EFIF Websites

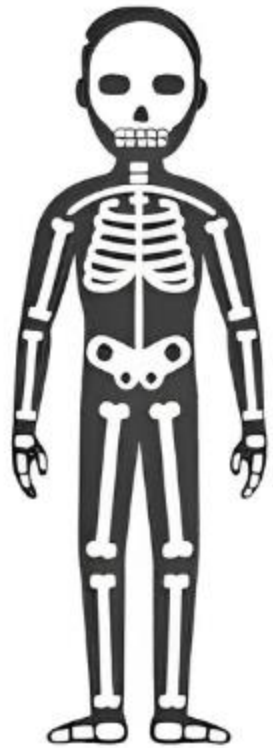
Client
(You)

REST API z.B. mit

- Flask(Python)
- node.js (JavaScript)

HTML

HTML the Skeleton

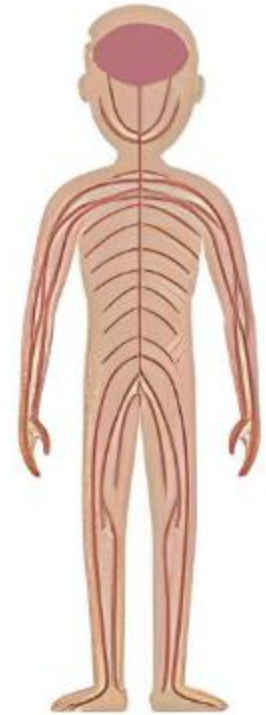


CSS

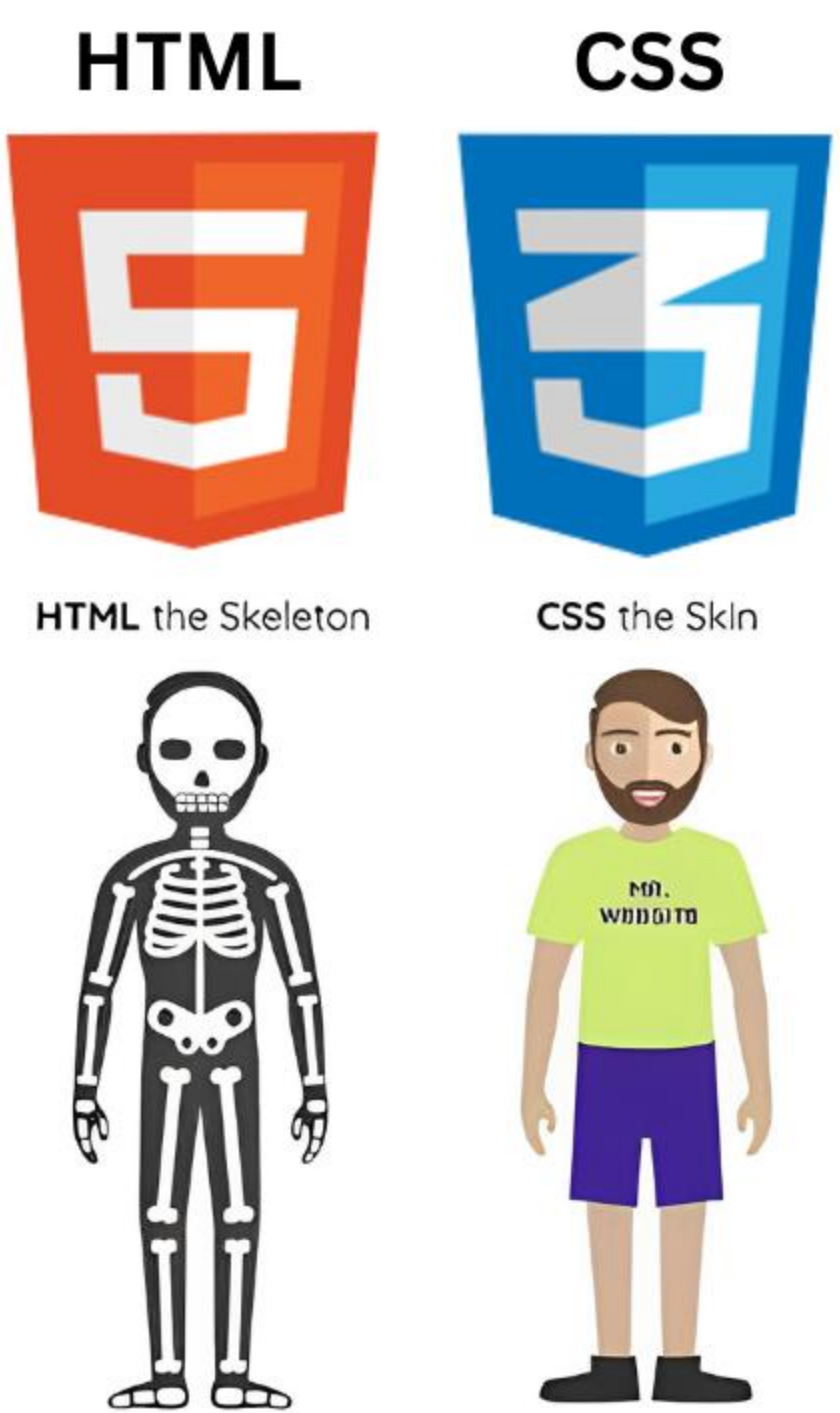
CSS the Skin

JS

Javascript the Brain



Part I: HTML & CSS



HTML-Grundstruktur

- Startseite jeder Website heisst **index.html**
- HTML gibt eine **Grundstruktur** für jede Datei vor:
  - **head**: Metadaten wie Titel etc.
  - **body**: Eigentlicher Inhalt der Website

```
<html>
<head>
    <meta charset="UTF-8">
    <title>Titel der Website</title>
</head>
<body>
    Hier kommt der Inhalt der Website hinein.
</body>
</html>
```

Block vs. Inline-Elemente

Block-Elemente

- *rechteckige Blöcke* im Inhalt über ganze Breite
- werden untereinander angeordnet

```
<h5>Überschriften</h5>
<p>Abschnitte (en. paragraph) strukturieren den Text</p>
<ul>
  <li>Listen (ul:
  <li>li: list i
</ul>
```

Überschriften

Abschnitte (en. paragraph) strukturieren den Text

- Listen (ul: unnumbered list)
- li: list item

Inline-Elemente

- sind Teil des Lauftexts

```
Text kann <em>betont (en. emphasis)</em> oder
<strong>stark betont</strong> werden.
```

Text kann *betont (en. emphasis)* oder **stark betont** werden.

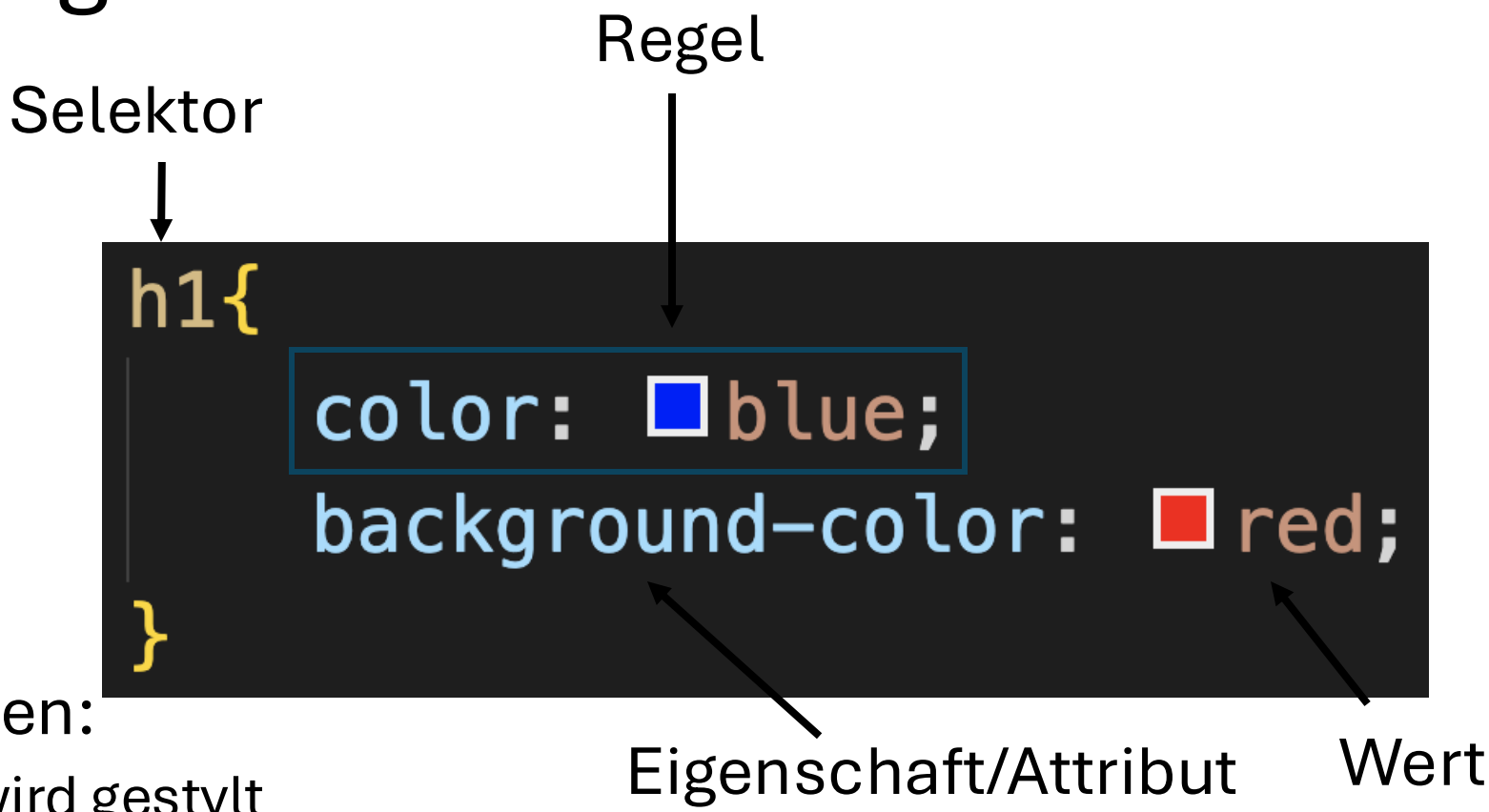
Wichtigste HTML Elemente

- **Überschriften** (h1,h2,...):
  ```
  <h1>Ich bin eine Überschrift</h1>
  ```
- **Paragraph** mit Text:
  ```
  <p>Ich bin eine Paragraph</p>
  ```
- **Hyperlink** (Tag: `<a>` für anchor) benötigen ein `href` Attribut (für hyper reference):
  ```
  <a href="https://www.ksr.ch">Link zur KSR</a>
  ```
- **Bild** (kein End-Tag!):
  ```
  <img src="name_bild.jpg"/>
  ```

CSS Einbinden

```html
<html>
<head>
    <meta charset="UTF-8">
    <title>Titel der Website</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    ...
</body>
</html>
```

CSS-Regeln



- Selektoren:
  - *Was* wird gestylt
  - Beispiele: h1, h2, strong, p, …
- Eigenschaften / Attribute: color, background-color, …

CSS-Attribute

Attribut	Bedeutung	Werte
`color`	Textfarbe	`red, rgb(30%, 50%, 100%)`
`font-family`	Schriftart	`sans-serif, monospace`
`font-size`	Textgrösse	`1cm, 10px`
`font-weight`	Textstil	`bold, normal`
`text-align`	Textausrichtung	`center`
`background-color`	Hintergrundfarbe	`red, rgb(30%, 50%, 100%)`
`background-image`	Hintergrundbild	`url("tigerente.jpg")`

Abstände

```
<p>Ich bin ein Abschnitt.</p>
```

```
p{

}
```

Ich bin ein Abschnitt.

```
p{
    background-color: red;
}
```

Ich bin ein Abschnitt.

```
p{
    background-color: red;
    padding: 15px;
}
```

Ich bin ein Abschnitt.

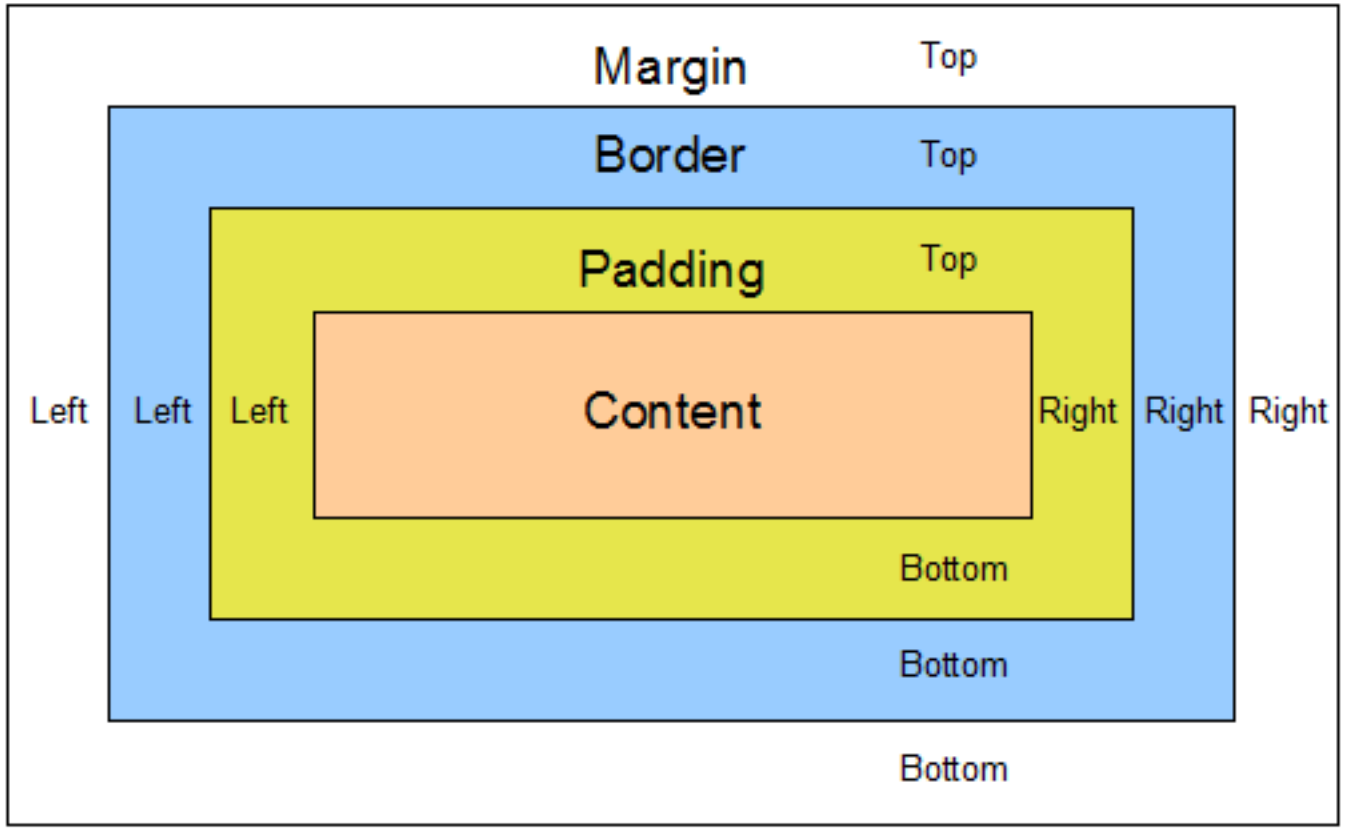
```
p{
    background-color: red;
    padding: 15px;
    border: solid blue 10px;
}
```

Ich bin ein Abschnitt.

```
p{
    background-color: red;
    padding: 15px;
    border: solid blue 10px;
    margin: 20px;
}
```

Ich bin ein Abschnitt.

Abstände & Ränder



- Kann vier Seiten (top, bottom, left, right) auch einzeln regeln, z.B.
- `margin-top: 20px`

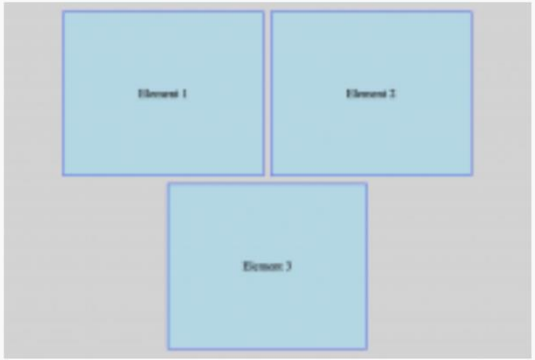
id & class

- Mehrere Elemente im CSS verändern:
```
<h2 class="blueH">Informatik</h2>
<h2 class="blueH">Mathe</h2>
```
- Einzelnes Element im CSS verändern:
```
<h1 id="greenH">Fächer</h1>
```
- CSS:
```
.blueH {}
#greenH{}
```

Ausrichten von Elementen

- Elemente nebeneinander ausrichten?
- Elemente dynamisch anordnen, z.B. Desktop vs. Mobile?
- Möglichkeit 1: Flexbox
  - https://sca.ksr.ch/doku.php?id=gf_informatik:web_sca:websites#flexbox
- Möglichkeit 2: Grid
  - https://sca.ksr.ch/doku.php?id=gf_informatik:web_sca:websites#grid_layout
- Kann auch Grid mit Flexbox kombinieren

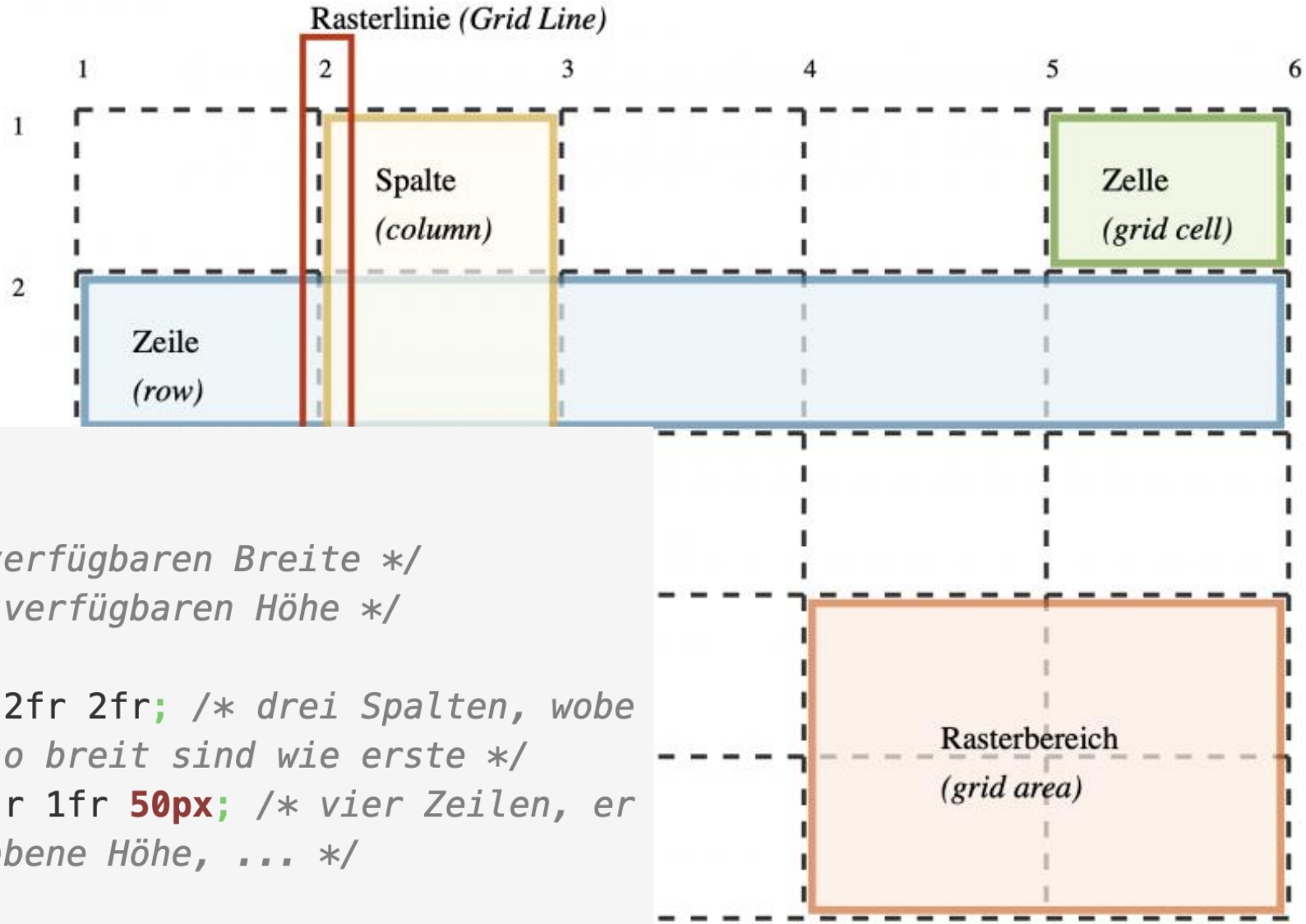
Flexbox

Breiter Bildschirm (z.B. Laptop)	Mittlerer Bildschirm (z.B. Tablet)	Schmaler Bildschirm (z.B. Smartphone)
		

```html
<div class="container">
    <div class="item">Element 1</div>
    <div class="item">Element 2</div>
    <div class="item">Element 3</div>
</div>
```

```css
.container{ /* da container class ist, muss Punkt vor Selektor schreiben */
    display: flex; /* flexbox aktivieren */
}
```

Grid




```css
.container {
    display: grid;
    width: 100vw; /* 100% der verfügbaren Breite */
    height: 100vh; /* 100% der verfügbaren Höhe */

    grid-template-columns: 1fr 2fr 2fr; /* drei Spalten, wobei die 2. und 3. beide doppelt so breit sind wie erste */
    grid-template-rows: 60px 1fr 1fr 50px; /* vier Zeilen, erste und letzte haben fix angegebene Höhe, ... */

    gap: 10px; /* gap um Elemente des Grids */
}
```

Link

- sca.ksr.ch / GF Informatik / GF Informatik 1 & 2 (A. Schärer) / Websites
- https://sca.ksr.ch/doku.php?id=gf_informatik:web_sca:websites

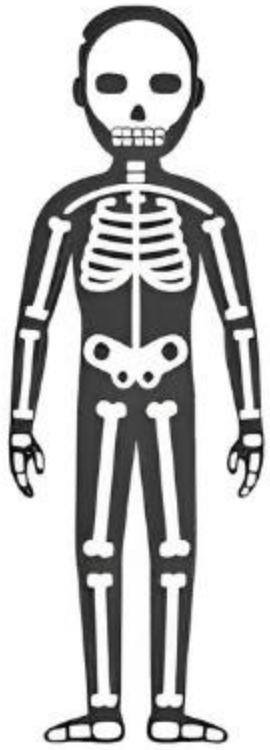
Auftrag

- Entscheide dich für Flexbox oder Grid (je nachdem was für dein Projekt besser scheint)
- Baue einfache Website mit versch. Elementen: Text (Lorem Ipsum), Bilder (Internet), Video einbinden (optional), …
- Richte Elemente nebeneinander und untereinander an (Flexbox oder Grid)
- Style Website mit CSS
- Website soll dynamisch anpassen, je nachdem, wie gross Browserfenster
- Website auf deinem Raspi hosten

Part II: JavaScript

HTML

HTML the Skeleton

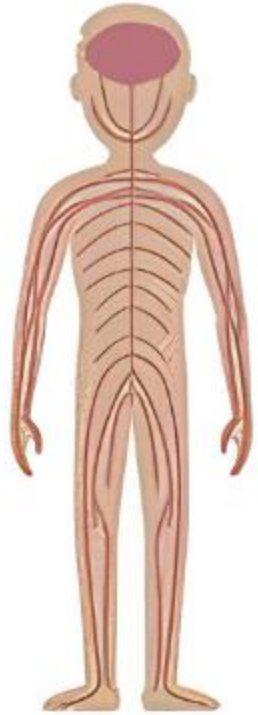


CSS

CSS the Skin

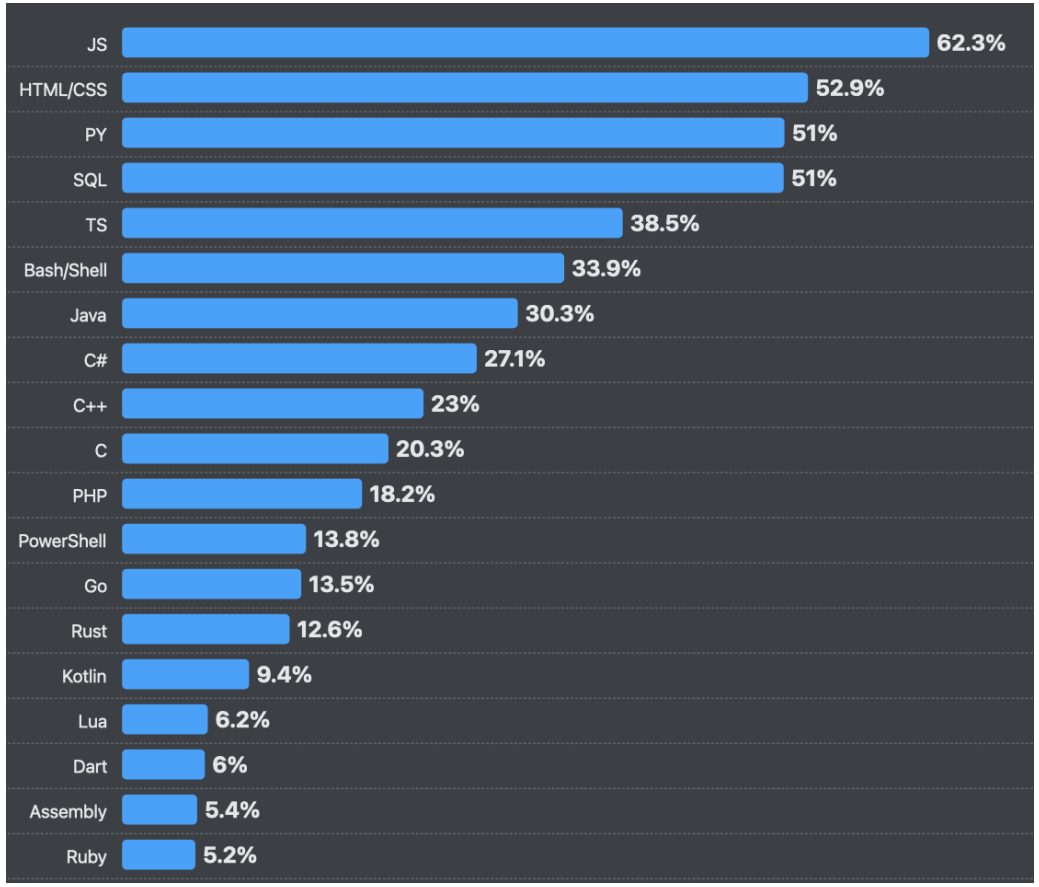
JS

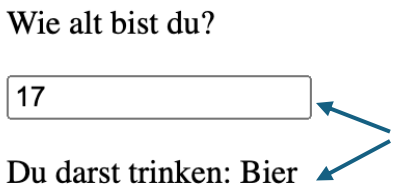
Javascript the Brain



JavaScript?

- JavaScript (JS) ist *die* Programmiersprache im Web
- Am meisten verwendete Programmiersprache seit vielen Jahren
- Primär läuft JS direkt im Browser
- Beispiel: **Schnaps-Bier-Sirup**





Benötigt **Logik**: je nach eingegebenem Alter wird ein anderes Getränk angezeigt. Nicht möglich in HTML (da *keine* Programmiersprache), benötigt JS!

Hello World!

JS einbinden

index.html

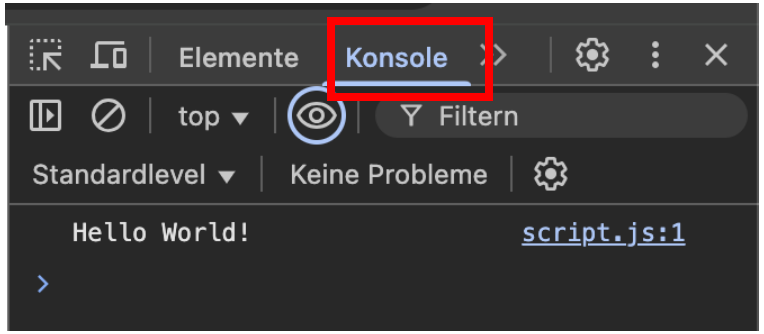
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>I'm a Website with JS!</title>
    <script src="script.js"></script>
</head>
<body>

</body>
</html>
```

script.js

```
console.log('Hello World!');
```

entspricht in etwa `print()` in Python

rechte Maustaste



ganz ähnlich in anderen Browsern

Basics

```
// IN KONSOLE AUSGEBEN (quasi das print() von JavaScript)
console.log(42);

// VARIABLEN & KONSTANTEN
let x = 42; // Variablen benötigen let
const y = 43; // Konstanten sind wie Variablen, ausser dass sie sich nicht verändern lassen.
// im Zweifelsfall: einfach let verwenden
```

Semikolon an Ende (fast) jeder Zeile.

```
// WHILE-SCHLEIFE
let i = 0;
while (i < 10) {
    console.log(i);
}

// FOR-SCHLEIFE
for (let i = 0; i < 10; i++) {
    console.log(i);
}
```

```
// VERZWEIGUNG
x = 42;
if (x > 0) {
    console.log("Zahl positiv");
} else if (x === 0) {
    console.log("Zahl Null");
} else {
    console.log("Zahl negativ");
}
```

Geschwungene Klammern, Einrückungen optional aber empfohlen

=== vs. ==

Bsp.:

5 == '5'; // true (macht zuerst Typ-Angleichung)

5 === '5'; // false (entspricht == in Python)

```
// FUNKTION
function mySquare(x) { // Deklaration der Funktion
    return x*x;
}

let sq = mySquare(5); // Funktionsaufruf
console.log(sq);
```

Ereignisse (Events)

- Ereignisse sind Aktionen, die im Browser passieren, z.B.:
  - Knopf wird gedrückt.
  - Eingabe in Eingabefeld wird verändert.
  - Mausereignisse
- JavaScript kann auf solche Ereignisse reagieren.
- HTML:

```
<button onclick="handleClick()">Click Me!</button>
```

Wenn Button geklickt wird, wird JS-Funktion ausgeführt.

- JavaScript:

```
function handleClick(event){
    console.log("Button Clicked!");
}
```

Auf HTML-Elemente zugreifen

- Oft möchte man in JS-Code auf HTML-Elemente zugreifen.
- Am einfachsten über **ID** eines HTML-Elements.
- Bsp.: Auf Zahl zugreifen, die in *Input*-Feld eingegeben wurde:


```
<input id="inp" type="number" placeholder="42" onchange="handleInpChanged(event)"></input>
```

```
function handleInpChanged(event){
    nr = document.getElementById("inp");
    console.log(nr.value);
}
```

über ID auf HTML-Element zugreifen

Achtung: `nr` ist nicht nur eingegebene Zahl, sondern beinhaltet alle Infos von Input-Feld. Mit `nr.value` kann nur auf Wert zugreifen

Ereignisse (Events)

Auftrag I: Schnaps-Bier-Sirup

- Programmiere:

Auftrag II: Dies und das

- Erstelle Website «Dies und das», die mehrere kleine (einigermassen) nützliche Tools beinhalten soll.
- Wähle zuerst 2-3 Tools aus dem Beispiel rechts …
- … und überlege dir danach eigene Ideen
- (oder weitere Ideen vom Beispiel)
- Auf Wiki findest du Tipps zu einzelnen Tools: https://sca.ksr.ch/doku.php?id=gf_informatik:web_sca:websites#auftragdies_und_das