DWA_07.4 Knowledge Check_DWA7

---

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

```
/**
 * Filters books based on user input, updates the list of filtered books, and displays them.
 *
 * @param {Event} event - The event object representing the form submission event.
 */
const filterAndDisplayBooks = event => {
  try {
    const result = getFilteredBooks(event);
    if (!result) throw new Error('There is something wrong with your result array');
    displayFilteredBooks(result);
    html.search.form.reset();
  } catch (e) {
    /* eslint no-console: ["error", { allow: ["warn", "error"] }] */
    console.error(e);
  }
};
```

Easy and concise for the developer to understand exactly what is happening and pass the result of the getFilteredBooks() to the following displayFilteredBooks().

```
/**
 * Creates a book element with its unique information.
 *
 * @param {Object} book
 * @param {string} book.author - The author of the book.
 * @param {string} book.id - The ID of the book.
 * @param {string} book.image - The image URL of the book.
 * @param {string} book.title - The title of the book.
 * @returns {HTMLButtonElement} The created book element
 */
export const createBookElement = ({ author, id, image, title }) => {
  const element = document.createElement('button');
  element.classList = 'preview';
  element.setAttribute('data-preview', id);

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

        <div class="preview__info">
            <h3 class="preview__title">${title}</h3>
            <div class="preview__author">${authors[author]}</div>
        </div>
    `;
  return element;
};
```

Created an easy template that can generate a book element whenever you need one.

```
export const html = {
  list: {
    button: document.querySelector('[data-list-button]'),
    message: document.querySelector('[data-list-message]'),
    overlay: document.querySelector('[data-list-active]'),
    close: document.querySelector('[data-list-close]'),
    items: document.querySelector('[data-list-items]'),
    blur: document.querySelector('[data-list-blur]'),
    image: document.querySelector('[data-list-image]'),
    title: document.querySelector('[data-list-title]'),
    subtitle: document.querySelector('[data-list-subtitle]'),
    description: document.querySelector('[data-list-description]'),
  },
  search: {
    button: document.querySelector('[data-header-search]'),
    overlay: document.querySelector('[data-search-overlay]'),
    cancel: document.querySelector('[data-search-cancel]'),
    form: document.querySelector('[data-search-form]'),
    authors: document.querySelector('[data-search-authors]'),
    genres: document.querySelector('[data-search-genres]'),
    title: document.querySelector('[data-search-title]'),
  },
  settings: {
    button: document.querySelector('[data-header-settings]'),
    overlay: document.querySelector('[data-settings-overlay]'),
    cancel: document.querySelector('[data-settings-cancel]'),
    form: document.querySelector('[data-settings-form]'),
    theme: document.querySelector('[data-settings-theme]'),
  },
};
```

Grouping all queries together by using Object literal keeps code more structured and maintainable.

---

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

```
/**
 * Checks if a given book matches the specified criteria.
 *
 * @param {object} book - The book object to be matched.
 * @param {object} filters - The criteria filters to match against.
 * @param {string} filters.genre - The genre to match. Set to 'any' to match any genre.
 * @param {string} filters.title - The title to match. Case-insensitive. Set to an empty string to ignore title matching.
 * @param {string} filters.author - The author to match. Set to 'any' to match any author.
 * @returns {boolean} Returns true if the book matches all specified criteria, otherwise false.
 */
const isBookMatch = (book, { genre, title, author }) => {
  const genreMatch = book.genres.includes(genre) || genre === 'any';
  if (!genreMatch) return false;
  const titleMatch = title.trim() === '' || book.title.toLowerCase().includes(title.toLowerCase());
  if (!titleMatch) return false;
  const authorMatch = author === 'any' || book.author === author;
  if (!authorMatch) return false;
  return true;
};
```

There could be a better way of writing this in a cleaner manner and easier for the developer to process.

```
/**
 * Finds the clicked book element in the event's path and displays its information in a modal.
 *
 * @param {Event} event - The event object representing the click event.
 */
export const getClickedBookAndDisplayModal = event => {
  const pathArray = Array.from(event.path || event.composedPath());
  let active = null;

  for (const node of pathArray) {
    if (active) break;

    if (node?.dataset?.preview) {
      let result = null;

      for (const singleBook of books) {
        if (result) break;
        if (singleBook.id === node?.dataset?.preview) result = singleBook;
      }

      active = result;
    }
  }
```

Not as well structured as my other code.

```
/**
 * Displays the filtered books on the page.
 * @param {Array} result - An array of book objects representing the filtered books.
 */
const displayFilteredBooks = result => {
  page = 1;
  matches = result;

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

  html.list.items.innerHTML = '';
  const newItems = document.createDocumentFragment();

  result.slice(0, BOOKS_PER_PAGE).forEach(book => newItems.appendChild(createBookElement(book)));
  html.list.items.appendChild(newItems);

  updateShowMoreButton();

  window.scrollTo({ top: 0, behavior: 'smooth' });
  html.search.overlay.open = false;
};
```

This could be abstracted further to make it more reusable.

---

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

Find ways to make it universally reusable in other components.
Ensure that each function has a single responsibility.
The code should be open for extension but closed for modification.
