

## Task 5: Form Handling and Validation for Your Profile Page

### Objective:

Build a user input form with various fields, add validation functions for each input field, and provide feedback to the user. This task aims to enhance your skills in form handling and validation using JavaScript.

### Pre-requisites:

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

### Concepts Covered:

- Building a User Input Form
- Writing Validation Functions for Each Input Field
- Providing Feedback to the User

### Concepts:

#### 1. Building a User Input Form:

Dynamically create a form with input fields for data like name, email, and phone number. Add labels and placeholders to guide user input.

```
<form id="profileForm">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" placeholder="Enter your
name" required>

  <label for="email">Email:</label>
  <input type="email" id="email" name="email" placeholder="Enter your
email" required>

  <label for="phone">Phone Number:</label>
  <input type="tel" id="phone" name="phone" placeholder="Enter your phone
number" required>

  <button type="submit">Submit</button>
</form>
<div id="feedback"></div>
```

#### 2. Writing Validation Functions for Each Input Field:

Create JavaScript functions that validate input for each field (e.g., email regex validation).

```

function validateName(name) {
    return name.trim().length > 0;
}

function validateEmail(email) {
    const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
    return emailRegex.test(email);
}

function validatePhone(phone) {
    const phoneRegex = /^\\d{10}$/;
    return phoneRegex.test(phone);
}

```

### 3. Providing Feedback to the User:

Add event listeners to the form to handle submission, validate inputs, and provide feedback to the user.

```

document.getElementById('profileForm').addEventListener('submit', function(event)
{
    event.preventDefault();

    const name = document.getElementById('name').value;
    const email = document.getElementById('email').value;
    const phone = document.getElementById('phone').value;

    let feedbackMessage = '';
    if (!validateName(name)) {
        feedbackMessage += 'Please enter a valid name.<br>';
    }
    if (!validateEmail(email)) {
        feedbackMessage += 'Please enter a valid email.<br>';
    }
    if (!validatePhone(phone)) {
        feedbackMessage += 'Please enter a valid phone number.<br>';
    }

    if (feedbackMessage) {
        document.getElementById('feedback').innerHTML = feedbackMessage;
        document.getElementById('feedback').style.color = 'red';
    } else {
        document.getElementById('feedback').innerHTML = 'Form submitted
successfully!';
        document.getElementById('feedback').style.color = 'green';
    }
});

```

### 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. Build a User Input Form with Various Fields (10 minutes):

- Dynamically create a form with input fields for data like name, email, and phone number.
- Add labels and placeholders to guide user input.
- Example:

```
<form id="profileForm">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" placeholder="Enter your
name" required>

  <label for="email">Email:</label>
  <input type="email" id="email" name="email" placeholder="Enter your
email" required>

  <label for="phone">Phone Number:</label>
  <input type="tel" id="phone" name="phone" placeholder="Enter your phone
number" required>

  <button type="submit">Submit</button>
</form>
<div id="feedback"></div>
```

#### 2. Write Validation Functions for Each Input Field (10 minutes):

- Create JavaScript functions that validate input for each field (e.g., email regex validation).
- Example:

```
function validateName(name) {
  return name.trim().length > 0;
}

function validateEmail(email) {
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  return emailRegex.test(email);
}

function validatePhone(phone) {
  const phoneRegex = /^\d{10}$/;
  return phoneRegex.test(phone);
}
```

#### 3. Provide Feedback to the User (10 minutes):

- Add event listeners to the form to handle submission, validate inputs, and provide feedback to the user.
- Example:

```
document.getElementById('profileForm').addEventListener('submit',
function(event) {
    event.preventDefault();

    const name = document.getElementById('name').value;
    const email = document.getElementById('email').value;
    const phone = document.getElementById('phone').value;

    let feedbackMessage = '';
    if (!validateName(name)) {
        feedbackMessage += 'Please enter a valid name.<br>';
    }
    if (!validateEmail(email)) {
        feedbackMessage += 'Please enter a valid email.<br>';
    }
    if (!validatePhone(phone)) {
        feedbackMessage += 'Please enter a valid phone number.<br>';
    }


    if (feedbackMessage) {
        document.getElementById('feedback').innerHTML = feedbackMessage;
        document.getElementById('feedback').style.color = 'red';
    } else {
        document.getElementById('feedback').innerHTML = 'Form submitted successfully!';
        document.getElementById('feedback').style.color = 'green';
    }
});
```

### 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:



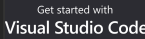

- 



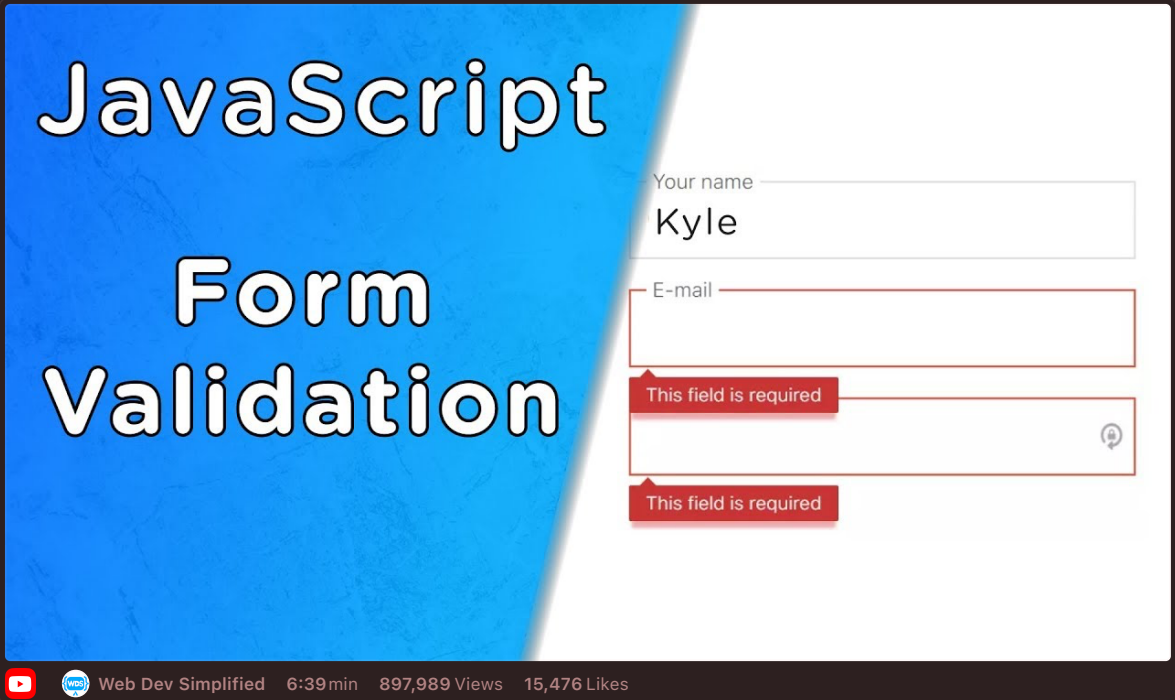
#### Regular expressions - JavaScript | MDN

Regular expressions are patterns used to match character combinations in strings. In Jav...


[https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular\\_Expressions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions)

-  **Client-side form validation - Learn web development | MDN**  
Client-side form validation sometimes requires JavaScript if you want to customize styling...  
 [https://developer.mozilla.org/en-US/docs/Learn/Forms/Form\\_validation](https://developer.mozilla.org/en-US/docs/Learn/Forms/Form_validation)
-  **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:

- 

### 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 16"
```

#### 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 5"
```

```
git push origin main
```

```
# Create a pull request on GitHub
```