Programmieren Starten

Zusammenfassung

Arrays

JavaScript - Dynamische Webseiten entwickeln

EINLEITUNG ARRAYS

Man verwendet Arrays, wenn man nicht nur einen Wert speichern möchte, sondern eine ganze Ansammlung an Werten. Ein Array kann man sich vereinfacht auch wie eine Liste vorstellen, auf der viele verschiedene Werte stehen.

Erstellung eines Arrays

Ein Array, das verschiedene Namen enthalten soll, kann wie im folgenden Beispiel erstellt werden. Die Elemente des Arrays, in diesem Fall sind das die Namen, werden mit einem Komma voneinander separiert aufgelistet.

Beispiel:

```
names = [“Jannick“, “Peter“];
```

Index

Beim Index handelt es sich um die **Position** eines Wertes im Array. Der Name „Jannick“ steht an **erster Stelle** und hat somit den **Index 0**. Über den Index kann beispielsweise ein Wert aufgerufen werden.

WICHTIG

Der Computer fängt bei 0 an zu zählen. Das heißt: Das **erste Element** eines Arrays hat den **Index 0**, das **zweite** Element den **Index 1**, usw.

Beispiel:

```
names = [“Jannick“, “Peter“];
console.log(names[0]);          //Ausgabe: Jannick
console.log(names[1]);          //Ausgabe: Peter
```

Ein Array kann auch Ganzzahlen enthalten.
Beispiel:

```
numbers = [1, 10, 20];
```

Das Besondere an JavaScript ist zudem, dass auch Arrays mit Elementen von verschiedenen Datentypen erstellt werden können:

```
random = [“X“, 10, “Jannick“];
```

ZUGRIFF AUF SPEZIFISCHE ARRAY ELEMENTE

Man kann bei Arrays auf spezifische Elemente zugreifen und diese beispielsweise ändern oder abfragen.

Elemente überschreiben

Im folgenden Beispiel wird über den Index 0 das erste Element des Arrays überschrieben und anschließend ausgegeben.
Beispiel:

```
fruits = [“Aplpe“, “Banana“, “Orange“];
fruits[0] = “Apple“;
console.log(fruits);      //Ausgabe: [‘Apple‘, ‘Banana‘, ‘Orange‘]
```

Elemente ausgeben

Im folgenden Beispiel wird das dritte Element über den Index 2 des Arrays ausgegeben.
Beispiel:

```
fruits = [“Apple“, “Banana“, “Orange“];
console.log(fruits[2]);      //Ausgabe: Orange
```

DIE ARRAY LENGTH-PROPERTY

length

Die length-Property ermöglicht es, die Länge eines Arrays herauszufinden. Es handelt sich dabei um die Anzahl der Elemente.
Beispiel:

```
fruits = [“Apple“, “Banana“, “Orange“];
console.log(fruits.length);              //Ausgabe: 3
```

Auf das letzte Array Element zugreifen

Im folgenden Beispiel wird auf das letzte Element des Arrays zugegriffen. In den eckigen Klammern nach „fruits“ in der Konsolenausgabe muss der Index angegeben werden. Da fruits.length den Wert 3 ergibt, in diesem Array aber nur Index 0, 1 und 2 existieren, schreibt man fruits.length – 1, um auf das letzte Element des Arrays zugreifen zu können.

Beispiel:

```
fruits = [“Apple“, “Banana“, “Orange“];
console.log(fruits[fruits.length - 1]);          //Ausgabe: Orange
```

ELEMENTE ZUM ARRAY HINZUFÜGEN

push()

Mithilfe der push-Methode (Funktion) kann einem Array ein neues Element hinzugefügt werden.

Beispiel:

```
var names = [“Jannick“, “Peter“, “Maria“];
names.push(“Klaus“);

console.log(names);      //Ausgabe: [‘Jannick‘, ‘Peter‘, ‘Maria‘, ‘Klaus‘]
```

ELEMENTE AN BESTIMMTER STELLE IM ARRAY EINFÜGEN

splice()

Mithilfe der splice-Methode können Elemente an einer bestimmten Stelle im Array hinzugefügt werden. Dabei wird in Klammern zuerst die Indexposition, an der gestartet wird, angegeben. Anschließend wie viele Elemente gelöscht und was eingesetzt werden soll.

Beispiel:

```
var numbers = [1, 3, 4, 5, 6];
numbers.splice(1, 0, 2);   //Bei Index 1 soll die Zahl 2 eingesetzt werden,
                             ohne dabei Elemente zu löschen
console.log(numbers);      //Ausgabe: [1, 2, 3, 4, 5, 6]
```

MEHRERE ARRAYS MITEINANDER VERBINDEN/VERKETTEN

concat()

Die concat-Methode fügt zwei oder mehr Arrays zusammen. Dabei wird ein neues Array zurückgegeben.

Beispiel:

```
var numbers = [1, 2, 3, 4];
var numbersTwo = [5, 6, 7, 8];

var newArray = numbers.concat(numbersTwo);
console.log(newArray);          //Ausgabe: [1, 2, 3, 4, 5, 6, 7, 8]
```

ELEMENTE AUS ARRAY LÖSCHEN

pop()

Die pop-Methode entfernt das **letzte** Element eines Arrays.

Beispiel:

```
var numbers = [1, 2, 3, 4, 5];
numbers.pop();

console.log(numbers); //Ausgabe: [1, 2, 3, 4]
```

shift()

Die shift-Methode entfernt das **erste** Element eines Arrays.

Beispiel:

```
var numbers = [1, 2, 3, 4, 5];
numbers.shift();

console.log(numbers); //Ausgabe: [2, 3, 4, 5]
```

splice()

Wie bereits oben erwähnt, können mithilfe der splice-Funktion auch Elemente an bestimmten Positionen im Array entfernt werden.

Beispiel:

```
var numbers = [1, 2, 3, 4, 5, 6, 7, 8];
numbers.splice(2, 3); //entfernt ab Index 2 drei Werte, also Zahl 3, 4 und 5
```

```
console.log(numbers);     //Ausgabe: [1, 2, 6, 7, 8]
```

ELEMENT/INDEX IM ARRAY FINDEN

find()

Mithilfe der find-Methode kann der Wert eines Elements im Array gesucht und zurückgegeben werden.

Beispiel:

```
var numbers = [1, 2, 3, 4, 5, 6, 7, 8];
var result = numbers.find(element => element == 4);

console.log(result);      //Ausgabe: 4
```

indexOf()

Mit dieser Methode kann festgestellt werden, bei welchem Index sich ein bestimmtes Element befindet.

Beispiel:

```
var numbers = [4, 6, 7, 2, 4, 3];
var result = numbers.indexOf(6);  //Bei welchem Index befindet sich der
                                  Wert 6?

console.log(result);              //Ausgabe: 1
```

lastIndexOf()

Diese Methode gibt den Index des letzten Wertes innerhalb eines Arrays zurück.

Beispiel:

```
var numbers = [4, 6, 7, 2, 4, 3, 4];
var result = numbers.lastIndexOf(4);

console.log(result);              //Ausgabe: 6
```

ARRAYS SORTIEREN BZW. UMDREHEN

sort()

Mithilfe der sort-Methode kann man Arrays sortieren, also beispielsweise in die korrekte alphabetische Reihenfolge bringen.

Beispiel:

```
var names = [“Jannick“, “Maria“, “Andreas“, “Tina“];
names.sort();

console.log(names);          //Ausgabe: [‘Andreas’, ‘Jannick’, ‘Maria’, ‘Tina’]
```

reverse()

Die reverse-Methode dreht die Reihenfolge der Array Elemente um.

Beispiel:

```
var names = [“Jannick“, “Maria“, “Andreas“, “Tina“];
names.reverse();

console.log(names);          //Ausgabe: [‘Tina’, ‘Andreas’, ‘Maria’, ‘Jannick’]
```