WTMA LAB MANUAL – 5 Javascript PROGRAMS

Experiment – Javascript Form Validation, Event Handling and DOM Manipulations

---

Programme 1 – Write JavaScript to validate the following fields of the Registration page.

a) First Name (Name should contains alphabets and the length should not be less than 6 characters).

b) Password (Password should not be less than 6 characters length).

c) E-mail id (should not contain any invalid and must follow the standard pattern name@domain.com)

d) Mobile Number (Phone number should contain 10 digits only).

e) Last Name and Address (should not be Empty).

Aim

To develop a Registration Web Page and implement **JavaScript form validation** for the following fields:

- First Name
- Password
- E-mail ID
- Mobile Number
- Last Name
- Address

Steps for Implementation

1. Create an HTML file named registration.html.
2. Design a registration form using <form> tag.
3. Add input fields for:
    - First Name
    - Last Name
    - Password
    - Email ID

- Mobile Number
- Address

4. Assign **id** attributes to each field for DOM access.
5. Create a JavaScript function named validateForm().
6. Use the following validations:
    - **First Name**
        - Alphabets only
        - Minimum 6 characters
    - **Password**
        - Minimum 6 characters
    - **Email**
        - Must follow pattern name@domain.com
    - **Mobile Number**
        - Exactly 10 digits
    - **Last Name & Address**
        - Should not be empty
7. Use **Regular Expressions** where required.
8. Display alert messages if validation fails.
9. Return false to stop form submission on error.
10. Return true when all validations pass.

Program Code

```
<!DOCTYPE html>

<html>

<head>

  <title>Registration Form Validation</title>


<script>
```

```
function validateForm() {

    // Fetch values
    var fname = document.getElementById("fname").value;
    var lname = document.getElementById("lname").value;
    var password = document.getElementById("password").value;
    var email = document.getElementById("email").value;
    var mobile = document.getElementById("mobile").value;
    var address = document.getElementById("address").value;

    // First Name Validation
    var namePattern = /^[A-Za-z]{6,}$/;

    if (!namePattern.test(fname)) {
        alert("First Name must contain alphabets and minimum 6 characters");
        return false;
    }

    // Password Validation
    if (password.length < 6) {
        alert("Password must be at least 6 characters long");
        return false;
    }

    // Email Validation
    var emailPattern =
        /^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$/;
```

```
if (!emailPattern.test(email)) {
    alert("Enter valid Email ID (name@domain.com)");
    return false;
}

// Mobile Number Validation
var mobilePattern = /^[0-9]{10}$/;

if (!mobilePattern.test(mobile)) {
    alert("Mobile number must contain exactly 10 digits");
    return false;
}

// Last Name Validation
if (lname == "") {
    alert("Last Name should not be empty");
    return false;
}

// Address Validation
if (address == "") {
    alert("Address should not be empty");
    return false;
}

// If all validations pass
alert("Form Validated Successfully!");
```

```
    return true; // allows submission to dummy file
}
</script>

</head>

<body>

<h2>Registration Form</h2>

<form
    action="success.html"
    method="post"
    name="regForm"
    onsubmit="return validateForm()"
>

First Name:
<input type="text" id="fname" name="fname"><br><br>

Last Name:
<input type="text" id="lname" name="lname"><br><br>

Password:
<input type="password" id="password" name="password"><br><br>

Email ID:
<input type="text" id="email" name="email"><br><br>
```

Mobile Number:

<input type="text" id="mobile" name="mobile">

Address:

<textarea id="address" name="address"></textarea>

<input type="submit" value="Submit">

</form>

</body>

</html>

Output

1. The Registration Form is displayed with the following input fields:
    - First Name
    - Last Name
    - Password
    - Email ID
    - Mobile Number
    - Address
    - Submit Button
2. When the user clicks the **Submit** button:
    - If any field is invalid, an alert message is displayed:
        - **First Name invalid** →
          "First Name must contain alphabets and minimum 6 characters"

- **Password invalid** →
  "Password must be at least 6 characters long"
- **Email invalid** →
  "Enter valid Email ID (name@domain.com)"
- **Mobile invalid** →
  "Mobile number must contain exactly 10 digits"
- **Last Name empty** →
  "Last Name should not be empty"
- **Address empty** →
  "Address should not be empty"

3. If all inputs are valid:
   - Pop-up message appears:
     "Form Validated Successfully!"
   - Form is submitted to the dummy file success.html.

Result

Thus, the Registration Form was validated successfully using JavaScript. All input fields were checked for correctness using conditions and Regular Expressions, and the form submission was allowed only after successful validation.

Programme 2 – Write a Javascript code to display a Pop Up "Form Submitted" when User clicks on submit Button on above Form

Aim

To write a JavaScript program to display a pop-up message **"Form Submitted"** when the user clicks on the Submit button of the Registration Form.

Steps for Implementation

1. Create an HTML file.
2. Design a simple Registration Form using <form> tag.
3. Add required input fields (First Name, Last Name, etc.).

4. Add a **Submit button**.
5. Use the onsubmit event in the <form> tag.
6. Create a JavaScript function named showPopup().
7. Inside the function, use alert() method to display the message.
8. Return true so that the form submits after showing the pop-up.

Program Code

```
<!DOCTYPE html>
<html>
<head>
   <title>Form Submission Popup</title>

<script>
function showPopup() {

   alert("Form Submitted");

   return true; // allows form submission
}
</script>

</head>

<body>

<h2>Registration Form</h2>

<form
```

```
   action="success.html"
   method="post"
   onsubmit="return showPopup()"
>

First Name:
<input type="text" name="fname"><br><br>

Last Name:
<input type="text" name="lname"><br><br>

Password:
<input type="password" name="password"><br><br>

Email ID:
<input type="text" name="email"><br><br>

Mobile Number:
<input type="text" name="mobile"><br><br>

Address:
<textarea name="address"></textarea><br><br>

<input type="submit" value="Submit">

</form>

</body>
```

</html>

Output

- When user clicks **Submit** → Pop-up appears:

"Form Submitted"

- After clicking **OK** → Form submits to dummy page.

Programme 3 - Create a web page with a text box or image that responds to mouse events such as mouseover, mouseout, and click. Use DOM event handling to change styles or display messages dynamically.

Aim

To create a web page with a text box or image that responds to mouse events such as **mouseover, mouseout, and click** using DOM event handling to change styles and display messages dynamically.

Steps for Implementation

1. Create an HTML file named mouseevents.html.
2. Design a web page with:
    - A text heading **or**
    - An image element.
3. Assign an **id** to the element for DOM access.
4. Create a <p> tag to display dynamic messages.
5. Write JavaScript functions for mouse events:
    - mouseOver()
    - mouseOut()
    - mouseClick()

6. Use DOM methods to:
    - Change text color.
    - Change background color.
    - Display messages.
7. Attach events using:
    - onmouseover
    - onmouseout
    - onclick
8. Save and run the file in a browser.

Program Code

```
<!DOCTYPE html>
<html>
<head>
   <title>Mouse Events Using DOM</title>

<script>

function mouseOver() {

   var element = document.getElementById("demo");

   element.style.color = "white";
   element.style.backgroundColor = "green";

   document.getElementById("msg").innerHTML =
      "Mouse Over Event Triggered";
}
```

```
function mouseOut() {

   var element = document.getElementById("demo");

   element.style.color = "black";
   element.style.backgroundColor = "lightgray";

   document.getElementById("msg").innerHTML =
      "Mouse Out Event Triggered";
}

function mouseClick() {

   document.getElementById("msg").innerHTML =
      "Mouse Click Event Triggered";
}

</script>

</head>

<body>

<h2>Mouse Events Demo</h2>

<div id="demo"
    onmouseover="mouseOver()"
```

```
    onmouseout="mouseOut()"
    onclick="mouseClick()"
    style="
      width:250px;
      padding:20px;
      background-color:lightgray;
      text-align:center;
      cursor:pointer;
    ">

   Move Mouse Here

</div>

<br>

<p id="msg"></p>

</body>
</html>
```

Output

- When mouse is placed over the box → Background turns **green** and message displays.
- When mouse leaves → Style resets.
- When clicked → “Mouse Click Event Triggered” message appears.

Result

Thus, the web page was created successfully, and mouse events such as **mouseover, mouseout, and click** were handled using JavaScript DOM.

Programme 4 – Write an HTML page including any required JavaScript that takes a number from text field in the range of 0 to 999 and shows it in words. It should not accept four and above digits, alphabets and special characters.

Aim

To create an HTML page that accepts a number (0–999) from a text field and displays the number in words using JavaScript.
The program should not accept:

- Four or more digits
- Alphabets
- Special characters

Steps for Implementation

1. Create an HTML file named numbertowords.html.
2. Design a web page with:
    - A text box to enter the number.
    - A button to process the input.
    - A label/paragraph to display the result.
3. Assign an **id** to the text box for DOM access.
4. Create a JavaScript function named convertToWords().
5. Read the number using getElementById().value.
6. Validate the input:
    - Allow digits only.
    - Length ≤ 3.
    - Range 0–999.
7. Create arrays for:
    - Units

- Teens
- Tens

8. Convert the number into words using logic.
9. Display the result dynamically using innerHTML.
10. Run the program in the browser.

Program Code

```
<!DOCTYPE html>
<html>
<head>
   <title>Number to Words</title>

<script>

function convertToWords() {

   var num = document.getElementById("num").value;

   // Validation – Only digits allowed
   if (!/^[0-9]+$/.test(num)) {
      alert("Enter digits only (0–999)");
      return;
   }

   // Validation – Max 3 digits
   if (num.length > 3) {
      alert("Only numbers from 0 to 999 are allowed");
      return;
```

```
}

var ones = [
    "", "One", "Two", "Three", "Four",
    "Five", "Six", "Seven", "Eight", "Nine"
];

var teens = [
    "Ten", "Eleven", "Twelve", "Thirteen",
    "Fourteen", "Fifteen", "Sixteen",
    "Seventeen", "Eighteen", "Nineteen"
];

var tens = [
    "", "", "Twenty", "Thirty", "Forty",
    "Fifty", "Sixty", "Seventy",
    "Eighty", "Ninety"
];

num = parseInt(num);

if (num == 0) {
    document.getElementById("result").innerHTML = "Zero";
    return;
}

var word = "";
```

```
    // Hundreds
    if (num >= 100) {
        word += ones[Math.floor(num / 100)] + " Hundred ";
        num = num % 100;
    }

    // Tens & Teens
    if (num >= 10 && num <= 19) {
        word += teens[num - 10];
    }
    else if (num >= 20) {
        word += tens[Math.floor(num / 10)] + " ";
        num = num % 10;
    }

    // Ones
    if (num > 0 && num < 10) {
        word += ones[num];
    }

    document.getElementById("result").innerHTML =
        "In Words: " + word;
}

</script>

</head>
```

```
<body>

<h2>Number to Words (0–999)</h2>

Enter Number:
<input type="text" id="num">

<button onclick="convertToWords()">
   Convert
</button>

<h3 id="result"></h3>

</body>
</html>
```

Output

Examples:

Input	Output
5	Five
12	Twelve
45	Forty Five
100	One Hundred
256	Two Hundred Fifty Six

Invalid inputs show alert messages.

Result

Thus, the HTML page was created successfully to convert numbers (0–999) into words using JavaScript with proper input validation.

Programme 5 - Write a JavaScript program that creates an Employee object with properties like Employee ID, Name, Designation, and Salary. Store multiple employee objects in an array and use functions to display employee details and calculate total salary expenditure.

Aim

To create an Employee object with properties such as Employee ID, Name, Designation, and Salary, store multiple employee objects in an array, and use functions to display employee details and calculate total salary expenditure.

Steps for Implementation

1. Create an HTML file named employee.html.
2. Design a simple webpage with:
    - A heading.
    - A button to display employee details.
    - A section to show output.
3. Create a JavaScript **Employee object** with properties:
    - Employee ID
    - Name
    - Designation
    - Salary
4. Create multiple employee objects.
5. Store all objects inside an **array**.

6. Write a function displayEmployees() to:
    - Loop through the array.
    - Display employee details.
7. Write another function calculateTotalSalary() to:
    - Add all salaries.
    - Display total expenditure.
8. Call functions using a button click event.
9. Run the program in a browser.

Program Code

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Employee Object Programme</title>

<style>
  body{
    font-family: Arial;
    margin:40px;
    background:#f2f2f2;
    text-align:center;
  }

  table{
    width:80%;
    margin:20px auto;
    border-collapse:collapse;
```

```
    background:white;
}

th, td{
    border:1px solid #ccc;
    padding:10px;
    text-align:center;
}

th{
    background:#333;
    color:white;
}

button{
    padding:12px 22px;
    font-size:15px;
    margin:10px;
    background:#0d6efd;
    color:white;
    border:none;
    border-radius:6px;
    cursor:pointer;
}

button:hover{
    background:#084298;
}
```

```
    .total{
        font-size:20px;
        margin-top:15px;
        font-weight:bold;
    }
</style>
</head>

<body>

<h2>Employee Details Using Object, Array & Function</h2>

<!-- BUTTONS -->
<button id="displayBtn">Display Employee Details</button>
<button id="salaryBtn">Calculate Total Salary</button>

<!-- TABLE -->
<table id="empTable">
    <tr>
        <th>Emp ID</th>
        <th>Name</th>
        <th>Designation</th>
        <th>Salary</th>
    </tr>
</table>

<!-- TOTAL -->
```

```
<div class="total" id="totalSalary"></div>



<script>

// Employee Array
const employees = [

   { id:101, name:"Arun", designation:"Manager", salary:50000 },
   { id:102, name:"Divya", designation:"Developer", salary:40000 },
   { id:103, name:"Karthik", designation:"Tester", salary:30000 },
   { id:104, name:"Meena", designation:"HR", salary:35000 }

];



// Function → Display Employees
function displayEmployees(){

   const table = document.getElementById("empTable");

   // Prevent duplicate rows
   if(table.rows.length > 1) return;

   employees.forEach(emp => {

      const row = table.insertRow();
```

```
        row.insertCell(0).innerHTML = emp.id;
        row.insertCell(1).innerHTML = emp.name;
        row.insertCell(2).innerHTML = emp.designation;
        row.insertCell(3).innerHTML = emp.salary;

    });

}


// Function → Calculate Salary
function calculateTotalSalary(){

    let total = 0;

    employees.forEach(emp => {
        total += emp.salary;
    });

    document.getElementById("totalSalary").innerHTML =
        "Total Salary Expenditure : ₹ " + total;

}


// Button Events (Pure JS)
document.getElementById("displayBtn")
        .addEventListener("click", displayEmployees);
```

```
document.getElementById("salaryBtn")
    .addEventListener("click", calculateTotalSalary);


</script>
</body>
</html>
```

Output

- Clicking **Display Employees** → Shows all employee details.
- Clicking **Calculate Total Salary** → Displays total salary expenditure.

Result

Thus, the JavaScript program was created successfully to store multiple Employee objects in an array, display their details, and calculate total salary expenditure using functions.