HTML5

chapter 4 / 6

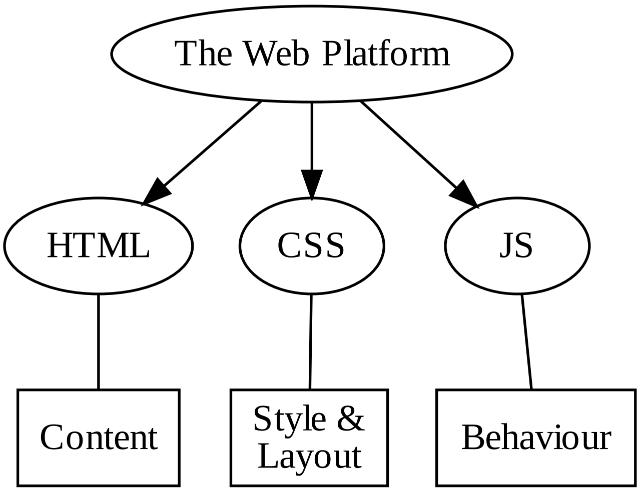
HTML5

Sections in this chapter:

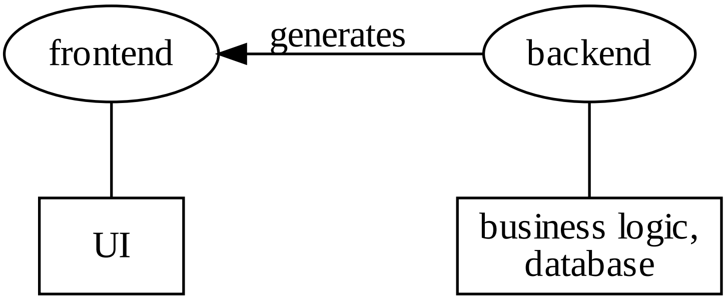
1. HTML basics
2. HTML content
3. Attributes
4. HTML Semantics
5. Two special elements

4-1. HTML basics

The **Web platform** is really the combination of three separate technologies: *4-1-1*

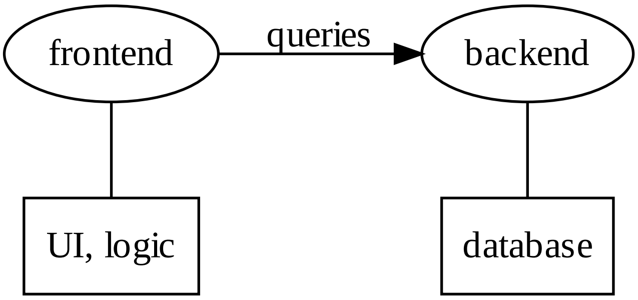


Traditionally, **web apps looked like this**: 4-1-2



Web tech was **just the UI**. The real coding was done in `php`/`ruby`/`java`/`.NET`.

But a **modern SPA (Single Page Application)** is more like this: 4-1-3



This means that web tech is on the rise and knowledge about the web platform will serve you well. 4-1-4

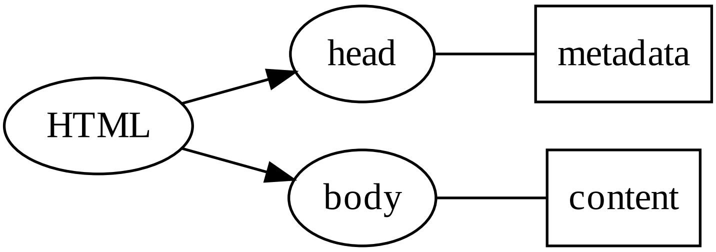
We will start by looking at HTML.

HTML (HyperText Markup Language) is used for encoding documents meant to be consumed on the web. 4-1-5

Much like Photoshop expects a `.psd` file, your favourite web browser wants `.html`.

HTML is an example of a **markup language**. What this means is that we, by marking up different sections of the document, are able to represent structural, presentational and semantic information, alongside content. 4-1-6

An HTML document has the following structure: 4-1-7



Translated to actual HTML, that becomes the following: 4-1-8

```
<html>
    <head>...</head>
    <body>...</body>
</html>
```

By adding a leading `<!DOCTYPE html>`, we tell the browser that this is a modern **HTML5** document 4-1-9

```
<!DOCTYPE html>
<html>
    <head>...</head>
    <body>...</body>
</html>
```

Notice that HTML elements consists of a **start tag**, **content** and an **end tag**. 4-1-10

As shown above in the previous section, the basic elements of an HTML5 page are: 4-1-11

```
<!DOCTYPE html>
<html>
    <head>
    Metadata about the document
    </head>
    <body>
        Content of the page
    </body>
</html>
```

Every HTML document begins with a !DOCTYPE, a **document type declaration** or **DTD**. 4-1-12

It is not an HTML element in itself, but by using this, we are telling the browser that this is an HTML document using the HTML5 standard. In older versions of HTML this tag included a reference to a document type declaration, see below example:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

In HTML5, this is not needed, since it is not based on SGML (Standard Generalized Markup Language), as was the case with HTML 4.01.

Here is a short explanation of the remaining elements:

The `<html>` element indicates that the code the page is made up from conforms to the standards of the HTML dictated in the **document type declaration**. 4-1-13

<head> contains metadata about the document, such as the page title and links to style sheets. 4-1-14

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF8">
        <link href="style.css" rel="stylesheet">
        <title>Title of the page</title>
    </head>
    <body>
        Content of the page
    </body>
</html>
```

<meta> provides additional metadata about the document, such as character encoding and keywords used by search engines. 4-1-15

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF8">
        <meta name="Author" content="Author name">
        <meta name="Description" content="Description">
        <meta name="Keywords" content="Keywords">
        <title>Title of the page</title>
    </head>
    <body>
        Content of the page
    </body>
</html>
```

Q Did you notice something strange in the previous section? 4-1-16

The <meta> element is missing an end tag! But why is that? 4-1-17

A The <meta> element is a **void element**, meaning it cannot have content. These elements are usually used to insert or embed something in the document. 4-1-18

4-1-19

The `<meta>` element is very useful when it comes to SEO (Search Engine Optimization).

Google has a Search Engine Optimization Starter Guide that is worth checking out. We will come back to this later in the course.

Some common HTML elements

Below is a short list of some common elements.

Lists

There are three kinds of lists:

Unordered lists, where the items are specified as bullets

```
<ul>
    <li>Gandalf</li>
    <li>Saruman</li>
</ul>
```

HTML output:

- Gandalf
- Saruman

Ordered lists, where the items are numbered

```
<ol>
    <li>Gandalf</li>
    <li>Saruman</li>
</ol>
```

HTML output:

1. Gandalf
2. Saruman

Description lists, where the items are specified as a descriptive text.

```
<dl>
    <dt>Gandalf</dt>
    <dd>Loves smoking his pipe</dd>
    <dt>Saruman</dt>
    <dd>All around bad guy</dd>
</dl>
```

HTML output:

Gandalf
Loves smoking his pipe

Saruman
All around bad guy

Tables

Tables are used to display data in a grid format. They are NOT to be used for layout purposes, although this was done in the days before CSS in browsers and actually continued some time after that.

A simple example of a table:

```
<table>
    <tr>
        <th>Name</th>
        <th>Price</th>
        <th>Number in stock</th>
    </tr>
    <tr>
        <td>Sushi bazooka</td>
        <td>499 SEK</td>
        <td>5</td>
    </tr>
</table>
```

HTML output:

Name	Price	Number in stock
Sushi bazooka	499 SEK	5

Forms

Forms are used to collect data from users visiting your site. The elements contained inside a form are called form controls, examples of these are:

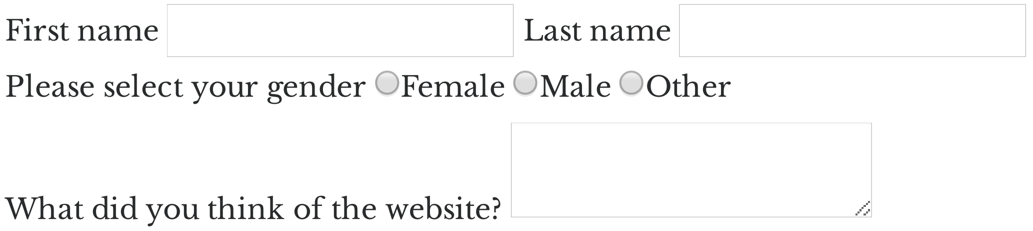
```
<input type="text">
<input type="radio">
<input type="checkbox">
<select>
<textarea>
<button>
```

Below is a simple example of a form, where a user can enter basic information:

```
<form>
    <label>First name</label>
    <input type="text" name="firstName">
    <label>Last name</label>
    <input type="text" name="lastName">
    <label>Please select your gender</label>
    <input type="radio" name="gender">Female
    <input type="radio" name="gender">Male
```

```
        <input type="radio" name="gender">Other
        <label>What did you think of the website?</label>
        <textarea name="comment"></textarea>
    </form>
```

HTML output:



First name ▭ Last name ▭

Please select your gender ○Female ○Male ○Other

What did you think of the website? ▭

The above output isn't very pretty, but we will be able to make it look a lot nicer when we have gone through CSS.

Audio and video

- `<audio src="sandman.mp3" autoplay>`
- `<video src="family-guy-pilot.mp4" autoplay>`

`src` is an attribute pointing to the actual media content. `autoplay` is an attribute controlling automatic playback. This is disabled by default.

Check out MDN Audio Element and MDN Video Element for more.

4-2. HTML content

As we said before the `<body>` element contains the actual **contents** of the document, that the browser will show on screen. *4-2-1*

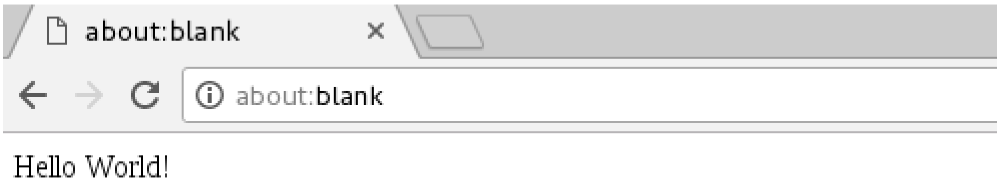
So, what does this look like?

Basically, the browser will display the **leaf text nodes**. *4-2-2*

So let's say we feed it this:

```
    <!DOCTYPE html>
    <html>
        <head></head>
        <body>
            Hello World!
        </body>
    </html>
```

Then the browser would display this: *4-2-3*



Hello World!

But of course, there are **elements with special meaning** to the browser. These fall into **two categories**: *4-2-4*

- **Block** elements *4-2-5*
- **Inline** elements

A **block** element takes up the full width of the document, meaning it starts and ends with a new line. *4-2-6*

The perhaps most common **block** element is `<div>`, short for divider, which is a generic block level element. *4-2-7*

```
<div>A block element</div>
```

The above `div` will position itself below other block elements, starting a new line. It will take all space it can horizontally and new content will go beneath. It may contain other block and inline elements. *4-2-8*

```
<div>A block element</div>
<div>Another block element</div>
```

HTML output:

A block element
Another block element

Examples of block elements: `<div>, <h1>, <p>, <ul>, <table>, <section>, <article>` *4-2-9*

An **inline element** flows with the text. Other inline elements will be displayed next to the previous element, on the same line. It is mainly a container for text and other inline elements. It may **not** contain block elements. *4-2-10*

A common **inline** element is the `<span>`, which is a generic inline element. *4-2-11*

```
<span>An inline element</span>
<span>Another inline element</span>
```

HTML output:

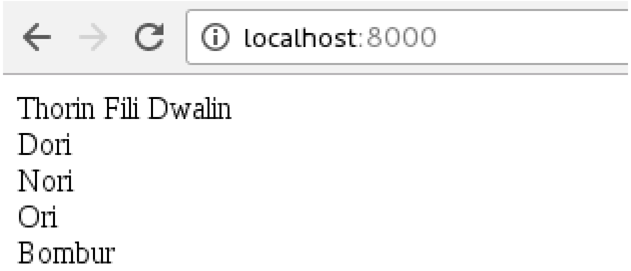
An inline element Another inline element

Examples of inline elements: `<span>, <td>, <a>, <img>` *4-2-12*

Q Knowing what you just learned, what would this look like? *4-2-13*

```
<!DOCTYPE html>
<html>
    <head></head>
    <body>
        <div>Thorin Fili
        <span>Dwalin</span></div>
        <div>Dori
        <div>Nori</div> Ori</div>
        <span>Bombur</span>
    </body>
</html>
```

A Well, like this of course! *4-2-14*



Thorin Fili Dwalin
Dori
Nori
Ori
Bombur

© Edument 2017

On the subject of generic elements - there are also some **generic attributes** that we can set on all elements. MDN has a full list here: *4-2-15*

https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes

4-3. Attributes

Attributes can be found after the opening tag of an element. *4-3-1*

They are key/value-pairs, consisting of a **name** and a **value**. *4-3-2*

The **name** tells us what kind of information the author is providing about the element's content.

Let's take a look at some generic attributes, mentioned in the previous chapter: *4-3-3*

The `id attribute` allows us to give a unique name to an element that for some reason is special: *4-3-4*

```
<div id="app-wrapper">...</div>
```

The id allows us to **easily target the element** from CSS.

The `class attribute` is a bit special. It is a sequence of 0 or more whitespace-separated **class names**, without a particular priority ordering between them, stating that this element belongs to these classes. Many elements can share the same class(es). *4-3-5*

```
<div class="wrapper article">...</div>
```

This allows us to **group elements together** in themed categories, and also to target these groups in CSS. *4-3-6*

4-3-7 And just to show an example that doesn't have to do with targeting - the `lang attribute`

```
<div lang="en">...</div>
```

4-3-8 allows the browser to **provide spell checking and dictionary lookup** for the correct language. It is also useful for for search engines, speech synthesizers, spell checkers etc.

4-3-9 **Note**: The naming convention for ids and classes are lowercase and hyphon-separated:

```
<div id="my-awesome-div">...</div>
```

HTML5 uses **boolean attributes** as well. For instance:

```
<div hidden>I'm invisible!</div>
```

The "boolean" part of it all, is that it's either applied, or not. Basically, you either **leave it out** to represent `false` or **add it in** to represent `true`, meaning that the **following** is not allowed according to the HTML5 specification:

```
<div hidden="false">I'm non compliant!</div>
```

4-3-10 Finally - if for whatever reason you want to insert a **comment** into HTML, or temporarily exclude some elements from the flow:

```
<div>a paragraph of text</div>
<!-- <div>a paragraph which has
been commented out...</div>
<div>...along with this one</div>-->
<div>but this one is still
visible!</div>
```

4-3-11 Beware that these **comments still shows up in the source code**, so no cursing!

4-4. HTML Semantics

You just met `<div>` and `<span>` which are generic elements. But, most elements you use will be **semantic**. 4-4-1

Semantic elements are elements which clearly define their meaning to both the browser and developer.

These new semantics was introduced after Google analyzed more than 1 billion pages to see what developers were using to describe their div's and other elements. New elements include `<header>`, `<footer>` and `<section>`. 4-4-2

Q So, semantic elements give meaning to their content. But what other advantages are there to using them? 4-4-3

A Because we are giving our content meaning, we are helping computers to make sense of our content. 4-4-4

Also, we are giving assisting technologies, like screen readers, help in reading and interpreting our page. Here is an example of a web page read by a screen reader:

ChromeVox lite demo page
Furthermore other developers, even those who are not native English speakers, will be able to make sense of our structure.

Q What is the difference between divs and these tags? 4-4-5

A The new semantic tags actually **do** claim a lot of the use case we used to use `div` for. Basically, while the semantic elements has specific meaning, the `div` tag **does not have any specific meaning**. 4-4-6

Q So we shouldn't use div? 4-4-7

A You're probably going to use it **less**, but you shouldn't stop using it. It has not disappeared from the standard. Basically, when you need a parent container that **does not carry any specific meaning**, then by all means use a `div`. *4-4-8*

And this is important - the primary purpose of HTML is to **explain to the browser what the content is**, so that the browser can make informed decisions on how to display it. *4-4-9*

Here are some **common elements** useful for **marking up text**: *4-4-10*

- A `<p>` element represents a **paragraph** of text. It will be displayed as a block with a bottom margin. *4-4-11*

- The `<h1>` - `<h6>` elements are **headlines**. 1 is the most important. They are blocks, with a bottom margin and a bigger and bolder look the lower the number. *4-4-12*

A general rule is to use one `<h1>` header per page, `<h2>` to denote sections and `<h3` and those below for concise investigation about each topic: *4-4-13*

```
<h1>Earth</h1>
<p>Some text about Earth</p>
<h2>Europe</h2>
<p>Some text about Europe</p>
<h3>France</h3>
<p>Some text about France</p>
<h2>Asia>/h2>
<p>Some text about Asia</p>
<h3>China</h3>
<p>Some text about China</p>
```

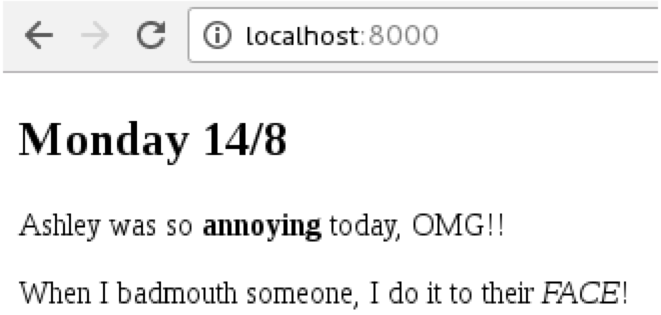
`<strong>` elements are used for an inline section of text that you want to **highlight**. Typically displayed in bold. *4-4-14*

Similarly there is an `<em>` element for **emphasis**, typically displayed in italics. *4-4-15*

So feeding the browser this: 4-4-16

```
<!DOCTYPE html>
<html>
  <head></head>
  <body>
    <h2>Monday 14/8</h2>
    <p>Ashley was so
        <strong>annoying</strong> today, OMG!!
    </p>
    <p>When I badmouth someone, I do it to their
        <em>FACE</em>!
    </p>
    </body>
</html>
```

...we get this: 4-4-17



Monday 14/8

Ashley was so **annoying** today, OMG!!

When I badmouth someone, I do it to their *FACE*!

There are **rules on where in the structure an element can appear**, which all depend on their semantic meaning. 4-4-18

For example, it makes no sense to nest a paragraph inside another paragraph, so that is forbidden. 4-4-19

In earlier HTML, many elements dealt with appearance. For example: 4-4-20

- there was a `<bold>` element
- and and <i> element (short for italics).

Now they are **deprecated in favour of semantic versions**, like `<strong>` and `<em>`. This is typical for HTML as a whole. 4-4-21

The fact that HTML content is semantic is also the reason why we haven't mentioned **layout** or **style** yet. *4-4-22*

Those things should **only live in CSS-land**. HTML is content, CSS is style and layout, and never the two shall meet, lest sadness and pandemonium ensues.

So, what **other elements** can we use? We will look at a few of them in the next chapter, but check out the full reference at MDN: *4-4-23*

https://developer.mozilla.org/en-US/docs/Web/HTML/Element

4-5. Two special elements

In this section we'll take a look at **two important elements**. First out is the `<img>` element, which lets us insert images into our documents: *4-5-1*

```
<img src="resources/fili-kili.jpg"
     title="fili and kili" alt="fili and kili">
```

We control the image mainly through two attributes: *4-5-2*

- `src` which is an **address to the image file**. It can be either
  - **relative**, pointing to a resource on the same server, or
  - **absolute**, pointing anywhere
- `title`, which will be displayed as a tooltip in most browsers

Note that as **an image will never have a child**, just like the meta element, we **don't need to close it**. 4-5-3

When you need all **details about a specific element** such as the `<img>` element, MDN again has you covered: 4-5-4

https://developer.mozilla.org/en-US/docs/Web/HTML/Element/img

Q Now for the second important element, which is arguably the **most important of all**. Can you think of which it is? 4-5-5

A The **anchor**, or <a> element as we know it! Without that, there would be no inter**net**. 4-5-6

The anchor has a **href attribute** which control what page it leads to. 4-5-7

```
<a href="https://developer.mozilla.org/en-US/">
    MDN
</a>
```

Just like image sources it can be **relative or absolute**.

Here is MDN:s full specification for the anchor element: 4-5-8

https://developer.mozilla.org/en-US/docs/Web/HTML/Element/a

The <a> element can also be used for internal linking, improving SEO, using the `href` attribute: 4-5-9

```
We will learn more about validating your HTML in
<a href="#validating">Section three</a>
...later in the course...
<h2 id="validating">Validating your HTML</h2>
```