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 eine 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;`

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

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

Wichtige Feld-Funktionen

All folgenden Beispiele beziehen sich auf 2 Felder:

```
$fa=array(1,2,3);  $fb=array(4,5,6);
```

Funktion	Bedeutung und Beispiel
`array_merge`	verbindet zwei Arrays zu einem Array `$fc=array_merge($fa,$fb);`
`array_pop`	Löscht letztes Element. Array danach um ein Element kürzer `array_pop($fa);`
`array_push`	fügt weitere Elemente an ein Array an. Gleicher Effekt kann mit weiteren Zuweisungen erreicht werden. `array_push($fa,10);`
`array_shift`	Löscht erstes Element. Die anderen rücken vor. `array_shift($fa);`
`array_unshift`	Fügt weitere Elemente am Anfang des Arrays an. `array_unshift($fa,12);`
`count`	Ermittelt die Anzahl der Array Elemente `for($i=0;$i<count($fa);$i++)`…
`in_array`	liefert true, wenn ein Wert in einem Feld vorhanden ist. `if(in_array(7,$fa))echo "es ist drin";`
`range`	erzeugt ein Array mit nummerischen Werten zwischen 2 Grenzen `$fc=range(1,10);`
`sort`	Sortiert ein Array `sort($fa);`
`shuffle`	"mischt ein Array" `srand(time(NULL));shuffle($fa);`
`unset`	löscht ein Array `unset($fa);`

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 Funktion **var_dump** gibt dazu noch die Datentypen an.
  `var_dump($Artikel);`
  Ausgabe: `array(3){ ["Nr"]=> int(101) ["Bezeich"]=> string(8) "Tastatur" ["Preis"]=> float(14.95) }`

Mehr- und multidimensionale Arrays

Da Feldelementen nicht nur Werte in unterschiedlichen Datentypen, sondern auch Felder zuweisen werden können, können sehr komplexe Datenstrukturen aufgebaut werden. Zur Analyse empfiehlt sich im Zweifelsfall die Funktion `print_r.`

```
$zD=array(array(11,12),array(21,22));
$mD[“anzZeilen“]=2;$mD[“anzSpalten“]=2;$mD[“Data“]=$zD;
```

```
Array (  [“anzZeilen“] => 2            //Ausgabe mit print_r, übersichtlich formatiert
         [“anzSpalten“]=> 2
         [“Data“] => Array (  [0] => Array (  [0] => 11
                                              [1] => 12 )
                              [1] => Array (  [0] => 21
                                              [1] => 22 ) ) )
```