Teil 1: PHP-Grundlagen

Der Homepage-Porsche

Mit PHP eröffnet sich eine Fülle neuer Möglichkeiten für Ihre Homepage. In einer dreiteiligen Serie lernen Sie nützliche Anwendungen der beliebten Skriptsprache kennen

Als Webdesigner kommen Sie kaum an PHP vorbei. Das Kürzel steht für PHP Hypertext Preprocessor. PHP bietet Ihnen ein Füllhorn zusätzlicher Funktionen für Ihre Homepage, fast so, als würden Sie einen VW Käfer mit einem Porsche-Motor aufrüsten. So kann PHP dynamische Webseiten erzeugen, eng mit Datenbanken zusammenarbeiten, dynamisch Grafiken erstellen und vieles mehr. Darüber hinaus stehen Ihnen rund 1800 vordefinierte Funktionen zur Verfügung, und PHP lässt sich, anders als etwa Perl, direkt und nahtlos in HTML einfügen. Dazu ist PHP relativ leicht zu erlernen.

Teil 1 soll Ihnen die wichtigsten Grundlagen von PHP vermitteln, auf denen die beiden folgenden Teile aufbauen. Danach stellt PHP für Sie keine geheimnisvolle Welt mehr dar, die nur den Programmier-Gurus vorbehalten ist.

Alle im Artikel erwähnten Beispiele finden Sie auf der com!-Heft-CD 1 in der Rubrik „HomeP@ge“, „Praxis & Tuning“.

PHP-Serie

Die dreiteilige com!-Serie macht Sie mit Grundlagen und Einsatzmöglichkeiten von PHP vertraut.

Teil 1: PHP-Grundlagen

Teil 2: **Fertige PHP-Programme einbauen und konfigurieren**

Teil 3: **MySQL und PHP im Zusammenspiel**

Wie PHP funktioniert

Als einzige Voraussetzung muss Ihr Provider PHP auf dem Webserver installiert haben, was bei der überwiegenden Mehrzahl der Fall ist.

Ein wesentlicher Unterschied zu HTML ist, dass Sie mit PHP dynamische Webseiten erzeugen. Darunter versteht man Webseiten, die erst im Augenblick des Abrufs zusammengesetzt werden und nicht starr und statisch auf dem Webserver liegen. So können Sie Daten interaktiv in die Webseiten einbauen, auf Nutzereingaben reagieren oder Datenbankinhalte darstellen.

Was passiert, wenn Sie eine HTML-Seite aufrufen? Der Webserver schickt die Seite an den Browser, und dieser zeigt sie an. Anders bei einer PHP-Seite: In diesem Fall verarbeitet der auf dem Webserver installierte PHP-Interpreter die PHP-Anweisungen und gibt das Ergebnis an den Browser zurück, der es dann anzeigt. Um PHP-Programme zu testen, müssen Sie diese also auf den Webserver übertragen und im Browser aufrufen. Oder Sie verwenden eine lokale Testumgebung auf Ihrem PC. (Wie Sie ein solches System einrichten, lesen Sie in com! 11/2003 ab Seite 104.) Damit der PHP-Interpreter auf dem Webserver erkennt, dass eine Seite mit PHP-Code kommt, muss die Dateiendung PHP lauten, in seltenen Fällen je nach Serverkonfiguration PHP4 oder PHTML. Ein Nebeneffekt der Server-sei-

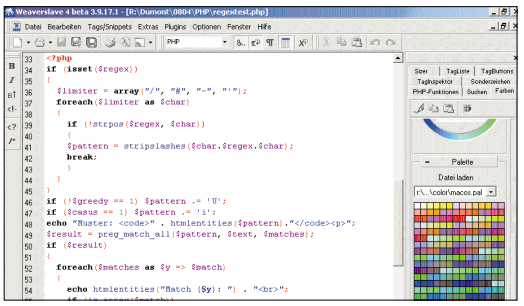
Teil 3

auf CD 1 unter HomeP@ge Praxis & Tuning

tigen Verarbeitung des Codes: Ihre Besucher bekommen nie das Skript zu sehen, sondern immer nur das Ergebnis.

Hallo Welt

Im Prinzip können Sie jeden beliebigen Editor verwenden, um PHP-Programme zu schreiben. Aber es geht auch komfortabler. Auf der Heft-CD finden Sie den Editor Weaverslave. Aktivieren Sie im Menü *Extras, Dateiprofile* den Punkt *PHP,* hilft Ihnen das Programm mit passendem Syntax-Highlighting.



Der Freeware-Editor Weaverslave bietet Syntax-Highlighting und andere nützliche Funktionen zum PHP-Programmieren

Es ist fast schon Tradition geworden, als ersten Schritt in einer noch unbekannten Programmiersprache ein Programm zu schreiben, das den Text „Hallo Welt" ausgibt. Also schreiben Sie ein solches Programm in PHP. Um PHP in eine HTML-Seite einzubetten, umgeben Sie den PHP-Code mit den beiden Tags **<?php** und **?>**. Der Befehl **echo** gibt Text zwischen Anführungszeichen auf dem Bildschirm aus. Das Seitengerüst erstellen Sie wie gewohnt in HTML. So sieht das kleine Programm dann aus:

```
<html>
<head>
<title>Mein erstes PHP-
Programm</title>
</head>
<body>
<?php
  echo "Hallo Welt";
?>
</body>
</html>
```

Eine wichtige Regel: Nach einer PHP-Anweisung steht grundsätzlich ein Strichpunkt. Speichen Sie das Programm unter dem Namen *hallo.php* und übertragen Sie es auf Ihren Webserver. Rufen Sie nun die Datei in Ihrem Browser auf. Voilà, Ihr erstes PHP-Skript funktioniert! Sie sehen eine weiße Webseite mit dem Text „Hallo Welt". Wenn Sie nichts sehen, unterstützt Ihr Provider sehr wahrscheinlich kein PHP.

Ändern Sie nun die Zeile mit dem **echo**-Befehl in

```
echo "<b>Hallo Welt</b>";
```

Sie sehen nun den Text in fetter Schrift. Da PHP und HTML eng miteinander verzahnt sind, können Sie folglich über den **echo**-Befehl auch HTML-Tags ausgeben.

Schauen Sie sich nun im Browser über *Ansicht, Seitenquelltext anzeigen* oder über *Ansicht, Quelltext* den Quellcode der Datei an.

Wohin ist denn Ihr PHP-Code verschwunden? Erinnern Sie sich, der Code wird auf dem Webserver verarbeitet, lediglich das Ergebnis wird an den Browser weitergeleitet. Sie sehen also nur lupenreines HTML.

Ein Tipp: Verwenden Sie in Ihren Programmen reichlich Kommentare, damit Sie diese auch nach einem halben Jahr noch verstehen. Um einen einzeiligen Kommentar einzufügen oder am Ende einer Zeile zu platzieren, benutzen Sie **//**. Um einen mehrzeiligen Kommentar einzufügen, umgeben Sie diesen mit den Zeichen **/*** und ***/**.

Variablen und Funktionen

Lernen Sie zunächst grundlegende Bausteine von PHP kennen, damit Sie die folgenden Skripts verstehen.

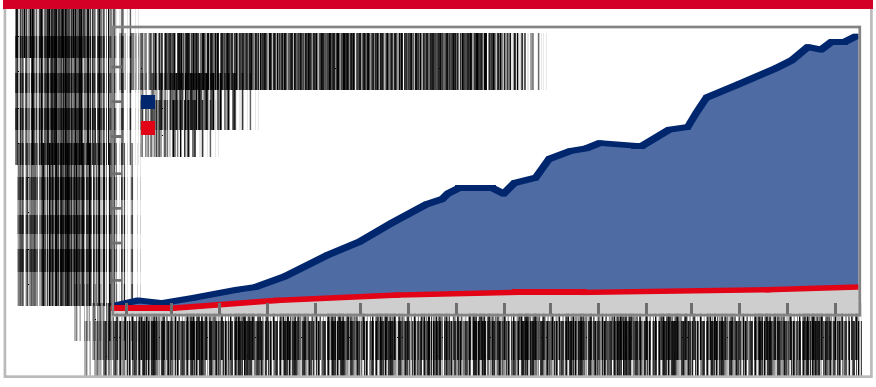
Variablen sind gewissermaßen die Seele jeder Programmiersprache und dienen als Behälter für Informationen. Die Variablennamen dürfen sich dabei aus Buchstaben, Zahlen und dem Unterstrich zusammensetzen, nicht aber mit einer Zahl beginnen.

Um eine Variable als solche kenntlich zu machen, stellen Sie ihr ein Dollarzeichen voran in der Art **$zahl=4**. PHP unterscheidet zwischen Groß- und Kleinschreibung, **$Zahl** und **$zahl** sind also zwei verschiedene Variablen.

Es gibt vier Typen von Variablen: ganze Zahlen, Gleitkommazahlen, Zeichenketten und die logischen Werte **TRUE** und **FALSE**. Der Dateityp ergibt sich aus der Zuweisung des Variablenwerts:

$zahl1=10; ganze Zahl
$zahl2=10.0; Gleitkommazahl ▶

PHP ist auf dem Vormarsch



Quelle: Netcraft

Eine Studie von Netcraft belegt: PHP erfreut sich immer größerer Beliebtheit

`$wort="Servus";` Zeichenkette
`$logik=TRUE;` logischer Wert
Funktionen schaffen Ordnung. Damit können Sie Codeabschnitte zusammenfassen und dann mehrmals aufrufen. Das hält den Programmcode klein, weil Sie nicht mehrmals den gleichen Code zu schreiben brauchen. Zudem ermöglicht dies schnelle Änderungen, da Sie lediglich eine Stelle im Code ändern.

Mit **`function`** gefolgt von dem Funktionsnamen definieren Sie eine neue Funktion. Hinter dem Funktionsnamen steht ein rundes Klammernpaar, in dem Sie bei Bedarf Parameter übergeben können. Deren Werte werden in der Funktion verarbeitet. Der Funktionskörper ist von geschweiften Klammern umschlossen. Zur Veranschaulichung ein Beispiel:

```
<?php
function multiplizieren($zahl1,
$zahl2) {
$produkt=$zahl1 * $zahl2;
echo "Das Produkt aus $zahl1 und
$zahl2 ist $produkt.";
}
multiplizieren(444, 329);
?>
```

Das Programm im Einzelnen: In der zweiten Zeile definieren Sie eine Funktion namens **`multiplizieren`**, die zwei Werte aufnehmen soll, **`$zahl1`** und **`$zahl2`**. Innerhalb der Funktion werden die beiden Zahlen multipliziert und das Ergebnis an die Variable **`$produkt`** übergeben. Danach schreibt der **`echo`**-Befehl einen entsprechenden Text auf den Bildschirm. Schließlich rufen Sie die Funktion auf und übergeben ihr die beiden zu multiplizierenden Zahlen, im Beispiel 444 und 329. Gar nicht so schwer, oder?

Ein anderes Beispiel: Angenommen, Ihr Verein feiert an Ostern immer ein rauschendes Fest, und Sie möchten auf Ihrer Homepage in einem Kalender die entsprechenden Termine der nächsten zehn Jahre eintragen. Sie können nun entweder die komplizierte Formel verwenden, die Carl Friedrich Gauß entwickelt hat, oder einfach die vordefinierte PHP-Funktion **`easter_days()`** einsetzen. Diese liefert für alle Jahre ab 1753 die Zahl der Tage, die zwischen dem 21. März und dem Ostersonntag liegen. Ostersonntag ist immer der erste Sonntag nach dem ersten Vollmond nach Frühlingsanfang.

Für eine sinnvolle Ausgabe sind noch ein paar zusätzliche Programmschritte nötig. Sehen Sie sich den Quellcode der Datei *ostern.php* an. In der Zeile **`$tage = easter_days($year);`** liefert die Funktion **`easter_days`** die Zahl der Tage zwischen Frühlingsbeginn und Ostersonntag des Jahres **`$year`** (dazu später) an die Variable **`$tage`**. Diese Zahl ist nun dort gespeichert.

Es folgt eine Abfrage, um zunächst den korrekten Monat zu ermitteln. Ist die ermittelte Zahl der Tage größer als 10 – **`if ($tage > 10)`**? Wenn ja, wird der Variablen **`$monat`** die Zeichenkette „April“ zugewiesen und die Zahl der Tage um 10 vermindert und in der Variablen **`$summe`** gespeichert – **`$summe = ($tage-10);`**. Wenn nein – **`else`** –, erhält die Variable **`$monat`** den Wert „März“, und die Zahl der Tage wird um 21 erhöht.

Im **`<body>`**-Bereich Ihrer Seite geben Sie schließlich einen entsprechenden Text mit Hilfe von **`echo`** aus.

Bleibt noch die Frage, wo denn das zu berechnende Jahr herkommt, denn die Variable **`$year`** ist nirgendwo definiert.

Des Rätsels Lösung erklärt eine weitere Besonderheit von PHP: die Möglichkeit, Variablen über den URL, also in der Adresszeile des Browsers zu definieren. Wenn Sie die Datei *ostern.php* im Browser aufrufen, erfolgt keine Wertzuweisung. Deshalb gibt die Funktion **`easter_days()`** den Wert für das laufende Jahr zurück. Wollen Sie aber wissen, wann im Jahr 1871 Ostern war, übergeben Sie den Wert im URL, indem Sie an den Dateinamen ein Fragezeichen anhängen sowie den Namen der Variablen und den gewünschten Wert: *www.meine site.de/ostern.php?year=1871*. Sie erhalten als korrekte Antwort den 9. April.

Besucherzähler in PHP

Als Nächstes schreiben Sie eine etwas komplexere Anwendung, die zudem einen praktischen Nutzen für Ihre Homepage hat: einen Besucherzähler. Nehmen Sie dazu die Datei *counter.php* unter die Lupe. Sie ist gut dokumentiert und nutzt vieles von dem, was Sie bereits gelernt haben. Der Quellcode im Detail:

In den Zeilen zwei und drei initialisieren Sie eine Zählvariable – **`$counter = 0;`** – und legen den Namen der Datei fest, in welcher der aktuelle Zählerstand gespeichert werden soll – **`$name = "counter.txt";`**.

Der Ausdruck **`if (file_exists ($name))`** prüft, ob die in **`$name`** definierte Datei – also *counter.txt* – bereits existiert.

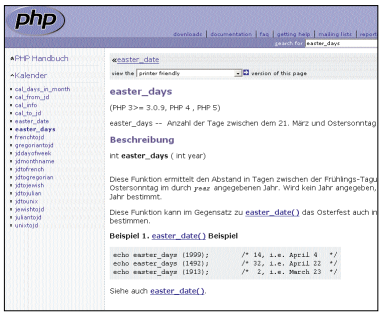
`$file = fopen($name, "r")` öffnet die Datei *counter.txt* (**`r`** steht für read, lesen) und übergibt einen Verweis an die neue Variable **`$file`**.

War dies erfolgreich – **`if ($file)`** –, wird der Zählerstand abgelesen und in der Variablen **`$counter`** gespeichert. Im Anschluss daran wird die Datei über die Funktion **`fclose()`** wieder geschlossen.

Informationsquellen zu PHP

Auf diesen Seiten finden Sie viele Hintergrundinformationen, Artikel und Anleitungen zu PHP:

- ***www.php.net:*** Die Mutter aller PHP-Seiten bietet ein umfangreiches Handbuch sowie eine detaillierte Funktionsreferenz



Bei *www.php.net* finden Sie ausführliche Beschreibungen aller PHP-Funktionen

- ***www.php-center.de:*** Deutschsprachige PHP-Seite mit Tutorials und Tipps & Tricks
- ***www.php-homepage.de:*** In der Rubrik *Scripts* finden Sie viele nützliche Code-Schnipsel
- ***www.php-resource.de:*** Ein Rund-um-sorglos-Paket mit deutschem Handbuch, aktuellen Informationen und vielen Tipps und Skripts
- ***www.php.de:*** Deutsches PHP-Forum. „Hier werden Sie geholfen“

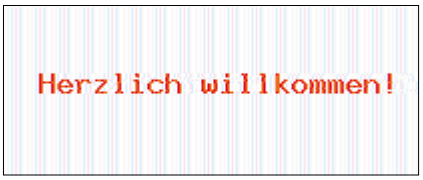
Grafiktypen und PHP

Wollen Sie genau wissen, welche Grafiktypen Sie mit PHP dynamisch erstellen können, rufen Sie die Datei *bildtyp.php* in Ihrem Browser auf. Sie erhalten eine Zahl, die sich aus der Summe der unterstützten Formate ergibt. Dabei ist 1=GIF, 2=JPEG, 4=PNG, 8=WBMP und 16=XPM. Bei den meisten Providern werden Sie 14 als Ergebnis erhalten, also 8+4+2. Folglich werden WBMP, PNG und JPEG unterstützt, nicht aber GIF. Eine Ausnahme bildet etwa 1 & 1, hier erhalten Sie als Ergebnis eine 15.

`$counter++;` erhöht den Zählerstand um eins. Danach wird die Datei *counter.txt* mit `$file = fopen($name, "w");` zum Schreiben geöffnet (`w` steht für write, schreiben), der neue Stand hineingeschrieben und diese wieder geschlossen. Schließlich geben Sie den `echo`-Befehl auf der Webseite aus.

Dynamische Grafiken

Mit PHP haben Sie die Möglichkeit, Grafiken dynamisch zu erzeugen. Sie können die Grafik-Funktionen von PHP zudem benutzen, um die Größe von JPEG-, GIF-, PNG- und SWF-Bilddateien zu ermitteln und zu verändern. In manchen Fällen erforderte dies, dass ein zusätzliches Programm-Modul namens gd auf dem Webserver installiert ist. Aber probieren Sie doch einfach aus, ob es auf Ihrer Website funktioniert. Mit der Datei *grafik.php* erzeugen Sie eine farbige Fläche mit einem Text. In Zeile 3 legen Sie über die Funktion `imagecreate()` die Maße der Fläche fest. `imagecolorallocate()` bestimmt die Farbe der Fläche, indem Sie je drei Dezimalwerte zwischen 0 und 255 für Rot, Grün und Blau angeben. `imagestring()` schreibt einen Text in die Fläche, die drei übergebenen Zahlenwerte kennzeichnen die Schriftgröße zwischen 1 und 5 sowie die x- und y-Koordinate des Textes, wobei der Ursprung in der linken oberen Ecke liegt. Die eigentliche Ausgabe des Bildes erledigt schließlich die Funktion `imagepng()`.



Mit dem Skript erzeugen Sie dynamisch eine farbige Fläche mit einem Schriftzug

Rufen Sie die Datei im Browser auf, sehen Sie die Grafik. Werfen Sie aber einen Blick in den Quellcode, erleben Sie eine Überraschung: keine Spur von HTML oder PHP, lediglich die Binärdaten des Bildes.

Im Rahmen einer Webseite können Sie damit also kaum etwas anfangen.

Als Abhilfe ändern Sie die Zeile 7 in

```
imagestring ($bild, 5, 30, 30,
$eingabe, $text);
```

und rufen die Datei von einer zweiten Datei aus auf. Ein Beispiel finden Sie in *aufruf.php* auf der Heft-CD. Dort geben Sie den Text, der in der farbigen Fläche erscheinen soll, in ein Formular ein. Die dynamisch erzeugte Grafik erscheint in einem neuen Browser-Fenster.

Kleine Helfer

In PHP sind über 1800 Klassen und Funktionen vordefiniert, die Ihnen das Programmieren erleichtern. Sie sind schnell, stabil und sofort einsetzbar. So gibt es Funktionen, um Grafiken zu erzeugen, mathematische Berechnungen durchzuführen oder mit Zeitangaben zu rechnen. Beispiele haben Sie mit den Funktionen `easter_days()` beziehungsweise `imagecreate()` bereits kennen gelernt.

Mit den vordefinierten Zeit-Funktionen können Sie zum Beispiel eine Stoppuhr bauen, die misst, wie lange ein Seitenskript für die Ausführung braucht. Vielleicht kennen Sie die kleinen Anzeigen bereits, die unter manchen Webseiten verkünden „Seite generiert in 1,565 Sekunden“. Das sieht zum einen wichtig aus und liefert Ihnen als Webmaster zudem nützliche Informationen. So merken Sie rechtzeitig, wenn ein Skript die Seite ausbremst und zu langsam wird. Die Stoppuhr besteht aus zwei Code-Schnipseln, die Sie ganz an den Anfang oder an das Ende der Seite stellen.

```
<?php
$start = explode(" ",
microtime());
$start = $start[0]+$start[1];
?>
<?php
$ende = explode(" ", microtime());
$ende = $ende[0]+$ende[1];
$zeit = round($ende-$start,5);
echo "Die Seite wurde in $zeit
Sekunden generiert";
?>
```

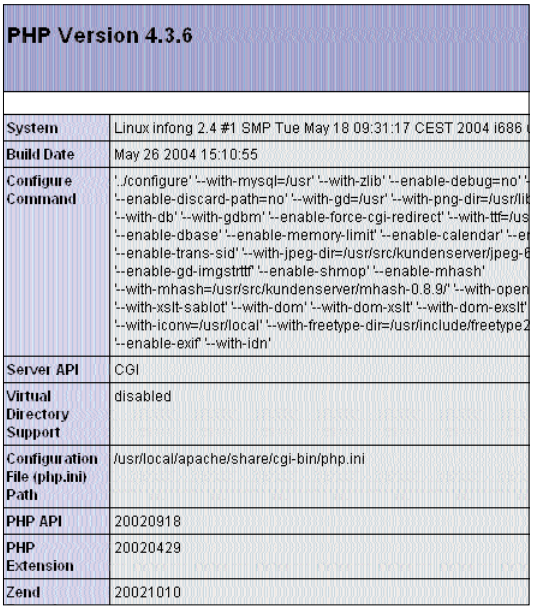
Eine Übersicht aller vordefinierten Funktionen finden Sie auf der Seite *http://de3.php.net/manual/de/funcref.php*. Suchen Sie Informationen über eine bestimmte Funktion, geben Sie deren Namen auf der Seite *www.php.net* in das Feld *search for* ein und wählen Sie bei *in the* den Eintrag *function list*.

Wer noch tiefer in die Welt der Funktionen einsteigen will, dem sei die Adresse *www.zend.com/phpfunc* empfohlen.

Sehr nützlich ist auch die Funktion `phpinfo()`. Probieren Sie es aus: Erstellen Sie eine neue Datei, und schreiben Sie folgende Zeilen hinein:

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

Übertragen Sie die Datei auf den Webserver und rufen Sie diese anschließend in Ihrem Browser auf. Nun können Sie sich zurücklehnen und haargenau studieren, wie Ihr Provider seinen Webserver konfiguriert hat und welche Dienste dort zur Verfügung stehen. ■



PHP Version 4.3.6

System	Linux infong 2.4 #1 SMP Tue May 18 09:31:17 CEST 2004 i686
Build Date	May 26 2004 15:10:55
Configure Command	'../configure' '--with-mysql=/usr' '--with-zlib' '--enable-debug=no' '--enable-discard-path=no' '--with-gd=/usr' '--with-png-dir=/usr/lib' '--with-db' '--with-gdbm' '--enable-force-cgi-redirect' '--with-ttf=/usr' '--enable-dbase' '--enable-memory-limit' '--enable-calendar' '--enable-trans-sid' '--with-jpeg-dir=/usr/src/kundenserver/jpeg-6' '--enable-gd-imgstrttf' '--enable-shmop' '--enable-mhash' '--with-mhash=/usr/src/kundenserver/mhash-0.8.9/' '--with-xslt-sablot' '--with-dom' '--with-dom-xslt' '--with-dom-exslt' '--with-iconv=/usr/local' '--with-freetype-dir=/usr/include/freetype2' '--enable-exif' '--with-idn'
Server API	CGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/apache/share/cgi-bin/php.ini
PHP API	20020918
PHP Extension	20020429
Zend	20021010

Mit der Funktion *phpinfo* erhalten Sie eine detaillierte Auskunft über den Webserver Ihres Providers

Andreas Dumont
homepage@com-online.de

Eine kurze Geschichte von PHP

Ende 1994 entwickelte der Däne Rasmus Lerdorf eine Sammlung von Makros und Skripts für den privaten Gebrauch. Er nannte sie PHP/FI, kurz für Personal Home Page/Forms Interpreter. Später schrieb er diese Tools in der Programmiersprache C um und veröffentlichte den Quellcode. Der nächste Meilenstein war PHP 3.0, das 1997 von den beiden israelischen Studenten Zeev Suraski und Andi Gutmans neu geschrieben wurde – mit vielen Erweiterungsmöglichkeiten und einer besseren Zusammenarbeit mit Datenbanken. Fortan stand PHP für das rekursive Akronym PHP Hypertext Preprocessor. Im Jahr 2000 erschien PHP 4.0, das mit einem neuen Sprachkern arbeitet, der so genannten Zend Engine. PHP 5, dem die neue Zend Engine 2.0 zu Grunde liegen wird, erscheint voraussichtlich Ende 2004. Aktuell ist die PHP-Version 4.3.7.