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.

---

1. What parts of encapsulating your logic were easy?

```
/**
 * Creates option elements for a dropdown menu.
 * @param {string} container - The dropdown container.
 * @param {string} defaultValue - The default value of the dropdown.
 * @param {Object} options - The options for the dropdown.
 */
function createOptionElements(container, defaultValue, options) {
  const fragment = document.createDocumentFragment();
  const firstElement = document.createElement('option');
  firstElement.value = defaultValue;
  firstElement.innerText = `All ${container}`;
  fragment.appendChild(firstElement);

  for (const [id, name] of Object.entries(options)) {
    const element = document.createElement('option');
    element.value = id;
    element.innerText = name;
    fragment.appendChild(element);
  }

  document.querySelector(`[data-search-${container}]`).appendChild(fragment);
}

createOptionElements('genres', 'any', genres);
createOptionElements('authors', 'any', authors);
```

- **createOptionElements(container, defaultValue, options)** : because this function can be encapsulated as a separate function declaration. It creates option elements for a dropdown menu based on the provided container, **default value, and options**.

---

2. What parts of encapsulating your logic were hard?

```
// Update page and matches with filtered results
page = 1;
matches = result;
```

The hardest parts to encapsulate in the given code are the variables and the code that directly manipulates the DOM.

1. **Variables**: The variables **page** and **matches** are declared outside of any specific function or object. Encapsulating these variables might require rethinking the structure of the code and determining the appropriate scope for these variables. One approach could be to encapsulate them within an object or a class, along with the functions that manipulate them. This was also difficult in the sense that I did not know whether I needed to put either **let** or **const.**

---

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

Abstracting the book preview is a good idea because:-

- Abstracting the book preview into a separate function promotes **reusability** allowing you to generate consistent book previews in different contexts by just calling the **createBookPreview** function and improves **readability** making it easier for other developers including myself, to understand the code's intention.

- It is also a good idea because it enhances **maintainability** for when you need to make changes or enhancements to the book preview functionality so you will

only need to modify the code in one place which is the **createBookPreview** function and it facilitates **effective testing** of the function helping identify bugs and promotes overall code quality.
