# L1\_3.7 Wiederholungen in PHP

Die PHP-Anweisungen innerhalb eines Programmes werden in der Regel sequenziell abgearbeitet. Kontrollstrukturen beeinflussen die Reihenfolge der Ausführung. Verzweigungen (Alternativen) ermöglichen das bedingte Ausführen von Programmabschnitten. Schleifen (Wiederholungen) dienen dazu, einen bestimmten Programmabschnitt mehrfach zu wiederholen.

**while-Schleife**

Ist die Anzahl der Durchläufe nicht im Vorhinein bekannt, dann eignet sich der Einsatz einer while-Schleife. Die Schleife wird so lange durchlaufen, wie eine gesetzte Bedingung den booleschen Wert „wahr“ liefert. Vor jedem neuen Schleifendurchlauf wird die Gültigkeit der Bedingung aufs Neue überprüft. Sobald die Bedingung das erste Mal den Wert „false“ liefert, wird die Schleife beendet. In der Regel werden die in der Bedingung genutzten Parameter im Schleifenrumpf verändert und damit der Abbruch der Schleife gesteuert.

|  |  |
| --- | --- |
| Syntax:  **while(** *Bedingung* **) {**  *Anweisungen*;  **}** | Beispiel:  $a = 1;  while( *$a < 10* ) {  *echo '$a ';*  $a = $a + 1;  } |

Vor dem Schleifenrumpf wird die Variable $a mit dem Wert 1 initialisiert. Solange $a kleiner 10 ist, wird der Schleifenrumpf – durch die geschweiften Klammern eingeschlossen – ausgeführt. Im Schleifenrumpf wird zuerst der Inhalt der Variablen $a ausgegeben und die Variable anschließend um eins erhöht. Nach Ausführung der letzten Anweisung des Schleifenrumpfs wird erneut die Bedingung geprüft. Das Ergebnis dieser Prüfung bestimmt, ob die Schleife ein weiteres Mal ausgeführt wird. $a vergrößert sich im Beispiel mit jedem Schleifendurchlauf um eins. Hat die Variable $a den Wert 10 angenommen, liefert die Bedingung den Wert „false“ und die Schleife ist beendet. Das Beispiel erzeugt die Ausgabe:



Achtung: Der Rumpf einer while-Schleife wird nicht zwangsweise ausgeführt. Liefert die Auswertung der Bedingung bereits vor dem ersten Durchlauf den Wert „false“, so wird die Schleife kein einziges Mal ausgeführt.

**do-while-Schleife**

Die do-while-Schleife ähnelt der while-Schleife. Im Gegensatz zur while-Schleife wird die Bedingung erst nach dem Schleifenrumpf platziert.

|  |  |
| --- | --- |
| Syntax:  **do {**  *Anweisungen*;  **} while(** *Bedingung* **);** | Beispiel:  $a = 1;  do {  *echo '$a ';*  $a = $a + 1;  } while( *$a < 10* ); |

Vor dem Schleifenrumpf wird die Variable $a mit dem Wert 1 initialisiert. Da die Prüfung des Wertes der Bedingung erst am Schleifenende geschieht, wird der Schleifenrumpf zumindest einmal ausgeführt. Im Schleifenrumpf wird zuerst der Inhalt der Variablen $a ausgegeben und die Variable anschließend um eins erhöht. Nach Ausführung der letzten Anweisung des Schleifenrumpfs wird nun die Bedingung geprüft. Das Ergebnis dieser Prüfung bestimmt, ob die Schleife ein weiteres Mal ausgeführt wird. $a vergrößert sich im Beispiel mit jedem Schleifendurchlauf um eins. Hat die Variable $a den Wert 10 angenommen, liefert die Bedingung den Wert „false“ und die Schleife ist beendet. Das Beispiel erzeugt die Ausgabe:



Achtung: Der Rumpf einer do-while-Schleife wird zumindest einmal ausgeführt, da die Bedingung erstmalig nach dem Durchlauf des Schleifenrumpfes geprüft wird.

**for-Schleife**

Ist die Anzahl der Schleifendurchläufe bekannt, biete sich der Einsatz einer for-Schleife – auch Zählerschleife genannt – an. Alle Parameter einer for-Schleife werden, anders als bei den bisher bekannten Schleifen, im Schleifenkopf definiert. Dazu gehören die Initialisierung, die Bedingung und die Anweisung zur Manipulation der Laufvariablen. Dem Schlüsselwort **for** folgen in runden Klammern drei durch Semikolon getrennte Komponenten, in welchen in der Regel die gleiche Variable auftaucht. Wir sprechen hier auch von der Laufvariablen – in obigem Beispiel ist dies die Variable $a. Die erste der drei Komponenten setzt $a auf einen Startwert. Die dritte Komponente dient dazu, die Schleifenvariable kontinuierlich zu ändern. Sehr häufig wird der Startwert bei jedem Durchlauf um eins erhöht. Die mittlere Komponente (Bedingung) bestimmt, wann die Schleife beendet wird. Die Schleife wird solange ausgeführt, solange die Bedingung erfüllt ist.

|  |  |
| --- | --- |
| Syntax:  **for(Initialisierung; Bedingung; Schrittweite) {**  *Anweisungen*;  **}** | Beispiel:  for($a = 1; $a<10; $a=$a+1) {  *echo '$a ';*  } |

Beispiel:

4

2

1

**for** ($a = 1; $a < 10; $a = $a + 1) **{**

3

…

**}**

5

Zu Beginn der Schleife wird der Anfangswert gesetzt (1). Dies geschieht im Leben einer Schleife nur einmal. Anschließend wird die Bedingung geprüft (2). Ist diese erfüllt, dann werden alle Anweisungen zwischen den geschweiften Klammern ausgeführt (3). Am Ende der Schleife, also an der schließenden geschweiften Klammer springt die Programmausführung zur Schrittweite (4). Der Wert des Schleifenzählers wird verändert. Nun wird erneut geprüft, ob die Bedingung auch mit dem geänderten Schleifenzähler noch erfüllt ist. Ist dies der Fall, dann folgt ein weiterer Schleifendurchlauf. Dieser Vorgang wiederholt sich solange, wie die Bedingung erfüllt ist. Erst dann, wenn die Bedingung nicht mehr erfüllt ist, ist die Schleife beendet und das Programm wird mit den Anweisungen, die nach der Schleife stehen fortgeführt (5).

Das Beispiel erzeugt die Ausgabe:



**Zusammenfassung**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **while-Schleife** | **do-while-Schleife** | **for-Schleife** |
| **Syntax** | **while(*Bedingung*) {**  *zu wiederholende Anweisungen;*  **}** | **do {**  *zu wiederholende Anweisungen;*  **}while(*Bedingung*);** | **for(Startwert; Bedingung; Schrittweite){**  *zu wiederholende Anweisungen;*  ***}*** |
| **Funktionsweise** | Der durch die geschweiften Klammern aufgespannte Anweisungsblock wird solange ausgeführt, solange die Bedingung erfüllt ist, also den Wert „wahr“ liefert.  Ist die Bedingung bereits vor der ersten Prüfung nicht erfüllt, wird der Schleifenrumpf kein einziges Mal ausgeführt. | Der durch die geschweiften Klammern aufgespannte Anweisungsblock wird einmal ausgeführt und danach solange, solange die Bedingung erfüllt ist, also den Wert „wahr“ liefert. | Zuerst wird der Startwert des Schleifenzählers gesetzt. Anschließend wird der durch die geschweiften Klammern aufgespannte Schleifenblock solange ausgeführt, solange die Bedingung erfüllt ist, also den Wert „wahr“ liefert. Am Ende eines jeden Schleifendurchlaufs wird der Schleifenzähler gemäß der Schrittweite geändert. |
| **Beispiel** | $a=1;  while($a<10) {  echo "$a ";  $a = $a+1;  } | $a=1;  do {  echo "$a ";  $a = $a+1;  } while($a<10) | for($a=1; $a<10; $a=$a+1) {  echo "$durchlauf ";  } |