Diego Jeftha

Capstone Project Presentation

IWA19

Query Selectors

```
const dataHeaderSearch = document.querySelector('[data-header-search]')
const dataHeaderSettings = document.querySelector('[data-header-settings]')
const dataListItems = document.querySelector('[data-list-items]')
const dataListMessage = document.querySelector('[data-list-message]')
const dataListButton = document.querySelector('[data-list-button]')
const dataListActive = document.querySelector('[data-list-active]')
const dataListBlur = document.querySelector('[data-list-blur]')
const dataListImage = document.querySelector('[data-list-image]')
const dataListTitle = document.querySelector('[data-list-title]')
const dataListSubtitle = document.querySelector('[data-list-subtitle]')
const dataListDescription = document.querySelector('[data-list-description]')
const dataListClose = document.querySelector('[data-list-close]')
const dataSearchOverlay = document.querySelector('[data-search-overlay]')
const dataSearchForm = document.querySelector('[data-search-form]')
const dataSearchTitle = document.querySelector('[data-search-title]')
const dataSearchGenres = document.querySelector('[data-search-genres]')
const dataSearchAuthors = document.querySelector('[data-search-authors]')
const dataSearchCancel = document.querySelector('[data-search-cancel]')
const dataSettingsOverlay = document.querySelector('[data-settings-overlay]')
const dataSettingsForm = document.querySelector('[data-settings-form]')
const dataSettingsTheme = document.querySelector('[data-settings-theme]')
const dataSettingsCancel = document.querySelector('[data-settings-cancel]')
```

First for loop creates a list of book previews,extracts first 36 books from array called ‘books’

```
/**
* for loop below creates a list of book previews
* extracts first 36 books from array called 'books'
*/
for (let i = 0; i < extracted.length; i++) {
    const { author: authorId, id, image, title } = extracted[i]
    const element = document.createElement('button')          //creates a new button
    element.classList = 'preview'
    element.setAttribute('data-preview', id)
    element.innerHTML = /* html */ `
        <img
            class="preview__image"
            src="${image}"
        />
        <div class="preview__info">
            <h3 class="preview__title">${title}</h3>
            <div class="preview__author">${authors[authorId]}</div>
        </div>
    `
    fragment.appendChild(element)       // appends button element to fragment doc
}

dataListItems.appendChild(fragment)
```

Creates dropdown list of book genres, creates new document named ‘genreOfBooks’ to store options element.

```
/**
 * creates a dropdownlist of book genres
 * creates a new docuemnt named 'genreOfBooks' to store options element
 */
const genreOfBook = document.createDocumentFragment()
const optionOfBooks = document.createElement('option')
optionOfBooks.value = 'any'
optionOfBooks.textContent = 'All Genres'
genreOfBook.appendChild(optionOfBooks)

/**
 * loops over the 'genre' object using a for...of loop
 * each time it loops, creates a new option element and sets its value attribute to current id
 * and textContent to current name
 * appends option element to genreOfBooks
 */
for (let [id, name] of Object.entries(genres)) {
    const optionsOfGenre = document.createElement('option')
    optionsOfGenre.value = id
    optionsOfGenre.textContent = name
    genreOfBook.appendChild(optionsOfGenre)
}

dataSearchGenres.appendChild(genreOfBook)
```

Loops over the ‘genre’ object using for…of loop. Each time it loops, creates a new option element and sets value attribute to current id and textConent to current name.

Appends option element to ‘genreOfBooks’

Show more Button

```
dataListButton.innerHTML = /* html */[
    `<span>Show more</span>
    <span class="list__remaining"> (${matches.length - [page * BOOKS_PER_PAGE] > 0 ?
matches.length - [page * BOOKS_PER_PAGE] : 0})</span>`,
]

dataListButton = `Show more (${books.length} - ${BOOKS_PER_PAGE})`
dataListButton.disabled = !(matches.length - (page * BOOKS_PER_PAGE) > 0)

dataListButton.addEventListener('click', () => {

document.querySelector('[data-list-items]').appendChild(createPreviewsFragment(matches
, page * BOOKS_PER_PAGE, (page + 1) * BOOKS_PER_PAGE, page));
    actions.list.updateRemaining();
});
```

Setting Button
Sets the theme to light or dark

```
const v = window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches ? 'night' : 'day';
dataSettingsTheme.value === window.matchMedia || window.matchMedia('(prefers-color-scheme: dark)').matches ? 'night' : 'day';
```

```
document.documentElement.style.setProperty('--color-dark', css[v].dark);
document.documentElement.style.setProperty('--color-light', css[v].light);
```

```
dataSettingsForm.addEventListener("submit", (event) => {
    event.preventDefault()
    const formSubmit = new FormData(event.target)
    const option = Object.fromEntries(formSubmit)
    if (option.theme === 'night') {
        document.documentElement.style.setProperty('--color-light', css[option.theme][1])
        document.documentElement.style.setProperty('--color-dark', css[option.theme][0])
    } else if (option.theme === 'day') {
        document.documentElement.style.setProperty('--color-light', css[option.theme][1])
        document.documentElement.style.setProperty('--color-dark', css[option.theme][0])
    }
    dataSettingsOverlay.close()
})
```

When the ‘dataHeaderSettings’ element is clicked, function focuses on the input field and checks if the settings overlay is open or not.

```
dataHeaderSettings.addEventListener('click', () => {
    dataSettingsTheme.focus();

    if (dataSettingsOverlay.open) {
        dataHeaderSettings.showModal()
    } else {
        dataSettingsOverlay.showModal();
    }
})
```

When the cancel button is clicked, closes the setting overlay.

```
dataSettingsCancel.addEventListener('click', () => {
    dataSettingsOverlay.close();
})
```

Search button

When the search button is clicked, function focuses on the input field and checks if the search overlay is open or not

```
dataHeaderSearch.addEventListener('click', () => {
    dataSearchTitle.focus();

    if (dataSearchOverlay.open) {
        dataHeaderSearch.showModal()
    } else {
        dataSearchOverlay.showModal();
    }
})
```

When the cancel button is clicked, closes the overlay

```
dataSearchCancel.addEventListener('click', () => {
    dataSearchOverlay.close();
})
```

Books Preview

When a book is clicked on, brings up a preview of the book

```
dataListItems.addEventListener('click', (event) => {
    const pathArray = Array.from(event.path || event.composedPath());
    let active = null;

    for (const node of pathArray) {
        if (active) {
            break;
        }
        const previewId = node?.dataset?.preview

        for (const singleBook of books) {
            if (singleBook.id === previewId) {
                active = singleBook
            }
        }
    }
if (!active) {
        return;
    }
    dataListActive .open = true;
    dataListBlur .src, dataListImage .src = active. image;
    dataListTitle .textContent = active. title;

    dataListSubtitle .textContent = `${authors [active .author ]} (${new
Date (active .published ) .getFullYear () }) `
    dataListDescription .textContent = active. description ;
})
```

Books Filter

Fixed syntax errors

```
let booksList = [];

dataSearchForm.addEventListener('click', (filters, event) => {
    preventDefault();    // prevents default form submit behavior
    const formData = new FormData(event.target);
    filters = Object.fromEntries(formData);
    const result = [];

    let titleMatch = null;
    let authorMatch = null;
    let genreMatch = null;

    for (const book of booksList) {
        titleMatch= filters.title.trim() = '' || book.title.toLowerCase().includes[filters.title.toLowerCase()];  //
use || instead of &&
        authorMatch= filters.author = 'any' || book.author === filters.author
    }

    if (filters.genre = 'any') {
        genreMatch= true;
    } else {
        genreMatch= false;
    }
```

```
for (const genre of books.genres) {
        if (genre = filters.genre) {
            genreMatch === true
        }

        if (titleMatch && authorMatch && genreMatch) {
            result.push(books)
        }

        if (display.length < 1) {
            dataListMessage.class.add('list__message_show')
        } else {
            dataListMessage.class.remove('list__message_show')
        }
```