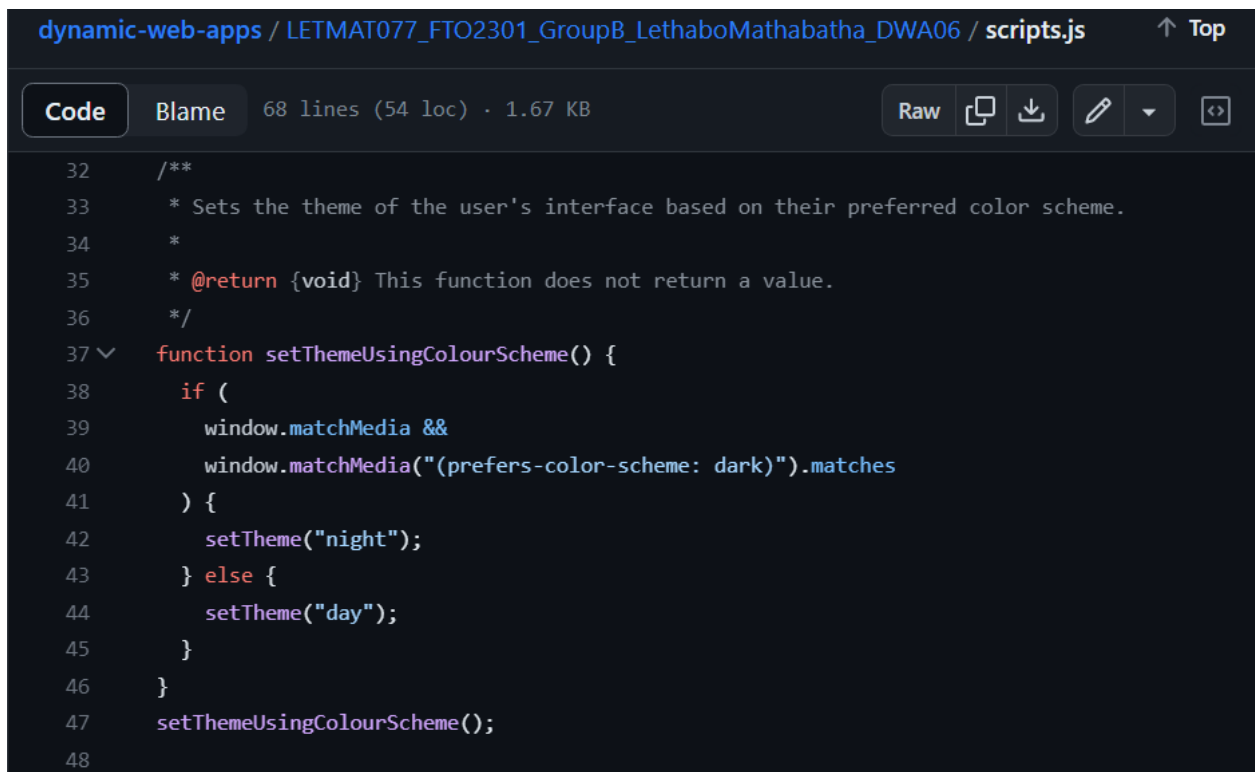


DWA_07.4 Knowledge Check_DWA7

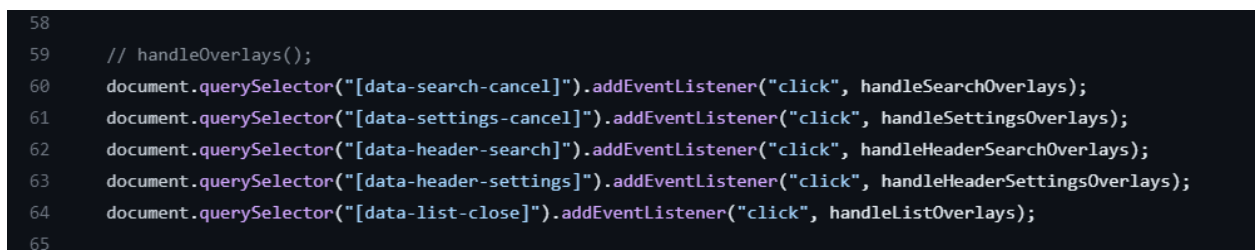
1. Which were the three best abstractions, and why?

a. Using a function to separate the purpose of the original them change



```
dynamic-web-apps / LETMAT077_FTO2301_GroupB_LethaboMathabatha_DWA06 / scripts.js ↑ Top
Code Blame 68 lines (54 loc) · 1.67 KB
32  /**
33   * Sets the theme of the user's interface based on their preferred color scheme.
34   *
35   * @return {void} This function does not return a value.
36   */
37  function setThemeUsingColourScheme() {
38      if (
39          window.matchMedia &&
40          window.matchMedia("(prefers-color-scheme: dark)").matches
41      ) {
42          setTheme("night");
43      } else {
44          setTheme("day");
45      }
46  }
47  setThemeUsingColourScheme();
48
```

b.



```
58
59  // handleOverlays();
60  document.querySelector("[data-search-cancel]").addEventListener("click", handleSearchOverlays);
61  document.querySelector("[data-settings-cancel]").addEventListener("click", handleSettingsOverlays);
62  document.querySelector("[data-header-search]").addEventListener("click", handleHeaderSearchOverlays);
63  document.querySelector("[data-header-settings]").addEventListener("click", handleHeaderSettingsOverlays);
64  document.querySelector("[data-list-close]").addEventListener("click", handleListOverlays);
65
```

c.

```
66 // settings handler
67 document
68   .querySelector("[data-settings-form]")
69   .addEventListener("submit", handleSettingsFormSubmit);
70
71
```

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

```
71
72 // search handler
73 processSearchForm(authors, books, BOOKS_PER_PAGE, page, matches);
74
75 // preview handler
76 handlePreviews(authors, books, BOOKS_PER_PAGE, page, matches);
77
78 // active book/list handler
79 handleActiveBooks(authors, books, BOOKS_PER_PAGE, page, matches);
80
```

dynamic-web-apps / LETMAT077_FTO2301_GroupB_LethaboMathabatha_DWA06 / helper.js

↑ Top

Code

Blame

356 lines (292 loc) · 10.8 KB

Raw



```
318
319 // data list items
320 export function handleActiveBooks(authors, books, BOOKS_PER_PAGE, page, matches) {
321   document
322     .querySelector("[data-list-items]")
323     .addEventListener("click", (event) => {
324       const pathArray = Array.from(event.path || event.composedPath());
325       let active = null;
326
327       for (const node of pathArray) {
328         if (active) break;
329
330         if (node?.dataset?.preview) {
331           let result = null;
332
333           for (const singleBook of books) {
334             if (result) break;
335             if (singleBook.id === node?.dataset?.preview) result = singleBook;
336           }
337
338           active = result;
339         }
340       }
341
342       if (active) {
343         document.querySelector("[data-list-active]").open = true;
344         document.querySelector("[data-list-blur]").src = active.image;
345         document.querySelector("[data-list-image]").src = active.image;
346         document.querySelector("[data-list-title]").innerText = active.title;
347         document.querySelector("[data-list-subtitle]").innerText = `${
348           authors[active.author]
349         } (${new Date(active.published).getFullYear()})`;
350         document.querySelector("[data-list-description]").innerText =
351           active.description;
352       }
353     });
354 }
```

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

```
dynamic-web-apps / LETMAT077_FTO2301_GroupB_LethaboMathabatha_DWA06 / helper.js
Code Blame 356 lines (292 loc) · 10.8 KB Raw Copy Download Edit View
318
319 // data list items
320 export function handleActiveBooks(authors, books, BOOKS_PER_PAGE, page, matches) {
321   document
322     .querySelector("[data-list-items]")
323     .addEventListener("click", (event) => {
324       const pathArray = Array.from(event.path || event.composedPath());
325       let active = null;
326
327       for (const node of pathArray) {
328         if (active) break;
329
330         if (node?.dataset?.preview) {
331           let result = null;
332
333           for (const singleBook of books) {
334             if (result) break;
335             if (singleBook.id === node?.dataset?.preview) result = singleBook;
336           }
337
338           active = result;
339         }
340       }
341
342       if (active) {
343         document.querySelector("[data-list-active]").open = true;
344         document.querySelector("[data-list-blur]").src = active.image;
345         document.querySelector("[data-list-image]").src = active.image;
346         document.querySelector("[data-list-title]").innerText = active.title;
347         document.querySelector("[data-list-subtitle]").innerText = `${
348           authors[active.author]
349         } (${new Date(active.published).getFullYear()})`;
350         document.querySelector("[data-list-description]").innerText =
351           active.description;
352       }
353     });
354 }
```

Two principles of SOLID can be used:

- **Single Responsibility Principle (SRP):** The handleActiveBooks function could have a single responsibility, which is handling the click event on the data list items. To improve SRP, I could extract the logic for finding the active book into a separate function.
- **Dependency Inversion Principle (DIP):** The handleActiveBooks function currently has direct dependencies on the authors and books variables. To improve DIP, I could pass these dependencies as function arguments instead of relying on variables from the outer scope.