aus: H. Schröder, R. Steyer: JavaScript 1.8, Herdt 2014

3.7 Datentypen

Es ist bei der Programmierung wichtig zu wissen, von was für einem Datentyp eine Variable oder ein Literal ist. Der Datentyp legt fest …

- ✔ wie viel Speicherplatz für eine Variable oder ein Literal reserviert wird,
- ✔ welche Werte erlaubt sind,
- ✔ wie klein bzw. groß die Zahlen minimal bzw. maximal sein können,.
- ✔ welche Operationen mit den Variablen durchgeführt werden können.

Lose Typisierung und Casting

JavaScript verfolgt bei der Zuweisung von Werten an Variablen das Konzept der sogenannten losen Typisierung (engl.: loose typing). Dies bedeutet, dass Sie zwar die Namen von Variablen festlegen, dabei jedoch die Datentypen nicht deklarieren können. Einer Variablen kann nur durch eine Wertzuweisung ein Datentyp zugewiesen werden. Die Typzuweisung einer Variablen erfolgt automatisch.

Grundsätzlich kann eine Variable innerhalb eines Skripts von unterschiedlichem Datentyp sein. Dies bedeutet, dass eine Variable in JavaScript z. B. zuerst eine Zahl sein kann, im nächsten Schritt aber zu einer Zeichenkette werden kann. Dieses Verhalten ist im Rahmen der losen Typisierung möglich. Ein explizites Typumwandeln, das sogenannte **Casting**, ist im Gegensatz zu vielen streng typisierten Sprachen wie Java nicht notwendig.

Lose Typisierung macht den Einstieg in JavaScript für Anfänger einfach, ist aber auch fehleranfällig. Lose Typisierung erhöht die Fehlerwahrscheinlichkeit im Quellcode und macht die Wartung unter Umständen aufwändig. Verwenden Sie deshalb für eine Variable **nur einen Datentyp**. In der neusten ECMAScript-Version 5 steht ein Modus zur Verfügung, bei dem der lockere Umgang mit Datentypen und Variablen unterbunden werden kann ("use strict"). Bis diese Version jedoch in der Praxis verwendet werden kann, werden wohl noch einige Jahre vergehen. Erst mit der Etablierung von JavaScript 2.0 wird der Modus in JavaScript zuverlässig zu verwenden sein.

Die verfügbaren Datentypen in JavaScript

JavaScript besitzt vier Haupt- und zwei Sonderdatentypen (`undefined` und `null`):

Datentyp	Beschreibung
`boolean`	Werte vom Typ `boolean` (auch boolesche Werte genannt – nach dem Mathematiker George Boole) sind Wahrheitswerte, die nur einen der beiden Werte `true` oder `false` (wahr oder falsch) annehmen. Dieser Datentyp wird im Rahmen von Vergleichen genutzt.
`number`	Dieser universelle numerische Datentyp kann in JavaScript sowohl Ganzzahl- als auch Gleitkommawerte enthalten. Dies ist recht ungewöhnlich, denn in den meisten Programmier-sprachen werden diese Typen getrennt.
`object`	Der allgemeine Datentyp kann einen Wert eines beliebigen Typs enthalten. Er wird für das Speichern von Objekten verwendet.
`string`	Dieser Datentyp steht für eine Zeichenkette und kann eine Reihe von alphanumerischen Zeichen enthalten. Maximal sind in JavaScript 256 Zeichen in einer Zeichenkette erlaubt. Bei Bedarf können aber mehrere Zeichenketten mit dem Operator `+` verknüpft werden. Wichtig ist, dass Sie mit dem Operator auch Werte anderer Datentypen mit dem String verbinden können. Das Resultat ist vom Typ `string`. Diese Möglichkeit ist bei Ausgaben sehr nützlich.

Datentyp	Beschreibung
`undefined`	Der Sondertyp beschreibt eine nicht definierte Situation. Eine Variable besitzt so lange diesen Wert, bis ihr nach dem Anlegen explizit ein Wert zugewiesen worden ist.
`null`	Dieser Sondertyp entspricht der Situation, wenn ein Objekt noch keinen Wert hat, und entspricht "keiner Bedeutung" oder null. Damit unterscheidet sich der Wert von einer leeren Zeichenkette oder dem Literal `0`. Der Sondertyp `null` kann beispielsweise zurückgegeben werden, wenn in einem Dialogfenster eine *Abbrechen*-Schaltfläche betätigt wird.

Ganze Zahlen

Ganze Zahlen besitzen keine Nachkommastellen und können negative sowie positive Werte und den Wert null annehmen. In JavaScript können ganze Zahlen in einer von drei möglichen Darstellungen eingesetzt werden:

- **Dezimaldarstellung**: Dies ist die gebräuchlichste Darstellung von Zahlen. Die Zahl wird im Dezimalsystem (Basis 10) dargestellt, d. h., dass die Ziffern 0..9 verwendet werden.
- **Hexadezimaldarstellung**: Die Zahlen werden im Hexadezimalsystem (Basis 16) dargestellt. Die verwendbaren Ziffern sind 0..9, A..F (A=10, B=11 ... F=15). Hexadezimale Zahlen werden durch ein einleitendes 0x (bzw. 0X) gekennzeichnet.
- Die **Oktaldarstellung** (zur Basis 8) ist seit JavaScript 1.5 nicht mehr Bestandteil von JavaScript und wird nur noch aus Gründen der Rückwärtskompatibilität unterstützt. Oktalzahlen werden durch eine vorangestellte 0 (Null) gekennzeichnet.

Dezimal	Hexadezimal
0	0x0
1	0x1
8	0x8
15	0xF
16	0x10
42	0x2A

Gleitkommazahlen

Gleitkommazahlen können Nachkommastellen besitzen. Das Kennzeichen ist entweder ein Dezimalpunkt, ein Exponent oder beides. Bei einer Fixkommazahl ist die Position des Kommas vorgegeben, bei einer Gleitkommazahl kann das Komma "gleiten".

In JavaScript wird die amerikanische Notation, also ein Punkt , verwendet, um die Dezimalstellen einer Zahl kenntlich zu machen.

Die größte und kleinste speicherbare Zahl können Sie z. B. durch die folgenden Anweisungen ausgeben:

Fixkommazahl	Gleitkommazahl
1000.0	1.0E3
2000.4	20004.0E-1
-4000.0	-4e3
-0.004	-4e-3

Beispiel

```
document.write("Max: ", Number.MAX_VALUE);
document.write("Min: ", Number.MIN_VALUE);
```

Beachten Sie, dass beim Überschreiten des Wertebereichs der ganzen Zahlen automatisch eine Konvertierung in Gleitkommazahlen erfolgt. Gleitkommazahlen unterliegen immer Rundungsfehlern. Beachten Sie dies bei Berechnungen mit großen Zahlen.

```
k = 12345678901234567890;
document.write(k);  // liefert 12345678901234567000, da die Zahl zuerst in eine
                    // Gleitkommazahl konvertiert und danach gerundet wird !
```