# L1\_3.8 Funktionen in PHP

In einer Funktion werden PHP-Anweisungen gekapselt. Funktionen sind mit einem Namen versehen, der dazu dient, die Funktion aufzurufen. Jeder Aufruf einer Funktion hat zur Folge, dass die in der aufgerufenen Funktion enthaltenen Anweisungen abgearbeitet werden. Funktionen kennen Sie bereits aus Python oder Javascript. Der Einsatz von Funktionen bietet sich immer dann an, wenn eine spezielle Funktionalität immer wieder benötigt wird. Das Auslagern von Programmteilen in einer Funktion erhöht in der Regel auch die Lesbarkeit des Programms.

Die PHP-Umgebung bringt bereits eine Vielzahl solcher Funktionen mit. Einige davon haben wir bereits kennengelernt: mt\_rand(…), sqrt(…), pow(…).

Wir können nun eigene Funktionen erstellen. Man spricht dann von „benutzerdefinierten Funktionen“.

**Funktionen ohne Parameter**

Die Definition einer Funktion in PHP ähnelt sehr stark der Definition einer Funktion in Javascript:

|  |  |
| --- | --- |
| Syntax:  **function** *funktionsname***() {**  *Anweisungen*  **}** | Beispiel:  function hallo() {  echo "Hallo";  } |

Die Definition der Funktion beginnt mit dem Schlüsselwort function. Danach folgt der Name der Funktion, für die dieselben Namensregeln gelten wie für Variablen. Anschließend folgen runde Klammern in denen, anders wie im obigen Beispiel, sogenannte Funktionsparameter stehen können. Am Schluss folgt ein Block von Anweisungen, die immer in geschweiften Klammern stehen. Parameterlose Funktionen benötigen keine weiteren Informationen, um ihren Job zu erledigen.

Der Aufruf einer Funktion erfolgt durch den Funktionsname:

|  |  |
| --- | --- |
| Syntax:  *funktionsname***();** | Beispiel:  hallo(); |

Der Funktionsaufruf besteht aus dem Namen der Funktion gefolgt von runden Klammern. Die Funktion hallo() kann beliebig oft aufgerufen werden. Bei jedem Aufruf springt das Programm zur gleichnamigen Funktionsdefinition und führt die dort enthaltenen Anweisungen aus.

**Funktionen mit Parameter**

Bisher haben wir Funktionen kennengelernt, die bei jedem Aufruf immer dasselbe Ergebnis liefert. Nun folgen Funktionen, denen man Informationen übermittelt, die dann verarbeitet werden und bei jedem Aufruf ein anderes Ergebnis liefern. Diese Informationen, die die Funktionen erhalten nennt man Parameter. Die Anzahl der Parameter ist nicht beschränkt. In der Praxis versucht man die Anzahl möglichst klein zu halten. Die Definition einer Funktion mir Parametern sieht in PHP wie folgt aus:

Syntax:

**function** *funktionsname***(*parameter1, parameter2,…*) {**

*Anweisungen*

**}**

Beispiel:

function ausgeben($vorname, $nachname) {

echo "$vorname, $nachname";

}

Hier wird eine Funktion mit dem Namen ausgeben(…) definiert. Sie besitzt die zwei Parameter $vorname und $nachname, die mit einem Komma voneinander getrennt sind. Beide Parameter werden genutzt um den kompletten Name auszugeben, wobei der Vorname vom Zunamen mit einem Komma getrennt dargestellt wird.

Der Aufruf einer Funktion mit Parametern erfolgt durch den Funktionsname:

|  |  |
| --- | --- |
| Syntax:  *funktionsname***($wert1, $wert2);** | Beispiel:  ausgeben("Max", "Mustermann"); |

Der Funktionsaufruf besteht aus dem Namen der Funktion gefolgt von runden Klammern, in welchen sich die Werte für die Parameter befinden. Der erste Wert wird mit dem ersten Parameter gematcht (blau), der zweite Wert mit dem zweiten Parameter (grün). Der Aufruf der Funktion ausgeben(…) hat nun zur Folge, dass die Zeichenkette Max, Mustermann ausgegeben wird.

**Funktionen mit Rückgabewerte**

Der Benutzer kann nicht nur Parameter an eine Funktion übergeben, sondern auch Ergebnisse von der Funktion erhalten. Diese Ergebnisse heißen Rückgabewerte. Rückgabewerte können sowohl Zahlen, Zeichenketten oder auch Wahrheitswerte sein. Man kann Rückgabewerte sofort ausgeben oder auch speichern, um sie später zu verwenden. Funktionen mit Rückgabewerte können über Parameter verfügen, müssen dies aber nicht.

Syntax:

**function** *funktionsname***(*parameter1, parameter2,…*) {**

*Anweisungen*

***return …;***

**}**

Beispiel:

function quadrat($zahl) {

return $zahl \* $zahl;

}

Eine Funktion mit Rückgabewert beinhaltet immer das Schlüsselwort return. Der Wert des hinter diesem Schlüsselwort befindlichen Ausdrucks bestimmt den Rückgabewert der Funktion.

Der Aufruf einer Funktion mit Parametern erfolgt durch den Funktionsname:

|  |  |
| --- | --- |
| Syntax:  *funktionsname***($wert1, $wert2);** | Beispiel:  echo quadrat(5);  $a = quadrat(5); echo quadrat(5);  if(quadrat(5) > 20) {  echo "Quadrat größer 20!";  } |

Der Funktionsaufruf erfolgt wie bisher. Da der Funktionsaufruf einen Wert zurückliefert, kann dieser Wert gleich verarbeitet werden. Im ersten Beispiel wird dieser direkt ausgegeben, im zweiten Beispiel wird dieser zuerst in einer Variablen gespeichert und anschließend ausgegeben. Der Rückgabewert einer Funktion kann auch direkt in einer Alternativen verwendet werden, wie das dritte Beispiel zeigt.

Ergänzende Anmerkungen:

* Funktionen müssen nicht an dem Ort definiert werden, an der sie das erste Mal aufgerufen werden. Funktionsdefinitionen können auch an den Anfang oder an das Ende eines PHP-Skripts verschoben werden, um die Programmstruktur überschaubar zu machen.
* Jede Funktion muss innerhalb eines Skriptes einen eindeutigen Namen haben.
* Bei Funktionen mit Parametern müssen Sie die Werte für diese Parameter beim Aufruf mitgeben. Sie können tatsächlich mehr Werte angeben als Parameter vorhanden sind, überzählige werden einfach ignoriert. Die Angabe von zu wenigen Parametern quittiert PHP allerdings mit einer Warnung.
* Innerhalb einer Funktion können Sie nicht auf Variablen zugreifen, die außerhalb der Funktion definiert sind. Eine Funktion hat einen eigenen, separaten Gültigkeitsbereich. Externe Variablen können über das Schlüsselwort global sichtbar gemacht werden, dies ist aber ein eher schlechtes Vorgehen. Benötigt eine Funktion Werte, die außerhalb der Funktion vorliegen, dann sollten diese Werte in Form von Parameter der Funktion bekannt gemacht werden.