DWA_07.4 Knowledge Check_DWA7

---

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

- The **applyTheme** function abstracts the reasoning behind applying a chosen theme to the webpage. It accepts a theme as input and modifies the documentElement's data-theme attribute accordingly. Because the theme application logic is contained within a separate function, this abstraction enhances the readability and maintainability of the code.

- The function of **createGenreOptionsHtml** is the process of adding genre options to the search dialog that is abstracted by this function. The options elements are dynamically created depending on the key-value pairs as it iterates across the genres object. By separating the production of genre alternatives from the primary logic, this abstraction increases code reuse and scalability.

- The **handleSearchSubmit** function, this function manages the search form submission and conducts the search depending on user input. By separating the search logic into a distinct function, it abstracts the functionality of searching. This abstraction enhances the readability and modularity of the code.

---

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

- The **handleSearchResults** function manages the presentation of search results. While it accomplishes what it needs to, it may be made better by separating the handling logic from the results rendering. By dividing this function into smaller functions, one of which would be in charge of handling the logic and the other of which would be in charge of rendering the results, the Single Responsibility Principle (SOLID) would be used.

- The **createAuthorOptionsHtml** function: the search dialog's author options are created by this function. It mixes the creation of alternatives with the event-handling logic, much like handleSearchResults does. The method might be broken up into several functions, isolating the option creation from the event handling, to better this abstraction.

- The handleSettingsSave function: This function manages user settings saving. It accomplishes the necessary jobs but might use a better way to separate its concerns. The storage of settings can be decoupled from the logic that applies the chosen theme by using the single responsibility principle. As a result, there would be two distinct functions, each in charge of a specific task.

---

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

- The SRP, or Single Responsibility Principle Separate the presentation of search results from the handling logic by breaking the handleSearchResults function into smaller functions. The code should be simpler to modify and maintainable by having a single responsibility for each function.

- Split the createAuthorOptionsHtml function into smaller functions in accordance with the single responsibility principle (SRP). The logic for handling events should be handled by one function, and the creation of author options should be handled by another. This division of duties enhances the readability and reusability of the code.

- The SRP or single responsibility principle Creates two distinct routines from the handleSettingsSave function. The logic for implementing the chosen theme should be handled by one function, while the saving of user settings should be handled by another. The functions become more focused and maintainable because of this division, which also helps code structure.
