DWA_07.4 Knowledge Check_DWA7

---

1. Which were the three best abstractions, and why?

BookList Class:
This class encapsulates the entire functionality of the BookConnect app. It abstracts away the detail of rendering, event handling and state management. It provides a high level interface for initializing the app and its features. It adheres to the principle of encapsulation and makes the code modular and more readable.

Dropdown Population Methods:
Through abstraction the logic for my dropdown population is isolated, making it reusable for different dropdowns. By encapsulating the logic the code becomes more maintainable and changes can be made in a single place. These methods contribute to the readability of my code by providing a clear and concise way of setting up dropdowns.

Search and Filter Methods:
These methods collectively form an abstraction for handling user searches and updating the list.These methods provide clear separation of what functionality they have within the code. This abstraction enhances maintainability and readability by isolating the search and filter logic.

---

2. Which were the three worst abstractions, and why?

Hardcoded Element Reference:
Several methods such as ‘openSearchOverlay’, ‘closeSearchOverlay’, ‘openSettingsOverlay’ and others directly reference HTML elements using document.querySelector. This creates a tight coupling between the JavaScript and HTML structure, making it less flexible. Passing elements as parameters to these methods would be more advantageous for abstraction.

Strings in Theme Update:
The updateTheme method uses string literals to determine the theme(‘day’ and ‘night’). By using constants for themes I can avoid typos and make the code more maintainable.

EventListener Setup Method:
My ‘setupEventListeners’ method directly references HTML elements. This tightly couples the JS and HTML making it less flexible. Passing relevant elements as parameters will make code more decoupled.

---

3. How can The three worst abstractions be improved via SOLID principles.

Hardcoded Element Reference:
Dependency Inversion Principle (DIP)
Instead of directly referencing HTML elements, pass elements as parameters or use injection to make code more flexible.

Strings in Theme Update:
Open/Closed Principle(OCP)
Open the code for extension but closed for modification. Using constants or an enum-like structure for themes, allows for easy extension, without modification.

EventListener Setup Method:
Dependency Inversion Principle(DIP)
Depend on abstractions instead of concrete implementations. Pass elements as parameters or use dependency injection to make code more flexible.
