DWA_09.3 Challenge 1

In this challenge, you will continue with your “Book Connect” codebase and further iterate on your abstractions.

Previously, you worked on adding abstraction around the book preview functionality of the project. Next, you must turn the book preview abstraction into a fully-working web component. Then, apply the techniques you’ve learned about this module to the book preview.

---

1. What problems did you encounter converting the book preview to a component?

Abstraction : Custom Elements and Rendering Logic operate at a higher level of abstraction compared to basic HTML and CSS. As a new developer with understanding how to structure simple web pages with standard tags like <div> and <p>. Custom Elements introduce a more abstract way of thinking, which can be initially confusing.
Creating custom HTML tags, like <book-preview>, can feel like inventing new things.

Understanding the logic flow which involves thinking about how a component should behave and present itself on the screen. This requires breaking down complex interactions into step-by-step instruction, which can be a significantly confusing leap for new developers.

Shadow DOM : Understanding the concept of creating a hidden, encapsulated DOM inside another DOM element can be challenging.

---

2. What other elements make sense to convert into web components? Why?

- Theme Switcher: This is like a tool we use everywhere in our app to change its look. Making a "theme-switcher" component means we build it once and use it everywhere, so our app looks the same no matter where we are.

- "Show More" Button: We often need a button that lets users see more stuff in our app. A "show-more-button" component makes it easy. We create it once, and we can use it everywhere in our app without doing the same work over and over.

- Search Overlay: When we have a search box that pops up in the same way all over our app, a "search-overlay" component is handy. It takes care of how the search box appears and disappears, saving us from writing the same code each time.

---

3. Why does keeping your HTML, CSS and JavaScript in a single file sometimes make sense?

From my understanding, having everything in one file can simplify things, especially for smaller projects. It's easier to distribute and share. But I'm also aware of the downsides. As projects get bigger, maintaining everything in a single file might become challenging. Separating concerns between HTML, CSS, and JavaScript is a good practice for readability.