DWA_07.4 Knowledge Check_DWA7

---

1. Which were the three best abstractions, and why?
Data Object (data): The data object serves as a container for various data attributes related to different sections of the application, such as header, list, search, and settings. It abstracts away the individual DOM element selections and organizes them into a structured object, making it easier to access and manage the data throughout the code.

Book Preview Creation (createPreview function): The createPreview function abstracts the process of creating a book preview element by encapsulating the HTML structure and rendering logic. It takes a book object as input and returns a button element with the appropriate HTML content. This abstraction promotes reusability and separation of concerns, making it easier to generate book previews in multiple parts of the application.

Theme Management (option.theme): The option.theme object abstracts the management of themes within the application. It dynamically sets the CSS properties for the color scheme based on the user's selection. By encapsulating the theme-related functionality in this object, it provides a clear and reusable approach for handling theme changes in the UI.

---

2. Which were the three worst abstractions, and why?
Inline CSS Styling: In several places within the code, inline CSS styling is used directly on DOM elements, such as setting the src attribute of an image or applying styles to elements using the style property. This approach can lead to code duplication, reduced maintainability, and decreased separation of concerns. It would be better to define CSS classes and apply them to elements using classList or modify the CSS styles through CSS files or stylesheets.

Nested Event Listeners: The code includes nested event listeners, where event listeners are attached inside other event listeners. This can result in complex and difficult-to-maintain code, as the logic becomes tightly coupled and nested. It is generally recommended to extract event listeners into separate functions or use event delegation to handle events in a more organized and modular manner.

Overuse of Global Variables: The code relies heavily on global variables, such as data, matches, page, and css. Overuse of global variables can lead to naming conflicts, unintended side effects, and difficulty in tracking variable states. It is preferable to encapsulate data and functionality within appropriate scopes and use local variables or modules to limit the global scope pollution

---

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

Improving these abstractions via SOLID principles may involve refactoring the code to adhere to principles like Single Responsibility, Open/Closed, and Dependency Inversion. By introducing better modularization, encapsulation, and separation of concerns, the code can become more maintainable, flexible, and extensible.
