HTML

**INTRODUCTION TO HTML:**

* **HTML** (Hyper Text Markup Language) is the language used to create websites, it provides a logical way to structure content for websites

+ *Markup Language* is a computer language that defines the structure and presentation of raw text, it works by surrounding raw text that is wrapped in HTML elements with information the computer can interpret and marking it up to be processed

+ *HyperText* is text displayed on a computer or device that provides access to other text through links, also known as “hyperlinks”

+ **Current HTML standard : HTML5**

* **!DOCTYPE**

+ Web browsers must know what language a document is written in before they can process its content => you can let them know that you are using HTML by starting your document with a document type declaration which looks like this <!DOCTYPE html> - it tells the browser what type of document to expect and the version of HTML being used in the document.

+ **<!DOCTYPE html>** must be the first line of code in all of the HTML document (to makes sure that the document is forever interpreted correctly) => if not, the code might not work

+ the “HTML” will tell the browser that you will use HTML in the document but it doesn’t add any HTML structure or content

* To create the **HTML structure** and content, we must add opening <html> and closing **</html>** tags (if not the browser could incorrectly interpret the HTML code). Anything between will be interpreted as HTML code, the <html> element will contain all HTML code

Ex:

<!DOCTYPE html>

<html>

<html>

* **HTML Anatomy** (a diagram shows how HTML elements are structured)

+ the paragraph element is made up of one opening tag, the text and a closing tag <p> (<,> are angle brackets; element content is the information contained in between)

+ Most elements require both opening and closing tags, but some call for a single self-closing tag.

* **Head element**: Give the browser information

+ contains the metadata, which is the information about the page that isn’t displayed directly on the web page

+ the opening and closing head tags **(<head></head>)** appear as the first item after your first HTML tag.

* **Page titles:**

+ the browser displays the title of the page because the title can be specified directly inside of the <head> element, by using a **<title>** tag. => when we open a file containing

+ The title appears in a browser’s tab

* **The Body:**

+ Only the content inside the opening body tag and the closing body tag **(<body></body>)** can be displayed to the screen (visible)

+ The opening body tag <body> should be placed below the closing </head> tag and the closing body tag </body> should be placed above the closing </html> tag

+ Once the file has a body, many different types of content – including text, images, and buttons – can be added to the body

* **Self-closing tag:**

+ A few elements only require 1 tag instead of an opening and a closing tag => they **can’t warp around raw text or other elements**

+ Self-closing elementscontain all the information the browser needs to render the element inside a single tag

Ex: <br /> a line break

(<p>line one<br /> line two</p>) => can stand in the middle of a line

* **HTML structure:**

+  When an element is contained inside another element, it is considered the child of that element

ex: <p> element is considered as a child of the <body> element

<body>

<p>Paragraph</p>

</body

+ As there are more elements inside, it can be extended to grandchildren… the outer element is the biggest which contain the element which contain another element and so on

+ The inner elements will inherit the characteristics of the outer elements

* **Whitespace**

+ As the code in an HTML file grows, it becomes more difficult to know how elements are related => whitespace (xuống dòng) and indentation (thụt lề) are used to visualize the relationship between elements

+ basically, this make it easier to read the code (the browser will still read the code in the same way)

* **Indentation:**

+ 2 spaces of indentation show that the next element is nested in the element above

+ The spaces are inserted using the spacebar on your keyboard.

* **Comments: (Chú thích)**

+ Comments **begin with <!-- and end with -->**

+ The characters in between will not be displayed on the browser, only you can see it

+ Comments may help you understand your code if you decide to come back and review it at a much late date

+ They allow you to experiment with new code, without having to delete old code

**COMMON ELEMENTS**

* **Visible content**: All content added to HTML webpages should be between opening and closing tags, such as the **<body>**tag
* **Headings:**

+ Headings can be used for a variety of purposes, like titling sections, articles, or other forms of content.

+ There are 6 available headings in HTML, ordering from largest to smallest in size

* **<h1> -** used for main headings, all other headings are used for subheadings.
* **<h2>**
* **<h3>**
* **<h4>**
* **<h5>**
* **<h6>**
* **Text content tags**

To add blocks of text in HTML, you can use:

+ Paragraphs **<p>** simply contain a block of plain text

+ **<div>** can contain any text or other HTML elements. They are primarily used to divide HTML documents into sections

+ **<span>** contains short pieces of text or other HTML. They are primarily used to wrap small pieces of content that are on the same line as other content and do not break text into different sections.

+ In order to organize text into sections, we can add **“id”** into <div> => **id** and **div** are attributes

* **Text style tags:**

To signal that the text within them should be “emphasized” or “strong”, **<em>** and **<strong>** are used

+ the <em> tag will generally render as italic (chữ nghiêng) emphasis

+ the <strong> will render as bold (chữ in đậm) emphasis

* **Line breaks**

+ The line break element is a self-closing tag. You can use it anywhere within your HTML code and a line break will be shown in the browser.

+ element: **<br /> or <br>**

* **Unordered list:**

+ In order to display content in an easy-to-read list, you can use an unordered list tag **<ul>** to create a list of items in no particular order.

+ It outlines individual list items with **a bullet point**

+ the <ul> element can’t hold raw text or automatically format raw text into an unordered list of item, either

+ each item must have the tag **<li>** as the first tag of the line to make it become a list

* **Ordered list:**

+ Ordered list is just like unordered list, except that each list item is **numbered**

+ You create this list by using the **<ol>** tag and then add the **<li>** tag as the first tag of each line

* **Images:**

+ The **<img>** tag allows you to add an image to a web page

+ This is a self-closing tag

+ This tag has a required attribute called “**src”**

ex: **<img src= “...” />**

+ The **src** attribute must be set to the image's source, or the location of the image

* **Attributes:**

+ It provides more information about an element’s content

+ They live directly inside of the opening tag of an element

+ They are made up of 2 parts: its *name* and *value*

* **Image Alts:**

+HTML helps support visually impaired users (ng dùng khiếm thị) with the **alt** attribute

+ More specifically, these users often use screen reading software, therefore, when the **alt** attribute is included, this software may read the image’s description out loud to the user. However, if the image has no meaningful information, the **alt** won’t be helpful

+ The **alt** attribute is applied specifically to the **<img>** element

+ By using the **alt** attribute, if an image can’t load on a web page, the user can mouse over the area originally intended for the image and read the brief description

ex: <**img src= “…” alt = “…” />**

* **Videos:**

+ Like the <img> tag, the **<video>** tag also requires a **src** attribute with a link to the video source, the source must link to a video file, not a video on another site

+ This is not a self-closing tag, therefore, it needs an opening and a closing tag

+ After the **src** attribute, the **width** and **height** attributes are used to set the size of the video displayed in the browser

+ After those attributes, the **controls** attribute instructs the browser to include basic video controls such as pause, play and skip

+ If the video is unable to be loaded, the text “Video not supported” will be displayed

ex: **<video src=”…” width=”…” height=”…” controls>**

video not supported

**</video>**

* **Linking out:**

+ You can add link to a web page by adding an anchor element **<a>** and including the text of the link in between the opening and closing tag

+ This element needs the **href** attribute to work, this attribute will address to where a file is located

ex: **<a href=…>** The name of the site that is linked or a guide for users to know what the link is **</a>**

* **Linking at will**

+ By wrapping elements with an anchor element **<a>**, you can turn nearly any element into a link

ex: <a href=… ><img src=… alt=… /> </a>

* **New page:**

+ By putting the **target** attribute in **<a>** after the **href**, the link will open in a new window

+ This attribute specifies that a link should open in a new window/tab

+ For this attribute to work, it requires **\_blank, remember the “”**

ex: <a href=… **target=”\_blank”**> … </a>

* **Relative:**

+ When making multi-page static websites, web developers often store HTML files in the **root directory**, or a main folder where all the files for the project are stored.

+ As the size of the projects you create grows, you may use additional folders within the main project folder to organize your code

+ If the browser is currently displaying 1 file, it knows that others are in the same folder with this one, which is also called the current folder. Meanwhile, the other files may be linked using a ***relative path***

+ It shows a path to a file on the same website versus a full URL which is stored in a different folder

ex: <a href=./contact.html> Contact </a>

The **“./”** tells the browser to look for the file in the current folder

* In this example, the **<a>**tag is used with a relative path to link from the current HTML file to the **contact.html** file in the same folder. On the web page, **Contact** will appear as a link
* **Same page:**

+ In order to link to a target on the same page, we must give the target an **id**

**+** The **id** should be descriptive to make it easier to remember the purpose of a link

+ The target link should contain the **#** and **the id**

**ex: <a href=#...>…</a>** will jump to **<div id=…> (this is also why we need <div> and <id>)**

**By clicking on this link, the web will automatically jump to the position that you want**

* **Navigation:**

+ Linking to elements on the same page or to other pages on the same site is called navigation.

+ The tag **<nav>** can be used to **wrap links** in order to organize the content on the web page

+ While there are non-semantic tags such as <div> do not describe the content inside of them, semantic tags such as <nav> will describe the content and help us modify and style our content later

**A FULL LIST OF AVAILABLE HTML TAGS**

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element>

**TABLES:**

* **Reason:**

A table is usually the best way of presenting data such as prices, scores, invoice data,…

* **Create a table:**

+ The **<table>** elementwill contain all of the tabular data you plan on displaying

+ In a table, it’s predefined that it has rows, columns and cells that will hold data

* **Table rows:**

+ By using the element **<tr>**, the rows will be added into the table

ex: <table>

<tr>

</tr>

<table>

* In this example, 1 row has been added
* **Table data**

+ By using the element **<td>**, the data will be added

ex: <tr>

<td>…</td>

<td>…</td>

</tr>

* The data is entered in one row and 2 columns
* **Table headings:**

+ to add titles to rows and columns, you can use the table heading element **<th>**

+ just like table data, a table heading must be placed within a table row

+ in some situations, we need a blank heading to create an extra table cell to align the table headings correctly over the data they correspond to

+ this element goes with the attribute **<scope>**

* **<row>** - this value makes it clear that the heading is for a row
* **<col>** - this value makes it clear that the heading is for a column

ex: <tr>

<th></th>

<th scope= “col”>…</th>

</tr>

<tr>

</tr>

* **Table borders**

+ to create the border for the table, you can use **CSS**

ex: table, td{border:1px solid black;}

* **Spanning columns (expand the column)- hàng ngang**

+ data can span columns using the **<colspan>** attribute

+ the attribute accepts an integer (số nguyên tố) that is greater than or equal to 1 to and it’s also the number of columns it spans across.

* **Spanning rows (expand the row)- hàng dọc**

+data can span multiple rows using the **<rowspan>** attribute

+ the attribute also accepts an integer (greater than or equal to 1) which is also the number of rows it spans across.

* **Table body:**

+ To manage the table easier when it contains too much data, we can use the element **<tbody>** so that the table can be sectioned off

+ The element should contain all of the table’s data, excluding the table heading

* **Table head**

+ to section off table’s headings, we use **<thead>** element

* **Table footer**

+ The bottom part of a long table can be sectioned off using the **<tfoot>** element

* **Styling with CSS**

Ex: table, th, td{border:1px solid black;font-family:Arial,sanserif;text-align:center}