

HTML MODULE 04

CREATING FORMS TO COLLECT AND VALIDATE USER INPUT

Summer 2021 – Web Development using ASP .Net Core MVC



HTML FORMS

- Use the **<form>** element to create an HTML form for user input.
 - The <form> element is a **container** for multiple **input elements**, such as: **text fields**, **checkboxes**, **submit buttons**, etc.
 - Note: the form itself is not visible!
 - Note: the default width of an input field is 20 characters

Type	Description
<input type="text">	Displays a single-line text input field
<input type="radio">	Displays a radio button (for selecting one of many choices)
<input type="checkbox">	Displays a checkbox (for selecting zero or more of many choices)
<input type="submit">	Displays a submit button (for submitting the form)
<input type="button">	Displays a clickable button

- Example:
 - `<form action="/login_page.php">`
 - `<label for="fname">First name:</label>
`
 - `<input type="text" id="fname" name="fname" >
`
 - `<label for="lname">Last name:</label>
`
 - `<input type="text" id="lname" name="lname" >
`
 - `<label for="pass">Enter your password:</label>
`
 - `<input type="password" id="pass" name="pass">

`
 - `<input type="submit" value="Submit">`
 - `</form>`

HTML Forms

First name:

Last name:

Enter your password:

- Note: the **name attribute** is used by the server (when processing the request)... if omitted, the value of **that input field will not be sent at all**.
- Note: the **for** attribute of the **<label>** tag must equal to the **id** attribute of the **<input>** element to bind them together.
- Note: **<button>** element produces varying results in different browsers and its use is discouraged. Encourage **<input type="button" />** instead
- Source: https://www.w3schools.com/html/html_forms.asp ← see also radio buttons, and checkboxes!



HTML FORMS – ATTRIBUTES

Example:

- `<form action="/login_page.php">`
- `<label for="fname">First name:</label>
`
- `<input type="text" id="fname" name="fname" >
`
- `<label for="lname">Last name:</label>
`
- `<input type="text" id="lname" name="lname" >
`
- `<label for="pass">Enter your password:</label>
`
- `<input type="password" id="pass" name="pass">

`
- `<input type="submit" value="Submit">`
- `</form>`



The **action** attribute defines the URL of the action to be called when the form is submitted.

- **GET**: appends name/value pairs to the **URL**, therefore **not secure**, useful if the user wants to bookmark searches
- **POST**: appends form data **inside body** of the request, **not in the URL**, has no size limitation

The **method** attribute specifies the HTTP method used when the form data is submitted.

- `<form action="/login_page.php" method="get">` ([test it!](#))
- `<form action="/login_page.php" method="post">` ([test it!](#))

Notes on GET:

- Appends the form data to the URL, in name/value pairs
- NEVER use GET to send sensitive data! (the submitted form data is visible in the URL!)
- The length of a URL is limited (2048 characters)
- Useful for form submissions where a user wants to bookmark the result
- GET is good for non-secure data, like query strings in Google

Notes on POST:

- Appends the form data inside the body of the HTTP request (the submitted form data is not shown in the URL)
- POST has no size limitations, and can be used to send large amounts of data.
- Form submissions with POST cannot be bookmarked

The **autocomplete** attribute specifies whether the browser should automatically complete values based on values entered before.

- `<form action="/login_page.php" autocomplete="on">`

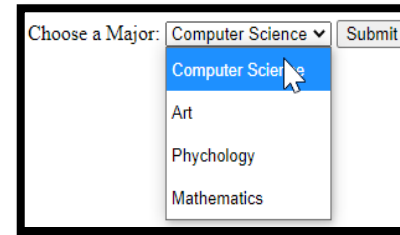
Source: https://www.w3schools.com/html/html_forms_attributes.asp ← check out the **target** attribute



HTML FORMS – ELEMENTS

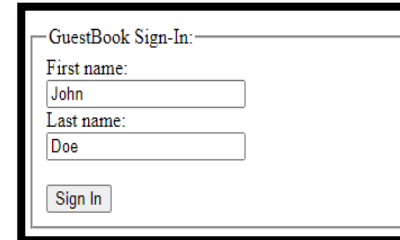
- We'll only see a few ones in here. You may want to check the link below to see other ones.
 - `<input>`, `<label>`, `<select>`, `<textarea>`, `<button>`, `<fieldset>`, `<legend>`, `<datalist>`, `<output>`, `<option>`, `<optgroup>`

- Use the `<select>` element to obtain a **drop-down list**:
 - The `<option>` elements define the available options
 - The `<optgroup>` element defines a group of related options
 - See an example: https://www.w3schools.com/tags/tryit.asp?filename=tryhtml_optgroup
 - The `selected` attribute used to preselect an option
 - The `size` attribute specifies the number of visible values
 - The `multiple` attribute used to allow selection of more than one value.

A screenshot of a web form. It has a label "Choose a Major:" followed by a dropdown menu. The dropdown menu is open, showing four options: "Computer Science", "Art", "Phychology", and "Mathematics". A mouse cursor is hovering over "Computer Science". To the right of the dropdown is a "Submit" button.

```
<form action="/action_page.php">
  <label for="cars">Choose a Major:</label>
  <select id="majors" name="majors">
    <option value="cs">Computer Science</option>
    <option value="art">Art</option>
    <option value="psy">Phychology</option>
    <option value="math">Mathematics</option>
  </select>
  <input type="submit">
</form>
```

- Use the `<fieldset>` element to group (& draw a box around) related data.
 - The `<legend>` element can be used to define a caption for the `<fieldset>` element.

A screenshot of a web form. It has a label "GuestBook Sign-In:" followed by a fieldset. Inside the fieldset, there are two text input fields: "First name:" with the value "John" and "Last name:" with the value "Doe". Below the input fields is a "Sign In" button.

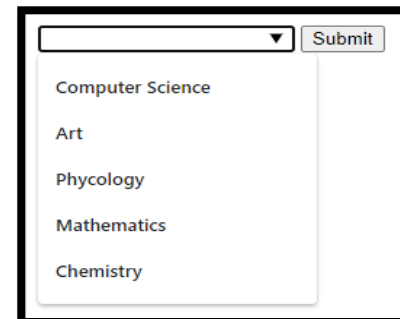
```
<form action="/action_page.php">
  <fieldset>
    <legend>GuestBook Sign-In:</legend>

    <label for="fname">First name:</label><br>
    <input type="text" id="fname" name="fname" value="John"><br>

    <label for="lname">Last name:</label><br>
    <input type="text" id="lname" name="lname" value="Doe"><br><br>

    <input type="submit" value="Sign In">
  </fieldset>
</form>
```

- Use the `<datalist>` element to specify a list of pre-defined options for an `<input>` element.
 - The `list` attribute of the `<input>` element, must refer to the `id` attribute of the `<datalist>` element.

A screenshot of a web form. It has a text input field with a dropdown arrow. Below the input field is a list of options: "Computer Science", "Art", "Phychology", "Mathematics", and "Chemistry". To the right of the input field is a "Submit" button.

```
<form action="/action_page.php">
  <input list="majors" name="majorSelection">
  <datalist id="majors">
    <option value="Computer Science">
    <option value="Art">
    <option value="Phychology">
    <option value="Mathematics">
    <option value="Chemistry">
  </datalist>
  <input type="submit">
</form>
```

- See also this `<output>` [example](#) (on your own)
- Sources:
 - https://www.w3schools.com/html/html_form_elements.asp
 - https://www.w3schools.com/tags/tag_optgroup.asp

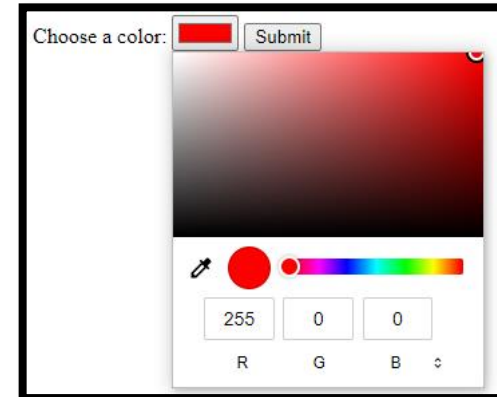
HTML INPUT TYPES

- Please read on your own, as needed

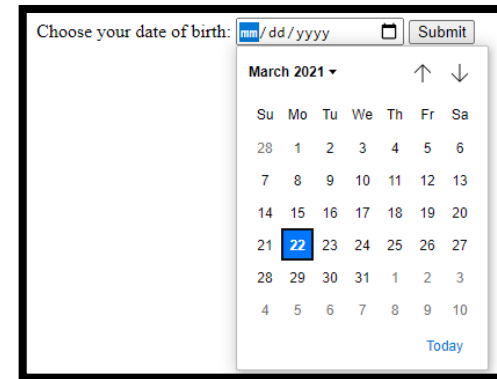
- `<input type="button">`
- `<input type="checkbox">`
- `<input type="color">`
- `<input type="date">`
- `<input type="datetime-local">`
- `<input type="email">`
- `<input type="file">`
- `<input type="hidden">`
- `<input type="image">`
- `<input type="month">`
- `<input type="number">`
- `<input type="password">`
- `<input type="radio">`
- `<input type="range">`
- `<input type="reset">`
- `<input type="search">`
- `<input type="submit">`
- `<input type="tel">`
- `<input type="text">`
- `<input type="time">`
- `<input type="url">`
- `<input type="week">`

```
<form action="/action_page.php">
  <label for="favcolor">Choose a color:</label>
  <input type="color" id="favcolor" name="favcolor" value="#ff0000">

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

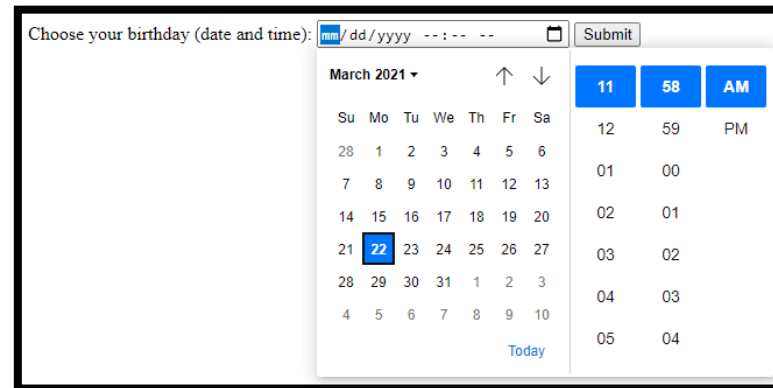


```
<form action="/action_page.php">
  <label for="birthday">Choose your date of birth:</label>
  <input type="date" id="birthday" name="birthday">
  <input type="submit" value="Submit">
</form>
```



```
<form action="/action_page.php">
  <label for="birthdaytime">Choose your birthday (date and time):</label>
  <input type="datetime-local" id="birthdaytime" name="birthdaytime">

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



- Source: https://www.w3schools.com/html/html_form_input_types.asp

HTML INPUT ATTRIBUTES

- There are several **attributes** for the HTML `<input>` element:
- Use the **value** attribute to specify an **initial value**.
- Use the **readonly** attribute to make the input field **read only**.
 - The user can still copy the value, select it, ...
 - Is this good for client-side validation? What about server-side validation?
- Use the **disabled** attribute to make the input field **disabled**.
 - The user can not copy the value, it can not select it either, ...
 - Is this good for client-side validation? What about server-side validation?
- Use the **pattern** attribute to specify a **regular expression** against which the value is checked.
 - See more on **regular expressions** in here: https://www.w3schools.com/js/js_regexp.asp
- Use the **placeholder** attribute to specify a **hint**.
- Use the **required** attribute to specify that a field **must be filled out** before submitting the form.
- The following are skipped. Please read them on your own, as needed:
 - size**, **multiple**, **step**, **autofocus**, **height**, **width**, **list**,
- Source: https://www.w3schools.com/html/html_form_attributes.asp

```
<form action="/action_page.php">  
  <label for="input1">Use of the Value Attribute:</label><br>  
  <input type="text" id="input1" value="Washington State"><br><br>  
  
  <label for="input2">Use of the Readonly Attribute:</label><br>  
  <input type="text" id="input2" value="Washington State" readonly><br><br>  
  
  <label for="input3">Use of the Disabled Attribute:</label><br>  
  <input type="text" id="input3" value="Washington State" disabled><br><br>  
  
  <label for="input4">Use of the Pattern Attribute:</label><br>  
  <input type="text" id="input4" pattern="[A-Za-z]{3}" title="Must use 3-letter values"><br><br>  
  
  <label for="input5">Use of the Placeholder Attribute:</label><br>  
  <input type="text" id="input5" placeholder="Washington State"><br><br>  
  
  <label for="input6">Use of the Required Attribute:</label><br>  
  <input type="text" id="input6" required><br><br>  
  
  <input type="submit" value="Submit">  
  
</form>
```

Use of the Value Attribute:

Use of the Readonly Attribute:

Use of the Disabled Attribute:

Use of the Pattern Attribute:

Use of the Placeholder Attribute:

Use of the Required Attribute:

```
<input type="text" id="input3" value="Changed State"  
disabled> == $0
```

Use of the Disabled Attribute:

VALIDATION

- We can implement validation
 - On the **client** side. ← What does this mean?
 - On the **server** side. ← What does this mean?
- Which one is better?
- Then why use both?
- **Client-side validation:**
 - Client validation alone is not sufficient!
 - We use it to improve user experience.
 - It can catch common user mistakes (what could happen if a user mistypes their email address?).
 - Users get immediate feedback without a request being sent to the server.
 - We also use it to minimize the number of requests sent the server
 - Client-side validation helps minimize the number of invalid requests sent to the server)
- **Server-side validation:**
 - The server still needs to validate the data received from client requests.
 - Why?
 - Check for malicious code (injection attacks, man-in-the-middle attacks, cross-site request forgery attacks, ...)
 - The server may have a better context of the data received from the client, so it can do a more thorough validation.



EXAMPLES OF CLIENT-SIDE VALIDATION

- One can make use of the following attributes to implement some client-side validation:
- Use the **required** attribute to specify that a field **must be filled out** before submitting the form.
- Use the **pattern** attribute to specify a **regular expression against which the value is checked**.
 - See more on **regular expressions** in here: https://www.w3schools.com/js/js_regexp.asp
- Use the **maxlength** attribute to specify the **maximum number of characters** allowed in an input field.
- Use the **min** and **max** attributes to specify the **minimum and maximum values** allowed in an input field.

```
<form action="/action_page.php">  
  <label for="input1">Use of the Required Attribute:</label><br>  
  <input type="text" id="input1" required><br><br>  
  
  <label for="input2">Use of the Pattern Attribute:</label><br>  
  <input type="text" id="input2" pattern="[A-Za-z]{3}" title="Must use 3-letter values"><br><br>  
  
  <label for="input3">Use of the Maxlength Attribute:</label><br>  
  <input type="text" id="input3" maxlength="4"><br><br>  
  
  <label for="input4">Use of the Min Attribute:</label><br>  
  <input type="date" id="input4" min="2003-03-25"><br><br>  
  
  <label for="input5">Use of the Min and Max Attribute:</label><br>  
  <input type="number" id="input5" min="1" max="5"><br><br>  
  
  <input type="submit" value="Submit">  
</form>
```

Use of the Required Attribute:

Use of the Pattern Attribute:

Use of the Maxlength Attribute:

Use of the Min Attribute:

Use of the Min and Max Attribute:

Submit

Use of the Required Attribute

! Please fill out this field.

Use of the Pattern Attribute:

! Please match the requested format.
Must use 3-letter values

Use of the Maxlength Attribute:

Use of the Min Attribute:

March 2003

Su	Mo	Tu	We	Th	Fr	Sa
23	24	25	26	27	28	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

Today

Use of the Min and Max Attribute:

! Value must be less than or equal to 5.

- Source: https://www.w3schools.com/html/html_form_attributes.asp



EXAMPLES OF CLIENT-SIDE VALIDATION

- One can also apply CSS styling.
- For example, we can make a red border around the required fields
- Another example: make a red border around input elements that are invalid, and green around those that pass client-side validation:

```
<style>
  input{ border: solid 1px; }
  input:required { border-color: red; }
</style>
```

Use of the Required Attribute:

Use of the Pattern Attribute:

Use of the Maxlength Attribute:

Use of the Min Attribute:

Use of the Min and Max Attribute:

```
<style>
  input{ border: solid 2px; }
  input:invalid { border-color: red;
                  background-color:yellow; }
  input:valid { border-color: lightgreen; }
</style>
```

Use of the Required Attribute:

Use of the Pattern Attribute:

Use of the Maxlength Attribute:

Use of the Min Attribute:

Use of the Min and Max Attribute:

VALIDATION THROUGH JAVASCRIPT

- How would you validate that an **input element** has a value that is an even number?
- For an **input** element, one can use the **oninput** attribute to call a method (an event handler) ← the **input** event
- How would you ensure that one **input element** contains a value larger than another **input element**?
- For a **form** element, one can use the **onsubmit** attribute to call a method (an event handler) ← the **submit** event
 - We can write the event handler to include any custom validation we may need.

```
<style>
  form { border: solid 1px; }
  input { border: solid 2px; }

  .invalid { border-color: red;
             background-color: yellow; }

  .valid { border-color: lightgreen; }
</style>
```

```
<form id = "myForm" action="/action_page.php" onsubmit="return validateMyForm()">
  <label for="input1">Enter an even number:</label><br>
  <input type="number" id="input1" oninput="return validateMyInput()"><br><br>

  <label for="input2">Enter a larger:</label><br>
  <input type="number" id="input2" required><br><br>

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

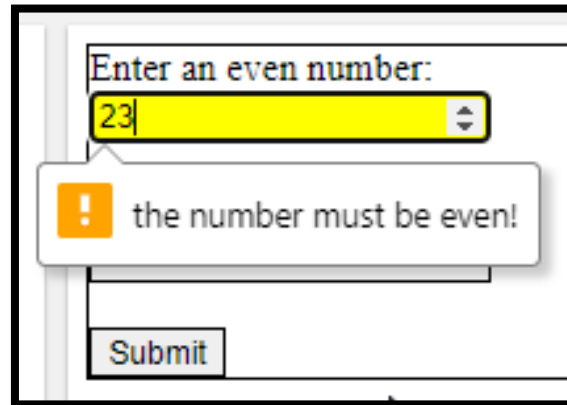
```
<script>
function validateMyForm(){
  if( document.getElementById("input2").value > document.getElementById("input1").value)
  {
    document.getElementById("myForm").className = "valid"
    return true;
  }
  else
  {
    document.getElementById("myForm").className = "invalid"
    return false;
  }
}

function validateMyInput(){
  if( (document.getElementById("input1").value) % 2 == 0)
  {
    document.getElementById("input1").className = "valid"
    return true;
  }
  else
  {
    document.getElementById("input1").className = "invalid"
    return false;
  }
}
</script>
```

- Alternatively, one can add an error message (add new elements to the page). We'll see later better ways to use validation.

VALIDATION THROUGH JAVASCRIPT

- On your lab, you'll also get to work with: **setCustomValidity()**
 - This method can be used to set the **validationMessage** property of an input element.
 - Call the **reportValidity()** method on the same input element to display the error



A screenshot of a web form. At the top, there is a label "Enter an even number:". Below it is a text input field containing the number "23". The input field has a yellow background. Below the input field, there is a red-bordered error message box with a red exclamation mark icon and the text "the number must be even!". At the bottom of the form, there is a "Submit" button.

Sources:

- https://www.w3schools.com/js/js_validation_api.asp
- <https://developer.mozilla.org/en-US/docs/Web/API/HTMLElement/setCustomValidity>

```
<!DOCTYPE html>
<html>
<head>
  <style>
    form { border: solid 1px; }
    input { border: solid 2px; }

    .invalid { border-color: red;
               background-color: yellow; }

    .valid { border-color: lightgreen; }
  </style>

  <script>
    function validateMyInput(){
      if( (document.getElementById("input1").value) % 2 == 0)
      {
        document.getElementById("input1").setCustomValidity("");
        document.getElementById("input1").className = "valid";
      }
      else
      {
        document.getElementById("input1").setCustomValidity("the number must be even!");
        document.getElementById("input1").className = "invalid";
        document.getElementById("input1").reportValidity();
      }
    }
  </script>
</head>
<body>

<form id = "myForm" action="/action_page.php">
  <label for="input1">Enter an even number:</label><br>
  <input type="number" id="input1" oninput="validateMyInput()"><br><br>

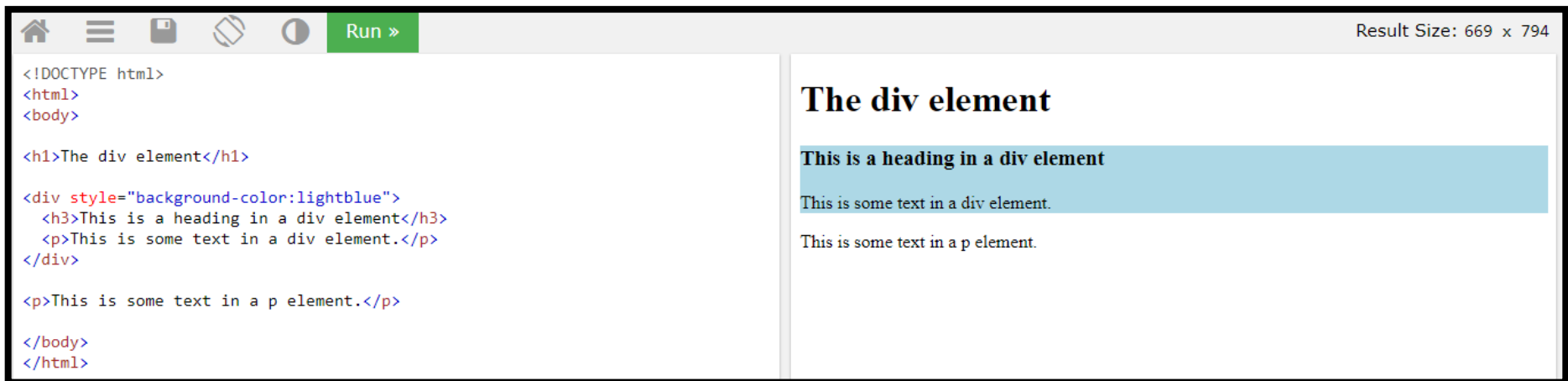
  <label for="input2">Enter a larger:</label><br>
  <input type="number" id="input2" required><br><br>

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

</body>
</html>
```

HTML BASICS – DIV: EXTRA

- **div**: defines a **division** or a **section** in an HTML document.
 - By default, browsers always place a line break before and after the <div> element. However, this can be changed with CSS
 - Source: https://www.w3schools.com/tags/tag_div.ASP
- Test it in here:
 - https://www.w3schools.com/tags/tryit.asp?filename=tryhtml_div_test
 - https://www.w3schools.com/tags/tryit.asp?filename=tryhtml_div_default_css



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

<h1>The div element</h1>

<div style="background-color:lightblue">
  <h3>This is a heading in a div element</h3>
  <p>This is some text in a div element.</p>
</div>

<p>This is some text in a p element.</p>

</body>
</html>
```

Result Size: 669 x 794

The div element

This is a heading in a div element

This is some text in a div element.

This is some text in a p element.



LAB SCENARIO

- For each lab, before performing the work, please make sure to read the given scenario.
 - This will help you understand what is it that you're trying to accomplish.
 - This will also help you understand what is the overall purpose of the lab and better understand the code.
- “The delegates who want to attend ContosoConf will need to register and provide their details. You have been asked to add a page to the ContosoConf website that implements an attendee registration form.
- The server-side code already exists to process the attendee data. However, the registration page performs minimal validation and is not user-friendly. You have decided to add client-side validation to the form to improve the accuracy of the registration data entered by attendees and to provide a better user experience”
- Source: https://github.com/MicrosoftLearning/20480-Programming-in-HTML5-with-JavaScript-and-CSS3/blob/master/Instructions/20480C_MOD04_LAB_MANUAL.md



IN-CLASS DEMO

Demonstration: Creating a Form and Validating User Input

- Source/Steps
- https://github.com/MicrosoftLearning/20480-Programming-in-HTML5-with-JavaScript-and-CSS3/blob/master/Instructions/20480C_MOD04_DEMO.md



PART OF DEMO

```
<form method="post" action="/registration/new" id="registration-form">
```

```
<!--
```

```
    TODO: Add form inputs
```

```
    FirstName - required string
```

```
    LastName - required string
```

```
    EmailAddress - required email address
```

```
    Password - required password string, at least 5 letters and numbers
```

```
    ConfirmPassword
```

```
    WebsiteUrl - optional url string
```

```
-->
```

```
<div class="field">
```

```
    <label for="first-name">First name:</label>
```

```
    <input type="text" id="first-name" name="FirstName" required="required" autofocus="autofocus"/>
```

```
</div>
```

```
<div class="field">
```

```
    <label for="last-name">Last name:</label>
```

```
    <input type="text" id="last-name" name="LastName" required="required"/>
```

```
</div>
```

```
<div class="field">
```

```
    <label for="email-address">Email address:</label>
```

```
    <input type="email" id="email-address" name="EmailAddress" required="required"/>
```

```
</div>
```

```
<div class="field">
```

```
    <label for="password">Choose a password:</label>
```

```
    <input type="password" id="password" name="Password" required="required" pattern="[a-zA-Z0-9]{5,}" title="At least 5 letters and numbers"/>
```

```
</div>
```

```
<div class="field">
```

```
    <label for="confirm-password">Confirm your password:</label>
```

```
    <input type="password" id="confirm-password" name="ConfirmPassword" required="required"/>
```

```
</div>
```

```
<div class="field">
```

```
    <label for="website">Website/blog:</label>
```

```
    <input type="url" id="website" name="WebsiteUrl" placeholder="http://"/>
```

```
</div>
```

```
<div>
```

```
    <button type="submit">Register</button>
```

```
</div>
```

```
</form>
```

```
<head>
```

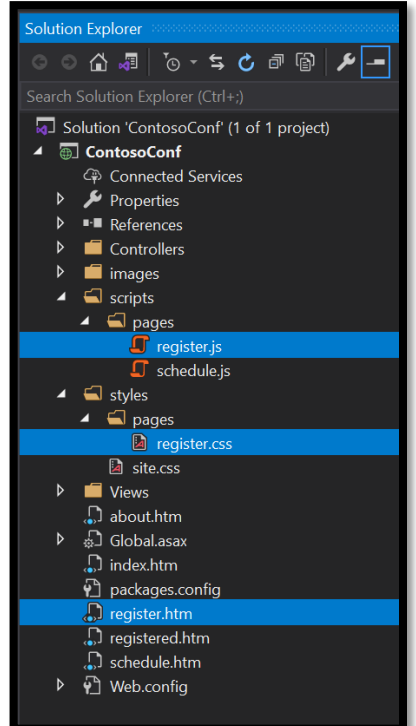
```
    <meta charset="utf-8"/>
```

```
    <title>Register for ContosoConf</title>
```

```
    <link href="/styles/site.css" rel="stylesheet" type="text/css" />
```

```
    <link href="/styles/pages/register.css" rel="stylesheet" type="text/css" />
```

```
</head>
```



```
    <script src="/scripts/pages/register.js" type="text/javascript"></script>
  </body>
</html>
```

PART OF DEMO

```
register.js  + - X
ContosoConf JavaScript Content Files  <global>

// Get the registration <form> element from the DOM.
const form = document.getElementById("registration-form");
const submitButton = form.querySelector("button");

// TODO: Task 1 - Get the password <input> elements from the DOM by ID
const passwordInput = document.getElementById("password");
const confirmPasswordInput = document.getElementById("confirm-password");

const checkPasswords = function () {
  // TODO: Task 2 - Compare passwordInput value to confirmPasswordInput value
  const passwordsMatch = passwordInput.value === confirmPasswordInput.value;

  // TODO: Task 3 - If passwords don't match then display error message on confirmPasswordInput (using setCustomValidity)
  // If passwords do match then clear the error message (setCustomValidity with empty string)
  if (passwordsMatch) {
    // Clear any previous error message.
    confirmPasswordInput.setCustomValidity("");
  } else {
    // Setting this error will prevent the form from being submitted.
    confirmPasswordInput.setCustomValidity("Your passwords don't match. Please type the same password again.");
  }
};

const addPasswordInputEventListeners = function () {
  // TODO: Task 4 - Listen for the "input" event on passwordInput and confirmPasswordInput.
  // Call the checkPasswords function
  passwordInput.addEventListener("input", checkPasswords, false);
  confirmPasswordInput.addEventListener("input", checkPasswords, false);
};

const formSubmissionAttempted = function() {
  form.classList.add("submission-attempted");
};

const addSubmitClickListener = function() {
  submitButton.addEventListener("click", formSubmissionAttempted, false);
};
```

Register for the conference

First name:

Last name:

Email address:

Choose a password:

Confirm your password:

Website/blog:

```
register.css  + - X

/* Styles for the register page */

.register form {
  max-width: 40rem;
}

.register .field {
  margin-bottom: 1rem;
}

.register label {
  display: block;
}

.register input {
  display: block;
  padding: .5rem;
  width: 100%;
  box-sizing: border-box;
}

.register button {
  padding: .5rem;
}

/* TODO: Task 5
Set invalid input elements background color to #f9b2b2,
when the form has the "submission-attempted" class */
.register form.submission-attempted input:invalid {
  background-color: #f9b2b2;
  outline: none;
}
```

EXTRA – DATA STRUCTURES ...

- Big Oh
- Singly linked Lists.
- If time, generics?



LAB/HOMEWORK

■ **Module 04**

- Exercise 1: Creating a Form and Validating User Input by Using HTML5 Attributes
- Exercise 2: Validating User Input by Using JavaScript

- You will find the **high-level** steps on the following page:

https://github.com/MicrosoftLearning/20480-Programming-in-HTML5-with-JavaScript-and-CSS3/blob/master/Instructions/20480C_MOD04_LAB_MANUAL.md

- You will find the **detailed** steps on the following page:

https://github.com/MicrosoftLearning/20480-Programming-in-HTML5-with-JavaScript-and-CSS3/blob/master/Instructions/20480C_MOD04_LAK.md

- For your homework submit one zipped folder with your complete solution.

