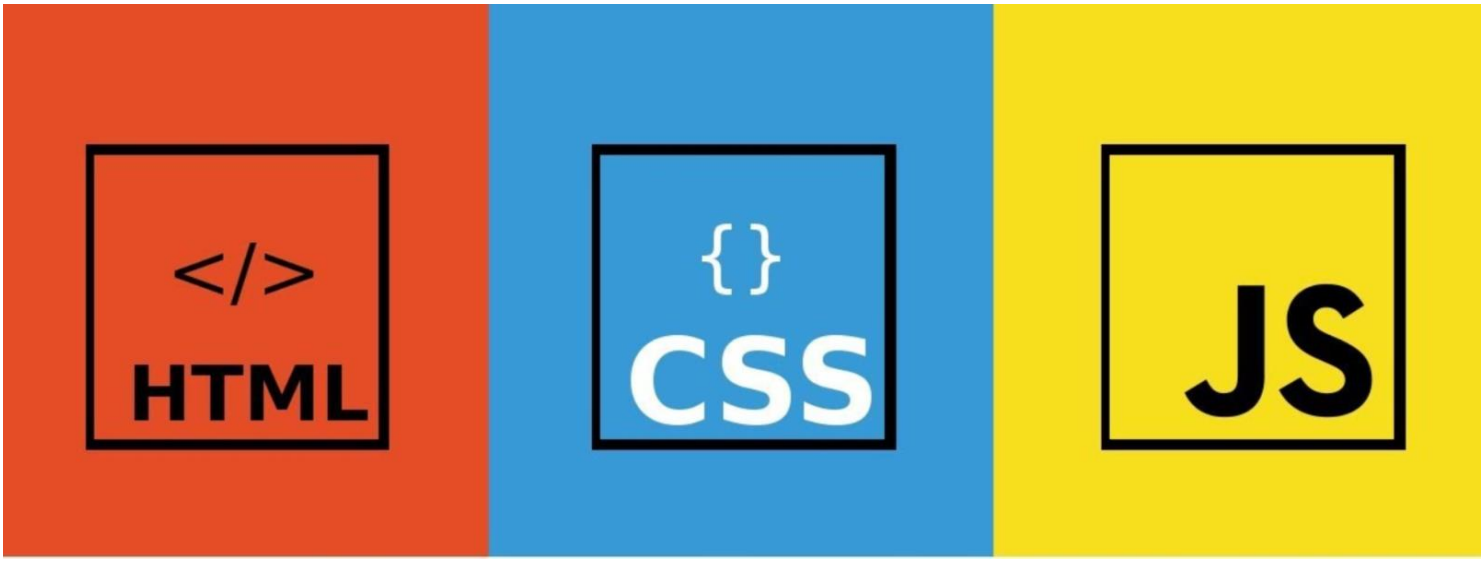
NGG-GAiN

HTML / CSS /JavaScript

Matteo Vögeli
28.05.2020



Inhalt

Software

In this course we will work with Visual Studio code. It is a free code editor from Microsoft. Visual Studio Code is available cross-platform for the operating systems Windows, macOS and Linux.

Download: https://code.visualstudio.com/download

HTML

What is «HTML»?

HTML (HyperText Markup Language) is the basic building block of the web. It describes and defines the content of a web page together with the basic layout of the web page. In addition to HTML, other technologies are generally used to describe the appearance (CSS) or functionality/behaviour (JavaScript) of a web page. The tags in HTML are case-insensitive. This means that they can be written in upper and lower case letters or in a mixture. Example <title> tag can be written as <title>, <TITLE> or in any other way. HTML consists of elements, each of which can be modified by a number of attributes. HTML documents are linked to each other by links.

Basic HTML tags

Document structure

```
<!DOCTYPE html>
<html>
    <head>
        <title>Page Title</title>
        <link rel="stylesheet" type="text/css" href="style.css">
    </head>
    <body>

    </body>
</html>
```

- <html>: Document type declaration
- <head>: (header data, e.g. information on title, etc.)
- <Body>: (body - content to be displayed, that is, text with headings, references, graphic references, and so on)

Head data

- <title>: (title of the page displayed in the tab)
- <Link>: (relationships to other documents in the Web space)
- <script>: (für Script-Anweisungen innerhalb des Dokumentes)

Text structuring

- <h1>, <h2> … <h6>: Headings
- <p>: Plain text
- : Ordered list
- : Unordered list
- : Elements in <ol> and
- : Container element

Links

- <a>: Link

Tables

```
<table>
  <tr>
    <td>Inhalt</td>
  </tr>
</table>
```

- <table>: Table definition tag
- <th>: Table head
- <tr>: Table row
- <td>: Table data

Multimedia and Graphics

```
<img src="image.png" alt="image">
```

- <img>: Integrate graphic files
- src: Path of the file
- alt: alternative text

Form

An HTML form is used to collect user input.

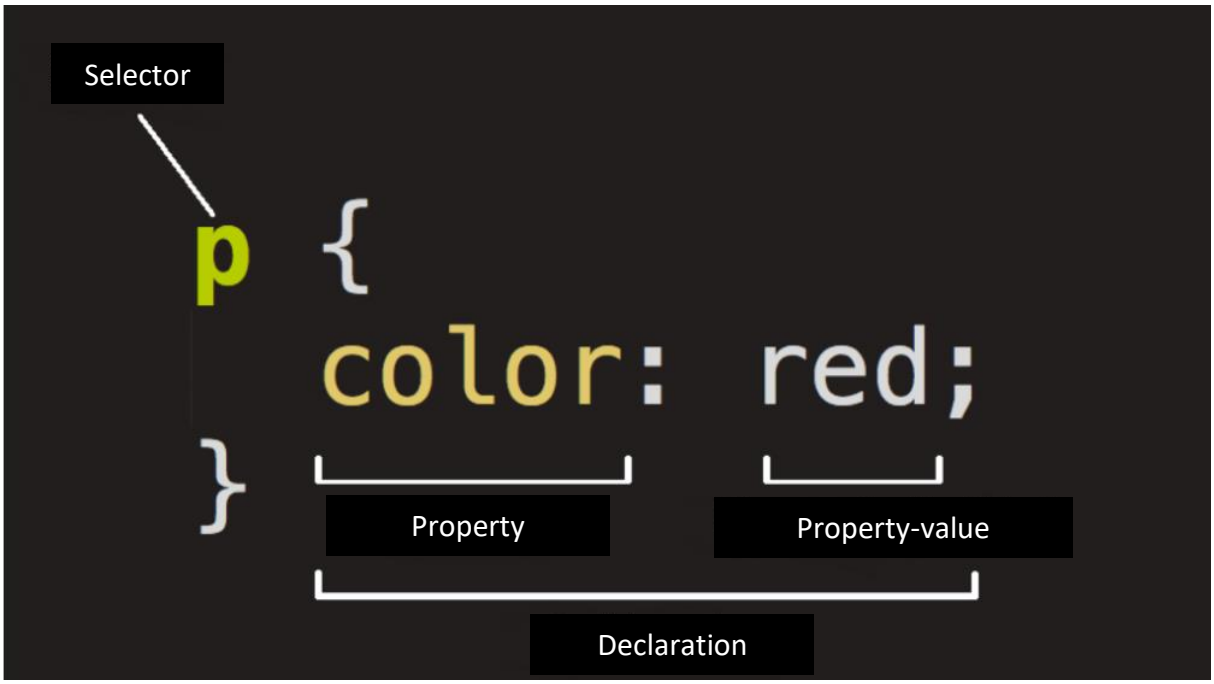
```
<form action="/action_page.php">
    <label for="fname">First name:</label>
    <input type="text" name="fname">
    <label for="lname">Last name:</label>
    <input type="text" name="lname">
    <input type="submit" value="Submit">
</form>
```

- <form>: Form definition tag
- <label>: defines a label for many form elements
- <input>: Input to collect data
- Type: Defines the type of the input

CSS

What is «CSS»?

Like HTML, CSS is not really a programming language. It is also not a markup language, but a stylesheet language that allows you to specify the appearance of elements on the page. For example, to select all paragraphs (<p>) and color their content.




```
/* All "p" elements */
p {
  color: red;
}

/* ALL elements with "header" class */
.header {
  height: 40px;
  width: 100%;
}

/* Element with "logo" id */
#logo {
  margin: 10px;
}
```

```
<p>I am a Paragraph.</p>

<p>I am a paragrah too!</p>

<div class="header"></div>

<img id="logo" src="./logo.jpg" />
```

Basics Selectors

Selector	Example	Result
.class	.header	Selects all elements with class="header"
#id	#logo	Selects the element with id="logo"
*	*	Selects all elements
element	p	Selects all <p> elements
Element, element…	div, p	selects all <p> and elements

Basics Properties

- Width
- Height
- font-size
- font-family
- Color
- Background
- Padding
- Margin
- Border

```css
.example {
  width: 100%;
  height: 40px;
  font-size: 12px;
  font-family: Arial;
  color: white;
  background: blue;
  padding: 20px 10px 40px 50px;
  margin: 40px;
  border: 1px solid black;
}
```

Integrate CSS in HTML

There are 3 ways to integrate CSS in HTML but usually only one proper way.

Version 1: Best way

```html
<!DOCTYPE html>
<html>
    <head>
      <title>Js outside Html</title>
      <link src="./style.css" rel="stylesheet" type="text/css">
    </head>

    <body>
      <h1>Hello World!</h1>
    </body>
</html>
```

Version 2:

```html
<!DOCTYPE html>
<html>
    <head>
      <title>Js outside Html</title>
    </head>

    <body>
      <h1>Hello World!</h1>

      <style>
        h1 {
          color: red;
        }
      </style>
    </body>
</html>
```

Version 3:

```
<div style="width: 100px; border: 1px solid red;"></div>
```

Combinators

```
div, p { color: red }
```
> Selects all "div" elements and all "p" elements

```
div p { color: red }
```
> Selects all "p" elements inside "div" elements

```
div > p { color: red }
```
> Selects all "p" elements placed directly in "div" element

```
div + p { color: red }
```
> Selects all "p" elements placed directly after a "div" element

```
div ~ p { color: red }
```
> Selects all "p" elements that are siblings to "div" elements

JavaScript

What is «JavaScript»?

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

Link a JavaScript file to a HTML file

There are 2 Ways to use JavaScript in HTML:

1: Inside HTML in <script> tags

```
<!DOCTYPE html>
<html>
    <head>
      <title>Js in Html</title>
    </head>

    <body>
      <h1>Hello World!</h1>

      <script> // JavaScript Block
        alert('Hello World!');
      </script>
    </body>
</html>
```

2: Import from .js file in <head> tag

```html
<!DOCTYPE html>
<html>
    <head>
      <title>Js outside Html</title>
      <script src="js/index.js"></script>
    </head>

    <body>
      <h1>Hello World!</h1>
    </body>
</html>
```

JavaScript basics

Variables

Variables are containers in which values can be stored. First, a variable is declared with the keyword var(new let & const), followed by any name under which this variable is to be addressed and an assigned value.

```
var name = "Gandalf";
var age = 9999;
var wizard = true;
```

New with "let" and "const"

Mutable (can be changed)

```
let name = "Gandalf";
let age = 9999;
let wizard = true;
```

Imutable (can not be changed)

```
const name = "Gandalf";
const age = 9999;
const wizard = true;
```

Data types

Data type, is a classification that specifies which type of value a variable has and what type of mathematical, relational or logical operations can be applied to it without causing an error.

```
// Number
var number = 1;

// String
var string = 'Hello';

// Boolean
var boolean = true;

// Undefined
var hello;
```

Concatenate variables

Variables can be concatenated (put together) to a string by adding a "+" between.

```
var variable = 'fellowship';

var text = 9 + ' people joined the ' + variable + '.';
// text = "9 people joined the fellowship."
```

Operators (Arithmetic)

An arithmetic operator is a mathematical symbol that generates a result based on two values (or variables).

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Exponentiation
%	Modulus
++	Increment
--	Decrement

```
// addition
5 + 5; // 10

// subtraction
5 - 5; // 0

// multiplication
5 * 5; // 25

// division
5 / 5; // 1
```

Operators (Comparison)

Operator	Description	Comparing	Returns
==	Equal to	5 == 8 5 == 5	false true
===	Equal value and equal type	5 === "5" 5 === 5	false true
!=	Not equal	5 != 8	true
!==	Not Equal value or not equal type	5 !== "5" 5 !== 5	true false
>	Greater than	5 > 8	false
<	Less than	5 < 8	true
>=	Greater than or equal to	5 >= 8	false
<=	Less than or equal to	5 <= 8	true

Control Structures

Conditional

- if/else
- switch

Looping

- for
- (do) while

If / else

The if/else condition is a code structure that allows you to check whether an expression is true or false and execute different code depending on the result.

If the "if" statement is false, the "else" block will be executed

```
var guest = 'Frodo';

if (guest == 'Frodo') {
  alert('It is the ring-keeper!')
}
else {
  alert('It is someone else.')
}
```

The "else" block will only be executed, when all "if" and "if else" blocks are false

```
var guest = 'Frodo';

if (guest == 'Frodo') {
  alert('It is the ring-keeper!')
}
else if (guest == 'Sam') {
  alert('It is his friend Samweis!')
}
else {
  alert('It is someone else.')
}
```

Switch

The switch statement is a selection statement that chooses a single block of code to execute from a list with predefined options.

```
var name = 'Gimli';

switch (name) {
  case 'Legolas':
    type = 'Elve';
    break;
  case 'Gimli':
    type = 'dwarf';
    break;
  default:
    type = 'undefined';
}
```

Because the value matches the second case, the variable "type" will be changed to "dwarf".

for

The for loop will execute a block of code as many times as it is declared.

The for loop is built up this way:

for (start; ending condition; increment after each execution)

```
var number = 0;

for (var i = 0; i < 10; i++) {
  number += 1;
}
```

What value will the "counter" variable have, once it is finish?

```
var seconds = 10;
var counter = 0;

for (var i = 0; i < seconds; i++) {
  counter += 5;
}
```

(do) while

The while or do while loop statement runs a block of code as long as the condition is true.

Because the condition is false, "counter" will not be increased.

```
var counter = 0;
var condition = true;

while (!condition) {
  counter++;
}
// counter = 0
```

Because the block will be executed at least once "counter" increased by 1.

```
var counter = 0;
var condition = false;

do {
  counter++;
} while (!condition)
// counter = 1
```

Events

An HTML event is something that happens to an HTML Element and JavaScript lets you execute code when such events are detected.

Here are some examples of HTML events:

- An HTML web page has finished loading
- An HTML input field was changed
- An HTML button was clicked

```
<button onclick="alert('Button clicked!')"></button> <!-- This is okay-->

<input onchange="myFunction()"></button> <!-- This is better -->
```

Functions

A JavaScript function is a block of code designed to perform a particular task.
A JavaScript function is executed when "something" invokes it (calls it).

```
function function_name(argument) {
  // body...
}
```

```
function myFunction() {
  document.getElementById('output').innerHTML = 'Input has changed'
}
```

```
var days = calculateYearDays(5); // days = 1'825

function calculateYearDays(years) {
  return years * 365;
}
```

JavaScript – Changing CSS

```
function myFunction(){
    document.getElementById("title").style.color = "blue";
    document.getElementById("title").style.fontFamily = "Arial";
    document.getElementById("inputfield").style.backgroundColor= "#3CBC8D";
    document.getElementById("inputfield").style.color= "white";
    document.getElementById("inputfield").style.boxSizing = "border-box";
    document.getElementById("inputfield").style.border= "none";
}
```

JavaScript – Changing HTML

```
function changeHtml() {
  document.getElementById('title').innerHTML = "New Text";
  document.getElementById('container').innerHTML = "<p>New Element</p>";
  document.getElementById('button').outerHTML = "<p>Button is now gone.</p>";
  document.getElementById('button').id = "New Id";
}
```

Arrays

An Array is a data structure, which can store more than one value at a time.

Class variables which can store only one value at a time

```
var name1 = 'Merry';
var name2 = 'Pippin';
var name3 = 'Sam';
```

While an Array can store mulitple

```
var names = ['Merry', 'Pippin', 'Sam'];
```

Access Array Element

- Array index start at 0.
- [0] is the first element. [1] is the second element.

```
//            [0]       [1]     [2]
var names = ['Merry', 'Pippin', 'Sam'];

var output = 'Frodos companion is ' + names[2];
```

Change Array Element directly

Array Elements can be changed by accessing them directly with the index.

```
var names = ['Merry', 'Pippin', 'Sam'];

names[1] = 'Peregrin';
// names = ['Merry', 'Peregrin', 'Sam']
```

Change Array Element with functions

Array Elements can also be changed with predefined functions.

```
var names = ['Merry', 'Pippin', 'Sam'];
```

Adding elements at the end of Array

```
names.push('Frodo'); // ['Merry', 'Pippin', 'Sam', 'Frodo']
```

Delete element at the end of Array

```
names.pop(); // ['Merry', 'Pippin', 'Sam']
```

Adding elements at the start of Array

```
names.unshift('Frodo', 'Merry'); // ['Frodo', 'Merry', Pippin', 'Sam']
```

Delete element at the start of Array

```
names.shift(); // ['Pippin', 'Sam']
```

Delete Array Element

Deleting a specific element in an Array works with the **splice()** function.

Splice() takes two arguments:
The index and how many elements from there it has to delete

```
Array.splice(index, itemCount);
```

```
var names = ['Merry', 'Pippin', 'Sam'];

names.splice(0, 1);

// names = ['Pippin', 'Sam']
```

Array length

The length property of an array returns the length of an array (the number of array elements).

```
var names = ['Merry', 'Pippin', 'Sam'];
```

Accessing first & last element of Array

To access the first element of an Array we can as we already know get it by index.

```
var firstElement = names[0];
```

Looping Array

We can iterate an entire Array with a **for** loop.

```
var places = ['shire', 'rohan', 'mordor', 'isengard'];
var listItems;

for (var i = 0; i < places.length; i++) {
  listItems += '<li>' + places[i] + '</li>';
}

document.getElementById('list').innerHTML = '<ul>' + listItems + '</ul>';
```