1

VARIABLEN, AUSDRÜCKE & OPERATOREN

Dieses Kapitel zeigt, wie Daten, die sich bei jeder Anforderung einer PHP-Seite ändern können, in Variablen gespeichert werden und wie Ausdrücke und Operatoren mit Variablenwerten umgehen.

Variablen stellen mit einem Namen einen Wert dar, der sich bei jedem Aufruf einer PHP-Seite ändern kann:

- Der **Name** beschreibt die Art der Daten, die die Variable enthält.
- Der **Wert** ist der Wert, den die Variable beim Aufruf der Seite enthalten soll.

Sobald die Seite fertig abgearbeitet und der HTML-Code an den Browser zurückgesendet wurde, vergisst der PHP-Interpreter die Variable wieder (und sie kann dann beim nächsten Aufruf der Seite einen anderen Wert enthalten).

PHP unterscheidet zwischen verschiedenen Arten von Werten, die Sie in einer Variablen speichern können (z. B. Text und Zahlen); diese werden als **Datentypen** bezeichnet:

- Ein Text wird als **Zeichenkette (String)** bezeichnet.
- Eine ganze Zahl heißt **Integer.**
- Ein Dezimalbruch entspricht dem Datentyp **Float.**
- Ein Wert, der wahr oder falsch (`true` oder `false`) sein kann, ist ein **boolescher Wert (Boolean)**.
- Eine Reihe zusammenhängender Namen und Werte lässt sich in einem **Array** speichern.

Wenn Sie sich erst einmal mit Variablen vertraut gemacht haben, werden Sie sehen, dass Ausdrücke aus mehreren Werten einen einzigen Wert bilden können. So lässt sich z. B. Text aus zwei Variablen zu einem Satz zusammenfügen oder eine in einer Variablen gespeicherte Zahl mit einer Zahl aus einer anderen Variablen multiplizieren.

Ausdrücke verwenden **Operatoren**, um einen einzelnen Wert zu erzeugen. Der Operator + wird beispielsweise zur Addition zweier Werte verwendet, der Operator – zur Subtraktion eines Werts von einem anderen.


The Candy Store
MULTI-BUY OFFER
Hello Ivy,
Buy 5 packs of Chocolate for $20
(usual price $25)!
Save $5

VARIABLEN

Variablen speichern Daten, die sich bei jedem Aufruf einer PHP-Seite verändern (variieren) können. Sie stellen also einen veränderlichen Wert mit einem feststehenden Namen dar.

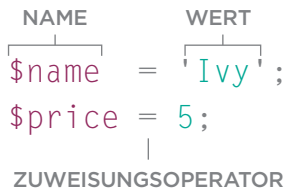
Um eine Variable zu erstellen und einen Wert darin zu speichern, benötigen Sie:

- einen **Variablennamen**, der mit einem Dollarzeichen beginnen muss, gefolgt von einem oder mehreren Wörtern, die die Art der von der Variablen zu speichernden Informationen beschreiben.
- ein **Gleichheitszeichen**, das auch als Zuweisungsoperator bezeichnet wird, weil es dem Variablennamen einen Wert zuweist.
- den **Wert**, der in der Variablen abgelegt werden soll.

Wenn die Variable Text enthält, muss dieser in Anführungszeichen gesetzt werden. Sie können einfache oder doppelte Anführungszeichen verwenden, aber sie müssen zusammenpassen. (Beginnen Sie zum Beispiel nicht mit einem einfachen Anführungszeichen und schließen dann mit einem doppelten Anführungszeichen ab.)

Wenn die Variable eine Zahl oder einen booleschen Wert (`true` oder `false`) enthält, wird dieser nicht in Anführungszeichen gesetzt.

Die Erstellung einer Variablen bezeichnen Programmierer als **Variablendeklaration**. Nach der Deklaration erfolgt dann üblicherweise die **Zuweisung** eines Werts.



Sobald eine Variable deklariert und ihr ein Wert zugewiesen wurde, können Sie den Variablennamen im PHP-Code überall dort nutzen, wo Sie den aktuellen Wert der Variablen benötigen.

Wenn der PHP-Interpreter auf einen Variablennamen stößt, ersetzt er den Namen durch den darin enthaltenen Wert. Im Folgenden zeigen wir den in der oben gezeigten Variablen `$name` gespeicherten Wert mit dem Befehl `echo` an.

VARIABLEN ERSTELLEN UND DARAUF ZUGREIFEN

PHP section_a/c01/variables.php

```
<?php
① $name  = 'Ivy';
② $price = 5;
?>
<!DOCTYPE html>
<html>
  <head>
    <title>Variables</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <h1>The Candy Store</h1>
③   <h2>Welcome <?php echo $name; ?></h2>
    <p>The cost of your candy is
④      $<?php echo $price; ?> per pack.</p>
  </body>
</html>
```

ERGEBNIS

In diesem Kapitel weisen wir den Variablen direkt im PHP-Code Werte zu. In künftigen Kapiteln stammen die Werte, die in die Variablen geschrieben werden, aus von den Besuchern abgeschickten HTML-Formularen, aus URL-Daten und aus Datenbanken.

In diesem Beispiel werden am Anfang der Seite zwei Variablen erstellt und mit Werten gefüllt:

1. `$name` enthält den Namen des aktuellen Website-Besuchers. Da es sich um Text handelt, steht dieser in Anführungszeichen.

2. `$price` enthält den Preis für eine einzelne Süßigkeit. Da es sich um eine Zahl handelt, steht der Wert nicht in Anführungszeichen.

Als Nächstes sehen Sie den HTML-Code, der an den Browser zurückgesendet wird. Darin wird:

3. der Benutzername mit dem Befehl `echo` in die Seite geschrieben

4. der Süßwarenpreis in die Seite geschrieben

Probieren Sie es: Ändern Sie in Schritt 1 den Wert der Variablen `$name` so ab, dass er Ihren Namen enthält. Speichern Sie die Datei und aktualisieren Sie dann die Seite in Ihrem Browser. Sie sehen daraufhin Ihren Namen.

Probieren Sie es: Ändern Sie in Schritt 2 den Preis auf 2. Speichern Sie die Datei und aktualisieren Sie dann die Seite. Nun wird der neue Preis angezeigt.

ZUR BENENNUNG VON VARIABLEN

Ein Variablename sollte die in der Variablen gespeicherten Daten beschreiben. Halten Sie folgende Regeln ein, um einen Variablennamen zu erstellen.

1

Beginnen Sie mit einem Dollarzeichen (`$`).

✓ `$greeting`

✗ `greeting`

Wenn Sie Variablennamen mit beschreibendem Charakter verwenden, ist Ihr Code leichter zu verstehen und nachzuvollziehen.

Wenn Sie mehrere Wörter verwenden, um die in einer Variablen enthaltenen Daten zu beschreiben, werden diese üblicherweise durch einen Unterstrich getrennt.

2

Darauf folgt ein Buchstabe oder ein Unterstrich (keine Zahl).

✓ `$greeting`

✗ `$2_greeting`

Die Groß- und Kleinschreibung ist bei der Namensgebung zu beachten, sodass `$Score` und `$score` zwei verschiedene Namen bilden würden. Sie sollten es jedoch generell vermeiden, zwei Variablen zu erstellen, die dasselbe Wort in unterschiedlicher Schreibweise verwenden, da Sie damit andere Personen, die Ihren Code lesen, verwirren könnten.

3

Nutzen Sie dann eine beliebige Kombination aus Buchstaben von A bis Z (Groß- und Kleinbuchstaben), Zahlen und Unterstrichen. (Bindestriche oder Punkte sind nicht erlaubt.)

✓ `$greeting_2`

✗ `$greeting-2`

✗ `$greeting.2`

Hinweis: `$this` hat eine besondere Bedeutung. Verwenden Sie es nicht als Variablenname.

✗ `$this`

Aus technischen Gesichtspunkten können Sie Zeichen aus verschiedenen Zeichensätzen verwenden (z. B. chinesische oder kyrillische Zeichen), aber es wird häufig als gängige Praxis angesehen, nur die Buchstaben A–z, Zahlen und Unterstriche zu verwenden (da die Unterstützung anderer Zeichen mit einigen Schwierigkeiten verbunden ist).

SKALARE (EINFACHE) DATENTYPEN

PHP unterscheidet zwischen drei skalaren Datentypen, die Text, Zahlen und boolesche Werte enthalten.

STRING-DATENTYP

Programmierer bezeichnen ein Textstück als Zeichenkette oder **String**. Der String-Datentyp kann aus Buchstaben, Zahlen und anderen Zeichen bestehen, die jedoch zur Darstellung von Text verwendet werden.

```
$name = 'Ivy';
```

Zeichenketten werden immer von einfachen oder doppelten Anführungszeichen umgeben. Das öffnende Anführungszeichen muss mit dem schließenden Anführungszeichen übereinstimmen.

```
$name = 'Ivy';
$name = "Ivy";
$name = "Ivy';
$name = 'Ivy";
```

NUMERISCHE DATENTYPEN

Numerische Datentypen ermöglichen es, mathematische Operationen mit den darin gespeicherten Werten durchzuführen, wie etwa Additionen oder Multiplikationen.

```
$price = 5;
```

Zahlen werden nicht in Anführungszeichen gesetzt. Wenn Sie Zahlen in Anführungszeichen setzen, werden sie statt als Zahlen als Zeichenketten behandelt.

PHP kennt zwei numerische Datentypen:

`int` steht für Integer, also für ganze Zahlen (z. B. 275).

`float` enthält Fließkommazahlen, die Dezimalbrüche darstellen (z. B. 2,75).

BOOLESCHE DATENTYPEN

Der boolesche Datentyp kann nur einen von zwei Werten annehmen: `true` oder `false`. Diese Werte sind in den meisten Programmiersprachen üblich.

```
$logged_in = true;
```

`true` und `false` sollten in Kleinbuchstaben geschrieben und nicht in Anführungszeichen gesetzt werden. Auf den ersten Blick wirken boolesche Werte vielleicht abstrakt, aber mit `true` oder `false` lassen sich viele Dinge ausdrücken, wie zum Beispiel:

- Ist jemand eingeloggt?
- Hat die Person den allgemeinen Geschäftsbedingungen zugestimmt?
- Ist ein Produkt für den kostenlosen Versand qualifiziert?

NULL-DATENTYP

PHP kennt auch den Datentyp `null`. Dieser kann nur den Wert `null` haben. Er zeigt an, dass für eine Variable noch kein Wert definiert wurde.

MIT DATENTYPEN JONGLIEREN

Auf den Seiten 60 und 61 sehen Sie, wie der PHP-Interpreter verschiedene Datentypen ineinander umwandeln kann (z. B. einen String in einen Zahlenwert).

EINEN VARIABLENWERT AKTUALISIEREN

Sie können den in einer Variablen abgelegten Wert ändern oder überschreiben, indem Sie ihr einen neuen Wert zuweisen. Das erfolgt auf die gleiche Weise, wie Sie der Variablen bei der Erstellung einen Wert zuweisen.

1. Die Variable `$name` wird **initialisiert**. Das bedeutet, dass sie deklariert und ihr ein Anfangswert zugewiesen wird. Dieser wird verwendet, sofern die Variable später auf der Seite nicht aktualisiert wird.

Der Anfangswert ist `Guest`; da es sich um Text handelt, wird er in Anführungszeichen geschrieben.

2. Der Variablen `$name` wird nachfolgend der neue Wert `Ivy` zugewiesen.

3. Die Variable `$price` enthält den Preis für eine einzelne Packung Süßigkeiten.

Als Nächstes sehen Sie den HTML-Code, der an den Browser zurückgesendet wird. Darin wird:

4. der Name mit dem Befehl `echo` in die Seite geschrieben.

Angezeigt wird der aktualisierte Wert, der der Variablen `$name` in Schritt 2 zugewiesen wurde.

5. der Süßigkeitenpreis auf die Seite geschrieben.

section_a/c01/updating-variables.php **PHP**

```
<?php
① $name  = 'Guest';
② $name  = 'Ivy';
③ $price = 5;
?>
<!DOCTYPE html>
<html>
  <head>
    <title>Updating Variables</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <h1>The Candy Store</h1>
④   <h2>Welcome <?php echo $name; ?></h2>
    <p>The cost of your candy is
⑤      $<?php echo $price; ?> per pack.</p>
  </body>
</html>
```

ERGEBNIS

Probieren Sie aus: Ändern Sie in Schritt 2 den Wert der Variablen `$name` so, dass er Ihren Namen enthält. Speichern Sie die Datei und aktualisieren Sie dann die Seite in Ihrem Browser. Nun wird Ihr Name angezeigt.

Probieren Sie aus: Fügen Sie nach Schritt 2 eine neue Zeile ein und geben Sie der Variablen `$name` einen anderen Namen. Speichern Sie die Datei und aktualisieren Sie die Seite. Sie sehen nun den neuen Namen.

ARRAYS

Eine Variable kann auch ein **Array** oder **Datenfeld** enthalten, in dem eine Reihe zusammengehöriger Werte gespeichert ist. Arrays werden auch zu den **zusammengesetzten Datentypen** gezählt, da sie mehr als einen Wert speichern können.

Ein Array ist wie ein Behälter, der eine Reihe zusammengehöriger Variablen enthält. Jeder Eintrag in einem Array wird als **Element** bezeichnet. Ebenso wie eine Variable einen Namen für die Darstellung eines Werts verwendet, besitzt auch jedes Element in einem Array:

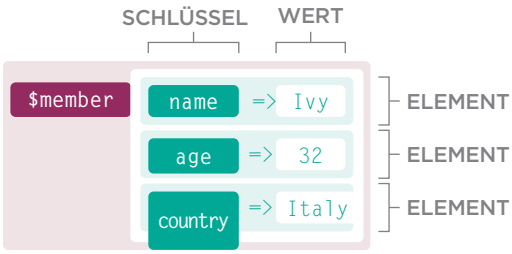
- einen **Schlüssel**, der sich genau wie ein Variablenname verhält
- einen **Wert**, also die zum Namen gehörigen Daten

PHP kennt zwei Arten von Arrays:

- In **assoziativen Arrays** bildet ein Name, der die darunter abgelegten Daten beschreibt, den Schlüssel für jedes Element.
- In **indizierten Arrays** bestehen die Elementschlüssel aus fortlaufenden Zahlen, den sogenannten **Indexwerten**.

ASSOZIATIVES ARRAY

Das nachstehende Array dient der Speicherung von Daten, die für ein Mitglied einer Website stehen. Bei jeder Verwendung des Arrays bleiben die Schlüsselnamen (die die in den einzelnen Elementen des Arrays gespeicherten Daten beschreiben) gleich.

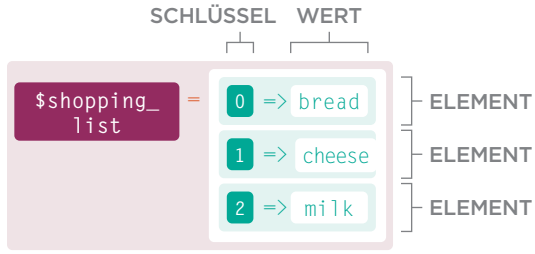


In diesen beiden Beispielen ist jeder im Array gespeicherte Wert ein skalarer Datentyp (ein einzelnes Datenelement).

Auf Seite 44 finden Sie Beispiele für Arrays, deren Elemente selbst auch wieder ein weiteres Array enthalten.

INDIZIERTES ARRAY (STANDARD-ARRAY)

Das nachstehende Array dient zur Erfassung einer Einkaufsliste. Derartige Listen können bei jeder Nutzung eine andere Anzahl von Elementen enthalten. Der Schlüssel nutzt keinen Namen zur Beschreibung der einzelnen Listenelemente, sondern einen aufsteigenden Indexwert (ganzzahlig und bei 0 beginnend).



Hinweis: Die Indexwerte beginnen bei 0, nicht bei 1. Das erste Listenelement hat den Index 0, das zweite Element hat den Index 1 und so weiter. Mit der Indexnummer wird häufig die Reihenfolge der Listenelemente beschrieben.

ASSOZIATIVE ARRAYS

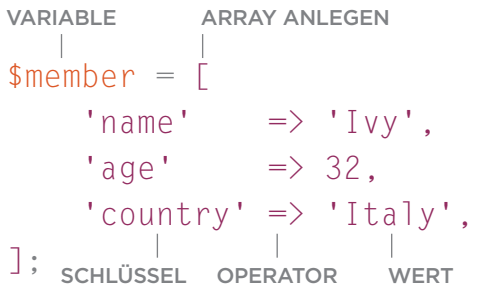
Um ein assoziatives Array zu erstellen, weisen Sie jedem Element (oder Eintrag) im Array einen **Schlüssel** zu, der die darin enthaltenen Daten beschreibt.

Um ein assoziatives Array in einer Variablen zu speichern, verwenden Sie:

- einen Variablenamen, der alle im Array enthaltenen Werte beschreibt
- den Zuweisungsoperator
- eckige Klammern zur Erstellung des Arrays

Innerhalb der eckigen Klammern verwenden Sie:

- den Schlüsselnamen in Anführungszeichen
- den Doppelpfeil-Operator =>
- den Wert für dieses Element (Zeichenketten stehen in Anführungszeichen, Zahlen und boolesche Werte nicht)
- ein Komma nach jedem Element



Ein assoziatives Array kann auch mit der unten dargestellten Syntax erstellt werden, wobei auf das Wort array dann runde (anstelle von eckigen) Klammern folgen.

```
$member = array(
  'name'    => 'Ivy',
  'age'     => 32,
  'country' => 'Italy',
);
```

Um auf ein Element eines assoziativen Arrays zuzugreifen, verwenden Sie:

- den Namen der Variablen, die das Array enthält
- gefolgt von eckigen Klammern und Anführungszeichen
- den Schlüssel des abzurufenden Elements

ASSOZIATIVE ARRAYS ANLEGEN UND DARAUF ZUGREIFEN

PHP section_a/c01/associative-arrays.php

```php
<?php
$nutrition = [
    'fat'   => 16,
    'sugar' => 51,
    'salt'  => 6.3,
];
?>
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Nutrition (per 100g)</h2>
    <p>Fat:   <?php echo $nutrition['fat']; ?>%</p>
    <p>Sugar: <?php echo $nutrition['sugar']; ?>%</p>
    <p>Salt:  <?php echo $nutrition['salt']; ?>%</p>
  </body>
</html>
```

ERGEBNIS

Probieren Sie aus: Fügen Sie in Schritt 1 ein weiteres Element in das Array ein. Verwenden Sie den Schlüssel `protein` und weisen Sie ihm einen Wert von 2.6 zu. Lassen Sie dann in Schritt 2 den Proteinwert auf der Seite anzeigen.

1. In diesem Beispiel erstellen wir ein assoziatives Array und speichern es in der Variablen `$nutrition`. Das Array wird in eckigen Klammern erstellt. Es besteht aus drei Elementen (jedes Element umfasst ein Schlüssel/Wert-Paar). Der Operator => ordnet den Schlüsseln die einzelnen Werte zu.

2. Um im Array gespeicherte Daten anzuzeigen, verwenden Sie:

- den `echo`-Befehl, der dafür sorgt, dass der nachfolgende Wert auf der Webseite ausgegeben wird
- gefolgt von dem Namen der Variablen, die das Array enthält
- gefolgt von eckigen Klammern und Anführungszeichen, die den Schlüsselnamen enthalten, auf dessen Wert Sie zugreifen möchten

Um beispielsweise den Zuckergehalt auf der Seite auszugeben, verwenden Sie:

```
echo $nutrition
['sugar'];
```

Probieren Sie aus: Ändern Sie in Schritt 1 die Werte des Arrays. Geben Sie dem Schlüssel:

- `fat` einen Wert von 42
- `sugar` einen Wert von 60
- `salt` einen Wert von 3.5

Speichern und aktualisieren Sie die Seite, um die aktualisierten Werte zu sehen.

INDIZIERTE ARRAYS

Wenn bei der Deklaration eines Arrays nicht für jedes Element ein Schlüssel angegeben wird, weist der PHP-Interpreter ihm eine Zahl zu, die Indexnummer. Indexnummern beginnen bei null (0), nicht bei eins (1).

Um ein indiziertes Array in einer Variablen zu speichern, verwenden Sie:

- einen Variablenamen, der alle im Array enthaltenen Werte beschreibt
- den Zuweisungsoperator
- eckige Klammern zur Erstellung des Arrays

Innerhalb der eckigen Klammern verwenden Sie:

- die Liste der Werte, die im Array gespeichert werden sollen (Zeichenketten stehen in Anführungszeichen, Zahlen und boolesche Werte nicht)
- ein Komma nach jedem Wert

Jedes Element erhält dabei eine Indexnummer.

Oben würde `bread` die Indexnummer 0, `cheese` die 1 und `milk` die 2 erhalten. Mit Indexnummern wird oft die Reihenfolge der im Datenfeld enthaltenen Elemente angegeben.

Ein indiziertes Array kann auch mit der unten dargestellten Syntax erstellt werden, wobei auf das Wort array dann runde (anstelle von eckigen) Klammern folgen.

```
$shopping_list = array('bread',
                       'cheese',
                       'milk');
```

Neue Werte können dem Array in derselben oder in einer neuen Zeile (so wie oben gezeigt) zugewiesen werden.

Um auf die Daten eines indizierten Arrays zuzugreifen, verwenden Sie:

- den Namen der Variablen, die das Array enthält
- gefolgt von eckigen Klammern (keine Anführungszeichen)
- dem Indexwert des Elements, auf das Sie zugreifen möchten (in den eckigen Klammern)

Der nachfolgende Code greift auf das dritte Array-Element zu, in diesem Beispiel als auf den Wert `milk`.

INDIZIERTE ARRAYS ANLEGEN UND DARAUF ZUGREIFEN

PHP section_a/c01/indexed-arrays.php

```php
<?php
$best_sellers = ['Chocolate', 'Mints', 'Fudge',
    'Bubble gum', 'Toffee', 'Jelly beans',];
?>
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Best Sellers</h2>
    <ul>
      <li><?php echo $best_sellers[0]; ?></li>
      <li><?php echo $best_sellers[1]; ?></li>
      <li><?php echo $best_sellers[2]; ?></li>
    </ul>
  </body>
</html>
```

ERGEBNIS

1. In diesem Beispiel erstellen wir zunächst die Variable `$best_sellers`. Als Variablenwert enthält sie ein Array mit einer Liste der meistverkauften Artikel auf der Website.

Dieses Array wird mittels eckiger Klammern erstellt, und seine Elemente stehen innerhalb dieser Klammern. Da es sich bei den Elementen im Array um Text handelt, stehen sie zudem in Anführungszeichen. (Zahlen und boolesche Werte gehören nicht in Anführungszeichen.) Auf jedes Element folgt ein Komma.

2. Hier werden die drei meistverkauften Artikel auf die Seite geschrieben:

- der `echo`-Befehl sorgt für die Ausgabe des nachfolgenden Werts
- gefolgt wird er vom Namen der Variablen, die das Array enthält
- danach steht die Indexnummer des abzurufenden Elements in eckigen Klammern. Denken Sie daran, dass Indexnummern bei 0 beginnen, nicht bei 1.

Probieren Sie aus: Fügen Sie in Schritt 1 hinter `Fudge` noch `Licorice` in die Auflistung ein. In Schritt 2 lassen Sie auch noch den 4. und 5. Artikel aus dem Array anzeigen.

ARRAYS AKTUALISIEREN

Nachdem Sie ein Array erstellt haben, können Sie ihm neue Elemente hinzufügen oder den Wert eines bestehenden Elements aktualisieren.

Um einen in einem assoziativen Array abgelegten Wert zu aktualisieren, verwenden Sie:

- den Namen der Variablen, die das Array enthält
- gefolgt von eckigen Klammern
- nun den Schlüsselnamen in Anführungszeichen
- einen Zuweisungsoperator
- den neuen Wert, den es enthalten soll

```
$member['name'] = 'Tom';
```

VARIABLE SCHLÜSSEL NEUER WERT

Um einem assoziativen Array ein neues Element hinzuzufügen, gehen Sie genauso vor wie oben beschrieben, nutzen dabei aber einen neuen Schlüsselnamen (keinen, der bereits im Array verwendet wurde).

Die Anführungszeichen umschließen den Schlüsselnamen, wenn es sich dabei um eine Zeichenkette handelt, da Anführungszeichen einen String-Datentyp anzeigen.

Um einen in einem indizierten Array abgelegten Wert zu aktualisieren, verwenden Sie:

- den Namen der Variablen, die das Array enthält
- gefolgt von eckigen Klammern
- die Indexnummer (ohne Anführungszeichen)
- einen Zuweisungsoperator
- den neuen Wert, den es enthalten soll

```
$shopping_list[2] = 'butter';
```

VARIABLE INDEX-NUMMER NEUER WERT

Wie man Elemente zu indizierten Arrays hinzufügt, erfahren Sie auf Seite 220. Der Ablauf ist etwas anders, da Sie die Position des neuen Elements im Array bestimmen können.

Indexnummern stehen nicht in Anführungszeichen, da numerische Datentypen nicht in Anführungszeichen geschrieben werden.

WELCHER ARRAY-TYP FÜR WELCHE ANWENDUNG?

Assoziative Arrays sind am besten geeignet, wenn Sie:

- genau wissen, welche Informationen das Array enthalten soll. Dies ist erforderlich, um für jedes Element einen Schlüsselnamen zu vergeben.
- einzelne Daten unter Verwendung eines Schlüsselnamens abrufen müssen.

Indizierte Arrays sind hilfreich, wenn Sie:

- nicht wissen, wie viele Daten in dem Array gespeichert werden sollen. (Der Index wird stetig erweitert, je mehr Elemente Sie der Liste hinzufügen.)
- eine Reihe von Werten in einer bestimmten Reihenfolge abspeichern möchten.

IN ARRAYS GESPEICHERTE WERTE VERÄNDERN

PHP section_a/c01/updating-arrays.php

```php
<?php
$nutrition = [
    'fat'   => 38,
    'sugar' => 51,
    'salt'  => 0.25,
];
$nutrition['fat']   = 36;
$nutrition['fiber'] = 2.1;
?>
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Nutrition (per 100g)</h2>
    <p>Fat:   <?php echo $nutrition['fat']; ?>%</p>
    <p>Sugar: <?php echo $nutrition['sugar']; ?>%</p>
    <p>Salt:  <?php echo $nutrition['salt']; ?>%</p>
    <p>Fiber: <?php echo $nutrition['fiber']; ?>%</p>
  </body>
</html>
```

ERGEBNIS

1. In diesem Beispiel wird zunächst ein Array in der Variablen `$nutrition` abgelegt.

Die Schlüssel und Werte, die die einzelnen Array-Elemente bilden, müssen nicht unbedingt auf einer neuen Zeile stehen (so wie hier gezeigt), allerdings erhöht diese Praxis die Lesbarkeit.

2. Der Wert für den Fettgehalt wird von 38 auf 36 aktualisiert.

3. Ein neues Element wird zum Array hinzugefügt. Der Schlüssel heißt `fiber` und sein Wert ist `2.1`.

4. Die Werte im Array werden auf der Seite ausgegeben.

Probieren Sie aus: Fügen Sie nach Schritt 3 einen weiteren Schlüssel für Eiweiß (`protein`) hinzu und weisen Sie ihm den Wert `7.3` zu.

ARRAYS IN EINEM ARRAY SPEICHERN

Der Wert eines jeden Elements innerhalb eines Arrays kann ein weiteres Array sein. Enthält jedes Array-Element ein weiteres Array, spricht man von einem **mehrdimensionalen Array**. Dieses eignet sich für die Darstellung von Daten, wie sie etwa in Tabellen vorkommen.

In einigen Fällen müssen Sie einen zusammenhängenden Wertesatz in einem Element eines Arrays speichern (z. B. bei der Darstellung von Daten, die sonst üblicherweise in Tabellen zu finden sind). Sehen Sie sich die rechte Tabelle mit drei Personen, ihrem Alter und ihren jeweiligen Aufenthaltsorten an.

NAME	ALTER	LAND
Ivy	32	UK
Emi	24	Japan
Luke	47	USA

Jede Zeile dieser Tabelle (jede Person) lässt sich durch ein Element eines indizierten Arrays darstellen. Jedes dieser Elemente kann dann wiederum ein assoziatives Array enthalten, in dem der Name, das Alter und das Land der jeweiligen Person gespeichert sind.

Die Indexnummern für das indizierte Array werden automatisch vom PHP-Interpreter zugewiesen. Das Komma nach jedem assoziativen Array zeigt das Ende des Werts für dieses Element an.

```
$members = [
    ['name' => 'Ivy',  'age' => 32, 'country' => 'UK',],
    ['name' => 'Emi',  'age' => 24, 'country' => 'Japan',],
    ['name' => 'Luke', 'age' => 47, 'country' => 'USA',],
];
```

Um das Array mit den Daten über Emi zu erhalten, verwenden Sie:

- den Namen der Variablen, die das indizierte Array enthält
- die Indexnummer des Elements, auf das Sie zugreifen möchten in eckigen Klammern (denken Sie daran, dass die Zählung bei 0 beginnt und dass Zahlen nicht in Anführungszeichen gesetzt werden).

```
$members[1];
```

Um das Alter von Luke abzurufen, verwenden Sie:

- den Namen der Variablen, die das indizierte Array enthält
- die Indexnummer des Elements, das die Informationen über Luke enthält in eckigen Klammern
- den Schlüssel des Elements, auf das Sie im Array über Luke zugreifen möchten in einem zweiten Satz eckiger Klammern (der Schlüssel ist eine Zeichenkette und gehört daher in Anführungszeichen gesetzt)

```
$members[2]['age'];
```

MEHRDIMENSIONALE ARRAYS

PHP section_a/c01/multidimensional-arrays.php

```php
<?php
$offers = [
    ['name' => 'Toffee', 'price' => 5, 'stock' => 120,],
    ['name' => 'Mints',  'price' => 3, 'stock' => 66,],
    ['name' => 'Fudge',  'price' => 4, 'stock' => 97,],
];
?>
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Offers</h2>
    <p><?php echo $offers[0]['name']; ?> -
      $<?php echo $offers[0]['price']; ?> </p>
    <p><?php echo $offers[1]['name']; ?> -
      $<?php echo $offers[1]['price']; ?> </p>
    <p><?php echo $offers[2]['name']; ?> -
      $<?php echo $offers[2]['price']; ?> </p>
  </body>
</html>
```

ERGEBNIS

1. Dieses Beispiel beginnt mit der Hinterlegung eines indizierten Arrays in der Variablen `$offers`.

Jedes Element im Array enthält ein assoziatives Array mit dem Namen, Preis und Lagerbestand eines angebotenen Artikels.

2. Der Name des ersten Produkts wird ausgegeben. (Die Indexnummer des ersten Produkts ist 0.)

3. Der Preis des ersten Produkts wird ausgegeben.

4. Der Name und der Preis des zweiten Produkts werden ausgegeben.

5. Der Name und der Preis des dritten Produkts werden ausgegeben.

Probieren Sie es: Fügen Sie in Schritt 1 ein weiteres Produkt mit dem Namen `Chocolate` zum Array hinzu. Setzen Sie den Preis auf `2` und den Lagerbestand auf `83`. Geben Sie dann nach Schritt 5 den Namen und den Preis des neu hinzugefügten Produkts aus.

Im nächsten Kapitel lernen Sie, wie Sie mithilfe einer Schleife den Namen und den Preis aller im Array `$offers` enthaltenen Produkte ausgeben können, und zwar unabhängig von der Anzahl der darin enthaltenen Produkte.

KURZFORM FÜR ECHO

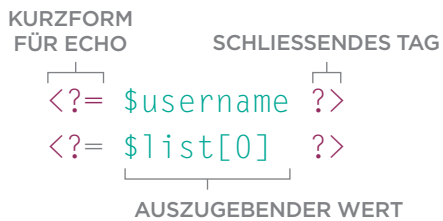
Wenn ein PHP-Block nur dazu dient, einen Wert in den Browser zu schreiben, können Sie anstelle von `<?php echo ?>` eine Abkürzung verwenden.

Anstatt `<?php echo $name; ?>` zu schreiben, können Sie die Abkürzung `<?= $name ?>` verwenden.

Dies ist die einzige Situation, in der Sie nicht das vollständige öffnende `<?php`-Tag brauchen.

Das können Sie weglassen:

- Die Buchstaben `php` im öffnenden Tag
- Den `echo`-Befehl
- Den Strichpunkt vor dem schließenden Tag



In vielen der Beispiele in den ersten Buchkapiteln wird deutlich, dass jede PHP-Datei aus zwei Teilen besteht:

- Zuerst speichert der PHP-Code Werte in Variablen oder Arrays. (Er kann auch bestimmte Aktionen mit den darin enthaltenen Daten durchführen.)
- Danach folgt der HTML-Code, der an den Browser zurückgesendet wird. Dieser zweite Teil der Seite gibt mit der oben gezeigten Kurzsyntax Werte aus, die in Variablen gespeichert wurden.

Wenn Sie zu Beginn jeder Seite die auf der Seite anzuzeigenden Werte erzeugen und in Variablen speichern, können Sie eine klare Trennung zwischen dem auf dem Server ausgeführten PHP-Code und dem HTML-Code, der letztlich im Browser dargestellt wird, herstellen.

Im zweiten Teil der Datei, in dem die HTML-Seite erstellt wird, sollte so wenig PHP-Code wie möglich vorkommen. In unseren ersten Beispielen schreibt der PHP-Code in diesem Teil der Seite lediglich in Variablen gespeicherte Werte in die HTML-Seite.

DIE KURZFORM FÜR ECHO VERWENDEN

PHP section_a/c01/echo-shorthand.php

```
<?php
① $name      = 'Ivy';
② $favorites = ['Chocolate', 'Toffee', 'Fudge',];
?>
<!DOCTYPE html>
<html>
  <head>
    <title>Echo Shorthand</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <h1>The Candy Store</h1>
③   <h2>Welcome <?= $name ?></h2>
    <p>Your favorite type of candy is:
④      <?= $favorites[0] ?>.</p>
  </body>
</html>
```

ERGEBNIS

In diesem Beispiel werden zwei verschiedene Variablen erstellt und am Anfang der Seite mit Werten gefüllt, bevor der HTML-Code beginnt:

1. `$name` enthält den Namen eines Website-Nutzers. Da es sich um Text handelt, wird er in Anführungszeichen gesetzt.

2. `$favorites` enthält eine Reihe von Lieblingssüßigkeiten dieses Mitglieds.

3. Der Name wird mithilfe der Kurzschreibweise des `echo`-Befehls in die Seite geschrieben.

4. Die Lieblingssüßigkeit des Mitglieds wird mit der Kurzschreibweise des `echo`-Befehls in die Seite geschrieben.

Probieren Sie es: Ändern Sie in Schritt 1 den in der Variablen `$name` gespeicherten Wert in Ihren eigenen Namen. In Schritt 2 setzen Sie Ihre Lieblingssüßigkeit an den Anfang des Arrays. Speichern Sie die Datei und aktualisieren Sie die Seite in Ihrem Browser. Daraufhin sehen Sie, wie sich der Inhalt ändert.