HTML Form Elements and JavaScript Events

Author: Anthoniraj Amalanathan

1. Introduction

HTML forms and JavaScript events are fundamental for building **interactive web applications**. HTML collects user input, while JavaScript processes, validates, and dynamically updates the webpage based on user actions.

These concepts are extensively used in:

- Online registration systems
- Login forms
- Feedback and survey forms
- Student information systems

2. HTML Form Elements

HTML form elements are used to **collect user inputs** and send them to JavaScript or a server for processing.

Commonly Used Form Elements

Element	Description
`<form>`	Container for all form elements and enables data submission
`<input>`	Accepts user input such as text, email, date, radio, file
`<select>`	Creates a dropdown list
`<textarea>`	Accepts multi-line text
`<button>`	Triggers actions like submit or reset

Example: Basic Form Structure

```
<form action="/register" method="get">
    <input type="text" name="fullname">
    <button type="submit">Submit</button>
</form>
```

3. Input Types and Their Usage

Text Input

```
<input type="text" id="fullname">
```

Email Input (Built-in Validation)

```
<input type="email" id="email">
```

Date Input

```
<input type="date" id="dob">
```

Radio Buttons (Single Selection)

```
<input type="radio" name="gender" value="Male"> Male
<input type="radio" name="gender" value="Female"> Female
```

Dropdown (Select Box)

```
<select id="school">
    <option value="SCOPE">SCOPE</option>
    <option value="SCORE">SCORE</option>
</select>
```

Textarea

```
<textarea rows="4">Vellore, Tamil Nadu</textarea>
```

File Upload

```
<input type="file" accept="image/*">
```

4. Events in HTML Forms

Events allow JavaScript to respond to user actions such as typing, clicking, or focusing on elements.

Common Form Events

Event	Description	Example
`onclick`	Triggered on mouse click	Button click
`oninput`	Triggered while typing	Live validation
`onfocus`	Triggered when element gets focus	Input highlight
`onblur`	Triggered when focus is lost	Validation on exit
`onchange`	Triggered when value changes	File upload

5. JavaScript Events – Practical Example

```
<input type="text"
       id="username"
       oninput="convertText()"
       onfocus="applyStyle()"
       onblur="removeStyle()">
```

```
<script>
function convertText() {
    const input = document.getElementById("username");
    input.value = input.value.toUpperCase();
}

function applyStyle() {
    const input = document.getElementById("username");
    input.style.backgroundColor = "lightblue";
}

function removeStyle() {
    const input = document.getElementById("username");
    input.style.backgroundColor = "";
}
</script>
```

6. Form Validation and Styling using JavaScript

Example: Name Length Validation

```
function changeCase() {
    const nameInput = document.getElementById("fullname");
    nameInput.value = nameInput.value.toUpperCase();

    if (nameInput.value.length >= 3) {
        nameInput.style.border = "2px solid green";
    } else {
        nameInput.style.border = "2px solid red";
    }
}
```

Concepts Used:

- DOM access
- Conditional logic
- Inline styling

7. Image Upload and Preview (Without Server)

HTML

```
<input type="file" id="photo" accept="image/*" onchange="previewImage()">
<img id="preview">
```

JavaScript

```
function previewImage() {
    const preview = document.getElementById("preview");
    const file = document.getElementById("photo").files[0];

    if (file) {
        preview.src = URL.createObjectURL(file);
        preview.style.display = "block";
    }
}
```

- Client-side file handling
- Dynamic image preview

8. JavaScript Naming Conventions

1. PascalCase (Classes)

```
class UserForm {
    constructor(name) {
        this.name = name;
    }
}
```

2. camelCase (Variables & Functions)

```
let userName = "Anand";

function calculateTotal(price, quantity) {
    return price * quantity;
}
```

3. CONSTANT_CASE (Constants)

```
const MAX_USERS = 100;
const API_URL = "https://api.example.com/";
```

9. JavaScript Built-in Objects

9.1 window Object

The `window` object represents the **browser window**.

```
window.alert("Hello!");
let name = window.prompt("Enter your name:");
let confirmStatus = window.confirm("Are you sure?");
```

Redirection

```
window.location.href = "https://www.vit.ac.in";
```

9.2 document Object

The `document` object represents the **HTML DOM**.

Access Elements

```
document.getElementById("title");
document.querySelector(".item");
```

Modify Content

```
element.textContent = "Plain Text";
element.innerHTML = "<b>Bold Text</b>";
```

Create Elements

```
let div = document.createElement("div");
div.textContent = "New Element";
document.body.appendChild(div);
```

10. Combined Example Using document and window

```
<h1 id="title">Hello, World!</h1>
<button onclick="changeTitle()">Change Title</button>
<button onclick="showPrompt()">Show Prompt</button>

<script>
function changeTitle() {
    let title = document.getElementById("title");
    title.textContent = "Title Changed!";
    title.style.color = "red";
}

function showPrompt() {
    let name = window.prompt("What's your name?");
    if (name) {
        alert("Hello, " + name);
    }
}
</script>
```