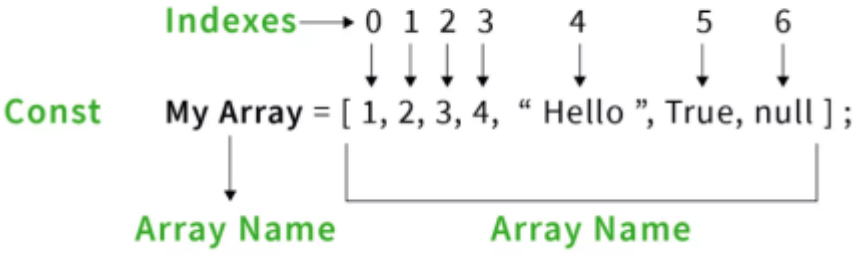
LU04a - Arrays

Einleitung

In der Programmierung stösst man schnell auf Situationen, in denen man nicht nur einen einzelnen Wert, sondern eine **Sammlung von Werten speichern** möchte – zum Beispiel eine Liste mit Namen, Zahlen oder Objekten.

In JavaScript gibt es dafür den Datentyp *Array*. Ein *Array* ist eine geordnete Liste, in der jedes Element über einen Index angesprochen werden kann (beginnend bei 0).



Arrays sind unglaublich vielseitig: Man kann Elemente hinzufügen, entfernen, sortieren oder mit speziellen Methoden wie *map* und *filter* weiterverarbeiten.

Was ist ein Array?

Ein Array ist eine geordnete Sammlung von Werten. Diese Werte können Zahlen, Strings, Objekte oder sogar andere Arrays sein. Arrays sind besonders praktisch, wenn man mehrere Werte unter einem Namen speichern möchte.

Beispiel:

let zahlen = [1, 2, 3, 4, 5];

```
let namen = ["Anna", "Ben", "Clara"];
```

===== Zugriff auf Array-Elemente ===== + Jedes Element in einem Array hat einen Index, beginnend bei 0. +

let farben = [„rot“, „gruen“, „blau“];

```
console.log(farben[0]); // "rot"
console.log(farben[1]); // "gruen"
console.log(farben[2]); // "blau"
```

Methoden

*Hinweis: **Wir müssen etwas vorgreifen mit dem Thema *Methoden*. In JavaScript sind Methoden nichts anderes als Funktionen, die an Objekte wie beispielsweise *ARRAYS* gebunden sind. * Eine Methode wird wie eine normale Funktion definiert, aber als Eigenschaft eines Objekts gespeichert. * D.h. man kann diese Objekte zum reagieren bringen wie beispielsweise: * Hey *myArray*, sag doch bitte wie lang Du bist –> myArray.length() Sie kann dann mit der Punkt-Notation aufgerufen werden.** Beispiel**

```
const person = {
  name: "Anna",
  greet: function() {
    return "Hallo, mein Name ist " + this.name;
  }
};
console.log(person.greet());
```

Array-Laenge

```
let zahlen = [10, 20, 30];
console.log(zahlen.length); // 3
```

Elemente hinzufügen oder entfernen

JavaScript stellt praktische Methoden bereit:

let tiere = [„Hund“, „Katze“];

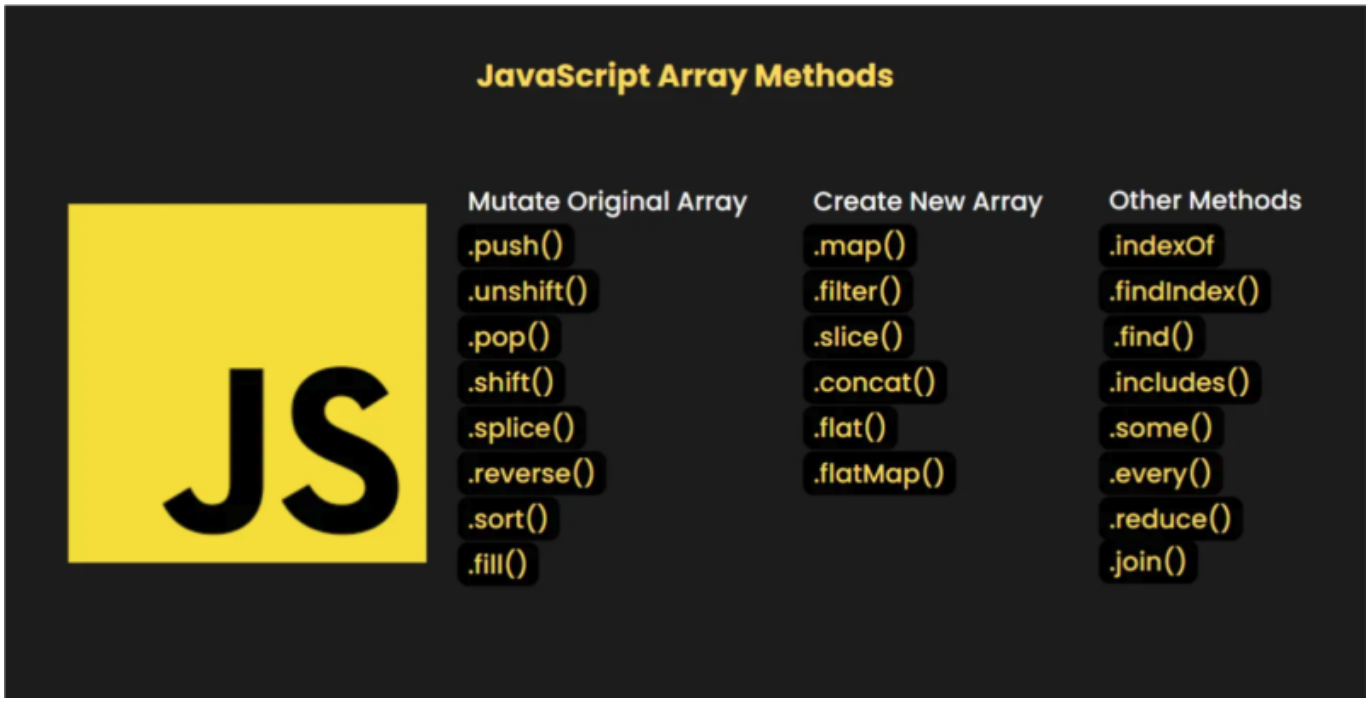
Hinzufügen tiere.push(„Maus“); ans Ende

```
tiere.unshift("Vogel"); // an den Anfang
console.log(tiere);     // ["Vogel", "Hund", "Katze", "Maus"]
// Entfernen    +
tiere.pop();    // entfernt letztes Element
tiere.shift();  // entfernt erstes Element
console.log(tiere); // ["Hund", "Katze"]
```

Besondere Methoden

- map → erstellt ein neues Array basierend auf jedem Element.
- filter → filtert Elemente nach Bedingung.
- find → findet das erste passende Element.
- includes → ist das Element vorhanden?

Übersicht über Array-Methoden



Lernvideos

JavaScript-Tutorial - 8:31 Minuten	JavaScript-Tutorial - 20:30 Minuten

Zusatzmaterial

- W3School- JavaScript Arrays
- SelfHTML - JS Arrays
