DWA_07.4 Knowledge Check_DWA7

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

*Book: The Book abstraction represents the structure of a book object and defines its properties. It encapsulates the book-related data and allows for consistent usage throughout the codebase. This abstraction promotes code organization, readability, and maintainability.

*createPreviewElement: This abstraction encapsulates the logic for creating a preview element for a book. It takes a book object as input and returns an HTML button element representing the book preview. By abstracting the creation of the preview element, this function promotes code reusability, separation of concerns, and improves the readability of the code that uses it.

*createOptionElements: This abstraction handles the creation of option elements for a dropdown menu. It takes a container, default value, and options as inputs, and generates the corresponding option elements. By encapsulating this logic, it promotes code modularity and reusability. This abstraction allows for easier modification and customization of the dropdown menu without affecting other parts of the code.

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

*createOptionElements: This abstraction violates the Open-Closed Principle (OCP) as it directly appends the created option elements to the DOM within the function. Instead, the function should return the created option elements, and the responsibility of appending them to the DOM should be delegated to the caller. By doing so, the function becomes more flexible, allowing for easier extension or modification of the DOM manipulation behavior.

*createPreviewElement: The createPreviewElement function tightly couples the HTML structure of the preview element with the logic for creating it. This makes it challenging to

*Modify or customize the appearance of the preview element without modifying the function itself. To improve this, consider separating the HTML structure into a separate template or component. By doing this, modifications or customization of the preview element's appearance can be achieved without altering the core logic of the function.

---

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

*Refactor app.init to separate the event listener setup for each UI element into separate functions or classes, following the SRP.

*Modify createOptionElements to make it return the created option elements instead of directly appending them to the DOM, adhering to the OCP.

*Decouple the HTML structure of the preview element from the createPreviewElement function by using a separate template or component, allowing for easier modification or customization without changing the core logic.
