Programmer Documentation

Technical Stack: Java (Swing, AWT, I/O)

2.1. System Overview

The application is a standard Java Swing GUI that manages a list of Storage objects. Data persistence is handled with Java Serialization (Serializable), storing the object graph directly into a binary file (store.dat).

2.2. Class Structure

1. Book (Model - Base Class)

- **Package:** bookstore
- **Role:** Represents the abstract data of a book.
- **Attributes:**
    - author, title, topic (Strings)
    - year, price (integers)
- **Key Interface:** Implements Serializable to allow file storage.

2. Storage (Model - Subclass)

- **Inheritance:** Extends Book.
- **Role:** Represents a concrete physical inventory item.
- **Attributes:**
    - shelfNumber: Location in the store.
    - piece: Current quantity in stock.
- **Key Methods:**
    - addPiece(int): Increments the inventory count.

3. Store (Data Access Object / Controller)

- **Role:** Manages the collection of books and handles File I/O.
- **Attributes:**
    - List<Storage> books: The in-memory database.
- **Key Methods:**
    - writeToFile(String filename): Serializes the list to store.dat.
    - readFromFile(String filename): Deserializes the list on startup.
    - addStorage(Storage): Adds a new object to the list.

4. StoreGUI (View & Main Logic)

- **Inheritance:** Extends JFrame.
- **Role:** The main entry point and UI handler.
- **Layout:** Uses BorderLayout containing a JPanel with a GridLayout for the menu buttons.
- **Event Handling:** Uses Lambda expressions to route actions to handleButton(String label).
- **Validation:**
    - Ensures numeric inputs (Year, Price, Stock) are non-negative.
    - Uses try-catch blocks to handle NumberFormatException.

2.3. Data Persistence Flow

1. **Startup:** StoreGUI initializes a Store object. The Store constructor calls readFromFile("store.dat"). If the file is missing or corrupt, a new empty ArrayList is created.
2. **Runtime:** Operations (Add, Modify, Delete, Upload) modify the books list in memory.
3. **Save:** Every time a modification occurs , the store.writeToFile("store.dat") method is called immediately to persist changes.

2.4. Business Logic Details

- **Low Stock Threshold:** Hardcoded as **10**.
    - Used in orderBook(): Filters list for items < 10.
    - Used in uploadBooks(): Only prompts user to restock items < 10.
- **Search Logic:** Linear search iterating through the books list.

2.5. Future Improvements (TODO)

- **ID System:** Currently, books are identified by the combination of Author + Title. Adding a unique ISBN or ID field would prevent duplicates and make searching more robust.
- **UI Separation:** Extract the logic from StoreGUI into a dedicated StoreController class to strictly follow MVC patterns.
- **Configuration:** Move the "Low Stock Threshold" (currently 10) to a constant or config file.