**Lektion 9: Functions (Funktionen)**

In den vorangegangenen Lektionen haben Sie gelernt, wie man die PHP-eigenen Functions wie z.B. [date()](http://php.net/manual/en/function.date.php) und [array()](http://php.net/manual/en/function.array.php) benutzt. In dieser Lektion werden Sie lernen, mit [function](http://php.net/manual/en/functions.user-defined.php) Ihre eigenen Funktionalitäten zu erstellen.

**Was ist eine Function (Funktion)?**

Eine Function verarbeitet *Eingaben (inputs)* und liefert eine *Ausgabe (output)*. Dies kann hilfreich sein, wenn Sie z.B. eine große Menge Daten haben, die sie mit immer wiederkehrenden Berechnungen oder Routinen verarbeiten müssen.

Eine Function hat die folgende Syntax:

 Function Name(Parameterliste) {

 Anweisung

 }

So z.B. können wir eine sehr einfache Function erstellen, die 1 zu einer anderen Zahl hinzuaddiert. Der Code dafür könnte wie folgt aussehen:

 function AddOne($x) {

 $x = $x + 1;

 echo $x;

 }

Unsere Function heißt **AddOne** (AddiereEins) und muss mit einer Zahl als Parameter aufgerufen werden - z.B. 34 ...

 echo AddOne(34);

... welches dann (welche Überraschung!) 35 ausgeben wird.

Das obige Beispiel verarbeitet eine Zahl, aber Functions arbeiten mit Text, Daten oder vielem andern mehr. Sie können auch Functions erstellen die mehrere verschiedene Parameter verarbeiten In dieser Lektion werden Sie verschiedene Beispiele von Functions kennenlernen.

**Beispiel 1: Function mit mehreren Parametern**

Wie bereits oben erwähnt, können Sie einfach Functions programmieren, die mit mehreren Parametern aufgerufen werden. Im nächsten Beispiel werden wir eine Function erstellen, die mit drei Zahlen als Parameter aufgerufen wird und die Summe der drei Zahlen ausgibt:

 <html>

 <head>

 <title>Functions (Funktionen)</title>

 </head>

 <body>

 <?php

 function AddAll($number1,$number2,$number3) {

 $plus = $number1 + $number2 + $number3;

 return $plus;

 }

 echo "123 + 654 + 9 equals " . AddAll(123,654,9);

 ?>

 </body>

 </html>

[Beispiel anzeigen](http://de.html.net/tutorials/php/lesson9_ex1.php)

OK! Aber das war jetzt fast zu einfach. Aber Sinn und Zweck war ja nur, zu zeigen, dass eine Function mit mehreren Parametern aufgerufen werden kann.

**Beispiel 2: Englisches Datum und Zeit**

Lassen Sie uns versuchen, eine ein wenig komplexere Function zu programmieren, Eine Function, die mit einem Datum und einer Zeit aufgerufen wird und dies im folgenden Format ausgibt: **Wednesday, 15 February, 2012, 10:00:00 AM**

 <html>

 <head>

 <title>Functions (Funktionen)</title>

 </head>

 <body>

 <?php

 function EnglishDateTime($date) {

 // Array mit den englischen Namen der Wochentage

 $arrDay = array("Monday","Tuesday","Wednesday","Thursday","Friday","Saturday","Sunday");

 // Array mit den englischen Namen der Monate

 $arrMonth = array("","January","February","March","April","May","June",

 "July","August","September","October","November","December");

 // Das Datum wird zusammengesetzt

 $EnglishDateTime = $arrDay[(date("w",$date))] . ", " . date("d",$date);

 $EnglishDateTime = $EnglishDateTime . " " . $arrMonth[date("m",$date)] . " " . date("Y",$date);

 $EnglishDateTime = $EnglishDateTime . ", " . date("H",$date) . ":" . date("i",$date);

 return $EnglishDateTime;

 }

 // Function testen

 echo EnglishDateTime(time());

 ?>

 </body>

 </html>

[Beispiel anzeigen](http://de.html.net/tutorials/php/lesson9_ex2.php)

Haben Sie bemerkt, dass '$arrMonth' und '$EnglishDateTime' über mehrere Zeilen hinweg aufgebaut wurden? Dies haben wir gemacht, damit Leser mit einer kleineren Bildschirmauflösung das Beispiel besser lesen können. Dies hat keinerlei Einfluss auf den Code.

Die obige Function funktioniert auf allen Webservern, egal welche Sprache eingestellt ist. Das heißt, Sie können eine solche Function auch nutzen, wenn Ihre Webseite auf deutschen Servern liegt, Sie aber ein englisches Datum benötigen.

An diesem Punkt werden wir nicht tiefer in der Thema 'Functions' eintauchen, aber jetzt wissen Sie ein wenig, wie das mit den Functions funktioniert.