1 JS and DOM Basics

1.1 What is the purpose of Javascript and how does it complement HTML and CSS?[1]
Javascript gilt als Programmiersprache, hnlich wie Java, Rust, php, ... Javascript wird mithilfe des <script>-Tags in ein HTML-Dokument eingebunden. Ein externales sowie internales Einbinden von Javascript ist zu jeder Zeit mglich, egal ob es sich dabei um Sourcecode innerhalb des Head oder des Body befindet. (Siehe demoHTML.html)

```
<!DOCTYPE html>
  <html>
    <head>
    <script src="demoJS.js"></script>
    </head>

    <body>
      <h1>A Web Page</h1>
      <p id="demo">A Paragraph</p>
      <button type="button" onclick="myFunction()">Try it</button>

    </body>
  </html>
```

Mithilfe von JS lassen sich Funktionen effizient schreiben. Es gilt dabei zu beachten, dass es zu den guten Praktiken gehrt, JS-Code von HTML und CSS zu trennen. Dementsprechend ist ein externales Einbinden anstrebbar.

- ◇ Vorteile und Nachteile von Internen Code
  - \+ Schnelles Testing & Debuggen
  - \- Lange Quellcodes fhren zu unbersichtlichen Code
  - \- Langer Wartungsaufwand bei dupliziertem Quellcode
- ◇ Vorteile und Nachteile von Externem Code
  - \+ bersichtlicher Code
  - \+ Kurzer Wartungsaufwand
  - \+ Zwischengespeicherter JS-Code beschleunigt Ladezeiten

1.2 What kind of typing is provided by Javascript? What are the risks? [2]
In Javascript unterscheidet man zwischen folgenden

Typen:

- ◦ String - Die allbekante Zeichenkette in "Text"
- ◦ Number - Eine beliebige Zahl mit/ohne Nachkommazahlen
- ◦ Boolean - Ein Wahrheitswert, True oder False

[1] Src.: *https://www.w3schools.com/js/js_whereto.asp*
[2] Src.: *https://www.w3schools.com/js/js_type_conversion.asp*

- Object - Ein Objekt, hnlich einer Klasse in Java
- Function - Eine Funktion, etwa fn()
- null - Ein Objekt ohne Wertzuweisung
- undefined - Ein nicht definiertes Objekt

Objekten:

- Object - Ein Objekt, hnlich einer Klasse in Java
- Date - Date() als eigene vordefinierte Klasse fr die Zeit
- Array - Ein Array aus Werten.

1.3 What is DOM?

Dom, Document Object Model, beschreibt ein neutrales Interface zwischen .js und .html. Es soll dynamischen Zugang & Update von Inhalt ermglichen. Levels of DOM und DOM Level sind 2 unterschiedliche Spezifikationen:

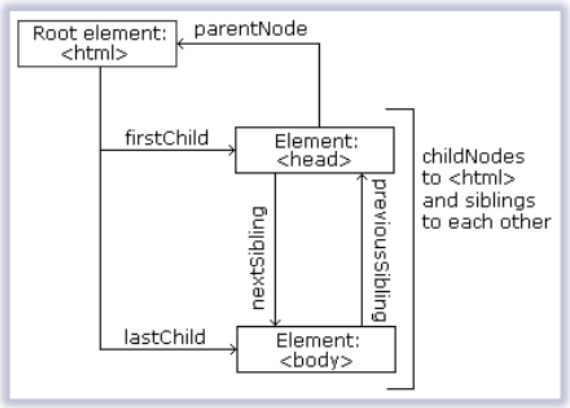
◇ Levels of DOM beschreibt die Hierarchie innerhalb eines Krpers. Beispielsweise gliedert sich

```
<p>Hello, <b>World</b>!</p>
```

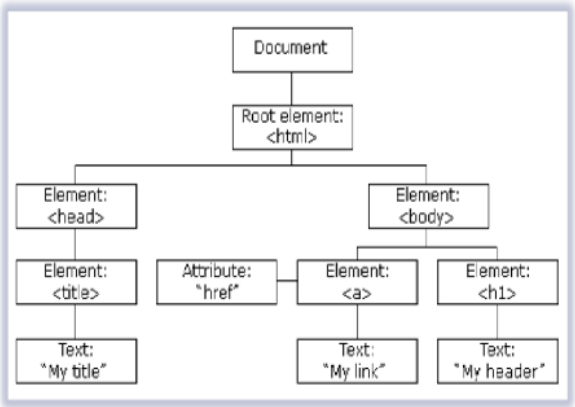
in <p> := Level 1 und <b> := Level 2

◇ DOM Level x beschreibt die derzeitig genutzte Norm fr HTML-Standards.
- DOM Level 1: HTML & XML Dokument-Model, inklusive Mglichkeit der Vernderung der Dokumente.
- DOM Level 2: Eventmodel & Funktionserweiterung "getElementById". Supporterweiterungen von XML und CSS.
- DOM Level 3: Support fr XPath, Keyboard-Event-Handling und Serialisierung von Dokumenten als XML.
- DOM Level 4: Work in Progress, verffentlicht 2015 als WHATWG living standard.

Prinzipiell soll man mithilfe von Funktionsaufrufen via .js oder .php auf DOM-Objekte zugreifen. DOM-Objekte sind grundstzlich spezifiziert im .html, wobei der Begriff "Objekt" auf die jeweiligen Tags abzielt.



.js Zugriffsdisplay innerhalb des DOM



Aufbau des DOM durch .html

Beispiel: Simple DOM-Tree Manipulation

```
<!DOCTYPE html>
  <html lang="en">
    <head>
      <meta charset="UTF-8">
      <title>DOM-Manipulation Demo</title>
    </head>
    <body>
      <p style="color: green;"
        onmouseover="hoverStart(this);" onmouseout="hoverEnd(this);">
        Hover me!
      </p>
      <p style="color: green;"
        onclick="this.innerHTML='You clicked me!';">
        Click me!
      </p>

<script src="demo_DOMscripts.js"></script>

</body>
</html>

// Javascript below
function hoverStart(element) {
element.innerHTML='You hover me!';
}
function hoverEnd(element) {
element.innerHTML='Thanks for not hovering me!';
}
```