PHP Grundlagen

Colletta Hagert

Unister

Überblick

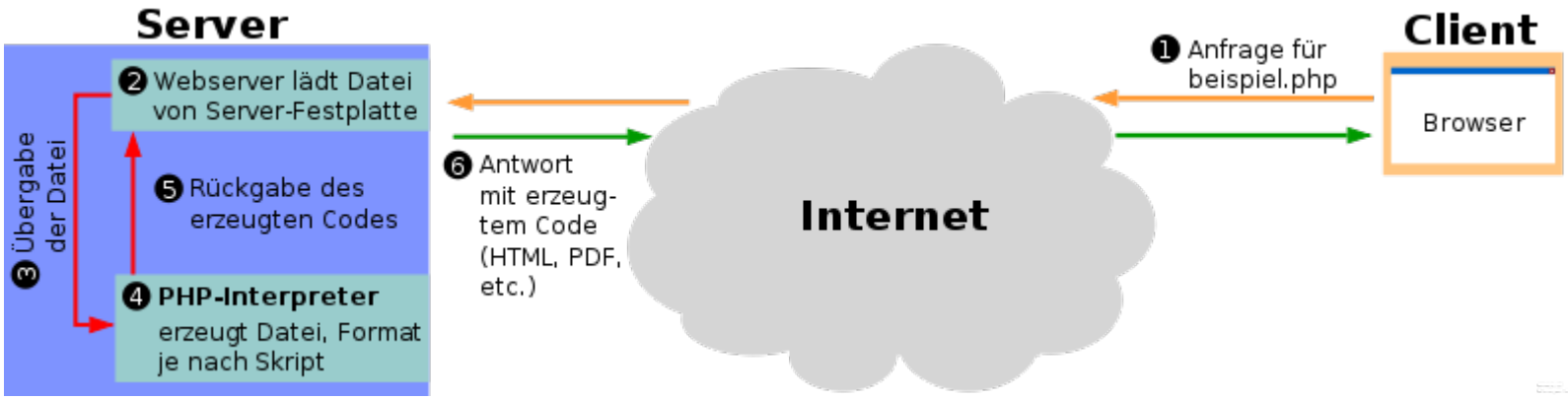
- Einführung
- Variablen
- Typ-Umwandlung
- Konstante
- Ausdrücke
- Operatoren
- Vordefinierte Variablen
- Magische Konstanten
- Zusammenfassung

Einführung

PHP (rekursives Akronym für PHP: Hypertext Preprocessor) ist eine weit verbreitete und für den allgemeinen Gebrauch bestimmte Open Source-Skriptsprache, welche speziell für die Webprogrammierung geeignet ist und in HTML eingebettet werden kann.

PHP-Anweisungen werden in HTML eingebettet

Der PHP-Code steht zwischen speziellen Anfangs- und Abschluss-Verarbeitungsinstruktionen <?php und ?>, mit denen man in den "PHP-Modus" und zurück wechseln kann.



Darstellung der Funktionsweise von PHP [Wiki]

PHP unterscheidet sich von clientseitigen Sprachen wie Javascript dadurch, dass der Code auf dem Server ausgeführt wird und dort HTML-Ausgaben generiert, die an den Client gesendet werden.

Beispiel: Das Skript in HTML integriert

```
<!DOCTYPE HTML PUBLIC "-//W3C/DTD HTML 4.01 Transitional//EN
"http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <title>Beispiel</title>
   </head>
   <body>
      <?php
         echo "Hallo Welt!";
      ?>
   </body>
</html>

// Ausgabe: Hallo Welt!
```

Drei Hauptgebiete, in denen PHP-Skripte genutzt werden, sind:

Serverseitige Programmierung: Dies ist das traditionelle und auch Hauptfeld von PHP. Sie benötigen drei Dinge, um damit arbeiten zu können: Den PHP-Parser (CGI oder Server-Modul), einen Webserver und einen Webbrowser.

Kommandozeilenprogrammierung (CLI:Command Line Interface): Es ist auch möglich PHP-Skripte schreiben, die ohne einen Server oder Browser arbeiten können. Dafür wird nur den PHP-Parser benötigt.

Schreiben von Desktop-Applikationen. PHP ist wahrscheinlich nicht die allerbeste Sprache, um Desktop-Anwendungen mit grafischer Oberfläche zu schreiben, aber PHP-GTK kann genutzt werden, um derartige Programme zu schreiben.

Variablen

Allgemeines

Variablen werden durch ein **$**-Zeichen zu Beginn gekennzeichnet

Es wird zwischen Groß- und Kleinschreibung unterschieden

Variablennamen

- Eine gültige Form beginnt mit einem Buchstaben oder einem Unterstrich '_' gefolgt von einer beliebigen Anzahl von Buchstaben, Zahlen oder Unterstrichen.

- Buchstaben von a bis z und von A bis Z

Beispiele

$test = „Richtich“;

$_test = „Auch richtig“;

// Richte Variablendefinitionen, da am Anfang eine Buchstabe bzw. eine Unterstrich steht

$2test = „Falsch“;

// Falsche Variablendefinition, da am Anfang eine Zahl steht

Wertzuweisung: Wertaufruf

Wenn man einer Variablen einen Ausdruck zuweist, wird der ganze Inhalt des Originalausdrucks in die Variable kopiert.

=> Eine Variablen deren Inhalt von einer anderen Variablen zugewiesen wurde, behält ihren Inhalt, auch wenn die Quell-Variabel verändert wurde.

=> Die Inhalte der Ziel- und Quell-variablen sind

insoweit unabhängig voneinander

- In PHP 3

Wertzuweisung: Referenzaufruf

Ab PHP 4 bietet sich eine andere Möglichkeit der Wertzuweisung bei Variablen:

=> Wertzuweisung durch Referenzierung

=> der Wert der neuen Variablen stellt eine Referenz zur Quell-variable dar.

Änderung im Inhalt der Quell-variablen ändern deshalb auch den Inhalt der neuen Variablen, und umgekehrt.

Wertzuweisung: Referenzaufruf

=> eine effektive Lösungsansatz bei umfangreichen Schleifen oder bei der Übertragung von großen Arrays und Objekten

- Die Zuweisung per Referenz wird durch das Voranstellen eines „**&**"-Operators der Quell-variable, die einer anderen Variablen zugewiesen werden soll.

- Lediglich werden Variablenzeiger zugewiesen

Beispiele

$vorname = „Martin“;

// der Variablenname $vorname wird der Wert „Martin“ zugewiesen

$name = &$vorname

// In $name wird ein Zeiger auf $vorname erzeugt

$name = „Christian“;

// $name wird verändert, deshalb ändert sich auch der Inhalt von $vorname auf „Christian“

Typ-Deklaration

Der Typ einer Variablen wird normalerweise durch den Zusammenhang bestimmt, in dem die Variable verwendet wird. PHP unterstützt folgende Typ-Deklarationen:

- Integer
- Fließkommazahl
- String/Zeichenkette
- Boolean /Wahrheitswerte
- Array
- Objekt

Typ-Umwandlung

1. Typ-Umwandlung mit Casting-Operatoren

Funktionsweise:

Der Name des geforderten Typs wird vor der umzuwandelnden Variablen (optional in Klammern) gesetzt.

• Folgende Umwandlungen sind in PHP möglich:

- **int, integer**: Umwandlung in Integer-Wert
- **real, double, float**: Umwandlung in Double-Wert
- s**tring**: Umwandlung in String-Wert
- **Boolean**: Umwandlung in Wahrheits-Wert
- **array**: Umwandlung in ein Array
- **object**: Umwandlung in ein Objekt

Bei der Konvertierung zum Typ boolean gelten die folgenden Werte als FALSE:

- boolean FALSE selbst
- integer 0 (zero)
- float 0.0 (zero)
- Der leere string, und der string "0"
- Ein array ohne Elemente
- Ein object ohne Eigenschaftsvariablen (nur PHP 4)
- Der spezielle Typ NULL (inklusive nicht gesetzter Variablen)
- SimpleXML Objekte die aus leeren Tags erzeugt wurden.

2. Typ-Umwandlung

Umwandlung einer Variablen in einen bestimmten Typ kann erzwungen werden:

=> Die Funktion **settype()** verwenden

Beispiel

```
<?php
   $foo = "5bar"; // string
   $bar = true;   // boolean
   settype($foo, "integer"); // $foo ist jetzt integer
   echo $foo; // Ausgabe: 5
   settype($bar, "string");  // $bar ist jetzt string
   echo $bar; // Ausgabe: '1'
```

3. Automatische Typ-Umwandlung

Der Plusoperator „+“ ist ein prädestiniertes Beispiel für die Automatische Typ-Konvertierung in PHP.

Typ des Ergebnisses der Addition:

- Double, falls einer der Additionswerte vom Typ double ist
- Integer, sonst

Werden zwei Werte mit einem String-Operator „.“ verknüpft, so resultiert daraus ein String.

Automatische Typ-Konvertierung

```
<?php
   $var = "0";
   // $var ist vom Typ String (ASCII 48)
   echo $var."<br />"; // Ausgabe: '0'
   $var++;
   // $var bleibt von Typ String (ASCII 49)
   echo $var."<br />"; // Ausgabe: '1'
   $var +=1;
   // $var ist jetzt vom Typ Integer
   echo $var."<br />"; // Ausgabe: 2
   $var2 = 5;
   $var .= $var2;
   // $var ist jetzt vom Typ String
   echo $var."<br />"; // Ausgabe: 25
```

4. Unbemerkte Konvertierung

Eine Konvertierung wird auch dann durchgeführt, wenn der Wert einer Integer-Variablen größer wird als der gültige Wertebereich.

```
<?php
   $wert = 2147483640; // Integer ist bis 2147483647 definiert
   var_dump($wert);    // Ausgabe: int(2147483640)
   $wert = $wert+8;    // wird größer als der gültige Wertebereich
   var_dump($wert);    // Ausgabe: float(2147483648)
   $wert-=2147483647; // $wert enthält jetzt 1
   var_dump($wert);    // Ausgabe: float(1): $wert bleibt float
```

=> Der Typ wird nicht automatisch zurück konvertiert, wenn das Ergebnis einer anderen Berechnung zurück in den gültigen Wertebereich gelangt.

5. Explizite Konvertierung mit unerwünschtem Ergebnis

Versucht man einen zu großen Wert in einem Integer zu konvertieren, führt das zu einem sogenannten „Überlauf“

=> eine negative Zahl wird erhalten

```
<?php
    $wert = 2147483640;      // Integer ist bis 2147483647 definiert
    $wert = $wert+9;         // wird zu groß und damit zum Float
    $wert = (int) $wert;     // $wert wird zum Integer gemacht
    var_dump($wert);         // Ausgabe: int -2147483647 aus
```

6. Automatisches Typ-Casting bei Vergleichen

Ein automatisches Typ-Casting wird nicht nur bei Berechnungen, sondern auch bei Vergleiche, also z. B. Bei der Bedingung einer if-Anweisung, durchgeführt.

Mit dem Identitätsoperator === wird auf Identität und nicht nur auf Gleichheit getestet.

=> Die Bedingung (false == 0) wird mit true bewertet, wohingegen ein (false === 0) mit false bewertet wird.

Identitätsoperator arbeitet auch ein wenig schneller als der Gleichheitsoperator.

=> Wenn in einer Bedingung den Inhalt einer Variable mit einer Zahl verglichen werden sollen, muss der Identitätsoperator verwendet werden.

Konstante

- Sind einmal definierte und unveränderte Werte

=> ähnlich einer Variable, die immer denselben Wert zugewiesen bekommt.

- Eine Konstante ist ein Bezeichner (Name) für einen einfachen Wert.

- PHP stellt einen Mechanismus zur Verfügung, mit dem man weitere Konstante zur Laufzeit definieren kann.

- Eine Konstante unterscheidet standardmäßig zwischen Groß- und Kleinschreibung (case-sensitive).
- Nach gängiger Konvention werden Konstanten immer in Großbuchstaben geschrieben.
- Wie bei superglobals ist der Gültigkeitsbereich einer Konstanten global.

Der Name einer Konstanten folgt den gleichen Regeln wie alle anderen Bezeichner in PHP

=> Ein gültiger Name beginnt mit einem Buchstaben oder einem Unterstrich, gefolgt von beliebig vielen Buchstaben, Ziffern oder Unterstrichen.

Beispiel

```
<?php
    // Gültige Namen für Konstanten
    define("FOO",     "Richtig");
    // Ungültige Namen für Konstanten
    define("2FOO",    "Richtig");
    // Gültige Namen für Konstanten, muss aber vermieden werden
    define("_FOO_",   "Auch Richtig");
    echo FOO; // Ausgabe: Richtig
    echo _FOO_; // Ausgabe: Auch Richtig

    // Funktioniert seit PHP 5.3.0
    const CONSTANT = 'Hallo Welt';
    echo CONSTANT;  // Ausgabe: Hallo Welt
```

Ausdrücke

Die einfachste, aber auch zutreffendste Definition für einen Ausdruck ist "alles, was einen Wert hat".

Ausdrücke (Expressions) sind die wichtigsten Bausteine von PHP.

Die grundlegendsten Formen von Ausdrücken sind Konstanten und Variablen.

Mit "$a = 5" wird $a den Ausdruck '5' zugewiesen

=> '5' ist ein Ausdruck mit dem Wert 5

=> $a ist ebenfalls ein Ausdruck mit dem Wert 5

Beispiele von Ausdrücken sind:

- Funktionen: sind Ausdrücke mit dem Wert ihres Rückgabewertes.

- Prä- und Post-Inkrement sowie die entsprechenden Dekremente

=> Die Notationen '++variable', 'variable++', '--variable' und 'variable--'.

- Vergleichsausdrücke: geben entweder FALSE oder TRUE zurück.

=> > (größer), >= (größer oder gleich), == (gleich), != (ungleich), < (kleiner), und <= (kleiner oder gleich), === (inhalts- und typgleich) und !== (nicht inhalts- oder typgleich)

- Der ternäre Operator: ? :

Operatoren

Folgende Arten von Operatoren werden unterschieden:

- Arithmetische Operatoren
- Zuweisungsoperatoren
- Vergleichsoperatoren
- Fehler-Kontroll-Operator
- Inkrementierungs- und Dekrementierungsoperator
- Logische Operatoren
- Zeichenketten-Operatoren
- Operatoren zur Programmausführung
- Bit-Operatoren

Typen von Operatoren:

- Der unäre Operator, der nur mit einem Wert umgehen kann: Beispiel sind:
  - !: der Verneinungsoperator
  - ++: der Inkrementoperator
  - -- : der Inkrementoperator

- Der Binäre Operator steht zwischen zwei Werten und verknüpfen diese zu einem neuen Wert.

=> diese Gruppe enthält die meisten Operatoren, die PHP unterstützt

- Der ternäre Operator : ?:

Unister

Arithmetische Operatoren

Addition - Summe von $a und $b:	$a + $b
Subtraktion - Differenz von $a und $b:	$a - $b
Multiplikation – Produkt von $a und $b:	$a * $b
Division – Quotient von $a und $b:	$a / $b
Modulus – Rest von $a geteilt durch $b:	$a % $b

Zuweisungsoperatoren

Der einfachste Zuweisungsoperator ist „="

Bedeutung:

Der Zuweisungsoperator bedeutet, dass dem linken Operanden der Wert des rechten Operanden zugewiesen wird.

Kann mit allen binären, arithmetische und String-Operatoren kombiniert werden

=> den Wert einer Variablen kann in einem Ausdruck

benutzt werden und das Ergebnis dieses Ausdruckes

als neuer Wert zugewiesen werden

Vergleichsoperatoren

Erlauben es uns, zwei Werte zu vergleichen.

Operator	Erklärung	Beispiel
==	Gleich- gibt TRUE zurück, wenn $a gleich $b ist	$a == $b
===	Identisch - gibt TRUE zurück, wenn $a gleich $b ist und beide auch vom gleichen Typ sind. (Ab PHP 4)	$a === $b
!=	Ungleich- gibt TRUE zurück, wenn $a nicht gleich $b ist	$a != $b
<	Kleiner als - gibt TRUE zurück, wenn $a kleiner ist als $b	$a < $b
>	Größer als - gibt TRUE zurück, wenn $a größer ist als $b	$a > $b
<=	Kleiner gleich - gibt TRUE zurück, wenn $a kleiner oder gleich als $b ist	$a <= $b
>=	Größer gleich - gibt TRUE zurück, wenn $a größer oder gleich als $b ist	$a >= $b
?:	Trinitätsoperator – gibt Ausdruck2 zurück, wenn Ausdruck1 gleich TRUE, andernfalls wird Ausdruck3 ausgewiesen	(ausdruck1) ? (ausdruck2) : (ausdruck3)

Fehler-Kontroll-Operatoren

- Das @-Symbol, der Fehler-Kontroll-Operator, ignoriert alle Fehlermeldungen, die von einem Ausdruck erzeugt werden, vor dem dieser Operator steht.

- Ist das **track_errors**-Feature in der Konfigurationsdatei php.ini aktiviert, werden alle Fehlermeldungen, die von diesem Ausdruck erzeugt, in der globalen Variablen **$php_errormsg** gespeichert.

Inkrementierungs- und Dekrementierungsoperatoren

PHP unterstützt Prä- und Post-Inkrementierungs- und Dekrementierungsoperatoren

Operator	Erklärung
++$a	Prä-Inkrement - erhöht den Wert von $a um eins und gibt anschließend den neuen Wert von $a zurück
$a++	Post-Inkrement – gibt zuerst den aktuellen Wert von $a zurück und erhöht diesen dann um eins
--$a	Prä-Dekrement - vermindert den Wert von $a um eins und gibt anschließend den neuen Wert von $a zurück
$a--	Post-Dekrement – gibt zuerst den aktuellen Wert von $a zurück und vermindert diesen dann um eins

Logische Operatoren

Operator	Erklärung	Beispiel
and	Und – gibt TRUE zurück, wenn sowohl $a als auch $b wahr ist	$a and $b
or	Oder – gibt TRUE zurück, falls entweder $a oder $b wahr ist	$a or $b
xor	Entweder Oder – gibt TRUE zurück, wenn entweder $a oder $b wahr ist; nicht wenn beide wahr sind	
!	Nicht – gibt TRUE zurück, wenn $a nicht wahr ist	!$a
&&	Und – gibt TRUE zurück, wenn sowohl $a als auch $b wahr ist	$a && $b
||	Oder – gibt TRUE zurück, falls entweder $a oder $b wahr ist	$a || $b

Zeichenketten-Operatoren

PHP bietet zwei Operatoren für Zeichenketten:

Die Vereinigungsoperatoren: '**.**' und '**.=**'

Der Rückgabewert von '**.**' ist eine Zeichenkette, die aus dem rechten und dem linken Argument besteht.

Beispiele

$teil1 = „Wie geht“;

$teil2 = „ es euch?“;

$gesamt = **$teil1 . $teil2**; **// Wie geht es euch?**

$gesamt .= „ Gut danke!“; **// äquivalent zu $gesamt = $gesamt . "Gut danke!";**

// Ausgabe: $gesamt = Wie geht es euch? Gut danke!

Operatoren zur Programmausführung

PHP unterstützt einen Operator zur Ausführung externer Programme:
Die sogenannten Backticks (``)

=> Entsprechen auf den meisten Tastaturen den Tasten

für die französischen Akzent.

PHP versucht, den Text zwischen den Backticks als Kommando-Befehl auszuführen.

Beispiel

```
$ausgabe = `ls -al`;
echo „<pre>$ausgabe</pre>“;
```

Bit-Operatoren

Erlauben es, bestimmte Bits in einem Integer auf 0 oder 1 zu setzen

=> ein- oder auszuschalten

Operator	Erklärung	Beispiel
&	Und – die Bits sind in $a und $b gesetzt	$a & $b
|	Oder – die Bits sind entweder in $a oder in $b gesetzt	$a | $b
^	Entweder oder (XOR) – die Bits sind in $a oder $b gesetzt	$a ^ $b
~	Nicht – die Bits, die in $a nicht gesetzt sind, sind gesetzt, und umgekehrt	~$a
<<	Nach links verschieben – Verschiebung der Bits von $a um $b Stellen nach links. Jede Stelle entspricht einer Multiplikation mit zwei	$a << $b
>>	Nach rechts verschieben – Verschiebung der Bits von $a um $b Stellen nach rechts.	$a >> $b

Typ Operatoren

- In PHP gibt es einen einzigen Typ Operator: instanceof.
- instanceof wird dazu verwendet um festzustellen, ob ein gegebenes Objekt ein Objekt ist, das zu einer bestimmten Klasse gehört.
- instanceof wurde in PHP 5 eingeführt.

Beispiel

```
class A { }
class B { }
$ding = new A;
if ($ding instanceof A) {
    echo 'A';
}
if ($ding instanceof B) {
    echo 'B';
}
```

Operator-Rangfolge

- Die Operator-Rangfolge legt fest, wie "eng" ein Operator zwei Ausdrücke miteinander verbindet.

- Die **Tabelle** zeigt die Rangfolge der Operatoren.

Vordefinierte Variablen

PHP stellt viele vordefinierte Variablen zur Verfügung.

Diese Variablen stehen automatisch in jedem Skript zur Verfügung

=> sie müssen nicht vorher mit „global“ definiert werden

Superglobals oder automatisch globale Variable sind Built-in-Variablen, die immer in allen Gültigkeits-Bereichen verfügbar sind.

- **$GLOBALS:**

Referenziert alle Variablen, die im globalen Gültigkeitsbereich vorhanden sind

- **$_SERVER:**

Informationen über Server und Ausführungsumgebung

- **$_GET:**

ein assoziatives Array mit Variablen, die an das laufende Skript mit einem GET-Aufruf übergeben wurden.

- **$_POST:**

ein assoziatives Array mit Variablen, die an das laufende Skript mit einem POST-Aufruf übergeben wurden.

- **$_FILE:**

ein assoziatives Array mit Elementen, die an das laufende Skript mit einem HTTP_POST hochgeladen wurden.

- **$_COOKIE:**

ein assoziatives Array mit Variablen, die an das laufende Skript mit HTTP-Cookies übergeben wurde

- **$_SESSION:**

Diese Variable enthält die momentane Session-variable des laufenden Skripts

- **$_REQUEST:**

Ein assoziatives Array mit den Inhalten von $_GET, $_POST, und $_COOKIE.

- **$_ENV:**

Ein assoziatives Array von Variablen, die dem aktuellen Skript mittels der Environment-Methode übergeben werden.

- **$php_errormsg:**

Diese Variable enthält die letzte Fehlermeldung im derzeitigen Gültigkeitsbereich.

=> Die Verwendung dieser Funktion setzt voraus, dass track_errors auf TRUE gesetzt ist.

Magische Konstanten

Einige "magische" PHP-Konstanten sind:

Es gibt sieben magische Konstanten, die, abhängig davon, wo sie eingesetzt werden, einen unterschiedlichen Wert haben.

__LINE__:	Die aktuelle Zeilennummer einer Datei.
__FILE__:	Der vollständige Pfad- und Dateiname einer Datei.
__DIR__:	Der Name des Verzeichnisses, in dem sich die Datei befindet.
__FUNCTION__:	Der Name der Funktion.
__CLASS__:	Der Name einer Klasse.
__METHOD__:	Der Name einer Klassenmethode.
__NAMESPACE__:	Der Name des aktuellen Namespace (Beachtung der Groß- und Kleinschreibung).

Zusammenfassung

Variablen

- werden durch ein $-Zeichen zu Beginn gekennzeichnet
- Es wird zwischen Groß- und Kleinschreibung unterschieden
- Der Variablenname hat eine gültige Form: beginnt mit einem Buchstaben oder einem Unterstrich gefolgt von einer beliebigen Anzahl von Buchstaben, Zahlen oder Unterstrichen

Wertzuweisung:

- Wertaufruf: einer Variablen wird einen Ausdruck zugewiesen, in dem der ganze Inhalt des Originalausdrucks in die Variable kopiert wird.
- Referenzaufruf: der Wert der neuen Variablen stellt eine Referenz zur Quell-variable dar.

Typ-Umwandlung

- Typ-Umwandlung mit Casting-Operatoren
- Typ-Umwandlung mit der Funktion settype()
- Automatische Typ-Umwandlung mit dem Plusoperator (+) und dem String-Operator (.)
- Unbemerkte Konvertierung: wenn der Wert einer Integer-Variablen größer als der gültige Wertebereich wird.
- Explizite Konvertierung mit unerwünschtem Ergebnis: bei einem Überlauf
- Automatisches Typ-Casting bei Vergleichen, z. B. bei if-Anweisung

Vordefinierte Variablen

Diese Variablen stehen automatisch in jedem Skript zur Verfügung

=> sie müssen nicht vorher mit „global" definiert werden

Vordefinierte Variablen

Diese Variablen stehen automatisch in jedem Skript zur Verfügung

=> sie müssen nicht vorher mit „global“ definiert werden

Konstante

- sind einmal definierte und unveränderte Werte.
- weitere Konstante können zur Laufzeit definiert werden.
- der Gültigkeitsbereich einer Konstanten ist auch global.

Magische Konstante

Haben abhängig davon, wo sie eingesetzt werden, einen unterschiedlichen Wert

Quellen

http://www.php.net/manual/de/language.constants.predefined.php

http://www.php.net/manual/de/reserved.constants.php

http://php.net/manual/de/reserved.variables.php

http://www.php.net/manual/de/language.basic-syntax.php