Task 3: Implementing Search Functionality for Profile Page

Objective:

Add a search field to the profile page, develop a search function to filter profiles based on the search query, and update the UI to display only the matching profiles.

Pre-requisites:

- Basic understanding of HTML, CSS, and JavaScript
- Familiarity with a code editor like Visual Studio Code

Concepts Covered:

- Adding a Search Field using JavaScript
- Developing a Search Function to Filter Results
- Updating the UI with Search Results

Concepts:

1. **Adding a Search Field using JavaScript:**

   Dynamically create an input field and a search button. Attach the input field and button to a search bar `div` .

```
const searchBar = document.createElement('div');
searchBar.classList.add('search-bar');

const searchInput = document.createElement('input');
searchInput.type = 'text';
searchInput.placeholder = 'Search profiles...';
searchBar.appendChild(searchInput);

const searchButton = document.createElement('button');
searchButton.innerText = 'Search';
searchBar.appendChild(searchButton);

document.body.insertBefore(searchBar, document.querySelector('.container'));
```

2. **Developing a Search Function to Filter Results:**

   Attach an event listener to the search button or input field (`keyup` event). In the event handler, filter your profile array based on the input field value.

```
searchButton.addEventListener('click', filterProfiles);
searchInput.addEventListener('keyup', filterProfiles);

function filterProfiles() {
    const query = searchInput.value.toLowerCase();
    const filteredProfiles = profiles.filter(profile =>
        profile.name.toLowerCase().includes(query) ||
        profile.bio.toLowerCase().includes(query) ||
        profile.skills.some(skill => skill.toLowerCase().includes(query))
    );
    displayProfiles(filteredProfiles);
}
```

3. **Updating the UI with Search Results:**

   Clear the current display of cards. Use the filtered array to display only the items that match the search query.

```
function displayProfiles(profiles) {
    const container = document.querySelector('.container');
    container.innerHTML = ''; // Clear current profiles

    profiles.forEach(profile => {
        const card = document.createElement('div');
        card.classList.add('card');

        const nameElement = document.createElement('h2');
        nameElement.innerText = profile.name;
        card.appendChild(nameElement);

        const bioElement = document.createElement('p');
        bioElement.innerText = profile.bio;
        card.appendChild(bioElement);

        const skillsElement = document.createElement('ul');
        profile.skills.forEach(skill => {
            const skillItem = document.createElement('li');
            skillItem.innerText = skill;
            skillsElement.appendChild(skillItem);
        });
        card.appendChild(skillsElement);

        container.appendChild(card);
    });
}

// Initial display of all profiles
displayProfiles(profiles);
```

Setup:

1. **Install Visual Studio Code (VS Code):**

   Download and install VS Code from Visual Studio Code.

2. **Web Browsers:**

   Use Google Chrome or Mozilla Firefox for viewing your webpage and utilizing their developer tools for debugging.

Tasks:

1. **Add a Search Field to the HTML using JavaScript (10 minutes):**
   - Dynamically create an input field and a search button.
   - Attach the input field and button to a search bar `div`.
   - Example:

   ```
   const searchBar = document.createElement('div');
   searchBar.classList.add('search-bar');

   const searchInput = document.createElement('input');
   searchInput.type = 'text';
   searchInput.placeholder = 'Search profiles...';
   searchBar.appendChild(searchInput);

   const searchButton = document.createElement('button');
   searchButton.innerText = 'Search';
   searchBar.appendChild(searchButton);

   document.body.insertBefore(searchBar, document.querySelector('.container'));
   ```

2. **Develop a Search Function to Filter Results (10 minutes):**
   - Attach an event listener to the search button or input field (`keyup` event).
   - In the event handler, filter your profile array based on the input field value.
   - Example:

   ```
   searchButton.addEventListener('click', filterProfiles);
   searchInput.addEventListener('keyup', filterProfiles);

   function filterProfiles() {
       const query = searchInput.value.toLowerCase();
       const filteredProfiles = profiles.filter(profile =>
           profile.name.toLowerCase().includes(query) ||
           profile.bio.toLowerCase().includes(query) ||
           profile.skills.some(skill => skill.toLowerCase().includes(query))
       );
       displayProfiles(filteredProfiles);
   }
   ```

3. **Update the UI with Search Results (10 minutes):**

  - Clear the current display of cards.
  - Use the filtered array to display only the items that match the search query.
  - Example:

```
function displayProfiles(profiles) {
    const container = document.querySelector('.container');
    container.innerHTML = ''; // Clear current profiles

    profiles.forEach(profile => {
        const card = document.createElement('div');
        card.classList.add('card');

        const nameElement = document.createElement('h2');
        nameElement.innerText = profile.name;
        card.appendChild(nameElement);

        const bioElement = document.createElement('p');
        bioElement.innerText = profile.bio;
        card.appendChild(bioElement);

        const skillsElement = document.createElement('ul');
        profile.skills.forEach(skill => {
            const skillItem = document.createElement('li');
            skillItem.innerText = skill;
            skillsElement.appendChild(skillItem);
        });
        card.appendChild(skillsElement);

        container.appendChild(card);
    });
}

// Initial display of all profiles
displayProfiles(profiles);
```

Instructions:

1. Write the required code in `index.html` and `script.js`.
2. Open the `index.html` file in your web browser to ensure the code displays correctly.
3. Use the browser's developer tools to debug and inspect the elements.

Resources:

- **Array - JavaScript | MDN**
  The Array object, as with arrays in other programming languages, enables storing a collecti...
  https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array

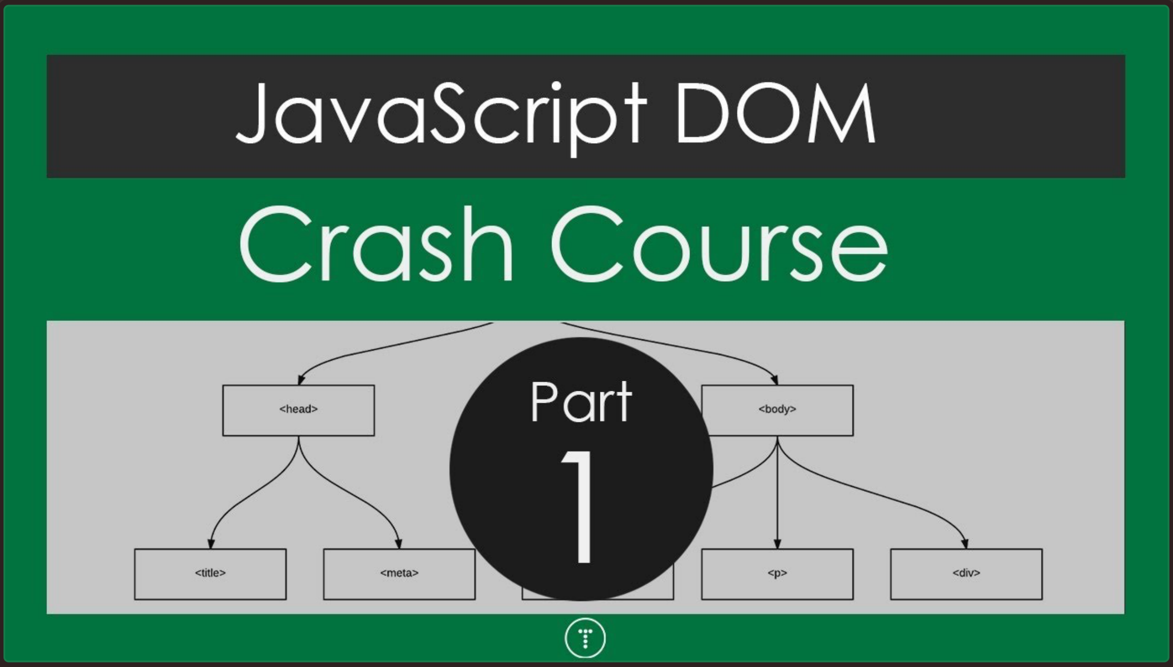
-

  Introduction to the DOM - Web APIs | MDN
  The Document Object Model (DOM) is the data representation of the objects that compris...
  https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction

-

  Documentation for Visual Studio Code
  Find out how to set-up and get the most from Visual Studio Code. Optimized for building and d...
  https://code.visualstudio.com/docs

Videos:

- 

  Traversy Media 39:01min 1,198,799 Views 22,283 Likes

GitHub Instructions:

1. **Open in Visual Studio Code:**

   After clicking on the "Open in Visual Studio Code" button from the GitHub Classroom confirmation page, VSCode will open the repository directly. If prompted, select "Open" or "Allow" to open the repository in VSCode.

2. **Open the Terminal in VSCode:**

   In VSCode, open a terminal by selecting Terminal > New Terminal from the top menu.

3. **Complete the Task:**

   In VSCode, write your solution in the `index.html` and `script.js` files.

4. **Run and Test Your Code:**

   Open your `index.html` file in a web browser to ensure it works correctly. Use the following command:

```
open index.html
```

5. **Commit Your Changes:**

   In the VSCode terminal, add your changes to git:

   ```
   git add index.html script.js
   ```

   Commit your changes with a meaningful message:

   ```
   git commit -m "Completed task 14"
   ```

6. **Push Your Changes to Your Repository:**

   Push your changes to your forked repository:

   ```
   git push origin main
   ```

7. **Create a Pull Request:**

   Go to your repository on GitHub.

   Click on the "Pull Requests" tab.

   Click the "New Pull Request" button.

   Ensure the base repository is the original template repository and the base branch is `main` .

   Ensure the head repository is your forked repository and the compare branch is `main` .

   Click "Create Pull Request".

   Add a title and description for your pull request and submit it.

Summary of Commands:

```
# Open in Visual Studio Code

# Open terminal in VSCode

# Complete the task by editing index.html and script.js

# Navigate to the directory containing index.html
cd path/to/your/index.html

# Run your code
open index.html

# Add, commit, and push your changes
git add index.html script.js
git commit -m "Completed task 3"
```

```
git push origin main

# Create a pull request on GitHub
```