Felder (Arrays) in PHP

Felder werden in PHP genau wie in C/C++ verwendet. Ein Feld ist eine Menge von Variablen mit gleichem Namen, die über diesen Namen und einen fortlaufenden Index angesprochen werden, welcher bei 0 beginnt. Die Deklaration wie in C/C++ entfällt. Der Datentyp entscheidet sich bei der Zuweisung und es können sogar unterschiedliche Typen im gleichen Feld untergebracht werden. Weiterhin braucht ein Feld nicht dimensioniert zu werden. Das Anhängen weiterer Feldelemente vergrößert automatisch das Feld.

Alle folgenden Beispiele erzeugen Felder:

Bsp 1 int-Feld	Bsp 2 int-Feld	Bsp 3 int-Feld	Bsp 4 untsch. Typen

`$feldA[0]=1;`
`$feldA[1]=2;`
`$feldA[2]=3;`
`...`
	`$feldB[]=1;`
	`$feldB[]=2;`
	`$feldB[]=3;`
		`$feldC=array(1,2,3);`	`$feldD[0]=1;`
			`$feldD[1]="Hallo";`
			`$feldD[2]=3.14;`

Wichtige Feld-Funktion mit dem Beispiel-Array: `$feld=array(1,2,3);`

`count`	Ermittelt die Anzahl der Array Elemente `for($i=0;$i<count($feld);$i++)…`

Felder lassen sich wie in C/C++ gewohnt mit Schleifen bearbeiten:

```
for($i=0;$i<count($feld);$i++)
  echo $feld[$i],"<br>";              //Ausgabe
```

Feldelemente in Variablen schreiben

Mit der Funktion `list`, können Feldelemente in einzelne Variablen geschrieben werden

```
  $wert=array(100,"Hallo",0.815);
  list($var1,$var2,$var3)=$wert;
  echo $var1,$var2,$var3;
```

Assoziative Arrays

In PHP können Feldelemente nicht nur über Indizes angesprochen werden, sondern sie können stattdessen beliebige Bezeichnungen erhalten. Damit ist es möglich, Datenstrukturen in Feldern abzubilden.

```
$Artikel["Nr"]=101;
$Artikel["Bezeich"]="Tastatur";
$Artikel["Preis"]=14.95;
echo $Artikel["Bezeich"];
```

- Mit den Funktionen **next** und **current**, kann in Feldern geblättert werden.

  ```
  next($Artikel); next($Artikel);  //Blättern
  echo current($Artikel);  //Ausgabe 3. Element
  ```

- Die Bezeichnungen (die sog. Schlüssel) eines assoziativen Feldes lassen sich mit der Funktion **array_keys** ermitteln und in ein Feld schreiben. Die “Schlüssel“ stehen danach als Werte in einem neuen Feld mit Indizes.

  ```
  $schluessel=array_keys($Artikel);
  for($i=0;$i<count($schluessel);$i++)
    echo $schluessel[$i],"<br>";
  echo $Artikel[$schluessel[1]]  //Zugriff auf 2. Element
  ```

- Die Funktion **extract** erzeugt aus einen assoziativen Array normale Variablen, welche wie die Schlüssel heißen.

```
extract($Artikel);
echo $Nr," + ",$Bezeichnung," + ",$Preis;
```

- Die Funktion **print_r** gibt Informationen über Belegung und Aufbau von Arrays aus. Sie ist vor allem bei der Analyse komplexer Arrays hilfreich.
  `print_r($Artikel);`
  Ausgabe: `Array ( [Nr] => 101 [Bezeich] => Tastatur [Preis] => 14.95 )`

Die foreach-Schleife

Im Gegensatz zur normalen for-Schleife, muss bei der foreach-Schleife die Anzahl der Arrayelemente nicht bekannt sein. Im Schleifenkopf wird eine Variable vereinbart, die nacheinander die Werte der Arrayelemente annimmt, s. nachfolgendes Beispiel:

```
$Adresse=array("Name","Strasse","PLZ","Ort");
foreach($Adresse as $Element)
{
  echo $Element,"<br>";
}
```

Funktionen in PHP

Funktionen lassen sich in PHP ähnlich wie in C/C++ definieren. Es geschieht mit dem Schlüsselwort **`function`**. Die Typangaben für Rückgabewert und formale Parameter entfallen.

```
function brutto($betrag,$mwst)     //Beispiel
{
  $betrag=doubleval($betrag);//sicherheitshalber in double wandeln
  $mwst=doubleval($mwst);
  return $betrag*$mwst/100; //Rückgabe mit return
}
echo "Ergebnis: ",brutto(10.0,16);//Funktionsaufruf
```

- Die Reihenfolge von Definition und Aufruf ist egal.
- In der Funktion deklarierte Variablen sind lokale Variablen.

Referenzparameter

Wird vor einen formalen Parameter ein &-Zeichen gesetzt, wird er als Referenz übergeben und Änderungen wirken sich auf den aktuellen Parameter aus.

```
function ref(&$param)      //Beispiel
{
  $param++;
}
$wert=123;
echo "vorher $wert <br>";
ref($wert);
echo "nachher $wert <br>";
```

Felder als Funktionsparameter

Felder können wie „einfache“ Variablen als Parameter übergeben werden. Dabei sind Wert- und Referenzparameter möglich. Felder können auch Rückgabewert sein.

Felder als Referenzparameter

Beispiel 1	Beispiel2

`function test(&$feld)`
`{`
`  for($i=0;$i<count($feld);$i++)`
`  $feld[$i]++;`
`}`
	`function test2(&$feld)`
	`{`
	`  foreach($feld as &$el)`
	`  $el--;`
	`}`

Eigene Funktionssammlungen in Bibliotheken

Funktionen können in Bibliotheksdateien gesammelt werden. So können Sie in mehreren Scripten verwendet werden.

Bibliotheksdateien lassen sich mit der Funktion `require` einbinden. `require` sollte am Anfang eines Scripts stehen.

Beispiel:

Datei `func.inc` (im gleichen Verzeichnis wie das Script)	`func.php`

`<?php`
`function test1()`
`{`
`  echo "Kuckuck";`
`}`
`?>`
	`<?php`
	`require("func.inc");`
	`test1();`
	`?>`