

# Web Apps

Javascript –  
Basiselemente von JS

# Variablen und Konstante

```
a = 1;  
b = "2";  
e=true;  
let c = null,  
    d;
```

Mit und ohne Deklaration möglich; keine Angabe von einem Datentyp

```
const COLOR_RED = "#F00";  
alert(a + " " + b + " " + c + " " + d);
```

127.0.0.1:5500 enthält

1 2 null undefined true

OK

null ist nicht undefined!

Weitere Datentypen: object, symbol

# Typenumwandlungen

---

```
a = 1;  
aString = String(a);  
b = "2"  
bNumber = Number(b);  
alert(typeof a + " " + aString + " " + typeof b + " " + bNumber);
```

Ohne Klammer!

**127.0.0.1:5500 enthält**

number 1 string 2

OK

# Vergleiche

---

```
a = 1;  
b = "1";  
alert(Boolean(a == b) + " X " + Boolean(a === b));
```

Vergleicht nur Werte!

Vergleicht auch Datentypen!



# Bedingungen

---

```
year = "2022";  
if (year == 2022) alert("1. Year is 2022!");
```

Wahr mit == da 2022 == „2022“

```
cond = year === 2022;  
if (cond) alert("2. Year is 2022!");
```

Falsch da 2022 === „2022“ nicht gilt!

```
value = year === 2022 ? "yes" : "no"  
alert(value);
```

Wenn year===2022, dann wird value=„yes“ sonst value=„no“



## Wiederholungen II

---

```
let i = 0;
while (++i < 5)
  alert("i: " + i);
```

Prinzipiell keine Unterschied zu Java, vor der Auswertung des Vergleiches wird i inkrementiert: 1,2,3,4 ist die Ausgabe!

```
for (let i = 0; i < 5; i++)
{
  if (i == 1)
    continue;
  if (i == 3)
    break;
  alert(i);
}
```

Führt den nächsten Durchlauf mit i=2 aus!

Bricht den Durchlauf komplett ab, so dass i=3 und für i>=3 alle weiteren Durchläufe nicht ausgeführt werden!

Ausgabe: 0, 2

# Zeichenketten Funktionen

---

```
let str="Mr. Big";
let mrStr=str.slice(0,2); //Start=0,länge=2
// mrStr="Mr"
let bigStr=str.slice(3); //Start=3,länge bis Ende
// bigStr="Big"
let lastChar=str.slice(-1); //1 Zeichen vom Ende(-)
// lastChar="g"
let secondChar=str[1]; //2.tes Zeichen
// secondChar="r"
let arrStr="A;BC;CEF".split(";"); //Aufspaltung in Array durch ;
// arrStr=["A","BC","CEF"]
let arrMrBig=str.split(""); //Aufspaltung in Array durch ;
// arrStr=["M","r",.....,"g"]
alert(arrMrBig[1]);
```

Weitere String Funktionen:  
substr(), replace(),  
toUpperCase(),  
charCodeAt()  
(ermittelt UTF16 code(0..65535))

Änderung ganz oder gar nicht! str[1]=„X“ bewirkt nichts!

# Arrays

---

```
let str=[1,"2",true];
let arrFromString=Array.from('123'); // String in array verwandeln
let twoElements=[ 1, 2, 3, 4 ].slice(1, 3); //Ab 2. Zeichen bis 3. Zeichen
let stringFromArr=[ 1, 2, 3, 4 ].toString(); //In string verwandeln: '1234'
alert([1,2,3].includes(1));
//Gibt true aus
let addedArr=[1,2,3].push(4);
// addedArr =[1,2,3,4]
// Alternative: Destructuring
//addedArr=[...[1,2,3],4];
[1,2,3].forEach(value=>alert("value:"+value));
// Gibt 1,2,3 aus
let reducedArr=[ 'a', 'b', 'c' ].reduce((acc, curr) => acc + curr, 'd');
// "dabc"
```

<https://simplernerd.com/js-arrays-cheat-sheet/>



# Funktionen

```
function testFunc(a, b = 2) {  
    if (a == undefined) {  
        alert("First Parameter not defined!");  
        return 0;  
    }  
    return a + b;  
}  
  
alert("testFunc(1,3) " + testFunc(1, 3));  
alert("testFunc(1) " + testFunc(1));  
alert("testFunc(1) " + testFunc());  
alert("testFunc " + testFunc());
```

Optional parameter b

a=1; b=3

a=1; b=2

a=undefined; b=2

127.0.0.1:5500 enthält

```
testFunc function testFunc(a, b = 2) {  
    if (a == undefined) {  
        alert("First Parameter not defined!");  
        return 0;  
    }  
    return a + b;  
}
```

OK

Ohne Parameter wird der ganze Quellcode ausgegeben!

# Pfeil-Funktionen

```
function addOld(a, b) {  
    return a + b;  
}
```

```
let add = (a, b) => a + b;  
let double = n => 2 * n;
```

```
let addMulti = (a, b) => {  
    alert("Do a something here!");  
    return a + b;  
}
```

Gleichwertige Variante

Weglassen Klammern bei einem Parameter

Variante, wenn im Funktionsrumpf  
zwei Anweisungen oder mehr stehen:

- {}
- return

```
alert("add(1,2) " + add(1, 2));  
alert("double(1,2) " + double(4));
```

# Klassen I

---

```
class User {  
  constructor(name) {  
    this.name = name;  
  }  
  
  sayHi() {  
    alert(this.name);  
  }  
}  
  
let user = new User("John");  
user.sayHi();
```

## Klassen II

---

```
class User {  
  constructor(name) {  
    this.name = name;  
    this.sayHi2 = () =>  
alert(this.name);  
  }  
  sayHi = () => alert(this.name);  
}
```

```
let user = new User("John");  
user.sayHi();  
user.sayHi2();
```