**Lektion 8: Arrays (Felder)**

In dieser Lektion beschäftigen wir uns mit Arrays (Feldern), wie sie verwendet werden und was man damit tun kann.

Arrays zu verstehen mag am Anfang ein wenig schwierig sein. Aber versuchen Sie es trotzdem ... wir haben probiert, es so einfach wie möglich zu machen.

**Was ist ein Array?**

Ein Array ist eine Menge von indizierten Elementen, wobei jedes eine eigene, eindeutige Identifikationsnummer hat.

Klingt verwirrend? Eigentlich ist es aber gar nicht so kompliziert.

Stellen Sie sich eine Liste von Wörtern vor, die mit Kommas voneinander getrennt sind. Dies nennt man kommaseparierte Liste und sie könnte z.B. so aussehen:

 Äpfel, Birnen, Bananen, Orangen, Zitronen

Dann versuchen Sie sich vorzustellen, die Liste an jedem Komma zu trennen. Geben Sie dann jedem Abschnitt eine eindeutige Identifikationsnummer. Etwa so:



Was Sie nun vorliegen haben ist ein Array. Wir können dieses Array jetzt z.B. "fruits" (engl. für Früchte) nennen. Das Konzept dahinter ist, dass wir uns jedes Element des Arrays ausgeben lassen können, indem wir es mit seiner Nummer ansprechen. Etwa so:

fruits(0) = Äpfel
fruits(1) = Birnen
fruits(2) = Bananen
fruits(3) = Orangen
fruits(4) = Zitronen

Dies ist die Idee hinter Arrays. Lassen Sie sie uns in der Praxis anwenden.

**Wie benutzt man einen Array?**

Wir werden mit dem Früchtebeispiel weitermachen. Schritt für Schritt werden wir die Liste zu einem richtigen Array machen. Zuerst bekommt die Fruchtliste eine Variable verpasst:

 <?php

 $fruitlist = "Äpfel, Birnen, Bananen, Orangen, Zitronen";

 ?>

Als nächstes benutzen wird die PHP-Function [explode](http://php.net/manual/en/function.explode.php), um die Liste an jedem Komma zu teilen:

 <?php

 $fruitlist = "Äpfel, Birnen, Bananen, Orangen, Zitronen";

 $arrFruits = explode(",", $fruitlist);

 ?>

Voila! "$arrFruits" ist jetzt ein Array!

Beachten Sie, dass wir die Function [explode](http://php.net/manual/en/function.explode.php) mit zwei Argumenten aufgerufen haben:

1. die Liste, die geteilt werden soll
2. und das Trennzeichen - also das Zeichen an dem getrennt werden soll (in unserem Falle das Komma) - in Anführungsstrichen: ",".

Hier benutzen wir das Komma als Trennzeichen, Sie können aber jedes andere Zeichen oder sogar Wort (bzw. Zeichenfolge) als Trenn'zeichen' benutzen.

Lassen Sie uns das Skript noch kommentieren und in eine PHP-Seite einfügen:

 <html>

 <head>

 <title>Arrays</title>

 </head>

 <body>

 <?php

 // kommaseparierte Liste

 $fruitlist = "Äpfel, Birnen, Bananen, Orangen, Zitronen";

 // Erstelle einen Array durch teilen der Liste (mit dem Komma als Trennzeichen)

 $arrFruits = explode(",", $fruitlist);

 // Ausgabe der Werte unseres Arrays

 echo "<p>Die Fruchtliste:</p>";

 echo "<ul>";

 echo "<li>" . $arrFruits[0] . "</li>";

 echo "<li>" . $arrFruits[1] . "</li>";

 echo "<li>" . $arrFruits[2] . "</li>";

 echo "<li>" . $arrFruits[3] . "</li>";

 echo "<li>" . $arrFruits[4] . "</li>";

 echo "</ul>";

 ?>

 </body>

 </html>

[Beispiel anzeigen](http://de.html.net/tutorials/php/lesson8_ex1.php)

Dieses Beispiel war ziemlich einfach und es mag sein, dass es ein wenig schwierig ist, daran die Vorteile des Arbeitens mit Arrays zu sehen. Aber warten Sie ... Arrays können für viele nützliche Sachen genutzt werden.

**Arrays mit Schleifen durchlaufen**

In [Lektion 5](http://de.html.net/tutorials/php/lesson5.php) hatten Sie Schleifen kennengelernt. Jetzt werden wir uns anschauen, wie Sie einen Array in einer Schleife durchlaufen lassen können.

Wenn Sie wissen, wieviele Elemente ein Array enthält, ist es kein Problem die Schleife zu definieren. Sie starten einfach bei 0 und lassen die Schleife, bis die Anzahl der Elemente erreicht ist, durchlaufen. Im Beispiel mit den Früchten, würden wir z.B. so vorgehen:

 <html>

 <head>

 <title>Arrays</title>

 </head>

 <body>

 <?php

 // kommaseparierte Liste

 $fruitlist = "apples, pears, bananas, oranges, lemons";

 // Erstelle einen Array durch teilen der Liste (mit dem Komma als Trennzeichen)

 $arrFruits = explode (",", $fruitlist);

 echo "<p>Die Fruchtliste:</p>";

 echo "<ul>";

 // Schleifendurchlauf durch den Array $arrFruits

 for ($x=0; $x<=4; $x++) {

 echo "<li>" . $arrFruits[$x] . "</li>";

 }

 echo "</ul>";

 ?>

 </body>

 </html>

[Beispiel anzeigen](http://de.html.net/tutorials/php/lesson8_ex2.php)

Wie Sie sehen, wurde die Variable *$x* (die sich in den Schleifendurchläufen von 0 auf 4 erhöht) benutzt um die Elemente des Arrays abzufragen.

**Wie kann man die Größe eines Arrays herausfinden**

Was aber, wenn wir eine weitere Frucht zu unserer Liste hinzufügen? Dann enthielte unser Array ein Element mehr - dieses erhielte die Identifikationsnummer **5**. Sehen Sie das Problem? Dann müssen wir die Schleife ändern, damit sie von 0 bis 5 durchlaufen wird oder es würden nicht alle Elemente eingeschlossen.

Wäre es nicht schöner, wenn man automatisch herausfinden könnte, wieviele Elemente ein Array enthält?

Die ist genau das, was die PHP-Function [foreach](http://php.net/manual/en/control-structures.foreach.php) macht. Jetzt können wir eine Schleife programmieren, die, ohne auf die Anzahl der Elemente achten zu müssen, funktioniert:

 <?php

 foreach ($arrFruits as $x) {

 echo $x;

 }

 ?>

Diese Schleife wird funktionieren, egal wie viele (oder wie wenige) Elemente der Arrays enthält.

**Ein weiteres Beispiel**

Nachfolgend ist ein anderes Beispiel, wie Sie einen Array nutzen können, um den Namen eines Monats auszugeben:

 <html>

 <head>

 <title>Array<title>

 </head>

 <body>

 // Erstellt einen Array mit den Namen der Monate.

 // Beachten Sie das Komma vor dem Januar - weil es keinen Monat mit der Nummer 0 gibt

 $arrMonths = array("","Januar","Februar","März","April","Mai","Juni","Juli","August","September","Oktober","November","Dezember");

 // Ruft den Array mit der Nummer des aktuellen Monats auf - und gibt es dem Client aus

 echo $arrMonths[date("m")];

 ?>

 </body>

 </html>

[Beispiel anzeigen](http://de.html.net/tutorials/php/lesson8_ex3.php)

Bitte beachten Sie, dass wir dieses Mal die PHP-Function [array](http://php.net/manual/en/language.types.array.php) anstatt von [explode](http://php.net/manual/en/function.explode.php) genutzt haben, um einen Array zu erstellen.

OK. Genug über Arrays! In der nächsten Lektion werden Sie lernen, wie Sie eigenen Functions schreiben können.