

Documentación: Gestor de una Biblioteca

(Eclipse, MySQL, JDK 1.8)



Programación 2

**Ingeniería en Sistemas Informáticos y de
Computación**

PRÓLOGO

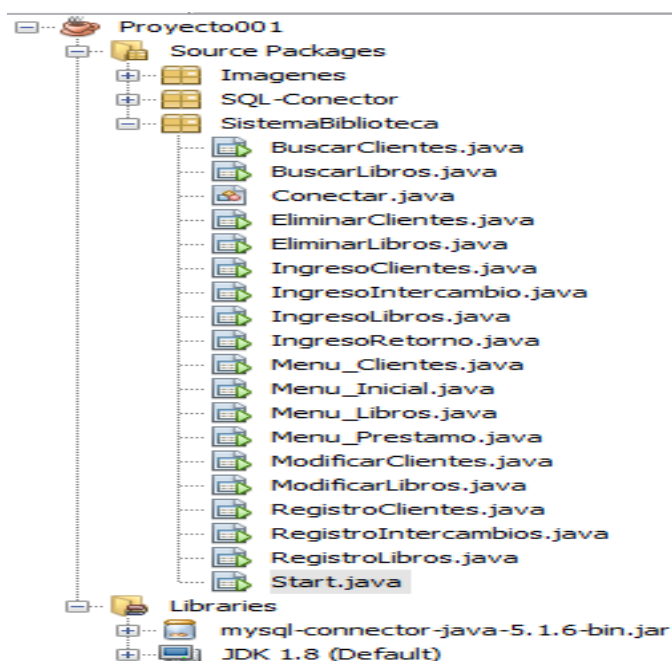
El Objetivo de esta práctica es el poder implementar los conocimientos impartidos a lo largo del semestre en la materia de Programación 2, mediante la creación de un sistema que permitirá el manejo y control de la entrada y salida de libros en una supuesta biblioteca.

Esta práctica pretende servir de ejemplo como aplicación de la Programación Orientada a Objetos. Decimos que estamos realizando POO cuando utilizamos clases, objetos y empleamos la herencia.

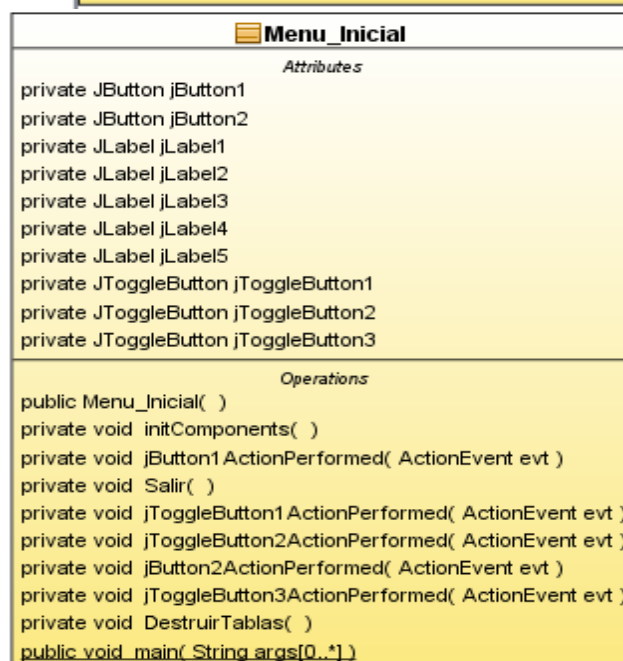
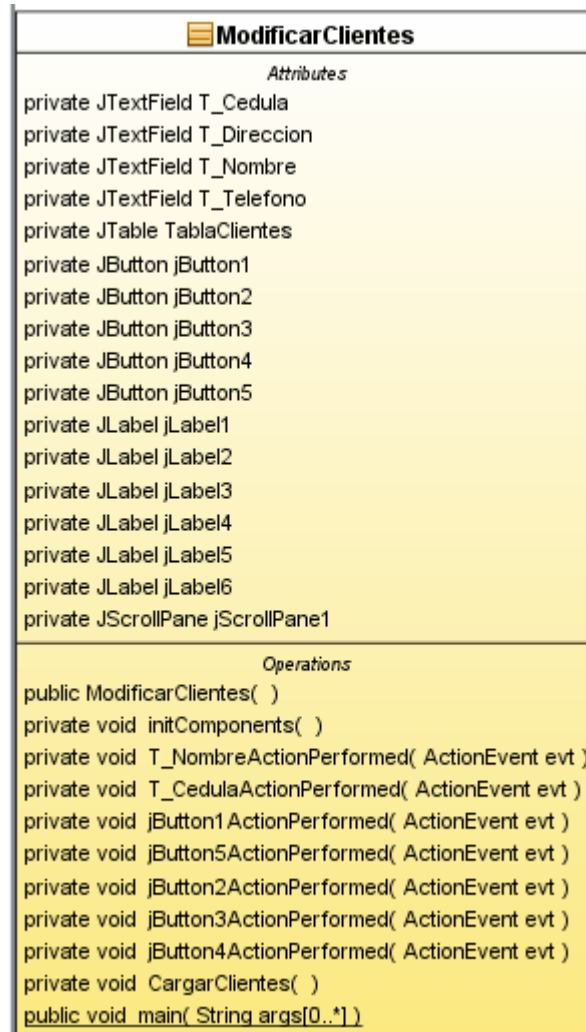
ESPECIFICACIÓN DE LA PRÁCTICA:

El programa consistirá en la construcción de una biblioteca orientada a objetos para una biblioteca N. Dicha biblioteca constará de simulación de préstamos (entrada y salida de libros), manejo de usuarios y de libros (nuevo, búsqueda, modificación y eliminación). Mediante el uso en conjunto de Netbeans con el servidor, que en este caso en específico se trabajará con XAMPP, para almacenar la información que se vaya trabajando.

CLASES Y ESTRUCTURA DEL PROGRAMA:



UML'S GENERADOS APARTIR DE SU CÓDIGO:



ModificarLibros

Attributes

```
private JTextField T_Autor  
private JTextField T_Avalúo  
private JTextField T_Año  
private JTextField T_Existencia  
private JTextField T_Libro  
private JTable Tb_Libros  
private JButton jButton1  
private JButton jButton2  
private JButton jButton3  
private JButton jButton4  
private JButton jButton5  
private JButton jButton6  
private JLabel jLabel1  
private JLabel jLabel2  
private JLabel jLabel3  
private JLabel jLabel4  
private JLabel jLabel5  
private JLabel jLabel6  
private JLabel jLabel7  
private JScrollPane jScrollPane1
```

Operations


```
public ModificarLibros( )  
private void initComponents( )  
private void T_LibroActionPerformed( ActionEvent evt )  
private void T_AutorActionPerformed( ActionEvent evt )  
private void T_AñoActionPerformed( ActionEvent evt )  
private void jButton4ActionPerformed( ActionEvent evt )  
private void jButton5ActionPerformed( ActionEvent evt )  
private void jButton6ActionPerformed( ActionEvent evt )  
private void jButton1ActionPerformed( ActionEvent evt )  
private void jButton2ActionPerformed( ActionEvent evt )  
private void jButton3ActionPerformed( ActionEvent evt )  
private void T_AvalúoActionPerformed( ActionEvent evt )  
private void CargarLibros( )  
public void main( String args[0..*] )
```


EliminarClientes
<p><i>Atributos</i></p> <pre>private JTable TablaClientes private JButton jButton1 private JButton jButton2 private JLabel jLabel1 private JScrollPane jScrollPane1</pre>
<p><i>Operations</i></p> <pre>public EliminarClientes() private void CargarClientes() private void initComponents() private void jButton1ActionPerformed(ActionEvent evt) private void jButton2ActionPerformed(ActionEvent evt) public void main(String args[0..*])</pre>


Start
<p><i>Atributos</i></p> <pre>private JButton jButton1 private JButton jButton2 private JLabel jLabel1 private JLabel jLabel2 private JLabel jLabel3 private JToggleButton jToggleButton1</pre>
<p><i>Operations</i></p> <pre>public Start() private void initComponents() private void jToggleButton1ActionPerformed(ActionEvent evt) private void jButton1ActionPerformed(ActionEvent evt) private void jButton2ActionPerformed(ActionEvent evt) private void abrirDocumentacion() public void main(String args[0..*])</pre>


IngresoInter cambi
<p><i>Atributos</i></p> <pre>private String fecha package int libroseleccionado private JTextField T_Unidades private JTable Tb_Clientes private JTable Tb_Libros private JButton jButton1 private JButton jButton2 private JLabel jLabel1 private JLabel jLabel3 private JLabel jLabel4 private JLabel jLabel5 private JPanel jPanel1 private JScrollPane jScrollPane1 private JScrollPane jScrollPane2</pre>
<p><i>Operations</i></p> <pre>public IngresoInter cambio() private void initComponents() private void jButton2ActionPerformed(ActionEvent evt) private void jButton1ActionPerformed(ActionEvent evt) private void CargarLibros() private void CargarClientes() private void getFecha() private void RegistrarInter cambio() private int ActuUnidades() private int getSelectedLibro() private int getSelectedCliente() public void main(String args[0..*])</pre>

IngresoLibro:
<p><i>Atributos</i></p> <pre>private JTextField T_autor private JTextField T_avaluo private JTextField T_año private JTextField T_existencia private JTextField T_titulo private JTable TablaLibros private JButton jButton1 private JButton jButton2 private JButton jButton3 private JLabel jLabel1 private JLabel jLabel10 private JLabel jLabel2 private JLabel jLabel3 private JLabel jLabel4 private JLabel jLabel5 private JLabel jLabel6 private JLabel jLabel7 private JLabel jLabel8 private JLabel jLabel9 private JScrollPane jScrollPane1</pre>
<p><i>Operations</i></p> <pre>public IngresoLibros() private void initComponents() private void T_tituloActionPerformed(ActionEvent evt) private void jButton1ActionPerformed(ActionEvent evt) private void jButton2ActionPerformed(ActionEvent evt) public void CargarLibros() private void jButton3ActionPerformed(ActionEvent evt) public void main(String args[0..*])</pre>

 Menu_Clientes
<i>Atributos</i>
<pre>private JLabel jLabel1 private JLabel jLabel2 private JLabel jLabel3 private JLabel jLabel4 private JLabel jLabel5 private JLabel jLabel6 private JLabel jLabel7 private JToggleButton jToggleButton1 private JToggleButton jToggleButton2 private JToggleButton jToggleButton3 private JToggleButton jToggleButton4 private JToggleButton jToggleButton5 private JToggleButton jToggleButton6</pre>
<i>Operaciones</i>
<pre>public Menu_Clientes() private void initComponents() private void jToggleButton1ActionPerformed(ActionEvent evt) private void jToggleButton5ActionPerformed(ActionEvent evt) private void jToggleButton6ActionPerformed(ActionEvent evt) private void jToggleButton4ActionPerformed(ActionEvent evt) private void jToggleButton3ActionPerformed(ActionEvent evt) private void jToggleButton2ActionPerformed(ActionEvent evt) public void main(String args[0..*])</pre>

 IngresoRetorno
<i>Atributos</i>
<pre>private String fecha private JTextField T_Libros private JTable Tb_Intercambios private JButton jButton1 private JButton jButton2 private JLabel jLabel1 private JLabel jLabel2 private JLabel jLabel3 private JScrollPane jScrollPane1</pre>
<i>Operaciones</i>
<pre>public IngresoRetorno() private void initComponents() private void T_LibrosActionPerformed(ActionEvent evt) private void jButton1ActionPerformed(ActionEvent evt) private void jButton2ActionPerformed(ActionEvent evt) private void GuardarRetorno() private void ActuarUnidades() private int getSelectedIntercambio() private int getLibrosRe() private void getFecha() public void CargarIntercambiosSalida() public void main(String args[0..*])</pre>

 Menu_Prestamo
<i>Atributos</i>
<pre>private JButton jButton1 private JButton jButton2 private JLabel jLabel1 private JLabel jLabel2 private JLabel jLabel3 private JLabel jLabel4 private JLabel jLabel5 private JToggleButton jToggleButton2 private JToggleButton jToggleButton3</pre>
<i>Operaciones</i>
<pre>public Menu_Prestamo() private void initComponents() private void jToggleButton3ActionPerformed(ActionEvent evt) private void jButton1ActionPerformed(ActionEvent evt) private void jToggleButton2ActionPerformed(ActionEvent evt) private void jButton2ActionPerformed(ActionEvent evt) public void main(String args[0..*])</pre>

 EliminarLibro
<i>Atributos</i>
<pre>private JTable Tb_Libros private JButton jButton1 private JButton jButton2 private JLabel jLabel1 private JLabel jLabel2 private JScrollPane jScrollPane1</pre>
<i>Operaciones</i>
<pre>public EliminarLibros() private void initComponents() private void jButton1ActionPerformed(ActionEvent evt) private void jButton2ActionPerformed(ActionEvent evt) private void CargarLibros() private void Eliminar() private int getSelección() public void main(String args[0..*])</pre>



RegistroIntercambio:

Attributes

```
private JButton jButton1
private JButton jButton2
private JButton jButton3
private JButton jButton4
private JLabel jLabel1
private JLabel jLabel2
private JLabel jLabel3
private JLabel jLabel4
private JLabel jLabel5
private JLabel jLabel6
private JLabel jLabel7
private JScrollPane jScrollPane1
private JTable jTable1
```

Operations

```
public RegistroIntercambios( )
private void initComponents( )
private void jButton1ActionPerformed( ActionEvent evt )
private void jButton3ActionPerformed( ActionEvent evt )
private void jButton4ActionPerformed( ActionEvent evt )
private void jButton2ActionPerformed( ActionEvent evt )
public void CargarIntercambiosTodos( )
public int BuscarcodCliente( int indice )
public int BuscarcodLibro( int indice )
public void CargarIntercambiosSalida( )
public void CargarIntercambiosEntrada( )
public void main( String args[0..*] )
```

RegistroClientes

Attributes

```
private JTable TablaClientes
private JButton jButton1
private JLabel jLabel1
private JLabel jLabel2
private JLabel jLabel3
private JScrollPane jScrollPane1
```

Operations

```
public RegistroClientes( )
private void initComponents( )
private void jButton1ActionPerformed( ActionEvent evt )
private void CargarClientes( )
public void main( String args[0..*] )
```

IngresoClientes

Attributes

```
private JTable TablaClientes
private JTextField cedula_cli
private JTextField direccion_cli
private JButton jButton1
private JButton jButton2
private JButton jButton3
private JLabel jLabel1
private JLabel jLabel2
private JLabel jLabel3
private JLabel jLabel4
private JLabel jLabel5
private JLabel jLabel6
private JLabel jLabel7
private JLabel jLabel8
private JScrollPane jScrollPane1
private JTextField nombre_cli
private JTextField telefono_cli
```

Operations

```
public IngresoClientes( )
private void initComponents( )
private void nombre_cliActionPerformed( ActionEvent evt )
private void cedula_cliActionPerformed( ActionEvent evt )
private void direccion_cliActionPerformed( ActionEvent evt )
private void jButton1ActionPerformed( ActionEvent evt )
private void CargarClientes( )
private void jButton3ActionPerformed( ActionEvent evt )
private void jButton2ActionPerformed( ActionEvent evt )
public void main( String args[0..*] )
```

BuscarLibros

Attributes

```
private String clavetitulo
private JTextField T_clave
private JTable TablaBusqueda
private JLabel jLabel1
private JLabel jLabel2
private JScrollPane jScrollPane1
private JToggleButton jToggleButton1
private JToggleButton jToggleButton2
```

Operations

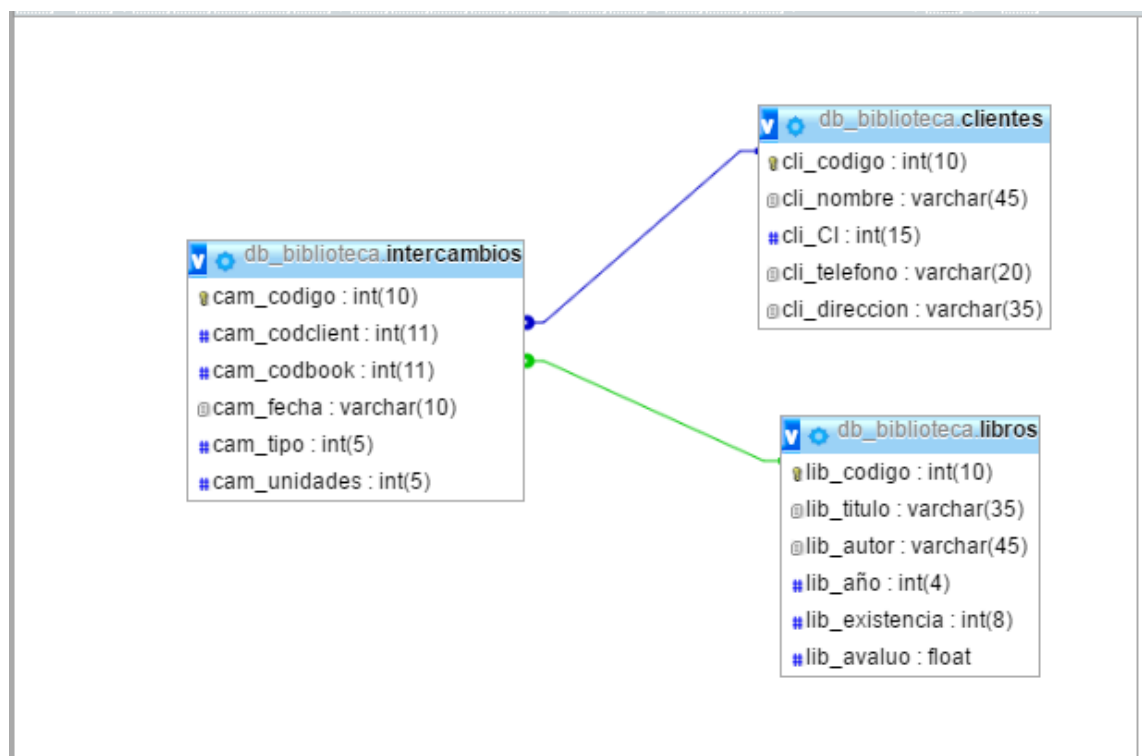
```
public BuscarLibros( )
private void initComponents( )
private void T_claveActionPerformed( ActionEvent evt )
private void jToggleButton1ActionPerformed( ActionEvent evt )
public void BuscarCoincidencias( )
public int BuscarExistencia( int indice )
public String BuscarTitulo( int indice )
private void jToggleButton2ActionPerformed( ActionEvent evt )
public void main( String args[0..*] )
```

Menu_Libros
<p>Atributos</p> <pre>private JButton jButton1 private JButton jButton2 private JButton jButton3 private JButton jButton4 private JButton jButton5 private JButton jButton6 private JLabel jLabel1 private JLabel jLabel2 private JLabel jLabel3 private JLabel jLabel4 private JLabel jLabel5 private JLabel jLabel6 private JLabel jLabel7</pre>
<p>Operaciones</p> <pre>public Menu_Libros() private void initComponents() private void jButton1ActionPerformed(ActionEvent evt) private void jButton5ActionPerformed(ActionEvent evt) private void jButton6ActionPerformed(ActionEvent evt) private void jButton3ActionPerformed(ActionEvent evt) private void jButton4ActionPerformed(ActionEvent evt) private void jButton2ActionPerformed(ActionEvent evt) public void main(String args[])</pre>

BuscarClientes
<p>Atributos</p> <pre>private String clave private JTextField T_Buscar private JTable Tb_Clientes private JButton jButton1 private JButton jButton2 private JLabel jLabel1 private JLabel jLabel2 private JScrollPane jScrollPane1</pre>
<p>Operaciones</p> <pre>public BuscarClientes() private void initComponents() private void T_BuscarActionPerformed(ActionEvent evt) private void jButton1ActionPerformed(ActionEvent evt) private void jButton2ActionPerformed(ActionEvent evt) private void BuscarCoincidencias() public String BuscarNombre(int indice) public void main(String args[])</pre>

RegistroLibro:
<p>Atributos</p> <pre>package ResultSet res private JLabel jLabel1 private JLabel jLabel2 private JScrollPane jScrollPane1 private JTable jTable1 private JToggleButton jButton1</pre>
<p>Operaciones</p> <pre>public RegistroLibros() public void CargarLibros() private void initComponents() private void jButton1ActionPerformed(ActionEvent evt) public void main(String args[])</pre>

DIAGRAMA DE TABLAS EN EL SERVIDOR:



DETALLES DE LAS CLASES:

Clase Start:



Es la clase que se encarga de inicial el sistema de la Biblioteca, crear las tablas de no existir y enlazarlas. Además de eso proporciona dos opciones, el ver los integrantes de grupo y abrir la documentación del programa.

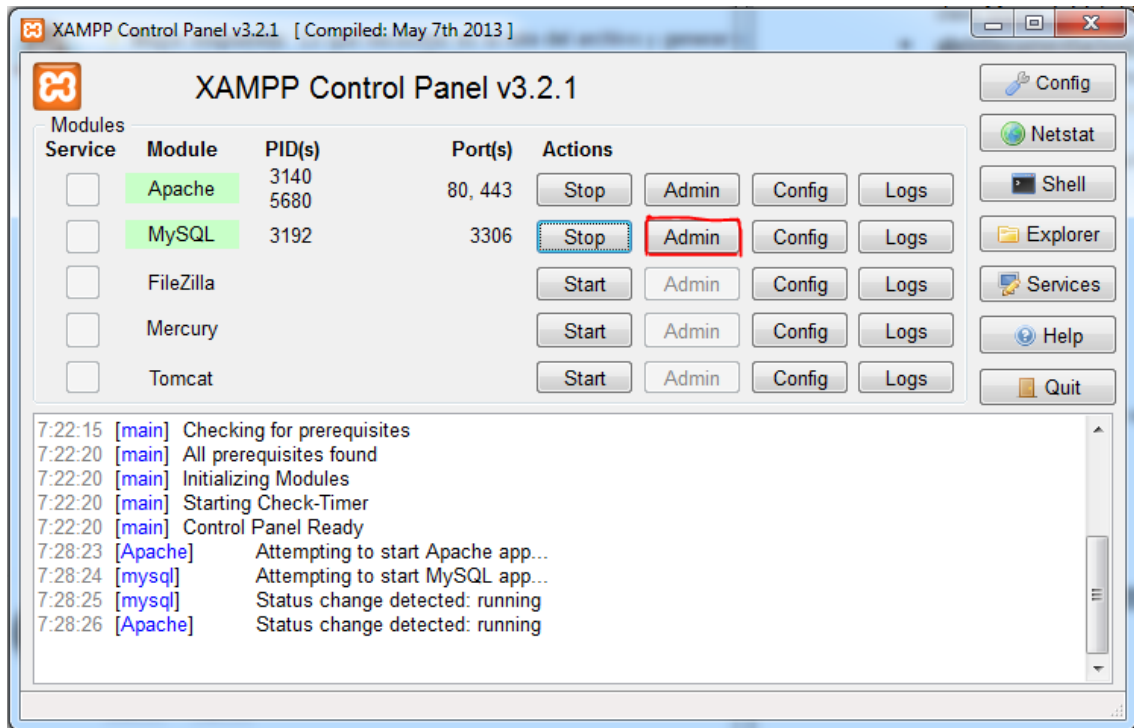
Métodos:

- **jButton1ActionPerformed(java.awt.event.ActionEvent evt).**- Este método es el encargado de controlar lo que sucederá al presionar el botón "Documentación". El mismo que llamará al método `abrirDocumentacion()` para que se pueda mostrar el PDF de la documentación.
- **private void jToggleButton1ActionPerformed(java.awt.event.ActionEvent evt).**- Es el método encargado de controlar el evento presionar el botón "GO", el mismo que pasará a crear las tablas en caso de no existir y de instanciar a la clase `Menu_Inicial` `obj=new Menu_Inicial()` y de derrar dicho `JPane`.
- **abrirDocumentacion().**- Es el método encargado de abrir el PDF de la documentación. Dicho PDF se encontrará en la dirección:

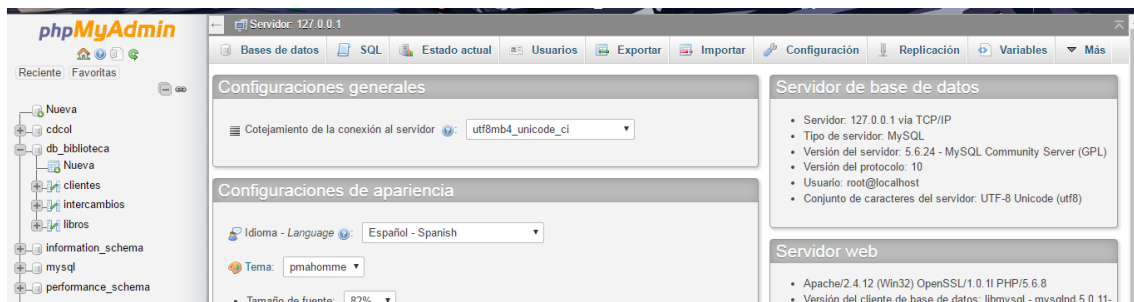
```
"C:\\Users\\User\\Documents\\NetBeansProjects\\Proyecto001\\src\\SQL-Conector\\documentacion.pdf"
```

Generación del Código para crear las tablas:

Procesemos ir a nuestro servidor:



Después procederemos a ir a Administrador (phpmyadmin):



Crearemos nuestra base de datos:

Bases de datos

Crear base de datos

db_biblioteca Cotejamiento Crear

Nota: Activar aquí las estadísticas de la base de datos podría causar tráfico pesado entre el servidor web y el servidor MySQL.

Desplegaremos la base de datos y damos clic a nueva tabla.

phpMyAdmin

Reciente Favoritas

- Nueva
- cdcol
- db_biblioteca
 - Nueva
 - clientes
 - intercambios
 - libros
- information_schema
- mysql
- performance_schema
- phishing
- phpmyadmin
- test
- webauth

Servidor: 127.0.0.1 Base de datos: db_biblioteca

Estructura SQL Buscar Generar una consulta Exportar Importar Operaciones Privilegios Rutinas Más

Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño	Residuo a depurar
clientes	Examinar Estructura Buscar Insertar Vaciar Eliminar	1	MyISAM	latin1_swedish_ci	2 KB	-
intercambios	Examinar Estructura Buscar Insertar Vaciar Eliminar	3	MyISAM	latin1_swedish_ci	3.1 KB	-
libros	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	MyISAM	latin1_swedish_ci	2.1 KB	528
3 tablas	Número de filas	4	InnoDB	latin1_swedish_ci	7.2 KB	528

Marcar todos / Marcar las tablas con residuo a depurar Para los elementos que están marcados:

Vista de impresión Diccionario de datos

Crear tabla

Nombre: Número de columnas: 4

Continuar

Llenamos los campos según lo que necesitamos:

phpMyAdmin

Reciente Favoritas

- Nueva
- cdcol
- db_biblioteca
 - Nueva
 - clientes
 - intercambios
 - libros
- information_schema
- mysql
- performance_schema
- phishing
- phpmyadmin
- test
- webauth

Servidor: 127.0.0.1 Base de datos: db_biblioteca

Estructura SQL Buscar Generar una consulta Exportar Importar Operaciones Privilegios Rutinas Eventos Más

Nombre de la tabla: libros Agregar 1 columna(s) Continuar

Nombre	Tipo	Longitud/Valores	Predeterminado	Cotejamiento	Atributos	Nulo	Índice	A_I	Comentarios
lib_codigo	INT		Ninguno						
lib_titulo	VARCHAR		Ninguno						
lib_autor	VARCHAR		Personalizado: N/A						
lib_año	INT		Ninguno						
lib_existencia	INT		Ninguno						
lib_avaluo	FLOAT		Ninguno						

Comentarios de la tabla: Motor de almacenamiento: InnoDB Cotejamiento:

definición de la PARTICIÓN:

Y damos clic en pre visualizar SQL:

lib_existencia INT Ninguno

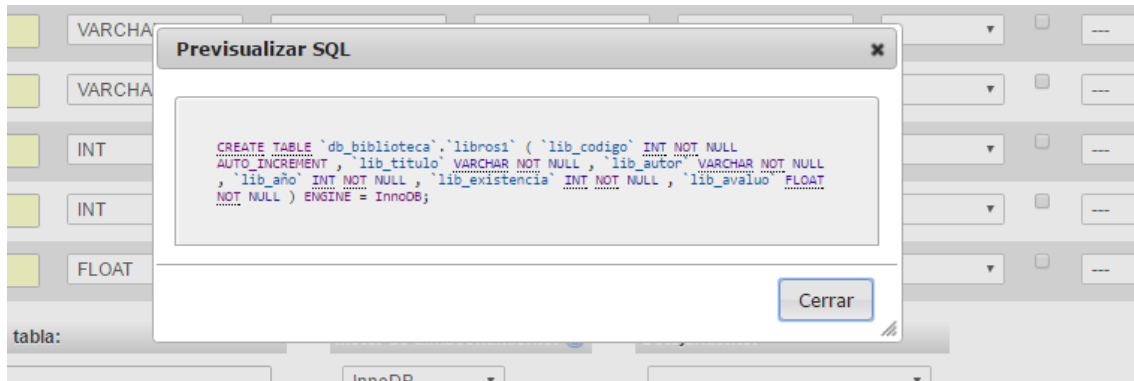
lib_avaluo FLOAT Ninguno

Comentarios de la tabla: Motor de almacenamiento: InnoDB Cotejamiento:

definición de la PARTICIÓN:

Previsualizar SQL Guardar

Y así obtenemos el código SQL para generar la tabla desde NETBEANS:

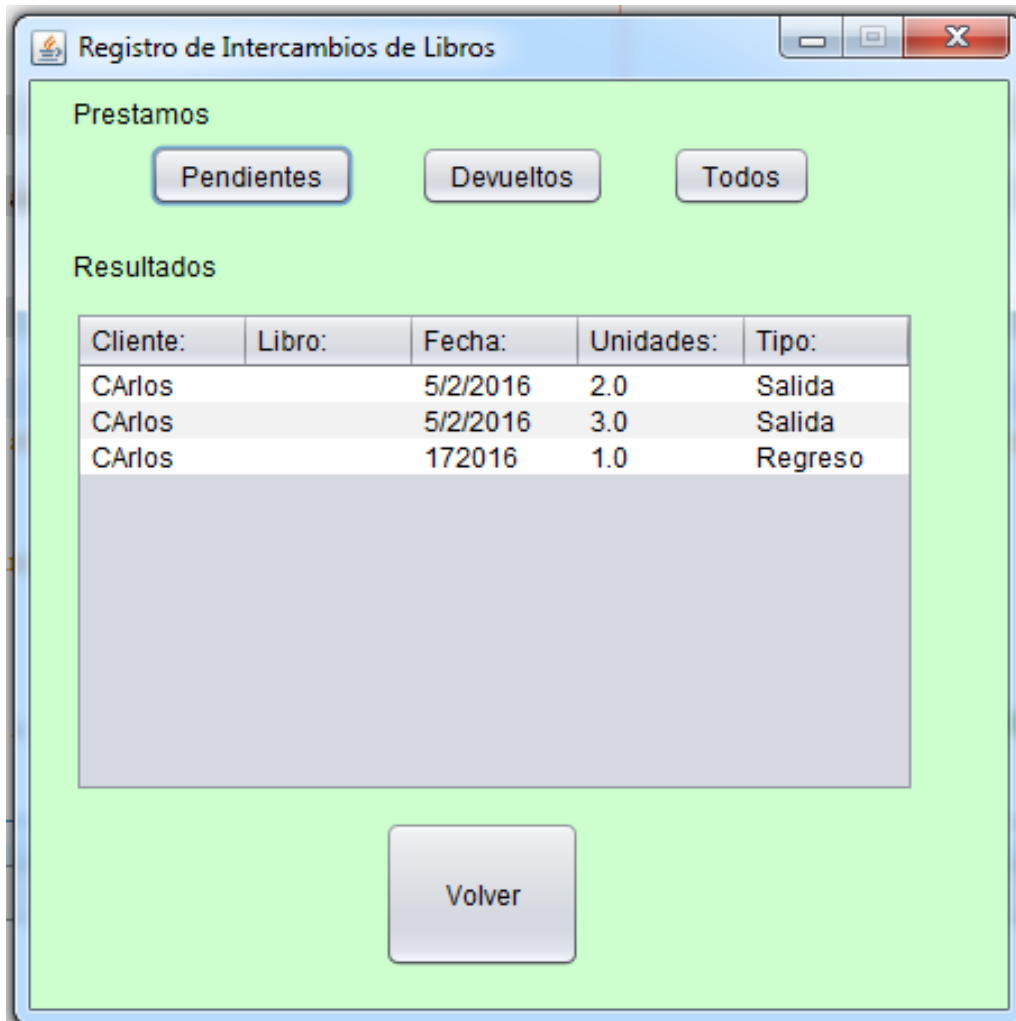


Fuentes para esta clase:

- Yahoo (¿Abrir .txt, .pdf o .ppt con un boton en java (netbeans?))
Fecha de extracción: 2016/08/03
url: <https://es.answers.yahoo.com/question/index?qid=20100508050850AAyvFt6>
- Youtube(Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>
- Youtube(Manejo de Ventanas con Netbeans - Java)
Actualizado el 1 nov. 2010
url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>
- Youtube (Conexion Mysql con JAVA(netbeans))
Publicado el 19 ago. 2013
url: <https://www.youtube.com/watch?v=zJBl4pGylFE>
- Youtube (Manejo de Ventanas con Netbeans - Java)
Actualizado el 1 nov. 2010
url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>

Clase Ingreso intercambio:

Esta clase está encargada de guardar el registro de salida de un libro en la base de datos. Además de poseer 3 botones que permiten filtrar los registros de intercambios.



Cliente:	Libro:	Fecha:	Unidades:	Tipo:
Carlos		5/2/2016	2.0	Salida
Carlos		5/2/2016	3.0	Salida
Carlos		172016	1.0	Regreso

Métodos:

CargarIntercambiosEntrada(). Este método está encargado de cargar la información que recibe desde el servidor(tabla intercambios), haciendo una pequeña validación antes de agregarlo a la tabla.

```
if(res.getInt(5)==2){  
    modelo.addRow(v);  
    coincidencias= coincidencias +1;  
}
```


El cual filtra el tipo de intercambio, en donde se ha definido al tipo 2, como intercambios de retorno de libros a la biblioteca. Y aumentando a la tabla la palabra "Regreso" en vez del número 2;

```
if(res.getInt(5)==1)
    v.add("Salida");
else if(res.getInt(5)==2)
    v.add("Regreso");
```

BuscarchLibro(int indice).- Es el método encargado de obtener el código del libro a partir del código de Registro, este método es llamado desde otras clases y se usa para lo que viene a ser el manejo de información en la base de datos.

BuscarchCliente(int indice).- Al igual que BuscarchLibro este método está encargado de obtener el código del cliente a partir del código de Registro, este método es llamado desde otras clases y se usa para lo que viene a ser el manejo de información en la base de datos.

Estos son necesarios debido a que en otras clases y métodos se tiende a usar tablas con los nombre de los libros y clientes, pero además se necesita de sus códigos, para lo que es actualización de información del servidor, entonces se instancia a esta clases y se usan estos métodos que son del tipo public.

CargarIntercambiosTodos ().- Este método es el encargado de cargar la información que recibe del servidor, a la tabla sin hacer ninguna validación para ello a diferencia de CargarIntercambiosEntrada() y de CargarIntercambiosSalida().

CargarIntercambiosSalida (): Carga los intercambios a la tabla que posean el atributo de salida (1), mediante la validación:

```
if(res.getInt(5)==1) {
    modelo.addRow(v);
    coincidencias= coincidencias +1;
}
```

jButton1ActionPerformed (): Controla la acción respecto al evento presionar el botón Salida, llama al método CargarIntercambiosEntrada();

jButton3ActionPerformed (): Controla la acción del evento presionar el botón Todos, llama al método CargarIntercambiosTodos().

jButton4ActionPerformed (): Controla la acción del evento presionar el botón Volver, instancia a Menu_Prestamo y cierra la ventana actual.

jButton2ActionPerformed (): Controla la acción del evento presionar el botón Devueltos, llama al método CargarIntercambiosSalida();

Clase Conectar:

Esta clase es la encargada de manejar la comunicación con el servidor desde la aplicación. Posee tres partes esenciales el conectar, que es el método encargado de cargar el driver además de dos métodos encargados de hacer consultas al servidor y ejecutar código SQL.

Métodos:

public Connection Conexion().- Este método está encargado de cargar el driver que va a hacer conexión con el servidor. Y nos responderá si la conexión es exitosa o existe algún error en el mismo. Enlazando la aplicación con el servidor.

```
public class Conectar {
    Connection con=null;
    public Connection Conexion(){
        try{
            //cargar driver
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost/db_biblioteca","root","");
            System.out.println("Coneccion establecida");
            //JOptionPane.showMessageDialog(null, "Coneccion establecida");
        } catch (ClassNotFoundException | SQLException e){
            System.out.println("Error de coneccion");
            JOptionPane.showMessageDialog(null, "Error de coneccion"+e);
        }
        return con;
    }
}
```

Consulta(String consulta).- Este método realiza las consultas al servidor. Dichas consultas las recibe en el String consulta. Y recibe una respuesta del servidor.

```
public ResultSet Consulta(String consulta){
    Conexion();
    Statement declara;
    try{
        declara = con.createStatement();
        ResultSet respuesta= declara.executeQuery(consulta);
        return respuesta;
    } catch (SQLException e){
        JOptionPane.showMessageDialog(null, "Error:"+ e.getMessage(), "Error de Coneccion",
            JOptionPane.ERROR_MESSAGE);
    }
    return null;
}
```

EjecutarSQL(String orden).- Este método tiene una función parecida a Consulta, su diferencia radica en que este no espera una respuesta una respuesta del servidor, simplemente ejecuta la orden. Dicha orden la recibe por parte del String orden.



```
public void EjecutarSQL(String orden) {  
    Conexion();  
    Statement declara;  
    try{  
        declara= con.createStatement();  
        declara.execute(orden);  
    }catch (SQLException e) {  
        JOptionPane.showMessageDialog(null, "Error:" + e.getMessage(), "Error de Conexion",  
            JOptionPane.ERROR_MESSAGE);  
    }  
}
```

Fuentes para esta clase:

- Youtube (Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>
- Youtube (Conexion Mysql con JAVA(netbeans))
Publicado el 19 ago. 2013
url: <https://www.youtube.com/watch?v=zJBI4pGylFE>
- Fuente de [Download Connector/J](#)
url: <https://dev.mysql.com/downloads/connector/j/>

MenuInicial:



Este Frame contiene el menú principal del programa, con él podemos ingresar a los menús de clientes, libros e intercambios.

Contiene cinco botones:

-**Conectar**: con este botón se puede comprobar la conexión al servido que contiene la base de datos.

- **Administrar Libros**: nos lleva al Menú de libros en el que se puede ingresar un nuevo libro, modificarlo, buscar un libro, consultar el registro de libros o eliminar un libro.

-**Administrar Clientes**: Este botón nos lleva al Menú de clientes en el que se puede ingresar un nuevo cliente, modificarlo, buscar un cliente, consultar el registro de clientes o eliminarlo.


- **Administrar Intercambios**: permite realizar un préstamo de libro, devolución de libro y Consultar el registro de intercambios.

Fuentes para esta clase:

- Youtube(Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>
- Youtube(Manejo de Ventanas con Netbeans - Java)
Actualizado el 1 nov. 2010
url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>

IngresoClientes

El JFrame IngresoClientes sirve para registrar Nuevos clientes en la base de datos.



Nombre:	Cédula:	Teléfono:	Dirección:
Liseth Serra...	1726153792	2627514	Quito Sur

Contiene 4 campos a llenar.

- Nombres; se guardan los nombres que se ingresen del cliente.
- Cédula; se registra la cédula del cliente.
- Teléfono: se guarda el teléfono del cliente
- Dirección: se registra la dirección del cliente.
- También contiene 4 botones:
- Nuevo: limpia los campos para poder ingresar un nuevo cliente.
- Registrar: Registra el cliente en la base y actualiza la tabla inferior con el nuevo cliente.
- Salir: regresa al usuario al menú clientes,

También aparece una tabla en la parte inferior y esta se va actualizando cada vez que se registra un nuevo cliente en la base de datos.

Fuentes para esta clase:

- Youtube (Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>

- Youtube (Conexion Mysql con JAVA(netbeans))

Publicado el 19 ago. 2013

url: <https://www.youtube.com/watch?v=zJBI4pGylFE>

- Youtube(Manejo de Ventanas con Netbeans - Java)

Actualizado el 1 nov. 2010

url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>

BuscarClientes:

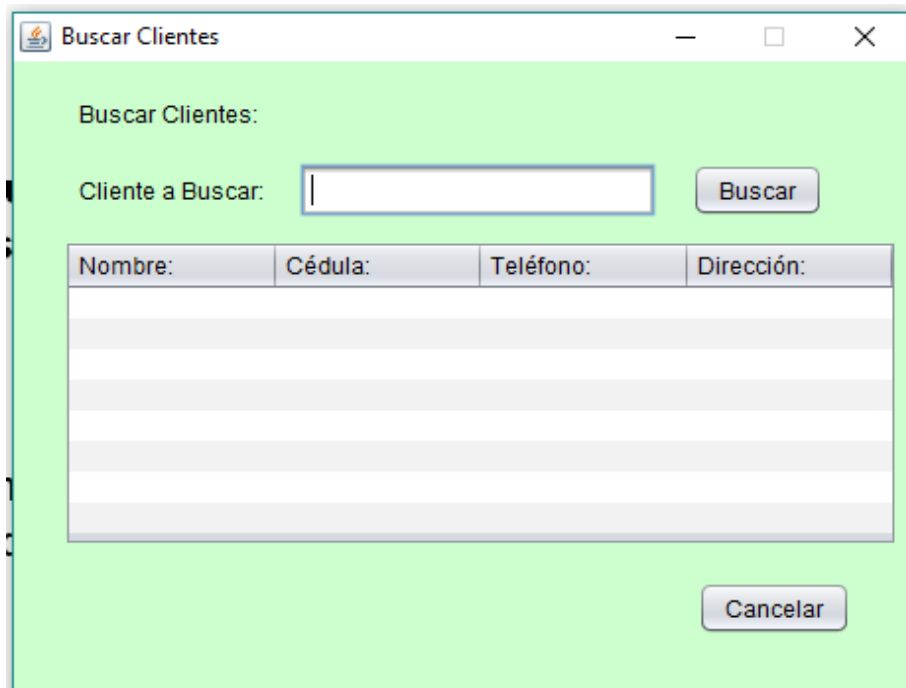
Este Frame sirve para buscar clientes en los registros que se han realizado. Funciona a través de un método que busca coincidencias en la base de datos.

Contiene un campo en el que se ingresa el nombre del usuario que se quiere buscar.

Y dos botones:

- Buscar: busca el cliente en la base de datos.
- Cancelar: regresa al menú clientes.

Y en la parte inferior se despliega una tabla con la información del cliente que se buscó.



Nombre:	Cédula:	Teléfono:	Dirección:

Fuentes para esta clase:

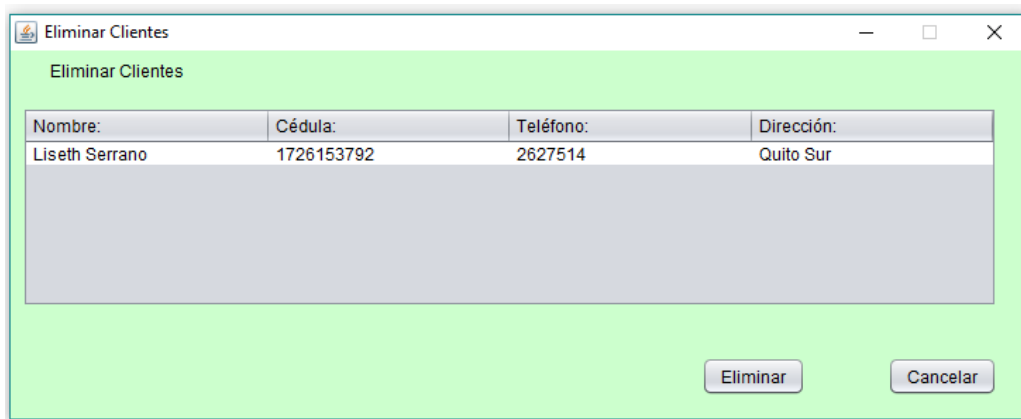
- Youtube (Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>
- Youtube (Conexion Mysql con JAVA(netbeans))
Publicado el 19 ago. 2013
url: <https://www.youtube.com/watch?v=zJBI4pGylFE>
- Youtube(Manejo de Ventanas con Netbeans - Java)
Actualizado el 1 nov. 2010
url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>

EliminarClientes:

Este Frame permite eliminar a un cliente de la base de datos.

Contiene una tabla de Datos en la que se selecciona un cliente y con el botón Eliminar permite borrarlo.

También contiene un botón de Cancelar con el que se puede regresar al menú de clientes.



Nombre:	Cédula:	Teléfono:	Dirección:
Liseth Serrano	1726153792	2627514	Quito Sur

Fuentes para esta clase:

- Youtube (Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>

- Youtube (Conexion Mysql con JAVA(netbeans))

Publicado el 19 ago. 2013

url: <https://www.youtube.com/watch?v=zJBI4pGylFE>

- Youtube(Manejo de Ventanas con Netbeans - Java)

Actualizado el 1 nov. 2010

url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>

- Youtube(Sistema Biblioteca hecho en java Netbeans Parte 1 y Parte 2)

Publicado el 3 may. 2015

Parte 1:

url: <https://www.youtube.com/watch?v=mVOoPlop5ic>

Parte 2:

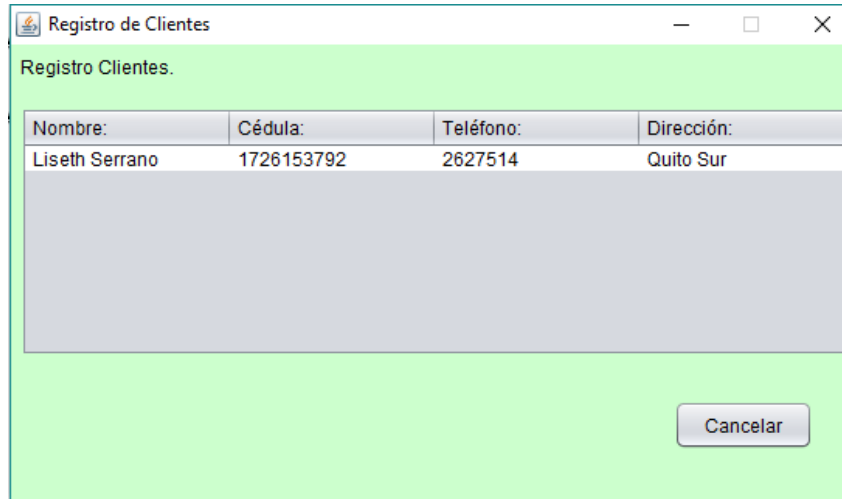
ulr: <https://www.youtube.com/watch?v=RPmvqGtRow4>

RegistroClientes:

Este frame permite consultar el registro de los clientes que se ha realizado, lo hace mediante el método CargarClientes.

Contiene una tabla con todos los clientes y sus datos.

Contiene un botón Cancelar que permite volver al menú de clientes.

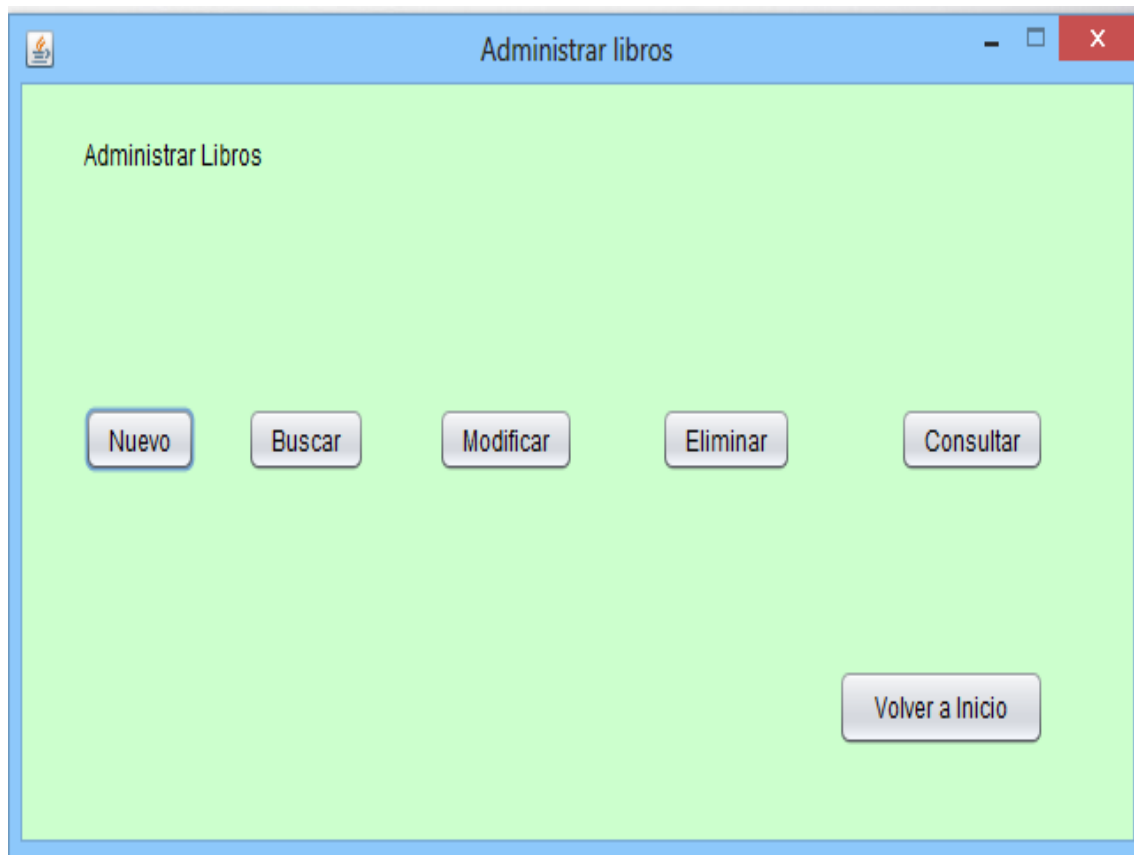


Nombre:	Cédula:	Teléfono:	Dirección:
Liseth Serrano	1726153792	2627514	Quito Sur

Fuentes para esta clase:

- Youtube (Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>
- Youtube (Conexion Mysql con JAVA(netbeans))
Publicado el 19 ago. 2013
url: <https://www.youtube.com/watch?v=zJBI4pGylFE>
- Youtube(Manejo de Ventanas con Netbeans - Java)
Actualizado el 1 nov. 2010
url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>
- Youtube(Sistema Biblioteca hecho en java Netbeans Parte 2)
Publicado el 3 may. 2015
ulr: <https://www.youtube.com/watch?v=RPmqvGtRow4>

Menu _Libros:



La tabla Libros contiene 6 entradas o botones los cuales están diseñados para que trabaje con los siguientes campos.

- **Nuevo**

Permitirá al usuario registrar un nuevo libro que desee ingresar a la biblioteca, una vez ingresado todos los campos se registrara y enseguida se actualizara en la base de datos

Abre una tabla con la información SQL. Libros

- **Buscar**

El usuario podrá buscar en la biblioteca cualquier libro que se encuentre registrado en la base de datos siempre y cuando los campos sean llenados con el nombre del libro o nombre del autor e inmediatamente se actualizara en la base de datos la operación realizada.

Abre una tabla con la información SQL. Libros

- **Modificar**

Modificar libros es una interfaz creada para que el usuario pueda rectificar datos erróneos al ingresar un libro desde esta interfaz se puede modificar todos los campos ingresados en el botón **Nuevo**.

- **Eliminar**

Eliminar permite al usuario desaparecer los datos ingresados.

Se instancia eliminar:

```
EliminarLibros obj=new EliminarLibros();
```

Para desaparecer la ventana anterior entonces utilizo

```
obj.setVisible(true);  
dispose();
```

- **Consultar**

Consultar permite al usuario verificar los datos o libros ingresados en los campos anteriores se puede añadir que solo permite una visualización al usuario no permite modificaciones ni cambios.



Abre una tabla con la información de SQL.libros.

```
private void jButton6ActionPerformed(java.awt.event.ActionEvent evt)  
{  
    RegistroLibros obj=new RegistroLibros();  
    obj.setVisible(true);  
    dispose();  
}
```

- **Volver al inicio**

Botón únicamente creado para volver al Menú Principal

```
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {  
    Menu_Inicial obj=new Menu_Inicial();  
    obj.setVisible(true);  
    dispose();  
}
```

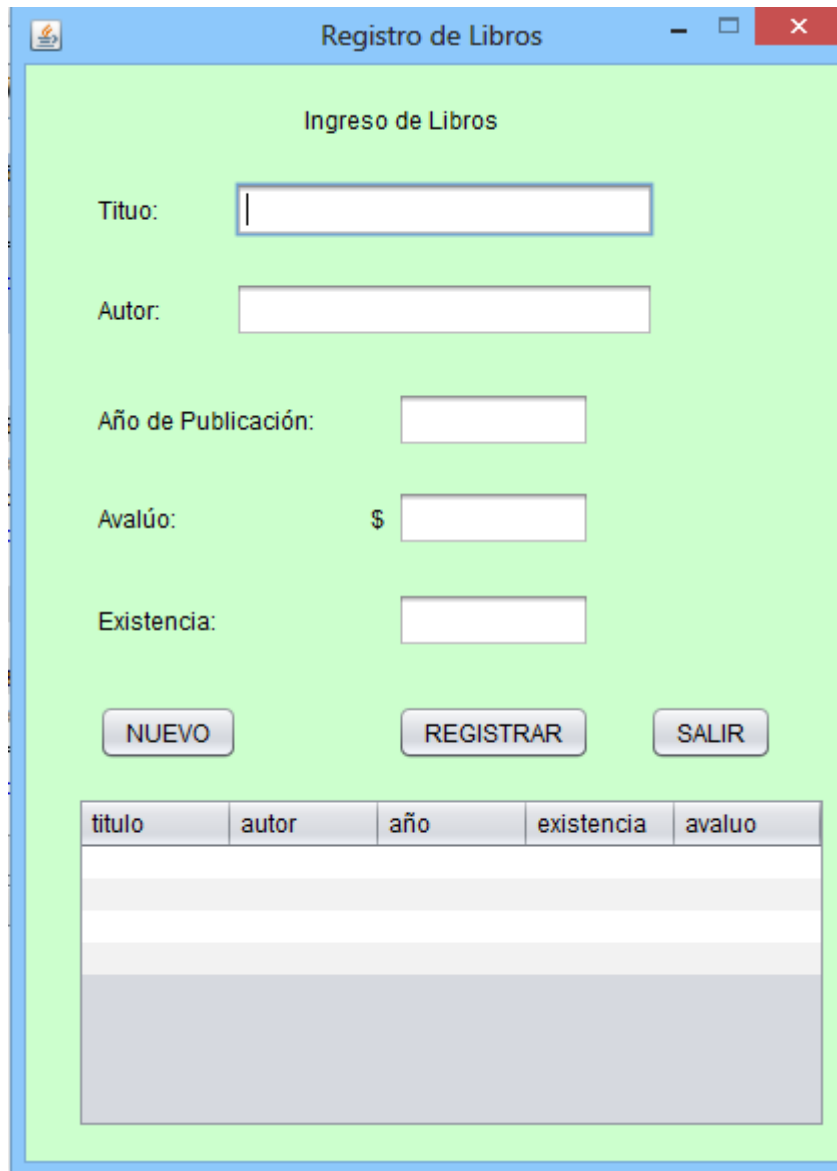
Fuentes para esta clase:

- Youtube (Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>
- Youtube (Conexion Mysql con JAVA(netbeans))

Publicado el 19 ago. 2013

url: <https://www.youtube.com/watch?v=zJBI4pGylFE>
- Youtube(Manejo de Ventanas con Netbeans - Java)
Actualizado el 1 nov. 2010
url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>

IngresoLibros:



titulo	autor	año	existencia	avaluo

Esta interfaz permite al usuario ingresar libros, con título , nombre del autor, Año de publicación Avalúo, Existencia registrándose y enseguida actualizándose en la base de datos.

- **NUEVO**
Borra los campos ingresados para ingresar otro libro.
- **REGISTRAR**
Registra los nombres y enseguida los actualiza en la base de datos creada.
Si no lo encuentra saltara un mensaje de error función try catch.

Registro de Libros

Ingreso de Libros

Título:

Autor:

Año de Publicación:

Avalúo: \$

Existencia:

titulo	autor	año	existencia	avaluo

Registro de Libros

Ingreso de Libros

Título:

Autor:


Año de Publicación:

Avalúo:

Existencia:

titulo	autor	año	existencia	avaluo

Mensaje

 Registrado con éxito

Registro de Libros

Ingreso de Libros

Título:

Autor:

Año de Publicación:

Avalúo: \$

Existencia:

titulo	autor	año	existencia	avaluo
ORION	GALILEO	1870	1	120.0

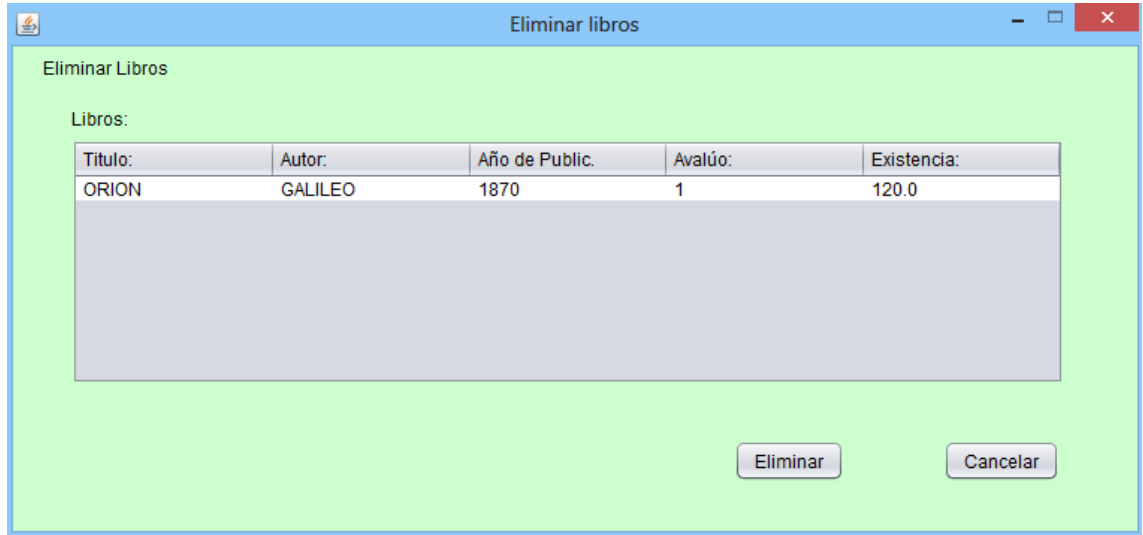
- **SALIR**

Botón que permite al usuario salir a la pantalla de administrar libros no tiene otra función.

Fuentes para esta clase:

- Youtube (Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>
- Youtube (Conexion Mysql con JAVA(netbeans))
Publicado el 19 ago. 2013
url: <https://www.youtube.com/watch?v=zJBI4pGylFE>
- Youtube(Manejo de Ventanas con Netbeans - Java)
Actualizado el 1 nov. 2010
url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>

EliminarLibros:



Eliminar Libros

Libros:

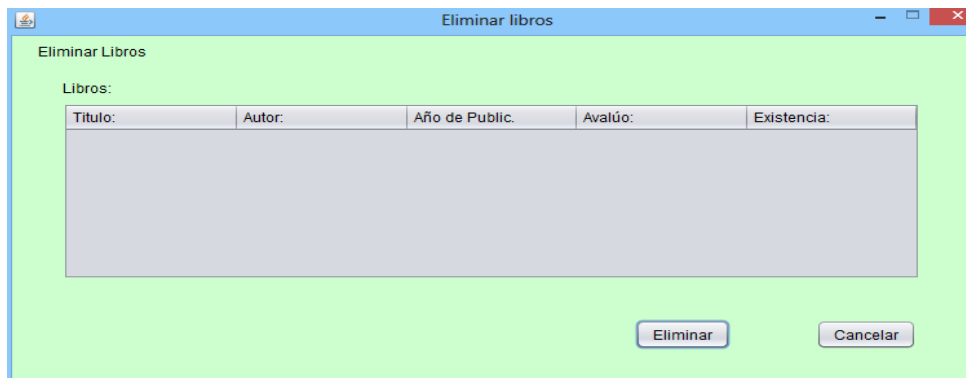
Titulo:	Autor:	Año de Public.	Avalúo:	Existencia:
ORION	GALILEO	1870	1	120.0

Eliminar Cancelar

Esta interfaz con solo 2 opciones Eliminar y Cancelar permite al usuario marcar el elemento y borrarlo de su registro enseguida actualizándolo en la base de datos

- **Eliminar**

Envía la orden al servidor para que elimine y vuelve a cargar el registro de libros.



Eliminar Libros

Libros:

Titulo:	Autor:	Año de Public.	Avalúo:	Existencia:
---------	--------	----------------	---------	-------------

Eliminar Cancelar

- **Cancelar**

Botón esencial que tiene un único trabajo volver a la pantalla Administrar Libros.

Fuentes para esta clase:

- Youtube (Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>
- Youtube (Conexion Mysql con JAVA(netbeans))

Publicado el 19 ago. 2013

url: <https://www.youtube.com/watch?v=zJBI4pGylFE>

- Youtube(Manejo de Ventanas con Netbeans - Java)
Actualizado el 1 nov. 2010
url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>

- Youtube(Sistema Biblioteca hecho en java Netbeans Parte 1, Parte 2)

Publicado el 3 may. 2015

Parte 1:

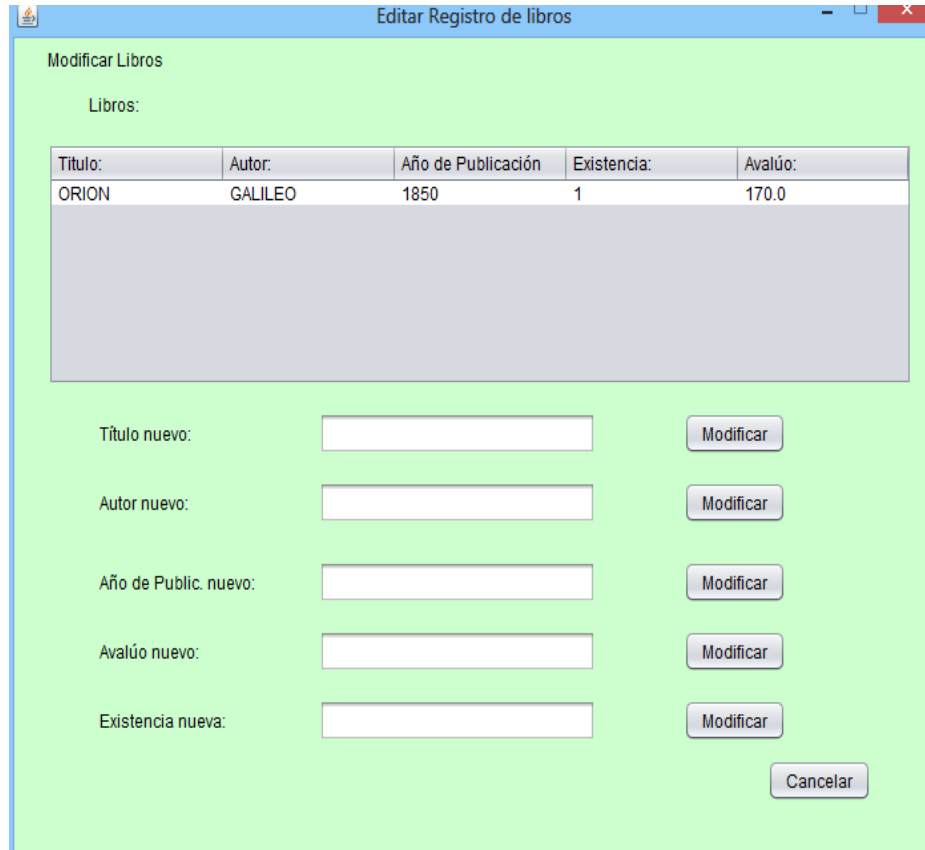
url: <https://www.youtube.com/watch?v=mVOoPlop5ic>

Parte 2:

url: <https://www.youtube.com/watch?v=RPmvqGtRow4>

- Youtube (3 NetBeans MySQL - Cargar tablas en JTable)
Publicado el 10 ene. 2014
url: <https://www.youtube.com/watch?v=YhsJx74T5-0>

ModificarLibros:



Titulo:	Autor:	Año de Publicación	Existencia:	Avalúo:
ORION	GALILEO	1850	1	170.0

Título nuevo:

Autor nuevo:

Año de Public. nuevo:

Avalúo nuevo:

Existencia nueva:

Modificar libros es una interfaz creada para que el usuario pueda rectificar datos erróneos al ingresar un libro desde esta interfaz se puede modificar todos los campos ingresados en el botón **Nuevo**.

- **Modificar Título**

Al seleccionar un libro el usuario escoge la opción modificar título entonces se modificara y se registrara el siguiente método representa la operación.

- **Modificar Autor**

Al seleccionar un libro el usuario escoge la opción modificar autor entonces se modificara y se registrara, el siguiente método representa la operación.

- **Modificar el Año de publicación**

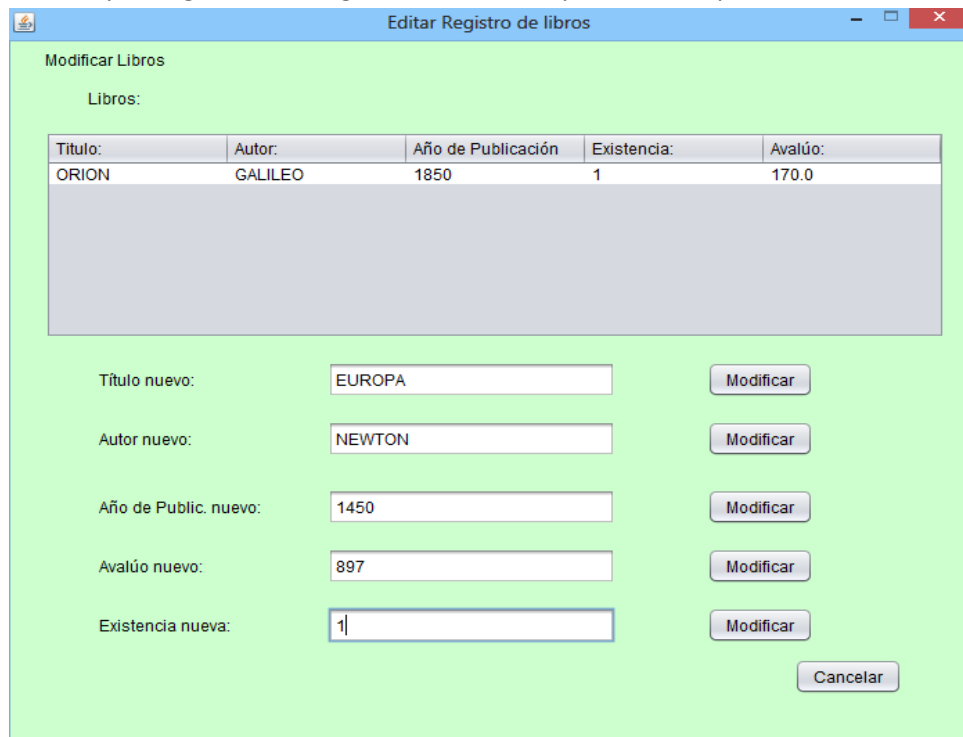
Al seleccionar un libro el usuario escoge la opción modificar **el Año de publicación** entonces se modificara y se registrara, el siguiente método representa la operación.

- **Modificar Avalúo**

Al seleccionar un libro el usuario escoge la opción modificar **Avalúo** entonces se modificara y se registrara , el siguiente método representa la operación.

- **Modificar Existencia**

Al seleccionar un libro el usuario escoge la opción modificar **Existencia** entonces se modificara y se registrará, el siguiente método representa la operación.



Editar Registro de libros

Modificar Libros

Libros:

Título:	Autor:	Año de Publicación	Existencia:	Avalúo:
ORION	GALILEO	1850	1	170.0

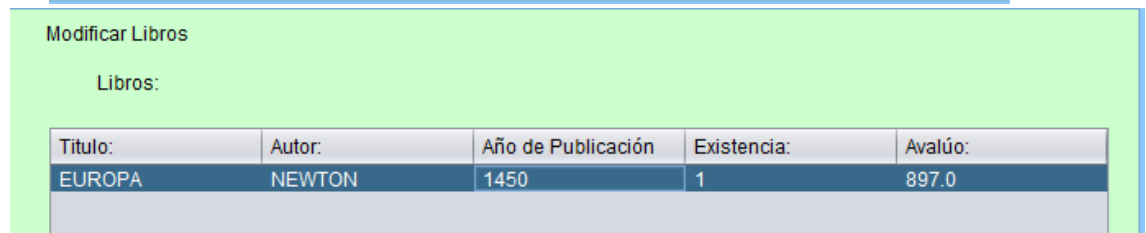
Título nuevo:

Autor nuevo:

Año de Public. nuevo:

Avalúo nuevo:

Existencia nueva:



Modificar Libros

Libros:

Título:	Autor:	Año de Publicación	Existencia:	Avalúo:
EUROPA	NEWTON	1450	1	897.0

- **Cancelar**

Botón diseñado únicamente para regresar a la pantalla de. Administrar Libros .
Desapareciendo la pantalla modificar libros.



Fuentes para esta clase:

- Youtube (Conexion Con Java y SQL server, Cargar datos en tablas)
Publicado el 24 ene. 2014.
url: <https://www.youtube.com/watch?v=9r00K04UUWg>
- Youtube (Conexion Mysql con JAVA(netbeans))
Publicado el 19 ago. 2013
url: <https://www.youtube.com/watch?v=zJBI4pGylFE>
- Youtube(Manejo de Ventanas con Netbeans - Java)
Actualizado el 1 nov. 2010
url: <https://www.youtube.com/watch?v=ZsVys0AG66Q>
- Youtube (3 NetBeans MySQL - Cargar tablas en JTable)
Publicado el 10 ene. 2014
url: <https://www.youtube.com/watch?v=YhsJx74T5-0>

VENTANA MENÚ CLIENTES:



En esta clase tenemos 6 botones que permiten elegir varias opciones del menú cliente. Entre ellas tenemos:

- Botón Nuevo.- Abre el Ingreso a Cliente.

Primero instanciamos un objeto el cual abrirá a la clase Ingreso clientes que permitirá su ingreso, este método `obj.setVisible(true)` nos ayuda a que la ventana anterior que está abierta se cierre en cuando vaya a la siguiente ventana, y la última línea de código ayuda a habilitar la liberación de recursos no administrados de forma determinista Botón Buscar: “Abre a la ventana para buscar a cualquier cliente”. (Microsoft , 2016)

`BuscarClientes obj=new BuscarClientes ()` .Instanciando un objeto que abrirá a buscar clientes.

- Botón Modificar: Abre la ventana para poder modificar a los datos de un cliente

`ModificarClientes obj= new ModificarClientes ()` . En la primera línea Instanciamos un objeto para ir a Modificar Clientes

- Botón Eliminar: Abre a la ventana eliminar para borrar a clientes.

`EliminarClientes obj = new EliminarClientes ()` .Instanciamos a EliminarClientes.

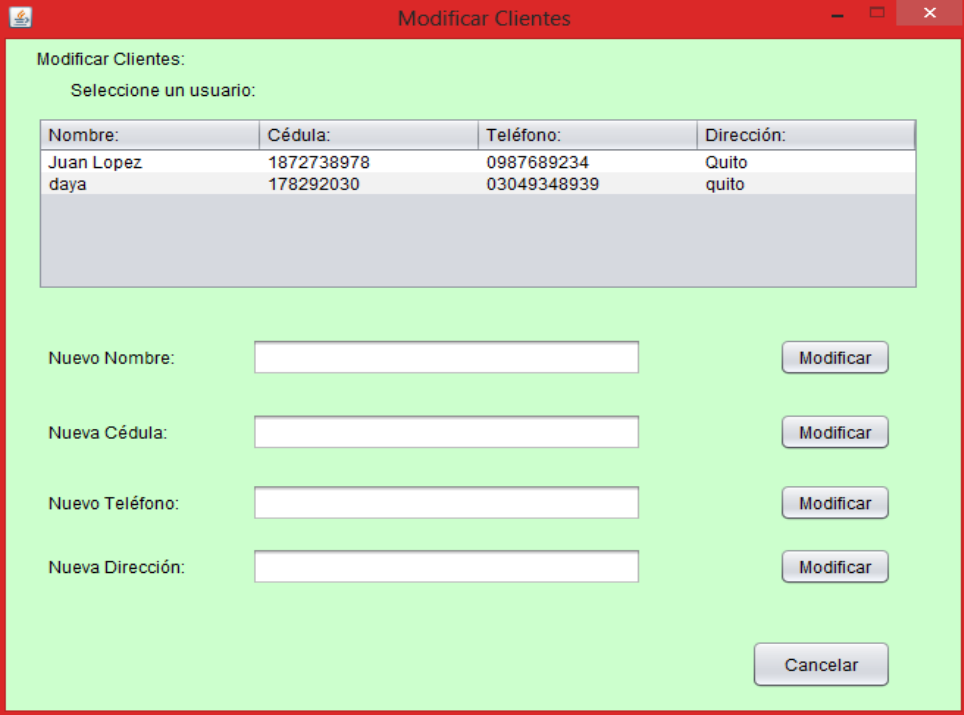
- Botón Consultar: Abre a la ventana donde se encuentra el registro de clientes.

`RegistroClientes obj = new RegistroClientes ()` . Instanciamos a Registro Clientes.

- Botón Volver: Permite volver al Menú Principal.

`Menu_Inicial obj= new Menu_Inicial ()` .Instanciamos un objeto que nos permite regresar a Menu_Principal.

VENTANA MODIFICAR CLIENTES:



Modificar Clientes:

Seleccione un usuario:

Nombre:	Cédula:	Teléfono:	Dirección:
Juan Lopez	1872738978	0987689234	Quito
daya	178292030	03049348939	quito

Nuevo Nombre:

Nueva Cédula:

Nuevo Teléfono:

Nueva Dirección:

En la tabla modificar clientes tenemos una tabla que permite mostrar al usuario que desea modificar y abajo están los campos que desea modificar como es el nombre, la cédula, el teléfono la dirección y junto a estos campos están los botones modificar que ayuda a cambiar los datos.

- Botón modificar nombre del cliente: Modifica el nombre en el servidor

En la primera línea de código `int s = TablaClientes.getSelectedRow ();` nos dice que nos devuelve un número de la fila seleccionada de la Tabla clientes por lo que declaramos una variable `d` tipo `int` utilizando el método `getSelectRow()`.luego creamos una variable `String` código para que me pase esos valores de la tabla libros evaluados en `s` y `0` .además declaramos un `String` en donde con un `getText` método devuelve el texto de su barra de título .Luego damos la orden a la base de datos , conectamos con la base de datos y mandamos la orden, por ultimo cargamos el método `CagarLibros` para que te aparezca la tabla y usamos un método `getText()` que sirve para mostrar cualquier argumento en el componente texto (Lescano, 2009).

- Botón modificar cedula del cliente: Modifica el número de cedula en el servidor.

La línea de código es igual que la anterior solo que aquí la variable es: `String newCedula= T_Cedula.getText ();`

- Botón modificar teléfono del cliente: Modifica el número de teléfono.

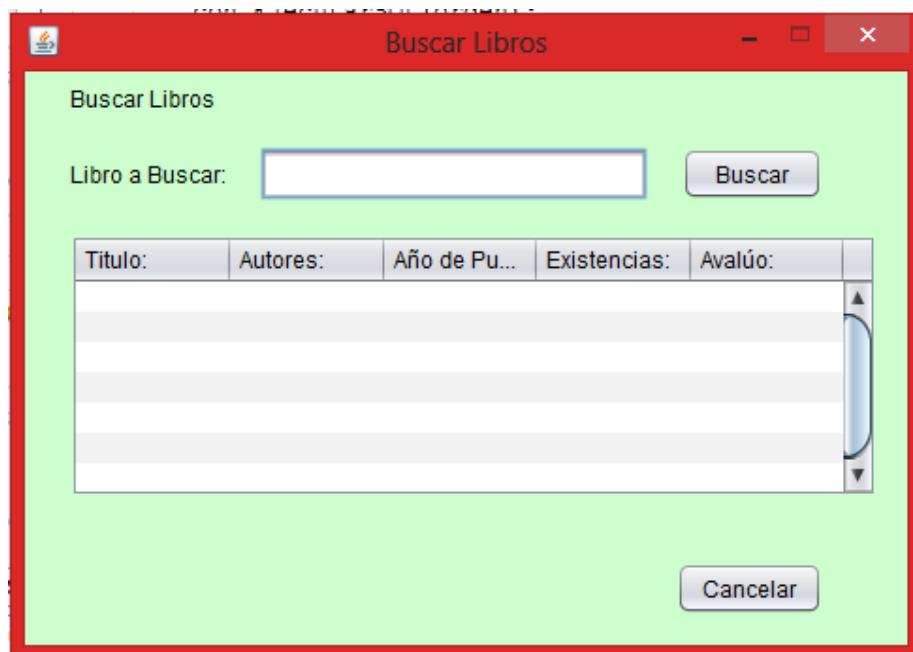
Tiene la misma línea de código pero con la variable: `String newTelefono=T_Telefono.getText();`

- Botón modificar dirección del cliente: Modifica la dirección en la base de datos.

Aquí también son las mismas líneas de código con la variable: `String newDireccion=T_Direccion.getText();`

- Botón cancelar: Regresa al menú clientes.

VENTANA BUSCAR LIBROS



Título:	Autores:	Año de Pu...	Existencias:	Avalúo:

En esta clase de JFrame aparece un JTextField que sirve para escribir un dato a buscar en este caso es un libro que lo buscaremos por su Título y a lado está el botón buscar que nos permitirá encontrar el libro y nos mostrara en la tabla que aparece de bajo de este (Oracle, 1996-2016).

- Botón Buscar libros: Busca en la base datos por el título a un libro.

Al principio de clase declaramos un atributo de tipo String `clavetitulo` que tendrá un parámetro `T_clave` que es el nombre de nuestro JTextField y con el método `getText()` y por ultimo llamamos a nuestro método `BuscarCoincidencias`.

- Botón Cancelar: Permite regresar al menú Libros.

VENTANA REGISTRO LIBROS:



En este JFrame podemos observar la Tabla de libros que nos muestra de la base de datos, para ver cuántos libros hay disponibles y después tenemos un botón volver.

```
public RegistroLibros()
```

El primer public RegistroLibros es el constructor que viene por defecto, en el cual aumentamos.

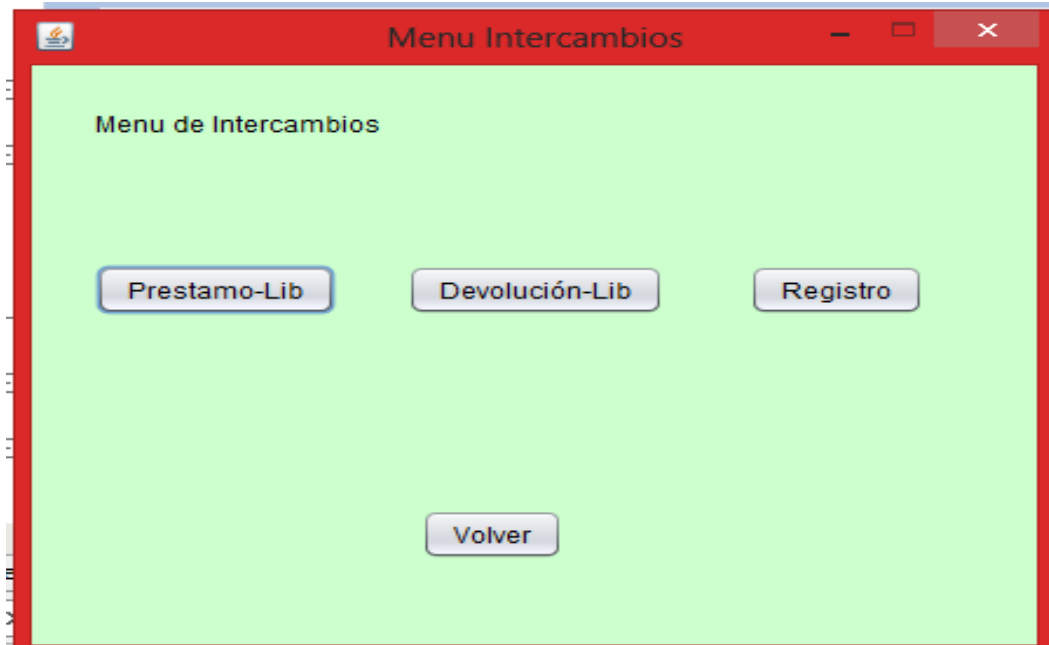
- el método `getContentPane().setBackground` que nos permite cambiar de color a la ventana que se muestra en pantalla al usuario.
- El método `setLocationRelativeTo()`: Establece la posición de la ventana relativa a un componente pasado como parámetro. Si se le pasa null como parámetro se posiciona en el centro de la pantalla.
- el Método `setResizable(false)`: Permite que no se pueda modificar el tamaño de la ventana del JFrame en tiempo de ejecución.
- El método `setDefaultCloseOperation(0)`: Retorna la operación que ocurre cuando el usuario cierra la ventana del frame.

Por ultimo llámanos al método CargarLibros.

- El método CargarLibros: primero instanciamos un modelo de datos : `DefaultTableModel modelo=(DefaultTableModel) jTable1.getModel()` y luego un **JTable** es un componente visual de java que nos permite dibujar una tabla, de forma que en cada fila o columna de la tabla podamos poner cualquier dato de la tabla; luego en el parámetro dado como modelo llámanos al método `setSelectRow()`

Luego nos conectamos con la base de datos y con la variable res que va dentro de un while () para que evalúe mientras se den las condiciones y vaya añadiendo en un vector que instanciamos nuestros datos. Y por último llámanos al jTable.

VENTANA MENU PRÉSTAMOS



Esta clase de JFrame tiene cuatro botones que son:

- Botón Prestamos-Libros: Instancia a Nuevo_Intercambio

IngresoIntercambio obj=new IngresoIntercambio ()

- Botón Devolución –Libros: Instancia a IngresoRetorno.

IngresoRetorno obj= new IngresoRetorno ()

- Botón Registro: Instancia a RegistroIntercambio.

RegistroIntercambios obj=new RegistroIntercambios ();

- Botón Volver: Nos permite volver al Menu_Inicial

Menu_Inicial obj=new Menu_Inicial()

Referencias

- Lescano, W. S. (2009). Ingresar datos con un control visual Java. JTextField y getText, clase y método. *Aprende a programar*, 6. Recuperado de :
[http://www.aprenderaprogramar.com/index.php?option=com_attachments&task=download&id=189\[\]](http://www.aprenderaprogramar.com/index.php?option=com_attachments&task=download&id=189[])
- Microsoft . (2016). IDisposable (Interfaz). *NET Framework (current version)*, 1. Recuperado de:
Microsoft: System.IDisposable
- Oracle. (1996-2016). Class JTextField. *Java™ Platform, Standard Edition 7*, 1. Recuperado de:
<https://docs.oracle.com/javase/7/docs/api/javax/swing/text/JTextComponent.html>
- Jiménez, J. (09 de Enero de 2014). Tecnología. Recuperado de Tecnología:
<http://www.genbetadev.com/herramientas/netbeans-1>.
- Salazar, J.M.(2009).Java desde cero y algo mas.BlogSpot.Recuperado: <http://javax-peru.blogspot.com/>

Conclusiones:

- Podemos concluir que una de las cosas más fundamentales al momento de realizar un trabajo es en primero lugar la investigación .Para esto debimos investigar muchas de las cosas que nos fueron útiles al momento de programar y otras que ya teníamos conocimiento; por ejemplo aprendimos que NetBeans es uno de los entornos profesionales y que todas sus funciones avanzadas estas en este IDE potente, con ventaja adicional que muestra una sencilla instalación y con este su kit de desarrollo.
- A pesar de que existen otros lenguajes más populares y convenientes en la actualidad, java sigue siendo un lenguaje muy sencillo en la actualidad y a la vez muy necesario porque dalas pautas o los cimientos de conocimientos para crear cosas aún más grandes.

Recomendaciones:

- Uso de Windows 7.
- Uso de XAMPP (como servidor).
- La creación de un botón en el menú inicial para probar la conexión con el servidor.
- El uso de Gestores de proyectos para poder encaminar el proyecto según se va avanzando (cono Github).
- Comprobar la disponibilidad de los puertos para el servidor.
- Dejar la contraseña y usuario del servidor en los por defecto (null/root).
- Todo lo que sea con el servidor, se realice mediante código.
- Usar nombres completos para las variables a usar.
- Usar diagramas para ver la navegación entre los JFrame.
- Procurar siempre tener un respaldo de todo lo que se realice.

Problemas surgidos en el desarrollo del proyecto:

- Al inicio del desarrollo del proyecto, usábamos el servidor WAMPSEVER, el cual hasta ese momento parecía bastante conveniente, hasta que llegamos al punto de crear los enlaces entre las tablas, lo cual no nos permitía.
- Cuando se Generó el UML a partir del código, se perdió el accedo al diseñador de JFrame, lo cual causo que se tuviera que reinstalar el IDE.
- Al inicio pensábamos que mediante la función consulta de Conectar podíamos realizar todo en el servidor, pero no nos funcionaban aquellos códigos que no devuelven respuestas, no supimos hasta revisar la documentación de Coneccion, de en que estábamos fallando.