Documentación de Refactorización

Introducción

En este documento hemos descrito las refactorizaciones realizadas en nuestro proyecto de gestión de biblioteca. El objetivo ha sido mejorar la claridad, legibilidad y mantenimiento del código sin modificar su funcionalidad.

Refactorizaciones realizadas

Clase `Libro`

Antes:

- El constructor permitía crear libros con campos inválidos sin validación.
- No se manejaba correctamente la inmutabilidad de algunos atributos.

Después:

- Añadida validación en el constructor para campos `titulo`, `autor`, `anyoPublicacion` y `genero`.
- Los atributos se definieron como `final` para garantizar que no cambien una vez creado el objeto.

Ventajas:

- Se evitan objetos `Libro` con datos inválidos.
- Se evita la modificación de libros ya guardados.
- Código más seguro y robusto.

Clase `Biblioteca`

Antes:

- Métodos de búsqueda no usaban streams, código menos limpio.
- No se manejaban casos de libros duplicados al agregar.

Después:

- Implementación de streams para búsquedas y filtrados, mejorando legibilidad.
- Control de duplicados en el método `agregarLibro`.

Ventajas:

- Código más compacto y legible.
- Evita añadir libros repetidos.

Clase `Main`

Antes:

- Código con menos estructura, entradas sin validación.

Después:

- Implementación de menú interactivo con opciones claras.
- Uso de `switch` con expresiones lambda para simplificar.

Ventajas:

- Mejor experiencia de usuario.
- Código más organizado y mantenible.

Conclusión

Los cambios realizados mejoran bastante el código, tanto a la hora de evitar errores como puedan ser duplicaciones de libros o falta de información a la hora de introducirlos, como a la hora de tener un codigo más limpio y que facilite la legibilidad de este mismo.