DWA_07.4 Knowledge Check_DWA7

---

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

‘createBookElement’ function.

1. The function abstracts away the complexity of creating the book element with HTML.
2. It encapsulates the code that sets up the button, image and information elements.
3. This function promotes readability and maintainability through isolating the creation of a book element.
4. If I need to change the structure of displaying a book, I only have to update this function.

‘populateDropdown’ function.

1. Abstracts logic for populating a dropdown with options.
2. Handles creation of option elements.
3. Encapsulates the logic, making it modular.
4. If I need to change how dropdowns are populated, I only need to focus on this function.

‘filterBooks’ function.

1. Encapsulates the logic for filtering books based on search criteria.
2. Abstracts away the detail of how filtering is done.
3. It separates the concerns of filtering books from other parts of app.
4. If adjustment had to be made, adjustment could be made in this function without affecting the rest of my code.

---

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

‘loadMoreBooks’ function.

1. This function takes the ‘bookList’ object and relies on external variables, which can lead to unintended side effects.

‘updateBookList’ function.

1. This function depends on the external state (‘bookList.matches’ and ‘bookList.booksPerPage’), making it less modular.
2. Handles both showing/hiding and updating the book list.

‘renderBooksFragment’ function.

1. This function is very tightly coupled to the structure of the book data (‘{author, id, image, title}’) and the createBookElement function, which makes it less flexible.

---

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

‘loadMoreBooks’ function.

1. Single Responsibility Principle: Refactor function to have a single responsibility.
2. Dependency Inversion Principle: Instead of relying on the ‘bookList’ object directly, pass required parameters explicitly to the function.

‘updateBookList’ function.

1. Single Responsibility Principle:This function could be improved by splitting the responsibilities of showing/hiding and updating the book list.
2. Dependency Inversion Principle: Pass required parameters explicitly, reducing dependency on external state.

‘renderBooksFragment’ function.

1. Single Responsibility Principle: Separate the responsibilities by creating a new function for creating a book element.
2. Dependency Inversion Principle: Function depends on specific structure of book data, which can be improved by allowing the caller to provide a function for extracting relevant data from items.
