PHP y MySQL

Etiqueta:

```
<?php …código en php… ?>
```

Comentarios:

```
/* Comentario para varias lineas. */
// Comentario de una sola linea.
```

Salida por pantalla:

```
print ("cadena de texto");
echo "cadena de texto";
printf (formato, variables, …);    // Funciona igual que el de C/C++.
                                    // Con '\n' incluido para saltar de linea.
```

(Nota: Alguno de los indicadores de variables en el formato son:

- %s Para cadenas de carácteres.
- %d Para números enteros.
- %f Para números reales.
- %c Para caracteres ASCII.

Aunque existen otros tipos, estos son los más importantes).

Variables:

```
$identificador = valor;          $num = 30;
print ("$identificador");       print ("$num<br>");
                                 echo $Var1, "<br>", $Var2;
                                 printf ("Valor = %d<br>", $num);
```

Para poder acceder a una variable global, ya que con $variable solo estaremos declarando variables locales a la función, tenemos que usar:

```
global miVariable;
```

Cuando se usan formularios, se pueden realizar obtención de datos haciendo referencia con el identificador puesto en la propiedad name:

```
<form action="first_name.php">
<p>Nombre: <input type="text" name="firstName"><br>
      <input type="submit" name="Enviar"></p></form>

print ("$firstName");
```

Además podemos pasar valores con la URL, ejemplo:

```
http://localhost/name.php?name=John&age=37
```

Operadores:

+ Aritméticos: + - * / % ++ --
+ Relacionales: == != < > <= >=
+ Lógicos: && and || or !

Estructuras de control:

Las sentencias de control del lenguaje PHP son:

```
if (condición) {
        …;
} else {
        …;
}

while (condición) { …; }

for (inicio; condición; sentencia) { …; }

break;

return;
```

```
switch ($variable) {
        case valor:
                …;
                break;
        …
        default:
}

do { …; } while (condición);

continue;
```

Una función que es importante para tratar con variables en nuestro código es isset. Esta sirve para ver si una variable tiene o no un valor asignado:

```
isset($variable);
      true = $variable tiene un valor establecido.
      false = No tiene valor alguno la variable.
```

Matrices:

```
$var = array (nombre1 => valor1,
              nombre2 => valor2,
              …);

$var[nombre1]   ←→   $var[0]
$var[nombre2]   ←→   $var[1]

$var = array (valor1, valor2, …);
```

Funciones:

```
function nombre ($arg1, $arg2, …) {
      …;
}
```

Ejemplo:

```
function hola ($nombre) {
        print ("Hola $nombre.");
}
```

Y para llamar a una función simplemente basta con:

```
hola ("Juan");
```

Manejar de las cadenas:

número = **strlen** (cadena);
Nos devuelve el número de carácteres de una cadena.

resultado = **split** (separador, cadena);
Divide una cadena en varias usando un carácter separador.

resultado = **sprintf** (formato, variables, ...);
Da una nueva cadena con las variables que le hemos pasado, formateadas de las forma que nosotros querramos.

resultado = **substr** (cadena, inicio, longitud);
Devuelve una subcadena de otra, empezando por inicio y terminando por inicio + longitud.

resultado = **chop** (cadena);
Elimina los saltos de línea y los espacios finales de una cadena.

número = **strpos** (cadena, patrón);
Busca en la cadena el patrón, devolviéndonos la posición en la que se encuentra.

resultado = **str_replace** (cadena1, cadena2, texto);
Reemplaza la cadena1 por la cadena2 en el texto.

resultado = **trim** (cadena);
Da la cadena sin los espacios en blanco de la derecha y la izquierda.

Funciones del tiempo:

date ("formato");	Da una cadena con la fecha.
time ();	Da el número de segundos desde 1970.

Manejo de las cookies:

setcookie (nombre, [valor, [expiración, [ruta, [dominio, [seguridad]]]]]);
+ nombre (string): Nombre de la cookie en el cliente.
+ valor (string): Valor que almacenará la cookie en el cliente.
+ expiración (int): Duración de la cookie (ejemplo: time() + 3600 = 1 hora).

+ ruta (string): Subdirectorio en donde tiene valor la cookie.
+ dominio (string): Dominio en donde tiene valor la cookie.
+ seguridad (int): Un valor que se usa para conexiones HTTPS.

(Nota: Al indicar el dominio es mejor usar algo como miurl.com, que usar www.miurl.com porque este último método no abarca el primero, y sin embargo el primero abarca ambos).

Ejemplo:

```
setcookie("usuario", "chris");
```

Manejo de los ficheros:

```
fichero = fopen (nombre, modo);
        Modos:      a       Añadir.
                    a+      Añadir y leer.
                    r       Leer.
                    r+      Leer y escribir.
                    w       Escribir.
                    w+      Escribir y leer.
fclose (fichero);

fpassthru (fichero);          Hace un print de todo el fichero.
fgets (fichero, numcar);      Lee y devuelve en una cadena el número de
                              caracteres indicado.
fgetss (fichero, numcar);     Parsea el php y el html del fichero.
fputs (fichero, valor);       Escribe un dato en el fichero.
fread (fichero, tamaño);      Lee un tamaño de bytes del fichero y lo devuelve.
fwrite (fichero, valor);      Escribe un dato en el fichero.
```

Modularización:

Podemos crearnos un fichero con código php, por ejemplo "misFuncs.php":

```
<?php
        function Saluda () {
                echo "Hola mundo.";
        }
?>
```

De hecho también podemos tener código HTML por en medio de dicho fichero, si nos interesara tenerlo, ya que con la directiva include, php insertará en ese punto del fichero, el contenido del otro fichero invocado:

```
<body>
<?php
        include ("misFuncs.php");
        Saluda ();
?>
```

```
</body>
```

Mandar un correo:

```
mail (destinatario, asunto, cuerpo, [encabezadosAdicionales]);
```

Ejemplo:

```
mail (yo@miurl.com, "Hola", "Te envio esto para saludarte…");
```

Funciones para MySQL:

```
$serverConnection = mysql_connect ("localhost", "usuario", "contraseña");
// Nos conecta al servidor MySQL que contiene la base de datos.

mysql_select_db ("NombreBD", $serverConnection);
// Selecciona la base de datos que queremos.

$result = mysql_query ("select * from tabla", $serverConnection);
// Con esta función se pueden lanzar cualquier sentencia SQL
// de las siguientes: select, insert, delete, update.

$fila = mysql_fetch_array ($result);
// Esta función devuelve la primera fila, pasando su puntero de lectura a la
// siguiente dentro de result (parecido a leer un fichero). Cuando ya no
// quedan más filas, devolvería un vacio y tendríamos que controlar ese
// final. Cada fila que devuelve es un array, así que podemos acceder al
// contenido usando la forma: $fila["Nombre de mi campo"];

mysql_close ($serverConnection);
// Cierra la conexión con la base de datos.
```

Ejemplo:

```
<?php
function Conectarse ()
{
	if (!($link=mysql_connect("localhost", "usuario", "password"))) {
		echo "Error conectando a la base de datos.";
		exit();
	}

	if (!mysql_select_db("BaseDatos", $link)) {
		echo "Error seleccionando la base de datos.";
		exit();
	}
	return $link;
}

function ListarNombres ()
{
```

```
        $link = Conectarse();
        $result = mysql_query("select * from alumnos", $link);
        while($row = mysql_fetch_array($result)) {
                printf("%s<br>", $row["Nombre"]);
        }
        mysql_free_result($result);
        mysql_close($link);
}
?>
```

Vistazo rápido a MySQL:

Y ahora un repaso rápido de sql para poder manejar una base de datos. Lo primero es aprender a realizar consultas sobre una tabla:

select { * | columna, … } from tabla, … [where condición] [order by columna];

Después de eso lo siguiente es aprender a insertar y borrar registros:

insert into tabla values (valor, …);
delete from tabla where condición;

Para actualizar un registro de una tabla se emplea lo siguiente:

update tabla set columna = valor, … where condición;

Sabiendo esto, que ya es bastante, podremos manejar una base de datos bastante bien desde nuestra aplicación web. Sin embargo necesitaremos en el inicio del proyecto crear las tablas, y para ello tenemos:

```
create table nombre (
        nombreCampo tipo [not null] [auto_increment],
        …,
        primarykey(campo));
```

Los tipos de datos que tenemos en SQL son: int, date, varchar(tamaño), text. Una vez creada una tabla podemos ver su descripción más adelante con:

describe nombreTabla;

O podemos alterar su contenido usando:

```
alter table nombreTabla  drop columna;
                         add nombre tipo;
```

Para borrar una tabla se realiza con:

delete table nombreTabla;

A un nivel superior tenemos la opción de crear y borrar una base de datos entera. De hecho para poder crear tablas tenemos que crear la base de datos primero:

```
create database nombre;
delete database nombre;
```

Otra forma de ver una posible descripción de una tabla es con:

```
show columns from tabla;
```

Y por último hay un error que dice: "Client does not support authentication protocol…", que se soluciona con las siguientes instrucciones en la consola de comandos de MySQL:

```
set password = old_password ('contraseña');
set password for 'root'@'localhost' = old_password('contraseña');
```

Y eso es todo lo imprescindible para poder manejar bases de dato MySQL con php, y de este modo hacer aplicaciones web.