JS – Sololean

JS in HTML einbinden:

- es ist praktisch die script Datei erst unten in der HTML-Datei einzubinden, das sorgt dafür, dass die Seite schon geladen ist, während JS noch lädt
- JS ist keine kompilierte Sprache, sie wird während der Luafzeit ausgeführt

```
<script src = "demo.js"><script>
```

Comments in JS:

```
// for a single line
/* for more
lines*/
```

Variables:

var x = 10;

let z=4; // nur im lokalen Block gültig

const z // Konstante, auf lokalen Block begrenzt

Naming Rules:

First character must be a letter, _ or $

No spaces, no mathematical or logical operator

Data Types:

- var Floating point number (var is a double), boolean, undefined or null
- Strings (use the \ before special characters – "Hello \"basterd\""
  -> outputs: Hello "basterd"

Increment & Decrement:

var++	Post Increment	a = 0, b = 10 -> a= b++	a = 10 , b = 11
++var	Pre Increment	a = 0, b = 10 -> a= ++b	a = 11 , b = 11
var--	Post Decrement	a = 0, b = 10 -> a= b--	a = 10, b = 9
--var	Pre Decrement	a = 0, b = 10 -> a= --b	A = 9, b = 9

Comparision Operators

===	Identical (equal &of same type)	
!==	Not identical	10 !== 10 flase

Functions

Funktionen aufrufen:

```
f2();
//ruft die Funktion auf, führt sie gleich aus:

(function f2(){})()
//IIFE – imediately invoked function expression
// wird gleich beim deklarieren ausgeführt

f2;
/*Referenz auf eine Funktion, Bsp: in einer
Klick-variable für später speichern/aktivieren
.führt die Funktion nicht gleich aus */
```

```
function name(){
     // code to be executed
}
```

```
function name(0, true,”b”){
    /*Argumente werden in einem array gespeichert,
    man kann hier true mit arguments[1] aufrufen.
    Es können mehr oder wenige Argumente eingegeben
    werden als die Methode vorsieht */
}
```

Infos and Feedback for User -Commands

Alert-Box:

```
alert("Hello\nHow are you?");//alert box pops up, use \n for line breaks
```

Prompt-Box: (often used to input value before entering, has ok and cancel button)

```
var user = prompt("Please enter your name","Max Mustermann");
//first parameter is the label, the second (optional) is the sample text in the textfield
//if clicked "OK" – it returns the user input, if clicked "Cancel" it returns null
```

Confirm-Box: (often used for have the user accept or verify smth. - user clicks either Ok or Cancel)

```
var result = confirm("Do you really want to leave this page?");

// if the user clicks OK it returns true, if Cancel is clicked, false is returned
```

Object

- are containers for named values, are variables too
- list of values written in **name:value** pairs
- theese values are called **properties**
- has a build in length property to count the characters of a value:

```
-> person.name.length //outputs 4
```

access object properties:

```
objectName.propertyName
//or
objectName["propertyName"]
```

Object methodes

(property containing a function definition):

```
objectName.methodName()
```

Create object:

```
Var person = {
     name: “John”, age: 31,
     favColor: “green”
};
```

Object constructor function:

```
Function person(name, age, color){
     this.name = name;
     this.age = age;
     This.favColor = color;
}
```

After the having an object constructor function, we can use the **new** keyword to create new objects:

```
 Var p1 = new person(“John”,42,”green);
```

Methods

- are functions that are stored as object properties
- with a method you can acess the object`s properties by using the **this** keyord

```
MethodName : function(){codelines}
```

Anonyme Funktion beim onClick-Event

h2.onclick=function(){testAgain(5)};

defining a method inside a constructor function:

```
function person(name,age){
     this.name = name;
     this.age = 43;
     this.changeName = function(name){
                this.name=name;
     } }
```

Array

- kann versch. Datentypen beinhalten
- Länge nicht festgelegt (anders als in Java)
- kann leere Stellen beinhalten
- Länge kann während der Laufzeit verkürzt oder verlängert werden
- dense(alle Plätze voll) und sparce(undefined zwischendurch) Arrays

```
var list = [1,2, ,4];
```

undefined

Durch den Array iterieren:

1.Möglichkeit – gibt alle plätze aus -normale For-Schleife

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

2. Möglichkeit – gibt keine undefined Werte zurück

```
for(elem in MyObject) {
        console.log(list[i]);
Array
```

Array Methoden:

.length
.concat(secondArray)
Concat joins 2 arrays to a new one, does not affect the old ones
.forEach(function)
The forEach() method calls a provided function once for each element in an array, in order, does not execute the function for array elements without values.

Associative arrays:

- statt Index benutzen diese Arrays einen String als Key, in JS ist das nicht möglich bei einem Array
- aber man kann den "named array syntax" benutzen, welcher ein Objekt erstellt:
- die benannten Index "name" und "age" werden zu properties
- besser ist, man benutzt einfach Objekte

```
Var person = []; //empthy array
person[“name”] = “John”;
person[“age”] = 46;
```

Timer

SetTimeout setIntervall:

- Methoden des Window-Objektes
- **Verzögerung**: `setTimeout(*)`
- **Wiederholte Ausführung:** `setInterval(*)`

* Zeit in ms

```
setTimeout (function(){
    //do smth
},2000);

setTimeout (spaeter, 2000);
setIntervall (immerWeiter, 3000);
```

Unterbechung des Timers:

- clearTimeaout(*)
- clearintervall(*)

* Zahl, die die jeweilige setIntervall/setTimeout Methode zurückgibt – ganzzahlige 1,2,3..

The Date object:

- consists of year, month, day, hour, minute, second and milisecond
- date onjects are static, they dont change
-

Methods:

```
.getFullYear()
.getMonth()
usw...
```

```
var d = new Date(); //creates a new date object with current date and time

new Date(miliseconds);

new Date(DateString);
New Date("January 2, 2015 10:42:00"); // outputs Fri Jan 02 2015 10:42:00

new Date(year, month, day, hours, minutes, seconds, milliseconds);
cew Date(88,5,11,11,42,0,0); // Sat Jun 11 1988 11:42:00
```

DOM

- all HTML elements are objects
- document objects has Methods that allow us to select the desired HTML element

```
document.getElementById(id)
document.getElementByClassName(name)
document.getElementByTagName(name)
document.getElementByName(id)
```

Relationship Methods:

```
.childNodes
.firstChild
.lastChild
.hasChildNoses
.nextSibling
.previousSbibling
.parentNode
```

Changing Elements

Change attributes of selected elements:

Change style:
(dont use dashes: backgroundColor instead of background-color)

```
x.style.color = "7700FF";
x.style.width = "100px";
```

Create Elements:

`element.cloneNode()` //clones an element and returns the resulting code
`document.createElement(element)` //creates a new element node
`document.createTextNode(text)` // creates new text node

created nodes will not appear in the document until you append it to an existing element with one of this methods:

`element.appendChild()` //adds a new childnode as the last child node
`element.insertBefore(node1, node 2)` //inserts node1 as child before node2

```
var node = document.createTextNode("Some new text")
 var div = getElementsByID("demo");
 div.appendChild(node);
```