PHP 7

Kompetenznachweis

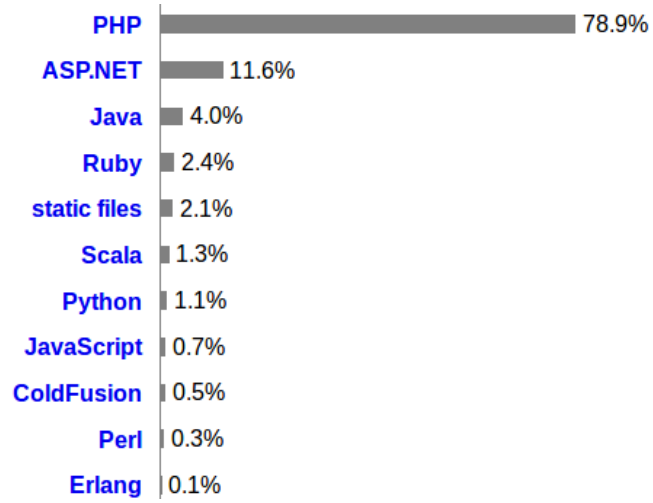
TEAM
Adriana Arteaga, Nuria Anaya,
Andreas Nydegger, Kevin Schleuniger

Inhalt

PHP

PHP ist die Abkürzung für „Hypertext Preprocessor“ ursprünglich „**P**ersonal **H**ome **P**age Tools/Forms Interpreter“ und ist eine Skriptsprache, die Grundsätzlich für Webseiten und Webanwendungen verwendet wird. PHP ist im Jahr 1995 erschienen. Die aktuelle Version ist PHP 7.3.10, die am 26. September 2019 herauskam. PHP wurde von Sprachen wie Perl, C, C++ und Java beeinflusst.

PHP ist laut W3Tech die meist genutzte Server-Programmiersprache-



PHP existiert schon lange, und da so viele die Sprache nutzen, wird sie wahrscheinlich noch lange im Web bleiben. Viele scheuen auch den Aufwand und Kosten, alles auf andere Sprachen umzustellen.

PHP ist einfach zu lernen und leicht in der Benutzung. Ausserdem gibt es ein sehr breites, preiswertes und einfaches Hosting-Angebot für PHP-Anwendungen. Ebenso besitzt PHP den mit Abstand größten Marktanteil (im Februar 2019).

Funktionsweise

PHP-Code wird serverseitig verarbeitet, d.h. dass der Quelltext nicht an den Webbrowser übermittelt wird, sondern an einen Interpreter auf dem Webserver. Die Ausgabe des PHP-Interpreters wird dann schliesslich an den Browser geschickt.

Unterschiede PHP 5 zu PHP 7

Von PHP 5 zu PHP 7 wurden viele Änderungen zur Verbesserung der Sprache durchgeführt.

Hier sind einige Beispiele:

Performance

Im Vergleich zu PHP 5, wurde die Performance stark verbessert. Diese Performanceverbesserung verdankt PHP 7 ihrer neuen Engine, PHP-NG (oder Next Generation).

Nach Zend Technologies, welche die Engines von PHP 7 und PHP 5, Zend II, entwickelte, ist PHP-NG durch verbesserten Speichergebrauch doppelt so schnell wie Zend II.

Return Typen

Mit PHP 7 wurden Return Typen für PHP eingeführt. Diese ermöglichen PHP-Programmierern zu überprüfen, ob der Datentyp der erhaltenen Variabel unerwünscht ist oder nicht und das Generieren von Exceptions je nach Datentyp.

Zurzeit sind vier Datentypen als Return Typ verfügbar: Boolean (bool), Integer (int), Strings (string) und Fliesskommazahlen (float).

Error Handling

In PHP 5 war es kompliziert, fatale Errors zu verarbeiten. In PHP 7 wurden viele grössere Errors durch Exceptions ersetzt, welche ohne grossen Aufwand verarbeitet werden können.

64-Bit Support

In PHP 7 wurde der Support für 64-Bit hinzugefügt. Dadurch werden auch 64-Bit Integer und der Gebrauch grösserer Files unterstützt.

Ausserdem laufen PHP Applikationen mit PHP 7 reibungslos auf Maschinen mit einer 64-Bit Architektur.

Neue Operatoren

PHP 7 fügt ausserdem noch zwei weitere Operatoren zu PHP hinzu, nämlich:

"Spaceship"

Der sogenannte "Spaceship" Operator wird aus einem kleiner als, Gleichheits- und grösser als Zeichen (<=>) zusammengesetzt.

Dieser Operator ist dazu da Code zu kürzen, welcher überprüft ob der erste Value kleiner, gleich wie, oder grösser als der Zweite ist.

Je nach Ergebnis der Überprüfung gibt der "Spaceship" Operator 1 (Erster Value grösser), 0 (Beide Values gleich), oder -1 (Erster Value kleiner) zurück.

Beispiel:

```php
<?php
	declare(strict_types = 1);

	//Test with if statements

	function testIf(int $first, int $second): int {
		if ($first < $second) {
			return 1;
		} elseif ($first == $second) {
			return 0;
		} else {
			return -1;
		}
	}

	//Test with Spaceship

	function testSpaceship(int $first, int $second): int {
		return $first <=> $second;
	}
?>
```

“Null coalescing”

Der “Null coalescing” Operator wird als zwei Fragezeichen (??) geschrieben.

Dieser Operator überprüft ob der erste Operand existiert und er nicht NULL ist. Trifft dies zu, gibt der Operator den ersten Operanden zurück. Wenn einer der beiden Bedingungen *true* zurückgibt, wird der Operator den zweiten Operanden zurückgeben.

Beispiel:

```php
<?php
	$name;

	//Test with if statements
	if (isset($_POST['username'])) {
		$name = $_POST['username'];
	} else {
		$name = null;
	}

	//Test with Spaceship
	$name = $_POST['username'] ?? null;
?>
```

Syntax

PHP ist nicht Case sensitive, wodurch alle Befehle, ausser Variabeln, da sie Case sensitive sind, in lower case, UPPERCASE, oder auch in MoCkSpEeCh geschrieben werden können.
Der PHP-Interpreter führt PHP-Code erst aus, wenn der Code in den vom PHP-Interpreter definierten Trennzeichen steht. Die meist gebrauchten Trennzeichen sind '<?php' und '?>'.
Das Ende einer Codezeile wird, wie in Sprachen wie C/C++, C#, Java, etc., mit einem Semikolon (;) gekennzeichnet.

Datentypen

PHP unterstürzt die folgenden Datentypen:

- String
- Integer
- Float (floating point numbers)
- Boolean
- Array
- Object
- NULL
- Resource

Variabeln

Variabeln werden mit einem Dollarzeichen ($) vor dem Variabelnamen definiert. Sie besitzen grundsätzlich keinen festen Datentyp und können Werte verschiedener Datentypen annehmen. Variabelnamen müssen immer mit einem Buchstaben oder Bodenstrich (_) anfangen. Variabeln dürfen nur Buchstaben, Ziffern und Bodenstriche (_) beinhalten.

Beispiel:

```
<?php
        $var = 2;
        $var = 'Hello World';
?>
```

Ausgabe

Mit dem Befehl 'echo' kann mit PHP zusätzlich auf die Webseite geschrieben werden.

Beispiel:

```
<?php
        $var = 'World';
        echo '<p>Hello' . $var . '</p>';
?>
```

Den zu ausgebenden Text kann auch mit HTML Tags umgeben werden, um diesen Tag in den HTML-Code einzufügen. Die Tags können auch mit den jeweiligen Attributen ergänzt werden.

Beispiel:

```
<?php
        echo '<p>Hello World</p>';
        echo '<a href="https://www.google.com">Hello World</a>';
?>
```

Funktionen

Funktionen werden ähnlich wie in anderen Sprachen deklariert. Der Unterschied ist aber, dass kein Return-Type angegeben wird.

Auch können Funktionsargumente definiert werden, indem sie zwischen die runden Klammern geschrieben werden. Es können Datentypen bei der Deklaration der Argumente angegeben werden, es wird aber nicht überprüft, ob

Es kann ein Return-Value ausgegeben werden, PHP wird automatisch den Return-Type zum Value assoziieren.

Beispiel:

```php
<?php
	function helloWorld() {
		echo 'Hello World';
	}

	helloWorld();
?>
```

Schleifen und Bedingungen

If, if – else, if – elseif – else, switch, (do) while loops, for loops, etc. werden alle von PHP unterstützt.

Hier einige Beispiele:

```php
<?php
	//if – elseif – else
	$t = date('H');

	If ($t < '10') {
		echo 'Have a good morning!';
	} elseif ($t < '20') {
		echo 'Have a good day!';
	} else {
		echo 'Have a good night!';
	}
?>
```

```php
<?php
	//switch case
	$favcolor = 'red';

	switch ($favcolor) {
		case 'red':
			echo 'Your favorite color is red!';
			break;
		case 'blue':
			echo 'Your favorite color is blue!';
			break;
		case 'green':
			echo 'Your favorite color is green!';
			break;
		default:
			echo 'Your favorite color is neither red, blue, nor green!';
			break;
	}
?>
```

```php
<?php
	//do while loop
	$x = 1;

	do {
		echo 'The number is: ' . $x . '<br>';
		$x++;
	} while ($x <= 5);
?>
```

```php
<?php
	//for loop
	for ($x = 0; $x <= 10; $x++) {
		echo 'The number is: ' . $x . '<br>';
	}
?>
```

Kommentare in PHP

Kommentare können mit „//“ oder mit „#“ sind nur für eine Zeile geltend, im Gegensatz zu „/* …… */“, dies ist für mehrzeilige Kommentare gedacht.

```php
<?php
        //single-line comment
        #single-line comment
        /*multi
        line
        comment
        */

        //comment in code
        $x = 5 /*+ 15*/ + 5;
        echo $x;
?>
```

Quellen

Informationen:

https://de.wikipedia.org/wiki/PHP
https://en.wikipedia.org/wiki/PHP
https://www.php.net/manual/de/
https://www.w3schools.com/php/default.asp
https://www.freelancinggig.com/blog/2018/04/23/major-differences-php-5-php-7/
https://www.geeksforgeeks.org/php-5-vs-php-7/
https://www.php-einfach.de/2019/02/migration-von-php5-auf-php7-leicht-gemacht/