JS 2

Daten & Strukturen in JS

Daten strukturieren

```
// ================================
// = Die "einfache" Form: **Arrays** =
// ================================

var a = []; // Ein leeres Array
```

```
// Arrays "speichern" Sequenzen von Werten.
// Die Werte werden einfach mit Kommata getrennt.

var a = [1,2,3,4,5];

console.log(a);
```

```
// Diese können auch verschiedene "Typen" an Daten enthalten.
a = [ "Klaus Dieter Müller", 2, "WS1213"];

// ... auch weitere Arrays:
a = [ "Klaus Dieter Müller", 2, ["WS1213",["Scriptings",2,3,90]];
```

```
// Leserlicher wird es mit Lerrzeichen und Umbrüchen.
// Lesbarer Code ist guter Code!!

a = [
  "Klaus Dieter Müller", // Name
  2, // Studiensemester
  [
    "WS1213", // Welches Semester
    ["Scriptings",2,3,90], // Veranstaltung 1
    ["poetische Apparate",2,3,90] // Veranstaltung 2
  ]
]; // Ende des Arrays!
```

```
// Einträge des Arrays direkt finden
// Die _Adressierung_ eines Arrays erfolgt mit einem
// sogenannten _Index_ in eckigen Klammern – [ und ].
// Dieser ist numerisch und begint bei 0 – also:

console.log("Name: "+a[0]);


// Die verschiedenen _Dimensionen_(Tiefe) eines Arrays
// erreicht man mit aufeinanderfolgenden Indizes:

console.log("Titel der Veranstaltung: "+[2][1][0]);
```

Sind Daten so einfach?

Everything is an Object

```
// ========================
// = Komplexer: Objekte =
// ========================

// Ein leeres Objekt:
var semester = {};
```

```
// Objekte können _Properties_ haben – Properties werden auch "keys" genannt.
// Ein Objekt stellt einen sogenannten "Key/Value Store" dar, d.h. nichts
// anderes, als: Jeder _Schlüssel_ hat einen _Wert_.

semester.welches = "WS";
//       ^^ key    ^^^^ value

semester.jahr = "2012/13";
```

```
// Eine alternative, direkte Notierung verwendet einen Doppelpunkt zwischen
// Schlüssel und Wert. Paare werden mit Kommata getrennt.

semester = {
  welches: "WS",
  jahr: "2012/13"
};

console.log(semester.welches,semester.jahr);
```

```
// Die _Schachtelung_ kann auch mehr als ein Ebene tief sein:
semester.student = {name:"Klaus Dieter Müller",studiensemester:2};
console.log(semester.student.name + " ist im " + semester.welches+semester.jahr);
```

```
// Auch hier gilt: Lesbarer Code ist guter Code!
semester = {
  student: {
    name: "Klaus Dieter Müller",
    studiensemester: 2
  },
  welches: "WS",
  jahr: "2012/13",
  veranstaltungen: [ // Achtung! Es folgt ein Array (im Objekt): andere Klammer
    {
      titel: "Scriptings",
      sws: 2,
      credits: 3,
      workload: 90
    },
    {
      titel: "poetische Apparate",
      sws: 2,
      credits: 3,
      workload: 90
    }
  ]
};
```

```
// Die _Addressierung_ der Werte in einem Objekt kann, wie oben schon in der
// Zuweisung mit . erfolgen:
console.log(semester.student.name+" studiert im "+semester.student.studiensemester+" Semester.");

// oder auch in Kombination:
console.log(semester.student.name+", willkommen im "+semester.veranstaltungen[0].titel+" Seminar!");
```

```
// Tip: Wenn Datensätze in Objekte abgebildet werden,
// sollte man zur späteren Verwendung _konsistent_,
// d.h. nach einem einheitlichen Strukturprinzip
// der _Schlüsselnamen_ und der _Typen der Werte_
// arbeiten!
```

```
// ========================================================
// = Aufgabe: Deine Semesterbelegung als strukturiertes Objekt =
// ========================================================

var mein_semester = {
  // bilde hier dein Semester ab!
};

// Als Zusatz: wandle dein Semester in ein JSON-String um und "parse" diesen
// wieder in ein Object zurück. Die Variablen sind schon angelegt.

var jsoned_semester,semester_from_json;
```