JavaScript 5

Inhalt:

1. Objekte
2. Arrays

1)JavaScript Objekte

Objekte besitzen normalerweise Eigenschaften und Methoden.

- Eine Eigenschaft ist wie eine Variable, die einem Objekt zugewiesen wird. Sie beschreibt das Objekt.
- Eine Methode ist wie eine Funktion, die aber zu einem Objekt gehört. Sie beschreibt Dinge, die das Objekt machen kann.

Objekte haben eine eigene Notation, also Schreibweise, beim Aufruf:

- Eine Methode wird immer mit dem Namen des Objekts,
- einem Punkt als Trennzeichen und
- dem Namen der Funktion (bzw. der Methode) aufgerufen.

Beispiel: dasObjekt.dieMethode()
Math.round()

Ein Objekt ist nicht nur eine verbesserte Funktion. Ein Objekt kann eine Vielzahl eigener Funktionen und viele eigene Variable haben. Dabei werden die Funktionen von Objekten oft als Methoden, die Variablen als Attribute bezeichnet. Methoden in Objekten werden auch mit dem Schlüsselwort „function“ definiert.

Objekte sind Datenelemente mit Eigenschaften und Methoden, also eine Ansammlung von thematisch zusammenpassenden Informationen und Funktionen. Diese lassen sich über eine Punktnotation abrufen, wie oben beschrieben.

Fertige Objekte

Es gibt feste, statische, fertige Objekte, auf die man zugreifen kann, z.B. „Math“.
Math hat viele unterschiedliche Methoden (Funktionen), z.B. Math.Pi oder Math.round

Achtung: das M bei Math ist immer groß!

Beispiele des vorhandenen und fertigen Objektes „Math“:

```
var pi = Math.PI
pi = Math.round(pi);
document.write(pi);
```

Ergebnis: 3

- In diesen drei Zeilen wird die Eigenschaft „PI“ des Math-Objektes abgerufen und in einer Variablen abgespeichert.
- Anschließend wird diese mit der Funktion „Math.round)()“ gerundet und
- per „document.write()“ ausgegeben.

Folgende Objekte mit ihren Eigenschaften und Methoden sind ein fester Bestandteil der JavaScript-Syntax. Es gib, je nach JavaScript-Implementierung, 30 bis 40 feste Objekte, auf die man zurückgreifen kann.

Die wichtigsten sind:

- document: enthält Infos über das HTML-Dokument
- Date: kann die aktuelle Zeit und Datum zurückgeben
- Math: kann gut rechnen
- Array: eine Kette von mehreren Variablen gleichen Typs

Erstellen: Ein Objekt kann man erstellen mit

- var
- let
- const

Übung 1 – eigenes Objekt erstellen

Erstelle ein neues Dokument namens „5.javascript.kind_objekt.html“.
Ziel ist die Ausgabe der Eigenschaften des Kindes in einem Alert-Fenster.

1. Variante 1: Ein neues Objekt wird mit „newObject();“ deklariert. Man muss jedoch auf Groß- und Kleinschreibung achten. Leerzeichen und Sonderzeichen sind in Namen und Eigenschaften nicht erlaubt.
2. Variante 2: erstellen mit {}
   innerhalb der geschwungenen Klammern können Eigenschaften definiert werden. Dies erfolgt mit „key – value“ Paaren. Gibt es mehr als eine, muss ein Komma dazwischen stehen.
   z.B.:

```
let Max = {
  groeße: 180
};
```

Übung Variante 2: Hier wird die 2. Art der Schreibweise genutzt, nämlich { }:

```
var person = {

        firstname: "Patrick",
        lastname: "Spongebob",
        hometown: "Digbiztown",
        birth: "2017"

};
```

Beispiel – es soll nur die Eigenschaft „name“ ausgegeben werden

obwohl ja mehrere Eigenschaften vorhanden sind.

Dafür schreibt den Namen der Variable (hier: person) und mit einem Punkt verknüpft die Eigenschaft danach hin.

Lösung: document.write(person.name);

Innerhalb eines Objektes kann man das selbige Objekt mit „**this**“ ansprechen. Siehe Zeile 12 im Beispiel.

Beispiel:

```
<script>
var person = {
        name: "Muster",
        vorname: "Heinz",
        getName: function() {return this.name + " " + this.vorname }
};
document.write(person.name);
</script>
```

```
1 ▼ <html>
2 ▼ <head>
3   <meta charset="utf-8">
4   <title>Unbenanntes Dokument</title>
5   </head>
6
```

```
<body>
<script>
var person = {
    name: "Muster",
    vorname: "Heinz",
    getName: function() {return this.name + " " + this.vorname }
};
document.write(person.name);
</script>
</body>
</html>
```

Sie können auch geschachtelt werden. Der Vorteil ist, dass dadurch einfach Strukturen erzeugt werden können.

2)Arrays

Eine Funktion ist eine benannte Gruppe von Codezeilen.

Ein ARRAY ist etwas Ähnliches.

- Array werden mit [ ] „eckigen Klammern“ definiert.
- Arrays sind besondere Arten von Variablen. Darin kann man nicht nur einen Wert speichern, sondern eine Liste von Werten.
- Es handelt sich dabei um eine benannte Gruppe von Variablen.
- Man kann Arrays verwenden, wenn man mit einer Liste von Werten desselben Datentyps arbeiten möchte.
- Ein neues Array wird ganz ähnlich wie ein Objekt mit dem Konstruktor „new Array()“ angelegt. Man kann das „new Array()“ aber auch weglassen.
- Anschließend weißt man der Arrayvariablen so viele Werte zu, wie man möchte, in eckigen Klammern, getrennt durch Kommas. Das Leerzeichen nach dem Komma ist keine Pflicht, aber es hilft bei der Lesbarkeit.

Hier werden, wie bei Objekten, sogenannte „key – value“ Paare erstellt. Der key ist die Reihung beginnend mit „0“.

Beispiel:

Erstelle das File „5.js.erstesarray.html“:

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Title</title>
6  </head>
7  <body>
8      <script>
9          var fruits;
10         fruits =  ['Orange', 'Apfel', 'Zitrone'];
11         //Ausgabe:
12         document.write(fruits[0]);
13     </script>
14 </body>
15 </html>
```

Code:

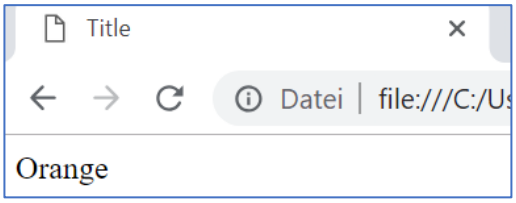
```
<script>
    var fruits;
    fruits =  ['Orange', 'Apfel', 'Zitrone'];
```

```
        //Ausgabe:
        document.write(fruits[0]);
    </script>
```

oder mit alert:

```
        alert('ich liebe ' + fruits[0] + ' .');
```

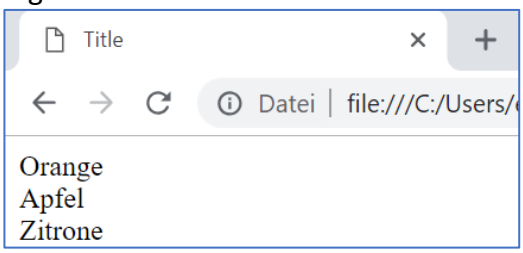
Ergebnis:



Alle ausgeben: FOR-Schleife verwenden

```
for (i = 0; i<3; i++)
        document.write(fruits[i] + "<br>");
```

Ergebnis:



bzw.

```
        for (i = 0; i<fruits.length; i++)
        document.write(fruits[i] + "<br>");
```

Funktion in einem Array: length

Die Länge des Arrays **(fruits.length)** wird in der Bedingung der for-Schleife verwendet. Statt die Länge mit 3 anzugeben, verwendet man die Eigenschaft „length“. Der Vorteil ist, dass die Schleife automatisch an die Größe des Arrays angepasst wird, auch wenn man Elemente hinzufügt oder entfernt.

Eine normale Variable kann nur einen Wert aufnehmen. Ein Array kann beliebig viele Elemente aufnehmen.

Kurzschreibweise:

Die einzelnen Werte werden nacheinander in die eckigen Klammern geschrieben. Getrennt werden sie jeweils durch ein Komma. Das Leerzeichen nach dem Komma ist keine Pflicht, aber es hilft bei der Lesbarkeit.

Nach dem letzten Element sollte kein weiteres Komma geschrieben werden.

```
var dieMannschaft = [ "Hans", "Josef"];
```

Zahlen können ohne Anführungszeichen geschrieben werden. Texte und Zahlen können auch ohne Probleme gemischt vorkommen.

```
var allesMoegliche = [ "Hans", 42, 25, "Patrick"];
```

Man kann auch ein Array so schreiben:

Mit dem Schlüsselwort „newArray " und runden Klammern kann man ebenfalls ein Array anlegen.

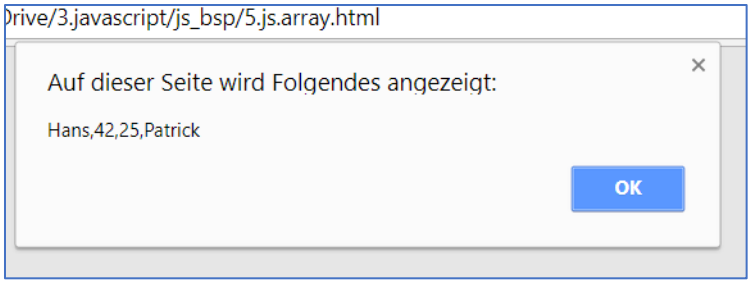
```
<script>
    var allesMoegliche = newArray("Hans", 42, 25, "Patrick");
    alert(allesMoegliche);
</script>
```

Das Schlüsselwort „new" ist auch in anderen Programmiersprachen bekannt. Es ist ein zentrales Wort der objektorientierten Programmierung. Damit erschafft man neue Objekte oder neue Arrays. Die runden Klammern deuten auf eine „Verwandtschaft" mit Funktionen hin.

Zugriff auf die Elemente:

- Man kann sich z.B. den gesamten Inhalt des Arrays mit „alert" ausgeben lassen.

```
<script>
    var allesMoegliche = [ "Hans", 42, 25, "Patrick"];
    alert(allesMoegliche);
</script>
```



- **Zugriff auf einzelne Elemente**

Jedes Element, jeder Wert eines Arrays bekommt automatisch eine eindeutige, fortlaufende Nummer. Über diese Nummer kann man jeden Wert auslesen, einer anderen Variablen zuordnen oder sogar direkt als Teil einer Berechnung verwenden.

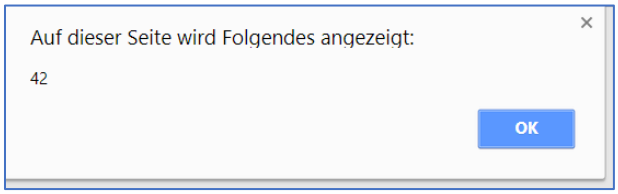
Abrufen lassen sich die Werte eines Arrays über den Namen der Variablen und die Position innerhalb des Arrays, ebenfalls in eckigen Klammern.

Die Position, also die Nummer in den eckigen Klammern, nennt man Schlüssel.

Jedoch beginnt der Computer mit „0“ zu zählen an, nicht bei „1“.

```
alert( allesMoegliche[1]);
```

Lösung:



Das Ergebnis ist also nicht „Hans“ sondern „42“.

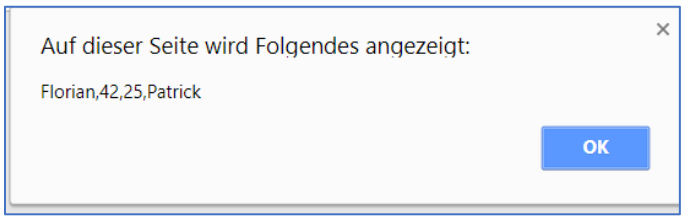
- **Einen Wert ändern**

Soll ein Wert geändert werden, kann man diese Position so ändern:

```
allesMoegliche[0] = "Florian";
alert(allesMoegliche);
```

Aber man soll auch die Ausgabe nicht vergessen.

Ergebnis:



Speichern unter 5.array_2.html.

Info: Zusätzliche Funktionen

- push() – am Ende hinzufügen
- unshift() – am Anfang hinzufügen

- und pop() – den letzten Wert löschen
- shift() – den ersten Wert löschen
- mit join() mehrere Arrays vereinen und
- mit sort() sortieren.

Quelle:

https://www.youtube.com/watch?v=cAiSbkWvBQU ab ca. 56:30 Minuten (Mario,2022)

Florian Franke, Johannes Ippen; in: Apps mit HTML5, CSS3 und JavaScript;
Rheinwerk Verlag, Bonn 2015; S. 106-107

Andy Harris, in: JavaScript für dummies; Wiley-Vch Verlag, 2017, Weinheim; S.128-143