IN5320 - Development in Platform Ecosystems

Lecture 2: *HTML, CSS, JavaScript*

27th of August 2018

Department of Informatics, University of Oslo
Magnus Li - magl@ifi.uio.no

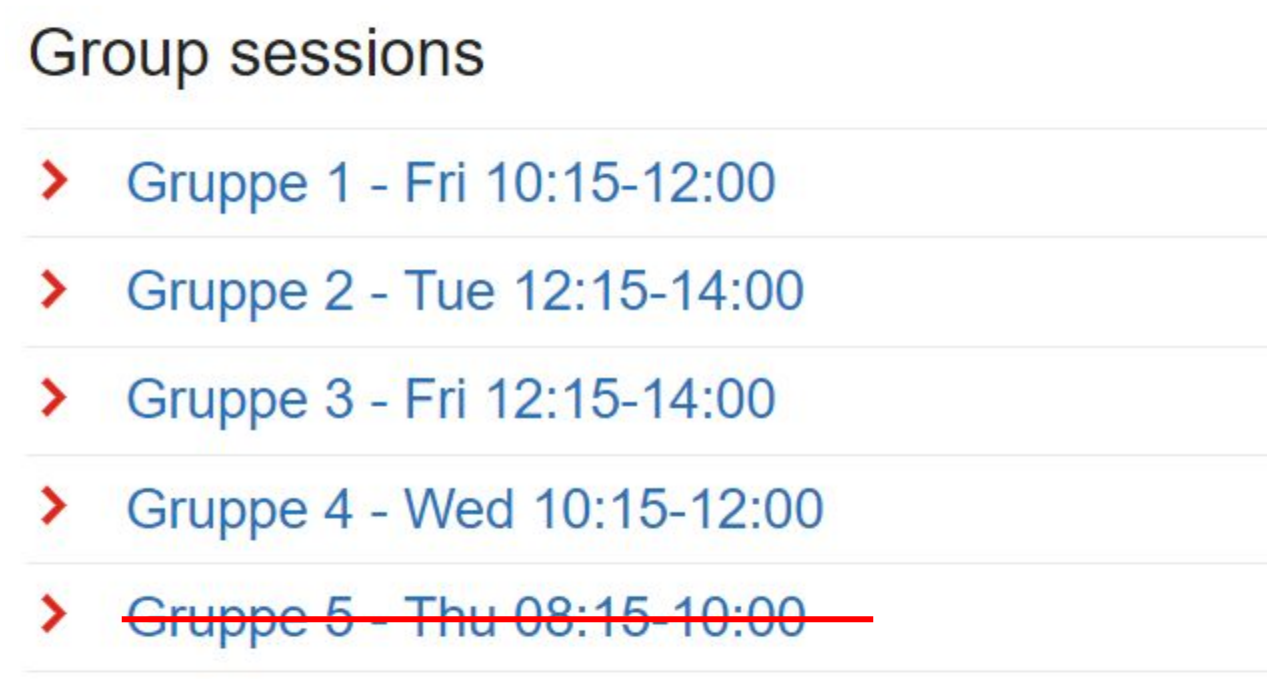
Today's lecture

1. Web, servers, clients, and programming
2. HTML and CSS basics
3. The Document Object Model (DOM)
4. JavaScript
5. jQuery

Seminar groups

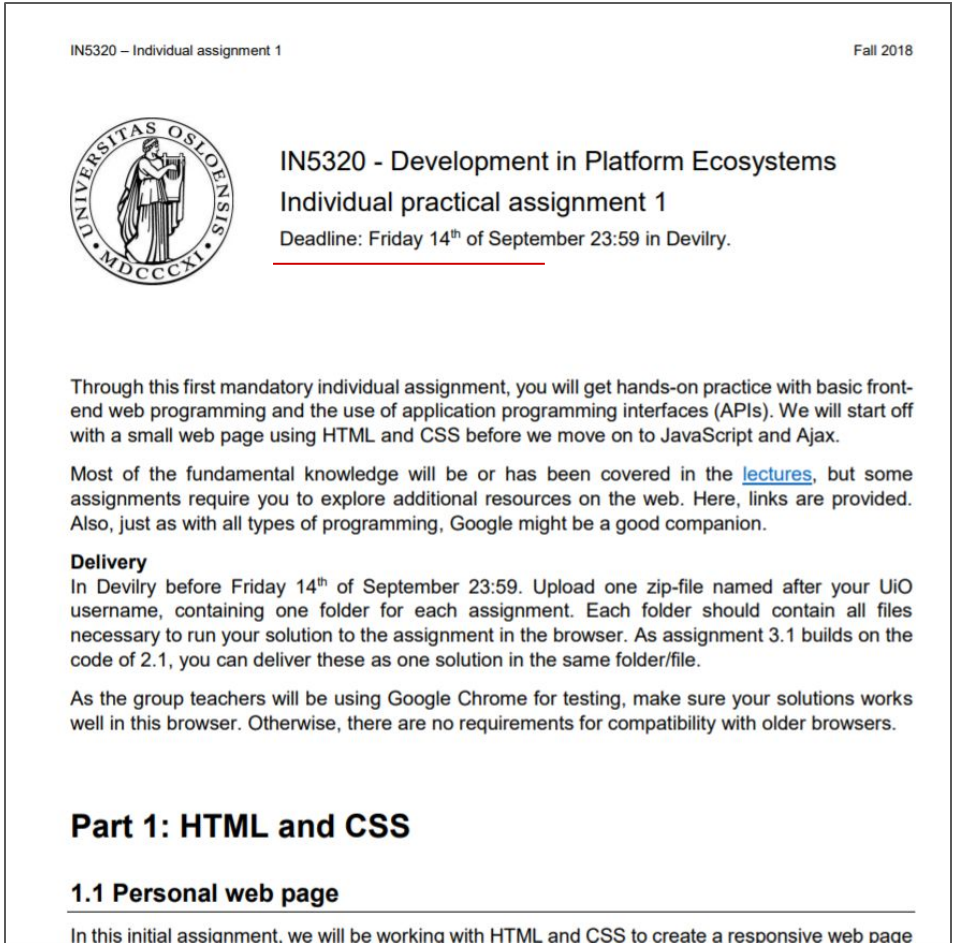
	34	No seminar
	35	*Help with individual assignment*
	36	
Assignment 1 ⇨	37	
	38	Workshop React
Assignment 2 ⇨	39	*Help with individual assignment*
	40	Find group and select case
	41	*Help with group project*
	42	
	43	Presentation 1
	44	*Help with group project*
	45	
	46	Presentation 2
	47	*Help with group project*
Final pres. ⇨	48	

Seminar groups


Group sessions

- Gruppe 1 - Fri 10:15-12:00
- Gruppe 2 - Tue 12:15-14:00
- Gruppe 3 - Fri 12:15-14:00
- Gruppe 4 - Wed 10:15-12:00
- ~~Gruppe 5 - Thu 08:15-10:00~~

Mandatory individual assignment 1



IN5320 – Individual assignment 1 Fall 2018

IN5320 - Development in Platform Ecosystems

Individual practical assignment 1

Deadline: Friday 14th of September 23:59 in Devilry.

Through this first mandatory individual assignment, you will get hands-on practice with basic front-end web programming and the use of application programming interfaces (APIs). We will start off with a small web page using HTML and CSS before we move on to JavaScript and Ajax.

Most of the fundamental knowledge will be or has been covered in the lectures, but some assignments require you to explore additional resources on the web. Here, links are provided. Also, just as with all types of programming, Google might be a good companion.

Delivery

In Devilry before Friday 14th of September 23:59. Upload one zip-file named after your UiO username, containing one folder for each assignment. Each folder should contain all files necessary to run your solution to the assignment in the browser. As assignment 3.1 builds on the code of 2.1, you can deliver these as one solution in the same folder/file.

As the group teachers will be using Google Chrome for testing, make sure your solutions works well in this browser. Otherwise, there are no requirements for compatibility with older browsers.

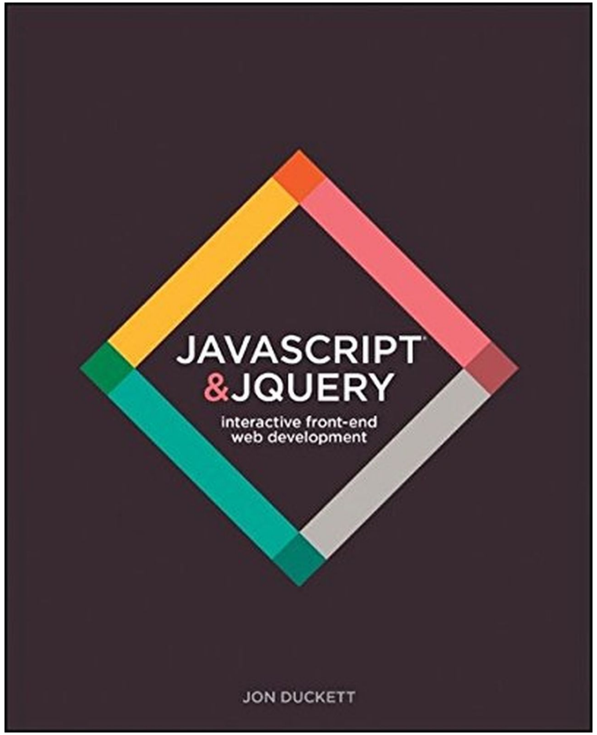
Part 1: HTML and CSS

1.1 Personal web page

In this initial assignment, we will be working with HTML and CSS to create a responsive web page

Recommended readings

HTML&CSS
design and build websites
JON DUCKETT



Recommended readings

codecademy

HTML & CSS

JavaScript

Learn ReactJS: Part I

Learn ReactJS: Part II

Web programming

- Involves several languages and frameworks for programming, scripting, database queries, markups and styles

Client-side

HTML CSS JS

React

jQuery

ANGULARJS by Google

Server-side

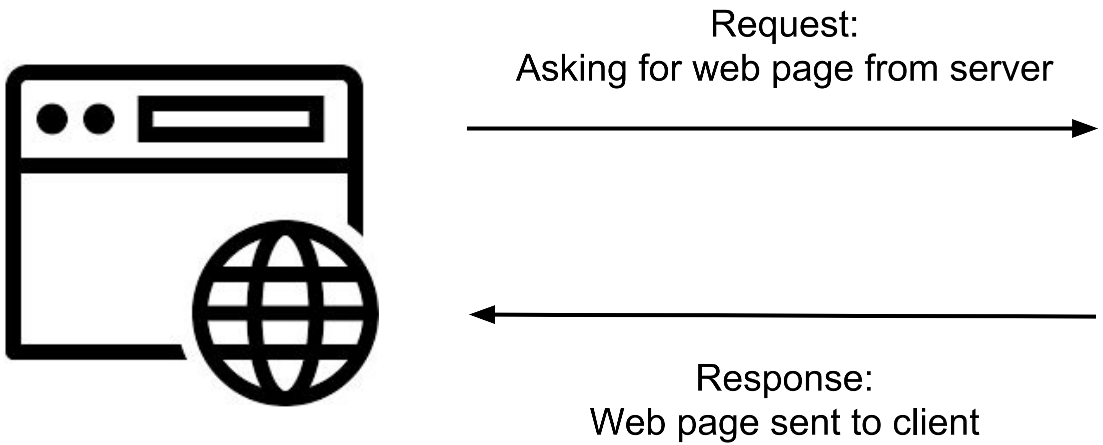
python Java php

node JS RAILS

ERLANG Cf

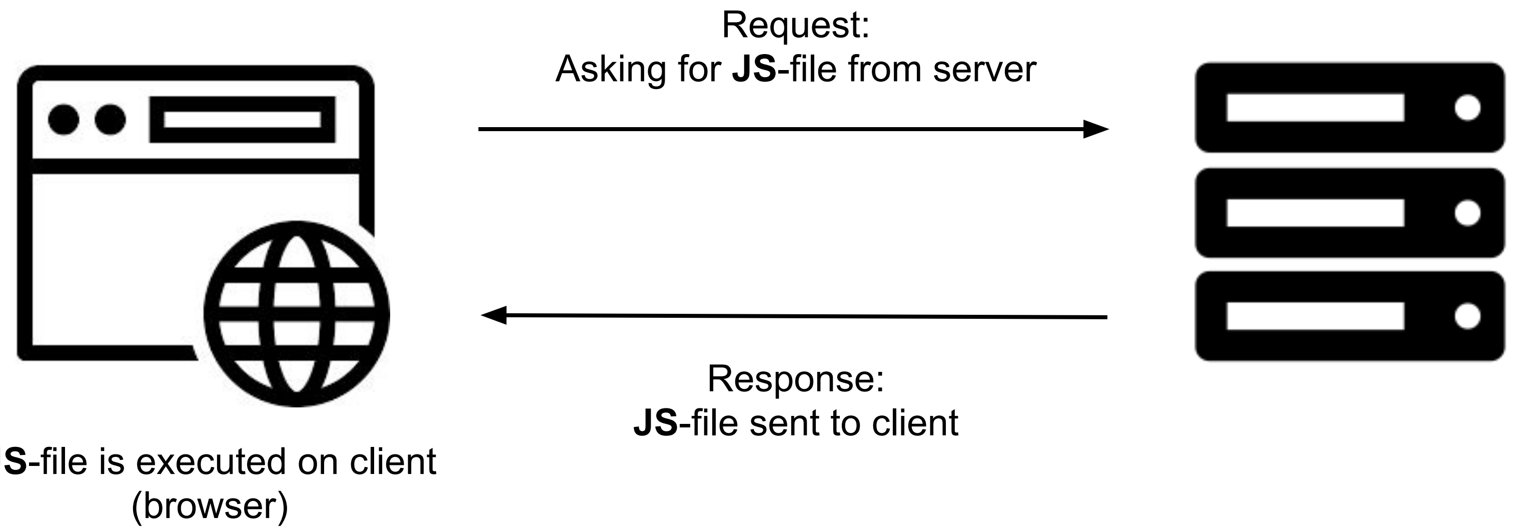
SQL

Servers and clients



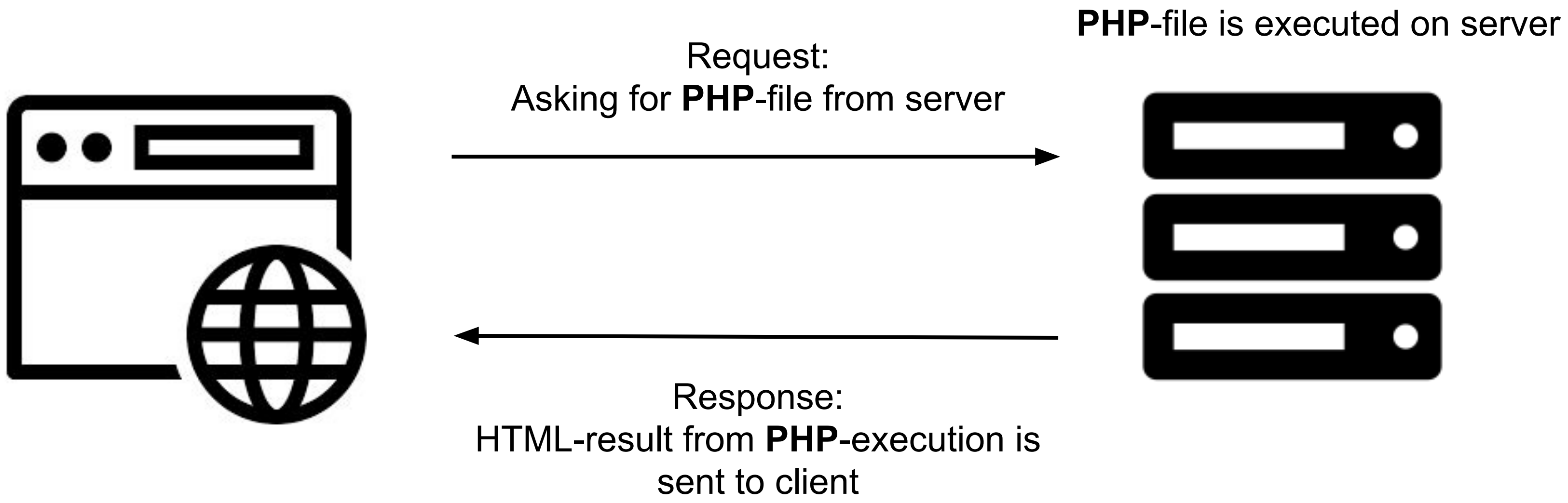
Client-side languages

- JavaScript (JS) is a client-side language.
- This means that it is executed on the client. (Often in the web-browser)


Request:
Asking for **JS**-file from server
Response:
JS-file sent to client
JS-file is executed on client (browser)

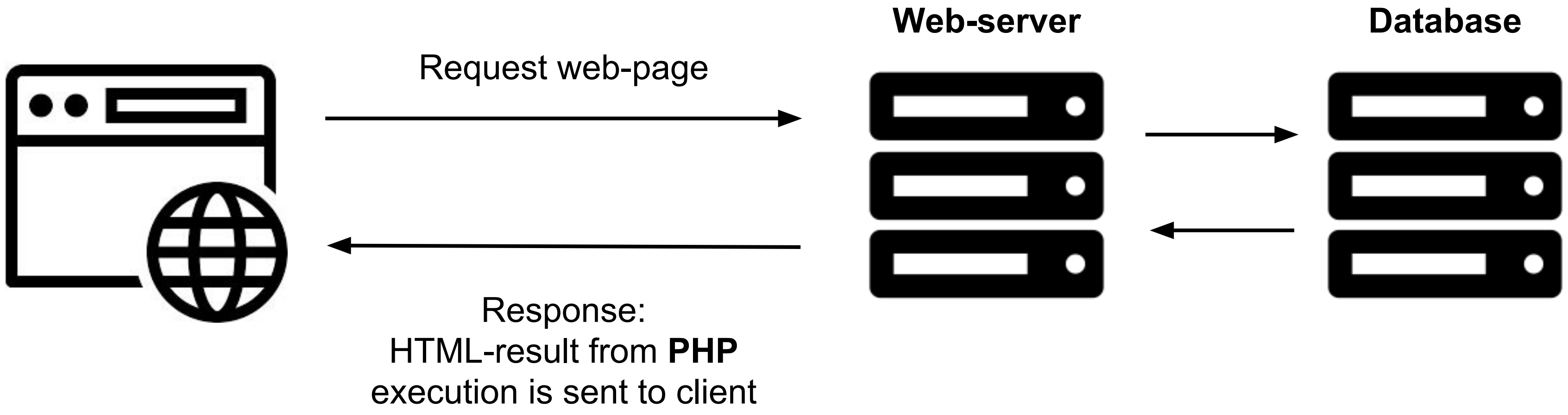
Server-side languages

- PHP is a server-side language.
- This means that it is executed on the server.
- The response contains the output from the execution in HTML


PHP-file is executed on server
Request:
Asking for **PHP**-file from server
Response:
HTML-result from **PHP**-execution is sent to client

Databases

- The web-server often communicate with a database
- A SQL query from PHP can be used to request data from the database
- It is then returned to the client in HTML-format



Client-side, front-end, server-side, back-end

- The terms client-side and front-end, and server-side and back-end partly overlaps

Client-side
Code that runs on the users computer. Often in the web-browser.

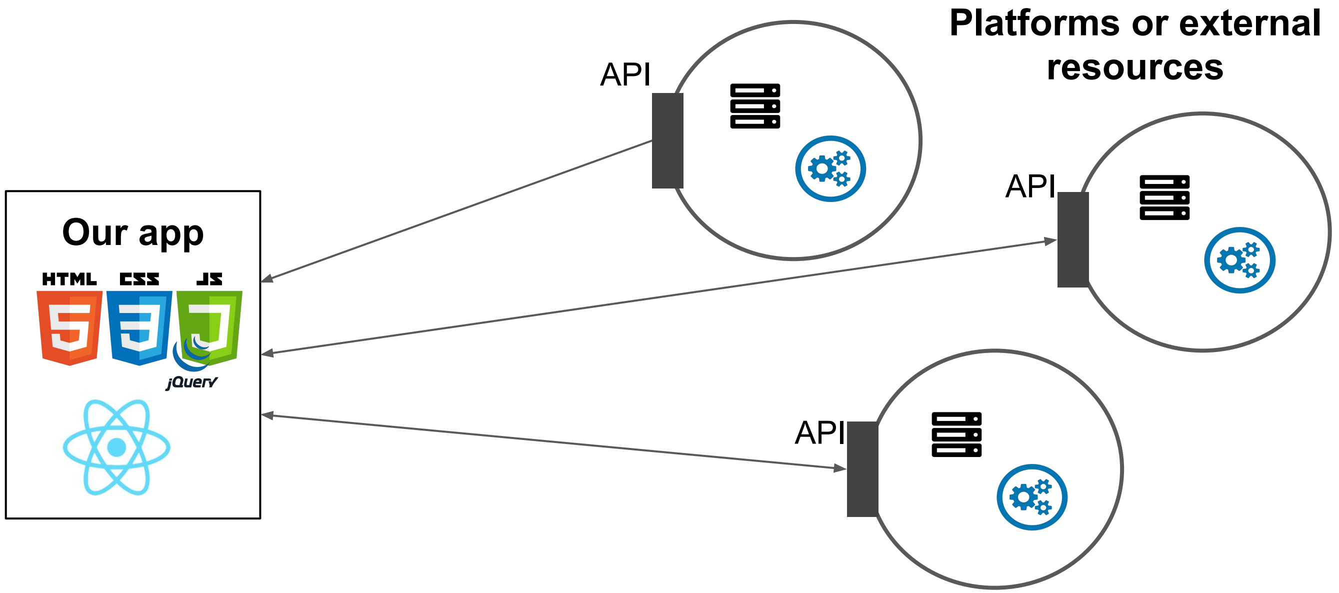
Server-side
Code that runs on the server, rather than the client (the user's computer/browser).

Front-end
All code/components/processes that face the end-user. Are generally client-side and hence executed in the browser.

Back-end
Underlying processes that do the "crunching" of data to be sent to the front-end. / processes that the user are "unaware" of. This often happens on the server-side, but not always.

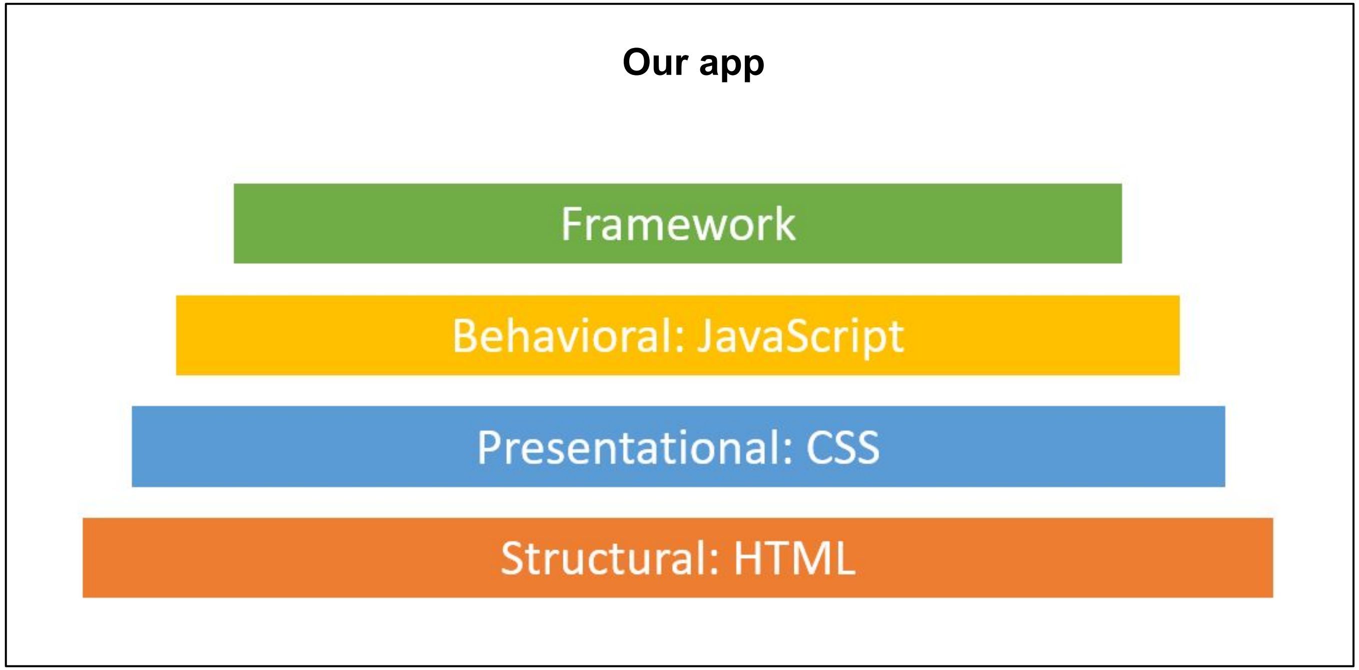
In this course

- This course focuses on client-side and front-end programming.
- We will use HTML, CSS, JavaScript, and some frameworks such as react.js to create web-based apps and prototypes.
- These will communicate with server-side and back-end systems through Application Programming Interfaces (APIs).



In this course

- We will use HTML for markup, CSS for styling and JavaScript for programming.
- We will also look at jQuery (JavaScript library) and React.js (JavaScript framework).



HTML and CSS basics

Hypertext Markup Language (HTML)

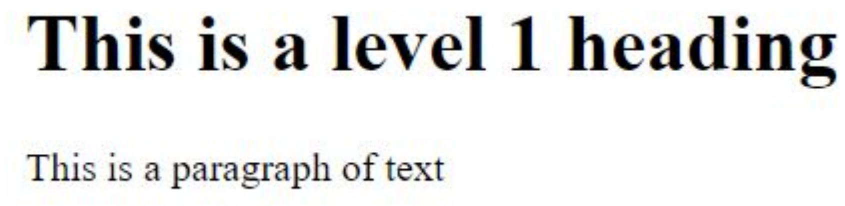
- Standardized markup-language that allows us to structure documents to be processed by the browser
- The standard contains a set of markups / tags that all browsers recognize.
- Elements are defined by an opening tag <p> and an closing tag </p>

```html
<h1>This is a level 1 heading</h1>

<p>This is a paragraph of text</p>
```

Hypertext Markup Language (HTML)

```
<h1>This is a level 1 heading</h1>
<p>This is a paragraph of text</p>
```



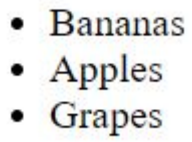
This is a level 1 heading

This is a paragraph of text

Hypertext Markup Language (HTML)

- There are a large set of default elements
- To create lists, tables, images, and hyperlinks
- For semantics and structure: sections, menus, headers, etc.

```html
<ul>
  <li>Bananas</li>
  <li>Apples</li>
  <li>Grapes</li>
</ul>
```



- Bananas
- Apples
- Grapes

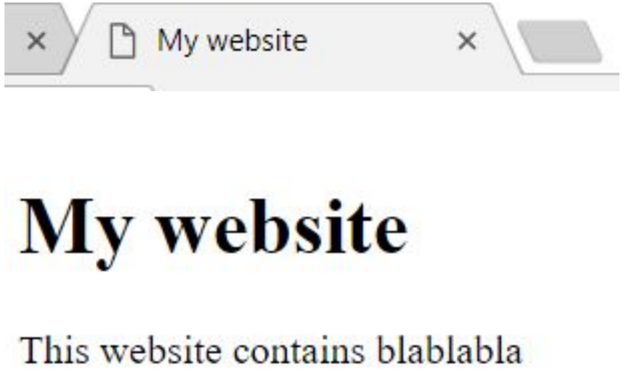
Hypertext Markup Language (HTML)

- Overall structure

```
<!DOCTYPE html>
<html>
<head>
  <title>My website</title>
</head>

<body>
  <h1>My website</h1>
  <p>This website contains blablabla</p>
</body>

</html>
```



My website

My website

This website contains blablabla

Cascading Style Sheets (CSS)

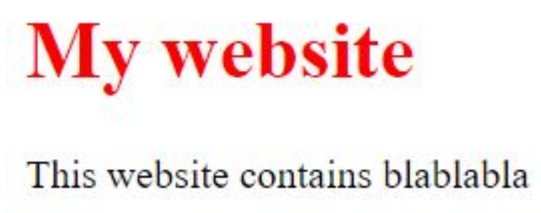
- With CSS, we can define our own styles for how HTML-elements should be displayed by the browser
- We do this by referring to an element, and define some rules for the browser to follow.

index.html

```
<body>
  <h1>My website</h1>
  <p>This website contains blablabla</p>
</body>
```

style.css

```
h1 {
  color: red;
}
```


My website

This website contains blablabla

Cascading Style Sheets (CSS)

- CSS can be written internally in the HTML document, or externally in its own .css file.

external

```
<head>
 <link rel="stylesheet" href="/html/styles.css">
</head>
```

internal

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

Classes and identifiers

- By referring to the element type, such as <h1>, all elements of that type will be affected.
- We can also add custom identifiers and classes to our HTML-elements to style specific elements or groups of elements
- Identifiers should be unique for one element
- Classes can encompass several elements.

```
<p id=”example_one”>
<p id=”example_two”>
<p id=”example_three”>
```

```
<p class=”example_class”>
<p class=”example_class”>
<p class=”example_class”>
```

Classes and identifiers

- We can refer to the elements in css by using # for identifiers and . for classes

Index.html

```
<p id="intro_text">This website contains blablabla</p>
```

style.css

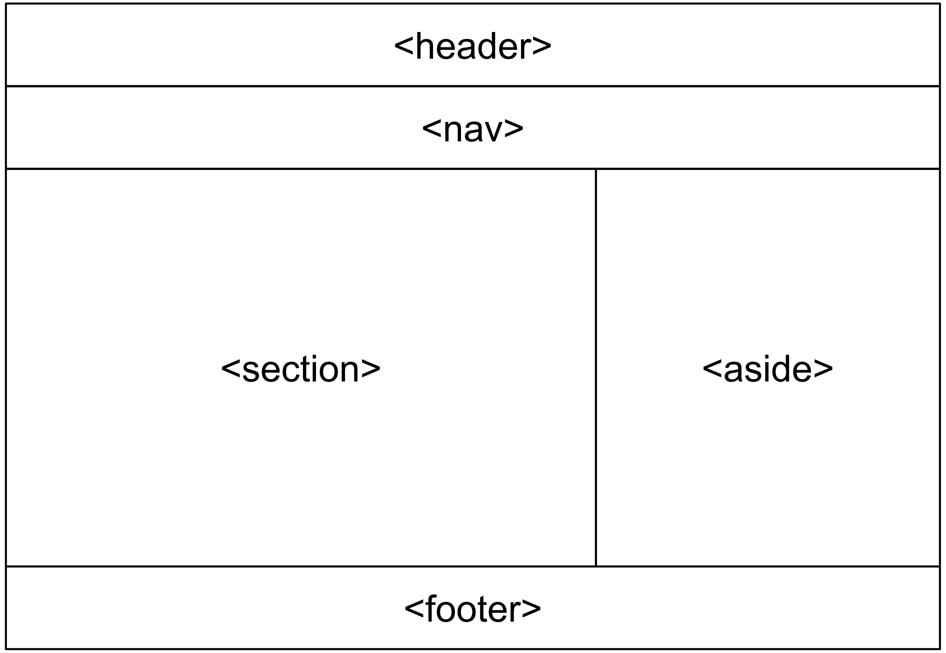
```
#intro_text {
  color: red;
}
```

```
<p class="text_snippet">This website contains blablabla</p>
```

```
.text_snippet {
  color: red;
}
```

Semantics

- Standard elements used to structure the document
- For the developer
- For robots and screen-readers



Example: basic page with menu

- Live code example.

JavaScript

JavaScript

- “JavaScript is the programming language of HTML and the Web”
- Allows us to create dynamic functionality on our web pages.
- It is a high-level and interpreted programming language
- It is multi-paradigm: allows imperative and functional programming

JavaScript - basics

- The basic syntax of JavaScript is quite similar to languages you already know

```
//if
if (x < y) {
}

//loops
for (var i = 0; i < 10; i++) {
}
while (i < 10) {
}
```

```
//arrays
var list = [1, 2, 3, 4, 5];
var list = ["Hey", "whats", "up"];
```

JavaScript - variables

- Unlike C++ or Java, JavaScript is weakly typed
- This means that we do not have to define the data type of variables.
- JavaScript will automatically recognize the relevant type, and make the appropriate adjustments.

```
var list = [1, 2, 3, 4, 5]; //array
var test = "hey"; //string
var test = 1; //int
var test = 1.22; //double
var test = true; //boolean
```

JavaScript - variables

- Unlike C++ or Java, JavaScript is weakly typed
- This means that we do not have to define data types for our variables.
- JavaScript will automatically recognize the relevant type, and make the appropriate adjustments.

```
var x = 2;
var y = 2;
var z = "one";
console.log(x+y+z);
```

```
4one
```

```
var x = 2;
var y = 2;
var z = "1";
console.log(x+y+z);
```

```
41
```

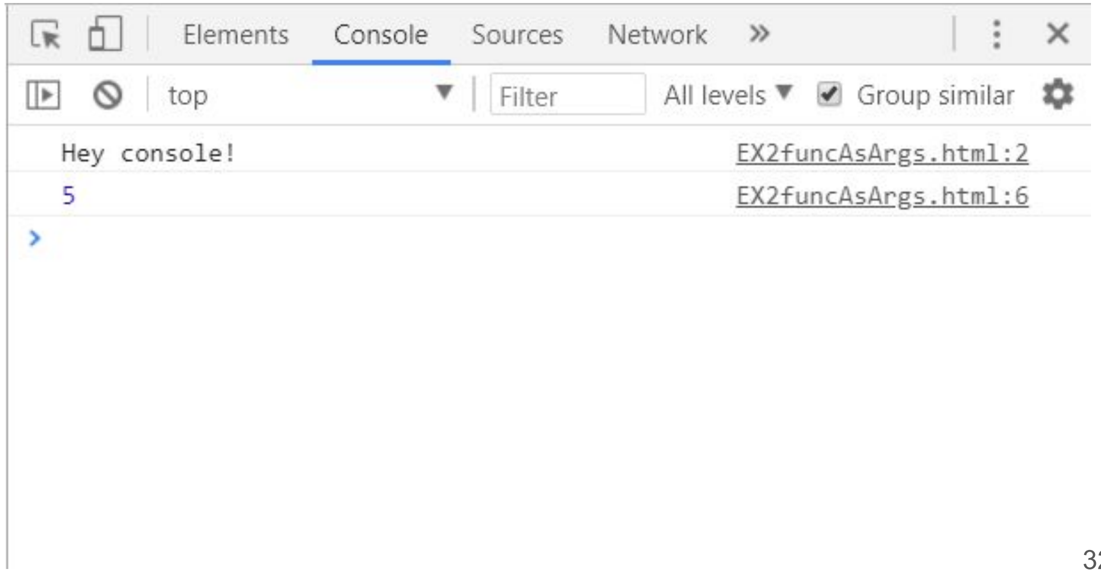
```
var x = 2;
var y = 2;
var z = true;
console.log(x+y+z);
```

```
5
```

JavaScript - console

- We can use the console found in our web browser for debugging.
- In Chrome, it can be opened by pressing ctrl + shift + J
- Similar to System.out.println() in Java, we can use Console.log() to print to the console in JavaScript.

```
console.log("Hey console!");
var x = 2;
var y = 2;
var z = true;
console.log(x+y+z);
```

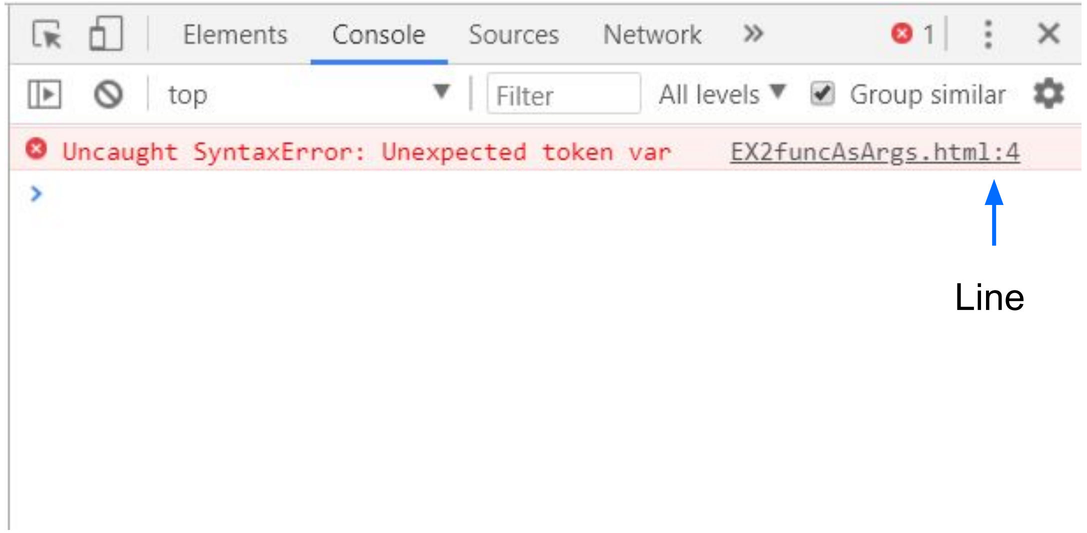


```
Hey console!                                    EX2funcAsArgs.html:2
5                                               EX2funcAsArgs.html:6
>
```

JavaScript - console

- The console will also display errors and warnings
- This includes on which line it occurs

```
2 console.log(“Hey console!”);
3 var x = 2<
4 var y = 2;
5 var z = true;
6 console.log(x+y+z);
```



```
Uncaught SyntaxError: Unexpected token var          EX2funcAsArgs.html:4
```

JavaScript - functions

- JavaScript supports both named and anonymous functions

Named function

```
function example() {
    //some code
}
```

Anonymous function

```
function(){
    //some code
});
```

JavaScript - passing functions as arguments

- JavaScripts allows passing functions as arguments to other functions
- Both anonymous and named functions.

```
someFunction(function(){
        //some code
    });
```

```
someFunction(example);
function example() {
    //some code
}
```

JavaScript - passing functions as arguments

- We typically use this when we want to define some event that will be triggered by a click or some other interaction with elements on the web page

```
button.addEventListener("click", function(){
        //some code
    });
```

```
button.addEventListener("click", example);
function example() {
    //some code
}
```

JavaScript - example: passing functions as arguments

- Here we have created a function that takes some list of something and a function as argument.
- It will go through the list and apply the provided function on each element, storing the returned value from the function in results.

```
function applyToElements(func, list) {
    var result = [];
    for (let i = 0; i < list.length; i++) {
       result.push(func(list[i]));
    }
    return result;
}
```

JavaScript - example: passing functions as arguments

- Here we have created a function that takes some list of something and a function as argument.
- It will go through the list and apply the provided function on each element, storing the returned value from the function in results.

```
function applyToElements(func, list) {
    var result = [];
    for (let i = 0; i < list.length; i++) {
       result.push(func(list[i]));
    }
    return result;
}
```

```
var numbers = [1, 2, 3, 4, 5, 6];
applyToElements(function(elem) {
    return ++elem;
}, numbers);
```

JavaScript - example: passing functions as arguments

- Here we have created a function that takes some list of something and a function as argument.
- It will go through the list and apply the provided function on each element, storing the returned value from the function in results.

```
var strings = ["Awk", "Ada", "Assembler", "Limbo", "C", "Caml", "Chill", "Cobol"];
var upperCase = applyToElements(function(elem) {
    return elem.toUpperCase();
}, strings);
console.log(upperCase);
```

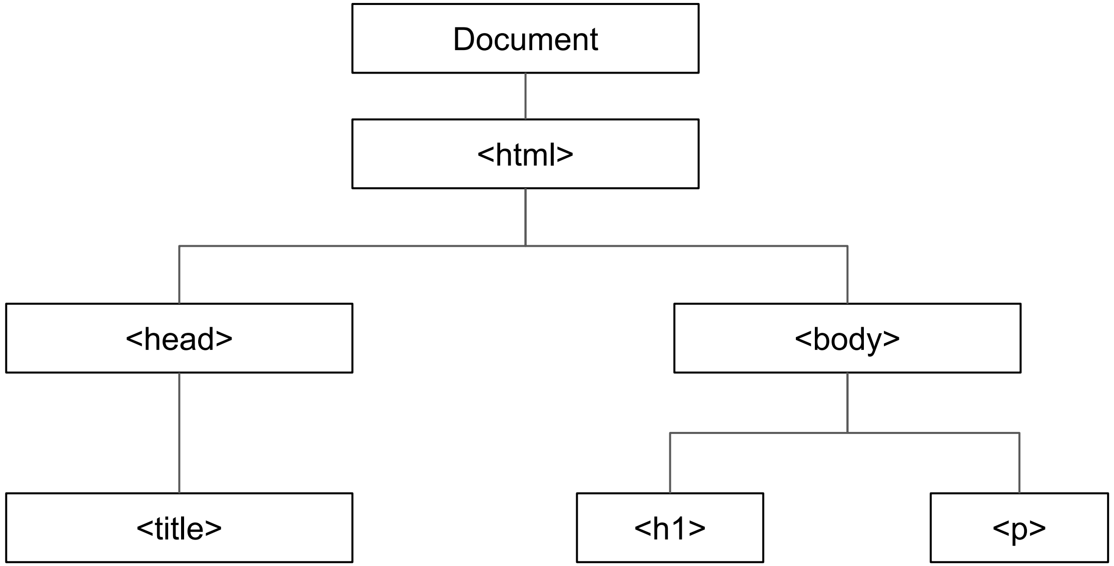
JavaScript and DOM

Document Object Model (DOM)

- When the web-browser loads a html-file, a Document Object structure is created.
- The DOM forms a tree structure of objects where every HTML-element is stored in nodes

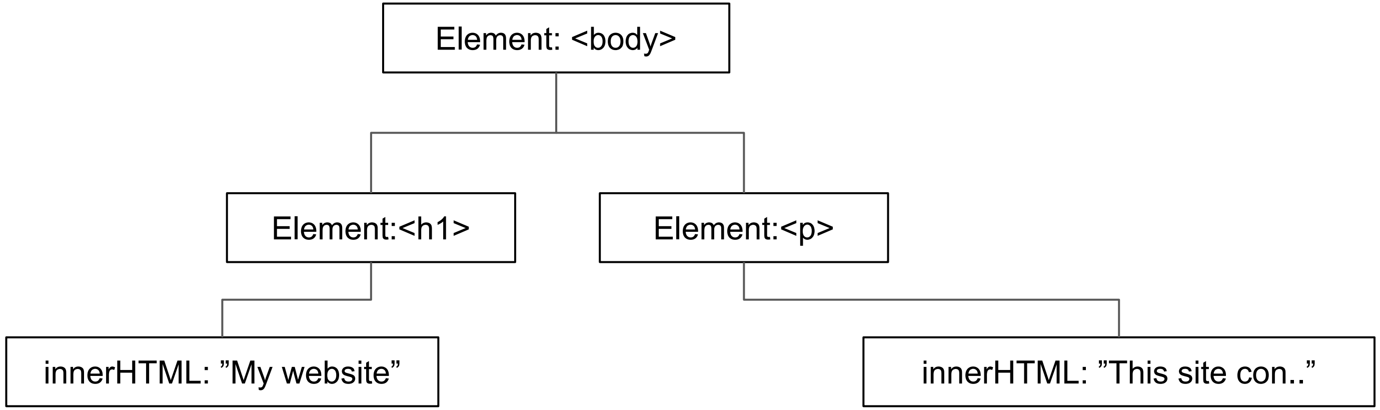
```
<html>
<head>
  <title>My website</title>
</head>
<body>
  <h1>My website</h1>
  <p>This website con</p>
</body>


</html>
```



Document Object Model

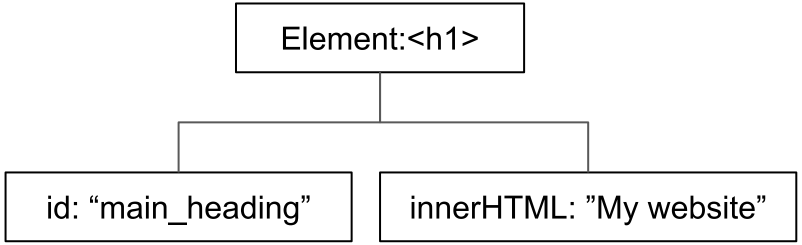
- Nodes with attributes and content is also a part of the tree
- For example, the inner content of the element is stored in a child node



JavaScript + DOM

- *"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."*
- JavaScript can easily access and modify all nodes.

```
<h1 id="main_heading">My website</h1>
```



```
var elem = document.getElementById("main_heading");
elem.innerHTML = “Hello”;
```

JavaScript + DOM: example - input + output

JavaScript + DOM: example - input + output

```
<body>
  <h1>My website</h1>
  <input type="text" id="search" placeholder="Keyword">
  <button id="search_button">Search!</button>
  <p id="result">This website contains blablabla</p>
</body>

<script>
    var button = document.getElementById("search_button");
    button.addEventListener("click", function(){
        var resElem = document.getElementById("result");
        var searchString = document.getElementById("search").value;
        resElem.innerHTML = searchString;
    });
</script>
```

jQuery

- The most widely used JavaScript library
- Simplifies navigation and manipulation of the DOM and CSS.
- Other functionality such as animations and ajax.

Pure JavaScript

```
var elem = document.getElementById("main_heading");
elem.innerHTML = “Hello”;
```

jQuery

```
var elem = $("#main_heading");
$(elem).html(“Hello”);
```

jQuery

- Can be imported by referring to it in the head of the HTML-file.
- Either local .js-file or online

Online from Google

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js" ></script>
<head>
```

Local

```
<head>
<script src="scripts/jquery-3.3.1.min.js" ></script>
<head>
```