What is HTML?

- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

Use of HTML

- Web Pages Development
- Navigating the Internet
- Responsive Designs
- Storage Function in the Browser
- Data Entry Support
- Creation of Web Documents

Different Version of html

1. HTML 1.0

One of the first versions of HTML, HTML 1.0, was released in 1993, laying down the foundations for building web pages. Some of its primary characteristics were:

- Structuring Elements
- Comparatively Simpler
- Font Support

Nonetheless, HTML 1.0 initiated the field of web development, and its features kept advancing with every updated version.

2. HTML 2.0

The HTML 2.0 version was released in 1995 and had considerable improvements from the previous version. Some of them were:

- Standardization of HTML

- Forms
- Tables
- Formation of the W3C (World Wide Web Consortium)

3. HTML 3.2

HTML 3.2 was the next major successor to HTML 2.0 and was developed in 1997. The updated features included in it are:

- Upgraded Form Elements
- CSS Support
- Enhanced Image Features
- Extended Character Set

4. HTML 4.01

HTML 4.01, released in 1999, brought several advancements to the HTML language. Here are some of the updated features:

- CSS Linking
- New Tags
- Table Enhancement

5. HTML5

WHATWG released the initial public draft of **HTML5** in 2008, but it officially became a W3C recommendation on October 28, 2014. The version brought extensive support for new **HTML tags**. Furthermore, HTML5 provided support for new form elements like input elements of different types and geo locations support tags, etc.

Here are some key features and tags added in HTML5:

- New Form Elements
- Audio Tag
- Semantic Tags
- Section Tag

Structure of html

```
<!DOCTYPE html>

<html>
    <head>
        <title>Page title</title>

    </head>

    <body>
        <h1>This is a heading</h1>

        <p>This is a paragraph.</p>

        <p>This is another paragraph.</p>

    </body>

</html>
```

- The <!DOCTYPE html> declaration defines that this document is an HTML5 document
- The <html> element is the root element of an HTML page
- The <head> element contains meta information about the HTML page
- The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- The <body> element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The <h1> element defines a large heading
- The <p> element defines a paragraph

HTML Form

An HTML form is used to collect user input. The user input is most often sent to a server for processing.

- **The <form> Element**

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

<form> </form>

The <form> element is a container for different types of input elements, such as: text fields, checkboxes, radio buttons, submit buttons, etc.

- **The Action Attribute**

The action attribute defines the action to be performed when the form is submitted. Usually, the form data is sent to a file on the server when the user clicks on the submit button.

In the example below, the form data is sent to a file called "action_page.php". This file contains a server-side script that handles the form data:

➔ **Example**

On submit, send form data to "action_page.php":

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

Tip: If the action attribute is omitted, the action is set to the current page.

- **The Target Attribute**

The target attribute specifies where to display the response that is received after submitting the form. The target attribute can have one of the following values:

Value	Description
_blank	The response is displayed in a new window or tab
_self	The response is displayed in the same frame (this is default)
_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

➔ **Example**

Here, the submitted result will open in a new browser tab:

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

- **The Method Attribute**

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.

➔ **Example**

This example uses the GET method when submitting the form data:

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

➔ **Example**

This example uses the POST method when submitting the form data:

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

➔ **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

Tip: Always use POST if the form data contains sensitive or personal information!

- **The Autocomplete Attribute**

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

➔ **Example**

A form with autocomplete on:

<form action="/action_page.php" autocomplete="on">

- **The Novalidate Attribute**

The novalidate attribute is a boolean attribute. When present, it specifies that the form-data (input) should not be validated when submitted.

➔ **Example**

A form with a novalidate attribute:

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

HTML Inputs

The <input> Element

The HTML <input> element is the most used form element. An <input> element can be displayed in many ways, depending on the type attribute.

Here are some examples:

- <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">

Input Type Submit

<input type="submit"> defines a button for **submitting** form data to a **form-handler**. The form-handler is typically a server page with a script for processing input data. The form-handler is specified in the form's action attribute:

Example

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

If you omit the submit button's value attribute, the button will get a default text.

Input Type Radio

<input type="radio"> defines a **radio button**. Radio buttons let a user select ONLY ONE of a limited number of choices:

Example

```
<p>Choose your favorite Web language:</p>

<form>
  <input type="radio" id="html" name="fav_language" value="HTML">
  <label for="html">HTML</label><br>
  <input type="radio" id="css" name="fav_language" value="CSS">
  <label for="css">CSS</label><br>
  <input type="radio" id="javascript" name="fav_language" value="JavaScript">
  <label for="javascript">JavaScript</label>
</form>
```

Input Type Checkbox

<input type="checkbox"> defines a **checkbox**. Checkboxes let a user select ZERO or MORE options of a limited number of choices.

Example

```
<form>
  <input type="checkbox" id="vehicle1" name="vehicle1" value="Bike">
  <label for="vehicle1"> I have a bike</label><br>
  <input type="checkbox" id="vehicle2" name="vehicle2" value="Car">
  <label for="vehicle2"> I have a car</label><br>
  <input type="checkbox" id="vehicle3" name="vehicle3" value="Boat">
  <label for="vehicle3"> I have a boat</label>
</form>
```

Input Type Button

<input type="button"> defines a **button**:

Example

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

Input Type File

The <input type="file"> defines a file-select field and a "Browse" button for file uploads.

Example

```
<form>
  <label for="myfile">Select a file:</label>
  <input type="file" id="myfile" name="myfile">
</form>
```

The <textarea> Element

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

Example

```
<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 <select> Element

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

```
<!-- Select box -->
<label for="cars">Choose a car:</label>
<select id="cars" name="cars" size="1" multiple>
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi" selected>Audi</option>
</select>
<p>You can preselect an option with the selected attribute:</p>
<p>Use the size attribute to specify the number of visible values.</p>
<p>Hold down the Ctrl (windows) / Command (Mac) button to select multiple options.</p>
```

File Control with its Attribute

```
<input

  type="file"

  id="docpicker"

accept=".doc, image/*" />
```

accept :- The accept attribute value is a string that defines the file types the file input should accept. This string is a comma-separated list of unique file type specifiers.

Capture:- The capture attribute value is a string that specifies which camera to use for capture of image or video data, if the accept attribute indicates that the input should be of one of those types. A value of user indicates that the user-facing camera and/or microphone should be used.

Multiple:- When the multiple Boolean attribute is specified, the file input allows the user to select more than one file.

Webkitdirectory:- The Boolean webkitdirectory attribute, if present, indicates that only directories should be available to be selected by the user in the file picker interface. See HTMLInputElement.webkitdirectory

Name attribute

- **Definition and Usage**

  The name attribute specifies the name of an <input> element.

  The name attribute is used to reference elements in a JavaScript, or to reference form data after a form is submitted.

  Note: Only form elements with a name attribute will have their values passed when submitting a form.

Value attribute

The value attribute specifies the value of an <input> element.

The value attribute is used differently for different input types:

- For "button", "reset", and "submit" - it defines the text on the button
- For "text", "password", and "hidden" - it defines the initial (default) value of the input field
- For "checkbox", "radio", "image" - it defines the value associated with the input (this is also the value that is sent on submit)

Note: The value attribute cannot be used with <input type="file">.

Class Attribute

The HTML class attribute is used to specify a class for an HTML element. Multiple HTML elements can share the same class.

- The HTML class attribute specifies one or more class names for an element
- Classes are used by CSS and JavaScript to select and access specific elements
- The class attribute can be used on any HTML element
- The class name is case sensitive
- Different HTML elements can point to the same class name

- JavaScript can access elements with a specific class name with the getElementsByClassName() method

Using The class Attribute

The class attribute is often used to point to a class name in a style sheet. It can also be used by a JavaScript to access and manipulate elements with the specific class name.

Tip: The class attribute can be used on **any** HTML element.

Note: The class name is case sensitive

Using The id Attribute

The id attribute specifies a unique id for an HTML element. The value of the id attribute must be unique within the HTML document.

The id attribute is used to point to a specific style declaration in a style sheet. It is also used by JavaScript to access and manipulate the element with the specific id.

The syntax for id is: write a hash character (#), followed by an id name. Then, define the CSS properties within curly braces {}.

IMG tag

The <img> tag is used to embed an image in an HTML page.

```
<img src="img_girl.jpg" alt="Girl in a jacket" width="500" height="600">
```

Images are not technically inserted into a web page; images are linked to web pages. The <img> tag creates a holding space for the referenced image.

The <img> tag has two required attributes:

- src - Specifies the path to the image
- alt - Specifies an alternate text for the image, if the image for some reason cannot be displayed

Note: Also, always specify the width and height of an image. If width and height are not specified, the page might flicker while the image loads.

Tip: To link an image to another document, simply nest the <img> tag inside an <a> tag (see example below).

```
<a href="https://www.w3schools.com">
<img src="w3html.gif" alt="W3Schools.com" width="100" height="132">
</a>
```

A tag

The <a> tag defines a hyperlink, which is used to link from one page to another.

Visit W3Schools.com!

Visit W3Schools.com!

The most important attribute of the <a> element is the href attribute, which indicates the link's destination.

By default, links will appear as follows in all browsers:

- An unvisited link is underlined and blue
- A visited link is underlined and purple
- An active link is underlined and red

Tips and Notes

Tip: If the <a> tag has no href attribute, it is only a placeholder for a hyperlink.

Tip: A linked page is normally displayed in the current browser window, unless you specify another target.

Tip: Use CSS to style links: CSS Links and CSS Buttons.

Meta tag

The <meta> tag defines metadata about an HTML document. Metadata is data (information) about data.

<meta> tags always go inside the <head> element, and are typically used to specify character set, page description, keywords, author of the document, and viewport settings.

Metadata will not be displayed on the page, but is machine parsable.

Metadata is used by browsers (how to display content or reload page), search engines (keywords), and other web services.

There is a method to let web designers take control over the viewport (the user's visible area of a web page), through the <meta> tag (See "Setting The Viewport" example below).

```
<head>
 <meta charset="UTF-8">
 <meta name="description" content="Free Web tutorials">
 <meta name="keywords" content="HTML, CSS, JavaScript">
 <meta name="author" content="John Doe">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
```

What is responsive web site, how user can do it?

- Web pages can be viewed using many different devices: desktops, tablets, and phones. Your web page should look good, and be easy to use, regardless of the device.

- Web pages should not leave out information to fit smaller devices, but rather adapt its content to fit any device:

- It is called responsive web design when you use CSS and HTML to resize, hide, shrink, enlarge, or move the content to make it look good on any screen.

- Users are used to scroll websites vertically on both desktop and mobile devices - but not horizontally!

- So, if the user is forced to scroll horizontally, or zoom out, to see the whole web page it results in a poor user experience.

- Some additional rules to follow:

  1. Do NOT use large fixed width elements - For example, if an image is displayed at a width wider than the viewport it can cause the viewport to scroll horizontally. Remember to adjust this content to fit within the width of the viewport.

  2. Do NOT let the content rely on a particular viewport width to render well - Since screen dimensions and width in CSS pixels vary widely between devices, content should not rely on a particular viewport width to render well.

  3. Use CSS media queries to apply different styling for small and large screens - Setting large absolute CSS widths for page elements will cause the element to be too wide for the viewport on a smaller device. Instead, consider using relative width values, such as width: 100%. Also, be careful of using large absolute positioning values. It may cause the element to fall outside the viewport on small devices.