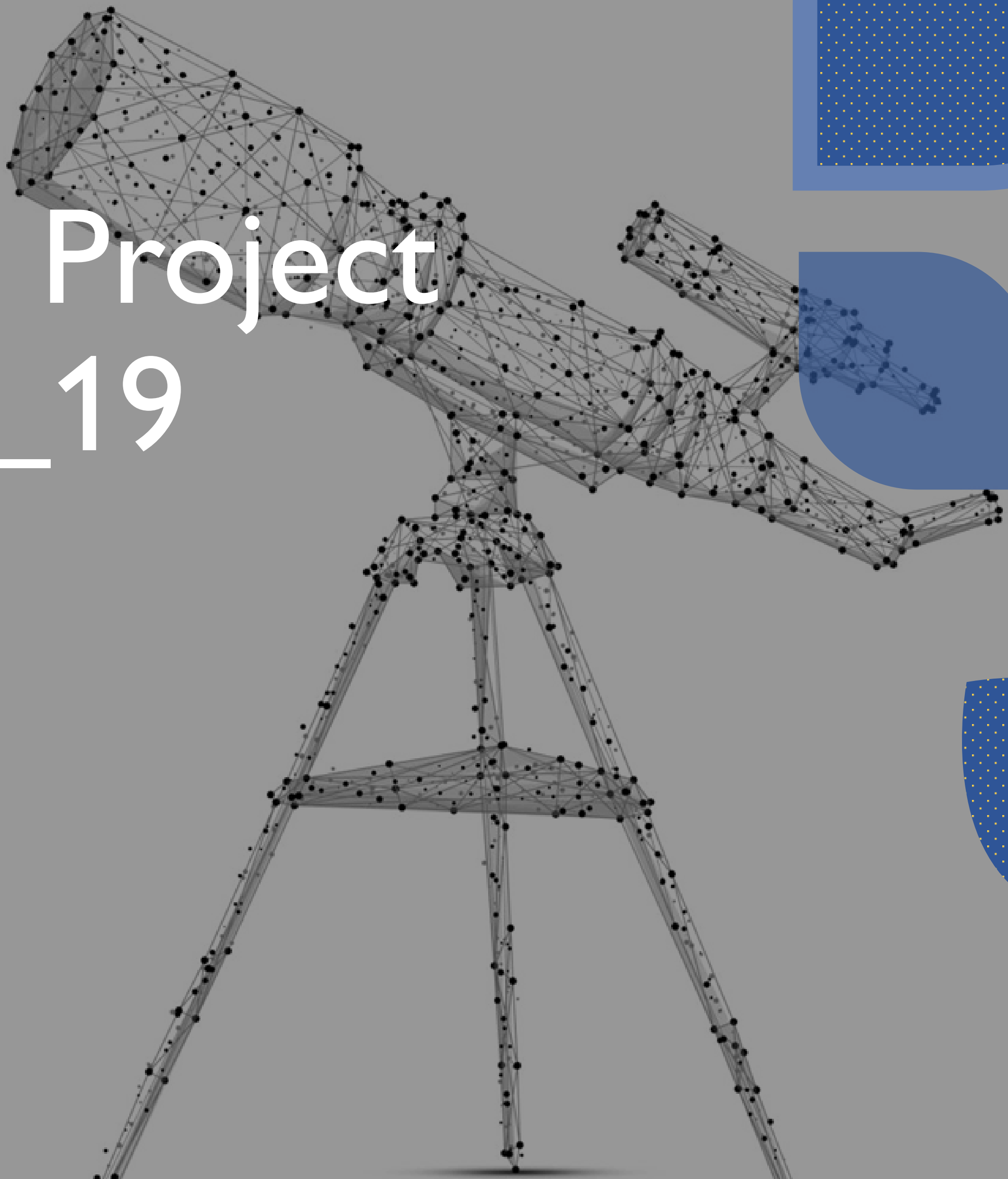
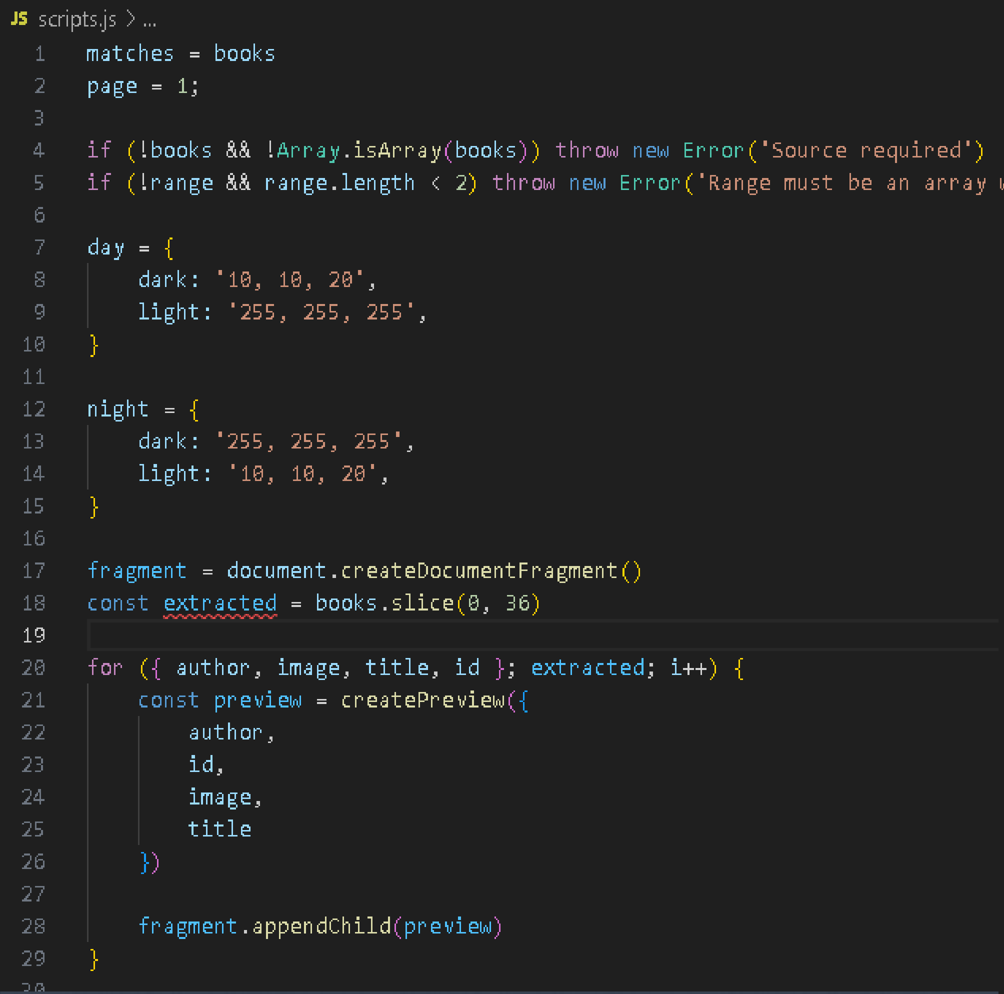
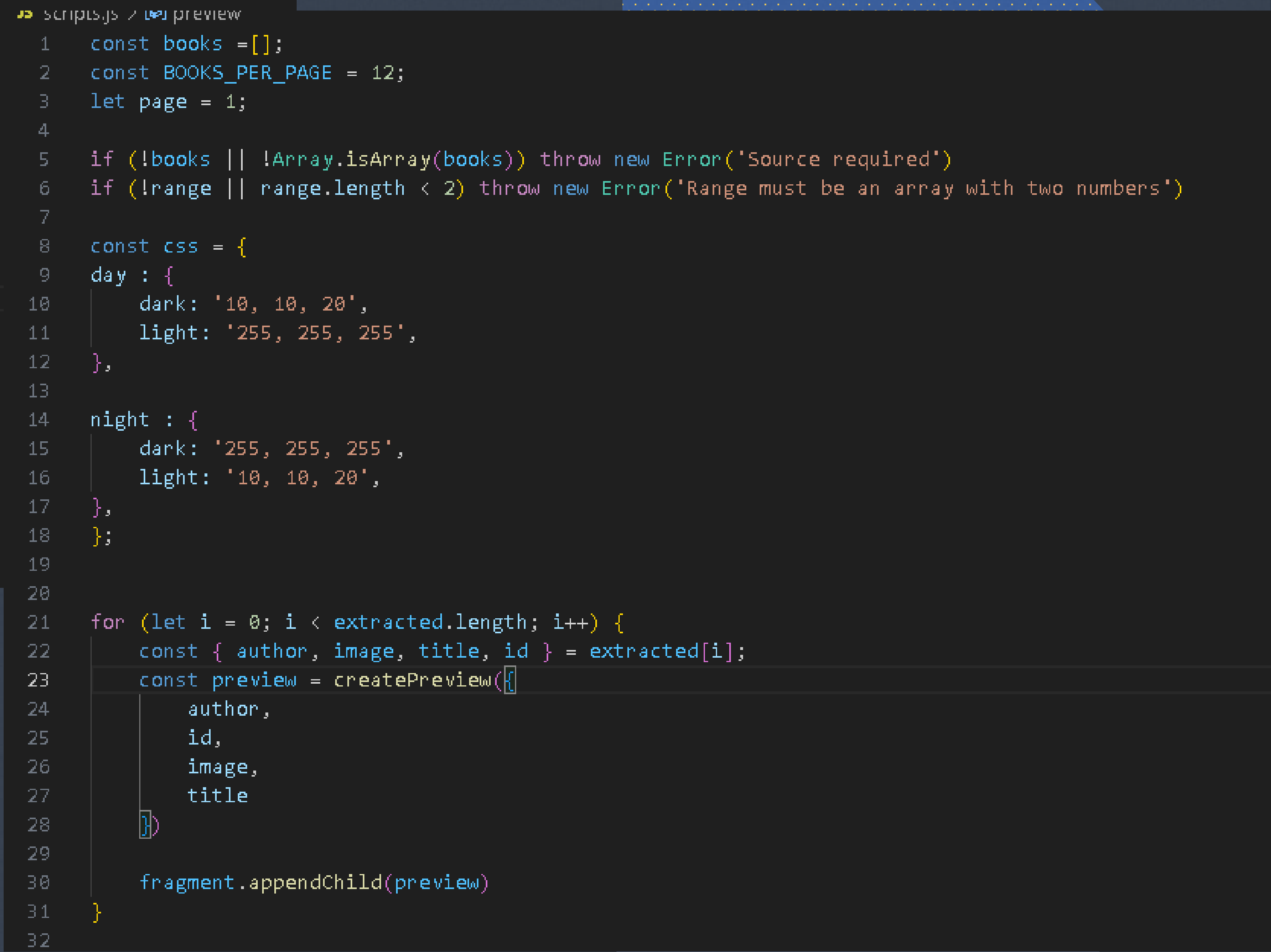
Capstone Project
IWA_19

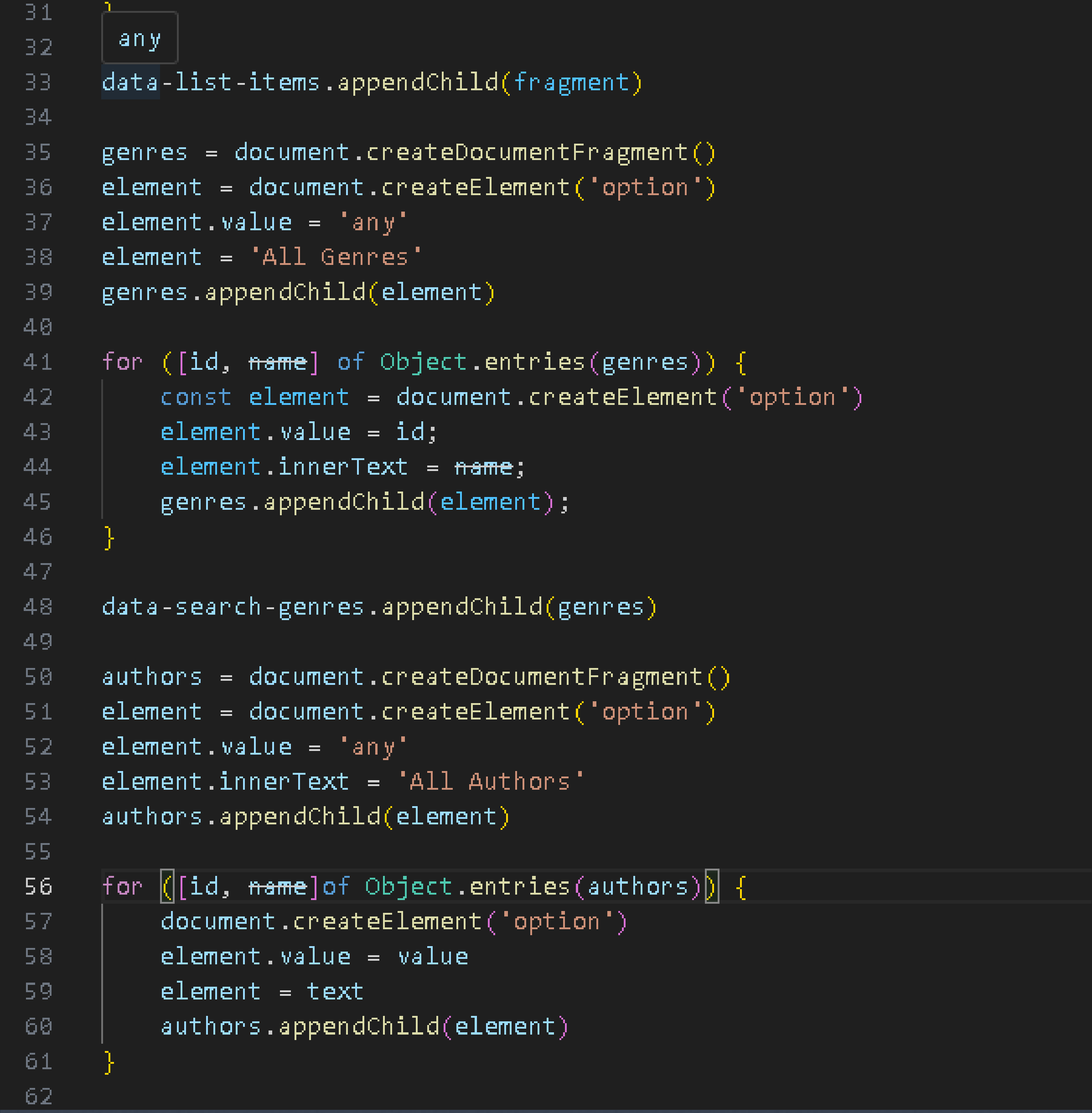




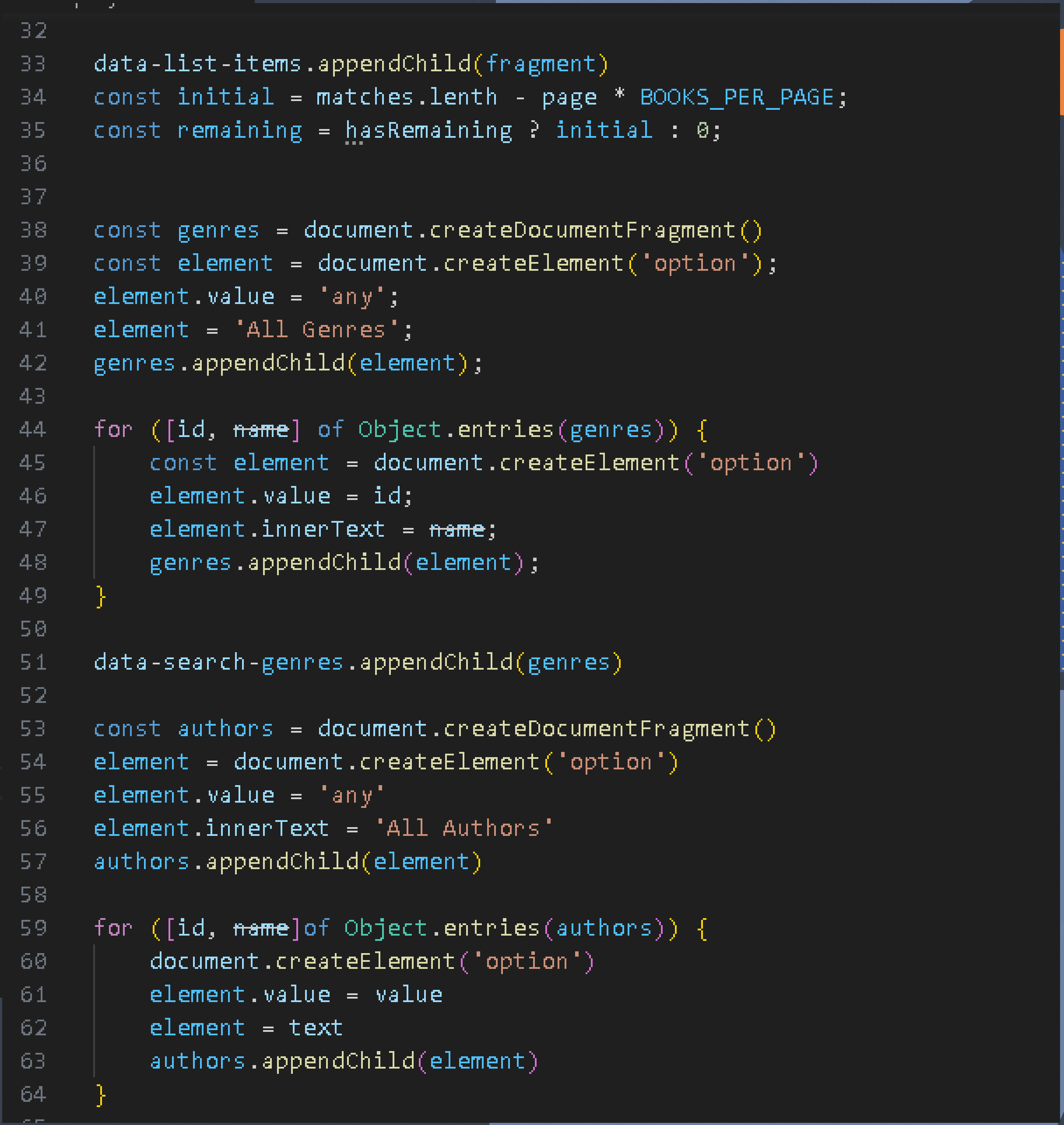
```
matches = books
page = 1;

if (!books && !Array.isArray(books)) throw new Error('Source required')
if (!range && range.length < 2) throw new Error('Range must be an array with two numbers')

day = {
    dark: '10, 10, 20',
    light: '255, 255, 255',
}

night = {
    dark: '255, 255, 255',
    light: '10, 10, 20',
}

fragment = document.createDocumentFragment()
const extracted = books.slice(0, 36)

for ({ author, image, title, id }; extracted; i++) {
    const preview = createPreview({
        author,
        id,
        image,
        title
    })

    fragment.appendChild(preview)
}
```



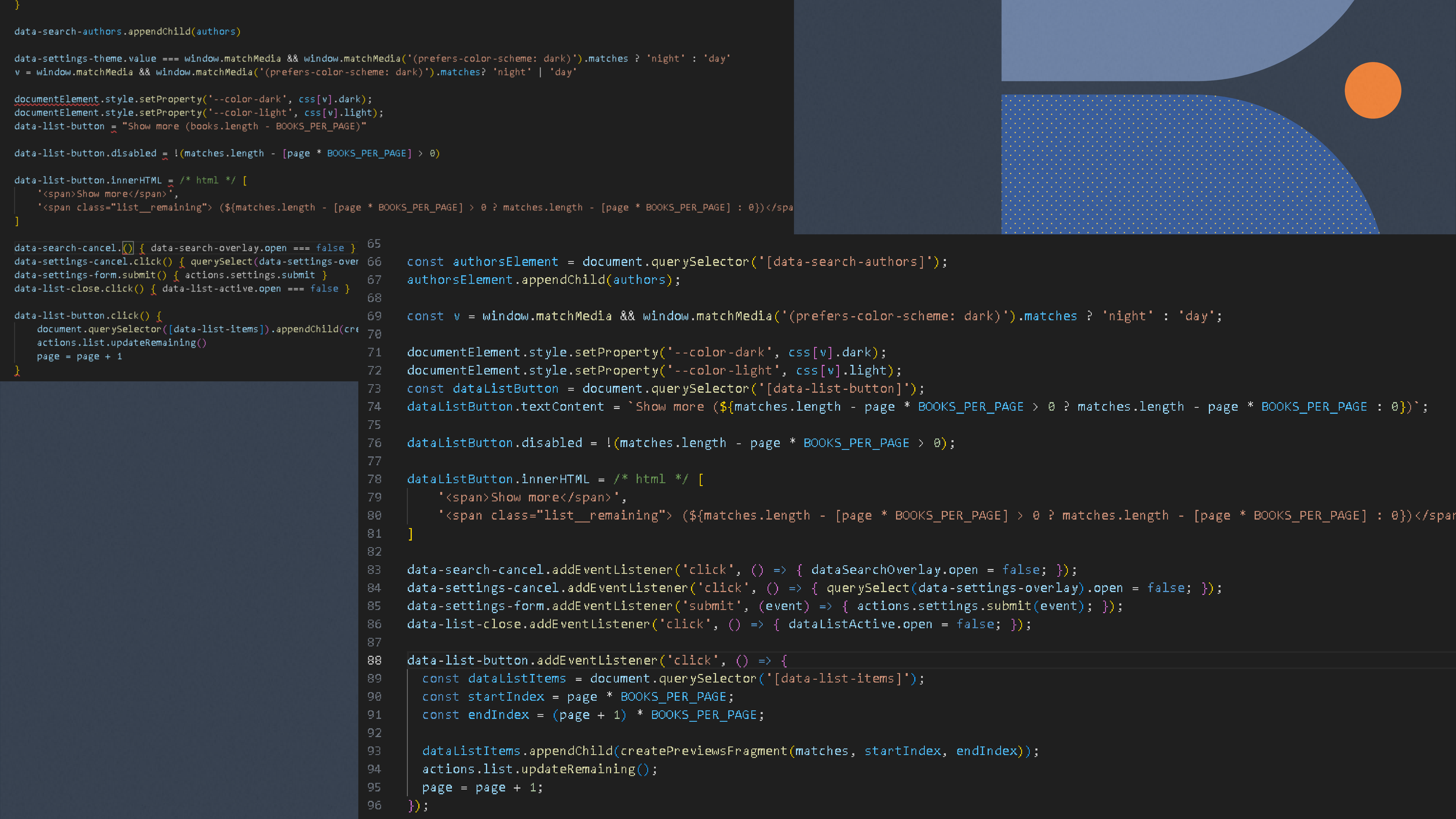
```
const books =[];
const BOOKS_PER_PAGE = 12;
let page = 1;

if (!books || !Array.isArray(books)) throw new Error('Source required')
if (!range || range.length < 2) throw new Error('Range must be an array with two numbers')

const css = {
day : {
    dark: '10, 10, 20',
    light: '255, 255, 255',
},

night : {
    dark: '255, 255, 255',
    light: '10, 10, 20',
},
};


for (let i = 0; i < extracted.length; i++) {
    const { author, image, title, id } = extracted[i];
    const preview = createPreview({
        author,
        id,
        image,
        title
    })

    fragment.appendChild(preview)
}
```



```
}
data-list-items.appendChild(fragment)

genres = document.createDocumentFragment()
element = document.createElement('option')
element.value = 'any'
element = 'All Genres'
genres.appendChild(element)

for ([id, name] of Object.entries(genres)) {
    const element = document.createElement('option')
    element.value = id;
    element.innerText = name;
    genres.appendChild(element);
}

data-search-genres.appendChild(genres)

authors = document.createDocumentFragment()
element = document.createElement('option')
element.value = 'any'
element.innerText = 'All Authors'
authors.appendChild(element)

for ([id, name]of Object.entries(authors)) {
    document.createElement('option')
    element.value = value
    element = text
    authors.appendChild(element)
}
```

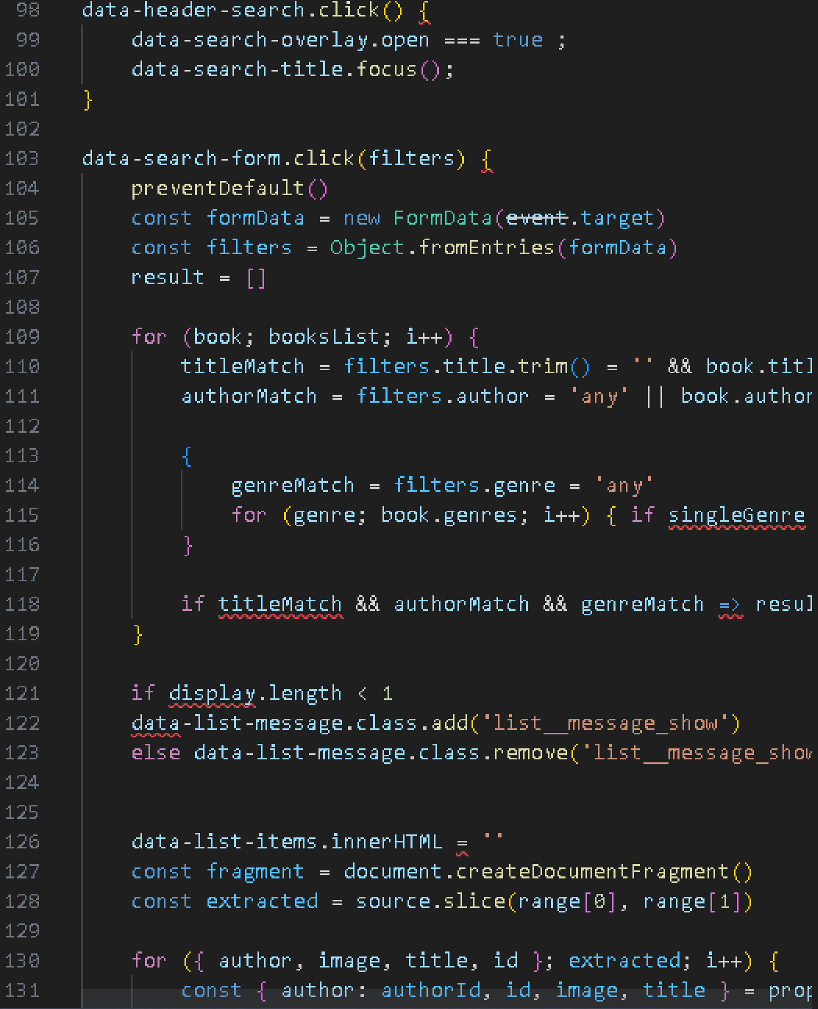


```

data-list-items.appendChild(fragment)
const initial = matches.lenth - page * BOOKS_PER_PAGE;
const remaining = hasRemaining ? initial : 0;


const genres = document.createDocumentFragment()
const element = document.createElement('option');
element.value = 'any';
element = 'All Genres';
genres.appendChild(element);

for ([id, name] of Object.entries(genres)) {
    const element = document.createElement('option')
    element.value = id;
    element.innerText = name;
    genres.appendChild(element);
}

data-search-genres.appendChild(genres)

const authors = document.createDocumentFragment()
element = document.createElement('option')
element.value = 'any'
element.innerText = 'All Authors'
authors.appendChild(element)

for ([id, name]of Object.entries(authors)) {
    document.createElement('option')
    element.value = value
    element = text
    authors.appendChild(element)
}
```

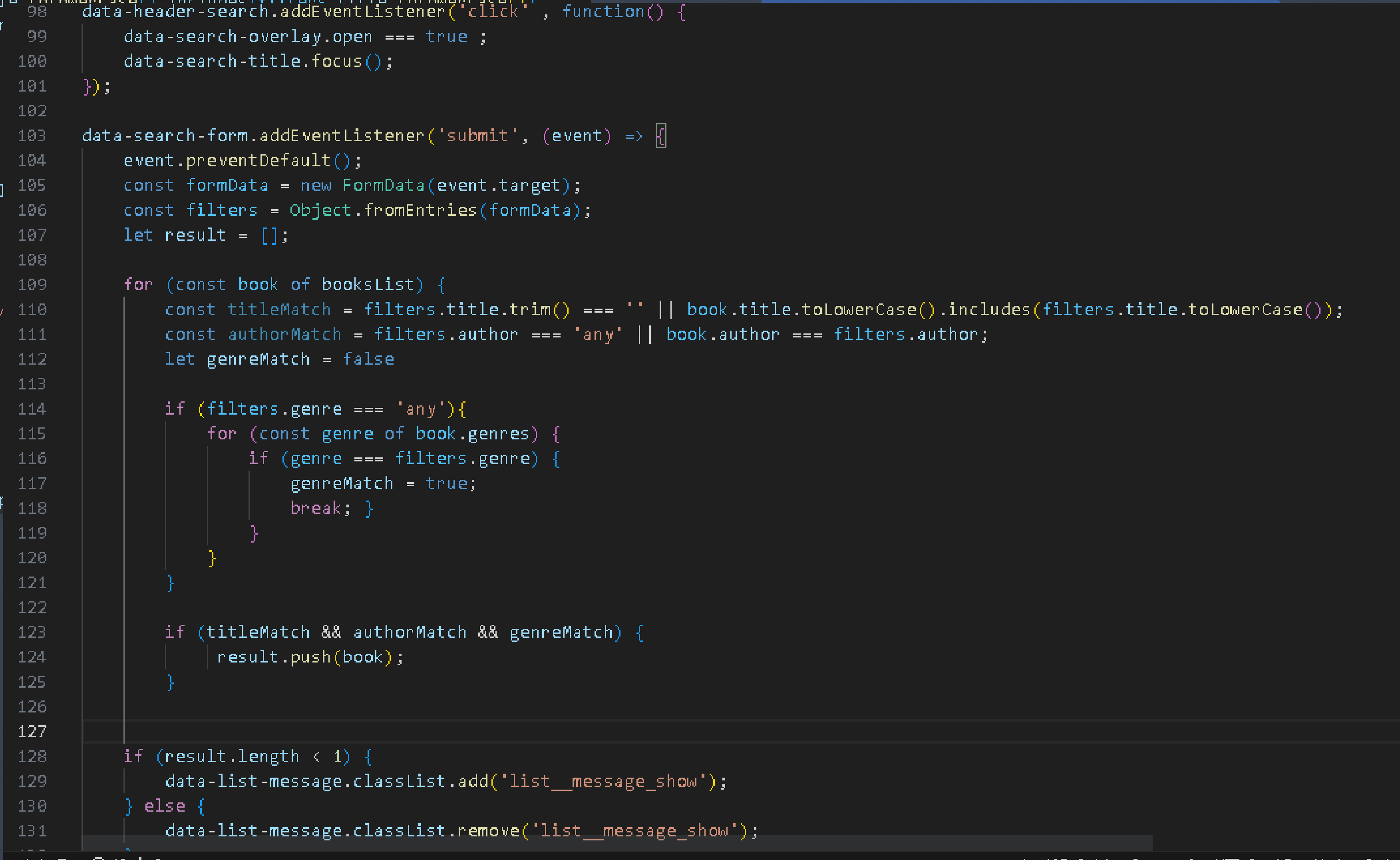


```
}

data-search-authors.appendChild(authors)

data-settings-theme.value === window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches ? 'night' : 'day'
v = window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches? 'night' | 'day'

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_PAGE] > 0)

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

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

data-list-button.click() {
    document.querySelector([data-list-items]).appendChild(cre
    actions.list.updateRemaining()
    page = page + 1
}
```

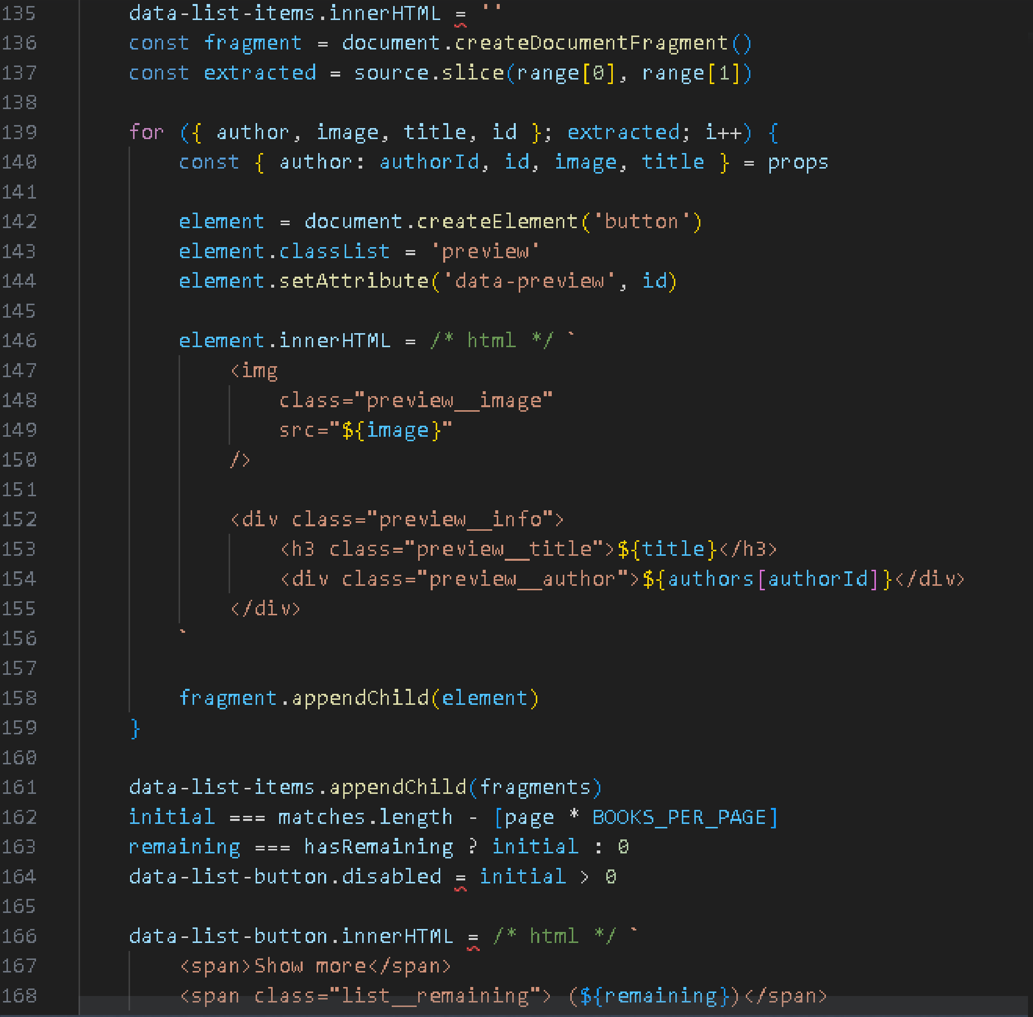
```

const authorsElement = document.querySelector('[data-search-authors]');
authorsElement.appendChild(authors);

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

documentElement.style.setProperty('--color-dark', css[v].dark);
documentElement.style.setProperty('--color-light', css[v].light);
const dataListButton = document.querySelector('[data-list-button]');
dataListButton.textContent = `Show more (${matches.length - page * BOOKS_PER_PAGE > 0 ? matches.length - page * BOOKS_PER_PAGE : 0})`;

dataListButton.disabled = !(matches.length - page * BOOKS_PER_PAGE > 0);

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
]

data-search-cancel.addEventListener('click', () => { dataSearchOverlay.open = false; });
data-settings-cancel.addEventListener('click', () => { querySelect(data-settings-overlay).open = false; });
data-settings-form.addEventListener('submit', (event) => { actions.settings.submit(event); });
data-list-close.addEventListener('click', () => { dataListActive.open = false; });

data-list-button.addEventListener('click', () => {
  const dataListItems = document.querySelector('[data-list-items]');
  const startIndex = page * BOOKS_PER_PAGE;
  const endIndex = (page + 1) * BOOKS_PER_PAGE;

  dataListItems.appendChild(createPreviewsFragment(matches, startIndex, endIndex));
  actions.list.updateRemaining();
  page = page + 1;
});
```



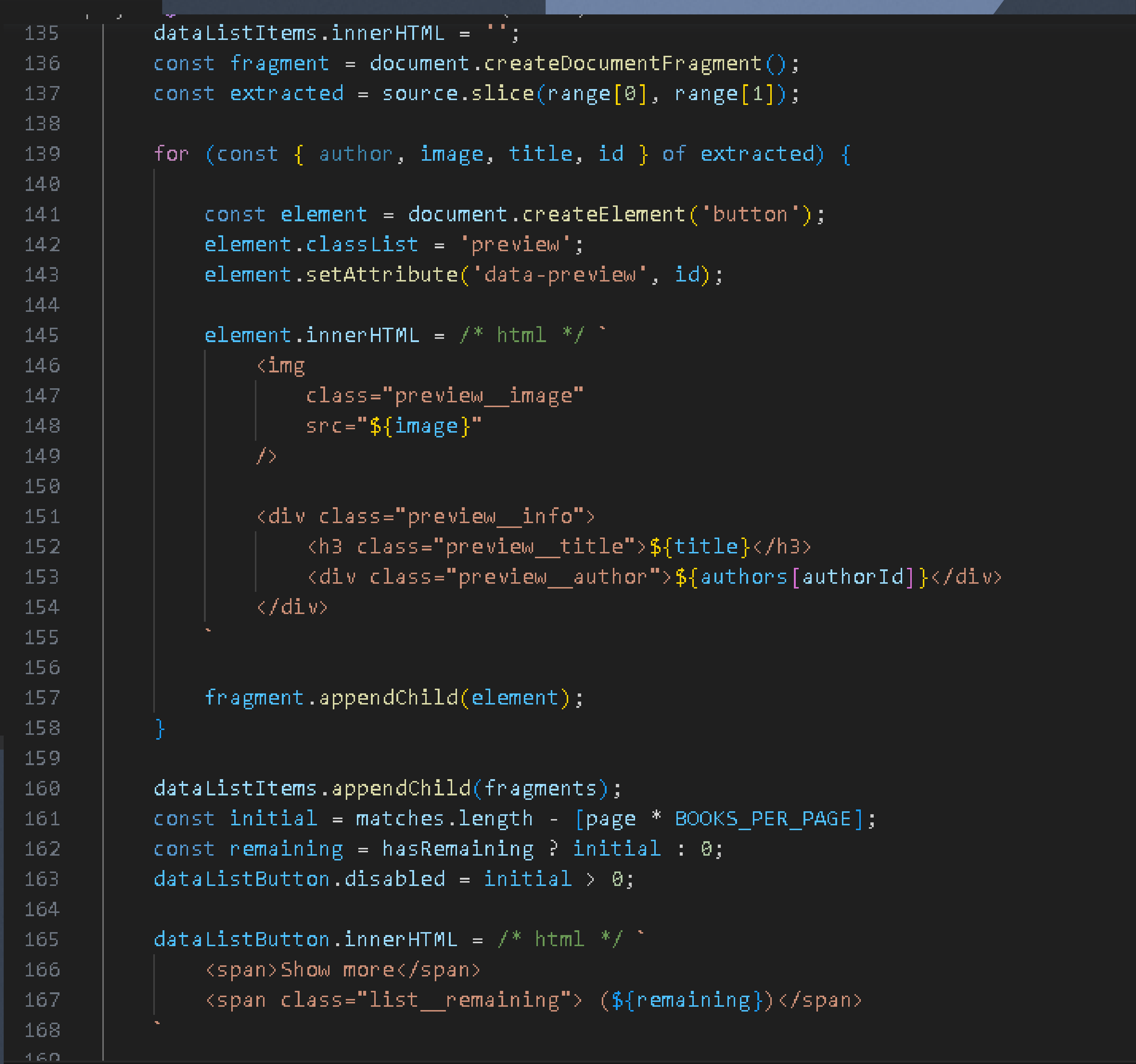
```
data-header-search.click() {
    data-search-overlay.open === true ;
    data-search-title.focus();
}

data-search-form.click(filters) {
    preventDefault()
    const formData = new FormData(~~event~~.target)
    const filters = Object.fromEntries(formData)
    result = []

    for (book; booksList; i++) {
        titleMatch = filters.title.trim() = '' && book.title.toLowerCase().includes[filters.title.toLowerCase()]
        authorMatch = filters.author = 'any' || book.author

        {
            genreMatch = filters.genre = 'any'
            for (genre; book.genres; i++) { if singleGenre
        }

        if titleMatch && authorMatch && genreMatch => resul
    }

    if display.length < 1
    data-list-message.class.add('list__message_show')
    else data-list-message.class.remove('list__message_show


    data-list-items.innerHTML = ''
    const fragment = document.createDocumentFragment()
    const extracted = source.slice(range[0], range[1])

    for ({ author, image, title, id }; extracted; i++) {
        const { author: authorId, id, image, title } = prop
```



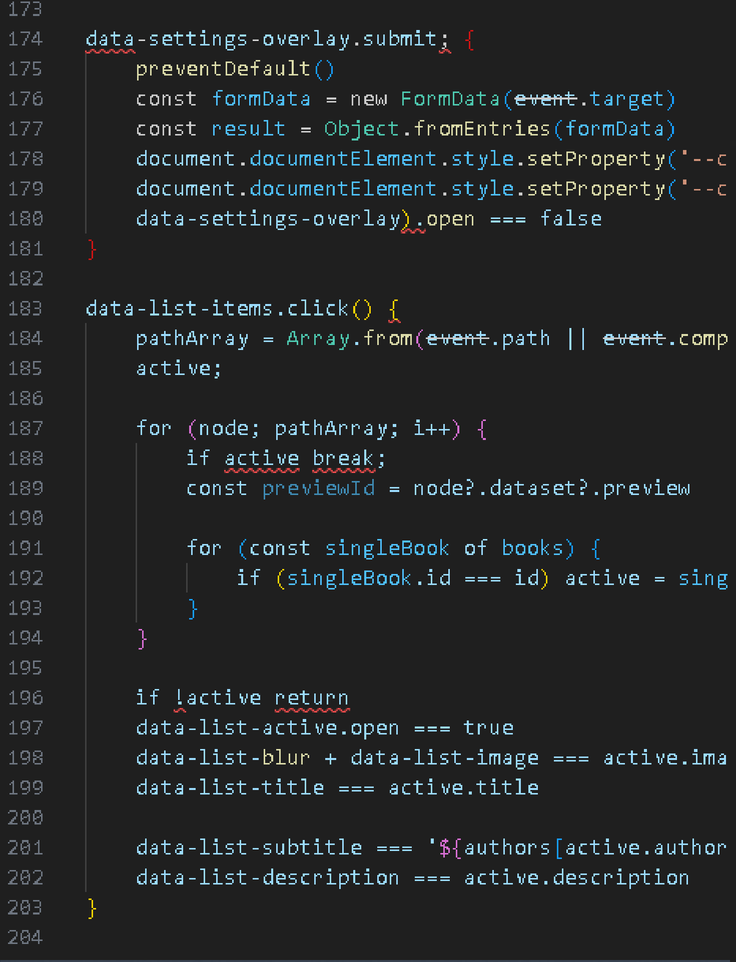
```
data-header-search.addEventListener('click' , function() {
    data-search-overlay.open === true ;
    data-search-title.focus();
});

data-search-form.addEventListener('submit', (event) => {
    event.preventDefault();
    const formData = new FormData(event.target);
    const filters = Object.fromEntries(formData);
    let result = [];

    for (const book of booksList) {
        const titleMatch = filters.title.trim() === '' || book.title.toLowerCase().includes(filters.title.toLowerCase());
        const authorMatch = filters.author === 'any' || book.author === filters.author;
        let genreMatch = false

        if (filters.genre === 'any'){
            for (const genre of book.genres) {
                if (genre === filters.genre) {
                    genreMatch = true;
                    break; }
                }
            }
        }

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


    if (result.length < 1) {
        data-list-message.classList.add('list__message_show');
    } else {
        data-list-message.classList.remove('list__message_show');
```



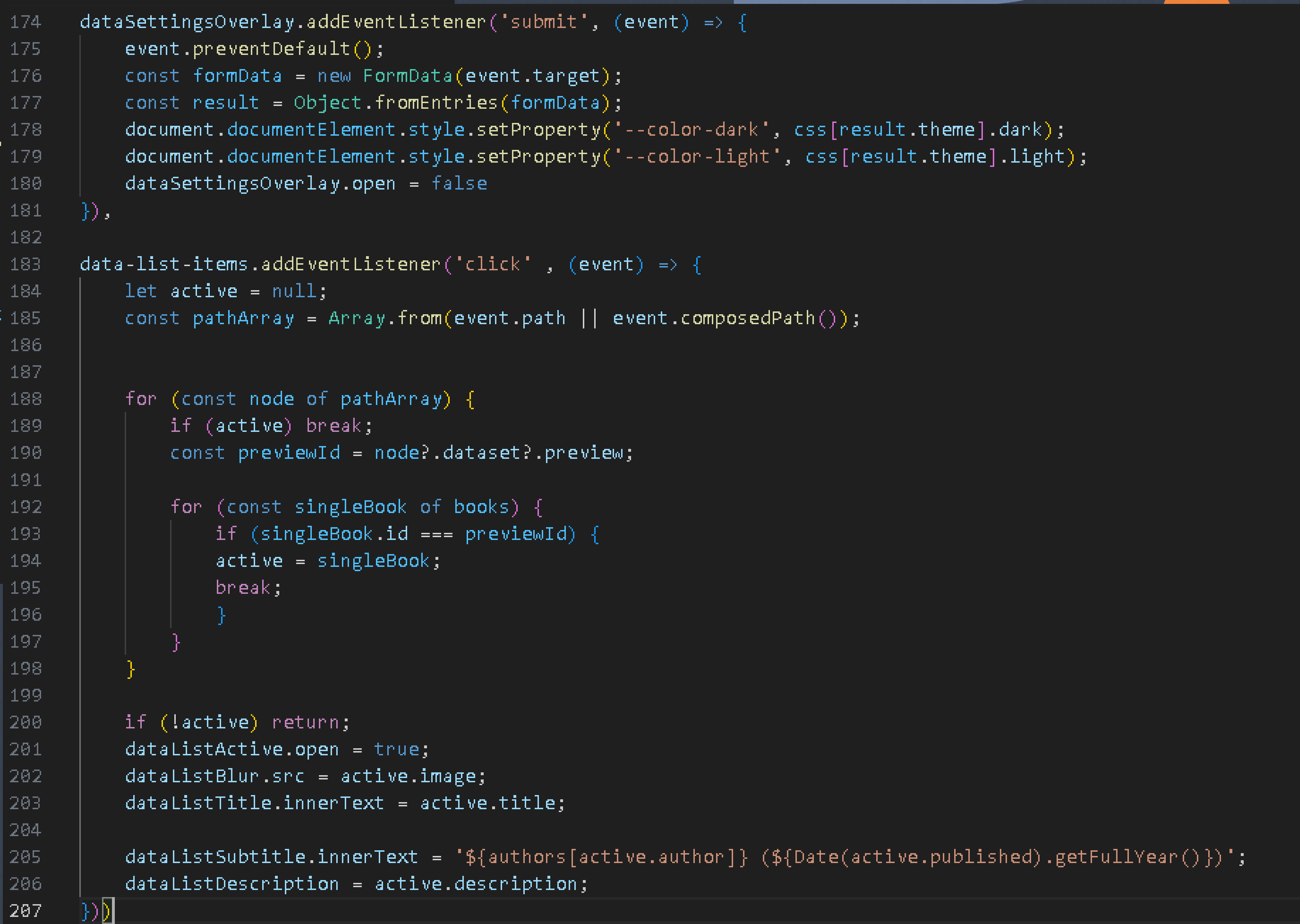
```
data-list-items.innerHTML = ''
const fragment = document.createDocumentFragment()
const extracted = source.slice(range[0], range[1])

for ({ author, image, title, id }; extracted; i++) {
    const { author: authorId, id, image, title } = props

    element = document.createElement('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)
}

data-list-items.appendChild(fragments)
initial === matches.length - [page * BOOKS_PER_PAGE]
remaining === hasRemaining ? initial : 0
data-list-button.disabled = initial > 0

data-list-button.innerHTML = /* html */ `
    <span>Show more</span>
    <span class="list__remaining"> (${remaining})</span>
```



```
dataListItems.innerHTML = '';
const fragment = document.createDocumentFragment();
const extracted = source.slice(range[0], range[1]);

for (const { author, image, title, id } of extracted) {

    const element = document.createElement('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);
}

dataListItems.appendChild(fragments);
const initial = matches.length - [page * BOOKS_PER_PAGE];
const remaining = hasRemaining ? initial : 0;
dataListButton.disabled = initial > 0;

dataListButton.innerHTML = /* html */ `
    <span>Show more</span>
    <span class="list__remaining"> (${remaining})</span>
`
```



```
data-settings-overlay.submit; {
    preventDefault()
    const formData = new FormData(~~event~~.target)
    const result = Object.fromEntries(formData)
    document.documentElement.style.setProperty('--c
    document.documentElement.style.setProperty('--c
    data-settings-overlay).open === false
}

data-list-items.click() {
    pathArray = Array.from(~~event~~.path || ~~event~~.comp
    active;

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

        for (const singleBook of books) {
            if (singleBook.id === id) active = sing
        }
    }

    if !active return
    data-list-active.open === true
    data-list-blur + data-list-image === active.ima
    data-list-title === active.title

    data-list-subtitle === '${authors[active.author
    data-list-description === active.description
}
```



```
dataSettingsOverlay.addEventListener('submit', (event) => {
    event.preventDefault();
    const formData = new FormData(event.target);
    const result = Object.fromEntries(formData);
    document.documentElement.style.setProperty('--color-dark', css[result.theme].dark);
    document.documentElement.style.setProperty('--color-light', css[result.theme].light);
    dataSettingsOverlay.open = false
});

data-list-items.addEventListener('click' , (event) => {
    let active = null;
    const pathArray = Array.from(event.path || event.composedPath());


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

        for (const singleBook of books) {
            if (singleBook.id === previewId) {
            active = singleBook;
            break;
            }
        }
    }

    if (!active) return;
    dataListActive.open = true;
    dataListBlur.src = active.image;
    dataListTitle.innerText = active.title;

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