



# JAVASCRIPT 01



# EINFÜHRUNG

# EINFÜHRUNG

- Einsatzzweck: dynamische Internetseiten erstellen
- Vielfältige Möglichkeiten um Aufbau, Layout und Inhalte zu verändern
- Zählt mittlerweile zu den etablierten Web-Technologien
- Aufgrund großer Beliebtheit hat sich Anwendungsbereich ausgeweitet
  - ▶ 2009 erschien bspw Node.js => Plattform um mit JavaScript Serveranwendungen programmieren zu können
  - ▶ Später auch Entwicklungsumgebungen wie Electron oder NW.js => möglich, Desktop-Anwendungen zu erstellen
- Browser-Anwendungen bleiben mit großem Abstand der häufigste Einsatzbereich von JS
- Läuft innerhalb einer Sandbox, dadurch kein Zugriff auf Funktionen des Betriebssystems, Hardware oder zu Netzwerken möglich

# ENTSTEHUNG

- Entstehung eng mit Entwicklung der Webbrowser verbunden
- Erste Version erschien 1995 unter der Bezeichnung „LiveScript“ und wurde im Netscape Navigator 2.0 umgesetzt
- Wenige Monate nach Ersterscheinung umbenannt in JavaScript (Marketingstrategie)

# ERSTE SCHRITTE

- Früher musste innerhalb des script-Tags das Attribut `type="text/javascript"` stehen => heute nicht mehr nötig
- `alert`-Befehl erzeugt eine kleine Info-Box
- Semikolon beendet JS Befehle

```
<script type="text/javascript">  
</script>
```

```
<script>  
    alert("Hallo Welt!");  
</script>
```

# KOMMENTARE

```
<script>  
    // Ich bin ein einzeliger Kommentar  
  
    /* Ich bin ein Kommentar  
       der über mehrere Zeilen  
       gehen kann */  
</script>
```

# JS IN EINER EIGENEN DATEI

## ➤ HTML Datei

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>JS Seite</title>

  </head>
  <body>

    <script src="meinScript.js"></script>

  </body>
</html>
```

## ➤ JS Datei

```
alert('hallo Welt');
```

## „USE STRICT“

- Fehler und schlecht implementierte Funktionen wurden nicht verbessert  
Grund: Entwickler legen viel Wert auf abwärtskompatible Programmiersprache (sonst kann es passieren, dass entsprechende Funktionen plötzlich nicht mehr funktionsfähig sind)
- 2009 neue JavaScript-Version die alle bekannten Schwachstellen entfernen sollte => ECMA SCRIPT 5 (ES5) wurde veröffentlicht => nicht mehr abwärtskompatibel
- Mit "use strict"; wird dem Browser mitgeteilt, dass er die neue Version verwenden soll, ansonsten wird ES5 vom Browser nicht berücksichtigt
- Befehl muss stets am Anfang eingefügt werden
- Befehl kann nicht rückgängig gemacht werden
- Empfehlenswert neue Programme in dieser moderneren Version zu verfassen

```
<script>  
  "use strict";  
  alert('hallo Welt!');  
</script>
```



# EINE EINGABE DES ANWENDERS AUFNEHMEN

- Einfache Möglichkeit für Interaktion mit Nutzer ist der prompt-Befehl
- Erzeugt ein neues Fenster, ähnlich dem alert-Befehl

```
<script>  
  "use strict";  
  prompt("Wie alt bist du?", "Geben Sie das Alter an");  
</script>
```

- der prompt-Befehl & alert-Befehl kombiniert

```
<script>  
  "use strict";  
  alert("Sie sind " + prompt("Wie alt bist du?", "Geben Sie das Alter an") + " Jahre alt.");  
</script>
```

# KLEINE ÜBUNG

- Schreibe ein Programm, das den Besucher zum Coding-Kurs begrüßt
- Frage den Besucher nach seinem Namen und erstelle eine personalisierte Begrüßung

# VARIABLEN IN JAVASCRIPT

- Können unterschiedliche Werte aufnehmen (am häufigsten sind es Zahlen)
  - Buchstaben, Zeichen, Wörter, längeren Text, Wahrheitswerte, ...
- Jede Variable hat einen Typ
- In JS nicht notwendig den Variablentyp anzugeben  
Interpreter wählt während der Ausführung automatisch eine passende Möglichkeit aus  
auch möglich zuerst eine Zahl und dann einen String in die gleiche Variable abzulegen
- Einige Operationen lassen sich aber nur mit bestimmten Variablentypen durchführen (zB.: Division)

```
<script>  
  let meineErsteVariable;  
</script>
```

- Variable deklarieren => Initialisierung

```
meineErsteVariable = 5;  
/* oder */  
meineErsteVariable = "Ich bin ein Text";
```

- Der Variable einen Wert zuweisen

```
let meineVariable = 5, text = "Hallo du", zahl = 10;
```

- Mehrere Variablen mit Wert initialisieren
- Nicht empfehlenswert wegen Übersichtlichkeit im Code = schlechter Programmierstil
- Wenn „use strict“ verwendet wird, ist diese Methode nicht möglich

## DER ALERT-BEFEHL

```
<script>  
  "use strict";  
  let text = "mein erster Text";  
  alert(text);  
</script>
```

# LET VS VAR

- In einem einfachen Programm gibt es keinen Unterschied
- Funktionsweisen sehr ähnlich
- Üblich, dass eine Variable, die innerhalb einer `if`-Abfrage deklariert wird, nur im Inneren dieses Blocks gültig ist  
im übrigen Programm käme es zu einem Fehler wenn sie aufgerufen wird
- Variablen, die mit `let` deklariert sind, beachten diesen Grundsatz  
mit `var`, sind sie auch außerhalb verwendbar, deshalb sollte man darauf verzichten
- Verhalten innerhalb von Funktionen:
  - ▶ Geht der Interpreter die gesamte Funktion durch und realisiert alle Variablen-Deklarationen (sofern mit `var` ausgeführt).  
Danach beginnt Abarbeitung des übrigen Programmcodes.
  - ▶ Folge: möglich, die Variablen am Anfang der Funktion zu verwenden und sie erst später zu deklarieren => `let`  
unterstützt das nicht
- Fazit: ältere Bezeichnung `var` unterstützt einige Verhaltensweisen, die nicht mehr gängigen Programmiertechniken entsprechen. Bezeichner `let` erfüllt diese Ansprüche

# KONSTANTEN VERWENDEN

- Werte die in der gesamten Verwendung gleich bleiben
- Verwendung kann Performance eines Programms etwas verbessern
- Beispiele:
  - ▶ Bestimmung von Farben für die Beeinflussung des Designs mit JavaScript

```
<script>  
  "use strict";  
  const a = "mein erster Text";  
  alert(a);  
</script>
```

# DATENTYPEN ERMITTELN

- Mit `typeof` kann ausgegeben werden, um welchen Typ von Variable es sich handelt
- Befehl: `document.write` schreibt Inhalte direkt in das Hauptfenster

```
"use strict";
let ganzeZahl = 3;
let kommazahl = 2.34;
let buchstabe = "a";
let wort = 'hallo';
let wahrheitswert = true;
let ohneInitialisierung;

document.write("ganzeZahl: " + typeof ganzeZahl + "<br>");
document.write("kommazahl: " + typeof kommazahl + "<br>");
document.write("buchstabe: " + typeof buchstabe + "<br>");
document.write("wort: " + typeof wort + "<br>");
document.write("wahrheitswert: " + typeof wahrheitswert + "<br>");
document.write("ohneInitialisierung: " + typeof ohneInitialisierung + "<br>");
```

---

ganzeZahl: number  
kommazahl: number  
buchstabe: string  
wort: string  
wahrheitswert: boolean  
ohneInitialisierung: undefined

- Ganze Zahlen und Fließkommazahlen behandelt JS gleich als number
- Typ einer Variable kann sich in JS im Laufe des Programms ändern

# DATENTYPEN VERÄNDERN

- Eingabeaufforderung durch prompt-Befehl
- Ausgabe des Datentyp (string)
- Ausgabe und versuch einen String mit einer Zahl zu addieren, es kommt zu einer Konkatenaktion
- Ausgabe von Zeilenumbrüchen
- Umwandlung des Eingabewerts in eine Zahl
- Ausgabe und addition der Zahl mit 3 = richtiges Ergebnis

```
let eingabe = prompt("Gib eine Zahl ein:");  
document.write("Datentyp: " + typeof eingabe + "<br>");  
document.write(eingabe + 3);  
document.write("<br><br>");  
eingabe = Number(eingabe);  
document.write("Datentyp: " + typeof eingabe + "<br>");  
document.write(eingabe + 3);
```

Datentyp: string

53

Datentyp: number

8



# OPERATIONEN MIT VARIABLEN DURCHFÜHREN

```
let meineVariable = 3;
meineVariable = meineVariable * 2;
document.write(meineVariable + "<br>");
meineVariable = 3;
meineVariable *= 2;
document.write("Kurzschreibweise: " + meineVariable + "<br>");
meineVariable = 3;
meineVariable = meineVariable + 5;
document.write(meineVariable + "<br>");
meineVariable = 3;
meineVariable += 5;
document.write("Kurzschreibweise: " + meineVariable + "<br>");
```

6

Kurzschreibweise: 6

8

Kurzschreibweise: 8

Zeichen	Beschreibung
+	Addition
-	Subtraktion
/	Division
*	Multiplikation
**	Potenz
%	Modulo
+=	Erhöhung um den angegebenen Wert
-=	Erniedrigung um den angegebenen Wert
*=	Variable wird mit dem angegebenen Wert multipliziert
/=	Variable wird durch den angegebenen Wert geteilt
++	Erhöhung um 1
--	Erniedrigung um 1

# ÜBUNGSAUFGABE

- Schreibe ein Programm, das drei Variablen unterschiedlichen Typs erstellt. Daraufhin sollen deren Wert und deren Typ mit dem alert-Befehl ausgegeben werden
- Frage vom Anwender zwei Werte mit dem prompt-Befehl ab. Wandle die Werte dann in Zahlen um und multipliziere sie. Gib das Ergebnis mit `document.write()` aus.

## IF-ABFRAGE

```
// Beispiel 1
let bedingung = true;
if(bedingung) {
    alert("Die Bedingung trifft zu.");
}

// Beispiel 2
let eingabe = Number(prompt("Was ist das Ergebnis aus 3 + 2?"));
if(eingabe == 5) {
    document.write("Richtige Lösung!");
}
if(eingabe < 5) {
    document.write("Wert ist zu klein<br>");
} else if(eingabe > 5) {
    document.write("Wert ist zu groß<br>");
}
if(eingabe != 5) {
    document.write("Wert ist falsch!");
}
```

```
// Beispiel 3
"use strict";
let eingabe1 = Number(prompt("Ergebnis aus 2 + 3?"));
let eingabe2 = Number(prompt("Ergebnis aus 2 * 3?"));

if(eingabe1 == 5 || eingabe2 == 6) {
    document.write("Mindestens eine Antwort ist richtig!");
} else {
    document.write("Beide Antworten sind falsch");
}
```

# SWITCH-STATEMENT

```
"use strict";
let artikel = prompt("Welches Produkt suchst du?");
switch(artikel){
  case "Bohrmaschine":
  case "Bandschleifer":
  case "Kreissäge":
    alert("Preis: 34,99 €");
    break;

  case "Schraubenzieher":
    alert("Preis: 2,99 €");
    break;

  case "Hammer":
    alert("Preis: 6,99 €");
    break;

  default:
    alert("Produkt leider nicht gefunden");
}
```

# ÜBUNG

- Erstelle ein Programm, das zwei Konstanten enthält, in denen ein Nutzernamen und ein Passwort enthalten sind. Fordere dann den Besucher zur Eingabe der entsprechenden Daten auf. Gib eine passende Meldung aus – je nachdem, ob die eingegebenen Werte richtig oder falsch sind.
- Stelle dem Anwender 5 Rechenaufgaben. Überprüfe nach jeder Aufgabe, ob das Ergebnis richtig ist. Erhöhe in diesem Fall den Wert der Variablen punkte um 1. Gebe daraufhin je nach Punktestand eine passende Nachricht aus. Verwende dafür ein switch-Statement. Die Werte 0 und 1 sowie 2 und 3 sollen dabei jeweils zusammengefasst werden und zur Ausgabe der gleichen Nachricht führen.

# ARRAYS

```
// Variante 1  
let meinArray = new Array();
```

- Entspricht Vorgaben für objektorientierte Programmierung (später mehr)

- Arrays sind in JS Objekte mit besonderen Eigenschaften

```
// Variante 2  
let neuesArray = [];  
let neuesArray = ["Hund", "Katze", "Maus"];  
alert(neuesArray);
```

- Häufiger im Einsatz
- Kann gleich befüllt werden

- Elemente müssen nicht den gleichen Datentyp aufweisen

```
// Produkt, Preis, Lieferbar  
let neuesArray = ["Brot", 2.99, true];  
alert(neuesArray);
```

# ARRAYS

```
// Ausgabe bestimmte Stelle des Array
let neuesArray = ["Hund", "Katze", "Maus"];
alert(neuesArray[0]);

// Wert an bestimmter Stelle ändern
neuesArray[0] = "Papagei";

// Wert hinzufügen
neuesArray[3] = "Hund";
// besser um leere Felder oder Überschreibungen zu vermeiden
neuesArray[neuesArray.length] = "Hund";
```

- length gibt die Länge des Arrays zurück, da der Index bei 0 anfängt wird so am Ende ein neues Feld hinzugefügt

# MEHRDIMENSIONALE ARRAYS

```
let meinArray = [];  
meinArray[0] = ["Brot", 1.99, true];  
meinArray[1] = ["Krapfen", 0.99, true];  
meinArray[2] = ["Reindling", 6.99, false];  
meinArray[3] = ["Linzer Torte", 13.99, true];  
meinArray[4] = ["Windbeute", 0.80, false];  
// An bestimmte Stelle zugreifen  
document.write(meinArray[0][1] + "<br><br>");  
document.write(meinArray[2][2] + "<br><br>");  
document.write(meinArray[4][0] + "<br><br>");  
document.write(meinArray);
```

1.99

false

Windbeute

Brot,1.99,true,Krapfen,0.99,true,Reindling,6.99,false,Linzer Torte,13.99,true,Windbeute,0.8,false



# ÜBUNG

1. Schreibe ein Programm, das den Besucher nach seinem Vornamen, nach seinem Nachnamen und nach seinem Alter fragt. Erstelle ein Array und füge die entsprechenden Werte ein. Gebe dessen Inhalt anschließend auf der Seite aus.
2. Erzeuge ein Array, das drei leere Arrays enthält. Wiederhole die Eingabe der einzelnen Datensätze dann drei Mal und füge die entsprechenden Werte ein. Gebe dem Anwender daraufhin die Möglichkeit, einen bestimmten Wert aus dem Datensatz abzurufen. Dazu muss er zuerst die Nummer der Person eingeben und daraufhin den Index der gewünschten Information. (multidimensionales Array)
3. Erstelle ein Programm, das die gleiche Aufgabe erfüllt wie Aufgabe 1. Nutze jedoch eine Datenstruktur, die es erlaubt, die Werte über Schlüsselbegriffe abzurufen.



**ENDE**

QUELLE: JAVAScript  
PROGRAMMIEREN FÜR EINSTEIGER  
ISBN: 978-3-96645-016-4