Section 3: Introduction To HTML

28th March 2020

Contents

[1. HTML Basics 2](#_Toc37766849)

[2. Introduction to Mozilla Developer Network 2](#_Toc37766850)

[3. HTML Boilerplate And Comments 3](#_Toc37766851)

[4. Basic Tags 5](#_Toc37766852)

[5. Lists 7](#_Toc37766853)

# HTML Basics

Hypertext Markup Language is the structure and content of the page, not the styling! It was produced between 1989 and 1990 back in the day when academics were exchanging research papers that had structured formatting. Back then there was no standardised way of exchanging these across the web. HTML was developed in response to the need for a way for researchers to send papers to each other in a format that allowed for the structure of the papers to remain consistent in structure and content.

HTML works in tags, it allows developers to designate what content is, and how it is structured (for example, this is a bullet point, this is a link, this is a paragraph). A big contribution was the inclusion of document linking via hyperlinks (It wasn’t revolutionary, but it suited what the language set out to do well).

HTML is built on the idea of <tags> </tags>. Content is sandwiched between an opening tag (<>) and a closing tag (</>). Each tag has a tag name to specify how the sandwiched content is to be structured. Tags can also have attributes to give each element additional properties (such as a href to another source or document).

For example, if we want a heading, we could do:

<h1>Abstract</h1>.

Or a paragraph:

<p>This is a paragraph tag. It is used to separate individual paragraphs.</p>.

# Introduction to Mozilla Developer Network

Mozilla Developer Network (or MDN) is a useful resource for guidance around HTML, CSS and JavaScript. Find it at <https://developer.mozilla.org>. The notes for this are enclosed in a separate document.

# HTML Boilerplate And Comments

First tag to write is the basic header 1 tag. This can be written as follows:

<h1>This is my first tag</h1>

This will now display a header when the file is loaded as shown below:

* Though this markup is valid in context of the language syntax, it is however, invalid by itself. All HTML documents need to implement the same boilerplate framework, consisting of a DOCTYPE, a <html>, a <head> and a <body> set of tags as shown below:

<!DOCTYPE html>

* + <html>
    - <head>
      * <title></title>
    - </head>
  + <body>
    - <h1>This is my first tag</h1>
  + </body>
  + </html>

**Note: In the Sublime editor, the easiest way to have this automatically implemented is to write “html” then press tab.**

* The following contains a brief description of each section of the boilerplate. These are explained in more detail in the Mozilla Developer Network notes for this section:
* <!DOCTYPE>: Doctype is an artifact of the old days of HTML where all HTML documents required a DOCTYPE to identify the page as HTML and a link to the set of HTML rules the page was adhering to. Nowadays, a HTML page just requires a Doctype to function as expected and to indicate the page is implementing HTML. The code below is the shortest valid DOCTYPE that can be implemented in order to ensure the page runs as expected.
* <html>: The HTML root element. The base of all HTML documents. All other elements are descendants of the root element.
* <head>: Metadata for the HTML doc goes here. This contains the "data about the data" such as the title of the page, the author, and the character set being used. Links to scripts (JavaScript) and style sheets (CSS) are also provided here.
* <body>: The body is where the HTML used to represent physical content to be displayed on the page goes.
* Some tags must have an opening tag and a closing tag. Empty tags (like the DOCTYPE example) only require one tag and contain attributes and data inside it.

Elements are defined as an object consisting of tags, and the content that is wrapped by the tag/s.

* Comments allow developers to document what certain markup does and why it has been written. Comments are not read by the browser and are not displayed as part of the content. In HTML comments are implemented with arrow like wrappers like the example below:

<!— This is a comment. It doesn’t do anything! -->

* A nifty trick for sublime is to highlight the line of text you want to comment and hit CTRL+/ to comment it out automatically.
* The <title> is the title of the document as a whole. It is used in the browser tab to display what the page is, and is used by search engines to display a page title in it’s search results.

# Basic Tags

* This section details some of the commonly used tags in a HTML document.
* MDN Element Refence (<https://developer.mozilla.org/en-US/docs/Web/HTML/Element>)
  + Contains all elements. About 100. Not all required, just look them up when required!
* Heading tags. Used to represent a main heading for the page, and then subheadings for hierarchical groups of content on the page. Ideally there is only one <h1> tag on the page (i.e. The page title). The developer should aim to use no more than three types of heading tag unless unavoidable. These tags should also follow a hierarchical pattern that makes sense (i.e. Use headings in order of size. Don't have a <h1> resting below a <h3>) as this will end up confusing the user.
  + Header tags are known as block level elements. Each section of content wrapped in these tags starts on it's own line on the document.
* <p> tags are paragraph tags. They are used to separate individual blocks of related content. In traditional media, this was blocks of text, however in HTML this can be anything from groups of images to form fields. These are block tags
* <b> and <i> tags are the bold and italic tags respectively. They are used to style text, though have been replaced in HTML 5 by the <strong> and <em> tags and help to provide semantics to content.
  + These are inline tags. Inline tags operate on a line of content rather than a block. They can embed within a block level tag but cannot contain a block level tag themselves.

The code below demonstrates the implementation of the tags discussed above:

<h1>HTML Basics</h1>

<h2>This is another heading</h2>

<h3>This is a smaller heading</h3>

<p>

This is some text. blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah sdjdskjhffkjdf hadfh asdkfhsadkjf hfksdahfsadkhdaskfj asdh fsadk

</p>

<p>

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut tincidunt venenatis lectus sit amet aliquet. Fusce hendrerit, eros et congue suscipit, enim mi finibus neque, ut facilisis orci ligula vehicula lectus. Phasellus imperdiet, lorem volutpat sodales tempor, quam urna ornare ante, quis suscipit erat magna non ex. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce vitae arcu est. Vestibulum porttitor mi id elementum sollicitudin. Pellentesque porta in sem quis lobortis. Curabitur dictum, magna eget malesuada suscipit, diam massa gravida leo, ac tempus neque sapien sed leo. Vestibulum blandit ullamcorper imperdiet. Aenean et elit diam.

</p>

<p>

Quisque at nibh lobortis, <b>sollicitudin eros nec</b>, feugiat erat. Praesent imperdiet blandit semper. <i>Sed sed tempus ex, at semper orci</i>. Fusce malesuada lacinia ligula vitae volutpat. <strong>Etiam volutpat augue vitae nibh interdum, in dignissim urna pulvinar</strong>. Ut dictum <em>justo id ultricies efficitur</em>. Nunc consectetur vel orci id porta. Praesent vel dignissim felis. Nam aliquet odio sit amet mauris aliquam finibus.

</p>

# Lists

* This section details how to make unordered lists and ordered (numbered) lsits.
* Ordered lists <ol> are block level elements used to display lists of content that take some form of ordered structure (for example, a list of instructions or directions). Ordered lists represent each item in the list with the list item (<li>) block level elements.
* Unordered lists <ul> are used to represent any unstructured list of items (such as a list of cars, ingredients for a recipe, the members of a team). They too also use list items.
* Any other inline level elements can be used (such as strong or em) to style each list item.
* Lists can be nested in other lists in instances where each list item has it’s own sub-categories or sub-steps.
  + **IMPORTANT: To comply with W3 standards, nested lists must be enclosed within each list item!**

This is demonstrated below:

<h3>Ordered Lists</h3>

<ol>

<li>Open The Door.</li>

<li>Put The Clothes In.</li>

<li>Add The Detergent.</li>

<li>

Close The Door

<ul>

<li>Make Sure The Door Is <strong>Closed Properly!</strong></li>

<li>Make Sure The Machine Is <strong>Not Overloaded!</strong></li>

</ul>

</li>

<li>Start The Washing Machine</li>

</ol>

<h3>Unordered Lists</h3>

<ul>

<li>Joy Division</li>

<li>Interpol</li>

<li>Nine Inch Nails</li>

</ul>

# Divs And Spans

Divs and spans are simply a generic means of grouping content together by wrapping a section of content. This can be applied to both block (div) and inline elements (span). They ideally should not be replaced with the existing semantics provided by the HTML syntax such as <section>, <article>, <nav>, <main> and <aside> tags for block level content, and <em>, <strong>, <abbr>, <datetime> etc. for inline content.

The following markup is an example:

<!DOCTYPE html>  
<html>  
<head>

<title>Divs And Spans</title>

<link rel="stylesheet" type="text/css" href="Divs\_And\_Spans.css">

</head>

<body>

<h1>Divs And Spans</h1>

<div>

<h2>Article 1</h2>

<p>

Lorem <span>ipsum dolor sit amet</span>, consectetur adipisicing elit, sed doeiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

</p>

</div>

<hr>

<div>

<h2>Article 2</h2>

<p>

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

</p>

</div>

</body>

</html>

# Attributes

Attributes take the value of a key value pair. That is a property and a value. MDN has a comprehensive reference for all the attributes.

## Images

The <img> tag is used to insert images onto webpages. These use the “src” attribute to set the source of the image where the browser goes to request the file. Img are self-closing tags. They do not require a closing tag. Images can be local to the directory or from a specific site. The image tag takes the following form:

<img src=”url”>

## Links

The <a> tag is used to insert hyperlinks into a document. These re-direct the user to a new webpage. The format is as follows:

<a href=”url”>This Is The Link That Appears In The Document<a/>

The <a> element is an inline element. It follows on from whatever came last.

Links to external sources explicitly use [http://](NULL) in the address. Links to elements hosted locally do not require http and can be constructed relative to where the file is in the file structure. It is advisable to use relatively links where possible for improved performance.

A “target” attribute can be added. If the value is set to “\_blank”, the link opens in a new tab.