# HTML Forms and Validation

# Validation

Validation is a way of making sure a user entered the correct information into an input field on your webpage. The benefits to this are:

- Ensuring user input it correct (username, password, etc)
- User input is in the correct data format for databases
- Puts security around user input fields

# How do to simple validation on a page

Javascript functions a great way to create reusable validation on a webpage.

This can be done by giving an input field an id and using document.getElementById("firstName").value to check what the user has put into that input field

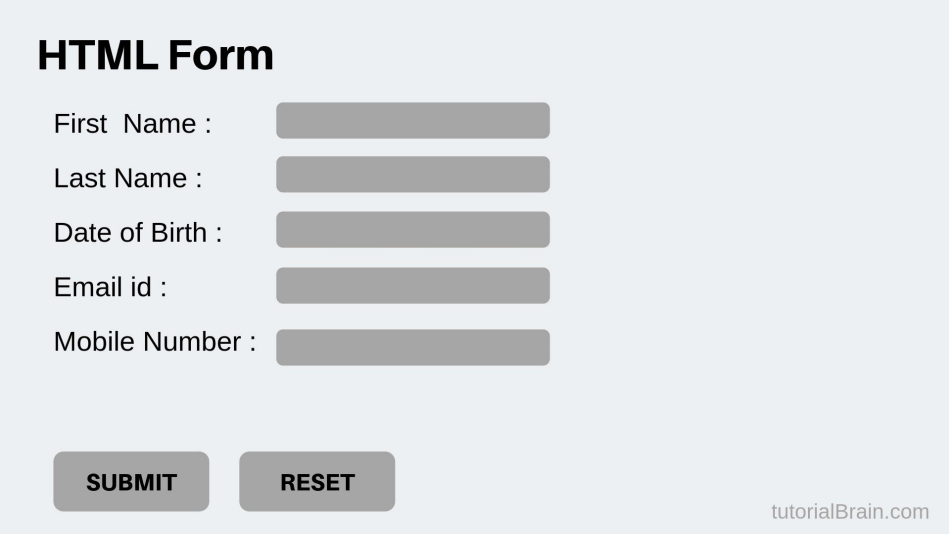
When validating multiple inputs on a webpage make sure you think about using functions to help reduce having to write duplicate code

# Form validation vs HTML Input

You can use regular HTML input fields as opposed to using HTML Forms. The benefit of using HTML forms is the pop up messages and error handling are built in where with HTML Inputs you will need to manually verify each field.

# HTML Forms

Forms are a way of getting and validating user input on your site



# Design principles for forms

Start with the easiest questions first and ensure there is a logical flow

Forms should follow a one column format and each field should be sized correctly

Text should be left aligned

If your using a longer form you should show a users progress

Errors or fields filled out incorrectly should show a positive, user friendly error message

You should always require as few fields as possible from your users

For shopping sites you should make the discount or coupon code field less prominent

# Backend principals for Forms

Validate validate validate!!!

Use keywords in the name or label for auto-complete (First name, Last name, Address)

Do not allow special characters in input fields

Use drop downs, radio button, or pre defined fields as much as possible

# Form tag

```
<form>

    Whatever you want!

</form>
```

Form tags are a way for other coders to better understand your code and to help keep you organized. There are also attributes you can use within the form tag.

# Input Attributes to use in a form

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

# Disabled

```
<input type="text" id="firstName" name="firstName" disabled>
```

# Max

```
<label for="numberEnter">Enter a number less than 5:</label>

<input type="number" id="humberEnter" name="numberEnter" max="5">
```

# Min

```
<label for="numberEnter">Enter a number greater than 5:</label>

<input type="number" id="numberEnter" name="numberEnter" min="5">
```

# Pattern

```
<form>

  <label for="pwd">Password:</label>

  <input type="password" id="pwd" name="pwd" pattern=".{10,}" title="Ten or more characters">

  <input type="submit">

</form>
```

**This one gets complicated really quick!!

# Required

```
<input type="text" id="username" name="username" required>
```

# Lets try it

```
<input type="text" id="firstName" disabled>

<input type="number" id="numberEnter" max="5">

<input type="number" id="numberEnter" min="100>

<input type="password" id="pwd" pattern=".{10,}" title="Ten or more characters">

<input type="text" id="username" required>

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

# Placeholder

Place holder allows you to pre populate fields

<input type="text" id="email" name="email" placeholder="Enter your email address" />

# Autocomplete

Autocomplete lets you specify if you want to let the user use the built in autocomplete fields on from their browsers data

```
<form autocomplete="on">

</form>
```

# Form onsubmit

When the submit button is pressed you can specify a javascript function to call. This works very similar to the onclick function in a button

The important part to note here is you must include return FUNCTIONNAME(this) in your call

Return is used to let you prompt the user if the function returns a true of false

This represents sending the form (or this form) to the function

```
<form onsubmit="return formValiador(this)">
    <text>User name:</text><br>
    <input type="text" id="uname" required><br>
    <text>Password:</text><br>
    <input type="password" id="pwd" pattern=".{10,}" title="Ten or more characters" required><br><br>
    <input type="submit" value="Submit">
</form>
```

# Javascript form validator

In the below example note the form.action line. This indicates what will happen if the function returns true.

You should always specify a true and false return when doing a form validator

```
function formValiador(form) {
    form.action = "loginSuccessful.html";
    var username = document.getElementById("uname").value;
    var password = document.getElementById("pwd").value;
    if(username =="jordan" && password == "0000000000"){
        alert("Congrats you logged in");
        return true;
    }
    else{
        alert("Incorrect Username/Pass")
        return false;
    }
}
```

# Lets try it

Create a simple form that allows for a username and password login

Try using some of the built in form validation attributes we have learned

```
pattern=".{10,}"

required

placeholder="Enter your email address"
```