JavaScript Form Validation

Introduction

Form validation ensures that user inputs meet the required criteria before submitting data to the server. There are two types of validation:

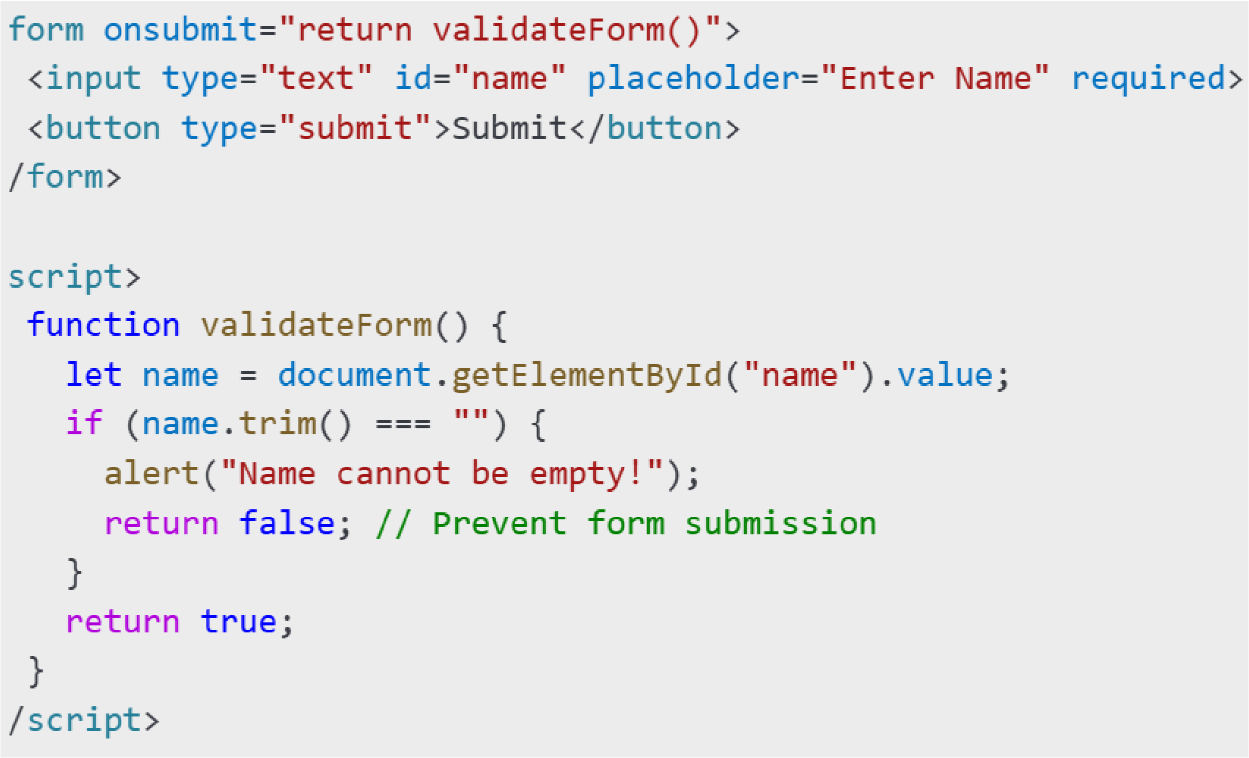
1. **Client-Side Validation (JavaScript-based)**: Checks data on the user's browser before submission.
2. **Server-Side Validation**: Verifies data on the server for security and correctness.

This lab focuses on **client-side validation** using JavaScript.

JavaScript Form Events & Validation Methods

1. onsubmit - Validate Before Submission

Description: This event is triggered when the user submits the form. We can use it to check if required fields are filled and prevent submission if validation fails.



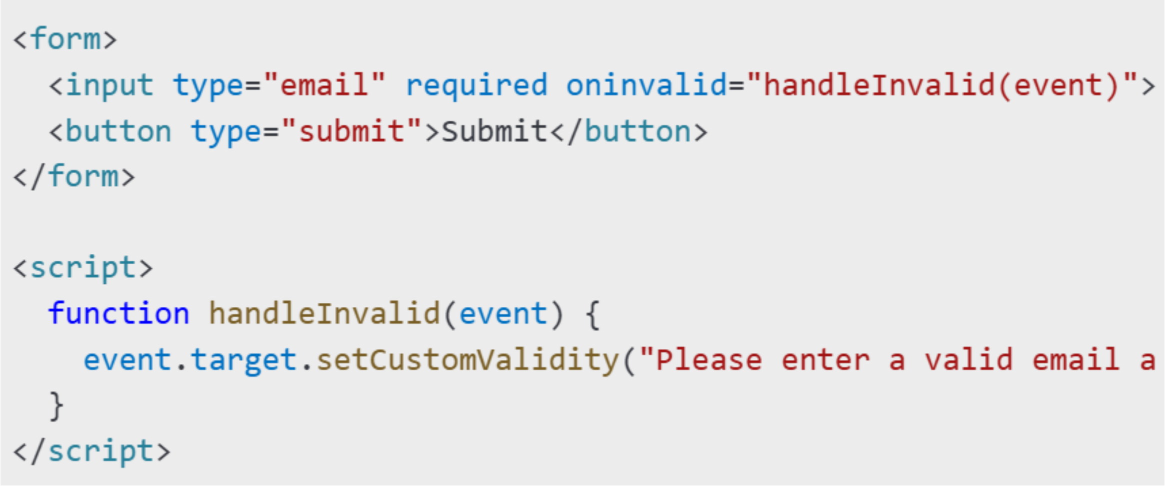
```
form onsubmit="return validateForm()">
 <input type="text" id="name" placeholder="Enter Name" required>
 <button type="submit">Submit</button>
/form>

script>
 function validateForm() {
   let name = document.getElementById("name").value;
   if (name.trim() === "") {
     alert("Name cannot be empty!");
     return false; // Prevent form submission
   }
   return true;
 }
/script>
```

Key Points:

- Runs when submit button is clicked
- Return false to block submission
- Check all form fields here
- Can show combined error messages

2. oninvalid – Custom Error Messages for Invalid Fields

Description: This event fires when a field fails the browser’s built-in validation. We can customize the error message.



```
<form>
  <input type="email" required oninvalid="handleInvalid(event)">
  <button type="submit">Submit</button>
</form>

<script>
  function handleInvalid(event) {
    event.target.setCustomValidity("Please enter a valid email a
  }
</script>
```

Key Points:

- Works with HTML5 required attribute
- Must clear with oninput to allow correction
- Better than default browser messages
- Can be styled with CSS

3. onload – Detect When the Page Loads

Description: The event fires when the page is completely loaded

```
<body onload="pageLoaded()">
  <p>The page has loaded!</p>
</body>

<script>
  function pageLoaded() {
    console.log("Page fully loaded!");
    alert("Welcome! The form is ready.");
  }
</script>
```

Key Points:

- Runs when page finishes loading
- Good for initial setup
- Can set default values
- Can focus first field

4. onkeydown & onkeyup – Detect Keystrokes

Description: These events are triggered when a key is pressed down or released.

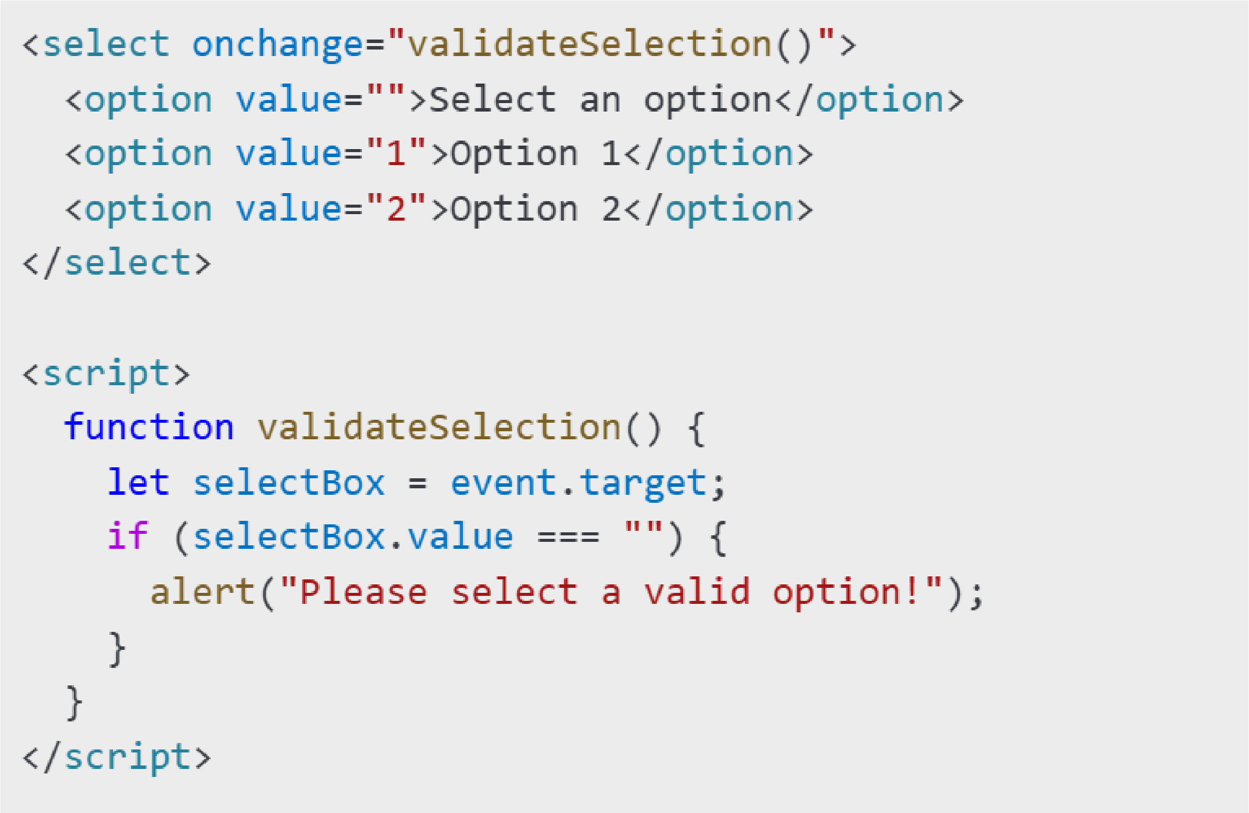
```
<input type="text" id="keyInput" placeholder="Press a key"
       onkeydown="logKey(event)" onkeyup="logKey(event)">

<script>
  function logKey(event) {
    console.log(`Key ${event.type}: ${event.key}`);
  }
</script>
```

Key Points:

- onkeydown: When key is pressed
- onkeyup: When key is released
- Access key with event.key
- Can block invalid keys

5. onchange – Validate Dropdown Selection

Description: This event occurs when the user selects a different option in a dropdown.



```
<select onchange="validateSelection()">
  <option value="">Select an option</option>
  <option value="1">Option 1</option>
  <option value="2">Option 2</option>
</select>

<script>
  function validateSelection() {
    let selectBox = event.target;
    if (selectBox.value === "") {
      alert("Please select a valid option!");
    }
  }
</script>
```

Key Points:

- Triggers after change AND blur
- Works with selects, checkboxes, radios
- Good for dependent fields
- Can show/hide fields based on selection

6. oninput – Validate Input While Typing

```
<input type="text" id="nameInput"
  placeholder="Type your name"
  oninput="validateName()">

<script>
  function validateName() {
    let input = document.getElementById("nameInput");
    if (input.value.length < 3) {
      input.style.borderColor = "red";
    } else {
      input.style.borderColor = "green";
    }
  }
</script>
```

Key Points:

- Triggers on every input change
- Provides instant feedback
- Good for format validation
- Less annoying than onsubmit-only validation

7. onfocus – Show Hints When Clicking Input

Description: This event triggers when an input field gains focus.

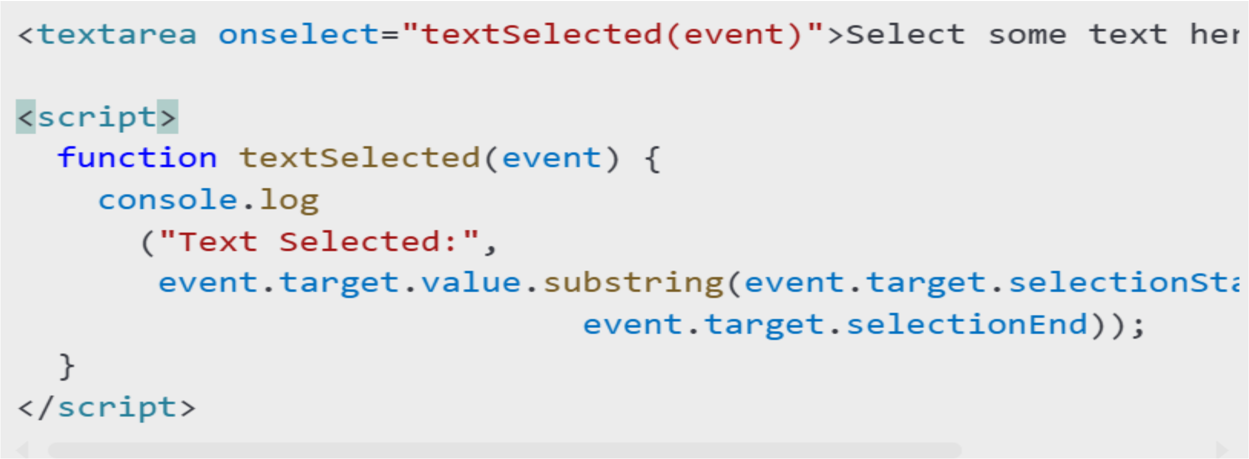
```
<input type="text" id="username"
  placeholder="Enter Username" onfocus="showHint()">
<p id="hint" style="color: gray;"></p>

<script>
  function showHint() {
    document.getElementById("hint").innerText
      = "Hint: Username must be at least 5 characters.";
  }
</script>
```

Key Points:

- Triggers when field gets focus
- Good for displaying instructions
- Often paired with onblur
- Can highlight active field

8. onselect – Detect Text Selection

Description: Triggers when the user selects text in a field.

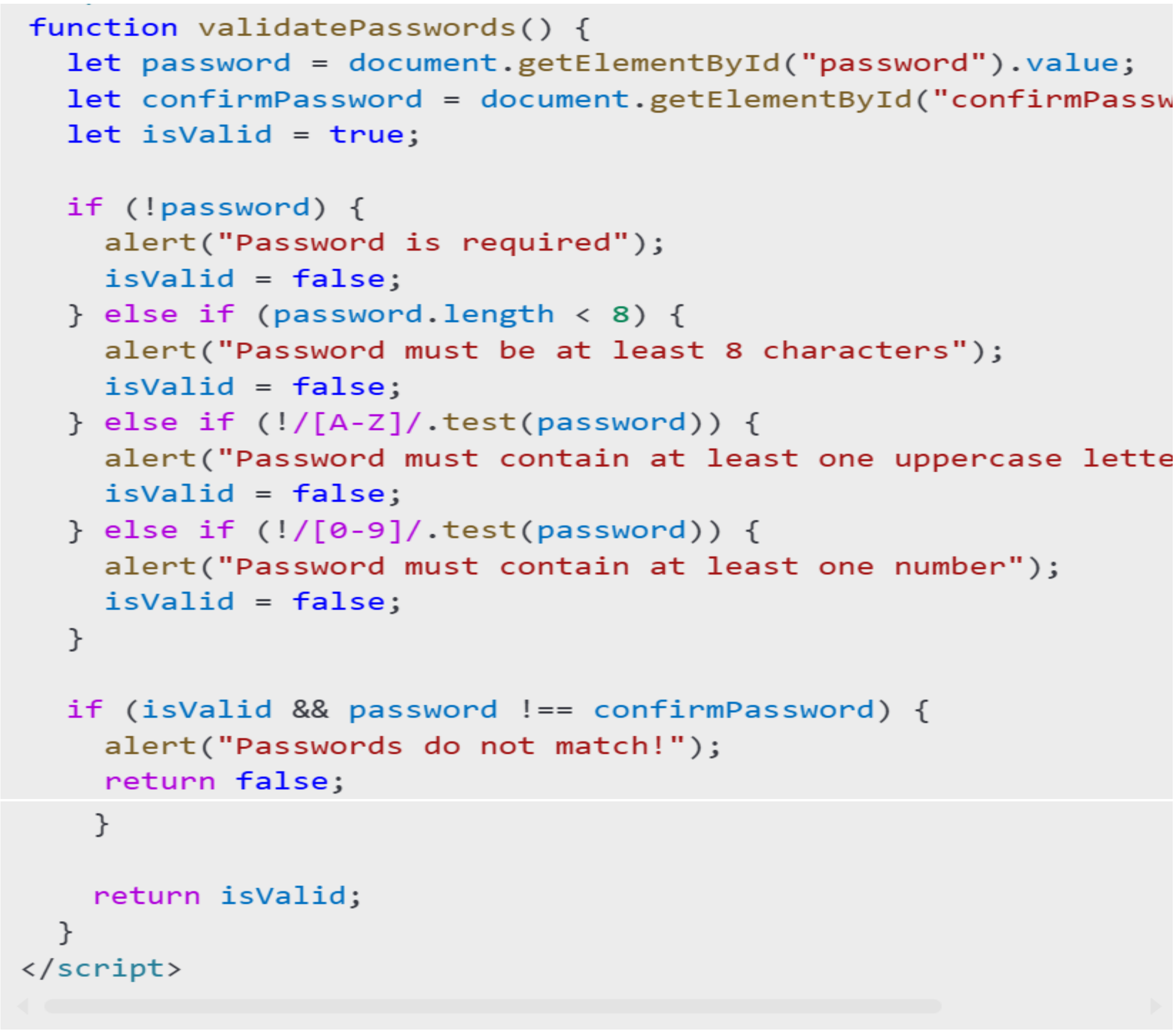


```
<textarea onselect="textSelected(event)">Select some text her

<script>
  function textSelected(event) {
    console.log
      ("Text Selected:",
       event.target.value.substring(event.target.selectionSta
                                    event.target.selectionEnd));
  }
</script>
```

Key Points:

- Triggers when text is selected
- Get selection with selectionStart/End
- Useful for text editing features
- Can enable/disable buttons based on selection

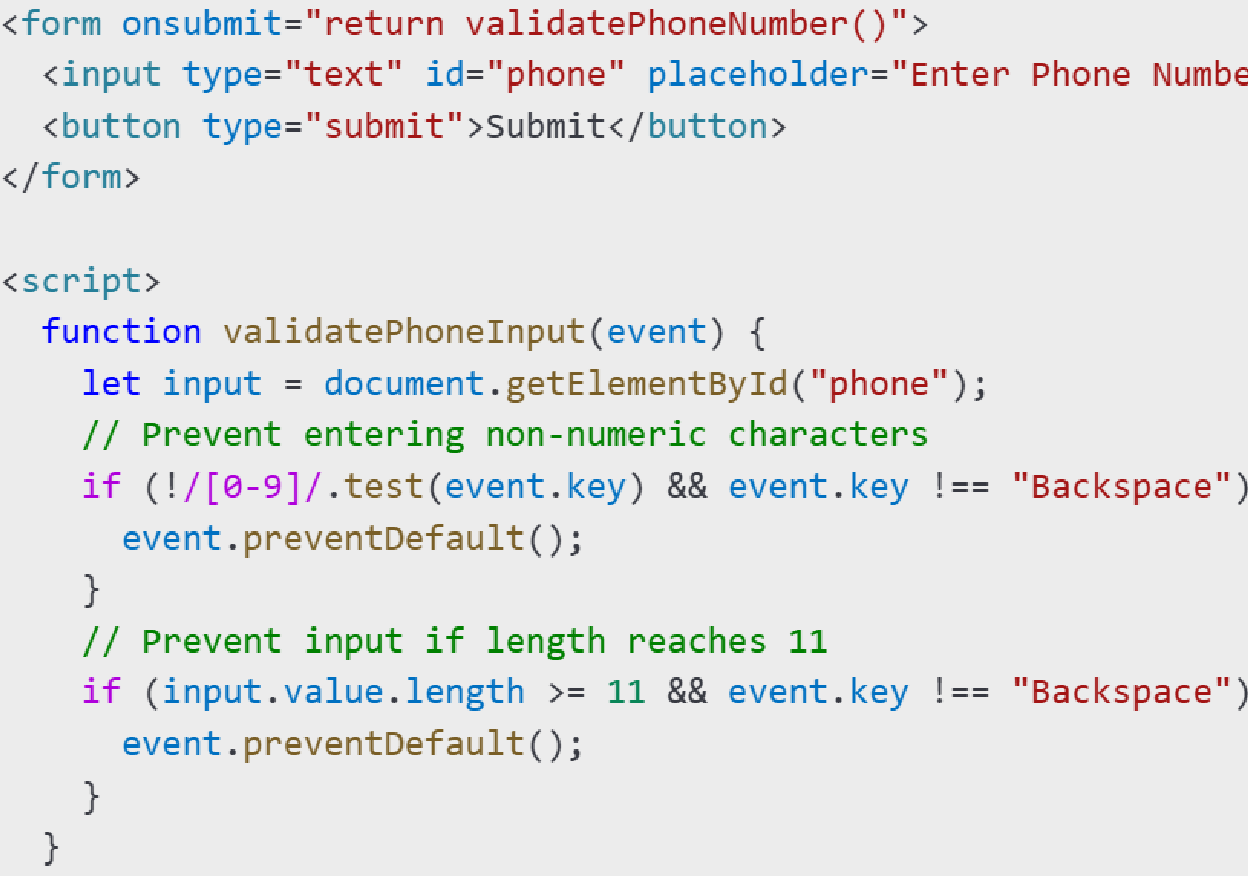
9. Password Confirmation & Strong Password Validation



```
function validatePasswords() {
  let password = document.getElementById("password").value;
  let confirmPassword = document.getElementById("confirmPassw
  let isValid = true;

  if (!password) {
    alert("Password is required");
    isValid = false;
  } else if (password.length < 8) {
    alert("Password must be at least 8 characters");
    isValid = false;
  } else if (!/[A-Z]/.test(password)) {
    alert("Password must contain at least one uppercase lette
    isValid = false;
  } else if (!/[0-9]/.test(password)) {
    alert("Password must contain at least one number");
    isValid = false;
  }

  if (isValid && password !== confirmPassword) {
    alert("Passwords do not match!");
    return false;
    }

    return isValid;
  }
</script>
```

10. Phone Number Validation

Description: Ensures that the phone number contains only digits, has exactly 11 characters, and prevents input beyond 11 characters.



```
<form onsubmit="return validatePhoneNumber()">
  <input type="text" id="phone" placeholder="Enter Phone Numbe
  <button type="submit">Submit</button>
</form>

<script>
  function validatePhoneInput(event) {
    let input = document.getElementById("phone");
    // Prevent entering non-numeric characters
    if (!/[0-9]/.test(event.key) && event.key !== "Backspace")
      event.preventDefault();
    }
    // Prevent input if length reaches 11
    if (input.value.length >= 11 && event.key !== "Backspace")
      event.preventDefault();
    }
  }
```