1. **FINISHED**
2. **Chapter 1: Getting started with HTML**
3. **Chapter 2: Doctypes**

* **Doctypes** - short for 'document type' - help browsers to understand the version of HTML the document is written in for better interpretability
* The <!DOCTYPE> declaration should always be included at the top of the HTML document, before the <html> tag
* HTML version ≥ 5 then declaration syntax: **<!DOCTYPE html>**

1. **Chapter 3: Heading**

- There are six heading tags. They are deﬁned with the <h1> to <h6> tags.

- Headings can be used to describe the topic.

**<h1>** deﬁnes the most important heading. **Syntax** **<h1>{Content}</h1>**

**<h6>** deﬁnes the least important heading.

1. **Chapter 4**: **Paragraphs**

* The tags define the paragraph in html: **<p>, <br>, <pre>**
* The HTML **<p>** element defines a paragraph (the browser will remove any extra spaces and extra lines when the page is displayed).
* The HTML **<br>** element defines a line break.
* The HTML **<pre>** element defines preformatted text (preserves both spaces and line breaks).

1. **Chapter 5**: **Text Formatting**

* HTML text formatting such as highlighting, bolding, underlining, subscript, and stricken text
* Highlighting: The **<mark>** element is new in HTML5 and is used to mark or highlight text in a document (A common standard formatting is black text on a yellow background).
* Bolding: use the **<strong>** or **<b>** tags to bold text.
  + Difference: **<strong>** used to indicate that the text is fundamentally or semantically important to the surrounding text. While **<b>** then not.
* Italic Text: use the **<em>** or **<i>** tags to italicize text.
  + Difference: **<em>** is used to indicate that the text should have extra emphasis that should be stressed. While **<i>** then not.
* Underline text: use the **<u>** tags to underline text
* Abbreviation: To mark some expression as an abbreviation, use **<abbr>** tag.
* Inserted, Deleted, or Stricken: use the **<ins>, <del>, <s>** tags to inserted, deleted or stricken.
* Superscript and Subscript: use the tags **<sup>** and **<sub>** to offset text either upward or downward

1. **Chapter 6: Anchors and Hyperlinks**

* **href**: Speciﬁes the destination address. It can be an absolute or relative URL, or the name of an anchor. Example**: <a** href="http://example.com/"**>**Link to example.com**</a>** (Link to another site/ absolute or relative URL)
  + **<h2** id="Topic1"**>**First topic**</h2>**
  + **….**
  + **<a** href='#Topic1'**>**Click to jump to the First Topic**</a>** (Link to an anchor)
* **ref**: Speciﬁes the relationship between the current document and the linked document. Example: **<a** href="http://example.com/" rel="external"**>**example site**</a>**
* **target:** Speciﬁes where to open the link**.**
  + Example:<a href="https://www.w3schools.com" target="\_blank">VisitW3Schools </a>
  + \_blank:Opens the linked document in a new window or tab
* **title:** Speciﬁes extra information about a link.
* We can use a relative path to link to pages on the same website
  + Example: **<a** href="/page"**>**Text Here**</a>**

1. **Chapter 7: List**

* Ordered List: Use ordinal sequences to indicate the order of list elements. Use the **<ol>** tag to create an ordered list, and each list item can be created with the **<li>** tag.
* Unordered List: Use a deﬁned symbol such as a bullet to list elements in no designated order. Use the **<ul>** tag to create an ordered list, and each list item can be created with the **<li>** tag.
* Description List: Use indents to list elements with their children.
* We can nest lists to represent sub-items of a list item

1. **Chapter 8: Tables**

* The HTML **<table>** element allows web authors to display tabular data (such as text, images, links, other tables, etc.) in a two dimensional table.
* The **<tr>** tag to create rows, the **<th>** tag to create header column and the **<td>** to create data column. (We can put whatever you want inside a **<td>** or **<th>**.)
* Table cells can span multiple columns or rows using the **colspan** and **rowspan** attributes. These attributes can be applied to <th> and <td> elements
* Column Groups: Sometimes you may want to apply styling to a column or group of columns. Or for semantic purposes, you may want to group columns together. To do this, use <colgroup> and <col> elements.
  + The **<colgroup>** tag specifies a group of one or more columns in a table for formatting.
  + The **<col>** tag specifies column properties for each column within a <colgroup> element.
* Table with thead, tbody, tfoot, and caption:
  + Caption: The **<caption>** tag defines a table caption
  + Thead: The **<thead>** tag is used to group header content in an HTML table.
  + Tbody: The **<tbody>** tag is used to group the body content in an HTML table.
  + Tfoot: The **<tfoot>** tag is used to group footer content in an HTML table.

1. **Chapter 9: Comments**
2. **Chapter 10: Class and IDs**

* *Classes* are identiﬁers for the elements that they are assigned to.
* The *class* attribute can be used on one or more tags**.** The *ID* attribute of an element is an identiﬁer which must be unique in the whole document.
* Acceptable Values:
  + ID: It must be unique in the document.
    - It must not contain any space characters.
    - It must contain at least one charactes.
  + Class: The rules for classes are essentially the same as for an id.
* Problems related to duplicated IDs: Having more than one element with the same ID is a hard to troubleshoot problem. The HTML parser will usually try to render the page in any case. Usually no error occurs. (But JavaScript fails to handle both elements have more than IDs )

1. **Chapter 11: Data Attribute**

* Data attributes introduced in HTML5, supported by all modern browsers, but older browsers are not.
* Although data attributes are not supported in older browsers, they still work and you can set and retrieve them using the same generic JavaScript *setAttribute* and *getAttribute* methods.
* Data Attribute Use:
  + HTML5 data-\* attributes provide a convenient way to store data in HTML elements. The stored data can be read or modiﬁed using JavaScript.
  + The data-\* attribute gives us the ability to embed custom data attributes on all HTML elements.
  + The stored (custom) data can then be used in the page's JavaScript to create a more engaging user experience.
  + The data-\* attribute consist of two parts:
    - The attribute name should not contain any uppercase letters, and must be at least one character long after the prefix "data-"
    - The attribute value can be any string

1. **Chapter 12: Linking Resoources**

* Javascript:
  + Synchronous: Standard practice is to place JavaScript **<script>** tags just before the closing **</body>** tag. Loading your scripts last allows your site's visuals to show up more quickly and discourages your JavaScript from trying to interact with elements that haven't loaded yet.
  + Asynchronous:
    - async: Speciﬁes that the script shall be executed asynchronously
    - defer : Speciﬁes that the script shall be executed when the page has ﬁnished parsing (only for external scripts). This attribute does not require any value.
* CSS
* Favicon:

1. **Chapter 13: Include JavaScript Code in HTML**

* Handling disabled Javascript: It is possible that the client browser does not support Javascript or have Javascript execution disabled, perhaps due to security reasons. To be able to tell users that a script is supposed to execute in the page, the **<noscript>** tag can be used. The content of **<noscript>** is displayed whenever Javascript is disabled for the current page.

1. **Chapter 14: Using HTML with CSS**
   1. External Stylesheet Use:

* using an external CSS file.
  1. Internal StyleSheet:
* using a **<style>** element in the **<head>** section
  1. Inline Style:
* using the style attribute in HTML elements
  1. Multiple Stysheets:
* It's possible to load multiple stylesheets:

**<head>**

**<link** rel="stylesheet" type="text/css" href="general.css">

**<link** rel="stylesheet" type="text/css" href="specific.css">

**</head>**

* Note that later ﬁles and declarations will override earlier ones

**Prioritize in css:** <https://evondev.com/do-uu-tien-trong-css/>

1. **Chapter 15: Image**

* Image tags do not have closing tags. The two main attributes you give to the img tag are src, the image source anh alt, which is alternative text describing.
* src: Speciﬁes the URL of the image
* alt: Alternative text that should be displayed if for some reason the image could not be displayed.
* Responsive image using picture element:
  + HTML5 introduced the **<picture>** element to add more flexibility when specifying image resources.
  + The **<picture>** element contains a number of **<source>** elements, each referring to different image sources. This way the browser can choose the image that best fits the current view and/or device.
  + Each **<source>** element have attributes describing when their image is the most suitable.

1. **Chapter 16: Image Map**

* An image maps is an image with clickable areas that usually act as hyperlinks.
* The **<map>** tag defines an image-map. An image-map is an image with clickable areas.
* The image is deﬁned by the **<img>** tag, and the map is deﬁned by a **<map>** tag with **<area>** tags to denote each clickable area. Use the usemap and name attributes to bind the image and the map.
* The coords attribute specifies the coordinates of an area in an image-map.
  + The coords attribute is used together with the shape attribute to specify the size, shape, and placement of an area.
  + shape attribute value: “rect”(rectangle), “circle”, “poly”(polygon).

1. **Chapter 17: Input Control Elements**

* A key component of interactive web systems, **<input>** tags are HTML elements designed to take a speciﬁc form of input from users.
* type: Identiﬁes the type of input control to display. Acceptable values are hidden, text, tel, url, email, password, date, time, number, range, color, checkbox, radio, file, submit, image, reset, and button. Defaults to text if not speciﬁed, if the value is invalid, or if the browser does not support the type speciﬁed
  + text: The most basic input type and the default input if no type is speciﬁed. This input type deﬁnes a single-line text ﬁeld with line-breaks automatically removed from the input value. All other characters can be entered into this.
    - He default width of a text ﬁeld input is 20 characters.
    - This can be changed by specifying a value for the size attribute.
    - size: distinctly diﬀerent than setting a width with CSS.
      * **Note:** Using the size attribute does not inherently limit the number of characters which can be entered into the box, only how wide the box is displayed. For limiting the length, see Input Validation.
      * If you need a multi-line text input for substantial amount of text, use a **<textarea>** element instead.
  + Checkbox and radio buttons:
  + Input validation:
    - Some newer input types (like email, url, tel, date and many more) are automatically validated and do not require your own validation constraints.
    - required attribute to indicate that a ﬁeld must be completed in order to pass validation.
    - minlength and maxlength attributes to indicate length requirements.
    - pattern attribute to specify any regular expression that must be matched in order to pass validation. We can also specify a title, which is included in the validation message if the ﬁeld doesn't pass.
    - accept attribute for input ﬁelds of type file, it is possible to accept only certain types of ﬁles, such as videos, images, audios, speciﬁc ﬁle extensions, or certain media types
  + color: Show a color picker.
  + password: creates a single-line text ﬁeld similar to the input type=text, except that text is not displayed as the user enters it.
    - placeholder text is shown in plain text and is overwritten automatically when a user starts typing.
  + file: File inputs allow users to select a ﬁle from their local ﬁlesystem for use with the current page. If used in conjunction with a form element, they can be used to allow users to upload ﬁles to a server (for more info see Uploading Files).
    - Adding the multiple attribute the user will be able to select more than one ﬁle
  + button: can be used for triggering actions to occur on the page, without submitting the form.
    - **<button>** element require a button that can be more easily styled or contain other elements.
      * The name of the button, which is submitted with the form data.
      * Inside a **<button>** element you can put content, like text or images. This is the difference between this element and buttons created with the <input type=”button”> element.
  + submit: creates a button which submits the form it is inside when clicked.
  + reset: An input of type reset creates a button which, when clicked, resets all inputs in the form it is contained in to their default state
  + tel:
  + search: Input type search is used for textual search. It will add magniﬁer symbol next to space for text on most browsers

1. **Chapter 18: Form**

* The HTML <form> element defines a form that is used to collect user input.
  1. The Action Attribute
  + The action attribute defines the action to be performed when the form is submitted.
  + Normally, the form data is sent to a web page on the server when the user clicks on the submit button.
  + In the example above, the form data is sent to a page on the server called "/action\_page.php". This page contains a server-side script that handles the form data:

<form **action="/action\_page.php**">

* + If the action attribute is omitted, the action is set to the current page.
  1. The Method Attribute
  + The method attribute specifies the HTTP method (GET or POST) to be used when submitting the form data.
  1. More attributes
  + accept-charset: Speciﬁes the character encodings that are to be used for the form submission.
  + 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").
  + name: Specifies the name of a form.
  + novalidate: Specifies the form should not be validated when submitted
  1. Target attribute in form tag.
  + The target attribute specifies if the submitted result will open in a new browser tab, a frame, or in the current window.
  + The default value is "\_self" which means the form will be submitted in the current window.
  1. Uploading Files
  2. Grouping a few input fields:
  + While designing a form, you might like to group a few input ﬁelds into a group to help organise the form layout

1. **IN PROGRESS**
2. **ISSUES**
3. **NEXT PLAN**
4. **SOURCE CODE**