Doing Web Apps

JavaScript 1

Autor: Rüdiger Marwein
Letzte Änderung: 2014-10-18
Version: 1.1

Dieses Dokument darf – mit Nennung des Autoren - frei vervielfältigt, verändert und weitergegeben werden.

Der Inhalt ist sorgfältig recherchiert, mit dem Dokument ist jedoch keinerlei Garantie auf Fehlerfreiheit gewährleistet.

Dieser Inhalt ist unter einem Creative Commons Namensnennung Lizenzvertrag lizenziert. Um die Lizenz anzusehen, gehen Sie bitte zu http://creativecommons.org/licenses/by/2.0/de/ oder schicken Sie einen Brief an Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

Inhaltsverzeichnis

1.Einleitung.....3
2.JavaScript in HTML.....3
3.Variablen.....3
4.Typen.....3
5.Zeichenketten.....4
6.Text ausgeben.....5
7.Rechnen.....5
8.Boolean.....5
9.Kontrollstrukturen.....7

1.Einleitung

JavaScript ist die meist mißverstandene Programmiersprache.

JavaScript ist nicht Java.

JavaScript gibt es unter anderem auch im Web-Browser. Aber auch in Flash, diversen Adobe-Produkten wie InDesign und sogar als serverseitige Scriptsprache mit Datenbank etc.

JavaScript verwendet keine Datentypen, aber JavaScript unterscheidet Datentypen.

JavaScript heisst auch ECMA Script.

JavaScript unterscheidet Groß- / Kleinschreibung.

Jede JavaScript Anweisung endet mit einem Semikolon;

Quellen zur Vertiefung:

http://www.w3schools.com/js/default.asp

http://www.amazon.de/dp/0596517742 (Buch: JavaScript – The good parts)

http://de.selfhtml.org/javascript/index.htm

http://openbook.galileocomputing.de/javascript/

http://www.codecademy.com/

http://jsbin.com/

http://addyosmani.com/blog/

2.JavaScript in HTML

In eine HTML-Seite, kann JavaScript auf 3 verschiedene Arten notiert werden.

Die lineare Verarbeitung ist bei der Platzierung zu berücksichtigen. Es wird dort ausgeführt, wo es steht.

Als externe Datei eingebunden (bevorzugt)

```
<script type="text/javascript" src="/js/myscript.js"></script>
```

In der HTML-Seite als Script-Block

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

Direkt an HTML-Tags

```
<div onclick="alert('Bei Klick ausgeführt');">Inhalt</div>
```

3.Variablen

Eine Variable ist ein Speicherort für einen Wert. Die Variable erhält einen frei wählbaren, sinnigen Namen. Der Wert kann nachträglich verändert werden.

Variablen-Deklarationen beginnen mit „var“. Es können eine oder mehrere Variablen auf einmal deklariert werden. Es wird empfohlen, alle verwendeten Variablen am Anfang des

Scripts zusammen zu deklarieren.

```
// einzeln
var name ="Peter";
// mehrere
var name ="Peter",
    alter=24,
    note=2;
```

Variablen, die außerhalb von Funktionen oder Objekten erstellt werden sind „global“ und sind somit überall verfügbar.Variablen, die innerhalb einer Funktion erstellt werden, sind nur lokal in dieser Funktion oder diesem Objekt verfügbar.

4.Typen

JavaScript kennt keine Datentypen. Es entscheidet selbst von welchem Typ eine Variable ist. Der zugewiesene Inhalt gibt Ausschlag.

```
var positiv = true; // boolean, Wahrheitswert
var name = "Peter"; // string, Zeichenkette, Text
var num = 32;       // Integer, Ganzzahl
var zahl = 2.333;   // Float, Fließkommazahl, Kommazahl
var liste = [];     // Array, Reihung, Liste
var data = {};      // Objekt
```

Ein Array ist tatsächlich auch ein Objekt in JavaScript, allerdings mit einer Eigenschaft „count“.

Bei Vergleichen wandelt JavaScript die Typen so um, dass sie vergleichbar werden. Es wird immer zum primitiveren Typ konvertiert.

```
"Peter" == 12 // false, falsch
"12" == 12    // true, wahr, Zeichenkette wird zu 12
"Peter" == 0  // true, wahr, Zeichenkette wird zu 0
"" == false   // true, wahr, leere Zeichenkette wird zu false
[] == {}      // false, falsch
{} == ""      // false, falsch
[] == ""      // true, Leeres Array wird zu Leer-String
```

Es wird empfohlen möglichst mit 3 Gleichheitszeichen zu vergleichen, da dann die Typumwandlung nicht vorgenommen wird. Das ist schneller und zuverlässiger.

```
"Peter" === 12 // false, falsch
"12" === 12    // false, falsch
"Peter" === 0  // false, falsch
"" === false   // false, falsch
[] === ""      // false, falsch
12 === 12      // true, wahr
```

5.Zeichenketten

Zeichenketten können in einfachen Anführungszeichen oder in Gänsefüßchen stehen.

Der Unterschied ist, dass Steuerzeichen innerhalb von Gänsefüßchen erlaubt sind.

Steuerzeichen:

```
\n   // Zeilenumbruch (Unix/Linux)
\r   // Wagenrücklauf (wie bei einer Schreibmaschine) (Mac)
```

```
\r\n // Zeilenumbruch mit Wagenrücklauf (DOS/Windows)
\t   // Tabulator, Einrückung
```

Zeichenketten können miteinander verknüpft werden mit dem Pluszeichen.

```
var text = "Peter hat" + " heute " + "Bauchweh";
```

Es können auch Variablen mit Zeichenketten verknüpft werden.

```
var loecher = 4;
var text = "Peter hat " + loecher + " Löcher im Zahn";
```

JavaScript akzeptiert umgebrochene Zeichenkette nicht bei der Zuweisung.

Fehler:

```
var text = "Liebes Tagebuch,
heute ist nichts passiert.";
```

Richtig:

```
var text = "Liebes Tagebuch,\nheute ist nichts passiert.";
```

Richtig auch:

```
var text = "Liebes Tagebuch,\n" +
           "heute ist nichts passiert.";
```

6.Text ausgeben

Zu Kontrollzwecken, ist es nützlich, den aktuellen Wert von Variablen auszugeben.

Ausgabe in einem Popup-Fenster:

```
var variable = "Mein Text";
alert( variable );
```

Ausgabe im HTML-Dokument:

```
var variable = "Mein Text";
document.write( variable );
```

Wichtig: Ist die HTML-Seite bereits fertig geladen und man verwendet document.write(), wird der Seiteninhalt geleert.

7.Benutzereingabe entgegennehmen

Im täglichen Leben wird Ihnen die folgende Funktion vermutlich nicht begegnen, für die Übungsaufgaben wird sie jedoch benötigt, um Benutzereingaben zu ermöglichen.

```
var name = prompt("Geben Sie Ihren Namen ein:", "");
```

Die Eingabe des Benutzers wird hier in die Variable „name“ gespeichert und kann weiterverarbeitet werden.

Durch Angabe des zweiten Parameters kann das Eingabefeld vorbelegt werden.

8.Benutzer-Entscheidung entgegennehmen

Die Funktion confirm() wird bspw. eingesetzt um vor Durchführung einer destruktiven Aktion den Benutzer um Zustimmung zu bitten.

var antwort = confirm("Soll die Webseite wirklich gelöscht werden?");

Klickt der Benutzer auf „Ja“, wird „true“ als Wert zurückgegeben, klickt er auf „Nein“, wird „false“ zurückgegeben. Man kann auf Basis der Benutzereingabe das Programm entsprechend reagieren lassen (ggf doch nicht löschen).

9.Rechnen

Es sei für jede Zeile in jeder Rechenart

var a = 5, b = 10;

+	+=
	 ++
		Addition, Plus	a = a + 1;
			 a += 1;
			 a++;
			 ++a;
			 a = a + b;
				a = 6
				 a = 6
				 a = 5 // danach 6 (Post-Inkrement)
				 a = 6
				 a = 15

-	-=
	 --
		Subtraktion, Minus	a = a - 1;
			 a -= 1;
			 a--;
			 --a;
			 a = a - b;
				a = 4
				 a = 4
				 a = 5 // danach 4 (Post-Dekrement)
				 a = 4
				 a = -5

*	*=	Multiplikation, „Mal“	a = a * b; a *= b;	a = 50 a = 50
/	/=	Division, „Durch“	a = b / a; b /= a;	a = 2 a = 2

%	%=	Modulo, „Rest“	b = b % a;
			 b = b % 3;
			 b %= 5
				b = 0 // 10/5 geht auf. Rest 0
				 b = 1 // 10/3 ergibt 3, Rest 2
				 b = 0 // 10/5 geht auf, Rest 0

Es gilt Punkt vor Strich. Modulo geht noch vor Punkt.

10.Boolean

Der Bool'sche Datentyp ist der wichtigste zur Programmablaufsteuerung.

Das Ergebnis eines Vegleichs ist „true“ oder „false“, je nachdem ob der Vergleich zutrifft.

Vergleiche werden in Schleifen und Bedingungen verwendet, um das Programm zu steuern.

Folgende Abfragen ergeben ein Ergebnis vom Typ Boolean (Wahrheitswert).

Es sei für jede Zeile in jeder Rechenart

var a = 5, b = 10;

==	Gleichheit mit Typumwandlung	5 == "5" false == "" 5 == a a == "Hallo"	true true true false

===	Gleichheit ohne Typumwandlung	5 === "5" false === "" 5 === 5 5 === a	false false true true
>=	Größer oder gleich	5 >= a	true
<=	Kleiner oder gleich	5 <= a	true
>	Größer als	5 > a	false
<	Kleiner als	5 < a	false

Wahrheitswerte können miteinander verknüpft werden, um neue Wahrheitswerte zu erhalten.

„Damit das Programm weiterlaufen kann, muss a>100 sein und b<=0“

Die sogenannten „Logischen Verknüpfungen“ sind programmatisch wie folgt:

Es sei für jede Zeile in jeder Rechenart

var a = 5, b = 10;

&&	UND. Wenn beide Seiten true sind	a ==5 && b == 10 a ==5 && b == 9	true false
||	ODER. Wenn eine der beiden Seiten true ist oder beide true sind	a ==5 || b == 10 a ==4 || b == 10 a ==4 || b == 9	true true false

Die logischen Verknüpfungen lassen sich kombinieren.

UND geht vor ODER. Ist es anders gewünscht, kann mit Klammern gearbeitet werden.

```
var a = 21, b = 5;
alert( a > 4 && a < 20 || b == 5 );
```

Die Klammerung ist implizit

```
(a > 4 && a < 20) || b == 5
```

Zuerst wird UND abgearbeitet, es ergibt

```
false || b == 5
```

Da bei ODER nur eine Seite wahr sein muss, ist der Ausdruck true.

Ist die Abarbeitung anders gewünscht, kann die implizite Klammerung durch eine explizite Klammerung aufgehoben werden.

```
a > 4 && (a < 20 || b == 5)
```

Die Klammerung ist nun explizit.

Es wird vor der Verknüpfung das ODER abgearbeitet, ergibt

```
a > 4 && (false || true)
```

ergibt

```
a > 4 && true
```

Da „a“ größer 4 ist und somit beide Werte „true“ sind, ist das Ergebnis „true“

Logische Verknüpfungen können beliebig lang sein.

11.Kontrollstrukturen

Mit Bedingungen wird das Programm kontrolliert über einen bestimmten Pfad ausgeführt, der dadurch bestimmt wird, ob gewisse Bedingungen zutreffen.

Eine Prüfung der Form wenn-dann-sonst, wird wie folgt notiert

```
var typ = prompt("Fahrzeuge","Fahrzeug-Typ angeben:");
var autos = 0, sonstige = 0;
if( typ === "Auto" ) {
  autos++;
} else {
  sonstige++;
}
alert("Autos: " + autos + "\nSonstiges: " + sonstige);
```

In Worten: „Wenn es sich um ein Auto handelt, zähle ein Auto hinzu, ansonsten zähle ein sonstiges Fahrzeug hinzu.“

Es lassen sich größere Strukturen bilden, indem innerhalb des if- oder else-Blocks weitere Prüfungen vorgenommen werden. Dies lässt sich beliebig fortführen, erhöht jedoch die Komplexität des Programms.

```
if( typ === "Auto" ) {
  if(schrott === false) {
    autos++;
  }
} else {
  if(typ === "Quad") {
    autos++;
  } else {
    sonstige++;
  }
}
```

Für einfache wenn-dann-sonst-Abfragen, kann eine verkürzte, sogenannte ternäre (dreiteilige) Schreibweise verwendet werden.

Aufbau: `Wenn-Bedingung ? dann : Sonst`

```
typ == "Auto" ? autos++ :  sonstiges++;
autos += typ == "Auto" ? 1 : 0;
```