

MANUAL TÉCNICO

Biblioteca Facultad de Ingeniería USAC

Por: Mario Ernesto Marroquín Pérez

Carné: 202110509

Observación: El software se encuentra escrito en Java.

```
Proyecto1.java 1 X
src > proyecto > pkg1 > Proyecto1.java > ...
1
2 package proyecto.pkg1;
3
4 /**
5  *
6  * @author marroquin
7  */
8 public class Proyecto1 {
9
10     /**
11      * @param args the command line arguments
12      */
13     public static void main(String[] args) {
14         MiInicio frame = new MiInicio();
15     }
16
17 }
18
```

Proyecto 1.java es nuestra clase principal en donde indicamos que se ejecutará primero, en este caso será la clase “MiInicio”, la cual nos muestra la visión de la biblioteca, así como nos da la opción de ingresar a la misma.

```
MiInicio.java 1 X
src > proyecto > pkg1 > MiInicio.java > {} proyecto.pkg1
1 package proyecto.pkg1;
2 import java.awt.*;
3 import java.awt.event.ActionEvent;
4 import java.awt.event.ActionListener;
5 import javax.swing.*;
6 import java.awt.event.ActionListener;
7
8 public class MiInicio extends JFrame implements ActionListener {
9
10     JLabel label, label1, label2, label3, label4, label5; //declaramos variables
11     JButton botonLogin, botonAbout;
12
13     MiInicio() {
14
15         //añadimos las respectivas imagenes
16         ImageIcon imgusu = new ImageIcon(getClass().getResource("imgs/usuario.png")).getImage().getScaledInstance(70, 70, Image.SCALE_SMOOTH);
17         label = new JLabel(); //creamos un nuevo objeto tipo label
18         label.setIcon(imgusu); //añadimos la imagen al objeto label
19         label.setVerticalAlignment(JLabel.TOP); //posicion vertical del label
20         label.setHorizontalAlignment(JLabel.LEFT); //posicion horizontal del label
21
22         ImageIcon logo = new ImageIcon(new ImageIcon(getClass().getResource("imgs/logo.png")).getImage().getScaledInstance(500, 225, Image.SCALE_SMOOTH));
23         label1 = new JLabel();
24         label1.setIcon(logo);
25         label1.setVerticalAlignment(JLabel.CENTER);
26         label1.setHorizontalAlignment(JLabel.CENTER);
27
28         //añadimos el texto
29         label2 = new JLabel();
30         label2.setText("<html><body>(b)VISIÓN</b></body></html>"); //Agregamos el texto
31         label2.setVerticalAlignment(JLabel.CENTER); //posicion vertical del label
32         label2.setHorizontalAlignment(JLabel.CENTER); //posicion horizontal del label
33     }
34 }
35
```



Al presionar iniciar sesión, vamos hacia la clase “MyRegistro” la cual valida nuestros datos para otorgarnos acceso a la biblioteca ya seamos un usuario normal o el administrador.

```

MyRegistro.java X
src > proyecto > pkg1 > MyRegistro.java > {} proyecto.pkg1
1 package proyecto.pkg1;
2
3 import java.awt.*;
4 import java.awt.event.ActionEvent;
5 import java.awt.event.ActionListener;
6 import javax.swing.*;
7
8 public class MyRegistro extends JFrame implements ActionListener {
9     static int num;
10    static String UsuarioI;
11    JLabel labelusu, labelpass, labelimg; //declaramos variables
12    JButton botonacp, botoncancel;
13    JTextField textusu, textpass;
14    JPasswordField contra;
15
16
17
18
19    MyRegistro(){
20
21        labelusu = new JLabel();
22        labelusu.setText("Usuario:");
23        labelusu.setVerticalAlignment(JLabel.CENTER); //posicion vertical del label
24        labelusu.setHorizontalAlignment(JLabel.LEFT);
25
26        labelpass = new JLabel();
27        labelpass.setText("Contraseña:");
28        labelpass.setVerticalAlignment(JLabel.CENTER); //posicion vertical del label
29        labelpass.setHorizontalAlignment(JLabel.LEFT);
30
31        ImageIcon imgusu = new ImageIcon(new ImageIcon(getClass().getResource("imgs/usuario.png")).getImage().getScaledInstance(70, 70, Image.SCALE_SMOOTH));
32        labelimg = new JLabel(); //creamos un nuevo objeto tipo label
33        labelimg.setIcon(imgusu); //asignamos el icono al objeto label
34    }
35
36    public void actionPerformed(ActionEvent e) {
37        // TODO Auto-generated method stub
38    }
39
40 }

```



Si somos administrador seremos enviados a la clase MenuAdmin, en donde se crean, modifican, eliminan y muestran los usuarios y bibliografías así como los préstamos.

```

MenuAdmin.java X
src > proyecto > pkg1 > MenuAdmin.java > {} MenuAdmin > actionPerformed(ActionEvent)
1 package proyecto.pkg1;
2 import java.awt.*;
3 import java.awt.event.ActionEvent;
4 import java.awt.event.ActionListener;
5 import javax.swing.*;
6 import javax.swing.event.*;
7
8
9
10 public class MenuAdmin extends JFrame implements ActionListener {
11     public String arg_usuario [] = new String [100];
12     public String arg_bibliografia [] = new String [100];
13
14     JLabel labellogo, labelusuA, labelReportes, labelBibliografia, labelUsuariosI;
15     JButton botonlogout, botonusuario, botonBibliografia, botonPrestamos, botonCrear, botonVer, botonModificar;
16     JButton botoneliminar, botonCrearI, botonVerI, botonModificarI, botoneliminarI;
17
18
19    MenuAdmin(){
20
21
22
23        ImageIcon logo = new ImageIcon(new ImageIcon(getClass().getResource("imgs/logoInge.png")).getImage().getScaledInstance(450, 100, Image.SCALE_SMOOTH));
24        labellogo = new JLabel();
25        labellogo.setIcon(logo);
26        labellogo.setVerticalAlignment(JLabel.CENTER);
27        labellogo.setHorizontalAlignment(JLabel.CENTER);
28
29        labelReportes = new JLabel();
30        labelReportes.setText("Reportes");
31        labelReportes.setVerticalAlignment(JLabel.CENTER);
32        labelReportes.setHorizontalAlignment(JLabel.LEFT);
33    }
34
35    public void actionPerformed(ActionEvent e) {
36        // TODO Auto-generated method stub
37    }
38
39 }

```



Si somos un usuario normal seremos enviados a la clase “MenuUsuario” en donde podemos visualizar las bibliografías de interés y ver los préstamos que hemos realizado.

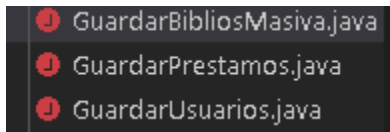
```

MenuUsuario.java X
src > proyecto > pkg1 > MenuUsuario.java > {} MenuUsuario > actionPerformed(ActionEvent)
1 package proyecto.pkg1;
2 import java.awt.*;
3 import java.awt.event.ActionEvent;
4 import java.awt.event.ActionListener;
5 import javax.swing.*;
6 import javax.swing.table.*;
7
8 import java.awt.event.*;
9
10
11 public class MenuUsuario extends JFrame implements ActionListener {
12
13     JLabel labellogo, labelusuA, textInfo;
14     JButton botonlogout, botonPrestamos, botonBuscar;
15     static JTextField textBuscar;
16     String buscar;
17
18
19
20
21
22    MenuUsuario(){
23
24
25
26        ImageIcon logo = new ImageIcon(new ImageIcon(getClass().getResource("imgs/logoInge.png")).getImage().getScaledInstance(400, 200, Image.SCALE_SMOOTH));
27        labellogo = new JLabel();
28        labellogo.setIcon(logo);
29        labellogo.setVerticalAlignment(JLabel.CENTER);
30        labellogo.setHorizontalAlignment(JLabel.CENTER);
31
32        ImageIcon imgusu = new ImageIcon(new ImageIcon(getClass().getResource("imgs/usuario.png")).getImage().getScaledInstance(70, 70, Image.SCALE_SMOOTH));
33        labelusuA = new JLabel();
34    }
35
36    public void actionPerformed(ActionEvent e) {
37        // TODO Auto-generated method stub
38    }
39
40 }

```



Estas clases se enlazan con otras clases que son donde se almacenan los textos de entradas, estas clases llevan por nombre “GuardarUsuarios”, “GuardarBiblios” y “GuardarUsuarios”, dentro de estas clases se encuentran los setters and getters.

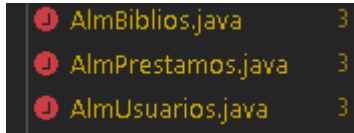


```
GuardarBibliosMasiva.java X
src > proyecto > pkg1 > GuardarBibliosMasiva.java > GuardarBibliosMasiva > GuardarBibliosMasiva(String, String, String, String, String, String, String, String, String, String)

4  /**
5   *
6   * @author marroquin
7   */
8  public class GuardarBibliosMasiva {
9      private String tipo;
10     private String autor;
11     private String titulo;
12     private String descripcion;
13     private int edicion;
14     private String []temas;
15     private String frecuenciaActual;
16     private int ejemplares;
17     private String area;
18     private int copias;
19     private int disponibles;
20
21     private String temasConcatenados;
22     private String strEdicion; //Presentación concatenada de los géneros, separados por coma
23     private String strEjemplares;
24     private String strCopias; //Presentación de tipo String del atributo copias
25     private String strDisponibles; //Presentación de tipo String del atributo existencia
26
27     public GuardarBibliosMasiva(
28         String tipo,
29         String autor,
30         String titulo,
31         String descripcion,
32         String edicion,
33         String temas,
34         String frecuenciaActual,
35         String ejemplares,
36         String area
```

```
71     public String getTipo() {
72         return this.tipo;
73     }
74
75     public void setTipo(String tipo) {
76         this.tipo = tipo;
77     }
78
79     public String getAutor() {
80         return this.autor;
81     }
82
83     public void setAutor(String autor) {
84         this.autor = autor;
85     }
86
87     public String getTitulo() {
88         return this.titulo;
89     }
90
91     public void setTitulo(String titulo) {
92         this.titulo = titulo;
93     }
94
95     public String getDescripcion() {
96         return this.descripcion;
97     }
98
99     public void setDescripcion(String descripcion) {
100         this.descripcion = descripcion;
101     }
102 }
```

Los datos encapsulados de las clases anteriores se enlazan con una clase de almacenamiento, la de usuario se enlaza con la clase “AlmUsuario” y “AlmPrestamos”, mientras que la clase administrador se enlaza con “AlmBibliografias” y “AlmPrestamos”, todas las clases realizan el mismo proceso, retornar los datos ingresados por el administrador para su manipulación.



Dentro de estas clases se encuentran distintos métodos, por ejemplo “ColocarBibliografia”, “Obtener bibliografia”, métodos los cuales podemos obtener arreglos de los datos ingresados.

```

AlmBiblios.java 3 x
src > proyecto > pkg1 > AlmBiblios.java > AlmBiblios > getBiblio()
1 package proyecto.pkg1;
2
3
4 public class AlmBiblios {
5
6     private static GuardarBibliosMasiva [] AlmBiblios = new GuardarBibliosMasiva [100];
7     private static int cantidadBiblios =0;
8
9
10
11
12
13
14
15     public static void colocarBiblio( GuardarBibliosMasiva nuevaBiblio){
16         for (int i=0; i<AlmBiblios.length; i++){
17             if (AlmBiblios[i]==null){
18                 AlmBiblios[i] = nuevaBiblio;
19                 cantidadBiblios++;
20                 return;
21             }
22         }
23     }
24
25     public static GuardarBibliosMasiva [] getBiblio(){
26         String [] biblios = new String [100];
27         int posicion =0;
28         for (GuardarBibliosMasiva biblio: AlmBiblios){
29             if (biblio != null){
30                 String [] fila ={
31                     biblio.getTipo(),
32                     biblio.getTitulo(),
33                     biblio.getAutor(),

```

Dentro de las clases para modificar usuarios o librerías se encuentra la interfaz grafica de estas clases, estas clases se enlazan con las clases de almacenamiento y manipulación para retornar los datos que se desean eliminar o modificar.

```

ModificarBiblios.java 1 x
src > proyecto > pkg1 > ModificarBiblios.java > ModificarBiblio > ModificarBiblio()
325
326     if (e.getSource()==botonCrear){
327         //CargaIndividual frame = new CargaIndividual();
328         String Autor = textAutor.getText();
329         String tipo = seleccionados;
330         String titulo = textTitulo.getText();
331         String Edicion = textEdicion.getText();
332         String Description = textDescription.getText();
333         String Temas = textTemas.getText();
334         String Frecuencia = textFrecuencia.getText();
335         String Ejemplares = textEjemplares.getText();
336         String Area = textArea.getText();
337         String Copias = textCopias.getText();
338         String Disponibles = textDisponibles.getText();
339
340
341         //int pos = AlmUsuario.verificarIngreso(usuario, contrasena);
342
343         AlmBiblios.ModificarBiblios(Buscar,tipo,titulo,Autor, Edicion, Description,
344             Temas, Frecuencia, Ejemplares,Area,Copias, Disponibles);
345
346
347         JOptionPane.showMessageDialog(null, "Bibliografía Modificada!", "Mensaje", JOptionPane.INFORMATION_MESSAGE);
348         this.dispose();
349         MenuAdmin menu = new MenuAdmin();
350     }
351
352
353     if(e.getSource()==botonBorrar){ //condicional para detectar
354         this.dispose(); //Se cierra la ventana de inicio y presenta otra ventana
355         MenuAdmin menu = new MenuAdmin();
356     }

```

Dentro de las clases “ReporteBiblio”, “ReporteUsuarios” y “ReportePrestamos” se encuentra los métodos para obtener los arreglos de las anteriores clases para elaborar el reporte en HTML, esta clase contiene código java así como lenguaje HTML, todas las clases hacen el mismo trabajo.

```
ReporteBibliografias.java X
src > proyecto > pkg1 > ReporteBibliografias.java > ReporteBibliografias
1 package proyecto.pkg1;
2
3 public class ReporteBibliografias {
4
5     private String antesDeTabla = "<DOCTYPE html>\n"+
6                                     "<html lang =\"en\">\n"+
7                                     "<head>\n"+
8                                     "<title>Reporte Bibliografias</title>\n"+
9                                     "</head>\n"+
10                                    "<body>\n"+
11                                    "<table border=\"1\">\n";
12
13
14     private String despuesDeTabla = "</table>\n"+
15                                     "</body>\n"+
16                                     "</html>";
17
18     private GuardarBibliosMasiva[] datos;
19     private String [] columnas;
20
21
22     public ReporteBibliografias(GuardarBibliosMasiva[] datos, String [] columnas){
23         this.datos = datos;
24         this.columnas = columnas;
25     }
26
27     public String obtenerReporteBiblios(){
28         String tablaColumnas = "";
29         tablaColumnas += "<tr>\n";
30
31         for (int i=0; i<columnas.length;i++){
32             tablaColumnas += "<th>"+columnas[i]+</th>\n";
33         }
34         tablaColumnas += "</tr>\n";
35
36         String tablaDatos="";
37
38         for (int i=0; i<datos.length; i++){
39             if (datos[i] != null){
40                 tablaDatos += "<tr>\n";
41                 tablaDatos += "<td>"+datos[i].getTipo()+</td>\n";
42                 tablaDatos += "<td>"+datos[i].getAutor()+</td>\n";
43                 tablaDatos += "<td>"+datos[i].getTitulo()+</td>\n";
44                 tablaDatos += "<td>"+datos[i].getDescription()+</td>\n";
45                 tablaDatos += "<td>"+datos[i].getStrEdicion()+</td>\n";
46                 tablaDatos += "<td>"+datos[i].getTemasConcatenados()+</td>\n";
47                 tablaDatos += "<td>"+datos[i].getFrecuenciaActual()+</td>\n";
48                 tablaDatos += "<td>"+datos[i].getStrEjemplares()+</td>\n";
49                 tablaDatos += "<td>"+datos[i].getArea()+</td>\n";
50                 tablaDatos += "<td>"+datos[i].getStrCopias()+</td>\n";
51                 tablaDatos += "<td>"+datos[i].getStrDisponibles()+</td>\n";
52                 tablaDatos += "</tr>\n";
53             }
54         }
55
56         return antesDeTabla+tablaColumnas+tablaDatos+despuesDeTabla;
57     }
58 }
59
60
61
```

Las clases “VerBibliografia”, “VerPrestamos” y “VerUsuarios” son las encargadas de retornarnos los datos de los arreglos en formas de tablas, estas clases entran en funcionamiento cuando el administrador da clic sobre la opción “ver” ya sea para bibliografias o usuarios.

```
VerBibliografias.java X
src > proyecto > pkg1 > VerBibliografias.java > VerBibliografias
1 package proyecto.pkg1;
2 import java.awt.*;
3 import java.awt.event.ActionEvent;
4 import java.awt.event.ActionListener;
5 import javax.swing.*;
6 import javax.swing.table.TableColumn;
7
8 import java.awt.event.ActionListener;
9
10
11 public class VerBibliografias extends JFrame implements ActionListener {
12
13     JButton botonCancelar;
14     private GuardarBibliosMasiva formCarreg;
15     private TablaBiblios mTabla;
16
17     VerBibliografias(){
18
19         //Tipo",
20         String[] columnNames = {"Tipo", "Autor", "Titulo", "Descripción", "Edición",
21                                 "Temas", "Frecuencia Actual", "Ejemplares", "Área", "copias", "Disponibles"};
22
23         String datos [][] = Almbiblios.obtenerBiblios();
24
25
26         JTable tabla = new JTable(datos, columnNames);
27         tabla.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);
28
29         tabla.setRowHeight(25);
30
31         JScrollPane sp = new JScrollPane(tabla);
32
33         botonCancelar = new JButton();
34
35     }
36 }
37
```



Las clases “VerReporteBibliografia”, “VerReportePrestamos” y “VerReporteUsuarios” son las encargadas de estructurar las tablas que se mostraran en lenguaje HTML, las tablas obtienen los arreglos de las clases “ReporteBiblio”, “ReporteUsuarios” y “ReportePrestamos”, luego de obtener los arreglos se hace llamado al método “reporteHTML” el cual se definió en las clases de Reportes y se imprime el HTML generado en un textArea.

```

VerReporteBiblios.java
src > proyecto > pkg1 > VerReporteBiblios.java > VerReporteBiblios()
1 package proyecto.pkg1;
2 import java.awt.*;
3 import java.awt.event.ActionEvent;
4 import java.awt.event.ActionListener;
5 import javax.swing.*;
6
7 public class VerReporteBiblios extends JFrame implements ActionListener {
8     JTextArea textCarga;
9     JScrollPane scrollPanel;
10    JButton botonAtras;
11
12
13    VerReporteBiblios(){
14
15        String []columnas = {"Tipo", "Autor", "Titulo", "Descripción", "Edición",
16            "Temas", "Frecuencia Actual", "Ejemplares", "Área", "Copias", "Disponibles"};
17        GuardarBibliosMasiva[] datos = AlmacenBiblios.getBiblios();
18        ReporteBibliografias generadorHtml = new ReporteBibliografias(datos,columnas);
19
20        String reporteHtml = generadorHtml.obtenerReporteBiblios();
21
22
23        textCarga = new JTextArea();
24        textCarga.setEditable(false);
25        textCarga.setLineWrap(true);
26        textCarga.setText(reporteHtml);
27        JScrollPane sp = new JScrollPane(textCarga);
28        //sp.setBounds(10,10,600,100);
29
30        botonAtras = new JButton();
31        botonAtras.setLayout(null);
32        botonAtras.setText("Atrás");
33        botonAtras.setFocusable(false);

```



Dentro de la carpeta “src” del proyecto se encuentra otra carpeta llamada “imgs” la cual contiene todas las imágenes utilizadas durante la ejecución del programa.

