6. Schleifen

Angenommen wir wollen die Zahlen von 1 bis 100 einzeln ausgeben lassen. Im Moment würden wir 100 Zeilen Code schreiben in der Form: alert(1); alert(2); alert(3); … ; alert(100);

Das ist sehr aufwendig und geht viel kürzer! Für solche Fälle hat sich der Programmierer das Konzept einer Schleife überlegt. Man wiederholt also unter bestimmten Bedingungen einen Block an Code mehrmals.

Es gibt zwei verschiedene Arten von Schleifen, und zwar die *For-Schleife* und die *While-Schleife*.

Die *For-Schleife* führt den Code eine bestimmte Anzahl x an Malen durch, also sie *zählt von einem Startwert durch bis zu einem Endwert* und hört dann auf. Mit unseren Zahlen von 1 bis 100, die wir ausgeben wollen, wäre der Startwert also zum Beispiel 1 und der Endwert 100, da wir 100 Mal ein alert ausführen möchten. Das heißt: Die Schleife fängt an zu zählen und zählt immer hoch, wenn der Code in der Schleife erfolgreich ausgeführt wurde und das bis wir bei dem Endwert angekommen sind.

Der Code für die Zahlen von 1 bis 100 würde so aussehen:

Beispiel 6.1

```
for (var i=1;i<=100;i++){
  alert(i);
}
```

Achtung! Bevor ihr das ausprobieren wollt, macht aus dem alert ein document.write! Sonst müsst ihr 100 Pop-Ups wegklicken!

Wie ist dieser Code aufgebaut?

Um eine For-Schleife einzuleiten benötigen wir das Schlüsselwort for. Im Anschluss erwartet das Programm zwei Klammern, in die 3 Operationen gehören.

1. Wir legen uns die *Zähl-Variable* fest und ihren Startwert. Wir erstellen uns also explizit für diese Schleife eine Variable, damit wir durchzählen können.
2. Wir legen die *Bedingung* fest, die, *solange sie erfüllt ist*, die *Schleife weiter durchlaufen lässt*. Damit verbunden ist dann auch immer unser Endwert, also im Beispiel 7.1 muss unsere Zähl-Variable bis 100 zählen, damit sie aufhören kann.
3. Wir setzen die *Rechnung zum Zählen* fest. Wir könnten ja zum Beispiel auch rückwärts zählen oder in 2er Schritten. Da kommt dann eine Zuweisung der Form i = i +1 , i = i -1 oder i = i + 2 hin. Man kann das normale Hochzählen in 1er Schritten auch abkürzen mit i++ oder das normale Runterzählen mit i--.

Den Code-Block, den wir wiederholen möchten, schreiben wir wie immer zwischen zwei geschweifte Klammern.

Warum schreiben wir unsere Variable i in das alert? Ganz einfach, weil genau diese Variable immer den Wert hat, den wir aktuell ausgeben möchten. Würden wir 100-mal den Text "Hallo!" ausgeben wollen, müsste entsprechend auch "Hallo!" in das alert. Also das kann je nach Programm und was das Ergebnis sein soll variieren, was in dem alert steht.

Die *While-Schleife* führt den Code aus solange die vorher festgesetzte Bedingung erfüllt ist.
So eine While-Schleife erstellen wir, indem wir das Schlüsselwort while verwenden und dann in runde Klammern unsere Bedingung schreiben. Danach kommen wieder die üblichen geschweiften Klammern für unseren Code-Block der Schleife.

Schauen wir uns auch hier mal ein paar Beispiele an:

Beispiel 6.2

```
var macheWeiter = true;

while (macheWeiter == true){
  alert("Das ist eine While-Schleife!");
}

alert("Fertig!");
```

In diesem Beispiel führen wir den Code in der Schleife aus solange auf der Variablen macheWeiter der Wert true gespeichert ist. Da gibt es aber ein kleines Problem: Da wir den Wert der Variablen nicht in der Schleife ändern, wird die Bedingung für immer erfüllt sein. Das heißt, dass diese While-Schleife niemals aufhören und unser alert mit "Fertig!" niemals ausgegeben wird.

Also was merken wir uns? Wir müssen die Faktoren unserer Bedingung innerhalb der Schleife immer wieder abändern, sodass die Schleife überhaupt enden kann.

Beispiel 6.3

```
var zahl = parseInt(prompt("Geben Sie eine Zahl ein:",""));

while (zahl%2==1){
  zahl = parseInt(prompt("Geben Sie eine andere Zahl ein:"));
}

alert("Juhu! Endlich eine gerade Zahl!");
```

Hier fragen wir den Benutzer nach einer Zahl und solange diese nicht gerade ist, fragen wir nach einer neuen.

Kommen wir nun zu unserem kleinen Problem aus dem Kapitel Arrays aus dem Anhang. Wir haben ein Array der Länge 4, auf dem Zahlen abgespeichert wurden. Diese wollen wir nun addieren. Statt immer den Arraynamen und dann die Position hinschreiben zu müssen, also zahlen[1]+…+zahlen[4], können wir unser Array auch in einer Schleife durchgehen.

Das Programm würde folgendermaßen aussehen:

```
var zahlen = [1,2,3,4];
var summe = 0;

for (var i = 0;i<zahlen.length;i++){
  summe = summe + zahlen[i];
}

alert(summe);
```

Was passiert da genau? Wir gehen mit einer For-Schleife unser Array einmal komplett durch, indem wir bei 0 anfangen zu zählen (erste Position eines Arrays) und bis zu der Länge des Arrays gehen. Wir gehen also so jeden einzelnen Eintrag durch. Und jedes Mal addieren wir den aktuellen Eintrag, also den Eintrag an der Position i, auf unsere Variable summe.

So sieht das ganze schon viel ordentlicher aus und ist bei langen Arrays auch deutlich weniger Schreibarbeit.