**¿Qué es PHP?**

Las páginas web se crean mediante HTML, y este es un lenguaje muy limitado para atender a los requerimientos que actualmente se exigen. Por ello han aparecido numerosas extensiones al lenguaje que permiten enriquecer las páginas web.

Para evitar problemas de compilación por el lado del cliente, dichos lenguajes y mejoras aparecieron en el lado del servidor. El cliente recibirá una página HTML normal y será el servidor el que traduzca el código script.

Ahora mismo PHP se trata del lenguaje script de servidor más popular, puesto que es interpretable en casi todos los servidores de aplicaciones web. Es un lenguaje basado en C y en Perl, que se ha diseñado pensando en darle la máxima versatilidad y facilidad de aprendizaje, por encima de la rigidez y coherencia semántica. Sus ventajas son:

* Es multiplataforma (Windows, Linux, Mac,…)
* Abierto y gratuito (de GNU de Linux)
* Gran comunidad de usuarios
* Diseñado para Apache y MySQL
* Extensiones
* Fácil de aprender

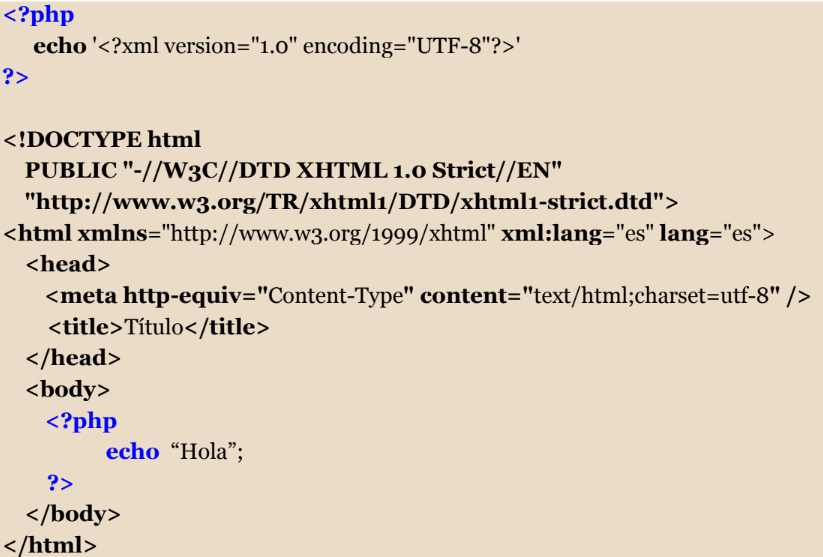
**¿Qué necesito para usar PHP?**

* Un servidor web compatible con PHP (recomendable Apache).
* Motor PHP: el intérprete que procesa y ejecuta código PHP en un servidor web, permitiendo la generación de contenido web dinámico.
* IDE para PHP: aunque se puede escribir código PHP en cualquier editor de texto, un buen entorno de desarrollo integrado facilita la edición de código (con coloreado especial de las palabras de PHP, corrección del código en línea, abreviaturas de código, plantillas,...), su prueba y depuración.
* Depurador PHP: Software que se añade al módulo PHP para darle la capacidad de depurar el código. Muchos IDEs tienen la suficiente capacidad como para sustituir esta herramienta.

El código PHP se escribe en un documento de texto indicando la extensión .php

El código PHP se añade entre las etiquetas <?php y ?> .

El código PHP se coloca en la zona de la página web donde más nos interese hacerlo. Un ejemplo de documento PHP podría ser (suponiendo que incrustamos PHP en un documento de tipo XHTML 1.0 estricto):



**El primer código PHP es necesario** porque actualmente se consideran las páginas web XHTML como un documento XML más y por ello deben de llevar la cabecera <?xml versión=”1.0” encoding=”utf-8”?> obligatoria en todo documento XML. EL problema es que ese código en el servidor web daría problemas porque podría entender que es código PHP (al empezar igual que la etiqueta <?php) e intentaría traducirlo. Por ello es mejor escribirlo con la función echo para explicar que es código del documento final HTML y no PHP.

En PHP se pueden hacer tres tipos de **comentarios**: tipo C (/\* esto es un comentario \*/), C++ ( // esto es un comentario ) y ShellScript ( #esto es un comentario ).

Las **normas básicas** para escribir lenguaje PHP se basan en los lenguajes padres de este, es decir C y Perl. Son:

* Todas las líneas de código deben de finalizar con un punto y coma
* Se puede agrupar el código en bloques que se escriben entre llaves
* Una línea de código se puede partir o sangrar (añadir espacios al inicio) para que sea más legible siempre y cuando no partamos una palabra o un valor.
* PHP obliga a ser estricto con las mayúsculas y las minúsculas en algunos casos como el nombre de las variables; sin embargo con las palabras reservadas del lenguaje no es estricto. Es decir: entiende WHILE, while e incluso wHiLe
* Para concatenar se usa la coma ( , ) o el punto ( **.** )

Aunque hay muchas funciones de escritura (para escribir en lo que será la página final) las fundamentales son echo y print.

echo es la más utilizada y en realidad es un comando del lenguaje. Tras echo se pasa uno o más textos (expresiones de cadena) que cuando son literales se escriben entre comillas. print funciona casi igual, pero tiene dos importantes diferencias: Sólo permite un texto (no varios), y devuelve un valor verdadero o falso dependiendo de si se pudo escribir el texto o no (en código complejo es útil).

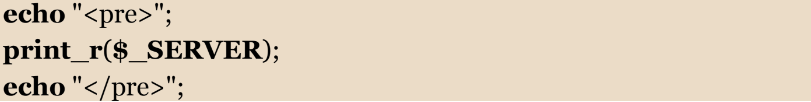
Las **variables** ocupan un espacio en la memoria RAM del servidor que ejecute el código para almacenar el dato al que se refieren. Cuando utilizamos el nombre de la variable, realmente estamos haciendo referencia a un dato que está en memoria. Como no es necesario declararlas, se puede poner directamente que $edad=15 sin necesidad de especificar el tipo de dato que almacena. Además, siempre cumplirán que:

* Empiezan por el símbolo $
* El segundo carácter puede ser guión bajo \_ o bien una letra.
* A partir del tercer carácter pueden incluirse números
* No hay límite de tamaño en el nombre, pero no puede haber espacios en blanco

El servidor web que aloje las páginas PHP pone a disposición del programador *variables de sistema* ya definidas para su uso en el programa. Todas suelen empezar por barra baja \_ además de escribirse en mayúsculas, y suelen ser arrays. Un ejemplo:



Esa variable ($\_SERVER[“SERVER\_PORT”];) escribirá el número de puerto por el que se comunica el servidor web. Para mostrar todas las variables predefinidas en el servidor podemos escribir el siguiente código en una página PHP:



Las etiquetas pre muestra la información formateada. Por su parte, print\_r es una función para mostrar el contenido de los arrays.

La función error\_reporting(E\_ALL); nos avisa de todos los errores.

Tipos de datos posibles:

* Enteros decimales, octales o hexadecimales (17, 071, 0XA2BC)
* Coma flotante, con o sin notación científica (23.4, 12.3e-4)
* Cadenas o Strings(“Jorge”, ‘Antonio dijo “hola” al llegar’)

Formas de castear tipos de datos:

* (int) o (integer). Convierte a entero
* (real), (double) o (flat). Convierte a coma flotante
* (string). Convierte a forma de texto
* (array). Convierte a forma de array
* (object). Convierte a un objeto

Caracteres especiales ***(sólo funcionan entre comillas dobles)***:

Para definir una **constante** utilizaré define(“nombre”, valor); las constantes no necesitan el signo $ de las variables. Se recomienda definir las constantes en MAYÚSCULAS.