DWA_08 Discussion Questions

In this module you will continue with your "Book Connect" codebase, and further iterate on your abstractions. You will be required to create an encapsulated abstraction of the book preview by means of a single factory function. If you are up for it you can also encapsulate other aspects of the app into their own abstractions.

To prepare for your session with your coach, please answer the following questions. Then download this document as a PDF and include it in the repository with your code.

---

1. What parts of encapsulating your logic were easy?

- Defining functions: I have created a standalone functions like
  createPreviewElement,
  appendPreviewElementToFragment,
  setTheme, updateListButton,
  handleSettingsFormSubmit,
  handleSearchFormSubmit,
  updateListItems,
  handleListButtonClick,
  handleListItemClick,
  and setup
  The functions are defined using the arrow function syntax and have clear purposes.

- Organizing code into functions: The code is organized into separate functions based on their functionality, which makes it easier to understand and maintain. Each function has a specific task and can be reused if necessary.

- Event handling: The code utilizes event listeners to handle various user interactions, such as form submissions, button clicks, and item clicks. Event handling is a common and relatively straightforward task in JavaScript.

---

2. What parts of encapsulating your logic were hard?

- Managing event handlers:

Handling events correctly and ensuring they are attached to the appropriate elements can be challenging, especially when dealing with complex UI interactions.
The code attached event listeners to different elements, such as buttons and form submissions.

- Updating and synchronizing data:
  Keeping the data in sync and ensuring the UI reflects the updated state can be tricky, especially when dealing with asynchronous operations or complex filtering conditions.

  The code maintains a list of matches and page variables, which need to be updated based on user actions and form submissions.

---

3. Is abstracting the book preview a good or bad idea? Why?

- Readability and Maintainability:
  By abstracting the book preview code into a separate function (createPreviewElement), the main code becomes more readable and easier to understand. The logic for creating the preview element is encapsulated in a single function, making it easier to maintain and modify if needed.

- Reusability:
  By abstracting the code, I can easily reuse the function in multiple places if I need to create the same or similar preview elements elsewhere in my application. This promotes code reuse and helps avoid duplication.

- Separation of Concerns:
  Abstracting the book preview code allows me to separate the UI-related logic from the rest of the code. This makes the code more modular and follows the principle of separation of concerns, where each function or module is responsible for a specific task.

- Testing:
  Abstracting the book preview code makes it easier to write unit tests for the function. You can test the createPreviewElement function independently, ensuring that it generates the correct HTML structure and attributes for the book

preview.