Engineer

PHP

Lenguaje en el que funciona la mayor parte de la web antigua

Qué es

PHP es un lenguaje de scripts que se ejecuta en el servidor web: llega una solicitud, se ejecuta el código PHP y al navegador le llega el HTML que produjo, nunca el propio código. Las letras significan hoy «PHP: Hypertext Preprocessor», un acrónimo recursivo, aunque al principio significaban «Personal Home Page». Un archivo PHP puede ser HTML normal con código entre las etiquetas `<?php` y `?>`, o código puro por el que un framework dirige cada solicitud. También ejecuta scripts desde la línea de comandos, pero su diseño se orienta a producir páginas web.

Su historia empieza en 1994, cuando Rasmus Lerdorf escribió un conjunto de binarios CGI en C para contar las visitas a su currículum en línea y los llamó Personal Home Page Tools. Publicó el código fuente en junio de 1995, y las herramientas evolucionaron hasta PHP/FI. En 1997, Andi Gutmans y Zeev Suraski empezaron a reescribir el analizador; el resultado se publicó como PHP 3.0 en junio de 1998 con el nombre recursivo. En su momento de mayor difusión estaba instalado en aproximadamente el 10 % de los servidores web de internet. Su siguiente motor, Zend Engine, cuyo nombre combina los de ambos, impulsó PHP 4.0 en mayo de 2000; PHP 5 llegó en julio de 2004.

En los servidores actuales, PHP suele ejecutarse mediante PHP-FPM, el gestor de procesos FastCGI que se distribuye con él. Un servidor web situado delante pasa cada solicitud PHP a un grupo de procesos de trabajo. Cada grupo puede ejecutarse con su propio usuario y grupo del sistema, su entorno y sus valores de php.ini; su gestor mantiene un número fijo de procesos, los inicia bajo demanda o varía su cantidad entre unos límites. El lenguaje avanza por ramas de versiones, y las cuatro últimas llegaron con un año de diferencia. Cada rama recibe dos años de soporte activo, con correcciones de errores y seguridad; después, otros dos años solo de correcciones de seguridad críticas, y entonces llega al fin de su vida

útil. PHP 8.2, publicado el 8 de diciembre de 2022, recibe correcciones de seguridad hasta el 31 de diciembre de 2026; PHP 8.5, publicado el 20 de noviembre de 2025, hasta el 31 de diciembre de 2029.

PHP convierte entre tipos cuando una operación lo requiere, y el operador de comparación débil `==` es donde más se aprecia. Si ambos lados son cadenas numéricas, o uno es un número y el otro una cadena numérica, la comparación es numérica. Antes de PHP 8.0.0, cualquier cadena comparada con un número se convertía primero en un número, por lo que `0 == "a"` era verdadero. El operador estricto `===` compara tanto el tipo como el valor y no convierte nada; el propio manual recomienda preferirlo en la mayoría de los casos.

```php
<?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
```

Utilice el concepto en una decisión

Aplique «¿Qué es PHP?» a una decisión real. Podemos ayudarle a relacionar el concepto con el sistema que está construyendo o comprando. La parte que el usuario no ve: dónde se guardan los datos, qué reglas se aplican y cómo pueden comunicarse otros sistemas con ella.

Explore nuestros servicios Póngase en contacto →

Compruebe los supuestos detrás del término

Utilice la explicación y los trabajos enlazados para formular preguntas sobre su proyecto. Podemos examinar los supuestos con usted antes de elegir una herramienta. Diseño y operación de 6 flujos ETL/ELT — 6 flujos ETL en tres motores de bases de datos

Diseño y operación de 6 flujos ETL/ELT Póngase en contacto →

Por qué importa

PHP importa sobre todo por lo que ya está escrito en él. WordPress y Joomla, los sistemas de gestión de contenidos con los que muchas organizaciones mantienen sitios que editan ellas mismas, son aplicaciones PHP: WordPress recomienda PHP 8.3 o superior, y Joomla 6 exige al menos PHP 8.3.0. WordPress aún funciona con PHP 7.4, aunque su página de requisitos advierte de que esas versiones han llegado al fin de su vida útil. Una empresa puede depender de PHP durante años sin que nadie de la empresa haya escrito una línea en él; por eso se olvida la versión que lo sustenta. Un sitio sigue sirviendo páginas cuando su rama llega al fin de vida: simplemente deja de recibir correcciones, y nada en pantalla lo indica.

Las actualizaciones también cambian el comportamiento sin avisar. PHP 8.0 cambió cómo se comparan las cadenas con los números, de modo que una condición verdadera en PHP 7 puede ser falsa en PHP 8 sin ningún error. PHP 8.4 elevó el coste bcrypt predeterminado de `password_hash()` de 10 a 12, lo que encarece el cálculo de cada hash de contraseña nuevo con el mismo hardware. Ambos cambios están documentados, y cada uno llega a producción solo mediante una actualización que alguien debe planificar, probar y asumir.

Recomendamos tratar la versión de PHP como una dependencia con una fecha de caducidad escrita a su lado. Conviene conservar una aplicación PHP existente que funciona: pásela a una rama con soporte, programe la siguiente actualización con suficiente antelación al fin del soporte de seguridad y asígnele su propio grupo FPM y usuario del sistema. Para un desarrollo nuevo, elija PHP si el trabajo es un sitio que el cliente editará con WordPress o Joomla, o si el equipo que lo mantendrá tras la entrega conoce bien PHP. En los demás casos, decida según las capacidades de ese equipo. El lenguaje rara vez es el riesgo; una versión sin responsable sí lo es.

Cómo se aplica en la práctica

La capa web de nuestro conjunto de tecnologías sitúa PHP en un lugar concreto. Las páginas que pueden ser archivos se generan de forma estática, lo que evita ejecutar una aplicación al atender sus solicitudes, y este sitio se construye con Hugo; Django y PHP se usan cuando hay que atender las solicitudes de una aplicación. La misma página explica cómo se decide: por el problema, por lo que el equipo del cliente podrá operar cuando nos marchemos y por cómo falla el sistema, no por cómo luce en una demostración. La regla incluye directamente a PHP: si la respuesta correcta es el conjunto de tecnologías que el cliente ya tiene, la página se compromete a decirlo.

De 2015 a 2022, la fundadora trabajó como responsable del equipo de desarrollo de software SIG, al frente de más de treinta proyectos SIG durante su desarrollo y despliegue, y su posterior soporte una vez en funcionamiento. PHP era uno de los cuatro lenguajes de su capa de aplicación, junto con JavaScript, Python y SQL; por debajo, todos los datos espaciales estaban en PostgreSQL con PostGIS y, delante, había mapas de Mapbox GL y Leaflet, algunos de los cuales llamaban a las API de ArcGIS o HERE. Fuera cual fuera el lenguaje, la responsabilidad continuaba después de la entrega y los problemas no se pasaban a otros: se diseña de otra manera cuando se sabe que uno mismo recibirá la llamada si algo falla.

Dónde surgen los problemas

- **Una comparación débil protege un secreto.** Un token o hash comprobado con `==` se compara según las reglas numéricas de PHP si ambos lados parecen números, de modo que dos cadenas distintas como `"10"` y `"1e1"` se consideran iguales. Utilice `password_verify()` para las contraseñas y `hash_equals()` para otros secretos: su tiempo de ejecución no revela nada sobre el contenido del secreto y espera la cadena de la persona usuaria como segundo argumento. Utilice `===` en los demás casos.
- **El SQL se construye con datos de la solicitud.** Las sentencias preparadas de PDO entrecomillan sus parámetros, y el manual afirma que una aplicación que solo las utiliza está protegida contra la inyección SQL, con una excepción expresa: cualquier parte de la consulta que aún se construya con entradas sin escapar. Un marcador de posición representa un valor completo, así que lo que no sea un valor, como la columna por la que se ordena una lista, sigue formando parte de la cadena. Compruebe esas partes frente a una lista fija en el código.
- **Los hashes de contraseñas se almacenan en columnas demasiado estrechas.** `PASSWORD_DEFAULT` es bcrypt hoy, y el manual advierte de que está diseñado para cambiar a medida que se añadan algoritmos más robustos. Recomienda una columna que pueda superar los 60 bytes y propone 255 como una buena elección. Bcrypt también trunca la contraseña a 72 bytes. Guarde el hash completo y llame a `password_needs_rehash()` tras un inicio de sesión correcto para actualizar los hashes antiguos a medida que las personas se conectan.
- **La rama supera su fecha de fin de vida.** La aplicación sigue funcionando, nadie programa la actualización y el salto final abarca varias ramas de golpe. Registre la fecha de fin del soporte de seguridad de la rama utilizada, actualice una rama cada vez y ejecute la batería de pruebas contra la siguiente antes de que caduque la actual.
- **El grupo FPM es compartido y está poco restringido.** Un grupo que atiende a varias aplicaciones con un solo usuario permite que un fallo en una dé acceso a los archivos de todas. `security.limit_extensions` tiene el valor predeterminado `.php .phar`, y el manual recomienda limitarlo a `.php` para que un servidor web mal configurado no pase otros archivos a PHP. Dé a cada aplicación su propio grupo y usuario, y ajuste `pm.max_children` a lo que pueda alojar la memoria del servidor.

Lecturas adicionales

- Manual de PHP: ¿qué es PHP? — la definición breve del proyecto, con el primer ejemplo de PHP incrustado en HTML.

- PHP: versiones con soporte — la política de soporte y las fechas de soporte activo y solo de seguridad de cada rama actual.
- Manual de PHP: operadores de comparación — comparación débil y estricta, las salidas de PHP 7 y PHP 8 en paralelo y la tabla de comparación por tipo.
- Manual de PHP: password_hash — algoritmos, coste, anchura de columna, el límite de 72 bytes y el cambio del coste predeterminado en 8.4.
- Manual de PHP: sentencias preparadas y procedimientos almacenados — cómo enlaza PDO los parámetros y dónde sigue siendo posible la inyección SQL.
- Manual de PHP: configuración de FPM — directivas de los grupos, modos del gestor de procesos, `clear_env` y `security.limit_extensions`.

Glosario

Elija según sus restricciones reales

Describa el flujo de trabajo, las interfaces existentes y los usuarios que necesitan el cambio. Podemos definir una entrega comprobable y un traspaso que permita a su equipo mantener lo entregado.

Vea cuánto cuesta Póngase en contacto →

Haga práctica la terminología

Pregunte cómo se aplica «¿Qué es PHP?» a su organización. Basta un ejemplo concreto de su sistema actual para iniciar la conversación.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

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

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://engineer.company/es/glossary/php/