# What to know about HTML5

This doc is meant to highlight basic information about HTML5 (including CSS and a few words about websites), which must be learned during Orientation for Software Engineering. This can also be used in the beginning of other courses, e.g. when JavaScript client-side programming will be introduced. In this document, there is nothing about Javascript.

## Version history

| | | |
|---|---|---|
| 2019-08-16 | First version based on a secondary part of Juhani Välimäki's "WhatToKnowBeforeJavaScriptClientSideProgramming.doc" | Kasper Valtakari |
| 2019-08-19 | Minor clarifications only | Kasper Valtakari |
| 2019-08-22 | Document split again (a separate folder/SCP part). Renamed this part WhatToKnowAboutWebsites → WhatToKnowAboutHTML5. | Kasper Valtakari |

Kasper Valtakari, Juhani Välimäki
Haaga-Helia
Business Information Technology
2019-2020

## Table of contents

# 1 Introduction

This is a collection of knowledge needed for basic understanding of HTML5.

If you haven't got prior HTML5 knowledge yet, this document is for you. If you know, then this serves as a reminder and recap of what you should know. Here we do not consider nice looks, graphics, colors or content. Technical skills only. There is a separate path for website design and digital services. Error-free, technically correct websites are appreciated. (Of course, in real business nice looks for the site are important.) Write as short HTML and CSS as possible.

Don't stress about not understanding everything. You'll have time to learn them during the first semester.

## 1.1 Outside of the scope – What is handled somewhere else

- Full details of HTML5, other concepts such as XML and older HTML versions
- Details of CSS
- Responsive HTML-CSS e.g. for narrower screens or portrait-landscape changes.
- JavaScript

## 1.2 Use Visual Studio Code (VS Code)

Here at Haaga-Helia, our preferred text editor (or an IDE, Integrated Development Environment) in all HH classrooms is VS Code (Visual Studio Code). For personal preference, you can use any editor (such as Atom, Sublime text, Notepad++ etc). Please avoid the plain Notepad, though.

Starting: Press the Windows logo key on the keyboard and then type "Visual" and select Visual Studio Code.

# 2 Website basics

## 2.1 What is a website?

Putting it in a **simplified way**, a website is a folder with web pages. And, to offer it publically to the Internet, there must be a computer with a program called the web server. You request a page writing the address to the client browser and the web server returns the HTML page to the browser as a response.

## 2.2 Static websites and dynamic websites

In our case, the website will be static. It means that the web server just gives your browser the HTML file that was asked for as it is, without changing that **static HTML** text file at all. That HTML file usually contains links to CSS files, images and JavaScript code files. The browser understands to ask for those other files as well when it parses the HTML markup code and finds those links or image elements.

Nevertheless, they are all **static, ready-made files**. The returned HTML file has always the same characters, the image will be the same, and JavaScript contains the same code.

Later (on the following courses) we will have something happening and the HTML page changed before web server returns it to the client browser application. For example. we will get information from the database and write it on the page. Then the construction of the page and the content on the page is **dynamic**. The web server runs some code that is used to create or change the page.

## 3 HTML Details

HTML document is a text file with file name extension **“.html”** or “.htm”. HTML5 is the current version of the language. HTML5 coding includes also CSS styling and can also include Javascript.

Here is a minimal valid HTML5 file structure so that it still makes sense.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>Hello (to the page tab)</title>
    <link rel="stylesheet" href="css/siteStyle.css" />
</head>
<body>
     <h1>Hello World</h1>

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

### 3.1 HTML5 semantic elements

Here is the same template with some **semantic elements** added as content of the **body**.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>A Simple OSE Example (this will show on the page tab)</title>
    <link rel="stylesheet" href="css/siteStyle.css" />
</head>
<body>
    <header>Some items such as cities listed in this document</header>

    <main>
        <article>
            <h2>Helsinki</h2>
            <p>Capital of blah blah blah...</p>
            <p>Suomenlinna est dolores ipse...</p>
        </article>
        <article>
            <h2>Kotka</h2>
            <p>The sea blah blah blah...</p>
            <p>The parks blah blah blah...</p>
            <p>Dolorem ipsum...</p>
        </article>
    </main>

    <footer>Copyright 2019 Kasper Valtakari, Juhani Välimäki</footer>

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

**Think!** Look at that HTML5 document template. You can use this as some kind of correct HTML5 page model. Think about the following questions that help you to remember the correct template:

1. header, main, article and footer are examples of **HTML5 semantic elements**. What does **semantic** mean? Can you explain the semantic elements in this example?

2. There are other HTML5 semantic elements. Which of them should be used when you want to say that here are the most important links for browsing around this website?

3. In addition to the person writing and reading HTML source code, who or what could benefit from having semantic elements instead of just a lot of <div> and <p> elements on the page?

## 3.2 Writing well-formed HTML markup

This example shows all syntax you need to know to be able to see, interpret and write well-formed HTML code. These are overall XML (eXtended Markup Language) rules which should apply also to all HTML we write. HTML can be written in a more relaxed way, so you may see HTML written with different logic.

```
<abc name="O'Neill" someMessage='He said: "OK!" to John'>
  <aaaBee>Copyright &copy;2015 – John Doe</aaaBee>
  <hoo><iii>koo</iii></hoo>
  <dee ceeEee="yes" />
</abc>

<!-- This is an example of an HTML comment,
     it will go publically to the client
     but not shown on browser -->
```

**Think!** The rules below are real and correct. The example above abides by these rules. Look at each rule and check that the example above follows the rule:

a) There is a start tag `<abc>` and end tag `</abc>` for every element (end tag with forward slash, remember it from the fact that your skill will progress on, forwards)

b) ..unless the element is a self-closing tag ending with `/>`
The whole element should look like this: `<xyz wyx="jkl" />`

c) None of the elements overlap. They are perfectly nested inside of some parent element. Or they contain content elements perfectly nested = opened and closed inside them. Correct: <a><b></b><c></c></a>, Incorrect because of overlapping elements: <a><b></a></b>

d) Element or attribute names contain only English letters A-Z, a-z, digits 0-9 or underscore. No spaces, no special characters in element or attribute names. (Not even Scandinavian and other non-English letters in any identifiers on our courses, please. Data/values/content naturally can have any characters, even spaces).

e) Element and attribute names should be written case-sensitively.The start tag and end tag must be written exactly same way.
Please write all HTML tags with small letter, e.g. <body> <main> and NOT: <BODY><MAIN>, the '90s way.

f) Attribute values are closed inside double (") or single (') quotation marks. If you have one of these characters in data, you may use the other ones to close the value. (If you have both in data you need to do more…)

g) Comments start by <!-- and end by -->. Note that all parsers/browsers do not anticipate/accept comments in all places.

h) Character entities start with "&" (ampersand) and end with ";" (semi-colon) e.g. "&copy;" marks the copyright symbol character. So, it is just one character written with six characters in the HTML markup code.

**Think!** Look at the HTML example again. Using it as an example connect the professional term and an item from the document. Some terms are several times and some items appear twice in different roles.

| | |
|---|---|
| 1. element | a) koo |
| 2. element | b) ceeEee |
| 3. attribute | c) iii |
| 4. attribute value | d) abc |
| 5. content | e) O'Neill |
| 6. content | f) iii |

## 3.3 Checking the HTML Validity

Programming is all about making errors. Some of the errors will cause e.g. the JavaScript NOT running at all. Some errors might jam or crash the browser. Some errors browser will be able to fix correctly. Some errors won’t get fixed correctly. But all in all, we should write valid HTML, not make browsers work hard to find what we wanted to say.

World Wide Web Consortium (W3C, http://w3.org) has validators. Check your first pages at https://validator.w3.org. ‘Direct input’ is for the cases when you don’t have your website published to the whole internet. You might need to tell that you want your pages to be validated as “HTML5” pages, not e.g. as “XHTML 1.1 Strict”.

**Think!** You published your website on your Myy Linux user account under public_html folder. Thus, you can now validate your pages also by giving the validator a public web link to your page!

# 4 CSS basics – this much you need to know about CSS

CSS details will be learned on the Digital Services path. On the OSE course, you only need to understand the basics, not to learn the CSS attributes.

Basically we should put **all** style definitions in CSS files and only the CSS file link in the HTML file. That makes it easier to read the shorter HTML document and thus understand the page structure.

## 4.1 How to tell HTML page to use a certain CSS stylesheet

Add a link element in the head element of the html document. The attribute ‘rel’ tells what the relationship between this html document and that linked file is: The linked file is the stylesheet of this page. The attribute ‘href’ is a hyperlink reference to the actual file. Thus it’s the file/folder path:

```
…
<head>
    …
    <link rel="stylesheet" href="css/siteStyle.css" />
</head>
```

Now the html page knows to read and apply the style rules defined in siteStyle.css. By the way, **Cascading Style Sheet (CSS)** means that definitions cascade like water in a fountain: water jets into the air and from several basins all water pours to the bottom of the fountain. You will learn the logic what happens when there are several rules affecting the same element. After all other priority rules, the rule that has been define later overrides a former rule. Latter means you read HTML document line by line. If there is a link to a CSS file, you apply those rules and if there is another CSS link later, the rules will be overwritten or expanded.

## 4.2 How CSS file can locate elements in the HTML document

Examples of selectors and style rules: file siteStyle.css

```
p {
    color: black;
}
.class1 {
    color: blue;
}
#someId {
    color: red;
}
/* The 'color' means color of the text */
/* Here, the line order happens to be so that the latter rule always applies.
    This is how you write comments in CSS files!
*/
```

The 1st **selector** ‘p’ selects all paragraph **elements** (<p>content</p>) in the HTML documents that link to this CSS file and applies the black text color **style rule** on them. The 2nd selector ‘.class1’ selects all elements that have **class** class1 defined in them and applies blue text color style rule. If any paragraphs had class1 definition, the black was replaced by blue. The 3rd selector ‘#someId’ selects the element with unique **id** ‘someId’. That element will now get the red text color, no matter whether it was a paragraph or did belong to class1 or not.

**Think!** How are style rules grouped (connected to the selector) in the CSS file?
How are these groups separated from each other?
What is the selector for element like? E.g. for standard element type ‘div’.
What is the selector for classes like? E.g. for class ‘important’.
What is the selector for ids like? E.g. for id ‘textName’.
With how many elements in one HTML doc can an id selector match?
What is the basic structure of a style rule inside a block?
How are the human-to-human comments separated from the CSS style rules?
List all 4-6 separators (single characters or two) that you can identify in the basic CSS syntax.

**Think!** Make one of your pages to test the cascading nature of the style rules. That is: put two Lorem Ipsum text paragraphs in an html document with same class attributes. Add an id to the other paragraph. Then link a CSS file to the document and define e.g. text color for paragraph elements, for that class and for that id. Test in each phase to see the colors changing.

If you want, you can also add another CSS file link after the first one and put different style rules there. And see which one overwrites/overrides. You can also put style definitions inside of the HTML (bad habit?) just to see what happens if you define style even later and closer to the actual element, thus testing the cascading nature of the style rules. You need to visit the w3.org HTML5 & CSS documentation for this, or look at the examples at the private Norwegian website w3schools.com.