

Autor: Laura Muñoz Palomo

Manual Técnico

Programación Práctica Gestión 3º Trimestre

10-06-2019

Manual Técnico

ÍNDICE

Introducción	3
Base de Datos	4
Diagrama Entidad-Relación (ERD)	4
Esquema Relacional (ER)	5
Diagrama de Workbench	5
Creación de Base de Datos	6
Creación de Tablas	6
Librerías	8
Instalación	8
Código Java	11
Modelo.java	11
ImprimirPDF.java	16
Login.java	19
MenuPrincipal.java	25
Socio.java	30
Profesor.java	41
Clase.java	52

Introducción

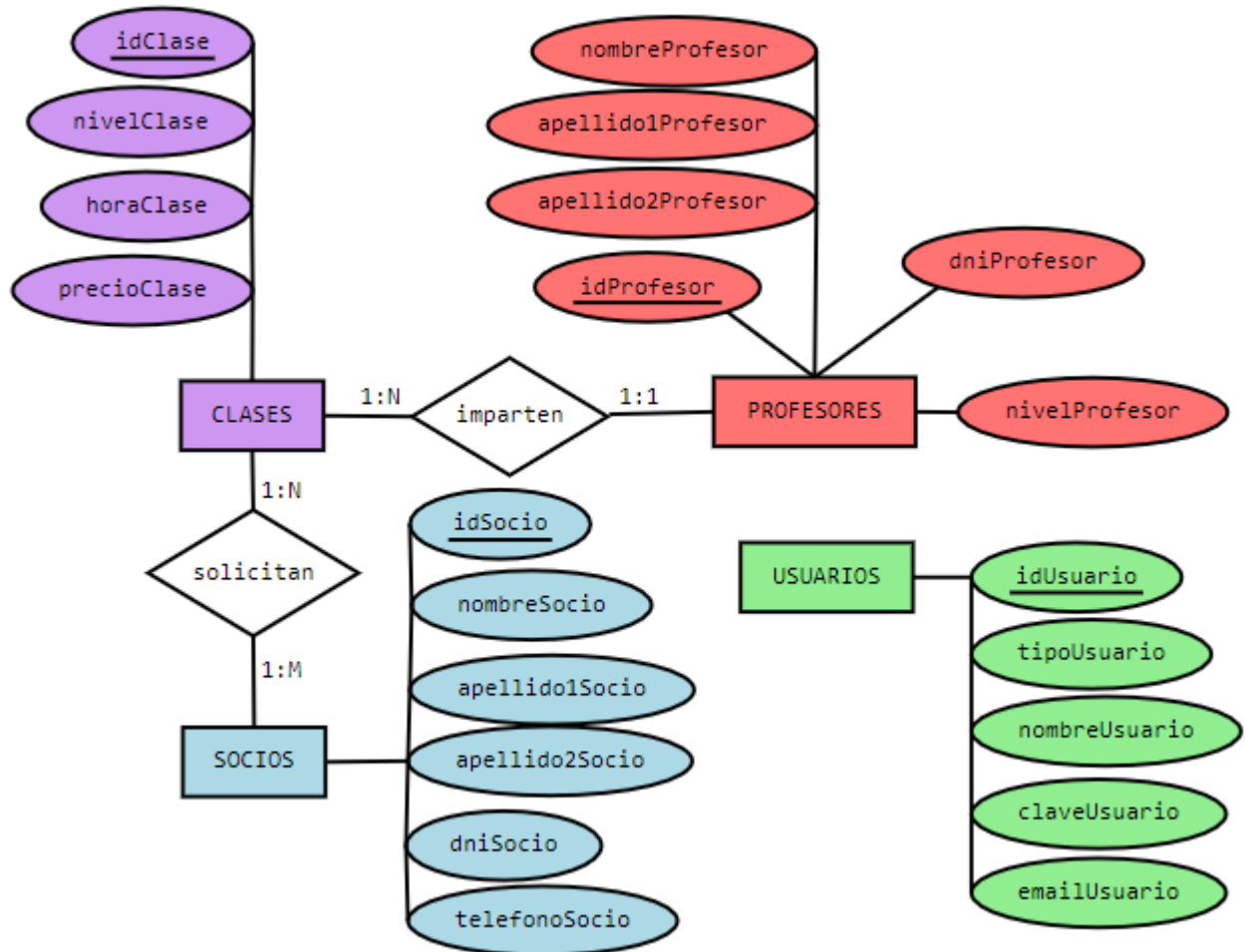
Este es el manual técnico del programa de gestión de una empresa, específicamente de un club de tenis. En este manual se describen los aspectos para la gestión de la base de datos y el código utilizado:

Más específicamente las siguientes características:

- ✓ Modelo de datos (diagrama E-R, esquema relacional y diagrama de Workbench).
- ✓ Sentencias SQL de creación de la base de datos y las tablas.
- ✓ Librerías externas utilizadas para el desarrollo de la aplicación.
- ✓ Las clases JAVA desarrolladas.

Base de Datos

Diagrama Entidad-Relación (ERD)



Esquema Relacional (ER)

PROFESORES (**idProfesor**, nombreProfesor, apellido1Profesor, apellido2Profesor, dniProfesor, nivelProfesor)

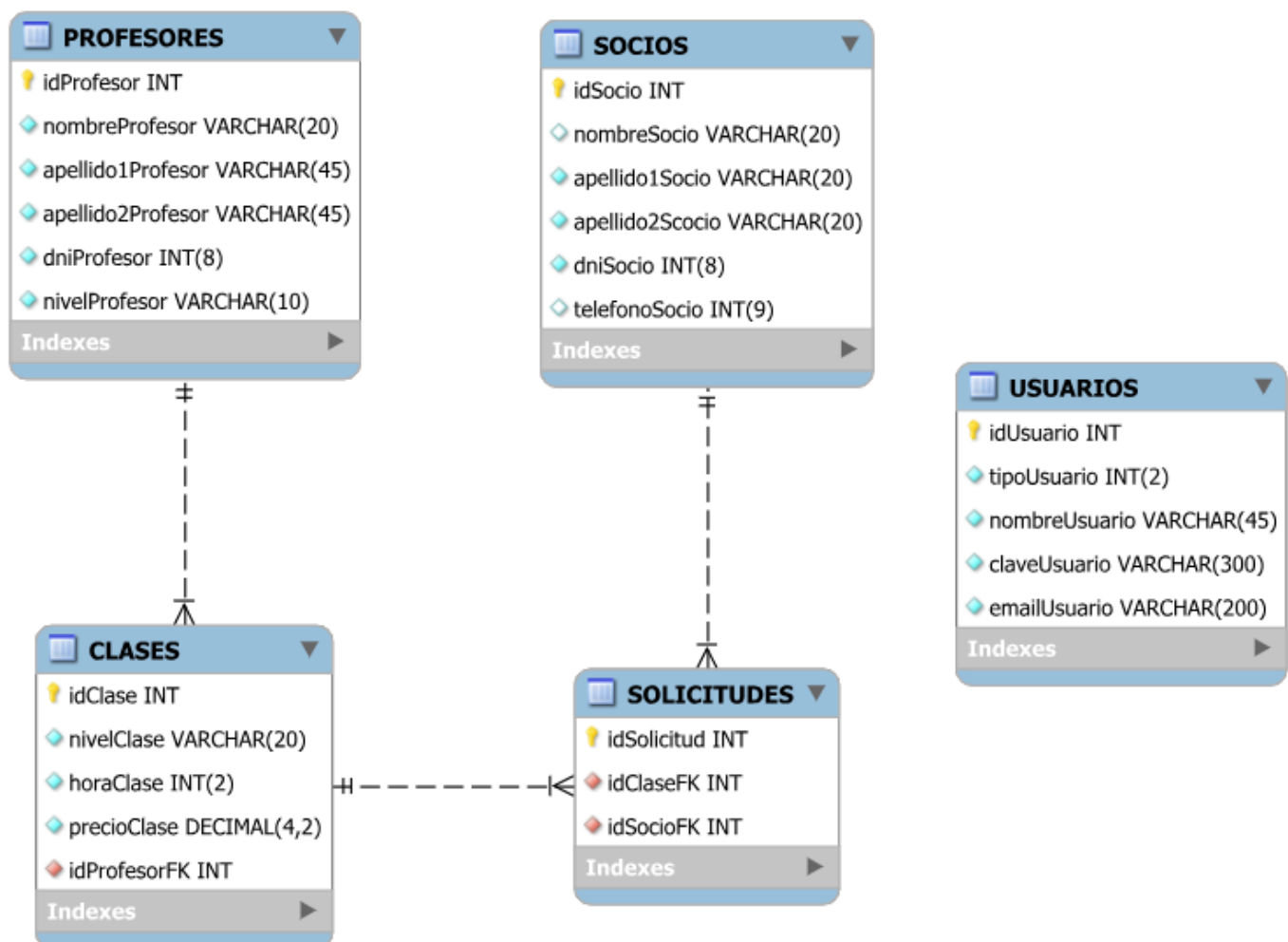
SOCIOS (**idSocio**, nombreSocio, apellido1Socio, apellido2Socio, dniSocio, telefonoSocio)

CLASES (**idClase**, nivelClase, horaClase, precioClase, **idProfesorFK**)

SOLICITUDES (**idSolicitud**, **idClasesFK**, **idSocioFK**)

USUARIOS (**idUsuario**, tipoUsuario, nombreUsuario, claveUsuario, emailUsuario)

Diagrama de Workbench



Creación de Base de Datos

```
MySQL 8.0 Command Line Client - Unicode
mysql> create database gestion charset utf8 collate utf8_spanish2_ci;
Query OK, 1 row affected, 2 warnings (0.24 sec)
mysql>
```

```
MySQL 8.0 Command Line Client - Unicode
mysql> show databases;
+-----+
| Database |
+-----+
| biblioteca |
| clubdetenis |
| ejercicio1 |
| ejercicio10 |
| ejercicio4 |
| ejercicio5 |
| ejercicio7 |
| ejercicio8 |
| empresa |
| gestion |
| information_schema |
| mysql |
| performance_schema |
| sys |
| tallerimportexport |
| videoclub |
+-----+
```

Creación de Tablas

PROFESORES

```
MySQL 8.0 Command Line Client - Unicode
mysql> create table profesores (idProfesor int auto_increment, nombreProfesor varchar(20), apellido1Profesor varchar(45),
apellido2Profesor varchar(45), dniProfesor int(8), nivelProfesor varchar(10), primary key(idProfesor));
Query OK, 0 rows affected (1.37 sec)
mysql> describe profesores;
+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+
| idProfesor | int(11) | NO | PRI | NULL | auto_increment |
| nombreProfesor | varchar(20) | YES | | NULL | |
| apellido1Profesor | varchar(45) | YES | | NULL | |
| apellido2Profesor | varchar(45) | YES | | NULL | |
| dniProfesor | int(8) | YES | | NULL | |
| nivelProfesor | varchar(10) | YES | | NULL | |
+-----+
6 rows in set (0.00 sec)
mysql>
```

SOCIOS

```
MySQL 8.0 Command Line Client - Unicode
mysql> create table socios(idSocio int auto_increment, nombreSocio varchar(20), apellido1Socio varchar(20), apellido2Socio varchar(20), dniSocio int(8), telefonoSocio int(9), primary key(idSocio));
Query OK, 0 rows affected (0.48 sec)

mysql> describe socios;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idSocio | int(11) | NO | PRI | NULL | auto_increment |
| nombreSocio | varchar(20) | YES | | NULL | |
| apellido1Socio | varchar(20) | YES | | NULL | |
| apellido2Socio | varchar(20) | YES | | NULL | |
| dniSocio | int(8) | YES | | NULL | |
| telefonoSocio | int(9) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)

mysql>
```

CLASES

```
MySQL 8.0 Command Line Client - Unicode
mysql> create table clases(idClase int auto_increment, nivelClase varchar(20), horaClase int(2), precioClase decimal(4,2), idProfesorFK int, primary key(idClase), foreign key(idProfesorFK) references profesores(idProfesor));
Query OK, 0 rows affected (0.87 sec)

mysql> describe clases;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idClase | int(11) | NO | PRI | NULL | auto_increment |
| nivelClase | varchar(20) | YES | | NULL | |
| horaClase | int(2) | YES | | NULL | |
| precioClase | decimal(4,2) | YES | | NULL | |
| idProfesorFK | int(11) | YES | MUL | NULL | |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.02 sec)

mysql>
```

SOLICITUDES

```
MySQL 8.0 Command Line Client - Unicode
mysql> create table solicitudes(idSolicitud int auto_increment, idClaseFK int, idSocioFK int, primary key(idSolicitud), foreign key(idClaseFK) references clases(idClase), foreign key(idSocioFK) references socios(idSocio));
Query OK, 0 rows affected (1.12 sec)

mysql> describe solicitudes;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idSolicitud | int(11) | NO | PRI | NULL | auto_increment |
| idClaseFK | int(11) | YES | MUL | NULL | |
| idSocioFK | int(11) | YES | MUL | NULL | |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql>
```

USUARIOS

```
MySQL 8.0 Command Line Client - Unicode
mysql> create table usuarios(idUsuario int auto_increment, tipoUsuario int(2), nombreUsuario varchar(45), claveUsuario varchar(300), emailUsuario varchar(200), primary key(idUsuario));
Query OK, 0 rows affected (0.54 sec)

mysql> describe usuarios;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idUsuario | int(11) | NO | PRI | NULL | auto_increment |
| tipoUsuario | int(2) | YES | | NULL | |
| nombreUsuario | varchar(45) | YES | | NULL | |
| claveUsuario | varchar(300) | YES | | NULL | |
| emailUsuario | varchar(200) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

mysql>
```

Librerías

Para el desarrollo de la aplicación se han utilizado las siguientes librerías externas:

- ✓ [MySQL Connector/J 5.1.35](#)
- ✓ [IText 5.5.5](#)

Instalación

Para poder usar este programa es necesario los siguientes componentes:

- ✓ MySQL Server (debidamente configurado).
- ✓ Eclipse IDE for Java Developers (Oxygen.2 Release (4.7.2)).
- ✓ Jdk 1.8.0_161
- ✓ Comprimido del programa.








Una vez se cumplan estos requisitos, el primer paso es descomprimir la carpeta del programa en el escritorio. Dentro de la carpeta “ClubDeTenis” encontraremos un archivo llamado “Instalar” como administrador. Lo ejecutaremos para que así se instale la base de datos necesaria, entre otras cosas.

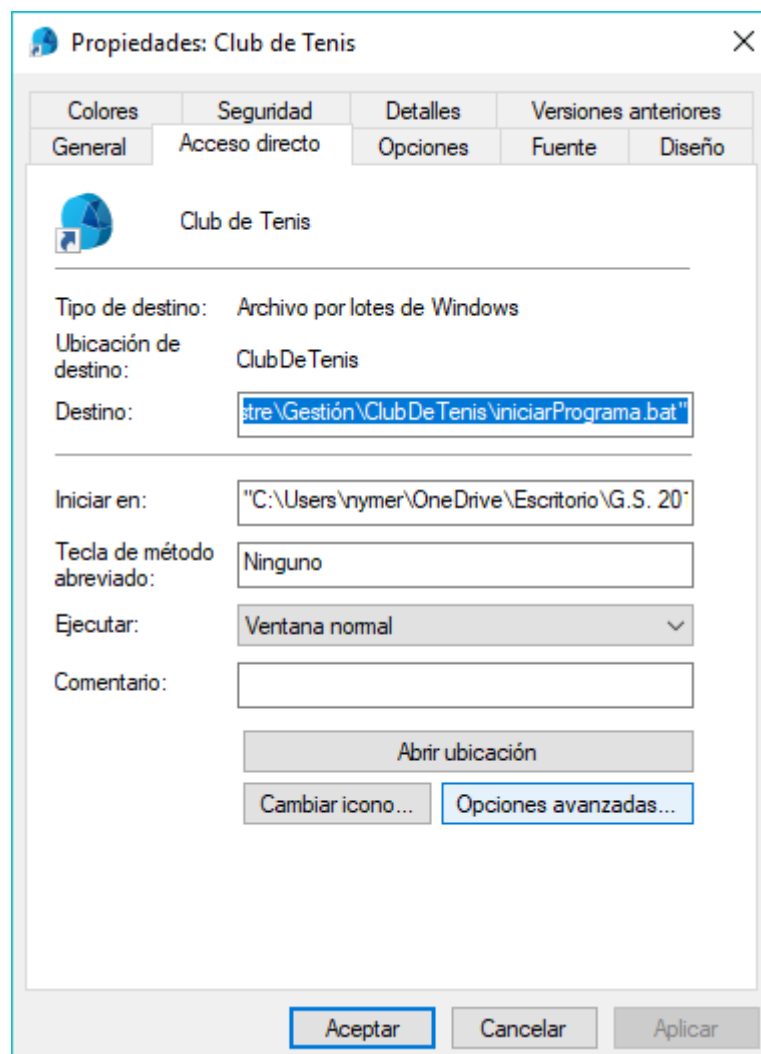
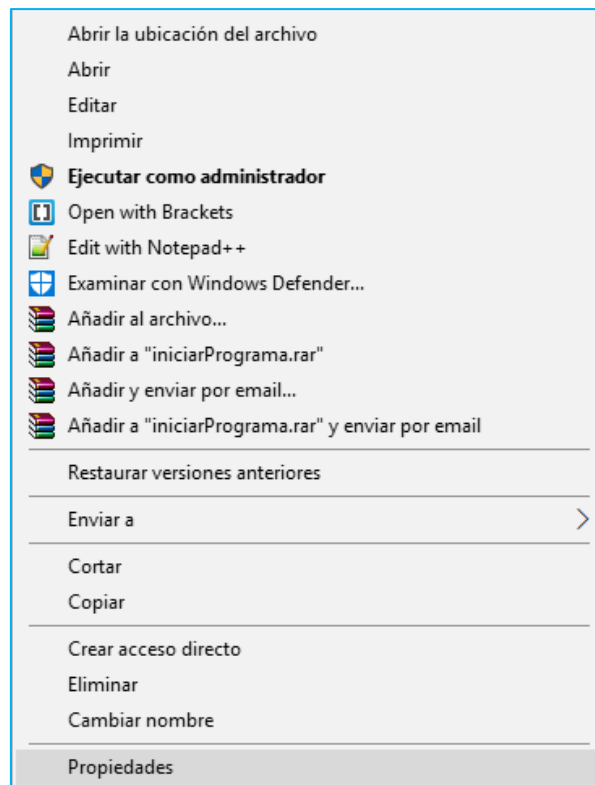
Antes de usarlo se debe crear la conexión ‘remoto’ con la contraseña ‘Studium2018;’.

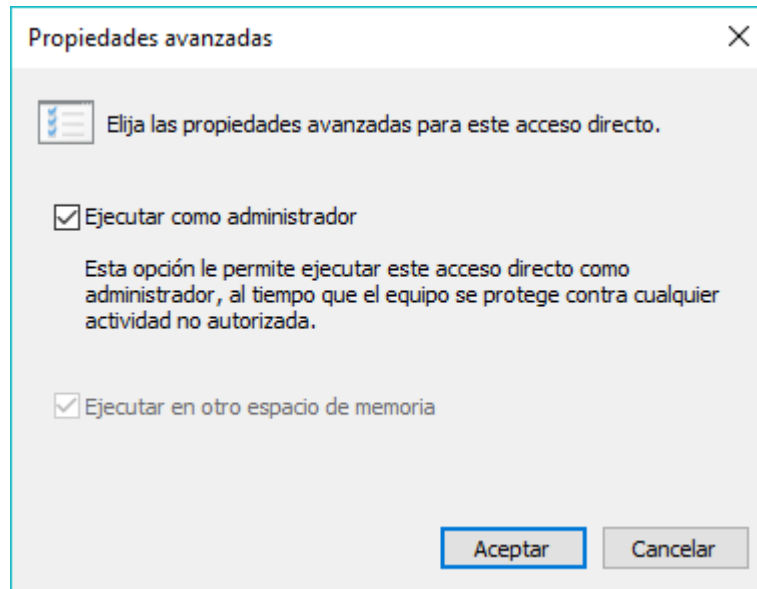
Podremos encontrar la carpeta con el programa y todos sus componentes en “C:\Program Files”.

Una vez que se cumplan estos requisitos, ya que está en una carpeta que necesita permisos de administrador, se puede abrir el programa de 3 maneras:

- Abrir un terminal como administrador y, estando en la ruta pertinente, escribir la siguiente sentencia: **java -jar ClubDeTenis.jar**
- Ir a la carpeta donde se encuentra el programa, buscar el archivo “Club de Tenis” y, pulsando sobre el con el botón derecho del ratón, seleccionar abrir como administrador.
- Ir a la carpeta donde se encuentra el programa, buscar el archivo “Club de Tenis” y, pulsando sobre el con el botón derecho del ratón, seleccionar “Propiedades”, “Opciones avanzadas...” y marcar la casilla “Ejecutar como administrador”. Luego le damos a “Aceptar/Aplicar/Aceptar”.

	ayudaClubDeTenis	08/06/2019 19:38	Carpeta de archivos	
	img	08/06/2019 19:38	Carpeta de archivos	
	Club de Tenis	08/06/2019 23:39	Acceso directo	3 KB
	ClubDeTenis	08/06/2019 22:46	Executable Jar File	13.621 KB
	gestion	08/06/2019 20:41	SQL Text File	7 KB
	iniciarPrograma	08/06/2019 23:26	Archivo por lotes ...	1 KB
	Instalar	08/06/2019 22:58	Archivo por lotes ...	1 KB





Para mayor comodidad se puede mover este archivo al escritorio. Si se sigue la última opción, se podrá usar a partir de entonces sin otro requerimiento o darle específicamente permiso de administrador de nuevo.

Código Java

Modelo.java

```
package ClubDeTenis;

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;

import javax.swing.DefaultListModel;
import javax.swing.JComboBox;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableColumnModel;

public class Modelo {

    /*#####*/
    /*## VARIABLES ##*/
    /*#####*/

    /*>> BASE DE DATOS <<*/
    String driver = "com.mysql.jdbc.Driver";
    String url =
"jdbc:mysql://localhost:3306/gestion?autoReconnect=true&useSSL=false";
    Connection connection = null;
    Statement statement = null;
    ResultSet rs = null;
    //USUARIO BASE DE DATOS
    String loginBD= "remoto";
    String passwordBD= "Stodium2018;";

    /*>> FECHA ORDENADOR <<*/
    Date fechaHora =new Date();
    DateFormat fechaHoraFormato = new SimpleDateFormat("dd-MM-yyyy HH:mm:ss");
```

```

/*#####*/
/*## METODOS ##*/
/*#####*/

/*>> METODOS BASE DE DATOS <<*/
//CONECTA MYSQL
public void conectaBase() {
    try{
        //CARGAR LOS CONTROLADORES PARA EL ACCESO A LA BASE DE DATOS
        Class.forName(driver);
        //ESTABLECER LA CONEXIÓN CON LA BASE DE DATOS
        connection = DriverManager.getConnection(url, loginBD,
passwordBD);
        //CREAR UNA SENTENCIA
        statement = connection.createStatement();
    }catch (ClassNotFoundException cnfe){
        System.out.println("Error 1: "+cnfe.getMessage());
    }catch (SQLException sqle){
        System.out.println("Error 2: "+sqle.getMessage());
    }
}
//DESCONECTA MYSQL
public void desconectaBase() {
    try{
        if(connection!=null){
            connection.close();
        }
    }catch (SQLException e){
        System.out.println("Error 3: "+e.getMessage());
    }
}
//BUSCA DATOS EN MYSQL MEDIANTE SELECT
public ResultSet buscarDatos(String sent) {
    try {
        rs=statement.executeQuery(sent);
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    return rs;
}
//DA DE ALTA, BAJA y MODIFICA LOS DATOS EN MYSQL
public void datosABM(String sent) {
    try {
        statement.executeUpdate(sent);
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

```

```

/*>> METODOS QUE RELLENAN COMPONENTES <<*/
//JCOMBOBOX PARA LAS COLUMNAS DE MYSQL
public void rellenarComboBoxCampos(String sent, JComboBox<String> caja ) {
    try {
        rs = statement.executeQuery(sent);
        int columnas = rs.getMetaData().getColumnCount();
        for (int i = 0; i<columnas; i++){
            caja.addItem(rs.getMetaData().getColumnLabel(i+1));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

//JCOMBOBOX PARA DATOS DE MYSQL
public void rellenarComboBoxDatos(String sent, JComboBox<String> caja ) {
    try {
        rs = statement.executeQuery(sent);
        while (rs.next()){
            caja.addItem(" "+rs.getString(1)+" - "+rs.getString(2)+"
"+ rs.getString(3)+" "+ rs.getString(4));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

//JCOMBOBOX NIVEL PROFESOR
public void rellenarComboBoxNivel(String sent, JComboBox<String> caja ) {
    try {
        caja.removeAllItems();
        caja.addItem("Selecciona uno...");
        String nivel;
        rs = statement.executeQuery(sent);
        rs.next();
        nivel= rs.getString(1);
        if (nivel.equals("Bajo")){
            caja.addItem("Bajo");
        }else if(nivel.equals("Medio")) {
            caja.addItem("Bajo");
            caja.addItem("Medio");
        }else {
            caja.addItem("Bajo");
            caja.addItem("Medio");
            caja.addItem("Alto");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

```

//JLIST SEGUN MYSQL
public void rellenarLista(String sent, DefaultListModel<String> lista ) {
    try {
        rs = statement.executeQuery(sent);
        while (rs.next()){
            lista.addElement(" "+rs.getString(1)+" -
"+rs.getString(2)+" "+ rs.getString(3)+" "+ rs.getString(4));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

//TABLA SEGUN MYSQL
public void rellenarTabla(String sent, JTable tabla) {
    try {
        TableColumnModel columnModel = tabla.getColumnModel();
        DefaultTableModel modeloTabla= (DefaultTableModel)
tabla.getModel();
        rs = statement.executeQuery(sent);
        int columnas = rs.getMetaData().getColumnCount();
        for (int i = 0; i<columnas; i++){

modeloTabla.addColumn(rs.getMetaData().getColumnLabel(i+1));
        }
        columnModel.getColumn(0).setPreferredWidth(15);
        columnModel.getColumn(1).setPreferredWidth(150);
        while (rs.next()){
            Object datos[] = new Object[columnas];
            for(int i=0; i<columnas; i++) {
                datos[i] = rs.getString(i+1);
            }
            modeloTabla.addRow(datos);
        }
    } catch (SQLException e) {
        System.out.println("Error rellenar tabla");
        e.printStackTrace();
    }
}

}

/*>> METODOS QUE CORRIGEN COMPONENTES <<*/
//ESCRIBE LA FECHA EN FORMATO MYSQL
public String formatoFechaSQL(String fecha) {
    String [] listaFecha = fecha.split("/");
    String newFecha = listaFecha[2]+"/"+listaFecha[1]+"/"+listaFecha[0];
    return newFecha;
}

//BORRA EL CONTENIDO DE JTEXTFIELD
public void borrarTxt (JTextField txt) {
    txt.selectAll();
    txt.setText("");
}
}

```

```
/*>> METODOS DE FICHEROS <<*/
//REGISTRAR MOVIMIENTOS
public void ficheroMovimiento(String mov) {
    try {
        //INDICA EL FICHERO DONDE SE GUARDA
        //TRUE: PARA QUE NO SE MACHAQUE EL TEXTO
        FileWriter fw= new FileWriter ("movimientos.log",true);
        //BUFFER PARA PODER ACCEDER AL FICHERO
        BufferedWriter bf=new BufferedWriter(fw);
        //PARA PODER ESCRIBIR EN EL FICHERO
        PrintWriter salida=new PrintWriter(bf);
        //PARA ESCRIBIR
        salida.println("[ "+fechaHoraFormato.format(fechaHora)+" ]"+mov);
        //CERRAR LO CREADO EN ORDEN INVERSO
        salida.close();
        bf.close();
        fw.close();
    }catch(IOException e) {
        System.out.println("Error");
    }
}
//ABRIR LA AYUDA
public void ficheroAyuda() {
    try{
        Runtime.getRuntime().exec("hh.exe
./ayudaClubDeTenis/ayudaClubDeTenis.chm");
    }catch (IOException e){
        e.printStackTrace();
    }
}
}
```

ImprimirPDF.java

```
package ClubDeTennis;

import java.awt.Desktop;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.net.MalformedURLException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JFrame;
import javax.swing.JOptionPane;

import com.itextpdf.text.BaseColor;
import com.itextpdf.text.Chunk;
import com.itextpdf.text.Document;
import com.itextpdf.text.DocumentException;
import com.itextpdf.text.Font;
import com.itextpdf.text.FontFactory;
import com.itextpdf.text.Image;
import com.itextpdf.text.Paragraph;
import com.itextpdf.text.pdf.PdfPTable;
import com.itextpdf.text.pdf.PdfWriter;

public class ImprimirPDF {

    /*#####*/
    /*## VARIABLES ##*/
    /*#####*/

    /*>> BASE DE DATOS <<*/
    String driver = "com.mysql.jdbc.Driver";
    String url =
"jdbc:mysql://localhost:3306/gestion?autoReconnect=true&useSSL=false";
    Connection connection = null;
    Statement statement = null;
    ResultSet rs = null;
    //USUARIO BASE DE DATOS
    String loginBD= "remoto";
    String passwordBD= "Studium2018";
```



```

/*#####*/
/*## METODO ##*/
/*#####*/

/*>> CREAR UN PDF <<*/
public ImprimirPDF(JFrame frame ,String tipoListado, String sent) throws
DocumentException, MalformedURLException, IOException, ClassNotFoundException{
    // SE CREA EL DOCUMENTO
    Document documento = new Document();
    try{
        // SE CREA EL OUTPUTSTREAM PARA EL FICHERO DONDE QUEREMOS DEJAR
EL PDF
        FileOutputStream ficheroPdf = new
FileOutputStream("Listado"+tipoListado+".pdf");
        // SE ASOCIA EL DOCUMENTO AL OUTPUTSTREAM Y SE INDICA QUE EL
ESPACIADO ENTRE
        // LINEAS SERA DE 20. ESTA LLAMADA DEBE HACERSE ANTES DE ABRIR
EL DOCUMENTO

        PdfWriter.getInstance(documento,ficheroPdf).setInitialLeading(20);
        // SE ABRE EL DOCUMENTO
        documento.open();
        // ESPECIFICA EL LOGO DE LA EMPRESA
        Image foto = Image.getInstance("./img/logo-tenis.png");
        foto.scaleToFit(100, 100);
        foto.setAlignment(Chunk.ALIGN_LEFT);
        // AÑADE LA FOTO AL DOCUMENTO
        documento.add(foto);
        // AÑADE LOS PARRAFOS AL PDF
        documento.add(new Paragraph("Club de Tenis",
FontFactory.getFont("arial", 22, Font.ITALIC,BaseColor.BLACK)));
        documento.add(new Paragraph(" "));
        Paragraph titulo = new Paragraph("Listado "+tipoListado,
FontFactory.getFont("arial", 35, Font.BOLD,BaseColor.BLACK));
        titulo.setAlignment(Chunk.ALIGN_CENTER);
        documento.add(titulo);
        documento.add(new Paragraph(" "));
        // CARGAR LOS CONTROLADORES PARA EL ACCESO A LA BASE DE DATOS
        Class.forName(driver);
        // ESTABLECE LA CONEXION CON AL BASE DE DATOS
        connection = DriverManager.getConnection(url, loginBD,
passwordBD);

        // CREA UNA SENTENCIA
        statement = connection.createStatement();
        rs = statement.executeQuery(sent);
        int columnas = rs.getMetaData().getColumnCount();
        // CREA LA TABLA
        PdfPTable tabla = new PdfPTable(columnas);
        // AÑADE LA CABECERA DE LA TABLA
        for (int i = 0; i<columnas; i++){
            tabla.addCell(rs.getMetaData().getColumnLabel(i+1));
        }
    }
}

```

```
// AÑADE LOS DATOS DE LA TABLA
while (rs.next())
{
    for(int i=0; i<columnas; i++) {
        tabla.addCell(rs.getString(i+1));
    }
}
// CIERRA LA CONEXIÓN A LA BASE DE DATOS
if(connection!=null){
    connection.close();
}
documento.add(tabla);
documento.close();
// MENSAJE DE CONFIRMACIÓN
JOptionPane.showMessageDialog(frame,"El PDF ha sido creado con
exito.");

// APERTURA DEL PDF
File objetofile = new File ("./Listado"+tipoListado+".pdf");
Desktop.getDesktop().open(objetofile);
} catch (SQLException | IOException e) {
    System.out.println("Error rellenar tabla");
    e.printStackTrace();
}
}
}
```

Login.java

```
package ClubDeTennis;

import java.awt.Color;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.HeadlessException;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;
import javax.swing.SwingConstants;

public class Login extends JFrame implements WindowListener, ActionListener,
MouseListener {
    private static final long serialVersionUID = 1L;

    /*#####*/
    /*## VARIABLES ##*/
    /*#####*/

    int type;
    String user, pass, mail;
    ResultSet resultado;
    //GENERAR MODELO
    Modelo objModelo = new Modelo();
```

```

/*>> COMPONENTES BÁSICOS <<*/
// ETIQUETAS
JLabel lblNom= new JLabel("Nombre:");
JLabel lblPass= new JLabel("Password:");
JLabel lblIcono= new JLabel();
JLabel lblIconoPreg= new JLabel();
JLabel lblOlvido= new JLabel("He olvidado mis datos");
JLabel lblEmail= new JLabel("Introduce tu email para recuperar tus
datos:");
JLabel lblTitulo= new JLabel("Club de Tennis");
// CASILLAS DE TEXTO
JTextField txtNom =new JTextField(20);
JPasswordField txtPass =new JPasswordField(20);
JTextField txtEmail =new JTextField(40);
// BOTONES
JButton btnLgn = new JButton("Login");
JButton btnBrr = new JButton("Borrar");
JButton btnAceptar = new JButton("Aceptar");
// PANELES Y DIÁLOGOS
JPanel pnl1 = new JPanel();
JDialog dlgOlv = new JDialog(this);

/*>> AÑADIR IMÁGENES <<*/
// LOGO
ImageIcon imagen= new ImageIcon("./img/logo-tenis.png");
Image img = imagen.getImage();
Image newimg = img.getScaledInstance(100, 100,
java.awt.Image.SCALE_SMOOTH);
ImageIcon imagentnf = new ImageIcon(newimg);
// ICONO PREGUNTA
ImageIcon imagenIconPreg= new ImageIcon("./img/iconoPregunta.png");
Image imgIconPreg = imagenIconPreg.getImage();
Image newimgIconPreg = imgIconPreg.getScaledInstance(50, 50,
java.awt.Image.SCALE_SMOOTH);
ImageIcon imagentntIconPreg = new ImageIcon(newimgIconPreg);

/*>> PERSONALIZACIÓN <<*/
// COLOR Y FUENTE
Color color=new Color(213, 245, 227);
Font fuenteTitulo = new Font("Calibri", 3, 20);
Font fuenteOlvidoIn = new Font("Dialog",1,13);
Font fuenteOlvidoOut = new Font("Dialog",1,12);
// PATRÓN EMAIL
String emailPatron = "^[_a-z0-9-]+(\\.[_a-z0-9-]+)*@" + "[a-z0-9-]+(\\.[a-
z0-9-]+)*(\\.[a-z]{2,4})$";
Pattern pattern = Pattern.compile(emailPatron);

```

```

/*#####*/
/*## CONSTRUCTOR ##*/
/*#####*/

/*>> LOGIN <<*/
public Login(){
    /*>> PARTES BÁSICAS DE LOGIN <<*/
    setTitle("Login");
    setLayout (new GridLayout(1,1));
    setSize(290, 360);
    setLocationRelativeTo(null);
    setResizable(false);
    pnl1.setLayout(null);
    // LOGO
    lblIcono.setIcon(imagentnf);
    pnl1.add(lblIcono);
    lblIcono.setBounds(95,15,100,100);
    // TITULO
    lblTitulo.setFont(fuenteTitulo);
    pnl1.add(lblTitulo);
    lblTitulo.setBounds(90,80,200,100);
    pnl1.add(lblNom);
    lblNom.setBounds(55,155,60,10);
    pnl1.add(txtNom);
    txtNom.setBounds(110,148,120,25);
    pnl1.add(lblPass);
    lblPass.setBounds(42,195,70,10);
    pnl1.add(txtPass);
    txtPass.setBounds(110,188,120,25);
    pnl1.add(btnLgn);
    btnLgn.setBounds(35,240,100,30);
    pnl1.add(btnBrr);
    btnBrr.setBounds(145,240,100,30);
    // ETIQUETA OLVIDO CONTRASEÑA
    lblOlvido.setHorizontalAlignment(SwingConstants.CENTER);
    pnl1.add(lblOlvido);
    lblOlvido.setBounds(65,300,150,30);
    lblOlvido.addMouseListener(this);
    pnl1.setBackground(color);
    add(pnl1);
    // DIÁLOGO EMAIL
    dlgOlv.setTitle("Restaurar datos");
    dlgOlv.setLayout (null);
    dlgOlv.setSize(350,150);
    dlgOlv.setResizable(false);
    dlgOlv.setLocationRelativeTo(null);
    dlgOlv.add(lblEmail);
    lblEmail.setBounds(60,10,300,20);
    lblIconoPreg.setIcon(imagentntIconPreg);
    dlgOlv.add(lblIconoPreg);
    lblIconoPreg.setBounds(5,10,50,50);
    dlgOlv.add(txtEmail);
    txtEmail.setBounds(60,35,260,25);

```

```
        dlgOlv.add(btnAceptar);
        btnAceptar.setBounds(130,70,100,30);
        /*>> LISTENERS Y VISIBILIDAD <<*/
        btnBrr.addActionListener(this);
        btnAceptar.addActionListener(this);
        btnLgn.addActionListener(this);
        addWindowListener(this);
        setVisible(true);
    }

    public static void main(String[] args) {

        new Login();

    }

    @Override
    public void windowActivated(WindowEvent e) {}
    @Override
    public void windowClosed(WindowEvent e) {}

    @Override
    public void windowClosing(WindowEvent e) {
        System.exit(0);
    }

    @Override
    public void windowDeactivated(WindowEvent e) {}
    @Override
    public void windowDeiconified(WindowEvent e) {}
    @Override
    public void windowIconified(WindowEvent e) {}
    @Override
    public void windowOpened(WindowEvent e) {}

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // BOTON BORRAR
        if(btnBrr.equals(arg0.getSource())) {
            objModelo.borrarTxt(txtNom);
            objModelo.borrarTxt(txtPass);
        }
    }
}
```

```

// BOTON LOGIN
if(btnLgn.equals(arg0.getSource())) {
    char[] passPers = txtPass.getPassword();
    pass = new String (passPers);
    user = txtNom.getText();
    String sentEntrada = "SELECT * FROM usuarios WHERE
nombreUsuario = '"+ user +"' AND claveUsuario = MD5('"+pass+"')";
    try {
        objModelo.conectaBase();
        resultado = objModelo.buscarDatos(sentEntrada);
        if(resultado.next()) {
            type=(int)resultado.getObject(2);
            if(type==0) {
                // FICHERO MOVIMIENTO
                String ficheroMovimiento =
"["+txtNom.getText()+"][Login]\n";
                objModelo.ficheroMovimiento(ficheroMovimiento);
                objModelo.borrarTxt(txtNom);
                objModelo.borrarTxt(txtPass);
                dispose();
                new MenuPrincipal(type, user);

            }else if(type==1){
                // FICHERO MOVIMIENTO
                String ficheroMovimiento =
"["+txtNom.getText()+"][Login]\n";
                objModelo.ficheroMovimiento(ficheroMovimiento);
                objModelo.borrarTxt(txtNom);
                objModelo.borrarTxt(txtPass);
                dispose();
                new MenuPrincipal(type, user);
            }
        }else {
            objModelo.borrarTxt(txtNom);
            objModelo.borrarTxt(txtPass);
            JOptionPane.showMessageDialog(dlgOlv,"El nombre o la
contraseña NO son correctos ", "Mensaje de ERROR",JOptionPane.ERROR_MESSAGE);
        }
    } catch (HeadlessException | SQLException e) {
        e.printStackTrace();
    }
    objModelo.desconectaBase();
}

// BOTON ACEPTAR EMAIL
if(btnAceptar.equals(arg0.getSource())) {
    mail=txtEmail.getText();
    String sentEmail = "SELECT * FROM usuarios WHERE emailUsuario =
'"+ mail +"'";
    objModelo.conectaBase();
    resultado = objModelo.buscarDatos(sentEmail);
    if (!txtEmail.getText().equals("")) {
        Matcher ObjPatron = pattern.matcher(txtEmail.getText());
        if (ObjPatron.matches()) {

```

```

        try {
            if(resultado.next()) {

                JOptionPane.showMessageDialog(dlgOlv,"Email de recuperación de datos
enviado ");

                dlgOlv.dispose();
            }else {

                JOptionPane.showMessageDialog(dlgOlv,"Este email no ha sido registrado
","Mensaje de ERROR",JOptionPane.ERROR_MESSAGE);
            }
        } catch (HeadlessException | SQLException e) {
            e.printStackTrace();
        }
    }else {
        JOptionPane.showMessageDialog(dlgOlv,"Formato de
email NO correcto ","Mensaje de ERROR",JOptionPane.ERROR_MESSAGE);
    }
    }else {
        JOptionPane.showMessageDialog(dlgOlv,"No se ha introducido
ningún email ","Mensaje de ERROR",JOptionPane.ERROR_MESSAGE);
    }
    objModelo.desconectaBase();
}
}
@Override
public void mouseClicked(MouseEvent arg0) {
    // LABEL CONTRASEÑA
    if(arg0.getComponent().equals(lblOlvido)) {
        objModelo.borrarTxt(txtEmail);
        dlgOlv.setVisible(true);
    }
}

@Override
public void mouseEntered(MouseEvent arg0) {
    // LABEL CONTRASEÑA
    lblOlvido.setFont(fuenteOlvidoIn);
}

@Override
public void mouseExited(MouseEvent arg0) {
    // LABEL CONTRASEÑA
    lblOlvido.setFont(fuenteOlvidoOut);
}

@Override
public void mousePressed(MouseEvent arg0) {}
@Override
public void mouseReleased(MouseEvent arg0) {}
}

```


MenuPrincipal.java

```
package ClubDeTenis;

import java.awt.Color;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
import javax.swing.JPanel;

public class MenuPrincipal extends JFrame implements WindowListener,
ActionListener, MouseListener{

    private static final long serialVersionUID = 1L;

    /*#####*/
    /*## VARIABLES ##*/
    /*#####*/

    String name;
    //GENERAR MODELO
    Modelo objModelo = new Modelo();

    /*>> COMPONENTES BÁSICOS <<*/
    // MENÚ
    JMenuBar barraMenu=new JMenuBar();
    JMenu menuSocios=new JMenu("Socios");
    JMenu menuProfesores=new JMenu("Profesores");
    JMenu menuClases=new JMenu("Clases");
    JMenu menuAyuda=new JMenu("Ayuda");
    JMenu menuCerrar=new JMenu("Cerrar sesión");
    JMenuItem miSociosAlta = new JMenuItem("Nuevo socio...");
    JMenuItem miSociosModificar =new JMenuItem("Modificar socio...");
    JMenuItem miSociosBaja=new JMenuItem("Eliminar socio...");
    JMenuItem miSociosListado= new JMenuItem("Listado socios...");
    JMenuItem miProfesoresAlta=new JMenuItem("Nuevo profesor...");
    JMenuItem miProfesoresModificar =new JMenuItem("Modificar profesor...");
    JMenuItem miProfesoresBaja=new JMenuItem("Eliminar profesor...");
```

```

JMenuItem miProfesoresListado=new JMenuItem("Listado profesores...");
JMenuItem miClasesAlta=new JMenuItem("Nueva clase...");
JMenuItem miClasesModificar =new JMenuItem("Modificar clase...");
JMenuItem miClasesListado=new JMenuItem("Listado clases...");
// ETIQUETAS
JLabel lblTitulo= new JLabel("Club de Tennis");
JLabel lblIcono= new JLabel();
// PANELES Y DIÁLOGOS
JPanel pnl1 = new JPanel();

/*>> PERSONALIZACIÓN <<*/
// COLOR Y FUENTE
Color color=new Color(213, 245, 227);
Font fuenteTitulo = new Font("Calibri", 3, 40);

/*>> AÑADIR IMÁGENES <<*/
// FONDO
ImageIcon imagen= new ImageIcon("./img/logo-tenis.png");
Image img = imagen.getImage();
Image newimg = img.getScaledInstance(300, 300,
java.awt.Image.SCALE_SMOOTH);
ImageIcon imagentnf = new ImageIcon(newimg);

/*#####*/
/*## CONSTRUCTOR ##*/
/*#####*/

/*>> MENÚ <<*/
public MenuPrincipal(int usu, String name){
    /*>> PARTES BÁSICAS DEL MENÚ <<*/
    this.name=name;
    setTitle("Menú");
    setLayout (new GridLayout(1,1));
    setSize(430, 460);
    setLocationRelativeTo(null);
    setResizable(false);
    pnl1.setLayout(null);
    // TÍTULO
    lblTitulo.setFont(fuenteTitulo);
    pnl1.add(lblTitulo);
    lblTitulo.setBounds(100,0,500,100);
    // LOGO
    lblIcono.setIcon(imagentnf);
    pnl1.add(lblIcono);
    lblIcono.setBounds(60, 80 , 300, 300);
    pnl1.setBackground(color);
    add(pnl1);
    // MENÚ
    menuSocios.add(miSociosAlta);
    menuProfesores.add(miProfesoresAlta);
    menuClases.add(miClasesAlta);
    if(usu==0) {
        menuSocios.add(miSociosModificar);
    }
}

```

```

        menuSocios.add(miSociosBaja);
        menuSocios.add(miSociosListado);
        menuProfesores.add(miProfesoresModificar);
        menuProfesores.add(miProfesoresBaja);
        menuProfesores.add(miProfesoresListado);
        menuClases.add(miClasesModificar);
        menuClases.add(miClasesListado);
    }
    barraMenu.add(menuSocios);
    barraMenu.add(menuProfesores);
    barraMenu.add(menuClases);
    barraMenu.add(menuAyuda);
    barraMenu.add(menuCerrar);
    setJMenuBar(barraMenu);
    /*>> LISTENERS Y VISIBILIDAD <<*/
    miSociosAlta.addActionListener(this);
    miSociosModificar.addActionListener(this);
    miSociosBaja.addActionListener(this);
    miSociosListado.addActionListener(this);
    miProfesoresAlta.addActionListener(this);
    miProfesoresModificar.addActionListener(this);
    miProfesoresBaja.addActionListener(this);
    miProfesoresListado.addActionListener(this);
    miClasesAlta.addActionListener(this);
    miClasesModificar.addActionListener(this);
    miClasesListado.addActionListener(this);
    menuAyuda.addMouseListener(this);
    menuCerrar.addMouseListener(this);
    setVisible(true);
}

```

```

@Override
public void actionPerformed(ActionEvent arg0) {

    /*>> SOCIOS <<*/
    if(miSociosAlta.equals(arg0.getSource())) {
        new Socio().nuevoSocio(this, name);
    }
    if(miSociosModificar.equals(arg0.getSource())) {
        new Socio().modificarSocio(this, name);
    }
    if(miSociosBaja.equals(arg0.getSource())) {
        new Socio().eliminarSocio(this, name);
    }
    if(miSociosListado.equals(arg0.getSource())) {
        new Socio().consultarSocio(this, name);
    }
}

```

```

/*>> PROFESORES <<*/
if(miProfesoresAlta.equals(arg0.getSource())) {
    new Profesor().nuevoProfesor(this, name);
}
if(miProfesoresModificar.equals(arg0.getSource())) {
    new Profesor().modificarProfesor(this, name);
}
if(miProfesoresBaja.equals(arg0.getSource())) {
    new Profesor().eliminarProfesor(this, name);
}
if(miProfesoresListado.equals(arg0.getSource())) {
    new Profesor().consultarProfesor(this, name);
}

/*>> CLASES <<*/
if(miClasesAlta.equals(arg0.getSource())) {
    new Clase().nuevaClase(this, name);
}
if(miClasesModificar.equals(arg0.getSource())) {
    new Clase().modificarClase(this, name);
}

if(miClasesListado.equals(arg0.getSource())) {
    new Clase().consultarClase(this, name);
}
}

@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}

@Override
public void windowClosing(WindowEvent arg0) {
    System.exit(0);
}

@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}

```

```
@Override
public void mouseClicked(MouseEvent e) {

    /*>> CERRAR SESIÓN <<*/
    if(e.getSource().equals(menuCerrar)) {
        int opcion = JOptionPane.showConfirmDialog(this,"¿Estás seguro
de cerrar la sesión?", "Cerrar sesión",JOptionPane.YES_NO_OPTION);
        if(opcion==JOptionPane.YES_OPTION) {
            String ficheroMovimiento = "["+name+"] [Cerrar sesión]\n";
            objModelo.ficheroMovimiento(ficheroMovimiento);
        }
        dispose();
        new Login();
    }

    /*>> ABRIR AYUDA <<*/
    if(e.getSource().equals(menuAyuda)) {
        objModelo.ficheroAyuda();
    }
}

@Override
public void mouseEntered(MouseEvent e) {}
@Override
public void mouseExited(MouseEvent e) {}
@Override
public void mousePressed(MouseEvent e) {}
@Override
public void mouseReleased(MouseEvent e) {}
}
```

Socio.java

```

package ClubDeTenis;

import java.awt.*;
import java.awt.event.*;
import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;

import com.itextpdf.text.DocumentException;

public class Socio extends JFrame implements WindowListener, ActionListener,
MouseListener{

    private static final long serialVersionUID = 1L;

    /*#####*/
    /*## VARIABLES ##*/
    /*#####*/

    JFrame menu = new JFrame();
    String name, campoMod;
    int indice;
    ResultSet resultado;
    //GENERAR MODELO
    Modelo objModelo = new Modelo();

    /*>> SENTENCIAS MYSQL <<*/
    // RELLENAR LISTA SOCIOS
    String sentenciaRellenarDatos = "SELECT idSocio, nombreSocio,
apellido1Socio, apellido2Socio FROM socios ORDER BY 2;";
    // RELLENAR COMBOBOX OPCIONES
    String sentenciaRellenarCampos = "SELECT nombreSocio AS 'Nombre',
apellido1Socio AS '1º Apellido', apellido2Socio AS '2º Apellido', dniSocio AS
'Dni', telefonoSocio AS 'Telefono' FROM socios ORDER BY 1;";
    // RELLENAR TABLA
    String sentenciaRellenarTabla = "SELECT idSocio AS 'ID',
concat(nombreSocio,' ',apellido1Socio,' ',apellido2Socio) AS 'NOMBRE', dniSocio
AS 'DNI', telefonoSocio AS 'TELEFONO' FROM socios ORDER BY 1;";

```

```

/*>> COMPONENTES BÁSICOS <<*/

/*>> ALTA <<*/
// ETIQUETAS
JLabel lblTituloAlt= new JLabel("NUEVO SOCIO");
JLabel lblNom= new JLabel("*Nombre:");
JLabel lblAp1= new JLabel("*Primer apellido:");
JLabel lblAp2= new JLabel(" Segundo apellido:");
JLabel lblDni= new JLabel("*DNI (Sin letra:)");
JLabel lblTel= new JLabel("Teléfono:");
JLabel lblAdv= new JLabel("--Los campos con * son obligatorios--");
// CASILLAS DE TEXTO
JTextField txtNomAlt =new JTextField(20);
JTextField txtAp1Alt =new JTextField(20);
JTextField txtAp2Alt =new JTextField(20);
JTextField txtDniAlt =new JTextField(20);
JTextField txtTelAlt =new JTextField(20);
// BOTONES
JButton btnBrrAlt = new JButton("Borrar todo");
JButton btnDarAlt = new JButton("Dar de alta");

/*>> MODIFICACIÓN <<*/
// ETIQUETAS
JLabel lblTituloMod= new JLabel("MODIFICAR SOCIO");
JLabel lblSelecMod= new JLabel("Socio seleccionado:");
JLabel lblCampMod= new JLabel("Seleccione campo a modificar:");
JLabel lblSocioMod =new JLabel();
// CASILLAS DE TEXTO
JTextField txtBuscMod =new JTextField(20);
JTextField txtModMod =new JTextField(20);
// BOTONES
JButton btnBuscMod = new JButton("Buscar");
JButton btnModMod = new JButton("Modificar");
JButton btnReinicioMod = new JButton("Reiniciar");
// LISTA Y JCOMBOBOX
DefaultListModel<String> modeloListaMod = new DefaultListModel<String>();
JList<String> lstMod = new JList<String>(modeloListaMod);
JComboBox<String> chclstAtrib = new JComboBox<String>();
// SEPARADOR Y SCROLL
JSeparator separador = new JSeparator();
JScrollPane scrollMod = new JScrollPane(lstMod);

```

```

/*>> BAJA <<*/
// ETIQUETAS
JLabel lblTituloBaj= new JLabel("ELIMINAR SOCIO");
JLabel lblSocioElm =new JLabel();
// BOTONES
JButton btnElmBaj = new JButton("Dar de baja");
JButton btnBuscElm = new JButton("Buscar");
JButton btnReiniciarElm = new JButton("Reiniciar");
// CASILLAS DE TEXTO
JTextField txtBuscElm =new JTextField(20);
// LISTA
DefaultListModel<String> modeloListaBaja = new DefaultListModel<String>();
JList <String> lstBaja = new JList <String>(modeloListaBaja);
// SCROLL
JScrollPane scrollBaja = new JScrollPane(lstBaja);

/*>> CONSULTA <<*/
// ETIQUETAS
JLabel lblTituloCons= new JLabel("LISTAR SOCIOS");
// TABLA
DefaultTableModel modeloTablaConsulta =new DefaultTableModel();
JTable tblSocios = new JTable(modeloTablaConsulta);
// BOTONES
JButton btnConsulta = new JButton("Crear PDF...");

// PANELES
JPanel pnl1 = new JPanel();
JPanel pnl2 = new JPanel();
// COLOR Y FUENTE
Color color=new Color(213, 245, 227);
Font fuenteTitulo = new Font("Calibri", 3, 20);

/*#####*/
/*## CONSTRUCTORES ##*/
/*#####*/

/*>> ALTA <<*/
public void nuevoSocio(JFrame menu, String name){
    /*>> PARTES BÁSICAS DEL MENÚ <<*/
    this.menu=menu;
    this.name=name;
    menu.setEnabled(false);
    setTitle("Nuevo socio");
    setLayout (new GridLayout(1,1));
    setSize(350, 320);
    setLocationRelativeTo(null);
    setResizable(false);
    pnl1.setLayout(null);
    // COMPONENTES
    lblTituloAlt.setFont(fuenteTitulo);
    pnl1.add(lblTituloAlt);
    lblTituloAlt.setBounds(120, 5, 300, 50);
    pnl1.add(lblNom);

```



```

        lblNom.setBounds(89, 60, 60, 20);
        pnl1.add(txtNomAlt);
        txtNomAlt.setBounds(150, 60, 150, 25);
        pnl1.add(lblAp1);
        lblAp1.setBounds(45, 90, 150, 20);
        pnl1.add(txtAp1Alt);
        txtAp1Alt.setBounds(150, 90, 150, 25);
        pnl1.add(lblAp2);
        lblAp2.setBounds(35, 120, 150, 20);
        pnl1.add(txtAp2Alt);
        txtAp2Alt.setBounds(150, 120, 150, 25);
        pnl1.add(lblDni);
        lblDni.setBounds(55, 150, 150, 20);
        pnl1.add(txtDniAlt);
        txtDniAlt.setBounds(150, 150, 150, 25);
        pnl1.add(lblTel);
        lblTel.setBounds(87, 180, 150, 20);
        pnl1.add(txtTelAlt);
        txtTelAlt.setBounds(150, 180, 150, 25);
        pnl1.add(lblAdv);
        lblAdv.setBounds(70, 260, 250, 30);
        pnl1.add(btnDarAlt);
        btnDarAlt.setBounds(35, 220, 120, 30);
        pnl1.add(btnBrrAlt);
        btnBrrAlt.setBounds(185, 220, 120, 30);
        pnl1.setBackground(color);
        add(pnl1);
        /*>> LISTENERS Y VISIBILIDAD <<*/
        btnDarAlt.addActionListener(this);
        btnBrrAlt.addActionListener(this);
        addWindowListener(this);
        setVisible(true);
    }

    /*>> MODIFICAR <<*/
    public void modificarSocio(JFrame menu, String name){
        this.menu=menu;
        this.name=name;
        menu.setEnabled(false);
        setTitle("Modificar socio");
        setLayout (new GridLayout(1,1));
        setSize(505, 290);
        setLocationRelativeTo(null);
        setResizable(false);
        pnl1.setLayout(null);
        // COMPONENTES
        lblTituloMod.setFont(fuenteTitulo);
        pnl1.add(lblTituloMod);
        lblTituloMod.setBounds(200, 5, 300, 50);
        pnl1.add(txtBuscMod);
        txtBuscMod.setBounds(10,60,140,25);
        pnl1.add(btnBuscMod);
        btnBuscMod.setBounds(155, 57, 80, 30);
    }

```

```

// LISTA SOCIOS
objModelo.conectaBase();
objModelo.rellenarLista(sentenciaRellenarDatos, modeloListaMod);
objModelo.desconectaBase();
pn1.add(scrollMod);
scrollMod.setBounds(10, 100, 225, 110);
pn1.add(btnReinicioMod);
btnReinicioMod.setBounds(70, 220, 100, 30);
pn1.add(lblSelecMod);
lblSelecMod.setBounds(285, 50, 260, 25);
pn1.add(lblSocioMod);
lblSocioMod.setBounds(300, 80, 200, 25);
pn1.add(lblCampMod);
lblCampMod.setBounds(285, 115, 260, 25);
// OPCIONES MODIFICAR
chcLstAtrib.addItem("Selecciona uno...");
objModelo.conectaBase();
objModelo.rellenarComboBoxCampos(sentenciaRellenarCampos,
chcLstAtrib);
objModelo.desconectaBase();
pn1.add(chcLstAtrib);
chcLstAtrib.setBounds(285, 140, 200, 25);
// SEPARADOR
separador.setOrientation(SwingConstants.VERTICAL);
pn1.add(separador);
separador.setBounds(260, 60, 2, 170);
pn1.add(txtModMod);
txtModMod.setBounds(285, 170, 200, 25);
pn1.add(btnModMod);
btnModMod.setBounds(335, 220, 100, 30);
pn1.setBackground(color);
add(pn1);
/*>> LISTENERS Y VISIBILIDAD <<*/
btnBuscMod.addActionListener(this);
btnModMod.addActionListener(this);
lstMod.addMouseListener(this);
chcLstAtrib.addActionListener(this);
btnReinicioMod.addActionListener(this);
addWindowListener(this);
setVisible(true);
}

/*>> MODIFICAR <<*/
public void eliminarSocio(JFrame menu, String name){
    this.menu=menu;
    this.name=name;
    menu.setEnabled(false);
    setTitle("Eliminar socio");
    setLayout (new GridLayout(1,1));
    setSize(290, 400);
    setLocationRelativeTo(null);
    setResizable(false);
    pn1.setLayout(null);

```

```

// COMPONENTES
lblTituloBaj.setFont(fuenteTitulo);
pn1.add(lblTituloBaj);
lblTituloBaj.setBounds(80, 5, 300, 50);
pn1.add(txtBuscElm);
txtBuscElm.setBounds(10,60,150,25);
pn1.add(btnBuscElm);
btnBuscElm.setBounds(170, 57, 100, 30);
// LISTA SOCIOS
objModelo.conectaBase();
objModelo.rellenarLista(sentenciaRellenarDatos, modeloListaBaja);
objModelo.desconectaBase();
pn1.add(scrollBaja);
scrollBaja.setBounds(10, 100, 260, 110);
pn1.add(btnReiniciarElm);
btnReiniciarElm.setBounds(10, 220, 260, 25);
pn1.add(lblSelecMod);
lblSelecMod.setBounds(10, 250, 260, 25);
pn1.add(lblSocioElm);
lblSocioElm.setBounds(10, 280, 260, 25);
pn1.add(btnElmBaj);
btnElmBaj.setBounds(90, 320, 100, 30);
pn1.setBackground(color);
add(pn1);
/*>> LISTENERS Y VISIBILIDAD <<*/
btnBuscElm.addActionListener(this);
btnReiniciarElm.addActionListener(this);
btnElmBaj.addActionListener(this);
lstBaja.addMouseListener(this);
addWindowListener(this);
setVisible(true);
}

/*>> LISTAR <<*/
public void consultarSocio(JFrame menu, String name){
    this.menu=menu;
    this.name=name;
    menu.setEnabled(false);
    setTitle("Listado socios");
    setLayout (new GridLayout(1, 1));
    setSize(570, 290);
    setLocationRelativeTo(null);
    setResizable(false);
    pn1.setLayout(null);
    pn2.setLayout(new BorderLayout());
    // COMPONENTES
    lblTituloCons.setFont(fuenteTitulo);
    pn1.add(lblTituloCons);
    lblTituloCons.setBounds(220, 10, 300, 50);
    objModelo.conectaBase();
    objModelo.rellenarTabla(sentenciaRellenarTabla, tblSocios);
    objModelo.desconectaBase();
    pn2.add(new JScrollPane(tblSocios), BorderLayout.CENTER);

```

```

pn11.add(pn12);
pn12.setBounds(18,70 , 530, 170);
pn12.add(btnConsulta, BorderLayout.SOUTH);
pn11.setBackground(color);
add(pn11);
/*>> LISTENERS Y VISIBILIDAD <<*/
btnConsulta.addActionListener(this);
addWindowListener(this);
setVisible(true);
}

@Override
public void actionPerformed(ActionEvent arg0) {

    /*>> ALTA <<*/
    // BOTÓN BORRAR
    if(btnBrrAlt.equals(arg0.getSource())) {
        objModelo.borrarTxt(txtNomAlt);
        objModelo.borrarTxt(txtAp1Alt);
        objModelo.borrarTxt(txtAp2Alt);
        objModelo.borrarTxt(txtDniAlt);
        objModelo.borrarTxt(txtTelAlt);
    }
    // BOTÓN DE ALTA
    if(btnDarAlt.equals(arg0.getSource())) {
        if((txtNomAlt.getText().equals("")) |
(txtAp1Alt.getText().equals("")) | (txtDniAlt.getText().equals(""))) {
            JOptionPane.showMessageDialog(this,"Uno o varios de los
campos obligatorios NO han sido rellenados ", "Mensaje de
ERROR",JOptionPane.ERROR_MESSAGE);
        }else {
            String sentencia ="INSERT INTO socios (nombreSocio,
apellido1Socio, apellido2Socio, dniSocio, telefonoSocio) VALUES
('"+txtNomAlt.getText()+"', '"+txtAp1Alt.getText()+"', '"+txtAp2Alt.getText()+"',
 '"+txtDniAlt.getText()+"', '"+txtTelAlt.getText()+"');";
            objModelo.conectaBase();
            objModelo.datosABM(sentencia);
            objModelo.desconectaBase();
            String ficheroMovimiento = "["+name+"]["+sentencia+"]\n";
            objModelo.ficheroMovimiento(ficheroMovimiento);
            JOptionPane.showMessageDialog (this,"El socio ha sido dado
de alta correctamente", "Confirmación",JOptionPane.INFORMATION_MESSAGE);
            menu.setEnabled(true);
            dispose();
        }
    }

    /*>> MODIFICACIÓN <<*/
    // BUSCAR SOCIO
    if(btnBuscMod.equals(arg0.getSource())){
        modeloListaMod.removeAllElements();
        String sentenciaBuscarMod ="SELECT idSocio, nombreSocio,
apellido1Socio,apellido2Socio FROM socios WHERE

```

```

concat(idSocio,nombreSocio,apellido1Socio,apellido2Socio) LIKE
'"'+txtBuscMod.getText()+"%' ORDER BY 2;";
    objModelo.conectaBase();
    objModelo.buscarDatos(sentenciaBuscarMod);
    objModelo.rellenarLista(sentenciaBuscarMod, modeloListaMod);
    objModelo.desconectaBase();
}
// REINICIAR SOCIO
if(btnReinicioMod.equals(arg0.getSource())){
    modeloListaMod.removeAllElements();
    objModelo.conectaBase();
    objModelo.buscarDatos(sentenciaRellenarDatos);
    objModelo.rellenarLista(sentenciaRellenarDatos,
modeloListaMod);
    objModelo.desconectaBase();
}
// OPCIONES ATRIBUTOS
if(chcLstAtrib.equals(arg0.getSource())) {
    if(chcLstAtrib.getSelectedIndex()!=0 &&
!lblSocioMod.getText().equals("")) {
        try {
            indice = chcLstAtrib.getSelectedIndex();
            String [] listaDato = lblSocioMod.getText().split("
");

            String sentenciaBuscarSocioMod ="SELECT * FROM
socios WHERE idSocio='"+listaDato[1]+"';";
            objModelo.conectaBase();

            resultado=objModelo.buscarDatos(sentenciaBuscarSocioMod);
            resultado.next();
            txtModMod.setText(resultado.getString(indice+1));
            objModelo.desconectaBase();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }else {
        objModelo.borrarTxt(txtModMod);
    }
}
// BOTÓN MODIFICAR
if(btnModMod.equals(arg0.getSource())){

    if(txtModMod.getText().equals("")|lblSocioMod.getText().equals("")|chcLstA
trib.getSelectedIndex()==0) {
        JOptionPane.showMessageDialog(this,"El socio
"+lblSocioMod.getText()+" NO ha sido modificado correctamente","Mensaje de
ERROR",JOptionPane.ERROR_MESSAGE);
    }else {
        try {
            String [] listaDato = lblSocioMod.getText().split("
");

            indice = chcLstAtrib.getSelectedIndex()+1;
            String sentenciaCampo ="SELECT * FROM socios;";

```

```

        objModelo.conectaBase();
        resultado= objModelo.buscarDatos(sentenciaCampo);
        campoMod =
resultado.getMetaData().getColumnName(indice);
        String sentenciaModificar ="UPDATE socios SET
"+campoMod+"='"+txtModMod.getText()+"'WHERE idSocio="+listaDato[1]+"";
        objModelo.datosABM(sentenciaModificar);
        objModelo.desconectaBase();
        String ficheroMovimiento =
 "["+name+"]["+sentenciaModificar+"]\n";
        objModelo.ficheroMovimiento(ficheroMovimiento);
        JOptionPane.showMessageDialog (this,"El socio
"+lblSocioMod.getText()+" ha sido modificado
correctamente", "Confirmación",JOptionPane.INFORMATION_MESSAGE);
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
menu.setEnabled(true);
dispose();
}

/*>> BAJA <<*/
// BUSCAR SOCIOS
if(btnBuscElm.equals(arg0.getSource())){
    modeloListaBaja.removeAllElements();
    String sentenciaBuscarMod ="SELECT idSocio, nombreSocio,
apellido1Socio,apellido2Socio FROM socios WHERE
concat(idSocio,nombreSocio,apellido1Socio,apellido2Socio) LIKE
'%" +txtBuscElm.getText()+"%" ORDER BY 2;";
    objModelo.conectaBase();
    objModelo.buscarDatos(sentenciaBuscarMod);
    objModelo.rellenarLista(sentenciaBuscarMod, modeloListaBaja);
    objModelo.desconectaBase();
}
// REINICIAR LISTA SOCIOS
if(btnReiniciarElm.equals(arg0.getSource())){
    modeloListaBaja.removeAllElements();
    objModelo.conectaBase();
    objModelo.buscarDatos(sentenciaRellenarDatos);
    objModelo.rellenarLista(sentenciaRellenarDatos,
modeloListaBaja);
    objModelo.desconectaBase();
}
// BOTÓN BAJA
if(btnElmBaj.equals(arg0.getSource())){
    if(!lblSocioElm.getText().equals("")) {
        int opcion = JOptionPane.showConfirmDialog(this,"¿Estás
seguro de eliminar a "+lblSocioElm.getText()+" ?", "Dar de
Baja",JOptionPane.YES_NO_OPTION);
        if(opcion==JOptionPane.YES_OPTION) {
            String [] listaDato = lblSocioElm.getText().split("
");

```

```

String sentencia ="DELETE FROM socios WHERE idSocio
= '"+listaDato[1]+'";
objModelo.conectaBase();
objModelo.datosABM(sentencia);
objModelo.desconectaBase();
String ficheroMovimiento =
 "["+name+"]["+sentencia+"]\n";
objModelo.ficheroMovimiento(ficheroMovimiento);
JOptionPane.showMessageDialog (this,"El socio
"+lblSocioElm.getText()+" ha sido eliminado
correctamente", "Confirmación",JOptionPane.INFORMATION_MESSAGE);
menu.setEnabled(true);
dispose();
    }
    }else {
        JOptionPane.showMessageDialog(this,"NO ha sido
seleccionado ningún socio", "Mensaje de ERROR",JOptionPane.ERROR_MESSAGE);
    }
}

/*>> CONSULTA <<*/
// BOTÓN CREAR PDF
if(btnConsulta.equals(arg0.getSource())) {
    try {
        String ficheroMovimiento = "["+name+"] [PDF Listado
Socios]\n";
        objModelo.ficheroMovimiento(ficheroMovimiento);
        new ImprimirPDF(this,"Socios", sentenciaRellenarTabla);
    } catch (DocumentException | IOException |
ClassNotFoundException e) {
        e.printStackTrace();
    }
}

@Override
public void windowActivated(WindowEvent e) {}
@Override
public void windowClosed(WindowEvent e) {}

@Override
public void windowClosing(WindowEvent e) {
    menu.setEnabled(true);
}

@Override
public void windowDeactivated(WindowEvent e) {}
@Override
public void windowDeiconified(WindowEvent e) {}
@Override
public void windowIconified(WindowEvent e) {}
@Override
public void windowOpened(WindowEvent e) {}

```



```
@Override
public void mouseClicked(MouseEvent e) {

    /*>> MODIFICACIÓN <<*/
    if(e.getSource().equals(lstMod)) {
        lblSocioMod.setText(lstMod.getSelectedValue());
        chcLstAtrib.setSelectedIndex(0);
        objModelo.borrarTxt(txtModMod);
    }

    /*>> BAJA <<*/
    if(e.getSource().equals(lstBaja)) {
        lblSocioElm.setText(lstBaja.getSelectedValue());
    }
}

@Override
public void mouseEntered(MouseEvent e) {}
@Override
public void mouseExited(MouseEvent e) {}
@Override
public void mousePressed(MouseEvent e) {}
@Override
public void mouseReleased(MouseEvent e) {}

}
```


Profesor.java

```
package ClubDeTenis;

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.swing.DefaultListModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JList;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JSeparator;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.SwingConstants;
import javax.swing.table.DefaultTableModel;

import com.itextpdf.text.DocumentException;

public class Profesor extends JFrame implements WindowListener, ActionListener,
MouseListener {

    private static final long serialVersionUID = 1L;

    /*#####*/
    /*## VARIABLES ##*/
    /*#####*/

    JFrame menu = new JFrame();
    String name, campoMod;
    int indice;
    ResultSet resultado;
    //GENERAR MODELO
    Modelo objModelo = new Modelo();

    /*>> SENTENCIAS MYSQL <<*/
    // RELLENAR LISTA PROFESORES
```

```

String sentenciaRellenarDatos = "SELECT idProfesor, nombreProfesor,
apellido1Profesor, apellido2Profesor FROM profesores ORDER BY 2;";
// RELLENAR COMBOBOX OPCIONES
String sentenciaRellenarCampos = "SELECT nombreProfesor AS 'Nombre',
apellido1Profesor AS '1º Apellido', apellido2Profesor AS '2º Apellido',
dniProfesor AS 'Dni', nivelProfesor AS 'Nivel' FROM profesores ORDER BY 1;";
// RELLENAR TABLA
String sentenciaRellenarTabla = "SELECT idProfesor AS 'ID',
concat(nombreProfesor,' ',apellido1Profesor,' ',apellido2Profesor) AS 'NOMBRE',
dniProfesor AS 'DNI', nivelProfesor AS 'NIVEL' FROM profesores ORDER BY 1;";

/*>> COMPONENTES BÁSICOS <<*/

/*>> ALTA <<*/
// ETIQUETAS
JLabel lblTituloAlt= new JLabel("NUEVO PROFESOR");
JLabel lblNom= new JLabel("*Nombre:");
JLabel lblAp1= new JLabel("*Primer apellido:");
JLabel lblAp2= new JLabel(" Segundo apellido:");
JLabel lblDni= new JLabel("*DNI (Sin letra:)");
JLabel lblNvl= new JLabel("*Nivel:");
JLabel lblAdv= new JLabel("--Los campos con * son obligatorios--");
// CASILLAS DE TEXTO
JTextField txtNomAlt =new JTextField(20);
JTextField txtAp1Alt =new JTextField(20);
JTextField txtAp2Alt =new JTextField(20);
JTextField txtDniAlt =new JTextField(20);
// BOTONES
JButton btnBrrAlt = new JButton("Borrar todo");
JButton btnDarAlt = new JButton("Dar de alta");
// JCOMBOBOX
JComboBox<String> chcLstNvl = new JComboBox<String>();

/*>> MODIFICACIÓN <<*/
// ETIQUETAS
JLabel lblTituloMod= new JLabel("MODIFICAR PROFESOR");
JLabel lblSeleccMod= new JLabel("Profesor seleccionado:");
JLabel lblCampMod= new JLabel("Seleccione campo a modificar:");
JLabel lblProfMod =new JLabel();
// CASILLAS DE TEXTO
JTextField txtBuscMod =new JTextField(20);
JTextField txtModMod =new JTextField(20);
// BOTONES
JButton btnBuscMod = new JButton("Buscar");
JButton btnModMod = new JButton("Modificar");
JButton btnReinicioMod = new JButton("Reiniciar");
// LISTA Y JCOMBOBOX
DefaultListModel<String> modeloListaMod = new DefaultListModel<String>();
JList<String> lstMod = new JList<String>(modeloListaMod);
JComboBox<String> chcLstAtrib = new JComboBox<String>();
// SEPARADOR Y SCROLL
JSeparator separador = new JSeparator();
JScrollPane scrollMod = new JScrollPane(lstMod);

```

```

/*>> BAJA <<*/
//  ETIQUETAS
JLabel lblTituloBaj= new JLabel("ELIMINAR PROFESOR");
JLabel lblProfElm =new JLabel();
// BOTONES
JButton btnElmBaj = new JButton("Dar de baja");
JButton btnBuscElm = new JButton("Buscar");
JButton btnReiniciarElm = new JButton("Reiniciar");
// CASILLAS DE TEXTO
JTextField txtBuscElm =new JTextField(20);
// LISTA
DefaultListModel<String> modeloListaBaja = new DefaultListModel<String>();
JList <String> lstBaja = new JList <String>(modeloListaBaja);
// SCROLL
JScrollPane scrollBaja = new JScrollPane(lstBaja);

/*>> CONSULTA <<*/
//  ETIQUETAS
JLabel lblTituloCons= new JLabel("LISTAR PROFESORES");
// TABLA
DefaultTableModel modeloTablaConsulta =new DefaultTableModel();
JTable tblProfesores = new JTable(modeloTablaConsulta);
// BOTONES
JButton btnConsulta = new JButton("Crear PDF...");

// PANELES
JPanel pnl1 = new JPanel();
JPanel pnl2 = new JPanel();
// COLOR Y FUENTE
Color color=new Color(213, 245, 227);
Font fuenteTitulo = new Font("Calibri", 3, 20);

/*****
/**** CONSTRUCTORES ****
*****/

/*>> ALTA <<*/
public void nuevoProfesor(JFrame menu, String name){
    /*>> PARTES BÁSICAS DEL MENÚ <<*/
    this.menu=menu;
    this.name=name;
    menu.setEnabled(false);
    setTitle("Nuevo profesor");
    setLayout (new GridLayout(1,1));
    setSize(350, 320);
    setLocationRelativeTo(null);
    setResizable(false);
    pnl1.setLayout(null);
    // COMPONENTES
    lblTituloAlt.setFont(fuenteTitulo);
    pnl1.add(lblTituloAlt);
    lblTituloAlt.setBounds(100, 5, 300, 50);

```

```

pn11.add(lblNom);
lblNom.setBounds(89, 60, 60, 20);
pn11.add(txtNomAlt);
txtNomAlt.setBounds(150, 60, 150, 25);
pn11.add(lblAp1);
lblAp1.setBounds(45, 90, 150, 20);
pn11.add(txtAp1Alt);
txtAp1Alt.setBounds(150, 90, 150, 25);
pn11.add(lblAp2);
lblAp2.setBounds(35, 120, 150, 20);
pn11.add(txtAp2Alt);
txtAp2Alt.setBounds(150, 120, 150, 25);
pn11.add(lblDni);
lblDni.setBounds(55, 150, 150, 20);
pn11.add(txtDniAlt);
txtDniAlt.setBounds(150, 150, 150, 25);
pn11.add(lblNvl);
lblNvl.setBounds(105, 180, 40, 20);
// DESPLEGABLE NIVEL PROFESORES
chcLstNvl.addItem("Selecciona uno...");
chcLstNvl.addItem("Bajo");
chcLstNvl.addItem("Medio");
chcLstNvl.addItem("Alto");
pn11.add(chcLstNvl);
chcLstNvl.setBounds(150, 180, 150, 25);
pn11.add(lblAdv);
lblAdv.setBounds(70, 260, 250, 30);
pn11.add(btnDarAlt);
btnDarAlt.setBounds(35, 220, 120, 30);
pn11.add(btnBrrAlt);
btnBrrAlt.setBounds(185, 220, 120, 30);
pn11.setBackground(color);
add(pn11);
/*>> LISTENERS Y VISIBILIDAD <<*/
btnDarAlt.addActionListener(this);
btnBrrAlt.addActionListener(this);
addWindowListener(this);
setVisible(true);
}

/*>> MODIFICAR <<*/
public void modificarProfesor(JFrame menu, String name){
    this.menu=menu;
    this.name=name;
    menu.setEnabled(false);
    setTitle("Modificar profesor");
    setLayout (new GridLayout(1,1));
    setSize(505, 290);
    setLocationRelativeTo(null);
    setResizable(false);
    pn11.setLayout(null);
    // COMPONENTES
    lblTituloMod.setFont(fuenteTitulo);

```

```

pn11.add(lblTituloMod);
lblTituloMod.setBounds(180, 5, 300, 50);
pn11.add(txtBuscMod);
txtBuscMod.setBounds(10,60,140,25);
pn11.add(btnBuscMod);
btnBuscMod.setBounds(155, 57, 80, 30);
// LISTA PROFESOR
objModelo.conectaBase();
objModelo.rellenarLista(sentenciaRellenarDatos, modeloListaMod);
objModelo.desconectaBase();
pn11.add(scrollMod);
scrollMod.setBounds(10, 100, 225, 110);
pn11.add(btnReinicioMod);
btnReinicioMod.setBounds(70, 220, 100, 30);
pn11.add(lblSelecMod);
lblSelecMod.setBounds(285, 50, 260, 25);
pn11.add(lblProfMod);
lblProfMod.setBounds(285, 80, 200, 25);
pn11.add(lblCampMod);
lblCampMod.setBounds(285, 115, 260, 25);
// OPCIONES MODIFICAR
chcLstAtrib.addItem("Selecciona uno...");
objModelo.conectaBase();
objModelo.rellenarComboBoxCampos(sentenciaRellenarCampos,
chcLstAtrib);
objModelo.desconectaBase();
pn11.add(chcLstAtrib);
chcLstAtrib.setBounds(285, 140, 200, 25);
// SEPARADOR
separador.setOrientation(SwingConstants.VERTICAL);
pn11.add(separador);
separador.setBounds(260, 60, 2, 170);
pn11.add(txtModMod);
txtModMod.setBounds(285, 170, 200, 25);
pn11.add(btnModMod);
btnModMod.setBounds(335, 220, 100, 30);
pn11.setBackground(color);
add(pn11);
/*>> LISTENERS Y VISIBILIDAD <<*/
btnBuscMod.addActionListener(this);
btnModMod.addActionListener(this);
lstMod.addMouseListener(this);
chcLstAtrib.addActionListener(this);
btnReinicioMod.addActionListener(this);
addWindowListener(this);
setVisible(true);
}

/*>> BAJA <<*/
public void eliminarProfesor(JFrame menu, String name){
    this.menu=menu;
    this.name= name;
    menu.setEnabled(false);
}

```

```

setTitle("Eliminar profesor");
setLayout (new GridLayout(1,1));
setSize(290, 400);
setLocationRelativeTo(null);
setResizable(false);
pn1.setLayout(null);
// COMPONENTES
lblTituloBaj.setFont(fuenteTitulo);
pn1.add(lblTituloBaj);
lblTituloBaj.setBounds(60, 5, 300, 50);
pn1.add(txtBuscElm);
txtBuscElm.setBounds(10,60,150,25);
pn1.add(btnBuscElm);
btnBuscElm.setBounds(170, 57, 100, 30);
// LISTA PROFESORES
objModelo.conectaBase();
objModelo.rellenarLista(sentenciaRellenarDatos, modeloListaBaja);
objModelo.desconectaBase();
pn1.add(scrollBaja);
scrollBaja.setBounds(10, 100, 260, 110);
pn1.add(btnReiniciarElm);
btnReiniciarElm.setBounds(10, 220, 260, 25);
pn1.add(lblSelecMod);
lblSelecMod.setBounds(10, 250, 260, 25);
pn1.add(lblProfElm);
lblProfElm.setBounds(10, 280, 260, 25);
pn1.add(btnElmBaj);
btnElmBaj.setBounds(90, 320, 100, 30);
pn1.setBackground(color);
add(pn1);
/*>> LISTENERS Y VISIBILIDAD <<*/
btnBuscElm.addActionListener(this);
btnReiniciarElm.addActionListener(this);
btnElmBaj.addActionListener(this);
lstBaja.addMouseListener(this);
addWindowListener(this);
setVisible(true);
}

/*>> LISTAR <<*/
public void consultarProfesor(JFrame menu, String name){
    this.menu=menu;
    this.name=name;
    menu.setEnabled(false);
    setTitle("Listado profesores");
    layout (new GridLayout(1,1));
    setSize(570, 290);
    setLocationRelativeTo(null);
    setResizable(false);
    pn1.setLayout(null);
    pn2.setLayout(new BorderLayout());
    // COMPONENTES
    lblTituloCons.setFont(fuenteTitulo);

```

```

pnl1.add(lblTituloCons);
lblTituloCons.setBounds(220, 10, 300, 50);
// TABLA
objModelo.conectaBase();
objModelo.rellenarTabla(sentenciaRellenarTabla, tblProfesores);
objModelo.desconectaBase();
pnl2.add(new JScrollPane(tblProfesores), BorderLayout.CENTER);
pnl1.add(pnl2);
pnl2.setBounds(18, 70, 530, 170);
pnl2.add(btnConsulta, BorderLayout.SOUTH);
pnl1.setBackground(color);
add(pnl1);
/*>> LISTENERS Y VISIBILIDAD <<*/
btnConsulta.addActionListener(this);
addWindowListener(this);
setVisible(true);
}

```

```

@Override
public void mouseClicked(MouseEvent arg0) {

    /*>> MODIFICACIÓN <<*/
    if(arg0.getSource().equals(lstMod)) {
        lblProfMod.setText(lstMod.getSelectedValue());
        chcLstAtrib.setSelectedIndex(0);
        objModelo.borrarTxt(txtModMod);
    }

    /*>> BAJA <<*/
    if(arg0.getSource().equals(lstBaja)) {
        lblProfElm.setText(lstBaja.getSelectedValue());
    }

}

```

```

@Override
public void mouseEntered(MouseEvent arg0) {}
@Override
public void mouseExited(MouseEvent arg0) {}
@Override
public void mousePressed(MouseEvent arg0) {}
@Override
public void mouseReleased(MouseEvent arg0) {}

@Override
public void actionPerformed(ActionEvent arg0) {

    /*>> ALTA <<*/
    // BOTÓN BORRAR
    if(btnBrrAlt.equals(arg0.getSource())) {
        objModelo.borrarTxt(txtNomAlt);
        objModelo.borrarTxt(txtAp1Alt);
        objModelo.borrarTxt(txtAp2Alt);
    }
}

```



```

        objModelo.borrarTxt(txtDniAlt);
        chcLstNvl.setSelectedIndex(0);
    }
    // BOTON ALTA
    if(btnDarAlt.equals(arg0.getSource())) {
        if((txtNomAlt.getText().equals("")) |
(txtAp1Alt.getText().equals("")) |
(txtDniAlt.getText().equals(""))|(chcLstNvl.getSelectedIndex()==0)) {
            JOptionPane.showMessageDialog(this,"Uno o varios de los
campos obligatorios NO han sido rellenados ", "Mensaje de
ERROR",JOptionPane.ERROR_MESSAGE);
        }else {
            String sentenciaAlta ="INSERT INTO profesores
(nombreProfesor, apellido1Profesor, apellido2Profesor, dniProfesor,
nivelProfesor) VALUES
('"+txtNomAlt.getText()+"', '"+txtAp1Alt.getText()+"', '"+txtAp2Alt.getText()+"',
'"+txtDniAlt.getText()+"', '"+chcLstNvl.getSelectedItem()+"')";
            objModelo.conectaBase();
            objModelo.datosABM(sentenciaAlta);
            objModelo.desconectaBase();
            String ficheroMovimiento =
 "["+name+"]["+sentenciaAlta+"]\n";
            objModelo.ficheroMovimiento(ficheroMovimiento);
            JOptionPane.showMessageDialog (this,"El profesor ha sido
dado de alta correctamente", "Confirmación",JOptionPane.INFORMATION_MESSAGE);
            menu.setEnabled(true);
            dispose();
        }
    }

    /*>> MODIFICACIÓN <<*/
    // BUSCAR PROFESOR
    if(btnBuscMod.equals(arg0.getSource())){
        modeloListaMod.removeAllElements();
        String sentenciaBuscarMod ="SELECT idProfesor, nombreProfesor,
apellido1Profesor,apellido2Profesor FROM profesores WHERE
concat(idProfesor,nombreProfesor,apellido1Profesor,apellido2Profesor) LIKE
'"+txtBuscMod.getText()+"%' ORDER BY 2;";
        objModelo.conectaBase();
        objModelo.buscarDatos(sentenciaBuscarMod);
        objModelo.rellenarLista(sentenciaBuscarMod, modeloListaMod);
        objModelo.desconectaBase();
    }
    // REINICIAR LISTA PROFESORES
    if(btnReinicioMod.equals(arg0.getSource())){
        modeloListaMod.removeAllElements();
        objModelo.conectaBase();
        objModelo.buscarDatos(sentenciaRellenarDatos);
        objModelo.rellenarLista(sentenciaRellenarDatos,
modeloListaMod);
        objModelo.desconectaBase();
    }
    // RELLENAR OPCIONES

```



```

        if(chcLstAtrib.equals(arg0.getSource())) {
            if(chcLstAtrib.getSelectedIndex()!=0 &&
!lblProfMod.getText().equals("")) {
                try {
                    indice = chcLstAtrib.getSelectedIndex();
                    //Separador de texto
                    String [] listaDato = lblProfMod.getText().split("
");

                    String sentenciaBuscarProfeMod ="SELECT * FROM
profesores WHERE idProfesor='"+listaDato[1]+'";
                    objModelo.conectaBase();

                    resultado=objModelo.buscarDatos(sentenciaBuscarProfeMod);
                    resultado.next();
                    txtModMod.setText(resultado.getString(indice+1));
                    objModelo.desconectaBase();
                } catch (SQLException e) {
                    e.printStackTrace();
                }
            }else {
                objModelo.borrarTxt(txtModMod);
            }
        }
        // BOTÓN MODIFICAR
        if(btnModMod.equals(arg0.getSource())){
            if(txtModMod.getText().equals("")|
lblProfMod.getText().equals("")|chcLstAtrib.getSelectedIndex()==0) {
                JOptionPane.showMessageDialog(this,"El profesor
"+lblProfMod.getText()+" no ha sido modificado correctamente","Mensaje de
ERROR",JOptionPane.ERROR_MESSAGE);
            }else {
                try {
                    String [] listaDato = lblProfMod.getText().split("
");

                    indice = chcLstAtrib.getSelectedIndex()+1;
                    String sentenciaCampo ="SELECT * FROM profesores";
                    objModelo.conectaBase();
                    resultado= objModelo.buscarDatos(sentenciaCampo);
                    campoMod =
resultado.getMetaData().getColumnName(indice);
                    String sentenciaModificar ="UPDATE profesores SET
"+campoMod+"='"+txtModMod.getText()+"'WHERE idProfesor='"+listaDato[1]+'";
                    objModelo.datosABM(sentenciaModificar);
                    objModelo.desconectaBase();
                    String ficheroMovimiento =
 "["+name+"]["+sentenciaModificar+"]\n";
                    objModelo.ficheroMovimiento(ficheroMovimiento);
                    JOptionPane.showMessageDialog (this,"El profesor
"+lblProfMod.getText()+" ha sido modificado
correctamente","Confirmación",JOptionPane.INFORMATION_MESSAGE);

                }catch(SQLException er) {
                    System.out.println("Error");
                }
            }
        }
    }
}

```

```

    }
}

menu.setEnabled(true);
dispose();

}

/*>> BAJA <<*/
// BUSCAR PROFESOR
if(btnBuscElm.equals(arg0.getSource())){
    modeloListaBaja.removeAllElements();
    //Rellenar lista segun txt
    String sentenciaBuscarMod ="SELECT idProfesor, nombreProfesor,
apellido1Profesor,apellido2Profesor FROM profesores WHERE
concat(idProfesor,nombreProfesor,apellido1Profesor,apellido2Profesor) LIKE
'%" +txtBuscElm.getText()+"%' ORDER BY 2;";
    objModelo.conectaBase();
    objModelo.buscarDatos(sentenciaBuscarMod);
    objModelo.rellenarLista(sentenciaBuscarMod, modeloListaBaja);
    objModelo.desconectaBase();
}
// REINICIAR LISTA PROFESOR
if(btnReiniciarElm.equals(arg0.getSource())){
    modeloListaBaja.removeAllElements();
    objModelo.conectaBase();
    objModelo.buscarDatos(sentenciaRellenarDatos);
    objModelo.rellenarLista(sentenciaRellenarDatos,
modeloListaBaja);
    objModelo.desconectaBase();
}
// BOTÓN BAJA
if(btnElmBaj.equals(arg0.getSource())){
    if(!lblProfElm.getText().equals("")) {
        int opcion = JOptionPane.showConfirmDialog(this,"¿Estás
seguro de eliminar a "+lblProfElm.getText()+" ?", "Dar de
baja",JOptionPane.YES_NO_OPTION);
        if(opcion==JOptionPane.YES_OPTION) {
            String [] listaDato = lblProfElm.getText().split("
");
            String sentenciaBorrar ="DELETE FROM profesores
WHERE idProfesor = '"+listaDato[1]+"'";
            objModelo.conectaBase();
            objModelo.datosABM(sentenciaBorrar);
            objModelo.desconectaBase();
            String ficheroMovimiento =
 "["+name+"]["+sentenciaBorrar+"]\n";
            objModelo.ficheroMovimiento(ficheroMovimiento);
            JOptionPane.showMessageDialog (this,"El profesor
"+lblProfElm.getText()+" ha sido eliminado
correctamente", "Confirmación",JOptionPane.INFORMATION_MESSAGE);
            menu.setEnabled(true);
            dispose();
        }
    }
}

```

```

    }
    }else {
        JOptionPane.showMessageDialog(this,"No ha sido
seleccionado ningún profesor","Mensaje de ERROR",JOptionPane.ERROR_MESSAGE);
    }
}

/*>> CONSULTA <<*/
// BOTÓN CREAR PDF
if(btnConsulta.equals(arg0.getSource())) {
    try {
        String ficheroMovimiento = "["+name+"][PDF Listado
Profesores]\n";
        objModelo.ficheroMovimiento(ficheroMovimiento);
        new ImprimirPDF(this,"Profesores",
sentenciaRellenarTabla);
    } catch (DocumentException | IOException |
ClassNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
}

@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}

@Override
public void windowClosing(WindowEvent arg0) {
    menu.setEnabled(true);
}

@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}
}

```

Clase.java

```
package ClubDeTenis;

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.swing.DefaultListModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JList;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JSeparator;
import javax.swing.JSpinner;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.SpinnerNumberModel;
import javax.swing.SwingConstants;
import javax.swing.table.DefaultTableModel;

import com.itextpdf.text.DocumentException;

public class Clase extends JFrame implements WindowListener, ActionListener,
MouseListener{

    private static final long serialVersionUID = 1L;

    /*#####*/
    /*## VARIABLES ##*/
    /*#####*/

    JFrame menu = new JFrame();
    String name, campoMod;
    int indice;
    ResultSet resultado;
    //GENERAR MODELO
    Modelo objModelo = new Modelo();
```

```

/*>> SENTENCIAS MYSQL <<*/
// RELLENAR LISTA PROFESORES
String sentenciaRellenarProfesores = "SELECT idProfesor, nombreProfesor,
apellido1Profesor, apellido2Profesor FROM profesores ORDER BY 2;";
// RELLENAR LISTA CLASE
String sentenciaRellenarClase = "SELECT idClase, horaClase,
concat(',',nivelClase,')', concat(nombreProfesor,' ', apellido1Profesor) FROM
clases, profesores WHERE clases.idProfesorFK=profesores.idProfesor ORDER BY
2;";
// RELLENAR COMBOBOX
String sentenciaRellenarCampos = "SELECT horaClase AS 'Hora', precioClase
AS 'Precio' FROM clases ORDER BY 1;";
// RELLENAR TABLA
String sentenciaRellenarTabla = "SELECT idClase AS 'ID',
concat(nombreProfesor,' ', apellido1Profesor) AS 'PROFESOR',nivelClase AS
'Nivel', horaClase AS 'HORA', precioClase AS 'PRECIO', Count(idSocioFK) AS 'Nº
Socios' FROM profesores, clases, solicitudes WHERE idClase=idClaseFK AND
idProfesorFK=idProfesor GROUP BY idClaseFK ORDER BY 1;";

/*>> COMPONENTES BÁSICOS <<*/

/*>> ALTA <<*/
// ETIQUETAS
JLabel lblTituloAlt= new JLabel("NUEVA CLASE");
JLabel lblHora= new JLabel("*Hora:");
JLabel lblPrecio= new JLabel("*Precio:");
JLabel lblProfe= new JLabel("*Profesor:");
JLabel lblNvl= new JLabel("*Nivel:");
JLabel lblAdv= new JLabel("--Los campos con * son obligatorios--");
// SPINNER
// DE 20 A 70, SALTO DE 5.0, EMPIEZA POR 45
SpinnerNumberModel modeloPrecio = new SpinnerNumberModel(45.0, 20.0, 70.0,
0.50);
JSpinner spnnrPrecioAlt = new JSpinner(modeloPrecio);
// DE 8 A 22, SALTO DE 1.0 EMPIEZA POR 18
SpinnerNumberModel modeloHora = new SpinnerNumberModel(18.0, 08.0, 22.0,
1.0);
JSpinner spnnrHoraAlt = new JSpinner(modeloHora);
// BOTONES
JButton btnBrrAlt = new JButton("Borrar todo");
JButton btnDarAlt = new JButton("Dar de alta");
// JCOMBOBOX
JComboBox <String> chcLstNvl = new JComboBox <String>();
JComboBox <String> chcLstProfe = new JComboBox <String>();

/*>> MODIFICACIÓN <<*/
// ETIQUETAS
JLabel lblTituloMod= new JLabel("MODIFICAR CLASE");
JLabel lblSelecMod= new JLabel("Clase seleccionada:");
JLabel lblSelecModSoc= new JLabel("Socio seleccionado:");
JLabel lblCampMod= new JLabel("Seleccione campo a modificar:");
JLabel lblClaseMod =new JLabel();
JLabel lblSocioMod =new JLabel();

```

```

// CASILLAS DE TEXTO
JTextField txtBuscMod =new JTextField(20);
JTextField txtBuscModSoc =new JTextField(20);
JTextField txtModMod =new JTextField(20);
// BOTONES
JButton btnBuscMod = new JButton("Buscar");
JButton btnModMod = new JButton("Modificar");
JButton btnBuscModSoc = new JButton("Buscar");
JButton btnReiniciarClase = new JButton("Reiniciar");
JButton btnReiniciarSocio = new JButton("Reiniciar");
// LISTA Y JCOMBOBOX
DefaultListModel<String> modeloListaMod = new DefaultListModel<String>();
JList <String> lstMod = new JList <String>(modeloListaMod);
DefaultListModel<String> modeloListaModSoc = new
DefaultListModel<String>();
JList <String> lstModSoc = new JList <String>(modeloListaModSoc);
JComboBox <String> chclstAtrib = new JComboBox <String>();
// SEPARADOR Y SCROLL
JSeparator separador = new JSeparator();
JSeparator separador2 = new JSeparator();
JScrollPane scrollMod = new JScrollPane(lstMod);
JScrollPane scrollModSoc = new JScrollPane(lstModSoc);

/*>> CONSULTA <<*/
// ETIQUETAS
JLabel lblTituloCons= new JLabel("LISTAR CLASES");
// TABLA
DefaultTableModel modeloTablaConsulta =new DefaultTableModel();
JTable tblClases = new JTable(modeloTablaConsulta);
// BOTÓN
JButton btnConsulta = new JButton("Crear PDF...");
// PANELES
JPanel pnl1 = new JPanel();
JPanel pnl2 = new JPanel();
// COLOR Y FUENTE
Color color=new Color(213, 245, 227);
Font fuenteTitulo = new Font("Calibri", 3, 20);

/*****/
/### CONSTRUCTORES ***/
/*****/

/*>> ALTA <<*/
public void nuevaClase(JFrame menu, String name){
    /*>> PARTES BÁSICAS DEL MENÚ <<*/
    this.menu=menu;
    this.name=name;
    menu.setEnabled(false);
    setTitle("Nueva clase");
    setLayout (new GridLayout(1,1));
    setSize(390, 300);
    setLocationRelativeTo(null);
    setResizable(false);
}

```

```

pn11.setLayout(null);
// COMPONENTES
lblTituloAlt.setFont(fuenteTitulo);
pn11.add(lblTituloAlt);
lblTituloAlt.setBounds(120, 5, 300, 50);
pn11.add(lblProfe);
lblProfe.setBounds(77, 60, 70, 20);
// DESPLEGABLE PROFESORES
chcLstProfe.addItem("Selecciona uno...");
objModelo.conectaBase();
objModelo.rellenarComboBoxDatos(sentenciaRellenarProfesores,
chcLstProfe);
objModelo.desconectaBase();
pn11.add(chcLstProfe);
chcLstProfe.setBounds(150, 60, 200, 25);
pn11.add(lblNvl);
lblNvl.setBounds(100, 90, 40, 20);
// DESPLEGABLE NIVEL PROFESORES
chcLstNvl.addItem("Selecciona uno...");
pn11.add(chcLstNvl);
chcLstNvl.setBounds(150, 90, 200, 25);
// SPINNERS
pn11.add(lblPrecio);
lblPrecio.setBounds(90, 120, 150, 20);
pn11.add(spnnrPrecioAlt);
spnnrPrecioAlt.setBounds(150, 120, 55, 25);
pn11.add(lblHora);
lblHora.setBounds(100, 150, 60, 20);
pn11.add(spnnrHoraAlt);
spnnrHoraAlt.setBounds(150, 150, 55, 25);
pn11.add(btnDarAlt);
btnDarAlt.setBounds(50, 185, 120, 30);
pn11.add(btnBrrAlt);
btnBrrAlt.setBounds(200, 185, 120, 30);
pn11.add(lblAdv);
lblAdv.setBounds(85, 225, 250, 30);
pn11.setBackground(color);
add(pn11);
/*>> LISTENERS Y VISIBILIDAD <<*/
btnDarAlt.addActionListener(this);
btnBrrAlt.addActionListener(this);
chcLstNvl.addActionListener(this);
chcLstProfe.addActionListener(this);
addWindowListener(this);
setVisible(true);
}

/*>> MODIFICAR <<*/
public void modificarClase(JFrame menu, String name){
    this.menu=menu;
    this.name=name;
    menu.setEnabled(false);
    setTitle("Modificar clase");
}

```



```

setLayout (new GridLayout(1,1));
setSize(790, 330);
setLocationRelativeTo(null);
setResizable(false);
pn1.setLayout(null);
// COMPONENTES
lblTituloMod.setFont(fuenteTitulo);
pn1.add(lblTituloMod);
lblTituloMod.setBounds(310, 5, 300, 50);
pn1.add(txtBuscMod);
txtBuscMod.setBounds(10,60,140,25);
pn1.add(btnBuscMod);
btnBuscMod.setBounds(155, 57, 80, 30);
// LISTA CLASE
objModelo.conectaBase();
objModelo.rellenarLista(sentenciaRellenarClase, modeloListaMod);
objModelo.desconectaBase();
pn1.add(scrollMod);
scrollMod.setBounds(10, 100, 225, 110);
pn1.add(btnReiniciarClase);
btnReiniciarClase.setBounds(70,220,100, 30);
// SEPARADOR
separador.setOrientation(SwingConstants.VERTICAL);
pn1.add(separador);
separador.setBounds(260, 60, 2, 200);
pn1.add(lblSelecMod);
lblSelecMod.setBounds(285, 50, 260, 25);
pn1.add(lblClaseMod);
lblClaseMod.setBounds(285, 80, 200, 25);
pn1.add(lblSelecModSoc);
lblSelecModSoc.setBounds(285, 110, 150, 25);
pn1.add(lblSocioMod);
lblSocioMod.setBounds(285, 140, 200, 25);
pn1.add(lblCampMod);
lblCampMod.setBounds(285, 170, 180, 25);
// OPCIONES MODIFICAR
chcLstAtrib.addItem("Selecciona uno...");
objModelo.conectaBase();
objModelo.rellenarComboBoxCampos(sentenciaRellenarCampos,
chcLstAtrib);
objModelo.desconectaBase();
chcLstAtrib.addItem("Agregar Socios");
chcLstAtrib.addItem("Eliminar Socios");
pn1.add(chcLstAtrib);
chcLstAtrib.setBounds(285, 200, 200, 25);
pn1.add(txtModMod);
txtModMod.setBounds(285, 230, 200, 25);
pn1.add(btnModMod);
btnModMod.setBounds(335, 260, 100, 30);
// SEPARADOR
separador2.setOrientation(SwingConstants.VERTICAL);
pn1.add(separador2);
separador2.setBounds(510, 60, 2, 200);

```



```

pn11.add(txtBuscModSoc);
txtBuscModSoc.setBounds(540,60,140,25);
pn11.add(btnBuscModSoc);
btnBuscModSoc.setBounds(685, 57, 80, 30);
// LISTA SOCIOS
pn11.add(scrollModSoc);
scrollModSoc.setBounds(540, 100, 225, 110);
pn11.add(btnReiniciarSocio);
btnReiniciarSocio.setBounds(610,220,100, 30);
pn11.setBackground(color);
add(pn11);
/*>> LISTENERS Y VISIBILIDAD <<*/
btnBuscModSoc.addActionListener(this);
btnBuscMod.addActionListener(this);
btnModMod.addActionListener(this);
lstMod.addMouseListener(this);
lstModSoc.addMouseListener(this);
chcLstAtrib.addMouseListener(this);
chcLstAtrib.addActionListener(this);
btnReiniciarClase.addActionListener(this);
btnReiniciarSocio.addActionListener(this);
addWindowListener(this);
setVisible(true);
}

/*>> LISTAR <<*/
public void consultarClase(JFrame menu, String name){
    this.menu=menu;
    this.name=name;
    menu.setEnabled(false);
    setTitle("Listado clases");
    setLayout (new GridLayout(1,1));
    setSize(570, 290);
    setLocationRelativeTo(null);
    setResizable(false);
    pn11.setLayout(null);
    pn12.setLayout(new BorderLayout());
    // COMPONENTES
    lblTituloCons.setFont(fuenteTitulo);
    pn11.add(lblTituloCons);
    lblTituloCons.setBounds(220, 10, 300, 50);
    // TABLA
    objModelo.conectaBase();
    objModelo.rellenarTabla(sentenciaRellenarTabla, tblClases);
    objModelo.desconectaBase();
    pn12.add(new JScrollPane(tblClases), BorderLayout.CENTER);
    pn11.add(pn12);
    pn12.setBounds(18,70 , 530, 170);
    pn12.add(btnConsulta, BorderLayout.SOUTH);
    pn11.setBackground(color);
    add(pn11);
}

```

```

/*>> LISTENERS Y VISIBILIDAD <<*/
btnConsulta.addActionListener(this);
addWindowListener(this);
setVisible(true);
}

@Override
public void mouseClicked(MouseEvent arg0) {

    /*>> MODIFICACIÓN <<*/
    if(arg0.getSource().equals(lstMod)) {
        modeloListaModSoc.removeAllElements();
        lblClaseMod.setText(lstMod.getSelectedValue());
        chcLstAtrib.setSelectedIndex(0);
        objModelo.borrarTxt(txtModMod);
    }
    if(arg0.getSource().equals(lstModSoc)) {
        lblSocioMod.setText(lstModSoc.getSelectedValue());
        objModelo.borrarTxt(txtModMod);
    }
}

@Override
public void mouseEntered(MouseEvent arg0) {}
@Override
public void mouseExited(MouseEvent arg0) {}
@Override
public void mousePressed(MouseEvent arg0) {}
@Override
public void mouseReleased(MouseEvent arg0) {}

@Override
public void actionPerformed(ActionEvent e) {

    /*>> ALTA <<*/
    // BOTÓN BORRAR
    if(btnBrrAlt.equals(e.getSource())) {
        chcLstProfe.setSelectedIndex(0);
        chcLstNvl.setSelectedIndex(0);
        spnnrHoraAlt.setValue(18);
        spnnrPrecioAlt.setValue(45);
    }
    // LISTA PROFESOR
    if(chcLstProfe.equals(e.getSource())) {
        if(chcLstProfe.getSelectedIndex()!=0) {
            chcLstNvl.setSelectedIndex(0);
            String [] listaDato = ((String)
chcLstProfe.getSelectedItem()).split(" ");
            String sentenciaNivelProfesor ="SELECT nivelProfesor FROM
profesores WHERE idProfesor='"+listaDato[1]+"';";
            objModelo.conectaBase();
            objModelo.rellenarComboBoxNivel(sentenciaNivelProfesor,
chcLstNvl);

```

```

        objModelo.desconectaBase();
    }
}
// DAR DE ALTA
if(btnDarAlt.equals(e.getSource())) {

    if((((""+spnnrHoraAlt.getValue()).equals(""))|(((""+spnnrPrecioAlt.getValue(
)).equals(""))|(chcLstProfe.getSelectedIndex()==0)|(chcLstNvl.getSelectedIndex(
)==0)) {

        JOptionPane.showMessageDialog(this,"Uno o varios de los
campos obligatorios NO han sido rellenados ","Mensaje de
ERROR",JOptionPane.ERROR_MESSAGE);
    }else {
        String [] listaDato = ((String)
chcLstProfe.getSelectedItem()).split(" ");
        String sentenciaAlta ="INSERT INTO clases (nivelClase,
horaClase, precioClase, idProfesorFK) VALUES
('"+chcLstNvl.getSelectedItem()+"', '"+spnnrHoraAlt.getValue()+"', '"+spnnrPrecio
Alt.getValue()+"', '"+listaDato[1]+"')";
        objModelo.conectaBase();
        objModelo.datosABM(sentenciaAlta);
        objModelo.desconectaBase();
        String ficheroMovimiento =
 "["+name+"]["+sentenciaAlta+"]\n";
        objModelo.ficheroMovimiento(ficheroMovimiento);
        JOptionPane.showMessageDialog (this,"La clase ha sido dado
de alta correctamente","Confirmación",JOptionPane.INFORMATION_MESSAGE);

        menu.setEnabled(true);
        dispose();
    }
}

/*>> MODIFICACIÓN <<*/
// BUSCAR CLASE
if(btnBuscMod.equals(e.getSource())){
    modeloListaMod.removeAllElements();
    //Rellenar lista segun txt
    String sentenciaBuscarMod ="SELECT idClase, horaClase,
concat('(',nivelClase,')'), concat(nombreProfesor,' ', apellido1Profesor) FROM
clases, profesores WHERE concat(idClase, horaClase, concat('(',nivelClase,')'),
concat(nombreProfesor,' ', apellido1Profesor)) LIKE
'%" +txtBuscMod.getText()+"%" AND clases.idProfesorFK=profesores.idProfesor
ORDER BY 2;";

    objModelo.conectaBase();
    objModelo.buscarDatos(sentenciaBuscarMod);
    objModelo.rellenarLista(sentenciaBuscarMod, modeloListaMod);
    objModelo.desconectaBase();
}

```

```

// BUSCAR SOCIO
if(btnBuscModSoc.equals(e.getSource())){
    if(!lstMod.isSelectionEmpty() &&
chcLstAtrib.getSelectedIndex()==6) {
        modeloListaModSoc.removeAllElements();
        //Rellenar lista segun txt
        String sentenciaBuscarModSoc = "SELECT idSocio,
nombreSocio, apellido1Socio, apellido2Socio FROM socios WHERE NOT(idSocio IN
(SELECT idSocioFK FROM solicitudes WHERE
idClaseFK="+lblClaseMod.getText().split(" ")[1]+")) AND
concat(idSocio,nombreSocio,apellido1Socio,apellido2Socio) LIKE
'%" +txtBuscModSoc.getText()+"%' ORDER BY 2;";
        objModelo.conectaBase();
        objModelo.buscarDatos(sentenciaBuscarModSoc);
        objModelo.rellenarLista(sentenciaBuscarModSoc,
modeloListaModSoc);
        objModelo.desconectaBase();
    }else if(chcLstAtrib.getSelectedIndex()==7 &&
!lstMod.isSelectionEmpty()) {
        String sentenciaRellenarSocios = "SELECT idSocio,
nombreSocio, apellido1Socio, apellido2Socio FROM socios WHERE idSocio IN
(SELECT idSocioFK FROM solicitudes WHERE
idClaseFK="+lblClaseMod.getText().split(" ")[1]+")) AND
concat(idSocio,nombreSocio,apellido1Socio,apellido2Socio) LIKE
'%" +txtBuscModSoc.getText()+"%' ORDER BY 2;";
        modeloListaModSoc.removeAllElements();
        objModelo.conectaBase();
        objModelo.buscarDatos(sentenciaRellenarSocios);
        objModelo.rellenarLista(sentenciaRellenarSocios,
modeloListaModSoc);
        objModelo.desconectaBase();
    }
}
// REINICIAR CLASE
if(btnReiniciarClase.equals(e.getSource())){
    modeloListaMod.removeAllElements();
    objModelo.conectaBase();
    objModelo.buscarDatos(sentenciaRellenarClase);
    objModelo.rellenarLista(sentenciaRellenarClase,
modeloListaMod);
    objModelo.desconectaBase();
}
// REINICIAR SOCIO
if(btnReiniciarSocio.equals(e.getSource())){
    if(!lstMod.isSelectionEmpty() &&
chcLstAtrib.getSelectedIndex()==6) {
        String sentenciaRellenarSocios = "SELECT idSocio,
nombreSocio, apellido1Socio, apellido2Socio FROM socios WHERE NOT(idSocio IN
(SELECT idSocioFK FROM solicitudes WHERE
idClaseFK="+lblClaseMod.getText().split(" ")[1]+")) ORDER BY 2;";
        modeloListaModSoc.removeAllElements();
        objModelo.conectaBase();
        objModelo.buscarDatos(sentenciaRellenarSocios);
    }
}

```

```

        objModelo.rellenarLista(sentenciaRellenarSocios,
modeloListaModSoc);
        objModelo.desconectaBase();
    }else if(chcLstAtrib.getSelectedIndex()==7 &&
!lstMod.isSelectionEmpty()) {
        String sentenciaRellenarSocios = "SELECT idSocio,
nombreSocio, apellido1Socio, apellido2Socio FROM socios WHERE idSocio IN
(SELECT idSocioFK FROM solicitudes WHERE
idClaseFK="+lblClaseMod.getText().split(" ")[1]+"") ORDER BY 2;";
        modeloListaModSoc.removeAllElements();
        objModelo.conectaBase();
        objModelo.buscarDatos(sentenciaRellenarSocios);
        objModelo.rellenarLista(sentenciaRellenarSocios,
modeloListaModSoc);
        objModelo.desconectaBase();
    }
}
// OPCIONES ATRIBUTOS
if(chcLstAtrib.equals(e.getSource())) {
    if(chcLstAtrib.getSelectedIndex()>0 &&
chcLstAtrib.getSelectedIndex()<3 && !lstMod.isSelectionEmpty()) {
        try {
            lblSocioMod.setText("");
            modeloListaModSoc.removeAllElements();
            indice = chcLstAtrib.getSelectedIndex();
            //Separador de texto
            String [] listaDato = lblClaseMod.getText().split("
");
            String sentenciaBuscarClaseMod = "SELECT * FROM
clases WHERE idClase='"+listaDato[1]+'";";
            objModelo.conectaBase();

            resultado=objModelo.buscarDatos(sentenciaBuscarClaseMod);
            resultado.next();
            txtModMod.setText(resultado.getString(indice+2));
            objModelo.desconectaBase();
        } catch (SQLException e1) {
            e1.printStackTrace();
        }
    }else {
        objModelo.borrarTxt(txtModMod);
        if(chcLstAtrib.getSelectedIndex()==3 &&
!lstMod.isSelectionEmpty()) {
            String sentenciaRellenarSocios = "SELECT idSocio,
nombreSocio, apellido1Socio, apellido2Socio FROM socios WHERE NOT(idSocio IN
(SELECT idSocioFK FROM solicitudes WHERE
idClaseFK="+lblClaseMod.getText().split(" ")[1]+")) ORDER BY 2;";
            modeloListaModSoc.removeAllElements();
            objModelo.conectaBase();
            objModelo.buscarDatos(sentenciaRellenarSocios);
            objModelo.rellenarLista(sentenciaRellenarSocios,
modeloListaModSoc);
            objModelo.desconectaBase();
        }
    }
}

```

```

        }else if(chcLstAtrib.getSelectedIndex()==4 &&
!lstMod.isSelectionEmpty()) {
            String sentenciaRellenarSocios = "SELECT idSocio,
nombreSocio, apellido1Socio, apellido2Socio FROM socios WHERE idSocio IN
(SELECT idSocioFK FROM solicitudes WHERE
idClaseFK='"+lblClaseMod.getText().split(" ")[1]+"') ORDER BY 2;";
            modeloListaModSoc.removeAllElements();
            objModelo.conectaBase();
            objModelo.buscarDatos(sentenciaRellenarSocios);
            objModelo.rellenarLista(sentenciaRellenarSocios,
modeloListaModSoc);
            objModelo.desconectaBase();
        }else{
            lblSocioMod.setText(lstModSoc.getSelectedValue());
            modeloListaModSoc.removeAllElements();
        }
    }
}
// BOTÓN MODIFICAR
if(btnModMod.equals(e.getSource())){

    if(lblClaseMod.getText().equals("")|chcLstAtrib.getSelectedIndex()==0|chcL
stAtrib.getSelectedIndex()>2 && lblSocioMod.getText().equals("")) {
        JOptionPane.showMessageDialog(this,"La clase
"+lblClaseMod.getText()+" NO ha sido modificado correctamente","Mensaje de
ERROR",JOptionPane.ERROR_MESSAGE);
    }else if(chcLstAtrib.getSelectedIndex()==3){
        String sentenciaAltaSolicitud ="INSERT INTO solicitudes
(idClaseFK, idSocioFK) VALUES ('"+lblClaseMod.getText().split("
")[1]+"','"+lblSocioMod.getText().split(" ")[1]+"')";
        objModelo.conectaBase();
        objModelo.datosABM(sentenciaAltaSolicitud);
        objModelo.desconectaBase();
        String ficheroMovimiento =
 "["+name+"]["+sentenciaAltaSolicitud+"]\n";
        objModelo.ficheroMovimiento(ficheroMovimiento);
        JOptionPane.showMessageDialog (this,"El socio ha sido
añadido a la clase
correctamente","Confirmación",JOptionPane.INFORMATION_MESSAGE);
        menu.setEnabled(true);
        dispose();
    }else if(chcLstAtrib.getSelectedIndex()==4){
        String sentenciaBajaSolicitud ="DELETE FROM solicitudes
WHERE idSocioFK='"+lblSocioMod.getText().split(" ")[1]+"'' AND
idClaseFK='"+lblClaseMod.getText().split(" ")[1]+"'";
        objModelo.conectaBase();
        objModelo.datosABM(sentenciaBajaSolicitud);
        objModelo.desconectaBase();
        String ficheroMovimiento =
 "["+name+"]["+sentenciaBajaSolicitud+"]\n";
        objModelo.ficheroMovimiento(ficheroMovimiento);
    }
}

```



```

JOptionPane.showMessageDialog(this, "El socio ha sido
eliminado de la clase
correctamente", "Confirmación", JOptionPane.INFORMATION_MESSAGE);
menu.setEnabled(true);
dispose();
} else {
String [] listaDato = lblClaseMod.getText().split(" ");
indice = chclstAtrib.getSelectedIndex()+2;
String sentenciaCampo = "SELECT * FROM clases;";
objModelo.conectaBase();
resultado= objModelo.buscarDatos(sentenciaCampo);
try {
    campoMod =
resultado.getMetaData().getColumnName(indice);
    String sentenciaModificar = "UPDATE clases SET
"+campoMod+"='"+txtModMod.getText()+"' WHERE idClase="+listaDato[1]+"";
    objModelo.datosABM(sentenciaModificar);
    objModelo.desconectaBase();
    String ficheroMovimiento =
    "["+name+"]["+sentenciaModificar+"]\n";
    objModelo.ficheroMovimiento(ficheroMovimiento);
    JOptionPane.showMessageDialog (this, "La clase ha
sido modificada correctamente", "Confirmación", JOptionPane.INFORMATION_MESSAGE);
    menu.setEnabled(true);
    dispose();
} catch (SQLException e1) {
    JOptionPane.showMessageDialog(this, "La clase
"+lblClaseMod.getText()+" NO ha sido modificado correctamente", "Mensaje de
ERROR", JOptionPane.ERROR_MESSAGE);
}
}
}

/*>> CONSULTA <<*/
// BOTÓN CREAR PDF
if(btnConsulta.equals(e.getSource())) {
    try {
        String ficheroMovimiento = "["+name+"] [PDF Listado
Clases]\n";

        objModelo.ficheroMovimiento(ficheroMovimiento);
        new ImprimirPDF(this, "Clases", sentenciaRellenarTabla);
    } catch (DocumentException | IOException |
ClassNotFoundException e1) {
        // TODO Auto-generated catch block
        e1.printStackTrace();
    }
}
}
}

```

```
@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}

@Override
public void windowClosing(WindowEvent arg0) {
    menu.setEnabled(true);
}

@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}

}
```