PHP 8.0

Dynamische Webseiten erstellen

Stephan Heller

1. Ausgabe, Mai 2021

ISBN 978-3-86249-992-2

GPHP8

Plus

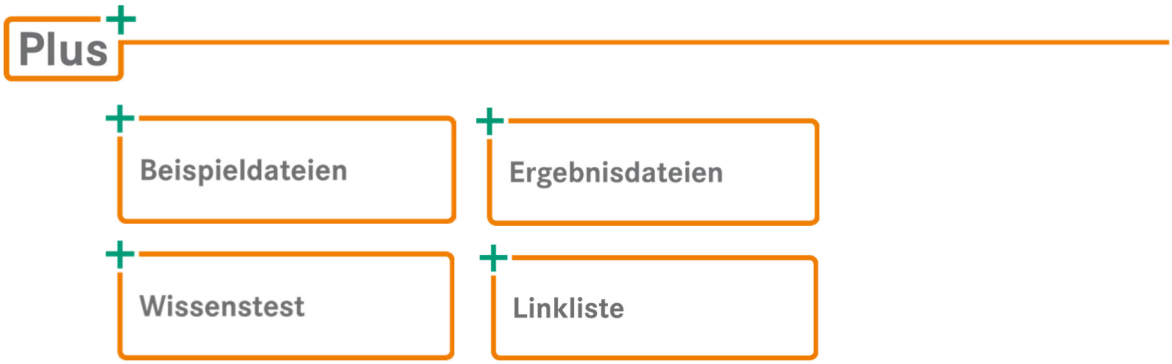
HERDT

Bevor Sie beginnen …

HERDT BuchPlus – unser Konzept:

Problemlos einsteigen – Effizient lernen – Zielgerichtet nachschlagen

Nutzen Sie dabei unsere maßgeschneiderten, im Internet frei verfügbaren Medien:



Wie Sie schnell auf diese BuchPlus-Medien zugreifen können, erfahren Sie unter *www.herdt.com/BuchPlus*.

Empfohlene Vorkenntnisse

- Umgang mit Betriebssystemen, Webbrowser
- HTML-Grundlagen (mindestens Zeichen- und Absatzformatierung, Tabellen, Formulare)
- Grundkenntnisse Datenbanken (MariaDB/MySQL)

Hinweise zu Soft- und Hardware

PHP ist eine Open-Source-Software. Sie benötigen zur Erstellung von Webseiten mit PHP-Anweisungen nicht mehr als einen Texteditor. Damit Sie Ihre PHP-Programme direkt an Ihrem Rechner testen können, empfehlen wir Ihnen folgende Software:

- die kostenfreie Entwicklungsumgebung XAMPP in der Version 8.0.1 vom 5. Januar 2021 (*https://www.apachefriends.org/de/*), unter anderem mit den Bestandteilen:
  - PHP, Version 8.0.1
  - Apache-Webserver, Version 2.4.46
  - MariaDB 10.4.17

  Die XAMPP-Software wurde in allen Beispielen mit den Standardeinstellungen verwendet. Installations- und Konfigurationshinweise zur Software finden Sie im Anhang.
- den Freeware-Texteditor Notepad++ in der Version 7.9.2 (*https://notepad-plus-plus.org/downloads/*);
- einen Webbrowser.

In diesem Buch sind alle Screenshots mit dem Mozilla Firefox-Browser auf Windows 10 erstellt. Grundsätzlich spielt der Browser für das Arbeiten mit PHP keine Rolle. Sie können ebenfalls Google Chrome, Microsoft Edge, Apple Safari oder auch Opera verwenden.

1

Einführung in PHP

Plus **Beispieldateien:** Dateien im Ordner *Kapitel_01*

1.1 PHP-Code in Webseiten

Was ist PHP?

PHP (Abkürzung für **P**HP: **H**ypertext **P**reprocessor, früher auch **P**ersonal **H**ome **P**age Tools) ist eine **Open- Source-Skriptsprache**, die speziell für den Einsatz im **Internet** entwickelt wurde.

Die Stärken von PHP liegen in der recht leichten Erlernbarkeit und in der breiten Funktionspalette.

PHP setzt dort an, wo HTML seine Grenzen erreicht: HTML-Seiten sind starr. Mithilfe von PHP können auf einer Webseite

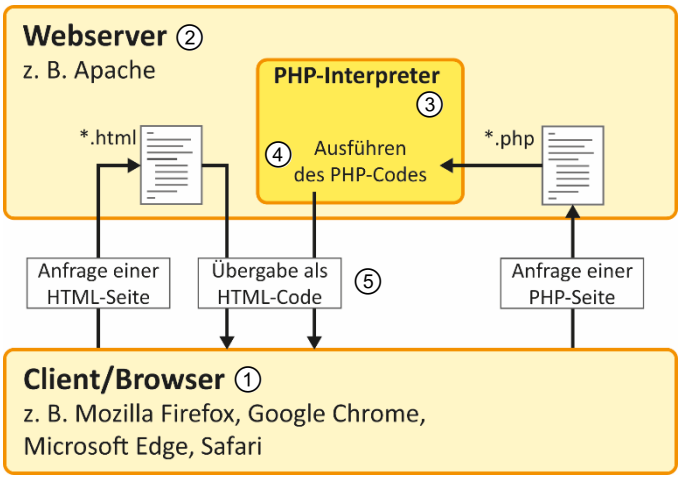
- Interaktionen eingebaut,
- Datenbanken gesteuert oder
- die Seite individuell an das Benutzerverhalten angepasst werden.

Webseiten mit PHP-Code verarbeiten

PHP-Code wird von einem Webserver, also serverseitig verarbeitet. Das Ergebnis wird danach vom Webserver als HTML-Code (Quellcode) an den Browser gesendet. Ob der HTML-Code aus einer statischen HTML-Datei stammt oder das Ergebnis eines ausgeführten PHP-Skripts ist, ist für den Browser nicht ersichtlich. Zur Darstellung erhält der Browser in beiden Fällen HTML-Code.

Öffnet der Betrachter eine PHP-Webseite im Browser ①,

- werden die Anweisungen von PHP auf dem Server ② interpretiert ③,
- ausgeführt ④ und
- das Ergebnis als HTML-Code an den Browser zurückgesendet ⑤.

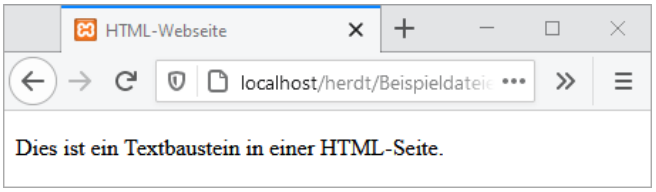


Der Nutzer sieht im Browser nur die Darstellung des zurückgelieferten HTML-Codes (Resultat nach den Vorgängen ③ und ④). Der PHP-Quellcode selbst ist dem Betrachter nicht zugänglich. Programmieralgorithmen, die Verarbeitung von Daten oder beispielsweise Zugriffe auf Datenbanken bleiben dem Nutzer verborgen.

Webseite ohne PHP-Code – Beispiel: *html-inhalt.html*

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>HTML-Webseite</title>
  </head>
  <body>
    <p>Dies ist ein Textbaustein in einer HTML-Seite.</p>
  </body>
</html>
```

HTML-Dateien können ohne weitere Software von jedem beliebigen Browser angezeigt werden. Es reicht ein Doppelklick auf den Dateinamen, um die Datei zu öffnen. In der Adresszeile des Browsers erscheint der Pfad des Dateisystems, in dem die HTML-Datei liegt.



Webseite um PHP-Code erweitern

- **Schritt 1:** Kopieren Sie die Datei *html-inhalt.html* und ändern Sie den Dateinamen in *html-inhalt-und-php.php*.

  Achten Sie auf die Angabe der Dateinamenerweiterung *php*. Sobald die Dateinamenerweiterung von *html* in *php* geändert ist, haben Sie eine lauffähige PHP-Datei erzeugt.

  PHP-Dateien benötigen im Gegensatz zu HTML-Dateien einen Webserver, der PHP verarbeitet (interpretiert). Webserver werden von Internet-Providern zur Verfügung gestellt. Mit dem XAMPP-Paket (vgl. Abschnitt A.1) können Sie einen lokalen Webserver für die Entwicklung von PHP auf Ihrem eigenen Rechner installieren.

Die Ausgabe der Datei *html-inhalt-und-php.php* ist identisch mit der Datei *html-inhalt.html*. Beide Dateien enthalten den gleichen HTML-Code und sonst keine weiteren Inhalte. HTML-Code wird vom PHP-Interpreter weder interpretiert noch ausgeführt, sondern direkt an den Browser gesendet und dort dargestellt.

Über PHP-Code soll zusätzlich der Text `Ich freue mich auf PHP!` ausgegeben werden:

- **Schritt 2:** Öffnen Sie die Datei *html-inhalt-und-php.php* im Editor (z. B. Notepad++) und fügen Sie PHP-Code in den HTML-Code ein:

Beispiel: *html-inhalt-und-php.php*

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>HTML-Webseite mit PHP</title>
  </head>
  <body>
    <p>Dies ist ein Textbaustein in einer HTML-Seite.<br>
      <?php
      echo "Ich freue mich auf PHP!";
      ?>
    </p>
  </body>
</html>
```

① Fügen Sie hinter dem vorhandenen Text einen **Zeilenumbruch** `<br>` ① ein, um die Ausgabe, die vom HTML- und PHP-Code erzeugt wird, in zwei Zeilen zu bewirken.

② Beginnen Sie einen PHP-Block im HTML-Code mit dem öffnenden PHP-Tag `<?php` ②.

③ Verwenden Sie zur Ausgabe des Textes den PHP-Befehl `echo`, gefolgt vom Text in Anführungszeichen. Befehle werden in PHP durch ein **Semikolon** abgeschlossen.

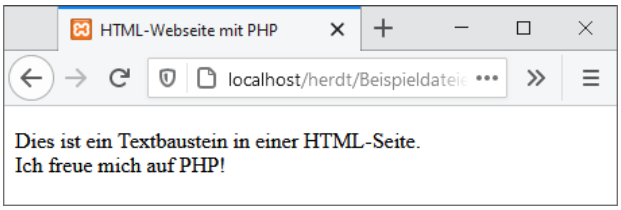
④ Beenden Sie den PHP-Block mit einem schließenden PHP-Tag `?>` ④.

PHP-Code kann an jeder **beliebigen** Stelle und **beliebig** oft im HTML eingefügt werden. Wichtig ist, dass jeder PHP-Block von dem öffnenden PHP-Tag **`<?php`** und dem schließendem PHP-Tag **`?>`** umschlossen ist.

- **Schritt 3:** Starten Sie den XAMPP-Webserver über das Control Panel, falls dieser nicht als Dienst installiert ist (vgl. Installationshinweise Abschnitt A.1). Rufen Sie die PHP-Datei im Browser mit folgender URL auf: *http://localhost/[Pfad zur Datei]/html-inhalt-und-php.php* bzw. *http://localhost:8080/[Pfad zur Datei]/html-inhalt-und-php.php* auf einem Mac.

  Falls Sie die Datei *html-inhalt-und-php.php* direkt im Ordner */htdocs* des Webservers gespeichert haben, lautet der Aufruf im Browser *http://localhost/html-inhalt-und-php.php*. Haben Sie die Datei in einem Unterordner abgelegt, müssen Sie *[Pfad zur Datei]* durch den Namen Ihrer Ordner ersetzen (vgl. Kapitel 2).

Die nebenstehende Abbildung zeigt die Webseite *„html-inhalt-und-php.php“* im Browser.



Den vom Server zurückgelieferten HTML-Code sehen Sie in nebenstehender Abbildung.

Die PHP-Anweisungen wurden durch den auszugebenden Inhalt ersetzt. Der Betrachter sieht den ursprünglichen PHP-Code im HTML-Code nicht.

Merkmale von PHP

- PHP wurde speziell für die Programmierung dynamischer Webseiten entwickelt. Funktionen in PHP sind deshalb auf die Programmierung von Internetanwendungen abgestimmt.
- PHP zeichnet sich durch einen großen Funktionsumfang aus.
- PHP ist plattformunabhängig. Webserver mit PHP-Unterstützung stehen sowohl für Linux, macOS und Windows zur Verfügung.
- Die PHP-Anweisungen werden direkt auf dem Server ausgeführt und nicht vom Browser (Client).
- Der PHP-Quellcode ist für den Betrachter nicht sichtbar, sondern nur die auf dem Webserver generierte HTML-Ergebnisdatei.
- PHP ist fehlertoleranter als andere Programmiersprachen wie beispielsweise JAVA.
- PHP ist kostenlos erhältlich.

1.2 Informationen zu PHP

Entwicklung von PHP

1994 begann Rasmus Lerdorf mit der Entwicklung einer Skriptsprache für das Internet. Die Sprache sollte einfach zu erlernen sein. Die erste Version nannte er PHP, abgeleitet von „Personal Home Page Tools“, einer Sammlung von diversen Skripten zum Einbinden in Webseiten. Der Interpreter beherrschte wenige Befehle, sodass der Funktionsumfang klein war.

Ein Jahr später erweiterte Lerdorf den Funktionsumfang, indem er den Interpreter neu programmierte. Hinzu kamen die Funktionen des vorhandenen `Formular Interpreters`, kurz `FI`. Dieser Interpreter war in der Lage, Formulardaten ohne CGI (Common Gateway Interface) zu verarbeiten. Die Kombination der Home Page Tools und des `Formular Interpreters` (PHP2) nannte er PHP/FI. Das Neue an dieser Version war die Anbindung an die MySQL-Datenbank.

Das Privatprojekt Lerdorfs wurde 1997 von einem Team von Programmierern übernommen. Mit dabei Andi Gutmans und Zeev Suraski, die Gründer der Firma *Zend Technologies Ltd.* Die Funktionen der PHP/FI-Version wurden in das PHP3-Format portiert und neue Befehle hinzugefügt.

Mit der Version PHP3 und ihrer Anbindung an verschiedenste Datenbanken begann die Verbreitung der Skriptsprache zur dynamischen Seitengenerierung. Konkurrenzprodukte sind andere serverseitige Programmiersprachen wie ASP.NET, Java Server Pages (JSP) oder ColdFusion.

Seit PHP4 bildet ein von Zend entwickelter Compiler zur schnelle(re)n Ausführung des Codes, die sogenannte „Zend Engine“, das Rückgrat der PHP-Programmierung. PHP wurde durch Erweiterungen seines Funktionsumfangs, wie z. B. das Sessionmanagement, verbessert.

PHP-Versionen und -Verbreitung

PHP5 ist seit Sommer 2004 auf dem Markt. Eine stark verbesserte Zend Engine II ist ebenso wie zahlreiche Verbesserungen, z. B. in der objektorientierten Programmierung und in Fragen rund um die Sicherheit der Programmierung, dafür verantwortlich, dass die Zahl der Webseiten mit PHP-Unterstützung schnell wächst.

Bereits ein Jahr, nachdem PHP 5 veröffentlich wurde, wurde 2005 mit der Entwicklung von PHP 6 begonnen. Unter anderem sollte eine native Unicode-Unterstützung implementiert werden. Die Entwicklung von PHP 6 wurde mittlerweile eingestellt.

2014 wurde mit der Entwicklung einer neuen Hauptversion von PHP begonnen. Da die Entwicklung von PHP 6 als gescheitert bezeichnet werden kann, haben sich die Entwickler entschieden, die 6er Version einfach zu überspringen und die neue PHP-Version PHP 7 zu nennen. Damit ist PHP 7 die direkte Nachfolgeversion von PHP 5.6.

PHP 7 ist aktuell (April 2021) bei den großen deutschen Providern die Standardversion. PHP 8 wurde im November 2020 veröffentlicht. Bevor Provider die nächste PHP-Version anbieten, warten diese eine Weile, bis die meisten Fehler in der neuen Version gefunden und behoben wurden. Erfahrungsgemäß (und unverbindlich) ist 2022 mit PHP 8 bei den meisten Providern zu rechnen. Bevor Sie also neue Features von PHP 8 verwenden wollen, prüfen Sie vorher, welche PHP-Version Ihr Provider anbietet.

PHP wird auf 79 % aller Webseiten eingesetzt (Stand: Ende 2019, Quelle: *https://de.wikipedia.org/wiki/PHP*) und ist damit die am häufigsten verwendete Programmiersprache zum Erstellen von Webseiten.

Funktionsumfang von PHP

PHP hat heute einen großen Funktionsumfang, mit dem weitestgehend alle gängigen Anforderungen im Webumfeld gelöst werden können. Eine Auswahl oft benötigter Funktionen finden Sie nachfolgend:

- HTML-Seiten generieren, in denen benutzerbezogene Daten verarbeitet sind
- Verarbeiten unterschiedlicher Datentypen, wie Integer, Fließkommazahlen, Zeichenketten oder Arrays

- Auslesen, Überprüfung und Verarbeiten von Formulardaten
- Daten aus Datenbanken auslesen, bearbeiten und wieder in Datenbanken speichern
- Zeichenketten auslesen, verändern, abschneiden, umwandeln, in bestimmten Formaten wieder ausgeben
- Datums- und Zeitangaben auslesen, erkennen, generieren, verändern
- E-Mails erstellen und versenden
- Grafiken öffnen oder generieren, verändern, speichern
- Dateien (z. B. Text- oder CSV-Dateien) öffnen, auslesen, verändern, speichern
- Über Schnittstellen Dienste anderer Webseiten aufrufen, einlesen, auswerten
- Cookies speichern und auslesen, Sessions verwalten

Über die Kernfunktionalitäten ist PHP zusätzlich um verschiedenste Bibliotheken erweiterbar, beispielsweise ImageMagick. Die Bibliothek ImageMagick bietet mehr Funktionalitäten als PHP-eigene Funktionen zur Bildverarbeitung.

Im Rahmen der Grundlagen von PHP 8.0 werden die Basisfunktionalitäten von PHP gezeigt, die ausreichend sind, um eine dynamische Webseite zu programmieren.

Die Kombination von PHP und der Datenbank MySQL bzw. MariaDB ist auf der Mehrzahl aller Webserver zu finden. PHP ist jedoch nicht auf MySQL beschränkt, sondern kann mit unterschiedlichen Datenbanken arbeiten – entweder direkt über eigene PHP-Funktionalitäten oder auch über ODBC-Schnittstellen. So kann PHP beispielsweise auch mit Oracle-Datenbanken arbeiten.

PHP unterstützt darüber hinaus die Kommunikation mit anderen Services, z. B. Protokollen wie LDAP, IMAP, SNMP, NNTP, POP3 oder HTTP.

Zudem ist PHP zur Kommandozeilenprogrammierung (z. B. für sogenannte „cronjob“-Skripte, die wiederkehrende Aufgaben automatisieren) oder für die Entwicklung von Desktop-Anwendungen einsetzbar.

Informationen zu PHP im Internet

Folgende ausgewählte Webseiten bieten ausführliche Informationen zu PHP:

https://www.php.net/	Die wichtigste Seite für PHP-Programmierer. Hier finden Sie die komplette Dokumentation von PHP, alle Funktionen einschließlich PHP-Code-Beispielen und PHP-Versionshinweise. Auch Probleme mit Funktionen werden dort besprochen.
https://www.php.net/manual/de/	Deutschsprachiges Onlinehandbuch der PHP-Dokumentationsgruppe
https://www.w3schools.com/php/ *https://www.w3schools.com/html/* *https://www.w3schools.com/css/*	Die Webseite von W3Schools bietet umfangreiche Tutorials zu PHP und allen weiteren Webtechniken wie HTML und CSS. Die Seite ist gut strukturiert, die Erklärungen sind ergänzt durch viele hilfreiche Beispiele.

https://www.php.net/ ist die Standardreferenz für PHP-Entwickler. Große Teile der Referenz stehen in Deutsch zur Verfügung. Gerade bei neueren Features lohnt sich aber der Wechsel in die englische Variante, da diese mitunter auf einem neueren Stand ist.

1.3 PHP-Version 8.0

Die herausragende Neuerung von PHP 8 ist wahrscheinlich der JIT-Compiler (just in time). PHP-Skripte werden beim Aufruf in eine Maschinensprache übersetzt und dann auf dem Rechner ausgeführt. Bislang wurde Code, der auch mehrfach durchlaufen wird, jedes Mal neu übersetzt. PHP 8 erkennt sich wiederholende Programmabschnitte, übersetzt diese nur 1-mal und speichert diese für die wiederholte Ausführung in einem Cache. Das zieht eine wesentlich schnellere Ausführung eines PHP-Skriptes nach sich. Das bedeutet damit auch, dass Anwendungen, aufgrund deren Performance bislang C oder C++ notwendig waren, mitunter auch durch PHP ersetzt werden können.

Die **Fehler-Level** wurden überarbeitet, sodass viele Programmierfehler, die bislang nur eine Warnung bewirkt haben, nun einen wirklichen Fehler erzeugen und – falls kein Fehler-Handling implementiert ist – Skripte zum Abbruch bringen können. Dies wird zwangsläufig zu Problemen führen, wenn Provider vollständig auf PHP 8 umstellen. Alte Skripte können eventuell nicht mehr laufen. Das gleiche bewirken einzelne Funktionen, die entweder entfernt oder in der Funktionalität verändert wurden. Damit ist PHP 8 nicht abwärtskompatibel. Die strengeren Fehler-Level bewirken aber gleichzeitig hochwertigere PHP-Skripte, da Entwickler zur sauberen Programmierung gezwungen sind.

Darüber hinaus umfasst PHP 8 viele Änderungen, die jedoch kaum Auswirkung auf das Lernen der Grundlagen haben. Relevant ist hier das neue Sprachkonstrukt `match()` (siehe Kapitel 4.7), welches eine zusätzliche Kontrollstruktur liefert sowie `str_contains()`, `str_starts_with()` und `str_ends_with()` (Kapitel 9.3), welche das Suchen in Zeichenketten intuitiver machen.

Die wichtigsten Änderungen und Neuerungen von PHP 8 sind nachfolgend aufgeführt:

Neuerungen und Änderungen von PHP 8

Neuerungen bzw. Änderungen	Erklärung	Kapitel
JIT-Compiler	Bessere Performance durch gecachten Maschinencode. Vermutlich die wichtigste Neuerung in PHP 8, was jedoch für die Syntax von PHP unrelevant ist	–
Überarbeitung der Error-Klassifikation	Verschiedene *notice*-Meldungen sind nun *warning* oder *error*.	2.4, 3.3
Concatenation Precedence	Rechenoperation hat Vorrang vor Konkatenation (vergleichbar mit Punkt-vor-Strich-Rechnung)	3.3
Constructor Promotion	Sichtbarkeitsmerkmale wie `public`, `private` oder `protected` können nun in der `_construct`-Methode verwendet werden und stellen so eine Kurzschreibweise dar.	–
`match()`	Neue Kontrollstruktur, ähnlich wie `switch()`	4.7
Trailing comma in parameter lists	Erlaubt ein Komma hinter der letzten Variablen in einem Funktionsaufruf. Bislang erzeugte dies in PHP einen Error.	7.1
Passing mandatory arguments after optional is deprecated	Die Reihenfolge von erforderlichen und optionalen Parametern in Funktionsköpfen ist nun beliebig (zuvor mussten optionale Parameter hinter Parametern ohne Vorbelegung stehen).	7.2
Union Types	Bei der Deklaration eines Parameters können nun mehrere optionale Variablentypen angegeben werden.	7.2
`mixed`	Neuer Wert zur Typ-Deklaration. Entspricht dem Union Type `array|bool|callable|int|float|object|resource|string|null`	–
Named Arguments	Funktionen können benannte Variablen übergeben werden. Die Benennung legt fest, welcher Parameter welcher Variablen zugewiesen wird.	7.2
`str_contains()`	Prüft, ob eine Zeichenkette in einer anderen Zeichenkette enthalten ist	9.3
`str_starts_with()` und `str_ends_with()`	Prüft, ob eine Zeichenkette mit einer bestimmten Zeichenkette anfängt bzw. endet	9.3
`fdiv($num1, $num2)`	Dividiert den 1 Wert durch den 2. Wert. Das Teilen durch 0 führt nicht zum Fehler. Die Funktion gibt in dem Fall einen Wert `INF` oder `NAN` zurück.	–
@-Operator entfernt	Der @-Operator zum Unterdrücken von Fehlermeldungen ist nicht mehr verfügbar.	–

2

Grundlegende Sprachelemente

Beispieldateien: Dateien im Ordner *Kapitel_02*

2.1 PHP in HTML einbinden

Dateiendung für PHP-Dateien

PHP kann direkt in den HTML-Code eingefügt werden. Damit der Webserver erkennt, dass es sich um eine Datei mit PHP-Anweisung(en) handelt, werden die Dokumente mit der Dateiendung (Dateinamenerweiterung) **.php* versehen. Die Dateiendung **.php* ist für den Webserver das Signal, den PHP-Interpreter aufzurufen, dieser führt dann die PHP-Datei aus und liefert das Ergebnis, also das generierte HTML an den Browser aus.

Die Dateinamenerweiterung ****.php***. ist die gängige Schreibweise und entspricht der Standardkonfiguration. In der Vergangenheit wurde auch **.php4*, **.php5* oder **.phtml* als Dateiendung für PHP-Dateien verwendet. Mit welchen Dateiendungen der Webserver Dateien als PHP-Dateien erkennt, kann in der Datei *httpd.conf* bzw. bei der XAMPP-Software in der Datei *httpd-xampp.conf* konfiguriert werden.

In normalen HTML-Dateien (Dateiendung **.htm* bzw. **.html*) werden PHP-Anweisungen nicht interpretiert.

PHP-Anweisungen einfügen

PHP-Code kann und darf an jeder beliebigen Stelle im HTML vorkommen. PHP-Anweisungen können Sie sowohl vor dem einleitenden HTML-Tag `<!DOCTYPE html>` einfügen, aber auch im `<head>` oder im `<body>` des HTML-Quelltextes, je nachdem, wo Sie den PHP-Code benötigen.

`<?php`
`  PHP-Anweisung(en)`
`?>`
	XML-konforme Standardschreibweise
	Ein sogenannter PHP-Block wird durch das Tag `<?php` geöffnet und nach den PHP-Anweisungen mit dem Tag `?>` geschlossen.
	Diese **XML-konforme** Schreibweise ist sowohl die empfohlene als auch gängige Art, PHP-Blöcke auszuzeichnen.

PHP-Code wird als PHP-Block in das HTML eingefügt. Ein PHP-Block wird mit einem öffnenden PHP-Tag `<?php` eingeleitet und durch ein schließendes PHP-Tag `?>` beendet, die PHP-Befehle schreiben Sie innerhalb dieser PHP-Tags. Der PHP-Interpreter erkennt anhand dieser Tags die PHP-Blöcke und führt den PHP-Code darin aus.

In einer HTML-Datei können beliebig viele PHP-Blöcke vorkommen. Auch innerhalb einer Zeile im HTML können mehrere PHP-Blöcke eingefügt werden.

Alternative PHP-Tags

`<?`
`  PHP-Anweisung(en)`
`?>`
	Kurzschreibweise
	Falls die Konfigurationsvariable `short_open_tag` in der Datei *php.ini* auf den Wert `On` gesetzt wurde, können Sie `php` im öffnenden PHP-Tag weglassen.

Alternativ zur Standardschreibweise können Sie auch PHP-Tags in der Kurzschreibweise `<? … ?>` verwenden, in der das `php` im öffnenden PHP-Tag entfällt. Gerade bei vielen PHP-Blöcken spart das Tipparbeit.

Allerdings müssen Sie die Kurzschreibweise zuvor in der *php.ini* aktivieren, was auf dem eigenen Rechner für die Entwicklung möglich ist. Das Aktivieren an sich stellt das geringere Problem dar. Das eigentliche Problem liegt darin, dass Ihr PHP-Skript später auf einem Webserver eines professionellen Providers laufen soll. Dort haben Sie jedoch keinen Zugriff auf die Konfigurationsdatei *php.ini* und können die Kurzschreibweise nicht aktivieren. Auch wenn Ihr PHP-Code während der Entwicklung funktioniert hat, läuft er später auf dem gemieteten Webserver nicht.

Vor PHP 7.0 gab es zwei weitere Schreibweisen, um PHP-Blöcke im HTML einzufügen. Zum einen die ASP-Schreibweise `<% … %>` zum anderen die Skript-Schreibweise `<script language="php"> … </script>`. Auch in älteren PHP-Versionen waren beide Schreibweisen nicht gängig, funktionierten aber, wenn diese entsprechend in der *php.ini* konfiguriert waren. Mit PHP 7.0 ist diese Syntax weggefallen und würde zum Fehler führen, wenn Sie diese Art der PHP-Einbindung verwenden würden.

Verwenden Sie nach Möglichkeit die allgemeingültige **XML-Schreibweise** (`<?php … ?>`) zum Einfügen von PHP-Anweisungen in Ihren Skripten. Dies ist die gängige Einstellung der Webserver von Internet-Providern. Damit stellen Sie sicher, dass Ihre Seite auf allen Servern läuft.

Informationsdatei zur PHP-Konfiguration anzeigen

Sie können sich mit einem Skript die PHP-Konfiguration des Servers anzeigen lassen, auf dem dieses Skript ausgeführt wird. Zu diesem Zweck stellt PHP die Funktion `phpinfo()` bereit. Damit erhalten Sie detaillierte Informationen zur installierten PHP-Version samt installierten Erweiterungen.

Wenn Sie mit dem empfohlenen XAMPP-Paket arbeiten, rufen Sie im Webbrowser Ihren lokalen Rechner als (PHP-)Webserver auf. Über die Adresse *http://localhost* – oder alternativ über die IP-Adresse *http://127.0.0.1* – wird automatisch das im Webserver definierte sogenannte *document root*-Verzeichnis geöffnet. Bei einer XAMPP-Standardinstallation ist das unter Windows der Ordner *C:\xampp\htdocs*. Unter macOS finden Sie das *root*-Verzeichnis unter *lampp/htdocs* (in der gemounteten Virtuellen Maschine, siehe Kapitel A am Buchende).

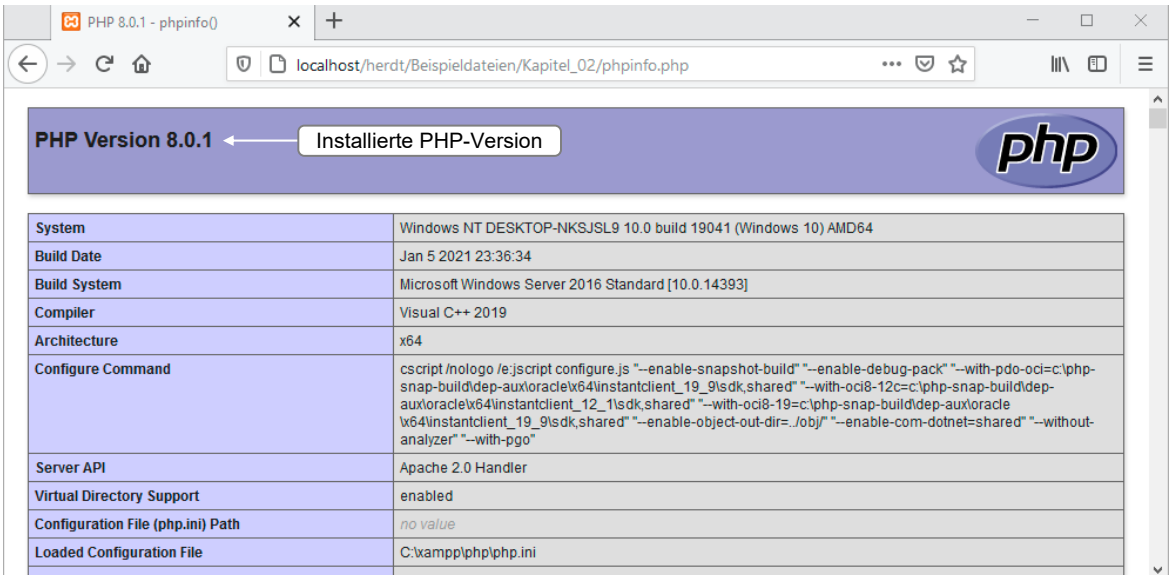
Beispiel: *phpinfo.php*

- Erstellen Sie folgende Datei und speichern Sie sie im Verzeichnis *C:\xampp\htdocs* (*lampp/htdocs* auf dem Mac) unter dem Dateinamen *phpinfo.php*.
- Um das Skript zu testen, rufen Sie im Browser die PHP-Datei mit folgender Adressangabe auf Ihrem lokalen Webserver auf:

  http://localhost/phpinfo.php oder *http://127.0.0.1/phpinfo.php* bzw. auf dem Mac *http://localhost:8080/phpinfo.php*

```
<?php
  phpinfo();
?>
```

Beispieldatei „phpinfo.php“



System	Windows NT DESKTOP-NKSJSL9 10.0 build 19041 (Windows 10) AMD64
Build Date	Jan 5 2021 23:36:34
Build System	Microsoft Windows Server 2016 Standard [10.0.14393]
Compiler	Visual C++ 2019
Architecture	x64
Configure Command	cscript /nologo /e:jscript configure.js "--enable-snapshot-build" "--enable-debug-pack" "--with-pdo-oci=c:\php-snap-build\dep-aux\oracle\x64\instantclient_19_9\sdk,shared" "--with-oci8-12c=c:\php-snap-build\dep-aux\oracle\x64\instantclient_12_1\sdk,shared" "--with-oci8-19=c:\php-snap-build\dep-aux\oracle\x64\instantclient_19_9\sdk,shared" "--enable-object-out-dir=../obj/" "--enable-com-dotnet=shared" "--without-analyzer" "--with-pgo"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	*no value*
Loaded Configuration File	C:\xampp\php\php.ini

Ausschnitt aus der Anzeige der PHP-Konfiguration

In der Kopfzeile der Seite wird Ihnen die installierte PHP-Version angezeigt. Auf der Seite selbst erscheinen dann diverse Informationen zur PHP-Installation, unter anderem Pfade zu Konfigurationsdateien, Umgebungsvariablen oder installierte Pakete.

PHP-Dateien können nicht mit einem Doppelklick auf die Datei geöffnet werden. Sie können die Datei zwar mit der Maus in den Browser ziehen, dann wird diese jedoch lediglich als HTML-Datei dargestellt, PHP-Anweisungen werden dann nicht interpretiert. Damit der PHP-Code ausgewertet wird, muss der Aufruf der PHP-Datei über einen PHP-Interpreter erfolgen, in den Beispielen in diesem Buch immer über die Serveradresse *http://localhost* oder *http://127.0.0.1* und anschließend jeweils Pfad- und Dateiname.