IWA19_01 FINAL CAPSTONE

Lesego Tacia Chiloane

Overview

These are my recommendations to improve the readability, performance and maintainability of the code for the book connect web app.

Scope of this presentation:

1. Most bugs in the code resulted from syntax errors.
2. This presentation will not focus on syntax errors.
3. Rather it will focus on how the could be better written if it were complete. That is, if there were no syntax errors.

Readability

Readable code is maintainable code. And vice versa. Therefore, it important to keep code readable for use in the long run.

Recommendations:

- Rearrange by functionality.
- Remove unused/unnecessary code.
- Use functions to break code up.
- Use destructuring to access objects & arrays.

Break up code by function

Page Structure

1. View books and previews with title, author, year & summary.
2. See 36 books at a time.
3. Scroll down using a button to show more.

Display Settings

1. Load page with system dark & light mode settings.
2. Toggle dark & light mode setting on page.

Book Search

1. Search for books by title, author & genre.
2. Do a strict search with parameters for title, author & genre.

Variable names

```
/**
 * For saving color values for day and night settings.
 * @type {object} css
 */
const css = {
    day : ['255, 255, 255', '10, 10, 20'],
    night: ['10, 10, 20','255, 255, 255']
};
```

```
data-search-authors.appendChild(authors)

data-settings-theme.value === window.matchMedia && window.matchM
v = window.matchMedia && window.matchMedia('(prefers-color-schem

documentElement.style.setProperty('--color-dark', css[v].dark);
documentElement.style.setProperty('--color-light', css[v].light)
data-list-button = "Show more (books.length - BOOKS_PER_PAGE)"

data-list-button.disabled = !(matches.length - [page * BOOKS_PER

data-list-button.innerHTML = /* html */ [
    '<span>Show more</span>',
    '<span class="list__remaining"> (${matches.length - [page *
]

data-search-cancel.click() { data-search-overlay.open === false
data-settings-cancel.click() { querySelect(data-settings-overlay
data-settings-form.submit() { actions.settings.submit }
data-list-close.click() { data-list-active.open === false }
```

Readability

Recommendations:

- Arrange code by functionality.
- Separate code into files. Longer code is difficult to read.
- Make descriptive variable names. Makes it easier to track the purpose of variables.

Using object destructuring

```
export const createPreview = (props) => {
    const { author, image, title, id } = props;

    const newElement = document.createElement('button');
    newElement.className = 'preview';
    newElement.setAttribute('data-preview', id);

    newElement.innerHTML = /* HTML */`
        <img
            class="preview__image"
            src="${image}"
        />

        <div class="preview__info">
            <h3 class="preview__title">${title}</h3>
            <div class="preview__author">${author}</div>
        </div>
    `

    return newElement;
};
```

Performance

Recommendations:

- Use import & export to access code between files..
- Use const or let to declare variables. Otherwise variables might be overwritten, resulting in bugs.
- Remove unused/unnecessary code.
- Store DOM elements in object.

Selecting elements

Store DOM elements in object as stored querySelector variables rather than calling querySelector for the same element multiple times.

```
export const html = {
    view: {
        mainHtml: document.querySelector('[data-list-items]'),
    },
    scroll: {
        moreButton: document.querySelector('[data-list-button]'),
    },
    preview: {
        summaryList: document.querySelectorAll('[data-preview]'),
        summaryOverlay: document.querySelector('[data-list-active]'),
        summaryBlur: document.querySelector('[data-list-blur]'),
        summaryImage: document.querySelector('[data-list-image]'),
        summaryTitle: document.querySelector('[data-list-title]'),
        summarySubTitle: document.querySelector('[data-list-subtitle]'),
        summaryDescription: document.querySelector('[data-list-description]'),
        summaryClose: document.querySelector('[data-list-close]'),
    },
    display: {
        settingsOverlay: document.querySelector('[data-settings-overlay]'),
        settingButton: document.querySelector('[data-header-settings]'),
        settingsTheme: document.querySelector('[data-settings-theme]'),
        settingsCancel: document.querySelector('[data-settings-cancel]'),
        settingsSubmit: document.querySelector('[data-settings-form]'),
    },
    search: {
        searchCancel: document.querySelector('[data-search-cancel]'),
        searchButton: document.querySelector('[data-header-search]'),
        searchOverlay: document.querySelector('[data-search-overlay]'),
        seacrhTitle: document.querySelector('[data-search-title]'),
        searchSubmit: document.querySelector('[data-search-form]'),
        searchAuthors: document.querySelector('[data-search-authors]'),
        searchAuthors: document.querySelector('[data-search-authors]'),
        searchGenres: document.querySelector('[data-search-genres]'),
        seachMessage: document.querySelector('[data-list-message]'),
```

Performance

Recommendations:

- Use defer to load javascript last.
- Use strict comparisons to account for type conversions. These may result in bugs later.
- Remove unused variables.
- Test code. Especially using the console log. Some error could have been fixed by looking console log errors.

Loading JavaScript

```
// Imports from other files.

import {
    BOOKS_PER_PAGE,
    authors,
    genres,
    books,
} from "./data.js"

import {
    html,
    showPreview,
    createPreviewsFragment,
    updateRemaining,
    searchAll,
    searchAuthor,
    searchGenre,
    searchNothing,
    searchTitle,
} from "./functions.js";

```

```
<link rel="apple-touch-icon" sizes="180x180" href="/meta/apple-touch-i
<link rel="icon" type="image/png" sizes="32x32" href="/meta/favicon-32
<link rel="icon" type="image/png" sizes="16x16" href="/meta/favicon-16
<link rel="manifest" href="/meta/manifest.json">
<link rel="mask-icon" href="/meta/safari-pinned-tab.svg" color="#0096f
<link rel="shortcut icon" href="/meta/favicon.ico">
<meta name="msapplication-TileColor" content="#0a0a14">
<meta name="msapplication-config" content="/meta/browserconfig.xml">
<meta name="theme-color" content="#0a0a14">

<link rel="stylesheet" href="./styles.css" />
<link rel="preconnect" href="https://fonts.googleapis.com" />
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />

<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@500;7

<!-- Use scripts as module & defer loading of JS. -->
<script src="./scripts.js" defer type="module"></script>
```

Maintanability

Recommendations:

- Add JSDoc comments for all functions.
- Add comments for variables with type.

JSDoc commenting

```
/**
 * Searches the books array for book objects given an author id from
 * the search form.
 *
 * @param {string} author id to search for in books
 * @param {number} index of current book object in books
 * @param {array} searchResult array where searchresults will be stored
 */
export const searchAuthor = (author, index, searchResult) => {
    let authorMatch;
    if (author !== 'any' && books[index].author.includes(author)) {
        authorMatch = books[index];
        searchResult.push(authorMatch);
    };
};
```