

**PROGRAMACION ORIENTADA A OBJETOS I**

**SEMANA 8**

Jorge Peñaloza Guerra

22 de mayo de 2023

ingeniería en informática

# **DESARROLLO**

Caso

Una biblioteca pública del país, requiere el desarrollo de un sistema de control de préstamos de libros. Para ello, te han contratado como técnico en informática.

En la biblioteca pública, desean desarrollar un sistema de préstamos de libros, en los cuales se pueden inscribir todos los estudiantes activos en cualquier universidad o colegio. Los estudiantes, que se inscriben en la biblioteca, luego de su inscripción, pueden solicitar préstamos de los libros.

Un estudiante está identificado por: número del carnet, nombre, dirección y carrera que estudia. Se desea que por medio del sistema se puedan realizar las consultas, para así saber la cantidad de libros que tiene el estudiante como préstamos, existe una condición donde un estudiante no puede tener más de 6 libros en préstamos. De los libros se conoce su código o isbn, título, autor y también saber si está o no disponible para el préstamo; se puede conocer en donde se encuentra ubicado el libro en la biblioteca.

A continuación, realiza lo siguiente:

1. Explica el concepto de interfaz gráfica y cómo usarías las librerías en java para mostrar un formulario para el sistema de préstamos de la biblioteca.

En el caso del sistema de préstamos de la biblioteca, se utiliza una librería gráfica en Java, como Swing o JavaFX, para mostrar un formulario en la interfaz gráfica. Estas librerías proporcionan clases y componentes predefinidos que permiten crear y personalizar la apariencia de la ventana y sus elementos.

Para mostrar un formulario de préstamos de libros, se pueden utilizar las clases como JFrame, JLabel, JTextField, JButton, entre otras, proporcionadas por la librería Swing. Estas clases permiten crear la ventana principal, etiquetas para los campos, campos de texto para ingresar datos y botones para realizar acciones.

Además, se puede utilizar el patrón de diseño Modelo-Vista-Controlador (MVC) para separar la lógica del formulario de la manipulación de datos y acciones. Esto implica tener una clase controladora que maneje los eventos de los componentes y realice las acciones correspondientes, una clase de modelo que represente los datos del formulario y una clase de vista que se encargue de la visualización y actualización de la interfaz gráfica.

En este caso se puede la interfaz gráfica en las figuras a continuación donde se aplican todos los elementos anteriormente descritos

* figuras 1: Ventana principal con 3 opciones.
* figuras 2: Inscripción de estudiante, tiene todas las funciones CRUD necesarias.
* figuras 3: Ventana de solicitud de préstamo, primero se solicita el carnet del estudiante
* figuras 4: Ventana de solicitud de préstamo, después se solicita el ISBN del libro
* figuras 5: Ventana de solicitud de préstamo, muestra que pasa cuando se piden mas libros de los permitidos por el sistema
* figura 6: Listado de libros prestados

2. Elabora un cuadro comparativo de las principales librerías gráficas Swing, e indica cómo se implementa cada una de las librerías en el desarrollo del sistema de control de préstamos de la biblioteca.

|  |  |  |
| --- | --- | --- |
| Característica | Swing | JavaFX |
| Tipo de librería | Librería estándar de Java | Librería de interfaz gráfica de Java |
| Plataforma soportada | Plataformas que soporten Java | Plataformas que soporten Java |
| Lanzamiento | Introducido en JDK 1.2 | Introducido en Java SE 8 |
| Arquitectura | Basada en AWT (Abstract Window Toolkit) | Basada en una arquitectura moderna y escalable |
| Estilo de programación | Basada en Componentes | Basada en Cascading Style Sheets (CSS) y FXML |
| Personalización de apariencia | Menos flexible | Mayor flexibilidad y capacidad de personalización |
| Gráficos y animaciones | Gráficos básicos y animaciones sencillas | Potentes capacidades gráficas y de animación |
| Herramientas de diseño | Menos herramientas de diseño gráfico | Herramientas de diseño gráfico más avanzadas |
| Interoperabilidad | Buena interoperabilidad con aplicaciones más antiguas | Mejor interoperabilidad y compatibilidad con tecnologías web |
| Popularidad y comunidad | Ampliamente utilizado y comunidad activa | Aumentando su popularidad y comunidad activa |

3. Mediante un organizador gráfico señala el uso de eventos dentro de las interfaces gráficas del sistema de control de préstamos de la biblioteca.

En la figura 1 se puede apreciar que:

* Dar clic en “Opciones” abre el menú de 3 opciones
* Dar clic en “Inscripción de estudiantes” carga la ventana de inscripción de estudiantes
* Dar clic en “Solicitud de préstamos” carga la ventana de prestamos de libros
* Dar clic en “Ver libros prestados” carga la ventana para ver libros prestados

En la figura 2 se puede apreciar que:

* Dar clic en “Buscar”, procede a buscar el carnet y si lo encuentro coloca los datos en pantalla, en caso contrario indica que no existe
* Dar clic en “Guardar”, procede a insertar un nuevo registro si este no existe o actualizar el registro existente si este ya existe
* Dar clic en “Eliminar”, elimina el registro indicado y sino existe muestra un mensaje
* Dar clic en cerrar permite cerrar la venta de inscripción

En la figura 3 se puede apreciar que:

* Se solicita carnet de estudiante, si existe sigue a la venta de la figura 4 y sino, da un mensaje indicando que estudiante no existe.
* Si el estudiante tiene menos de 6 libros pedidos sigue a la ventana de la figura 4, pero si el estudiante tiene más de 5 libros pedidos dar un mensaje indicando que no se pueden prestar mas libros.

En la figura 4 se puede apreciar que:

* Pide el ISBN del libro, si este existe se procede al préstamo, y sino se da aviso por pantalla

En la figura 5 se puede apreciar que:

* Se cumple el requerimiento funcional de que no preste mas de 6 libros al mismo estudiante

En la figura 6 se puede apreciar:

* Listado de libros prestados y la persona a quien fue prestado

Además, se entrega como respaldo el código de programación en:

* Clase de JAVA 1: Clase principal con código de entrada al sistema
* Clase de JAVA 2: Clase con entorno grafico de inscripción de estudiantes
* Clase de JAVA 3: Clase con entorno grafico de préstamos de libros
* Clase de JAVA 4: Clase con entorno grafico de vista de li8bros prestados
* Clase de JAVA 5: Clase con abstracción de estudiante
* Clase de JAVA 6: Clase con abstracción de libro

Figura 1

Pantalla principal

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

Fuente: Elaboración propia

Figura 2

Ventana CRUD de inscripción de estudiantes

Tabla

Descripción generada automáticamente

Fuente: Elaboración propia

Figura 3

Solicitud de libro

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Fuente: Elaboración propia

Figura 4

Solicitud de código de libro a prestar

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Fuente: Elaboración propia

Figura 5

Solicitud de código de libro a prestar cuando se piden mas de 6 libros

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Fuente: Elaboración propia

Figura 6

Listado de libros prestados

Tabla

Descripción generada automáticamente

Fuente: Elaboración propia

Clase de JAVA 1:

Archivo: SistemaDeBiblioteca

package Biblioteca;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.util.ArrayList;  
  
public class LoanForm extends JFrame implements ActionListener {  
 private ArrayList<Estudiante> listaEstudiantes;  
 private ArrayList<Libro> listaLibros;  
  
 private JLabel lblCarnet;  
 private JTextField txtCarnet;  
 private JButton btnSolicitar;  
  
 public LoanForm(ArrayList<Estudiante> listaEstudiantes, ArrayList<Libro> listaLibros) {  
 this.listaEstudiantes = listaEstudiantes;  
 this.listaLibros = listaLibros;  
  
 // Configuración de la ventana y los componentes  
 setTitle("Formulario de Solicitud de Préstamo");  
 setDefaultCloseOperation(JFrame.*DISPOSE\_ON\_CLOSE*);  
 setLayout(new FlowLayout());  
  
 lblCarnet = new JLabel("Carnet:");  
 txtCarnet = new JTextField(10);  
 btnSolicitar = new JButton("Solicitar");  
 btnSolicitar.addActionListener(this);  
  
 add(lblCarnet);  
 add(txtCarnet);  
 add(btnSolicitar);  
  
 pack();  
 setLocationRelativeTo(null); // Centrar la ventana en la pantalla  
 setVisible(true);  
 }  
  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 if (e.getSource() == btnSolicitar) {  
 String carnet = txtCarnet.getText();  
  
 // Buscar el estudiante en la lista de estudiantes  
 Estudiante estudianteEncontrado = null;  
 for (Estudiante estudiante : listaEstudiantes) {  
 if (estudiante.getCarnet().equals(carnet)) {  
 estudianteEncontrado = estudiante;  
 break;  
 }  
 }  
  
 // Verificar si el estudiante existe y si puede solicitar más libros  
 if (estudianteEncontrado != null) {  
 if (estudianteEncontrado.getLibrosPrestados().size() < 5) {  
 // Solicitar el préstamo del libro  
 String isbn = JOptionPane.*showInputDialog*("Ingrese el ISBN del libro a solicitar:");  
 Libro libroEncontrado = null;  
 if (listaLibros != null)  
 {  
 for (Libro libro : listaLibros)  
 {  
 System.*out*.println(libro);  
 if (libro.getIsbn().equals(isbn))  
 {  
 libroEncontrado = libro;  
 break;  
 }  
 }  
 }  
 if (libroEncontrado != null && libroEncontrado.isDisponible()) {  
 estudianteEncontrado.agregarLibroPrestado(libroEncontrado);  
 libroEncontrado.setDisponible(false);  
 JOptionPane.*showMessageDialog*(this, "Préstamo realizado exitosamente");  
 } else {  
 JOptionPane.*showMessageDialog*(this, "El libro no está disponible");  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(this, "El estudiante ha alcanzado el límite máximo de préstamos");  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(this, "No se encontró un estudiante con el carnet especificado");  
 }  
 }  
 }  
}

Fuente: Elaboración propia

Clase de JAVA 2:

Archivo: StudentForm

package Biblioteca;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.util.ArrayList;  
  
public class StudentForm extends JFrame implements ActionListener {  
 // Componentes del formulario  
 private JLabel lblCarnet;  
 private JLabel lblNombre;  
 private JLabel lblDireccion;  
 private JLabel lblCarrera;  
 private JTextField txtCarnet;  
 private JTextField txtNombre;  
 private JTextField txtDireccion;  
 private JTextField txtCarrera;  
 private JButton btnBuscar;  
 private JButton btnGuardar;  
 private JButton btnEliminar;  
 private JButton btnCerrar;  
 private ArrayList<Estudiante> listaEstudiantes;  
 public StudentForm(ArrayList<Estudiante> listaEstudiantes) {  
 this.listaEstudiantes = listaEstudiantes;  
 // Configuración de la ventana  
 setTitle("Formulario de Estudiantes");  
 setDefaultCloseOperation(JFrame.*DISPOSE\_ON\_CLOSE*);  
 setLayout(new GridLayout(6, 2));  
  
 // Creación de los componentes  
 lblCarnet = new JLabel("Carnet:");  
 lblNombre = new JLabel("Nombre:");  
 lblDireccion = new JLabel("Dirección:");  
 lblCarrera = new JLabel("Carrera:");  
 txtCarnet = new JTextField();  
 txtNombre = new JTextField();  
 txtDireccion = new JTextField();  
 txtCarrera = new JTextField();  
 btnBuscar = new JButton("Buscar");  
 btnGuardar = new JButton("Guardar");  
 btnEliminar = new JButton("Eliminar");  
 btnCerrar = new JButton("Cerrar");  
  
 // Agregar los componentes al formulario  
 add(lblCarnet);  
 add(txtCarnet);  
 add(lblNombre);  
 add(txtNombre);  
 add(lblDireccion);  
 add(txtDireccion);  
 add(lblCarrera);  
 add(txtCarrera);  
 add(btnBuscar);  
 add(btnGuardar);  
 add(btnEliminar);  
 add(btnCerrar);  
  
 // Configuración de los botones  
 btnBuscar.addActionListener(this);  
 btnGuardar.addActionListener(this);  
 btnEliminar.addActionListener(this);  
 btnCerrar.addActionListener(this);  
  
 // Mostrar la ventana  
 pack();  
 setSize(400,220);  
 setLocationRelativeTo(null);  
 setVisible(true);  
 }  
  
 public void actionPerformed(ActionEvent e) {  
 if (e.getSource() == btnBuscar) {  
 // Lógica para buscar el estudiante  
 boolean sw = false;  
 JButton source = (JButton) e.getSource();  
 if (source.getText().equals("Buscar")) {  
 String carnetBuscado = txtCarnet.getText();  
  
 for (Estudiante estudiante : listaEstudiantes) {  
 if (estudiante != null && estudiante.getCarnet().equals(carnetBuscado)) {  
 txtCarnet.setText(estudiante.getCarnet());  
 txtNombre.setText(estudiante.getNombre());  
 txtDireccion.setText(estudiante.getDireccion());  
 txtCarrera.setText(estudiante.getCarrera());  
 sw = true;  
 break;  
 }  
 }  
 if(!sw) {  
 JOptionPane.*showMessageDialog*(this, "Estudiante no existe");  
 }  
 }  
 } else if (e.getSource() == btnGuardar) {  
 String carnet = txtCarnet.getText();  
 String nombre = txtNombre.getText();  
 String direccion = txtDireccion.getText();  
 String carrera = txtCarrera.getText();  
  
 Estudiante estudianteExistente = null;  
 for (Estudiante estudiante : listaEstudiantes) {  
 if (estudiante != null && estudiante.getCarnet().equals(carnet)) {  
 estudianteExistente = estudiante;  
 break;  
 }  
 }  
  
 if (estudianteExistente != null) {  
 // Actualizar los datos del estudiante existente  
 estudianteExistente.setNombre(nombre);  
 estudianteExistente.setDireccion(direccion);  
 estudianteExistente.setCarrera(carrera);  
 JOptionPane.*showMessageDialog*(this, "Datos actualizados correctamente");  
 } else {  
 // Crear un nuevo estudiante y agregarlo al array  
 Estudiante nuevoEstudiante = new Estudiante(carnet, nombre, direccion, carrera);  
 listaEstudiantes.add(nuevoEstudiante);  
 JOptionPane.*showMessageDialog*(this, "Estudiante agregado correctamente");  
 }  
 } else if (e.getSource() == btnEliminar) {  
 String carnet = txtCarnet.getText();  
  
 // Buscar el estudiante en el ArrayList  
 Estudiante estudianteEncontrado = null;  
 for (Estudiante estudiante : listaEstudiantes) {  
 if (estudiante.getCarnet().equals(carnet)) {  
 estudianteEncontrado = estudiante;  
 txtCarnet.setText("");  
 txtNombre.setText("");  
 txtDireccion.setText("");  
 txtCarrera.setText("");  
 break;  
 }  
 }  
  
 // Eliminar el estudiante si fue encontrado  
 if (estudianteEncontrado != null) {  
 listaEstudiantes.remove(estudianteEncontrado);  
 JOptionPane.*showMessageDialog*(this, "Estudiante eliminado");  
 // Aquí puedes agregar la lógica adicional que desees después de eliminar el estudiante  
 } else {  
 JOptionPane.*showMessageDialog*(this, "No se encontró un estudiante con el carnet especificado");  
 }  
 } else if (e.getSource() == btnCerrar) {  
 // Cerrar la ventana  
 dispose();  
 }  
 }  
}

Fuente: Elaboración propia

Clase de JAVA 3:

Archivo: LoanForm

package Biblioteca;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.util.ArrayList;  
  
public class LoanForm extends JFrame implements ActionListener {  
 private ArrayList<Estudiante> listaEstudiantes;  
 private ArrayList<Libro> listaLibros;  
  
 private JLabel lblCarnet;  
 private JTextField txtCarnet;  
 private JButton btnSolicitar;  
  
 public LoanForm(ArrayList<Estudiante> listaEstudiantes, ArrayList<Libro> listaLibros) {  
 this.listaEstudiantes = listaEstudiantes;  
 this.listaLibros = listaLibros;  
  
 // Configuración de la ventana y los componentes  
 setTitle("Formulario de Solicitud de Préstamo");  
 setDefaultCloseOperation(JFrame.*DISPOSE\_ON\_CLOSE*);  
 setLayout(new FlowLayout());  
  
 lblCarnet = new JLabel("Carnet:");  
 txtCarnet = new JTextField(10);  
 btnSolicitar = new JButton("Solicitar");  
 btnSolicitar.addActionListener(this);  
  
 add(lblCarnet);  
 add(txtCarnet);  
 add(btnSolicitar);  
  
 pack();  
 setLocationRelativeTo(null); // Centrar la ventana en la pantalla  
 setVisible(true);  
 }  
  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 if (e.getSource() == btnSolicitar) {  
 String carnet = txtCarnet.getText();  
  
 // Buscar el estudiante en la lista de estudiantes  
 Estudiante estudianteEncontrado = null;  
 for (Estudiante estudiante : listaEstudiantes) {  
 if (estudiante.getCarnet().equals(carnet)) {  
 estudianteEncontrado = estudiante;  
 break;  
 }  
 }  
  
 // Verificar si el estudiante existe y si puede solicitar más libros  
 if (estudianteEncontrado != null) {  
 if (estudianteEncontrado.getLibrosPrestados().size() < 5) {  
 // Solicitar el préstamo del libro  
 String isbn = JOptionPane.*showInputDialog*("Ingrese el ISBN del libro a solicitar:");  
 Libro libroEncontrado = null;  
 if (listaLibros != null)  
 {  
 for (Libro libro : listaLibros)  
 {  
 System.*out*.println(libro);  
 if (libro.getIsbn().equals(isbn))  
 {  
 libroEncontrado = libro;  
 break;  
 }  
 }  
 }  
 if (libroEncontrado != null && libroEncontrado.isDisponible()) {  
 estudianteEncontrado.agregarLibroPrestado(libroEncontrado);  
 libroEncontrado.setDisponible(false);  
 JOptionPane.*showMessageDialog*(this, "Préstamo realizado exitosamente");  
 } else {  
 JOptionPane.*showMessageDialog*(this, "El libro no está disponible");  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(this, "El estudiante ha alcanzado el límite máximo de préstamos");  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(this, "No se encontró un estudiante con el carnet especificado");  
 }  
 }  
 }  
}

Fuente: Elaboración propia

Clase de JAVA 4:

Archivo: LibrosPrestadosViewer

package Biblioteca;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.util.ArrayList;  
  
public class LibrosPrestadosViewer extends JFrame {  
 private ArrayList<Estudiante> listaEstudiantes;  
  
 public LibrosPrestadosViewer(ArrayList<Estudiante> listaEstudiantes) {  
 this.listaEstudiantes = listaEstudiantes;  
  
 // Configuración de la ventana  
 setTitle("Libros Prestados");  
 setDefaultCloseOperation(JFrame.*DISPOSE\_ON\_CLOSE*);  
 setLayout(new BorderLayout());  
  
 // Creación del modelo de tabla  
 String[] columnas = {"Carnet", "Nombre", "Libros Prestados"};  
 Object[][] datos = obtenerDatosTabla();  
 JTable tablaLibrosPrestados = new JTable(datos, columnas);  
  
 // Configuración de la tabla  
 JScrollPane scrollPane = new JScrollPane(tablaLibrosPrestados);  
 add(scrollPane, BorderLayout.*CENTER*);  
  
 // Mostrar la ventana  
 pack();  
 setSize(500, 300);  
 setLocationRelativeTo(null);  
 setVisible(true);  
 }  
  
 private Object[][] obtenerDatosTabla() {  
 int totalLibrosPrestados = 0;  
 for (Estudiante estudiante : listaEstudiantes) {  
 totalLibrosPrestados += estudiante.getLibrosPrestados().size();  
 }  
  
 Object[][] datos = new Object[totalLibrosPrestados][3];  
 int fila = 0;  
  
 for (Estudiante estudiante : listaEstudiantes) {  
 ArrayList<Libro> librosPrestados = estudiante.getLibrosPrestados();  
 for (Libro libro : librosPrestados) {  
 datos[fila][0] = estudiante.getCarnet();  
 datos[fila][1] = estudiante.getNombre();  
 datos[fila][2] = libro.getTitulo();  
 fila++;  
 }  
 }  
  
 return datos;  
 }  
}

Fuente: Elaboración propia

Clase de JAVA 5:

Archivo: Estudiante

package Biblioteca;  
  
import java.util.ArrayList;  
  
*/\*\*  
 \* La clase Estudiante representa a un estudiante en el sistema de préstamo de libros.  
 \*/*public class Estudiante {  
 // Atributos de la clase  
 private String carnet;  
 private String nombre;  
 private String direccion;  
 private String carrera;  
 private ArrayList<Libro> librosPrestados;  
  
 */\*\*  
 \* Constructor de la clase Estudiante.  
 \* @param carnet El número del carnet del estudiante.  
 \* @param nombre El nombre del estudiante.  
 \* @param direccion La dirección del estudiante.  
 \* @param carrera La carrera que el estudiante está estudiando.  
 \*/* public Estudiante(String carnet, String nombre, String direccion, String carrera) {  
 this.carnet = carnet;  
 this.nombre = nombre;  
 this.direccion = direccion;  
 this.carrera = carrera;  
 this.librosPrestados = new ArrayList<>();  
 }  
  
 // Getters y Setters  
 */\*\*  
 \* @return El número del carnet del estudiante.  
 \*/* public String getCarnet() {  
 return carnet;  
 }  
  
 */\*\*  
 \* @param carnet El nuevo número de carnet del estudiante.  
 \*/* public void setCarnet(String carnet) {  
 this.carnet = carnet;  
 }  
  
 */\*\*  
 \* @return El nombre del estudiante.  
 \*/* public String getNombre() {  
 return nombre;  
 }  
  
 */\*\*  
 \* @param nombre El nuevo nombre del estudiante.  
 \*/* public void setNombre(String nombre) {  
 this.nombre = nombre;  
 }  
  
 */\*\*  
 \* @return La dirección del estudiante.  
 \*/* public String getDireccion() {  
 return direccion;  
 }  
  
 */\*\*  
 \* @param direccion La nueva dirección del estudiante.  
 \*/* public void setDireccion(String direccion) {  
 this.direccion = direccion;  
 }  
  
 */\*\*  
 \* @return La carrera que el estudiante está estudiando.  
 \*/* public String getCarrera() {  
 return carrera;  
 }  
  
 */\*\*  
 \* @param carrera La nueva carrera que el estudiante está estudiando.  
 \*/* public void setCarrera(String carrera) {  
 this.carrera = carrera;  
 }  
  
 public ArrayList<Libro> getLibrosPrestados() {  
 return librosPrestados;  
 }  
  
 public void setLibrosPrestados(ArrayList<Libro> librosPrestados) {  
 this.librosPrestados = librosPrestados;  
 }  
  
 public void agregarLibroPrestado(Libro libro) {  
 librosPrestados.add(libro);  
 }  
}

Fuente: Elaboración propia

Clase de JAVA 6:

Archivo: Libro

package Biblioteca;  
*/\*\*  
 \* La clase Libro representa a un libro en el sistema de préstamo de libros.  
 \*/*public class Libro {  
 // Atributos de la clase  
 private String isbn;  
 private String titulo;  
 private String autor;  
 private boolean disponible;  
 private String ubicacion;  
  
 */\*\*  
 \* Constructor de la clase Libro.  
 \* @param isbn El código o ISBN del libro.  
 \* @param titulo El título del libro.  
 \* @param autor El autor del libro.  
 \* @param disponible Si el libro está o no disponible para préstamo.  
 \* @param ubicacion Donde se encuentra el libro en la biblioteca.  
 \*/* public Libro(String isbn, String titulo, String autor, boolean disponible, String ubicacion) {  
 this.isbn = isbn;  
 this.titulo = titulo;  
 this.autor = autor;  
 this.disponible = disponible;  
 this.ubicacion = ubicacion;  
 }  
  
 // Getters y Setters  
 */\*\*  
 \* @return El código o ISBN del libro.  
 \*/* public String getIsbn() {  
 return isbn;  
 }  
  
 */\*\*  
 \* @param isbn El nuevo código o ISBN del libro.  
 \*/* public void setIsbn(String isbn) {  
 this.isbn = isbn;  
 }  
  
 */\*\*  
 \* @return El título del libro.  
 \*/* public String getTitulo() {  
 return titulo;  
 }  
  
 */\*\*  
 \* @param titulo El nuevo título del libro.  
 \*/* public void setTitulo(String titulo) {  
 this.titulo = titulo;  
 }  
  
 */\*\*  
 \* @return El autor del libro.  
 \*/* public String getAutor() {  
 return autor;  
 }  
  
 */\*\*  
 \* @param autor El nuevo autor del libro.  
 \*/* public void setAutor(String autor) {  
 this.autor = autor;  
 }  
  
 */\*\*  
 \* @return Si el libro está o no disponible para préstamo.  
 \*/* public boolean isDisponible() {  
 return disponible;  
 }  
  
 */\*\*  
 \* @param disponible El nuevo estado de disponibilidad del libro.  
 \*/* public void setDisponible(boolean disponible) {  
 this.disponible = disponible;  
 }  
  
 */\*\*  
 \* @return Donde se encuentra el libro en la biblioteca.  
 \*/* public String getUbicacion() {  
 return ubicacion;  
 }  
  
 */\*\*  
 \* @param ubicacion La nueva ubicación del libro en la biblioteca.  
 \*/* public void setUbicacion(String ubicacion) {  
 this.ubicacion = ubicacion;  
 }  
}

Fuente: Elaboración propia

**REFERENCIAS BIBLIOGRÁFICAS**

IACC. (2022). Programación Orientada a Objetos con Java. En *Programación orientada a objetos I. Semana 8.*