Table of contents

This document in one large webpage...

Introduction

The purpose of this course is to provide an intro to web technologies (HTML, CSS) and dynamic webpages via PHP.

Resources

The main documentation for this course is a site an can be found at https://asoete.github.io/howest-webtechnology. The main advantage of a using a site as a textbook is that the included examples and snippets can be rendered/formatted by the browser on the fly.

A PDF version of this document is also available. But keep in mind that some HTML features can get lost in the conversion to a pdf.

There are no slides available, all key aspects of this course will introduced via examples during the lessons. These examples and the solutions of the exercises made during this course will be published here.

Keep in mind that this course is a very practical one and that making exercises is the best way to learn, get familiar, with all the aspects featured in this course.

Additional resources

An in depth guide/reference/manual for PHP can be found at http://php.net

For an HTML and CSS reference see http://www.w3schools.com/html and http://www.w3schools.com/css

Contact

If you have questions about this course and/or it's content please ask... You can contact me via arne.soete@howst.be

Code in this document

This course will feature a lot of code. The source-code of all the snippets in this manual can be found here.

The source and the output of each snippets are always displayed whenever a snippet is included.

Example:

introduction/embed_example | *src*

```
<?php

echo "<h1>This is an embed example</h1>";

if( array_key_exists( 'KEY', $_POST ) ) {

    unset($_POST['KEY']);
}

?>

<p>
  Markup is interpreted by the browser and formatted accordingly..
</p>
```

output of introduction/embed_example

> # This is an embed example
>
> Markup is interpreted by the browser and formatted accordingly..

Getting started

This course requires some software to be installed.

- A web browser: firefox
- A text editor: gedit
- PHP
- GIT

Normally these packages are already installed on the provided VM. If not, they can easily be installed by running:

```
sudo dnf install firefox gedit php git
```

Press `y` when prompted `Is this ok [y/N]:`.

This document and all the exercises/examples are hosted on GitHub. This means a local copy of the source can be obtained easily **and kept in sync with the latest changes and updates**.

If you choose not to use the command line and git. Snapshots of each lessons exercises and examples will be made available for download here.

Init workspace

```
mkdir ~/Documents/webtechnology
cd  ~/Documents/webtechnology
```

- Create your own exercises directory

```
mkdir exercises
```

Get local copy of the *exercises and examples* solutions

- Get an initial copy of the repository:

```
git clone https://github.com/asoete/howest-webtechnology-examples.git examples-solutions
```

  This will create a `examples-solutions`-folder which will hold example solutions for the exercises on a per lesson basis.
- To get the latest version/updates run:

```
# in examples-solutions folder
git pull origin master
```

> **Warning:** If you made local modifications to any of the files in this repository, this update command (`git pull`) will most likely fail. So don't modify the contents in this folder...

> **Info:** When you do encounter errors while pulling, run:
>
> ```
> git fetch --all
> git reset --hard origin/master
> ```
>
> This will reset the repository to be identical to the one on GitHub. **Be warned: local modifications will be lost...**

Get local copy of this site. (Optional)

- Get an initial copy of the repository:

```
git clone https://github.com/asoete/howest-webtechnology.git webtechnology-site
```

- To get the latest version/updates

```
# in webtechnology-site folder
git pull origin master
```

- Start a local instance of the site:

  ```
  # in webtechnology-site folder
  make serve
  ```

  And open http://localhost:8000 in a web browser.

Final result

If you complete all of the steps above, you will end up with a workspace that looks like this:

```
~/Documents/webtechnology
├── examples-solutions
├── exercises
└── site
```

HTML Basics

> **Info:** This course is based on the HTML specification and can differ from older specifications like XHTML and HTML

HTML is an XML subset. This means it is composed out of tags with, optionally, attributes.

Tags

A tag is delimited by `<` and `>`, for example: `<body>`.

There are two types of HTML-tags:

- Non self-enclosing tags
- Self-enclosing tags

Non self-enclosing tags

Non self-enclosing tags exist out of two parts:

1. An opening part: `<tag>`
2. And a closing part: `</tag>`.
   The closing part is identified by the forward slash (`/`) before the tag-name.

These *parts* are used to contain/format certain content.

```
<tag> {{content}} </tag>`
```

Example: `<strong>Bold Font</strong>` (This tags formats its content in a bold font: **Bold Font**)

Self-enclosing tags

A self-enclosing tag has no content to format. So the closing part is left of:

```
<tag>
```

Example: `<br>` (this will insert a newline into your HTML)

Sometimes you may see self-closing tags used like `<tag />`, this trailing tag is optional since HTML5 and can be left of.

Attributes

Attributes modify the behaviour of a tag.

For example the `a`-tag converts a peace of text into a clickable link.

```
<a>My text to click</a>
```

The `href`-attribute defines where the link should take you:

```
<a href="http://go-here-when-clicked.com">My text to click</a>
```

Attributes are also used to modify the appearance of a tag. Later in this course we'll see more detailed examples of this.

A valid HTML document

A valid HTML5 document requires a bit of boilerplate:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <!-- Your webpages metadata -->
</head>
<body>
    <!-- your webpage specific content -->
</body>
</html>
```

This minimal markup tells the browser to treat the document as HTML5.

The document head

The `head`-tag allows the developer to define meta-data about the webpage. It is a wrapper around multiple other tags.

```
<head>
    <!-- meta tags here -->
</head>
```

The `head`-tag may only be defined once in the complete document.

Title

The title tag sets the web-page title. This title is displayed by the browser in the browser-tab.

```
<head>
  <title>My web-page's title...</title>
</head>
```

Style

We will address styling later in this course but for now it is sufficient to know that style information should be included in the head of a web-page.

Raw style

The `style`-tag allows to include raw CSS rules in the documents

```
<head>
  <style type="text/css">
    /* style information here */
  </style>
</head>
```

External style sheets

The `link`-tag allows to external style sheets into the document. (Do not confuse this tag with the `a`-tag...).

```
<head>
  <link href="/link/to/file.css" type="text/css" rel="stylesheet">
</head>
```

We will only ever include CSS-files to style our web-pages. The provided attributes in the example are required to include a CSS-file and avoid browser quirks.

The document body

The `body`-tag should wrap all the content to be displayed.

```
<body>
  <!-- all displayed tags and content go here -->
</body>
```

> ### Exercise:
>
> - Create a valid HTML-document with
>   - Title: Hello World
>   - Content: `Hello World from the my first web-page`

HTML: Hello World

Create a text file name `hello-world.html`

```
gedit hello-world.html
```

With contents:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Hello World</title>
</head>
<body>
    Hello World from my first web-page.
</body>
</html>
```

Open the local HTML file in the browser.

```
firefox hello-world.html
```

Headers: H*

The purpose of a header is to indicate the start of a new block a add an appropriate heading.

The `hn`-tags come in 6 variations. From to highest order header `h1` to the lowest `h6`.

The browser auto-formats these headers accordingly from largest to smallest font-size.

html-intro/headers | src

```
<h1>Header 1</h1>
<h2>Header 2</h2>
<h3>Header 3</h3>
<h4>Header 4</h4>
<h5>Header 5</h5>
<h6>Header 6</h1>
```

output of html-intro/headers

Header 1

Header 2

Header 3

Header 4

Header 5

Header 6

Containers

The purpose of these types of tags is to wrap other content. Why the content should be wrapped can vary:

- To indicate semantic meaning (new paragraph, a quote, ...)
- To position and/or style the contents in the container.

They are also referred to as *block*-elements

Paragraphs `p`

The `p` -tag encloses a blob of related text into a paragraph

html-intro/p-tag | *src*

```
Content before...
<p>
  Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy
  eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam
  voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita
  kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.
</p>
Content after...
```

output of html-intro/p-tag

Content before...

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Content after...

Generic container `div`

The `div` defines a *division* in the document. It is used al lot to wrap some content and apply styles.

It has no special styles by default

html-intro/div-tag | *src*

```
Content before...
<div>
  Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy
  eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam
  voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita
  kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.
</div>
Content after...
```

output of html-intro/div-tag

Content before...
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.
Content after...

Blockquote `blockquote`

The `blockquote` -tag is used to denote some block of text as a quote from another source.

Content before...
<blockquote>
  Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.
</blockquote>
Content after...

html-intro/blockquote-tag | *src*

Content before...

> Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Content after...

output of html-intro/blockquote-tag

Inline tags

These tags are inline because they do not start a new block (identified by new lines) as the previous tags.

Their purpose is either to give a specific style and semantic meaning to an element or to extend a certain functionality to the element.

Anchors (links): `a`

The `a` is used to link to other web-pages.

In order the function, the `href` -attribute is required on the `a` -element.

```
<a href="http://google.com">Link to goole</a>
```

html-intro/a-tag | *src*

Link to goole

output of html-intro/a-tag

Exercise:

Make a web-page with links to:

- google.com
- howest.be
- github.com

A newline: `br`

The `br` insert a newline into the document.

```
This is the first line.
This is the second line, but in html all white space is replaced by a single space...<br> The "br"-tag however instructs the browser to continue on a new line...<br><br>Cool right?
```

html-intro/br-tag | *src*

> This is the first line. This is the second line, but in html all white space is replaced by a single space...
> The "br"-tag however instructs the browser to continue on a new line...
>
> Cool right?

output of html-intro/br-tag

Emphasise text: `em`

The `em`-tag allows to emphasise certain text.

```
This is <em>emphasised</em> inline...
```

html-intro/em-tag | *src*

> This is *emphasised* inline...

output of html-intro/em-tag

Small text: `small`

The `small`-tag indicates the browser to use a smaller font-size to visualise this content.

```
This is <small>small</small> inline...
```

html-intro/small-tag | *src*

> This is small inline...

output of html-intro/small-tag

Inline wrap text: `span`

The `a`

```
This is <span>span</span> inline...
```

html-intro/span-tag | *src*

> This is span inline...

output of html-intro/span-tag

Strike text: `strike`

The `a`

```
This is <strike>strike</strike> inline...
```

html-intro/strike-tag | *src*

> This is ~~strike~~ inline...
>
> *output of html-intro/strike-tag*

Bold text: `strong`

The `a`

This is ****strong**** inline...

html-intro/strong-tag | *src*

> This is **strong** inline...
>
> *output of html-intro/strong-tag*

Inline quote text: `q`

The `a`

This is **<q>**quote**</q>** inline...

html-intro/quote-tag | *src*

> This is "quote" inline...
>
> *output of html-intro/quote-tag*

Exercise:

- Print the following text so the sentences are broken up as below.

  > HTML is a markup language browser understand to format documents.
  > CSS is a way to style this markup.
  > PHP is a programming language.
  > It is used to dynamically generate HTML-markup.

- Print the following text so `hello world` is emphasised.

  > Let's emphasise hello world in this sentence.

- Print the following text so `hello world` is smaller

  > Let's make hello world smaller in this sentence.

- Print the following text so `hello world` is bold

  > Let's make hello world bold in this sentence.

- Print the following text so `hello world` is crossed of.

  > Let's strike through hello world in this sentence.

Attributes

Exercises

title: PHP basics

PHP Basics

Execute a PHP-script

Variables

Scalars

Arrays

Loops

Conditionals

HTML extended: beyond the basics

External stylesheets

Forms

Last modified by Arne Soete at 2016-09-25