HTML5

WEEK 1

- What HTML is and how we got from the original version to HTML5
- The "magic" behind the Internet and how your web page isn't just one file, but many pieced put together by your browser using something called the Response/Request Cycle
- The syntax behind the tags -- how to write good syntactic HTML.
- The semantics behind the tags - some tags have special meaning and are extremely useful if you want to make your pages accessible. Accessible pages are ones that can be accessed by the widest range of people and that includes those who have physical or cognitive disabilities.
- Getting your page on the web - This class will not require you to post your site on the Internet. However, you will learn how to publish your site if you choose. (And we hope you do and share your work with me and others in the class.)

Video 01.01. History and evolution

HTML: Hypertext Markup Language

Not the same as programming language, use tag to annotate documents

In HTML the tags indicate where headings, images, lists, links, line breaks, and other components should go.

.html files

When your computer opens a .html file, it knows to open it in an Internet browser (Chrome, Firefox, Safari, etc.)
The browser can read this file and know how to display it on the screen.
Screen readers and other assistive devices can also utilize the HTML tags to present the information is special ways.

syntax, semantics

HTML1

HTML (1) was created in 1990 as a way to electronically connect documents via hyperlinks (hence a “web” of connections)

Mosaic:

In 1993, Mosaic emerged as the first graphical browser

WWW proliferates at a 341,634% annual growth rate of service traffic

Mosaic had challengers though in the form of Netscape (1994), IE (1995) and others

The Browser Wars

Browsers had proprietary tags

- <marquee>...</marquee> (scrolling text)
- <blink>...</blink> (blinking text).!

Other tags that went against the spirit of the original tenets of HTML were added, e.g. <font>, <center> and <bgcolor>
Origination of “Best viewed on” messages.

Web standards

No one "run" the Internet or the Web, some groups do take proactive roles:

- Internet Engineering Task Force (IETF)
- World Wide Web Consortium (W3C)
- The Web Accessibility Initiative (WAI)

Evolution of HTML

1993	HTML 1.0	Developed by Tim Berners-Lee to link document
1995	HTML 2.0	Developed by IETF RFC to include stylized text and tables
1996	CSS1	
1997	HTML 3.2	Developed by W3C and included browser specific feature
1997	HTML 4.0	A move back to normalizing the pages across platforms
1998	CSS2	
1999	HTML 4.1	Introduced different document types
2012	HTML 5	Back to HTML plus media and semantic tags

HTML5

HTML5 is a cooperation between W3C and the Web Hypertext Application Technology Working Group (WHATWG)

Established Guidelines:

- New features should be based on HTML, CSS, the DOM, JS
- Reduce the need for external plugins (Flash)
- More markup to replace scripting
- HTML5 should be device-independence

📌 Browser translate HTML documents into viewable webpges
HTML was intended to facilitate content types
When desingers want to do something new they write non-standard code to force browsers to do it.
New standards are written to handle new requirements and browser adopt to new standards

---

Video 01.03. How It Works

The Request/Response Cycle

Client/Server Relationship

Server:

- Servers are basically machines that hold all the resources.
- Our hope is that they're always connected to the network.

Clients:

- Machines for peronal use (laptops, phones, etc.)

Request/Response Cycle

- This is what happen when your computer (client) requests a page and a server responds with the appropriate files..

URL - Uniform Resource Locator

- Protocol - how to connect
- Domain - the server
- (optional) Document - the specific file needed (most pages are made up of multiple files)

Protocol:

- HTTP - Hypertext transfer protocol
- HTTPS - Secure Hypertext transfer protocol
- FTP - File transfer protocol

Domain names:

- Identifies the entity you want to connect to (umich.edu, google.com, etc.)
- Each has different top-level domain; Deteremined by Internet Corporation for Assigned Names and Numbers (ICAAN)

Domain name map into IP address

Internet Protocol Version 4 (Ipv4) uses number format of `xxx.xxx.xxx.xxx` to identify each domain → can represent over 4 billion unique combinations (2^32)
DNS looks up the domain and returns the IP address

IP address:

- Internet Protocol Version 6 (IPv6) is the communication protocol that identifies computers on networks
- Every computer has a unique IP address

`xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx where x can have 16 different values`

- Can represent over 300 trillion unique combination (2^128)

Domain Name Server (DNS)

- You don't need to remember the IP address of a domain
- The DNS will lookup the IP address based on the URL you type in

Document:

- URLs can specify a specific document

  http://www.intro-webdesign.com/contact.html

  http://www.intro-webdesign.com/Ashtabula.harbor.html

- If no document is specified, the default document is returned. (convention is *index.html)*

The request

Once the IP address is determined, the browser creates an HTTP request

Lots of hidden information in this request (header, cookies, form, data, etc.)

The response

The server returns files, not "web pages". It is up to the browser to decide what to do with those files

If the server can't fulfill the request it will send back files with error codes: 404 (File not found), 500 (Server down), etc.

Browser

So please remember that many times a page that doesn't work in one browser will work in another. It is a great time saver if you make sure to have two on your computer so that you can do a quick check if you ever get stuck on a site.

High-level domain name examples

Original	Country	Generic
.org	.au	.airforce
.net	.br	.biz
.int	.de	.community
.edu	.ie	.jobs
.gov	.uk	.travel
.arpa	.us	.wiki

📌 A URL has three parts.!
Request/Response cycle typically requires multiple rounds of communication between the client and server.

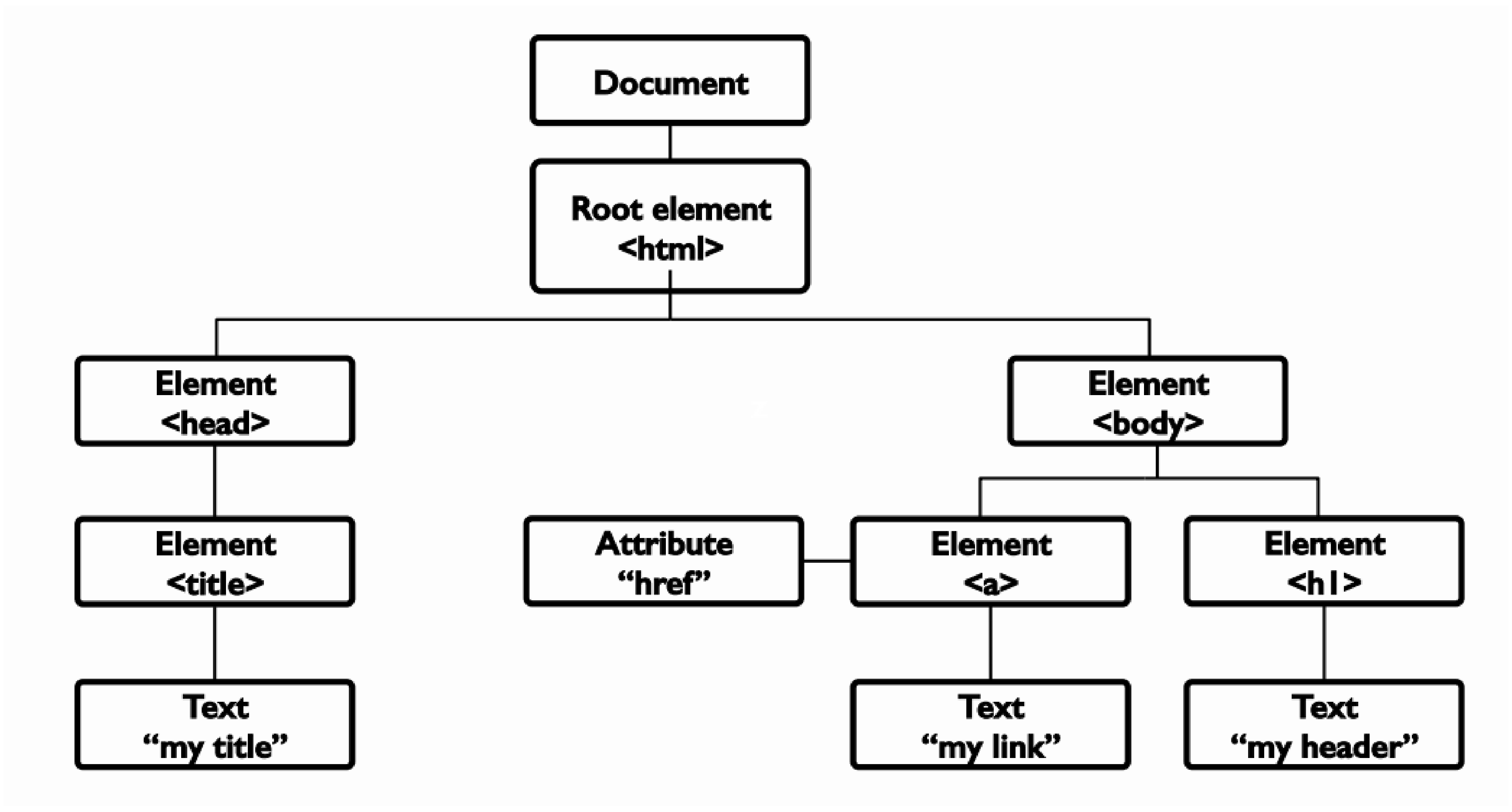
WEEK 2

Video 02.01 - The Document Object Model (DOM)

Basis of HTML5 is "New features should be based on HTML, CSS, the DOM, and Javascript,...)

DOM provides common **tree-like structure** that all pages should follow

Computer Scientist love trees (the mathematical kind) because you can test them



Three parts of a well-formed document

- Doctype: version of HTML that you will be using
- Head: Metadata
- Body: Displayable content

Doctype

HTML5: <!DOCTYPE html>

Previous versions dictated backwards compatibility:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org">

Head

Additional information used by the browser:

- Metadata - language, title
- Supporting files - Javascript, Styling, Add-ons

Other than title, meta-data is not displayed

Body

Bulk of your page

Important to write well-formatted (tree-like) code

Most of the content is dipslayed by the browser, but there may be some meta-data too.

Validating files

You can also validate file that aren't on the Internet.

📌 Well-formed pages use the DOM structure

- Use beginning and end tags
- Close inner tags before outer ones
- Use valid atrributes

Browser will "fix" bad code, but not always well. Use a validator to check your code

02.02 - HTML5 Tags and Syntax

HTML Reference
A free guide to all HTML elements and attributes.
https://htmlreference.io/

htmlreference.io
Free guide to all HTML5 elements and attributes

Special entities

< → &lt;

> → &rt;

© → &copy;

blank space → &nbsp;

¢→ &cent;

& → &amp;

02.03 - Semantic tags

`<header>` A group of introductory or navigational aids: title, navigation links, etc.

`<nav>` A section of the page that links to other pages or to parts within the page; Often found in the tag

`<footer>` A section that contains info such as copyright data, related documents, and links to social media

`<figure>` More semantics than. Can include:
+ caption
+ multiple multi-media resources

```
<figure>
  <img src="sunset.jpg" alt="">
  <figcaption>A sunset over Lake Erie.</figcaption>
</figure>
```

Other New Tags

Structural Elements	article, aside, main, menuitem, summary, section
Form Elements	datalist, keygen, output
Input Types	color, date, email, list
Graphic Elements	canvas, svg
Media Elements	audio, embed, source, track, video

02.06 - Hyperlinks

Anchor links

The <a> tag stands for anchor link

Needs a hyper-reference AND content:
• href: reference to location of new content
• content: the “clickable” part (text or image)

Types of links:
+ Abolute

+ Relative
+ Internal
+ Graphical

Using Images as the Link:
+ Make sure the clickable component has an informative name
+ Information in the images should be available to those who can't see the image

Target
+ _self - default action
+ _blank - open in new tab or window
+ _top, _parent

02.07 - Multimedia

HTML5 Multimedia:
+ Designed to avoid the use of extra software to play music/video
+ Not fully implemented

<video src="" height width autoplay loop controls> or embedded <source>

<audio autoplay controls loop buffered muted volume> use src attribute to link file (.mp3, .wav)

Settings clips

You can set both the video and audio elements to play clips by adding to the src attribute.
+ .ext#t=5, 25
+ .ext#t=, 39
+ .ext#t=, 1:38:45
+ .ext#t=42

Plugin

Before HTML5 there was no standard for video display, plugins were required

→ Flash

02.08 - Table

<th>..</th> -- table heading

Spanning Multiple Cells

combine multiple rows and/or columns using the `rowspan` and `colspan` attributes.

The Border Attribute

Captions
A heading (h2, h3, etc.) will look good, butdoesn't provide semantics. → Use <caption>

02.09 - Useful tabs

Generic: <p>, <div>

Semantic: <header>, <nav>, <footer>, <figure>

Block Tags

- Containers: <article>, <aside>, <section>, <main>, …
- <hr>
- <address>
- <blockquote> - has cite attribute
- <details> with <summary>

Inline tags

- <span> was the original inline tag for plain text
- <cite>
- <abbr>
- <time>
- <code>
- <sub> and <sup>

Tags that need “more”!

- <button>
- <meter>
- <progress>
- <iframe> – often used to embed documents
- <bdo> attribute dir (ltr or rtl)
- <map> with <area> -- creates “clickable element in image” but needs JavaScript

WEEK 3

03.01 - Accessible Web

What is web accessibility?

• Making the web accessible for the widest possible audience
• This audience includes Temporarily Able-Bodied users (TABs)
• Currently, online infrastructure is hostile to those with disabilities
• Inseparable from SEO, mobile, and usability: improve one and you improve the others
• Best way to accomplish accessibility? Adherence to standards.

W3C WCAG 2.0

W3C Web Content Accessibility Guidelines are principle-, not technology-based!

The four principles (POUR):
– Perceivable
– Operable
– Understandable
– Robust

Three approaches!
• Validate by URI!
• Validate by Filename!
• Validate by Direct Input

03.02 - Hosting your site

Requirements
• Domain name
• Hosting company

Domain names

• Typically purchased for multiple years at cheap rate
• Most common is .com, but other extensions are gaining acceptance
• Domain names are useless on their own

Hosting

What is your URL right now?!
You need a registered IP address to connect with your domain name
Hosting services vary
• Free

• Mid-range
• Full-service

03.05 - Using Secure File Transfer Protocol

SFTP – A faster way to transfer files

Secure File Transfer

A common way to transfer files is with FTP/SFTP

Flashback, what is a protocol?

Allows you to “drag and drop” as oppose to one-at-a-time file upload

What you need

Find/install a FTP client
• PC – WinSCP
• Mac – Fugu/Cyberduck
Find the ftp address for your host

> 📌 You can upload your files many ways!
> Make sure you know your login information

<a href= "shayhowe@awesome.com?subject=Hi&body=How%20are%20you">Email</a>

HTML tables

```
<table>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
</table>
```

Table Cells

Each table cell is defined by a `<td>` and a `</td>` tag.

`td` stands for table data.

Table Rows

Each table row starts with a `<tr>` and end with a `</tr>` tag.

`tr` stands for table row.

Table Headers

Sometimes you want your cells to be headers, in those cases use the `<th>` tag instead of the `<td>` tag:

Tag	Description
<table>	Defines a table
<th>	Defines a header cell in a table

Tag	Description
<tr>	Defines a row in a table
<td>	Defines a cell in a table
<caption>	Defines a table caption
<colgroup>	Specifies a group of one or more columns in a table for formatting
<col>	Specifies column properties for each column within a <colgroup> element
<thead>	Groups the header content in a table
<tbody>	Groups the body content in a table
<tfoot>	Groups the footer content in a table

Vertical Table Headers

```
<table>
  <tr>
    <th>Firstname</th>
    <td>Jill</td>
    <td>Eve</td>
  </tr>
  <tr>
    <th>Lastname</th>
    <td>Smith</td>
    <td>Jackson</td>
  </tr>
  <tr>
    <th>Age</th>
    <td>94</td>
    <td>50</td>
  </tr>
</table>
```

Header for Multiple Columns

use the `colspan` attribute on the `<th>` element

```
<table>
  <tr>
    <th colspan="2">Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

use the `rowspan` attribute on the `<th>` element

```
<table style="width:100%">
  <tr>
    <th>Name</th>
    <td>Jill</td>
  </tr>
  <tr>
    <th rowspan="2">Phone</th>
    <td>555-1234</td>
  </tr>
  <tr>
    <td>555-8745</td>
  </tr>
</table>
```

Zebra Stripes

```
tr:nth-child(even) {
  background-color: #D6EEEE;
}
```

```
/* vertical striped */
td:nth-child(even), th:nth-child(even) {
  background-color: #D6EEEE;
}
```

Table Colgroup

```
<table>
  <colgroup>
    <col span="2" style="background-color: #D6EEEE">
  </colgroup>
  <tr>
    <th>MON</th>
    <th>TUE</th>
    <th>WED</th>
    <th>THU</th>
...
```

HTML List Tags

Tag	Description
<ul>	Defines an unordered list
<ol>	Defines an ordered list
<li>	Defines a list item
<dl>	Defines a description list
<dt>	Defines a term in a description list
<dd>	Describes the term in a description list

HTML Forms

`<form>` element is used to create an HTML form for user input:

`<input>` element is the most used form element.

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

`<label>` tag defines a label for many form elements.

when the user clicks the text within the `<label>` element, it toggles the radio button/checkbox.

The `for` attribute of the `<label>` tag should be equal to the `id` attribute of the `<input>` element to bind them together.

The Submit Button

The `<input type="submit">` defines a button for submitting the form data to a form-handler.

The form-handler is typically a file on the server with a script for processing input data.

The form-handler is specified in the form's `action` attribute.

If the `name` attribute is omitted, the value of the input field will not be sent at all.

Form Attributes

The `action` attribute defines the action to be performed when the form is submitted.

The `target` attribute specifies where to display the response that is received after submitting the form.

Value	Description
_blank	The response is displayed in a new window or tab
_self	The response is displayed in the current window
_parent	The response is displayed in the parent frame
_top	The response is displayed in the full body of the window
framename	The response is displayed in a named iframe

The `method` attribute specifies the HTTP method to be used when submitting the form data.

The form-data can be sent as URL variables (with `method="get"`) or as HTTP post transaction (with `method="post"`).

The default HTTP method when submitting form data is GET.

The `autocomplete` attribute specifies whether a form should have autocomplete on or off.

The `novalidate` attribute is a boolean attribute.

List of All <form> Attributes

Attribute	Description
accept-charset	Specifies the character encodings used for form submission
action	Specifies where to send the form-data when a form is submitted
autocomplete	Specifies whether a form should have autocomplete on or off
enctype	Specifies how the form-data should be encoded when submitting it to the server (only for method="post")
method	Specifies the HTTP method to use when sending form-data
name	Specifies the name of the form
novalidate	Specifies that the form should not be validated when submitted
rel	Specifies the relationship between a linked resource and the current document
target	Specifies where to display the response that is received after submitting the form

Form Elements

The HTML `<form>` element can contain one or more of the following form elements:

- `<input>`
- `<label>`
- `<select>`
- `<textarea>`
- `<button>`
- `<fieldset>`
- `<legend>`
- `<datalist>`
- `<output>`
- `<option>`
- `<optgroup>`

The `<select>` element defines a drop-down list:

```
<label for="cars">Choose a car:</label>
<select id="cars" name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>
```

The `<option>` elements defines an option that can be selected.

By default, the first item in the drop-down list is selected.

To define a pre-selected option, add the `selected` attribute to the option:

The `<textarea>` element defines a multi-line input field (a text area):

```
<textarea name="message" rows="10" cols="30">
The cat was playing in the garden.
</textarea>
```

The `rows` attribute specifies the visible number of lines in a text area.

The `cols` attribute specifies the visible width of a text area.

The `<button>` element defines a clickable button:

```
<button type="button" onclick="alert('Hello World!')">Click Me!</button>
```

💡 **Note:** Always specify the `type` attribute for the button element. Different browsers may use different default types for the button element.

The `<fieldset>` element is used to group related data in a form.

The `<legend>` element defines a caption for the `<fieldset>` element.

```
<form action="/action_page.php">
  <fieldset>
    <legend>Personalia:</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="Submit">
  </fieldset>
</form>
```

The `<datalist>` element specifies a list of pre-defined options for an `<input>` element.

Users will see a drop-down list of the pre-defined options as they input data.

The `list` attribute of the `<input>` element, must refer to the `id` attribute of the `<datalist>` element.

```
<form action="/action_page.php">
  <input list="browsers" name="browser">
  <datalist id="browsers">
    <option value="Internet Explorer">
    <option value="Firefox">
    <option value="Chrome">
    <option value="Opera">
    <option value="Safari">
  </datalist>
  <input type="submit">
</form>
```

The `<output>` element represents the result of a calculation (like one performed by a script).

HTML Input Types

Here are the different input types you can use in HTML:

- `<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="button">`
- `<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">`

Input Restrictions

Attribute	Description
checked	Specifies that an input field should be pre-selected when the page loads (for type="checkbox" or type="radio")
disabled	Specifies that an input field should be disabled
max	Specifies the maximum value for an input field
maxlength	Specifies the maximum number of character for an input field
min	Specifies the minimum value for an input field
pattern	Specifies a regular expression to check the input value against
readonly	Specifies that an input field is read only (cannot be changed)
required	Specifies that an input field is required (must be filled out)
size	Specifies the width (in characters) of an input field
step	Specifies the legal number intervals for an input field
value	Specifies the default value for an input field

Range

```
<form action="/action_page.php" method="get">
  <label for="vol">Volume (between 0 and 50):</label>
  <input type="range" id="vol" name="vol" min="0" max="50">
  <input type="submit" value="Submit">
</form>
```

HTML Input Attributes

The input `value` attribute specifies an initial value for an input field

The input `readonly` attribute specifies that an input field is read-only.

The input `disabled` attribute specifies that an input field should be disabled.

The input `size` attribute specifies the visible width, in characters, of an input field.

The input `maxlength` attribute specifies the maximum number of characters allowed in an input field.

The input `min` and `max` attributes specify the minimum and maximum values for an input field.

The input `multiple` attribute specifies that the user is allowed to enter more than one value in an input field.

The input `pattern` attribute specifies a regular expression that the input field's value is checked against, when the form is submitted.

The input `placeholder` attribute specifies a short hint that describes the expected value of an input field (a sample value or a short description of the expected format).

The input `required` attribute specifies that an input field must be filled out before submitting the form.

The input `step` attribute specifies the legal number intervals for an input field.

The input `autofocus` attribute specifies that an input field should automatically get focus when the page loads.

The input `height` and `width` attributes specify the height and width of an `<input type="image">` element.

The input `list` attribute refers to a `<datalist>` element that contains pre-defined options for an <input> element.

The input `autocomplete` attribute specifies whether a form or an input field should have autocomplete on or off.