Engineer

PHP

Sprache, auf der der größte Teil des älteren Webs läuft

Was es ist

PHP ist eine Skriptsprache, die auf dem Webserver ausgeführt wird: Eine Anfrage trifft ein, der PHP-Code läuft, und im Browser kommt das erzeugte HTML an, niemals der Code selbst. Die Buchstaben stehen heute für „PHP: Hypertext Preprocessor“, ein rekursives Akronym. Ursprünglich bedeuteten sie „Personal Home Page“. Eine PHP-Datei kann gewöhnliches HTML mit Code zwischen den Tags `<?php` und `?>` sein oder reiner Code, durch den ein Framework jede Anfrage leitet. PHP führt auch Skripte auf der Kommandozeile aus, ist aber auf das Erzeugen von Webseiten ausgerichtet.

Die Geschichte beginnt 1994, als Rasmus Lerdorf einige CGI-Binärprogramme in C schrieb, um die Besuche seines Online-Lebenslaufs zu zählen. Er nannte sie Personal Home Page Tools. Im Juni 1995 veröffentlichte er den Quellcode, und aus den Werkzeugen wurde PHP/FI. 1997 begannen Andi Gutmans und Zeev Suraski, den Parser neu zu schreiben. Das Ergebnis erschien im Juni 1998 unter dem rekursiven Namen als PHP 3.0. Auf seinem Höhepunkt war es auf rund 10 % der Webserver im Internet installiert. Ihre nächste Engine, die nach beiden benannte Zend Engine, bildete im Mai 2000 die Grundlage für PHP 4.0. PHP 5 folgte im Juli 2004.

Auf heutigen Servern läuft PHP meist über PHP-FPM, den mitgelieferten FastCGI Process Manager. Ein vorgeschalteter Webserver übergibt jede PHP-Anfrage an einen Pool von Worker-Prozessen. Jeder Pool kann mit eigenem Benutzer und eigener Gruppe, eigener Umgebung und eigenen php.ini-Werten laufen. Sein Prozessmanager hält eine feste Worker-Zahl vor, startet sie bei Bedarf oder variiert ihre Zahl innerhalb festgelegter Grenzen. Die Sprache entwickelt sich in Versionszweigen, von denen die letzten vier im Jahresabstand erschienen. Jeder Zweig erhält zwei Jahre aktive Unterstützung mit Fehler- und Sicherheitskorrekturen, danach zwei weitere Jahre nur kritische Sicherheitskorrekturen und erreicht anschließend sein Lebensende. PHP 8.2,

veröffentlicht am 8. Dezember 2022, erhält Sicherheitskorrekturen bis zum 31. Dezember 2026. PHP 8.5, erschienen am 20. November 2025, bis zum 31. Dezember 2029.

PHP wandelt Typen um, wenn eine Operation es verlangt. Am deutlichsten zeigt sich das beim losen Vergleichsoperator `==`. Sind beide Seiten numerische Zeichenfolgen oder ist eine Seite eine Zahl und die andere eine numerische Zeichenfolge, erfolgt ein numerischer Vergleich. Vor PHP 8.0.0 wurde jede mit einer Zahl verglichene Zeichenfolge zuerst in eine Zahl umgewandelt. Deshalb war `0 == "a"` wahr. Der strikte Operator `===` vergleicht neben dem Wert auch den Typ und wandelt nichts um. Das Handbuch selbst empfiehlt ihn für die meisten Fälle.

```
<?php
var_dump(0 == "a");      // PHP 7: true, PHP 8: false
var_dump("1" == "01");   // true in both: two numeric strings
var_dump("10" == "1e1"); // true in both: 10 equals 1e1
var_dump(100 == "1e2");  // true in both
var_dump(0 === "a");     // false in both: === never converts
```

Das Konzept für eine Entscheidung nutzen

Nutzen Sie „Was ist PHP?“ für eine konkrete Entscheidung. Wir können helfen, das Konzept mit dem System in Verbindung zu bringen, das Sie entwickeln oder einkaufen. Der Teil, den Nutzer nie sehen: wo die Daten liegen, welche Regeln gelten und wie andere Systeme darauf zugreifen dürfen.

Unsere Leistungen entdecken Kontakt aufnehmen →

Die Annahmen hinter dem Begriff prüfen

Nutzen Sie die Erklärung und die verlinkten Arbeiten, um Fragen für Ihr Projekt zu formulieren. Wir können die Annahmen gemeinsam mit Ihnen prüfen, bevor ein Werkzeug ausgewählt wird. 6 ETL/ELT-Pipelines entworfen und betrieben — 6 ETL-Pipelines über drei Datenbankengines hinweg

6 ETL/ELT-Pipelines entworfen und betrieben Kontakt aufnehmen →

Warum es wichtig ist

PHP ist vor allem wegen der bereits darin geschriebenen Software wichtig. WordPress und Joomla sind PHP-Anwendungen. Viele Organisationen nutzen diese Content-Management-Systeme, um selbst bearbeitbare Websites zu betreiben. WordPress empfiehlt PHP 8.3 oder höher, Joomla 6 verlangt mindestens PHP 8.3.0. WordPress läuft noch mit PHP 7.4, obwohl seine Anforderungsseite darauf hinweist, dass solche Versionen ihr Lebensende erreicht haben. Ein Unternehmen kann jahrelang von PHP abhängen, ohne dass jemand dort eine Zeile der Sprache geschrieben hat. Deshalb gerät die

darunterliegende Version in Vergessenheit. Nach dem Lebensende ihres Zweigs liefert die Website weiterhin Seiten aus. Sie erhält lediglich keine Korrekturen mehr, ohne dass der Bildschirm darauf hinweist.

Auch Upgrades verändern das Verhalten unauffällig. PHP 8.0 änderte den Vergleich von Zeichenfolgen mit Zahlen. Eine unter PHP 7 wahre Bedingung kann unter PHP 8 ohne Fehlermeldung falsch werden. PHP 8.4 erhöhte den Standardkostenfaktor von bcrypt in `password_hash()` von 10 auf 12. Dadurch wird jeder neue Passwort-Hash auf derselben Hardware aufwendiger zu berechnen. Beides ist dokumentiert und gelangt nur durch ein Upgrade in die Produktion, das jemand planen, testen und verantworten muss.

Unsere Empfehlung ist, die PHP-Version wie eine Abhängigkeit mit daneben notiertem Ablaufdatum zu behandeln. Eine bestehende PHP-Anwendung, die funktioniert, sollte erhalten bleiben: Bringen Sie sie auf einen unterstützten Zweig, planen Sie das nächste Upgrade deutlich vor Ende der Sicherheitsunterstützung und geben Sie ihr einen eigenen FPM-Pool und eigenen Systembenutzer. Wählen Sie PHP für ein neues Vorhaben, wenn der Kunde die Website mit WordPress oder Joomla selbst bearbeiten soll oder wenn das nach der Übergabe zuständige Team PHP gut kennt. Entscheiden Sie andernfalls nach den Fähigkeiten dieses Teams. Das Risiko ist selten die Sprache. Es ist eine Version, für die niemand Verantwortung trägt.

Wie es in der Praxis aussieht

Die Webschicht unseres Technologie-Stacks weist PHP einen bestimmten Platz zu. Seiten, die Dateien sein können, werden statisch gebaut. Für diese Seiten muss bei einer Anfrage kein Anwendungscode ausgeführt werden. Diese Website wird mit Hugo gebaut. Django und PHP kommen dort zum Einsatz, wo eine Anwendung Anfragen bedienen muss. Dieselbe Seite beschreibt die Auswahlkriterien: das Problem, die Betriebsfähigkeiten des Kundenteams nach unserer Übergabe und das Fehlerverhalten statt der Vorführwirkung. Diese Regel gilt unmittelbar für PHP: Wenn der vorhandene Stack des Kunden die richtige Antwort ist, verpflichtet sich die Seite, das auch zu sagen.

Von 2015 bis 2022 arbeitete die Gründerin als Teamleiterin für GIS-Softwareentwicklung und begleitete mehr als dreißig GIS-Projekte durch Entwicklung, Bereitstellung und Support im laufenden Betrieb. PHP war neben JavaScript, Python und SQL eine von vier Sprachen ihrer Anwendungsschicht. Darunter lagen sämtliche räumlichen Daten in PostgreSQL mit PostGIS; davor

standen Karten mit Mapbox GL und Leaflet, von denen manche die APIs von ArcGIS oder HERE aufriefen. Unabhängig von der Sprache reichte die Verantwortung über die Auslieferung hinaus. Probleme wurden nicht weitergereicht: Man entwirft anders, wenn man weiß, dass man selbst bei einem Ausfall den Anruf erhält.

Wo es schiefgeht

- **Ein loser Vergleich schützt ein Geheimnis.** Ein mit `==` geprüftes Token oder ein solcher Hash wird nach PHPs Zahlenregeln verglichen, sobald beide Seiten wie Zahlen aussehen. Verschiedene Zeichenfolgen wie `"10"` und `"1e1"` gelten deshalb als gleich. Verwenden Sie `password_verify()` für Passwörter und `hash_equals()` für andere Geheimnisse. Dessen Laufzeit verrät nichts über den Inhalt des Geheimnisses; es erwartet die Zeichenfolge des Nutzers als zweites Argument. Verwenden Sie überall sonst `===` .
- **SQL wird aus Anfragedaten zusammengesetzt.** Vorbereitete PDO-Anweisungen quotieren ihre Parameter. Laut Handbuch ist eine Anwendung, die ausschließlich diese verwendet, vor SQL-Injection geschützt. Die ausdrücklich genannte Ausnahme sind Abfrageteile, die weiterhin aus nicht maskierter Eingabe zusammengesetzt werden. Ein Platzhalter steht für einen vollständigen Wert. Alles, was kein Wert ist, etwa die Sortierspalte einer Liste, landet deshalb weiterhin in der Zeichenfolge. Prüfen Sie diese Teile gegen eine feste Liste im Code.
- **Passwort-Hashes werden zu schmal gespeichert.** `PASSWORD_DEFAULT` ist heute bcrypt. Das Handbuch weist darauf hin, dass sich dieser Standard mit neuen, stärkeren Algorithmen ändern soll. Es empfiehlt eine über 60 Byte hinaus erweiterbare Spalte; 255 sind eine gute Wahl. Bcrypt kürzt außerdem das Passwort auf 72 Byte. Speichern Sie den gesamten Hash und rufen Sie nach erfolgreicher Anmeldung `password_needs_rehash()` auf, damit alte Hashes bei der Anmeldung der Nutzer aktualisiert werden.
- **Der Zweig überschreitet sein Lebensende.** Die Anwendung läuft weiter, sodass niemand das Upgrade plant. Der spätere Sprung überspannt dann mehrere Zweige zugleich. Halten Sie das Ende der Sicherheitsunterstützung des verwendeten Zweigs fest, aktualisieren Sie jeweils einen Zweig weiter und führen Sie die Testsuite gegen den nächsten aus, bevor der aktuelle abläuft.
- **Der FPM-Pool ist gemeinsam genutzt und zu offen konfiguriert.** Bei einem Pool, der mehrere Anwendungen unter einem Benutzer bedient, kann eine Schwachstelle in einer Anwendung Zugriff auf die Dateien aller Anwendungen ermöglichen. `security.limit_extensions` steht standardmäßig auf `.php .phar` . Das Handbuch empfiehlt eine Begrenzung auf `.php` , damit ein falsch konfigurierter Webserver keine anderen Dateien an PHP übergeben kann. Geben Sie jeder Anwendung einen eigenen Pool und Benutzer und setzen Sie `pm.max_children` passend zum verfügbaren Serverspeicher.

Weiterführende Lektüre

- PHP-Handbuch: Was ist PHP? — die eigene Kurzdefinition des Projekts mit dem ersten Beispiel für in HTML eingebettetes PHP.
- PHP: Unterstützte Versionen — die Unterstützungsregeln sowie die Zeiträume aktiver und reiner Sicherheitsunterstützung aller aktuellen Zweige.
- PHP-Handbuch: Vergleichsoperatoren — loser und strikter Vergleich, die Ausgaben von PHP 7 und PHP 8 nebeneinander sowie die Vergleichstabelle nach Typ.
- PHP-Handbuch: password_hash — Algorithmen, Kostenfaktor, Spaltenbreite, 72-Byte-Grenze und die Änderung des Standardkostenfaktors in 8.4.
- PHP-Handbuch: Vorbereitete Anweisungen und gespeicherte Prozeduren — wie PDO Parameter bindet und wo SQL-Injection möglich bleibt.
- PHP-Handbuch: FPM-Konfiguration — Pool-Direktiven, Prozessmanagermodi, `clear_env` und `security.limit_extensions`.

Glossar

Für Ihre tatsächlichen Rahmenbedingungen auswählen

Beschreiben Sie den Arbeitsablauf, die vorhandenen Schnittstellen und die Nutzer, die die Veränderung benötigen. Wir können eine prüfbare Lieferung definieren und die Übergabe so gestalten, dass Ihr Team die gelieferten Bestandteile warten kann.

Preise ansehen Kontakt aufnehmen →

Die Begriffe praktisch nutzbar machen

Fragen Sie, wie „Was ist PHP?“ auf Ihre Organisation zutrifft. Ein konkretes Beispiel aus Ihrem bestehenden System reicht aus, um das Gespräch zu beginnen.

Unsere Leistungen entdecken

welcome@engineer.company +45 50 10 61 84 Kopenhagen, Dänemark

Nehmen Sie diese Seite mit

Kontaktkarte

Mit Sorgfalt und Liebe gestaltet von D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – heute · Engineer ApS · Alle Rechte vorbehalten.

Gedruckt von https://engineer.company/de/glossary/php/