

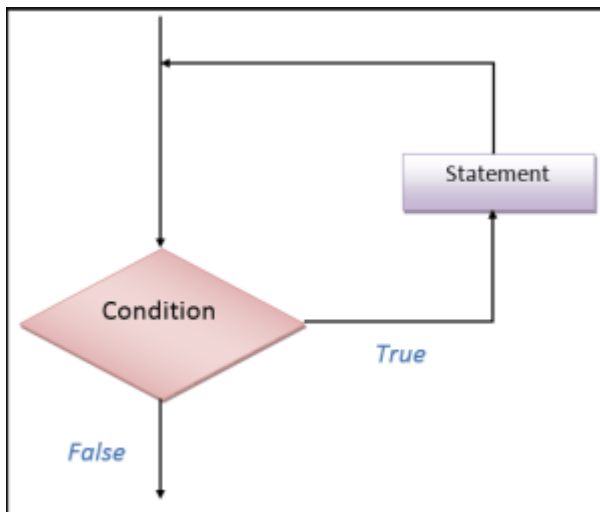
LU05a - Loops/Schleifen Übersicht

Einleitung

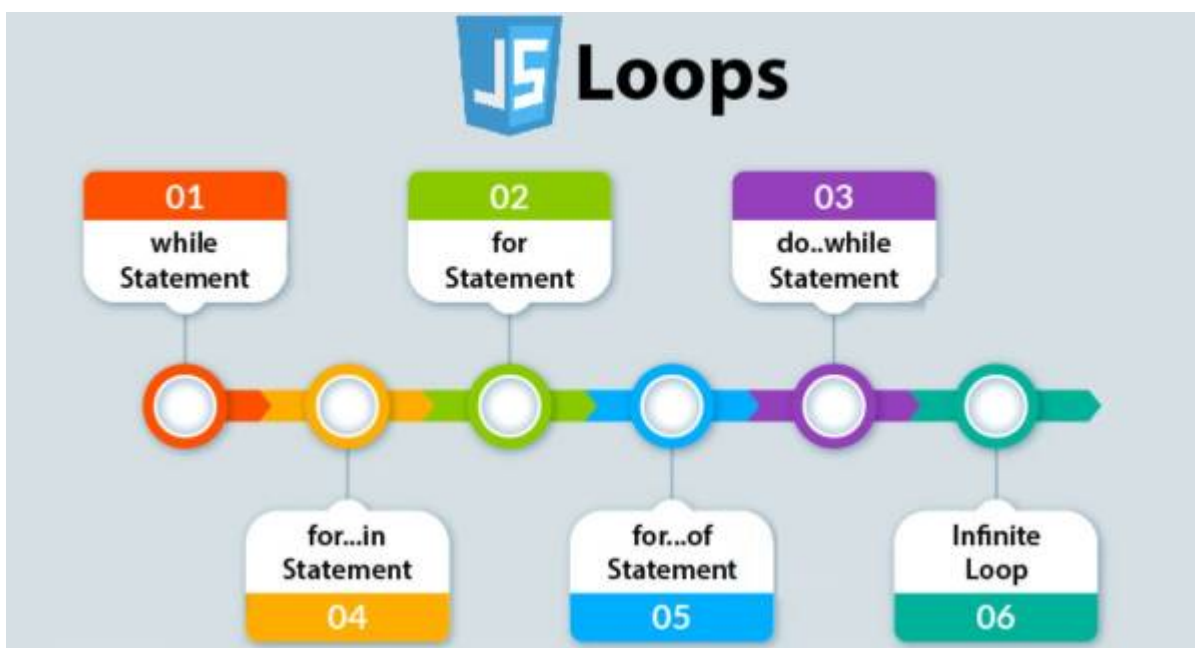
Stellen Sie sich vor, Sie sollen 100 Zahlen auf die Konsole schreiben. Das ginge mit:

```
console.log(1);  
console.log(2);  
console.log(3);  
// ... bis 100
```

Sehr unpraktisch. Stattdessen: Schleifen - sie wiederholen Anweisungen, solange eine bestimmte Bedingung erfüllt ist.



In fast jeder Programmiersprache, so auch in JavaScript, gibt es die nachfolgende Typen:



Wir haben nicht Zeit für alle Loop-Arten. Innerhalb des Moduls 288 werden wir uns daher auf die

nachfolgenden vier Loops konzentrieren.

1. for-Schleife: Zählerschleife, kopfgesteuert
2. while-Schleife: Bedingung, kopfgesteuert
3. for ... of: Solange es hat, kopfgesteuert
4. for ... in: Bedingung, kopfgesteuert

Die Schleifenarten

for-Schleife

Die klassische for-Schleife ist eine kopfgesteuerte Schleife, d.h. die Schleifenbedingung wird im Kopf der Schleife abgefragt. Sie ist ideal, wenn wir genau wissen, wie oft etwas wiederholt werden soll. Z.B. wir wollen unsere Produkte jeweils 10 Stück auf einer Seite präsentieren.

Beispiel

```
for (let i = 0; i < 5; i++) {  
  console.log("Zahl:", i);  
}  
// 1. Startwert (let i = 0)  
// 2. Bedingung (i < 5)  
// 3. Anweisung ausführen  
// 4. Schritt (i++)
```

Ausgabe

```
Zahl: 0  
Zahl: 1  
Zahl: 2  
Zahl: 3  
Zahl: 4
```

while-Schleife

Die *while-Schleife* ist eine kopfgesteuerte Schleife, d.h. die Schleifenbedingung wird im Kopf der Schleife abgefragt. Der Code, der sich innerhalb der Schleife befindet, solange wiederholt, solange eine Bedingung **wahr** ist.

Beispiel

```
let i = 0;  
while (i < 5) {  
  console.log("while:", i);  
  i++;  
}
```

Ausgabe

```
while: 0
while: 1
while: 2
while: 3
while: 4
```

Wichtig: Vergisst man `i++`, hängt man in einer Endlosschleife, weil die Bedingung im Kopf der Schleife nie erfüllt wird.

do...while-Schleife

Die *do...while*-Schleife ist ähnlich wie die *while*-Schleife, mit dem Unterschied, dass die Schleifenbedingung im Fuss der Schleife abgefragt wird. Folglich wird der Code innerhalb der Schleife mindestens einmal ausgeführt, auch wenn die Bedingung sofort falsch ist.

Beispiel

```
let i = 5;
do {
  console.log("do-while:", i);
  i++;
} while (i < 5);
```

Ausgabe

```
do-while: 5 // (obwohl die Bedingung von Anfang an falsch war).
```

for...of-Schleife

Dieser Schleifentyp ist ebenfalls kopfgesteuert. Sie ist ideal, um über Arrays (oder Strings) zu iterieren (durchzugehen).

```
let fruits = ["Apfel", "Banane", "Kirsche"];
for (let fruit of fruits) { // es wird ein interner Schleifenzähler
  console.log(fruit);
}
```

Ausgabe

```
Apfel, Banane, Kirsche
```

for...in-Schleife

Diese Schleife ist ebenfalls kopfgesteuert. Sie läuft über die Eigenschaften eines Objekts (oder Indizes

eines Arrays).

Beispiel

```
let person = {name: "Anna", age: 25, city: "Berlin"};
for (let key in person) {
  console.log(key, ":", person[key]);
}
```

Ausgabe

```
name : Anna
age  : 25
city : Berlin
```

Kontrollanweisungen in Loops

Bei sehr lange Listen kann es passieren, dass man die aktuell laufende Schleife vorzeitig abbrechen möchte. Hier gibt es zwei Möglichkeiten:

- **break** → beendet die Schleife komplett
- **continue** → überspringt den aktuellen Durchlauf und springt zur nächsten Runde

Beispiel

```
for (let i = 0; i < 5; i++) {
  if (i === 2) continue; // überspringt 2
  if (i === 4) break;    // stoppt bei 4
  console.log(i);
}
```

Ausgabe

```
0, 1, 3
```

Wann ist welche Schleife passend?

| Typ | Wann nutzen? |
|----------|--|
| for | Wenn die Anzahl der Wiederholungen bekannt sind. |
| while | Wenn nur eine Bedingung überprüfen werden soll. |
| for...of | Wenn über ein Array durchgelaufen werden soll. |
| for...in | Wenn Objekte durchgegangen werden sollen. |

Achtung: Endlosschleifen sind der Klassiker für eingefrorene Programme.

Lernvideos

[Schleifen im JavaScript](#)

Zusatzmaterial

- [W3School- JavaScript Loops](#)
- [SelfHTML - Schleifen](#)



Volkan Demir

From:

<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:

<https://wiki.bzz.ch/de/modul/m288/learningunits/lu05/01?rev=1758604765>

Last update: **2025/09/23 07:19**

