

# JavaScript Form validation

- JavaScript provides a way to validate form's data on the client's computer before sending it to the web server. Form validation generally performs two functions.
- **Basic Validation** – First of all, the form must be checked to make sure all the mandatory fields are filled in. It would require just a loop through each field in the form and check for data.
- **Data Format Validation** – Secondly, the data that is entered must be checked for correct form and value. Your code must include appropriate logic to test correctness of data.

# Automatic HTML Form Validation (by using **required** attribute)

**Ques.** WAP using JavaScript to create an input field(Fname) If this form field is empty, the **required** attribute should prevent this form from being submitted.

**Soln**

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<form action="/action_page.php" method="post">
```

```
<input type="text" name="fname" required>
```

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

```
</form>
```

```
</body>
```

```
</html>
```

# To validate numeric input

Ques: WAP to create an input field that accepts only the numbers between 1 and 10.

Soln

```
<html><body>
<input type= text id="numb">
<button type="button" onclick="myFunction()">Submit</button>
<p id="demo"></p>
<script>
function myFunction() {
    var x, text;

    // Get the value of the input field with id="numb"
    x = document.getElementById("numb").value;

    // If x is Not a Number or less than one or greater than 10
    if (isNaN(x) || x < 1 || x > 10) {
        text = "Input not valid";
    } else {
        text = "Input OK";
    }
    document.getElementById("demo").innerHTML = text;
}
</script></body></html>
```

# Data Validation

- Data validation is the process of ensuring that user input is correct

## **EXAMPLE OF DATA VALIDATION:**

- has the user filled in all required fields?
- has the user entered a valid date?
- has the user entered text in a numeric field?

Validation can be defined by many different methods, and deployed in many different ways.

- **Server side validation** is performed by a web server, after input has been sent to the server.
- **Client side validation** is performed by a web browser, before input is sent to a web server.

# HTML Constraint Validation

## Constraint Validation HTML Input Attributes

Attribute	Description
disabled	Specifies that the input element should be disabled
max	Specifies the maximum value of an input element
min	Specifies the minimum value of an input element
pattern	Specifies the value pattern of an input element
required	Specifies that the input field requires an element
type	Specifies the type of an input element

# The disabled Attribute

- The disabled attribute specifies that the input field is disabled.
- A disabled input field is unusable and un-clickable, and its value will not be sent when submitting the form:

# example

```
<html>
<body>
<form action="">
First name:<br>
<input type="text" name="firstname" value ="John" disabled>
<br>
Last name:<br>
<input type="text" name="lastname">
</form>

</body>
</html>
```

First name:

John

Last name:



# min and max

- min and max attributes specify the minimum and maximum values for an <input> element.
- The min and max attributes work with the following input types: number, range, date, datetime-local, month, time and week.

Example:

```
<input type="number" name="quantity" min="1"
max="5"><br>
```

# The pattern Attribute

- pattern attribute works with the following input types: text, search, url, tel, email, and password.

Expression	Description
[abc]	Find any character between the brackets
[^abc]	Find any character NOT between the brackets
[0-9]	Find any character between the brackets (any digit)
[^0-9]	Find any character NOT between the brackets (any non-digit)
(x y)	Find any of the alternatives specified

# WAP to create an input field that can contain only three letters (no numbers or special characters):

```
<html>
```

```
<body>
```

```
<form action="/action_page.php">
```

```
Country code: <input type="text" name="country_code" pattern="[A-Za-z]{3}" >
```

```
<input type="submit">
```

```
</form>
```

```
</body>
```

```
</html>
```

WAP to create <input> element with type="password" that must contain 6 or more characters

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<form action="/action_page.php">
```

```
  Password: <input type="password" name="pw" pattern=".{6,}" >
```


```
  <input type="submit">
```

```
</form>
```

```
</body>
```

```
</html>
```

Password:

 Please match the requested format.

Dot (.) is used to find a single character, except  
newline or line terminator.

**Ques.: WAP to create an <input> element with type="password" that must contain 8 or more characters that are of at least one number, and one uppercase and lowercase letter.**

## Soln

```
<html><body>  
<form action="/action_page.php">  
Password: <input type="password" name="pw" pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,}">  
  
<input type="submit">  
</form>  
</body></html>
```

Here “\d” is used to find a digit

The  $n^*$  quantifier matches any string that contains zero or more occurrences of  $n$ .

The  $?=n$  quantifier matches any string that is followed by a specific string  $n$ .

# Validity Properties

The rangeOverflow Property

Task: If the number in an input field is greater than 100 (the input's max attribute), display a message:

# solution

```
<!DOCTYPE html>
<html>
<body>

<p>Enter a number and click OK:</p>

<input id="id1" type="number" max="100">
<button onclick="myFunction()">OK</button>

<p>If the number is greater than 100 (the input's max attribute), an error message will be displayed.</p>

<p id="demo"></p>

<script>
function myFunction() {
  var txt = "";
  if (document.getElementById("id1").validity.rangeOverflow) {
    txt = "Value too large";
  } else {
    txt = "Input OK";
  }
  document.getElementById("demo").innerHTML = txt;
}
</script>

</body>
</html>
```



## The rangeUnderflow Property

Task : If the number in an input field is less than 100 (the input's min attribute), display a message

```
<html>
<body>

<p>Enter a number and click OK:</p>

<input id="id1" type="number" min="100">
<button onclick="myFunction()">OK</button>

<p>If the number is less than 100 (the input's min attribute), an error message will be displayed.</p>

<p id="demo"></p>

<script>
function myFunction() {
  var txt = "";
  if (document.getElementById("id1").validity.rangeUnderflow) {
    txt = "Value too small";
  } else {
    txt = "Input OK";
  }
  document.getElementById("demo").innerHTML = txt;
}
</script>

</body>
</html>
```

**Question: write code to create a text box. If the value of text box is blank, then alert box must be displayed “username cannot be null”, else open a web page named “page1.html”**

```
<html> <head> <title>Login page</title></head>
  <body>
    <form name="login">
      Username<input type="text" name="userid"> <br>
      Contact No.<input type="text" name="Contact"> <br>
      <input type="button" onclick="check(this.form)" value="enter details"><br>
        <input type="reset" value="Reset">
    </form>
    <script language="javascript">
      function check(form)
      {
        if(form.userid.value == "" || form.userid.value == null)
        {
          alert("user name cannot be null")
        }
        else
        {
          window.open("page1.html")
        }
      }
    </script></body></html>
```

## Question: Validation:

If userID is 123 and password is abc, only then the user can login, otherwise alert box is displayed "incorrect user name and password"

```
<html><head><title>Login page</title></head>
<body>
<form name="login">
Username<input type="text" name="userid"><br>
Password<input type="password" name="pswrd"><br>
<input type="button" onclick="check(this.form)" value="Login"><br>
<input type="reset" value="Reset">
</form>
<script language="javascript">
function check(form)
{
    if(form.userid.value == "123" && form.pswrd.value == "abc")
    {
        window.open("page1.html")
    }
    else
    {
        alert("Incorrect Password or Username")
    }
}
</script></body></html>
```

# Some more HTML input attributes

# multiple Attribute

- The multiple attribute specifies that the user is allowed to enter more than one value in the `<input>` element.
- The multiple attribute works with the following input types: email, and file.

# Example: multiple

```
<html>
```

```
<body>
```

```
<form action="/action_page.php">
```

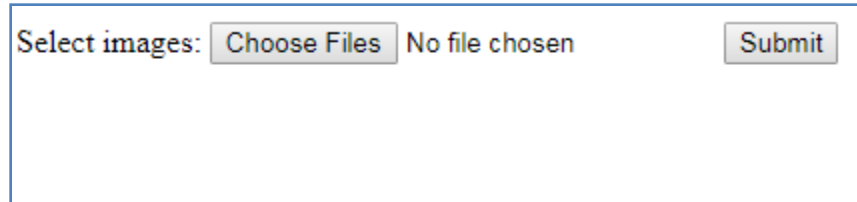
```
  Select images: <input type="file" name="img"
    multiple>
```

```
  <input type="submit">
```

```
</form>
```

```
</body>
```

```
</html>
```

A screenshot of a web form rendered from the provided HTML code. It features a text label "Select images:" followed by a file selection button labeled "Choose Files" and a status text "No file chosen". To the right is a "Submit" button. Below these elements is a large, empty rectangular area for image uploads.

# The placeholder Attribute

- The placeholder attribute specifies a hint that describes the expected value of an input field (a sample value or a short description of the format).
- The hint is displayed in the input field before the user enters a value.
- The placeholder attribute works with the following input types: text, search, url, tel, email, and password.

# Example: placeholder

```
<html>
<body>

<form action="/action_page.php">
  <input type="text" name="fname" placeholder="First
    name"><br>
  <input type="text" name="lname" placeholder="Last
    name"><br>
  <input type="submit" value="Submit">
</form>
</body>
</html>
```

---

First name
Last name
Submit



# The autocomplete Attribute

- The autocomplete attribute specifies whether a form or input field should have autocomplete on or off.
- When autocomplete is on, the browser automatically completes the input values based on values that the user has entered before.

# Example: autocomplete

```
<form action="/action_page.php" autocomplete="on">  
First name:<input type="text" name="fname"><br>  
Last name: <input type="text" name="lname"><br>  
E-mail: <input type="email" name="email" autocomplete="off"><br>  
<input type="submit">  
</form>
```

# novalidate Attribute

- The novalidate attribute is a <form> attribute.
- When present, novalidate specifies that the form data should not be validated when submitted.

## Example

**Indicates that the form is not to be validated on submit:**

```
<form action="/action_page.php" novalidate>  
E-mail: <input type="email" name="user_email">  
<input type="submit">  
</form>
```

## TASK:

Create THE GIVEN FORM form

---

QUANTITY

PINCODE

COUNTRY

- First field quantity(range 1 to 3)
- Pin code: 6 digits only
- Country (textbox must be disabled)