¿Qué es PHP?

PHP es un lenguaje de programación para desarrollar aplicaciones y crear sitios web que conquista cada día más seguidores. Fácil de usar y en constante perfeccionamiento es una opción segura para aquellos que desean trabajar en proyectos calificados y sin complicaciones.

Con los años, principalmente debido a su accesibilidad, el lenguaje PHP ha ganado muchos seguidores, formando una gran comunidad de apoyo.

Por lo tanto, quienes lo utilizan para programar pueden resolver dudas, aprender más y estar en constante desarrollo.

También existen muchas ventajas al usar el lenguaje PHP, que también refuerzan este escenario tan positivo.

La programación es una actividad muy valorada y con gran demanda en el mercado, ya que la transformación digital es una realidad, especialmente para páginas web, blogs y otros canales web.

Sin embargo, aquellas personas que no tienen tanto conocimiento sobre lenguajes de programación, generalmente no saben qué significa PHP.

Con los años, principalmente debido a su accesibilidad, el lenguaje PHP ha ganado muchos seguidores, formando una gran comunidad de apoyo.

Por lo tanto, quienes lo utilizan para programar pueden resolver dudas, aprender más y estar en constante desarrollo.

También existen muchas ventajas al usar el lenguaje PHP, que también refuerzan este escenario tan positivo.

¿Qué es PHP y cómo funciona?

PHP es un lenguaje de programación destinado a desarrollar aplicaciones para la web y crear páginas web, favoreciendo la conexión entre los servidores y la interfaz de usuario.

Entre los factores que hicieron que PHP se volviera tan popular, se destaca el hecho de que es de código abierto.

Esto significa que cualquiera puede hacer cambios en su estructura. En la práctica, esto representa dos cosas importantes:

No hay restricciones de uso vinculadas a los derechos. El usuario puede usar PHP para programar en cualquier proyecto y comercializarlo sin problemas.
está en constante perfeccionamiento, gracias a una comunidad de desarrolladores proactiva y comprometida.
El PHP generalmente es definido como un lenguaje del lado del servidor. Esto significa que se aplica en la programación que tiene lugar en el servidor web responsable de ejecutar la aplicación o, más a menudo, en un sitio web.

Este trabajo previo permite cargar los elementos de una página antes de mostrarlos al usuario que accede a un sitio web, por ejemplo.

El código PHP se ejecuta en el servidor que, al leer los comandos, puede activar todos los elementos funcionales y la interfaz visual del sitio web.

La simplicidad para aprender a usarlo y el desarrollo del código abierto le facilita el trabajo a los profesionales que eligen estructurar sitios web utilizando la plataforma, debido a que a medida que avanzan las configuraciones y ediciones se simplifican aún más.

¿En qué situaciones se puede utilizar PHP?
No es muy difícil entender qué es PHP, pero se vuelve más simple saber su utilidad y operación cuando usamos ejemplos concretos de aplicaciones.

Básicamente, como dijimos, su uso es para la web, gracias a su capacidad de conectar el servidor y la interfaz de usuario, tomando todo el código HTML.

Hoy en día, una gran parte de las grandes compañías globales que son parte de nuestra vida cotidiana tienen el PHP como base de sus aplicaciones.

Aplicaciones en sitios web

Una de las características principales de PHP es que es un lenguaje mucho más dinámico que la mayoría de las otras opciones que existen.

Por lo tanto, es esencial para desarrollar sitios que tienen aplicaciones más complejas y, para eso, necesitamos dos cosas: agilidad en el tiempo de respuesta y conexión a una gran base de datos. Por ejemplo, ¡nada menos que Facebook nació con PHP!

La idea de usar este lenguaje es disminuir el tiempo de carga de las páginas, permitiendo que el servidor trabaje con más suavidad para cargar plugins y aplicaciones en los sitios web.

De esta manera, es posible desarrollar con agilidad sitios con un gran rendimiento, incluso si están llenos de recursos, y con la garantía de la sostenibilidad del desempeño a largo plazo utilizando el lenguaje PHP

Mejoras en PHP 8

La mejora más importante, que nos aporta el PHP 8 con respecto a versiones anteriores aunque en el PHP 7.4 hubiera una versión experimental, el compilador JIT (Just in Time).

Como sabemos PHP es un lenguaje interpretado y cuando se ejecuta es cuando se convierte a código máquina. Esta diferencia hace que el PHP no tenga el mismo rendimiento que los lenguajes compilados (java, c), pero esta mejora ha nacido para mejorar este rendimiento.

¿Qué es el compilador JIT (Just in Time)?

El compilador JIT (Just in Time) nos permite compilar en tiempo de ejecución ciertas partes del código, para su posterior reutilización en futuras peticiones por parte del usuario.
¿Qué ganamos con ello y qué lo hace tan importante?
Debido que al evitar nuevas conversiones del código interpretado a código máquina, esto nos hace mejorar el rendimiento de las aplicaciones en algunos casos ¡¡¡hasta 4 veces!!!.

Esto nos abre 2 nuevas vías de mejoras muy importantes. Una de ellas es mejorar el rendimiento de cualquier proyecto realizando con los diversos Framework de PHP como son, Symfony, Laravel, WordPress, Drupal… etc. Aunque en un proyecto web no suele repetirse un código muchas veces, donde sí vamos a ver y notar esta mejora es cuando se ejecuten procesos de cálculos muy pesados y repetitivos, cálculos de KPI, informes, etc…

El compilador JIT también nos va a permitir y no deja de ser importante, que, al ser multiplataforma, funciona tanto en Linux, Windows o Mac, esto abre el camino a poder realizar aplicaciones PHP fuera del mundo Web como tal, con el que únicamente ha sido vinculado.

¿Cómo funciona?

Como hemos indicado solo compila ciertas partes del código ,¿cómo sabe esta nueva funcionalidad qué código tiene que compilar?

Esto se controla mediante un proceso interno que monitoriza la vigilancia del código durante su ejecución y cuando detecta que se ejecuta varias veces esta parte determinada del código, la marca como candidata a compilar. Al realizar las compilaciones de este código repetitivo, y las guarda en caché (extensión OPcache) y la próxima vez que se requieran estas partes del código no sirve el código interpretado si no que sirven el código ya compilado, mejorando el rendimiento enormemente como hemos comentado anteriormente estas operaciones ya que han sido compiladas anteriormente.

Variables en PHP

Las variables son uno de los primeros temas a conocer en PHP y en la mayoría de los lenguajes de programación.

Conceptos y tipos de datos que podremos encontrar en el lenguaje PHP:

Anteriormente en JavaScript, ya introducimos el concepto de variable. No obstante podemos entender una variable como un dato almacenado en una referencia. Técnicamente una variable apunta a una posición de la memoria, donde se almacena un dato. Las variables se utilizan en los lenguajes de programación para darle un nombre a ese dato, a esa posición de la memoria, de manera que se pueda entender o saber lo que contiene. Al final, esos datos almacenados son los que se utilizan para conseguir los resultados de los programas.

Por su parte, un tipo de datos es la característica de un dato almacenado. Es decir, si el dato posee información numérica, alfanumérica, etc. La mayoría de los lenguajes clasifican de alguna manera los tipos de datos, aunque algunos son más permisivos que otros a la hora de realizar operaciones con variables de distintos tipos.

Signo $ en el nombre de las variables

Para PHP, las variables eran definidas comenzando siempre por el símbolo $.

Las variables siempre deberían tener un nombre descriptivo sobre lo que ellas van a almacenar. Por tanto, al nombre de una variable en PHP le colocaremos el símbolo $ y un nombre que ayude a entender la funcionalidad del dato almacenado en el sistema que se está desarrollando.

Por ej:

<?php $totalticket = 300 ?>

Tipos de datos en PHP

Dependiendo de la información que contenga, una variable puede ser considerada de diferentes tipos:

Variables numéricas

Este tipo de variables almacena cifras, números, que pueden tener dos clasificaciones distintas:

Enteros $entero=2002; //Números sin decimales

Reales $real=3.14159; //Números con o sin decimal

Variables alfanuméricas

Este tipo de datos almacena textos compuestos, cadenas de caracteres, que pueden contener letras, símbolos y números o cifras.

Cadenas

Almacenan variables alfanuméricas por ej: $cadena="Feliz año 2020!";

Booleanas

Este tipo de variables almacena un valor lógico, que puede valer verdadero o falso. Es muy común en la programación este tipo de variables booleanas.

Booleano verdadero $verdadero = true;

Booleano falso $falso = false;

Matrices, tablas o arrays

Es un tipo de datos en el que, en lugar de tener un dato, podemos almacenar un conjunto de ellos, a los que accedemos a través de índices. Cada una de las casillas de un array o los datos de nuestra matriz a su vez poseen informaciones numéricas, alfanuméricas etc.

Arrays Son las variables que guardan las tablas

$sentido[1]="ver";

$sentido[2]="tocar";

$sentido[3]="oir";

$sentido[4]="gusto";

$sentido[5]="oler";

Objetos

Se trata de conjuntos de variables y funciones asociadas. Presentan una complejidad mayor que las variables vistas hasta ahora pero su utilidad es más que interesante. Entraremos con detalle en los objetos más adelante.

PHP tiene tipado dinámico

A diferencia de otros lenguajes, PHP posee una gran flexibilidad a la hora de operar con variables. En efecto, cuando definimos una variable asignándole un valor, el ordenador le atribuye un tipo. Si por ejemplo definimos una variable entre comillas, la variable será considerada de tipo cadena:

```
$variable="5"; //esto es una cadena
```

Sin embargo, si pedimos en nuestro script realizar una operación matemática con esta variable, no obtendremos un mensaje de error sino que la variable cadena será asimilada a numérica (PHP hará todo lo posible por interpretar nuestra operación):

```
<?

$cadena="5"; //esto es una cadena

$entero=3; //esto es un entero

echo $cadena+$entero

?>
```

Este script dará como resultado "8". La variable cadena ha sido asimilada en entero (aunque su tipo sigue siendo cadena) para poder realizar la operación matemática. Del mismo modo, podemos operar entre variables tipo entero y real. No debemos preocuparnos de nada, PHP se encarga durante la ejecución de interpretar el tipo de variable necesario para el buen funcionamiento del programa.

Nota: Los lenguajes como PHP que permiten mayor flexibilidad en los tipos de las variables se dicen que tienen tipado dinámico. En ellos una variable puede tener distintos tipos a lo largo de su vida, es decir, a medida que el programa se ejecuta una variable podrá cambiar de tipo. Generalmente durante el procesamiento del programa se va infiriendo los tipos de las variables, en tiempo de ejecución, según el tipo de datos del valor que se le asigna o las operaciones que se realizan sobre ellas. Otra manera de referirse a este tipo de lenguajes de programación es "levemente tipados", aunque esta segunda denominación es menos correcta, porque puede inducir a una comprensión errónea, ya que en la realidad las variables siempre tienen tipos, aunque estos puedan variar con el tiempo.

PHP es sensible a las mayúsculas y minúsculas

PHP entiende de manera distinta las mayúsculas y minúsculas. En el caso del nombre que le damos a una variable, no es lo mismo escribirla con mayúscula o minúscula, o mezclando mayúsculas y minúsculas de distinta manera. Por tanto, hay que tener mucho cuidado a la hora de escribir los nombres de variables, y no cambiar mayúsculas por minúsculas, ya que PHP entenderá dos variables distintas aunque nosotros podamos intentar referirnos a la misma. Cuando estamos empezando quizás sea un buen consejo trabajar asignando nombres a las variables siempre en minúsculas, para evitar este tipo de malentendidos a veces muy difíciles de localizar.

En el caso que tengamos una variable con un nombre compuesto de varias palabras, en PHP es una práctica común colocar la variable toda en minúscula y separar las palabras por guiones bajos.

<?php $mi_variable = "me gusta PHP" ?

Variables de sistema en PHP

PHP es un lenguaje que se ejecuta en el servidor, bajo demanda de un cliente. Por tanto, la ejecución de PHP se produce dentro de un marco muy concreto, donde intervienen varios actores, principalmente el cliente (generalmente el usuario que entra usando su navegador) y el servidor (donde se ejecuta el código PHP, que básicamente debe producir la salida que se enviará al cliente).

Dentro de una página PHP tendremos por tanto acceso a toda una serie de variables que nos informan sobre nuestro servidor y sobre el cliente que ha solicitado una determinada página. A estas informaciones, que podemos recoger en forma de variables, las llamamos "variables de sistema".

La información de estas variables es atribuida por el servidor y muchas varían de valor según el momento en que se consultan los datos que almacena.

$_SERVER

La mayoría de las variables de sistema las podemos recibir a partir de un array denominado $_SERVER.

$_SERVER es un array asociativo, cuyos índices son cadenas de texto y no números.

Técnicamente $_SERVER se conoce como una "variable superglobal" y posee una multitud de datos asociados al array $_SERVER, algunos sin utilidad aparente y otros realmente interesantes y con una aplicación directa para nuestras aplicaciones web. Enumeramos algunas de estas variables y la información que nos aportan:

$_SERVER["HTTP_USER_AGENT"] Nos informa principalmente sobre el sistema operativo y tipo y versión de navegador utilizado por el internauta. Su principal utilidad radica en que, a partir de esta información, podemos redireccionar nuestros usuarios hacia páginas optimizadas para su navegador o realizar cualquier otro tipo de acción en el contexto de un navegador determinado.

$_SERVER["HTTP_ACCEPT_LANGUAGE"] Nos devuelve la o las abreviaciones de la lengua considerada como principal por el navegador. Esta lengua o lenguas principales pueden ser elegidas en el menú de opciones del navegador. Esta variable resulta también extremadamente útil para enviar al internauta a las páginas escritas en su lengua, si es que existen.

$_SERVER["HTTP_REFERER"] Nos indica la URL desde la cual el internauta ha tenido acceso a la página. Muy interesante para generar botones de "Atrás" dinámicos o para crear nuestros propios sistemas de estadísticas de visitas.

$_SERVER["PHP_SELF"] Nos devuelve una cadena con la URL del script que está siendo ejecutado. Muy interesante para crear botones para recargar la página.

$_SERVER["HTTP_GET_VARS"] Se trata de un array que almacena los nombres y contenidos de las variables enviadas al script por URL o por formularios GET.

$_SERVER["HTTP_POST_VARS"] Se trata de un array que almacena los nombres y contenidos de las variables enviadas al script por medio de un formulario POST.

$_SERVER["HTTP_COOKIE_VARS"] Se trata de un array que almacena los nombres y contenidos de las cookies. Veremos qué son más adelante.

$_SERVER["PHP_AUTH_USER"] Almacena la variable usuario cuando se efectúa la entrada a páginas de acceso restringido. Combinado con $_SERVER["PHP_AUTH_PW"] resulta ideal para controlar el acceso a las páginas internas del sitio.

$_SERVER["PHP_AUTH_PW"] Almacena la variable password cuando se efectúa la entrada a páginas de acceso restringido. Combinado con $_SERVER["PHP_AUTH_USER"] resulta ideal para controlar el acceso a las páginas internas del sitio.

$_SERVER["REMOTE_ADDR"] Muestra la dirección IP del visitante.

$_SERVER["DOCUMENT_ROOT"] Nos devuelve el path físico en el que se encuentra alojada la página en el servidor.

$_SERVER["PHPSESSID"] Guarda el identificador de sesión del usuario. Veremos más adelante en qué consisten las sesiones.

No todas estas variables están disponibles en la totalidad de servidores o en determinadas versiones de un mismo servidor. además, algunas de ellas han de ser previamente activadas o definidas por medio de algún acontecimiento. Así, por ejemplo, la variable $HTTP_REFERER no tendrá un valor definido, a menos que el usuario acceda al script a partir de un enlace desde otra página.

Si quieres ver cuál es el conjunto completo de las variables del sistema que dispones dentro de $_SERVER en tu entorno, es suficiente con escribir y ejecutar una página PHP que contenga este código:

```
<?php

var_dump($_SERVER);

?>
```

Eso realizará un listado de todo el contenido del array asociativo $_SERVER y lo mostrará como salida en la página web.

Variables superglobales

A partir de PHP 4.1.0, se dispone de un conjunto de variables de tipo array que mantienen información del sistema, llamadas "superglobales" porque se definen automáticamente en un ámbito global y a las que se puede acceder desde cualquier punto del código PHP.

Estas variables ya existían anteriormente en PHP, aunque se accedían desde otros arrays.

La lista de estas variables superglobales de PHP es la siguiente:

$GLOBALS

Contiene una referencia a cada variable disponible en el espectro de las variables del script. Las llaves de esta matriz (índices del array) son los nombres de las variables globales. $GLOBALS existe dese PHP 3.

$_SERVER

Variables definidas por el servidor web ó directamente relacionadas con el entorno en don el script se esta ejecutando. Es equivalente a lo que antes se conocía como $HTTP_SERVER_VARS. Son las variables de sistema que hemos explicado antes en este artículo.

$_GET

Variables proporcionadas al script por medio de HTTP GET. Es equivalente a lo que antes se conocía como $HTTP_GET_VARS.

$_POST

Variables proporcionadas al script por medio de HTTP POST. Es equivalente a lo que antes se conocía como $HTTP_POST_VARS.

$_COOKIE

Variables proporcionadas al script por medio de HTTP cookies. Es equivalente a lo que antes se conocía como $HTTP_COOKIE_VARS.

$_FILES

Variables proporcionadas al script por medio de la subida de ficheros via HTTP . Es equivalente a lo que antes se conocía como $HTTP_POST_FILES.

$_ENV

Variables proporcionadas al script por medio del entorno. Es equivalente a lo que antes se conocía como $HTTP_ENV_VARS.

$_REQUEST

Variables proporcionadas al script por medio de cualquier mecanismo de entrada del usuario. La presencia y el orden en que aparecen las variables en esta matriz es definido por la directiva de configuración variables_order. Esta matriz no tiene un análogo en versiones anteriores a PHP 4.1.0.

$_SESSION

Variables registradas en la sesión del script. Es equivalente a lo que antes se conocía como $HTTP_SESSION_VARS. Vea también la sección Funciones para el manejo de sesiones para más información.