



Informatik 7bi8bi Schuljahr 2021/2022 als PDF exportieren

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122&do=export_pdfns&book_ns=inf:inf7bi8bi_202122&book_title=Info

Informatik 7. Klasse - Schuljahr 2021/22

Lehrinhalte

- [Lehrplaninhalte](#)

[Remote-Zugriff auf Schulserver](#)

Kapitel

- [1\) PHP](#)
- [2\) Datenbanken](#)
- [3\) MS Office \(Excel\)](#)
- [4\) 3D Modellierung & 3D Druck](#)
- [5\) Bildbearbeitung](#)
- [6\) C++ Strukturen & Klassen](#)
- [7\) CMS Systeme](#)

From:

<http://elearn.bgamstetten.ac.at/wiki/> - **Wiki**

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122

Last update: **2021/09/16 08:21**



1. PHP

1.1) Was ist PHP?

- [1.1.1\) Was ist PHP](#)
- [1.1.2\) How to use PHP](#)

1.2) Grundlegende Sprachelemente

- [1.2.1\) Kommentare](#)
- [1.2.2\) Ausgabe](#)
- [1.2.3\) ÜBUNG 1](#)

1.3) Variablen und Operatoren

- [1.3.1\) Variablen und Operatoren](#)

1.4) Interaktive Webseiten

- [1.4.1\) Formulare](#)
- [1.4.2\) ÜBUNG 2](#)

1.5) Anführungszeichen

- [1.5.1\) Verwenden von Anführungszeichen](#)

1.6) Kontrollstrukturen

- [1.6.1\) Vergleichsoperatoren](#)
- [1.6.2\) if-Anweisung](#)
- [1.6.3\) switch-Anweisung](#)
- [1.6.4\) ÜBUNG 1.7](#)

1.7) Schleifen

- [1.7.1\) for-Schleife](#)
- [1.7.2\) while-Schleife](#)
- [1.7.3\) do-while-Schleife](#)
- [1.7.4\) foreach-Schleife](#)

- [1.7.5\) ÜBUNG 8-11](#)

1.8) Felder

- [1.8.1\) Grundlagen zu Felder](#)
- [1.8.2\) Indizierte Felder](#)
- [1.8.3\) Assoziative Felder](#)
- [1.8.4\) Mehrdimensionale Felder](#)
- [1.8.5\) Weitere Informationen zu Feldern](#)
- [1.8.6\) ÜBUNG 12-18](#)
- [1.8.7\) KONTROLLFRAGEN](#)

3.9) Formulare

- [1.9.1\) Hidden-Feld](#)
- [1.9.2\) Textarea](#)
- [1.9.3\) Checkbox](#)
- [1.9.4\) Radio-Button](#)
- [1.9.5\) Auswahllisten](#)
- [1.9.6\) ÜBUNG 20](#)

1.10) Externe Dateien

- [1.10.1\) Theorie: Nutzung von externen Dateien](#)
- [1.10.2\) Dateien öffnen, lesen und schließen](#)
- [1.10.3\) Aus Dateien lesen](#)
- [1.10.4\) In Dateien schreiben](#)
- [1.10.5\) ÜBUNG 21 Gästebuch](#)
- [1.10.6\) ÜBUNG 22 Besucher zählen](#)
- [1.10.7\) ÜBUNG 23-25](#)

1.11) Datum und Zeit

- [1.13.1\) Datum und Zeit](#)
- [1.13.2\) ÜBUNG 26](#)

1.12) Funktionen

- [1.12.1\) Funktionen erstellen und aufrufen](#)
- [1.12.2\) Funktionen verwenden](#)
- [1.12.3\) Gültigkeitsbereich von Variablen](#)
- [1.12.4\) PHP-Dateien einbinden](#)

- [1.12.5\) Andere Dateitypen einbinden](#)
 - [1.12.6\) ÜBUNG 27-29](#)
-

From:

<http://elearn.bgamstetten.ac.at/wiki/> - **Wiki**

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1



Last update: **2021/09/15 08:05**

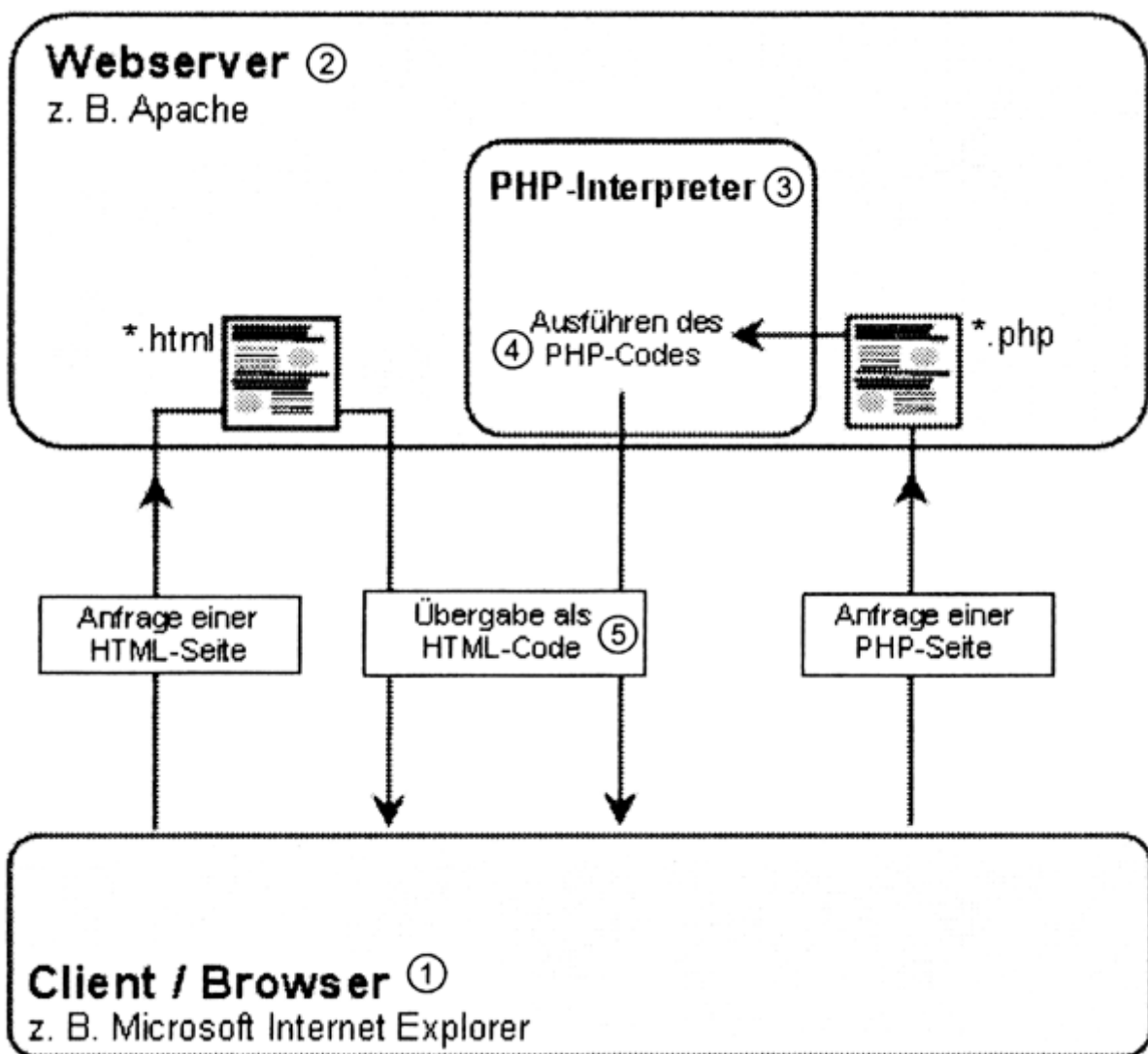
1.1.1) Was ist PHP?

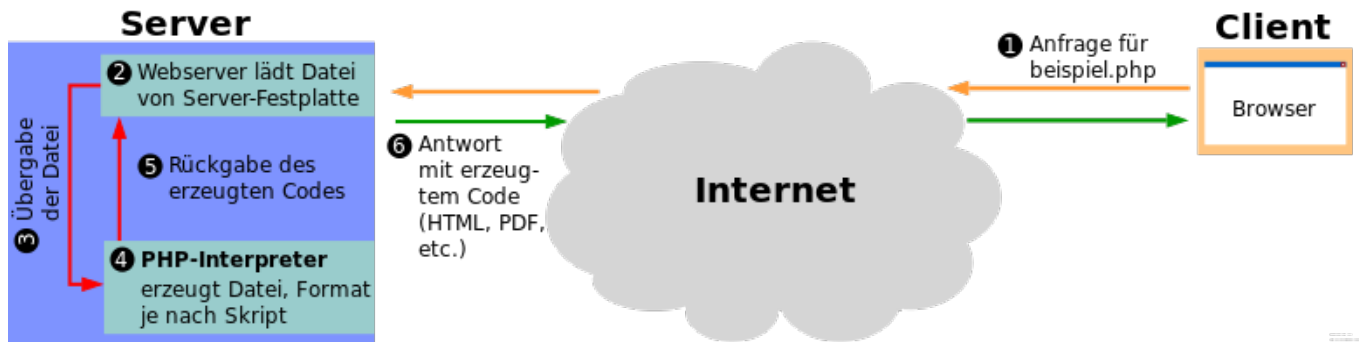
PHP (= PHP Hypertext Preprocessor) ... Skriptsprache für Internet, wird am **Webserver ausgeführt**.

PHP, die Kurzform von Personal HomePage oder auch PHP Hypertext Preprocessor, ist eine Skriptsprache, die speziell für den **Einsatz im Internet** entwickelt wurde. Die auszuführenden Anweisungen können **direkt in den HTML-Code integriert** werden.

Bei einer mit **HTML erstellten Webseite ist keine Dynamik** möglich. Mithilfe von **PHP** können auf einer Webseite **Interaktionen** eingebaut oder die Seite in irgendeiner Form, beispielsweise durch **Reaktionen auf Eingaben**, angepasst werden.

Öffnet der Betrachter in seinem Browser (1) eine solche Webseite über einen **PHP-fähigen Webserver** (2), werden die Anweisungen von PHP interpretiert (3), ausgeführt (4) und das Ergebnis als HTML-Code zurückgesendet (5). Ein klarer Vorteil dabei ist, dass der PHP-Quellcode dem Betrachter nicht zugänglich ist. Er sieht im Browser nur den zurückgelieferten HTML-Code.





Um dynamische Webseiten zu erstellen und später auf Datenbanken zugreifen zu können, benötigt man neben PHP einen Webserver und einen Datenbankserver. Unter Windows kann man einen lokalen Webserver installieren. Als Entwicklungssoftware hat sich XAMPP sehr bewährt.

Unterschied zu Javascript

Im Gegensatz zu Javascript benötigt PHP einen Webserver, der eine Interaktion mit dem Benutzer ermöglicht und somit eine Webseite dynamisch erscheinen lässt.

Server-seitiges vs. Client-seitiges Skripting

PHP ist eine serverseitige Skriptsprache. Das bedeutet, dass es auf dem Webserver läuft, im Gegensatz zu einem Client-Rechner.

Serverseitige Programmierung ist nützlich, um Benutzern dynamische Inhalte (typischerweise aus einer Datenbank) zu liefern, wie z.B. eine Willkommensnachricht („Hi, Claire!“), wenn sich ein Benutzer anmeldet.

Noch wichtiger ist, dass serverseitiges Scripting im E-Commerce eingesetzt wird. Zum Beispiel gibt es über 100 WooCommerce-Erweiterungen, die sich über APIs (Application Programming Interfaces) mit verschiedenen Zahlungsanbietern verbinden, um Transaktionen zu verarbeiten.

JavaScript ist eine clientseitige Sprache, also läuft es auf dem Laptop, dem Telefon oder dem Tablet des Nutzers.

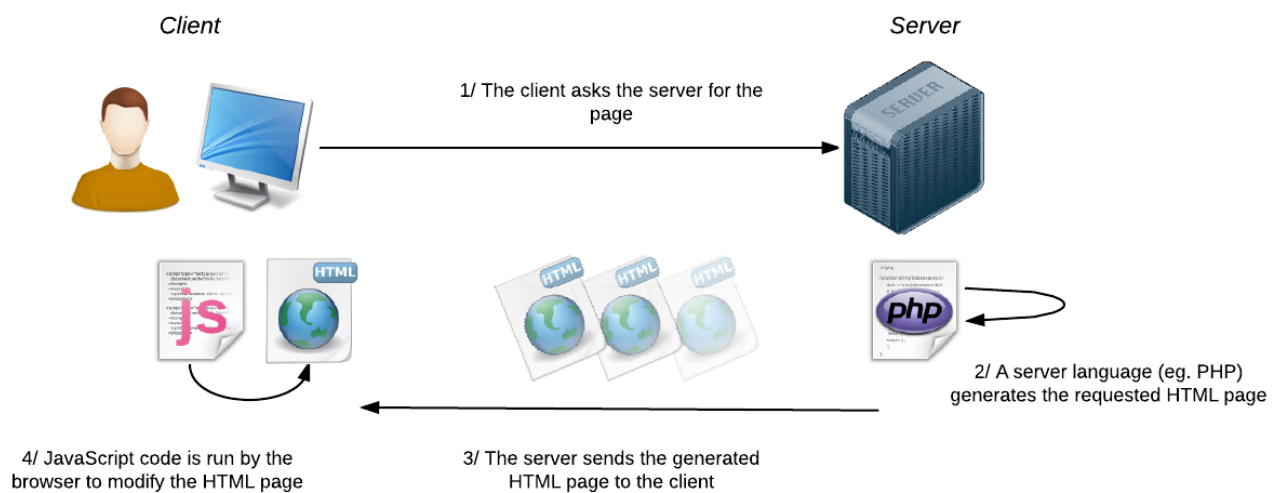
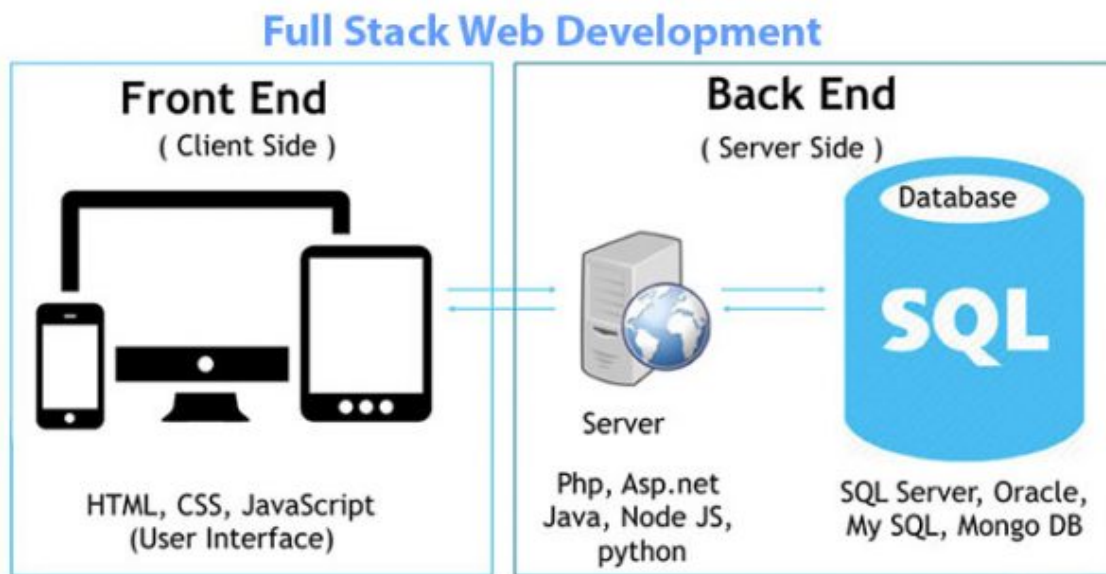
JavaScript kann das DOM manipulieren, was für Document Object Model steht und man kann es sich als eine baumartige Struktur vorstellen, die aus dem HTML einer Webseite gebildet wird.

Wenn du jemals auf ein Akkordeon oder einen Toggle gestoßen bist, vielleicht als Teil eines FAQ Plugins, hast du clientseitiges JavaScript in Aktion gesehen. Wenn du auf eine Frage klickst oder tippst, schalten JavaScript-Event-Handler die CSS-Anzeige- oder Sichtbarkeits-Eigenschaften ein oder aus und zeigen oder verstecken die entsprechende Antwort.

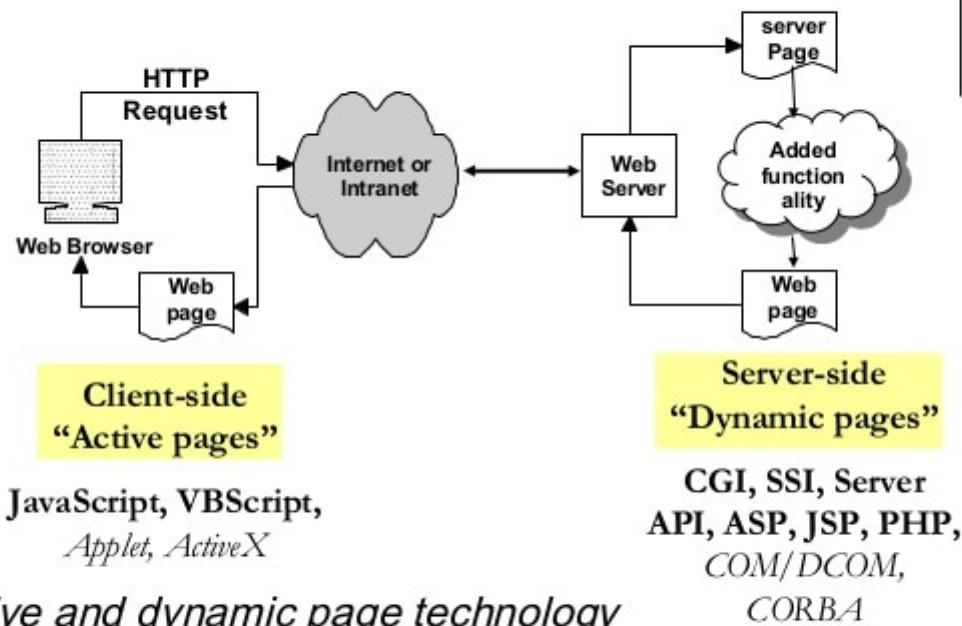
Frontend vs Backend

PHP läuft im Backend einer Webseite – dem Teil, den Besucher nicht sehen! In WordPress bedeutet das, dass PHP seine ganze Arbeit auf dem Webserver und im WordPress-Admin macht.

JavaScript lief traditionell im Frontend, aber das änderte sich 2009, als Node.js, eine Backend-Runtime, eingeführt wurde. Heute ist JavaScript wirklich eine Full-Stack-Sprache.



Where does PHP fit ?



Active and dynamic page technology can be used together – server-side program generates customized active pages.

Download von XAMPP

- <http://www.apachefriends.org/de/xampp.html>
- localhost greift auf den Ordner .../htdocs zu. In diesen Ordner müssen die html- bzw. php-Dateien gespeichert werden.

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_01

Last update: **2021/09/16 08:04**



1.1.2) How to use PHP

PHP-Befehle einfügen

PHP-Code wird direkt in den HTML-Code eingebettet. PHP-Blöcke können vollständig im Dokumentkopf `<head>` oder im Dokumenttrumpf `<body>` stehen. Weiterhin hat man die Möglichkeit, innerhalb einer Datei mehrmals zwischen PHP- und HTML-Code zu wechseln.

Dateinamenerweiterungen mit PHP-Code

Damit der Webserver erkennt, dass es sich um eine Datei mit einem PHP-Aufruf handelt, werden die Dokumente mit der Dateinamenerweiterung `.php` oder `.php5` gespeichert.

XML-Schreibweise

Der auszuführende PHP-Code wird zwischen dem öffnenden Tag `<?php` und dem schließenden Tag `?>` geschrieben

```
<?php
    hier steht ein (oder mehrere) PHP-Befehl(e)
?>
```

Diese Schreibweise ist am gebräuchlichsten. Alternativ dazu kann man auch einen Skript-Tag verwenden. Diese Schreibweise entspricht im Wesentlichen der JavaScript- bzw. CSS-Angabe:

```
<script language="php">
    hier steht ein (oder mehrere) PHP-Befehl(e)
</script>
```

PHP-Anweisungen voneinander trennen

Jede PHP-Anweisung wird durch einen Strichpunkt beendet.

```
<?php
    PHP-Code;
    PHP-Code;
?>
```

Es empfiehlt sich, nach einer Anweisung in einer neuen Zeile weiterzuschreiben.

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_02



Last update: **2021/09/16 08:05**

1.2.1) Kommentare einfügen

- Kommentare werden nicht vom Interpreter ausgeführt.
- Kommentare dienen zur näheren Beschreibung des Quellcodes

Syntax und Bedeutung der Kommentare

```
<html>
<body>
<?php
    echo "Für den Anfang: "; //einzeiliger Kommentar am Ende eines PHP-Codes
    // einzeiliger K0mmentar

    /*
        mehrzeilige Kommentare sind ebenfalls möglich,
        um den Quellcode ausführlicher zu beschreiben
    */
    echo "Hallo Welt!";
?>
</body>
</html>
```

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_03



Last update: **2021/09/16 08:05**

1.2.2) Daten im Browser ausgeben

Befehl "echo"

Da PHP eine eingebettete Skriptsprache ist, die vom Server ausgeführt wird, bleibt der Quellcode im Gegensatz zu HTML für den Client unsichtbar. Zeichenketten und Daten werden über den Befehl echo auf dem Bildschirm ausgegeben.

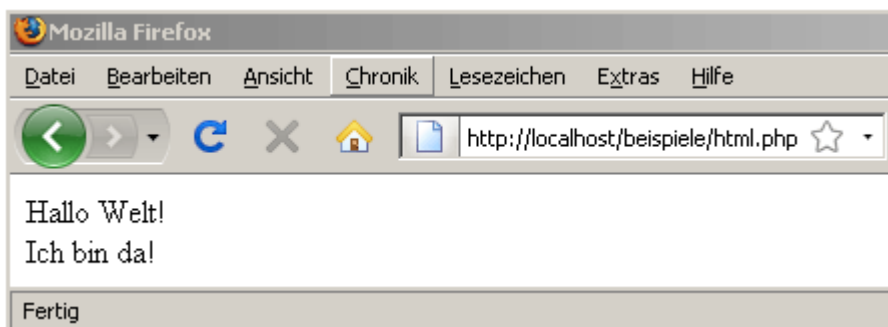
```
echo "Hallo Welt";
```

Umgang mit HTML-Syntax bei der Ausgabe

Um Zeichenketten anzuzeigen, werden diese direkt an den Browser weitergegeben. Daher kann man innerhalb des Befehls echo auch jeden HTML-Tag (z.B.
) angeben. Dieser wird dann vom Browser als normaler HTML-Befehl interpretiert.

```
<html>
<body>
<?php
    echo "Hallo Welt! <br>"; //Der HTML-Tag <br> wird vom Browser korrekt
interpretiert.
    echo "Ich bin da!";
?>
</body>
</html>
```

Screenshot Browser



Im Screenshot wird der HTML-Tag
 richtig interpretiert. Der Browser fügt eine Leerzeile ein. Der Kommentar wird natürlich nicht angezeigt!

Quelltext HTML

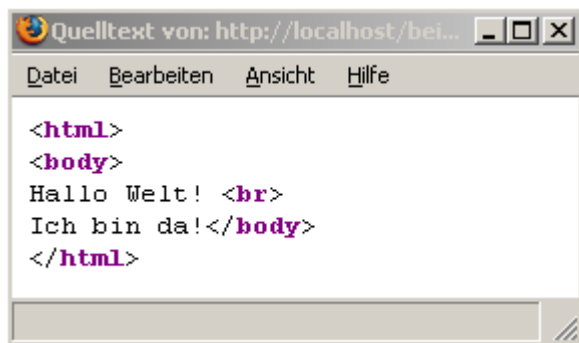


Wie man sieht, wird der PHP-Kommentar auch im HTML-Quelltext nicht angezeigt. Weiters erscheint der übersetzte HTML-Code in einer Zeile.

Der HTML-Quellcode sollte ebenfalls wie der PHP-Quellcode übersichtlich erscheinen. Mittels `\n` kann man in PHP erwirken, dass im HTML-Quelltext eine Leerzeile eingefügt wird. `\n` hat keinen Einfluss auf das Erscheinungsbild im Browser.

```
<html>
<body>
<?php
    echo "Hallo Welt! <br>\n"; // \n bewirkt eine Leerzeile im HTML-Quellcode!
    echo "Ich bin da!";
?>
</body>
</html>
```

Das Ergebnis im HTML-Quelltext sieht wie folgt aus:



Sonderzeichen ausgeben

Wenn man Hochkommas oder einen Backslash ausgeben möchte, setzt man vor den Zeichen einen Backslash:

```
<?php
    echo "<p>Der \"Erfinder\" von PHP war Rasmus Lerdorf. </p>";
    echo "Das Programm wird installiert im Verzeichnis C:\\Programme";
?>
```

Zeilenumbruch im Quelltext

Möchte man im Quelltext einen Zeilenumbruch einfügen, verwendet man \n

```
<?php
    echo "Das ist ein Text<br>\n";
    echo "Das ist noch ein Text";
?>
```

From:

<http://elearn.bgamstetten.ac.at/wiki/> - **Wiki**

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_04



Last update: **2021/09/16 08:05**

1.2.3) Aufgabe 1

Schreibe ein einfaches PHP-Skript, z.B. eine einfache Webseite mit Informationen über dich (Name, Wohnort, Hobbies, ...)

Folgende Elemente sollen dabei vorkommen (eingebunden in PHP-Code):

- Kommentare
- Sonderzeichen, wie z.B. Anführungsstriche
- Farbformatierungen
- Listen
- Tabellen
- ...

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_05

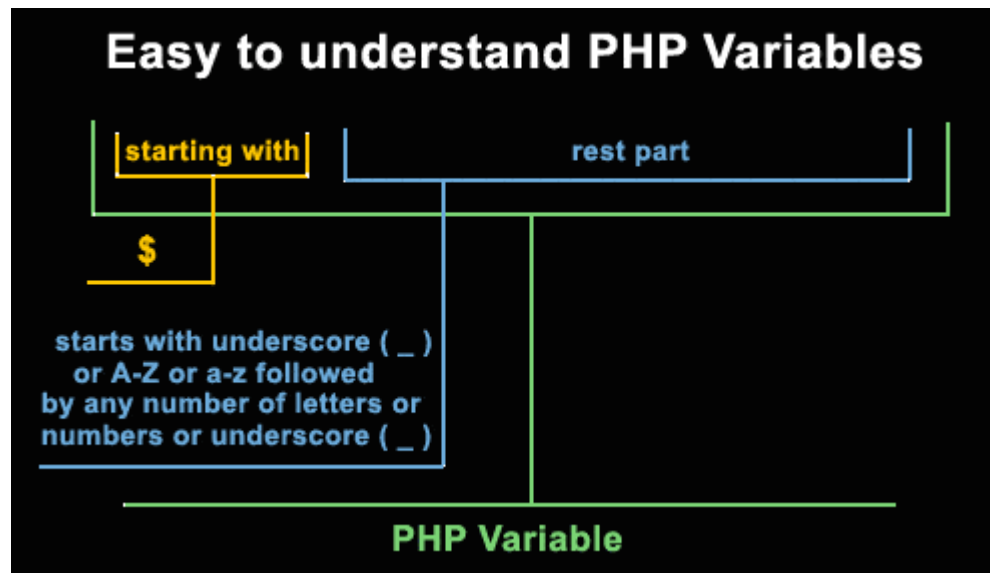


Last update: **2021/09/16 08:05**

1.3.1) Variablen und Operatoren

Variablen

Variablen müssen im Gegensatz zu vielen Programmiersprachen nicht extra deklariert werden. Sie werden mit einem vorangestellten Dollar- Zeichen gekennzeichnet und beim ersten Aufruf definiert.



Folgende Datentypen werden von PHP unterstützt:

Datentyp	Bezeichnung	Beispiel
Ganze Zahlen	integer	1911 oder -1911
Dezimalzahl	double	19.11 oder -19.11
Zeichenketten	string	„Kette von Zeichen“ oder 'Kette von Zeichen'
Felder (ein- oder mehrdimensional)	array	(„Frankfurt“, „Berlin“, „Zürich“)
Objekte	object	Verweis auf eine bestimmte Variable

Namensgebung bei Variablen

Bei der Vergabe von Namen für die Variablen gibt es folgende Regeln. Der Name einer Variablen

- muss mit dem Dollarzeichen beginnen und darf kein Leerzeichen enthalten
- darf nur aus Buchstaben und Ziffern bestehen, aber keine Umlaute oder „ß“ enthalten
- muss als zweites Zeichen einen Buchstaben enthalten
- darf keine Sonderzeichen außer dem Unterstrich „_“ enthalten
- kann Groß- oder Kleinbuchstaben enthalten, wobei zwischen Groß- und Kleinschreibung unterschieden wird (\$PrimZahl ist nicht gleich \$primzahl)
- darf nicht identisch sein mit einem so genannten reservierten Wort (and, break, case, class, continue, default, do, else, elseif, empty, endfor, endif)

Variablen und Operatoren für Zahlen

Numerische Datentypen

Die numerischen Datentypen werden in Ganzzahl-Datentypen `integer` und Fließkommazahl-Datentypen `double` unterteilt.

Beispiel: `preis.php`

```
<?php
$preis_apfel = 2.50;
$menge = 4;
$gesamtpreis = $preis_apfel * $menge;
echo $gesamtpreis;
?>
```

Arithmetische Operatoren

Operator	Name	Bedeutung
+	Addition	<code>\$a + \$b</code> ergibt die Summe
-	Subtraktion	<code>\$a - \$b</code> ergibt die Differenz
*	Multiplikation	<code>\$a * \$b</code> ergibt das Produkt
/	Division	<code>\$a / \$b</code> ergibt den Quotienten
%	Modulo	<code>\$a % \$b</code> ist der Rest der ganzzahligen Division von <code>\$a / \$b</code>
++	Prä-, bzw. Postinkrement	<code>\$a++</code> (<code>++\$a</code>) erhöht die Variable <code>a</code> um 1 nach (vor) der weiteren Verwendung
--	Prä-, bzw. Postdekrement	<code>\$a--</code> (<code>--\$a</code>) verringert die Variable <code>a</code> um 1 nach (vor) der weiteren Verwendung

Beispiel: `berechnung.php`

```
<?php
$preis_apfel = 2.50;
$menge1 = 4;
$menge2 = 10;
$menge3 = 15;
$gesamtpreis = $preis_apfel * ($menge1 + $menge2 + $menge3);
echo $gesamtpreis;
?>
```

Variablen und Operatoren für Zeichenketten

Zeichenkettenoperator

Mittels Zeichenkettenoperator `„.“` können mehrere Zeichenketten miteinander verknüpft werden:

```
<?php
$a = "Hallo ";
$b = "Welt!";
$c = $a.$b;
echo $c;
?>
```

Ergebnis: Hallo Welt!

Typumwandlung

Der Datentyp einer Variablen wird durch die erste Wertzuweisung bestimmt.

Durch eine vorangestellte Typendeklaration kann dieser Datentyp geändert werden.

Typumwandlung in PHP funktioniert oft wie in C. Der Name des gewünschten Typs wird vor der umzuwandelnden Variablen in Klammern gesetzt, dies wird auch als Cast-Operation bezeichnet.

```
<?php
echo (int)$z;
echo "<br>";
var_dump($z); //Gibt die Variable mit zugehörigen Datentyp aus!
echo "<br>";
echo $z;
?>
```

Der obige PHP-Code ergibt folgende Ausgabe:

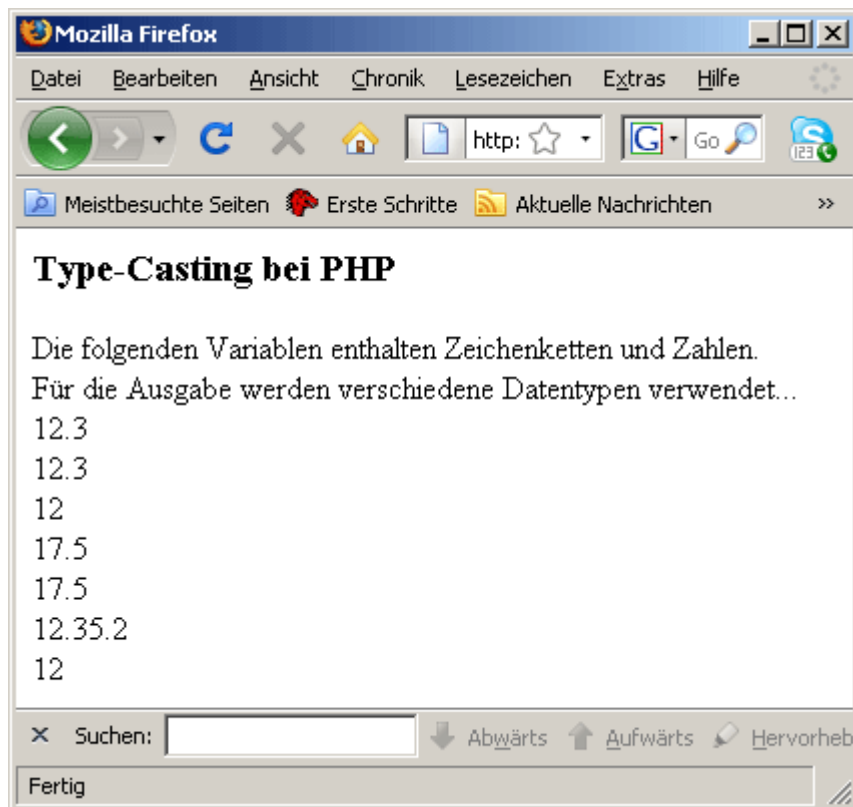
```
10
float(10.5)
10.5
```

Folgende Umwandlungen sind möglich:

- (int), (integer) – nach integer
- (bool), (boolean) – nach boolean
- (float), (double), (real) – nach float
- (string) – nach string
- (array) – nach array
- (object) – Wandlung zum Objekt

Anstatt eine Variable in einen String umzuwandeln, können Sie die Variable auch in doppelte Anführungszeichen einschließen.

Beachten Sie, dass Tabulatoren und Leerzeichen innerhalb der Klammern erlaubt sind. Deshalb sind die folgenden Beispiele identisch:



From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_06



Last update: **2021/09/16 08:06**

1.4.1) Formulare

Mit PHP kann man **interaktive Webseiten** erstellen, bei denen **Benutzereingaben aus Formularen durch PHP ausgewertet** werden. Die **Darstellung der Formulare** wird mit **HTML-Code** umgesetzt. Die Daten aus dem Formular werden dabei **per HTTP** mit den **Methoden POST oder GET** an das **auswertende PHP-Programm** übermittelt.



Methoden der Datenübertragung

POST

- Angabe von `method="POST"` im einleitenden HTML-Tag `<form>`
- Formulardaten werden direkt an die Webadresse versendet.
- Daten können nicht in der Adresszeile des Browsers manipuliert werden. Optionen sind nur über das Formular möglich.
- Daten werden nicht im Logfile des Servers gespeichert.
- Formulardaten sind nicht im Verlauf des Browsers sichtbar.
- Längere Texte oder Daten können in Formularen übermittelt werden, da keine Beschränkung der Größe besteht.

GET

- Angabe von `method="GET"` im einleitenden HTML-Tag `<form>`
- Formulardaten werden in der URL übermittelt und werden dann durch den Server in einer speziellen Umgebungsvariablen zwischengespeichert.

- Daten sind in der Adresszeile des Browsers änderbar, ohne das Formular erneut ausfüllen zu müssen.
- Der Aufruf des Skripts mit Angabe der Daten kann als Favorit abgelegt werden.
- Daten sind auf ca. 2KB beschränkt.

Da bei der Methode GET die Daten über die URL an den Server weitergegeben werden, ist es möglich, die Daten zu manipulieren. Die sicherere und am häufigsten genutzte Methode ist daher die Methode POST.

Formulare auswerten mit PHP

Formulardaten eingeben

Zur Eingabe von Formulardaten, genügt eine HTML-Datei, in der ein Formular integriert ist:

formular_eingabe.html

[formular_eingabe.html](#)

```
<html>
<body>
<p>Bitte füllen Sie die nachfolgenden Eingabefelder aus:</p>
<form action="formular_auswertung.php" method="POST">
<pre>
<p>
Vorname: <input type="Text" name="vorname"><br>
Nachname: <input type="Text" name="nachname"><br>
Wohnort: <input type="Text" name="ort"><br>
</p>
<input type="Submit" value="Abschicken">
<input type="Reset" value="Zurücksetzen">
</pre>
</form>
</body>
</html>
```

Formulardaten übertragen

Klickt man auf den Button „Abschicken“, wird das in action angegebene Skript (formular_auswertung.php) gestartet.

Formulardaten auswerten

Die mit der Methode POST übermittelten Daten wird mit dem Befehl `$_POST[„Parameter“]` angesprochen. Der Name des Eingabefeldes im Formular (z.B. `name=„vorname“`) wird automatisch Element der Variablen `$_POST` (z.B. `$_POST[„vorname“]`) Der Eintrag des Eingabefeldes im Formular wird bei der Auswertung zum Wert der Variablen, z.B. `$_POST[„vorname“] = „Max“`.

formular_auswertung.php

formular_auswertung.php

```
<?php
echo "<p>Folgende Daten wurden übermittelt:</p>\n";
echo "Vorname: " . $_POST["vorname"] . "<br>\n";
echo "Nachname: " . $_POST["nachname"] . "<br>\n";
echo "Wohnort: " . $_POST["ort"] . "<br>\n";
?>
```

Formular und Auswertung in einer Datei

Das Formular und die Auswertung können in einer Datei zusammengefasst werden. Mit einer if-Abfrage kann überprüft werden, ob der Submit-Button gedrückt wurde oder nicht.

pizzabestellung.php

pizzabestellung.php

```
Pizzabestellung<br>
Pizza Margarita zum günstigen Preis von 5.50 Euro bestellen! <br>
<pre>
<form action="pizzabestellung.php" method="POST">
  Name: <input type="Text" name="name" />
  Lieferadresse: <input type="Text" name="adresse" />
  Anzahl der Pizzen: <input type="Text" name="anzahl" /><br><br>
  <input type="Submit" name="schicken" value="Bestellung
abschicken"><br>
  <input type="reset" value="Zurücksetzen">
</form>
</pre>
<hr>
<?php
  if ($_POST["schicken"]=="Bestellung abschicken")
  {
    echo "<i>Für folgende Bestellung wird gedankt: </i><br><br>";
    echo "<b>Name:</b> " . $_POST["name"] . "<br>";
    echo "<b>Lieferadresse:</b> " . $_POST["adresse"] . "<br>";
    if ($_POST["anzahl"]<>0)
```

```
{
    echo "Menge beträgt <b>".$_POST["anzahl"]." Stück</b> zum Preis von
    <b>".$_POST["anzahl"]*5.50." Euro</b>";
}
}

?>
```

Überprüfen des mitgeschickten Arrays mittels POST

```
print_r($_POST);
var_dump($_POST);
```

`print_r()` Zeigt den Inhalt der Variablen in lesbarer Form an. `var_dump()` geht einen Schritt weiter, es können auch, durch Beistriche getrennt, mehrere Variablen angegeben werden. Die Informationen sind detaillierter. Sie enthalten auch Typ und Länge der Variablen. `var_dump(Variable 1[, Variable 2,...])`; Das gesamte mittels POST übermittelte Array wird ausgegeben.

Beispiel Newsletter

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_07

Last update: **2021/09/16 08:06**



1.4.2) Aufgabe 2

Erstelle eine kleine Website für eine Apfelbestellung:

- bioladen.html
 - Eingabe von Name, Adresse, Menge (in kg) 1. Sorte Golden Delicious (1,20 pro kg), 2. Sorte: Idared (1,50 pro kg)
- bioladen-auswertung.php
 - Ausgabe von Name, Adresse, bestellter Menge und Gesamtpreis
 - ab insg. 10 kg bestellter Menge: keine Versandkosten, sonst 5,00 Euro.

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_08



Last update: **2021/09/16 08:07**

1.5.1) Anführungszeichen

Doppelte oder einfache Anführungszeichen

```
$foo = 'Heute';

echo "$foo ist ein schöner Tag!"; // Ausgabe: Heute ist ein schöner Tag!
echo '$foo ist ein schöner Tag!'; // Ausgabe: $foo ist ein schöner Tag!
```

Wenn ein PHP-Programmierer einen Link aus Variablen erstellt, kann man häufig folgenden Code sehen:

```
echo '<a href="' . $link . ' " id="' . $id . ' " class="' . $class . ' ">' .
$linktext . '</a>';
```

oder

```
?>
<a href="<?php echo $link; ?>" id="<?php echo $id; ?>"
    class="<?php echo $class; ?>"<?php echo $linktext; ?>"</a>
<?php
```

oder

```
echo "<a href=\"\$link\" id=\"\$id\" class=\"\$class\">\$linktext</a>";
```

Abgesehen vom persönlichen Stil des Programmierers haben die Code-Beispiele eines gemeinsam: sie sind schlecht zu lesen. „Klar!“, werden jetzt die Programmierer sagen, „In HTML müssen die Attribute in doppelten Anführungszeichen stehen und in PHP muss man eine der oben gezeigten Methoden verwenden“. Sicher? Müssen Attribute wirklich in doppelten Anführungszeichen stehen? Die einfache Antwort ist: Nein! Ein Blick in die Spezifikation bringt Folgendes an den Tag:

[...]Standardmäßig verlangt SGML, dass alle Attributwerte entweder von doppelten Anführungszeichen (ASCII dezimal 34) oder einfachen Anführungszeichen (ASCII dezimal 39) begrenzt werden. Einfache Anführungszeichen können im Attributwert enthalten sein, wenn der Wert durch doppelte Anführungszeichen begrenzt ist und umgekehrt.[...]

Doppelte oder einfache Anführungszeichen. Das macht die Sache doch viel übersichtlicher:

```
echo "<a href='$link' id='$id' class='$class'>\$linktext</a>";
```

```
echo 'Micro$oft'; // ergibt: Micro$oft
echo "Micro$oft"; // ergibt: Micro + Inhalt der Variable $oft
echo "Micro\$oft"; // ergibt: Micro$oft

echo 'c:\temp'; // ergibt: c:\temp
```

```
echo "c:\temp"; // ergibt: c: + Tabulator + emp
echo "c:\\temp"; // ergibt: c:\temp
echo 'c:\\temp'; // ergibt: c:\temp

echo 'Kein Hochkomma: \x27'; // ergibt: Kein Hochkomma: \x27
echo " Ein Hochkomma: \x27"; // ergibt: Ein Hochkomma: '
echo ' Ein Hochkomma: \''; // ergibt: Ein Hochkomma: '

echo "<input name='foo' value='$bar'>"; // gültiges HTML
```

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_09



Last update: **2021/09/16 08:07**

1.6.1) Vergleichsoperatoren

Eine Bedingung ist eine Möglichkeit, den Ablauf eines Skripts durch Entscheidungen zu beeinflussen. In einer Bedingung werden Ausdrücke verglichen. Das Ergebnis kann entweder „Ja“ (TRUE) oder „Nein“ (FALSE) sein.

Operator	Name	Bedeutung
==	Gleichheit	\$a == \$b ergibt TRUE, wenn \$a und \$b gleich sind.
===	Identisch	\$a === \$b ergibt TRUE, wenn \$a und \$b gleich und vom selben Datentyp sind.
!=	Ungleichheit	\$a != \$b ergibt TRUE, wenn \$a und \$b ungleich sind.
<	Kleiner	\$a < \$b ergibt TRUE, wenn \$a kleiner \$b ist.
>	Größer	\$a > \$b ergibt TRUE, wenn \$a größer \$b ist.
<=	Kleiner gleich	\$a <= \$b ergibt TRUE, wenn \$a kleiner oder gleich \$b ist.
>=	Größer gleich	\$a >= \$b ergibt TRUE, wenn \$a größer oder gleich \$b ist.

Verknüpfung von Bedingungen

Operator	Name	Bedeutung
and bzw. &&	UND	\$a and \$b ergibt 1, wenn \$a und \$b 1 sind, ansonsten wird 0 zurückgegeben
or bzw.	ODER	\$a or \$b ergibt 1, wenn mindestens eine der beiden Variablen 1 ist.
xor	ausschließendes ODER	\$a xor \$b ergibt 1, wenn entweder \$a oder \$b 1 sind, aber nie beide gleichzeitig
!	NICHT	!\$a ergibt die Umkehrung des Wahrheitswertes

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_10



Last update: **2021/09/16 08:07**

1.6.2) Die einfache if-Anweisung

Syntax

```
if (Bedingung)
{
    Anweisungen;
}
```

- Die Bedingung steht in runden Klammern.
- Die Anweisungen stehen in geschwungenen Klammern.
- Liefert die Bedingung TRUE zurück, werden die Anweisungen ausgeführt, ist die Bedingung FALSE, werden die Anweisungen ignoriert
- Wird nur eine Anweisung ausgeführt, können die geschwungenen Klammern entfallen.

Beispiel: if1.php

```
<?php
$menge = 7;
if ($menge > 5)
    echo "Sie haben mehr als 5 Kilo bestellt.";
?>
```

- Der Variable \$menge wird der Wert 7 zugewiesen. Da \$menge größer als 5 ist, wird die Anweisung ausgeführt.

Beispiel: if2.php

```
<?php
$menge = 7;
if ($menge > 5)
{
    echo "Sie haben mehr als 5 Kilo bestellt.<br>\n";
    echo "Der Versand ist deswegen kostenfrei.";
}
?>
```

Die if-Anweisung mit else Zweig

Syntax

```
if (Bedingung)
{
    Anweisungsblock1;
}
```

```
else
{
    Anweisungsblock2;
}
```

- Falls die Bedingung erfüllt ist, wird Anweisungsblock1 ausgeführt, sonst Anweisungsblock2.

Beispiel: ifelse.php

```
<?php
$menge=4;
if($menge > 5)
{
    echo "Sie haben mehr als 5 Kilo bestellt.<br>\n";
    echo "Der Versand ist deswegen kostenfrei.";
}
else
{
    echo "Sie haben $menge Kilo bestellt.<br>\n";
    echo "Der Versand kostet pauschal 7,00 EUR";
}

?>
```

Verschachtelte if-Anweisungen

Syntax

```
if (Bedingung1)
{
    Anweisungsblock1;
    if (Bedingung2)
    {
        Anweisungsblock2;
    }
    else
    {
        Anweisungsblock3;
    }
}
else
{
    Anweisungsblock4;
}
```

From:

<http://elearn.bgamstetten.ac.at/wiki/> - **Wiki**

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_11



Last update: **2021/09/16 08:08**

1.6.3) Fallauswahl mit switch-Anweisung

Wenn eine Variable mit verschiedenen Werten verglichen werden soll, kann die Programmierung mit einer if-Anweisung sehr aufwändig sein. Als Alternative dazu kann man in diesem Fall die switch-Anweisung einsetzen:

Syntax

```
switch ($variable)
{
    case Wert1:
        Anweisungsblock1;
        break;
    case Wert2:
        Anweisungsblock2;
        break;
    default:
        Anweisungsblock3;
}
```

- Stimmt der Wert der Variable mit einem der angeführten Auswahlwerte überein, dann wird der Anweisungsblock unmittelbar bis zur Anweisung break ausgeführt. Die restlichen Blöcke werden nicht ausgeführt.
- Ist kein break am Ende eines case-Teils angegeben, so werden auch alle nachfolgenden case-Blöcke ausgeführt, bis eine break-Anweisung erfolgt.
- Stimmt der Wert der Variablen mit keinem der angegebenen Werte überein, wird der Anweisungsblock nach der default-Anweisung durchgeführt.

Bsp: switch-case.php

```
<?php
$sorte = "Gala";
echo "Der Preis für 1kg $sorte beträgt <br>\n";
switch($sorte) //Preiszuordnung abhängig von Sorte
{
    case "Jonagold":
        echo "1,50 EUR.";
        break;
    case "Delicious":
        echo "1,60 EUR.";
        break;
    case "Gala":
        echo "1,65 EUR.";
        break;
    case "Elstar":
        echo "2,00 EUR.";
        break;
}
```

```
default:
    echo "Diese Sorte haben wir leider nicht im Angebot.";
?>
```

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_12



Last update: **2021/09/16 08:08**

1.6.4) Aufgaben 3-7

Aufgabe 3

Apfelbestellung - Versandkosten

Erweitere die [Aufgabe 2](#) folgendermaßen:

- Wenn mehr als 10 kg Äpfel bestellt werden, ist der Versand kostenfrei, ansonsten werden 7.00 Euro verrechnet.
- Ausgabe:
 - Rückmeldung, ob Versandkosten zu bezahlen sind oder nicht.
 - Gesamtpreis.

Aufgabe 4

Minimum - Maximum

- In einem Formular (zahlen_eingabe.html) sollen drei Zahlen eingegeben werden (mittels drei Input-Feldern)
- Als Rückmeldung (minmax_ausgabe.php) wird das Maximum sowie das Minimum der drei Zahlen ausgegeben.

Aufgabe 5

Kennwortabfrage mit if

Viele Webseiten arbeiten mit einer Anmeldung des Benutzers. Um Zugang zu weiteren Teilen der Webseite zu bekommen, muss sich der Benutzer einem „Loginverfahren“ unterziehen.

- Erstelle hierfür ein Formular (kennwort.html), in welchem der Benutzer ein Kennwort eingeben kann. Verwende dazu ein `<input>`-Feld, bei dem das Kennwort bei der Eingabe nicht sichtbar ist.
- Nach Klick auf den Button „Abschicken“ soll eine Rückmeldung (kennwort_auswertung.php) erfolgen, ob das Passwort richtig eingegeben wurde.
- Baue das Formular so aus, dass vor dem Passwort auch der Nickname eingegeben werden muss. Eine positive Rückmeldung (z.B. „Zutritt erfolgreich...“) wird nur dann gegeben, wenn Nickname **und** Passwort korrekt eingegeben wurden.

Aufgabe 6

Verschachtelte if-Anweisungen

Baue die Aufgabe 3 folgendermaßen aus:

- Wenn mehr als 20 kg Äpfel bestellt werden, gibt es zur Bestellung zusätzlich ein kleines Präsent.
- Ausgabe:
 - Rückmeldung, ob Versandkosten zu bezahlen sind oder nicht und ob es ein Präsent gibt.
 - Gesamtpreis.

Aufgabe 7

switch-Anweisung

Erweitere die Aufgabe 5 folgendermaßen aus:

- Bei der Bestellung soll die bestellte Apfelart in einem `<input>`-Feld eingegeben werden.

Apfelsorte	Preis
Jonagold	1,50 EUR
Delicious	1,60 EUR
Gala	1,65 EUR
Elstar	2,00 EUR

- Ausgabe
 - Welche Äpfelsorte bestellt wurde (inkl. Preisangabe), bei falscher Eingabe soll eine entsprechende Meldung erfolgen und kein Gesamtpreis berechnet werden.
 - Gesamtpreis je nach bestellter Sorte.

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_13



Last update: **2021/09/16 08:08**

1.7.1) for-Schleife

In der for-Schleife wird genau angegeben, wie oft die Schleife durchlaufen werden soll.

Syntax

```
for (Initialisierung; Bedingung; Operation)
{
    Anweisungsblock;
}
```

Beispiel: Ausgabe der ersten n ungeraden Zahlen und die Summe dieser Folge

```
<?php
    echo "<h2>Ausgabe der ersten n ungeraden Zahlen und die Summe dieser
Folge</h2>\n";
    $n=10;
    $summe=0;
    for ($i=1;$i<=$n;$i++)
    {
        $ung=$i*2-1;
        $summe=$summe+$ung;
        echo $i.". ungerade Zahl: ".$ung."<br>\n";
    }
    echo "Die Summe der ersten $n ungeraden Zahlen beträgt: ".$summe;
?>
```

Beispiele für Operatoren in for-Schleifen

Bedingung in der for-Schleife	Werte von \$i
for (\$i=1; \$i<=5, \$i++)	1, 2, 3, 4, 5
for (\$i=1; \$i<5, \$i++)	1, 2, 3, 4
for (\$i=15; \$i>=10, \$i--)	15, 14, 13, 12, 11, 10
for (\$i=15; \$i>10, \$i--)	15, 14, 13, 12, 11
for (\$i=0; \$i<=100, \$i=\$i+10)	10, 20, 30, 40, 50, 60, 70, 80, 90, 100
for (\$i=1; \$i<=12, \$i=\$i+1.2)	1, 2.4, 3.6, 4.8, 7.2, 8.4, 9.6, 10.8, 12

Geschachtelte for-Schleifen

Mit geschachtelten for-Schleifen können zweidimensionale Strukturen abgearbeitet werden.

Beispiel: Erstellen einer Tabelle mit 3 Zeilen und 4 Spalten mittels zweier for-Schleifen

```
<?php
    echo "<table border='1'>\n";
    for ($i=1;$i<=3;$i++)
    {
        echo "<tr>\n";
        for ($j=1;$j<=4;$j++)
        {
            echo "<td> $i. Zeile, $j. Spalte </td>\n";
        }
        echo "</tr>\n";
    }
    echo "</table>\n";
?>
```

From:

<http://elearn.bgamstetten.ac.at/wiki/> - **Wiki**

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_14



Last update: **2021/09/16 08:09**

1.7.4) Foreach-Schleife

Die foreach-Schleife ermöglicht es, auf einfache Weise ein Array zu durchlaufen. foreach funktioniert nur in Verbindung mit Arrays. Wenn Sie versuchen, foreach mit einer Variablen eines anderen Datentyps oder einer nicht initialisierten Variablen zu benutzen, gibt PHP einen Fehler aus.

Definition

```
foreach (array_expression as $value) {  
    Anweisung/en;  
}
```

Mithilfe von foreach wird das Durchlaufen eines Arrays wesentlich vereinfacht. Gegenüber der while-Schleife mit list und each ist die foreach-Schleife syntaktisch deutlich im Vorteil.

Beispiel while

```
<?php  
$zahlen = array (10, 20, 30, 40);  
while (list(, $value) = each ($zahlen)) {  
    echo "Wert: $value<br>\n";  
}  
?>
```

Beispiel foreach

```
<?php  
foreach ($zahlen as $value) {  
    echo "Wert: $value<br>\n";  
}  
?>
```

Das Array wird von Anfang bis Ende durchlaufen und bei jedem Schleifendurchlauf wird das aktuelle Element der Variablen array_expression zugewiesen. Jedes Arrayelement kann wiederum ein Array sein, dann könnte mit einer verschachtelten foreach-Schleife auch dieses Array ausgewertet werden.

Beispiel - mit assoziativem Array

```
$personen = array("Matthias", "Caroline", "Gülten");  
  
foreach ($personen as $person) {  
    echo "Name: $person<br>\n";  
}
```

```
// Ausgabe:  
// Name: Matthias  
// Name: Caroline  
// Name: Gülten
```

Typischerweise ist es erlaubt, einzelne Elemente eines solchen Arrays mit unterschiedlichen Datentypen zu belegen. Das können auch weitere Arrays sein.

Erweiterte Syntax

Sollten Sie Arrays mit Schlüssel-Wert-Paaren bauen, kann foreach mit einer erweiterten Syntax diese Paare direkt auslesen. Die grundlegende Syntax lautet:

Definition

```
foreach (array as $key => $value) {  
    Anweisung/en;  
}
```

Hier wird der Operator $\Rightarrow$ eingesetzt, der schon bei der Konstruktion des Arrays verwendet wurde.

Beispiel

```
<?php  
$person = array("Vorname" => "Caroline",  
               "Nachname" => "Kannengiesser",  
               "Alter" => 25,  
               "Ort" => Berlin);  
foreach ($person as $key => $val) {  
    echo "Feld $key hat den Wert: $val<br>";  
}  
?>
```

Ausgabe

```
Feld Vorname hat den Wert: Caroline  
Feld Nachname hat den Wert: Kannengiesser  
Feld Alter hat den Wert: 25  
Feld Ort hat den Wert: Berlin
```

From:

<http://elearn.bgamstetten.ac.at/wiki/> - **Wiki**

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_14_1



Last update: **2021/09/16 08:09**

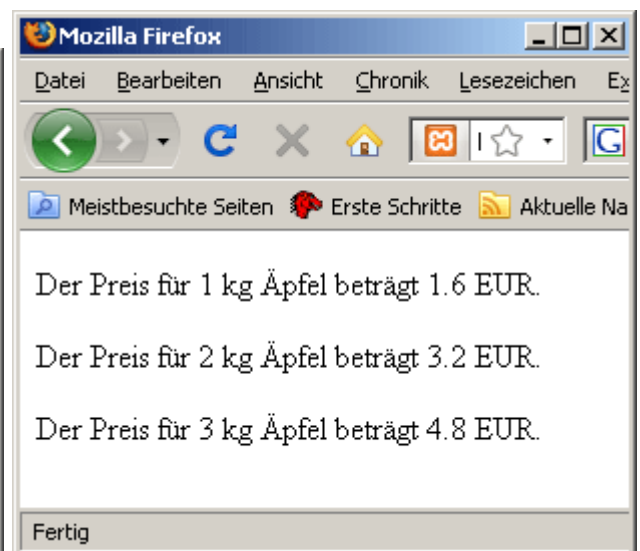
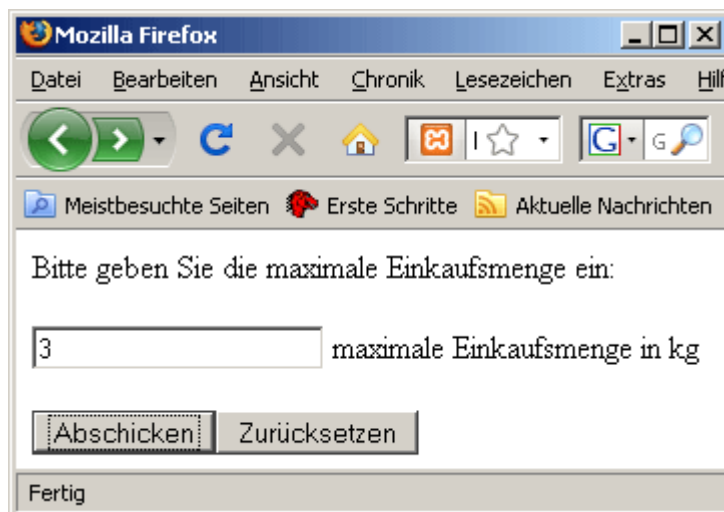
1.7.2) while-Schleife

Syntax

```
while(Bedingung)
{
    Anweisungsblock;
}
```

Bsp: while.php

```
<?php
$menge = 1;
$preis = 1.6;
while ($menge <= $_POST["maxmenge"])
{
    $gesamtpreis = $preis * $menge;
    echo "<p>Der Preis für $menge kg Äpfel beträgt $gesamtpreis EUR.</p>\n";
    $menge++;
}
?>
```



From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_15

Last update: **2021/09/16 08:09**



1.7.3) do-while-Schleife

Die do-while Schleife unterscheidet sich von der while-Schleife, dadurch, dass die Abbruchbedingung nicht am Anfang, sondern am Ende jedes Schleifendurchlaufs geprüft wird. Die Anweisung wird somit zumindest einmal durchlaufen.

Syntax

```
do
{
    Anweisungsblock;
}
while (Bedingung);
```

Bsp: dowhile.php

```
<?php
    $i=1;
    do
    {
        echo "<br>".$i*$i;
        $i++;
    }
    while ($i<=10);
?>
```

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_16



Last update: **2021/09/16 08:09**

1.7.5) Aufgabe 8 bis 11

Aufgabe 8

Erstelle ein PHP-Skript, welches eine Folge von n Zahlen erzeugt, in der jede Zahl den doppelten Wert der vorhergehenden hat.

Aufgabe 9

Erstelle ein PHP-Skript, das das „Ein-mal-eins“ für die Faktoren zwischen 11 und 20 ausgibt („großes Ein-mal-eins“). Die einzelnen Rechnungen sollen dabei in einer Tabelle aufscheinen.

Aufgabe 10

Schreibe ein PHP-Skript, das die Fibonacci-Zahlen bis zu einer eingegebenen Höchstgrenze erzeugt und ausgibt.

Eine Fibonacci-Zahl wird als Summe der beiden vorhergehenden Fibonacci-Zahlen gebildet. Die erste und die zweite Fibonacci-Zahl sind gleich 1.

Beispiel: 1, 1, 2, 3, 5, 8, 13, ...

Aufgabe 11

Teiler einer Zahl

Erstelle ein HTML- und PHP-Skript, welches die Teiler von Zahlen ermittelt. Weiters soll zusätzlich bestimmt werden, ob es sich um eine Primzahl (Anzahl der Teiler: 2) handelt

a) Version 1: Es werden die Teiler der eingegebenen Zahl ermittelt:

Ausgabe z.B.: 27 hat die Teiler 1, 3, 9, 27

b) Version 2: Es werden die Teiler sämtlicher Zahlen bis zur eingegebenen Zahl ermittelt:

Ausgabe z.B.:

...

23 hat die Teiler 1, 23 (Primzahl)

24 hat die Teiler 1, 2, 3, 4, 6, 8, 12, 24

25 hat die Teiler 1, 5, 25

26 hat die Teiler 1, 2, 13, 26

27 hat die Teiler 1, 3, 9, 27

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_17



Last update: **2021/09/16 08:10**

1.8.1) Was sind Felder?

Felder, auch Arrays oder Feldvariablen genannt, sind in der Lage, mehrere Variablen unter einem gemeinsamen Bezeichner zu speichern. Beispielsweise erhalten Sie in einer Berechnung 20 Ergebnisse. Die Ausgabe der 20 Werte soll erst erfolgen, wenn alle Berechnungen abgeschlossen wurden. Sie könnten 20 Variablen (\$ergebnis1, \$ergebnis2, \$ergebnis3, \$ergebnis4, ...) definieren, um alle Ergebnisse der Berechnungen zu speichern. Einfacher und schneller kann die Aufgabe gelöst werden, wenn die Ergebnisse der Berechnung in einem Feld abgelegt werden. Der Zugriff auf die einzelnen Ergebnisse erfolgt jeweils über ihre Position innerhalb des Feldes.



Felder in PHP

Felder unterscheiden sich in der Art, wie auf die Werte innerhalb des Feldes zugegriffen werden kann, und in ihrem Aufbau.

Auf Werte in Feldern zugreifen:

- Im numerisch indizierten Feld werden die einzelnen Werte (Value) innerhalb des Feldes über eine laufende Nummer, auch **Index oder Schlüssel** genannt, angesprochen. Der **Schlüssel** ist eine **laufende Nummer**, die automatisiert vergeben wird und bei **Null beginnt**.

```
$myArray = array();  
  
$myArray[0] = "Susi";  
$myArray[1] = "Maxi";  
  
//oder  
  
$myArray = array("Susi", "Maxi");
```

Schlüssel & Inhalt des Arrays \$myArray

key (Ort)	value (PLZ)
0	Susi
1	Maxi

- Im assoziativen Feld werden die einzelnen Werte (Value) innerhalb des Feldes über einen eindeutigen Schlüssel, Key genannt, angesprochen.

```
$myArray = array();  
$myArray['key'] = "value";  
  
//oder  
  
$staedte = array('Guetersloh' => 98000,  
                 'Bielefeld' => 326715);  
  
echo $staedte["Guetersloh"]; //gibt 98000 aus
```

Schlüssel & Inhalt des Arrays \$staedte

key (Ort)	value (PLZ)
Guetersloh	98000
Bielefeld	326715

Aufbau von Feldern

Felder können ein- oder mehrdimensional sein:

- In einem eindimensionalen Feld können Sie beispielsweise eine Liste von Städten darstellen.
- Wenn Sie z. B. eine Tabelle mit verschiedenen Feldern abbilden möchten, können Sie dies mit einem zweidimensionalen Feld tun.
- Sie haben auch die Möglichkeit, Felder mit mehr als zwei Dimensionen zu nutzen.

Eigenschaften von Feldern

- In PHP müssen die Elemente von Feldern nicht vom selben Datentyp sein.
- Felder-Indizes bei numerisch indizierten Feldern beginnen mit 0. Das erste Element hat dementsprechend den Index 0, das zweite Element hat den Index 1, das dritte Element hat den Index 2 und so weiter. Der Index eines Feldes mit \$n\$ Elementen reicht somit von bis \$n-1\$\$. Der Index eines Feldes mit vier Elementen reicht somit von 0 - 3.
- Beim Anlegen eines Feldes muss nicht angegeben werden, wie viele Elemente in diesem Feld gespeichert werden. Sie können auch nach der Erstellung dem Feld zusätzliche Elemente hinzufügen.

From:

<http://elearn.bgamstetten.ac.at/wiki/> - **Wiki**

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_18



Last update: **2021/09/16 08:10**

1.8.2) Indizierte Felder

Indizierte Felder erstellen

Ein indiziertes Feld anlegen

PHP bietet Ihnen eine einfache und schnelle Möglichkeit, Felder mit verschiedenen Werten zu füllen. Dies wird über die array-Anweisung realisiert.

Syntax und Bedeutung der array() -Anweisung

```
$Feldvariable = array(Wert1,Wert2,Wert3,...) ;
```

- Die array() -Anweisung beginnt mit dem reservierten Wort array.
- Der Variablen \$Feldvariable werden die einzelnen Werte übergeben, indem sie durch Kommata voneinander getrennt angegeben werden.
- Die Indizierung der einzelnen Elemente erfolgt automatisch in der Reihenfolge der Angabe.

Beispiel

```
$staedte = array("Frankfurt","Berlin","Bern");
```

Es befinden sich drei Werte in dem Feld \$staedte; Frankfurt, Berlin und Bern. Die Reihenfolge, in der die Werte zugewiesen worden sind, bestimmt die Indizes der Werte.

Index	0	1	2
Wert	Frankfurt	Berlin	Bern

Auf indizierte Felder zugreifen

Um auf einzelne Elemente in einem Feld zugreifen zu können, muss ihnen eine eindeutige Kennung zugewiesen werden. Bei numerisch indizierten Feldern entspricht der Kennung der automatisch erzeugte fortlaufende Index.

Auf einen bestimmten Wert eines Feldes greifen Sie zu, indem Sie zusätzlich zum Namen der Feldvariablen in den eckigen Klammern den Indexwert angeben.

```
$staedte[Index];
```

Beispiel

```
$stadt = $staedte[2] ;
```

Der Variablen `$stadt` wird der dritte Wert des Feldes `$staedte`, die Zeichenkette `Bern`, zugewiesen.

Mit indizierten Feldern arbeiten

Indizierte Felder ändern

Um einen Wert innerhalb einer Feldvariablen zu ändern, geben Sie bei der Wertzuweisung bei der Feldvariablen den entsprechenden Index des zu ändernden Wertes an und weisen Sie den neuen Wert zu.

```
$staedte[1] = "München";
```

In dem bisherigen Beispiel würde somit die Zeichenkette `Berlin` durch `München` ersetzt werden.

Indizierte Felder erweitern

Soll ein Feld um einen weiteren Wert am Ende des Feldes ergänzt werden, erzeugt PHP den entsprechenden Index automatisch.

Syntax und Bedeutung

```
$Feldvariable[] = Wert Zuweisung;
```

- Um einem Feld einen weiteren Wert (oder mehrere Werte) hinzuzufügen, brauchen Sie bei der Wertzuweisung keinen Index anzugeben. Das Feld wird am Ende um den Wert (bzw. die Werte) ergänzt und der Index automatisch erhöht.
- Sollte das Feld noch nicht bestehen, erstellt PHP automatisch ein Feld, sobald Sie der Variablen einen Wert zugewiesen haben.

Beispiel

```
$staedte = array("Frankfurt", "Berlin", "Bern");  
$staedte[] = "Graz";  
$staedte[] = "Rom";
```

ergibt:

Index	0	1	2	3	4
Wert	Frankfurt	Berlin	Bern	Graz	Rom

Anzahl der Feldelemente ermitteln

Der Befehl `COUNT($bereich)`; liefert die Anzahl der Feldelemente des Arrays `$bereich`.

From:

<http://elearn.bgamstetten.ac.at/wiki/> - **Wiki**

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_19



Last update: **2021/09/16 08:10**

1.8.3) Assoziative Felder

Assoziative Felder erstellen

Ein assoziatives Feld anlegen

Eine weitere Möglichkeit, Feldvariablen anzulegen, sind die assoziativen Felder. Bei ihnen werden für den Zugriff keine fortlaufenden Indizes benutzt, sondern Schlüssel, über deren Wert auf die einzelnen Werte zugegriffen werden kann, so genannte Schlüsselwörter.

Syntax der `array()`-Anweisung bei assoziativen Feldern

Auch die assoziativen Felder lassen sich über die `array()`-Funktion füllen.

```
$Feldvariable = array (Schlüssel1 => Wert1, Schlüssel2 => Wert2, ...);
```

Beispiel:

```
$hauptstaedte = array("Schweiz" => "Bern", "Frankreich" => "Paris");
```

- Die `array()`-Anweisung beginnt mit dem reservierten Wort `array`.
- Der Variablen werden die einzelnen Wertpaare, bestehend aus Schlüsselwort (Key) und Wert (Value), übergeben, indem sie durch Kommata voneinander getrennt angegeben werden.
- Die Indizierung der einzelnen Werte erfolgt über die Angabe eines Schlüssels in Form einer beliebigen, eindeutigen Zeichenkette oder eindeutiger Integerwerte. Die Zuweisung erfolgt über `=>`.

Auf assoziative Felder zugreifen

Beim Zugriff auf einen Wert des Feldes kann der Schlüsselwert direkt angegeben oder als Variable übergeben werden:

	Ergebnis
<code>\$stadt = \$hauptstaedte["Schweiz"];</code>	Variable <code>\$stadt</code> hat den Wert „Bern“

oder:

	Ergebnis
<code>\$k = "Schweiz"; \$stadt = \$hauptstaedte[\$k];</code>	Variable <code>\$stadt</code> hat den Wert „Bern“

Beispiel: `arr_assoz.php` Hier wurde die Zuweisung der Städtenamen mit der `array()`-Anweisung realisiert.

```
<html>
<body>
<?php
$hauptstaedte = array("Schweiz" => "Bern",
                      "Frankreich" => "Paris",
                      "Deutschland" => "Berlin",
                      "Österreich" => "Wien");

$k = "Österreich";
echo "<p>Hauptstadt von $k: " . $hauptstaedte[$k] . "</p>";
echo "Die Hauptstadt von $k ist " . $hauptstaedte["Österreich"];
?>
</body>
</html>
```

- Das assoziative Feld wird über die `array()`-Funktion und die Zuweisung der Schlüssel sowie Feldinhalte angelegt.
- Der Variablen `$k` wird die Zeichenkette Österreich zugewiesen.
- Mithilfe der Variablen `$k` wird das entsprechende Element aus dem Feld ausgelesen und angezeigt.
- Hier wird das Element durch die direkte Angabe des Schlüssels herausgefiltert. Die Bezeichnung des Schlüssels wird als Zeichenkette in den eckigen Klammern angegeben.

Mit assoziativen Feldern arbeiten

Daten an Variablen übergeben

Mit einer `foreach()`-Schleife sind Sie in der Lage, die Werte der Feldelemente einzeln auszulesen. Dabei wird jedes Element in einer neuen Variable zwischengespeichert.

```
foreach($Feld as $wert) Anweisung;
foreach($Feld as $index => $wert) Anweisung;
```

- Die Funktion `foreach()` erwartet die Angabe des assoziativen Feldes, dessen Elemente durchlaufen werden sollen.
- In der ersten Variante werden mithilfe von `foreach` bei jedem Schleifendurchlauf der Variablen (`$wert`) nacheinander die Werte der Arrayelemente zugewiesen.
- Mithilfe der zweiten Variante können Sie zusätzlich den Index des Arrayelements auslesen und der Variablen `$index` zuweisen, was beispielsweise bei assoziativen Feldern hilfreich ist.

Beispiel: `foreach.php`

Verschiedene Länder der Welt und deren Hauptstädte werden in einem assoziativen Feld gespeichert. Zur Ausgabe sollen das Land (= Schlüssel) und die dazugehörige Hauptstadt (= Wert) in einer Tabelle angezeigt werden:

```
<?php

$hauptstaedte = array("Schweiz" => "Bern",
```

```
"Frankreich" => "Paris",
"Deutschland" => "Berlin");
$hauptstaedte["Polen"] = "Warschau";
$hauptstaedte["Italien"] = "Rom";
$hauptstaedte["Österreich"] = "Wien";

echo "<table border>";
echo "<tr><td><b>Land</b></td>" ;
echo "<td><b>Hauptstadt</b></td></tr>" ;
foreach($hauptstaedte as $land=>$stadt)
{echo "<tr><td>$land</td><td align='left'>$stadt</td></tr>" ;
}
echo "</table>";

?>
```

- Das assoziative Feld \$hauptstaedte wird über die array-Anweisung gefüllt.
- Dem assoziativen Feld \$hauptstaedte werden weitere Elemente hinzugefügt.
- Der Tabellenkopf wird erstellt.
- Mithilfe einer foreach-Schleife können Sie auf jedes Element des Feldes zugreifen. Die Schleife wird so lange durchgeführt, bis alle Elemente des Feldes angesprochen wurden. In jedem Schleifendurchlauf wird der Variablen \$land der jeweilige Schlüsselwert und der Variablen \$stadt der Wert des Feldes an der durch das Schlüsselwort festgelegten Position zugeordnet.
- Die Werte werden in die Tabelle eintragen und über den Ausgabebefehl echo am Bildschirm ausgegeben.



From:

<http://elearn.bgamstetten.ac.at/wiki/> - **Wiki**

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_20



Last update: **2021/09/16 08:11**

1.8.4) Mehrdimensionale Felder

Mehrdimensionale indizierte Felder erstellen

Grundlagen zu mehrdimensionalen Feldern

Im folgenden Beispiel sehen Sie mehrere Angaben (Vorname, Nationalität und Alter), die jeweils einen Eintrag in einer Feldvariablen bilden. Damit ist dieses Feld ein mehrdimensionales Feld. Bei jeder der Angaben sind Mehrfachwerte möglich. Mehrdimensionale Felder sind verschachtelte Felder, man spricht hier von äußeren und inneren Feldern.

Mehrdimensionale Felder können sowohl numerisch indiziert als auch assoziativ sein. Eine Mischform aus numerisch indizierten und assoziativen Feldern ist ebenfalls möglich. Im Beispiel bietet sich durch mögliche Mehrfachwerte z. B. ein indiziertes Feld an, das jeden Satz an Angaben mit einem eindeutigen numerischen Index versieht.

Index 1	Index 2		
	0	1	2
0	Oliver	spanisch	37 Jahre
1	Maria	deutsch	23 Jahre
2	Oliver	englisch	46 Jahre

In dieser Tabelle wird bereits die Arbeitsweise mehrdimensionaler Felder ersichtlich. Um beispielsweise an die Informationen von Maria zu gelangen, suchen Sie die Zeile mit der Angabe Maria und lesen die einzelnen Werte dieser Zeile aus: Maria, deutsch, 23 Jahre.

Mit mehrdimensionalen indizierten Feldern arbeiten

Ein mehrdimensionales indiziertes Feld hat statt *eines* Indexes - in Abhängigkeit von der Anzahl der Verschachtelungen - *mehrere* Indizes.

Syntax eines mehrdimensionalen indizierten Feldes

Die bereits erwähnten Informationen sollen im Folgenden als mehrdimensionales Feld realisiert werden.

Auf eine bestimmte Angabe in einem mehrdimensionalen indizierten Feld zugreifen

Ausschnitt aus der Beispieldatei „mehrdimensional.php“:

```
$person = array(array("Oliver", "spanisch", "37 Jahre"),
                  array("Maria", "deutsch", "23 Jahre"),
```

```

        array("Oliver", "englisch", "46 Jahre"));
$vorname1 = $person[1][0] ;
$nationalitaet1 = $person[1][1] ;
$alter1 = $person[1][2] ;

```

Ergebnis:

```

$vorname1 = "Maria";
$nationalitaet1 = "deutsch";
$alter1 = "23 Jahre".

```

Ein mehrdimensionales indiziertes Feld erweitern

```

$person[3][0] = "Johanna";
$person[3][1] = "schwedisch";
$person[3][2] = "19 Jahre";

```

oder alternativ:

```

$person[] = array("Johanna", "schwedisch", "19 Jahre");

```

Mit mehrdimensionalen assoziativen Feldern arbeiten

Syntax eines mehrdimensionalen assoziativen Feldes

Die assoziativen Felder besitzen zum Ansprechen der einzelnen Elemente einen sogenannten Schlüssel. Um ein mehrdimensionales Feld anzulegen, wird jedem Schlüssel ein weiteres Feld mit Schlüssel-Wert-Paaren übergeben (Sonderfall zweidimensional). Verwenden Sie aus Gründen der Übersichtlichkeit und Nachvollziehbarkeit der Programmierung nicht mehr als drei Dimensionen bzw. Verschachtelungen.

Beispiel: mehrdimensional.php

```

$land = array("Spanien" => array("Hauptstadt" => "Madrid",
                                "Sprache" => "Spanisch",
                                "Waehrung" => "Euro",
                                "Flaeche" => "504645 qkm"),
              "England" => array("Hauptstadt" => "London",
                                "Sprache" => "Englisch",
                                "Waehrung" => "Pfund Sterling",
                                "Flaeche" => "130395 qkm"),
              "Portugal" => array("Hauptstadt" => "Lissabon",
                                "Sprache" => "Portugiesisch",
                                "Waehrung" => "Euro",
                                "Flaeche" => "92345 qkm"));

```

Hiermit ist jeder Satz an Eintragungen mit entsprechenden Länderinformationen über einen Schlüssel mit aussagekräftiger Bezeichnung - Spanien, England oder Portugal - direkt ansprechbar.

Auf eine bestimmte Angabe in einem mehrdimensionalen assoziativen Feld zugreifen

Ausschnitt aus Beispieldatei, mehrdimensional.php

```
$hauptstadt2 = $land["Portugal"]["Hauptstadt"];  
$sprache2    = $land["Portugal"]["Sprache"];  
$waehrung2    = $land["Portugal"]["Waehrung"];  
$flaeche2     = $land["Portugal"]["Flaeche"];
```

Ergebnis:

```
$hauptstadt2 = "Lissabon";  
$sprache2    = "Portugiesisch";  
$waehrung2    = "Euro";  
$flaeche2     = "92345 qkm";
```

Ein mehrdimensionales assoziatives Feld erweitern

```
$land["Ungarn"]["Hauptstadt"] = "Budapest";  
$land["Ungarn"]["Sprache"]    = "Ungarisch";  
$land["Ungarn"]["Waehrung"]    = "Forint";  
$land["Ungarn"]["Flaeche"]     = "93036 qkm";
```

oder alternativ:

```
$land["Ungarn"] = array("Hauptstadt" => "Budapest",  
                        "Sprache"     => "Ungarisch",  
                        "Waehrung"    => "Forint",  
                        "Flaeche"     => "93036 qkm");
```

Daten aus mehrdimensionalen Feldern extrahieren

Bei der Arbeit mit mehrdimensionalen Feldern kommt es häufig vor, dass Sie die kompletten Daten eines inneren Feldes benötigen. Mit einer `foreach`-Schleife können Sie alle Werte einer Feldvariablen auslesen. Durch die Verwendung einer `foreach`-Schleife müssen Sie keinen Index oder Schlüssel kennen. Bei jedem Schleifendurchlauf wird das aktuelle Element in einer von Ihnen angegebenen Variablen zwischengespeichert. Hierbei handelt es sich auch um eine Feldvariable mit Angaben zum aktuellen „Datensatz“.

Syntax:

```
foreach($feld as $wert);
```

Um die einzelnen Elemente auszulesen und jeden Wert verschiedenen Variablen zu übergeben, verwenden Sie innerhalb der Schleife die `list()`-Funktion.

Syntax

```
list($variable1, $variable2, ...) = $wert;
```

- Der Funktion `list()` wird das entsprechende Element übergeben, das in die angegebenen Variablen aufgeteilt werden soll.
- Die Variablen werden nacheinander, durch Kommata voneinander getrennt, in den runden Klammern der Funktion `list()` angegeben.
- Die Anzahl und die Position der Variablen richten sich nach der Anzahl der Elemente im Feld.
- Bei der an dieser Stelle verwendeten Variablen `$wert` handelt es sich um die Variable, die als letztes Argument in der `foreach()`-Schleife definiert wurde.

Beispiel: `array_zdim_list.php`

Verschiedene Angaben zu Ländern werden in einem mehrdimensionalen Feld gespeichert. Zur Ausgabe sollen der Name des Landes, die Hauptstadt, die Sprache und die Landeswährung in separaten Variablen abgelegt werden.

```
<body>
<table width="500">
<tr>
<td width="125"><b>Land</b></td>
<td width="125"><b>Hauptstadt</b></td>
<td width="125"><b>Sprache</b></td>
<td width="125"><b>W&auml;hrung</b></td>
</tr>

<?php
    $staedte = array(
        "Japan"           => array("Tokio", "Japanisch", "Yen"),
        "Niederlande"     => array("Amsterdam", "Niederländisch", "Euro"),
        "Polen"            => array("Warschau", "Polnisch", "Z&#322;oty"),
        "Indien"           => array("Neu Delhi", "Indisch", "Rupie"),
        "Island"           => array("Reykjavik", "Isländisch", "Krone"),
        "Italien"          => array("Rom", "Italienisch", "Euro"),
        "Frankreich"       => array("Paris", "Französisch", "Euro"),
        "Spanien"          => array("Madrid", "Spanisch", "Euro"),
        "England"          => array("London", "Englisch", "Pfund Sterling")
    );
```

```
②      foreach($staedte as $key=>$ausgabe)
        {
③          list($hauptstadt, $sprache, $waehrung) = $ausgabe;
④          echo "<tr><td>" . $key . "</td>";
          echo "<td>" . $hauptstadt . "</td>";
          echo "<td>" . $sprache . "</td>";
          echo "<td>" . $waehrung . "</td></tr>";
        }

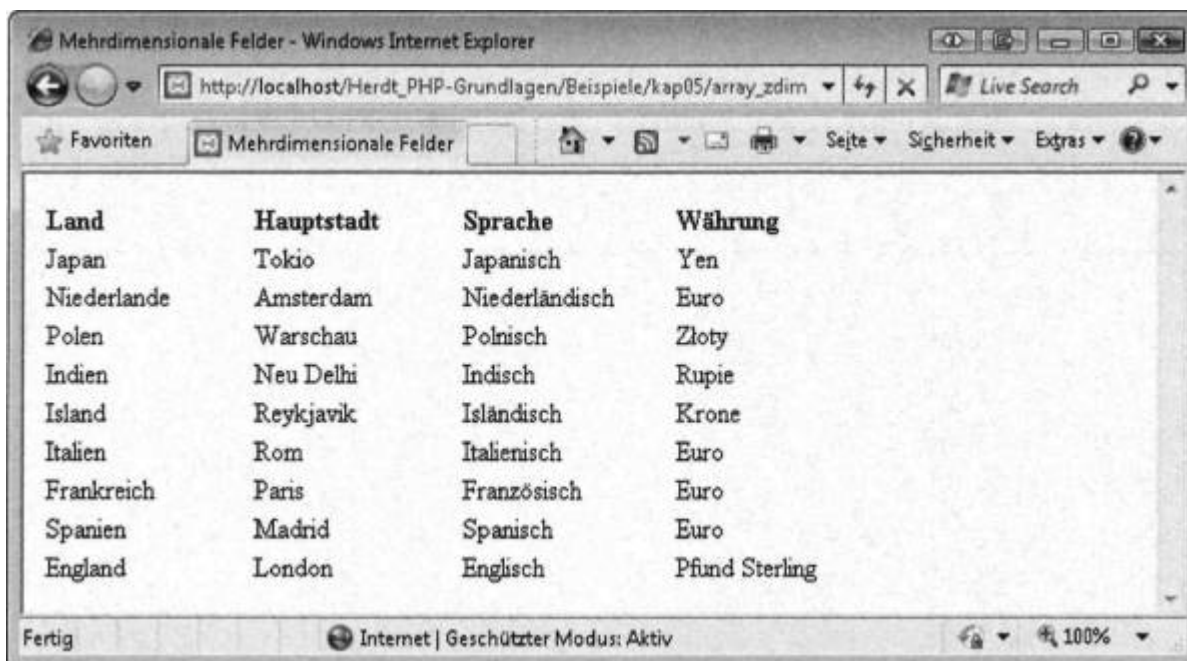
    ?>
</table>
</body>
```

(1) Das assoziative Feld `$staedte` wird neu definiert und mit Werten gefüllt. Die Daten enthalten Informationen zur Hauptstadt eines Landes, zur Sprache und zur im Land verwendeten Währung. Der Name des Landes selbst wird als eindeutiger Schlüssel verwendet.

(2) Mithilfe einer `foreach`-Schleife können Sie auf jedes Element des Feldes zugreifen, indem die Schleife so oft durchgeführt wird, bis auch das letzte Element angesprochen wurde. Da es sich in diesem Beispiel um mehrere Werte pro Element handelt, müssen die zusammenhängenden Werte eines Landes in dem neuen Feld `$ausgabe` zwischengespeichert werden. Im ersten Schleifendurchlauf enthält die Variable `$ausgabe` nur die Werte des zuerst definierten Landes Japan (Tokio, Japanisch, Yen), im zweiten Schleifendurchlauf die Werte der Niederlande etc.

(3) Über den Befehl `list()` werden die Daten zum gerade abgerufenen Land (Hauptstadt, Sprache und Währung) in die entsprechenden Variablen `$hauptstadt`, `$sprache` und `$waehrung` aufgeteilt.

(4) Über den Ausgabebefehl `echo` werden die Werte der Variablen am Bildschirm ausgegeben. Bei der Variablen `$key` handelt es sich um den Schlüssel (Landesnamen), der in (2) definiert wurde.



The screenshot shows a web browser window titled "Mehrdimensionale Felder - Windows Internet Explorer". The address bar shows the URL `http://localhost/Herdt_PHP-Grundlagen/Beispiele/kap05/array_zdim`. The page displays a table with the following data:

Land	Hauptstadt	Sprache	Währung
Japan	Tokio	Japanisch	Yen
Niederlande	Amsterdam	Niederländisch	Euro
Polen	Warschau	Polnisch	Zloty
Indien	Neu Delhi	Indisch	Rupie
Island	Reykjavik	Isländisch	Krone
Italien	Rom	Italienisch	Euro
Frankreich	Paris	Französisch	Euro
Spanien	Madrid	Spanisch	Euro
England	London	Englisch	Pfund Sterling

Beispiel aus SÜ

```
<?php
// Nicht assoziatives Array
$matrix = array (
    array(1,2,3,4),
    array(5,6,7,8),
    array(9,10,11,12),
    array(13,14,15,16),
    array(17,18,19,20)
);
// Ausgabe als Zeilen und Spalten
for ($i=0;$i<count($matrix);$i++){
    for ($j=0;$j<count($matrix[$i]);$j++) {echo $matrix[$i][$j].' ';}
    echo '<br>';
}

// Assoziatives Array
$user = array(
    'person'=>array('name'=>'Thomas', 'nickname'=>'Baumi'),
    'lang'
=>array('deutsch','mostviertlerisch','english','spanisch','französisch'),
    'kontakt'=>array('email'=>'thomas.baumgartner@bgamstetten.ac.at',
'web' =>'www.baumi.net')
);

// Ausgabe
foreach ($user as $eigenschaft => $arrayinhalt) {
    echo $eigenschaft.':';
    foreach ($arrayinhalt as $index =>$wert)
        echo ' '.$index.' ist '.$wert.' | ';
    echo '<br>';
}
?>
```

Verwenden des passenden Feldtyps

Der passende Feldtyp hängt von den Daten ab, die Sie in der Feldvariablen speichern wollen. Die folgende Tabelle soll Ihnen bei der Auswahl des Feldvariablentyps helfen:

Daten	Anzahl an Informationen pro Eintrag	Bevorzugter Feldtyp
Einfache Liste (z. B. Namen, Länder, Auto-marken etc.)	1	✓ eindimensional/indiziert
Wertepaare (z. B. Länder - Hauptstädte, Artikel - Preis etc.)	2	✓ eindimensional/assoziativ, sofern ein Wert eindeutig ist (= Schlüssel), sonst ✓ mehrdimensional/indiziert oder assoziativ
Mehrere evtl. mehrfach verschachtelte Werte (z. B. Mitarbeiter und ihre Stammdaten, Artikelkategorien - Artikel - Artikeldetails etc.)	> 2	✓ mehrdimensional/indiziert oder assoziativ

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_21Last update: **2021/09/16 08:11**

1.8.5) Weitere Informationen zu Feldern in PHP

Felder begegnen uns in PHP nicht nur, wenn wir selbst Feldvariablen definieren. PHP arbeitet intern automatisch mit Feldern, und zwar bei folgenden Aktionen:

- Die in einem HTML-Formular übermittelten Daten speichert PHP in einer Feldvariablen.
- Wenn man mit Sessions arbeitet, werden die zur Session gehörigen Daten automatisch in einem Feld abgelegt.
- Bei Abfragen aus Datenbanken befinden sich die Ergebnisse ebenfalls üblicherweise in einer Feldvariablen.

Weitere Feld-Funktionen

PHP kennt annähernd 100 Funktionen für den Umgang mit Feldern. Für nahezu jede Fragestellung liefert PHP eine passende Funktion. Folgende Aufgabengebiete werden abgedeckt:

- Auslesen
- Auswerten (u.a. Dopplungen finden)
- Sortieren
- Suchen
- Teilen und Zusammensetzen
- Verändern (u.a. Füllen, Hinzufügen, Löschen)

Eine Auswahl zusätzlicher wichtiger Feldfunktionen wird kurz in der nachfolgenden Tabelle vorgestellt:

Feldfunktion	Beschreibung
<code>array_flip(\$feld)</code>	Im Feld werden Indizes mit Werten vertauscht.
<code>array_key_exists(wert,\$feld)</code>	Prüfung, ob ein Schlüssel in einem Feld vorhanden ist.
<code>array_keys(\$feld)</code>	Liefert die Indizes des angegebenen Feldes zurück.
<code>array_merge(\$feld1,\$feld2...)</code>	Fügt die Elemente mehrerer Felder zu einem Feld zusammen.
<code>array_push(\$feld,werte)</code>	Das Feld wird um den oder die angegebenen Werte am Ende des Feldes erweitert.
<code>array_search(wert,\$feld)</code>	Das Feld wird nach dem angegebenen Wert durchsucht.
<code>array_sum(\$feld)</code>	Addiert die Werte des Feldes und liefert das Ergebnis zurück.
<code>array_unique(\$feld)</code>	Es wird ein neues Feld erstellt, aus dem doppelte Werte des angegebenen Feldes gelöscht wurden.
<code>array_values(\$feld)</code>	Alle Werte des Feldes werden zurückgeliefert.
<code>count(\$feld)</code>	Gibt die Anzahl der Elemente des Feldes zurück.
<code>sort(\$feld)</code> <code>rsort(\$feld)</code>	Sortiert die Werte des angegebenen Feldes aufsteigend bzw. absteigend.

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_22



Last update: **2021/09/16 08:11**

1.8.6) Aufgaben 12 bis 18

Aufgabe 12 - Indizierter Array

1. Schreibe mittels PHP-Code ein „indiziertes Array“, das folgende Elemente auflistet:

- Englisch
- Informatik
- Deutsch
- Geografie
- Bewegung und Sport

2. Gib alle eingegebenen Fächer aus.

3. Schreibe folgenden Satz im PHP-Block und greife auf den Array zu, um ihn zu vervollständigen:
„Mein Lieblingsfach ist [.....] „

Aufgabe 13 - Assoziativer Array

1. Schreibe mittels PHP-Code ein „assoziatives Array“. Beziehe folgende Werte ein:

- Hauptfach - Englisch
- Nebenfach - Geschichte

2. Schreibe die folgenden Sätze im PHP-Block und greife auf den Array zu, um den Satz zu vervollständigen:

[Englisch] ist ein Hauptfach. [Geschichte] ist ein Nebenfach.

Aufgabe 14 - Jahreszeit

Erstelle ein Dokument namens jahreszeit.php. Hier sollen die vier Jahreszeiten in einer Werteliste namens

```
$jahreszeit[0]
```

abgebildet werden können. Nutze dafür die **Kurzform**, indem du alle Werte auf einmal zuweist. Gib den Herbst aus.

Aufgabe 15 - Monatsnamen

Erstelle ein Dokument namens monat.php. Speichere alle Monatsnamen in einem Array namens \$monat. Nutze dafür die **Langform**, indem du die Monate einzeln zuweist! Gib sodann folgenden

String aus:

„Ein Jahr hat 12* Monate. Der aktuelle Monat ist der Mai.“ (* Zäihlfunktion einsetzen)

Überlege dabei, welcher Arraytyp sinnvoll ist (indiziert oder assoziativ)!

Aufgabe 16 - Zufallszahlen

Ein Array soll mit einer vorgegebenen Anzahl von Zufallszahlen zwischen den Grenzen ug und og belegt werden. Dabei können sich die Zahlen wiederholen.

Zufallszahlen	
Bitte geben Sie die gewünschte Anzahl der Zufallszahlen bzw. die obere und untere Grenze ein:	
Anzahl der Zufallszahlen:	5
Untere Grenze:	10
Obere Grenze:	20
Berechnung starten Zurücksetzen	
<i>Folgende Zufallszahlen wurden ermittelt:</i>	
Untergrenze 10	Obergrenze 20
Zufallszahlen: 18 15 16 12 11	

Hinweis: Wie man eine Zufallszahl zwischen den Grenzen ug und og zu erhalten kann, zeigt die folgende Zeile:

```
$zahl[$i]=rand($ug,$og);
```

1. Nach der Generierung der Zufallszahlen soll das Array ausgegeben werden.
2. Der Mittelwert aller Zahlen soll ausgegeben werden.
3. Die Standardabweichung soll ausgegeben werden.

Aufgabe 17 - Primzahlenberechnung mit dem Sieb des Erathostenes

Zur Berechnung aller Primzahlen bis zu einer Obergrenze n gibt es ein Verfahren, das vom griechischen Mathematiker *Erathostenes* stammt. Es wird das *Sieb des Erathostenes* genannt und basiert auf der Idee, dass die Vielfachen einer Primzahl mit Sicherheit keine Primzahlen sind. Wenn wir z.B. wissen, dass 2 eine Primzahl ist, können wir alle Vielfache von 2 aus der Menge der Primzahlenkandidaten streichen. Die kleinste Zahl der dann verbleibenden Menge ist die nächste Primzahl. Wir eliminieren wiederum alle ihre Vielfache.

Implementieren lässt sich dieser Algorithmus sehr schön mit Hilfe eines Arrays, das zunächst lauter „p“ (Primzahl) in den Feldern stehen hat. Wird eine Zahl gestrichen, ersetzt man das „p“ durch ein „z“

(zusammengesetzte Zahl).

Die Primzahlen sind dann die Indizes jener Felder, in denen „p“ steht.

Aufgabe 18 - Leichtathletik

Erstelle eine neue Datei unter dem Namen `uebung_mehrdim.php` mit einem mehrdimensionales indiziertes Feld mit folgenden Inhalten und gib die Daten anschließend in Tabellenform (mit Überschrift) auf dem Bildschirm aus:

Beginn	Disziplin	Ort	Bemerkung
09:30 Uhr	Diskuswurf	Nebenplatz	Jugendmeisterschaften
10:00 Uhr	5-km-Lauf	Stadion - Laufbahn	Offener Lauf
11:00 Uhr	Halbmarathon	Waldgebiet	Teilnahme ab 18 Jahren
12:00 Uhr	Stabhochsprung	Stadion - Stabhochsprunganlage	Nur Frauen

From:

<http://elearn.bgamstetten.ac.at/wiki/> - Wiki

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_23



Last update: **2021/09/16 08:12**

1.8.7) Fragen zu Feldern

- Erkläre die Begriffe „eindimensionales“ und „mehrdimensionales Feld“ und gib jeweils ein sinnvolles bzw. typisches Beispiel an.
- Erkläre die Begriffe „indiziertes“ und „assoziatives Feld“. Worin besteht der Unterschied.
- Wie kann man bei indizierten bzw. bei assoziativen Felder auf die Feldwerte zugreifen?
- Welche typischen Befehle für den Zugriff auf Feldern sind dir bekannt?

From:

<http://elearn.bgamstetten.ac.at/wiki/> - **Wiki**

Permanent link:

http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf7bi8bi_202122:1:1_24



Last update: **2021/09/16 08:12**