Engineer

PHP

språket det meste av det gamle nettet kjører på

Hva det er

PHP er et skriptspråk som kjøres på webserveren: en forespørsel kommer inn, PHP-koden kjører, og det som når nettleseren, er HTML-en den produserte, aldri selve koden. Bokstavene står i dag for «PHP: Hypertext Preprocessor», et rekursivt akronym, men de begynte som «Personal Home Page». En PHP-fil kan være vanlig HTML med kode plassert mellom taggene `<?php` og `?>`, eller ren kode som et rammeverk sender hver forespørsel gjennom. Språket kjører også skript fra kommandolinjen, men utformingen heller mot å lage nettsider.

Historien begynner i 1994, da Rasmus Lerdorf skrev et sett CGI-binærfiler i C for å telle besøkene på CV-en sin på nettet og kalte dem Personal Home Page Tools. Han ga ut kildekoden i juni 1995, og verktøyene vokste til PHP/FI. I 1997 begynte Andi Gutmans og Zeev Suraski å skrive om parseren, og resultatet kom ut som PHP 3.0 i juni 1998 under det rekursive navnet; på det meste var det installert på omtrent 10 % av webserverne på internett. Den neste motoren deres, Zend Engine, oppkalt etter dem begge, bar PHP 4.0 i mai 2000, og PHP 5 fulgte i juli 2004.

På en server i dag kjøres PHP som regel av PHP-FPM, FastCGI Process Manager, som følger med språket. En webserver foran sender hver PHP-forespørsel videre til en pulje (pool) med arbeidsprosesser. Hver pulje kan kjøre som sin egen bruker og gruppe, med sitt eget miljø og egne verdier i php.ini, og prosessbehandleren holder enten et fast antall arbeidsprosesser, starter dem ved behov eller lar antallet variere mellom to grenser. Språket utvikles i utgivelsesgrener (release branches), og de fire siste kom med ett års mellomrom. Hver gren får to års aktiv støtte, med rettinger av både feil og sikkerhetshull, deretter to år til med bare kritiske sikkerhetsrettinger, og så når den end of life, slutten av levetiden. PHP 8.2, utgitt 8. desember 2022, får sikkerhetsrettinger til 31. desember 2026; PHP 8.5, utgitt 20. november 2025, får dem til 31. desember 2029.

PHP konverterer mellom typer når en operasjon krever det, og det er i den løse sammenligningsoperatoren `==` det viser seg tydeligst. Er begge sider numeriske strenger, eller den ene et tall og den andre en numerisk streng, blir sammenligningen numerisk. Før PHP 8.0.0 ble enhver streng som ble sammenlignet med et tall, først gjort om til et tall, så `0 == "a"` var sann. Den strenge operatoren `===` sammenligner typen i tillegg til verdien og konverterer ingenting, og manualen anbefaler selv å foretrekke den i de fleste tilfeller.

```
<?php
var_dump(0 == "a");      // PHP 7: true, PHP 8: false
var_dump("1" == "01");   // true i begge: to numeriske strenger
var_dump("10" == "1e1"); // true i begge: 10 er lik 1e1
var_dump(100 == "1e2");  // true i begge
var_dump(0 === "a");     // false i begge: === konverterer aldri
```

Bruk begrepet i en beslutning

Bruk «Hva er PHP?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Delen brukerne aldri ser: hvor dataene ligger, hvilke regler som gjelder, og hvordan andre systemer får lov til å snakke med den.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Designet og driftet 6 ETL/ELT-pipelines — 6 ETL-pipelines på tvers av tre databasemotorer

Designet og driftet 6 ETL/ELT-pipelines Ta kontakt →

Hvorfor det er viktig

PHP betyr mest på grunn av det som allerede er skrevet i det. WordPress og Joomla, publiseringssystemene svært mange organisasjoner bruker til nettsteder de redigerer selv, er PHP-applikasjoner: WordPress anbefaler PHP 8.3 eller nyere, og Joomla 6 krever minst PHP 8.3.0. WordPress kjører fortsatt på PHP 7.4, men siden med systemkravene advarer om at slike versjoner har nådd slutten av levetiden. En virksomhet kan være avhengig av PHP i årevis uten at noen der har skrevet en eneste linje i språket, og derfor blir versjonen under glemt. Et nettsted fortsetter å levere sider etter at grenen har nådd end of life; det slutter bare å få rettinger, og ingenting på skjermen sier fra.

Oppgraderinger endrer også oppførsel, i det stille. PHP 8.0 endret hvordan strenger sammenlignes med tall, så en betingelse som var sann på PHP 7, kan være usann på PHP 8 uten noen feilmelding. PHP 8.4 hevet standardkostnaden for bcrypt i `password_hash()` fra 10 til 12, noe som gjør hver ny passordhash

dyrere å regne ut på den samme maskinvaren. Begge endringene er dokumentert, og hver av dem når produksjon bare gjennom en oppgradering som noen må planlegge, teste og eie.

Vår anbefaling er å behandle PHP-versjonen som en avhengighet med en utløpsdato skrevet ved siden av. En eksisterende PHP-applikasjon som virker, er verdt å beholde: flytt den til en gren som har støtte, planlegg neste oppgradering i god tid før sikkerhetsstøtten tar slutt, og gi den sin egen FPM-pulje og sin egen systembruker. Ved et nytt prosjekt bør dere velge PHP når jobben er et nettsted kunden skal redigere i WordPress eller Joomla, eller når teamet som skal drifte det etter overleveringen, kan PHP godt, og ellers velge ut fra hva det teamet kan. Språket er sjelden risikoen; en versjon ingen eier, er det.

Slik ser det ut i praksis

Weblaget på siden om teknologiene vi bruker gir PHP en bestemt plass. Sider som kan være filer, bygges statisk, slik at disse sidene ikke krever kjøring av applikasjonskode for hver forespørsel, og dette nettstedet bygges av Hugo; Django og PHP brukes der det er en applikasjon å levere. Den samme siden sier hvordan valget tas: etter problemet, etter hva kundens team kan drifte etter at vi har gått, og etter hvordan tingen svikter, ikke hvordan den tar seg ut i en demo. Den regelen gjelder PHP direkte: når det riktige svaret er stakken kunden allerede har, lover siden å si det.

Fra 2015 til 2022 arbeidet grunnleggeren som teamleder for GIS-programvareutvikling og førte mer enn tretti GIS-prosjekter gjennom utvikling, produksjonssetting og driftsstøtte etter at de var satt i drift. De romlige dataene lå i PostgreSQL med PostGIS, og webkartene var bygd på Mapbox GL og Leaflet, noen ganger mot API-ene til ArcGIS eller HERE, med applikasjonslaget i JavaScript, Python, PHP og SQL. PHP var ett av fire språk i det laget, og hovedpoenget på siden gjelder hvert av dem: ansvaret strakk seg forbi leveransen, problemer ble ikke sendt videre, og du designer annerledes når du vet at det er du som blir ringt opp hvis noe ryker.

Hvor det går galt

- **En løs sammenligning vokter en hemmelighet.** Et token eller en hash som sjekkes med `==`, sammenlignes etter PHPs tallregler hver gang begge sider ser ut som tall, så to ulike strenger som `"10"` og `"1e1"` regnes som like. Bruk `password_verify()` for passord og `hash_equals()` for andre hemmeligheter: kjøretiden avslører ingenting om innholdet i hemmeligheten, og funksjonen forventer brukerens streng som andre argument. Bruk `===` overalt ellers.
- **SQL settes sammen av data fra forespørselen.** PDO tar selv hånd om siteringen av parameterne i forberedte spørringer (prepared statements), og manualen sier at en applikasjon som bare bruker dem, er beskyttet mot SQL-injeksjon, med ett uttalt unntak: de delene av spørringen som fortsatt bygges av input som ikke er escapet. En plassholder står for en hel verdi, så alt som ikke er en verdi, f.eks. kolonnen en liste sorteres etter, havner fortsatt i strengen. Sjekk de delene mot en fast liste i koden.
- **Passordhasher lagres for smalt.** `PASSWORD_DEFAULT` er bcrypt i dag, og manualen advarer om at den er laget for å endres etter hvert som sterkere algoritmer kommer til. Den anbefaler en kolonne som kan vokse ut over 60 byte, med 255 som et godt valg. Bcrypt kutter dessuten passordet ved 72 byte. Lagre hele hashen, og kall `password_needs_rehash()` etter en vellykket innlogging, så eldre hasher oppgraderes etter hvert som folk logger inn.
- **Grenen passerer datoen for end of life.** Applikasjonen fortsetter å virke, så ingen planlegger oppgraderingen, og hoppet som til slutt kommer, spenner over flere grener på én gang. Noter datoen da sikkerhetsstøtten tar slutt for grenen som er i bruk, oppgrader én gren om gangen, og kjør testsuiten mot neste gren før den nåværende går ut.
- **FPM-puljen er delt og løst satt opp.** Én pulje som betjener flere applikasjoner som én bruker, lar en feil i den ene nå filene til alle. `security.limit_extensions` står som standard på `.php .phar`, og manualen råder til å begrense den til `.php`, så en feilkonfigurert webserver aldri sender andre filer til PHP. Gi hver applikasjon sin egen pulje og bruker, og sett `pm.max_children` til det minnet på serveren kan bære.

Les mer

- PHP Manual: What is PHP? — prosjektets egen korte definisjon, med det første eksempelet på PHP bygd inn i HTML.
- PHP: Supported Versions — støttepolicyen og datoene for aktiv støtte og ren sikkerhetsstøtte for hver gren som fortsatt får rettinger.
- PHP Manual: Comparison Operators — løs og streng sammenligning, resultatene fra PHP 7 og PHP 8 side om side, og sammenligningstabellen ordnet etter type.

- PHP Manual: password_hash — algoritmer, kostnad, kolonnebredde, grensen på 72 byte og den endrede standardkostnaden i 8.4.
- PHP Manual: Prepared statements and stored procedures — hvordan PDO binder parametere, og hvor SQL-injeksjon fortsatt er mulig.
- PHP Manual: FPM configuration — direktivene for puljer, modusene til prosessbehandleren, `clear_env` og `security.limit_extensions`.

Ordliste

Velg ut fra deres faktiske rammer

Beskriv arbeidsflyten, eksisterende grensesnitt og brukerne som trenger endringen. Vi kan definere en testbar leveranse og en overlevering som gjør teamet i stand til å vedlikeholde løsningen.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er PHP?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

welcome@engineer.company +45 50 10 61 84 København, Danmark

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/php/