UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

ESCUELA DE CIENCIAS Y SISTEMAS

PROYECTO 1:
Manual técnico

Nombre	Carne	CUI
José Eduardo Galdámez González	202109732	285299328301

Profesor: William Estuardo Escobar Argueta

Auxiliar: Héctor Josué Orozco Salazar

Fecha: 08/03/2022

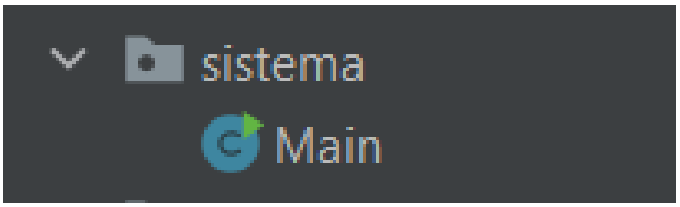
Manual técnico:

En el siguiente proyecto de programación se ha enfocado la utilización de arreglos e interfaces gráficas, utilizando a la programación orientada a objetos de JAVA, en este en el apartado grafico se trabajo con la Librería JavaFX, modelándola desde SceneBuilder y en la parte de funciones dentro de la aplicación fue en IntelliJ IDEA.

La finalidad de este proyecto es promover las buenas prácticas y poner a prueba nuestra habilidad como programadores, también el uso de la POO (programación orientada a objetos) desarrollarlo y entender su funcionalidad al trabajarlo.

Estructura del Código: En este caso especificamos y separamos toda la estructura del código. Iremos en orden para que se entienda la estructura del mismo.

1. **Sistema:** En esta parte almacenara lo que es la clase Main, la cual inicia el proyecto y el que maneja el cambio de ventanas, en este caso trabajamos un método el cual nos ayudara a abrir la ventana que deseamos y que nos ayudara a optimizar el uso de memoria.

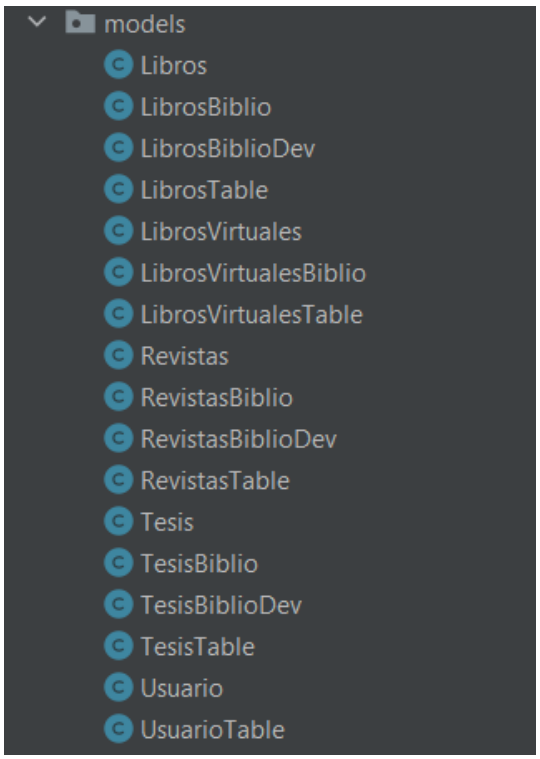



```
public class Main extends Application {


    private final String CarpetaViews="../views/";
    private Stage escenarioPrincipal;

    @Override
    public void start(Stage escenarioPrincipal) throws Exception{
        this.escenarioPrincipal=escenarioPrincipal;
        this.escenarioPrincipal.setTitle("IPC1-Proyecto 1");
        this.cambiarEscenaMain();
        this.escenarioPrincipal.show();
    }
```

```java
public void cambiarEscenaMain(){
    try{
        MainController controlador = (MainController) this.cambiarEscena( escena: "MainView.fxml", ancho: 509, alto: 416);
        controlador.setEscenarioPrincipal(this);
    }catch(IOException e){
        e.printStackTrace();
    }

}
```

```java
    public Initializable cambiarEscena(String escena, int ancho, int alto) throws IOException {
        Initializable resultado = null;
        FXMLLoader cargadorFXML = new FXMLLoader(getClass().getResource( name: this.CarpetaViews + escena));
        AnchorPane root = (AnchorPane) cargadorFXML.load();
        Scene scene = new Scene(root, ancho, alto);
        scene.getStylesheets().add("org/ipc_p1/resources/style/estilo.css");
        this.escenarioPrincipal.setScene(scene);
        this.escenarioPrincipal.sizeToScene();
        resultado = (Initializable) cargadorFXML.getController();
        return resultado;
    }


    public static void main(String[] args) { launch(args); }
}
```

2. **Models:** En esta parte almacenara lo que son las clases a utilizar como en este caso serían Usuarios, Revistas, Libros, Libros Virtuales y Tesis, y se le pondrán sus atributos correspondientes (Hay clases como Bilbio y BiblioDev que ayudaron en el apartado de las tablas de préstamos y de Bibliografía o Usuario).




```java
public class Usuario{
    //Variables para recibir.
    private String username;
    private String password;
    private int dpi;
    private String nombre;
    private String apellido;
    private String rol;
```

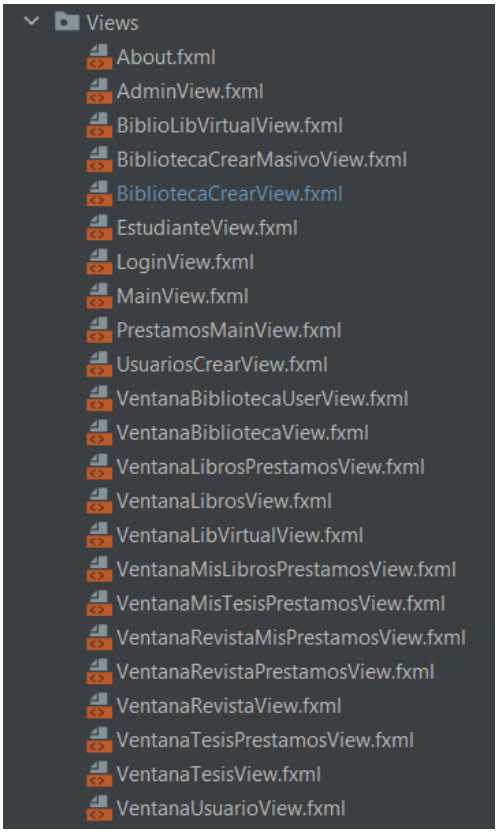
- **Constructor:** este es un elemento de una clase cuyo identificador coincide con el de la clase correspondiente y que tiene por objetivo obligar a y controlar cómo se inicializa una instancia de una determinada clase.

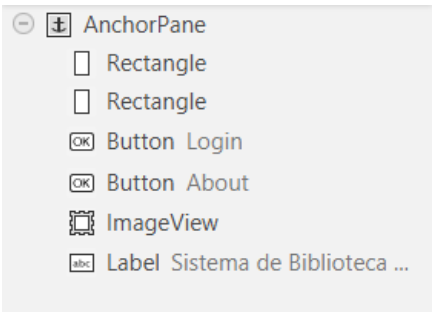
```
public Usuario(int dpi, String username, String password, String nombre, String apellido, String rol){
    this.username=username;
    this.password=password;
    this.dpi=dpi;
    this.nombre=nombre;
    this.apellido=apellido;
    this.rol=rol;
}
```

- **Getters & Setters:** son métodos de acceso a los campos/atributos de una clase. En este caso se resalta para la obtención de datos o añadimientos se utiliza estos métodos.

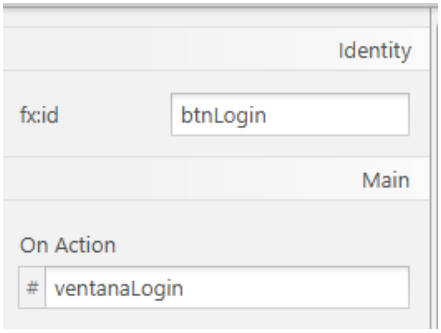
```
public String getUsername() { return username; }

public void setUsername(String username) { this.username = username; }

public String getPassword() { return password; }

public void setPassword(String password) { this.password = password; }

public int getDpi() { return dpi; }

public void setDpi(int dpi) { this.dpi = dpi; }

public String getNombre() { return nombre; }

public void setNombre(String nombre) { this.nombre = nombre; }

public String getApellido() { return apellido; }

public void setApellido(String apellido) { this.apellido = apellido; }

public String getRol() { return rol; }

public void setRol(String rol) { this.rol = rol; }
```

3. **Views:** En este aparatado se trabajó toda la parte gráfica, dándole el diseño a cada ventana desde SceneBuilder, también colocando los botones, tablas, cajas texto a utilizar dentro del programa, ya las funcionalidades se trabajaron en el apartado de "Controladores y Funciones" pero eso lo veremos más adelante.





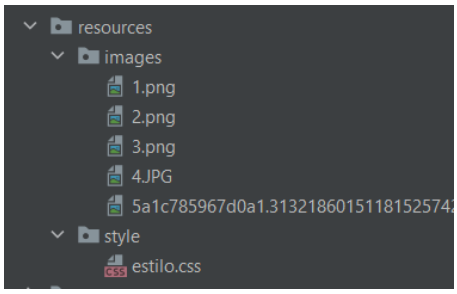
- En este caso debemos de recordar que colocamos los nombres de los objetos como serian los botones y si en dado caso tienen una función se le colocara en el apartado de acción el nombre de determinada función.



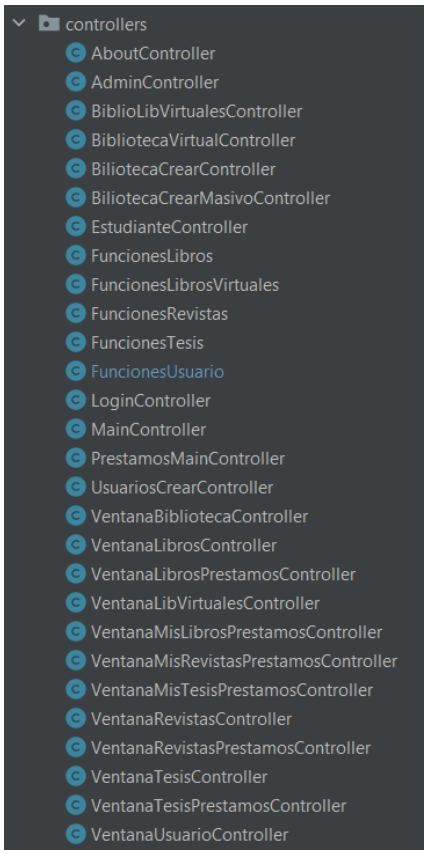
- En otra cosa importante es el asignarle el controlador definido para esa ventana para que le de sus funciones y que este ayude a que puedan cambiar entre ventanas de forma gráfica.

```
fx:controller="org.ipc_p1.controllers.MainController">
```

También en este el proyecto contiene imágenes y CSS dentro del proyecto y se guardaron en esta carpeta.



4. **Controladores y Funciones:** En este aparatado se trabajó lo que fue las funciones de como agregar, modificar, mostrar, eliminar, y demás apartados en cada clase, se dividió para que no hubiera repitencia de datos y tenga orden, en el caso de los controladores se centró en las funciones de las tablas y la obtención de datos desde las interfaces graficas.



Funciones: Para que los controladores nos puedan mandar los datos y puedan ejecutar funciones tendremos a nuestras funciones, en este caso veremos cómo es el Funciones Usuario.

```
//VARIABLES PRINCIPALES
public static int limite=100, loginNoFound=0, cont=2, c, actual,a;
private static Usuario users[]=new Usuario[100];
public static String rl;
```

```
//DATOS A MOSTRAR
public static int[] dpi=new int[100];
public static String[] nom=new String[100];
public static String[] ape=new String[100];
public static String[] usern=new String[100];
public static String[] pass=new String[100];
public static String[] rol=new String[100];
public static Usuario user=new Usuario( dpi: 2020010689, username: "admin", password: "1234", nombre: "José", apellido: "Galdámez", rol: "Administrador");
```

```
//LOGIN
public void login(String username, String password){
    for (int i=0; i<cont; i++){
        if(username.equals(users[i].getUsername()) && password.equals(users[i].getPassword())){
            Alert aviso = new Alert(Alert.AlertType.CONFIRMATION);
            aviso.setTitle("SISTEMA DE BIBLIOTECA USAC");
            aviso.setHeaderText("Login");
            aviso.setContentText("Bienvenido de nuevo "+users[i].getNombre()+ " "+ users[i].getApellido()+"!");
            aviso.show();
            res="1";
            rl=users[i].getRol();
            loginNoFound=2;
            actual=i;
            i=limite;
        }else{
            loginNoFound=1;
            res="2";
            rl="";
        }
    }

}
```

Controladores: Ya teniendo en este caso lo que es la función login procedemos a ingresar los datos en la interfaz grafica y en este apartado los controladores nos ayudaran a capturar los valores y mandárselos a nuestro login para que pueda hacer todo el proceso.

```
public void autenticar(){
    String ps, us;
    us=txtUsuario.getText();
    ps=txtPassword.getText();


    if(us.isEmpty() || ps.isEmpty()){
        Alert aviso = new Alert(Alert.AlertType.ERROR);
        aviso.setTitle("SISTEMA DE BIBLIOTECA USAC");
        aviso.setHeaderText("Login");
        aviso.setContentText("Debe de llenar los campos.");
        aviso.show();
    }else{
        funcionesUsuario.login(us, ps);
        if(FuncionesUsuario.loginNoFound==1 && funcionesUsuario.res.equals("2")){
            Alert aviso = new Alert(Alert.AlertType.ERROR);
            aviso.setTitle("SISTEMA DE BIBLIOTECA USAC");
            aviso.setHeaderText("Login");
            aviso.setContentText("Datos erroneos, porfavor vuelva a intentar.");
            aviso.show();
        }
    }
    if (funcionesUsuario.rl.equals("Administrador") && funcionesUsuario.res.equals("1")) {
        this.escenarioPrincipal.cambiarEscenaAdmin();
    } else if (funcionesUsuario.rl.equals("Usuario") && funcionesUsuario.res.equals("1")) {
        this.escenarioPrincipal.cambiarEscenaEstu();
    }
}
```

Finalmente es una breve descripción de como funciona, ya que esto pasara cada que queramos a hacer un seteado de valores, comprobación, creación, etc.

```
//CREAR USUARIO
public static void crearUsuario(int dp, String nombre, String apellido, String usuario, String password, String rol){
    int r=1;
    for (int i=0; i<cont; i++){
            if (users[i].getDpi() == dp) {
                r = 1;
                Alert aviso = new Alert(Alert.AlertType.ERROR);
                aviso.setTitle("SISTEMA DE BIBLIOTECA USAC");
                aviso.setHeaderText("NO REGISTRADO");
                aviso.setContentText("El DPI no puede repertirse en otro usuario.");
                aviso.show();
                i = cont;
                dp = 0;
            } else {
                r = 0;

            }
    }
    if(r==0){
        for (int i=1; i<=(cont); i++){
            if(users[i]==null){
                users[i]=new Usuario(dp, usuario,password, nombre,apellido,rol);
                Alert aviso = new Alert(Alert.AlertType.CONFIRMATION);
                aviso.setTitle("SISTEMA DE BIBLIOTECA USAC");
                aviso.setHeaderText("Registro guardado exitosamente!");
                aviso.setContentText("Bienvenido de nuevo "+users[i].getNombre()+ " "+ users[i].getApellido()+"!");
                aviso.show();
                dpi[i]=users[i].getDpi();
                cont=cont+1;
                i=cont;
            }
```

```
//MODIFICAR USUARIO
public static void modificarUsuario(int dp, String nombre, String apellido, String usuario, String password, String rol){
    if(users[0].getDpi()==dp){
        Alert aviso = new Alert(Alert.AlertType.ERROR);
        aviso.setTitle("SISTEMA DE BIBLIOTECA USAC");
        aviso.setHeaderText("NO MODIFICADO");
        aviso.setContentText("No puede modificar al Administrador principal.");
        aviso.show();
    }
    for (int i=1; i<(cont); i++){
        if(users[i].getDpi()==dp){
            users[i]=new Usuario(dp, usuario,password, nombre,apellido,rol);
            System.out.println(users[i].getUsername());
            Alert aviso = new Alert(Alert.AlertType.CONFIRMATION);
            aviso.setTitle("SISTEMA DE BIBLIOTECA USAC");
            aviso.setHeaderText("Registro modificado exitosamente!");
            aviso.setContentText("Puede continuar!");
            aviso.show();
            i=limite;
        }
    }
}
```

```java
//ELIMINAR USUARIO
public static void eliminarUsuario(int dp){
    if(users[0].getDpi()==dp){
        Alert aviso = new Alert(Alert.AlertType.ERROR);
        aviso.setTitle("SISTEMA DE BIBLIOTECA USAC");
        aviso.setHeaderText("NO ELIMINADO");
        aviso.setContentText("No puede ser eliminado el Administrador principal.");
        aviso.show();
    }
    for (int i=1; i<(cont); i++){
        if(users[i].getDpi()==dp ){
            users[i]=null;
            users[i]=users[cont-1];
            users[cont-1]=null;
            System.out.println(users[i].getUsername());
            Alert aviso = new Alert(Alert.AlertType.CONFIRMATION);
            aviso.setTitle("SISTEMA DE BIBLIOTECA USAC");
            aviso.setHeaderText("Registro eliminado exitosamente!");
            aviso.setContentText("Puede continuar!");
            aviso.show();
            cont=cont-1;
            i=limite;
        }
    }
}
```

```java
//MOSTRAR USUARIOS
//DPI
public static int mostrarDPI(int o){
    dpi[o]=users[o].getDpi();
    return dpi[o];
}

//NOMBRE
public static String mostrarNom(int o){
    nom[o]=users[o].getNombre();
    return nom[o];
}

//APELLIDO
public static String mostrarApe(int o){
    ape[o]=users[o].getApellido();
    return ape[o];
}

//USUARIO
public static String mostrarUser(int o){
    usern[o]=users[o].getUsername();
    return usern[o];
}

//PASSWORD
public static String mostrarPass(int o){
    pass[o]=users[o].getPassword();
    return pass[o];
}
```