Päivä 2 - HTML, CSS, JS

2021-10-28
AaltoPRO - Websovelluskehitys
Lassi Haaranen

Ohjelma

- Aamupäivä 9-12
  - HTML
  - CSS
  - HTTP
- Lounas 12-13
- Iltapäivä 13-15:30
  - JavaScript
  - jQuery

- Päivän harjoitukset: https://github.com/aaltopro-weblearners/project-02-html-css-js

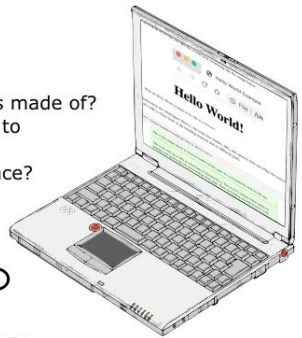
HTML

Olet täällä

Client-side JavaScript
How to program and how to use JavaScript to add interaction to web pages?

Server-side runtime environment
How to create server-side programs?
Our runtime is mainly Deno

Web pages
What are web pages made of?
How to add content to a page and how to change its appearance?



Requests and responses
What is a server and how pages get transferred between the server and the client (HTTP)?

Routing & Sessions
How to get the correct page or resource from a server and how to store data in the browser?

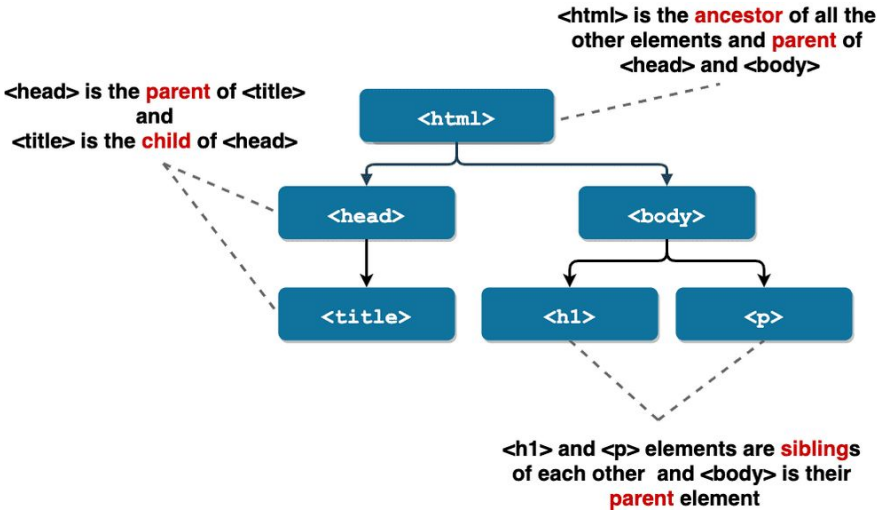
Libraries
What different types of libraries exist and what can you create with them?

Data
How to retrieve, modify, and store data on the server?

+ Testing, deployment, logging, analytics, and so much more...

HTML

- https://acos.cs.aalto.fi/html/codeannotation/code-annotation-iwdap/html_structure



- Introduction to HTML - Learn web development | MDN

HTML Validiteetti (validity)

- Jokaisessa HTML-dokumentissa pitää olla tietyt elementit (esim. <html> ja <body>)
- Tämän lisäksi elementtien pitää alkaa ja loppua (joitakin poikkeuksia) ja niiden hierarkian pitää olla sallittua
- Validiteetin voi tarkistaa esim. https://validator.w3.org/

HTML - Block & Inline -elementit

- Näkyvät HTML-elementit ovat joko block-tyyppisiä (ennen ja jälkeen elementtiä rivinvaihto) tai inline-elementtejä (ei rivinvaihtoja)
- Inline elements - HTML: HyperText Markup Language | MDN
- Block-level elements - HTML: HyperText Markup Language | MDN

	Sallittu?	Esimerkkejä
Block-elementti toisen block-elementin sisällä	Kyllä	Content
Inline-elementti block-elementin sisällä	Kyllä	Content
Inline-elementti toisen inline-elementin sisällä	Kyllä	Content
Block-elementti inline-elementin sisällä	Ei	Content

HTML - Entiteetit & Kommentit

- Entity - MDN Web Docs Glossary: Definitions of Web-related terms | MDN
- Tietyt merkit eivät ole suoraan käytettävissä, jolloin dokumenttiin tarvitsee lisätä merkit HTML entiteeteillä Esim.
  - & ⟶ &
  - < ⟶ <
- Täysi listaus entiteeteistä: https://html.spec.whatwg.org/multipage/named-characters.html#named-character-references
- Kommentit `<!-- tämä on kommentti -->`
  - Huom. kommentit ovat nähtävissä selaimessa tarkastelemalla

HTML Attribuutit (attributes)

- Elementeillä voi olla lisämääreitä eli attribuutteja.
- Alla <h1> elementillä on class-attribuutti, jolla arvo `title` ja style-attribuutti, jolla arvo `font-size: 200%;`

```
<h1 class="title" style="font-size: 200%;">

    Hello, World!

</h1>
```

HTML Elementtejä

- Käytettävät attribuutit riippuvat elementistä
- HTML attribute reference - HTML: HyperText Markup Language | MDN

```
<a href="https://www.aalto.fi/">Aalto University's front page</a>
```

```
<ul class="list">
    <li>Flour</li>
    <li>Salt</li>
</ul>
```

```
<img src="phalaenopsis.jpg" alt="Phalaenopsis. Photo by Bill Higham." />
```

HTML id ja class-attribuutit

- id - HTML: HyperText Markup Language | MDN

```
<h1 id="main-title">Ids are unique</h1>
```

```
h1#main-title {
    color: red;
}
```

```
let NoteElement document.querySelector("#lastNote");
```

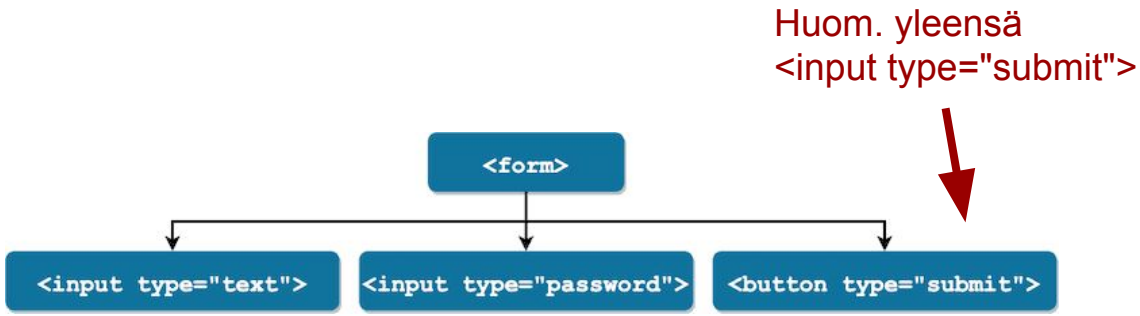
- class - HTML: HyperText Markup Language | MDN

```
<p class="highlight">This is some text</p>
```

```
.highlight {
    border: 1px solid green;
}
```

HTML Forms

- [<form>: The Form element - HTML: HyperText Markup Language | MDN]



HTML Form-harjoitus - project-02-html-css-js/synth_form.html

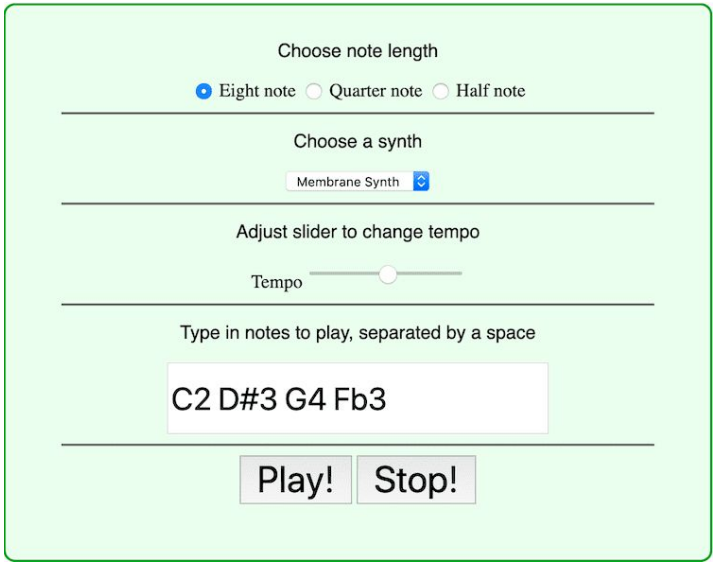
ks. <input>: The Input (Form Input) element - HTML: HyperText Markup Language | MDN

It should include a possibility to select the note length and one of three available synthesizers. The form should contain a slider that allows the user to control tempo. There should be a text input where the user can type in notes (see next paragraph). Finally, the form should include buttons for starting and stopping play.

The notes are typed into the form by giving the letter for each note, followed by a number between 2 and 9, e.g. C4. The user can type in multiple notes by separating them with spaces, e.g. "C2 D3 G4 F3". If you are familiar with music theory, you can also add sharp and flat notes using `#` and `b`, e.g. "C2 D#3 G4 Fb3".

Here is what the form should contain in more detail:

- Start by creating a <form> element in your template. The element should contain "synth-form" as its id. It also needs to include a method attribute with post and its action attribute should be #.
- The form should contain three radio inputs, which have the following `ids`: "eight-note", "quarter-note", and "half-note". The corresponding value attributes should be "8n", "4n", and "2n". All of the radio inputs should use the same name value of `note-length`. By default, the `eight-note` should be checked.
- The form should also contain a <select> element with the id of "synth-select". There should be three options with value "default-synth", "membrane-synth", and "pluck-synth".
- To adjust the tempo, the form should include an <input> element with the `type` attribute set to `range`. The element should specify the id of "tempo". Additionally, you need to set the `min` and `max` attributes to "1" and "30", respectively.
- The form should also present an <input> with the type of "text" with an id of "notes-to-play". As a tip, you can also set a default value for this, which could be "C4 D4 F4".
- Finally, you need to add two buttons. The first is an <input> button that specifies the type of "submit". The value of the input could be anything (e.g. "Play"). The second button is an <input> that has a "button" as its type and an id of "stop". Again, you can put anything for its value<, e.g. "Stop".



CSS

CSS - Cascading StyleSheets

- Tyylien määrittäminen HTML-sisällölle
- https://acos.cs.aalto.fi/html/codeannotation/code-annotation-iwdap/css_rule_example

- CSS basics - Learn web development | MDN
- CSS: Cascading Style Sheets | MDN

CSS - Spesifiteetti

Specificity - CSS: Cascading Style Sheets | MDN

```
<!DOCTYPE html>
<html lang="en">
    <head>
      <title>Another CSS Example</title>
      <style>
        body { color: green; }
        h1 { color: red; }
        p em { color: blue; }
      </style>
    </head>
    <body>
      <h1>A Heading in <em>Red</em></h1>
      <p>This paragraph has green text, but the emphasis is <em>blue</em>.</p>
    </body>
</html>
```

CSS - Specificity

```
body { font-size: 18px; }
div { padding: 10px; }
p { background-color: lightgray }
em { color: red; }
em.special { font-size: 150%; }
em#critical { color: blue; padding: 20px;}
```

```
<div>
    <p> This is a text paragraph inside &lt;div&gt; that also has a
        <em>couple</em>
        <em class="special">of different</em>
        <em id="critical" class="special">emphasis elements</em>.
    </p>
</div>
```

CSS - Värien määrittely

<color> - CSS: Cascading Style Sheets | MDN

- Kaikki nämä muuttavat tekstin punaiseksi

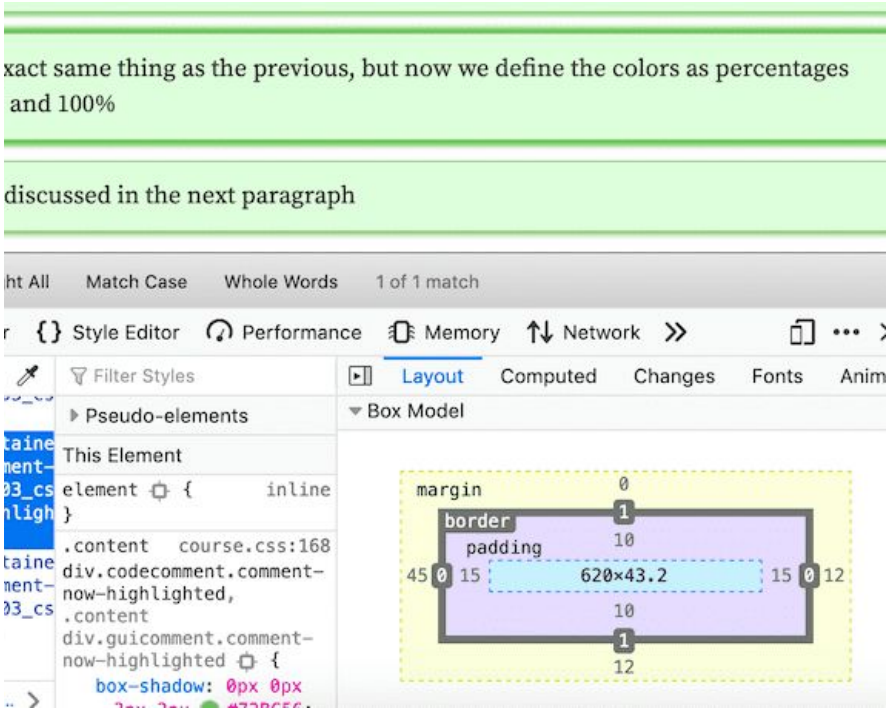
```
body {
    color: red;
    color: rgb(255, 0, 0);
    color: rgb(100%, 0%, 0%);
    color: #FF0000;
}
```

CSS - Box model

- Content, padding, border, margin

- [Introduction to the CSS basic box model - CSS: Cascading Style Sheets | MDN]

- [The box model - Learn web development | MDN]



CSS - Pseudo-class & pseudo-element

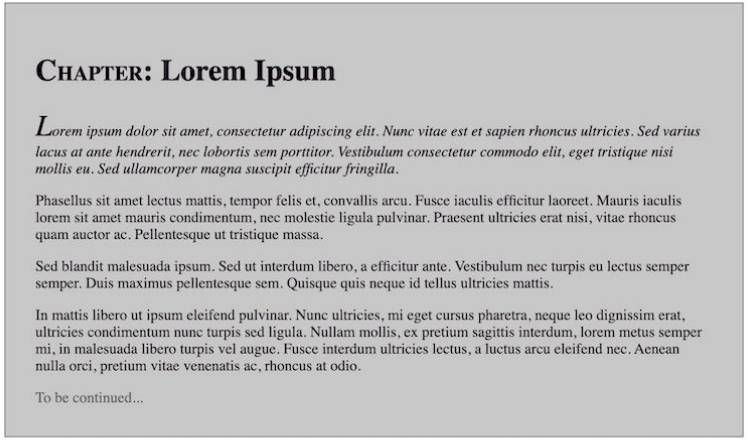
- Pseudo-classes - CSS: Cascading Style Sheets | MDN
- Pseudo-classes and pseudo-elements - Learn web development | MDN

```
button:hover {
  color: blue;
}

p::first-letter {
  font-size: 130%;
}
```

CSS - Harjoitus - project-02-html-css-js/css/hello_cascading.css

- The that wraps around all of the content should have a 1px wide solid border with a color of rgb(100,100,100). It should also have a padding that's 32px on all sides and margin of 16px all around it. It should also have a background color of rgb(200,200,200)
- Unless otherwise specified all the paragraphs should have the text in rgb(40,20,40).
- The heading text should be black which is rgb(0,0,0).
- Before the heading, there should be a text that reads "Chapter: " and it should have a font-variant small-caps. Hint: MDN has an article on font-variant property.
- The first paragraph after a heading should have italic font-style
- The first letter of the first paragraph should also have the font-size doubled to 36px
- Finally, after the last paragraph of the page, there should be text that says "To be continued..." that has the color rgb(100,100,100). Hint: MDN has an article describing :last-child pseudo-class. You can combine this with another pseudo-element to `add content that comes after` a particular selection



CHAPTER: Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc vitae est et sapien rhoncus ultricies. Sed varius lacus at ante hendrerit, nec lobortis sem porttitor. Vestibulum consectetur commodo elit, eget tristique nisi mollis eu. Sed ullamcorper magna suscipit efficitur fringilla.

Phasellus sit amet lectus mattis, tempor felis et, convallis arcu. Fusce iaculis efficitur laoreet. Mauris iaculis lorem sit amet mauris condimentum, nec molestie ligula pulvinar. Praesent ultricies erat nisi, vitae rhoncus quam auctor ac. Pellentesque ut tristique massa.

Sed blandit malesuada ipsum. Sed ut interdum libero, a efficitur ante. Vestibulum nec turpis eu lectus semper semper. Duis maximus pellentesque sem. Quisque quis neque id tellus ultricies mattis.

In mattis libero ut ipsum eleifend pulvinar. Nunc ultricies, mi eget cursus pharetra, neque leo dignissim erat, ultricies condimentum nunc turpis sed ligula. Nullam mollis, ex pretium sagittis interdum, lorem metus semper mi, in malesuada libero turpis vel augue. Fusce interdum ultricies lectus, a luctus arcu eleifend nec. Aenean nulla orci, pretium vitae venenatis ac, rhoncus at odio.

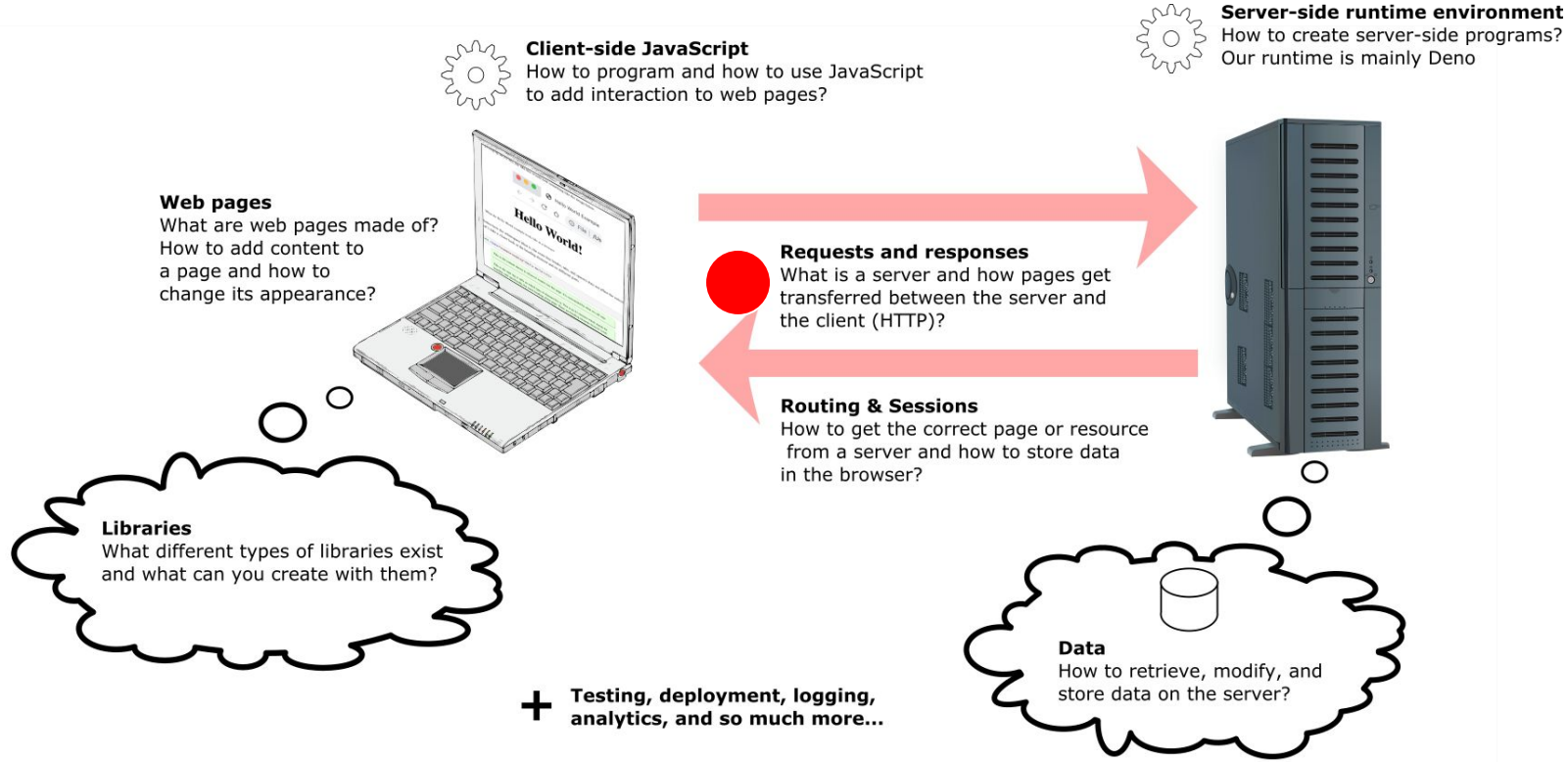
To be continued...

Vinkkejä:

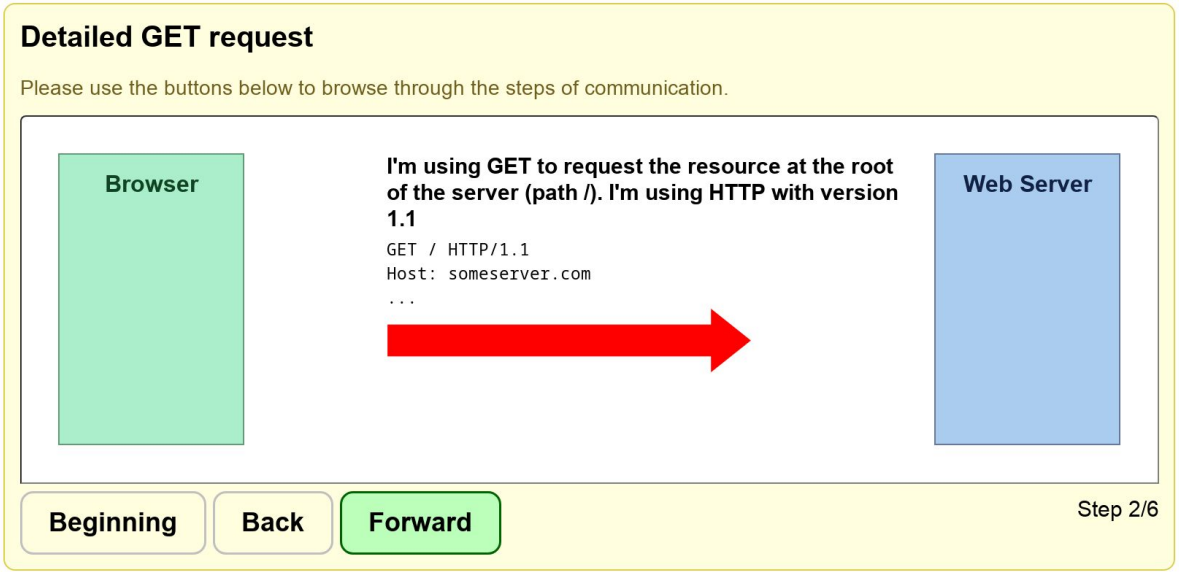
- Class selectors - CSS: Cascading Style Sheets | MDN
- Descendant combinator - CSS: Cascading Style Sheets | MDN
- Adjacent sibling combinator - CSS: Cascading Style Sheets | MDN

HTTP

● Olet täällä



HTTP - Pyyntö (request) ja vastaus (response)



https://acos.cs.aalto.fi/html/webdev/webdev-csmv/detailed_get

An overview of HTTP - HTTP | MDN

URL - Uniform Resource Locator

- Esim. https://wsd.cs.aalto.fi:443/1-introduction-and-tooling/4-http-protocol/#http-status-codes?foo=bar
  - Protokolla: https
  - Alidomainit: wsd ja cs
  - Domain: aalto
  - TLD* fi
  - Portti: 443
  - Polku: /1-introduction-and-tooling/4-http-protocol/
  - Ankkuri: #http-status-codes
  - Query: ?foo=bar
- What is a URL? - Learn web development | MDN

* TLD, Top-Level Domain

HTTP - Request Methods

- GET (ei muuta palvelimen tilaa) ja POST (lähettää dataa palvelimelle, tila yleensä muuttuu)

- GET - HTTP | MDN ja POST - HTTP | MDN

- HTTP request methods - HTTP | MDN

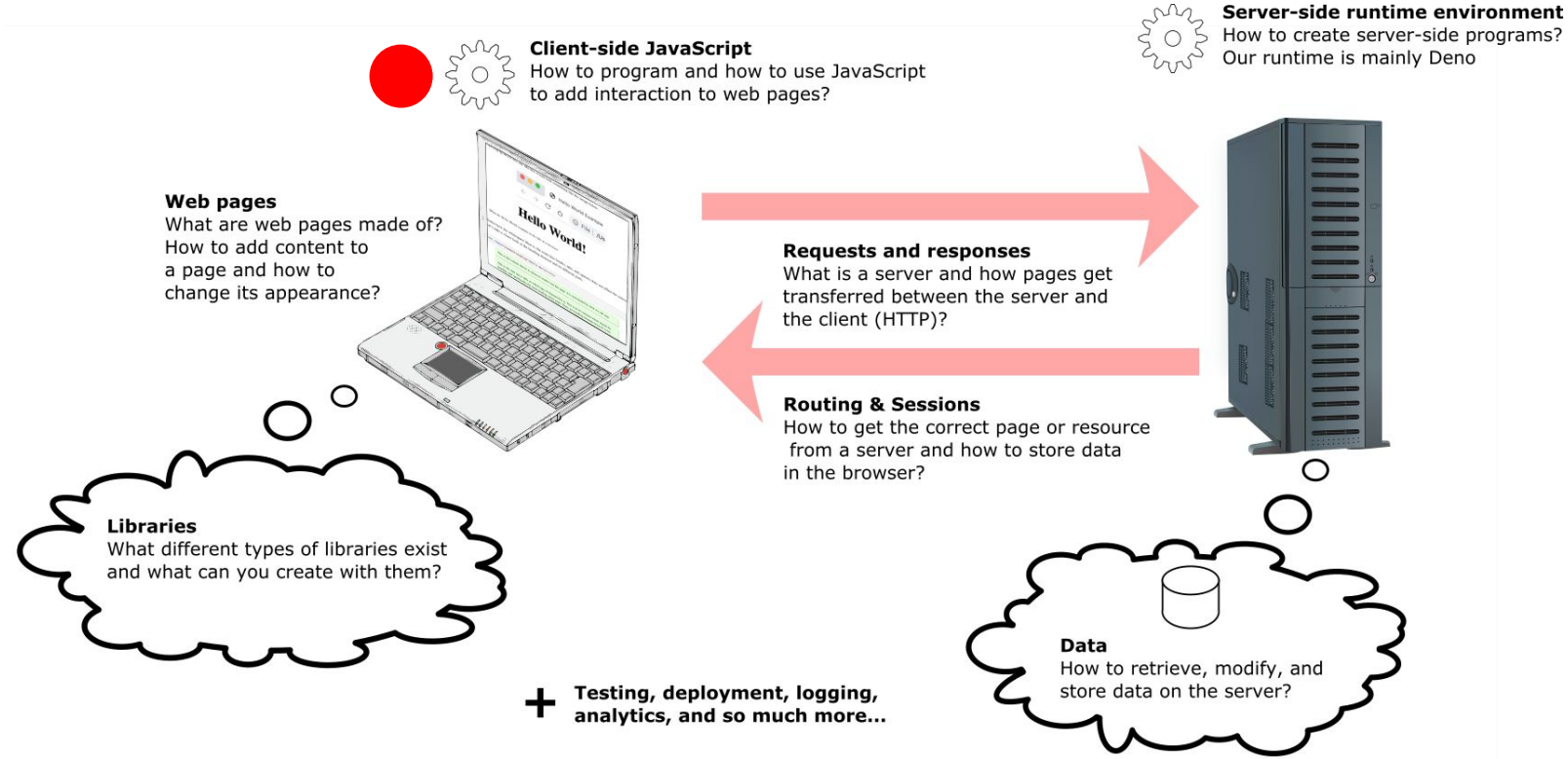
HTTP - Response

- Palvelin antaa pyyntöön vastauksen
- Vastauksella aina koodi:
  - Informaatio-koodit 100-199, esim. 101 Switching protocol
  - Onnistumiset 200-299, esim. 200 OK
  - Uudelleenohjaukset 300-399, esim. 301 Moved permanently
  - Asiakkaan virheet 400-499, esim. 404 Not found
  - Palvelimen virheet 500-599, esim. 500 Internal server error

- HTTP response status codes - HTTP | MDN

JavaScript

Olet täällä



JS Scope

- Muuttujien määrittely
  - Block-scope let - JavaScript | MDN
  - Function/Global scope var - JavaScript | MDN

- Vakioiden määrittely
  - Block-scope const - JavaScript | MDN

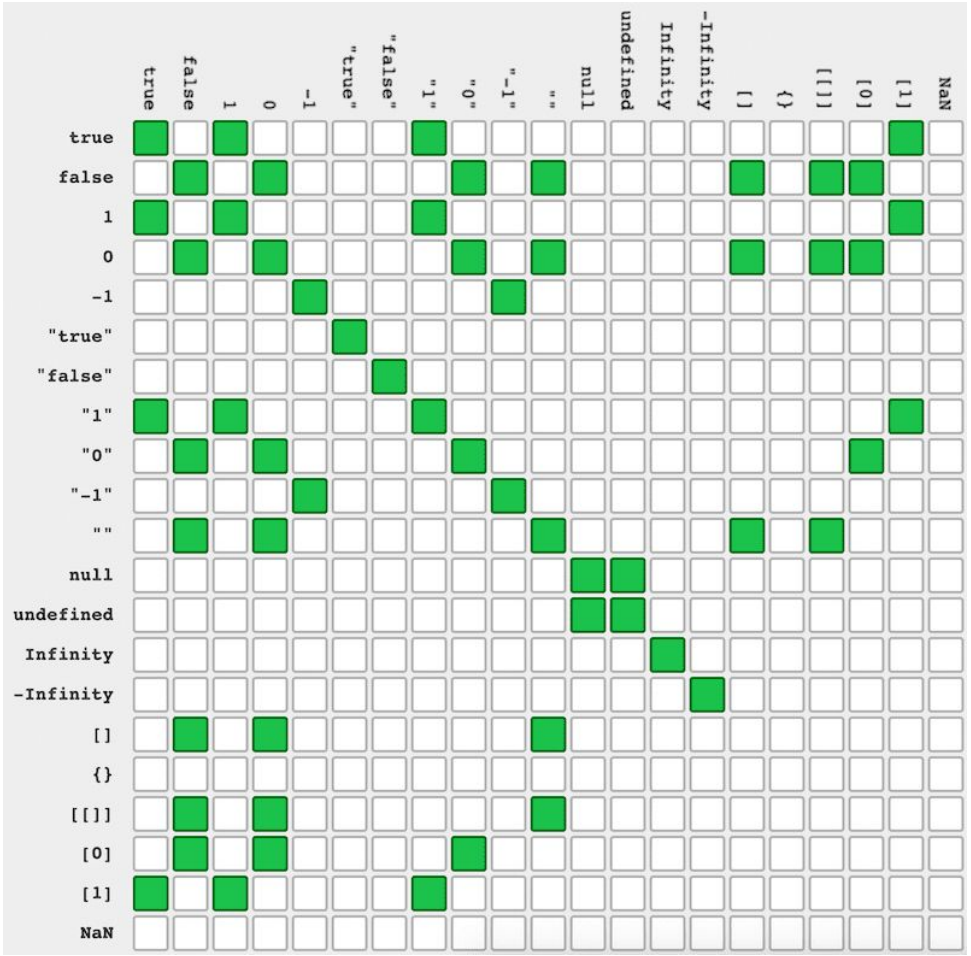
JS - Muuttujat (variables)

```
let numberOfPeaches = 4;
let numberOfApricots = 3;
numberOfPeaches = numberOfPeaches + 1;
let fruitsEaten = numberOfPeaches + numberOfApricots;

const PI = 3.14;
PI = 3;
Uncaught TypeError: Assignment to constant variable.
```

JS - Truthy & Falsy

- == vs. ===
- 1 == true; //true
- 1 === true; //false



https://dorey.github.io/JavaScript-Equality-Table/ (CC 4.0 BY-SA)

JS - Ehtolauseet (Conditionals)

```
let feelsLike;
let temperature = 18;

if (temperature > 20) {
  feelsLike = 'warm';
} else {
  feelsLike = 'cold';
}
```

[Making decisions in your code — conditionals - Learn web development | MDN]

```
let temperature = 18;
let feelsLike = temperature > 20 ? 'warm' : 'cold';
```

JS - Toistorakenteet (loops)

Loops and iteration - JavaScript | MDN

- For

```
console.log("This is easy as...");
for(let i = 3; i > 0; i--) {
  console.log(i);
}
console.log("Done!");
```

- While
  - https://acos.cs.aalto.fi/html/jsvee/jsvee-python-json/transpiler?url=https:%2F%2Fusers.aalto.fi%2F~haaranl1%2Fwhile_loop.json

JS - Funktiot

- "Normaalit" funktiot - Functions - JavaScript | MDN

```
function addThree(someNumber) {
  let newNumber = someNumber + 3;
  return newNumber;
}
addThree(5); //this will return 8
```

- Nuolinotaatio - Arrow function expressions - JavaScript | MDN

```
let addThree = (x) => { return x+3; }

//tai lyhyemmin
let addThree = x => x+3;
```

JS - Listat (arrays)

Array - JavaScript | MDN

```
let numbers = [1,2,3]
numbers.push(4) // [1,2,3,4]
let another = numbers.splice(1);
console.log(another)  // [2, 3, 4]
another[0] // 2
another[1] // 3
another[2] // 4
another.length // 3
let squares = another.map( x => x*x) //ks.Array.prototype.map() - JavaScript | MDN
console.log(squares) // [4, 9, 16]
let both = another.map( x => [x, x*x])
```

JS - Lisää listoista

```
let numbers = [1,2,3]
let both = numbers.map( x => [x, x*x] ) //[[1,1], [2,4], [3,9]]
both.length // 3
both[1] // [2,4]
both[1][0] // 2
let justSquares = both.map( x => x[1] )
justSquares // [1, 4, 9]

// arrayn viimeinen elementti
justSquares[ justSquares.length - 1] //9
```

JS - Objektit (objects) - Working with objects - JavaScript | MDN

```
let person = {
    name: "Brendan Eich",
    yearOfBirth: 1961,
    language: "javascript" // funktion olisi voinut määritellä täällä
};

person.name         // "Brendan Eich"
person["name"]      // "Brendan Eich"

person.approximateAge = function() {
    return new Date().getFullYear() - this.yearOfBirth;
}
person.approximateAge() // 60
```

JS - Template literals

[Template literals (Template strings) - JavaScript | MDN]

```
console.log(`${person.name} is ${person.approximateAge()} years old.`);

--> Brendan Eich is 60 years old.
```

JavaScript Object Notation (JSON)

- Tapa säilöä tietoa, jota on helppo lukea JavaScriptillä (ja monilla muilla kielillä)
- https://www.json.org/json-en.html

```
{
  "name" : "Kauppalista",
  "owner" : "Lassi",
  "items" : [
    [false, "Kahvia"],
    [true, "Tomaatteja"],
    [false, "Paprika"]
  ]
}
```

Accessing elements

Locating DOM elements using selectors - Web APIs | MDN

Element.getElementsByTagName() - Web APIs | MDN

Document.getElementById() - Web APIs | MDN

Document.querySelector() - Web APIs | MDN

Document.querySelectorAll() - Web APIs | MDN

Document.getElementsByClassName() - Web APIs | MDN

JS - Elementtien manipulointi

```
let listItems = document.querySelectorAll('#special-list li');
for (let i = 0; i < listItems.length; i++) {
  console.log(listItems[i].innerText);
}

let specialList = document.getElementById('special-list');
let newItem = document.createElement('li');
newItem.innerText = "New item";
specialList.appendChild(newItem);

for (let i = 0; i < listItems.length; i++) {
  console.log(listItems[i].innerText);
}
```

JS - Removing elements

```
let simpleKeyboard = document.getElementById('simple-keyboard');
let keyButtons = simpleKeyboard.children;

console.log(keyButtons); //HTML Collection of n buttons
let enharmonicKey = keyButtons[2];
enharmonicKey.remove();
console.log(keyButtons); //HTML Collection of n-1 buttons
```

jQuery

jQuery

"jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript."

[jQuery API Documentation]

```
$('#someItem').text(); // Näyttää elementin tekstisisällön

// lisää uuden span-elementin #someItem elementin loppuun,
// on myös vastaava prepend, joka lisää alkuun
$('#someItem').append('<span>some text</span>');

// poistaa ensimmäisen lapsielementin #someItem-elementistä
$('#someItem').children().first().remove()
```

Tehtäviä

- Project-02-html-css-js
  - Typing.html - Ks. kommentti alussa
  - js/lists.js - Toteuta useamman listan tuki