



PHP. Nivel I

marzo, 2019



Objetivos del nivel

- Entender la dinámica de las aplicaciones Web.
- Conocer el lenguaje PHP.
- Aprender a crear contenido HTML dinámicamente.
- Aprender a procesar los datos enviados en formularios.
- Trabajar con arreglos.

Hacia quien está Orientado el curso

Hacia diseñadores de páginas Web que desean agregarle mayor versatilidad a sus sitios.

Prerrequisitos del nivel

- HTML 5 y CSS 3 Nivel I
- Lógica de Programación Nivel III

Acerca de este manual

Este manual pertenece al Centro de Asesoramiento y Desarrollo Informático C.A. (CADIF1). Para obtener más información sobre este u otros cursos visite nuestra sitio Web www.cadif1.com, escribanos a la dirección de correo cadi@cadif1.com o visítenos en nuestra sede ubicada en la Av. Pedro León Torres con calle 59, Centro Comercial Sotavento, piso 2 oficina 27, Barquisimeto estado Lara, Venezuela. Tlf. 0251-7179247, 0251-4410268.

Las marcas mencionadas en este manual son propiedad de sus respectivos dueños. Copyright 2019. Todos los derechos reservados.



Contenido del nivel

Capítulo 1. Instalación Del Servidor Web

- 1.1.- El Servidor Apache.
- 1.2.- Easy Php.
- 1.3.- Acceso al Servidor web.
- 1.4.- Alias.

Capítulo 2. Introducción de Php

- 2.1.- El Lenguaje Php.
- 2.2.- Insertar Código Php en el Html.
- 2.3.- El Php.ini.

Capítulo 3. Conociendo Php

- 3.1.- Aspectos Básicos Del Lenguaje Php.
- 3.2.- Mostrando Información en la Página.
- 3.3.- Manejo de Cadenas.

Capítulo 4. Entorno de Desarrollo Para Php

- 4.1.- El Ide de Netbeans.
- 4.2.- Crear un Proyecto.
- 4.3.- Crear un Archivo Php Nuevo.
- 4.4.- Crear un Proyecto.

Capítulo 5. Php + Html. Parte 1

- 5.1.- Introducción.
- 5.2.- Uso de Listas.

Capítulo 6. Instrucciones Condicionales

- 6.1.- Operadores de comparación.
- 6.2.- Condiciones.
- 6.3.- Switch.

Capítulo 7. Instrucciones Iterativas

- 7.1.- For.
- 7.2.- While.
- 7.3.- Do While.

Capítulo 8. Trabajando Con Tablas

- 8.1.- Contenido Dinámico de Las Celdas.
- 8.2.- Contenido Condicionado.
- 8.3.- Generación de Filas y Celdas.

Capítulo 9. Arreglos. Parte 1

- 9.1.- Declaración.
- 9.2.- Acceso.
- 9.3.- Sum y Count.

Capítulo 10. Arreglos. Parte 2

- 10.1.- Buscar Elemento.
- 10.2.- Ordenar Elementos.
- 10.3.- Explode e Implode.

Capítulo 11. Arreglos. Parte 3

- 11.1.- Arreglos Asociativos.
- 11.2.- Declaración.
- 11.3.- Foreach.

Capítulo 12. Manipulación de Formularios. Parte 1

- 12.1.- Recepción de Datos.
- 12.2.- Recibiendo Datos Enviados con Post.
- 12.3.- Recibiendo Datos Enviados Con Get.

Capítulo 13. Manipulación de Formularios. Parte 2

- 13.1.- La Función Isset.
- 13.2.- Redireccionamiento de Páginas.

13.3.- Procesar Datos en la Misma Página.

Capítulo 14. Manipulación de Formularios. Parte 3

14.1.- Trabajar Con Checkbox.

14.2.- Enviar Datos Como Arreglos.

Capítulo 15. Manipulación de Formularios. Parte 4

15.1.- Llenar Select Dinámicamente.

15.2.- Enviar Select Múltiples.

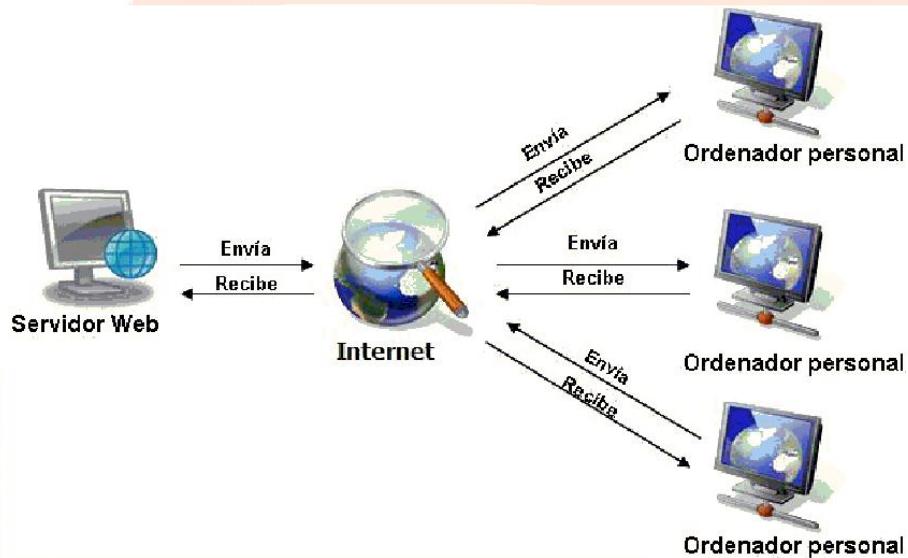


Capítulo 1. INSTALACIÓN DEL SERVIDOR WEB

1.1.- El Servidor Apache

Un servidor web es un programa que implementa el protocolo HTTP. Este se encarga de mantenerse a la espera de peticiones HTTP llevadas a cabo por un cliente HTTP (típicamente, un navegador). El navegador realiza una petición al servidor y éste le responde con el contenido que el cliente solicita.

Cabe destacar el hecho de que la palabra servidor identifica tanto al programa como a la máquina en la que dicho programa se ejecuta. Existe, por tanto, cierta ambigüedad en el término. Entre estos servidores web están: Apache e Internet Information Server (IIS).



El servidor HTTP Apache es un software libre, de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. El servidor Apache se desarrolla dentro del proyecto HTTP Server (`httpd`) de la Apache Software Foundation.

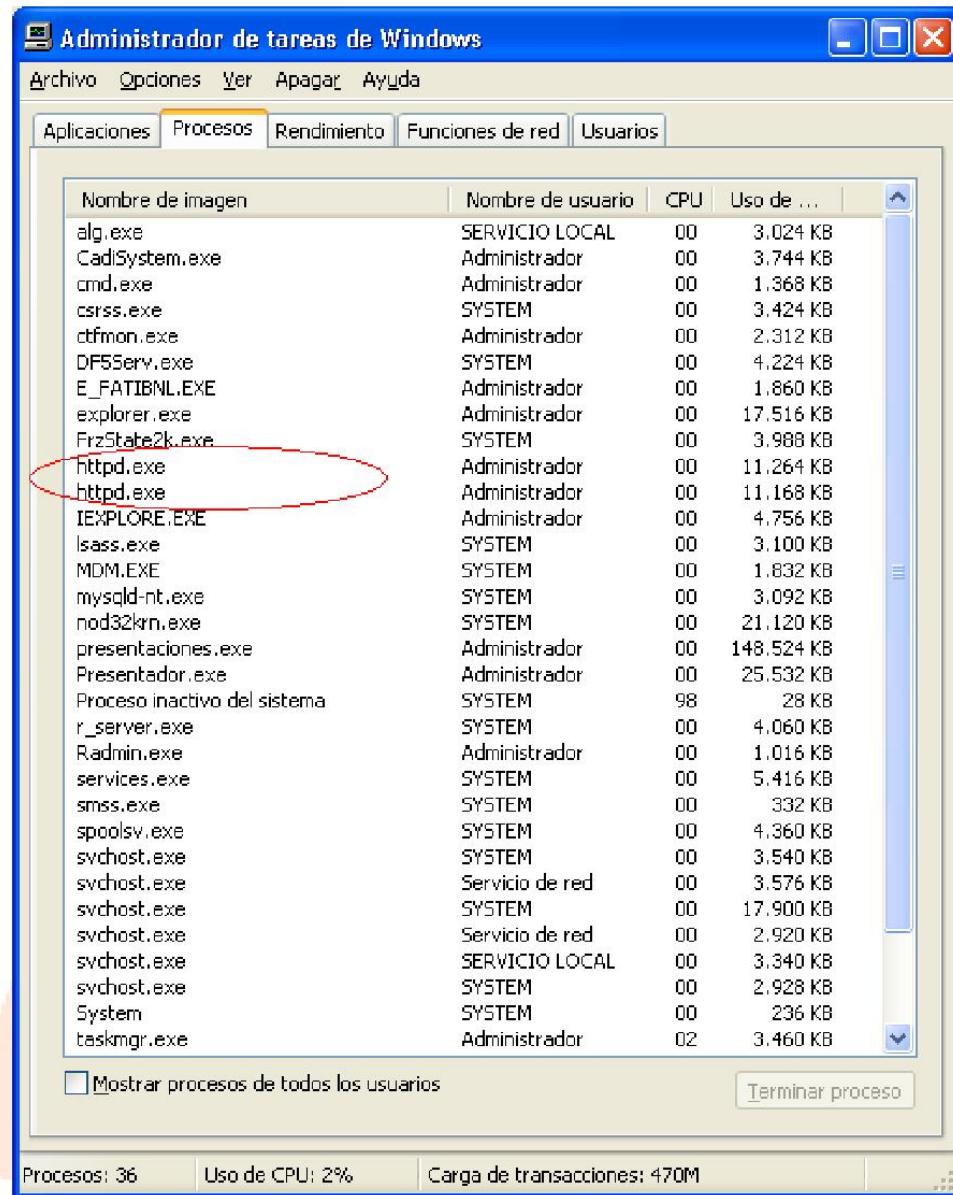
Apache tiene amplia aceptación en la red: desde 1996, es uno de los servidores HTTP más usados. Apache puede instalarse en Windows como un servicio o puede ser

ejecutado como un programa cualquiera a través de los accesos directos que crea el Appserv al instalarse.



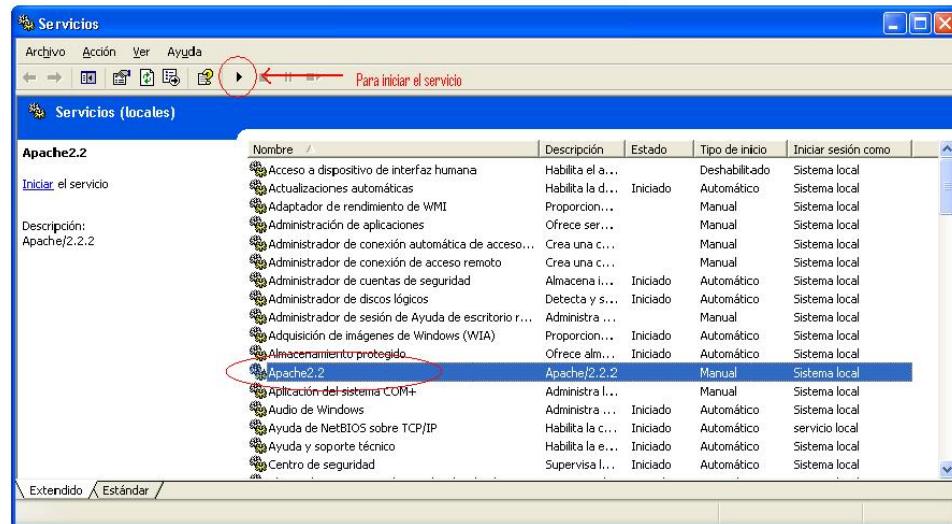
Lo primero que debemos saber es si el servidor está ejecutándose. Una forma sencilla de saberlo es abriendo el administrador de tareas de windows y buscar un servicio llamado httpd.exe.

Si no aparece podemos iniciararlo haciendo click en "Inicio"->"Programas"->"Appserv"-> "Control server by service" -> "Apache start" o tambien "Inicio"->"Programas"->"Appserv"-> "Control server by manual" -> "Apache start".



En la carpeta de instalación de Apache se almacenan varios archivos de configuración, entre estos, el más importante es el archivo httpd.conf, el cual contiene opciones de configuración tales como: puerto, carpeta raíz (document root), entre otras. Todas estas opciones pueden hacer que el comportamiento del servidor cambie. Normalmente, al hacer cambios en este archivo se debe reiniciar el servicio.

Otra forma de iniciararlo es buscando el servicio "Apache 2.2" en el administrador de servicios de Windows, que podemos encontrarlo en el panel de control -> Herramientas Administrativas -> Servicios, seleccionarlo y hacer click en el botón "Iniciar servicio".



1.2.- Easy Php

Para trabajar con PHP, se necesita el servidor web y el interpretador de PHP (que se instala aparte), además hace falta configurar el servidor WEB para que pueda hacer uso del interpretador de PHP. Realizar esta tarea manualmente puede ser muy engorroso. Es por eso que se han creado paquetes que instalan y configuran automáticamente estas dos herramientas, la mayoría usando como servidor web Apache.

Adicionalmente a Apache y PHP, estos paquetes instalan MySQL Server y PHPMyAdmin (por lo que se les conoce como paquetes AMP). Existen en el mercado varias opciones: Appserv, Xampp, Wamp, Easy PHP, entre otros. Se puede usar cualquiera de estos, dado que lo que varía entre cada uno de ellos es el proceso de instalación y las herramientas de administración que poseen.

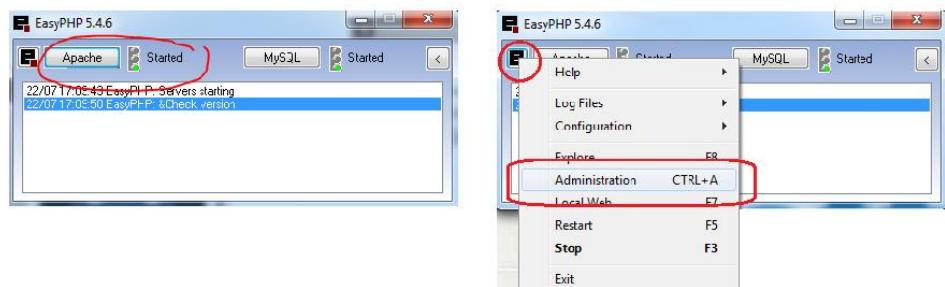


EasyPHP es un paquete open source de descarga gratuita para Windows, muy fácil de instalar y de poner en marcha. A diferencia de otros paquetes, Easyphp se ejecuta como un programa de escritorio y es en ese momento cuando inician los servicios Apache y Mysql, evitando así cargar al equipo ejecutando software que no es necesario en un momento dado. La instalación es tan sencilla como cualquier aplicación para Windows.

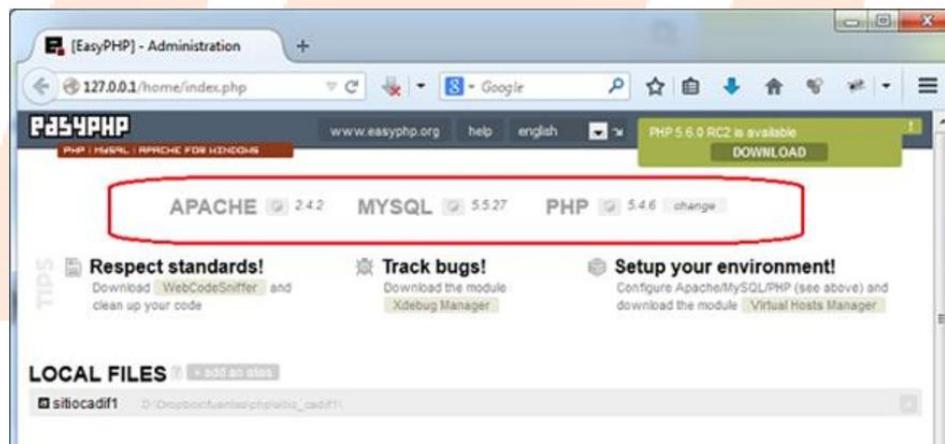


Luego de instalado, se debe ejecutar. Al ejecutar aparecerá una ventana que indicará si los servidores Apache y Mysql se iniciaron. Si se inician exitosamente, se mostrará

el semáforo con la luz verde, de lo contrario, se mostrará el semáforo con una luz roja. Hay que tener en cuenta que en ocasiones ya se tiene instalado otra versión de Apache o de otro software que ocupa el puerto 80 que es el puerto por defecto de Apache. En ese caso el servidor no iniciará. Hay que detener el servicio ya existente o cambiar la configuración del puerto de Apache. Para acceder a la administración de EasypHP se debe hacer click en el ícono con la letra "e". Aparecerá un menú donde se selecciona la opción "Administration".



Se abrirá en el navegador por defecto con una página donde se podrá visualizar las versiones de cada paquete que trae EasypHP.



1.3.- Acceso al Servidor web

Todos los computadoras que están conectados a internet (y en general a cualquier red) deben tener una dirección que los identifique como únicos. A esta dirección se le

denomina "dirección IP" y está formada por 4 números con valores entre 0 y 255, por ejemplo: 200.44.12.32. Para acceder a un servidor Web necesitamos saber su dirección IP o el nombre de dominio asociado a él.

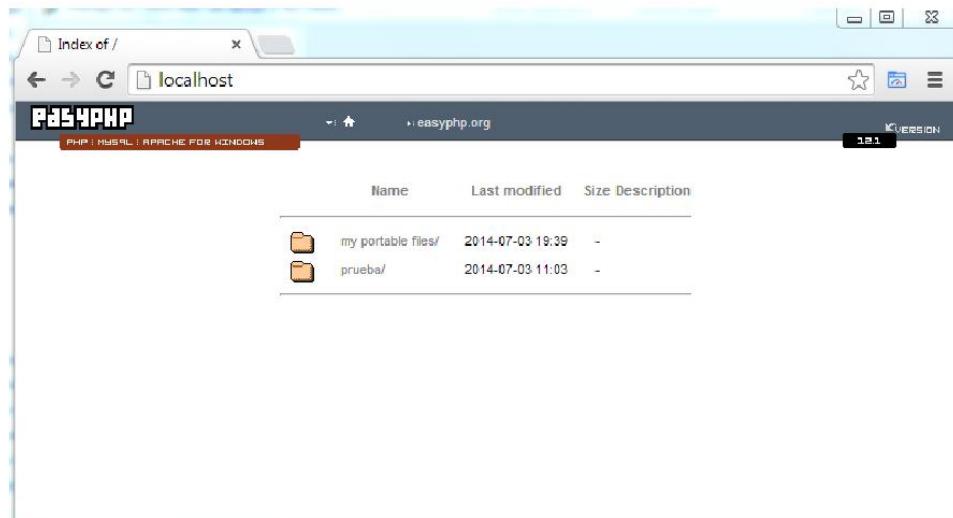
El problema de usar la dirección IP es lo difícil que puede ser para el usuario recordarla, además de que existe la posibilidad de que ésta cambie. Para eso existe un servicio llamado DNS, que nos permite acceder a los servidores usando un nombre y él se encarga de hacer la traducción del "nombre de dominio" a su correspondiente dirección IP.

Por ejemplo, la dirección IP del servidor www.cadif1.com es 72.232.237.163. Imagínese que para visitar esta página haya que recordar todos esos números, sin mencionar el problema que tendríamos si la dirección IP del servidor cambiara.

Para acceder al servidor web local no se necesita un dominio ni un servidor DNS, ni conexión a internet, ni siquiera se necesita saber la dirección IP del equipo, lo único que se necesita es el navegador. Esto debido a que se puede referir a la misma computadora con el nombre "localhost", que tiene asociada la dirección IP 127.0.0.1.

Adicionalmente, se debe entender que el objetivo de un servidor Web es publicar información, pero ¿ cual información ?, ¿ donde debe estar ubicada la información para que éste pueda publicarla ?, o ¿ se va a publicar todo el disco duro ?. El servidor web posee una carpeta (llamada carpeta raíz del servidor) donde se deben colocar todos los sitios (uno en cada carpeta), páginas y/o archivos que deseamos publicar. En el caso de Apache con EasypHP la carpeta raíz esta en c:/archivos de programas/easy php/www (esta ruta puede cambiar ligeramente de un sistema operativo a otro y de una versión a otra de easy php).

Para empezar a hacer uso del servidor web, se abre el navegador y se coloca <http://localhost>. Cuando se instala Apache con EasypHP aparecerá una página como la que se muestra en la siguiente imagen, que muestra el contenido de la carpeta raíz de Apache:

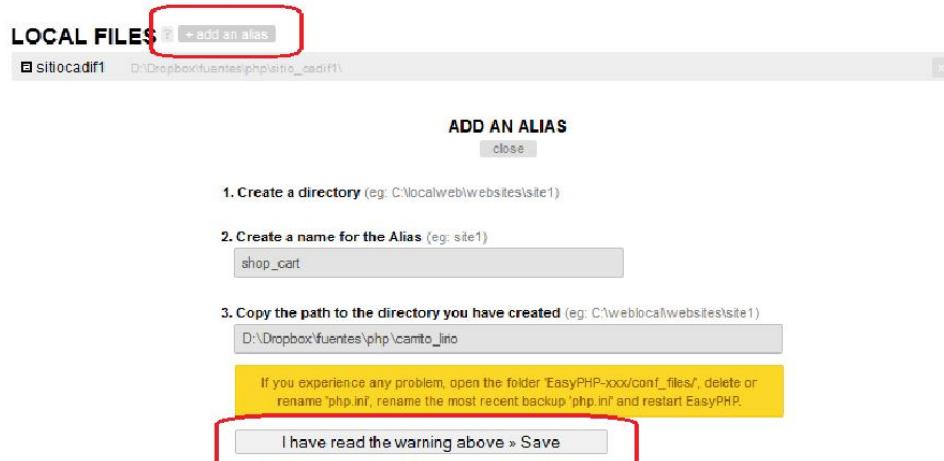


1.4.- Alias

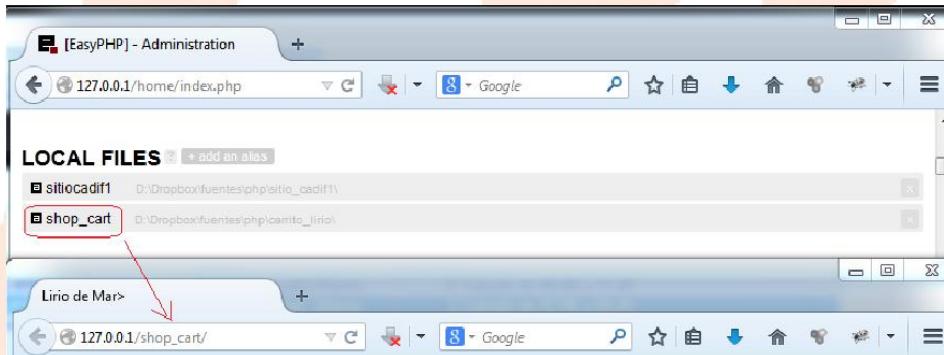
Apache posee una herramienta llamada "alias". Un alias es la ruta de una carpeta donde esta alojado un sitio web (conjunto de archivos Php y html), bajo un sobre nombre que el administrador especifica.

Un alias evita tener que guardar todos nuestros proyectos en la carpeta raíz por defecto de apache, que generalmente esta en la raíz del disco "C". Esto es especialmente útil se tiene un proyecto en una memoria usb o en otra partición.

Para agregar un alias se debe abrir la página de administración de EasyPhp, hacer click en el botón "Add an Alias". Aparece las opciones "Name for the alias" donde se coloca el nombre del alias (se recomienda un nombre sin espacios en blanco) y "the path to the directory" que es la ruta donde se encuentran los archivos php:



Luego de crear el alias se puede acceder a los archivos de esa carpeta colocando en el navegador: <http://localhost/alias>. En el ejemplo de la imagen se utiliza la dirección 127.0.0.1 que es equivalente a localhost.

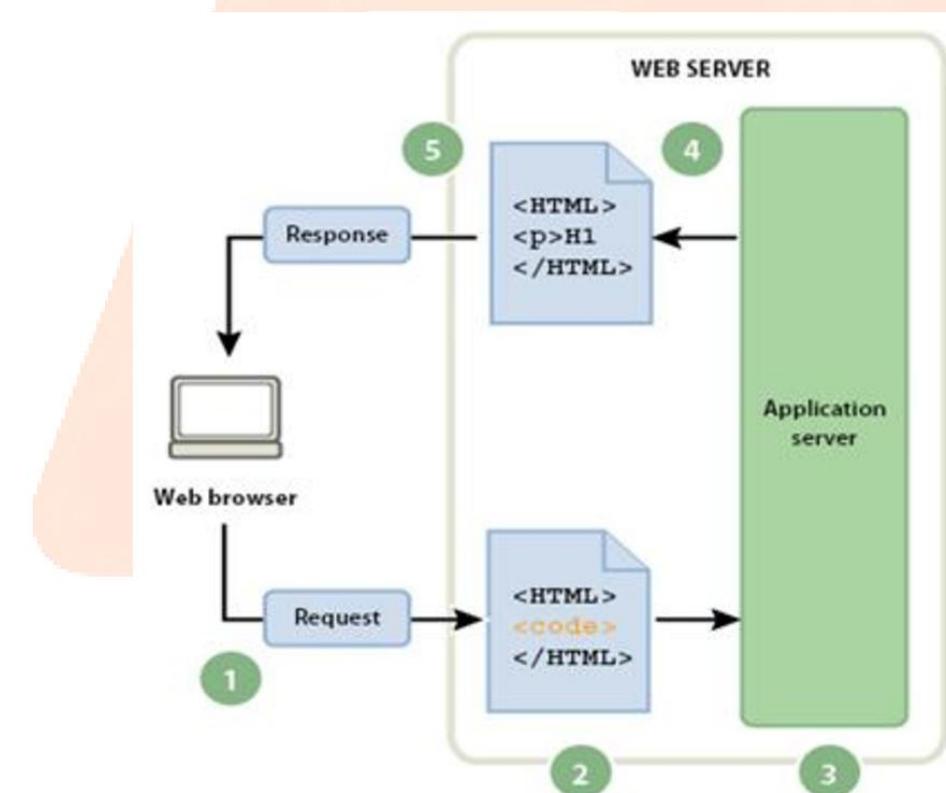


Capítulo 2. INTRODUCCIÓN DE PHP

2.1.- El Lenguaje Php

Un lenguaje del lado del servidor es aquel lenguaje de programación cuyo código es reconocido, ejecutado e interpretado en servidor web (con un intérprete agregado a éste) y que cuyo resultado se envía al cliente (navegador) en un formato comprensible para éste (HTML, XML o JSON).

Los scripts (segmentos de código) son almacenados en el servidor (ya sea incrustados en el código HTML o en archivos aparte) quien los ejecuta apoyándose en un intérprete para generar una salida, que posteriormente es enviada al cliente (FrontEnd).



PHP es un lenguaje creado por una gran comunidad de personas. El sistema fue desarrollado originalmente en el año 1994 por Rasmus Lerdorf como un CGI escrito en C que permitía la interpretación de un número limitado de comandos. El sistema fue denominado Personal Home Page Tools, pero luego su nombre fue cambiado a "PHP Hypertext Pre-processor".

El lenguaje PHP es un lenguaje de programación de estilo clásico, es decir, es un lenguaje de programación con variables, sentencias condicionales, ciclos (bucles), funciones. No es un lenguaje de marcado como podría ser HTML, XML o WML.

A diferencia de JavaScript que se ejecuta en el navegador, PHP se ejecuta en el servidor, por eso permite acceder a los recursos que tenga el servidor, como por ejemplo podría ser una base de datos, carpetas, servidor de correo, entre otros.

Al ser PHP un lenguaje que se ejecuta en el servidor no es necesario que el navegador requiera un software especial para acceder a archivos con código Php. Php tiene algunas ventajas, entre ellas estan:

- * Es un lenguaje multiplataforma.
- * Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL
- * Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados exts o extensiones).
- * Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- * Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- * Permite usar el paradigma de Programación Orientada a Objetos.

2.2.- Insertar Código Php en el Html

El código PHP se inserta (se empotra) en cualquier lugar del código HTML, siempre y cuando este encerradas en las etiquetas correctas. Se puede hacer de varias maneras:

- <? código php ?>
- <?php código php ?>
- <script language="php"> código php </script>

Por ejemplo:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title></title>
  </head>
  <body>
    <?php
      // coloque su código php aquí
    ?>
  </body>
</html>
```

Existe una función que permite conocer información del servidor donde se ejecutan los scripts y verificar la configuración actual de Php. La manera de hacerlo es insertar en una página html una etiqueta Php para verificar, de la siguiente manera:

```
<?php
  phpinfo();
?>
```

Si todo está bien con esta etiqueta se generará una página completa de información sobre el servidor Web, sobre la versión de php y sobre la configuración que este tiene en el momento.

PHP Version 5.1.4



System	Windows NT CADIASESOR 5.1 build 2600
Build Date	May 4 2006 10:30:29
Configure Command	ccscript /nologo configure.js "--enable-snapshot-build" "--with-gd=shared"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\WINDOWS\php.ini
PHP API	20041225
PHP Extension	20050922
Zend Extension	220051025
Debug Build	no
Thread Safety	enabled
Zend Memory Manager	enabled
IPv6 Support	enabled
Registered PHP Streams	php, file, http, ftp, compress.zlib
Registered Stream Socket Transports	tcp, udp
Registered Stream Filters	converticonv.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert*, consumed, zlib.*

This program makes use of the Zend Scripting Language Engine:
| Zend Engine v2.1.0 Copyright (c) 1998-2006 Zend Technologies |

Powered By 

2.3.- El Php.ini

PHP utiliza un archivo para definir su comportamiento en ciertos aspectos. Esta información se almacena en un archivo con el nombre PHP.ini (ubicado generalmente en la carpeta e Windows c:winnt), el cual contiene una gran cantidad de directivas que pueden ser modificadas.

Lo único que hay que hacer es ubicar el archivo, abrirlo con cualquier editor de texto plano, ubicar la directiva deseada y especificarle el valor deseado. Para conocer el significado de muchas de las directivas consulte el manual oficial de Php, aunque el archivo contiene comentarios que ayudan a entenderlo y modificarlo. Un cambio en el php.ini puede implicar reiniciar el servidor web para que tenga efecto.

Capítulo 3. CONOCIENDO PHP

3.1.- Aspectos Básicos Del Lenguaje Php

PHP está basado en lenguaje C, por eso heredó mucha de las características de este lenguaje. Entre esas características están:

- * Sensibilidad a mayúsculas y minúsculas (Contador <> contador).
- * El separador de instrucciones es el ";".
- * Los comentarios se hacen con #, // o con /* */.

Por ejemplo:

```
<?php
    // Comentario tipo C++ para una linea
    # esto tambien es un comentario de linea

    /*
        Esto es un comentario multilinea
        otra linea mas de comentario
    */
?>
```

El tipo de una variable normalmente no lo indica el programador; en su lugar, lo decide PHP en tiempo de ejecución dependiendo del contexto en el que se utilice esa variable. PHP soporta los siguientes tipos de datos:

- matrices.
- números en punto flotante.
- enteros.
- objetos.
- cadenas.

En PHP las variables se identifican con un signo \$ seguido por el nombre de la variable. Las variables se declaran automáticamente al asignársele un valor. Las variables se pueden declarar en cualquiera de los bloques de código de PHP y se pueden usar en cualquier otro bloque de la misma página siempre y cuando esté después del bloque donde se declaró. Las variables son locales al documento donde se declaran.

Existen unas variables globales predefinidas por PHP, llamadas también variables del sistema. Entre estas variables están: \$_POST, \$_GET, \$SERVER, \$GLOBALS, \$_SESSION, entre otras. Ejemplos de declaración de variables:

```
<body>
<?php
$Nombre = "Santana";
$nombre = "Pedro";
$edad = 10;
$sueldo = 1500.5;

// la siguiente linea produciría un error
$ aumento=$sueldominimo*2;
?>
</body>
```

3.2.- Mostrando Información en la Página

La manera tradicional de mostrar información es usando la palabra reservada "echo". Se pueden mostrar cadenas o variables, o combinaciones de ambas. También existe el procedimiento "printf" heredado de lenguaje C. El printf tiene el mismo comportamiento que en lenguaje C. Su sintaxis es:

```
<?php  
    // Ejemplos del uso del printf son:  
    $edad=50;  
    printf("La edad de la persona es %d", $edad);  
  
    // Ejemplos del uso del echo son  
  
    echo 'Este contenido se genero con php';  
    echo "El nombre del cliente es jose luis";  
?>
```

En este punto hay que hacer una distinción, la interpretación que hace PHP de las comillas simples y las comillas dobles. En el segundo caso PHP interpretará el contenido de la cadena. El siguiente es un ejemplo:

```
<?php  
  
    $a = "Mundo";  
    echo 'Hola $a'; //Esto escribirá "Hola $a"  
    echo "Hola $a"; //Esto escribirá "Hola Mundo"  
  
?>
```

Existe otra forma de mostrar información. Es una función llamada "print_r" que imprime información legible para humanos sobre una variable. Si se le da string, integer o float, el valor en sí mismo será impreso. Si le dan un array, los valores serán presentados en un formato que muestra las claves y los elementos. Una notación similar se utiliza para objects.

```
<?php  
$a = array ('a' => 'manzana', 'b' => 'banana', 'c' => array ('x', 'y', 'z'));  
print_r ($a);  
?>
```

El resultado del ejemplo sería:

```
Array  
(  
    [a] => manzana  
    [b] => banana  
    [c] => Array  
        (  
            [0] => x  
            [1] => y  
            [2] => z  
        )  
)
```

Var_dump muestra información sobre una variable. Esta función muestra información estructurada sobre una o más expresiones incluyendo su tipo y valor. Las matrices o arreglos y los objetos son explorados recursivamente con valores sangrados para mostrar su estructura.

```
<?php  
  
$b = 3.1;  
$c = true;  
var_dump($b, $c);  
  
?>
```

El resultado del ejemplo sería:

```
float(3.1)  
bool(true)
```

3.3.- Manejo de Cadenas

El único operador de cadenas que existen es el de concatenación, el punto (.), sin embargo, PHP dispone de toda una gama de funciones que permitirán manipular cómodamente las cadenas de texto. La forma de concatenar en Php es la siguiente:

```
<?php  
  
$a = "Hola";  
$b = $a . " Mundo"; // Ahora $b contiene "Hola Mundo"  
  
echo $b . " jose "; // esto imprime "Hola Mundo jose"  
  
?>
```

Capítulo 4. ENTORNO DE DESARROLLO PARA PHP

4.1.- El Ide de Netbeans

Netbeans es un IDE de software libre y multiplataforma diseñado originalmente para desarrollar aplicaciones para Java. Fue creado de una forma modular de manera tal que se le podían agregar módulos para desarrollar en lenguajes distintos a Java.

En la versión 6.5 se le añadió el módulo para Php el cual va a permitir desarrollar aplicaciones web con funciones como:

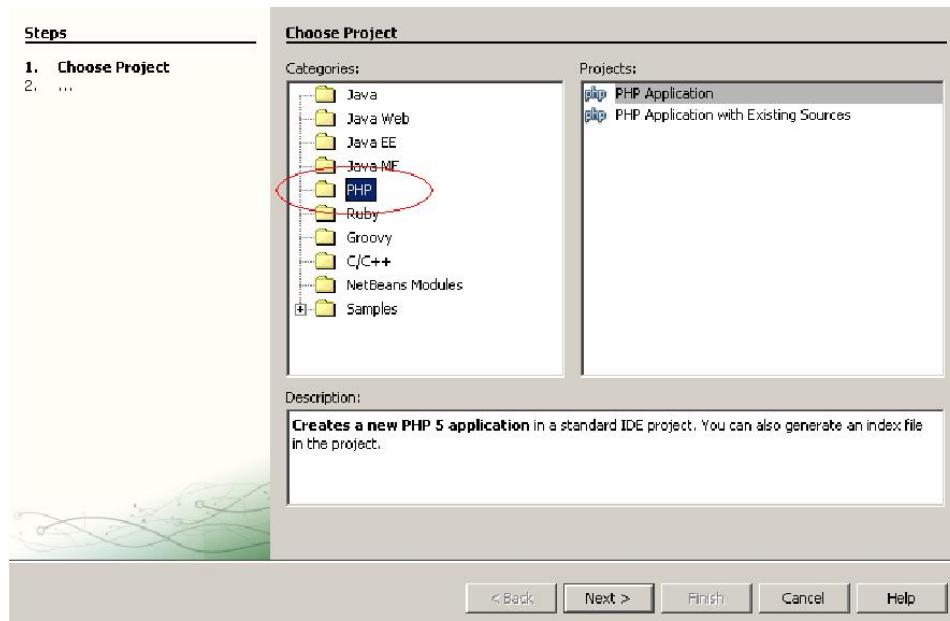
- auto completación de código.
- depurador integrado.
- verificación automática de errores.
- entre otras cosas.

Para instalar Netbeans se debe tener instalado el JDK o el JRE.

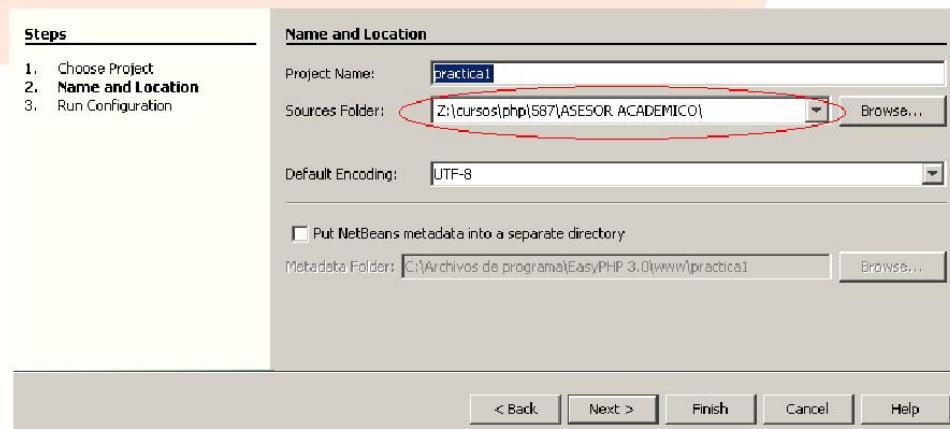


4.2.- Crear un Proyecto

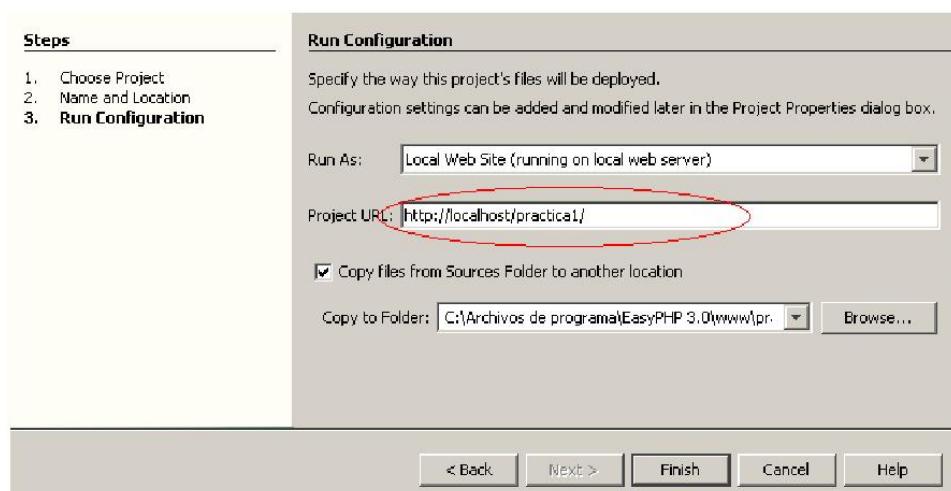
Para crear un proyecto nuevo para Php se debe hacer click en File -> new -> Project. Luego se inicia un asistente donde se debe seleccionar "PHP" en categorías, tal como se muestra en la imagen:



Luego, se debe seleccionar el directorio donde se van a guardar los fuentes del proyecto:

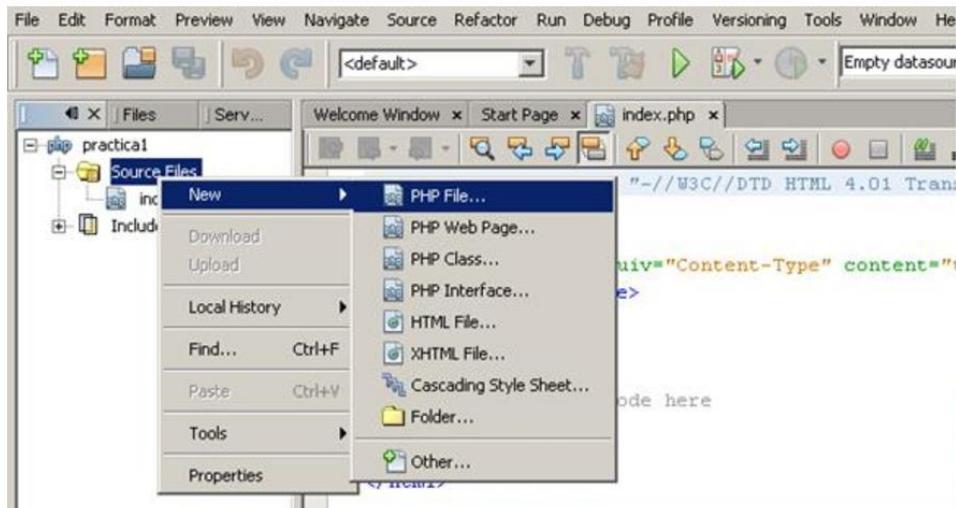


Por último se indica la ubicación de la carpeta raíz del servidor web que se está usando. Aquí se debe tener cuidado en indicar la carpeta raíz del servidor web que se esté usando. Netbeans automáticamente va a copiar los archivos en esa carpeta para que puedan ser encontrados por el servidor.

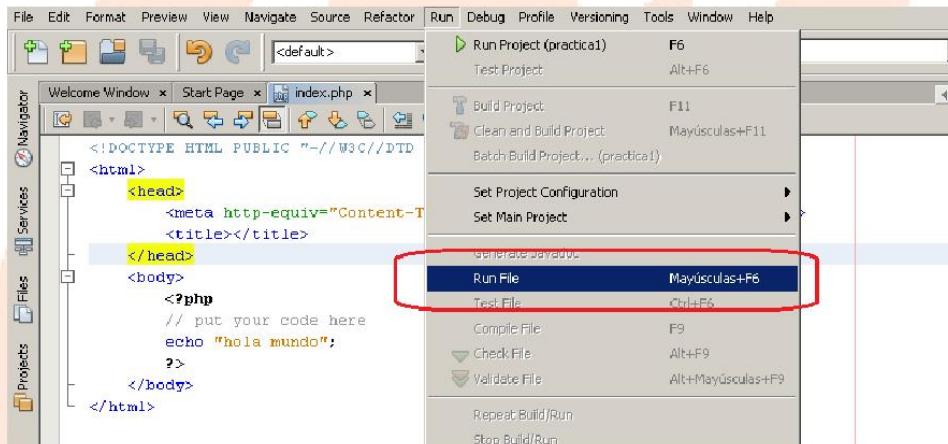


4.3.- Crear un Archivo Php Nuevo

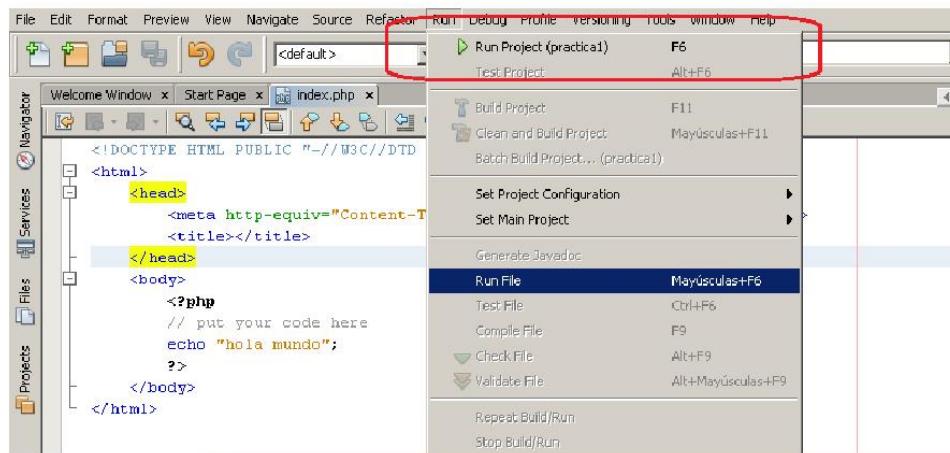
Para crear un archivo nuevo, se debe hacer click con el botón derecho sobre el proyecto y seleccionar "New" -> "File", como se muestra en la imagen:



Para ejecutar el archivo actual, se hace click en la opción del menú principal "Run" y luego en "Run File" o se presiona la combinación de teclas Shift+F6.

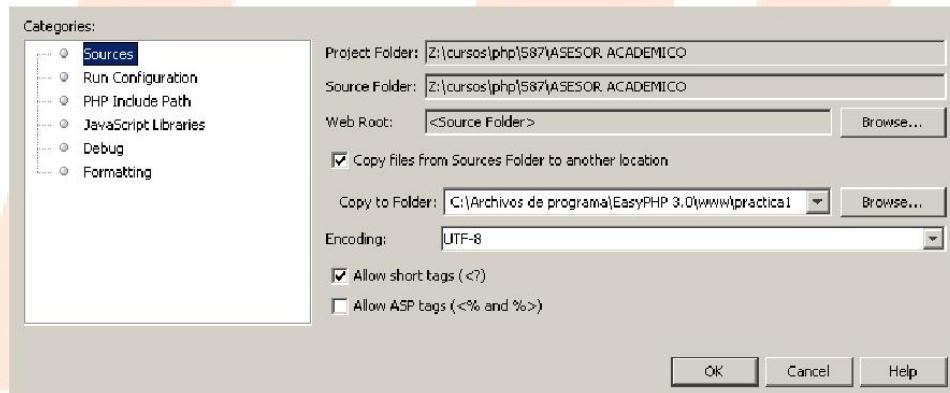


Otra forma de ejecutar es seleccionando la opción "Run Project" o la tecla F6. Esta opción ejecuta la página principal del proyecto actual:

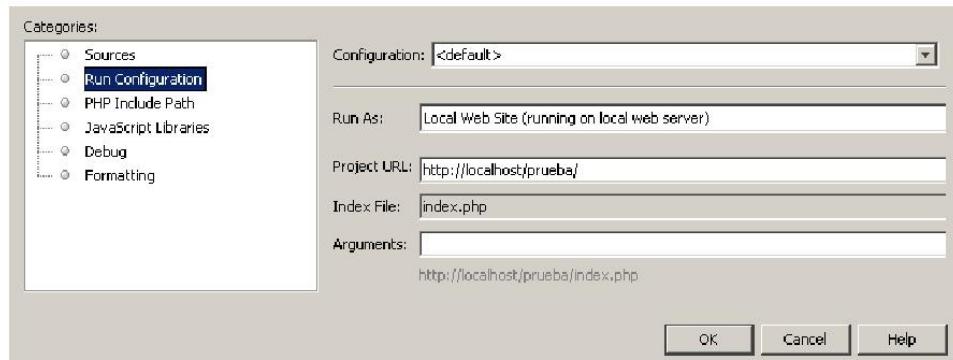


4.4.- Crear un Proyecto

Las propiedades del proyecto se consiguen al hacer click con el botón derecho sobre el proyecto y luego en la opción "Properties", donde se puede cambiar la carpeta raíz del servidor web:



También se puede cambiar la configuración de ejecución, especificando cual es la dirección URL local y cuál será la página de inicio del proyecto, que generalmente se llama index.php. La página de inicio es la que se ejecuta cuando se selecciona la opción "Run Project":



Capítulo 5. PHP + HTML. PARTE 1

5.1.- Introducción

El código Php se inserta (o empotra) dentro del código HTML. Cuando el usuario desde un navegador solicita una página con extensión Php el servidor ejecuta el código que está en la página y envía el código resultante, que luego será interpretado y mostrado por el navegador. En base a esto, se puede combinar el código Php con el HTML de manera tal que sea más sencilla la generación del HTML resultante. Por ejemplo, conociendo la etiqueta HTML de un hipervínculo se puede generar dinámicamente de la siguiente forma:

```
<body>
<?php
$texto="Pagina google";
$página="http://www.google.com";

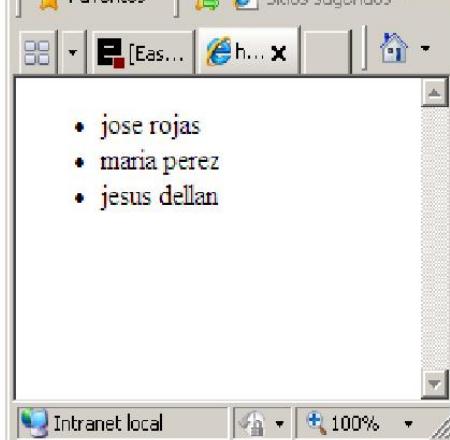
echo "<a href=$página> $texto </a>";
?>
</body>
```

También se puede generar así:

```
<body>
<?php
$texto="Pagina google";
$página="http://www.google.com";
?>
<a href=<?php echo $página?>> <?php echo $texto?> </a>
</body>
```

5.2.- Uso de Listas

Si es necesario llenar una lista HTML (ol o ul) con varios nombres, combinando Php con HTML quedaría algo como lo siguiente:



```
<body>
<?php
$nombre1="jose rojas";
$nombre2="maria perez";
$nombre3="jesus dellan";
?>
<ul>
<li><?php echo $nombre1?></li>
<li><?php echo $nombre2?></li>
<li><?php echo $nombre3?></li>
</ul>
</body>
```

Otra forma de obtener el mismo resultado es el siguiente:

```
<body>
<?php
$nombre1="jose rojas";
$nombre2="maria perez";
$nombre3="jesus dellan";

echo "  <ul>
        <li>$nombre1</li>
        <li>$nombre2</li>
        <li>$nombre3</li>
    </ul>" ;
?>
</body>
```

Capítulo 6. INSTRUCCIONES CONDICIONALES

6.1.- Operadores de Comparación

Los operadores de comparación o relacionales, como su nombre indica, permiten comparar dos valores. Los operadores relacionales en PHP se resumen en la siguiente tabla:

Ejemplo	Nombre	Resultado
<code>\$a == \$b</code>	Iqual	TRUE si \$a es igual a \$b.
<code>\$a === \$b</code>	Idéntico	TRUE si \$a es igual a \$b, y son del mismo tipo.
<code>\$a != \$b</code>	Diferente	TRUE si \$a no es igual a \$b.
<code>\$a <> \$b</code>	Diferente	TRUE si \$a no es igual a \$b.
<code>\$a !== \$b</code>	No idénticos	TRUE si \$a no es igual a \$b, o si no son del mismo tipo.
<code>\$a < \$b</code>	Menor que	TRUE si \$a es estrictamente menor que \$b.
<code>\$a > \$b</code>	Mayor que	TRUE si \$a es estrictamente mayor que \$b.
<code>\$a <= \$b</code>	Menor o igual que	TRUE si \$a es menor o igual que \$b.
<code>\$a >= \$b</code>	Mayor o igual que	TRUE si \$a es mayor o igual que \$b.

De estos operadores, los más particulares son `==` (idéntico) y `!=` (no idéntico). Estos operadores existen porque PHP al ser tan flexible con los tipos de datos tiene comportamientos singulares al momento de comparar 2 valores de distintos tipos. Por ejemplo:

```
<?php
```

```
$x = 5;
$y = "5";

// es el valor de $x igual al de $y
var_dump($x == $y);
// es el valor y el tipo de datos de $x
// igual al de $y
var_dump($x === $y);
```

```
127.0.0.1/ed
127.0.0.1 160% ...
Most Visited Getting Started
bool(true) bool(false)
```

El mismo ejemplo pero usando los operadores diferente (!=) y no idéntico (!==), donde se nota que los resultados son contrarios:

```
<?php
```

```
$x = 5;
$y = "5";

// es el valor de $x igual al de $y
var_dump($x != $y);
// es el valor y el tipo de datos de $x
// igual al de $y
var_dump($x !== $y);
```

6.2.- Condiciones

La sentencia if permite ejecutar un bloque de instrucciones si la condición es verdadera. Es importante tener en cuenta que la condición que evaluada ha de estar encerrada entre paréntesis. La forma general de hacer una condición es la siguiente:

```
if (condicion) instrucion1;

o

if (condicion) {
    instrucion1;
    instrucion2;
    instrucionN;
}
```

Si no se utilizan llaves es porque una sola instrucción estará condicionada.

Un ejemplo del uso del if es el siguiente:

```
<?php
$a = rand(1,100);
?>
<body style="<?php if ($a>10) echo 'color:red' ?>">
    <h1>Capitulo 5</h1>
    <div>
        <?php
            echo "contenido del div"
        ?>
    </div>
```

Otro ejemplo donde se necesitan llaves es el siguiente:

```
<body style="<?php if ($a>10) echo 'color:red' ?>">
    <h1>Capitulo 5</h1>
    <?php
        if ($a < 50)
        {
        ?>
        <div>
            <?php
                echo "contenido del div"
            ?>
        </div>
        <?php
        }
        ?>
```

La sentencia if...else permite ejecutar un bloque de instrucciones si la condición es verdadera y otro bloque de instrucciones si ésta es falsa. La forma general de usar la instrucción if..else es la siguiente:

```
if (condición) {
    Este bloque se ejecuta si la condición es VERDADERA
} else
    instrucionElse;
```

o

```
if (condición) {
    Este bloque se ejecuta si la condición es VERDADERA
} else {
    instrucionElse1;
    instrucionElse2;
}
```

El siguiente ejemplo muestra como usar el "else":

```
<?php
$a = rand(1,100);
?>
<body style="color:<?php
    if ($a>10)
        echo 'red';
    else
        echo 'green'
?>">
```

6.3.- Switch

Una alternativa a if...elseif...else, es la sentencia switch, la cual evalúa y compara cada expresión de la sentencia case con la expresión que se evalúa, si llega al final de la lista de case y no encuentra una condición verdadera, ejecuta el código que haya en el bloque "default". Adicionalmente, se debe agregar un break para que la sentencia switch no siga buscando en la lista de case al encontrar una condición verdadera.

La forma general para usar el switch es la siguiente:

```
switch (variable)
{
    case valor1: sentencia1;
                    sentencia2;
                    break;
    case valor2: sentencia3;
                    sentencia4;
                    break;
    default:    sentencia5;
}
```

Por ejemplo:

```
<?php
$dia = rand(1,7);
switch ($dia){
    case 1:
        echo "Lunes";
        break;
    case 2:
        echo "Martes";
        break;
    case 3:
        echo "Miercoles";
        break;
    case 4:
        echo "Jueves";
        break;
    case 5:
        echo "Viernes";
        break;
    default :
        echo "Fin de semana";
}
?>
```

Capítulo 7. INSTRUCCIONES ITERATIVAS

7.1.- For

En PHP existen tres tipos de ciclos repetitivos, estos son: while, do..while y for. El ciclo for resulta muy útil cuando se debe ejecutar un bloque de código cuando una variable se encuentre entre un valor mínimo y otro máximo. El formato general del for es el siguiente:

```
for ( inicialización ; verificación o evaluación de una condición; actualización)
    instrucción1;
```

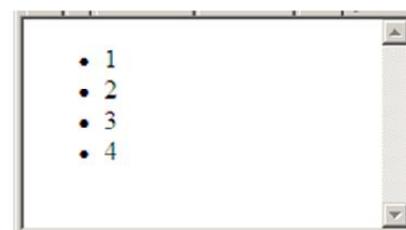
ó

```
for ( inicialización ; verificación o evaluación de una condición; actualización) {
    instrucción1;
    instrucción2;
    instrucciónN;
}
```

El uso de las llaves depende de la cantidad de instrucciones que se necesitan repetir.

Por ejemplo, si se necesita generar una lista con números, se puede hacer usando el siguiente ciclo for:

```
<body>
<ul>
<?php
    for ($i=1;$i<=5;$i++)
        echo "<li>$i</li>";
?
</ul>
</body>
```



Las 3 partes del ciclo for (inicialización, verificación y actualización) pueden ser muy variadas. En el siguiente ejemplo se muestra como hacer un conteo regresivo usando listas no ordenadas:

```
<?php  
    $n = rand(10,20);  
    echo "<ul>";  
    for ($i=$n;$i>0;$i--) {  
        echo "<li>";  
        echo $i;  
        echo "</li>";  
    }  
    echo "</ul>";
```

7.2.- While

La sentencia while ejecuta un bloque de código mientras se cumpla una determinada condición. La forma general de usar el ciclo while es la siguiente:

```
while (condicion)  
    instrucion1;
```

ó

```
while (condicion) {  
    instrucion1;  
    instrucion2;  
    instrucionN  
}
```

Si la condición no se cumple la primera vez, el ciclo no ejecuta ni siquiera una iteración.

En el siguiente ejemplo, se genera un número aleatorio y se mostrara en una lista ordenada hasta que el número generado sea el 5, en cuyo caso se cierra la lista:

```
<?php  
    $n = rand(0,10);  
    echo "<ul>";  
    while ($n != 5) {  
        echo "<li>";  
        echo $n;  
        echo "</li>";  
        $n = rand(0,10);  
    }  
    echo "</ul>";
```

7.3.- Do While

La sentencia do..while es similar a while, salvo que con esta sentencia primero se ejecuta el bloque de código y después se evalúa la condición, por lo que el bloque de código se ejecuta siempre al menos una vez. La forma general del ciclo do while es la siguiente:

```
do{  
    instrucion1;  
}while ( condicion );
```

ó

```
do{  
    instrucion1;  
    instrucion2;  
    instrucionN  
}while ( condicion );
```

En el ciclo do while siempre se utilizan llaves, independientemente de que se repita una instrucción o varias.

El mismo ejemplo usado con while se puede realizar con do while:

```
<?php
    echo "<ul>";
    do
    {
        $n = rand(0,10);
        echo "<li>";
        echo $n;
        echo "</li>";
    }while ($n != 5);
    echo "</ul>";
```

Capítulo 8. TRABAJANDO CON TABLAS

8.1.- Contenido Dinámico de Las Celdas

En el caso de una tabla también se puede hacer que su contenido sea dinámico, por ejemplo:

```
<body>
    <?php
        $descrip1="Cuentas por pagar";
        $descrip2="Ventas";
        $monto1=500000;
        $monto2=15400;
    ?>
    <table border="1" width="250">
        <tr>
            <td>Descripcion</td>
            <td>Monto</td>
        </tr>
        <tr>
            <td><?php echo $descrip1; ?></td>
            <td><?php echo $monto1;?></td>
        </tr>
        <tr>
            <td><?php echo $descrip2;?></td>
            <td><?php echo $monto2;?></td>
        </tr>
    </table>
</body>
```

Descripcion	Monto
Cuentas por pagar	500000
Ventas	15400

Otra forma de colocar el mismo código y obtener el mismo HTML es la siguiente:

```
<table border="1" width="250">
    <tr>
        <td>Descripcion</td>
        <td>Monto</td>
    </tr>
    <?php
        echo "<tr>
                    <td>$descrip1</td>
                    <td>$monto1</td>
                </tr>
                <tr>
                    <td>$descrip2</td>
                    <td>$monto2</td>
                </tr>"?
    </table>
```

8.2.- Contenido Condicionado

A continuación un ejemplo de cómo hacer condiciones para generar código HTML:

```
?>
<table border="1" width="350">
  <tr>
    <td>Descripcion</td>
    <td>Monto</td>
    <td>Estatus</td>
  </tr>
  <tr>
    <td><?php echo $descrip1; ?></td>
    <td><?php echo $monto1;?></td>
    <td><?php if ($monto1>$meta)echo "Sobre la meta";
           else echo "Debe aumentar"; ?></td>
  </tr>
  <tr>
    <td><?php echo $descrip2;?></td>
    <td><?php echo $monto2;?></td>
    <td><?php if ($monto2>$meta)echo "Sobre la meta";
           else echo "Debe aumentar"; ?></td>
  </tr>
</table>
```

Descripcion	Monto	Estatus
Cuentas por pagar	500000	Sobre la meta
Ventas	15400	Debe aumentar



Otra forma de obtener el mismo código HTML resultante es la siguiente:

```
<table border="1" width="350">
  <tr>
    <td>Descripcion</td>
    <td>Monto</td>
    <td>Status</td>
  </tr>
  <?php
  echo "<tr>
            <td>$descrip1</td>
            <td>$monto1</td>
            <td>";
  if ($monto1>$meta) echo "Sobre la meta";
  else echo "Debe aumentar";
  echo " </td>
        </tr>
        <tr>
            <td>$descrip2</td>
            <td>$monto2</td>
            <td>";
  if ($monto2>$meta) echo "Sobre la meta";
  else echo "Debe aumentar";
  echo " </td>
        </tr>"?
  </table>
```

8.3.- Generación de Filas y Celdas

Con PHP se puede generar cualquier etiqueta HTML o contenido, por lo tanto, las filas y las celdas de una tabla se pueden generar en tiempo de ejecución. Un ejemplo para generar varias filas de una tabla usando un ciclo while es el siguiente:

```
<table border="1" width="250">
    <tr>
        <td>Número</td>
        <td>Tipo</td>
    </tr>
    <?php
    $i=1;
    while ($i<=5)
    {
    ?>
    <tr>
        <td><?php echo $i; ?></td>
        <td><?php if ($i%2==0) echo "Par";
            else echo "Impar"?></td>
    </tr>
    <?php
    $i++;
    }
    ?>
</table>
```

Número	Tipo
1	Impar
2	Par
3	Impar
4	Par
5	Impar

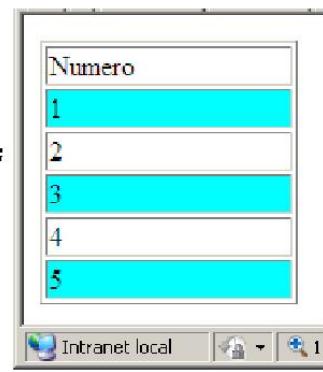


Otra forma de obtener el mismo resultado es la siguiente:

```
<table border="1" width="250">
    <tr>
        <td>Número</td>
        <td>Tipo</td>
    </tr>
    <?php
        $i=1;
        while ($i<=5)
        {
            echo "<tr>";
            <td>$i</td>
            <td>";
            if ($i %2==0) echo "Par";
            else echo "Impar";
            echo " </td>";
            </tr>";
            $i++;
        }
    ?>
</table>
```

El mismo ejemplo se puede hacer con el ciclo do while:

```
<table border="1" width="150">
    <tr>
        <td>Número</td>
    </tr>
    <?php
        $i=1;
        do
        {
            echo "<tr>";
            if ($i%2!=0) echo " bgcolor=#00FFFF";
            echo ">";
            echo "<td> $i</td>";
            echo "</tr>";
            $i++;
        }while ($i<=5);
    ?>
</table>
```



Capítulo 9. ARREGLOS. PARTE 1

9.1.- Declaración

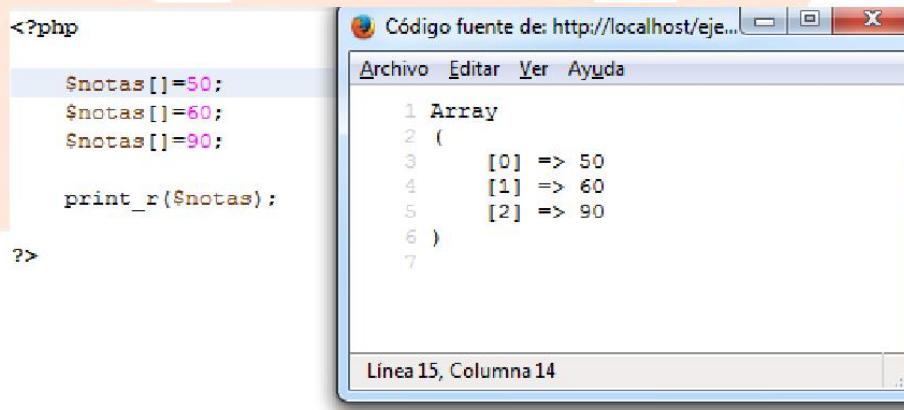
En PHP se pueden declarar los arreglos de varias maneras. La más adecuada sería utilizando la palabra reservada array, de la siguiente manera:

```
$variable = array( valor1, valor2, valor3, ..., valorn);
```

A continuación, varios ejemplos de como declarar arreglos en php:

```
<?php  
  
$notas=array(15, 12, 18, 4, 19, 20); // arreglo de enteros  
$nombres=array("juan", "pedro", "maria", "miguel"); // arreglo de strings  
$sueldos=array(); // arreglo vacio  
  
?>
```

Es una situación muy frecuente necesitar agregar elementos al final de un arreglo. Para ello, Php permite hacerlo asignando un valor usando los corchetes vacíos. El siguiente es un ejemplo:



```
<?php  
  
$notas[] = 50;  
$notas[] = 60;  
$notas[] = 90;  
  
print_r($notas);  
  
?>
```

Código fuente de: http://localhost/eje... | Archivo | Editar | Ver | Ayuda

```
1 Array  
2 ( [0] => 50  
3 [1] => 60  
4 [2] => 90  
5 )  
6  
7
```

Línea 15, Columna 14

9.2.- Acceso

El índice (o clave) de un arreglo puede ser un entero o un string. En este tema se verá como acceder a los elementos de un arreglo cuando los índices son enteros. Los elementos del arreglo se enumeran desde la posición 0 hasta la posición tamaño-1. Por ejemplo:

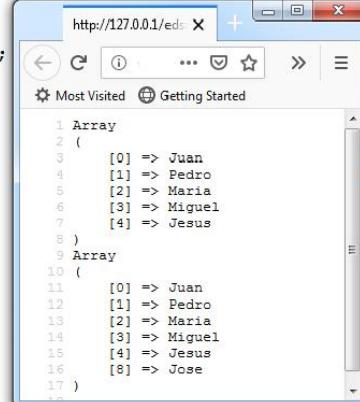
```
<?php  
$nombres=array("Juan","Pedro","Maria","Miguel");  
  
echo $nombres[1];  
if ($nombres[2]=="Ana")  
    echo "Ana esta en la posicion 2";  
else  
    echo "Ana no esta ahí";  
  
echo "El primer nombre es $nombres[0]";  
?>
```

Si se intenta acceder (para leer) a una posición que no existe, se genera un error. El siguiente es un ejemplo:

```
<?php  
$nombres=array("Juan","Pedro","Maria","Miguel");  
  
echo $nombres[4];
```

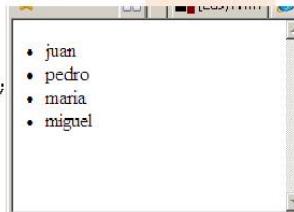
Notice: Undefined offset: 4 in Z:\recursos\Php\ejemplos\nivell\cap9_2_a.php on line 5

Si se intenta acceder a una posición que no existe, pero para escritura no ocurrirá un error, en cambio, se creará la nueva posición. Por ejemplo:



```
<?php
$nombr=array("Juan","Pedro","Maria","Miguel");
$nombr[4]="Jesus";
print_r($nombr);
$nombr[8]="Jose";
print_r($nombr);
```

Para recorrer el arreglo se puede usar un ciclo for, que permita ir desde el primer elemento (posición 0) hasta el último. La información del arreglo puede imprimise con cualquier etiqueta HTML: una tabla, una lista, etc. Por ejemplo:



```
<body>
<?php
$nombr=array("juan","pedro","maria","miguel");
for ($i=0;$i<4;$i++)
    echo "<li>$nombr[$i]</li>";
?>
</body>
```

9.3.- Count y Sum

En ciertas circunstancias puede que no se conozca el tamaño de un arreglo. Para determinar el tamaño de un arreglo se debe usar la función "count", la cual recibe como parámetro el arreglo y retorna un número entero con la cantidad de elementos que tenga el arreglo. Por ejemplo:

```

<body>
<?php
$nombrres=array("juan", "pedro", "maria", "miguel");
?>
<table border="1" width="200">
    <tr>
        <td>Nombres</td>
    </tr>
    <?php
    for ($i=0;$i<count ($nombrres);$i++)
    {
        echo " <tr>
            <td> $nombrres[$i] </td>
        </tr>" ;
    }
    ?>
</table>

```

Nombres
juan
pedro
maria
miguel

Para sumar los elementos contenidos en un arreglo se utiliza la función array_sum. Esta función recibe por parámetro el arreglo y retorna un número con la sumatoria de los elementos contenidos en el arreglo. Por ejemplo:

```

<?php

$ingresos = array(2750,1600,1452,8410,1624,2315);
$total = array_sum($ingresos);

echo "El total ingresado es $total $";

```

Si es necesario calcular el promedio (o average) de los elementos contenidos en el arreglo, se puede combinar con la función array_sum con la función count. Por ejemplo:

```
<?php  
  
$ingresos = array(2750,1600,1452,8410,1624,2315);  
$promedio = array_sum($ingresos)/count($ingresos);  
  
echo "El promedio ingresado es $promedio $";
```



Capítulo 10. ARREGLOS. PARTE 2

10.1.- Buscar Elemento

Para buscar un elemento en un arreglo se utiliza la función `in_array`. Esta función recibe 2 parámetros: el valor que se desea buscar y el arreglo donde se va a buscar. Por ejemplo:

```
<?php

$porcentajes = array("15","20","25","35");
$x = 20;

if (in_array($x,$porcentajes))
    echo "$x existe";
else
    echo "$x no esta en el arreglo";
```

La función `in_array` no verifica el tipo de dato del elemento buscado y el tipo de dato de los elementos del arreglo donde está buscando. En el siguiente ejemplo, el valor buscado es una cadena y los valores son numéricos, aun así, `in_array` retornará true si hay una coincidencia:

```
<?php

$porcentajes = array(15,20,25,35);
$x = "20";

if (in_array($x,$porcentajes))
    echo "$x existe";
else
    echo "$x no esta en el arreglo";
```

Si fuera necesario verificar el tipo de dato del elemento buscado con respecto a los elementos del arreglo, `in_array` tiene un tercer parámetro opcional, cuyo valor por defecto es `false`. Si ese parámetro se pasa con el valor `true`, el resultado será diferente:

```
<?php  
  
$porcentajes = array(15,20,25,35);  
$x = "20";  
  
if (in_array($x,$porcentajes,true))  
    echo "$x existe";  
else  
    echo "$x no esta en el arreglo";
```

Cuando se utilizan cadenas de caracteres para hacer búsquedas, `in_array` es sensible a mayúsculas y minúsculas. En el siguiente ejemplo, la ciudad "Caracas" es encontrada al buscarla en el arreglo porque coinciden las mayúsculas y las minúsculas de uno de los elementos que está en éste:

```
<?php  
  
$ciudades = array("Barquisimeto","Caracas","Merida","Valencia");  
$x = "Caracas";  
if (in_array($x,$ciudades))  
    echo "$x existe";  
else  
    echo "$x no esta en el arreglo";
```

En cambio, en el siguiente ejemplo al buscar "caracas", la función `in_array` retornará `false` porque no hay coincidencia en las mayúsculas y las minúsculas:

```
<?php

$ciudades = array("Barquisimeto", "Caracas", "Merida", "Valencia");
$x = "caracas";
if (in_array($x,$ciudades))
    echo "$x existe";
else
    echo "$x no esta en el arreglo";
```

10.2.- Ordenar Elementos

También es una operación muy común ordenar los elementos de un arreglo. Para este fin, Php cuenta con varias funciones:

- sort: Los elementos serán ordenados de menor a mayor (ascendentemente).
- rsort: Ordena un array en orden inverso.
- natsort: Ordena un array usando un algoritmo de "orden natural", es decir, como lo haría un humano.
- usort: Ordena un array según sus valores usando una función de comparación definida por el usuario.

Otra funciones son especiales para el uso de arreglos asociativos (se verá más adelante):

- asort: Ordena un array y mantiene la asociación de índices.
- arsort: Ordena un array en orden inverso y mantiene la asociación de índices.
- ksort: Ordena un array por clave.
- krsort: Ordena un array por clave en orden inverso.
- uasort: Ordena un array con una función de comparación definida por el usuario y mantiene la asociación de índices.
- uksort: Ordena un array según sus claves usando una función de comparación definida por el usuario.

El siguiente es un ejemplo del uso de las funciones "sort" y "rsort":

```

<?php

$valores=array(80,10,15,22,56);
print_r($valores);
sort($valores);
print_r($valores);
rsort($valores);
print_r($valores);

1 Array
2 (
3     [0] => 80
4     [1] => 10
5     [2] => 15
6     [3] => 22
7     [4] => 56
8 )
9 Array
10 (
11    [0] => 10
12    [1] => 15
13    [2] => 22
14    [3] => 56
15    [4] => 80
16 )
17 Array
18 (
19    [0] => 80
20    [1] => 56
21    [2] => 22
22    [3] => 15
23    [4] => 10
24 )
25

```

La función sort tiene un segundo parámetro opcional llamado "sort_flags", que puede ser usado para modificar el modo de ordenación. Los posibles valores son:

- SORT_REGULAR: compara elementos normalmente (no cambia los tipos). Es el tipo de ordenamiento usado por defecto cuando no se pasa el valor del parámetro.
- SORT_NUMERIC: compara elementos de forma numérica.
- SORT_STRING: compara elementos como cadenas.
- SORT_NATURAL: compara elementos como cadenas usando el "orden natural" (equivalente a natsort()).
- SORT_FLAG_CASE: se puede combinar con SORT_STRING o SORT_NATURAL para ordenar cadenas de forma insensible a mayúsculas/minúsculas.

10.3.- Explode e Implode

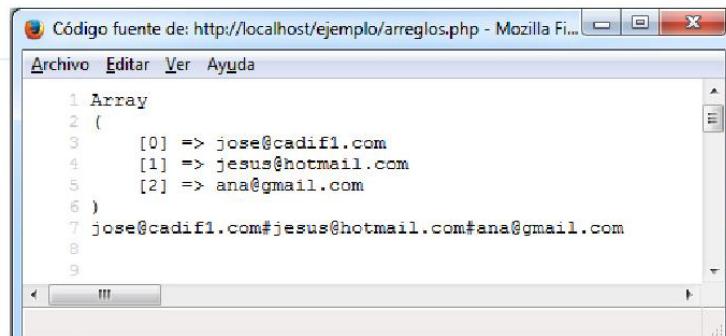
En algunas situaciones hace falta manipular elementos de una cadena como un arreglo o unir los elementos de un arreglo en una cadena. Para estas 2 tareas se utilizan las funciones:

- explode: divide un string en varios string. Retorna un arreglo de string.
- implode: une elementos de un array en un string, unidos por un separador.

Algunos ejemplos del uso de estas funciones:

```
<?php
    $correos_str="jose@cadif1.com,jesus@hotmail.com,ana@gmail.com";
    $correos_arr=explode(",",$correos_str);
    print_r($correos_arr);

    $cadena = implode("#", $correos_arr);
    print_r($cadena);
?>
```



```
Código fuente de: http://localhost/ejemplo/arreglos.php - Mozilla Fi...
Archivo Editar Ver Ayuda
1 Array
2 (
3     [0] => jose@cadif1.com
4     [1] => jesus@hotmail.com
5     [2] => ana@gmail.com
6 )
7 jose@cadif1.com#jesus@hotmail.com#ana@gmail.com
8
9
```

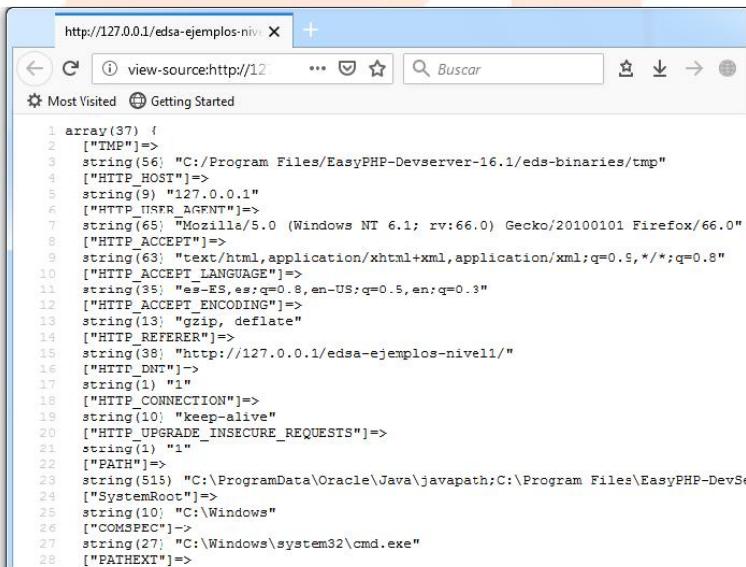
Capítulo 11. ARREGLOS. PARTE 3

11.1.- Arreglos Asociativos

PHP posee un tipo de arreglo muy particular, son los llamados arreglos asociativos. Es un arreglo tal cual como el explicado anteriormente, con la diferencia de que el índice (o clave) es un string en vez de un número entero. Muchas de las variables globales de PHP son arreglos asociativos: `$_SERVER`, `$_POST`, `$_GET` y `$_SESSION` son algunos ejemplos. La variable `$_SERVER` contiene información valiosa sobre la ejecución de un script, entre ellas:

- `PHP_SELF`: El nombre del archivo de script ejecutándose actualmente.
- `SERVER_ADDR`: dirección Ip del servidor donde se está ejecutando el script.
- `REMOTE_ADDR`: La dirección IP desde la cual está viendo la página actual el usuario.
- `SCRIPT_FILENAME`: La ruta del script ejecutándose actualmente en forma absoluta.

Al mostrar el contenido de la variable `$_SERVER` se observan todos los valores que contiene con sus respectivas claves:



```
<?php
var_dump($_SERVER);

```

The screenshot shows a browser window with the URL `http://127.0.0.1/edsa-ejemplos-nivel1/`. The page content displays the output of the `var_dump($_SERVER)` PHP code. The output is a large array with 37 elements, each consisting of a key (string) and its corresponding value (string). Some of the keys and their values are:

- `"TMP"` => `"C:/Program Files/EasyPHP-Devserver-16.1/eds-binaries/tmp"`
- `"HTTP_HOST"` => `"127.0.0.1"`
- `"HTTP_USER_AGENT"` => `"Mozilla/5.0 (Windows NT 6.1; rv:66.0) Gecko/20100101 Firefox/66.0"`
- `"HTTP_ACCEPT"` => `"text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8"`
- `"HTTP_ACCEPT_LANGUAGE"` => `"es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3"`
- `"HTTP_ACCEPT_ENCODING"` => `"gzip, deflate"`
- `"HTTP_REFERER"` => `"http://127.0.0.1/edsa-ejemplos-nivel1/"`
- `"HTTP_DNT"` => `"1"`
- `"HTTP_CONNECTION"` => `"keep-alive"`
- `"HTTP_UPGRADE_INSECURE_REQUESTS"` => `"1"`
- `"PATH"` => `"C:\ProgramData\Oracle\Java\javapath;C:\Program Files\EasyPHP-DevServer-16.1\bin;C:\Windows\system32;C:\Windows;C:\Windows\SYSTEM32;C:\Windows\cmd.exe"`
- `"SystemRoot"` => `"C:\Windows"`
- `"COMSPEC"` => `"cmd.exe"`
- `"PATHEXT"` => `".COM;.EXE;.BAT;.CMD;.VBS;.VBE;.WSF;.WSH;.MSC"`

La forma de acceder a los elementos de un arreglo asociativo es colocando el nombre del índice, en vez de colocar un número (\$arreglo[2]) se coloca un texto (\$arreglo["loquesea"]). Por ejemplo:

```
<?php  
  
echo 'La direccion Ip es '. $_SERVER["REMOTE_ADDR"] ;  
echo 'y El nombre del archivo es '. $_SERVER["PHP_SELF"] ;
```

Resultado:

La dirección Ip es 127.0.0.1 y El nombre del archivo es /edsa-ejemplos-nivel1/cap11_1_b.php

Si se intenta acceder a un elemento cuya clave no existe se generará un error de ejecución:

```
<?php  
  
echo 'La direccion Ip es ' . $_SERVER["REMOTE"] ;  
echo 'y El nombre del archivo es ' . $_SERVER["PHP_SELF"] ;
```

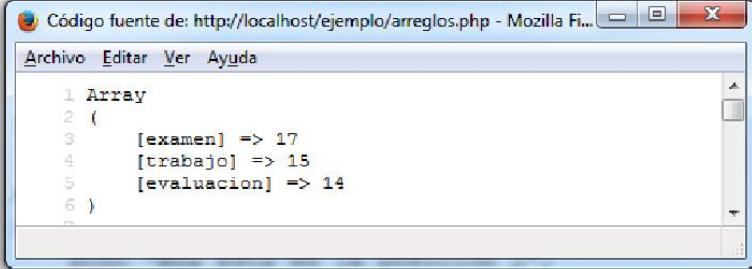
Notice: Undefined index: REMOTE in Z:\recursos\Php\ejemplos\nivel1\cap11_1_b.php on line 3
La dirección Ip es y El nombre del archivo es /edsa-ejemplos-nivel1/cap11_1_b.php

11.2.- Declaración

Además de usar los arreglos asociativos predefinidos en PHP se pueden definir arreglos asociativos particulares. Un ejemplo de como definir un arreglo asociativo es el siguiente:

```
<?php
    // arreglos asociativos
    $notas=array("examen"=>17,"trabajo"=>15,"evaluacion"=>14);

    print_r($notas);
```

?> 

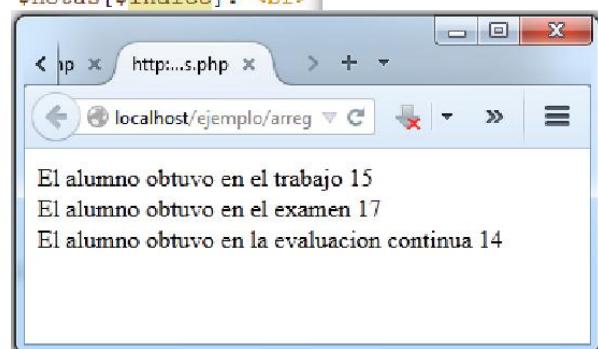
```
Archivo Editar Ver Ayuda
1 Array
2 (
3     [examen] => 17
4     [trabajo] => 15
5     [evaluacion] => 14
6 )
```

En el siguiente ejemplo se muestra como acceder a los elementos del arreglo declarado:

```
<?php
    // arreglos asociativos
    $notas=array("examen"=>17,"trabajo"=>15,"evaluacion"=>14);

    echo "El alumno obtuvo en el trabajo ".$notas['trabajo']."<br>";
    echo "El alumno obtuvo en el examen ".$notas['examen']."<br>";

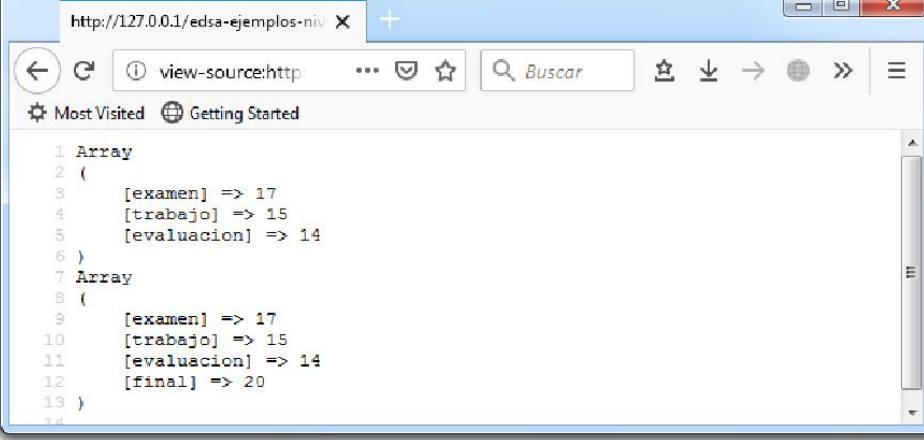
    $indice="evaluacion";
    echo "El alumno obtuvo en la evaluacion continua ".
        $notas[$indice]."<br>"
```

?> 

```
localhost/ultimo/arreglos.php
El alumno obtuvo en el trabajo 15
El alumno obtuvo en el examen 17
El alumno obtuvo en la evaluacion continua 14
```

Así como se pueden agregar nuevos elementos a un arreglo tradicional, también se pueden crear nuevas claves a un arreglo ya existente. Por ejemplo:

```
<?php  
  
$notas=array("examen"=>17,"trabajo"=>15,"evaluacion"=>14);  
print_r($notas);  
$notas["final"] = 20;  
print_r($notas);
```



```
1 Array  
2 ( [examen] => 17  
3 [trabajo] => 15  
4 [evaluacion] => 14  
5 )  
6 Array  
7 ( [examen] => 17  
8 [trabajo] => 15  
9 [evaluacion] => 14  
10 [final] => 20  
11 )  
12  
13 )  
14
```

11.3.- Foreach

El constructor "foreach" proporciona un modo sencillo de iterar sobre arrays. Foreach funciona sólo sobre arrays y objetos, y emitirá un error al intentar usarlo con una variable de un tipo diferente de datos o una variable no inicializada. Existen dos sintaxis para el foreach. La primera es la siguiente:

```
foreach (expresión_array as $valor)  
    sentencias
```

Recorre el array dado por expresión_array. Cuando foreach inicia su ejecución, el puntero interno del array se pone automáticamente en el primer elemento del array. En cada iteración, el valor del elemento actual se asigna a \$valor y el puntero interno del arreglo avanza una posición (así en la próxima iteración se estará accediendo al siguiente elemento).

Ejemplo:



```
<?php // foreach
$names=array("Juan","Pedro","Maria","Miguel");

echo "El listado nombres es:"; 
echo "<ul>";
foreach($names as $n)
    echo "<li> $n </li>";
echo "</ul>";

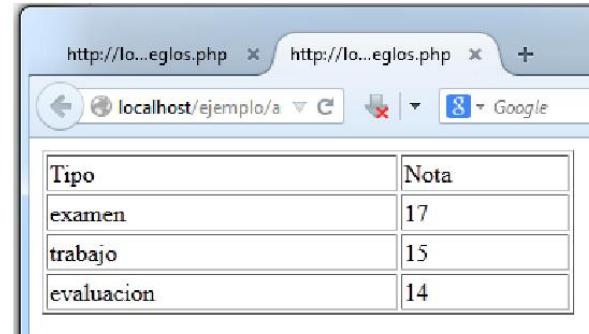
?>
```

La segunda forma de usar "foreach" es para arreglos asociativos. La forma general es la siguiente:

```
foreach (expresión_array as $clave => $valor)
    sentencias
```

Funciona exactamente igual que la forma anterior, con el adicional de que se asigna la clave del elemento actual a la variable \$clave en cada iteración. Por ejemplo:

```
<table border="1" width="350">
<tr>
    <td>Tipo</td>
    <td>Nota</td>
</tr>
<?php
$notas=array("examen"=>17,"trabajo"=>15,"evaluacion"=>14);
foreach($notas as $tipo => $nota)
echo "<tr>
        <td>$tipo</td>
        <td>$nota</td>
    </tr>";
?>
</table>
```



Tipo	Nota
examen	17
trabajo	15
evaluacion	14

Capítulo 12. MANIPULACIÓN DE FORMULARIOS. PARTE 1

12.1.- Recepción de Datos

Para manipular los datos que recibe una página a través de un formulario HTML se debe conocer el método que se usó para enviar los datos. Es importante recordar que existen varios métodos HTTP para enviar datos al servidor, aunque los más usados son: el Post y el Get. En ambos métodos la manera de acceder a los datos enviados es muy similar.

PHP posee 3 variables super globales para manipular los datos que recibe una página, estas son: `$_POST`, `$_GET` y `$_REQUEST`. Las 3 son arreglos asociativos, donde cada posición del arreglo es un dato que está recibiendo nuestra página (posiblemente a través de un formulario), el cual es identificado por el valor de la propiedad "name" del elemento que está adentro del formulario.

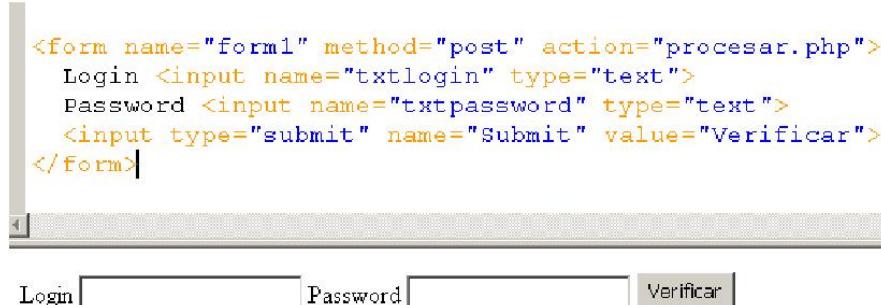
Aquí radica la importancia de decidir el método de envío y el de colocarle a los elementos de los formularios nombres significativos.

En ocasiones puede que no se conozca el método de envío que se usó para enviar la información al servidor, o no se usaron los métodos Get ni Post. Para estos casos se puede usar la variable global `$_REQUEST` que también es una arreglo asociativo como `$_GET` y `$_POST`. Esta variable va a contener los valores recibidos por la página, independientemente del método de envío.

El problema de usar esta variable, es que técnicamente no se sabe por dónde vienen los datos, lo cual es extremadamente peligroso y se convierte en un potencial problema de seguridad que debe evitarse.

12.2.- Recibiendo Datos Enviados con Post

Cuando los datos capturados en un formulario son enviados con el método Post, se usa la variable global `$_POST` para acceder a ellos. Por ejemplo, si una página que posee un formulario como el mostrado en la imagen, cuando el usuario haga click en el botón "Verificar" los datos contenidos en el formulario se enviarán a la página "procesar.php":



Por lo tanto, en la página procesar.php se puede acceder usando la variable `$_POST` a los datos que el usuario escribió en los cuadros de texto que están contenidos en el formulario, de la siguiente manera:

```
<?php
if ($_POST['txtlogin']=="jose" and
    $_POST['txtpassword']=="1234")
{
    echo "Bienvenido !!!";
} else
    echo "Nombre de usuario o contraseña invalidos";
?>
```

Es importante resaltar que a la página procesar.php no se podrá acceder directamente, es decir, la única manera de visitar esta página es visitando la página del formulario y haciendo click en el botón "Verificar".

Es importante recordar que no es un simple hipervínculo, sino enviar los datos a una página que va a procesarlos y a generar una respuesta, por lo tanto, si se intenta acceder a la página procesar.php directamente ocurrirá un error.

En este momento se hace vital hacer una validación donde se pueda verificar si efectivamente la página procesar.php está siendo solicitada por la página que contiene el formulario, en caso de no ser así, se debe mostrar un mensaje de error (manejado por nosotros) o redireccionar a la página del formulario.

12.3.- Recibiendo Datos Enviados Con Get

Cuando una página tiene un hipervínculo a otra página, a través del hipervínculo se le puede transferir información a la página que se está llamando. Esto puede notarse por ejemplo cuando visitamos la página de google.com y hacemos una búsqueda de "cadif1". Se carga en la barra de direcciones una cadena con el texto:

<http://www.google.co.ve/search?hl=es&q=cadif1&meta=>

A esta cadena se le llama QueryString. Esto también puede notarse cuando se usa como método de envío el método Get en vez del Post. Con el método Get los datos capturados en un formulario se envían a través de la URL y son visibles en la barra de direcciones. La desventaja de usar este método es que se puede ver toda la información que se transmite y se puede manipular a discreción.

Los parámetros que se envían a una página se colocan después del nombre de la página y el signo de interrogación "?". Los parámetros deben tener nombres sin espacios en blanco y se separan entre sí con un "&". Por ejemplo:

<http://www.cadif1.com/buscarcurso.php?nombre=php&tipo=1&turno=manana>

En este ejemplo se está solicitando la página buscarcurso.php del dominio cadif1.com, y se le están enviando 3 parámetros: el parámetro "nombre" con el valor "php", "tipo" con el valor "1" y otro de nombre "turno" con el valor "manana". Se pueden enviar tantos parámetros como haga falta y con valores que tenga cualquier longitud (pueden inclusive no tener valor).

La página que recibe los datos debe ser una página dinámica obligatoriamente, y la forma de manejar estos datos recibidos es a través de la variable global `$_GET`. Cada parámetro recibido es una posición del arreglo `$_GET`. Siguiendo con el ejemplo anterior donde se reciben los parámetros `nombre=php`, `tipo=1` y `turno=manana`:

```
<?php  
// este código es de la página buscarcurso.php  
  
echo "El nombre del curso es " . $_GET["nombre"];  
echo "El tipo de curso buscado es " . $_GET["tipo"];  
echo "El turno buscado es " . $_GET["turno"];  
  
?>
```

Capítulo 13. MANIPULACIÓN DE FORMULARIOS. PARTE 2

13.1.- La Función Isset

La función "isset" verifica si una variable existe (si ya ha sido declarada). Si la variable existe, isset retornará true, sino retornará false. Isset no toma en cuenta si la variable está vacía ("") o tiene un cero (0), ya que una variable puede haberse declarado asignándole una cadena vacía o un cero (0) y por lo tanto desde ese momento empieza a existir. Por ejemplo:

```
$var = "";  
  
// Esto evaluara a TRUE asi que el texto se imprimira.  
if (isset($var)) {  
    echo "Esta variable esta definida, asi que se imprimira esto.";  
}
```

En este ejemplo la variable \$var se declaró al inicializarla con una cadena vacía y por lo tanto isset devuelve true. Para verificar si una variable está vacía utilice la función empty().

El isset es útil para hacer la validación que resuelve el problema planteado en el capítulo anterior. Si al verificar con isset la existencia de uno de los inputs del formulario (cualquiera de ellos) este devuelve false, se puede asumir que el llamado de la página "procesar.php" no fue desde la página del formulario, y por lo tanto no se debe procesar la información del mismo. El código sería como el siguiente:

```
<?php
if (isset($_POST['txtlogin'])==false)
{
    echo "Error. No puede visitar esta pagina directamente";
    echo "Visite esta <a href='formulario.html'>pagina</a>";
} else
{
    if ($_POST['txtlogin']=="jose" and
        $_POST['txtpassword']=="1234" )
    {
        echo "Bienvenido !!!";
    } else
    {
        echo "Nombre de usuario o contraseña incorrectos";
    }
}
?>
```

13.2.- Redireccionamiento de Páginas

El redireccionamiento de páginas sucede cuando un usuario escribe el nombre de una página en la barra de direcciones de su navegador y se carga otra página. Este redireccionamiento se puede hacer con código del lado de cliente o con código del lado del servidor.

Aunque en algunos casos esto sea imperceptible para el usuario, el redireccionamiento del lado del servidor puede ser más rápido que el del lado del cliente. La función de php para hacer redireccionamientos se llama header, y se usa de la siguiente manera:

```
header('location: pagina')
```

Donde "pagina" puede ser una página dinámica o estática, de nuestro sitio o de otro sitio. Por ejemplo:

```
header('location: http://www.cadif1.com');
```

La validación hecha en el tema anterior con isset también se puede hacer redireccionando a la página que pide los datos (en vez de mostrar un mensaje de error). La manera de hacerlo es la siguiente:

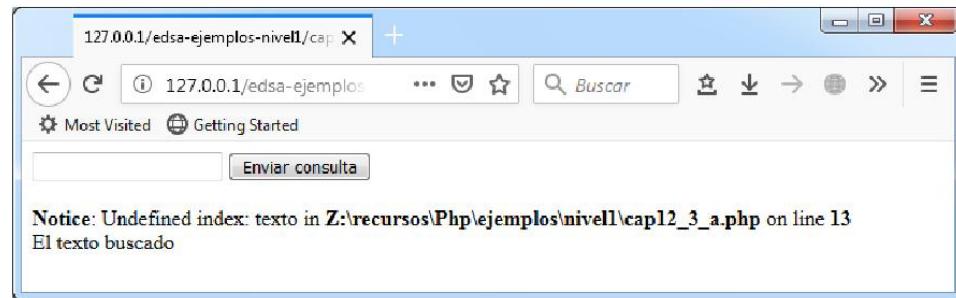
```
<?php
if (isset($_POST['txtlogin'])==false)
{
    header("location: formulario.html");
} else
{
    if ($_POST['txtlogin']=="jose" and
        $_POST['txtpassword']=="1234" )
    {
        echo "Bienvenido !!!";
    } else
    {
        echo "Nombre de usuario o contraseña incorrectos";
    }
}
?>
```

13.3.- Procesar Datos en la Misma Página

Además de procesar datos de un formulario en un archivo Php distinto a la página que contiene el formulario, también se pueden procesar los datos enviados en la misma página donde está el formulario. Para lograrlo, el valor del atributo "action" de la etiqueta <form> debe tener el nombre del archivo que contiene el formulario. En el siguiente ejemplo se muestra como usar la variable \$_SERVER["PHP_SELF"]:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="post">
            <input type="text" name="texto" />
            <input type="submit" name="boton" />
        </form>
        <?php
            echo "El texto buscado ". $_POST["texto"];
        ?>
    </body>
</html>
```

El problema es que se genera un error la primera vez que se carga la página porque al intentar acceder a los datos enviados, éstos no han sido enviados aún:



Adicionalmente, antes de usar los datos enviados se debe hacer uso de la función `isset` para verificar efectivamente que se estén enviando los datos y así evitar que se muestren errores en la primera carga de la página. Por ejemplo:

```
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
  </head>
  <body>
    <form action="<?php echo $_SERVER['PHP_SELF']?>" method="post">
      <input type="text" name="texto" />
      <input type="submit" name="boton" />
    </form>
    <?php
      if (isset($_POST["texto"]))
        echo "El texto buscado ".$_POST["texto"];
    ?>
  </body>
</html>
```

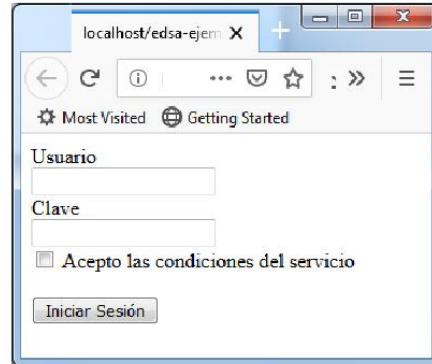
Capítulo 14. MANIPULACIÓN DE FORMULARIOS. PARTE 3

14.1.- Trabajar Con Checkbox

Al momento de procesar los datos de un formulario hay ciertas particularidades. Por ejemplo, cuando un input es de type "checkbox", el value de un checkbox es enviado sólo si el usuario lo marca, en cuyo caso, se envía el value asociado al checkbox. Por lo tanto, se debe usar la función isset para verificar si ha sido enviado o no. Por esta razón no se puede usar un checkbox para hacer la validación de envío de los datos a una página. El siguiente ejemplo muestra un formulario con un checkbox:

```
<form method="post">
    Usuario<br>
    <input type="text" name="usuario" /><br>
    Clave<br>
    <input type="password" name="clave" /><br>

    <input type="checkbox" name="aceptar" value="1"/>
    Acepto las condiciones del servicio <br><br>
    <input type="submit" name="iniciar" value="Iniciar Sesión" />
</form>
```



La página para procesar los datos del formulario debe tener un código como el siguiente:

```
<?php

if (!isset($_POST["aceptar"])){
    echo "Debe aceptar las condiciones del servicio";
} else
    echo "El value asociado al check es ".$_POST["aceptar"];

?>
```

14.2.- Enviar Datos Como Arreglos

Para responder a ciertos requerimientos particulares, es posible enviar los datos de un formulario bajo un mismo nombre, haciendo que los datos sean enviados como un arreglo. Para lograrlo, es necesario que los elementos del formulario que se necesitan enviar bajo un mismo nombre (`<input>`, `<select>`, `<textarea>`) tengan el mismo valor en el atributo "name" y unos corchetes vacíos, por ejemplo: "nombres[]". Estos corchetes son los que hacen que sea un arreglo, de lo contrario, sólo se enviará un solo elemento.

```
<form method="post">
    Nota 1<input type="text" name="notas[]" />
    Nota 2<input type="text" name="notas[]" />
    Nota 3<input type="text" name="notas[]" />

    <input type="submit" value="Registrar" />
</form>
```

Para procesar todos los datos enviados, se debe acceder a éstos como un arreglo:

```
<?php

if (isset($_POST["notas"])){
    $notas = $_POST["notas"];
    for ($i=0;$i<count($notas);$i++){
        echo $notas[$i];
    }
}

?>
```

Cuando se necesita enviar el valor de varios checkboxes como arreglo es diferente, porque se agregan al arreglo sólo los checkboxes que fueron marcados por el usuario, si ninguno fue marcado, al evaluar con "isset" va a retornar falso. Por ejemplo:

Formulario:

```
<form method="post">
    <h2>Idiomas</h2>
    <input type="checkbox" name="idiomas[]" value="ingles" />Ingles
    <input type="checkbox" name="idiomas[]" value="espanol" />Español
    <input type="checkbox" name="idiomas[]" value="frances" />Frances

    <input type="submit" value="Registrar" />
</form>
```

Procesamiento:

```
<?php

if (isset($_POST["idiomas"])){
    $idiomas = $_POST["idiomas"];
    echo "Los idiomas seleccionados son:";
    for ($i=0;$i<count($idiomas);$i++){
        echo $idiomas[$i];
    }
} else
    echo "No selecciono ningun idioma";

?>
```

Capítulo 15. MANIPULACIÓN DE FORMULARIOS. PARTE 4

15.1.- Llenar Select Dinámicamente

Los elementos de un select (<option value="valor"> Texto </option>) se pueden agregar estáticamente o dinámicamente con PHP. Una forma para generar un select dinámicamente con PHP se muestra en la siguiente imagen:

```
<body>
  <form>
    Dia
    <select name="dia">
      <?php
        for ($i=1;$i<=5;$i++)
          echo "<option>$i</option>";
      ?>
    </select>
  </form>
```

La etiqueta <option> tiene el atributo "value", al cual se le puede asignar un valor distinto a lo que se le muestra al usuario. Cuando se asigna el atributo value, lo que contenga este atributo es lo que se enviará, si no se especifica un "value", se enviará el texto que está entre las etiquetas < option>. El siguiente ejemplo muestra como llenar un select con el contenido de un arreglo, asignando el atributo "value":

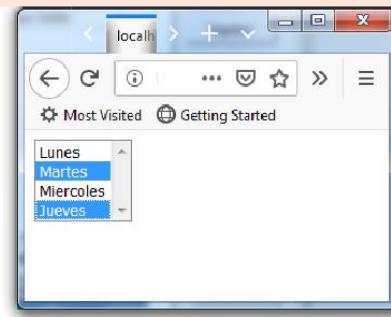
```
<?php
$dias = array("Lunes", "Martes", "Miercoles", "Jueves");
?>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
  </head>
  <body>
    <select name="dia">
      <?php
        for ($i=0;$i<count($dias);$i++)
          echo "<option value='".$i."'>".
            $dias[$i].
          "</option>";
      ?>
    </select>
  </body>
</html>
```

En el ejemplo anterior, al seleccionar el día "Martes" se enviará el valor 1.

15.2.- Enviar Select Multiples

La etiqueta "select" tiene un atributo especial que permite al usuario visualizar y seleccionar varios valores de la lista. El atributo se llama "multiple" y se coloca en la etiqueta de apertura del select. El siguiente ejemplo es muy parecido al ejemplo del tema anterior, con la diferencia de que se mostrarán y se podrán seleccionar varios valores a la vez:

```
<body>
    <select name="dia" multiple>
        <?php
            for ($i=0;$i<count($dias);$i++)
                echo "<option value='\$i'>" .
                    $dias[$i] .
                "</option>";
        ?>
    </select>
</body>
```



Cuando se utiliza este atributo, para poder enviar todos los valores seleccionados por el usuario se debe asignar un nombre colocándole al final un corchete que abre y uno que cierra ([]), de lo contrario, sólo se enviará el último valor seleccionado por el usuario. El procesamiento es idéntico al realizado cuando se coloca a varios elementos el mismo nombre usando corchetes, es decir, manejando los valores como un arreglo.

Formulario:

```
<select name="dia[]" multiple>
<?php
    for ($i=0;$i<count($dias);$i++)
        echo "<option value='\$i'>" .
            $dias[$i] .
        "</option>";
?>
</select>
<input type="submit" value="Enviar" />
```

Procesamiento:

```
<?php
    if (isset($_POST["dia"])){
        echo "Días seleccionados:" ;
        $dias = $_POST["dia"];
        for ($i=0;$i<count($dias);$i++)
            echo $dias[$i];
    }else
        echo "No selecciono ningún dia";
?>
```