# Javascript Grundlagen

## Repetition 2

WEB PROFESSIONALS

# Fragen zu Übungen?

# Switch Statement

```
switch (expression) {
  case value1:
    // falls expression mit value1 übereinstimmt
    [break;]
  case value2:
    // falls expression mit value2 übereinstimmt
    [break;]
  ...
  default:
    // falls keine der case-Klauseln mit expression
übereinstimmt
    [break;]
}
```

# Do While

```
var i = 0;
do {
    i += 1;
    console.log(i);
} while (i < 5);
```

# Spezialwörter (return)

Code ausführung der Funktion beenden & Wert zurückgeben

Hilfreich für Return Early

```
if (nameKorrekt(name) {
    if (vornameKorrekt(vorname) {
        if (telKorrekt(tel) {
            // Kunde Speichern
        }
    }
}
```

```
if (!nameKorrekt(name)
    || !vornameKorrekt(vorname)
    || !telKorrekt(tel) {
    return;
}
// Kunde Speichern
```

# Spezialwörter (continue)

Überspringt diesen Schlaufendurchgang

```
for (var i=0; i<a.length; i++) {
  if (i % 2 == 0) {// Jede zweite Zeile überspringen
    continue; // Überspringen dieses Durchganges
  }
  // Wird nicht ausgeführt, wenn continue; ausgeführt
}
```

# Spezialwörter (break)

Bricht die Schleife komplett ab

```
for (var i=0; i<a.length; i++) {
  if (i == 3) {// Jede zweite Zeile überspringen
    break; // Schlaufendurchlauf beenden
  }
}
```

# String Operationen

| | | |
|---|---|---|
| "ABCDEFGH".toLowerCase() | => | "abcdefgh" |
| "Abcdefgh".toUpperCase() | => | "ABCDEFGH" |
| "ABCDEFGH".length | => | 8 |
| "ABCDEFGH".slice(1, 4) | => | "BCD" |
| "123123123123".split("3") | => | ["12", "12", "12", "12"] |
| "ABCDEFGH".substr(2) | => | "CDEFGH" |

# Array Slice

Neuen Array aus Array Bereich erstellen

**arr.slice([begin[, end]])  (bis und ohne end)**

```
var fruits = ['Banana', 'Orange', 'Lemon', 'Apple', 'Mango'];
var citrus = fruits.slice(1, 3);

// fruits enthält  ['Banana', 'Orange', 'Lemon', 'Apple', 'Mango']
// citrus enthält ['Orange','Lemon']
```

# Array Push

Element(e) ans Ende des Arrays hängen

```
var fruits = ['Banana', 'Orange', 'Lemon', 'Apple', 'Mango'];
fruits.push('Ananas');

// fruits enthält  ['Banana', 'Orange', 'Lemon', 'Apple', 'Mango', 'Ananas']
```

# Array Pop

Letztes Element aus Array nehmen

```
var fruits = ['Banana', 'Orange', 'Lemon', 'Apple', 'Mango'];
var mango = fruits.pop();

// fruits enthält  ['Banana', 'Orange', 'Lemon', 'Apple']
// mango enthält 'Mango'
```

# Array Shift

Erstes Element aus Array nehmen

```
var fruits = ['Banana', 'Orange', 'Lemon', 'Apple', 'Mango'];
var banana = fruits.shift();

// fruits enthält  ['Orange', 'Lemon', 'Apple', 'Mango']
// banana enthält 'Banana'
```

# Array Join

Arrays verketten

```
var a = ['Wind', 'Rain', 'Fire'];
console.log(a.join(','));      // 'Wind,Rain,Fire'
console.log(a.join(', '));  // 'Wind, Rain, Fire'
console.log(a.join(' + ')); // 'Wind + Rain + Fire'
console.log(a.join(''));    // 'WindRainFire'
```

# Array IndexOf

Im Array suchen

```
var array = [2, 9, 9];
array.indexOf(2);        // 0
array.indexOf(7);        // -1
```

```
// Finde Hans
var array = ["Hugo", "Hans", "Henrieta"];
var idx = array.indexOf("Hans");
if (idx !== -1) {
    console.log("Hans ist an Position " + (idx + 1));
} else {
    console.log("Hans nicht gefunden");
}
```

# Array Sort

Array sortieren

```
var fruit = ['cherries', 'apples', 'bananas'];
console.log(fruit.sort());          // ['apples', 'bananas', 'cherries']
var scores = [1, 10, 21, 2];
console.log(scores.sort());         // [1, 10, 2, 21]
                                    // Achtung: 10 kommt vor 2
                                    // Später mehr dazu
```

# Array Sort mit Funktion

```
var scores = [1, 10, 21, 2];
function compare(a, b) {
  if (a < b) {
    return -1;
  }
  if (a > b) {
    return 1;
  }

  return 0;
}
var sortierteScores = scores.sort(compare)
console.log(sortierteScores) // [1, 2, 10, 21]
```

# Array Splice

Aus Arrays entfernen oder einfügen

```
array.splice(index, deleteCount[, element1[, element2 [, ...]]])
```

```
var personen = ['Hans', 'Martin', 'Peter', 'Fabian'];
var letzteZwei = myFish.splice(2, 2);

console.log(letzteZwei);     // => [Peter", "Fabian"]
console.log(personen);       // => ["Hans", "Martin"]
```

```
var personen = ['Hans', 'Martin', 'Peter', 'Fabian'];
var fabian = myFish.splice(3, 1);

console.log(fabian);         // => ["Fabian"]
console.log(personen);       // => ["Hans", "Martin", "Peter"]
```

# "Gefährlich"

# Array Map

Eine Funktion auf alle Elemente des Arrays anwenden

```
var zahlen = [1, 4, 9];
var wurzeln = zahlen.map(Math.sqrt);
// wurzeln ist [1, 2, 3], numbers ist immernoch [1, 4, 9]
var verdoppelt = zahlen.map(function(zahl) { return zahl * 2; });
// verdoppelt ist [2, 8, 18]
```

# Array Filter

Anhand einer Funktion einen Array filtern

```
var zahlen = [0, 1, 2, 3];
var groesser3 = zahlen.filter(function(zahl) {
    return zahl > 3;
});
console.log(groesser3)                // [0, 1, 2]
```

# Array Reduce

Alle Elemente des Arrays auf ein Element reduzieren

```
var summme = [0, 1, 2, 3].reduce(function(start, zahl) {
      return start + zahl;
   }, 0);
// Summe ist 6
// Wird so berechnet:  (((0 + 1) + 2) + 3)
```

# Übungen zu Array & String Operationen

Kopiert aus dem Google Drive Datei aus:
**Module/Javascript/lektion2.zip** in euer **htdocs** Verzeichnis in einen eigenen Ordner und macht:

- uebung_1_0.html
- uebung_1_1.html
- uebung_1_2.html
- uebung_1_3.html
- uebung_1_4.html

# Funktionen

Funktionen erlauben es Aktionen die mehrfach verwendet werden, zu verallgemeinern.

Funktionen haben einen oder mehrere Parameter und einen Rückgabewert

# Funktionen ohne Parameter

```
// Definition
function gibWasAus() {
    console.log('Hallo');
}
// Aufruf
gibWasAus();                          // Auf der Konsole: Hallo
```

# Funktionen mit Parameter

```
// Definition
function gibWasAus(ausgabe) {
    console.log(ausgabe);
}
// Aufruf
gibWasAus("Hallo Peter");                    // Auf der Konsole: Hallo Peter
```

# Funktionen mit Parametern

```
function addiere(a, b) {
    console.log(a + b);
}
// Aufruf
addiere(5, 2)                    // => Auf der Konsole: 7
```

# Funktionen mit Rückgabewert

```
function addiere(a, b) {
    return a + b;
}
var resultat = addiere(5, 2);        // Rückgabewert in Variable
console.log(resultat);               // 7 auf Konsole
console.log(addiere(10, 2));         // 7 auf Konsole
```

# Funktionen Default Werten

```
function summe(a, b, c) {
    a = a || 0;
    b = b || 0;
    c = c || 0;
    return a + b + c;
}
summe(1);
summe(1,2);
summe(1,2,3)
```

# Funktionen mit flexiblen Parametern

```
function summe() {
    var summe = 0;
    for (var i=0; i < arguments.length; i++) {
        summe += arguments[i];
    }
    return summe;
}
summe(1);
summe(2,3);
summe(2,3,4,5);
```

# Wichtiges zu Funktionen

Es gibt 2 Arten von Funktionen (**deklarierte Funktionen** und **"named Functions"**)

Funktionen sind auch Objekte

Funktionen müssen deklariert sein, bevor sie aufgerufen werden können

Variablen innerhalb von Funktionen sind ausserhalb nicht Gültig.

Funktionen können in Funktionen vorkommen

**Funktionsnamen dürfen nicht gleich wie Variablen heissen**

# Named Functions "funktionen als Variable"

```
var addiere = function (a, b) {
    return a + b;
}
addiere(2,3); // => 5
```

```
function addiere (a, b) {
   return a + b;
}
addiere(2,3) // => 5
```

Was ist der Unterschied?

# Named Functions "funktionen als Variable"

```
addiere(2,3); // Fehler
var addiere = function (a, b) {
    return a + b;
}
```

Fehler

```
addiere(2,3) // => 5
function addiere (a, b) {
   return a + b;
}
```

Kein Fehler

# Funktionen Verschachtelung

```
function getSumme() {
  var num1 = 2,
      num2 = 3;

  function addieren() {
    return name + ' scored ' + (num1 + num2);
  }

  return addieren();
}
```

# Funktionen als Parameter

```
function mwstCH(preis) {
    return preis * 1.078;
}
function mwstDE(preis) {
    return preis * 1.2;
}
function kosten(mwstFunktion) {
  var kostenOhneMwst = 200;

  return mwstFunktion(kostenOhneMwst);
}
kosten(mwstCH); // 215.60
kosten(mwstDE); // 240
```

# Variablen: Gültigkeit (Scope)

```
var a = "a-ausserhalb";
function test() {
    var a = "a-innerhalb";
    var b = "b-innerhalb";
    console.log(a);
    console.log(b);
}
test();
console.log(a);
console.log(b);
```

```
-> a-innerhalb
-> b-innerhalb
-> a-ausserhalb
-> Uncaught ReferenceError: b is not defined
```

# Var Gültigkeit Verschachtelte Funktionen

```
function getSumme() {
  var num1 = 2,
      num2 = 3;

  function addieren() {
    return name + ' scored ' + (num1 + num2);
  }

  return addieren();
}
```

# Übungen zu Funktionen

Kopiert aus dem Google Drive Datei aus:
**Module/Javascript/lektion2.zip** in euer **htdocs** Verzeichnis in einen eigenen Ordner und macht:

- uebung_2_0.html
- uebung_2_1.html
- uebung_2_2.html
- uebung_2_3.html
- uebung_2_4.html
- uebung_2_5.html (anspruchsvoll)