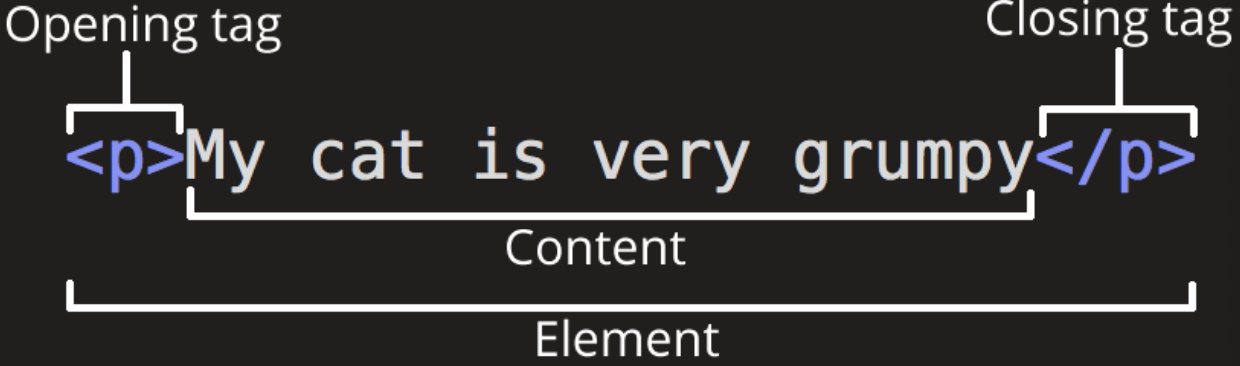
HTML

Overview

Hyper Text Markup Language (HTML) is the language used to tell a computer browser how to structure the webpages visited. It consists of a series of different elements enclosed in <> which are used to wrap/markup different parts of content, making them appear in a certain way. This is how webpages have images, links, buttons, forms, etc.

Tags



Opening tag - Opens with ‘<’ contains the name of the element, any elements attributes such as ID etc and then indicates where the elements starts to take effect with ‘>’

Content - The inner HTML of the element, can contain any other HTML such as text, or buttons

Closing tag - same as opening tag but has a slash before the element name, however some elements are empty, such as an image so require no closing tag.

Nesting

When elements are placed inside other elements, it is called nesting. Make sure that elements are nested properly and closed before and closing tags which they are nested inside.

Levels

There are two categories of element in HTML:

* Block-Level Elements - form visible block on the page. These elements will appear on a newline no matter what element is before it. The block level elements tend to be structual elements on the page, such as paragraphs, lists, footers, etc
* Inline Elements - Those which are contained in a block-level element and surround small parts of content, not paragraphs. Inline elements do not cause a newline to appear. E.g. <span> (text), <a> element (hyperlink), <em> (emphasis) or <strong> (bold).

Attributes

Attributes contain extra information of an element which you don’t want to appear in the inner HTML. Some common attributes are:

* class=”<name”
* id=”<anme>”

For an attribute to be correctly recognised, it must have a space either between it and another, and have its value placed between apostrophes (single or double depending on style). If a quote wants to be included inside the attribute value, HTML entities must be used to escape the charater.

For boolean attributes it is possible to make just include the attribute it is required to be true, no need for assignment to true.

Hyperlinks

For the <a> element the following attributes are added:

* title = gives title
* href = gives link
* target = provides browsing context, ie ‘\_blank” will display the link in a new tab

Document

<!DOCTYPE html> is required at the top of all to allow for error checking.

<html> whole doc </html> html tags wrap the entire document

<head> the head element acts as a container for all the items which are to be included on the page and make it work. This means items like CSS, keywords, and page descriptions, not anything which it to be actually displayed on the page.

<meta carset=”utf-8> specifys the charater set for the html document, required to include emojis etc

<title> sets title of page which will be displayed in the browser tab

<body> contains all the content which is to be displayed: text, images, video, javascript

Whitespace

When writing HTML documents whitespace is parsed down to one space, no matter how much is there. Therefore, formatting like tabs, paragraphs, etc have no effect on what is actually displayed.

Whitespace is generally used to increase readibility for a user, with tabs showing nested objects.

Special Charaters

The charaters < > “ ‘ &, are all special charters in html since they are used in the syntax. Therefore to use the literal version of these in html text they must be escaped as references:

* < - &lt;
* > - &gt;
* “ - &quot;
* ‘ - &apos;
* & - &amp;

Comments

Comments can be added using the following syntax:

start of comment =>. *<!-- some text*

end of comment => *-->*

HTML Parsing Process

Once donwloaded HTML is parsed progressively from top to bottom. Once the parser comes across an asset, the browser will attempt to download and run it, in most browsers only 2 requests are processed in parallel. Since the assests are downloaded progressivley two factors must be taken into account:

* Requests download and run large files will block the script - therefore large files/scripts which take time to run should either be put into async mode, or placed just before the closing tag of the html body. Therefore they will not hold up the parasing of the rest of the html.
* Since parsing is progressive, if a script which is requested and run refers to an element which has not yet be parsed and added to the DOM, it will not be able to find the element and error. Therefore similar to large downloads, add these files at the bottom of the html body, or use the event listener for page load to run the scripts.

CSS files should be requested in the top of the header since they are rendered at the sametime as the DOM and are therefore required early on for the page to render correctly.

The parsings of the DOM/CSS is the most intensive for the CPU, so to make webpage loading quick and responsive, make sure to use the minimum html elements possible, and most efficient CSS instructions, i.e. removing redundant ones.

Base HTML and requests to external sources should be kept to a minimum to improve response time.

DOM

The Domain Object Model (DOM) is a programming interface for HTML and XML documents, containing nodes and objects which represent the page so programs can change the documents structure, style and content.

All the properties, methods and events available for manipulating and creating webpages are organised into element objects, with the document object representing the itself allowing for them to be accessed and manipulated using a language like JavaScript.

Datatypes in the DOM:

* Document - References the root document object itself (DOM)
* Node - Every object located within a document is a node of some kind, so every object is a node and some type of node such as element, text, or attribute
* Element - An element is a specific type of node which can be directly specified in HTML with a HTML tag (div or section), can have properties such as id or class, and can have children, etc.
* NodeList - Array of elements like that returned by the method document.getElementByTagName(), accessed by index.
* NamedNodeMap - Array but items are acced by name or index

Brief DOM interface (there are many more methods):

* document.getElementById(id)
* document.getElementsByTagName(name)
* document.createElement(name)
* parentNode.appendChild(node)
* element.innerHTML
* element.style.left
* element.setAttribute()
* element.getAttribute()
* element.addEventListener()
* window.content
* window.onload
* window.scrollTo()