregará un separador al menú descendente como se muestra en la figura 49.

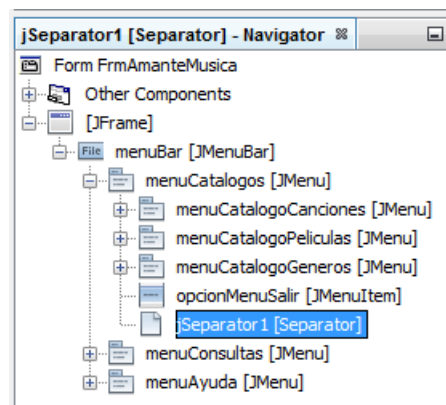


Figura 49

4. El separador se creó debajo de la opción de menú **opcionMenuSalir**. Para subir el separador para que quede encima de la opción de menú **opcionMenuSalir** haremos clic con el botón derecho y en el menú emergente haremos clic en la opción **Move Up**, como se muestra en la figura 50.
5. NetBeans subirá el separador una posición como se muestra en la figura 51.

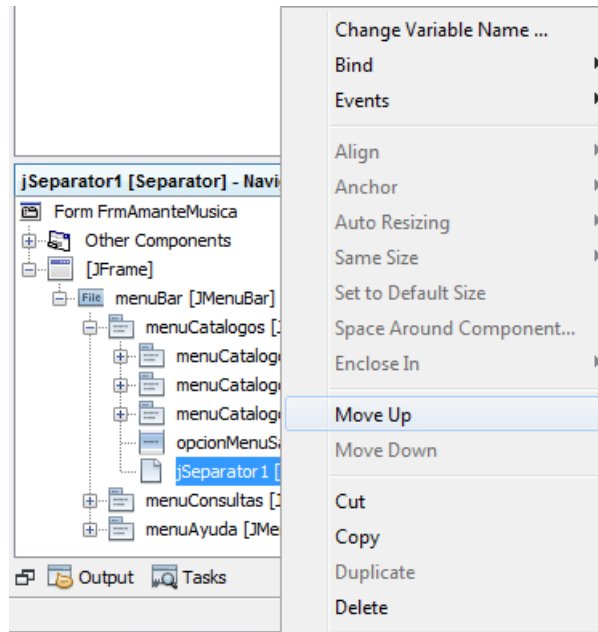


Figura 50

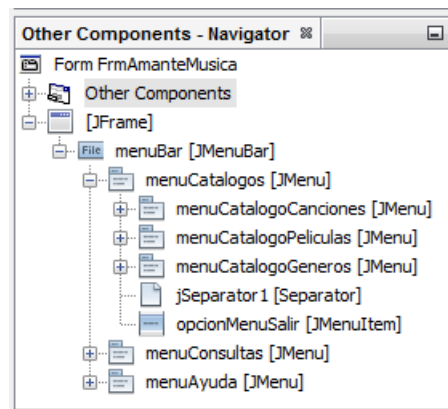


Figura 51

Establecimiento de los Métodos Oyentes de los Eventos Generados por las Opciones de Menús.

Cuando hacemos clic sobre una opción de menú, ésta genera un evento del tipo **ActionEvent**. Si queremos que nuestro programa reaccione a ese evento, debemos de establecer un método oyente del tipo **actionPerformed()** en el contenedor de ese menú, que normalmente es una ventana. Para establecer un método oyente para una opción de menú de un menú, seguiremos el siguiente procedimiento:

1. Haga clic en la opción de menú a la que se le desea agregar un método oyente. En este caso a la opción de menú **opcionMenuAgregarCancion** del menú anidado **menuCatalogoCanciones** del menú descendente **menuCatalogos**, de la barra de menús de la clase **FrmAmanteMusica**, figura 52.

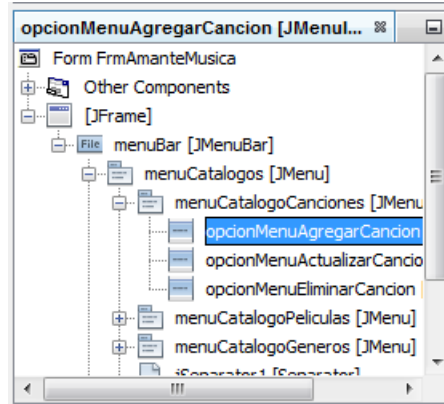


Figura 52

- Haga clic en el selector **Events** del Editor de Propiedades. Aparecerá la lista de eventos que puede generar el elemento de menú, figura 53.

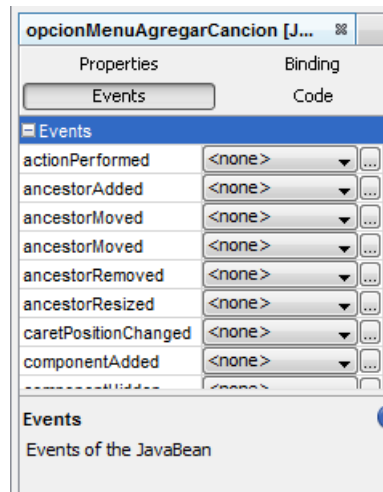


Figura 53

- Haga clic en la caja combinada a la derecha del evento **actionPerformed** y seleccione la opción **opcionMenuAgregarCancionActionPerformed** que será el nombre sugerido por NetBeans para el método oyente del tipo **actionPerformed()**, figura 54.
- NetBeans genera el esqueleto del método oyente y nos lo muestra en la Vista de Código Fuente, como se muestra en la figura 55. En el cuerpo de ese método colocaremos el código que queramos que se ejecute cuando se haga clic en la opción de menú **opcionMenuAgregarCancion**.

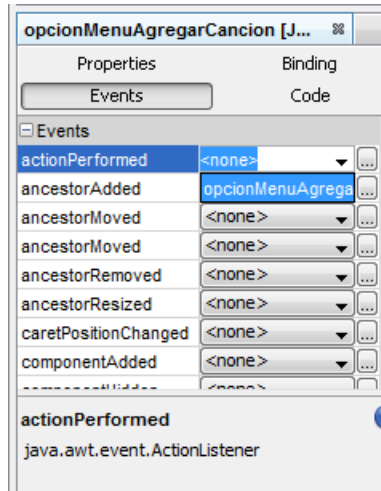


Figura 54

```

256
257 private void opcionMenuAgregarCancionActionPerformed(java.awt.event.ActionEvent
258 // TODO add your handling code here:
259 }
260

```

Figura 55

- Repita los pasos 1 a 4 para agregar un método oyente para cada una de las demás opciones de menú de la ventana **FrmAmanteMusica**:
 - opcionMenuActualizarCancion**,
 - opcionMenuEliminarCancion**,
 - opcionMenuAgregarPelicula**,
 - opcionMenuActualizarPelicula**,
 - opcionMenuEliminarPelicula**,
 - opcionMenuAgregarGenero**,
 - opcionMenuActualizarGenero**,
 - opcionMenuEliminarGenero**,
 - opcionMenuConsultasCancionesTodas**,
 - opcionMenuConsultasCancionesTitulo**,
 - opcionMenuConsultasCancionesInterprete**,
 - opcionMenuConsultasCancionesAutor**
 - opcionMenuConsultasCancionesGenero**,
 - opcionMenuConsultasCancionesAlbum**,
 - opcionMenuConsultasCancionesPeriodo**,
 - opcionMenuConsultasPeliculasTodas**,
 - opcionMenuConsultasPeliculasTitulo**,
 - opcionMenuConsultasPeliculasActor**,
 - opcionMenuConsultasPeliculasDirector**,
 - opcionMenuConsultasPeliculasGenero**,
 - opcionMenuConsultasPeliculasPeriodo**,
 - opcionMenuConsultasGenerosTodos**,
 - opcionMenuConsultasGenerosMedio**,
 - opcionMenuAcercaDe** y
 - opcionMenuContenido**.

Edición de la Ventana Principal de la Aplicación

A continuación se modificará el tamaño de la ventana principal de la aplicación y se le agregarán una etiqueta y una tabla en la que se desplegarán los resultados de las consultas.

Para modificar el tamaño de la ventana principal de la aplicación seguimos el siguiente procedimiento:

1. En la Vista de Diseño mueva el cursor a una de las orillas del cuadro que representa la ventana de la aplicación y déjelo ahí unos segundos. Aparecerá una etiqueta mostrando el tamaño de la ventana de la aplicación y un mensaje indicando que para cambiar el tamaño de la ventana se arrastre el ratón o se haga doble clic en la orilla, figura 56.

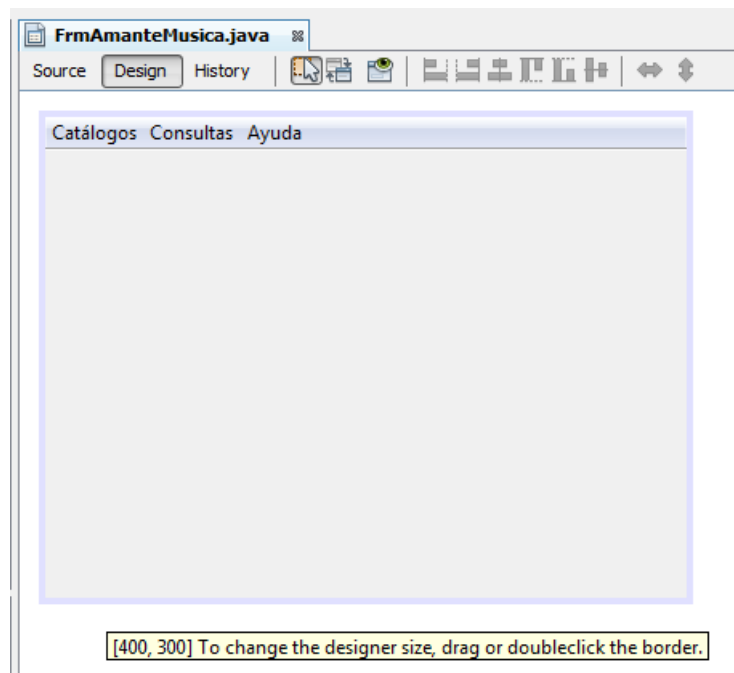


Figura 56

2. Al hacer doble clic en la orilla aparecerá un cuadro de diálogo con un campo de texto mostrando las dimensiones actuales de la ventana, 400 pixeles de ancho y 300 pixeles de alto, figura 57.
3. Teclearemos los nuevos valores: ancho y alto separados por una coma. En este caso 600, 450 y haremos clic en el botón **OK**. Al hacerlo, el cuadro que representa la ventana de la aplicación cambiará de tamaño, figura 58.

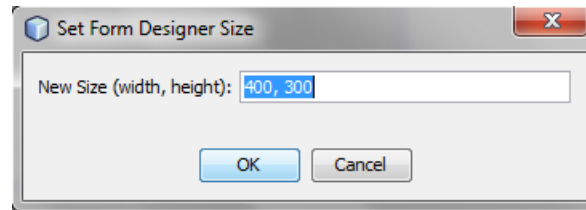


Figura 57

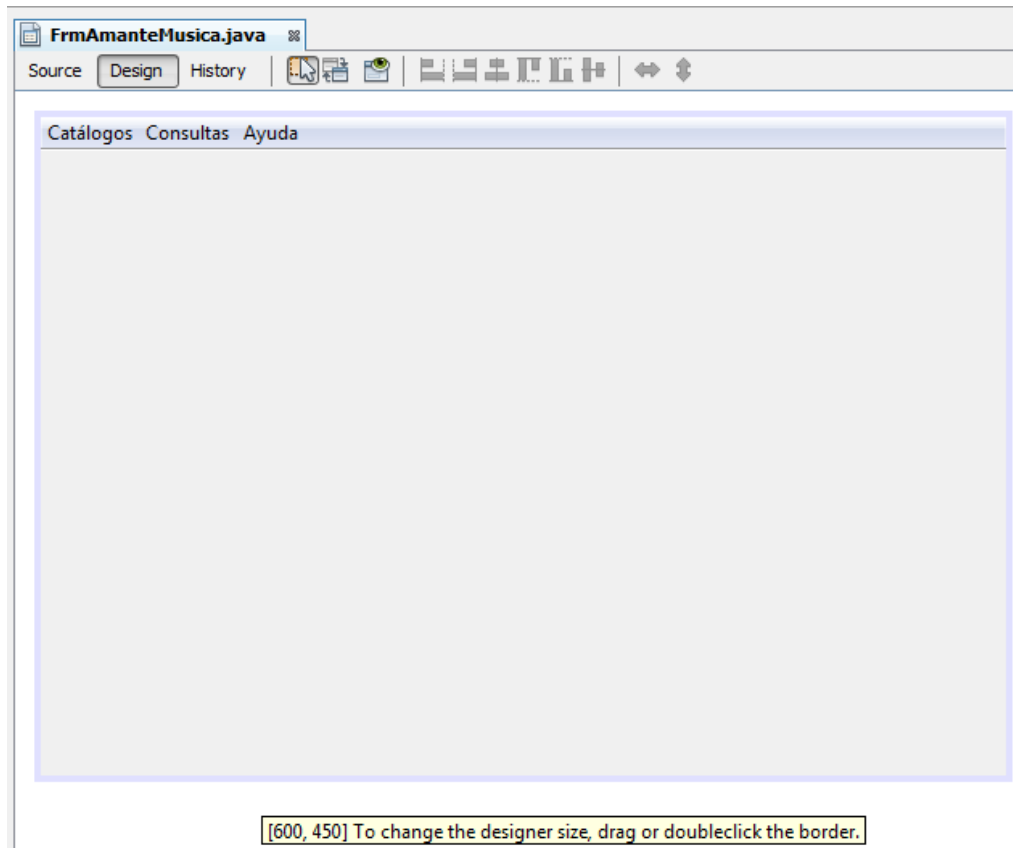


Figura 58

A continuación le agregaremos una etiqueta a la ventana de aplicación que contendrá el título de la tabla en la que se desplegarán los resultados de las consultas. Para agregar una etiqueta seguiremos el siguiente procedimiento:

1. De la Paleta de Componentes seleccione el icono que representa una etiqueta, la componente **JLabel**, figura 59.
2. Arrastre con el ratón el icono de la etiqueta y suéltelo, digamos a un centímetro por debajo y un centímetro a la derecha de la esquina superior izquierda de la ventana de la aplicación, figura 60.
3. Usando el Editor de Propiedades, cambie el nombre de la variable de la etiqueta de **jLabel1** a **títuloTabla**.

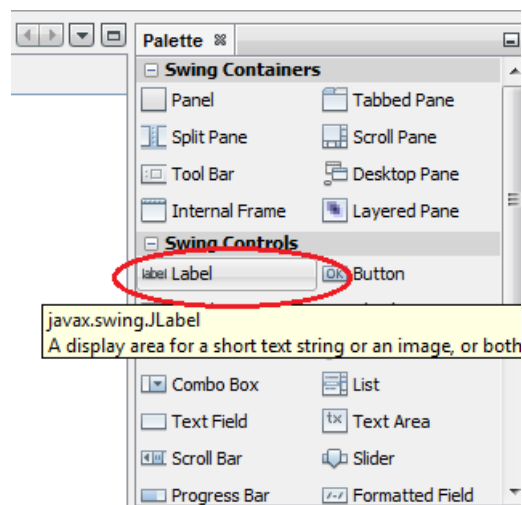


Figura 59

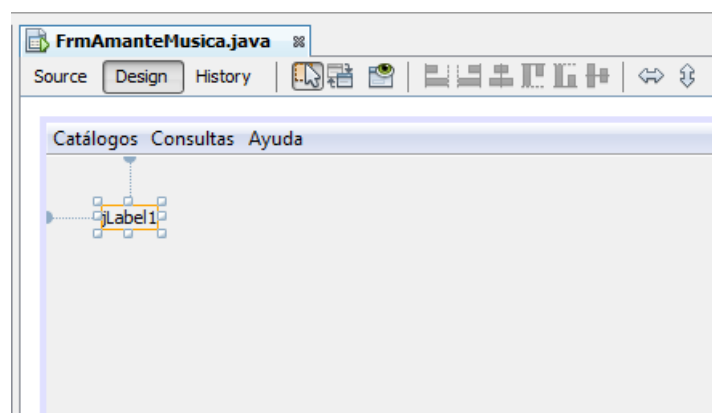


Figura 60

4. En el Editor de Propiedades haremos clic en el botón que se encuentra del lado derecho de la propiedad **font**, figura 61, para abrir el cuadro de dialogo que nos permite establecer las propiedades de la fuente empleada en el texto de la etiqueta, figura 62.
5. Modificaremos la propiedad **Size** cambiando su valor a 18 y haciendo clic en el botón **OK**. Al hacerlo el tamaño del texto de la etiqueta aumenta, reflejando el cambio, figura 63.

A continuación se le agregará a la ventana de la aplicación un panel con barras de deslizamiento, **JScrollPane**, que servirá como contenedor para la tabla en la que se desplegarán los resultados de las consultas. El propósito de este panel es el de utilizar las barras de deslizamiento para ir mostrando porciones de una tabla cuyo tamaño exceda al tamaño que será visible. Para agregar un panel con barras de deslizamiento haremos lo siguiente:

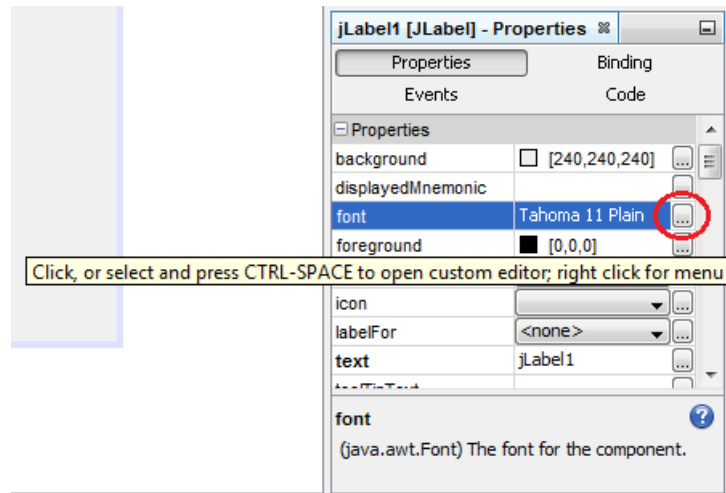


Figura 61

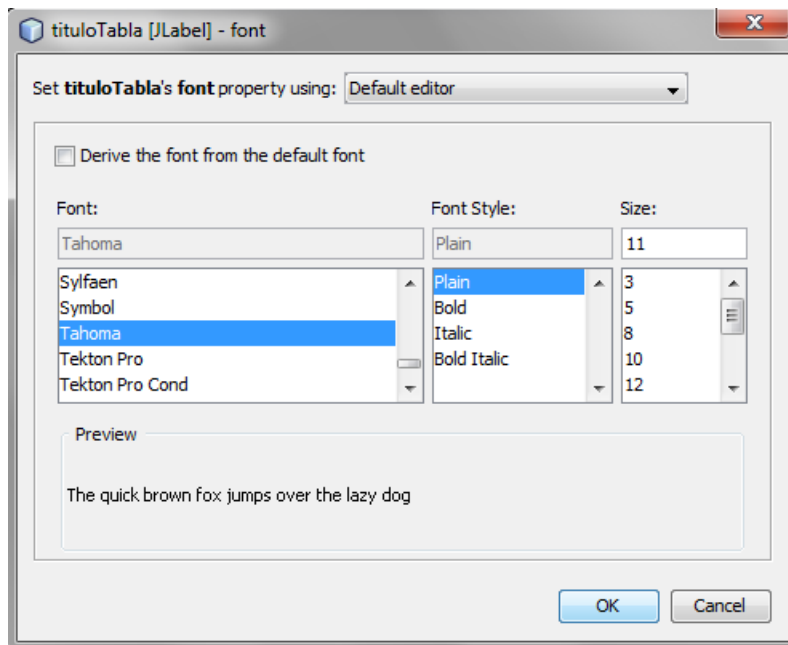


Figura 62

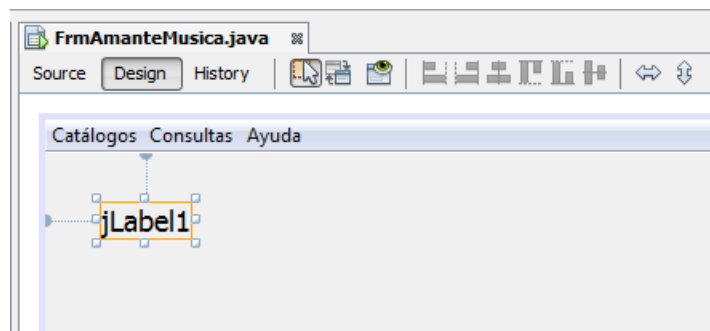


Figura 63

1. En la paleta de componentes seleccione el icono que representa un panel con barras de deslizamiento, la componente **JScrollPane**, figura 64.

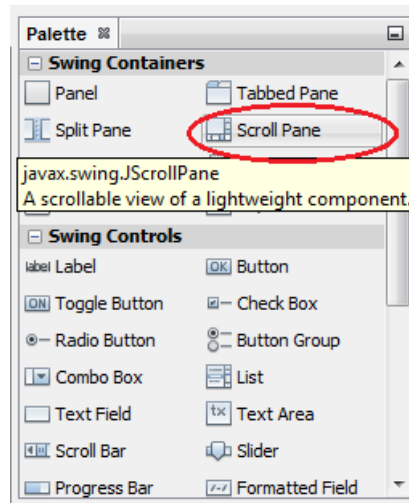


Figura 64

2. Arrastre con el ratón el icono del panel con las barras de deslizamiento por debajo de la etiqueta hasta que la guía de alineación indique que el margen izquierdo del panel está alineado con la etiqueta, figura 65, y suéltelo. Aparecerá lo mostrado en la figura 66.

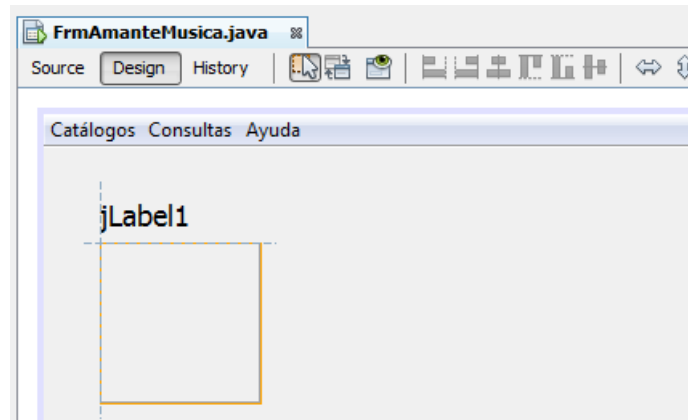


Figura 65

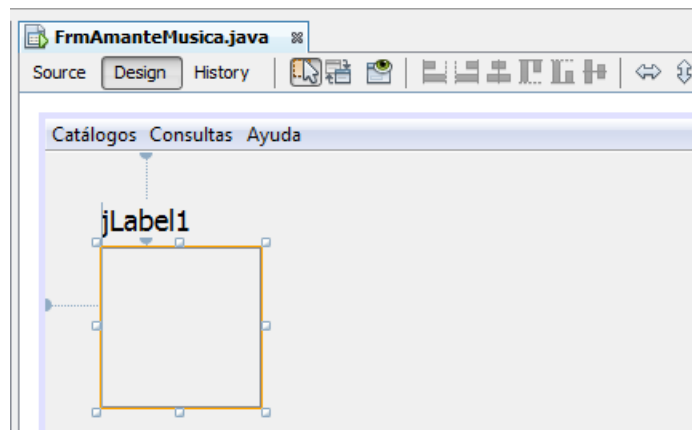


Figura 66

3. Utilice el Editor de Propiedades para cambiar el valor de la propiedad **Horizontal Size** de 100 a 525 y el valor de la propiedad **Vertical Size** de 100 a 300 para cambiar el tamaño del panel con barras de deslizamiento. Al hacerlo obtendremos lo mostrado en la figura 67.

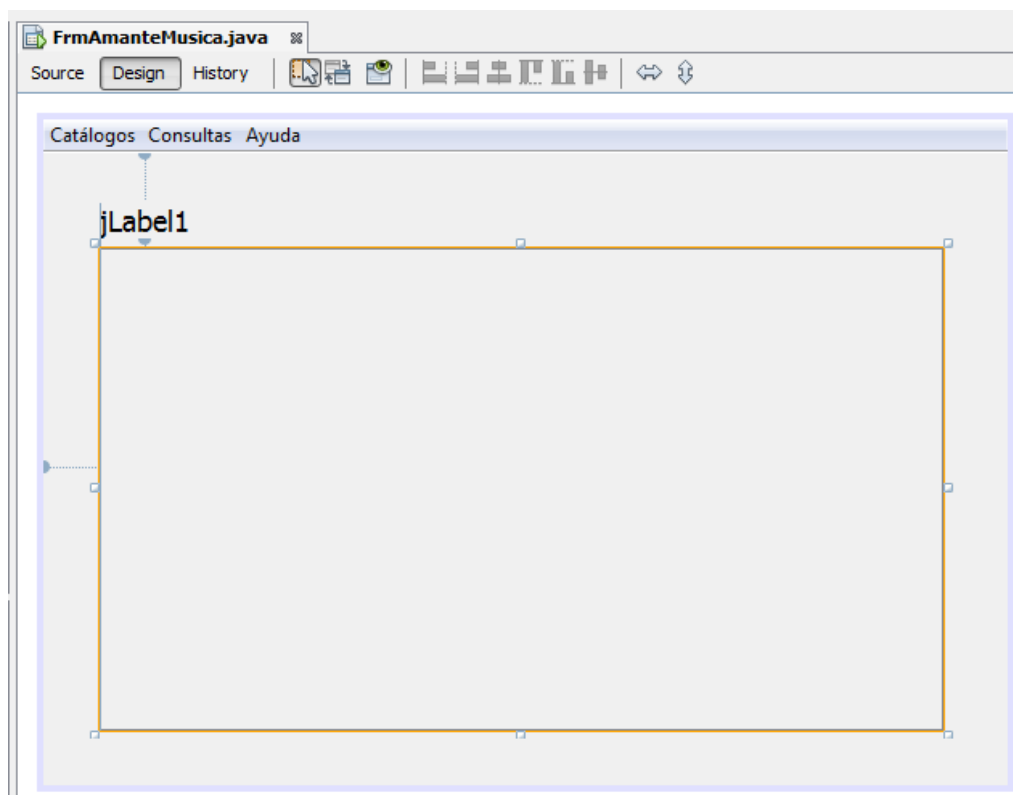


Figura 67

4. Utilice el Editor de Propiedades para cambiar el valor de las propiedades **Horizontal Resizable** y **Vertical Resizable** de **false** a **true** marcando las casillas de verificación de esas propiedades.

- Para modificar el tipo de marco del panel con las barras de deslizamiento haremos clic en el botón que se encuentra del lado derecho de su propiedad **border**, en el Editor de Propiedades figura 68.

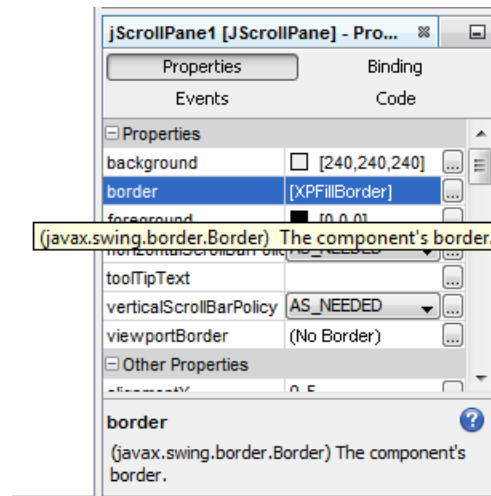


Figura 68

- Aparece el cuadro de dialogo que nos permite seleccionar el tipo de marco, figura 69. Seleccionaremos la primera opción: **(No Border)** para eliminar el marco alrededor del panel.

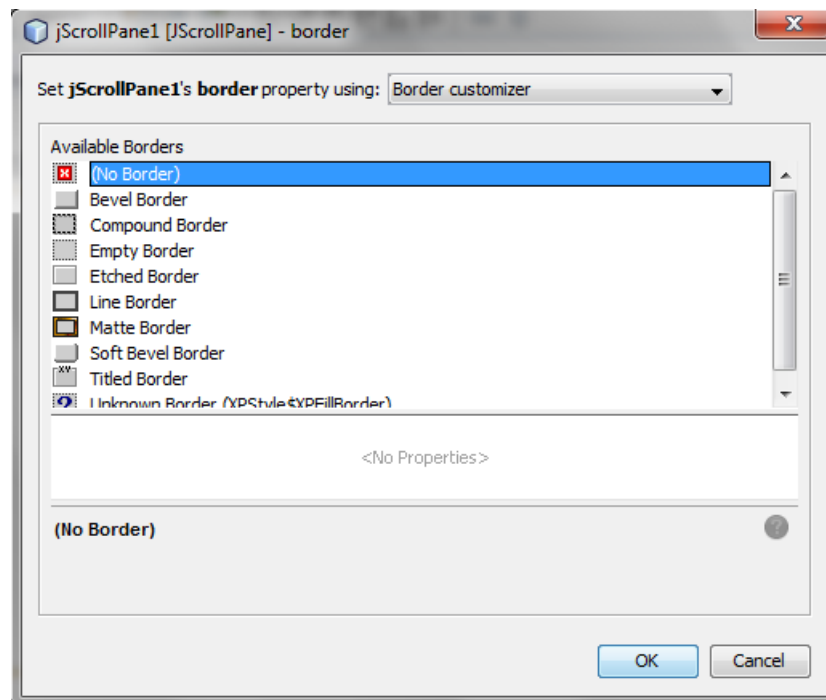


Figura 69

A continuación, en el Editor de Propiedades modificaremos la propiedad **text** de la etiqueta, **tituloTabla**, borrando su valor por omisión. El texto que se desplegará en esta

etiqueta dependerá de la tabla desplegada y se establecerá en el programa. Al borrar el valor de la etiqueta se borrará en el Editor de Código y sólo se verán las guías que indican la alineación de la etiqueta, figura 70.

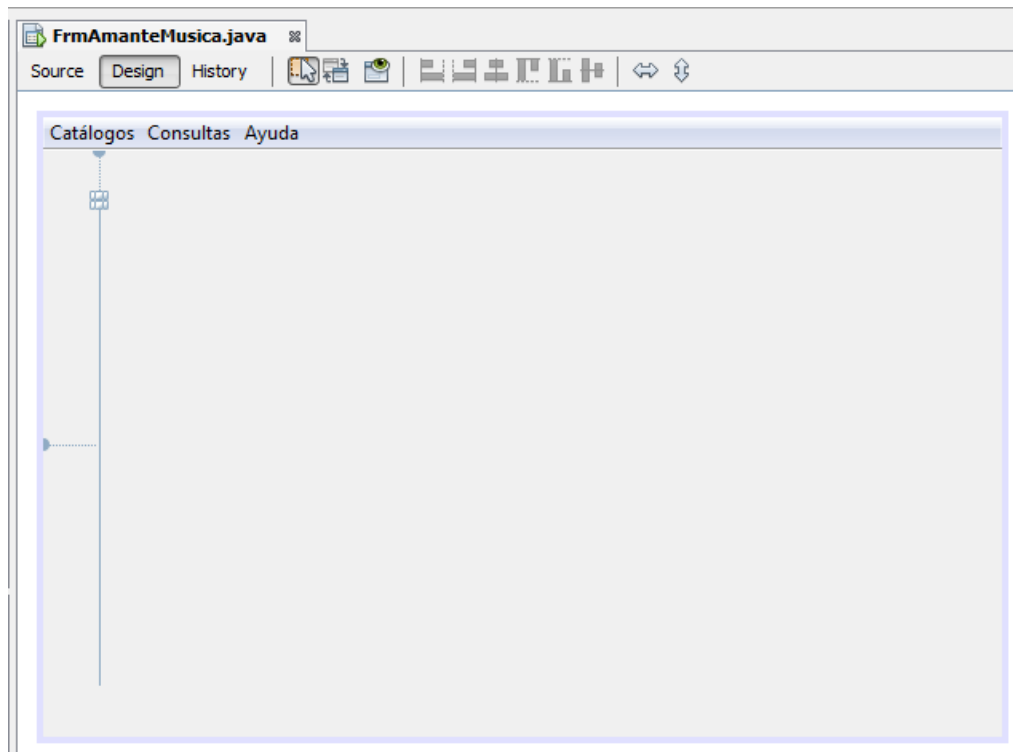


Figura 70

Por último agregaremos el código para agregarle una tabla al panel con las barras de deslizamiento:

1. Seleccione la Vista de Código Fuente del Panel de Edición. Vaya al final del código se encuentran las declaraciones de los atributos de la clase.
2. Agregue la declaración de la tabla:

```
private javax.swing.JTable jtabla;
```

3. Después del método `centraVentana()` empleado para centrar la ventana de la aplicación en la pantalla agregue el siguiente método que crea una tabla y la despliega dentro de un panel con barras de deslizamiento. El parámetro `titulo` se usa para desplegar el título de la tabla en la etiqueta `tituloTabla`. El parámetro `modeloTabla` es un objeto de la clase `DefaultTableModel`, que encapsula los títulos de las columnas de la tabla y los valores de la tabla.

```
/**
 * Este método crea un objeto del tipo JTable dentro de un panel con
 * barras de deslizamiento y la despliega
 */
```

```

    * @param tabla objeto TableModel con los datos de una tabla
    */
    public void despliegaTabla(Tabla tabla) {
        // Crea la tabla a partir del modelo de la tabla con los valores
        // de los titulos de las columnas y los valores de las celdas
        jTablela = new javax.swing.JTable(tabla.getModeloTabla());

        // Establece el título de la tabla
        tituloTabla.setText(tabla.getTitulo());

        // Hace que el control del tamaño de la tabla y la porción visible
        // lo tenga la barra de deslizamiento y no la tabla
        jTablela.setAutoResizeMode(javax.swing.JTable.AUTO_RESIZE_OFF);
        jTablela.setAutoscrolls(false);

        // Hace visible la tabla dentro del panel con barras de
        // deslizamiento
        jScrollPane.setViewportViewView(jTablela);
    }
}

```

Creación de un Cuadro de Diálogo

Los cuadros de diálogo se utilizan principalmente para capturar, editar o desplegar datos. Los cuadros de diálogo normalmente tienen varios botones: para aceptar y cancelar la operación o para restaurar los valores de los campos del cuadro de diálogo a su valor original. Los botones para aceptar o cancelar la operación normalmente cierran el cuadro de diálogo. La operación a realizar por el cuadro de diálogo y el botón presionado para cerrar el cuadro de diálogo se establecerán como parámetros del constructor de la clase que representa al cuadro de diálogo. Los diferentes valores que pueden tomar esos parámetros se definen en la clase `UtileriasGUI`.

```

/*
 * ConstantesGUI.java
 *
 * @author mdomitsu
 */
package interfazUsuario;

/**
 * Esta clase define constantes empleadas en los cuadros de dialogo
 */
public class ConstantesGUI {
    // Tipos de operaciones en los que se van a usar los cuadros de dialogos
    public static int AGREGAR = 0;
    public static int ACTUALIZAR = 1;
    public static int ELIMINAR = 2;
    public static int DESPLEGAR = 3;

    // Tipo de boton presionado al salir de los cuadros de dialogos
    public static String ACEPTAR = "Aceptar";
    public static String CANCELAR = "Cancelar";
}

```

Edite la clase en el paquete `interfazUsuario`.

El procedimiento para crear un cuadro de diálogo es el siguiente:

1. De la barra de menú de NetBeans, seleccione la opción **Files/New File**, presione las teclas **Ctrl+ N** o haga clic en el icono **New File**, como se muestra en la figura 6:
2. Aparecerá el primer cuadro de diálogo del asistente para crear una clase, figura 7.
3. Del recuadro **Categories:** seleccione el nodo **Java GUI Forms** y del recuadro **File Types** seleccionaremos el tipo **JDialog Form** que nos creará el esqueleto de un cuadro de diálogo, figura 71. Presione el botón **Next**.

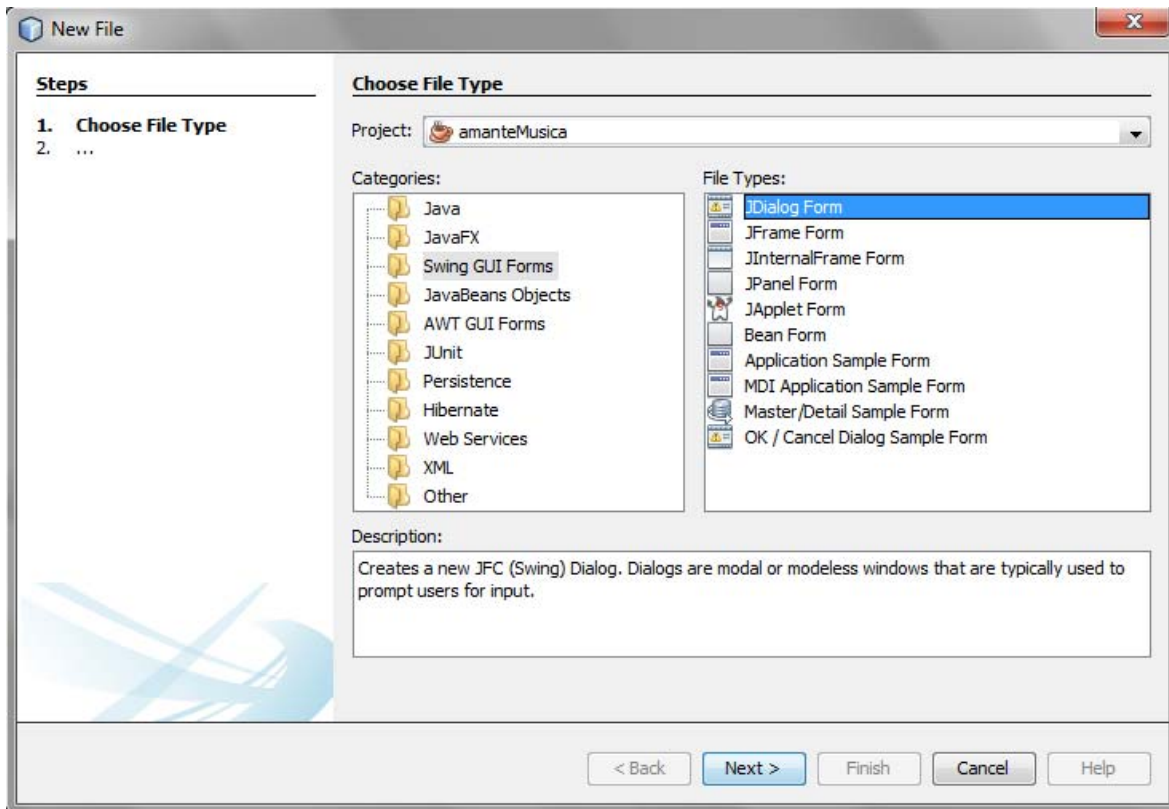


Figura 71

4. Aparecerá el segundo cuadro de diálogo del asistente para crear clases, mostrada en la figura 72. Aquí seleccionamos el nombre y la ubicación de la clase del cuadro de diálogo.

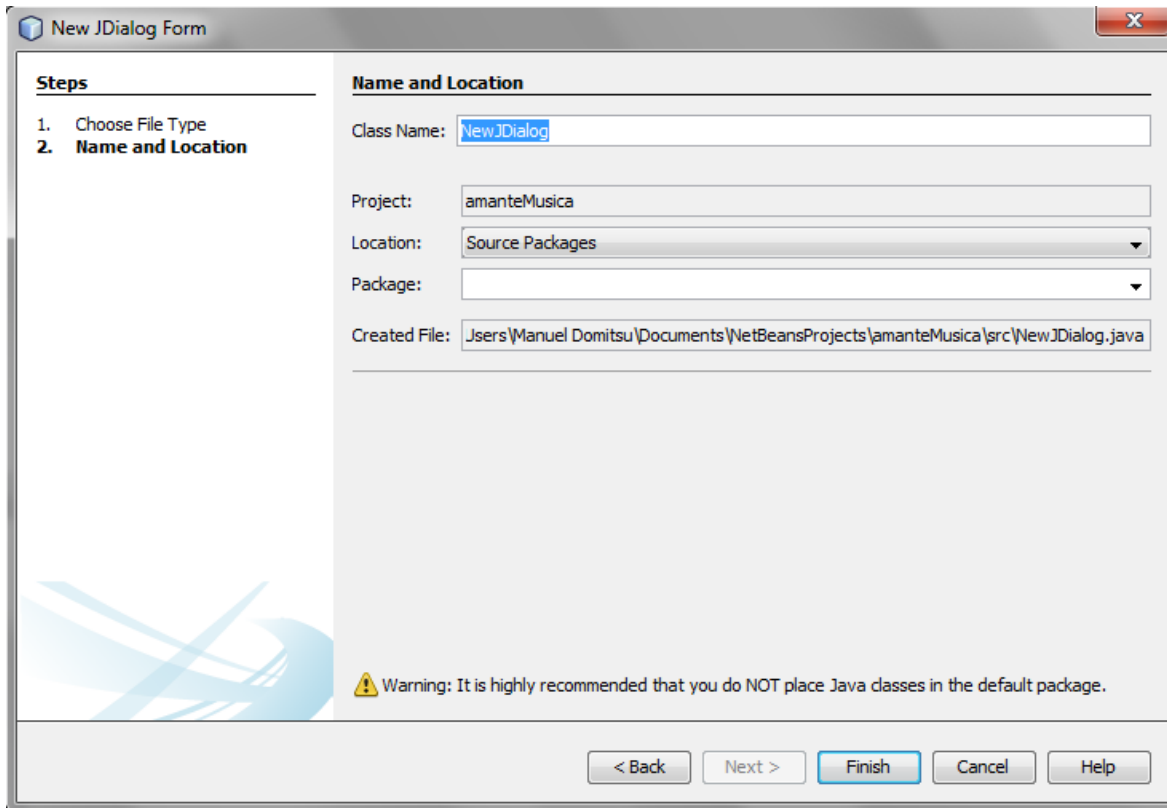


Figura 72

- a) En el cuadro de texto **Class Name:** establezca el nombre de la clase. Por ejemplo, “**DlgCancion**” ya que este cuadro de diálogo se empleará para capturar, editar o mostrar los datos de la clase `Cancion`.
 - b) En el cuadro de texto **Package:** establezca el paquete donde estará la clase. Por ejemplo, “**interfazUsuario**”.
 - c) Presione el botón **Finish**.
5. Desaparecerá el asistente para crear una nueva clase y aparecerá lo mostrado en la Figura 73.
 6. Cámbiese a la Vista de Código.
 6. Agréguele a la clase las siguientes directivas import:

```
import javax.swing.DefaultComboBoxModel;

import objetosServicio.Fecha;
import objetosNegocio.Cancion;
```

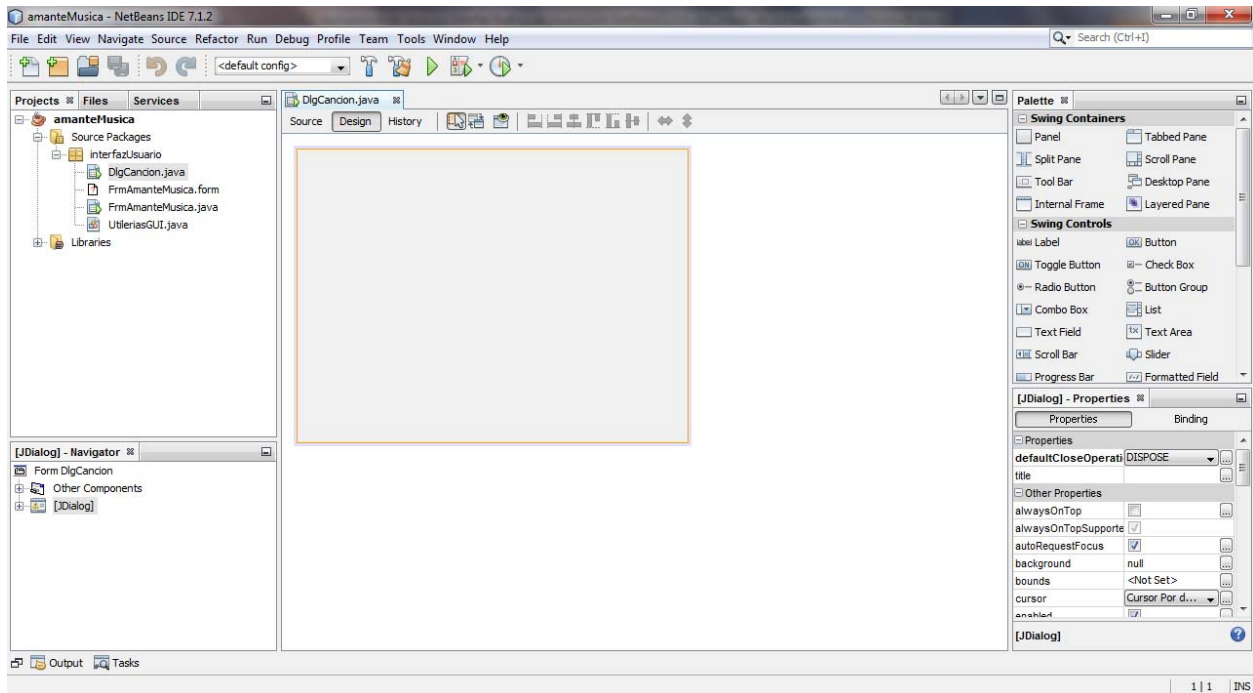



Figura 73

7. Al crear un cuadro de diálogo, Netbeans le agrega un método `main()`. Como nuestro cuadro de diálogo va a formar parte de la aplicación **AmanteMusica** cuya ventana principal ya tiene un método `main()`, el del cuadro de diálogo no se requiere y por lo tanto podemos eliminarlo. Busque el método `main()` del cuadro de diálogo y elimínelo.
8. Agréguele a los atributos de la clase, al final del código de la clase, figura 74, los siguientes atributos:

```
private Cancion cancion;
private DefaultComboBoxModel listaGenerosCanciones;
private int operacion;
private StringBuffer respuesta;
```

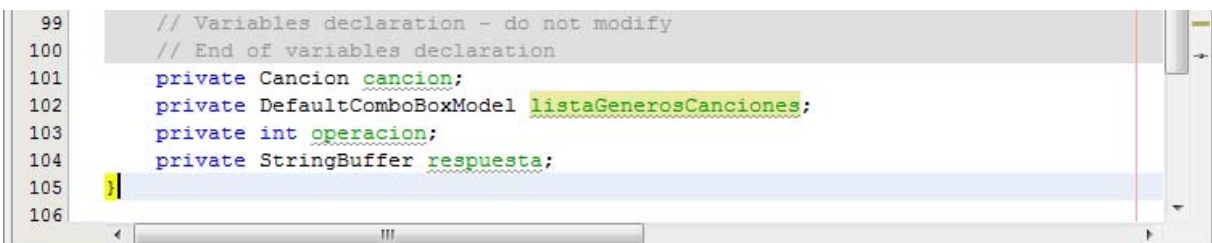


Figura 74

El atributo `cancion` contendrá los datos de la canción a agregar, actualizar o borrar. El atributo `listaGenerosCanciones` es una lista de los posibles

géneros que puede tener una canción. El atributo `operacion` representa la operación que se desea realizar: agregar, actualizar o borrar. El atributo `respuesta` contiene un código establecido para el botón presionado al cerrar el cuadro de diálogo.

9. Modifique el constructor de la clase del cuadro de diálogo para que inicialice los atributos agregados en el paso 9, de la siguiente manera:

```
/**
 * Constructor que establece las características del cuadro de diálogo
 * y la operación a realizar con él
 * @param parent Ventana sobre la que aparecerá el cuadro de diálogo
 * @param title Título del cuadro de diálogo
 * @param modal true si permite acceder fuera de los límites del cuadro
 * de diálogo, false en caso contrario
 * @param cancion Canción a capturar, editar o desplegar
 * @param listaGenerosCanciones Lista de los generos de una canción
 * @param operacion Operación a realizar en el cuadro de diálogo:
 *         AGREGAR = 0, ACTUALIZAR = 1, ELIMINAR = 2, DESPLEGAR = 3;
 * @param respuesta Boton presionado al salir de los cuadros de
 * diálogos: ACEPTAR = "Aceptar", CANCELAR = "Cancelar".
 */
public DlgCancion(java.awt.Frame parent, String title, boolean modal,
                  Cancion cancion,
                  DefaultComboBoxModel listaGenerosCanciones,
                  int operacion, StringBuffer respuesta) {
    super(parent, title, modal);
    this.cancion = cancion;
    this.listaGenerosCanciones = listaGenerosCanciones;
    this.operacion = operacion;
    this.respuesta = respuesta;

    initComponents();
}
```

Agregado de Etiquetas al Cuadro de Diálogo

A continuación le agregaremos al cuadro de diálogo los componentes gráficos. Primero le agregaremos al cuadro de diálogo una serie de etiquetas que identificarán los campos de texto y la caja combinada empleados para capturar los datos de una canción. El procedimiento es el siguiente:

1. Agregue una etiqueta, digamos a un centímetro por debajo y un centímetro a la derecha de la esquina superior izquierda del cuadro de diálogo.
2. Usando el editor de propiedades cambie el valor de la propiedad **text** de **JLabel1** a **Clave**. Esto establece el texto desplegado en la etiqueta. Al hacerlo tendremos lo mostrado en la figura 75.

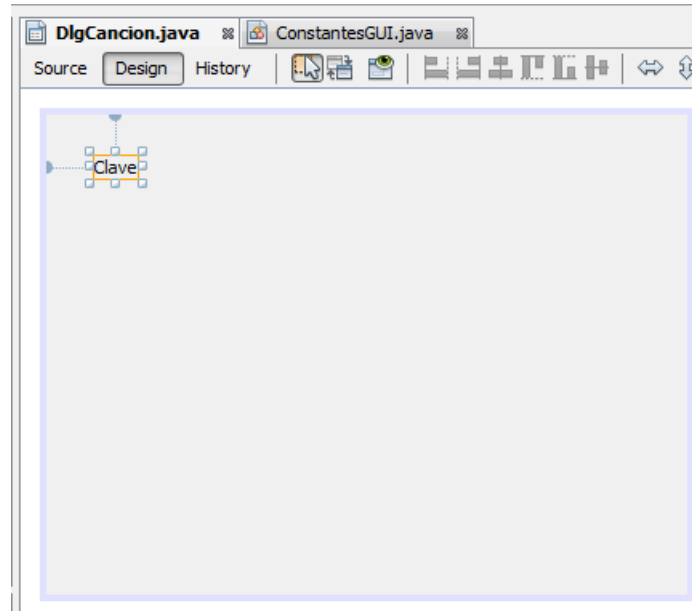


Figura 75

3. Agregue una etiqueta por debajo de la etiqueta **Clave**, arrastrando el icono hasta que la guía de alineación vertical muestre que la etiqueta está alineada con la etiqueta **Clave** y aparezca una guía horizontal en la etiqueta indicando que las etiquetas se encuentran separados por la distancia vertical predeterminada y suéltela, figura 76. Usando el Editor de Propiedades cambie el valor de la propiedad **text** a **Título**.

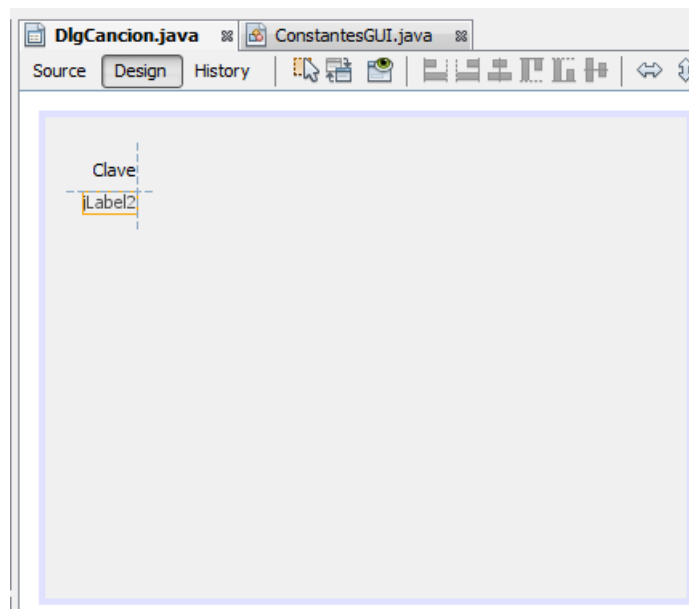


Figura 76

4. Repita el paso 3 para agregar una etiqueta por debajo de la etiqueta **Título** y modifique su propiedad **text** a **Intérprete**.

5. Repita el paso 3 para agregar una etiqueta por debajo de la etiqueta **Intérprete** y modifique su propiedad **text** a **Autor**.
 6. Repita el paso 3 para agregar una etiqueta por debajo de la etiqueta **Autor de la letra** y modifique su propiedad **text** a **Género**.
 7. Repita el paso 3 para agregar una etiqueta por debajo de la etiqueta **Género** y modifique su propiedad **text** a **Álbum**.
 8. Repita el paso 3 para agregar una etiqueta por debajo de la etiqueta **Disquera** y modifique su propiedad **text** a **Duración**.
 9. Repita el paso 3 para agregar una etiqueta por debajo de la etiqueta **Duración** y modifique su propiedad **text** a **Fecha (dd/mm/aaaa)**.
10. Al hacerlo tendremos lo mostrado en la figura 77.

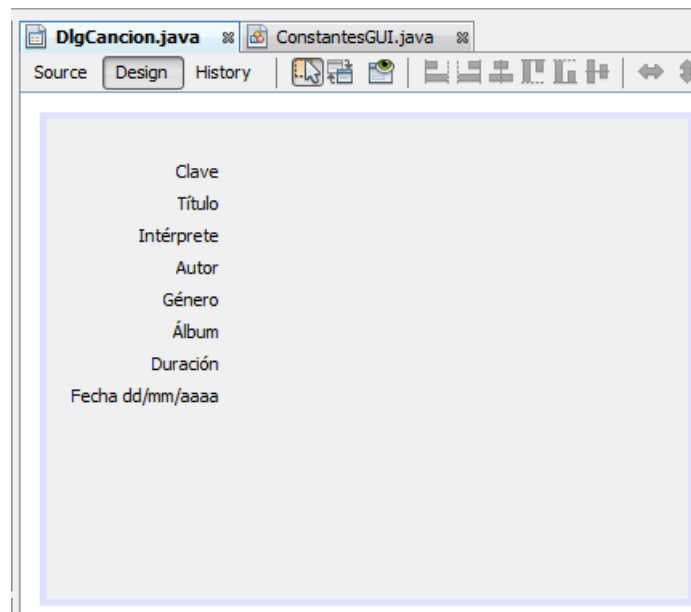


Figura 77

Agregado de Campos de Texto al Cuadro de Diálogo

Enseguida le agregaremos al cuadro de diálogo los campos de texto para capturar los datos, menos el género, de una canción. El procedimiento es el siguiente:

1. De la Paleta de Componentes seleccione el icono que representa un campo de texto, la componente **JTextField**, figura 78.

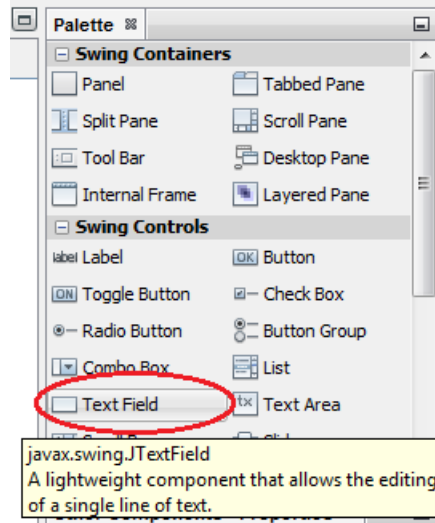


Figura 78

2. Arrastre con el ratón el icono del campo de texto hasta que la guía de alineación horizontal muestre que el texto de la etiqueta **Clave** y la del campo de texto están alineados y aparezca una guía vertical en el campo de texto indicando que los elementos se encuentran separados por la distancia horizontal entre elementos predeterminada y suéltelo, figura 79.

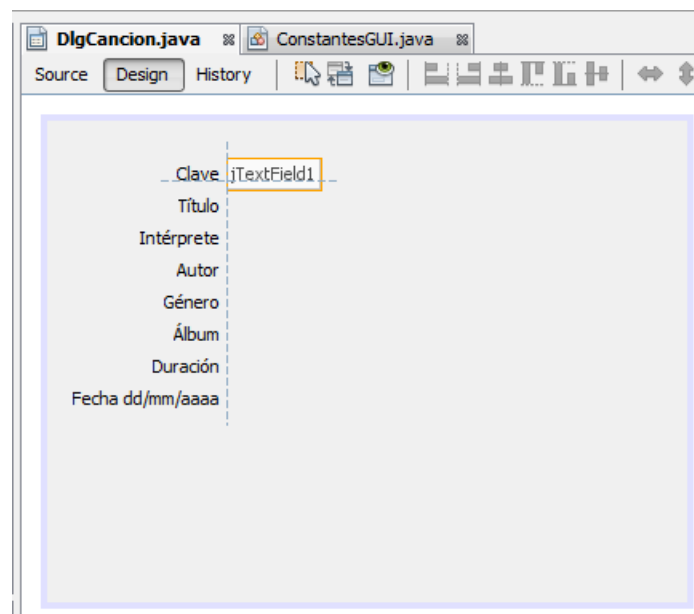


Figura 79

3. Usando el Editor de Propiedades haga lo siguiente:
 - a) Cambie el valor de la propiedad **columns** de **0** a **10**. Esto establece el ancho del campo de texto en caracteres.

- b) Cambie el valor de la propiedad **editable** de **true** a **false** desmarcando la casilla de verificación. Esto hace que el campo de de texto sea de solo lectura.
 - c) Modifique la propiedad **text**, borrando su valor por omisión.
 - d) Cambie el nombre de la variable del campo de texto de **jTextField1** a **campoTextoClave**.
4. Al hacerlo tendremos lo mostrado en la figura 80.

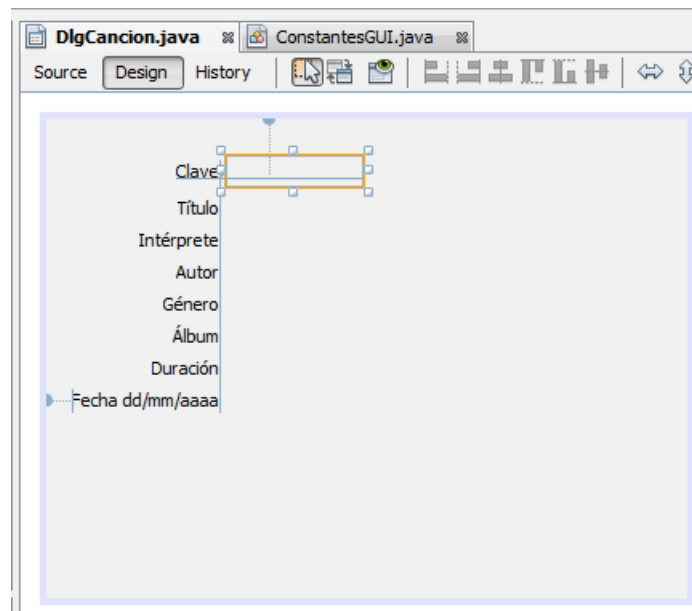


Figura 80

5. Agregue un campo de texto por debajo del campo de texto **campoTextoClave**, arrastrando el icono hasta que la guía de alineación horizontal muestre que la etiqueta está alineada con la etiqueta **Título** y aparezca una guía horizontal en la etiqueta indicando que el campo de texto se encuentra separado de la etiqueta por la distancia horizontal predeterminada y suéltela, figura 81.
6. Usando el Editor de Propiedades, cambie el valor de la propiedad **columns** a **35**, borre el valor de la propiedad **text** y cambie el nombre de la variable del campo de texto de **jTextField1** a **campoTextoTitulo**.
7. Repita los pasos 6 y 7 para agregar un campo de texto por debajo del campo de texto **campoTextoTitulo** y cambie el valor de la propiedad **columns** a **35**, borre el valor de la propiedad **text** y cambie el nombre de la variable del campo de texto de **jTextField1** a **campoTextoInterprete**.

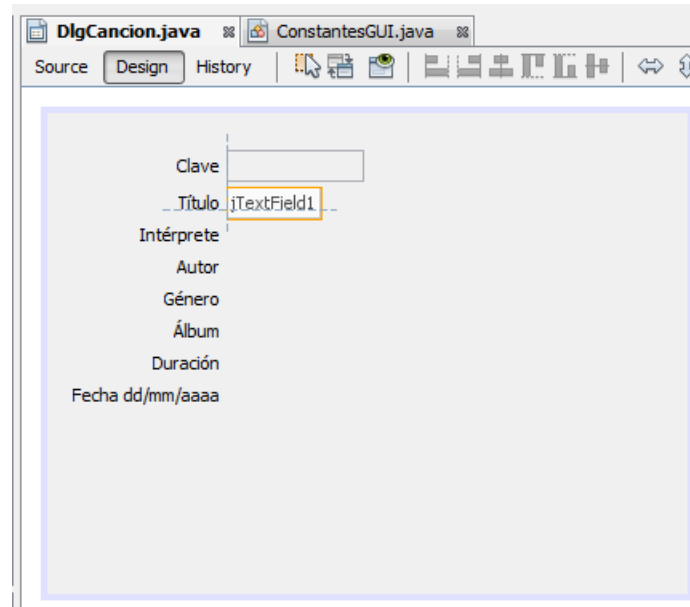


Figura 81

8. Repita los pasos 6 y 7 para agregar un campo de texto por debajo del campo de texto **campoTextoInterprete** y cambie el valor de la propiedad **columns** a **35**, borre el valor de la propiedad **text** y cambie el nombre de la variable del campo de texto de **jTextField1** a **campoTextoAutor**.
9. Repita los pasos 6 y 7 para agregar un campo de texto a un lado de la etiqueta **Álbum** (Note que al lado de la etiqueta **Género** no se agrega un campo de texto, ya que ahí le agregaremos una caja combinada). Cambie el valor de la propiedad **columns** a **35**, borre el valor de la propiedad **text** y cambie el nombre de la variable del campo de texto de **jTextField1** a **campoTextoAlbum**.
10. Repita los pasos 6 y 7 para agregar un campo de texto por debajo del campo de texto **campoTextoDisquera** y cambie el valor de la propiedad **columns** a **5**, borre el valor de la propiedad **text** y cambie el nombre de la variable del campo de texto de **jTextField1** a **campoTextoDuracion**.
11. Repita los pasos 6 y 7 para agregar un campo de texto por debajo del campo de texto **campoTextoDuracion** y cambie el valor de la propiedad **columns** a **10**, borre el valor de la propiedad **text** y cambie el nombre de la variable del campo de texto de **jTextField1** a **campoTextoFecha**.
12. Al hacerlo tendremos lo mostrado en la figura 82.

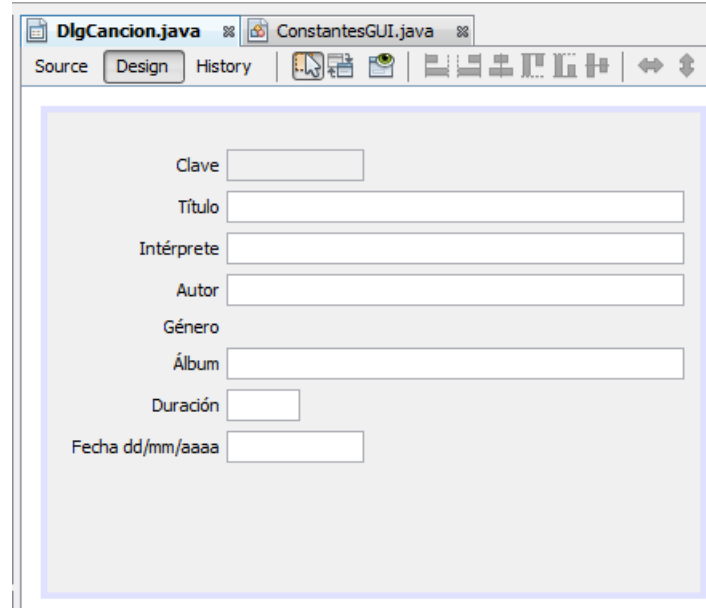


Figura 82

Agregado de una Caja Combinada al Cuadro de Diálogo

A continuación le agregaremos al cuadro de diálogo una caja combinada al lado de la etiqueta **Género**, para seleccionar el género de la canción. El procedimiento es el siguiente:

1. De la Paleta de Componentes seleccione el icono que representa una caja combinada, la componente **JComboBox**, figura 83.

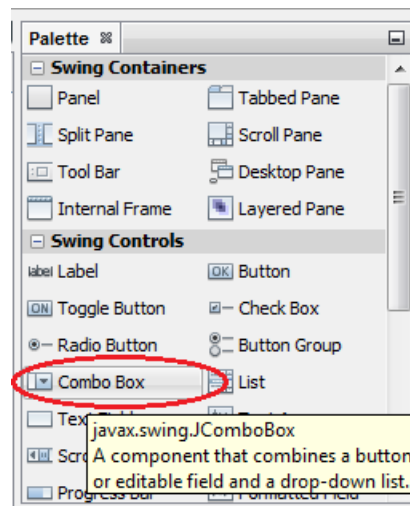


Figura 83

2. Arrastre con el ratón el icono de la caja combinada hasta que la guía de alineación horizontal muestre que el texto de la etiqueta **Género** y la de la caja combinada están alineados y aparezca una guía vertical en la caja combinada indicando que los elementos se encuentran separados por la distancia horizontal entre elementos predeterminada y suéltelo, figura 84.

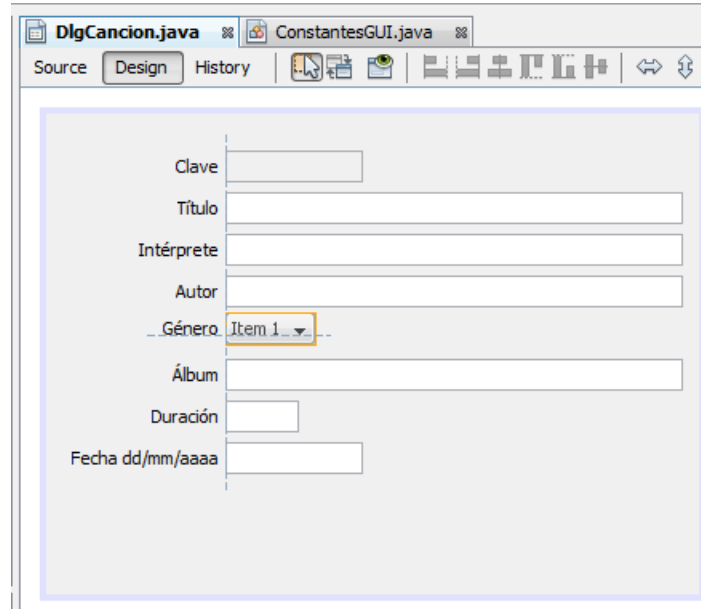


Figura 84

3. Si hacemos clic en el icono **Preview Design** para obtener una vista previa del cuadro de diálogo y en éste hacemos clic en la caja combinada, veremos que la lista de opciones es **Item 1, Item 2, Item 3, Item 4**, figura 85.

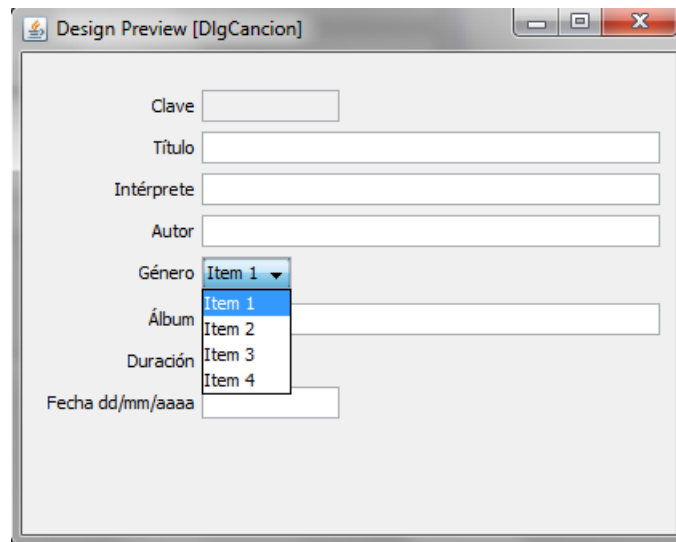


Figura 85

- Podemos modificar esa lista de opciones modificando la propiedad **model** de la caja combinada en el Editor de Propiedades, figura 86. Para ello haga clic en el botón que se encuentra a la derecha de la propiedad **model**. Aparecerá el editor de las opciones para la caja combinada mostrado en la figura 87.

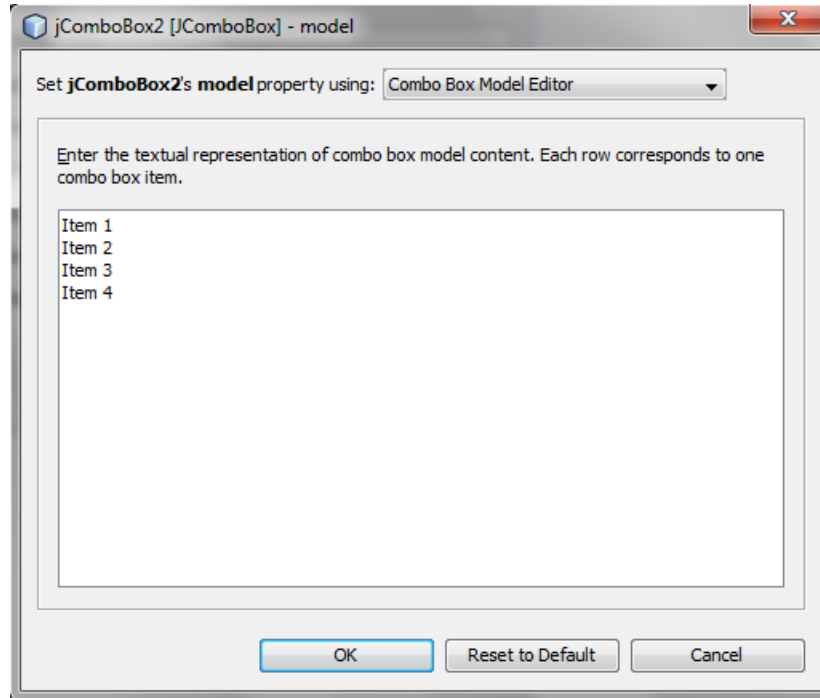


Figura 86

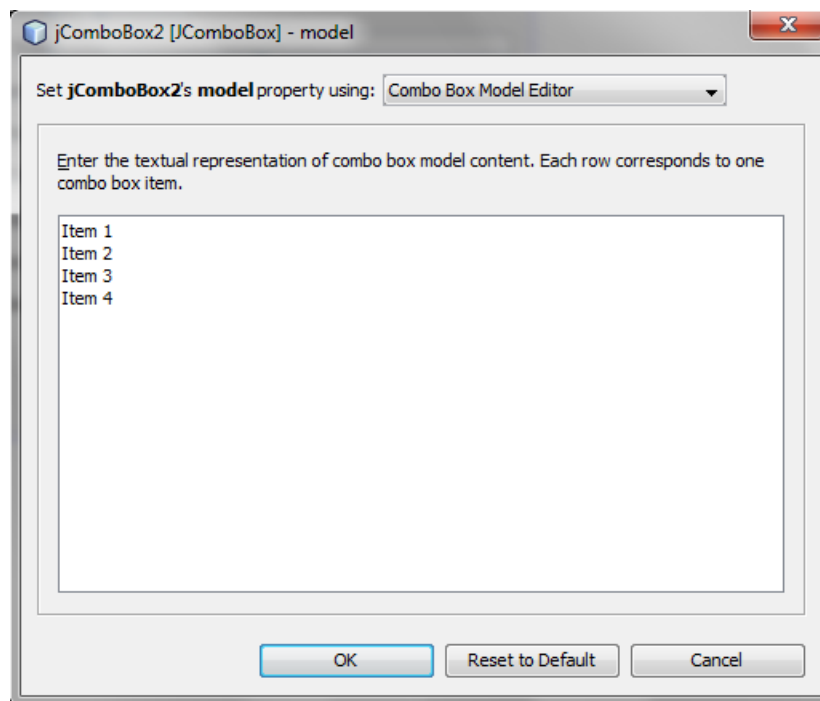


Figura 87

5. Podemos editar la lista que aparece en el área de texto. Hay otras alternativas para establecer la lista de opciones: De un atributo o de la invocación de un método, ambos del tipo `DefaultComboBoxModel`. En este caso usaremos un atributo. Para ello, en el editor de las opciones para la caja combinada mostrado en la figura 87, haremos clic en la caja combinada que aparece en la parte superior, figura 88.

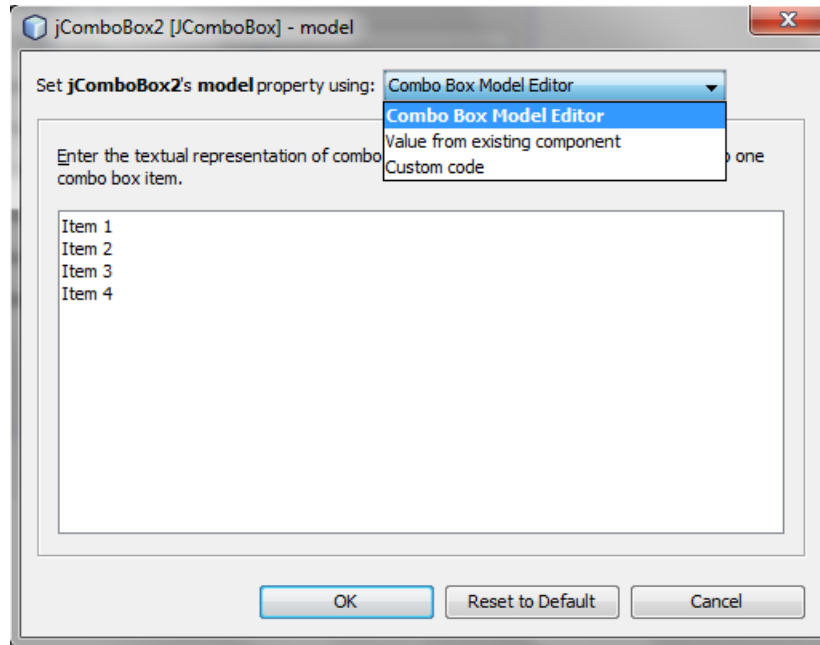


Figura 88

6. De la caja combinada seleccione la opción **Custom code**. Aparecerá el cuadro de diálogo de la figura 89. Haga clic en el botón de radio **Property**, para establecer la propiedad que contiene lista de opciones de la caja combinada.

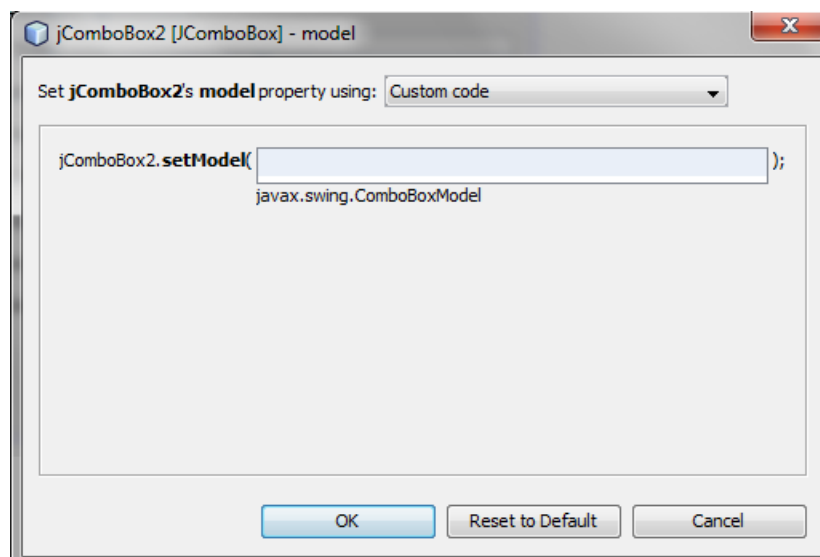


Figura 89

7. En el campo de texto teclee **listaGenerosCanciones** que es un atributo del cuadro de diálogo `DlgCancion` y que fue inicializado mediante el constructor. Este atributo contiene la lista de opciones de la caja combinada, figura 90. Haga clic en el botón **OK**.

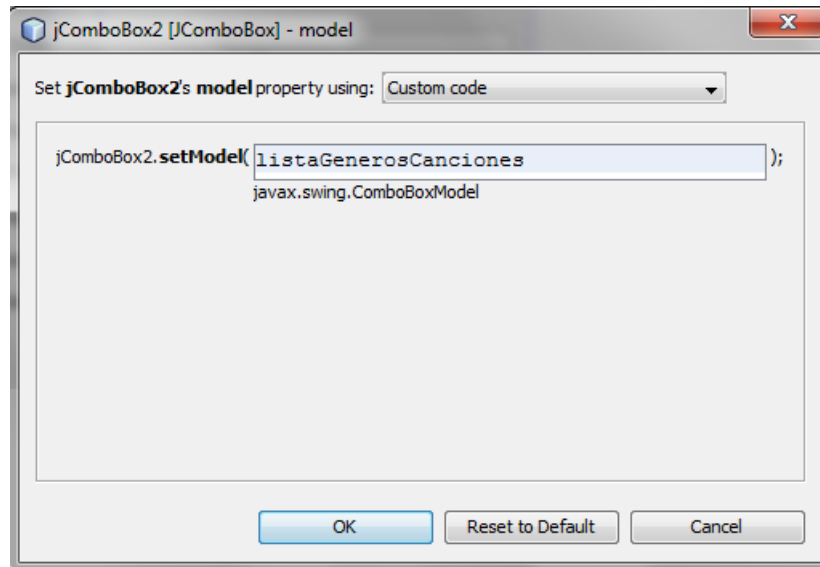


Figura 90

8. Al hacerlo se cierra el cuadro de diálogo y en el Editor de Propiedades la propiedad `model` aparece con el valor **<User Code>**, figura 91.

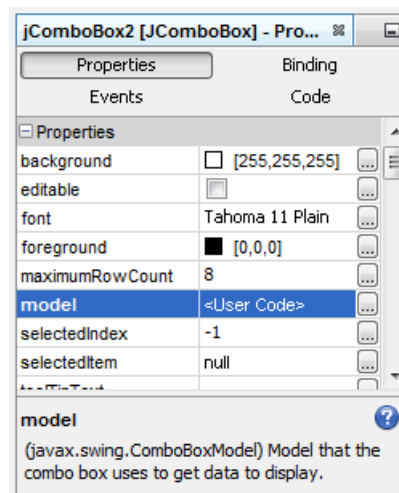


Figura 91

9. Usando el Editor de Propiedades, cambie el nombre de la variable de la caja combinada de **jComboBox1** a **cajaCombinadaGenerosCanciones**. Al hacerlo tendremos lo mostrado en la figura 92.

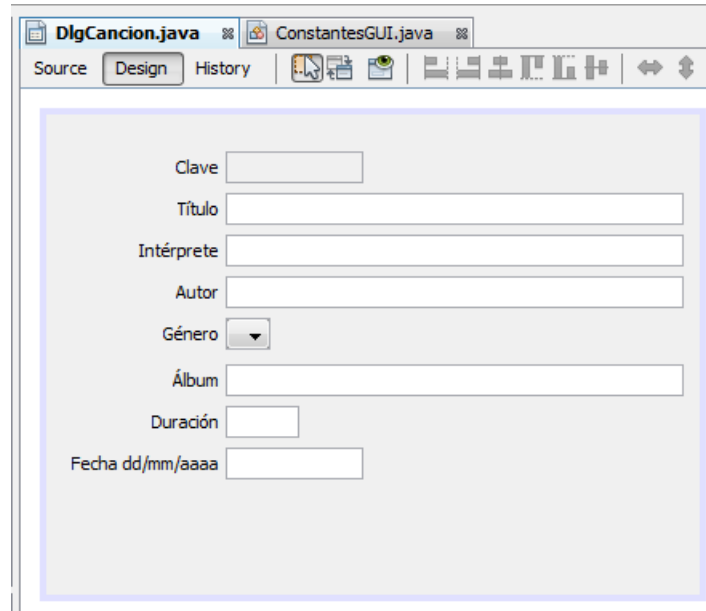


Figura 92

Ajuste del Tamaño del Cuadro de Diálogo

Si en la figura 92 vemos que el cuadro de diálogo se ha llenado por completo y no hay lugar para agregar los botones que permitan aceptar, restablecer los datos y cancelar la operación dada por el cuadro de diálogo; o si el espacio entre la orilla izquierda y la etiqueta inferior es mayor que la distancia entre la orilla derecha y los campos de texto de mayor longitud, podemos modificar el tamaño del cuadro de diálogo. Para ello haremos lo siguiente:

1. Coloque el cursor en la orilla derecha del cuadro y arrastre el ratón a la derecha hasta igualar los márgenes izquierdo y derecho.
2. Coloque el cursor en la orilla inferior del cuadro y arrastre el ratón hacia abajo para dejar un espacio de unos tres centímetros.

Agregado de Botones al Cuadro de Diálogo

Enseguida agregaremos botones que permitan aceptar, restablecer los datos y cancelar la operación dada por el cuadro de diálogo. El procedimiento es el siguiente:

1. De la Paleta de Componentes seleccione el icono que representa un botón, la componente **JButton**, figura 93.
2. Arrastre con el ratón el icono del botón a una posición por debajo de la etiqueta **Fecha** y hasta que la guía de alineación vertical muestre que el botón y la etiqueta están alineados verticalmente y suéltelo, figura 94.

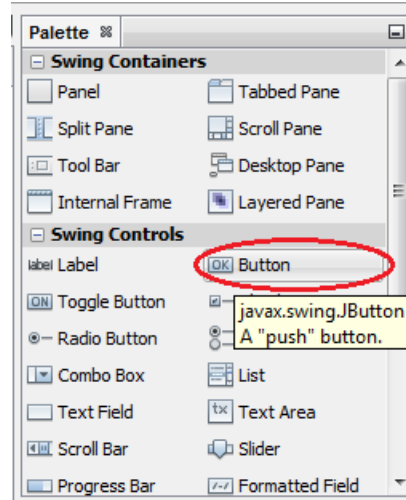


Figura 93

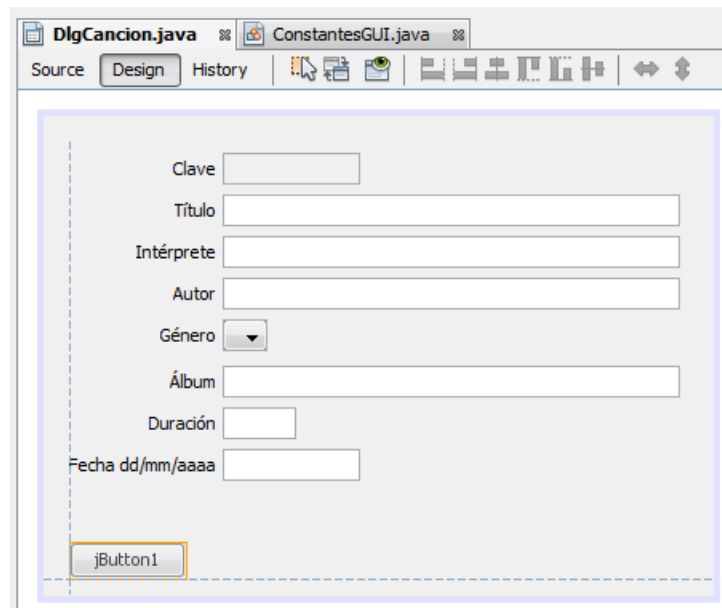


Figura 94

3. Usando el Editor de Propiedades de las componentes, cambie el valor de la propiedad **text** de **jButton1** a **Aceptar**. Esto establece el texto desplegado en el botón.
4. Usando el Editor de Propiedades de las componentes, cambie el nombre de la variable del botón de **jButton1** a **botonAceptar**.
5. Al hacerlo tendremos lo mostrado en la figura 95.

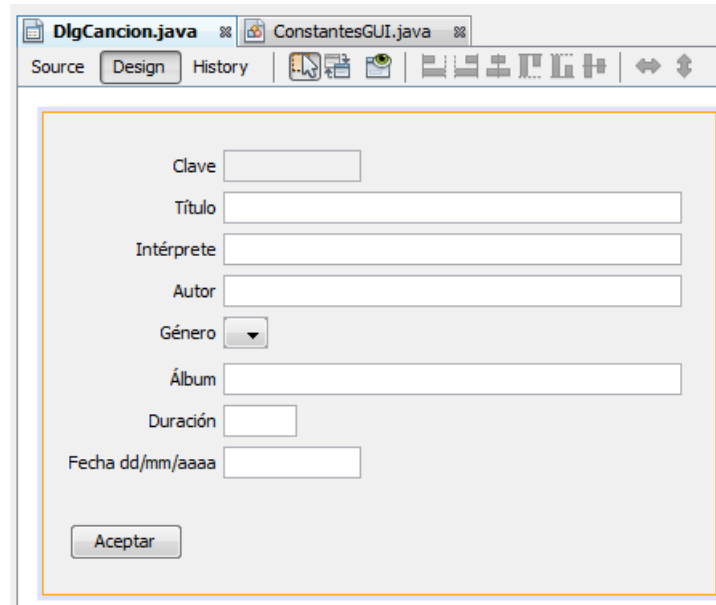


Figura 95

6. Agregue un segundo botón al cuadro de diálogo a la derecha del primer botón y por debajo del campo de texto **campoTextoFecha** arrastrando su icono hasta que la guía de alineación horizontal indique que los botones están alineados horizontalmente y la guía de alineación vertical indique que el botón está alineado con el campo de texto, figura 96.

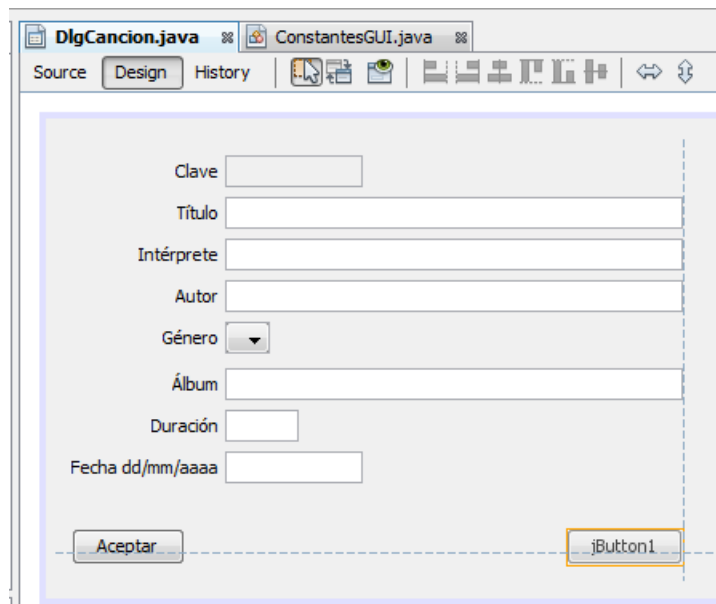


Figura 96

7. Usando el Editor de Propiedades de las componentes, cambie el valor de la propiedad **text** de **jButton1** a **Cancelar** y el nombre de la variable del botón de **jButton1** a **botonCancelar**.

8. Agregue un tercer botón al cuadro de diálogo entre los dos botones anteriores arrastrando su icono hasta que la guía de alineación horizontal indique que los botones están alineados horizontalmente y el botón esté centrado horizontalmente en el cuadro de diálogo.
9. Usando el Editor de Propiedades de las componentes, cambie el valor de la propiedad **text** de **jButton1** a **Restaurar** y el nombre de la variable del botón de **jButton1** a **botonRestaurar**.
10. Al hacerlo aparecerá lo mostrado en la figura 97.

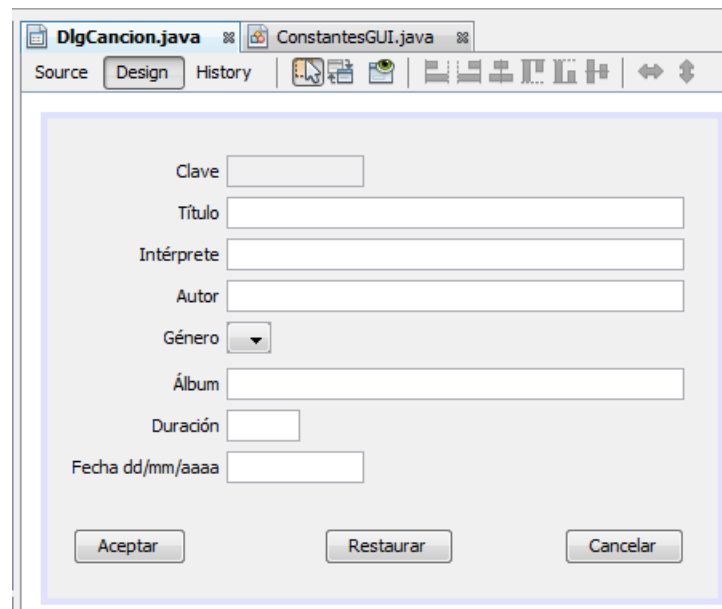


Figura 97

11. Podemos visualizar el aspecto del cuadro de diálogo **DlgCancion** haciendo clic en el icono **Preview Design** que se encuentra encima de la Ventana de Edición del Código, figura 35.
12. Obtendremos lo mostrado en la figura 98.

Establecimiento de los Métodos Oyentes de los Eventos Generados por los Botones.

Al igual que con los elementos de menú, si hacemos clic sobre botón, éste genera un evento del tipo **ActionEvent**. Si queremos que nuestro programa reaccione a ese evento, debemos de establecer un método oyente del tipo **actionPerformed()** en el contenedor de ese botón, que normalmente es una ventana o un cuadro de diálogo. Para establecer un método oyente del botón **botonAceptar** del cuadro de diálogo **DlgCancion**, seguiremos el siguiente procedimiento:

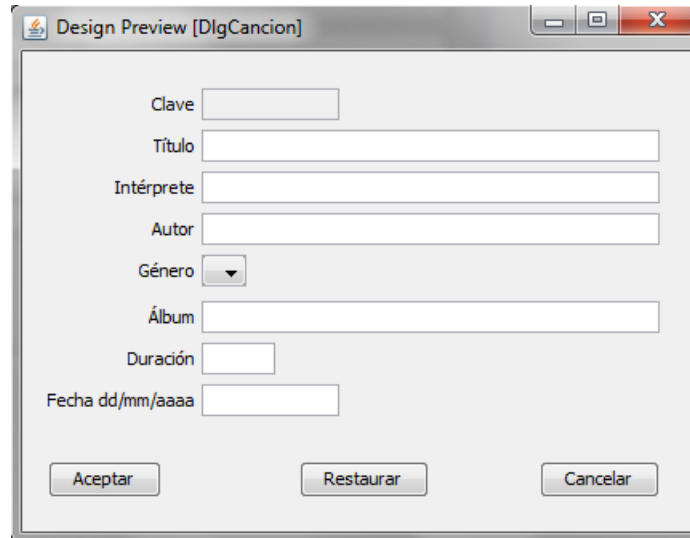


Figura 98

1. En la Vista de Diseño de la Ventana de Edición de Código haga clic sobre el botón **Continuar**. También puede hacer clic sobre el nodo que representa a ese botón en el panel Navegador.
2. Haga clic en el selector **Events** del Editor de Propiedades. Aparecerá la lista de eventos que puede generar el botón, figura 99.

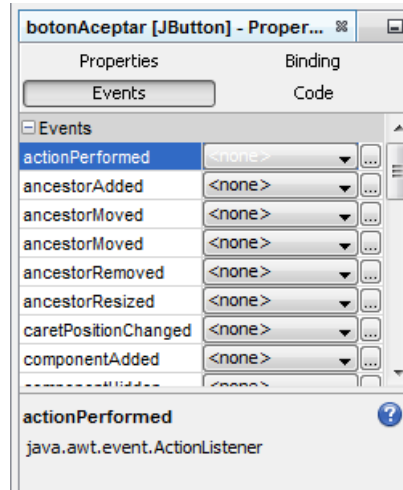


Figura 99

3. Haga clic en la caja combinada a la derecha del evento **actionPerformed** y seleccione la opción **botonAceptarActionPerformed** que será el nombre sugerido por NetBeans para el método oyente del tipo **actionPerformed()**, figura 100.

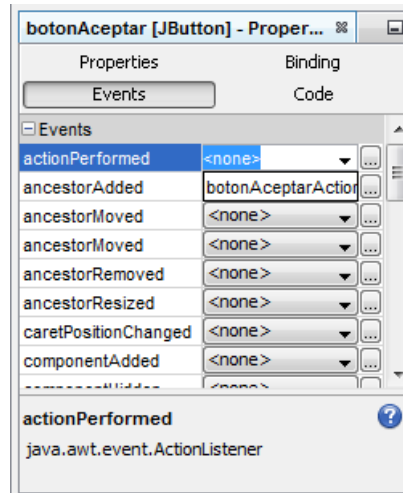


Figura 100

- NetBeans genera el esqueleto del método y nos lo muestra en la Vista de Código Fuente, como se muestra en la figura 101. En el cuerpo de ese método colocaremos el código que queramos que se ejecute cuando se haga clic en el botón **botonAceptar** del cuadro de diálogo.

```

201
202     private void botonAceptarActionPerformed(java.awt.event.ActionEvent evt) {
203         // TODO add your handling code here:
204     }
205

```

Figura 100

- Repita los pasos 1 a 4 para agregar un método oyente para cada uno de los demás botones del cuadro de diálogo: **botonRestaurar** y **botonCancelar**.

Edición del constructor del Cuadro de Diálogo

El cuadro de diálogo que estamos creando se va a utilizar para capturar, editar o desplegar los datos de canción. Modifique el constructor del cuadro de diálogo para que reciba o regrese los datos de una canción y para que modifique su apariencia dependiendo de la operación deseada, de la siguiente manera.

```

/**
 * Constructor que establece las características del cuadro de diálogo
 * y la operación a realizar con él
 * @param parent Ventana sobre la que aparecerá el cuadro de diálogo
 * @param title Título del cuadro de diálogo
 * @param modal true si permite acceder fuera de los límites del cuadro
 * de diálogo, false en caso contrario
 * @param cancion Canción a capturar, editar o desplegar
 * @param listaGenerosCanciones Lista de los generos de una canción
 * @param operacion Operación a realizar en el cuadro de diálogo:
 *         AGREGAR = 0, ACTUALIZAR = 1, ELIMINAR = 2, DESPLEGAR = 3;
 * @param respuesta Boton presionado al salir de los cuadros de

```

```

* diálogos: ACEPTAR = "Aceptar", CANCELAR = "Cancelar".
*/
public DlgCancion(java.awt.Frame parent, String title, boolean modal,
                  Cancion cancion,
                  DefaultComboBoxModel listaGenerosCanciones,
                  int operacion, StringBuffer respuesta) {
    super(parent, title, modal);
    this.cancion = cancion;
    this.listaGenerosCanciones = listaGenerosCanciones;
    this.operacion = operacion;
    this.respuesta = respuesta;

    initComponents();

    // Modifica el título del botón botonAceptar y establece si los
    // botones botonRestaurar y botonCancelar estarán habilitados.
    // Si la operación es agregar
    if(operacion == ConstantesGUI.AGREGAR)
        botonAceptar.setText("Guardar");
    // Si la operación es actualizar
    else if(operacion == ConstantesGUI.ACTUALIZAR)
        botonAceptar.setText("Actualizar");
    // Si la operación es eliminar
    else if(operacion == ConstantesGUI.ELIMINAR) {
        botonAceptar.setText("Eliminar");
        botonRestaurar.setEnabled(false);
    }
    // Si la operación es desplegar
    else if(operacion == ConstantesGUI.DESPLEGAR) {
        botonAceptar.setText("Continuar");
        botonRestaurar.setEnabled(false);
        botonCancelar.setEnabled(false);
    }

    // Despliega la clave de la canción
    campoTextoClave.setText(cancion.getClave());

    // Si la operación es de actualizar, eliminar o desplegar,
    if((operacion == ConstantesGUI.ELIMINAR) ||
        (operacion == ConstantesGUI.ACTUALIZAR) ||
        (operacion == ConstantesGUI.DESPLEGAR)) {
        // despliega el resto de los datos de la canción
        campoTextoTitulo.setText(cancion.getTitulo());
        campoTextoInterprete.setText(cancion.getInterprete());
        campoTextoAutor.setText(cancion.getAutor());
        cajaCombinadaGenerosCanciones.setSelectedItem(cancion.getGenero());
        campoTextoAlbum.setText(cancion.getAlbum());
        campoTextoDuracion.setText(new Integer(cancion.getDuracion())
            .toString());
        campoTextoFecha.setText(cancion.getFecha().toString());
    }

    // Si la operación es de eliminar o desplegar
    if((operacion == ConstantesGUI.ELIMINAR) ||
        (operacion == ConstantesGUI.DESPLEGAR)) {
        // hace los campos de texto de sólo lectura
        campoTextoTitulo.setEditable(false);
    }

```

```

        campoTextoInterprete.setEditable(false);
        campoTextoAutor.setEditable(false);
        cajaCombinadaGenerosCanciones.setEnabled(false);
        campoTextoAlbum.setEditable(false);
        campoTextoDuracion.setEditable(false);
        campoTextoFecha.setEditable(false);
    }

    // Establece el valor por omisión para respuesta, por si se cierra el
    // cuadro de diálogo presionando el botón cerrar o el botón cancelar
    respuesta.append(ConstantesGUI.CANCELAR);

    // centra el cuadro de dialogo sobre la ventana de la aplicación
    centraCuadroDialogo(parent);

    // Muestra el cuadro de diálogo
    setVisible(true);
}

```

Centrado del Cuadro de Diálogo sobre la Ventana de la Aplicación

Note que al final del constructor del cuadro de diálogo, se invoca al método `centraCuadroDialogo()` para centrar el cuadro de diálogo sobre la ventana de la aplicación. Coloque el código de su método después del constructor.

```

/**
 * Este método centra el cuadro de dialogo sobre la ventana de la
 * aplicación.
 * @param parent Ventana sobre la que aparecerá el cuadro de diálogo
 */
private void centraCuadroDialogo(java.awt.Frame parent) {
    // Obtiene el tamaño y posición de la ventana de la aplicación
    Dimension frameSize = parent.getSize();
    Point loc = parent.getLocation();

    // Obtiene el tamaño del cuadro de diálogo
    Dimension dlgSize = getPreferredSize();

    // Centra el cuadro de diálogo sobre la ventana padre
    setLocation( (frameSize.width - dlgSize.width) / 2 + loc.x,
                 (frameSize.height - dlgSize.height) / 2 + loc.y);
}

```

Edición de los Métodos Oyentes de los botones del Cuadro de Diálogo

Este cuadro de diálogo tiene tres botones: **botonAceptar**, **botonRestaurar** y **botonCancelar**.

El método oyente del botón **botonAceptar** hace lo siguiente:

1. Si la operación es Agregar o Actualizar, toma los valores capturados en los campos de texto y almacénalos en el parámetro `cancion`.
2. Establece que se presionó el botón **botonAceptar** para cerrar el cuadro de diálogo.
3. Cierra el cuadro de diálogo.

El código del método oyente del botón **botonAceptar** es el siguiente:

```
/**
 * Método oyente del botón botonAceptar
 *
 * @param evt Evento al que escucha
 */
private void botonAceptarActionPerformed(java.awt.event.ActionEvent evt) {
    // Si la operación es Agregar o Actualizar
    if((operacion == ConstantesGUI.AGREGAR) ||
        (operacion == ConstantesGUI.ACTUALIZAR)) {
        // Toma los valores capturados en los campos de texto y en la caja
        // combinada y almacénalos en el parámetro cancion.
        cancion.setTitulo(campoTextoTitulo.getText());
        cancion.setInterprete(campoTextoInterprete.getText());
        cancion.setAutor(campoTextoAutor.getText());
        cancion.setGenero(
            (Genero)cajaCombinadaGenerosCanciones.getSelectedItem());
        cancion.setAlbum(campoTextoAlbum.getText());
        cancion.setDuracion(Integer.parseInt(campoTextoDuracion.getText()));
        cancion.setFecha(new Fecha(campoTextoFecha.getText()));
    }

    // Borra el contenido de respuesta
    respuesta.delete(0, respuesta.length());

    // Establece que se presionó el botón botonAceptar
    respuesta.append(ConstantesGUI.ACCEPTAR);

    // Destruye el cuadro de diálogo
    dispose();
}
```

El método oyente del botón **botonRestaurar** restaura el contenido de los campos de texto a su valor original. Si la operación es que en este caso son cadenas vacías. El código del método oyente del botón **botonRestaurar** es el siguiente:

```
/**
 * Método oyente del botón botonRestaurar
 *
 * @param evt Evento al que escucha
 */
```

```

private void botonRestaurarActionPerformed(java.awt.event.ActionEvent evt){
    // Restaura el contenido de los campos de texto a su valor original
    // Si la operación es Agregar
    if(operacion == ConstantesGUI.AGREGAR) {
        campoTextoTitulo.setText("");
        campoTextoInterprete.setText("");
        campoTextoAutor.setText("");
        cajaCombinadaGenerosCanciones.setSelectedIndex(0);
        campoTextoAlbum.setText("");
        campoTextoDuracion.setText("");
        campoTextoFecha.setText("");
    }

    // Si la operación es Actualizar
    if(operacion == ConstantesGUI.ACTUALIZAR) {
        campoTextoTitulo.setText(cancion.getTitulo());
        campoTextoInterprete.setText(cancion.getInterprete());
        campoTextoAutor.setText(cancion.getAutor());
        cajaCombinadaGenerosCanciones.setSelectedItem(cancion.getGenero());
        campoTextoAlbum.setText(cancion.getAlbum());
        campoTextoDuracion.setText(new Integer(cancion.getDuracion())
            .toString());
        campoTextoFecha.setText(cancion.getFecha().toString());
    }
}

```

El método oyente del botón **botonCancelar** cierra el cuadro de diálogo.

El código del método oyente del botón **botonCancelar** es el siguiente:

```

/**
 * Método oyente del botón botonCancelar
 *
 * @param evt Evento al que escucha
 */
private void botonCancelarActionPerformed(java.awt.event.ActionEvent evt) {
    // Destruye el cuadro de diálogo
    dispose();
}

```