JavaScript-Grundlagen

1 Einbinden in HTML

Im Quellcode

```
<script> ... </script>
```

In extra Datei

```
<script src='datei.js'></script>
```

2 Ausgabe

Konsole

Firefox: Extras → Web-Entwickler → Web-Konsole
`console.log('Text');`

Popup-Fenster

`alert('Text');`

Im HTML-Dokument

`document.write('Text');`

3 Grundlegende Datentypen

JavaScript verwendet dynamische Typisierung: Der Datentyp einer Variablen wird bei Wertzuweisung festgelegt und kann sich bei Zuweisung eines anderen Wertes ändern.

Datentyp ermitteln

`typeof(variable)` oder `typeof variable` ergibt `undefined`, `string`, `number`, `boolean`, `function`, `object` oder `symbol`.

undefined

`let irgendwas; //` Variablen, denen noch kein Wert zugewiesen wurde, haben den Datentyp `undefined`.

string

`let text = 'Hallo';`
Text kann in Single-Quotes `'Text'` oder Double-Quotes `"Text"` angegeben werden. Es gibt keinen speziellen Datentyp für einzelne Zeichen.

Template literals mit Backticks

```
let zahl = 3;
let text = `${zahl} hoch 2 = ${zahl * zahl}` ;
```

number

```
let ganzZahl = 17;
let kommaZahl = -13.5;
let hexZahl = 0xff; // 255
let keineZahl = "text/ 3; // NaN
let unendlich = 3 / 0; // Infinity
let minusUnendlich = -3 / 0; // -Infinity
```

Es gibt keinen speziellen Datentyp für ganze Zahlen.

boolean

```
let ok = true;
let wahrheit = false;
```

4 Umwandlung von Datentypen

Die Konvertierung von Datentypen erfolgt in vielen Fällen automatisch. Bestimmte Umwandlungen sind aber explizit möglich:

string → number

```
let zahl = Number('42');
```

number → string

```
let text = String(42);
```

5 String: Methoden und Eigenschaften

```
let zeichen = 'Text'.charAt(1); // 'e'
let laenge = 'Text'.length; // 4
```

6 Math: Methoden und Eigenschaften

```
let zufall = Math.random(); // 0.00 - 0.99
let abgerundet = Math.floor(3.5); // 3
let pi = Math.PI;
```

7 Array []

Ein Array ist ein Object und damit eine Liste aus Schlüssel-Wert-Paaren. Die Schlüssel sind normalerweise die Zahlen 0, 1, 2, ..., die Werte eines Arrays können von verschiedenen Datentypen sein. Es ist möglich, assoziative Arrays mit anderen Schlüsseln als 0, 1, 2, ... zu erstellen. Für assoziative Arrays sollten aber besser reine Objects verwendet werden.

Beispiel

Schlüssel	Wert
0	'Anna'
1	45
2	'Berta'

erstellen

```
let feld = [];
feld[0] = 'Anna';
feld[1] = 45;
feld[2] = 'Berta';
```

oder

```
let feld = [ 'Anna', 45, 'Berta' ];
```

Anzahl der Elemente

```
let anzahl = feld.length; // 3
```

8 Function ()

Eine JavaScript-Funktion ist ein Objekt und kann deshalb in einer Variablen gespeichert werden.

Definition

Function Declaration

```
function quadrat(zahl) {
    return zahl * zahl;
}
```

oder

Function Expression

```
let quadrat = function (zahl) {
    return zahl * zahl;
}
```

oder

Arrow Functions

```
let quadrat = (zahl) => zahl * zahl;
```

Aufruf

```
let ergebnis = quadrat(3);
```

9 Object {}

Ein Objekt ist eine Liste aus Schlüssel-Wert-Paaren. Werte können auch Funktionen sein, dann werden sie Methoden genannt. Die Eigenschaften und Methoden von Objekten können auch zur Laufzeit hinzugefügt werden.

Beispiel: Object 'hund'

Schlüssel	Wert
`name`	`'Bello'`
`alter`	`5`
`wirdaelter`	`function (jahre) { this.alter += jahre; }`

einzelnes Objekt erstellen

```
let hund = {};
hund.name = 'Bello';
hund.alter = 5;
hund.wirdaelter = function (jahre) {
    this.alter += jahre;
}
```

oder

```
let hund = {
    name : 'Bello',
    alter : 5,
    wirdaelter : function (jahre) {
        this.alter += jahre;
    }
};
```

Objekt verwenden

```
hund.wirdaelter(3);
let alter = hund.alter; // 8
```

10 Mehrere gleichartige Objekte erstellen

mit Konstruktorfunktion

Es gibt in JavaScript keine Klassen. Als Schablonen für Objekte dienen Konstruktoren. Die Namen von Konstruktoren sollten mit einem Großbuchstaben beginnen.

```
function Hund(name, alter) {
    this.name = name;
    this.alter = alter;
    this.wirdaelter = function (jahre) {
        this.alter += jahre;
    };
}
```

mit class

```
class Hund {
    constructor(name, alter) {
        this.name = name;
        this.alter = alter;
    }

    wirdAelter(jahre) {
        this.alter += jahre;
    }
}
```

Objekte erstellen

```
let hund1 = new Hund('Bello', 3);
let hund2 = new Hund('Hasso', 7);
```

11 Module

JavaScript-Code kann auf mehrere Module (= Dateien) aufgeteilt werden.

beispiel.js

```
export class Beispiel {
  ...
}
```

datei.js

```
import { Beispiel } from './beispiel.js';

const beispiel = new Beispiel();
```

datei.html

```
<script type="module" src="datei.js"></script>
```