DWA_07.4 Knowledge Check_DWA7

---

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

- `const createPreview = ({ ... }) => { ... }`: This function abstracts the creation of a preview button for a book, encapsulating the logic for generating the button's HTML structure. It takes a book object as input and produces the corresponding HTML button element.

- `const SearchOptions = (Cartegory) => { ... }`: This function abstracts the creation of filter options for authors or genres as document fragments. It encapsulates the logic for generating these filter options based on a provided category.

- `const sliceBooks = (booksSlice) => { ... }`: This function abstracts the process of looping through a portion of books and creating preview buttons for them. It takes a slice of books as input and generates a document fragment containing preview buttons.

---

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

- dayMode(), and nightMode()
  - ❖ functions are not abstracted. It would be better to create a function for theme switching to abstract this logic.

---

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

- For **Single Responsibility Principle (SRP)**
  A function or class should have only one reason to change with a single responsibility. In my codes **`createPreview`, `SearchOptions`, `nightMode`, and `dayMode`** all have clear and distinct responsibilities.

- For **Open/Closed Principle (OCP)**
  Your code appears to follow this principle fairly well. For example, when you want to add more options or filters for authors or genres, you can do so without modifying the existing code by calling **`appendingMoreOptions`**. Similarly, adding more themes can be done without changing the existing code.
