MySQL

1. Entorno MySQL
2. PhpMyAdmin
3. Workbench.
4. Acceso a un servidor de Base de datos de prueba.
5. Crear, eliminar y seleccionar una base de datos.
6. Crear, eliminar y seleccionar una tabla.
7. Concepto de Entidad, atributo y tipo de datos.
8. Estructura de una consulta sql y Cláusula SELECT
9. Alias de Tablas
10. Cláusula FROM y Cláusula WHERE
11. Cláusula Insert / Delete y Update

En el desarrollo de un proyecto web, podemos publicar páginas estáticas o dinámicas. ¿Cuáles son las diferencias entre cada una?

Las páginas web dinámicas son generadas en el servidor por la ejecución del lenguaje Script sobre el cual se encuentran programadas y el resultado es una salida HTML enviada al navegador. Sin embargo, una web estática es un archivo de texto HTML que envía el servidor hacia el navegador, por lo que requiere menor procesamiento del lado servidor, ya que solo necesita localizar el archivo HTML, leerlo y emnviarlo al navegador del usuario (cliente).

Las páginas web dinámicas no se hicieron necesarias hasta el momento en el que crecieron tantas opciones de administración de datos y multimedia que surgió la necesidad de separar el contenido en una estructura mejor adaptada para su administración y lo mejor fue usar una base de datos para gestionarlo.

PÁGINAS WEB ESTÁTICAS

Son ideales para las entidades que no quieren muchas pretensiones con su sitio web, sino simplemente informar a sus clientes de sus productos y dar a conocer su perfil de organizacional entre otros. La principal ventaja es lo económico que pueden resultar. Su diseño y desarrollo es más rápido que el de una web dinámica pero el usuario no puede seleccionar, ordenar o modificar los contenidos o el diseño de la página. Sus características más relevantes son:

- Ausencia de actualización de datos.
- Realizadas generalmente HTML / CSS.
- Es necesario acceder al servidor para cambiar contenidos de la página.
- Sin opciones de búsquedas del visitante en la página.
- Artesanal y manual proceso de actualización de información.

PÁGINAS WEB DINÁMICAS

Estas permiten la creación de aplicaciones dentro de la propia web y ofrecen una mayor interactividad con los usuarios que la visiten. Su creación es más compleja, ya que se requiere de conocimientos específicos de lenguajes de programación y gestión de bases de datos. Con este tipo de páginas web se puede manejo de datos, multimedia, edicion de fotografía etc. Es realmente una web dinámica , esa en la cual los usuarios interactúan con la información contenida en la página, ya que dicha información varía en tiempo real de acuerdo a las opciones tomadas por el usuario. Las características son:

- Gran cantidad de posibilidades en su diseño y desarrollo.
- El visitante puede alterar el diseño, contenidos o presentación.
- Se utilizan varias técnicas de programación.

- El proceso de actualización es más sencillo.
- Permite muchas funcionalidades como bases de datos, foros, etc.
- Consumo de Api´s y web services proporcionados por otras páginas web.

En definitiva, si una entidad u organización solo quiere mostrar su historia, productos o servicios (algo más institucional), lo mejor sería una página web estática, pero si lo que necesitas es una web mediante la cual puedas hacer ventas, recibir pagos, almacenar estadísticas, etc, la elección debe estar orientada al desarrollo de una página web dinámica.

¿Por qué eligimos PHP?

Para comenzar porque con PHP es posible desarrollar cualquier cosa. PHP está enfocado principalmente a la programación de scripts del lado del servidor, por lo que se puede hacer cualquier cosa que pueda hacer otro programa CGI, como recopilar datos de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies. Aunque PHP puede hacer mucho más. Existen principalmente tres campos principales donde se usan scripts de PHP.

- Scripts del lado del servidor. Este es el campo más tradicional y el foco principal. Son necesarias tres cosas para que esto funcione: el analizador de PHP (módulo CGI o servidor), un servidor web y un navegador web. Es necesario ejecutar el servidor con una instalación de PHP conectada. Se puede acceder al resultado del programa de PHP con un navegador, viendo la página de PHP a través del servidor. Todo esto se puede ejecutar en tu máquina con la programación de PHP.

- Scripts desde la línea de comandos. Se puede crear un script de PHP y ejecutarlo sin necesidad de un servidor o navegador. Solamente es necesario el analizador de PHP para utilizarlo de esta manera. Este tipo de uso es ideal para scripts que se ejecuten con regularidad empleando cron (en *nix o Linux) o el Planificador de tareas (en Windows). Estos scripts también pueden usarse para tareas simples de procesamiento de texto.

- Escribir aplicaciones de escritorio. Probablemente PHP no sea el lenguaje más apropiado para crear aplicaciones de escritorio con una interfaz gráfica de usuario, pero si se conoce bien PHP, y se quisiera utilizar algunas características avanzadas de PHP en aplicaciones del lado del cliente, se puede utilizar PHP-GTK para escribir dichos programas. También es posible de esta manera escribir aplicaciones independientes de una plataforma. PHP-GTK es una extensión de PHP, no disponible en la distribución principal. Si estás interesado en PHP-GTK, podés visitar su sitio web.
PHP puede utilizarse en todos los sistemas operativos principales, incluyendo Linux, muchas variantes de Unix (incluyendo HP-UX, Solaris y OpenBSD), Microsoft Windows, macOS, RISC OS y probablemente otros más. PHP admite la mayoría de servidores web de hoy en día, incluyendo Apache, IIS, y muchos otros. Esto incluye cualquier servidor web que pueda utilizar el binario de PHP FastCGI, como lighttpd y nginx. PHP funciona tanto como módulo como procesador de CGI. De modo que con PHP, se tiene la libertad de elegir el sistema operativo y el servidor web. Además, se tiene la posibilidad de utilizar programación por procedimientos o programación orientada a objetos (POO), o una mezcla de ambas.
Con PHP estás limitado a generar HTML. Entre las capacidades de PHP se incluyen la creación de imágenes, ficheros PDF e incluso películas Flash (usando libswf y Ming) generadas sobre la marcha. También se puede generar fácilmente cualquier tipo de texto, como XHTML y cualquier otro tipo de fichero XML. PHP puede autogenerar estos ficheros y guardarlos en el sistema de ficheros en vez de imprimirlos en pantalla, creando una caché en el lado del servidor para contenido dinámico.
Una de las características más potentes y destacables de PHP es su soporte para una gran cantidad de diferentes bases de datos. Escribir una página web con acceso a una base de datos es increíblemente simple utilizando una de las extensiones específicas de bases de datos (p.ej., para mysql), o utilizar una capa de abstracción como PDO, o conectarse a cualquier base de datos que admita el estándar de Conexión Abierta a Bases de Datos por medio de la extensión ODBC.

PHP también cuenta con soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) y muchos otros. También se pueden crear sockets de red puros e interactuar usando cualquier otro protocolo. PHP tiene soporte para el intercambio de datos complejos de WDDX entre virtualmente todos los lenguajes de programación web. Y hablando de interconexión, PHP tiene soporte para la instalación de objetos de Java y emplearlos de forma transparente como objetos de PHP.
PHP tiene útiles características de procesamiento de texto, las cuales incluyen las expresiones regulares compatibles con Perl (PCRE), y muchas extensiones y herramientas para el acceso y análisis de documentos XML. PHP estandariza todas las extensiones XML sobre el fundamento sólido de libxml2, y amplía este conjunto de características añadiendo soporte para SimpleXML, XMLReader y XMLWriter.

En resumen, como podrás apreciar, estas lineas no son suficiente para enumerar todas las características y beneficios que ofrece PHP, por lo que siempre te invitamos a investigar más en la web y ahondar conocimientos en este enorme y robusto lenguaje.

¿Qué es XAMPP?

XAMPP incluye los productos: Apache + MariaDB + PHP + Perl

Sin dudas, XAMPP es el entorno más popular de desarrollo con PHP y es una distribución de Apache completamente gratuita y fácil de instalar que contiene MariaDB, PHP y Perl. El paquete de instalación de XAMPP ha sido diseñado para ser increíblemente fácil de instalar y usar.

Que es Apache?

Apache HTTP Server es un software de servidor web gratuito y de código abierto para plataformas Unix con el cual se ejecutan el 46% de los sitios web de todo el mundo. Es mantenido y desarrollado por la Apache Software Foundation.

Le permite a los propietarios de sitios web servir contenido en la web, de ahí el nombre de «servidor web». Es uno de los servidores web más antiguos y confiables, con la primera versión lanzada hace más de 20 años, en 1995.

Cuando alguien quiere visitar un sitio web, ingresa un nombre de dominio en la barra de direcciones de su navegador. Luego, el servidor web envía los archivos solicitados actuando como un repartidor virtual.

Que es MaríaDB?

MariaDB Server es una de las bases de datos relacionales de código abierto más populares. Está hecho por los desarrolladores originales de MySQL y así se garantiza que seguirá siendo de código abierto. Es parte de la mayoría de las ofertas en la nube y es el producto predeterminado en la mayoría de las distribuciones de Linux.

Se basa en los valores de rendimiento, estabilidad y apertura, y MariaDB Foundation garantiza que las contribuciones serán aceptadas por méritos técnicos. La nueva funcionalidad reciente incluye agrupación avanzada con Galera Cluster 4, características de compatibilidad con Oracle Database y Temporal Data Tables, lo que permite consultar los datos tal como estaban en cualquier momento del pasado.

Que es Perl?

Perl es un lenguaje de propósito general originalmente desarrollado para la manipulación de texto y que ahora es utilizado para un amplio rango de tareas incluyendo administración de sistemas, desarrollo web, programación en red, desarrollo de GUI y más. Apache incluye también la posibilidad de utilizar este lenguaje.

Podés obtener su versión para Windows / Linux / Mac desde su página oficial: https://www.apachefriends.org/es/index.html

Para ser más gráficos, te invitamos a ver un video de como instalar XAMPP en una pc con sistema operativo Windows:

Primeros Pasos en PHP

En esta unidad vamos a explicar la sintaxis básica y cómo en una página HTML podemos mezclar el código del lenguaje de marcación (HTML) con el código del lado del servidor (PHP). Vas a ver que es muy sencillo, motivo por el cual a los desarrolladores que ya saben HTML les resulta fácil comenzar con PHP. Además veremos algunas cosas básicas y consejos para que tu código PHP se pueda ejecutar perfectamente en cualquier tipo de servidor.

Apertura y cierre del código PHP
PHP se escribe dentro de la propia página web, junto con el código HTML y, como para cualquier otro tipo de lenguaje incluido en un código HTML, en PHP necesitamos especificar cuáles son las partes del código escritas en este lenguaje. Esto se hace, como en otros casos, delimitando nuestro código por etiquetas de apertura y cierre. Podemos utilizar distintos modelos de etiquetas en función de nuestras preferencias y costumbres.

Estos son los modos de abrir y cerrar las etiquetas que delimitan el código PHP:

<? y ?>
<?php y ?>

El modo de funcionamiento de una página PHP, a grandes rasgos, no difiere del clásico para una página dinámica de lado servidor: El servidor va a reconocer la extensión correspondiente a la página PHP (Generalmente .php, pero podría configurarse el servidor para que busque código PHP en otras extensiones de archivo...) para ejecutar los bloques de scripts PHP.

El servidor, antes de enviar la página al navegador se encargará de interpretar y ejecutar todo aquello que se encuentre entre las etiquetas correspondientes al lenguaje PHP. El resto, lo enviara sin más ya que, asumirá que se trata de código HTML absolutamente comprensible por el navegador.

En PHP la apertura del código con el tag en su versión corta (<?) no se encuentra siempre activado por defecto. Es algo que depende del servidor y de la versión de PHP que esté instalada en él. Para evitar problemas debidos a la plataforma donde se ejecuta PHP no te recomendamos utilizarlo. No obstante, si tienes la oportunidad de alterar la configuración del lenguaje PHP (mediante la edición del archivo php.ini correspondiente, del que hablaremos en otro momento), podrías definir que también se interprete esa etiqueta mediante la directiva "short-open-tags".

Uso de ; para delimitar sentencias
Otra característica general de los scripts en PHP es la forma de separar las distintas instrucciones. Para hacerlo, hay que acabar cada instrucción con un punto y coma ";". Para la ultima expresión, la que va antes del cierre de etiqueta, este formalismo no es necesario.

<?php echo 'código PHP' ?>
Aunque la sentencia "echo" anterior (que sirve para escribir desde PHP salida en la propia página) no acaba en ";" el código es perfectamente válido, porque inmediatamente después tenemos el cierre del script PHP.

Comentarios en PHP
La sintaxis de comentarios, funcionan muy similares a los de otros lenguajes como Java, C o Javascript.

Nota: Un comentario, es una frase o palabra que nosotros incluimos en el código para comprenderlo más fácilmente al volverlo a leer un tiempo después y que, por supuesto, el ordenador tiene que ignorar ya que no va dirigido a su ejecución, sino a nosotros mismos u otros desarrolladores que puedan leer ese código más adelante. Los comentarios tienen una gran utilidad ya que es muy fácil olvidarse del funcionamiento de un script programado un tiempo atrás y resulta muy útil si queremos hacer rápidamente comprensible nuestro código a otra persona.

Veamos esto con un primer ejemplo de script:

```
<?php
$mensaje="Hola Mundo!!"; //Comentario de una linea
echo $mensaje; #Este comentario también es de una linea
/*En este caso
mi comentario ocupa
varias lineas, lo ves? */
?>
```

Si usamos doble barra (//) o el símbolo # podemos introducir comentarios de una línea. Mediante /* y */ creamos comentarios multilínea. Por supuesto, nada nos impide de usar estos últimos en una sola línea.

Las variables en PHP se definen anteponiendo un símbolo de dólar ($) y la instrucción echo sirve para sacar en pantalla lo que hay escrito a continuación.

Recordamos que todo el texto insertado en forma de comentario es completamente ignorado por el servidor. Resulta importante acostumbrarse a dejar comentarios, es algo que se agradece con el tiempo.

Ejemplo completo de una página PHP
Ahora veamos un código completo de lo que podría ser una página PHP. Comenzamos con una página básica escrita con HTML en la que hemos insertado código PHP. El código de momento es lo de menos, lo importante es ver cómo se integra el código PHP en una página HTML.

```
<!doctype html>
<html lang="es">
<head>
   <meta charset="UTF-8">
   <title>Primera página PHP</title>
</head>
<body>
   <h1>Esto es HTML</h1>

   <?php
         echo '<p>Esto viene de PHP</p>';
   ?>

</body>
</html>
```

Para poder probar esta página PHP deberás nombrarla con extensión ".php". Podría ser algo como "pag1.php" o "index.php". Luego tendrás que colocarla en el directorio de publicación de tu

servidor ("document root" en inglés), cuyo depende de cuál sea el servidor que estés usando para poder comenzar con PHP. Normalmente esa carpeta es c:\Xampp\htdocs\ . Luego, teniendo el servidor Apache encendido, tendrás que acceder a la página a través de tu navegador web con la dirección "http://localhost/index.php" según lo mostramos en el video anterior. Aunque esto depende mucho de cómo tengas configurado tu entorno de trabajo.

Configuración de PHP.
El archivo php.ini

PHP generalmente funciona bien sin necesidad de hacer cambios en su configuración, pero los creadores de PHP hicieron posible poder manipular el comportamiento del lenguaje para las necesidades de cada desarrollador. El archivo de configuración php.ini permite alterar el comportamiento de PHP relacionado con directorios, sesiones, parámetros de las bases de datos y extensiones.

El archivo se divide en secciones, cada una con variables relacionadas con esa sección:

[MiSección]

variable1="valor1"

variable2="valor2"

Como en las variables de php, son sensibles a mayúsculas y no pueden contener espacios. Los valores pueden ser numéricos, strings o booleanos. Los booleanos pueden ser: true, on o yes y false, off, no o none.

El punto y coma al principio de cada línea indica que es un comentario, siendo así fácil poder activar y desactivar características.

El archivo de configuración php.ini se lee cuando arranca PHP. Si funciona como módulo de servidor (Ej: Apache), sólo se leerá cuando se inicia el servidor, por lo que será necesario reiniciarlo para que se produzcan los cambios.

Dónde y como cambiar la configuración en PHP

Algunos ajustes pueden realizarse directamente en el script PHP con ini_set(), otros es necesario hacerlos en el php.ini o httpd.conf.

Por ejemplo el ajuste output_buffering es PHP_INI_PERDIR, por lo que no puede establecerse usando _initset. En cambio display_errors es PHP_INI_ALL y se puede establecer desde cualquier lugar, incluyendo init_set().

Ejecutar PHP como un módulo Apache

Cuando se usa PHP como un módulo de Apache, se puede cambiar la configuración desde httpd.conf y .htaccess. Se necesitan los privilegios AllowOverride Options o AllowOverride All para poder hacerlo.

Configuración del parser

Una de las configuraciones más importantes es la del intérprete del lenguaje. La primera opción es la variable que controla el motor de PHP, que ha de ser On o Off. Si se cambia a Off el código PHP no será analizado por el servidor.

engine = On

La variable _short_open_tag_ controla si el parser debería reconocer las etiquetas <?...?>, además de <?php...?>. Si esta etiqueta genera conflictos con otros lenguajes, es mejor que esté en Off.

short_open_tag = Off

Ajustes de seguridad

Antes de PHP 5.4 existían la directivas _safemode, que limitaban al usuario las cosas que podía hacer a través de PHP, como restringir directorios, pero fueron eliminadas en dicha versión.

Se puede restringir la manipulación de archivos con la variable _openbasedir, que establece el directorio nombrado como el directorio root para operaciones con archivos. Si este valor está establecido, archivos fuera de este directorio y subsidiarios serán inaccesibles para PHP:

open_basedir = /home/web

La variable _max_executiontime establece el numero máximo de segundos que PHP esperará para que un script finalice su ejecución, antes de forzar su cierre:

max_execution_time = 30

Configuración de subida de archivos y variables de formularios

Se puede incrementar la seguridad desactivando la subida de archivos con la variable _fileuploads o limitando el tamaño de archivo máximo con _upload_maxfilesize:

file_uploads = On

upload_max_filesize = 2M

En cuanto a los formularios, la variable _post_maxsize controla la cantidad máxima de datos que PHP aceptará en un formulario de vez con el método POST:

post_max_size = 8M

Otra variable es _max_inputtime, que limita en segundos el tiempo de recibir datos a través de POST, GET y PUT:

max_input_time = 90

Ajustes de rendimiento

Hay algunas variables que pueden mejorar el rendimiento del intérprete PHP. La variable _memorylimit especifica la memoria máxima que un simple script puede utilizar:

memory_limit = 8M //(Este valor deberá ser mayor que post_max_size)

Otra opción para mejorar el rendimiento es desactivar las variables $argc y $argv, que guardan el número de argumentos pasados a una aplicación desde la línea de comandos así como sus valores:

register_argc_argv = false

La función ini_set()

PHP lee toda la configuración al iniciarse desde el archivo php.ini, pero también permite sobreescribirla con la función _ini_set()_, que acepta dos argumentos, el nombre de la variable a configurar y su nuevo valor. El siguiente ejemplo incrementa el tiempo de ejecución:

```
<?php


ini_set('max_execution_time', 900);
```

// código

La configuración afecta sólo al script en donde está configurado. Cuando el script termina, el valor de la variable vuelve a su valor inicial.