
Manual Técnico

DESCRIPCION

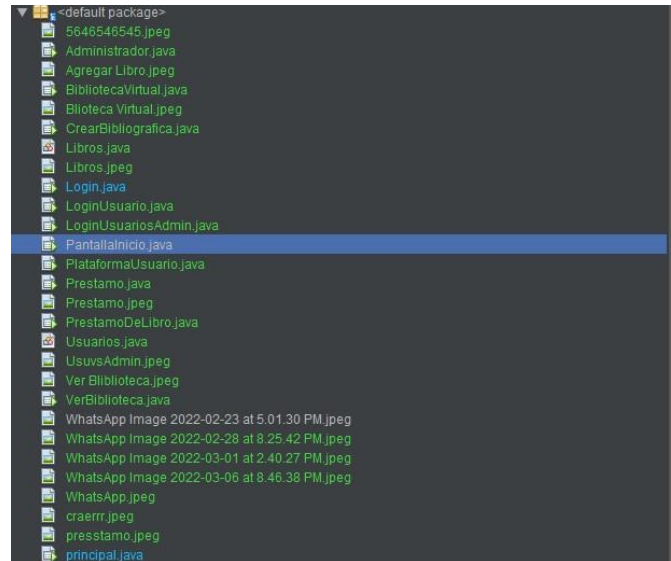
MANUAL GENERAL

PROYECTO GENERAL

El proyecto general esta echo en base de jframe para el uso

Del panel de visualización, teniendo en cuenta que cada apartado tiene su propio java class para almacenar datos

Siendo, el jframe principal.



```
1 public class Usuarios {
2     private String ID;
3     private String Nombre;
4     private String Apellido;
5     private String User;
6     private String Rol;
7     private String Clave;
8
9
10    public Usuarios(String ID, String Nombre, String Apellido, String User, String Rol, String Clave) {
11        this.ID = ID;
12        this.Nombre = Nombre;
13        this.Apellido = Apellido;
14        this.User = User;
15        this.Rol = Rol;
16        this.Clave = Clave;
17    }
18
19    public String getID() {
20        return ID;
21    }
22
23    public void setID(String ID) {
24        this.ID = ID;
25    }
26
27    public String getNombre() {
28        return Nombre;
29    }
30
31    public void setNombre(String Nombre) {
32        this.Nombre = Nombre;
33    }
34
35    public String getApellido() {
36        return Apellido;
37    }
38
39    public void setApellido(String Apellido) {
40        this.Apellido = Apellido;
41    }
42 }
```

JAVA CLASS

Utilizando Varias Java Class para poder almacenar varios datos y hacer poder llamarlos en otros jframes para su uso como por ejemplo el java class de Usuarios.

JAVA CLASS DE BIBLIOTECA

Al igual que los Usuarios la biblioteca tiene su propio Java Class en el cual se almacenan los datos de cada uno de los libros registrados.

```
1
2 public class Libros {
3
4     public String Autor;
5     public String Fecha;
6     public String Titulo;
7     public String Descripcion;
8     public String Palabras;
9     public String Edicion;
10    public String Temas;
11    public String Copias;
12    public String Area;
13    public String Categoria;
14    public String isbn;
15
16    public Libros(String Autor, String Fecha, String Titulo, String Descripcion, String Palabras, String Edicion, String Temas,
17                  this.Autor = Autor;
18                  this.Fecha = Fecha;
19                  this.Titulo = Titulo;
20                  this.Descripcion = Descripcion;
21                  this.Palabras = Palabras;
22                  this.Edicion = Edicion;
23                  this.Temas = Temas;
24                  this.Copias = Copias;
25                  this.Area = Area;
26                  this.Categoria = Categoria;
27                  this.isbn = isbn;
28    }
29
30    public String getAutor() {
31        return Autor;
32    }
33
34    public void setAutor(String Autor) {
35        this.Autor = Autor;
36    }
37
38    public String getFecha() {
39        return Fecha;
40    }
41
42    public void setFecha(String Fecha) {
43        this.Fecha = Fecha;
44    }
45
46    public String getTitulo() {
47        return Titulo;
48    }
49
50    public void setTitulo(String Titulo) {
51        this.Titulo = Titulo;
52    }
53
54    public String getDescripcion() {
55        return Descripcion;
56    }
57
58    public void setDescripcion(String Descripcion) {
59        this.Descripcion = Descripcion;
60    }
61
62    public String getPalabras() {
63        return Palabras;
64    }
65
66    public void setPalabras(String Palabras) {
67        this.Palabras = Palabras;
68    }
69
70    public String getEdicion() {
71        return Edicion;
72    }
73
74    public void setEdicion(String Edicion) {
75        this.Edicion = Edicion;
76    }
77
78    public String getTemas() {
79        return Temas;
80    }
81
82    public void setTemas(String Temas) {
83        this.Temas = Temas;
84    }
85
86    public String getCopias() {
87        return Copias;
88    }
89
90    public void setCopias(String Copias) {
91        this.Copias = Copias;
92    }
93
94    public String getArea() {
95        return Area;
96    }
97
98    public void setArea(String Area) {
99        this.Area = Area;
100    }
101
102    public String getCategoria() {
103        return Categoria;
104    }
105
106    public void setCategoria(String Categoria) {
107        this.Categoria = Categoria;
108    }
109
110    public String getIsbn() {
111        return isbn;
112    }
113
114    public void setIsbn(String isbn) {
115        this.isbn = isbn;
116    }
117
118 }
```

```
1
2 public class Usuarios {
3     private String ID;
4     private String Nombre;
5     private String Apellido;
6     private String User;
7     private String Rol;
8     private String Clave;
9
10    public Usuarios(String ID, String Nombre, String Apellido, String User, String Rol, String Clave) {
11        this.ID = ID;
12        this.Nombre = Nombre;
13        this.Apellido = Apellido;
14        this.User = User;
15        this.Rol = Rol;
16        this.Clave = Clave;
17    }
18
19    public String getID() {
20        return ID;
21    }
22
23    public void setID(String ID) {
24        this.ID = ID;
25    }
26
27    public String getNombre() {
28        return Nombre;
29    }
30
31    public void setNombre(String Nombre) {
32        this.Nombre = Nombre;
33    }
34
35    public String getApellido() {
36        return Apellido;
37    }
38
39    public void setApellido(String Apellido) {
40        this.Apellido = Apellido;
41    }
42
43    public String getUser() {
44        return User;
45    }
46
47    public void setUser(String User) {
48        this.User = User;
49    }
50
51    public String getRol() {
52        return Rol;
53    }
54
55    public void setRol(String Rol) {
56        this.Rol = Rol;
57    }
58
59    public String getClave() {
60        return Clave;
61    }
62
63    public void setClave(String Clave) {
64        this.Clave = Clave;
65    }
66
67 }
```

INFORMACION DEL JAVA CLASS

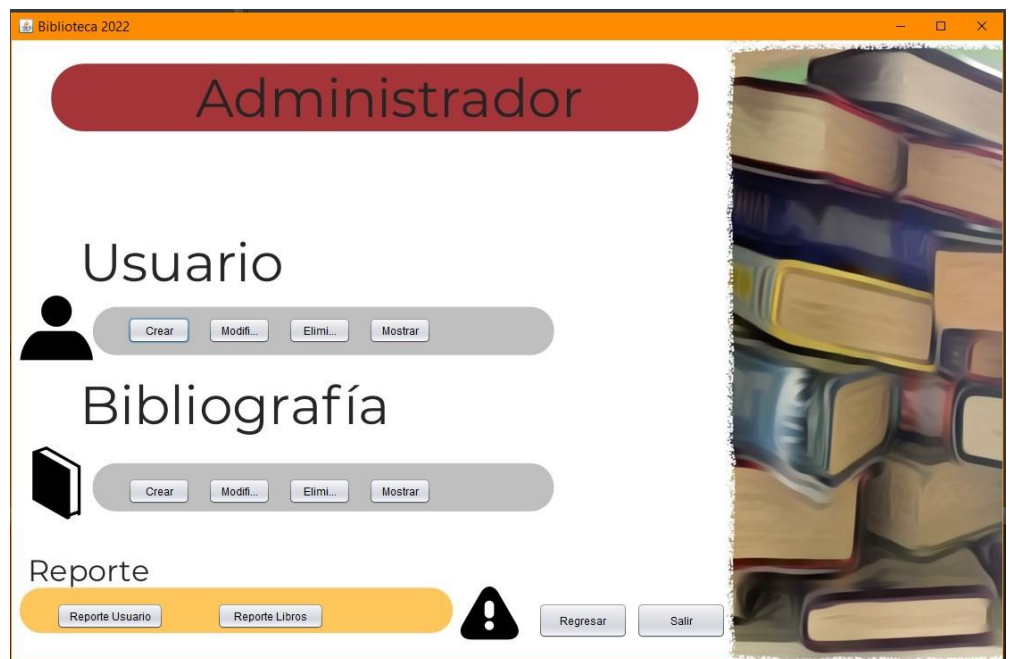
Cada java class tiene su propios String como Int para la declaración de variables utilizando el constructor de códigos podemos estructurar rápidamente una java Class para almacenamientos de datos.

jLabel

Los JLabel los utilizamos para agregar las imágenes que tanto resaltan en cada uno de los JFrame para poder dar una interfaz amigable y completa a cada Usuario con la

Intención que sea agradable al Ojo humano.

- Inicio
- Login
- Administrador
- Usuario
- Creación de Usuarios
- Creación de Libros
- Biblioteca virtual
- Ver Bibliografía



CREACION DE USUARIOS

La creacion de usuarios fue creada por medio de java class y por medio de for para implementar un llamado de la java class de usuarios en la ventana de login.en el cual se le solicita al aministrador los siguientes datos a nuestro usuario.

- ID: DPI
- Nombre: El nombre del usuario
- Apellido: El apellido del usuario
- User: Nickname para acceder más rápido al sistema.
- Rol: “estudiante” o “catedratico”.
- Password: Contraseña del usuario.

```
private void guardarActionPerformed(java.awt.event.ActionEvent evt) {  
    agregar();  
    if (!hayespacio(listaUsuario)) {  
        System.out.println("Dimension:" + getListaUsuario().length);  
        listaUsuario = redimensionalArreglo(getListaUsuario());  
        System.out.println("Logitud:" + getListaUsuario().length);  
    }  
    for (int i = 0; i < getListaUsuario().length; i++) {  
        if (getListaUsuario()[i] == null) {  
            listaUsuario[i] = new Usuarios(IDUsu.getText(), NombreUsu.getText(), ApellidoUsu.getText(),  
            break;  
        }  
    }  
}  
  
private void refrescarActionPerformed(java.awt.event.ActionEvent evt) {  
    for (int i = 0; i < getListaUsuario().length; i++) {  
        if (getListaUsuario()[i] != null) {  
            System.out.println("*****");  
            System.out.println(getListaUsuario()[i].getID());  
            System.out.println(getListaUsuario()[i].getNombre());  
            System.out.println(getListaUsuario()[i].getApellido());  
            System.out.println(getListaUsuario()[i].getUser());  
            System.out.println(getListaUsuario()[i].getRol());  
            System.out.println(getListaUsuario()[i].getClave());  
            System.out.println("*****");  
        }  
    }  
    String[][] matriz = new String[getListaUsuario().length][6];  
    for (int i = 0; i < getListaUsuario().length; i++) {  
        if (getListaUsuario()[i] != null) {  
            matriz[i][0] = getListaUsuario()[i].getID();  
            matriz[i][1] = getListaUsuario()[i].getNombre();  
            matriz[i][2] = getListaUsuario()[i].getApellido();  
            matriz[i][3] = getListaUsuario()[i].getUser();  
            matriz[i][4] = getListaUsuario()[i].getRol();  
            matriz[i][5] = getListaUsuario()[i].getClave();  
        }  
    }  
}
```

Cada Usuario se puede editar desde la interfaz Grafica en la cual es bastante agraciada con el administrador

Biblioteca 2022

Crear Usuario

ID:

Nombre:

Apellido:

User:

Rol:

Contraseña:

C. Contraseña:

ID	Nombres	Apellidos	User	Rol	Clave

Regresar Enter Usuario Guardar Actualizar Eliminar Limpiar

OPCIONES DEL ADMINISTRADOR

VER USUARIO

Las opciones del administrador fueron creadas por medio de `principal prin = new principal ();`

```
26 //
27 @SuppressWarnings("unchecked")
28 Generated Code
29
169
170 private void ReporteBibliActionPerformed(java.awt.event.ActionEvent evt) {
171     // TODO add your handling code here:
172 }
173
174 private void MostrarBibliActionPerformed(java.awt.event.ActionEvent evt) {
175     Login prin = new Login ();
176     prin.setVisible(true);
177     dispose();
178 }
179
180 private void CrearUsuActionPerformed(java.awt.event.ActionEvent evt) {
181     principal prin = new principal ();
182     prin.setVisible(true);
183     dispose();
184 }
185
186 private void CrearUsuMouseClicked(java.awt.event.MouseEvent evt) {
187     // TODO add your handling code here:
188 }
189
190 private void MostrarBibli2ActionPerformed(java.awt.event.ActionEvent evt) {
191     LoginUsuariosAdmin prin = new LoginUsuariosAdmin ();
192     prin.setVisible(true);
193     dispose();
194 }
195
196 private void MostrarUsuActionPerformed(java.awt.event.ActionEvent evt) {
197     principal prin = new principal ();
198     prin.setVisible(true);
199     dispose();
200 }
201
202 private void CrearBibliActionPerformed(java.awt.event.ActionEvent evt) {
203     CrearBibliografica prin = new CrearBibliografica ();
204     prin.setVisible(true);
205     dispose();
206 }
207
```

Varios opciones

El administrador ya tiene su propio usuario para indicar los movimientos de cada una de las plataformas de la plataforma de Usuario al igual podemos concluir que el administrador tiene la opción de indicar y modificar cada opción.

```
private void EliminarUsuActionPerformed(java.awt.event.ActionEvent evt) {
    principal prin = new principal ();
    prin.setVisible(true);
    dispose();
}

private void ModificarBibliActionPerformed(java.awt.event.ActionEvent evt) {
    CrearBibliografica prin = new CrearBibliografica ();
    prin.setVisible(true);
    dispose();
}

private void EliminarBibliActionPerformed(java.awt.event.ActionEvent evt) {
    CrearBibliografica prin = new CrearBibliografica ();
    prin.setVisible(true);
    dispose();
}

private void MostrarBibli1ActionPerformed(java.awt.event.ActionEvent evt) {
    CrearBibliografica prin = new CrearBibliografica ();
    prin.setVisible(true);
    dispose();
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    LookAndFeelSettingCode(Optional);

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new Administrador().setVisible(true);
        }
    });
}
```

TABLAS DE DATOS

Algo fundamental de este proyecto fueron el manejo de tablas para poder administrar los datos dentro de cada JFrame tomando en cuenta que esta, está enlazada previamente dicha con el Java class para el almacenamiento de datos.

```
public PrestamoDeLibro() {
    initComponents();
    initComponents2();
    String[] titulo=new String[]{"No", "Titulo", "Autor","Agregar"};
    dtm.setColumnIdentifiers (titulo);
    TablaDatos.setModel (dtm);
}

private void initComponents2(){
    this.setTitle("Biblioteca 2022");
    setLocationRelativeTo(null);
}

void agregar (){
    dtm.addRow(new Object[]{
        L1.getText(),L2.getText(),L3.getText()});}

void agregar2 (){
    dtm.addRow(new Object[]{
        R1.getText(),R2.getText(),R3.getText()});}

void agregar3 (){
    dtm.addRow(new Object[]{
        T1.getText(),T2.getText(),T3.getText()});}
```


UTILIZACION DE TABLAS

Las tablas fueron creada desde el public class para tenerlo en el proyecto general y no tener algún problema en poder llamarlos utilizando el método Void y así poder implementar en cada botón, sus diferentes opciones como eliminar modificar y agregar como limpiar la tabla.

TABLAS

Las tablas son una herramienta muy útil para implementar información de forma ordena y simple con la cual hay varias maneras de poder modificar.

[illegible]

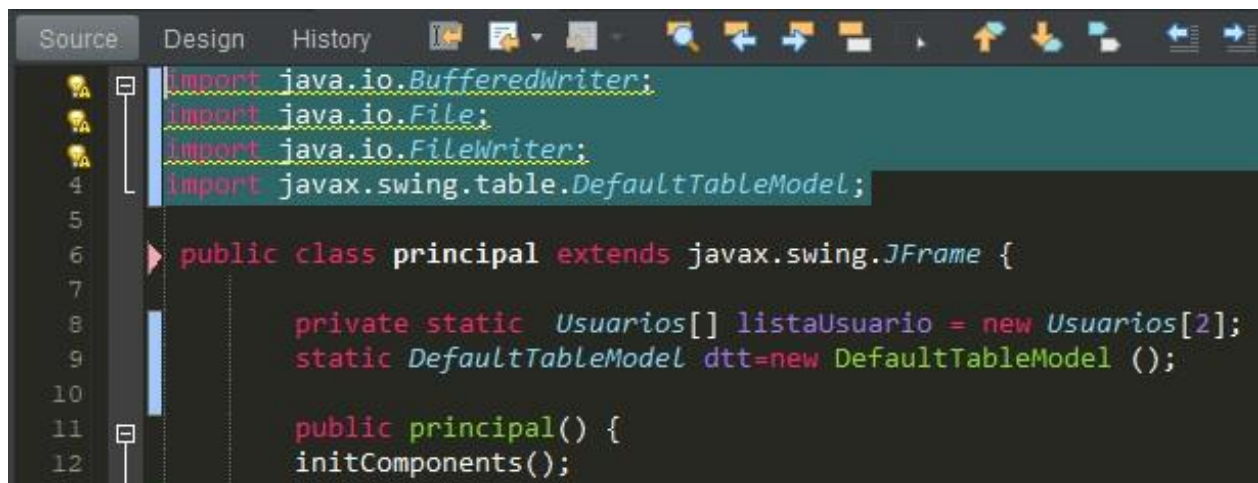
LIBRERIAS UTILIZADAS

BufferedWriter;

File;

.FileWriter;

DefaultTableModel;



```
Source  Design  History  [Icons]
1  import java.io.BufferedWriter;
2  import java.io.File;
3  import java.io.FileWriter;
4  import javax.swing.table.DefaultTableModel;
5
6  public class principal extends javax.swing.JFrame {
7
8      private static Usuarios[] listaUsuario = new Usuarios[2];
9      static DefaultTableModel dtt=new DefaultTableModel ();
10
11      public principal() {
12          initComponents();
```

