NAVEGAR en INTERNET

Creación de un portal con PHP y MySQL

4ª Edición

Jacobo Pavón Puertas

Ra-Ma®

Creación de un portal con PHP y MySQL

4.ª Edición

Creación de un portal con PHP y MySQL

4.ª Edición

Jacobo Pavón Puertas

Ra-Ma®

La ley prohíbe
copiar o imprimir este libro

CREACIÓN DE UN PORTAL CON PHP Y MySQL. 4ª EDICIÓN
© Jacobo Pavón Puertas

© De la Edición Original en papel publicada por Editorial RA-MA
ISBN de Edición en Papel: 978-84-9964-024-2

Editado por:
RA-MA, S.A. Editorial y Publicaciones
Calle Jarama, 33, Polígono Industrial IGARSA
28860 PARACUELLOS DE JARAMA, Madrid
Teléfono: 91 658 42 80
Fax: 91 662 81 39
Correo electrónico: editorial@ra-ma.com
Internet: www.ra-ma.es y www.ra-ma.com

Diseño Portada: Antonio García Tomé

ISBN: 978-84-9964-382-3

E-Book desarrollado en España en septiembre de 2014

A mis dos amores: mi mujer
Gema y mi hijo Alejandro,
y a mi familia.

ÍNDICE

INTRODUCCIÓN

Con la lectura de esta obra, se pretende enseñar al lector cómo crear aplicaciones para nuestras páginas web. Para ello, no es necesario ningún conocimiento de programación, ya que lo que se pretende con este libro es enseñar desde cero a programar las aplicaciones más avanzadas empleadas en portales de Internet. Se mostrarán todos los contenidos necesarios en cuanto a la programación en PHP y la utilización de la base de datos MySQL, esta última realmente útil.

Con este lenguaje, se consigue crear interactividad y webs dinámicas para una mejor dinámica entre las páginas web.

Al lector no le resultará nada difícil el aprender a realizar sus propias aplicaciones para sus páginas web, porque gracias a los ejemplos que se muestran, y a los conocimientos que se adquieren al inicio de la lectura, será capaz de aprender y de modificar códigos a su gusto, según las necesidades del lector.

Sería de gran utilidad para el lector que tuviera unos conocimientos básicos de programación HTML, ya que según se avance en la lectura de este libro, se irán introduciendo nuevos

términos, como pueden ser formularios, etc., y tener una idea al menos básica de este lenguaje.

Si durante la lectura de esta obra, o posteriormente, al lector le surgen dudas acerca de algún contenido, puede contactar mediante correo electrónico con el autor de la obra escribiendo a: jacobopavon@yahoo.es.

Capítulo 1

PHP Y MySQL

1.1 PHP

¿Qué es PHP?

PHP es un lenguaje de alto nivel que se ejecuta en el servidor.

¿Qué quiere decir que se ejecuta en el servidor?

Un lenguaje de servidor es aquél que se ejecuta en el servidor donde están alojadas las páginas, al contrario que otros lenguajes que son ejecutados en el propio navegador.

¿Qué ventajas tiene el ser un lenguaje de servidor?

La principal ventaja es que, al ejecutarse el código en el servidor, todas nuestras páginas van a poder ser vistas en cualquier ordenador, independientemente del navegador que tenga. En cambio, el gran problema de que se ejecute el código en el navegador es que

muchos navegadores no son capaces de entender todo el código, lo que presentaría errores al mostrar el resultado de las páginas.

¿Qué otras ventajas presenta el lenguaje PHP?

Principalmente, que se trata de un lenguaje de programación gratuito y, por tanto, todo el mundo puede utilizarlo sin ningún coste, frente a otros lenguajes cuyo *software* es necesario comprar para su utilización.

En este libro tratamos el lenguaje PHP en su última versión, la versión 5.3.

En la figura 1.1 podemos ver en un gráfico el proceso que se realiza a la hora de visitar una página PHP.

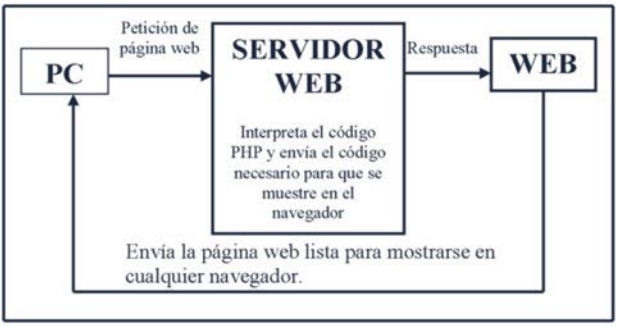

Figura 1.1

1.2 MySQL

MySQL es, por otro lado, la base de datos elegida por la gran mayoría de programadores en PHP. Soporta el lenguaje SQL y la conexión de varios usuarios, pero, en general, se utiliza para aplicaciones de tamaño pequeño-medio.

Al igual que PHP, su principal ventaja reside en que es una base de datos gratuita.

En este libro mostraremos cómo se instala y el uso de la base de datos MySQL con PHP con la última versión existente de MySQL, la versión 5.1.

Capítulo 2

CONFIGURACIÓN DEL *SOFTWARE* NECESARIO

2.1 INSTALACIÓN Y CONFIGURACIÓN DE APACHE

Para instalar Apache, nos dirigiremos a la página web http://www.apache.org, donde podremos descargar directamente la última versión. O también podremos insertar directamente en nuestro navegador la siguiente dirección, donde podremos descargarlos directamente sin necesidad de buscarlo por su web. La dirección sería: http://archive.apache.org/dist/directory/installers/apachedr-1.0-RC1 donde seleccionamos el fichero:

Apache.1.0-RC1-win32-setup.exe

NOTA: es conveniente comprobar la versión actual del fichero a descargar, ya que durante la edición de este libro las versiones existentes eran las que se citan en el mismo, pero es muy probable que estas hayan sido actualizadas con el paso del tiempo, pero esto no varía en absoluto el funcionamiento de nuestras aplicaciones siguiendo los pasos de este libro.

Una vez que tengamos este fichero en nuestro ordenador, lo ejecutaremos para poder configurar nuestro servidor Apache. Iremos recorriendo todas las pantallas hasta que aparezca una ventana donde se debe configurar el dominio, el servidor, el correo electrónico y el puerto que utilizaremos.

Esta ventana que debemos configurar nos la encontramos en la figura 2.1.

Figura 2.1

En ella introduciremos los siguientes datos en las casillas correspondientes:

- Network Domain: localhost.
- Server Name: Mi_servidor.
- Administrator's Email Address: tuemail@tudominio.com.
- Y marcamos la opción: for All Users, on Port 80, as a Service-Recommended.

Para finalizar, sólo nos quedaría indicar que nos instale la opción típica y ya estarán listos para instalarse los ficheros en nuestro ordenador de forma automática.

Ahora sólo quedaría configurar el fichero **httpd.conf** para que Apache funcione según nuestras necesidades; seguiremos los pasos de la imagen correspondiente a la figura 2.2 para poder abrir el archivo httpd.conf y realizar los cambios.

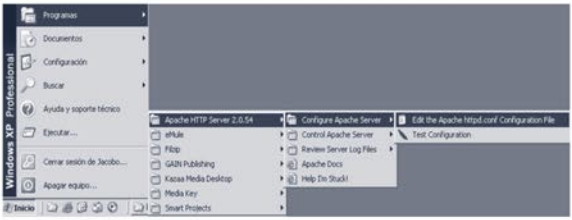

Figura 2.2

Lo primero que haremos es cargar el módulo de PHP para Apache, para lo que buscaremos la cadena de texto **LoadModule**:

 # LoadModule foo_module modules/mod_foo.so

Y añadiremos justo debajo la línea siguiente:

LoadModule php5_module C:/php/php5apache2.dll

Quedando finalmente algo así:

 # Example:

 # LoadModule foo_module modules/mod_foo.so

#

LoadModule php5_module C:/php/php5apache2.dll

La ruta **C:/php/php5apache2.dll** es donde está guardado el módulo correspondiente a PHP 5; luego, cuando instalemos PHP, debemos asegurarnos de que la ruta donde estará PHP será **C:/php/**.

El siguiente paso es configurar la ruta donde se guardarán y cargarán nuestros ficheros. Para ello, buscamos la cadena de texto: **DocumentRoot**.

Y encontraremos algo así:

 # DocumentRoot: The directory out of which you will serve your

 # documents. By default, all requests are taken from this directory, but

 # symbolic links and aliases may be used to point to other locations.

#

DocumentRoot "C:/Archivos de programa/Apache Group/Apache/htdocs"

Pues bien, el cambio que debemos realizar es el de la ruta, donde pondremos algo así:

DocumentRoot "C:/ficheros"

Para ello, debemos crear en nuestra raíz C:/ un directorio llamado ficheros, que será donde guardaremos todos nuestros ficheros.

A continuación buscamos la siguiente cadena de texto: **DirectoryIndex**.

Encontraremos algo así:

 # DirectoryIndex: sets the file that Apache will serve if a directory

 # is requested.

#

 # The index.html.var file (a type-map) is used to deliver content-

 # negotiated documents. The MultiViews Option can be used for the

 # same purpose, but it is much slower.

#

DirectoryIndex index.html index.html.var

Modificaremos la última línea para que quede así:

DirectoryIndex index.html index.htm index.php index.php3 index.php4 index.phtml index.html.var

Así, si creamos varios directorios, al acceder a alguno de ellos, ejecute el index predeterminado si existe y así evitará un mensaje de error.

A continuación de esta última línea que hemos modificado insertaremos las siguientes líneas:

AddType application/x-httpd-php .php .php3 .php4 .phtml .php5

AddType application/x-httpd-php-source .phps

2.2 INSTALACIÓN Y CONFIGURACIÓN DE PHP

Para la instalación de PHP, necesitamos la última versión del mismo, que es la 5.1. Para descargarla, nos dirigimos al siguiente sitio de PHP, ***http://www.php.net/*** y en la sección *download* seleccionamos el siguiente fichero:

PHP 5.3.3 (tar.bz2)

> **NOTA:** es conveniente comprobar la versión actual del fichero a descargar, ya que durante la edición de este libro las versiones existentes eran las que se citan en el mismo, pero es muy probable que estas hayan sido actualizadas con el paso del tiempo, pero esto no varía en absoluto el funcionamiento de nuestras aplicaciones siguiendo los pasos de este libro.

Una vez hayamos descargado el fichero, el siguiente paso será descomprimirlo en nuestro ordenador. Para ello, lo que haremos es crear en **C:/** un directorio llamado **php** y descomprimimos ahí todos los ficheros.

A continuación, una vez descomprimido, copiaremos los ficheros con extensión **.dll** (librerías) y los pegaremos en el directorio **System** si utilizamos Windows 9x, o los pegaremos en **System32** si nuestro sistema operativo es Windows NT, 2000, XP o Vista.

El siguiente paso es configurar el fichero **php.ini**. Para ello accedemos a la carpeta **C:/php** donde encontraremos el fichero **php.ini-dist**, que debemos de renombrarlo a **php.ini**. Abrimos este fichero con un editor de textos para poder modificarlo.

Lo primero que buscamos es la cadena: **register_globals**, cuyo valor es *Off*, y debemos de modificarlo a *On*, quedando algo así:

; register_globals to be on; Using form variables as globals can easily lead

; to possible security problems, if the code is not very well thought of.

register_globals = On

Con esta modificación lo que hemos hecho es admitir variables globales.

A continuación, vamos a indicar a PHP dónde se guardan las extensiones. Para ello buscamos la cadena: ***extensión_dir*** y la modificaremos para que quede así:

;Directory in which the loadable extensions (modules) reside.

extension_dir = "c:/php/ext"

donde ***C:/php/ext*** es la ruta donde tenemos guardadas las librerías con la extensiones de PHP.

A continuación, vamos a activar la extensión necesaria para que PHP nos permita manejar funciones relacionadas con MySQL. Para ello vamos a buscar la cadena de texto: **Windows Extensions**.

Nos encontraremos una serie de extensiones, todas ellas en principio desactivadas, pero a nosotros la que ahora mismo nos interesa es la extensión en la que pone: *;extension=php_mysql.dll*, donde deberemos quitar el ";" para poder activarla.

Por último, en este libro viene un ejemplo más adelante en el que se explica cómo crear una aplicación para subir ficheros al servidor. Para realizar este ejemplo, es necesario indicar dónde queremos que se almacenen los ficheros que se suban al servidor. Para ello buscamos la cadena: **upload_tmp_dir**.

Encontraremos algo así:

; Temporary directory for HTTP uploaded files (will use system default if not

; specified).

;upload_tmp_dir =

; Maximum allowed size for uploaded files.

upload_max_filesize = 2M

Y debemos cambiarlo para que quede de este modo:

; Temporary directory for HTTP uploaded files (will use system default if not

; specified).

upload_tmp_dir = "c:/ficheros/upload/"

; Maximum allowed size for uploaded files.

upload_max_filesize = 5M

Como se puede observar, hemos quitado el ";" delante de **upload_tmp_dir** y le hemos indicado la ruta donde queremos guardar los ficheros subidos (esta carpeta debemos crearla, ya que en

principio no existe). Y por último, hemos modificado el tamaño máximo de los ficheros a subir al servidor de 2M a 5M, para evitar problemas si tenemos algún fichero algo mayor.

Una última modificación que haremos es indicar el directorio donde queremos que se almacenen los ficheros temporales de las sesiones. Para ello buscamos la cadena: **session.save_path**.

Nos encontramos algo así:

; session.save_path = "N;/path"

Y lo modificamos para que quede de este modo:

session.save_path = "C:/ficheros/sesiones/"

Por último, debemos guardar el fichero **php.ini** y copiarlo en la carpeta Windows.

2.3 INSTALACIÓN Y CONFIGURACIÓN DE MySQL

Para la instalación de MySQL, necesitamos la última versión de la base de datos, en este caso la versión 5.0.26. Para ello nos dirigimos al siguiente sitio de MySQL, **http://www.mysql.com/**, accedemos a la sección **download** y seleccionamos **MySQL Comunity Server** y después descargar **MySQL versión 5.1.49**.

> **NOTA:** es conveniente comprobar la versión actual del fichero a descargar, ya que durante la edición de este libro las versiones existentes eran las que se citan en el mismo, pero es muy probable que estas hayan sido actualizadas con el paso del tiempo, pero esto no varía en absoluto el funcionamiento de nuestras aplicaciones siguiendo los pasos de este libro.

En este caso, hemos optado por un fichero no autoinstalable, por lo que primero procederemos a descomprimir el fichero descargado en nuestro disco duro. Para ello creamos una carpeta que llamaremos *mysql*, que será donde descomprimiremos los ficheros.

Una vez descomprimido, accedemos a esa carpeta, nos encontraremos la carpeta: **mysql-5.1.49-win32**, accedemos a ella y hallaremos más carpetas y ficheros. De éstos, nos centramos en la carpeta **bin**, accedemos a ella y ejecutamos el fichero **winmysqladmin.exe**, que nos llevará a configurar MySQL.

Lo primero que nos encontraremos será una ventana, como la de la figura 2.3, en la que debemos introducir un nombre de usuario y una contraseña.

Con estos pasos, ya tenemos lista nuestra base de datos MySQL para utilizarla.

Es muy importante recordar el nombre de usuario y la contraseña introducida al configurar MySQL, ya que los utilizaremos siempre que queramos conectarnos a una base de datos.

Figura 2.3

2.4 INSTALACIÓN Y CONFIGURACIÓN DE PHPMYADMIN

Para la instalación de phpMyAdmin, nos dirigiremos a la página web *http://www.phpmyadmin.net* y accedemos a la sección **download** para descargarnos la última versión del programa, la versión 3.3.5.

> **NOTA:** es conveniente comprobar la versión actual del fichero a descargar, ya que durante la edición de este libro las versiones existentes eran las que se citan en el mismo, pero es muy probable que estas hayan sido actualizadas con el paso del tiempo, pero esto no varía en absoluto el funcionamiento de nuestras aplicaciones siguiendo los pasos de este libro.

Una vez hayamos descargado el fichero, lo descomprimimos en la carpeta donde vamos a guardar nuestros fichero PHP, que ya indicamos anteriormente, es decir, hay que descomprimirlo dentro de la carpeta: **c:/ficheros**.

Cuando lo hayamos descomprimido, es conveniente cambiar el nombre de la carpeta que nos ha creado, **/phpMyAdmin-3.3.5/**, por **/phpmyadmin/**, quedando finalmente algo así: **c:/ficheros/phpmyadmin**.

Una vez descomprimido, debemos configurar phpMyAdmin. Para ello entramos en la carpeta **phpmyadmin** y editamos el fichero **config.inc.php**, donde modificaremos una serie de parámetros, como a continuación se indica:

Lo primero que debemos hacer es buscar la cadena: ***$cfg['PmaAbsoluteUri']= ' '***; y modificarla, poniendo: ***$cfg['PmaAbsoluteUri'] = 'http://localhost/phpmyadmin'***; con lo que indicamos la ruta donde se encuentra phpMyAdmin.

El siguiente paso es buscar la cadena:

$cfg['Servers'][$i]['user']='root', donde debemos modificar el parámetro **'root'** por nuestro usuario que utilizamos en MySQL.

Y, por último, buscamos la cadena:

$cfg['Servers'][$i]['password'] = ' ', donde introduciremos nuestra contraseña en el espacio entrecomillado.

Ya tenemos listo phpMyAdmin para ser utilizado. Para comprobar su funcionamiento, abrimos nuestro navegador y tecleamos *http://localhost* y, accedemos a phpMyAdmin, y aparecerá algo similar a la imagen de la figura 2.4.

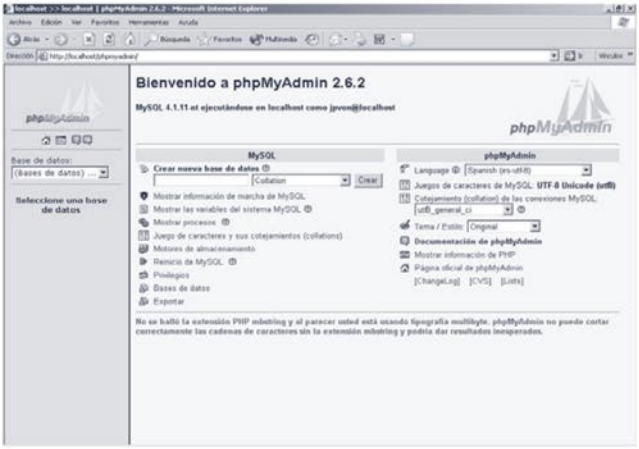

Figura 2.4

En el capítulo 12, empezaremos a explicar cómo trabajar con MySQL, donde entraremos en profundidad a manejar la aplicación phpMyAdmin que, como veremos más adelante, nos va a ayudar mucho en la gestión de nuestras bases de datos, ya que gracias a este

entorno gráfico nos va a resultar muy sencillo realizar todo tipo de operaciones con bases de datos.

Existe un apartado exclusivo donde aprenderemos a manejar esta aplicación para crear bases de datos, crear tablas, borrar tablas, borrar bases de datos, insertar registros, modificar registros, borrar registros y un sinfín de operaciones más.

2.5 OTRAS OPCIONES DE INSTALACIÓN

En este apartado se explicarán dos aplicaciones libres de uso y muy sencillas de manejar para poder trabajar con Apache, PHP, MySQL y phpMyAdmin. Se trata de las aplicaciones WAMP y AppServer. Entraremos en mayor profundidad a explicar la instalación de WAMP.

2.5.1 WAMP

La aplicación WAMP es posible descargarla de forma gratuita desde la web *http://www.wampserver.com*, donde podemos descargar accediendo a la sección **download**, la versión más reciente, que se trata de WAMP5 2.0.

> **NOTA:** es conveniente comprobar la versión actual del fichero a descargar, ya que durante la edición de este libro las versiones existentes eran las que se citan en el mismo, pero es muy probable que estas hayan sido actualizadas con el paso del tiempo, pero esto no varía en absoluto el funcionamiento de nuestras aplicaciones siguiendo los pasos de este libro.

Con esta aplicación dispondremos de inmediato de todas las aplicaciones necesarias para poder empezar a trabajar; en concreto, con esta última versión de WAMP5, dispondremos de las siguientes versiones:

- Apache 2.2.11.
- PHP 5.3.0.
- MySQL 5.1.36.
- PhpMyAdmin 3.2.0.1.

Como podemos comprobar, existe una aplicación de la cuál no hemos hablado hasta este momento: se trata del gestor de bases de datos SQLitemanager. En este libro no entraremos a explicar esta aplicación, ya que con el resto de servicios que hemos explicado podemos realizar todo tipo de aplicaciones sin necesidad de emplear SQLitemanager.

Lo primero que debemos hacer es ejecutar el archivo que nos hemos descargado para instalar WAMP5 en nuestro ordenador.

Una vez ejecutado, el primer paso es seleccionar la carpeta donde queremos instalar la aplicación; en este caso, como muestra la figura 2.5, por defecto nos aparece la carpeta **wamp**.

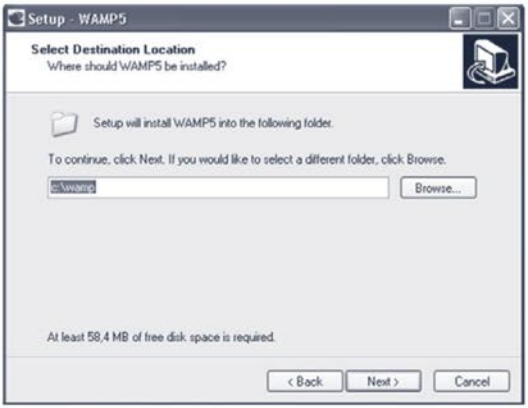

Figura 2.5

El siguiente paso en el proceso de instalación es indicar el nombre de la carpeta que queremos que aparezca en el menú de programas del menú Inicio; por defecto podemos dejar el que aparece: **WampServer**.

A continuación, nos pregunta si queremos que se inicie WAMP cada vez que iniciemos nuestro ordenador; de este modo nos ahorramos el tiempo de tener que estar arrancando todos los servicios cada vez que arranquemos nuestro sistema. Los servicios que arranca WAMP por defecto, son el servidor Apache y el servidor de bases de datos MySQL. Podemos ver esta opción de instalación en la siguiente imagen, figura 2.6.

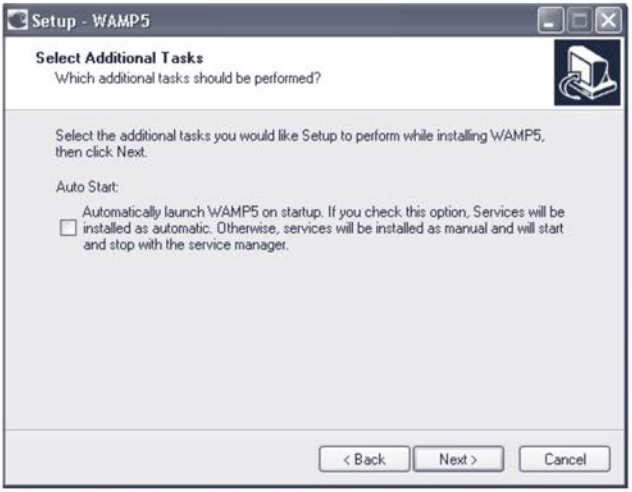

Figura 2.6

A continuación, debemos indicar la carpeta con la que vamos a trabajar, es decir, dónde queremos almacenar nuestros archivos para poder visualizarlos. En nuestro caso, como se puede ver en la siguiente imagen, figura 2.7, hemos creado una carpeta llamada

ficheros, que será donde se almacenarán todos los archivos. Dentro de esta carpeta se pueden crear subcarpetas para diferenciar cada uno de los proyectos donde estamos trabajando.

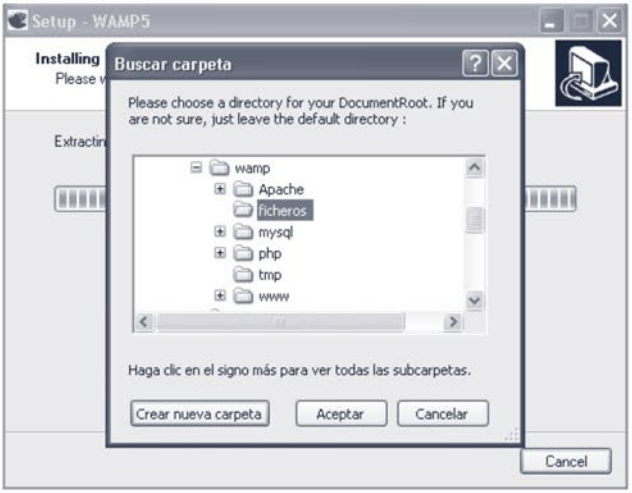

Figura 2.7

El último paso consiste en indicar el navegador que queremos utilizar por defecto al trabajar con WAMP. En nuestro caso y para los ejemplos de este libro hemos empleado el navegador Internet Explorer, pero se podría emplear cualquier otro, como por ejemplo podría ser Firefox.

Si nos fijamos en la barra de tareas, veremos junto al reloj que nos ha aparecido un nuevo icono: es el correspondiente a la instalación que hemos realizado de WAMP5, y si pulsamos sobre él con el botón izquierdo, podremos acceder a todos los servicios de nuestra nueva aplicación. Podemos ver el menú de opciones que se despliega cuando pulsamos sobre el icono de WAMP5 situado en la barra de tareas en la siguiente imagen, figura 2.8.

Figura 2.8

Desde este menú podemos realizar las siguiente funciones:

- **Localhost**: accede a la raíz de nuestros ficheros.
- **phpMyAdmin**: accederemos al servidor de bases de datos a través de la aplicación phpMyAdmin.
- **www directory**: abre la carpeta con los archivos donde almacenamos nuestros ficheros.
- **Config files**: desde aquí podemos acceder a los tres ficheros de configuración: **httpd.conf**, **php.ini** y también **my(wamp).ini**.
- Por último, en la siguiente sección nos encontramos con cinco opciones desde las que podemos arrancar, parar y reiniciar los servidores Apache y MySQL.

2.5.2 AppServer

La otra opción de instalación se trata de AppServer.

No entraremos en los detalles de instalación, ya que prácticamente el proceso de instalación es idéntico al de WAMP, por lo que lo único que se pretende en este apartado es describir esta nueva aplicación para que el usuario decida cuál cree más conveniente emplear para sus aplicaciones.

Para descargar AppServer accedemos a la siguiente dirección web *http://www.appservnetwork.com* y una vez ahí, seleccionamos la opción **download**, que nos llevará a la web de SourceForge.net y desde donde podremos descargar la versión 2.5.10.

NOTA: es conveniente comprobar la versión actual del fichero a descargar, ya que durante la edición de este libro las versiones existentes eran las que se citan en el mismo, pero es muy probable que estas hayan sido actualizadas con el paso del tiempo, pero esto no varía en absoluto el funcionamiento de nuestras aplicaciones siguiendo los pasos de este libro.

Por último, indicar las versiones que contiene AppServer:

- Apache 2.2.8.
- PHP 5.2.6.
- MySQL 5.0.51b.
- PhpMyAdmin 2.10.3.

Capítulo 3

PRIMERAS PRUEBAS

A continuación, vamos a realizar las pruebas necesarias para comprobar que hemos instalado y configurado correctamente todo el *software* necesario para comenzar a trabajar.

Para ello, nos dirigimos a la barra de tareas de nuestro ordenador, en la cual encontraremos un icono como el que se muestra en la figura 3.1 , donde comprobaremos, pulsando sobre el mismo, que se encuentra funcionando; de no ser así, pulsaremos sobre **Start.**

Figura 3.1

Para comprobar que las páginas web creadas funcionan correctamente, haremos uso de la línea de código que PHP ofrece

para tal efecto, y que además permite ver los parámetros con los que tenemos configurados el intérprete de PHP.

Para ello, abriremos nuestro editor de páginas web y escribiremos las siguientes líneas de código:

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

Y guardaremos este archivo con el nombre **pruebaphp.php** en la carpeta **c:\ficheros** (que es la que le indicamos al configurar el servidor Apache para que se almacenen los archivos que vamos creando).

Luego iniciaremos nuestro navegador habitual, donde teclearemos la siguiente dirección web: **http://localhost**, que será la ruta para acceder desde nuestro navegador a los ficheros que tengamos almacenados en la carpeta configurada en el servidor web.

Tecleando esta ruta en el ordenador realizamos la misma operación que cuando navegamos por una página web en Internet, es decir, conectamos con un servidor que tiene almacenados una serie de ficheros o páginas web, pues en este caso es lo mismo, con la diferencia de que el servidor lo tenemos instalado en nuestro ordenador y nos va a servir para poder visualizar nuestras páginas web programadas con PHP.

Una vez hayamos conectado con nuestro servidor a través del navegador, aparecerá una pantalla con los ficheros que tenemos almacenados en el mismo.

En este caso, nos interesa pulsar sobre el fichero llamado **pruebaphp.php**, que es el que acabamos de crear, para ver el resultado de este ejemplo.

En la figura 3.2 podemos ver el resultado de conectar desde nuestro navegador con el servidor cuando está funcionando Apache.

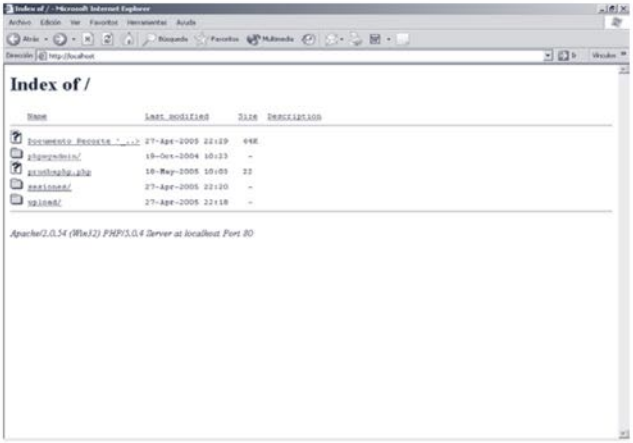

Figura 3.2

Para comprobar que todo se ha hecho correctamente y que realmente funciona, seleccionaremos el archivo creado anteriormente **pruebaphp.php** y se cargará una página con la configuración que tiene nuestro intérprete de PHP. Y nos cercioraremos que hemos configurado todo correctamente, por lo que podremos seguir creando páginas en PHP.

Al seleccionar el fichero **pruebaphp.php**, aparecerá la configuración de PHP. Entre otras cosas, podremos ver datos de nuestro sistema y del servidor que utilizamos, la configuración que hemos hecho de PHP, del servidor Apache y de MySQL, así como otra serie de funciones extras de PHP. Así mismo, al final del documento, nos encontraremos información acerca de la licencia de PHP, que como ya sabemos se trata de un *software* gratuito.

En la figura 3.3 podemos ver el resultado de seleccionar el fichero **pruebaphp.php**.

Figura 3.3

3.1 EMPEZANDO CON PHP

Entre otras muchas cuestiones que veremos más adelante, es importante saber escribir cuando nos referimos a una instrucción PHP.

A continuación se muestran varias opciones:

<?

Código PHP

?>

<%

Código PHP

En la figura 3.3 podemos ver el resultado de seleccionar el fichero **pruebaphp.php**.

Figura 3.3

3.1 EMPEZANDO CON PHP

Entre otras muchas cuestiones que veremos más adelante, es importante saber escribir cuando nos referimos a una instrucción PHP.

A continuación se muestran varias opciones:

<?

Código PHP

?>

<%

Código PHP

%>

<?php

Código PHP

?>

<script language="php">

Código PHP

En los ejemplos que se desarrollan en este libro vamos a emplear siempre el formato:

<?

Código PHP

?>

3.2 COMENTARIOS

Los comentarios en PHP, al igual que en cualquier otro lenguaje, son muy importantes, ya que ayudan a otras personas a comprender lo que hacemos con nuestras líneas de código, al igual que también nos ayudará a nosotros cuando programemos una página web y, pasado un tiempo, queramos hacer modificaciones.

Los comentarios, nos ayudarán a comprender de una forma sencilla cualquier parte del código.

Los comentarios que emplearemos pueden ser de cualquier tipo. Cada programador sigue unas pautas a la hora de realizar sus códigos, y los hay que no utilizan los comentarios.

Para personas que conozcan otros entornos de programación, como pueden ser C o C++, no les resultará nada complicado el aprender a hacer comentarios en PHP, ya que se hace exactamente igual que en esos mismos lenguajes de programación.

En primer lugar, podemos utilizar //, pero sólo nos servirá para hacer comentarios en una sola línea; si queremos utilizar varias líneas para realizar nuestros comentarios, utilizaremos para comenzar el comentario: /*, y para terminarlo: */.

3.3 EJEMPLO

```
<?
phpinfo ();// Con este ejemplo vemos la  configuración del  intérprete
// de PHP,  podemos comprobar qué parámetros tenemos activados
o
// desactivados y, si  se dispone  de algún conocimiento, podremos
// modificar alguno de estos parámetros al igual que hemos hecho
// anteriormente en los ficheros de configuración.
//Como podemos comprobar,esta forma de incluir comentarios en
este
// ejemplo no es la más correcta, ya que utilizamos varias líneas y  es
// menos eficaz al tener que escribirlo ocupando un mayor espacio.
?>
```

Lo más correcto, para realizar el ejemplo anterior sería realizarlo de este otro modo, ya que evitamos el tener que poner // cada vez que modifiquemos este comentario, y utilizando /* */ podemos modificar el texto del comentario sin tener que estar pendientes de agregar de nuevo el inicio del comentario con //.

```
<?
phpinfo ();
/* Con este ejemplo vemos la configuración del intérprete de PHP,
podemos comprobar qué parámetros tenemos activados o
desactivados y, si se dispone de algún conocimiento, podremos
```

modificar alguno de estos parámetros al igual que hemos hecho anteriormente en los ficheros de configuración.

*Como podemos comprobar, esta forma de incluir comentarios en este ejemplo es más correcta, ya que al utilizar varias líneas es más eficaz que el ejemplo anterior. */*

?>

Capítulo 4

VARIABLES Y CONSTANTES

La forma de representar las variables en PHP es anteponiendo el símbolo $ a la palabra que utilizaremos como variable, es decir, cuando en un código PHP veamos algo que comienza por $, como por ejemplo **$variable**, sabremos que en esa línea estamos definiendo una variable.

Debemos prestar especial atención a las mayúsculas y minúsculas, ya que PHP reconoce la diferencia entre las dos; no es lo mismo escribir, por ejemplo, $valor que escribir $VALOR, ya que el simple hecho de cambiar cualquier letra por una mayúscula hará que estemos hablando de variables diferentes. Para asignar las variables, utilizaremos el símbolo "=".

¿Para qué se utilizan las variables? Un lenguaje de programación no tendría sentido sin el uso de las variables, ya que son la base de todo entorno de programación. Gracias a ellas podremos realizar operaciones aritméticas, operaciones con cadenas, así como todo tipo de operaciones que a lo largo del libro veremos.

Las variables las utilizaremos cómo y cuándo queramos en una misma página, ya que no existe limitación en cuanto al uso de las mismas.

Otra cuestión muy importante referente al uso de las variables es que hay que prestar atención a no provocar errores a la hora de utilizar palabras reservadas por el lenguaje PHP y utilizarlas como variables, ya que existen unas variables por defecto que no se pueden alterar. Por ejemplo, nunca podremos utilizar en nuestros ejemplos para definir una variable algo así: **$os**, ya que **$os** es una variable predefinida y no podremos utilizarla para asignarle un valor que nosotros queramos. Más adelante, veremos algunas de estas palabras reservadas. También se deben evitar errores al empezar una variable por un número. Por ejemplo si escribimos **$25variable = "12"**, estaríamos cometiendo un error, ya que hemos comenzado a nombrar la variable con un número.

Más adelante, veremos cuáles son algunas de estas variables reservadas o predefinidas, a las que no podemos asignar ningún valor como si fueran cualquier otra variable y veremos cuál es la utilidad que se les puede dar.

En PHP no es necesario especificar el tipo de variable, pero sí debemos saber cuándo utilizar comillas, a la hora de definirlas, ya que dependiendo de si las utilizamos o no, estaremos hablando de una cadena o no.

Por ejemplo, para asignar el valor 5 a la variable cuenta, escribiremos: **$cuenta = 5**. Es decir, cuando queremos utilizar valores numéricos para asignar un valor a una variable, no es necesario entrecomillar el valor. Pero, por el contrario, si queremos asignar una palabra o una cadena a una variable, hay que entrecomillar el valor. Por ejemplo: **$nombre= "Pepe"**.

4.1 MOSTRANDO VARIABLES

Para mostrar las variables en pantalla, podemos hacerlo de varias maneras: una de ellas puede ser utilizando la opción **echo ( );** y otra puede ser **print ( );**. Más adelante y mediante ejemplos, veremos que estas dos instrucciones realizan exactamente la misma función al ser ejecutadas.

4.2 EJEMPLO

```
<head>

<title>Creación de un portal con PHP y MySQL</title>

</head>

<?

$a = 5; // Asignamos a la variable a el valor 5 (numérico).

$b = "7"; /* Asignamos a la variable b el valor 5 como cadena, es decir, la variable b no podrá ser utilizada para realizar operaciones aritméticas, ya que, aunque contenga el valor 5, lo reconocerá como una cadena de texto, no como un valor numérico.*/

echo "<b><h1>"; /* Con esta línea, hacemos que el texto que aparece en pantalla se muestre en negrita y con tamaño de letra grande. */

echo ($a); // Mostramos en pantalla el valor 5, que es la variable a.

echo "<br>" /* Realizamos un salto de línea para poder visualizar las dos variables por separado. */

echo ($b); // Mostramos en pantalla el valor 5, que es el valor de la variable b, pero la diferencia con la variable $a es que $b la podemos utilizar como una cadena.

echo "</b></h1>";

?>
```

Guardaremos este ejemplo en nuestra carpeta de ficheros con el nombre **ejemplo4-1.php**.

Al ejecutar este ejemplo en nuestro navegador, podremos ver que el resultado que obtenemos en pantalla son los valores 5 y 7, separados por un salto de línea. El salto de línea se ha puesto para separar las dos variables.

El resultado de ejecutar en nuestro navegador el fichero **ejemplo4-1.php** podemos verlo en la figura 4.1.

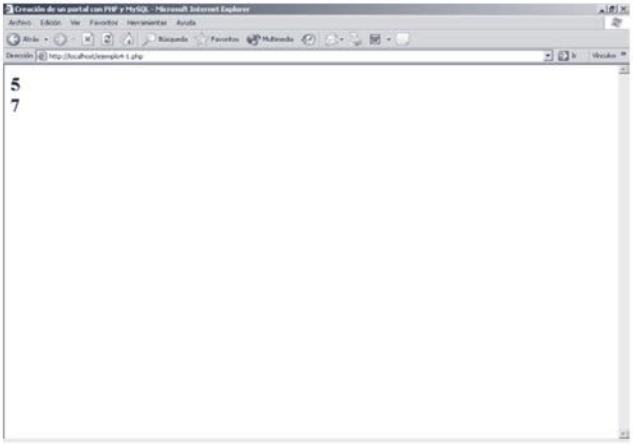

Figura 4.1

4.3 CONSTANTES

La primera y gran diferencia que existe con las variables y las constantes es que las últimas van a tener un valor fijo, es decir, su valor no se va a poder modificar durante la ejecución de una página. Por el contrario, una misma variable puede tomar varios valores en una misma ejecución.

Las forma de definir las constantes es mediante el uso de la instrucción **define.** Su sintaxis será: *define ("nombre_variable", "valor_variable").*

4.3.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
define ("capitalespana" , "Madrid");  /* capitalespana tendrá durante la ejecución de la página siempre el valor Madrid. */
define ("habitantes" , 4.000.000); /* habitantes tendrá durante la ejecución de la página siempre el valor 4.000.000. */
?>
```

Capítulo 5

OPERADORES

Los operadores son utilizados para realizar operaciones con variables y constantes. Podemos distinguirlos en cinco bloques diferentes: aritméticos, de comparación, lógicos, de unión de cadenas y de asignación. A continuación, vamos a ver cada uno de ellos con unos ejemplos para comprobar cuál es su funcionamiento.

5.1 OPERADORES ARITMÉTICOS

Dentro del bloque de los operadores aritméticos podemos distinguir siete clases diferentes que se muestran en la siguiente tabla, donde, además, hemos asignado a las variables $x y $z los valores 8 y 4, respectivamente, para poder ver el resultado que se obtiene al realizar las operaciones aritméticas con estos operadores aritméticos.

Operador	Operación	Sintaxis	Resultado
+	Suma	$x + $z	12
-	Resta	$x - $z	4
*	Multiplicación	$x * $z	32
/	División	$x / $z	2
%	Módulo (resto de la división)	$x % $z	0
++	Incremento (incrementa en 1)	$x++	9
--	Decremento (decrementa en 1)	$z--	3

5.1.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
$x = 9;
$y = 3;
$z = 5;
$suma = $x + $y;
$resta = $x - $ z;
$multiplicacion = $suma * $resta;
$final = $multiplicación ++;
```

echo $multiplicacion; / Si analizamos, paso a paso, las operaciones que realizamos, hemos de llegar al resultado de: 49. Paso a paso: (9 + 3) * (9 – 5) = 48 ++ si lo incrementamos en 1 = 49, que es el resultado que se mostrará al ejecutar la página. */*

?>

5.2 OPERADORES DE COMPARACIÓN

Los operadores de comparación se utilizan para comprobar el resultado de una operación. El resultado que obtenemos de estas operaciones será *True*, en caso de ser verdadero, y *False*, en caso de ser falso.

Dentro de los operadores de comparación podemos distinguir siete operadores. Para realizar la siguiente tabla, asignamos valores a las variables: $x = 6 y $z = 4.

Operador	Operación	Sintaxis	Resultado
==	Igual ($x y $z tienen el mismo valor)	$x == $z	*False*
===	Idéntico ($x y $z tienen el mismo valor y además son del mismo tipo)	$x === $z	*False*
!=	Diferente ($x y $z son de diferente valor)	$x != $z	*True*
<	Menor ($x menor que $z)	$x < $z	*False*
>	Mayor ($x mayor que $z)	$x > $z	*True*
<=	Menor o igual ($x menor o igual que $z)	$x <= $z	*False*
>=	Mayor o igual ($x mayor o igual que $z)	$x >= $z	*True*

5.2.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
$x = 5;
$y = 4;
echo ($x == $z);  /* Nos mostrará en pantalla el valor False, ya que las variables x e y no son iguales.*/
echo ($x >= $z);  /* Nos mostrará en pantalla el valor True, ya que la variable x, como podemos comprobar, es mayor que y.*/
?>
```

5.3 OPERADORES LÓGICOS

Los operadores lógicos son utilizados para combinar varias condiciones y para que las diferentes condiciones puedan ser evaluadas con una sola expresión.

Podemos distinguir seis operadores lógicos diferentes que podemos ver en la siguiente tabla:

Operad.	Operación	Sintaxis	Resultado
&&	Y ($a y $b)	$a && $b	*True* (si $a y $b son verdaderos)
AND	Y ($a y $b)	$a AND $b	*True* (si $a y $b son verdaderos)
||	O ($a o $b)	$a || $b	*True* (si $a o $b son verdaderos)

OR	O ($a o $b)	$a OR $b	*True* (si $a o $b son verdaderos)
XOR	O exclusiva ($a o (exclusiva) $b)	$a XOR $b	*True* (si $a es verdadero o $b es verdadero, pero no los dos)
!	Negación	!$a	*True* (si $a no es verdadero)

Como se puede observar en la tabla anterior, las operaciones && y AND equivalen a la misma operación, y lo mismo ocurre con || y OR. Son dos sintaxis diferentes, pero con un mismo resultado final.

5.3.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
$x = 4;
$y = 5;
if (($x==4) && ($y==5))
{
print ("Estás en lo correcto");
}
echo '<br>';
if (($x==4) OR ($y==3))
{
print ("La segunda operación también es correcta");
}
?>
```

Al probar este ejemplo en nuestro navegador, obtendremos en pantalla el mensaje "Estás en lo correcto" y otro que dirá "La segunda operación también es correcta", y nos podemos preguntar ¿por qué se imprimen los dos mensajes? En el primer caso, si la variable *x* es igual a 4 y la variable *y* es igual a 5, muestra el mensaje "Estás en lo correcto" y en la segunda operación, se imprime el mensaje "La operación también es correcta", ya que si *x* es igual a 4 o *y* es igual a 3, lo imprime.

5.4 OPERADORES DE UNIÓN DE CADENAS

Este operador, como bien dice su nombre, se encarga de unir cadenas.

Para unir cadenas, es necesario al menos disponer de dos variables para que se produzca la unión de las mismas, o también una misma variable la podemos utilizar en varias ocasiones para la unión de cadenas.. Como se puede ver a continuación en el siguiente ejemplo, tenemos cinco variables con las que hacemos una unión de cadenas. Para la unión de cadenas mediante variables se emplea el punto (.).

5.4.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<?
$t = 'Ejemplo';
```

```
$w = 'unión';
$x = 'de';
$y = 'cadenas';
$z = ' ';
$resultado1 = $t;
$resultado2 = $t . $z . $x . $z . $w . $z . $x . $z . $y;
echo '<b><h1>';
echo $resultado1; // Insertamos una cabecera.
echo '<hr>';
/* Con esta línea insertamos una línea horizontal, que puede ser utilizada para dividir textos o imágenes en nuestras páginas web. En este caso, lo utilizamos para dividir un titular de la web, con el ejemplo de unión de cadenas.*/
echo $resultado2;
/* Mostramos en pantalla el resultado de la variable $resultado que, como podemos ver, contiene la frase: "Ejemplo de unión de cadenas".*/
echo '</b></h1>';
?>
```

En este ejemplo, hemos realizado la unión de varias palabras mediante la operación de unión de cadenas. También podemos observar cómo una misma variable puede ser empleada tantas veces como se desee en una misma página, es decir, la palabra ejemplo y el carácter espacio, que en realidad son dos variables, son utilizadas más de una vez en la misma página.

En la siguiente imagen, figura 5.1, podemos ver el ejemplo de ejecutar este fichero en nuestro navegador.


Archivo Edición Ver Favoritos Herramientas Ayuda
Ejemplo
Ejemplo de unión de cadenas

Figura 5.1

Capítulo 6

ESTRUCTURAS DE CONTROL

Las estructuras de control son instrucciones utilizadas en programación para llevar a cabo una serie de acciones en las aplicaciones que vamos creando.

6.1 INSTRUCCIONES CONDICIONALES

Con este tipo de instrucciones lo que hacemos es ejecutar una parte de código si se cumple una determinada condición.

Tenemos varias instrucciones de condición:

Instrucción *If*. Esta instrucción se utiliza para hacer preguntas. Si la pregunta se cumple en la condición, se ejecutará el código que contiene.

Poniendo un ejemplo, en nuestro lenguaje es como si dijéramos: Si *tengo más de 18 años*, soy mayor de edad. Es decir, si se cumple la condición, será que eres mayor de edad, de lo contrario no podrás ser mayor de edad.

Instrucción *else* y *else if*. Estas dos instrucciones se utilizan cuando el resultado obtenido es falso tras un *if*. Por ejemplo, como en el caso anterior, si dijéramos: Si *tengo más de 18 años*, soy mayor de edad. Si esta condición fuera falsa, es decir, que por ejemplo tuviéramos 15 años, no se mostraría nunca el texto *soy mayor de edad*, para lo que podemos darle otra condición posterior y mostrar un texto en caso de ser falso. Por ejemplo, podemos decir: Si *tengo más de 18 años*, soy mayor de edad. Si no, *soy menor de edad.*

La forma de referirnos a las instrucciones condicionales será poniendo entre paréntesis la condición y cerrando con corchetes la parte de código que queremos que se ejecute si se cumple la condición, como a continuación se muestra:

```
<?
If (condición) {
Hacer esto
Y esto
.....
.....
.....
Tantas como queramos
}
?>
```

6.1.1 Ejemplo 1

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
```

```
$color = "rojo";  // Asignamos a la variable color el valor rojo.
if ($color = "rojo")
{
/* Le preguntamos si la variable color es igual a rojo y efectivamente así es, por lo que se ejecuta la siguiente parte de código que está dentro de la condición.*/
print ("Efectivamente el color es rojo");  /* Como se cumple la condición, se mostrará este mensaje en pantalla.*/
}
?>
```

En la siguiente imagen, figura 6.1, podemos ver el resultado del ejemplo anterior en nuestro navegador.

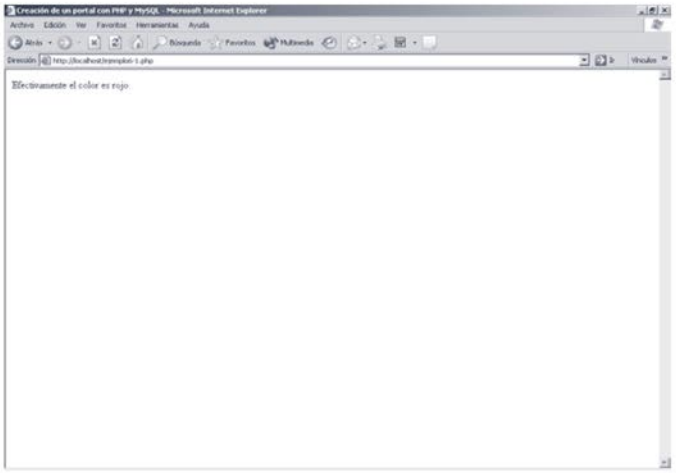

Figura 6.1

6.1.2 Ejemplo 2

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
$x = 10;
$y = 15;
if ($x = $y)
{ // Si x es igual que y, muestra el siguiente mensaje.
print ("X e Y son iguales");     /* Este mensaje no se muestra, ya que x e y no son iguales. */
}
elseif ($x > $y)
{ /* Si x es mayor que y, muestra el siguiente mensaje.*/
print ("X es mayor que Y");     /* Este mensaje no se muestra, ya que x no es mayor que y. */
}
elseif ($x < $y)
{ /* Si x es menor que y, se muestra el siguiente mensaje. */
print ("X es menor que Y");     /* Este mensaje sí se mostrará en pantalla, ya que x es menor que y. */
}
?>
```

6.2 INSTRUCCIONES DE BUCLE

Las instrucciones de bucle son utilizadas para ejecutar un determinado número de veces un código o cuando se cumple una condición.

Tenemos varias instrucciones de bucle:

Instrucción *while*: indica que mientras no se cumpla una determinada condición, no se saldrá del bucle y no saltará a la siguiente línea de código.

Por ejemplo, nosotros diríamos: *mientras tu edad no sea 18 años, no serás mayor de edad.*

Otra de las instrucciones de bucle es *do...while*, que lo que hace es ejecutar una parte de código mientras que no se cumpla una condición.

Por ejemplo, en nuestro lenguaje podríamos decir: *serás un menor, mientras no tengas más de 18 años.*

Por último, la instrucción *for* será la utilizada para ejecutar un bucle un determinado número de veces (hasta que se cumpla una condición). Esta instrucción está formada por tres partes: la primera que será donde inicializaremos la variable; la segunda, donde se establece la condición que queremos que se cumpla, y una última, donde iremos modificando el valor de la variable.

6.2.1 Ejemplo 1

```
<head>

<title>Creación de un portal con PHP y MySQL</title>

</head>

<?
```

```
$x = 10;

while (--$x)

{

// Decrementa en 1 la variable x.

echo "<big>"; /* Con esta línea vamos incrementando el tamaño de la fuente.*/

echo "<b>";

print ("Número: " . $x);

// Mostrará en pantalla "Número ..." desde el 9 hasta el 1.

echo "<br>";

echo "<hr>";

}

?>
```

Con esta pequeña aplicación lo que mostraremos en pantalla es:

Número 9

....

Número 1

Podemos ver este ejemplo en la figura 6.2:

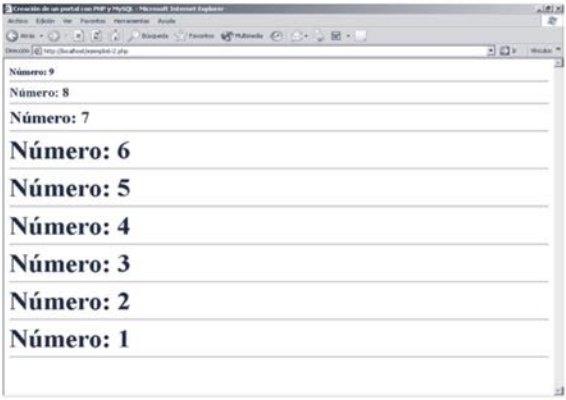

Figura 6.2

6.2.2 Ejemplo 2

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<?
for ($x = 5; $x = 10;  $x++)
{
/* Inicializamos la variable x a 5 y le decimos que hasta que no llegue a 10 no salga del bucle, para que cada vez que hace el bucle, lo incrementemos en 1. Es decir, incrementamos x en 1, desde su valor inicial 5, hasta que llegue a tomar el  valor 10.*/
print ("Número: " .$x . "<br>");  /* Cada vez que haga el bucle escribirá en pantalla:
```

```
Número: 5
Número: 6
.....
Hasta llegar al 9.*/
}
?>
```

6.3 OTRAS INSTRUCCIONES

Distinguiremos entre otras tres instrucciones, *require ( ), include ( ) y switch.*

La instrucción *require ( )* sirve para incluir ficheros en nuestras páginas, siendo sólo necesario hacer referencia a este fichero con la instrucción *require ( )*. Por ejemplo, podemos poner lo siguiente: require ("texto.php"); y lo que hará esta línea de código cuando se ejecute en nuestra página web será solicitar al fichero la información que contiene el fichero **texto.php**.

Su uso principal es para definir variables, y estas estarán listas, una vez hagamos una llamada al fichero donde estén guardadas. En el ejemplo anterior, en el fichero **texto.php,** podremos tener almacenadas una serie de variables que podrán ser utilizadas a lo largo de la página, ya que, con hacer la llamada a este fichero, las variables que contiene el mismo pasan a formar parte de nuestra página para ser utilizadas en cualquier momento.

La principal ventaja de esta instrucción es que con una sola línea de código podemos estar utilizando infinitas variables en multitud de ficheros a la vez, con el consabido ahorro de líneas que esto nos puede suponer.

Tiene una desventaja, y es que no se puede utilizar en un bucle para llamar a diferentes ficheros.

La instrucción *include ( )* su funcionamiento es igual que el de la instrucción *require ( )*, pero con la diferencia de que sí puede procesar el código tantas veces como llamemos a esa página externa. Su sintaxis será igual que la de *require ( )*. Por ejemplo: include ("texto.php").

Por último, la instrucción *switch* se utiliza para comprobar un dato entre varias posibilidades.

6.3.1 Ejemplo 1

```
<?
$x = "Incluyendo";
$y = "ficheros";
$z = " ";
?>
```

Este código lo podemos guardar en un fichero con el nombre **variables.php**. Y, a continuación, lo incluiremos en la siguiente página que vamos a crear.

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
include ("variables.php");
/* Utilizamos el fichero variables.php, que es el que anteriormente hemos creado. */
echo "<br>";
echo "<br>";
print ("$x" . "$z" . "$y");
```

/ Mostramos en pantalla el texto "Incluyendo ficheros", que es el resultado de unir las variables que hemos definido en el fichero "variables.php". */*

?>

Podemos ver el resultado de ejecutar este ejemplo en la siguiente imagen, figura 6.3.

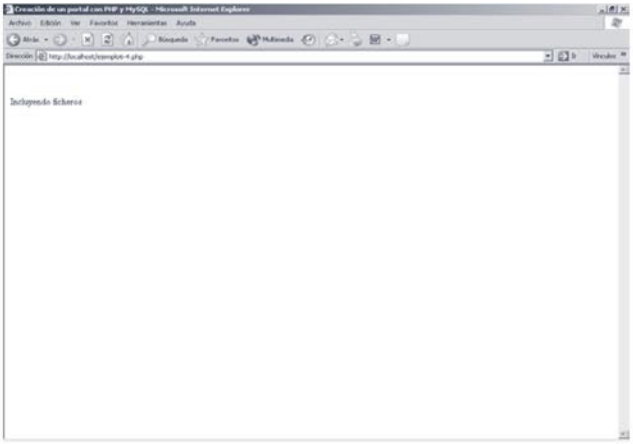

Figura 6.3

Como podemos comprobar, la principal ventaja de utilizar estas instrucciones es ahorrar líneas de código, ya que el fichero **variables.php** lo podemos utilizar en las páginas que queramos, con el correspondiente ahorro de líneas de código en cada una de ellas.

Además, no siempre tenemos por qué utilizar todas las variables que contenga el fichero **variables.php**, ya que, por ejemplo, además de todas las variables que hemos definido anteriormente, podemos tener otras cinco variables, que serán utilizadas en otros ficheros que no tendrán nada que ver con el

anteriormente creado. Por ejemplo, podemos tener el fichero **variables.php** de la siguiente forma:

```
<?
$x = "Incluyendo";
$y = "ficheros";
$z = " ";
$a = 3;
$b = 7;
$c = 5;
$d = 12;
$e = 9;
?>
```

Este fichero podrá llamarse igualmente **variables.php**, sustituyendo al anterior y no hay ningún problema en que haya otras variables que no vamos a utilizar en el ejemplo que hayamos creado anteriormente.

6.3.2 Ejemplo 2

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
$color = "negro";
switch ($color)
        {
        case "blanco":
```

```
            $sector = "claro";
            break;
        case "naranja":
            $sector = "normal";
            break;
        case "negro":
            $sector = "oscuro";
            break;
        }
print ($sector);
/* Mostrará en pantalla el valor oscuro, que es el que se
corresponde con la variable $color = "negro".*/
?>
```

Lo que hacemos con este código es que, con una variable que definimos, la comprobamos con algunos valores mediante la instrucción *switch* (las comprueba una por una, independientemente de que la primera o cualquier otra sean las válidas). Una vez que ya ha terminado de analizar cada caso, le diremos que nos muestre cuál es la correcta.

Capítulo 7

FUNCIONES

Una función es un bloque de código que introducimos en nuestra página y que puede ser utilizado a lo largo de todo nuestro código PHP. La principal ventaja de las funciones es que nos permiten ahorrar código.

7.1 FUNCIONAMIENTO

La sintaxis para definir funciones es mediante la sentencia *function*. Por ejemplo, para definir una función escribiríamos *function suma ($x)*.

Las funciones pueden recibir tantos argumentos como sean necesarios, separándolos con comas.

7.1.1 Ejemplo 1

<head>

<title>Creación de un portal con PHP y MySQL</title>

```
</head>
<?
echo "<h1>";
function suma ($x, $y)
{
$z = $x + $y;
return $z;
}
$resultado = suma (5,12);
/* Vamos a utilizar la función suma, asignando a las variables x e y
los valores 5 y 12 respectivamente.*/
echo "<br>";
echo $resultado;
// Nos devuelve el resultado, en este caso, 17.
echo "</h1>";
?>
```

7.1.2 Ejemplo 2

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
function suma ($suma)
{
// Creamos la función, en este caso, la función suma.
```

```
return $suma + $suma;
// Devuelve el resultado de la suma.
}
print ("Suma: " . suma (5) );
/* Sacará en pantalla el resultado de utilizar la función con la
variable $suma tomando el valor 5.*/
?>
```

El resultado de ejecutar el ejemplo del apartado 7.1.1. podemos verlo en la siguiente imagen, figura 7.1.

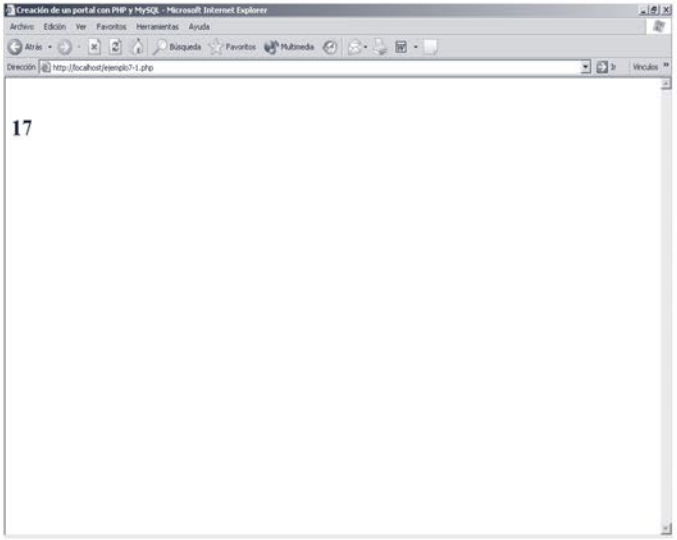

Figura 7.1

7.2 ALCANCE DE LAS VARIABLES

Cuando hablamos del alcance de las variables, nos referimos a qué partes del código podemos acceder de las mismas. Al hablar de este nuevo término, tenemos dos nuevos conceptos, como son las variables globales y las variables locales.

Las variables globales son aquellas que tienen un mismo valor durante toda la ejecución de una página web, pero podemos encontrarnos con una variable con el mismo nombre definida dentro de una función. Esta será una variable local, y su valor sólo será valido mientras ejecutemos la función; fuera de la función será válido el valor de la variable global.

Mediante ejemplos entenderemos mejor su funcionamiento.

7.2.1 Ejemplo 1

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
$var = 5;
function suma ( )
{
$var = 12;
}
suma ( );
echo "<br>";
echo "<br>";
print ($var);
```

```
/* En este caso concreto se mostrará en pantalla el valor 5, ya que
en este caso la variable $var es global, porque al modificar la
variable $var con valor 12, sólo afecta a la función.
*/
?>
```

7.2.2 Ejemplo 2

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
$var = 5;
function suma ( )
{
$var = 12;
print ($var);
/* Por el contrario, en este ejemplo, se mostrará en pantalla el valor
12, ya que estamos mostrando la variable como variable local y
estamos diciendo que nos muestre el contenido de la variable dentro
de la función.*/
}
?>
```

Capítulo 8

FUNCIONES PARA MANIPULACIÓN DE CADENAS

A lo largo de este capítulo explicaremos una serie de funciones de PHP que están especialmente destinadas a manipular cadenas de texto.

En este capítulo explicaremos ocho de estas funciones, aunque existen otras muchas, ya que nos serán de gran utilidad a la hora de manipular cadenas de texto.

8.1 FUNCIÓN SUBSTR ( )

Para el uso de esta función utilizaremos obligatoriamente, al menos dos argumentos: el primero será la cadena de texto que vayamos a tratar y el siguiente será el que nos indique la posición a partir de la cual nos devolverá la cadena de texto que estemos tratando.

En definitiva, con esta función lo que conseguimos es mostrar la parte que nos interesa de una cadena de texto.

8.1.1 Ejemplo

```
<head>

<title>Creación de un portal con PHP y MySQL</title>

</head>

<body bgcolor = "#303030">

<?

/* Color de fondo de la página; en este caso, este sería un gris oscuro.*/

 ?>

<body text = "#A0A0A0">

<?

/* Color de la fuente; en este caso, el color sería un plata.*/

?>

<body leftmargin = "200">

<?

/* Distancia del eje horizontal a partir de la cual se muestra el texto o imágenes (en este caso, a partir de 200 píxeles hacia la derecha).*/

?>

<body topmargin = "200">

<?

/* Distancia del eje vertical a partir del cual se muestra el texto o imágenes (en este caso, a partir de 200 píxeles hacia abajo).*/
```

```
?>

<font face = "Tahoma">

<?

/* Tipo de fuente que utilizamos; en este caso seleccionamos
Tahoma.*/ ?>

<font size = "3">

<?

// Tamaño de la fuente.

?>

<hr size = "9" color = "FFFFFF" width = "30%" align = "left">

<?

/* Con esta última línea insertamos una línea horizontal, de un
tamaño vertical (size) de 9 pixel, en color blanco (FFFFFF), un
tamaño horizontal del 30% (width) y alineada a la izquierda
(align="left").*/

/* Con todas estas líneas hemos empezado a dar formato a nuestras
páginas. Como podemos ver, mejoramos la vistosidad de las mismas,
añadiendo pequeños detalles, como pueda ser cambiar el color de
fondo de la página web, el color de la fuente, el tamaño y el tipo.*/

print (substr ("Bienvenido al Portal de Coches", 14));

print ("<br>");

print (substr ("Bienvenido al Portal de Coches", -6));

?>

<hr size = "9" color = "ffffff" width = "30%" align = "left">
```

El resultado de este ejemplo se puede ver en la figura 8.1.

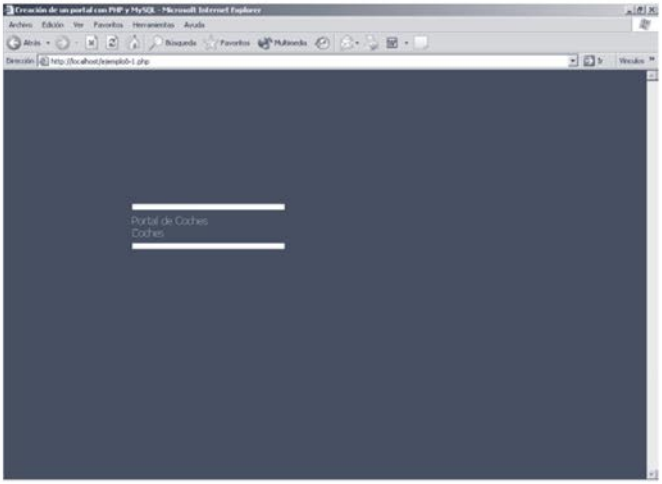

Figura 8.1

Con este ejemplo que hemos creado conseguimos mostrar en pantalla dos mensajes. El primero de ellos muestra el resultado: *Portal de Coches*, porque le hemos indicado que nos empiece a mostrar a partir del carácter número 14 de la cadena de texto. Y el siguiente mensaje de texto que aparece es *Coches*, porque al indicarle el número 6 con un "–" delante, hacemos que empiece a descontar al contrario, desde el lado derecho de la cadena de texto.

8.2 FUNCIÓN ORD ( )

La función ord ( ) tiene un cometido muy peculiar, ya que su misión es convertir código ASCII a caracteres.

Es decir, le indicamos un carácter en código ASCII y nos mostrará el correspondiente carácter. Esto puede ser muy útil a la

hora de utilizar la unión de cadenas, y si, por ejemplo, queremos tabular un texto que se muestra en pantalla, guardaremos en una variable el correspondiente carácter ASCII y sólo tendremos que hacer una llamada a esa variable cada vez que queramos tabular un texto.

8.2.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#A0A0A0">
<body leftmargin = "20">
<body topmargin = "20">
<font face = "Tahoma">
<font size = "3">
<hr size = "9" color = "FFFFFF" width = "40%" align = "left">
<?
for ($i=1; $i<=255; $i++)
{
/* Incrementamos la variable $i desde 1 hasta 255, para poder representar todos los caracteres. */
print ("ASCII: ".$i." -< caracter: ". chr($i). "<br>");
/* Mostramos en pantalla cada código ASCII con su correspondiente carácter. */
}
?>
```

Con este ejemplo, lo que hacemos es crear un listado con los 255 códigos ASCII y sus correspondientes caracteres.

El resultado de ejecutar este ejemplo en el navegador se puede ver en la figura 8.2.

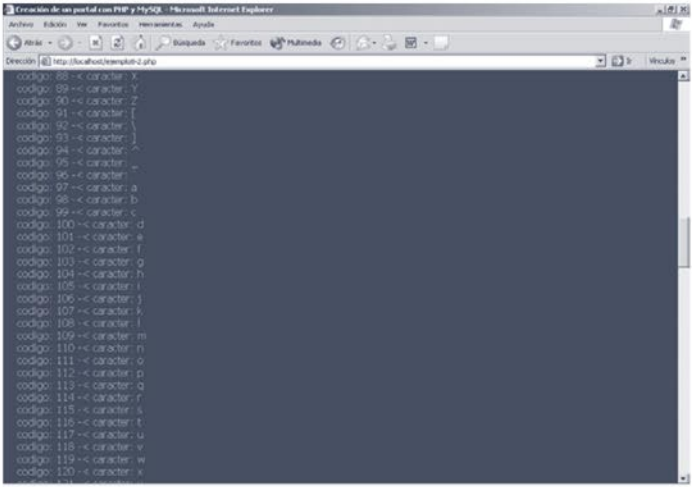

Figura 8.2

8.3 FUNCIONES PRINTF ( ) Y SPRINTF ( )

Estas dos instrucciones tienen la misma función: construir cadenas de texto en función de las instrucciones que se muestran en la siguiente tabla. Para utilizar esta tabla mostramos la instrucción *printf* o *sprintf* y la variable a mostrar precedida de la letra correspondiente a la función que queremos que realice.

Instrucción	Acción
b	Devuelve entero binario
d	Devuelve entero decimal
c	Devuelve carácter ASCII correspondiente
f	Devuelve decimal (utiliza signo decimal)
s	Devuelve cadena

La única diferencia entre estas dos funciones es que *sprintf* () no mostrará nunca el resultado, sino que lo podremos almacenar en una variable para más adelante utilizarla.

Estas dos instrucciones irán siempre acompañadas del signo "%", que será necesario utilizar en su sintaxis, es decir, su sintaxis puede ser algo así: *printf* ("%b", $variable).

8.3.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#A0A0A0">
<body leftmargin = "20">
<body topmargin = "20">
<font face = "Tahoma">
<font size = "3">
<hr size = "9" color = "FFFFFF" width = "40%" align = "left">
<?
$edad = "25 años";
```

```
printf ("%d", $edad);
?>
```

En el ejemplo anterior, mostraremos en pantalla el valor *25 (se omite la palabra años)*, ya que al utilizar la función *printf* junto con la instrucción *%d*, extraemos de la variable $edad el valor decimal de esa variable. Podemos ver el resultado de ejecutar este ejemplo en la siguiente imagen, figura 8.3.

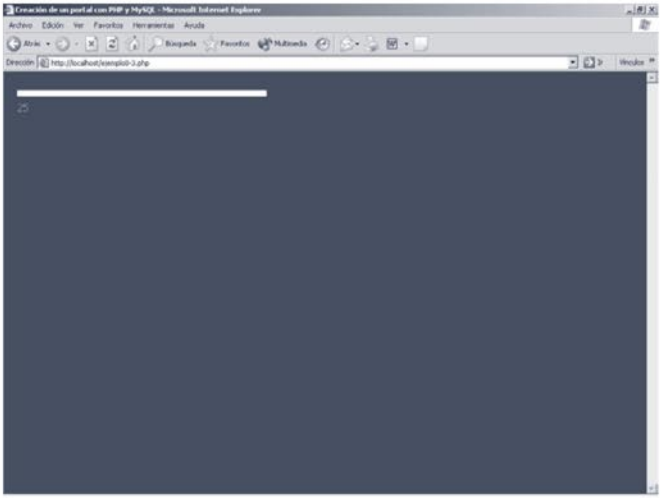

Figura 8.3

8.4 FUNCIONES STRTOLOWER ( ) Y STRTOUPPER ( )

Estas dos funciones tienen la misión de convertir en mayúsculas [strtoupper ( )] o en minúsculas [strtolower ( )], una cadena de texto.

Su utilidad está muy definida si, por ejemplo, queremos destacar un titular o, simplemente, convertir todo el texto o parte de una página web. Con estas dos instrucciones convertiremos el texto a minúsculas o mayúsculas, según nuestras necesidades en cada caso.

8.4.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#A0A0A0">
<body leftmargin = "200">
<body topmargin = "200">
<font face = "Tahoma">
<font size = "3">
<hr size = "9" color = "FFFFFF" width = "40%" align = "left">
<?
$var = "Pepito";
print "Texto en minúsculas:";
echo "<br>";
echo ( strtolower ($var));
/* Esta función mostrará en pantalla el resultado siguiente: pepito.
Es decir, nos ha convertido toda la cadena Pepito a minúscula.*/
echo "<br>";
echo "<br>";
print "Texto en mayúsculas:";
echo "<br>";
```

```
echo (strtoupper ($var));

/* Esta función mostrará en pantalla el resultado siguiente:
PEPITO. Es decir, nos ha convertido toda la cadena de texto Pepito
a caracteres en mayúscula.*/

?>

<hr size = "9" color = "FFFFFF" width = "40%" align = "left">
```

Con el ejemplo anterior lo que hacemos es convertir una variable a minúsculas y mayúsculas.

En la siguiente imagen, figura 8.4, podemos ver el resultado de ejecutar este ejemplo.

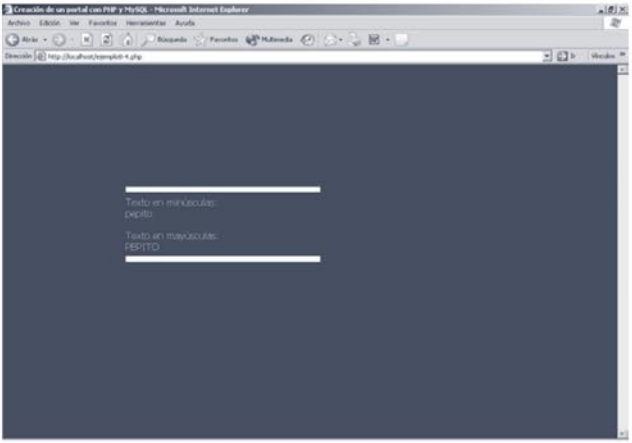

Figura 8.4

8.5 FUNCIONES EREG ( ) Y EREGI ( )

El uso de estas dos funciones es muy interesante, en especial para el uso en formularios, ya que muchas veces el usuario no

completa bien los campos de un formulario a la hora de introducir el e-mail, porque no pone bien el símbolo "@", aunque se nos pueden ocurrir infinidad de ideas a las que aplicar estas dos funciones.

La diferencia entre estas dos funciones es que *eregi ( )* no diferencia entre mayúsculas o minúsculas, algo que a la hora de rellenar un formulario nos puede ser indiferente, ya que lo que nos interesa es recibir un texto legible.

8.5.1 Ejemplo

A continuación, vamos a crear un ejemplo en el que supuestamente tenemos una variable que ha tomado el campo e-mail de un formulario y vamos a comprobar que este es correcto ¿Cómo? Pues como anteriormente indicábamos, comprobando que se ha escrito correctamente el carácter "@". Este método no será del todo fiable, ya que aun así se pueden seguir equivocando a la hora de escribir el e-mail, pero sí es un paso importante a la hora de evitar un fallo muy común, como es el de escribir mal el símbolo "@".

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
$mail = "pepe@dominio.com";
/* En este caso, hemos puesto una variable con el contenido
pepe@dominio.com, para comprobar el funcionamiento de la
función eregi ( ), pero lo normal es que el contenido de la variable
$mail lo obtengamos de un formulario.*/
if ( eregi ( "@", $mail))
{
print ("Los datos insertados en el formulario son correctos");
```

```
/* Si no se encontrase el carácter @ en la variables $mail, se
mostraría este mensaje, es decir, que es correcta la dirección de e-
mail.*/

}

else {

print ("La dirección de e-mail no es correcta, por favor vuelva a
introducirla");

/* Si no se encuentra el carácter @ en la variables $mail, se
mostrará este mensaje, es decir, que la dirección de correo que se ha
introducido no es correcta.*/

}

?>

/* En este caso, evidentemente se mostrará el primer mensaje, es
decir: "Los datos insertados en el formulario son correctos", porque
sabemos el valor de la variable $mail y sabemos que contiene el
carácter @. Pero cuando diseñemos un formulario para que inserten
sus datos los usuarios, puede ser que se equivoquen y no lo
introduzcan bien, por lo que el segundo mensaje les advertirá del
error al introducir el e-mail. Para ver que este ejemplo funciona con
una dirección de correo electrónico incorrecta, podemos probar
también este ejemplo, poniendo como valor a la variable $mail, por
ejemplo, $mail=pepemidominio.com, y veremos que se nos muestra
el mensaje de error en la dirección de correo electrónico
introducida.*/
```

Capítulo 9

MANEJO DE FICHEROS

PHP ofrece una extensa gama de funciones para acceso a ficheros, funciones que se pueden utilizar para abrir, guardar, leer, modificar ficheros, etc. Mención especial merece la función de subir ficheros a un servidor, por ejemplo, puede ser subir una fotografía a una página web, para poder utilizarla en un foro o, por ejemplo, si tenemos una página web que se dedica a promocionar personas para una agencia de modelos, esta será muy buena opción para poder recibir las fotografías de nuestros usuarios.

A continuación, vamos a ver una serie de instrucciones utilizadas en PHP para el manejo de ficheros, así como una tabla en la que se explica cuál es el funcionamiento, según el valor que tome la instrucción.

Función **fopen ( )**: se utiliza para abrir ficheros. Debemos prestar atención a la siguiente tabla a la hora de abrir ficheros, ya que, según el valor que elijamos, se dará una serie de permisos u otros a la hora de trabajar con ficheros.

Función **fclose ( )**: cuando hemos utilizado un archivo, debemos terminar cerrándolo, para lo que utilizaremos esta función.

Valor	Descripción
a	Abre el fichero sólo para añadir datos. Si el fichero no existe, se creará.
a+	Abre el fichero para añadir y leer datos. Si el fichero no existe, se creará.
r	Abre el fichero sólo para lectura.
r+	Abre el fichero para lectura y escritura.
w	Abre el fichero para escritura. Si el fichero no existe, se creará.
w+	Abre el fichero para escritura y lectura. Si el fichero no existe, se creará.

Función **fread ( )**: se utiliza para mostrar sólo determinadas partes de un fichero. Esta función necesitará, además, un segundo argumento, que determinará la cantidad de caracteres que se desea leer del fichero al que estamos accediendo.

Función **fwrite ( )**: es utilizada para escribir ficheros. Primero hay que abrirlo, según se explicó anteriormente, prestando atención a que para escribir debemos utilizar permiso de escritura (es decir, los que llevan el símbolo +).

9.1 DIRECTORIOS

Como hemos visto anteriormente con los ficheros, también podremos trabajar con directorios. Una de las principales funciones para el manejo de directorios es la instrucción **chdir ( )**, que determinará el nuevo directorio que queramos especificar para trabajar (en nuestro caso, el directorio con el que trabajamos por defecto es el que determinamos al haber configurado inicialmente PHP y el servidor Apache, en el cual guardamos las páginas que vamos creando).

Funciones **mkdir ( )** y **rmdir ( )**: son funciones utilizadas para crear y para borrar directorios, respectivamente. El único parámetro que añadiremos a estas funciones es el de especificar el nombre del directorio que queramos crear o borrar.

9.1.1 Ejemplo 1

En este ejemplo vamos a abrir un fichero sólo para su lectura, que se encuentra en nuestro disco duro (hablamos de nuestro disco duro, ya que debemos recordar que por el momento estamos trabajando en modo local). Para este ejemplo, hemos creado un fichero llamado **texto.txt** y lo hemos guardado en el directorio **c:/ficheros/**, es decir, donde guardamos los ficheros PHP.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<?
$abre = "c:/ficheros/texto.txt";
/* Determinamos el fichero y la ruta donde se encuentra, en este caso el fichero es texto.txt y se encuentra en c:/ficheros/.*/
if (fopen ($abre, r))
```

9.1 DIRECTORIOS

Como hemos visto anteriormente con los ficheros, también podremos trabajar con directorios. Una de las principales funciones para el manejo de directorios es la instrucción **chdir ( )**, que determinará el nuevo directorio que queramos especificar para trabajar (en nuestro caso, el directorio con el que trabajamos por defecto es el que determinamos al haber configurado inicialmente PHP y el servidor Apache, en el cual guardamos las páginas que vamos creando).

Funciones **mkdir ( )** y **rmdir ( )**: son funciones utilizadas para crear y para borrar directorios, respectivamente. El único parámetro que añadiremos a estas funciones es el de especificar el nombre del directorio que queramos crear o borrar.

9.1.1 Ejemplo 1

En este ejemplo vamos a abrir un fichero sólo para su lectura, que se encuentra en nuestro disco duro (hablamos de nuestro disco duro, ya que debemos recordar que por el momento estamos trabajando en modo local). Para este ejemplo, hemos creado un fichero llamado **texto.txt** y lo hemos guardado en el directorio **c:/ficheros/**, es decir, donde guardamos los ficheros PHP.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<?
$abre = "c:/ficheros/texto.txt";
/* Determinamos el fichero y la ruta donde se encuentra, en este caso el fichero es texto.txt y se encuentra en c:/ficheros/.*/
if (fopen ($abre, r))
```

```
{
/* Abrimos ese fichero en modo lectura, para lo que utilizamos el parámetro "r" .*/
print ("El fichero se ha abierto.");
/* Si lo encuentra y lo puede abrir, se muestra este mensaje.*/
}
else
{
print ("El fichero no se encuentra.");
/* Si no lo encuentra o no lo puede abrir, se muestra este otro mensaje. */
}
?>
```

9.1.2 Ejemplo 2

En este ejemplo vamos a crear un directorio en nuestro disco duro.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<?
$nuevodirectorio = "nuevo";
mkdir ($nuevodirectorio);
/* Nuevo será el nombre del directorio que hemos creado con la instrucción mkdir.*/
?>
```

9.2 SUBIR FICHEROS AL SERVIDOR

Esta es una de las funciones que mayor funcionalidad y dinamismo da a nuestras páginas web, ya que nos permite subir cualquier tipo de documento al servidor, aunque también, como programadores, podremos limitar mediante una serie de condiciones el que sólo se suban los ficheros que cumplan una serie de requisitos que nosotros fijemos.

Para poder utilizar esta función, debemos crear un formulario para la recepción de estos ficheros.

Cuando bajamos un fichero a nuestro servidor en modo local, lo colocaremos de forma temporal en el directorio que le hemos determinado en el fichero **php.ini**, en este caso el directorio será: **c:/ficheros/upload/** . En cambio, cuando trabajemos con un servidor que se encuentra en Internet y en el que alojamos nuestras páginas web, el tratamiento puede ser diferente, ya que podemos almacenar estos ficheros en el mismo servidor o, por ejemplo, hacer que nos lleguen mediante un mensaje a nuestro de correo electrónico.

9.2.1 Ejemplo

Con este ejemplo, lo que hacemos es crear en HTML un formulario para enviar las fotografías a otra página PHP que en este caso será **guarda.php**, que será la que procese el fichero que le enviamos.

<head>

<title>Creación de un portal con PHP y MySQL

</title>

</head>

```
<body bgcolor = "#303030">

<body text = "#FFFFFF">

<body leftmargin = "60">

<body topmargin = "60">

<font face = "Tahoma">

<font size = "3">

<form enctype="multipart/form-data" action="recepcion.php" method="post">

<input type="hidden" name="lim_tamano" value="500000">

<b>

<font size="6">

Formulario para el envío de ficheros:

</b>

</font size>

<p><b>Archivo a transferir<b><br>

<input type="file" name="archivo"></p>

<p><input type="submit" name="enviar" value= "Aceptar"> </p>

</form>

</head>

<b>Instrucciones de uso: Pulse el botón Examinar y seleccione el archivo que desee y luego pulse el botón Enviar.

</b>
```

En la figura 9.1 vemos el resultado de ejecutar este ejemplo.

Figura 9.1

Ahora, sólo nos quedaría crear un fichero llamado, por ejemplo, **recepcion.php**, en el que le diremos qué debe hacer con el fichero que hemos subido con el formulario creado anteriormente. En este caso, el código que vamos a crear, nos va a dar información del fichero que hemos subido y, además, se encargará de almacenarlo en nuestra carpeta temporal.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#FFFFFF">
<body leftmargin = "60">
```

```
<body topmargin = "60">
<font face = "Tahoma">
<font size = "3">
<?
if ($archivo != "none" AND $archivo_size != 0 AND $archivo_size<=$lim_tamano){
if (copy ($archivo, "c:/ficheros/upload/".$archivo_name)) {
echo "<h2>Se ha transferido el archivo $archivo_name</h2>";
/* Indicamos el nombre del archivo transferido. */
echo "<br>Su tamaño es: $archivo_size bytes<br>";
/* Indicamos el tamaño del archivo transferido */
echo "<br>El fichero es tipo: $archivo_type <br>";
/* Por último, indicamos a qué tipo de archivo corresponde. */
}
} else {
echo "<h2>No ha podido transferirse el fichero</h2>";
echo"<h3>su tamaño no puede exceder de $lim_tamano bytes
</h2>";
}
echo "<a href='.$archivo_name.'>";
?>
```

El fichero **recepción.php** es el encargado de procesar el fichero que hemos enviado. Como podemos ver en su código, este fichero se encarga de mostrar en pantalla datos del fichero que nos han enviado, nombre, tamaño, tipo.

Además, si desde nuestro navegador, a través de la dirección *http://localhost*, accedemos al directorio **/upload/**, veremos estos archivos que hemos subido almacenados en esta carpeta temporal.

El resultado de ejecutar este ejemplo podemos verlo en la siguiente imagen, figura 9.2.

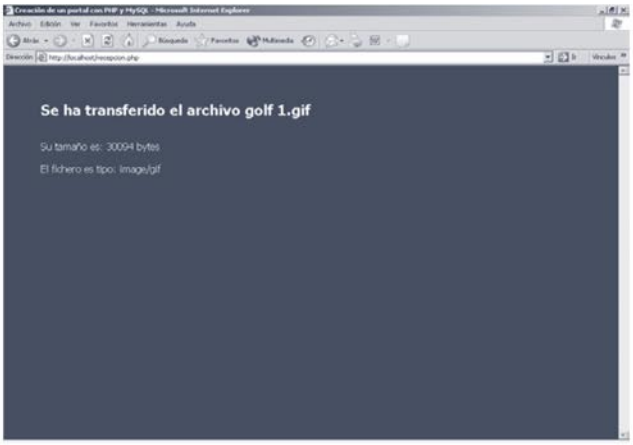

Figura 9.2

Capítulo 10

COOKIES Y SESIONES

10.1 COOKIES

Las cookies son pequeños ficheros de texto que maneja nuestro servidor para conocer datos de los usuarios y emplearlos si es necesario en cada una de sus visitas. Algunos de los usos que se suelen dar de las cookies pueden ser: recordar el nombre del usuario y sus preferencias.

Las cookies sólo almacenan datos que facilitan la navegación de los usuarios; en ningún caso guardan datos personales o de otro tipo que puedan violar su intimidad.

La siguiente cuestión referente al tema de las cookies es cómo se crean. Para ello empleamos la función *setcookie ( )*, con al menos dos argumentos fundamentales: el primero será el valor de la cookie y el segundo, el nombre de la variable.

Un ejemplo de uso de esta función, podría ser este: **setcookie ("pepe", $nombre);**.

Si probásemos el ejemplo anterior, al ser ejecutada esta instrucción en nuestro servidor se generará una cookie que tendrá el contenido: $nombre=pepe.

10.1.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#ffffff">
<body leftmargin = "60">
<body topmargin = "60">
<font face = "Tahoma">
<font size = "3">
<form enctype="multipart/form-data" action="ejemplo10-2.php" method="post">
<input type="hidden" name="action" value="setcookie">
Nombre: <input type ="text" name ="nombre"> <br>
<input type = "submit" value = "Enviar">
<?
if ($cookie == "setcookie") {
setcookie ("nombre", $tunombre);
}
?>
```

10.2 SESIONES

Podemos definir las sesiones como una serie de variables almacenadas en nuestro servidor que ofrecen información acerca de nuestros usuarios y que son diferentes para cada uno.

Hemos de recordar que al principio de este libro se explica cómo se debe de configurar nuestro fichero **php.ini**, para saber dónde almacenamos nuestras sesiones. En nuestro caso, hemos creado un directorio dentro de **c:/ficheros/**, llamado **sesiones**.

Existen dos funciones que son las más importantes siempre que hablemos de sesiones:

session_start ( ). Esta función se utiliza para crear una nueva sesión.

session_id ( ). Esta función se encarga de devolver el identificador de la sesión que ha creado el usuario y que lo identifica y distingue respecto al resto de usuarios que puedan estar conectados a esa página web.

10.2.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#ffffff">
<body leftmargin = "60">
<body topmargin = "60">
<font face = "Tahoma">
```

```
<font size = "3">
<?
if ($cookie == "setcookie")
{
setcookie ("nombre", $tunombre);
session_start ( );
}
?>
```

Si nos dirijimos a la carpeta **sesiones** que se encuentra en nuestro *localhost*, podremos ver las sesiones que creamos cada vez que probamos este código.

En la siguiente imagen, podemos ver algunas de las sesiones que se crean en nuestro directorio **sesiones**.

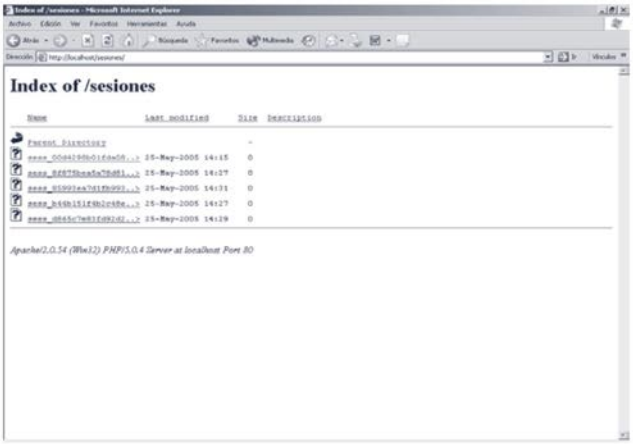

Figura 10.1

Capítulo 11

VARIABLES PREDEFINIDAS

¿Qué son las variables predefinidas? Son aquellas que, como su nombre indica, están previamente definidas; son variables que no pueden ser utilizadas ni modificadas.

A continuación, se muestran algunas de estas variables, así como unos pequeños ejemplos de aplicación de algunas de ellas.

$HTTP_REFERER: esta variable nos devolverá la dirección (URL) de la que procede el usuario. Podemos saber cuál ha sido la última web que visitó el usuario antes de acceder a nuestro portal utilizando esta variable.

$HTTP_ACCEPT_LANGUAGE: nos devuelve el lenguaje en el que tiene configurado el navegador el usuario que ahora nos está visitando.

$HTTP_USER_AGENT: nos indicará cual es el navegador utilizado por el usuario.

$REMOTE_ADDR: nos indicará cuál es la IP que tiene asignada el usuario.

$OS: devuelve el sistema operativo donde se ejecuta la página web.

$REQUEST_METED: indica el método de petición por el cual se accede a la página.

$SERVER_NAME: devolverá el nombre del servidor donde se ejecuta el script.

$SERVER_SOFTWARE: indica bajo qué servidor se ejecuta el *script*.

$DOCUMENT_ROOT: nos indica el directorio raíz donde se encuentra almacenado el fichero que estamos ejecutando.

$SERVER_ADMIN: esta variable almacena la persona de contacto administradora del servidor, en concreto el correo electrónico del administrador/a.

$SERVER_PORT: indica el puerto del equipo servidor que se está usando.

$SERVER_SIGNATURE: contiene la versión del servidor y el nombre del servidor virtual.

$SCRIPT_NAME: indica el nombre del fichero/script que se está ejecutando actualmente.

Estas sólo son algunas de las variables predefinidas, pero existen muchas más. Para conocerlas y saber cuáles son, sólo debemos crear una página web con el siguiente código y así podremos ver un listado con muchas más de estas variables:

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

11.1 EJEMPLO 1

Con el siguiente ejemplo, vamos a averiguar de dónde procede el usuario, cuál es el navegador que utiliza y cuál es su IP, así como otros datos de interés.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<hr size="8" color="ffffff">
<?
echo "Bienvenido a nuestro portal.";
echo "<br><br>";
echo "Usted usa el navegador: ";
echo $HTTP_USER_AGENT;
echo "<br><br>";
echo "Su dirección IP es: ";
echo $REMOTE_ADDR;
echo "<br><br>";
```

```
echo "El puerto que utiliza para conectarse al servidor web es: ";
echo $SERVER_PORT;
echo "<br><br>";
echo "Y viene de visitar: ";
echo $HTTP_REFERER;
echo "<br><br>";
echo "El nombre del servidor al que se conecta es: ";
echo $SERVER_NAME;
echo "<br><br>";
echo "El directorio en el que se almacenan los ficheros es: ";
echo $DOCUMENT_ROOT;
echo "<br><br>";
echo "El fichero PHP que está ejecutando se llama: ";
echo $SCRIPT_NAME;
echo "<br><br>";
echo "El correo del administrador del servidor es: ";
echo $SERVER_ADMIN;
echo "<br><br>";
echo "Versión del servidor y nombre del servidor virtual: ";
echo $SERVER_SIGNATURE;
echo "<br><br>";
echo "Sistema operativo: ";
echo $OS;
?>
<hr size="8" color="ffffff">
```

El resultado de ejecutar este ejemplo en el navegador nos muestra en pantalla los siguientes datos: navegador, dirección IP, puerto que se utiliza, dirección web de la que procedemos, nombre del servidor al que nos hemos conectado, ruta donde se encuentran almacenados los ficheros, nombre del fichero que se está ejecutando, datos del administrador del servidor, la versión y el nombre del servidor y el sistema operativo que estamos utilizando.

Como se puede observar, con unas sencillas líneas de código, podemos conocer datos muy interesantes de nuestros clientes, así como del servidor que estamos utilizando.

En la siguiente imagen, figura 11.1, vemos cómo se mostrará en nuestro navegador el resultado de este ejemplo cada vez que un usuario visite nuestro portal.

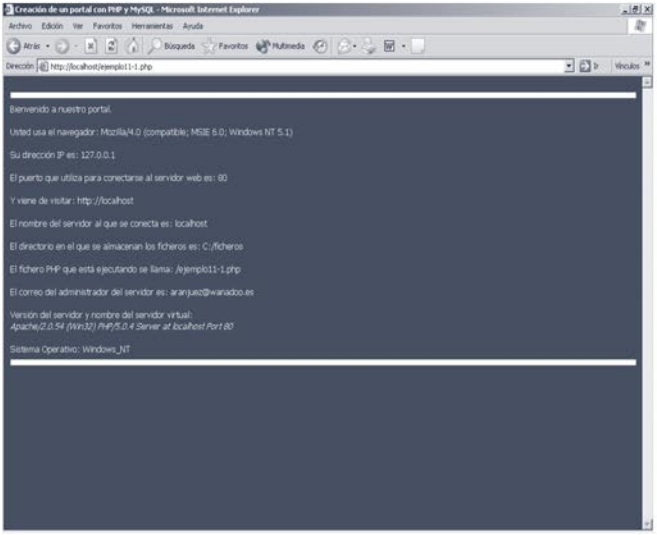

Figura 11.1

11.2 EJEMPLO 2

A continuación, se muestra un ejemplo muy útil, ya que si disponemos de un portal con código JavaScript o cualquier otro lenguaje que no es posible ejecutar por los actuales navegadores, redireccionará a otra página sin ese código, así como si tenemos nuestro portal en varios idiomas, detectará el idioma y lo redireccionará a la página en su idioma.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<hr size="8" color="ffffff">
<?
if (eregi ("MSIE", $HTTP_USER_AGENT))
{
print ( "Usted usa el navegador Microsoft Internet Explorer. \n");
}
 else
{
print ("Usted usa un navegador diferente a Microsoft Internet Explorer.\n");
}
/* Dependiendo del navegador que utilice el usuario que nos visita, mostrará en pantalla un mensaje u otro.*/
```

```
echo "<br>";
if ($HTTP_ACCEPT_LANGUAGE= es)
{
$a = Español;
/* Asignamos a $a el valor español, ya que lo primero que hacemos
es preguntar si la variable &HTTP_ACCEPT_LANGUAGE= es, que
corresponde al idioma español. Si no hiciésemos esto, se mostraría
en pantalla =es, pero, al hacer este cambio, lo que conseguimos es
que si se utiliza español, en pantalla mostrará español.
También podemos hacer un bucle para que distinga entre varios
idiomas.*/
echo "El idioma de su navegador es: ";
echo $a;
}
?>
```

Como podemos ver, con todos estos datos obtenidos de nuestros usuarios podremos redireccionarles a las páginas que se adapten más a las características de su equipo informático. Es decir, podemos tener creada una página para que se ejecute correctamente en Internet Explorer, pero que, por ejemplo, cierta parte de nuestro código no lo reconozca cualquier otro navegador que usen los usuarios de nuestra web. En este caso redireccionaríamos a nuestros usuarios a la página que se adapte a su navegador, e igualmente haríamos con el idioma, es decir, según sepamos el idioma de procedencia del usuario, podemos tener varias páginas en distintos idiomas y a las que redireccionaríamos a nuestros usuarios según el mismo.

En la siguiente imagen, figura 11.2, podemos ver el resultado de ejecutar este ejemplo, en el que se nos indica el navegador que utiliza el usuario, así como también el idioma en el que lo tiene configurado, por lo que podemos intuir su procedencia.

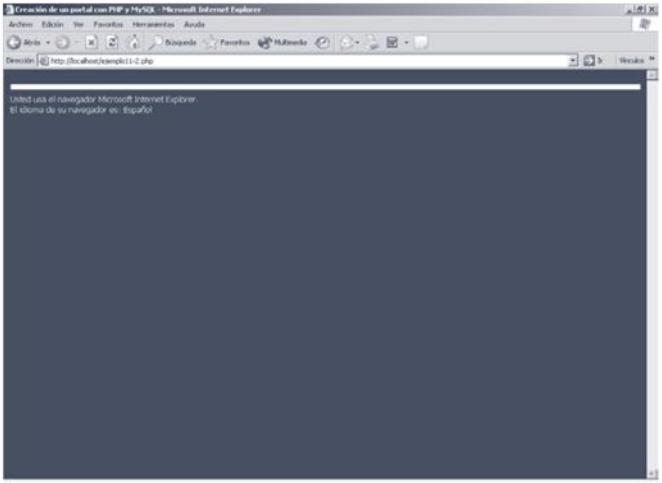
Creación de un portal con PHP y MySQL - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Búsqueda Favoritos Multimedia
Dirección http://localhost/ejemplo11-2.php
Usted usa el navegador Microsoft Internet Explorer.
El idioma de su navegador es: español

Figura 11.2

Capítulo 12

COMENZANDO CON MySQL

Una de las principales ventajas de crear una base de datos en una página web es, entre otras muchas aplicaciones, que se consiguen páginas dinámicas.

Para el uso y manejo de la base de datos MySQL, vamos a emplear una aplicación muy usual y extendida entre los usuarios que es phpMyAdmin.

12.1 PHPMYADMIN

Como ya se explicó anteriormente, phpMyAdmin es una aplicación que nos va a ayudar a gestionar y administrar nuestras bases de datos. Entre otras muchas, las funciones que podemos realizar con esta aplicación son: crear bases de datos, crear tablas, insertar datos en tablas, realizar consultas, borrar datos de tablas, borrar tablas, borrar bases de datos, etc.

12.2 CREAR UNA BASE DE DATOS

Para crear una base de datos, lo primero que debemos hacer es, desde el localhost, acceder a la carpeta **phpmyadmin/**, y nos encontraremos con el entorno de phpMyAdmin. Nos encontramos en la página principal, donde pone *Crear una base de datos*, pues es ahí donde debemos poner el nombre de la base de datos que vamos a crear. El menú desplegable que nos encontramos al lado lo dejaremos como está, con la opción *Collation*, y pulsaremos en el botón *Crear* para finalizar el proceso.

En la siguiente imagen, figura 12.1, podemos ver el detalle para crear una base de datos.

Figura 12.1

12.2.1 Ejemplo

Vamos a crear una base de datos llamada usuarios. Este sería el proceso:

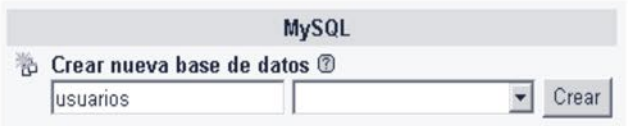

Figura 12.2

Ponemos el nombre de la base de datos: **usuarios**; seleccionamos donde pone *Collation* y lo dejamos en blanco o podemos seleccionar *utf8_spanish_ci*, que es el equivalente al español tradicional, y pulsamos el botón *Crear*. Una vez hayamos

pulsado el botón *Crear*, nos aparecerá la siguiente pantalla, figura 12.3, en la que nos indica que la base de datos se creó y que si queremos realizar otras operaciones con la base de datos que hemos creado.

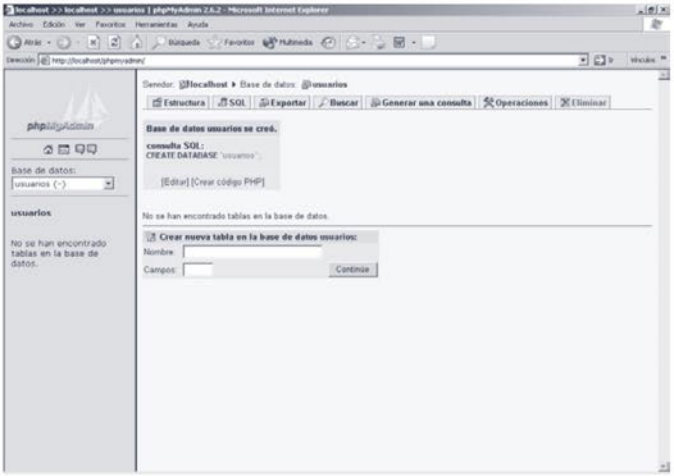

Figura 12.3

12.3 CREAR UNA TABLA

Para crear una tabla, seguiremos los siguientes pasos: en la base de datos que hemos creado anteriormente aparece un espacio para crear una tabla. En este espacio hemos de indicar el nombre de la tabla y los campos que contendrá la misma. Si, por ejemplo, hemos perdido esa página posterior a la de crear una base de datos porque hemos cerrado el navegador, no hay problema, ya que siempre que queramos crear una tabla no tenemos más que pulsar en el nombre de la base de datos, situado a la izquierda, donde pone el nombre de la base de datos (**usuarios**), y aparecerá de nuevo para crear una base de datos.

En la siguiente imagen, figura 12.4, podemos ver el espacio donde podemos crear una tabla en nuestra base de datos.

Figura 12.4

12.3.1 Ejemplo

Vamos a crear la tabla **clientes** dentro de la base de datos **usuarios**. Esta tabla va a tener siete campos, que van a ser: id, nombre, apellidos, edad, localidad, teléfono y e-mail.

Figura 12.5

Al pulsar sobre el botón *Continúe*, nos aparecerá un menú que debemos rellenar para cada uno de los campos de nuestra tabla. Podemos verlo en la figura 12.6.

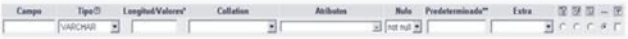

Figura 12.6

Cada uno de los campos de este formulario debemos rellenarlo en función al campo que corresponde en la tabla. Los únicos campos que rellenaremos o modificaremos serán: Campo, Tipo, Longitud/Valores, Collation, Extra y, de los iconos situados a la derecha, referentes a las acciones, sólo modificaremos uno de ellos, el resto lo dejamos tal cual.

A continuación, se representa una tabla con los valores que hemos introducido para crear esta tabla de forma resumida. Estos valores no tienen por qué ser exactamente así, ya que podemos cambiar tanto el tipo como la longitud. Pero para este ejemplo se ha estimado que con estos valores es suficiente.

Campo	Tipo	Longit. /Valor.	Collation	Extra	Acción
id	int	3		auto_increment	primaria
nombre	varchar	10	utf8_spanish_ci		
apellidos	varchar	20	utf8_spanish_ci		
edad	int	2			
población	varchar	12	utf8_spanish_ci		
teléfono	int	9			
e-mail	varchar	25	utf8_spanish_ci		

Quedando finalmente nuestra tabla como podemos ver en la siguiente imagen, figura 12.7.

	Campo	Tipo	Collation	Atributos	Nulo	Predeterminado	Extra	Acción
☐	id	int(3)			No		auto_increment	
☐	nombre	varchar(10)	utf8_spanish_ci		No			
☐	apellidos	varchar(20)	utf8_spanish_ci		No			
☐	edad	int(2)			No	0		
☐	poblacion	varchar(12)	utf8_spanish_ci		No			
☐	telefono	int(9)			No	0		
☐	email	varchar(25)	utf8_spanish_ci		No			

Revisar todos/as / Desmarcar todos Con marca:

Figura 12.7

12.4 INSERTAR DATOS EN UNA TABLA

Ya tenemos creada nuestra base de datos con su tabla. Ahora es el momento de introducir datos en nuestra tabla.

Para ello, desde nuestra tabla, nos fijaremos en la parte superior, donde encontraremos, entre otros, un icono con el texto *Insertar*. Podemos verlo en la figura 12.8.

Figura 12.8

Una vez hayamos seleccionado la opción *Insertar*, aparecerá un menú para introducir los datos en nuestra tabla.

En la figura 12.9 podemos ver el formulario para insertar datos en nuestra tabla.

Figura 12.9

12.4.1 Ejemplo

Vamos a introducir datos en nuestra tabla. Insertamos cuatro registros. Algo muy importante a la hora de rellenar el formulario con los datos es dejar el campo id sin rellenar, ya que al ser un campo auto_increment, cada vez que insertemos un registro nuevo se le asignará un id automáticamente de manera incremental.

Cada vez que insertemos un registro, podremos ver el resultado en pantalla con el correspondiente código SQL que lo inserta en nuestra tabla. Podemos verlo en la figura 12.10.

Figura 12.10

En la siguiente tabla se pueden ver los datos que hemos introducido para realizar este ejemplo.

id	Nombre	Apellidos	Edad	Población	Teléfono	e-mail
1	José	Mata Pérez	31	Madrid	293439281	jose@sucorreo.com
2	Juan	López Gómez	43	Barcelona	431298448	juan@sucorreo.com
3	Pedro	Lara Manjón	28	Sevilla	122432987	pedro@sucorreo.com
4	José	Martín Ortiz	28	Madrid	458446158	josemartin@sucorreo.com

12.5 CONSULTAR DATOS DE UNA TABLA

Una vez que hayamos terminado de introducir registros en nuestra tabla, pulsaremos sobre el botón *Buscar* para poder realizar consultas.

Figura 12.11

Y aparecerá un formulario en el que podemos optar por dos formas de buscar registros en nuestra tabla. En el primero de ellos, seleccionaremos los campos que queremos mostrar; pueden ser todos o algunos de ellos; debemos indicar el número de registros por página que queremos mostrar, si queremos mostrarlos en orden ascendente o descendente y las condiciones que debe cumplir esa búsqueda. Respecto a las condiciones, podemos poner tantas como queramos. Por ejemplo, una condición es que aparezcan los registros donde la edad sea igual a 40 años. En este caso, en el campo condiciones, insertaríamos algo así: edad>40. Otra condición puede ser que queremos que nos muestre los datos de los que se llaman José. Para ello, en el campo condición pondríamos nombre = "jose".

En la siguiente imagen, figura 12.12, podemos ver el formulario para consulta de registros.

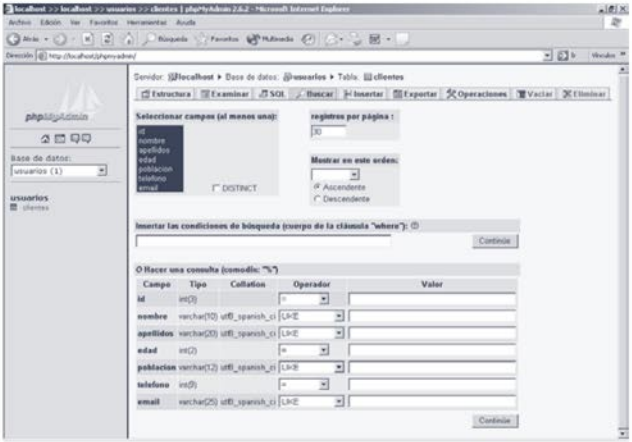

Figura 12.12

Otra forma de consultar registros en este formulario es introduciendo en los campos destinados a la búsqueda unas condiciones y seleccionar el operador necesario. Por ejemplo, si

queremos buscar a los clientes mayores de 40 años, en el campo operador introduciríamos > y en el campo valor pondríamos 40. Otro ejemplo podría ser buscar clientes que se llaman José. Para ello, en el campo operador del nombre podríamos LIKE y en el campo valor introducimos José.

12.5.1 Ejemplo 1

En este ejemplo vamos a hacer una consulta utilizando el formulario de la parte superior, en el que vamos a buscar los registros cuyos clientes tengan más de 40 años, y queremos mostrar solamente su nombre, apellidos, edad y teléfono. Para ello, en la siguiente imagen, figura 12.13, se muestra cómo debemos rellenar el formulario para ejecutar esa consulta.

Figura 12.13

En la figura 12.14 vemos el resultado de ejecutar la consulta.

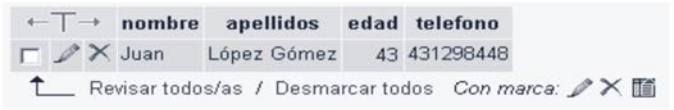

Figura 12.14

12.5.2 Ejemplo 2

En este ejemplo vamos a hacer una consulta utilizando el formulario de la parte inferior, en el que vamos a buscar los registros

cuyos clientes se llamen José. Podemos ver cómo se realiza la consulta en la siguiente imagen, figura 12.15.

Figura 12.15

El resultado de esta consulta podemos verlo en la figura 12.16.

Figura 12.16

12.6 ACTUALIZAR DATOS DE UNA TABLA

Lo ideal es que una base de datos esté lo más actualizada posible. Bien, pues para ello tenemos una opción con la que podemos actualizar los datos de la misma en cualquier momento. Imaginemos por ejemplo que un cliente de nuestra base de datos quiere modificar su número de teléfono. Para ello, lo que hacemos es buscar al cliente en la base de datos y, una vez que lo localicemos, seleccionamos la opción *Editar*.

Figura 12.17

12.6.1 Ejemplo

Imaginemos que el cliente José Martín Ortiz quiere actualizar su número de teléfono. Para ello, una vez localizado en la base de datos, seleccionamos la opción *Editar*, y aparecerá la siguiente pantalla, figura 12.18, donde modificaremos el número de teléfono, y pulsaremos el botón *Continuar* para validar.

Figura 12.18

12.7 BORRAR DATOS DE UNA TABLA

Para borrar datos de una tabla tenemos disponibles dos opciones: una de ellas sería un borrado de registros parcial o total y otra sería realizar un vaciado completo de los registros que contiene la tabla que estamos empleando.

Prestaremos especial atención antes de realizar cualquiera de estas operaciones, ya que las acciones que ejecutemos serán irreversibles, por lo tanto los datos que borremos no podremos volver a recuperarlos.

Una de las opciones es la que muestra la figura 12.19, en la que borramos únicamente dos registros de nuestra tabla. Como se puede observar, debemos marcar la casilla situada a la izquierda, para eliminar los registros deseados.

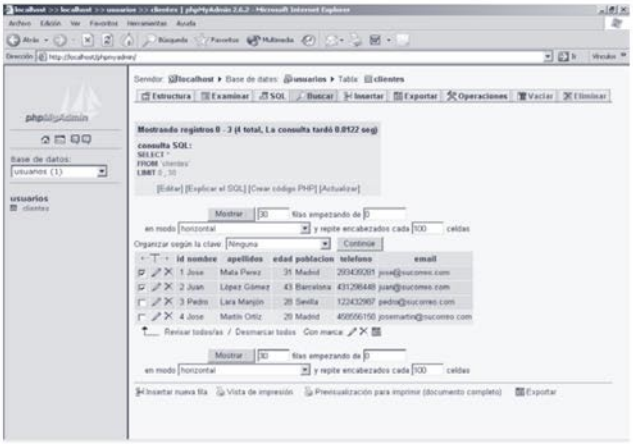

Figura 12.19

Como podemos observar en la imagen, primero seleccionamos los registros que queremos eliminar y luego pulsamos en la parte superior de la pantalla en la opción *Eliminar*.

Otra de las opciones es, sin necesidad de seleccionar ningún registro, pulsar en la parte superior derecha de la pantalla en la opción *Vaciar*. Con esta última, lo que hacemos es borrar todos los registros de la tabla.

12.8 BORRAR UNA TABLA

Para borrar una tabla, lo que hacemos es seleccionar la base de datos que contiene esa tabla, por lo que seleccionaremos la base de datos **usuarios**, en la parte situada a la izquierda de nuestra pantalla, y nos encontraremos una imagen como la que podemos ver en la figura 12.20.

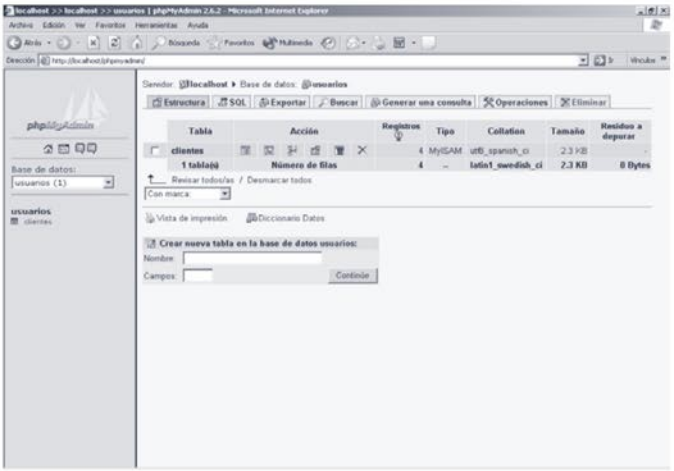

Figura 12.20

Sólo quedaría pulsar en el botón *Eliminar* para borrar completamente la tabla.

En la siguiente imagen, figura 12.21, podemos ver el icono para eliminar completamente la tabla.

Figura 12.21

12.9 BORRAR UNA BASE DE DATOS

Para borrar una base de datos, nos situamos en la página principal de nuestra base de datos y pulsamos sobre el botón *Eliminar* para borrar la base de datos en la que nos encontremos actualmente.

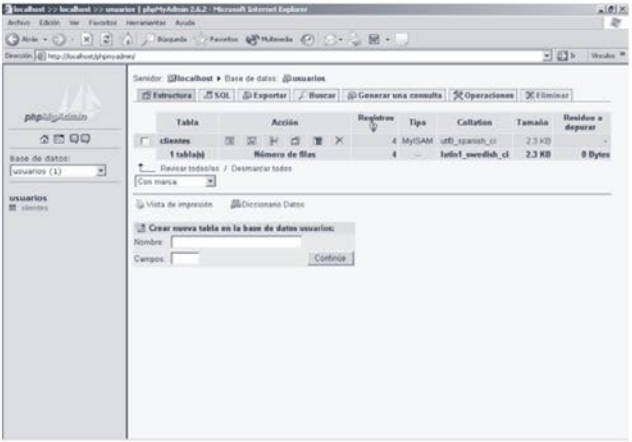

Figura 12.22

Y finalmente pulsamos sobre el enlace que pone *Eliminar*, para poder borrar la base de datos de forma definitiva.

En la siguiente imagen, figura 12.23, podemos ver en enlace en el que debemos pulsar para borrar la base de datos.

Figura 12.23

Capítulo 13

PHP Y MySQL

La combinación del lenguaje PHP junto con la base de datos MySQL es utilizada en un gran número de páginas web que podemos encontrar mientras navegamos por Internet, debido a la potencia que se consigue utilizando estas dos aplicaciones juntas.

PHP dispone de una amplia lista de funciones para utilizarlas con la base de datos MySQL, y a lo largo de este capítulo veremos algunas de ellas, así como ejemplos de las mismas.

PHP es un lenguaje de programación de páginas web muy potente y muy extenso, pero su uso sin una base de datos, y más en concreto de la base de datos MySQL, sería desaprovechar muchas de las posibilidades que nos ofrece este lenguaje.

A continuación, vamos a explicar las instrucciones más importantes, así como necesarias, para poder operar entre PHP y MySQL como, por ejemplo, para conectar con una base de datos, seleccionar una base de datos, seleccionar registros de una base de datos, etc.

13.1 CONECTAR A UNA BASE DE DATOS

Para conectar desde PHP a una base de datos creada en MySQL, utilizaremos la instrucción *mysql_connect ( )*, a la que le deben acompañar tres parámetros: el primero de ellos es determinar el *host* al que nos conectamos, y luego le indicaremos el nombre de usuario y la contraseña.

Cuando trabajemos en modo local, el *host* al que conectamos será, por ejemplo, "127.0.0.1", y el nombre de usuario y contraseña serán los mismos que definimos al instalar MySQL. Si alguno de estos datos no se introduce de forma correcta, no podremos conectarnos a la base de datos. Recibiremos un mensaje de error que nos indicará que los datos de conexión no son los correctos, por lo que hemos de prestar mucha atención a la hora de preparar la conexión a la base de datos que vamos a utilizar en cada caso.

13.1.1 Ejemplo

```
<?
$host = "127.0.0.1";
$usuario = "user"; // Cambiar por su nombre de usuario.
$password = "pass"; // Cambiar por su password.
$conectar = mysql_connect ($host, $usuario, $password);
?>
```

Con este ejemplo lo que hacemos es preparar la conexión a una base de datos, algo que haremos muy a menudo al crear nuestras páginas web. Es la primera acción que se realiza, por lo que la aplicación que hemos preparado podemos guardarla en un fichero que llamaremos, por ejemplo, **conectar.php**, para luego incluirlo posteriormente en nuestras páginas en las que sea necesario conectar a una base de datos, y nos ahorrará muchas líneas de código. Si, por ejemplo, tenemos 15 páginas en las que conectamos con una base de datos, nos estamos ahorrando 90 líneas de código (6 líneas x 15

páginas). Posteriormente, podemos incluir este fichero utilizando la instrucción *include ( )* que vimos anteriormente.

Un problema muy frecuente que nos podemos encontrar a la hora de utilizar esta instrucción es que los datos de conexión que utilizamos no sean correctos. Debemos procurar poner bien esos datos, prestando especial atención en que el nombre de usuario y la contraseña que introducimos sean los mismos que pusimos cuando configuramos MySQL.

13.2 SELECCIONAR UNA BASE DE DATOS

Mediante la instrucción **mysql_select_db ( )**, seleccionamos una base de datos de entre todas las que tengamos creadas para trabajar con ella.

13.2.1 Ejemplo

```
<?
include ("conectar.php");
mysql_select_db ("prueba", $conectar);
?>
```

En este ejemplo, primero incluimos el fichero **conectar.php** que creamos anteriormente para conectarnos, y posteriormente le indicamos la base de datos a la que nos conectaremos, que en este caso será la base de datos llamada **prueba**.

13.3 EJECUTAR UNA CONSULTA EN UNA BASE DE DATOS

Ejecuta una consulta a la base de datos activa en el servidor asociado al identificador de conexión.

13.3.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<body leftmargin = "50">
<body topmargin = "50">
<font face = "tahoma">
<font size = "2">
<?
echo "<p align=center>";
echo "A continuación se muestra el resultado de seleccionar todos los registros de las tablas nombre y apellidos.";
$host = "127.0.0.1";
$usuario = "user"; // Cambiar por su nombre de usuario.
$password = "pass"; // Cambiar por su password.
$conectar = mysql_connect ($host, $usuario, $password);
mysql_select_db ("usuarios", $conectar);
$consulta = "SELECT nombre, apellidos FROM clientes";
$query = mysql_query ($consulta, $conectar);
echo "<table align=center border=1 bgcolor=#6B6BFF cellspacing=5>";
while ($reg = mysql_fetch_row($query)){
echo "<tr>";
echo "<br>";
foreach($reg as $cambia){
```

```
echo "<td>",$cambia,"</td>";
}
}
echo "</table>";
?>
```

En este ejemplo, al igual que en el que hicimos anteriormente, incluimos el fichero para conectar a la base de datos y, posteriormente a esto, ejecutamos una consulta. Concretamente, le decimos que nos seleccione los campos nombre y apellidos de la tabla **clientes** y nos muestre los registros correspondientes a esta consulta.

En la figura 13.1 podemos ver la imagen correspondiente al resultado de ejecutar este ejemplo y, como podemos ver, nos muestra todos los registros, en concreto los cuatro que corresponden a la tabla clientes que creamos anteriormente y en la que insertamos esos cuatro registros.

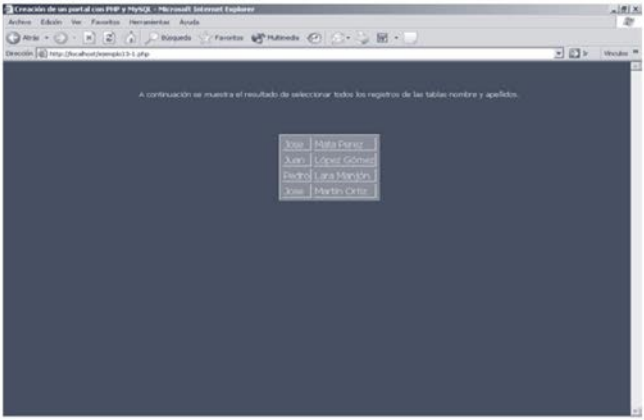

Figura 13.1

13.4 DEVOLVER CONSULTAS EN UN ARRAY

Esta instrucción nos devuelve un array con el resultado de una consulta.

13.4.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<?
include ("conectar.php");
mysql_select_db ("usuarios", $conectar);
$consulta = "SELECT nombre, apellidos FROM clientes";
$query = mysql_query ($consulta, $conectar);
$array = mysql_fetch_array ($query);
echo ($array ["nombre"]. "<br>");
echo ($array ["apellidos"]. "<br>");
?>
```

En este ejemplo almacenamos los datos de la consulta que realizamos en un array.

13.5 NÚMERO DE REGISTROS OBTENIDOS EN UNA CONSULTA

Esta instrucción devuelve el número de registros obtenidos en una consulta.

13.5.1 Ejemplo

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<body leftmargin = "50">
<body topmargin = "50">
<font face = "tahoma">
<font size = "2">
<?
echo "<p align=center>";
echo "A continuación se muestra el resultado de seleccionar todos los registros de las tablas nombre y apellidos.";
$host = "127.0.0.1";
$usuario = "user";
$password = "pass";
$conectar = mysql_connect ($host, $usuario, $password);
mysql_select_db ("usuarios", $conectar);
$consulta = "SELECT nombre, apellidos FROM clientes";
$query = mysql_query ($consulta, $conectar);
echo "<table align=center border=1 bgcolor=#6B6BFF cellspacing=5>";
while ($reg = mysql_fetch_row($query)){
echo "<tr>";
echo "<br>";
foreach($reg as $cambia){
echo "<td>",$cambia,"</td>";
}
}
```

```
echo "</table>";
echo "<br>";
print ("Y aquí se muestra el número de registros encontrados");
echo "<br><br>";
$numregistros = mysql_num_rows ($query);
print ("Registros encontrados:"." "."$numregistros");
?>
```

Este ejemplo es prácticamente igual que el del apartado 13.3.1., pero con la diferencia de que aquí recibiremos en pantalla el mensaje *El número de registros encontrados es: (valor_numérico)*. Es decir, nos mostrará el mensaje con un valor que dependerá del número de registros encontrados al ejecutar la consulta realizada.

En la figura 13.2 podemos ver el resultado de ejecutar este ejemplo.

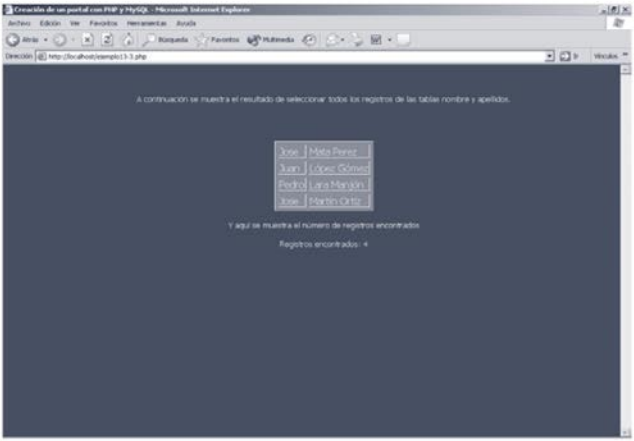

Figura 13.2

Capítulo 14

PRIMERAS APLICACIONES PARA NUESTRA WEB

A continuación, en este capítulo, vamos a desarrollar una serie de aplicaciones que se van a poder emplear en cualquiera de las páginas web que desarrollemos, ya que son pequeñas aplicaciones fáciles de insertar en cualquier parte de nuestro sitio.

Por ejemplo, un contador de visitas es una aplicación que no tiene sentido emplearla en una única página web, sino que lo más lógico es utilizarla en cuantas páginas de nuestra web deseemos para ver el tráfico entre unas y otras, ver cuál de estas es la más visitada, etc. Y así con cualquiera de las aplicaciones que a continuación se van a explicar, como insertar la fecha y hora, mostrar el tiempo de carga de una página, personalizar la página de error, etc., serán aplicables a cualquiera de nuestras páginas web.

14.1 FECHA Y HORA EN NUESTRAS PÁGINAS

Para el manejo tanto de fechas como de horas, utilizaremos la función **date ( )**, y mediante la siguiente tabla veremos los códigos necesarios para mostrarla en pantalla en el formato que deseemos. Podemos elegir entre mostrar sólo la hora, sólo la fecha o ambas opciones juntas e incluso mostrar los días y meses, pero en inglés. Para ello emplearemos el comando **echo date ( )**.

Código	Resultado
a	am o pm
A	AM o PM
h	Hora en formato 1 – 12
H	Hora en formato 0 – 23
i	Minutos
s	Segundos
j	Día del mes sin ceros
d	Día del mes con ceros
D	Abreviatura del día de la semana en inglés
I	Nombre del día en inglés
z	Número de día del año, de 1 a 365
m	Número del mes, de 1 a 12
M	Abreviatura del mes en inglés
F	Nombre del mes en inglés
y	Año, con formato de 2 dígitos
Y	Año, con formato de 4 dígitos

14.1.1 Ejemplo

Este pequeño código lo podemos poner en nuestra página principal o en todas las demás, para mostrar la fecha y la hora actuales.

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<body leftmargin = "50">
<body topmargin = "50">
<font face = "tahoma" size="2">
<?
echo "Hoy es día" ." ", date ("d/m/Y") , " y la hora actual es" ." ", date ("h:i:s"), ". <br> <br> Queremos daros la bienvenida a nuestro portal.";
echo "<br>";
echo "<br>";
echo "<br>";
/* Como podemos fijarnos, el resultado que nos da esta última línea es un poco a modo de saludo, por lo que si lo que queremos es darle un toque más sencillo, y emplearlo en cualquier lugar de nuestra web para indicar sólo la fecha y la hora, bastaría con eliminar la línea anterior y utilizar la línea de código que viene a continuación, que es algo más sencilla para utilizarla y mostrar sólo el día y la hora. */
echo date ("d/m/Y"), " ---- ". Date ("h:i");
?>
```

En una línea, y de forma sencilla, mostramos la siguiente información:

Hoy es día 27/05/2005 y la hora actual es 10:15:24. Queremos daros la bienvenida a nuestro portal.

Y, además, se ha creado una línea más sencilla en la que mostramos igualmente la fecha y la hora, por si algún usuario quiere emplearla en todas sus páginas, sólo a modo de información.

En la siguiente imagen, figura 14.1, podemos ver el resultado de ejecutar este ejemplo.

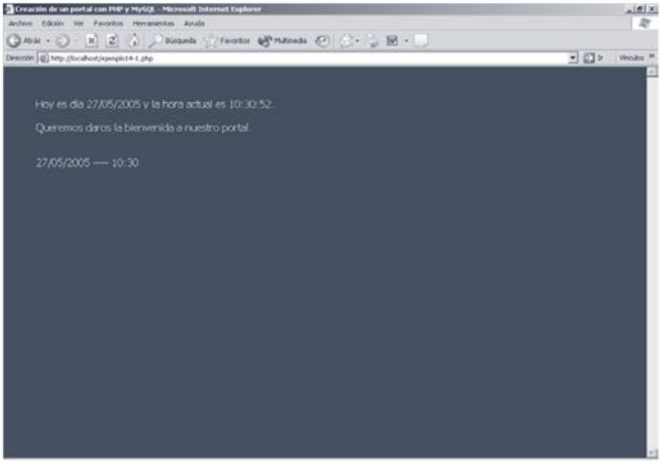

Figura 14.1

14.2 CONTADOR DE VISITAS

A continuación, lo que vamos a explicar es una sencilla aplicación que nos puede ser de una gran utilidad, ya que podremos insertarla en cualquiera de nuestras páginas, y tendremos tantos contadores de visitas independientes como queramos. E incluso vamos a crear un contador de visitas global que, como veremos más adelante, nos va a servir para poder generar, en una página, un resumen con cada una de las visitas que recibe cada página, en vez de tener que ir viendo una por una las estadísticas de cada una de las páginas en las que tengamos insertado un contador. De este modo evitaremos perder el tiempo para saber el número de visitantes que tenemos en cada una de ellas.

A continuación, en este ejemplo crearemos e insertaremos este código en la página principal, pero para poder agregarlo en varias páginas, lo único necesario sería crear tantos ficheros ***.txt** como contadores queramos insertar en ellas y hacer una llamada a los respectivos ficheros.

14.2.1 Ejemplo

Para comenzar, debemos crear un fichero llamado **visitas.txt** y dejarlo en blanco, ya que será necesario que no contenga nada. A continuación, insertaremos este código en nuestra página principal, donde deseemos que nos aparezca el número de visitantes que tenemos.

```
<html>
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<?
$cuenta = "visitas.txt";
function contador($cuenta)
{
$fp = fopen($cuenta, rw);
$num = fgets($fp,5);
$num += 1;
print "Número de visitas: ";
```

```
echo "$num";
exec( "rm -rf $cuenta");
exec( "echo $num > $cuenta");
}
if (!file_exists($cuenta))
{
exec( "echo 1 > $cuenta");
}
contador($cuenta);
?>
```

14.2.2 Recuento de visitas de todo el portal

A continuación, se explica cómo podemos crear una página en la que visualicemos todas las visitas al completo que recibe nuestro portal, es decir, las visitas de cada página en un listado.

Esto es de gran utilidad ya que así no tenemos que ir viendo página por página para comprobar cuáles son las más visitadas o el número de visitas que tiene cada una. Es obvio que la página que más visitas puede contener será la página princial (index), ya que por esta accederá la mayoría de nuestros usuarios, aunque puede que muchos otros usuarios que ya conocen nuestro portal accedan directamente desde un enlace a otra sección de la página web que desean visitar.

14.2.2.1 EJEMPLO 1

A continuación, vamos a ver un pequeño ejemplo de cómo poder visualizar los contadores de cuatro de las páginas de las que se compone nuestro portal. Recuerde que podemos modificarlo para poder visualizar todas las que deseemos.

En cada una de nuestras páginas hemos de incluir el código que se muestra en el *ejemplo 14.2.1*, para lo que simplemente habrá

que modificar la variable *$cuenta,* ya que cada una de las páginas que lleve contador deberá hacer referencia a un fichero ***.txt** diferente, es decir, **visitas1.txt**, **visitas2.txt**, **visitas3.txt**, etc.

Este sería el código de ejemplo para poder visualizar las visitas de cuatro de las páginas web de nuestro portal. Si fuesen más de cuatro, el proceso sería igual de sencillo que este, tan sólo sería necesario repetirlo, modificando el nombre de los ficheros.

```
<html>
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<h3><p align="center">RESUMEN DE VISITAS DEL PORTAL</p></h3>
<hr size="8" color="ffffff">
<?
print ("Las visitas de la página A son: ");
include ("visitas.txt");
echo "<hr size = 2 color = ffffff width = 35% align = left>";
echo "<br>";
print ("Las visitas de la página B son: ");
include ("visitas2.txt");
echo "<hr size = 2 color = ffffff width = 35% align = left>";
echo "<br>";
print ("Las visitas de la página C son: ");
```

```
include ("visitas3.txt");
echo "<hr size = 2 color = ffffff width = 35% align = left>";
echo "<br>";
print ("Las visitas de la página D son: ");
include ("visitas4.txt");
echo "<hr size = 2 color = ffffff width = 35% align = left>";
?>
```

Así haríamos hasta incluir tantos contadores como tengamos en nuestro portal. Como se puede observar, los ficheros donde se almacenan las visitas de cada página los hemos ido nombrando como **visitas.txt**, **visitas2.txt**, etc, siendo cada uno el correspondiente a cada una de las páginas de nuestra web.

14.2.2.2 EJEMPLO 2

Pero aún podemos rizar un poco más el rizo y mejorar la imagen de este resumen de visitas de nuestro portal, para poder interpretar de una manera más eficaz los datos obtenidos. Para ello, lo que vamos a hacer es implementar el código del ejemplo anterior para dar a esta misma página un sentido visual que nos ayude a interpretar los datos obtenidos, incluyendo un espacio en el que se mostrará el tanto por ciento de visitas que recibe cada página mostrado además con una barra.

Para realizar el siguiente ejemplo, es necesario emplear una imagen que será la que creará una barra implementado el tanto por ciento de visitas de cada web, la imagen será un pequeño cuadrado de reducidas dimensiones, como el que podemos ver en la siguiente imagen, figura 14.2.

Figura 14.2

El ejemplo finalmente al completo quedaría de la siguiente manera:

```
<html>
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<h3>
<p align="center">RESUMEN DE VISITAS DEL PORTAL</p>
</h3>
<hr size="8" color="ffffff">
<?
print ("Las visitas de la página A son: ");
include ("visitas.txt");
echo "<hr size = 2 color = ffffff width = 25% align = left>";
echo "<br>";
print ("Las visitas de la página B son: ");
include ("visitas2.txt");
echo "<hr size = 2 color = ffffff width = 25% align = left>";
echo "<br>";
print ("Las visitas de la página C son: ");
include ("visitas3.txt");
echo "<hr size = 2 color = ffffff width = 25% align = left>";
echo "<br>";
print ("Las visitas de la página D son: ");
include ("visitas4.txt");
```

```
echo "<hr size = 2 color = ffffff width = 25% align = left>";
echo "<br><br>";
/*A partir de aquí se crean los gráficos que mostrarán el porcentaje de visitas de cada una de las páginas.*/
echo "<hr size=8 color=ffffff>";
$archivo1 = "visitas.txt";
$archivo2 = "visitas2.txt";
$archivo3 = "visitas3.txt";
$archivo4 = "visitas4.txt";
$abre1 = fopen($archivo1, "r");
$abre2 = fopen($archivo2, "r");
$abre3 = fopen($archivo3, "r");
$abre4 = fopen($archivo4, "r");
$total1 = fread($abre1, filesize($archivo1));
$total2 = fread($abre2, filesize($archivo2));
$total3 = fread($abre3, filesize($archivo3));
$total4 = fread($abre4, filesize($archivo4));
$visitas=$total1+$total2+$total3+$total4;
$por1=$total1*100/$visitas;
$por1=intval ( $por1 ,10);
$por2=$total2*100/$visitas;
$por2=intval ( $por2 ,10);
$por3=$total3*100/$visitas;
$por3=intval ( $por3 ,10);
$por4=$total4*100/$visitas;
$por4=intval ( $por4 ,10);
echo "Página A: <b>$total1</b>visitas - <b>$por1 %</b>"." ";
echo "<img height=15 width=$por1 src=figura14-2.jpg>";
echo "<br><br>";
echo "Página B: <b>$total2</b> visitas - <b>$por2 %</b>"." ";
echo "<img height=15 width=$por2 src=figura14-2.jpg>";
echo "<br><br>";
```

```
echo "Página C: <b>$total3</b> visitas - <b>$por3 %</b>"." ";
echo "<img height=15 width=$por3 src=figura14-2.jpg>";
echo "<br><br>";
echo "Página D: <b>$total4</b> visitas - <b>$por4 %</b>"." ";
echo "<img height=15 width=$por4 src=figura14-2.jpg>";
echo "<br><br><br>";
$todo=$por1+$por2+$por3+$por4;
echo "<hr size = 2 color = ffffff width = 30% align = left>";
echo "Total Visitas: <b>$visitas</b> de un <b>$todo % </b>";
echo "<img height=15 width=$todo src=figura14-2.jpg>";
?>
```

Aunque el código es muy extenso, se puede observar que no por eso es complicado de comprender, ya que es repetitivo.

En la siguiente imagen, figura 14.3, podemos ver el resultado de ejecutar este ejemplo.

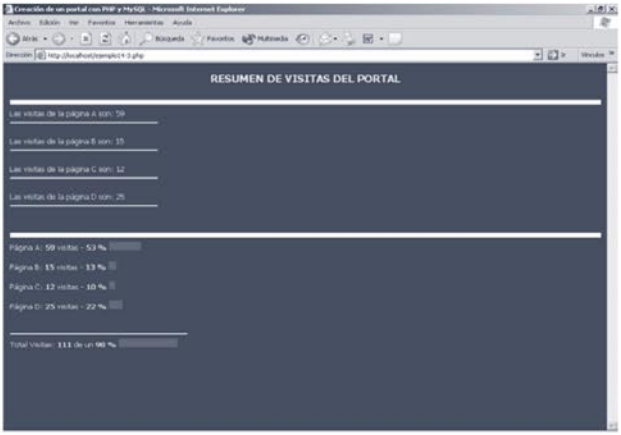

Figura 14.3

14.3 MOSTRAR EL TIEMPO DE CARGA DE NUESTRAS PÁGINAS

Cuántas veces hemos creado páginas que consideramos un poco pesadas en cuanto al volumen de imágenes, textos, operaciones, etc.

Con el ejemplo que vamos a desarrollar a continuación vamos a ser capaces de averiguar el tiempo que tarda en crearse nuestra página web. Podemos ver algo parecido a esto en Google, que cada vez que se realiza una búsqueda indica el tiempo que ha tardado en realizarla.

A continuación vamos a ver dos ejemplos sencillos para ver la diferencia de tiempo que hay entre la carga de una página sencilla con una imagen y el tiempo de carga de otra con alguna operación en bucle.

14.3.1 Ejemplo 1

En el siguiente ejemplo vemos el tiempo que tarda en cargarse una página sencilla que tiene una imagen de tan sólo 52 Kb.

```
<?
$tiempo= microtime ( );
$tiempo= explode (" ", $tiempo);
$tiempo= $tiempo[1] + $tiempo[0];
$tiempoinicial= $tiempo;
?>
<html>
<head>
```

```
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<body leftmargin = "300">
<body topmargin = "150">
<img src=foto2.jpg>
<?
echo "<br><br><br>";
$tiempo= microtime ( );
$tiempo = explode (" ", $tiempo);
$tiempo = $tiempo[1] + $tiempo[0];
$tiempofinal= $tiempo;
$tiempototal= ($tiempofinal - $tiempoinicial);
echo "La página tardó en crearse ".$tiempototal." segundos";
?>
```

En la siguiente imagen, figura 14.4, podemos ver el tiempo que tarda en cargarse este ejemplo, que en este caso en concreto es de 0.000182867050171 segundos, como nos ha indicado al terminar la carga de la página.

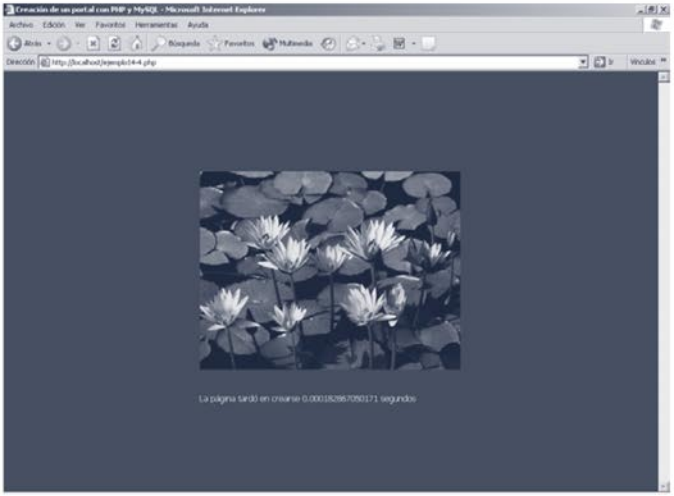

Figura 14.4

14.3.2 Ejemplo 2

A continuación, vamos a desarrollar otro ejemplo en el que creamos un bucle, para ver el tiempo que tarda en cargarse y ver la diferencia de tiempo entre ambos ejemplos.

Aquí vemos que, aunque apenas es apreciable el tiempo, si es sensiblemente mayor el tiempo que tarda en crearse esta página respecto al otro ejemplo. En este caso el tiempo que tarda en crearse es de 0.0136001110077 segundos.

```
<?
$tiempo= microtime ( );
$tiempo= explode (" ", $tiempo);
$tiempo= $tiempo[1] + $tiempo[0];
$tiempoinicial= $tiempo;
```

```
?>
<html>
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<?
$i=1;
while ($i<1000)
{
print ("Número ".$i);
echo "<br>";
$i++;
}
?>
<?
echo "<br><br><br>";
$tiempo= microtime ( );
$tiempo= explode (" ", $tiempo);
$tiempo= $tiempo[1] + $tiempo[0];
$tiempofinal= $tiempo;
$tiempototal= ($tiempofinal - $tiempoinicial);
echo "La página tardó en crearse ".$tiempototal." segundos";
?>
```

En la siguiente imagen, figura 14.5, podemos ver este ejemplo.

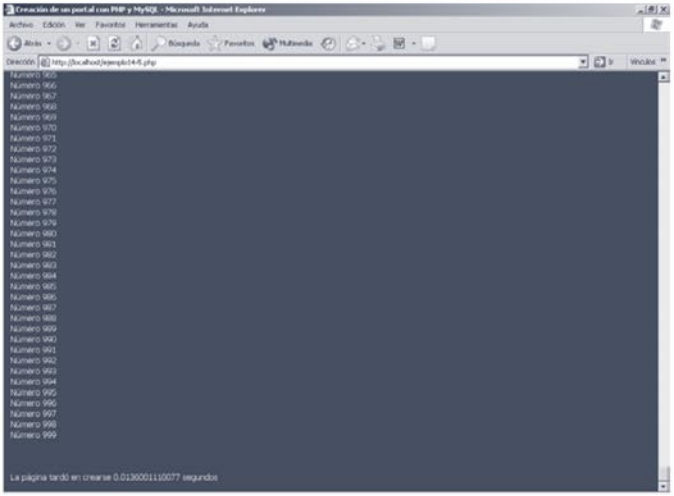

Figura 14.5

14.4 FRASES ALEATORIAS AL RECIBIR A LOS USUARIOS

En este apartado vamos a crear una aplicación para insertar frases aleatorias en nuestra web. Podemos crear varias frases representativas, crear frases a modo de anuncio, o emplearlo como sea necesario.

Lo que hacemos con este ejemplo es que, cada vez que se carga esta página, muestra una frase entre todas las que tengamos creadas. La carga de estas frases se hará de forma aleatoria, ya que para esto empleamos una nueva función, la función *rand ( )*, que lo

que realiza es que nos da es un número aleatorio entre un intervalo que previamente le indiquemos.

14.4.1 Ejemplo

En el ejemplo que hemos creado a continuación, en vez de una frase se muestran tres. El motivo es ver el funcionamiento de la aplicación, así como poder comprobar que se puede repetir la misma frase, ya que como indicamos antes los números son aleatorios y en ocasiones se pueden repetir.

```
<html>
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<?
$frases = array (
1=> "Hola navegante.",
2=> "Bienvenido a mi web.",
3=> "Gracias por visitarnos.",
4=> "Te recomiendo visitar nuestro foro.",
5=> "Puedes enviarnos las sugerencias que quieras.",
6=> "No dejes de visitarnos estos días, tendremos nuevas sorpresas.",
);
```

```
$aleatorio = rand (1,6);
echo "$frases[$aleatorio]";
echo "<br><br>";
$aleatorio = rand (1,6);
echo "$frases[$aleatorio]";
echo "<br><br>";
$aleatorio = rand (1,6);
echo "$frases[$aleatorio]";
?>
```

En la siguiente imagen, figura 14.6, podemos ver este ejemplo.

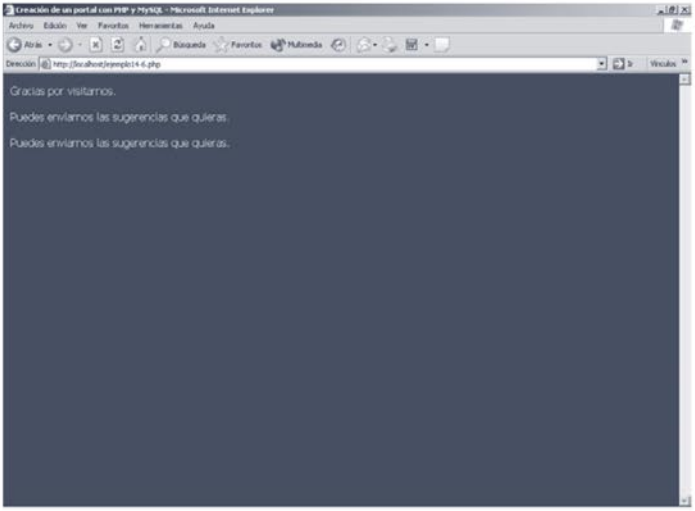

Figura 14.6

14.5 RECOMENDAR NUESTRA WEB A UN AMIGO

A continuación, en este capítulo, vamos a explicar cómo realizar una aplicación para que los usuarios de nuestra web se la recomienden a otro usuarios. Como se puede entender, esta aplicación es de lo más útil, ya que como bien sabemos uno de los canales más importantes para la promoción de nuestra web son los mismos usuarios, es decir, si un usuario visita nuestra web y le parece interesante en cuanto a diseño o contenidos, seguro que, en primer lugar, volverá a visitarnos y, en segundo lugar, seguro que se la recomendará a sus conocidos.

Para la realización de esta pequeña aplicación, nos encontramos con una nueva instrucción de PHP, la función **mail ( )**, muy útil para enviar correos electrónicos sin necesidad de disponer de un gestor de correo electrónico.

La función **mail ( )** tiene que estar formada por tres partes principalmente, que serán la dirección, el asunto del mensaje y el mensaje, pero además de estas se pueden añadir otras.

La sintaxis es: mail (<emaildestino>,<asunto>, <mensaje>, [otros]);.

14.5.1 Ejemplo

A continuación, vamos a realizar el ejemplo para que los usuarios recomienden nuestra web. Algo muy importante que debe saber el lector de este libro es que al realizar esta aplicación en modo local no verá el resultado de la misma, ya que trabajando en local no recibiremos e-mail, a no ser que tengamos instalado en nuestro ordenador un servidor de correo electrónico. Por lo que para probarlo correctamente existen dos opciones. Una de ellas sería instalar un servidor de correo electrónico o probar este ejemplo en un servidor alojado en Internet, con el cual sí recibiríamos los correos para comprobarlo.

Para realizar este ejemplo, debemos crear dos ficheros: el primero de ellos será un formulario HTML, donde el usuario introduce los datos, y el otro será un fichero PHP, que procesa los datos recibidos en el formulario para enviar el correo electrónico. Al formulario lo llamaremos **ejemplo14-7** y al fichero PHP que procesa los datos lo llamaremos **recomendar.php**.

Primero creamos el formulario:

```
<html>
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<form method="POST" action="recomendar.php">
<p>
<h2>
```

Recomienda este sitio a un amigo:

```
</h2>
</p>
<p>

Tu nombre:
<br>
<input type="text" name="tunombre" size="20">
<br>
Tu email:
<br>
<input type="text" name="tuemail" size="20">
<br>

Nombre de tu amigo:
<br>
<input type="text" name="nombreamigo" size="20">
<br>

Email de tu amigo:
<br>
<input type="text" name="emailamigo" size= "20">
<br>
<br>
<input type="submit" value="Recomienda" name="B1">
</p>
</form>
```

En la figura 14.7 vemos el formulario:

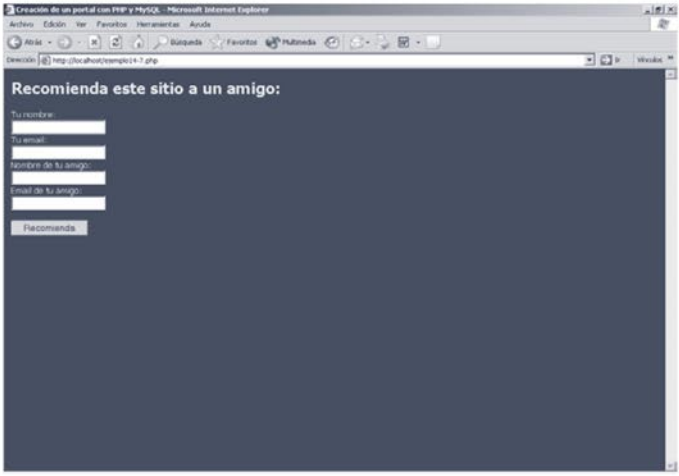

Figura 14.7

A continuación, vemos el código del fichero que procesa los datos recibidos del formulario, **redomendar.php**:

```
<html>
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
```

```
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<?
$asunto = "Te recomiendo visitar este portal.";
$mensaje = "Hola ".$nombreamigo.", soy ".$tunombre." Y te recomiendo visitar. www.tudominio.com, un portal en el que podrás encontrar información muy interesante, un foro muy sencillo y con muchísima información. Espero que lo visites pronto, ya que estoy convencido que te va a gustar. Por cierto, si es de tu agrado, no dejes de firmar el libro de visitas.";
mail($emailamigo, $asunto, $mensaje, "From: ".$tuemail);
?>
```

Mediante este ejemplo, el usuario cuyo e-mail introduzcamos en el formulario recibirá un mensaje con el título: *Te recomiendo visitar este portal*, indicándole quién es la persona que se lo envía, así como el texto del mensaje, que será el siguiente: *Hola "nombredelamigo", soy "aquitunombre". Y te recomiendo visitar www.tudominio.com, un portal en el que podrás encontrar una información muy interesante, un foro muy sencillo y con muchísima información. Espero que lo visites pronto, ya que estoy convencido de que te va a gustar. Por cierto, si es de tu agrado, no dejes de firmar el libro de visitas.*

14.6 CAMBIAR UNA IMAGEN SEGÚN EL DÍA DE LA SEMANA

A continuación, con el siguiente ejemplo, vamos a desarrollar una sencilla aplicación con la que vamos a poder insertar una imagen para cada día de la semana.

14.6.1 Ejemplo

En este sencillo ejemplo volvemos a hacer uso de la función *date ( )*. Lo único necesario es tener creada una imagen para cada día

de la semana en inglés, ya que la función *date ( )* maneja el idioma inglés Los nombres serían: Lunes=monday.gif, Martes= tuesday.gif, Miércoles=wednesday.gif, Jueves=thursday.gif, Viernes=friday.gif, Sábado=saturday.gif, Domingo=sunday.gif.

```
<html>
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<body leftmargin = "300">
<body topmargin = "150">
<?
$fecha = date ("l");
$fecha = $fecha.".gif";
echo "<img src = \"$fecha\">";
?>
```

En la figura 14.8 podemos ver el resultado de este ejemplo.

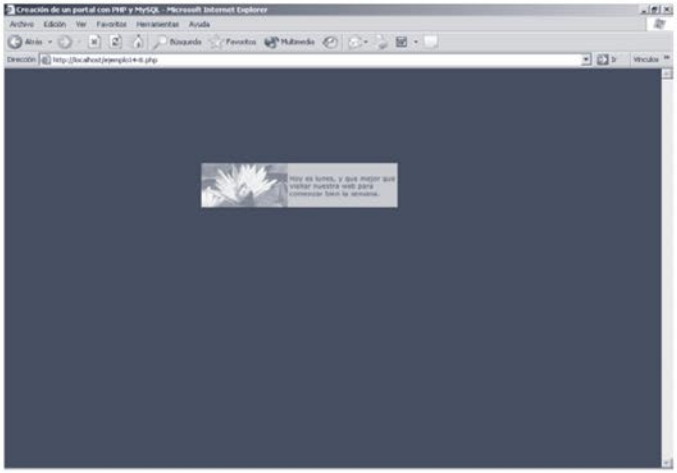

Figura 14.8

En este ejemplo obtenemos una imagen que se mostrará según el día de la semana.

Este ejemplo podríamos utilizarlo creando un formato de imagen como si fueran banner. Por ejemplo, si nuestro portal fuese una tienda, cada día de la semana podríamos mostrar un banner que una oferta o promoción diaria.

14.7 PROTEGER PÁGINAS CON CONTRASEÑA

Con esta pequeña y sencilla aplicación conseguimos crear una página protegida con un nombre de usuario y contraseña, para que sólo accedan a ella usuarios que conozcan estos datos.

14.7.1 Ejemplo

En primer lugar, creamos un pequeño formulario que recoje los datos del usuario, para posteriormente enviarlos a la página **comprueba.php**, que es la que verifica que el usuario y la contraseña son correctos. Si son correctos, muestra un mensaje de bienvenida; si no lo son, muestra un mensaje de error.

```
<html>
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<body leftmargin = "300">
<body topmargin = "150">
<table border="0">
<form method="POST" action="comprueba.php">
<tr><td>Usuario: </td>
<td><input type="text" name="usuario" size="20"></td></tr>
<tr>
<td>Contraseña: </td>
```

```
<td><input type="password" name="pass" size="20"></td> </tr>
<tr>
<td>
<input type="submit" value="Enviar" name="privado">
</td>
</tr>
</table>
```

En la siguiente imagen, figura 14.9, podemos ver el formulario principal.

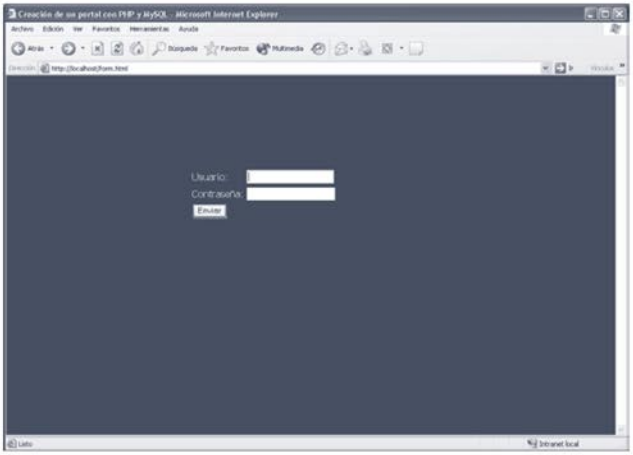

Figura 14.9

Este es el código correspondiente al fichero **comprueba.php** que se encargará de realizar la verificación de los datos recibidos por el formulario.

```
<html>
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<body leftmargin = "300">
<body topmargin = "150">
<?
if ($usuario=="tuusuario" && $pass=="tuclave"){
$valido="si";
} else {
$valido="no";}
if ($valido=="si"){
?>
Bienvenido, has introducido correctamente el usuario y la contraseña.
<?
} else {
?>
Vuelve a intentarlo, el usuario o la contraseña son incorrectos.
<? } ?>
```

Capítulo 15

APLICACIONES MUY ÚTILES PARA NUESTRA WEB

15.1 CREACIÓN DE UN FORO

El foro es una aplicación casi imprescindible hoy en día en cualquier portal que nos encontremos. En él los usuarios podrán exponer sus dudas para que expertos en alguna materia o nosotros mismos (el *webmaster*) podamos resolverlas. A su vez, también puede ser utilizado a modo de tablón de anuncios.

Si navegamos por Internet, comprobaremos que ya hoy en día cualquier portal dispone de su propio foro, e incluso algunos disponen de varios.

En el siguiente ejemplo vamos a mostrar cómo se crea un sencillo foro, pero totalmente operativo y con todas las funciones necesarias.

15.1.1 Ejemplo

Para poder crear el foro, lo primero que debemos hacer es crear una tabla y después necesitaremos, además cuatro ficheros.

Lo primero que vamos a hacer es crear la base de datos que llamaremos **foro** y la tabla para el foro. Podemos crear diferentes foros, que podemos ir llamando sucesivamente **foro2**, **foro3**, etc., empleando una única base de datos y creando simplemente nuevas tablas de igual forma que la que vamos a crear a continuación.

Lo primero que vamos a hacer es en phpMyAdmin crear la base de datos **foro** y la tabla **foro1**, que contendrá los campos id, autor, título mensaje, fecha, respuestas e identificador.

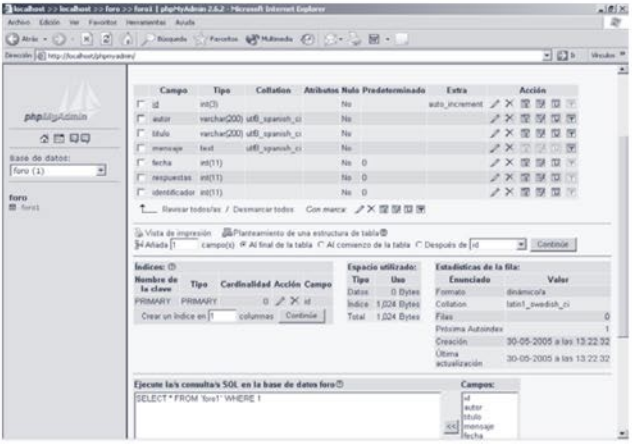

Figura 15.1

El primer fichero que vamos a crear en este ejemplo lo llamaremos **indexforo.php**. Y será la página principal del foro. En ella veremos todos los mensajes que existen ordenados según su fecha de creación, así como el número de respuestas que contienen.

Por último, pondremos un enlace para añadir nuevos mensajes. Al lado de cada título habrá un link con la palabra *Ver*.

Fichero indexforo.php.

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<font size="4">
<u>Foro del portal de coches </u>
</font></p>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
<br><br>
<tr>
<td width="5%"></td>
<td width="35%">
<b>TITULO</b>
</td>
<td width="30%">
<b>FECHA</b>
</td>
<td width="30%">
```

```
<b>RESPUESTAS</b>
</td></tr></table>
<?
$host="127.0.0.1";
$user="usuario";  // Poner aquí nuestro nombre de usuario
$password="pass";  // Poner aquí nuestra contraseña
$db="foro";
$connect=mysql_connect($host,$user,$password);
mysql_select_db("foro", $connect);
$consulta = mysql_query("SELECT * from foro1 WHERE identificador = 0 ORDER BY fecha DESC",$connect);
$lado=mysql_num_rows($consulta);
echo "<hr size = 10 color = ffffff width = 100% align = left>";
while($row = mysql_fetch_array($consulta)) {
$titulo= $row ["titulo"];
$id=$row["id"];
$titulo=$row["titulo"];
$fecha=$row["fecha"];
$respuestas=$row["respuestas"];
echo("<table width='100%' border='0' cellspacing='0' cellpadding= '0'>\n");
echo("<tr>\n");
echo("<td width='5%'><a href= foroforo.php?id=$id> Ver</a></td>\n");
echo("<td width='30%'>$titulo</a></td>\n");
echo("<td width='30%'>". date("d-m-y",$fecha)."</td>\n");
echo("<td width='30%'>$respuestas</td>\n");
echo("</tr>\n");
```

```
echo("</table>\n");
echo "<hr size = 2 color = ffffff width = 100% align = left>";
}
?>
<br><p align = "center">
<font face="arial" size="1">
<a href= "formularioforo.php?respuestas=0">
Añadir mensaje</a></p>
</font>
```

En la figura 15.2 podemos ver la imagen correspondiente a este fichero, que es la página principal del foro, donde podemos ver las cabeceras de los mensajes del foro.

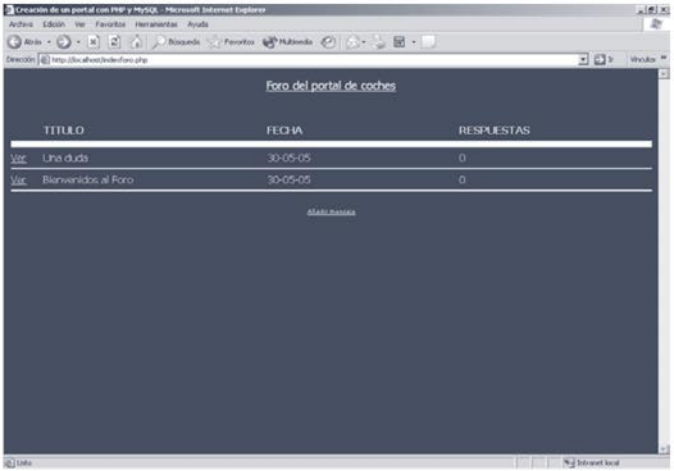

Figura 15.2

El siguiente fichero será **formularioforo.php**, un pequeño formulario en el que los usuarios podrán insertar nuevos mensajes o respuestas a algunos mensajes ya existentes en el foro. Será un sencillo fichero con código HTML con un pequeño formulario.

Fichero formularioforo.php.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align="center">
<font size="4">
<u>Formulario para insertar un mensaje en el foro</u>
</font>
</p>
<form action="addforo.php">
<input type="hidden" name="respuestas" value="<? echo $respuestas;?>">
<input type="hidden" name="identificador" value="<? echo $id; ?>">
```

```
AUTOR:<input type="text" name="autor" size ="25">
<br><br>
TITULO:<input type="text" name="titulo" size="25">
<br>
<br>
MENSAJE: <textarea name="mensaje">
</textarea>
<br>
<br>
<input type=submit value="Enviar">
</form>
```

El siguiente fichero, **addforo.php**, permite ver el fichero anterior, el cual nos remite cuando pulsamos el botón *Enviar*. Su función es reunir los datos que introducen los usuarios en el formulario para insertarlos en la base de datos y así poder visualizar los mensajes o respuestas en el foro.

Para responder a algún mensaje ya publicado del foro se emplea este fichero y el anterior con dos funciones diferentes para nuestra página: insertar nuevos mensajes o insertar respuestas.

En la siguiente imagen, figura 15.3, podemos ver el resultado de ejecutar el fichero **formularioforo.php**, que como podemos ver simplemente es un formulario donde insertar los datos para que se muestre el mensaje.

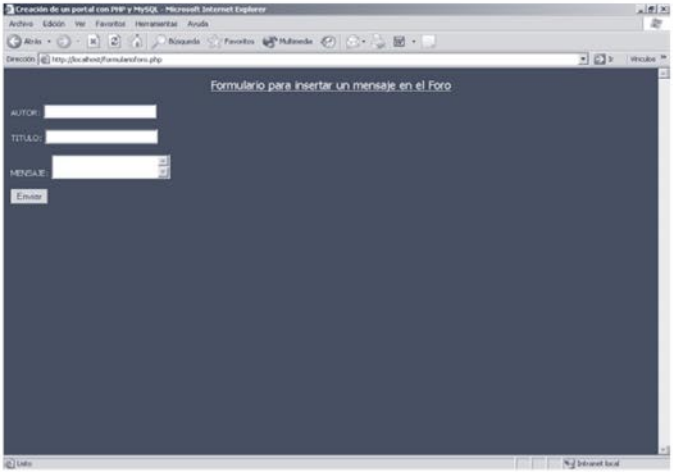

Figura 15.3

Fichero addforo.php.

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<font size="2">
<?
$host="127.0.0.1";
```

```
$user="usuario";  // Poner aquí nuestro nombre de usuario
$password="pass";  // Poner aquí nuestra contraseña
$db="foro";
$enlace = mysql_connect($host, $user, $password);
mysql_select_db($db,$enlace);
$fecha = time();
if(empty($identificador))
{ $identificador=0; }
$respuesta = $respuestas+1;
$sql = "INSERT INTO foro1 (autor, titulo, mensaje, fecha, identificador) VALUES ('$autor', '$titulo', '$mensaje', '$fecha', '$identificador') ";
mysql_query($sql);
$sql2 ="UPDATE foro1 SET respuestas = '$respuesta' WHERE id = '$identificador'";
mysql_query($sql2);
$resultado=mysql_query("SELECT '$mensaje' FROM foro1 WHERE mensaje='$mensaje'",$enlace);
while ($registro = mysql_fetch_row($resultado))
{
echo "<tr>";
foreach($registro as $clave)
{
echo "<td>",$clave,"</td>";
}
}
echo "<br><br>";
echo "<a href=indexforo.php>Volver al foro</a> </font> </center>";
?>
```

Para terminar con este ejemplo del foro, el último fichero que necesitamos lo llamaremos **foroforo.php** y será el encargado de mostrarnos el contenido de los mensajes y respuestas. Cada vez que pulsemos en un mensaje para visualizarlo, podremos ver el contenido del mismo y el de las respuestas si las tuviera.

En esta imagen vemos el resultado que se obtiene cada vez que insertamos un mensaje en el foro, figura 15.4

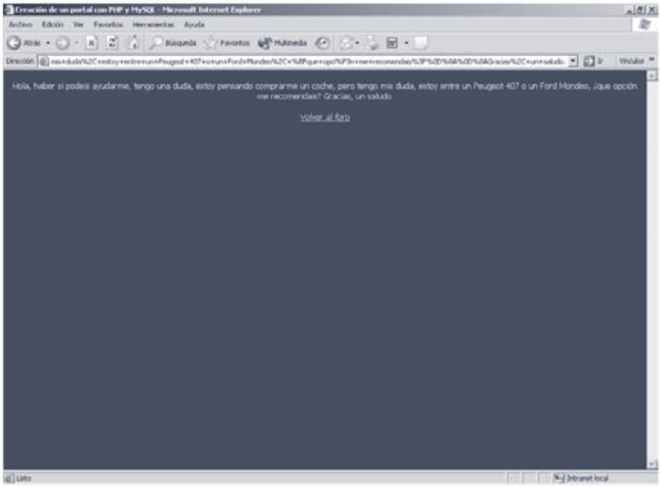

Figura 15.4

Fichero foroforo.php.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
```

```
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<font size="4">
<u>Leyendo mensaje del Foro</u>
</font></p>
<?
$host="127.0.0.1";
$user="usuario";   // Poner aquí nuestro nombre de usuario
$password="pass";   // Poner aquí nuestra contraseña
$db="foro";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$consulta = mysql_query("SELECT * FROM foro1 WHERE id=
'$id' ORDER BY fecha DESC",$enlace);
while($row = mysql_fetch_array($consulta)) {
$titulo=$row["titulo"];
$autor=$row["autor"];
$mensaje=$row["mensaje"];
$id=$row["id"];
$fecha=$row["fecha"];
$respuestas=$row["respuestas"];
echo "<table><tr><td>TITULO: $titulo</td><tr>";
```

```
echo "<td>AUTOR: $autor</td></tr>";
echo "<tr><td>$mensaje</td></tr></table>";
echo "<center><font face=arial size=1>";
echo "<a href=formularioforo.php?id= $id&respuestas=
$respuestas>";
echo "<br><br>";
echo "Añadir mensaje</a> ";
echo "<a href=indexforo.php>Volver al foro</a></font>
</center>";
}
$consulta2 = mysql_query("SELECT * FROM foro1 WHERE
identificador = '$id' ORDER BY fecha DESC",$enlace);
echo "RESPUESTAS:<br><hr>";
while($row = mysql_fetch_array($consulta2)){
$titulo=$row['titulo'];
$autor=$row['autor'];
$mensaje=$row['mensaje'];
$id=$row['id'];
$fecha=$row['fecha'];
$respuestas=$row['respuestas'];
echo "<table><tr><td>TITULO: $titulo</td></tr>";
echo "<tr><td>AUTOR: $autor</td></tr>";
echo "<tr><td>MENSAJE: $mensaje</td></tr></table>";
} ?>
```

En la siguiente imagen, figura 15.5, podemos ver el resultado de seleccionar un mensaje del foro. A su vez, y si existen, se visualizarán las respuestas que contenga el mismo mensaje.

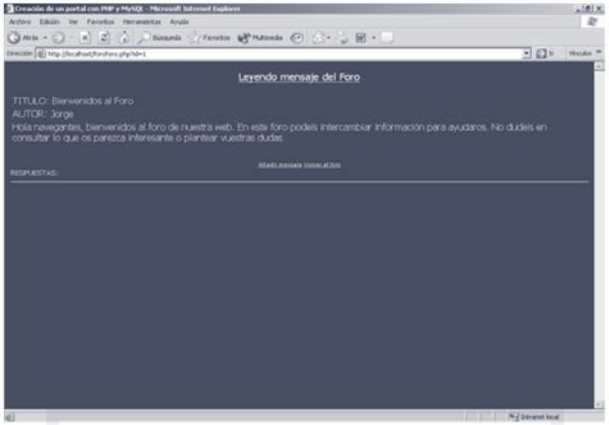

Figura 15.5

15.1.2 phpBB

Tras haber creado nuestro primer foro, el cual podemos mejorar en muchos aspectos, principalmente en cuanto a diseño, podemos comprobar que existen algunas alternativas desarrolladas por algunos miembros de la comunidad de desarrolladores de PHP. Se trata de la aplicación phpBB.

¿Qué es phpBB? Es una completa aplicación para la gestión dinámica de foros. Cabe destacar, además, que gracias a su potencia y versatilidad, phpBB es una de las aplicaciones para foros más utilizadas en Internet.

¿Qué ganamos al emplear phpBB? Gracias a la utilización de phpBB obtendremos una serie de ventajas, como puede ser control absoluto sobre la utilización incorrecta del mismo por parte de usuarios, como por ejemplo el envío de SPAM a los foros. Gracias a los moderadores, estas actividades pueden ser detectadas con suficiente facilidad. Otra de las grandes ventajas es que disponemos

de amplias posibilidades en su administración, ya que se trata de una aplicación bastante completa.

La web oficial de esta aplicación es *http://www.phpbb.com.* Desde esta misma web podemos encontrar toda la información para utilizar esta aplicación en nuestra web. Si accedemos a la sección **download**, podremos descargar la aplicación para poder utilizarla en nuestra web.

Una vez lo hayamos descargado, descomprimimos el archivo en la misma carpeta que utilizamos para almacenar nuestro archivos de prueba.

El siguiente paso será abrir en nuestro navegador la dirección *http://localhost*, donde podemos ver que aparece una nueva carpeta, en este caso **phpBB2**. Pulsamos sobre ella para acceder a la aplicación y veremos que lo primero que tenemos que hacer es configurar la aplicación para poder utilizarla en nuestra web. Podemos ver la pantalla inicial en la siguiente imagen, figura 15.6.

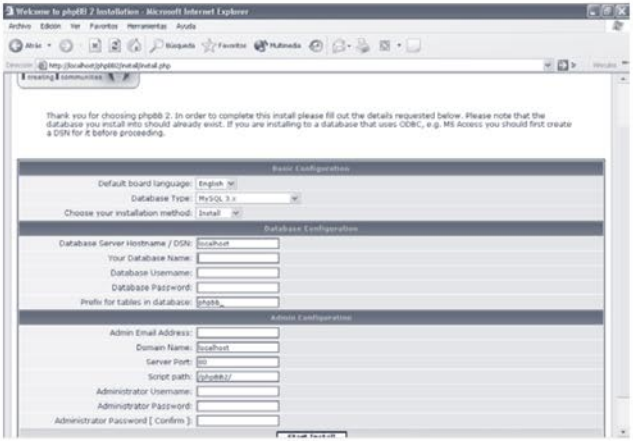

Figura 15.6

En esta página inicial de nuestro foro phpBB, debemos de configurarlo de la siguiente forma:

- En el primer campo del formulario, el idioma lo dejamos como está, ya que no existe otra opción.

- En el campo **Database Type**, seleccionaremos MySQL 4.x/5.x, que es el que estamos utilizando a lo largo de este libro.

- El campo **Installation** lo dejamos como aparece, con la opción *Install*.

- El siguiente paso es configurar la base de datos. El nombre del *host* lo dejaremos como aparece: **localhost**. Los siguientes pasos serán indicar el nombre de la base de datos que queramos darle, en este caso le pondremos **forophpbb**, y también indicarle el nombre de usuario y la contraseña que empleamos para acceder a la base de datos. Después debemos también de indicar el prefijo que irá delante y con el que queremos que se nombren las tablas de la base de datos que vamos a crear.

- Por último, configuramos los datos necesarios para el Administrador del foro. En primer lugar, le indicamos el e-mail del administrador, pero el *host*, el puerto y el *path* lo dejamos como aparece por defecto. Para finalizar sólo debemos indicar un nombre de usuario y una contraseña para el acceso como administradores al foro.

Una vez hayamos completado los datos, comenzaremos la instalación pulsando en *Start Install*. Cuando haya terminado el proceso, estaremos en disposición de poder empezar a utilizar nuestro nuevo foro phpBB.

15.2 CREACIÓN DE UN LIBRO DE VISITAS

Otra parte importante en la creación de un portal será un libro de visitas, ya que en él los usuarios podrán dejar "su huella" con comentarios acerca de nuestro portal, como pueden ser felicitaciones o críticas, aunque lo ideal es que estas sean menos, mejoras que podríamos realizar, comentarios acerca del diseño o la utilidad que le han encontrado a nuestro portal, etc.

La forma de crear el libro de visitas será muy parecida a la de crear el foro, pero con pequeñas diferencias, ya que, por ejemplo, en este caso no tendremos respuestas a cada mensaje del libro de visitas, ya que en él, el usuario sólo escribe una vez y su mensaje no ha de tener respuesta.

15.2.1 Ejemplo

Necesitaremos crear una base de datos con una tabla y, además, los mismos cuatro ficheros que para el foro, pero con pequeñas variaciones.

La base de datos que crearemos se llamará **libro** y la tabla **libro1**. Para crear esta tabla seguiremos los mismos pasos que en el ejemplo del foro, sólo cambian los nombres y que no debemos crear el campo respuestas.

El primer fichero que crearemos se llamará **indexlibro.php**. Será la página principal del libro de visitas y en ella veremos todos los mensajes ordenados según su antigüedad y con la fecha de creación.

Al igual que en el foro, junto al título de cada mensaje pondremos un enlace con la palabra **Ver** donde podremos pulsar para leer el mensaje del libro de visitas que haya puesto el visitante.

Fichero indexlibro.php.

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<font size="4">
<u>Libro de Visitas </u>
</font>
</p>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
<br><br><tr>
<td width="5%">
</td>
<td width="55%">
<b>TITULO</b>
</td>
<td width="40%">
<b>FECHA</b>
</td></tr>
</table>
<?
$host="127.0.0.1";
```

```
$user="user";    // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="libro";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$consulta = mysql_query("SELECT * from libro1 WHERE identificador = 0 ORDER BY fecha DESC",$enlace);
$lado=mysql_num_rows($consulta);
echo "<hr size = 10 color = ffffff width = 100% align = left>";
while($row = mysql_fetch_array($consulta))
{
$titulo= $row ["titulo"];
$id=$row["id"]; $titulo=$row["titulo"]; $fecha=$row["fecha"];
$respuestas=$row["respuestas"];
echo("<table width='100%' border='0' cellspacing='0' cellpadding= '0'>\n");
echo("<tr>\n");
echo("<td width='5%'><a href= libro.php?id=$id>Ver</a></td>\n");
echo("<td width='55%'>$titulo</a></td>\n");
echo("<td width='40%'>". date("d-m-y",$fecha)."</td>\n");
echo("</tr>\n");
echo("</table>\n");
echo "<hr size = 2 color = ffffff width = 100% align = left>";
}
?>
<font face="arial" size="1">
<a href= "formulariolibro.php">Añadir mensaje
</a></font>
```

Al siguiente fichero lo vamos a llamar **formulariolibro.php**, y será un pequeño formulario en el que los usuarios que accedan a él podrán insertar los mensajes que deseen en el libro de visitas.

En la siguiente imagen, figura 15.7, podemos ver el resultado de ejecutar el fichero **indexlibro.php**, que será la página principal del libro de visitas.

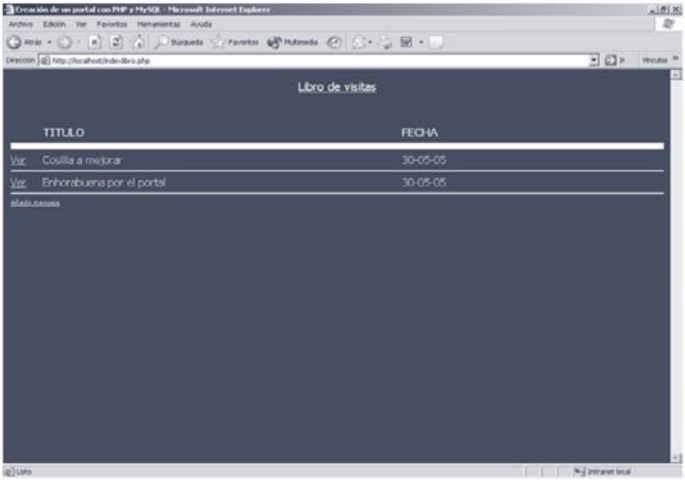

Figura 15.7

Fichero formulariolibro.php.

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
```

```
<body bgcolor = "#303030"><body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align="center"><font size="4">
<u>
Formulario para insertar un mensaje en el Libro de Visitas
</u>
</font>
</p>
<form action="addlibro.php">
<input type="hidden" name="identificador" value="<? echo $id; ?>">
AUTOR: <input type="text" name="autor" size ="25">
<br>
<br>
TITULO: <input type="text" name="titulo" size ="25">
<br>
<br>
MENSAJE: <textarea name="mensaje">
</textarea>
<br>
<br>
<input type=submit value="Enviar">
</form>
```

En esta imagen, figura 15.8, se muestra el formulario para insertar mensajes en el libro de visitas.

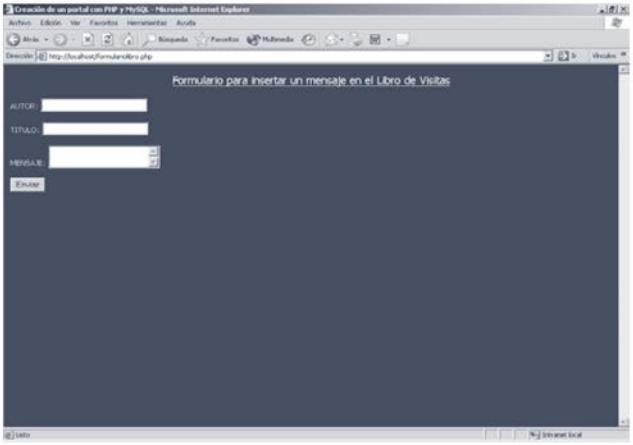

Figura 15.8

El siguiente fichero de obligada creación es **addlibro.php**, este será el encargado de procesar el mensaje para que lo insertemos a la base de datos y que quede almacenado para poder visualizarlo con el resto.

Fichero addlibro.php.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
```

```
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<?
$host="127.0.0.1";
$user="user";   // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="libro";
$enlace = mysql_connect($host, $user, $password);
mysql_select_db($db,$enlace);
$fecha = time();
if(empty($identificador))
{
$identificador=0;
}
$respuesta = $respuestas+1;
$sql = "INSERT INTO libro1 (autor, titulo, mensaje, fecha, identificador) VALUES ('$autor', '$titulo', '$mensaje', '$fecha', '$identificador') ";
mysql_query($sql);
$resultado=mysql_query("SELECT '$mensaje' FROM libro1 WHERE mensaje='$mensaje'",$enlace);
while ($registro = mysql_fetch_row($resultado))
{
echo "<tr>";
foreach($registro as $clave)
{
```

```
echo "<td>",$clave,"</td>";
}
}
echo "<br>";
echo "<br>";
echo "<a href=indexlibro.php> Volver a la página principal</a></font> </center>";
?>
```

Para terminar con el libro de visitas, el último fichero que necesitamos es **librolibro.php**, que será el encargado de mostrarnos el contenido de los mensajes del libro que queramos visualizar.

Cada vez que pulsemos en un mensaje para visualizarlo, podremos ver su contenido. Para lo que necesitaremos el siguiente fichero **librolibro.php**.

Fichero librolibro.php.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<font size="4">
<u>
```

```
Leyendo mensaje del Libro de Visitas
</u>
</font>
</p>
<?
$host="127.0.0.1";
$user="user";   //Poner aquí nuestro nombre de usuario
$password="pass";   //Poner aquí nuestra contraseña
$db="libro";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$consulta = mysql_query("SELECT * FROM libro1 WHERE id='$id' ORDER BY fecha DESC",$enlace);
while($row = mysql_fetch_array($consulta))
{
$titulo=$row["titulo"];
$autor=$row["autor"];
$mensaje=$row["mensaje"];
$id=$row["id"];
$fecha=$row["fecha"];
echo "<table><tr><td>TITULO: $titulo</td><tr>";
echo "<td>AUTOR: $autor</td></tr>";
echo "<tr><td>$mensaje</td></tr></table>";
echo "<center><font face=arial size=1>";
echo "<br><br>";
echo "<a href=indexlibro.php>Volver al foro</a></font>
</center>";
}
?>
```

15.3 FORMULARIO DE CONTACTO

A continuación, en este ejemplo, vamos a mostrar cómo realizar un formulario de contacto con varios campos donde todos estos serán enviados al correo electrónico del *webmaster*.

15.3.1 Ejemplo

Lo primero será crear el formulario de contacto, al que llamaremos **form.html**, y cuyo código es:

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<STRONG>FORMULARIO DE CONTACTO</STRONG>
<br><br>
Inserte los datos que a continuación se solicitan y en breve nos pondremos en contacto con usted.
<form name="contacto" method="post" action="contacto.php">
<table width="90%" border="2" align="center" bordercolor="#cccccc">
<tr>
<td width="15%"><p><strong>NOMBRE:</strong></p></td>
<td colspan="3"><input name="nombre" type="text" value="" size="109">
```

```
</td>
</tr>
<tr>
<td><p><strong>APELLIDOS:</strong></p></td>
<td colspan="3"><input name="apellidos" type="text"
size="109">
</td>
</tr>
<tr>
<td><p><strong>DIRECCION:</strong></p></td>
<td colspan="3"><input name="direccion" type="text"
size="109">
</td>
</tr>
<tr>
<td width="15%"><p>
<strong>LOCALIDAD:</strong>
</p></td>
<td width="25%">
<input name="localidad" type="text" size="36">
</td>
<td width="15%"><p>
<strong>PROVINCIA:</strong>
</p>
</td>
<td width="25%">
<input name="provincia" type="text" size="36">
</td>
```

```
</tr>
<tr>
<td>
<p>
<strong>TELEFONO:
</strong>
</p>
</td>
<td>
<input name="telefono" type="text" size="36">
</td>
<td>
<p>
<strong>E-MAIL:
</strong>
</p>
</td>
<td>
<input name="email" type="text" size="36">
</td>
</tr>
</table>
<p align="center">
<input type="submit" name="Submit" value="Enviar datos">
</p>
</form>
```

En la siguiente imagen, figura 15.9, podemos ver el formulario de contacto.

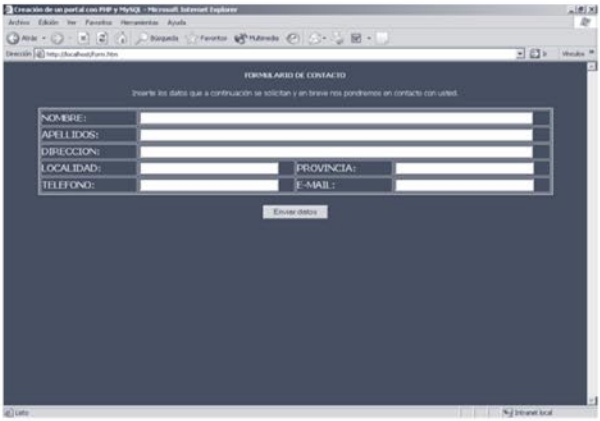

Figura 15.9

El siguiente paso es crear el fichero que reune los datos del formulario para enviarlos al correo electrónico del *webmaster* (o al correo electrónico que nos sea conveniente). Este fichero se llamará **contacto.php**, como ya indicamos en el formulario de contacto, y este será el código:

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
```

```
<font size = "2">

<body link = "#E5E5E5" vlink ="E0E0E0">

<p align = "center">

<STRONG>

Su mensaje ha sido enviado. En breve contactaremos con usted.
Gracias.

</STRONG>

<?

$fecha=date("d-m-Y");

$hora=date("H:i");

$destinatario="tucorreo@tucorreo.com";

/* El correo electrónico es un valor que nunca se va a modificar, es
decir, los correos que nos envíen siempre llegarán a la dirección de
correo electrónico que aquí pongamos. Por eso hay que prestar
mucha atención si algún día modificamos nuestra dirección de
correo electrónico. De cambiarla, en este código también debe
hacerse, porque de lo contrario cada vez que nos manden algún
correo no lo recibiremos.*/

$asunto="Contacto de cliente";

echo "<br><br><br>";

echo "Compruebe si sus datos son correctos, de lo contrario pinche
en <a href=http://localhost/form.htm>Volver</a>";

$texto="Nombre:"."\n".$nombre."<br>"."Apellidos:"."\n".$apellido
s."<br>"."Dirección:"."\n".$direccion."<br>"."Localidad:"."\n".$lo
calidad."<br>"."Provincia:"."\n".$provincia."<br>"."Teléfono:"."\
n".$telefono."<br>"."Email:"."\n".$email."<br>"."Fecha:"."\n".$fe
cha."<br>"."Hora:"."\n".$hora;

/* Lo que hacemos con la variable $texto, es unir dentro de ella
todos los campos que recibimos del formulario para poder enviar
```

*con la función mail ( ) en una sola variable todo el contenido con los datos de contacto. */*

```
echo "<br><br><br>";

echo $texto;

echo "<br><br>";

mail($destinatario,$asunto,$texto);

?>
```

En la siguiente imagen, figura 15.10, podemos ver el resultado en pantalla cuando el usuario ha enviado los datos. Como se puede observar, el usuario verá en pantalla todos sus datos y, si estos no son correctos, podrá volver de nuevo al formulario para corregirlos.

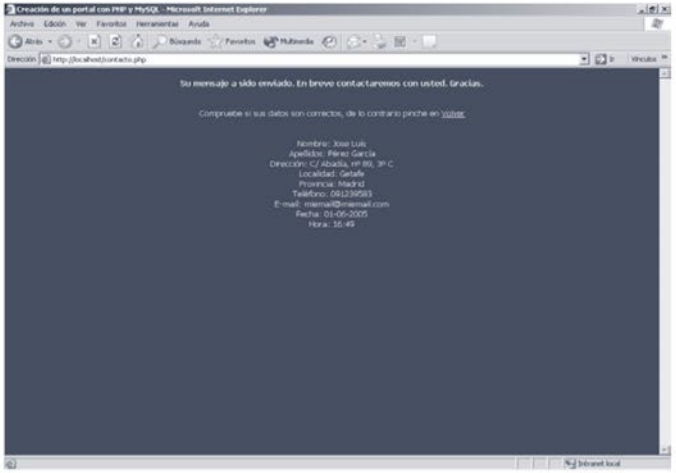

Figura 15.10

15.4 REGISTRO Y RECONOCIMIENTO DE USUARIOS

En este apartado vamos a desarrollar un ejemplo de una zona de registro de usuarios y también una zona donde identificarse. Estas aplicaciones las utilizan muchos portales para captar usuarios y ofrecerles servicios exclusivos.

15.4.1 Ejemplo

A continuación, empezamos creando un pequeño formulario para que los usuarios se registren en nuestra web, al que llamaremos **registro.htm**, y cuyo código será:

```
<head>

<title>Creación de un portal con PHP y MySQL

</title>

</head>

<body bgcolor = "#303030">

<body text = "#E5E5E5">

<font face = "tahoma">

<font size = "2">

<body link = "#E5E5E5" vlink ="E0E0E0">

<p align = "center">

<STRONG>FORMULARIO DE REGISTRO DE USUARIOS</STRONG>

<br><br>

Inserte los datos que a continuación se solicitan y en breve nos pondremos en contacto con usted.
```

```
<form name="contacto" method="post" action="registro.php">
<p align = "left">
<strong>NOMBRE:</strong>
<br>
<input name="nombre" type="text" value="" size="50">
<br><br>
<strong>APELLIDOS:</strong>
<br>
<input name="apellidos" type="text" size="50">
<br><br>
<strong>NOMBRE DE USUARIO:</strong>
<br>
<input name="usuario" type="text" size="50">
<br><br>
<strong>CONTRASEÑA:</strong>
<br>
<input name="cont" type="password" size="50">
<br><br>
<strong>E-MAIL:</strong>
<br>
<input name="email" type="text" size="50">
<br>
```

```
<br>
<input type="submit" name="Submit" value="Enviar datos">
</p>
</form>
```

Y este será el aspecto que tendrá el formulario de registro.

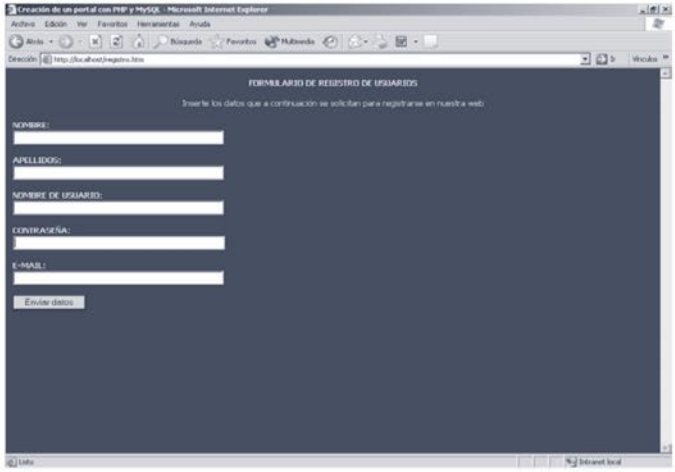

Figura 15.11

El siguiente paso para que el usuario quede registrado en nuestra web es crear una base de datos. En ella se almacenarán todos los datos del usuario para que acceda a nuestra web con sólo poner el nombre de usuario y la contraseña.

Creamos una base de datos que contenga una tabla con los campos id, nombre, apellidos, usuario, contraseña y e-mail a la que llamamos **registrados** y creamos dentro la tabla **usuarios**. En la siguiente imagen podemos ver la base de datos creada, figura 15.12.

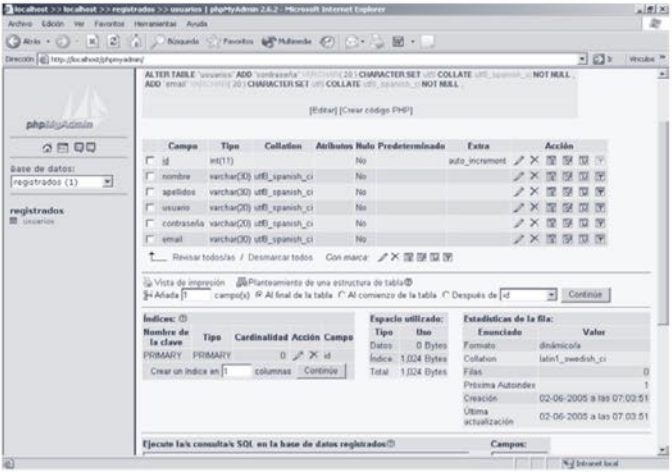

Figura 15.12

El siguiente paso es crear el fichero **registro.php**, que será el encargado de recibir los datos del formulario e insertarlos en la base de datos, para que el usuario quede registrado. A continuación, se muestra el código del fichero **registro.php**:

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
```

```
<STRONG>SU REGISTRO SE HA COMPLETADO CON ÉXITO</STRONG>
<br>
<br>
<?
$host="127.0.0.1";
$user="user";    // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="registrados";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$consulta = mysql_query("insert into usuarios (nombre,apellidos,usuario,contraseña,email) values ('$nombre', '$apellidos', '$usuario', '$cont', '$email')",$enlace);
echo "<hr size = 10 color = ffffff width = 100% align = left>";
echo "<STRONG>Bienvenido a nuestra web $nombre</STRONG>";
?>
```

Por último, sólo quedaría crear un fichero para que los usuarios que estén ya registrados puedan identificarse en la web; para ello, necesitamos crear dos ficheros: el primero será un formulario con dos campos, uno para el nombre de usuario y otro para la contraseña, y el segundo será un fichero que compruebe esos datos y nos indique si son correctos.

A continuación vamos a ver el contenido del código del formulario, que llamaremos **formregistrados.htm**:

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
```

```
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<STRONG>
IDENTIFICARSE
</STRONG>
<br>
<br>
<form name="contacto" method="post" action="identifica.php">
<p align = "left">
<strong>
NOMBRE DE USUARIO:
</strong>
<br>
<input name="usuario" type="text" size="50">
<br>
<br>
<strong>
CONTRASEÑA:
</strong>
<br>
<input name="cont" type="password" size="50">
<br>
```

```
<br>
<input type="submit" name="Submit" value="Enviar datos">
</p>
</form>
```

Con este código hemos creado un pequeño formulario para que los usuarios se identifiquen en nuestra web.

En la siguiente imagen podemos ver el formulario para identificarse, figura 15.13.

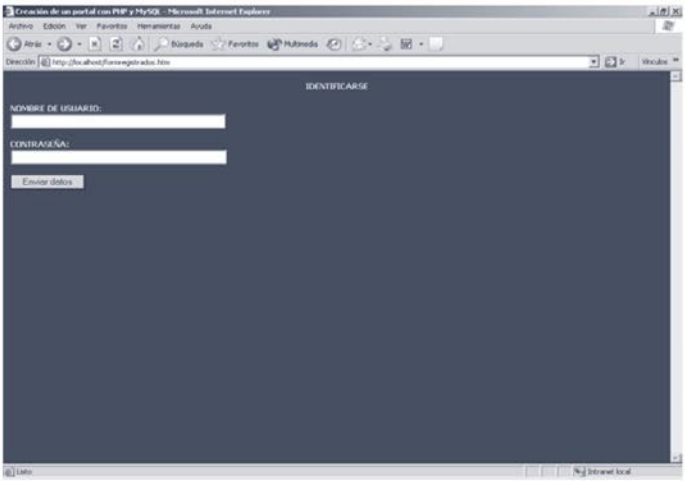

Figura 15.13

El siguiente fichero que crearemos se llamará **identifica.php** y será el encargado de darnos respuesta cuando un usuario intente identificarse.

Este es el código del fichero **identifica.php**:

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<STRONG>RESPUESTA A SU IDENTIFICACIÓN</STRONG>
<br>
<br>
<?
$host="127.0.0.1";
$user="user";    // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="registrados";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$consulta = mysql_query("SELECT nombre FROM usuarios
WHERE usuario LIKE '$usuario' and contraseña LIKE
'$cont'",$enlace);
$dato= mysql_fetch_array ($consulta);
$cambia= $dato["nombre"];
echo "<hr size = 10 color = ffffff width = 100% align = left>";
if ($dato =="") {
```

```
echo "Los datos no son correctos, <a href=formregistrados.php>Volver";
} else {
echo "<STRONG>Bienvenido a nuestra web $cambia</STRONG>";
}
?>
```

Como se puede ver en este código, cuando un usuario no está registrado o ha introducido mal su nombre de usuario o contraseña, aparecerá un mensaje diciendo que es incorrecto y que vuelva atrás para intentarlo de nuevo, en cambio si todo es correcto y ha introducido bien los datos, aparecerá en la pantalla un mensaje de bienvenida indicándole su nombre.

En la siguiente imagen, figura 15.14, podemos ver el resultado de probar con un nombre de usuario y contraseña que previamente hemos registrado.

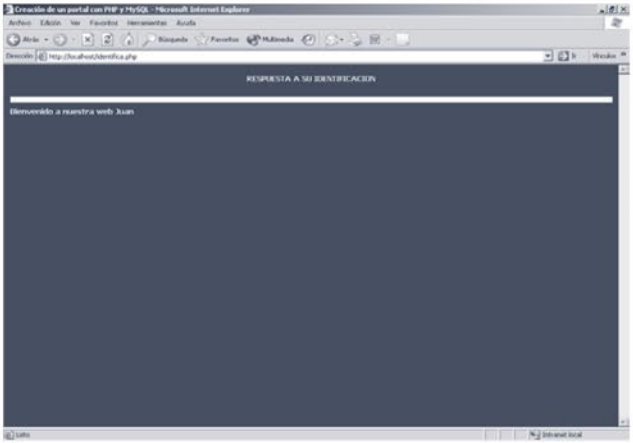

Figura 15.14

15.5 CODIFICAR CONTRASEÑAS CON MD5 ( )

A continuación, vamos a conocer una nueva función de php. Es la función *md5 ( )* y es una función muy útil, ya que nos va a servir para realizar un ejemplo como el que hemos hecho en el apartado anterior, pero ofreciéndonos una mayor seguridad, ya que lo que hace la función *md5 ( )* es encriptar las contraseñas.

15.5.1 Ejemplo

Lo primero que vamos a hacer es crear el formulario de registro, que como veremos a continuación es exactamente igual que el anterior. Lo único que variamos es el nombre del fichero al que enviamos las variables con los datos del formulario. Al formulario de registro le llamamos **registro2.htm**.

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<STRONG>
FORMULARIO DE REGISTRO DE USUARIOS
</STRONG>
<br>
<br>
```

Inserte los datos que a continuación se solicitan para registrarse en nuestra web.

```
<form name="contacto" method="post" action="registrocodificado.php">
<p align = "left">
<strong>
NOMBRE:
</strong>
<br>
<input name="nombre" type="text" value="" size="50">
<br><br>
<strong>
APELLIDOS:
</strong>
<br>
<input name="apellidos" type="text" size="50">
<br><br>
<strong>
NOMBRE DE USUARIO:
</strong>
<br>
<input name="usuario" type="text" size="50">
<br><br>
<strong>
CONTRASEÑA:
</strong>
<br>
<input name="cont" type="password" size="50">
```

```
<br>
<br>
<strong>
E-MAIL:
</strong>
<br>
<input name="email" type="text" size="50">
<br>
<br>
<input type="submit" name="Submit" value="Enviar datos">
</p>
</form>
```

El siguiente paso sería, al igual que en el ejemplo del apartado anterior, crear la base de datos con su correspondiente tabla, pero vamos a trabajar con la misma base de datos, ya que en este sentido no será necesario realizar ningún cambio en la misma.

Como ya tenemos la base de datos, el siguiente paso será crear el fichero que recopila los datos del formulario de registro. Este fichero será **registrocodificado.php**.

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
```

```
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<STRONG>
SU REGISTRO SE HA COMPLETADO CON ÉXITO.
</STRONG>
<br>
<br>
<?
$host="127.0.0.1";
$user="user";    // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="registrados";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$cont2= md5 ($cont);
$consulta = mysql_query("insert into usuarios (nombre,apellidos,usuario,contraseña,email) values ('$nombre', '$apellidos', '$usuario', '$cont2', '$email')",$enlace);
echo "<hr size = 10 color = ffffff width = 100% align = left>";
echo "<STRONG>Bienvenido a nuestra web $nombre</STRONG>";
?>
```

Al igual que en el caso anterior, vemos que el código de este fichero cambia poco respecto del ejemplo anterior. Sólo existe una variación y es en la línea: *$cont2= md5 ($cont);* que como vemos lo que hacemos es codificar la contraseña con la función *md5 ( )* y la insertamos en la base de datos. Es decir cada vez que se registre un usuario en la web, ya tenga su contraseña el número de dígitos o letras que tenga, con la función *md5 ( )* vamos a registrar en nuestra base de datos una contraseña con 32 caracteres.

A continuación, vamos a crear el código de la página para que los usuarios registrados se identifiquen.

Para diferenciar este fichero que creamos, del anterior (**formregistrados.htm**) le hemos añadido un número al nombre, **formregistrados2.htm**. Entre ellos sólo cambia el fichero al que son enviados los datos del formulario.

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<STRONG>
IDENTIFICARSE
</STRONG>
<br>
<br>
<form name="contacto" method="post" action=
"identificacodificado.php">
<p align = "left">
<strong>
```

```
NOMBRE DE USUARIO:
</strong>
<br>
<input name="usuario" type="text" size="50">
<br>
<br>
<strong>
CONTRASEÑA:
</strong>
<br>
<input name="cont" type="password" size="50">
<br>
<br>
<input type="submit" name="Submit" value="Enviar datos">
</p>
</form>
```

A continuación, creamos el fichero que procesa los datos del formulario para comprobar que son correctos e identificarlo. Este fichero es **identificacodificado.php**.

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
```

```
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<STRONG>
RESPUESTA A SU IDENTIFICACIÓN
</STRONG>
<br>
<br>
<?
$host="127.0.0.1";
$user="user";    // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="registrados";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$cont2 = md5 ($cont);
$consulta = mysql_query("SELECT nombre FROM usuarios WHERE usuario LIKE '$usuario' and contraseña LIKE '$cont2'",$enlace);
$dato= mysql_fetch_array ($consulta);
$cambia= $dato["nombre"];
echo "<hr size = 10 color = ffffff width = 100% align = left>";
if ($dato =="")
{
echo $dato;
```

```
echo "Los datos no son correctos, <a href=formregistrados2.htm>Volver";
}
else
{
echo $dato;
echo "<STRONG>Bienvenido a nuestra web $cambia</STRONG>";
}
?>
```

15.6 INSERTAR, ACTUALIZAR, CONSULTAR Y BORRAR DATOS DE UNA TABLA

En el siguiente ejercicio que se nos plantea vamos a realizar las cuatro acciones más usuales con una base de datos. Estas acciones y sus correspondientes parámetros son: *INSERT* para insertar datos en una tabla, *SELECT* para consultar datos, *UPDATE* para actualizar los registros y, por último, *DELETE* para borrar los datos de una tabla.

15.6.1 Ejemplo

En el ejemplo que vamos a realizar en este apartado, crearemos una base de datos con una tabla en la que incluir coches. Para ello, lo principal será crear la base de datos que llamaremos **coches** y su respectiva tabla que llamaremos **ocasión**.

Esta tabla contendrá los campos, id, marca, modelo, combustible, color, fecha y precio del mismo.

En la figura 15.15, podemos ver la imagen correspondiente a la base de datos que hemos creado para insertar vehículos.

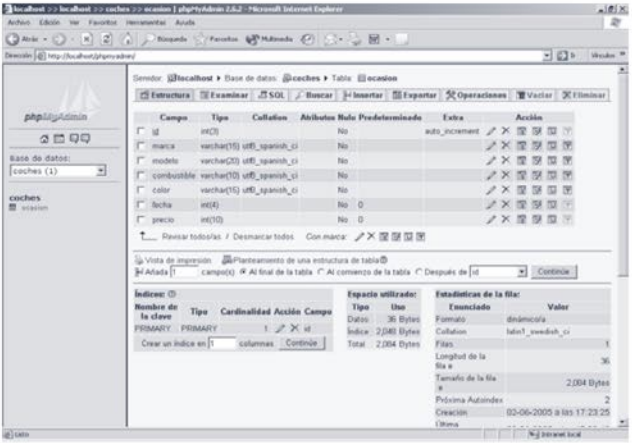

Figura 15.15

A continuación, vamos a crear el primer fichero, que será el encargado de insertar registros en nuestra base de datos. Para ello creamos un formulario que llamaremos **forminserta.htm**. Este es su código:

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
```

```
<form name="form" action= "insertacoches.php"  method="post">
<strong>
<h2>
FORMULARIO PARA INSERTAR VEHiCULOS EN LA BASE DE DATOS.
</h2>
</strong>
<hr size = "8" color = "ffffff" width = "100%" align = "left">
<h5>Seleccione la marca de su vehículo:
<select name="marca">
<option value="alfaromeo">Alfa Romeo </option>
<option value="audi">Audi </option>
<option value="bmw">BMW </option>
<option value="chrysler">Chrysler </option>
<option value="citroen">Citroen </option>
<option value="daewoo">Daewoo </option>
<option value="fiat">Fiat </option>
<option value="ford">Ford </option>
<option value="honda">Honda </option>
<option value="hyundai">Hyundai </option>
<option value="jeep">Jeep </option>
<option value="kia">Kia </option>
<option value="lancia">Lancia </option>
<option value="lexus">Lexus </option>
```

```
<option value="mazda">Mazda </option>
<option value="mercedes">Mercedes </option>
<option value="mitsubishi">Mitsubishi </option>
<option value="nissan">Nissan </option>
<option value="opel">Opel </option>
<option value="peugeot">Peugeot </option>
<option value="porsche">Porsche </option>
<option value="renault">Renault </option>
<option value="rover">Rover </option>
<option value="saab">Saab </option>
<option value="seat">Seat </option>
<option value="skoda">Skoda </option>
<option value="toyota">Toyota </option>
<option value="volkswagen">Volkswagen </option>
<option value="volvo">Volvo </option>
</select>
<br>
</h5>
<h5>
Indique el modelo:
<input name="modelo" type="text" size="45">
</h5>
<h5>
```

```
Indique el color:
<input name="color" type="text" size="48">
</h5>
<h5>
Combustible:
<input name="combustible" type="radio" value="diesel" checked>Diesel
<input name="combustible" type="radio" value="gasolina"> Gasolina
</h5>
<h5>
Año de matriculación:
<input name="fecha" type="text" size="10">
</h5>
<h5>
Precio:
<input type="text" name="precio" size="10">
</h5>
<h5>
<hr size = "4" color = "ffffff" width = "100%" align = "left">
<input name="Enviar" type="submit" value="Enviar">
</h5>
</form>
```

En la siguiente imagen, figura 15.16, podemos ver el formulario.

Figura 15.16

Lo siguiente es desarrollar el código que recopila los datos del formulario y los inserta en la base de datos. Para ello creamos a continuación el fichero **insertacoches.php**, que será el encargado de procesar estos datos. Este es su código:

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<h2>DATOS DEL VEHÍCULO INSERTADOS
```

```
</h2>
<?
$host="127.0.0.1";
$user="user";   // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="coches";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$result = mysql_query("insert into ocasion (marca, modelo, combustible, color, fecha, precio) values ('$marca', '$modelo', '$combustible', '$color', '$fecha', '$precio')", $enlace);
echo "a insertado los siguientes datos:";
echo "<br><br>";
echo "Marca:$marca";
echo "<br>";
echo "Modelo:$modelo";
echo "<br>";
echo "Combustible:$combustible";
echo "<br>";
echo "Color:$color";
echo "<br>";
echo "Fecha:$fecha";
echo "<br>";
echo "Precio:$precio";
echo "<br><br>";
?>
<a href="http://localhost/forminserta.htm">Volver
```

A continuación, podemos ver el resultado que obtenemos cada vez que insertamos un vehículo en la base de datos. Podemos verlo en la siguiente imagen, figura 15.17.

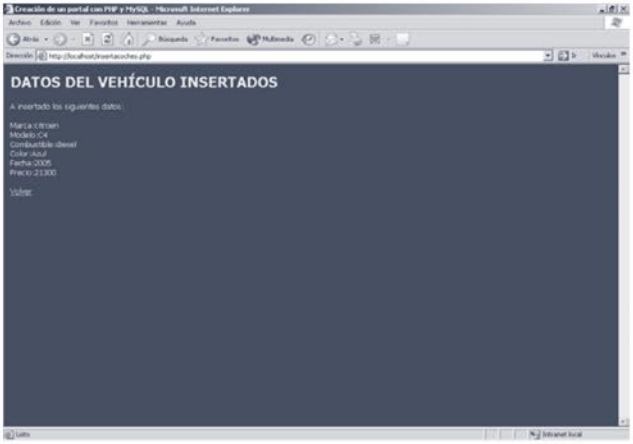

Figura 15.17

El siguiente paso es crear el fichero de consulta a la base de datos. Para ello necesitamos igualmente dos ficheros: uno de ellos será un formulario sobre el que realizamos la búsqueda y otro el que nos devuelve el resultado de la búsqueda.

A continuación, se muestra el código del formulario para búsqueda.

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
```

```
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<form name="form" action= "buscacoches.php"  method="post">
<strong>
<h2>
FORMULARIO PARA BÚSQUEDA DE VEHÍCULOS.
</h2>
</strong>
<hr size = "8" color = "ffffff" width = "100%" align = "left">
<h5>Seleccione la marca de  su vehículo:
<select name="marca">
<option value="alfaromeo">Alfa Romeo </option>
<option value="audi">Audi </option>
<option value="bmw">BMW </option>
<option value="chrysler">Chrysler </option>
<option value="citroen">Citroen </option>
<option value="daewoo">Daewoo </option>
<option value="fiat">Fiat </option>
<option value="ford">Ford </option>
<option value="honda">Honda </option>
<option value="hyundai">Hyundai </option>
<option value="jeep">Jeep </option>
<option value="kia">Kia </option>
<option value="lancia">Lancia </option>
<option value="lexus">Lexus </option>
```

```
<option value="mazda">Mazda </option>
<option value="mercedes">Mercedes </option>
<option value="mitsubishi">Mitsubishi </option>
<option value="nissan">Nissan </option>
<option value="opel">Opel </option>
<option value="peugeot">Peugeot </option>
<option value="porsche">Porsche </option>
<option value="renault">Renault </option>
<option value="rover">Rover </option>
<option value="saab">Saab </option>
<option value="seat">Seat </option>
<option value="skoda">Skoda </option>
<option value="toyota">Toyota </option>
<option value="volkswagen">Volkswagen </option>
<option value="volvo">Volvo </option>
</select>
<br>
</h5><h5>
Indique el modelo:
<input name="modelo" type="text" size="45">
</h5>
<h5>
Año de matriculación:
<input name="fecha" type="text" size="10">
</h5>
<h5>
Precio:
```

```
<input type="text" name="precio" size="10">
</h5><h5>
<hr size = "4" color = "ffffff" width = "100%" align = "left">
<input name="Enviar" type="submit" value="Enviar">
</h5>
</form>
```

A continuación, vamos a crear el que sería el código que procesa los datos del formulario de búsqueda. Al fichero le llamamos **buscacoches.php**.

En la siguiente imagen, figura 15.18, podemos ver el formulario de búsqueda de vehículos, que, como podemos ver, sólo utiliza cuatro campos de búsqueda: marca, modelo, año de matriculación y precio.

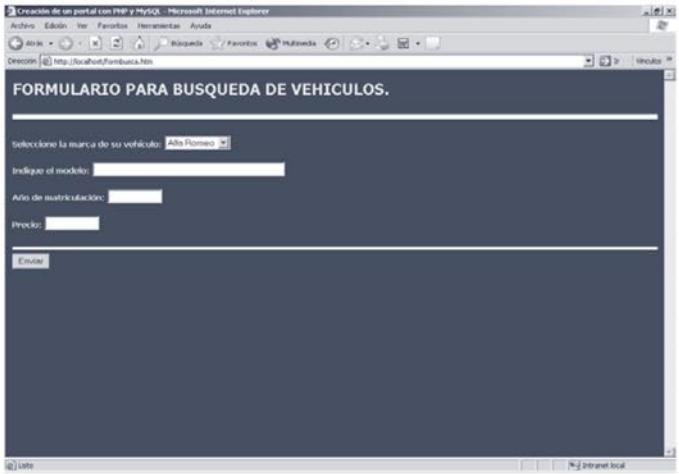

Figura 15.18

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<h2>RESULTADO DE LA BUSQUEDA</h2>
<?
$host="127.0.0.1";
$user="user";    // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="coches";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$consulta = mysql_query("SELECT * FROM ocasion WHERE
marca LIKE '$marca' or modelo LIKE '$modelo' or fecha LIKE
'$fecha' or precio LIKE '$precio'",$enlace);
while($row = mysql_fetch_array($consulta))
{
$id= $row ["id"];
$marca= $row ["marca"];
$modelo= $row ["modelo"];
$fecha= $row ["fecha"];
```

```
echo("<table width='100%' border='0' cellspacing='0' cellpadding= '0'>\n");
echo("<tr>\n");
echo("<td width='12%'><a href=modificarcoche.php?id=$id>Modificar</a></td>\n");
echo("<td width='12%'><a href=borrarcoche.php?id=$id>Borrar</a></td>\n");
echo("<td width='26%'>$marca</a></td>\n");
echo("<td width='26%'>$modelo</td>\n");
echo("<td width='24%'>$fecha</td>\n");
echo("</tr>\n");
echo("</table>\n");
echo "<hr size = 2 color = ffffff width = 100% align = left>";  }
?>
<a href="http://localhost/formbusca.htm">Volver
```

Como podemos observar en este código, para realizar la búsqueda hemos utilizado la instrucción OR en vez de AND. Esto hará que nos muestre una búsqueda de vehículos con todos los resultados que introduzcamos en el buscador, es decir, si en el campo marca ponemos Alfa Romeo y en el campo modelo ponemos León, nos dará todos los vehículos de la marca Alfa Romeo que encuentre igual que todos los modelos de León; si queremos que esto no sea así, debemos modificar las instrucciones OR por AND, quedando de este modo:

```
$consulta = mysql_query("SELECT * FROM ocasion WHERE marca LIKE '$marca' and modelo LIKE '$modelo' and fecha LIKE '$fecha' and precio LIKE '$precio'",$enlace);
```

En la siguiente imagen, figura 15.19, podemos ver el resultado de buscar vehículos de la marca Alfa Romeo y el modelo León.

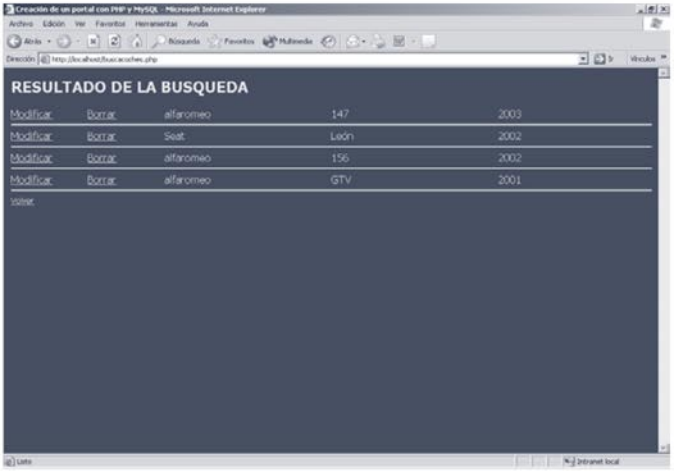

Figura 15.19

Junto a cada registro que aparece en la búsqueda nos encontramos dos opciones, que son *Modificar* y *Borrar*, Con ellas vamos a aprovechar para explicar cómo se realizaría la modificación de registros y cómo se borrarían registros.

Cuando pulsemos en *Modificar*, nos lleva a una página llamada **modificarcoche.php**, y con el id que toma del registro correspondiente al que apunta, vamos a poder modificar el vehículo. Este es el código del fichero **modificarcoche.php**:

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
```

```
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<form name="form" action= "modificarcoche2.php" method="post">
<strong>
<h2>
MODIFIQUE LOS DATOS DEL VEHÍCULO.
</h2>
</strong>
<hr size = "8" color = "ffffff" width = "100%" align = "left">
<h5>
Seleccione la marca de  su vehiculo:
<select name="marca">
<option value="alfaromeo">Alfa Romeo </option>
<option value="audi">Audi </option>
<option value="bmw">BMW </option>
<option value="chrysler">Chrysler </option>
<option value="citroen">Citroen </option>
<option value="daewoo">Daewoo </option>
<option value="fiat">Fiat </option>
<option value="ford">Ford </option>
<option value="honda">Honda </option>
<option value="hyundai">Hyundai </option>
```

```
<option value="jeep">Jeep </option>
<option value="kia">Kia </option>
<option value="lancia">Lancia </option>
<option value="lexus">Lexus </option>
<option value="mazda">Mazda </option>
<option value="mercedes">Mercedes </option>
<option value="mitsubishi">Mitsubishi </option>
<option value="nissan">Nissan </option>
<option value="opel">Opel </option>
<option value="peugeot">Peugeot </option>
<option value="porsche">Porsche </option>
<option value="renault">Renault </option>
<option value="rover">Rover </option>
<option value="saab">Saab </option>
<option value="seat">Seat </option>
<option value="skoda">Skoda </option>
<option value="toyota">Toyota </option>
<option value="volkswagen">Volkswagen </option>
<option value="volvo">Volvo </option>
</select>
<br>
</h5>
<h5>
Indique el modelo:
<input name="modelo" type="text" size="45">
</h5>
```

```
<h5>
Indique el color:
<input name="color" type="text" size="48">
</h5>
<h5>
Combustible:
<input name="combustible" type="radio" value="diesel" checked>
Diesel
<input name="combustible" type="radio" value="gasolina">
Gasolina
</h5>
<h5>
Año de matriculación:
<input name="fecha" type="text" size="10">
</h5>
<h5>
Precio:
<input type="text" name="precio" size="10">
</h5>
<h5>
<hr size = "4" color = "ffffff" width = "100%" align = "left">
<input type="hidden" name="id" value="<?=$id?>">
<input name="Enviar" type="submit" value="Enviar">
</h5>
</form>
```

Si nos fijamos en este fichero, podemos comprobar que es prácticamente igual que el de insertar coches. Sólo cambian dos cosas: una de ellas es el fichero al que envían los datos del formulario, que en este caso es **modificarcoche2.php**, y otra es que hemos introducido un campo oculto en el formulario para que nos lo envíe a la siguiente página, que es el campo id, donde viene almacenada la id del coche que queremos modificar.

A continuación, creamos el fichero **modificarcoche2.php**. Este será su código:

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<h2>
DATOS DEL VEHÍCULO MODIFICADOS
</h2>
<?
$host="127.0.0.1";
$user="user";    // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
```

```
$db="coches";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$result = mysql_query("update ocasion set marca='$marca', modelo='$modelo', combustible='$combustible', color='$color', fecha='$fecha', precio='$precio' WHERE id='$id'", $enlace);
echo "A actualizado los siguientes datos:";
echo "<br><br>";
echo "Marca:$marca";
echo "<br>";
echo "Modelo:$modelo";
echo "<br>";
echo "Combustible:$combustible";
echo "<br>";
echo "Color:$color";
echo "<br>";
echo "Fecha:$fecha";
echo "<br>";
echo "Precio:$precio";
echo "<br><br>";
?>
<a href="http://localhost/formbusca.htm">
Volver
```

En la siguiente imagen, figura 15.20, se muestra el resultado de modificar un registro de la base de datos.

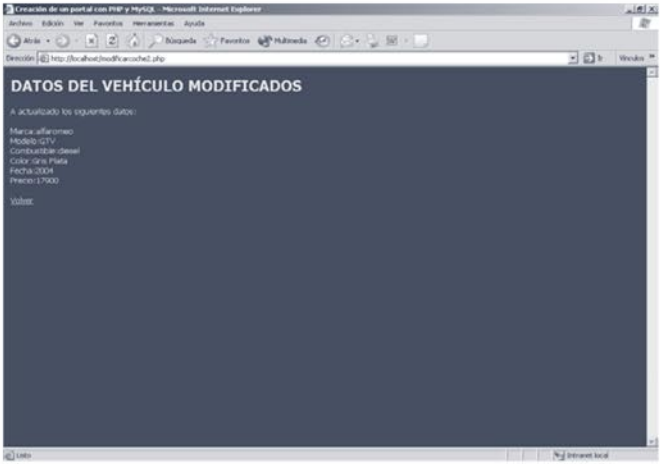

Figura 15.20

Por último, sólo nos quedaría crear la página para borrar registros de la base de datos. Para ello debemos crear el código del fichero **borrarcoches.php**, que será el siguiente:

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
```

```
<p align = "center">
<h2>
BORRAR VEHÍCULOS
</h2>
<?
$host="127.0.0.1";
$user="user";   // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="coches";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$result = mysql_query("delete from ocasion where id='$id'",
$enlace);
?>
<a href="http://localhost/formbusca.htm">
Volver
```

15.7 ENLACES A CADA RESULTADO DE UNA CONSULTA

A continuación, siguiendo con el ejemplo del apartado anterior, vamos a desarrollar un ejemplo para ver cómo se puede enlazar cada resultado obtenido en una consulta con una página que nos mostrará el contenido de ese vehículo.

15.7.1 Ejemplo

Lo primero que tenemos que hacer, será modificar el fichero anterior **buscacoches.php**. Tan sólo cambiaremos una línea de su código, quedando de este modo:

```
<head>
<title>Creación de un portal con PHP y MySQL</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<h2>
RESULTADO DE LA BÚSQUEDA
</h2>
<?
$host="127.0.0.1";
$user="user";    // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="coches";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$consulta = mysql_query("SELECT * FROM ocasion WHERE
marca LIKE '$marca' or modelo LIKE '$modelo' or fecha LIKE
'$fecha' or precio LIKE '$precio'",$enlace);
while($row = mysql_fetch_array($consulta)) {
$id= $row ["id"];
$marca= $row ["marca"];
```

```
$modelo= $row ["modelo"];

$fecha= $row ["fecha"];

echo("<table width='100%' border='0' cellspacing='0' cellpadding= '0'>\n");

echo("<tr>\n");

echo("<td width='12%'><a href=modificarcoche.php?id=$id> Modificar</a></td>\n");

echo("<td width='12%'><a href=borrarcoche.php?id=$id> Borrar</a></td>\n");

echo("<td width='26%'><a href=coches.php?id=$id>$marca </a></td>\n");

/* Esta línea anterior es la única modificación que hemos hecho en todo el código. Hemos incluido un enlace junto al modelo de vehículo que nos lleva a otra página (coches.php) para mostrarnos los datos de ese vehículo. Otra opción sería poner también ese enlace en el modelo, pero eso será a nuestro gusto, como mejor nos venga en cada caso.*/

echo("<td width='26%'>$modelo</td>\n");

echo("<td width='24%'>$fecha</td>\n");

echo("</tr>\n");

echo("</table>\n");

echo "<hr size = 2 color = ffffff width = 100% align = left>"; }

?>

<a href="http://localhost/formbusca.htm">Volver
```

Como se puede ver hasta aquí, apenas hemos modificado nada. Sólo nos quedaría lo más importante, que es crear la página que llamaremos **coches.php**, en la que se mostrarán los datos del vehículo seleccionado.

A continuación, mostramos el código del fichero **coches.php**, que será el que nos muestre los datos del vehículo seleccionado.

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<form name="form" action= "buscacoches.php"  method="post">
<strong><h2>
DATOS DEL VEHÍCULO SELECCIONADO
</h2>
<?
$host="127.0.0.1";
$user="user";    // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="coches";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
```

```
$consulta = mysql_query("SELECT * FROM ocasion WHERE id=$id",$enlace);
while($row = mysql_fetch_array($consulta)) {
$id= $row ["id"];
$marca= $row ["marca"];
$modelo= $row ["modelo"];
$fecha= $row ["fecha"];
$color= $row ["color"];
$combustible= $row ["combustible"];
$precio= $row ["precio"];
echo "Los datos del vehículo que ha solicitado son los siguientes:";
echo "<br>";
echo "Vehículo marca: $marca";
echo "<br>";
echo "Modelo: $modelo";
echo "<br>";
echo "El color es: $color";
echo "<br>";
echo "El combustible que utiliza es: $combustible";
echo "<br>";
echo "Es  del año: $fecha";
echo "<br>";
echo "El precio de este coche es de: $precio €";
}
?>
```

En la figura 15.21 podemos ver el resultado del fichero **coches.php** cuando seleccionamos un vehículo de los resultados obtenidos en el buscador.

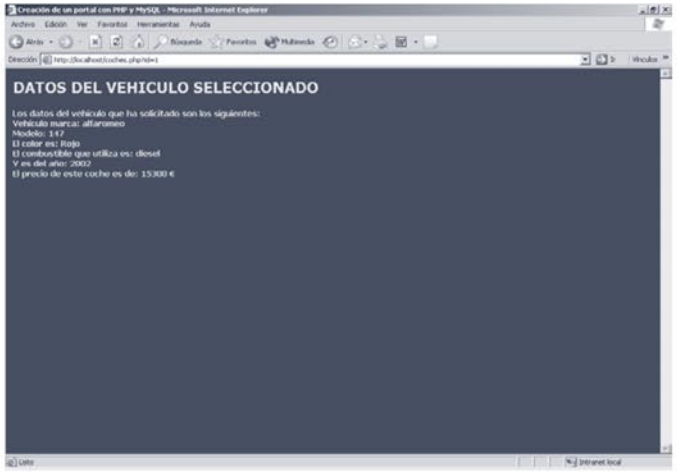

Figura 15.21

15.8 SISTEMA DE ENCUESTAS

A continuación, vamos a desarrollar un sistema con el que intentaremos obtener información de nuestros visitantes. Se trata de un sistema de encuestas a través del cual los usuarios van a poder votar.

El ejemplo que aquí se plantea está diseñado para saber la opinión de los usuarios acerca de nuestra web. También se puede crear semanal o mensualmente otro tipo de encuesta diferente para ir captando información acerca de nuestros usuarios e ir adaptando nuestra web y los servicios de la misma a la opinión que los usuarios nos van ofreciendo.

Más adelante, en este ejemplo, veremos que tiene cierto parecido con el que ya realizamos anteriormente en el apartado 14.2., que era el desarrollo de un contador de visitas, en el que se mostraban los resultados de varias páginas.

15.8.1 Ejemplo

Para realizar el ejemplo necesitaremos cinco archivos: uno de ellos será una imagen, que llamaremos **barra.jpg**, y será la que nos mostrará de forma gráfica el resultado de la encuesta; dos archivos de texto, que serán los que almacenarán el resultado de la encuesta (cada uno almacena un resultado); y una página para la votación y otra para visualizar el resultado.

Comenzemos por la imagen **barra.jpg**. Será un sencillo gráfico como el que se muestra a continuación en la figura 15.22, en el que podremos ver como se va implementando sucesivamente este gráfico para formar una barra.

Figura 15.22

A continuación, crearemos dos archivos de texto, uno llamado **si.txt** y otro que llamaremos **no.txt** y cuyo contenido será exclusivamente 0 como valor inicial.

En la siguiente imagen, figura 15.23, podemos ver el formulario en el que los usuarios podrán realizar las votaciones para la encuesta que vamos a desarrollar. Como se puede observar, tiene sólo dos posibles respuestas, pero para ampliarla se hará de un modo muy sencillo: sólo bastará con poner tantos ficheros .txt como respuestas queramos que tengan nuestra encuesta y modificar el siguiente fichero que tenemos que crear.

Figura 15.23

A continuación, pasamos a desarrollar la página que contendrá la pregunta de la encuesta y las dos posibles respuestas. Al fichero le llamaremos **vota.php**.

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<h3>
```

```
<p align="center">
SISTEMA DE ENCUESTA
</h3>
</p>
<hr size="8" color="ffffff">
<p>
<font size="2" face="Arial, Helvetica, sans-serif">
<p align = "center">
<strong>
¿Te gusta esta página?
</strong>
</font>
</p>
<form name="form1" method="post" action="resultado.php">
<p>
<font size="2" face="Arial, Helvetica, sans-serif">
<p align = "center">
<input type="radio" name="op" value="a">
Sí
</font>
</p>
<p>
<font size="2" face="Arial, Helvetica, sans-serif">
<p align = "center">
<input type="radio" name="op" value="b">
No
```

```
</font>
</p>
<p align = "center">
<input type="submit" name="Submit" value="Enviar">
</p>
</form>
<hr size="3" color="ffffff" width = 45%>
<p align = "center">
Le agradecemos que utilice este servicio de nuestra página web,
</p>
<p align = "center">
ya que nos sirve para mejorar cada día su opinión.
</p>
```

Y, a continuación, mostraremos el resultado que se obtiene al pulsar el botón *Enviar* con los resultados de las votaciones. A este fichero le llamaremos **resultado.php** y será el encargado de ir sumando los votos a un fichero o a otro y de mostrar los resultados parciales de la votación.

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
```

```
<body link = "#E5E5E5" vlink ="E0E0E0">
<h3>
<p align="center">
RESULTADOS DE LA ENCUESTA
</h3>
</p>
<hr size="8" color="ffffff">
<?
$archivo1 = "si.txt";
$archivo2 = "no.txt";
$abre1 = fopen($archivo1, "r");
$abre2 = fopen($archivo2, "r");
$total1 = fread($abre1, filesize($archivo1));
$total2 = fread($abre2, filesize($archivo2));
fclose($abre1);
fclose($abre2);
if($op=="a")
{
$abre1 = fopen($archivo1, "w");
$total1=$total1+1;
$grabar1 = fwrite($abre1, $total1);
fclose($abre1);
}
else if($op=="b")
{
```

```
$abre2 = fopen($archivo2, "w");
$total2=$total2+1;
$grabar2 = fwrite($abre2, $total2);
fclose($abre2);
}
$votos=$total1+$total2;
$por1=$total1*100/$votos;
$por1=intval ( $por1 ,10);
$por2=$total2*100/$votos;
$por2=intval ( $por2 ,10);
echo "<br>";
echo "<br>";
echo "<p align=center>";
echo "<img height=15 width=$por1 SRC=figura14-2.jpg>";
echo "<br>";
echo "<br>";
echo "Si: <b>$total1</b> votos - <b>$por1 %</b>";
echo "<br>";
echo "<br>";
echo "<p align=center>";
echo "<IMG HEIGHT=15 WIDTH=$por2 SRC=figura14-2.jpg>";
echo "<br>";
echo "<p align=center>";
echo "No: <b>$total2</b> votos - <b>$por2 %</b>";
```

```
echo "<br>";
echo "<br>";
echo "<br>";
echo "<p align=center>";
echo "Total Votos: <b>$votos</b>";
echo "<br>";
echo "<br>";
echo "<p align=center>";
echo "<a href=vota.php>Volver</a>";
?>
```

Como hemos podido observar, sólo se da la posibilidad de elegir entre dos opciones para la votación, pero se pueden poner muchas más opciones. Siguiendo el código que se muestra, será muy sencillo para el lector poder modificarlo según sus necesidades e incluir tantas respuestas como considere necesarias.

Analizando el código, podemos ver lo sencillo que es: a tantas opciones, tantos ficheros .txt para almacenar e ir sumando los votos. Y en cuanto al código del fichero **resultado.php**, tendremos que agregar repetidas líneas como las que ya tenemos, en las que incluiremos los ficheros .txt nuevos que necesitemos.

A continuación, podemos ver el resultado de la página de la encuesta en la siguiente imagen, figura 15.24, donde podemos observar más detenidamente el incremento del gráfico de la barra (imagen **barra.jpg**), en función del porcentaje de votos a favor o en contra de la página web, obteniendo una imagen real en función del porcentaje de votos.

Figura 15.24

Como ya se dijo al principio de este ejemplo, si se observa detenidamente el código de sus ficheros, podemos ver que es muy similar al que realizamos cuando creamos el ejemplo del contador de visitas: los códigos son muy parecidos en cuanto a los ficheros que procesan los datos, pero las aplicaciones que hemos desarrollado son totalmente diferentes; además, en este caso, la diferencia es que para obtener los resultados hemos tenido que desarrollar un formulario para que nos dé el porcentaje de votos a favor o en contra y así modificar la página de datos.

15.9 POSTALES SIN BASE DE DATOS

A continuación, vamos a desarrollar una sencilla aplicación que nos va a servir para que nuestros usuarios puedan enviar postales a sus amigos desde nuestra web.

El proceso que utilizaremos es el siguiente: diseñaremos una página con una serie de fotografías, en concreto, postales de coches

(que es la temática del portal que estamos realizando) y, en la parte inferior de la página, tendremos un pequeño formulario mediante el cual podremos enviar una postal de las que anteriormente hemos visto seleccionando la fotografía de una lista desplegable. Además, el usuario podrá insertar un pequeño mensaje que también se enviará junto con la postal.

15.9.1 Ejemplo

Creamos primero la página en la que tendremos las fotos y más abajo el formulario para enviar la postal. Al fichero le llamaremos **postal.php**.

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<h3>
<p align="center">
ENVÍO DE POSTALES
</h3>
</p>
<hr size="8" color="ffffff">
<table border="1" width="40%" align="center">
```

```
<tr>
<td width="10%">
<img src="coche1.jpg">
SMART
</td>
<td width="10%">
<img src="coche2.jpg">
BMW
</td>
<td width="10%">
<img src="coche3.jpg">
MERCEDES
</td>
<td width="10%">
<img src="coche4.jpg">
PORSCHE
</td>
</tr>
</table>
<form name="form" action= "enviapostal.php" method="post">
<p>
Enviar postal a:;
<input type="text" name="email" size="20">
</p>
<p>Titulo:
```

```
<input type="text" name="asunto" size="20">
</p>
<p>Quién la envía:
<input type="text" name="quien" size="20">
</p>
<p>
<select name="foto">
<option>
coche1.jpg
</option>
<option>
coche2.jpg
</option>
<option>
coche3.jpg
</option>
<option>
coche4.jpg
</option>
</select>
</p>
<p>
<input type="submit" value="Enviar" name="Enviar">
</p>
</form>
```

El funcionamiento del mismo es muy sencillo: introducimos el correo electrónico del destinatario de la postal, un título para la postal, ponemos el nombre de la persona que lo envía y por último, en el menú desplegable, podemos seleccionar de las imágenes de arriba la imagen que deseamos enviar y pulsamos el botón *Enviar*.

Una vez que pulsemos en el botón *Enviar*, nos llevará al fichero **enviapostal.php**, que será el encargado de procesar los datos del formulario para hacer llegar la postal al remitente indicado.

A continuación, vamos a crear el código del fichero enviapostal.php, que será un sencillo fichero de respuesta, en el que trabajaremos con la función *mail ( )* para poder enviar la postal al usuario.

En la figura 15.25, podemos ver la imagen correspondiente al formulario que hemos diseñado para el envío de postales.

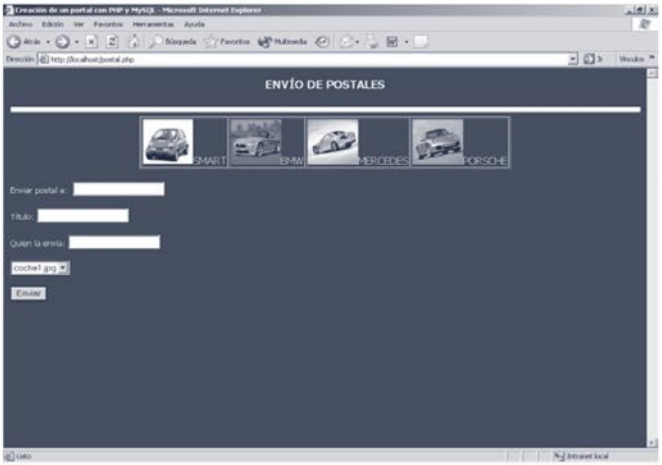

Figura 15.25

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<h3>
<p align="center">
ENVÍO DE POSTALES
</h3></p>
<hr size="8" color="ffffff">
<?
mail ($email,$asunto,$foto, 'From:'.$quien);
echo "Su postal ha sido enviada con éxito a: $email";
echo "<br><br>";
echo "El título de la postal que ha mandado es: $asunto";
echo "<br><br>";
echo "La foto que ha enviado es: $foto";
?>
```

El resultado que se obtiene cuando enviamos la postal será el de la imagen siguiente, figura 15.26.

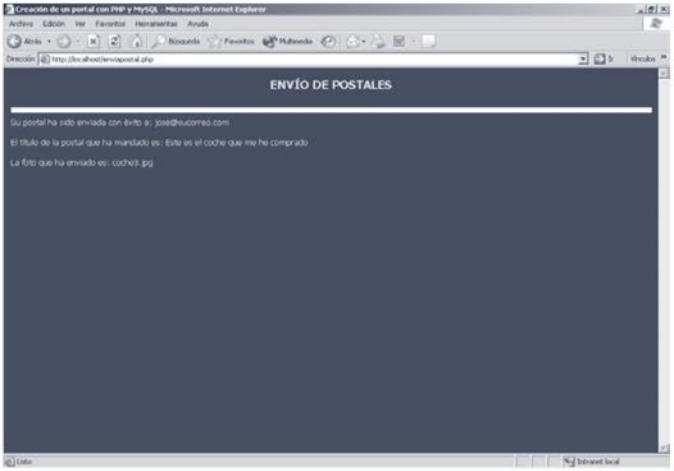

Figura 15.26

15.10 GENERAR NÚMEROS ALEATORIOS

PHP dispone de una función mediante la cual es posible obtener número aleatorios. Esta es la función *rand ( )*.

A continuación, vamos a crear un ejemplo para que los usuarios se registren en la web, pero en vez de que tengan que introducir ellos el nombre de usuario y la contraseña, sólo van a introducir el nombre de usuario, ya que la contraseña se la vamos a proporcionar nosotros empleando la función *rand ( )*.

15.10.1 Ejemplo

Este sería el código del fichero del formulario de registro regusuarios.htm.

```
<head>
<title>
Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<STRONG>
FORMULARIO DE REGISTRO DE USUARIOS
</STRONG>
<br><br>
Inserte los datos que a continuación se solicitan:
<form name="contacto" method="post" action=
"registrousuarios.php">
<p align = "left">
<strong>NOMBRE:</strong>
<br>
<input name="nombre" type="text" value="" size="50">
<br><br>
<strong>APELLIDOS:</strong><br>
<input name="apellidos" type="text" size="50">
<br>
<br>
<strong>NOMBRE DE USUARIO:</strong><br>
```

```
<input name="usuario" type="text" size="50">
<br>
<br>
<strong>E-MAIL:</strong>
<br>
<input name="email" type="text" size="50">
<br><br>
<input type="submit" name="Submit" value="Enviar datos">
</p>
</form>
```

Esta sería la imagen correspondiente al formulario de registro, figura 15.27.

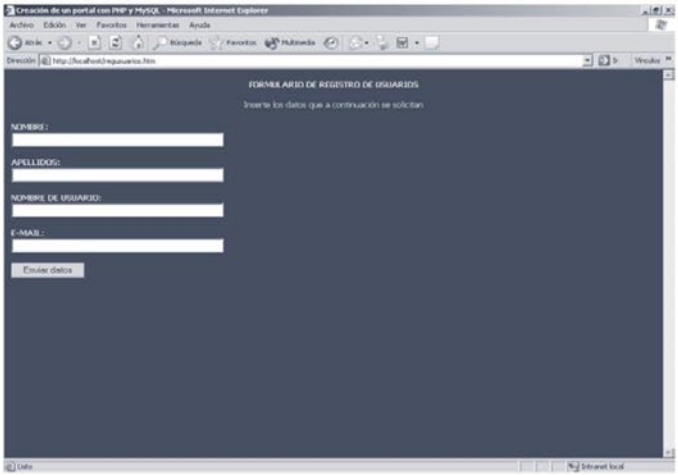

Figura 15.27

A continuación, vamos a crear el fichero de respuesta al formulario que ya tenemos, en el que además se le indicará al usuario qué contraseña le ha sido asignada de forma aleatoria. Una vez más, al igual que hicimos en los apartados 15.4. y 15.5., vamos a utilizar la base de datos registrados, que ya habíamos creado anteriormente en el apartado 15.4.

El código del fichero que recibe los datos del formulario, **registrousuarios.php**, será el siguiente:

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<STRONG>
SU REGISTRO SE HA COMPLETADO CON ÉXITO
</STRONG>
<br><br>
<?
$host="127.0.0.1";
$user="user";    // Poner aquí nuestro nombre de usuario.
$password="pass";   // Poner aquí nuestra contraseña.
$db="registrados";
$enlace = mysql_connect($host,$user,$password);
mysql_select_db($db,$enlace);
$cont = rand (100000, 99999999);
```

```
$consulta = mysql_query("insert into usuarios (nombre,apellidos,usuario,contraseña,email) values ('$nombre', '$apellidos', '$usuario', '$cont', '$email')",$enlace);
echo "<hr size = 10 color = ffffff width = 100% align = left>";
echo "<STRONG>Bienvenido a nuestra web $nombre</STRONG>";
echo "<br>";
echo "La contraseña que le hemos asignado es: $cont";
?>
```

Como podemos observar, cuando el usuario pulsa en el formulario el botón *Enviar*, le llevará a esta página, en la que se generará aleatoriamente una clave, que será la que deberá emplear cuando quiera identificarse en la web. Para crear la clave, hemos indicado que nos genere un número entre 1.000.000 y 99.999.999.

En la siguiente imagen, figura 15.28, podemos ver el resultado de ejecutar este ejemplo.

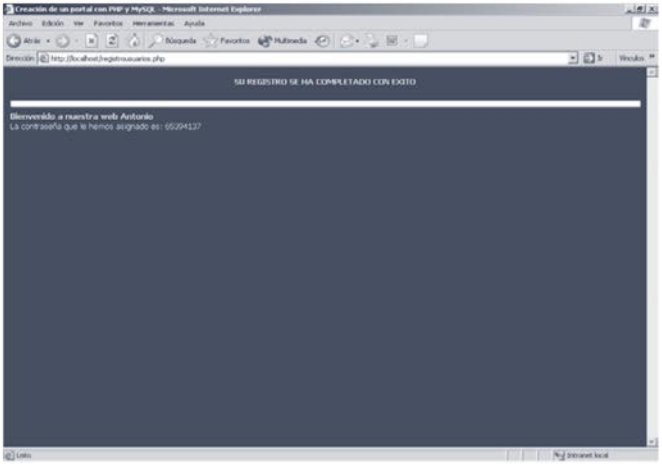

Figura 15.28

15.11 ROTADOR DE BANNER

La aplicación que vamos a desarrollar a continuación es, como su nombre bien indica, un rotador de banner. Esto puede sernos muy útil en nuestra web si utilizamos varios banner de varios patrocinadores, ya que con este ejemplo no siempre se mostrarán los mismos, sino que serán aleatorios.

15.11.1 Ejemplo

Este será el código del rotador de banner que podemos incluir en cualquiera de nuestras páginas web:

```
<head>
<title>Creación de un portal con PHP y MySQL
</title>
</head>
<body bgcolor = "#303030">
<body text = "#E5E5E5">
<font face = "tahoma">
<font size = "2">
<body link = "#E5E5E5" vlink ="E0E0E0">
<p align = "center">
<STRONG>ROTADOR DE BANNER</STRONG><br><br>
<?
$banner = 4;
$aleatorio = rand (1,$banner);
$imagen = array ( );
```

```
$url = array ( );
$texto = array ( );
$imagen[1]="banner1.jpg";
$url[1]= "http://localhost/pagina1.php";
$texto[1]="Te recomendamos visitar este banner 1";
$imagen[2]="banner2.jpg";
$url[2]= "http://localhost/pagina2.php";
$texto[2]="Te recomendamos visitar este banner 2";
$imagen[3]="banner3.jpg";
$url[3]= "http://localhost/pagina3.php";
$texto[3]="Te recomendamos visitar este banner 3";
$imagen[4]="banner4.jpg";
$url[4]= "http://localhost/pagina4.php";
$texto[4]="Te recomendamos visitar este banner 4";
echo "<a href = '$url[$aleatorio]'><img src='$imagen[$aleatorio]'
alt='$texto[$aleatorio]'></a>";
?>
```

Como se puede observar en este ejemplo, hemos creado el código suponiendo que tenemos cuatro banner, pero esto se puede modificar a tantos banner como queramos.

En la siguiente imagen, figura 15.29, podemos ver el ejemplo del rotador de banner.

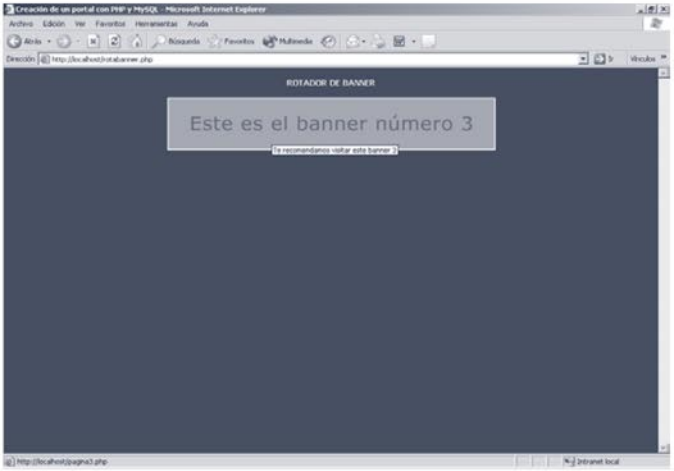
Creación de un portal con PHP y MySQL - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Búsqueda Favoritos Multimedia
Dirección http://localhost/rotabanner.php
Vínculos
ROTADOR DE BANNER
Este es el banner número 3
Te recomendamos visitar este banner !!
http://localhost/pagina3.php
Intranet local

Figura 15.29

GLOSARIO DE TÉRMINOS

CHDIR

Mediante esta instrucción determinaremos, a la hora de trabajar con ficheros, el directorio que vamos a utilizar.
Sintaxis:
CHDIR ([nombre_del_directorio]);

DATE

Date es la función utilizada tanto para el manejo de fecha como de hora. Sólo tenemos que hacer una llamada a esta instrucción indicando el formato que deseamos y nos mostrará o almacenará la fecha y/u hora, con el formato que queramos que aparezca. Es decir, sólo día y mes, o día y hora, con las combinaciones que elijamos.
Sintaxis:
DATE ("[formato]");

DEFINE

Mediante esta instrucción definimos constantes. Una constante no puede variar nunca su valor una vez que la hayamos definido para utilizarla en una página web. Para definir la constante, debemos primero asignar un nombre a esta y un valor.

Sintaxis:

DEFINE ([variable],[valor]);

DELETE

Esta instrucción la utilizaremos para borrar datos (registros) de una tabla. Para ello, sólo hemos de hacer una llamada a esta instrucción e indicarle qué es lo que queremos borrar.

Sintaxis:

DELETE [lo_que_desees_borrar] FROM [nombre_tabla];

ECHO

La instrucción ECHO, posiblemente sea una de las más empleadas a la hora de programar en PHP, ya que se empleará casi siempre para mostrar cualquier resultado que deseemos en pantalla.

Sintaxis:

ECHO "[parte_a_mostrar]";

ELSE

La instrucción ELSE forma parte de las instrucciones condicionales y siempre irá precedida de una instrucción IF. Otorga otra serie de condiciones a una condición.

Sintaxis:

} ELSE ([condición] {;

EREG

EREG es una instrucción empleada para el reconocimiento de cadenas de texto, es decir, la usaremos cuando necesitemos verificar que existe una determinada cadena en una variable.
Sintaxis:
EREG ("[cadena]", [variable]);

EREGI

La instrucción EREGI hace exactamente la misma función que la anterior instrucción EREG. La única diferencia es que distingue entre mayúsculas y minúsculas, mientras que EREG no lo hace.
Sintaxis:
EREGI ("[cadena]", [variable]);

FCLOSE

La instrucción FCLOSE se emplea cuando se ha terminado de escribir un fichero.
Sintaxis:
FCLOSE ([nombre_fichero]);

FOPEN

Utilizaremos esta instrucción cuando necesitemos trabajar con un fichero en concreto, ya sea para escribir contenido en él o simplemente para leerlo.
Sintaxis:
FOPEN ([nombre_fichero]);

FOR

FOR es una de las instrucciones de bucle. La emplearemos para ejecutar un determinado número de veces un bucle, hasta que se haya cumplaido la condición que le indiquemos. Esta

instrucción se compone de tres partes: en la primera definimos inicialmente la variable a emplear, en la segunda le indicamos qué valor debe tener la variable para salir del bucle, y en la última le indicaremos qué variaciones debe cumplir la variable para alcanzar el valor hasta salir del bucle.

Sintaxis:

FOR([variable_y_valor_inicial], [variable_y_valor_deseado], [pasos_a_seguir_para_valor_deseado]);

FREAD

Utilizaremos esta instrucción para acceder a la lectura de un fichero. Para ello sólo tenemos que hacer una llamada a un fichero e indicarle cuántos caracteres queremos que nos muestre de este fichero.

Sintaxis:

FREAD ([nombre_fichero], [cantidad_a_mostrar]);

FUNCTION

Utilizaremos FUNCTION para emplear funciones que tengamos creadas en una página web. En una misma página podemos tener creadas las funciones que queramos y utilizarlas posteriormente cuando sea necesario.

Sintaxis:

FUNCTION [nombre_funcion] ([variable]);

FWRITE

La instrucción FWRITE la utilizaremos cuando nos interese escribir contenido en algún fichero, pero debemos prestar atención a que este fichero que vamos a escribir esté anteriormente abierto y que tenga el permiso de escritura.

Sintaxis:

FWRITE [fichero];

IF

La instrucción IF forma parte de las instrucciones condicionales. Su utilización nos indica que si se cumple la condición que contiene el IF, ejecutará cierta parte del código de nuestra página. Si no cumple esta condición, saltará y seguirá adelante, sin hacer caso de esta instrucción.

Sintaxis:

IF ([condición]) { haz_esto };

INCLUDE

INCLUDE lo podemos utilizar cuando sea necesario hacer una llamada a un fichero que contenga código y que necesitamos que se ejecute en la página actual. Esta instrucción es muy empleada para realizar conexiones a bases de datos, y su principal característica es que nos puede ahorrar muchas líneas de código.

Sintaxis:

INCLUDE [fichero];

INSERT

La instrucción INSERT la utilizaremos para insertar registros en una tabla.

Sintaxis:

INSERT into [nombre_tabla] (campos_tabla) values ("valor_del_campo");

MAIL

MAIL es la instrucción que utilizaremos cuando queramos crear una aplicación para enviar correos electrónicos. Requiere definir tres parámetros: el primero de ellos es el destinatario del correo electrónico, el segundo será el encabezado del correo electrónico que se va a enviar y, por último, el cuerpo del mensaje que enviaremos.

Sintaxis:
MAIL ([email_destinatario], [titulo_mensaje], [cuerpo_mensaje]);

MKDIR

MKDIR se utiliza para crear directorios.
Sintaxis:
MKDIR ([nombre_directorio]);

MYSQL_CONNECT

Utilizaremos la instrucción MYSQL_CONNECT para conectar a una base de datos MySQL. Esta instrucción requiere tres parámetros: el primero de ellos es donde determinamos el *host* al que queremos conectarnos; el segundo, el nombre de usuario para acceder a la base de datos y el tercero, el *password* (contraseña).
Sintaxis:
MYSQL_CONNECT ([host], [nombre_usuario], [password]);

MYSQL_FETCH_ARRAY

Esta instrucción nos devuelve un array con el resultado de una consulta.
Sintaxis:
MYSQL_FETCH_ARRAY ([consulta_SELECT]);

MYSQL_NUM_ROWS

Esta instrucción nos devuelve un valor que indica el número de registros encontrados en una consulta.
Sintaxis:
MYSQL_NUM_ROWS ([consulta_SELECT]);

MYSQL_QUERY

Ejecuta una consulta a la base de datos activa en el servidor asociado al identificador de conexión. Requiere dos parámetros: la consulta que deseemos realizar y los datos de conexión a la base de datos (*host*, nombre de usuario y *password*).

Sintaxis:

MYSQL_QUERY ([consulta_SELECT], [conexión_base_de_datos]);

MYSQL_SELECT_DB

Utilizamos esta instrucción para seleccionar una base de datos entre todas las que tengamos. Requiere dos parámetros: el primero de ellos es la base de datos que queramos utilizar y el segundo, la conexión para la base de datos.

Sintaxis:

MYSQL_SELECT_DB ([base_de_datos], [conexión_base_de_datos]);

ORD

ORD nos convierte un carácter en su correspondiente valor en código ASCII.

Sintaxis:

ORD ([carácter]);

PRINTF

Esta instrucción imprime en pantalla una cadena de texto, previamente formateada.

Sintaxis:

PRINTF ([formato], "[cadena_de_texto]");

RANDOM

Esta instrucción se emplea para generar números de forma aleatoria. Podemos indicar el rango entre el cual queremos que se nos muestre el número.

Sintaxis:

RANDOM (valor1, valor2);

REQUIRE

Realiza la misma función que la instrucción INCLUDE.

Sintaxis:

REQUIRE ([fichero]);

RMDIR

RMDIR se utiliza para borrar directorios.

Sintaxis:

RMDIR ([nombre_directorio]);

SELECT

La instrucción SELECT se utiliza para seleccionar registros de una base de datos. El uso de esta instrucción puede llevar a que se determinen ciertas condiciones en la selección de los registros, es decir, que deban cumplir una serie de condiciones.

Sintaxis:

SELECT [condición] from [nombre_tabla];

SPRINTF

Hace la misma función que PRINTF, pero la diferencia es que esta, en vez de mostrarlo en pantalla, lo almacena en una variable.

Sintaxis:
$variable = SPRINTF ([formato],"[cadena_de_texto]");

STRTOLOWER

Convierte una cadena de caracteres a minúsculas.
Sintaxis:
STRTOLOWER ([cadena]);

STRTOUPPER

Convierte una cadena de caracteres a mayúsculas.
Sintaxis:
STRTOUPPER ([cadena]);

SUBSTR

La función SUBSTR se encarga de mostrar una parte determinada de una cadena de texto. Para su utilización será necesario que le indiquemos primero la cadena a mostrar y, posteriormente, a partir de qué carácter queremos que nos muestre esa cadena.
Sintaxis:
SUBSTR ("[cadena]", [caracteres a mostrar]);

UPDATE

UPDATE es la instrucción encargada de actualizar registros en una base de datos. Para poder actualizar un registro, debemos indicar qué condiciones ha de tener el registro que vamos a actualizar para poder modificarlo.
Sintaxis:
UPDATE [tabla] set [condición];

ÍNDICE ALFABÉTICO

H

I

J

L

M

N

O

P

R

S

U

Creación de un portal con PHP y MySQL

4ª Edición

¿Le gustaría tener su propio portal en Internet? Con ayuda de este libro aprenderá a programar aplicaciones en PHP y a utilizar la base de datos MySQL; combinando estas dos potentes herramientas y sin que tenga ningún conocimiento previo de programación ni de manejo de base de datos, en poco tiempo, usted será capaz de programar sus páginas web, desarrollando aplicaciones hoy en día empleadas en muchos portales, como pueden ser formularios, foros, libros de visitas, contadores de visitas, rotadores de banner, etc.

Para esta cuarta edición del libro se ha realizado una actualización de las versiones de los programas utilizados: tanto en PHP como en MySQL, en la aplicación para la gestión de base de datos phpMyAdmin, y también las últimas versiones de los paquetes de aplicaciones WAMP y AppServer.

Con ayuda de este libro le será muy sencillo aprender a realizar su propia página web, incluso proyectos profesionales, ya que está desarrollado de manera que cada apartado incluye ejemplos con el código necesario para hacerlas funcionar.

9 788499 640242

ra-ma.es

Ra-Ma®