COURSEWARE

Hypertext Markup Language

Contents

Overview

Hypertext Markup Language (HTML) is the standard markup language for creating Web pages. Specifically, HTML is used to determine the *general structure of*, and *actual contents stored within*, a Web page.

Tutorial

HTML is most commonly used alongside **Cascading Style Sheets (CSS)** and **JavaScript** in Web pages.

If HTML determines the structure and content of a page, then CSS determines **how the page looks to the user**, while JavaScript determines **what the page does**.

As an analogy, if a Web page is like a building:

- the CSS is the architecture style - how the building looks
- the JavaScript is its purpose - what the building is used for
- the HTML is the bricks, mortar, foundations, and every item stored in the building

The latest version of HTML - **HTML5** - includes a lot of changes to its previous version, which starts to blur the lines somewhat between JavaScript and HTML when it comes to basic functionality - HTML5, for instance, is capable of playing video without JavaScript to help it.

You can tell if a Web page is written in HTML5 by checking if the following declarative tag is written on line 1:

```
<!DOCTYPE html>
```

Structure

HTML works on a tag-based system using angled-braces (`<>`), which determine the content which gets displayed on the page, as well as any metadata (stuff that isn't displayed but is needed to make the site render correctly).

These tags are known as **elements**.

Usually, each element has an opening block and a closing block, e.g.:

```
<html>
</html>
```

This is analagous to the pairs of curly-braces (`{}`) used in conventional programming languages such as Java.

Some elements might contain modifiers, or **attributes**, which allow us to specify their function:

```
<meta charset="UTF-8">
```

Here, we're setting the `charset` attribute to comply to the UTF-8 character set encoding.

Let's look at a basic Web page to see how it's structured:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Some Title</title>
        <meta charset="UTF-8">
    </head>

    <body>

        <!-- Your content would go here -->

        <script src="javascript.js"></script>
    </body>
</html>
```

The indentation should look familiar to you! Just like traditional programming languages, each element tag is nested according to their *scope*:

- `<html>` is the root element of the Web page - without it, the page wouldn't load
- `<head>` is the container in which metadata is stored - the browser interprets any elements placed within it as something that should not be displayed on the Web page itself
- `<title>` displays the page's title in the tab/window the Web page occupies
- `<meta>` stores the relevant metadata of the page, such as character encoding, search engine keywords, etc.
- `<body>` is the container in which all the visible elements of the page are stored - the browser interprets any elements placed within it as something that should be displayed on the Web page itself
- `<script>` is where you would store JavaScript, either as references to external code (using the `src` attribute, as above) or as inline scripting (not generally recommended)

The Document Object Model

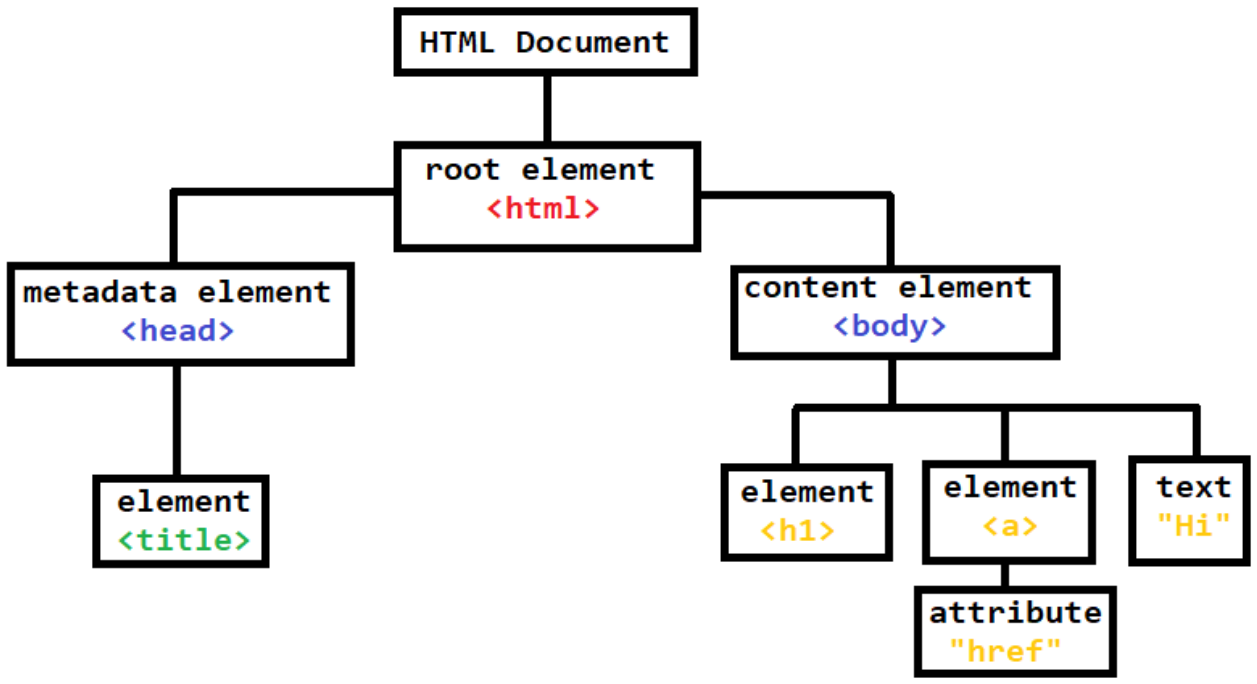
You don't see tags everywhere whenever you're reading a Web page because your browser is designed to read and interpret the HTML tags for you. In that sense, if HTML is the blueprint for a building, the browser is the architect.

When a Web page is loaded, the browser creates a **Document Object Model (DOM)** for that page, which is based on the layout (i.e. the tags) of its HTML.

The Web page's HTML file is parsed into **tokens** based on the element tags, which then is converted into *nodes* that are linked together in a tree data structure.

This preserves all the different **relationships** between the elements, allowing the browser to understand how everything should be laid out.

A typical page might look something like this:



You'll notice a couple of things here:

- JavaScript can run on the entire Web page - the `document` - so it goes at the top of the DOM
- the `<head>` and `<body>` elements are at the same level - remember `<head>` is hidden
- elements may contain **attributes**, which affect what the element does - here, our `<a>`, which is a link element, contains an `href` which would contain the URL we want to link to
- raw text may occasionally show up outside of an element

Bear these in mind when you're writing your own HTML in the future.

Entities

Characters such as `<` and `>` have a special meaning in HTML, as they're read by the browser when it parses our element tags into objects.

As a result, if we want to write the actual character of `<` (for instance, if we're doing some sort of complicated maths on-screen) then we'll need to render it using an **entity** character.

Some of the most common ways to write different entity characters are listed below:

Character	Entity Name	Entity Number
<	&lt;	&#60
>	&gt;	&#62
&	&amp;	&#38
"	&quot	&#34
'	&apos	&#39
Non-breaking space		
¢	&cent	&#162
£	&pound	&#163
¥	&yen	&#165
€	&euro	&#8364
©	&copy	&#169

Character	Entity Name	Entity Number
®	`&reg`	`&#174`

A fuller reference is available here.

Exercises

Try to create a simple Web page using HTML5 which contains:

- a greeting
- your name
- a title saying "Hello there!"

(*note: HTML documents are saved with the* `.html` *extension.*)