Programación en PHP a través de ejemplos

Apuntes de la asignatura “Programación para Internet”,
Ingeniería Técnica en Informática de Gestión

Manuel Palomo Duarte
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Cádiz
Ildefonso Montero Pérez
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Sevilla

1 Introducción

El lenguaje PHP (cuyo nombre es acrónimo de PHP: Hipertext Preprocessor) es un lenguaje interpretado con una sintaxis similar a la de C++ o JAVA. Aunque el lenguaje se puede usar para realizar cualquier tipo de programa, es en la generación dinámica de páginas web donde ha alcanzado su máxima popularidad. En concreto, suele incluirse incrustado en páginas HTML (o XHTML), siendo el servidor web el encargado de ejecutarlo.

Algunas de las características de su enorme popularidad son:

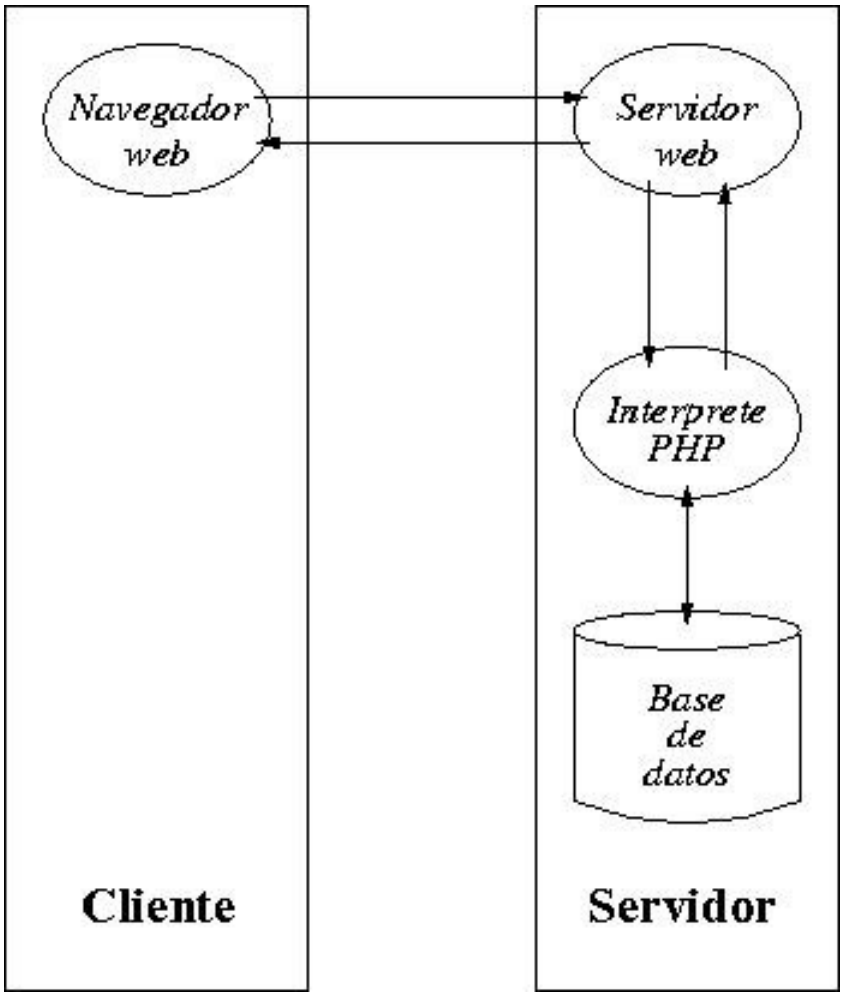
- Es un lenguaje libre. Puede descargarse de http://www.php.net.
- Está disponible para muchos sistemas (GNU/Linux, Windows, UNIX, etc).
- Tiene una extensa documentación oficial en varios idiomas (disponible libremente en http://www.php.net).
- Existen multitud de extensiones: para conectar con bases de datos, para manejo de sockets, para generar documentos PDF, para generar dinámicamente páginas en Flash, etc
- Al ejecutarse en el servidor, los programas PHP lo pueden usar todo tipo de máquinas con todo tipo de sistemas operativos.
- En caso de que un cliente falle (por error hardware, virus, etc) se puede seguir usando el sistema desde otro cualquiera que tenga un navegador web con conexión al servidor.

Este documento enseña a manejar PHP a personas con conocimientos básicos de programación. Para realizar programas son necesarios algunas nociones de HTML (o XHTML), pero se puede ir aprendiendo sobre la marcha con los ejemplos.

2 Inclusión de código PHP en una página HTML

Para incluir código PHP basta con precederlo de la etiqueta <?php, y cerrarlo con ?>. Si el servidor web está correctamente configurado, detectará código PHP y, en vez de proporcionarle el contenido de la página directamente al cliente (lo que significaría que recibiría el código fuente del programa), ejecuta el programa y devuelve su resultado al navegador.

Así pues, el esquema de una petición sería como sigue:



3 Configuración del entorno de trabajo

3.1 Configuración del servidor

Los pasos para configurar un servidor web con soporte para PHP en un sistema GNU/Linux son los siguientes:

- Instalar el sistema GNU/Linux con soporte de red (aunque no tenga tarjeta de red, se puede usar el loopback). Con el comando *ifconfig* se puede comprobar si está activado.
- Instalar el paquete *Apache* con sus dependencias (que contiene el servidor web). Si el manual está disponible (*apache-doc*) se recomienda instalarlo también
- Lanzar el servidor (también conocido como demonio) httpd, invocando al script */etc/init.d/apache2* con el parámetro *start.*
- Probar que *Apache* sirve peticiones. Abrir un navegador web y escribir la URL *localhost* (o 127.0.0.1). Deberá de dar una página de bienvenida como respuesta o decir que no la hay, pero no dar un error de petición rechazada.
- Instalar el paquete *php* (que incluye el lenguaje) y *apache-php* (el paquete que permite conectar Apache con PHP). También se recomienda *php-manual*, el manual oficial.
- Se puede probar PHP desde línea de comando ejecutando *echo "<? print(2+2) ?>" | php* . El resultado debe ser 4.
- Por último hay que comprobar que Apache ejecuta código PHP. Para ello hay que ver el directorio donde Apache busca las páginas web: *grep DocumentRoot /etc/apache2/** . En el directorio que nos indique ejecutamos *echo "<? print(2+3) ?>" > p.php* . Y solicitamos a Apache la URL *localhost/p.php* . Si el resultado es 5, todo está correcto.

Si fuera necesario modificar el comportamiento de PHP, su fichero de configuración es /etc/php.ini

La extensión que suelen tener los programas en PHP es .php o .php seguido del número mayor de la versión de PHP que se usa (.php3 , .php4 , .php5)

Si además se desea trabajar con bases de datos MySQL:

- Instalar el paquete mysql (en algunas distribuciones se llama *mysql-server*) con sus dependencias. Si el manual está disponible se recomienda instalarlo también
- Lanzar el servidor (también conocido como demonio) mysqld, invocando al script */etc/init.d/mysql* con el parámetro *start.*
- Probar que *mysql* sirve peticiones. Abrir una consola y escribir *mysqlshow*. La respuesta que tiene que dar es el listado de bases de datos del sistema.

También se recomienda la instalación de algún entorno para facilitar el trabajo con la base de datos,

como puede ser phpMyAdmin (que está disponible libremente). Este programa se puede instalar como paquete de la distribución o bien bajar el código fuente y colocarlo en un directorio de donde *Apache* ejecute páginas webs.

3.2 Trabajo remoto

También si se desea, se puede trabajar con una máquina remota. Para ello lo más común es tener una cuenta a la que se suban las páginas (por FTP, scp, etc) y solicitar la URL correspondiente a la máquina destino.

3.3 Entorno de trabajo

Se recomienda usar el editor Quanta Plus, que es el editor web del proyecto KDE (forma parte del paquete *kdewebdev*. Existen otras alternativas interesantes: Bluefish, Eclipse, etc. Es importante que el entorno permite funciones como coloreado de sintaxis PHP, entorno gráfico, gestión de proyectos, previsualización de resultado, sincronización con deirectorios remotos, etc

4 Primeros programas en PHP

El objetivo del resto del documento es enseñar PHP a través de ejemplos. Las explicaciones de sintaxis serán mínimas, pues (salvo que se indique lo contrario) la sintaxis es idéntica a la de C.

Los nombres de variables en PHP comienzan por un carácter y van seguidos de números y caracteres sin espacios. Para hacer referencia a una variable debemos anteponer a su nombre el símbolo del dólar ($). Los tipos básicos son:

- Entero: número entero con signo
- Flotante: número decimal con signo
- Booleano: vale true o false
- Cadena de caracteres: cadena de caracteres delimitada por comillas. Las comillas simples interpretan el texto literalmente, mientras que las dobles sustituyen las variables.

No es necesario declarar las variables, simplemente el intérprete averiguará el tipo de dato que almacenará y se declarará automáticamente. Si es necesaria una conversión de tipos, al igual que en C, se puede anteponer el tipo al que se desea promocionar entre paréntesis.

Los comentarios pueden ser de dos tipos:

Para comentarios de unas sólo línea (o parte de ella) se pueden usar indistintamente // o #, que comentan todo lo que se encuentre a continuación de ellos hasta el fin de la línea.

Si se desean comentario de varias líneas, se abren con /* y se cierran con */

Para imprimir en pantalla se puede usar la orden echo o print:

La orden echo es muy similar a la de shell Bash. Recibe como primer y único parámetro una cadena. Esa cadena, si está entre comillas simples, se imprimirá literalmente. Si por el contrario deseamos que se sustituyan las variables que contenga por sus valores hay que usar comillas dobles. Por ejemplo, *$cant=8; echo 'Son $cant euros';* dará como resultado *Son $cant euros*. Pero *$cant=8; echo "Son $cant euros";* imprimirá *Son 8 euros*. Como se observa, las órdenes PHP van terminadas por el carácter ;

Por el contrario, la orden print recibe sus parámetros entre paréntesis. Es una orden que admite muchos más parámetros y opciones. Una de sus principales diferencias es que evalua su parámetro y después lo imprime. Por ejemplo *echo "doble(8)"* da doble(8), pero *print(doble(8))* da 16.

Ejercicio 4.1: Concatena dos cadenas con el operador punto (.) e imprimir su resultado.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <title>ejercicio 4.1</title>
```

```
</head>
<body>

<?php
$ini = "Hola ";
$fin = " a todos";
$todo = $ini.$fin;
echo $todo;
?>

</body>
</html>
```

Ejercicio 4.2: Hacer un programa que sume dos variables que almacenan dos números distintos.

```
<html>
<head>
  <title>ejercicio 4.2</title>
</head>
<body>

<?php

$n1=1;
$n2=2;
$suma=$n1+$n2;
echo "suma = ".$suma. "<br>";
echo "$n1+$n2";

?>

</body>
</html>
```

Ejercicio 4.3: hacer un programa que muestre en pantalla información de PHP con la función phpinfo(). Muestre la información centrada horizontalmente en la pantalla.

```
<html>
<head>
  <title>ejercicio 4.3</title>
</head>

<body>
<center>
<?php

echo phpinfo();

?>
```

```
</center>
</body>
</html>
```

Ejercicio 4.4: Mostrar en pantalla una tabla de 10 por 10 con los números del 1 al 100

```
<html>
<head>
  <title>ejercicio 4.4</title>
</head>

<body>

<?php

echo "<table border=1>";
$n=1;
for ($n1=1; $n1<=10; $n1++)
{
   echo "<tr>";
   for ($n2=1; $n2<=10; $n2++)
   {
      echo "<td>", $n, "</td>";
      $n=$n+1;
   }

   echo "</tr>";
}
echo "</table>";
?>

</body>
</html>
```

Ejercicio 4.5: ídem a 4.4 anterior, pero colorear las filas alternando gris y blanco. Además, el tamaño será una constante: define(TAM, 10)

```
<html>
<head>
  <title>ejercicio 4.5</title>
</head>

<body>

<?php

define(TAM,10);

echo "<table border=1>";
```

```
$n=1;
for ($n1=1; $n1<=TAM; $n1++)
{
   if ($n1 % 2 == 0)
      echo "<tr bgcolor=#bdc3d6>";
   else
      echo "<tr>";
   for ($n2=1; $n2<=TAM; $n2++)
   {
      echo "<td>", $n, "</td>";
      $n=$n+1;
   }

   echo "</tr>";
}
echo "</table>";
?>

</body>
</html>
```

Ejercicio 4.6: mostrar una tabla de 4 por 4 que muestre las primeras 4 potencias de los números del uno 1 al 4 (hacer una función que las calcule invocando la función pow). En PHP las funciones hay que definirlas antes de invocarlas. Los parámetros se indican con su nombre ($cantidad) si son por valor y antecedidos de & (&$cantidad) si son por referencia.

```
<html>
<head>
  <title>ejercicio 4.6</title>
</head>

<body>

<?php

define(TAM,4);
function potencia ($v1, $v2)
{
   $rdo= pow($v1, $v2);
   return $rdo;
}

echo "<table border=1>";
for ($n1=1; $n1<=TAM; $n1++)
{
   echo "<tr>";
   for ($n2=1; $n2<=TAM; $n2++)
      echo "<td>". potencia($n1,$n2). "</td>";

   echo "</tr>";
```

```
}
echo "</table>";
?>

</body>
</html>
```

Ejercicio 4.7: hacer un programa que muestre en una tabla de 4 columnas todas las imágenes de el directorio "fotos". Para ello consulte el manual (en concreto la referencia de funciones de directorios). Suponga que en el directorio sólo existen fotos.

```
<html>
<head>
  <title>ejercicio 4.7</title>
</head>
<body>

<?php

if ($gestor = opendir('fotos'))
{
   echo "<table border=1>";
   echo "<tr>";
   $i=0;
   while (false !== ($archivo = readdir($gestor)))
   {
      if ($archivo!="." && $archivo!="..")
      {
             if ($i==4)
         {
                $i=0;
                    echo "</tr>";
                    echo "<tr>";
             }
         $i++;
             echo "<td>";
             echo "<a href=fotos/$archivo><img src=fotos/$archivo>
</a>";
             echo "</td>";
          }
   }
   echo "</tr>";
   echo "</table>";
   closedir($gestor);
}
?>

</body>
</html>
```

Ejercicio 4.8: ídem al anterior, pero que muestre las fotos en 100x100 y que al pulsar abra la foto entera. Compruebe que sólo muestra fotos con extensión .jpg, .png, bmp o .gif (haga una función que lo compruebe usando las expresiones regulares como aparecen en el manual).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>ejercicio 4.8</title>
</head>
<body>

<?php
echo "<h1>Tabla de Fotos con Enlace</h1>";

function valida_foto($fotos)
{
  $rdo=0;
  if (ereg("[Jj][Pp][Gg]$", $fotos)) rdo=1;
  if (ereg("[Gg][Ii][Ff]$", $fotos)) rdo=1;
  if (ereg("[Pp][Nn][Gg]$", $fotos)) rdo=1;
  if (ereg("[Bb][Mm][Pp]$", $fotos)) rdo=1;

  return $rdo;
}

echo "<table border=1>";
$puntero = opendir('fotos');
$i=1;
while (false !== ($foto = readdir($puntero)))
{
  if ($foto!="." && $foto!=".." && valida_foto($foto))
  {
    if ($i==1)
      echo "<tr>";
    echo "<td><a href='fotos/$foto'>";
    echo "<img src='fotos/$foto' width=100 height=100></img>";
    echo "</a></td>";
    if ($i==4)
    {echo "</tr>"; $i=0;}
    $i++;
  }
}

closedir($puntero);
echo "</table>";
?>
</body>
</html>
```

Ejericio 4.9: ídem al anterior, pero que por cada foto tenga una miniatura. Para la foto playa.jpg la miniatura será MINI-playa.jpg

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>ejercicio 4.9</title>
</head>

<body>
<?php
echo "<h1>Galeria de imagenes con thumbnails</h1>";

function valida_foto($fotos)
{
  $rdo=0;
  if (ereg("[Jj][Pp][Gg]$", $fotos)) rdo=1;
  if (ereg("[Gg][Ii][Ff]$", $fotos)) rdo=1;
  if (ereg("[Pp][Nn][Gg]$", $fotos)) rdo=1;
  if (ereg("[Bb][Mm][Pp]$", $fotos)) rdo=1;

  return $rdo;
}

echo "<table border=1>";
$puntero = opendir('fotos');
$i=1;
while (false !== ($foto = readdir($puntero)))
{
  if ($foto!="." && $foto!=".." && valida_foto($foto))
  {
    if ($i==1)
      echo "<tr>";
    echo "<td><a href='fotos/tumbs/MINI-$foto'>";
    echo "<img src='fotos/$foto' width=100 height=100></img>";
    echo "</a></td>";
    if ($i==4)
    {echo "</tr>"; $i=0;}
    $i++;
  }
}

closedir($puntero);
echo "</table>";
?>
</body>
</html>
```

Ejercicio 4.10: ídem al anterior, pero que si no existe la miniatura de una foto debe de crearla. Para generar la miniatura se usa el programa *convert* (hay que invocarlo en línea de comandos desde PHP son la función *system*).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>ejercicio 4.10</title>
</head>

<body>
<?php
echo "<h1>Galeria de imagenes con thumbnails</h1>";

function valida_foto($fotos)
{
  $rdo=0;
  if (ereg("[Jj][Pp][Gg]$", $fotos)) rdo=1;
  if (ereg("[Gg][Ii][Ff]$", $fotos)) rdo=1;
  if (ereg("[Pp][Nn][Gg]$", $fotos)) rdo=1;
  if (ereg("[Bb][Mm][Pp]$", $fotos)) rdo=1;

  return $rdo;
}

function crea_tumbs($foto)
{
 if (!is_dir('fotos/tumbs'))
   mkdir ('fotos/tumbs', 0777);
 if (!is_file('fotos/tumbs/MINI-$foto'))
   system ("convert -sample 40x40 /fotos/$foto /fotos/tumbs/MINI-
$foto");
}

echo "<table border=1>";
$puntero = opendir('fotos');
$i=1;
while (false !== ($foto = readdir($puntero)))
{
  if ($foto!="." && $foto!=".." && valida_foto($foto))
  {
    crea_tumbs($foto);

    if ($i==1)
      echo "<tr>";
    echo "<td><a href='fotos/tumbs/MINI-$foto'>";
    echo "<img src='fotos/$foto' width=100 height=100></img>";
```

```
    echo “</a></td>";
    if ($i==4)
    {echo "</tr>"; $i=0;}
    $i++;
  }
}

closedir($puntero);
echo "</table>";
?>

</body>
</html>
```

Ejercicio 4.11: PHP desde línea de comandos. Suponga que tenemos un servidor que no soporta PHP. Genere una página estática con la galería de fotos del ejercicio anterior.

Las razones para usar PHP generando contenidos estáticos pueden ser, además de la indicada anteriormente: para facilitar la indexación de contenidos (con spiders), para cargar menos el servidor, para realizar una página que funciona off-line (por ejemplo, una recopilación de información para grabarla en CD/DVD), etc

Simplemente abría que invocar, desde la línea de comandos *php ejercicio4.10.php > pag.html*

Ejercicio 4.12: vectores. Almacene en un vector los 10 primeros número pares. Imprímalos cada uno en una línea (recuerde que el salto de línea en HTML es
).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>ejercicio 4.12</title>
</head>

<body>
<?php

for ($i=0;$i<10;$i++)
 $v[$i]=$i*2;

for ($i=0;$i<10;$i++)
 echo “$v[$i]<br>”;

?>
</body>
</html>
```

Ejercicio 4.13: Imprima los valores del vector asociativo siguiente usando la estructura de control foreach:

```
$v[1]=90;
$v[30]=7;
$v['e']=99;
$v['hola']=43;
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>Ejercicio 4.13</title>
</head>

<body>
<?php
$v[1]=90;
$v[30]=7;
$v['e']=99;
$v['hola']=43;

foreach ($v as $indice => $valor)
{
   echo “El elemento de indice $indice vale $valor <br>”;
}

?>
</body>
</html>
```

5 Interacción con los clientes

Todos los ejemplos vistos anteriormente trabajan de manera independiente del usuario. No existe ninguna interacción a nivel de programa. En este apartado comenzaremos a interactuar. Para ello el elemento clave es el formulario HTML (o XHTML)

Los formularios HTML están delimitados por las marcas *<FORM>* y *</FORM>*. Los formularios que usaremos, en principio, tienen las siguientes características:

- Tienen que contener (entre sus marcas) toda la información necesaria para operar posteriormente. Si hay varios formularios en la página estos no pueden estar anidado y, por lo tanto, serán independientes.

- La marca *<FORM>* tiene que incorporar dos parámetros (por ahora). Uno es constante, e indica cómo se enviarán los datos: *METHOD=POST* y otro indica la página PHP que procesará la información del formulario: *ACTION=pagina.php*. Es importante que la dirección del atributo *ACTION* sea relativa, porque si es absoluta (de la forma *ACTION=http://www.dominio.com/pagina.php* o *ACTION=/directorio/pagina.php*) sólo funcionará en un servidor (o una estructura de directorios) determinado.

- Todo formulario incluirá un elemento tipo *SUBMIT*, que será el que permita al usuario ordenar el procesado de la información.

- Todo elemento del formulario necesitará un nombre para que pueda ser procesado posteriormente (y se recomienda que también incorpore un *VALUE*).

Los elementos principales que se pueden incluir dentro de un formulario son los siguientes (si desea una referencia más completa puede dirigirse al estándar en http://www.w3c.org):

- Cajas de texto: el atributo VALUE indica el valor por defecto, SIZE el tamaño en pantalla y MAXLENGTH la cantidad de caracteres que se podrán escribir en ella.
  <INPUT TYPE="text" NAME="ciudad" VALUE=”pepe” SIZE=8 MAXLENGTH=20>

- Cajas de texto para claves: idénticas a las cajas de texto, pero el texto que se escribe no está visible al usuario.
  <INPUT TYPE="password" NAME="ciudad" SIZE=8 MAXLENGTH=20>

- Botones de selección: permiten elegir uno (y sólo uno) de los elementos agrupados. Es importante que todos los elementos agrupados tengan exactamente el mismo nombre (para que sean excluyentes) y distintos valor en VALUE (que será lo que identifique el seleccionado). Además, si se desea obligar al usuario a que seleccione uno de los elementos hay que poner el atributo *CHECKED* en alguno de ellos (pues en otro caso no aparecería ninguno seleccionado por defecto).

```
<BR><INPUT TYPE="radio" NAME="musica" VALUE="1" checked>Flamenco
<BR><INPUT TYPE="radio" NAME="musica" VALUE="2">Pop
<BR><INPUT TYPE="radio" NAME="musica" VALUE="3">Rock
```

- Cajas de selección: similares a los botones de selección, pero se pueden seleccionar los

elementos que desee (uno, varios o ninguno). En este caso son independiente, por lo que casa uno tiene su nombre y el valor CHECKED si deseamos que por defecto aparezca marcado

<INPUT TYPE=”checkbox” NAME=”publi” VALUE=1> Marque si desea publicidad

- Botón de envío: es un botón que realiza la petición a la página indicada en el atributo *ACTION* del *FORM*. El texto que tiene es el del *VALUE*.

<INPUT TYPE=”submit” VALUE=”Procesar”>

- Campos ocultos: sirven para que el formulario envíe datos que el usuario no vea en pantalla (pero que podría ver en el código fuente de la página, no están ocultos realmente). Se suele usar en las modificaciones de registros de bases de datos. El valor que envían se especifica en *VALUE*.

<INPUT TYPE=”hidden” NAME=”identificador” VALUE=”8”>

- Cajas de texto multilínea: son similares a las cajas de texto, pero especifican sus dimensiones con *COLS* (columnas) y *ROWS* (filas). Además la marca tiene que cerrarse y el valor por defecto se especifica entre la marca que abre y la que cierra (porque puede ser multilínea).

<TEXTAREA NAME=”comentario” COLS=”20” ROWS=”4”>
Ponga aquí su comentario
u opinión
</TEXTAREA>

- Listas desplegables:permite elegir entre uno o varios valores mostrados. Entre la marca de apertura y la de cierre puede haber tantos elementos de selección como se desee. Existe el atributo SIZE que indica las opciones que se verán simultáneamente en pantalla y MULTIPLE que indica si es posible realizar una selección de más de un valor (con el atributo MULTIPLE su función es similar a la de las cajas de selección y sin él a la de los botones de selección).

<SELECT NAME=”provincia”>
<OPTION VALUE=”1” CHECKED>Sevilla
<OPTION VALUE=”2”>Huelva
</SELECT>

Una vez en cliente ha introducido los valores adecuados en los elementos y ha pulsado el botón *SUBMIT*, la página indicada en el atributo *ACTION* del *FORM* se ejecutará.

Para trabajar con los valores de los elementos del formulario se debe poner la función import_request_variables("gp","f_"). Tras ejecutar esta función por cada elemento del formulario de nombre "edad" existirá una variable "$f_edad" con su valor.

En versiones anteriores de PHP no era necesario usar la función anterior, sino que directamente existía para un elemento edad la variable $edad. Ese efecto se consigue activando la directiva *register_globals=on* en el fichero de configuración de PHP. Sin embargo, esta directiva puede acarrear problemas de seguridad, por lo que PHP la trae por defecto desactivada, y es raro encontrar servidores de hosting (alojamiento) que la tengan activada. Más información en el manual.

Una vez explicados los conceptos básicos, vamos a ver ejemplos. A partir de ahora los ejemplos

serán, por lo general dos páginas (una con el formulario y otra que precese), por lo que antes de la marca *DOCTYPE* se pondrá en nombre del fichero.

Ejercicio 5.1: Hacer un euroconversor de euros a pesetas.

ej5.1.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>Ejercicio 5.1</title>
</head>
<body>

<form method=post action=ej5.1.php>
Introduzca la cantidad de euros: <input type=text name=euros
size=10>
<input type=submit name=ok value=enviar>
</form>

</body>
</html>
```

ej5.1.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>ejercicio 5.1</title>
</head>
<body>

<?php
import_request_variables("pg","f_");

echo "Son ";
echo $f_euros*166.386;
echo " pesetas";
?>

</body>
</html>
```

Ejercicio 5.2 Hacer un conversor de euro a pesetas o a dolares (que el usuario elija una moneda y sólo una)

ej5.2.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>Ejercicio 5.2</title>
</head>
<body>

<form method=post action=ej5.2.php>
Introduzca la cantidad: <input type=text name=cantidad size=10>
<input type=submit name=ok value=enviar>
<br>
Seleccione el tipo de conversion:<br>
<input type=radio name=conv value=1 checked>Euros<br>
<input type=radio name=conv value=2>dolares<br>
</form>

</body>
</html>
```

ej5.2.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>ejercicio 5.2</title>
</head>
<body>

<?php
import_request_variables("pg","f_");

echo "Son ";
if ($f_conv==1)
        {
        echo $f_cantidad/166.386;
        echo " euros";
        }

else
        {
        echo $f_cantidad/180.386;
        echo " dolares";
        }
?>

</body>
</html>
```

Ejercicio 5.3 Amplie el ejercicio de la galería de fotos realizada anteriormente y permita al usuario añadir nuevas fotos.
Para ello hay que poner el atributo enc_type=multipart/form-data en el FORM y usar la variable $_FILES['foto']

ej5.3.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>

  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>Galeria 5.3</title>
</head>
<body>

<form enctype="multipart/form-data" action="ej5.3.php"
method="post">

Enviar foto: <input name="foto" type="file">
<input type="submit" value="Enviar">
</form>

</body>
</html>
```

ej5.3.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<body>

<?php
if (is_uploaded_file($_FILES['foto']['tmp_name'])) {
        $nombre = date(DATE_RFC822);
        print($nombre);
        copy($_FILES['foto']['tmp_name'], "fotos/$nombre.jpg");
} else
   echo "Possible file upload attack. Filename: " .
$_FILES['foto']['name']. "---".$_FILES['foto']['tmp_name'];
?>
</body>

</html>
```

Ejercicio 5.4 Realizar el conversor de monedas en una única página creando una máquina de estados.

Máquinas de estados: a veces puede interesar que una misma página se envíe a sí misma información. Para ello se implementa una máquina de estado. La idea es que al principio de la

página se determina si es la primera invocación de la página o si es una llamada a sí misma con información, y se realiza una acción u otra. Por ejemplo:

```
if (is_set($f_estado) && $f_estado==1)
{ echo ...
}

resto_del_prog_principal con HIDDEN
```

Al introducir el elemento HIDDEN el mismo cliente nos indicará (sin que lo sepa) el estado en el que está.

La ventaja que tiene el uso de máquinas de estado es que permite reutilizar código (aunque realmente sería más adecuada escribir el código a reutilizar en un fichero aparte e incluirlo donde se desee) y que se reduce el número de páginas de un proyecto (lo que facilita su gestión, manejo de versiones, etc). Además, al reducir el número de páginas también se reduce las posibilidades de fallo. A veces puede ser que falle una determinada funcionalidad que dependa de dos páginas, lo que implica revisar las dos páginas y comprobar su comunicación. Con una sólo página (con máquina de estados) es más sencillo.

ej5.4.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-
equiv="content-type">
  <title>Ejercicio 5.4</title>
</head>
<body>

<?php
import_request_variables("pg","f_");

if (is_set($f_estado) && $f_estado==1)
{
echo "Son ";
if ($f_conv==1)
  echo $f_cantidad/166.386." euros<BR>";
else
  echo $f_cantidad/180.386." dolares<BR>";
}
?>

<form method=post action=ej5.4.php>
Introduzca la cantidad: <input type=text name=cantidad size=10>
<input type=submit name=ok value=enviar>
<br>
Seleccione el tipo de conversion:<br>
<input type=radio name=conv value=1 checked>Euros<br>
<input type=radio name=conv value=2>dolares<br>
<input type=hidden name=estado value=1>
```

```
</form>

</body>
</html>
```

6 PHP + MySQL

Una de las principales razones de la popularidad de PHP es su capacidad para comunicarse con el sistema gestor de bases de datos MySQL.

MySQL es un sistema gestor de bases de datos libre que funciona sobre una gran cantidad de sistemas operativos (tanto tipo UNIX/Linux/BSD como Windows) y plataformas hardware. Entre sus principales características destacan su bajo consumo de recursos manejando grandes cantidades de datos. Está soportado por la empresa MySQL

Los pasos para trabajar con una base de datos son los siguientes:

Primero es necesario realizar una conexión con el sistema de bases de datos. Para ello se usa la función mysql_connect. Esta función recibe tres cadenas como parámetros: dirección IP del servidor, usuario y clave. Devuelve un manejador de conexión (o cero si se ha producido un error).

```
if(!$link=mysql_connect('127.0.0.1','pepe','secreto'))
 { echo “<a href=/index.html>Error1</a>” ; exit ; }
```

En caso de error hay que comprobar si el equipo tiene activado el servidor de MySQL, si no tiene ningún cortafuegos que no permita acceder a él y si existe el par de usuario y clave suministrado.

Y segundo es seleccionar la base de datos con la que se va a trabajar.

```
if (!mysql_select_db(“biblioteca”)
 { echo “<a href=/index.html>Error2</a>” ; exit; }
```

En caso de error puede ser que no exista la base de datos o que no se tengan permisos para manejarla. En este último caso se podrían dar permisos de la siguiente manera:

```
grant all privileges on acme.* to acme@'localhost' identified by
'acme';
```

Una vez se tiene una conexión y una base de datos seleccionada se puede interactuar con el sistema de bases de datos.

A continuación describiremos brevemente los comandos SQL más usados para las cuatro operaciones más usadas: altas, bajas, modificaciones y listados sobre una tabla con escritores:

- Altas:

Sintaxis para alta de registro:
INSERT INTO tabla (columna1, columna2)
 VALUES ('valor1', valor2);

Ejemplo:
INSERT INTO autores (nombre, apellidos)
VALUES ('Neal','Stephenson');

Tabla Autores

Nombre	*Apellidos*	*Nacionalidad*
Lawrence	Lessig	Estadounidense
Richard M.	Stallman	Estadounidense
Alberto	Noguera	Español
Neal	Stephenson	NULL

- Modificaciones:

Sintaxis para modificación de registro/s:
UPDATE tabla SET columna1='valor1', columna2='valor2', columna3='valor3'
[WHERE columnaN='valorN'];

Ejemplo:
UPDATE autores SET nacionalidad='estadounidense'
WHERE nombre='Neal' and apellidos='Stephenson';

Tabla Autores

Nombre	*Apellidos*	*Nacionalidad*
Lawrence	Lessig	Estadounidense
Richard M.	Stallman	Estadounidense
Alberto	Noguera	Español
Neal	Stephenson	estadounidense

- Bajas:

Sintaxis para baja de registro/s:
DELETE FROM tabla
WHERE campo1='valor1';

Ejemplo:
DELETE FROM autores
WHERE nombre='Neal' or apellido='Noguera';

Tabla Autores

Nombre	*Apellidos*	*Nacionalidad*
Lawrence	Lessig	Estadounidense
Richard M.	Stallman	Estadounidense

- Listados:

Sintaxis para consulta de registro/s:
SELECT columna1, columna2
FROM tabla
WHERE columna3='valor1'

ORDER BY columna2 [ASC | DESC]

Se puede usar * (todas las columnas)

Ejemplo:
SELECT * FROM autores
WHERE nacionalidad='estadounidense'
ORDER BY apellidos DESC

Resultado de la consulta

Nombre	*Apellidos*	*Nacionalidad*
Richard M.	Stallman	Estadounidense
Lawrence	Lessig	Estadounidense

Para manejo de los resultados de una consulta existen varias funciones:

El número de registros: mysql_numrows($result);

La información del campo1 para el registro i-ésimo: $dato=mysql_result($result,$i,"campo1");

Por ejemplo, para recorrer el resultado de una consulta imprimiendo el campo nombre en pantalla:

```
if (mysql_numrows($result) > 0)
 for ($i=0;$i<mysql_numrows($result);$i++)
  echo mysql_result($result,$i,"nombre").“<br>”;
```

Cuando no haga falta la conexión hay que cerrarla: mysql_close($link);

Un mismo manejador puede ejecutar muchas consultas, pero sólo se puede acceder al resultado de la última. El sistema suele cerrar automáticamente las conexiones al terminar de ejecutarse el script. No obstante hay que hacerlo correctamente

Vamos a ver ejemplos de páginas que hacen altas, bajas, modificaciones y listados sobre una tabla llamada empresas que están en una base de datos denominada “buscador”, con los siguientes campos: id (entero autoincrementado, calve primaria), nombre (cadena), web (cadena), telef (cadena), sector (cadena), descrip (cadena que mostraremos multilínea), karma (entero que servirá para ordenar las empresas)

- Alta

alta.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>Alta1</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
```

```
</head>
<body>
<h2>Alta de empresa</h2>
<center>

<FORM action='alta2.php' method='POST'>

<TABLE border=0>
<TR>
<TD>Nombre</TD>
<TD><INPUT type='text' name='nombre' size='30'
maxlength='30'></TD>
</TR>
<TR>
<TD>Web</TD>
<TD><INPUT type='text' name='web' size='30' maxlength='30'></TD>
</TR>
<TR>
<TD>Telef</TD>
<TD><INPUT type='text' name='telef' size='20' maxlength='20'></TD>
</TR>
<TR>
<TD>Sector</TD>
<TD><INPUT type='text' name='sector' size='30'
maxlength='30'></TD>
</TR>
<TR>
<TD>Descrip</TD>
<TD><INPUT type='text' name='descrip' size='50'
maxlength='50'></TD>
</TR>
<TR>
<TD>Karma</TD>
<TD><INPUT type='text' name='karma' size='3' maxlength='3'></TD>
</TR>
</table>

<INPUT type='submit' value='Aceptar'>
</FORM>

</center>
</body>
</html>
```

alta2.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>alta2</title>
```

```
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php

import_request_variables("P","f_");

$linea1="INSERT INTO empresas (nombre, web, telef, sector,
descrip, karma) ";
$linea2=" VALUES ('$f_nombre', '$f_web', '$f_telef', '$f_sector',
'$f_descrip', '$f_karma') ";

$consulta=$linea1.$linea2;

//echo $consulta;

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}


if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}


echo "<br>Alta correcta";
echo "<br><br><a href='alta.html'>Otra alta</a>";
echo "<br><br><a href='index.html'>Inicio</a>";

mysql_close($link);
?>

</body>
</html>
```

- Baja

baja.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>baja1</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php

$linea1="SELECT * FROM empresas ";

$consulta=$linea1;

//echo $consulta;

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}

if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}

echo "<h2>Seleccione empresa/s a dar de baja</h2>";

echo "<CENTER>";
echo "<FORM ACTION=baja2.php METHOD=POST>";

echo "<TABLE BORDER=1>";

for ($i=0;$i<mysql_numrows($result);$i++)
{
     $id=mysql_result($result,$i,"id");
     $nombre=mysql_result($result,$i,"nombre");
```

```
     echo "<TR><TD><INPUT type='checkbox'
name='borrar[$id]'></TD><TD>$nombre</TD></TR>";
}

echo "</TABLE>";
echo "<INPUT type='submit' value='Borrar'>";
echo "</FORM>";

echo "</CENTER>";

mysql_close($link);
?>

</body>
</html>
```

baja2.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title></title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php

import_request_variables("P","f_");

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}

foreach ($f_borrar as $indice => $valor)
{
     if ($valor=="on")
     {
```

```
     $linea1="DELETE FROM empresas ";
     $linea2=" WHERE id='$indice' ";
     $consulta=$linea1.$linea2;

     //echo "$consulta";

     if ( ! $result=mysql_query($consulta,$link))
     {
          echo "<a href=index.html>Error en el borrardo</a>";
          exit;
     }
     }
}

echo "<br>Borrado correcto";
echo "<br><br><a href='baja.php'>Otra baja</a>";
echo "<br><br><a href='index.html'>Inicio</a>";

mysql_close($link);
?>

</body>
</html>
```

- Modificaciones

modif.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>modif1</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php

$linea1="SELECT * FROM empresas ";

$consulta=$linea1;

//echo $consulta;

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
```

```
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}


if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}

echo "<h2>Seleccione empresa/s a dar modificar</h2>";

echo "<CENTER>";
echo "<FORM ACTION=modif2.php METHOD=POST>";

echo "<TABLE BORDER=1>";

for ($i=0;$i<mysql_numrows($result);$i++)
{
     $id=mysql_result($result,$i,"id");
     $nombre=mysql_result($result,$i,"nombre");

     echo "<TR><TD><INPUT type='radio' name='modif'
value='$id'></TD><TD>$nombre</TD></TR>";
}

echo "</TABLE>";

echo "<INPUT type='submit' value='Modif'>";

echo "</FORM>";
echo "</CENTER>";

mysql_close($link);
?>

</body>
</html>
```

modif2.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
```

```
<head>
  <title>modif2</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php

import_request_variables("P","f_");

$linea1="SELECT * FROM empresas  ";
$linea2=" WHERE id='$f_modif' ";

$consulta=$linea1.$linea2;

//echo $consulta;

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}


if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}
?>

<h2>Modif de empresa</h2>
<center>

<FORM action='modif3.php' method='POST'>

<TABLE border=0>
<TR>
<TD>Nombre</TD>
<TD><INPUT type='text' name='nombre' size='30' maxlength='30'
value='<?php print(mysql_result($result,0,'nombre')); ?>' ></TD>
</TR>
<TR>
```

```
<TD>Web</TD>
<TD><INPUT type='text' name='web' size='30' maxlength='30'
value='<?php print(mysql_result($result,0,'web')); ?>'></TD>
</TR>
<TR>
<TD>Telef</TD>
<TD><INPUT type='text' name='telef' size='20' maxlength='20'
value='<?php print(mysql_result($result,0,'telef')); ?>'></TD>
</TR>
<TR>
<TD>Sector</TD>
<TD><INPUT type='text' name='sector' size='30' maxlength='30'
value='<?php print(mysql_result($result,0,'sector')); ?>'></TD>
</TR>
<TR>
<TD>Descrip</TD>
<TD><INPUT type='text' name='descrip' size='50' maxlength='50'
value='<?php print(mysql_result($result,0,'descrip')); ?>'></TD>
</TR>
<TR>
<TD>Karma</TD>
<TD><INPUT type='text' name='karma' size='3' maxlength='3'
value='<?php print(mysql_result($result,0,'karma')); ?>'></TD>
</TR>
</table>

<INPUT type='hidden' name='id' value='<?php
print(mysql_result($result,0,'id')); ?>'>

<INPUT type='submit' value='Aceptar'>
</FORM>

</center>

<?php
mysql_close($link);
?>

</body>
</html>
```

modif3.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>modif3</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
```

```
</head>
<body>

<?php

import_request_variables("P","f_");

$linea1="UPDATE empresas ";
$linea2="  SET nombre='$f_nombre', web='$f_web', telef='$f_telef',
sector='$f_sector', descrip='$f_descrip', karma='$f_karma' ";
$linea3=" WHERE id='$f_id' ";

$consulta=$linea1.$linea2.$linea3;

echo $consulta;

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}


if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}


echo "<br>Modif correcta";
echo "<br><br><a href='modif.php'>Otra modif</a>";
echo "<br><br><a href='index.html'>Inicio</a>";

mysql_close($link);
?>
</body>
</html>
```

- Consulta de todas las empresas ordenadas por nombre

consulta1.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
```

```
<head>
  <title>Consulta1</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php

$linea1="SELECT * FROM empresas ORDER BY nombre";

$consulta=$linea1;

//echo $consulta;

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}

if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}

echo "<h2>Empresas</h2>";

echo "<CENTER>";

echo "<TABLE BORDER=1>";

echo "<TR><TD>Nombre</TD><TD>Web</TD><TD>Telef.</TD>
<TD>Sector</TD><TD>Descrip.</TD><TD>Karma</TD></TR>";
for ($i=0;$i<mysql_numrows($result);$i++)
{
     echo "<TR>";

     $nombre=mysql_result($result,$i,"nombre");
     echo "<TD>$nombre</TD>";

     $web=mysql_result($result,$i,"web");
```

```
      echo "<TD>$web</TD>";

      $telef=mysql_result($result,$i,"telef");
      echo "<TD>$telef</TD>";

      $sector=mysql_result($result,$i,"sector");
      echo "<TD>$sector</TD>";

      $descrip=mysql_result($result,$i,"descrip");
      echo "<TD>$descrip</TD>";

      $karma=mysql_result($result,$i,"karma");
      echo "<TD>$karma</TD>";

      echo "</TR>";
}

echo "</TABLE>";
echo "</CENTER>";

mysql_close($link);
?>

</body>
</html>
```

- Consulta de todas las empresas ordenadas por sector y karma con franjas opcionales

consulta2.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>consulta2</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<FORM action='consulta2.php' method='POST'>
  <INPUT type='checkbox' name='cambio'> Alternar franjas <br>
<INPUT type='submit' value='Generar informe'>
</FORM>

</body>
</html>
```

consulta2.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>consulta2</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php

import_request_variables("p","f_");

$linea1="SELECT * FROM empresas ORDER BY sector, karma DESC";

$consulta=$linea1;

//echo $consulta;

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}


if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}

echo "<h2>Empresas</h2>";

echo "<CENTER>";

echo "<TABLE BORDER=1>";

echo
"<TR><TD>Nombre</TD><TD>Web</TD><TD>Telef.</TD><TD>Sector</TD><TD>
Descrip.</TD><TD>Karma</TD></TR>";
```

```
for ($i=0;$i<mysql_numrows($result);$i++)
{
     if ($f_cambio && ($i%2))
          echo "<TR bgcolor='#B6B7B7'>";
     else
          echo "<TR bgcolor='white'>";


     print("<TD>".mysql_result($result,$i,"nombre")."</TD>");

     print("<TD>".mysql_result($result,$i,"web")."</TD>");

     print("<TD>".mysql_result($result,$i,"telef")."</TD>");

     print("<TD>".mysql_result($result,$i,"sector")."</TD>");

     print("<TD>".mysql_result($result,$i,"descrip")."</TD>");

     print("<TD>".mysql_result($result,$i,"karma")."</TD>");

     echo "</TR>";
}

echo "</TABLE>";

echo "</CENTER>";

mysql_close($link);
?>

</body>
</html>
```

- Consulta de todas las empresas limitando la descripción y con hiperenlaces

consulta3.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>consulta3</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php
```

```
$linea1="SELECT * FROM empresas ORDER BY nombre";

$consulta=$linea1;

//echo $consulta;

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}

if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}

echo "<h2>Empresas</h2>";
echo "<CENTER>";
echo "<TABLE BORDER=1>";

echo "<TR><TD>Nombre</TD><TD>Telef.</TD><TD>Sector</TD>
<TD>Descrip.</TD><TD>Karma</TD></TR>";
for ($i=0;$i<mysql_numrows($result);$i++)
{
     echo "<TR>";

     $web=mysql_result($result,$i,"web");

     print("<TD><A
href=http://$web>".mysql_result($result,$i,"nombre")."</A></TD>");

     print("<TD>".mysql_result($result,$i,"telef")."</TD>");

     print("<TD>".mysql_result($result,$i,"sector")."</TD>");

     $descrip=mysql_result($result,$i,"descrip");
     if (strlen($descrip)>15)
          print("<TD>".substr($descrip,0,14)."...</TD>");
     else
          print("<TD>$descrip</TD>");

     print("<TD>".mysql_result($result,$i,"karma")."</TD>");
```

```
     echo "</TR>";
}

echo "</TABLE>";
echo "</CENTER>";

mysql_close($link);
?>

</body>
</html>
```

- Consulta de todas las empresas ordenando como se desee

consulta4.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>consulta4</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php

import_request_variables("gp","f_");

function enlace ($campo, $orden, $actual)
{
     if (strcmp($campo,$actual))
          $rdo= "<A
href='consulta4.php?campo=$actual&orden=ASC'>".ucfirst($actual)."<
/A>";
          else
          if ( ! strcmp($orden,"ASC"))
               $rdo="<A
href='consulta4.php?campo=$actual&orden=DESC'>".ucfirst($actual)."
</A>";
          else
               $rdo="<A
href='consulta4.php?campo=$actual&orden=ASC'>".ucfirst($actual)."<
/A>";

     return $rdo;
}
```

```
if ( ! isset ($f_campo))
     $linea1="SELECT * FROM empresas ORDER BY nombre";
else
     $linea1="SELECT * FROM empresas ORDER BY $f_campo $f_orden";

$consulta=$linea1;

//echo $consulta;

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}

if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}

echo "<h2>Empresas</h2>";

echo "<CENTER>";

echo "<TABLE BORDER=1>";

echo "<TR>";

print ("<TD>".enlace($f_campo, $f_orden, "nombre")."</TD>");
print ("<TD>".enlace($f_campo, $f_orden, "web")."</TD>");
print ("<TD>".enlace($f_campo, $f_orden, "telef")."</TD>");
print ("<TD>".enlace($f_campo, $f_orden, "sector")."</TD>");
print ("<TD>".enlace($f_campo, $f_orden, "descrip")."</TD>");
print ("<TD>".enlace($f_campo, $f_orden, "karma")."</TD>");

echo "</TR>";

for ($i=0;$i<mysql_numrows($result);$i++)
{
     echo "<TR>";

     $nombre=mysql_result($result,$i,"nombre");
```

```
     echo "<TD>$nombre</TD>";

     $web=mysql_result($result,$i,"web");
     echo "<TD>$web</TD>";

     $telef=mysql_result($result,$i,"telef");
     echo "<TD>$telef</TD>";

     $sector=mysql_result($result,$i,"sector");
     echo "<TD>$sector</TD>";

     $descrip=mysql_result($result,$i,"descrip");
     echo "<TD>$descrip</TD>";

     $karma=mysql_result($result,$i,"karma");
     echo "<TD>$karma</TD>";

     echo "</TR>";
}

echo "</TABLE>";

echo "</CENTER>";

mysql_close($link);
?>

</body>
</html>
```

- Consulta de empresas de un sector

funciones.php

```
<?php

function conectar($bdd)
{
if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db($bdd))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}

return $link;
```

```
}

function generar_select($campo)
{
conectar("buscador");

    echo "<select name=$campo>";

    $linea1="select * from empresas group by $campo";
    $sql = $linea1;
    $result = mysql_query($sql);


    if (!$result)
    {
        echo "F generar_select: Error al acceder a la base de
datos" ;
        echo "<a href='./index.html'>Continuar</a><BR>\n";
        exit;
    }
    else
    {
        for($i=0; $i < mysql_numrows($result); $i++)
        {
        $v=mysql_result($result, $i, $campo);

        echo "<option value='$v'>";
        echo "$v";
        echo "</option>";
        echo "\n";
        }
    }

    echo "</select>";

}

?>
```

consulta5.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>consulta5</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>
```

```
<FORM action='consulta5-2.php' method='POST'>
  Seleccione sector

<?php
include_once("funciones.php");
generar_select("sector");
?>

<INPUT type='submit' value='Generar informe'>
</FORM>

</body>
</html>
```

consulta5-2.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>consulta5-2</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php

import_request_variables("p","f_");

$linea1="SELECT * FROM empresas WHERE sector='$f_sector' ORDER BY
nombre ";

$consulta=$linea1;

//echo $consulta;

if ( ! $link=mysql_connect('localhost','root',''))
{
     echo "<a href=index.html>Error al conectar</a>";
     exit ;
}

if ( ! mysql_select_db("buscador"))
{
    echo "<a href=index.html>Error al seleccionar BDD</a>";
    exit;
}
```

```
if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}

echo "<h2>Empresas</h2>";

echo "<CENTER>";

echo "<TABLE BORDER=1>";

echo
"<TR><TD>Nombre</TD><TD>Web</TD><TD>Telef.</TD><TD>Sector</TD><TD>
Descrip.</TD><TD>Karma</TD></TR>";

for ($i=0;$i<mysql_numrows($result);$i++)
{
     if ($f_cambio && ($i%2))
          echo "<TR bgcolor='#B6B7B7'>";
     else
          echo "<TR bgcolor='white'>";


     print("<TD>".mysql_result($result,$i,"nombre")."</TD>");

     print("<TD>".mysql_result($result,$i,"web")."</TD>");

     print("<TD>".mysql_result($result,$i,"telef")."</TD>");

     print("<TD>".mysql_result($result,$i,"sector")."</TD>");

     print("<TD>".mysql_result($result,$i,"descrip")."</TD>");

     print("<TD>".mysql_result($result,$i,"karma")."</TD>");

     echo "</TR>";
}

echo "</TABLE>";

echo "</CENTER>";

mysql_close($link);
?>

</body>
</html>
```

- Consulta telefónica

consulta6.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>consulta6</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<FORM action='consulta6.php' method='POST'>
  Introduzca telef. <INPUT type='text' name='telef'> <br>
<INPUT type='submit' value='Generar informe'>
</FORM>

</body>
</html>
```

consulta6.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
  <title>consulta6</title>
  <meta name="GENERATOR" content="Quanta Plus">
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>

<?php
include_once("funciones.php");

import_request_variables("p","f_");

$link=conectar("buscador");

//$linea1="SELECT * FROM empresas where telef='$f_telef' ORDER BY
nombre";
$linea1="SELECT * FROM empresas where telef LIKE '%$f_telef%'
ORDER BY nombre";

$consulta=$linea1;
```

```
//echo $consulta;


if ( ! $result=mysql_query($consulta,$link))
{
     echo "<a href=index.html>Error en la consulta</a>";
     exit;
}

echo "<h2>Empresas</h2>";

echo "<CENTER>";

echo "<TABLE BORDER=1>";

echo "<TR><TD>Nombre</TD><TD>Web</TD><TD>Telef.</TD>
<TD>Sector</TD><TD>Descrip.</TD><TD>Karma</TD></TR>";

for ($i=0;$i<mysql_numrows($result);$i++)
{
     echo "<TR>";
     print("<TD>".mysql_result($result,$i,"nombre")."</TD>");
     print("<TD>".mysql_result($result,$i,"web")."</TD>");
     print("<TD>".mysql_result($result,$i,"telef")."</TD>");
     print("<TD>".mysql_result($result,$i,"sector")."</TD>");
     print("<TD>".mysql_result($result,$i,"descrip")."</TD>");
     print("<TD>".mysql_result($result,$i,"karma")."</TD>");

     echo "</TR>";
}

echo "</TABLE>";

echo "</CENTER>";

mysql_close($link);
?>

</body>
</html>
```

7 Sesiones en PHP

Una de las principales limitaciones del protocolo HTTP es que no permite relacionar peticiones consecutivas en sesiones. El concepto de sesión es importante, porque permite que no tengan que reenviar los mismos datos en todas las peticiones (y respuestas) entre cliente y servidor.

Para solucionar este problema, PHP proporciona manejo de sesiones, que permite conservar datos de una peterminada petición en peticiones de la misma máquina. Para ello cada petición de un cliente recibe un identificador único llamado "session_id". Este valor se puede almacenar en una cookie o bien propagarlo en la URL.

Para crear una sesión se usa la función session_start(). A partir del momento en que se invoque, se pueden almacenar variables en ella con la siguiente sintaxis:
$_SESSION['variable1'] = 8;

Para disponer de las variables en páginas sucesivas tienen que invocar la función session_start(). Con la función session_register() también se puede iniciar sesiones, almacenar variables en ellas y hacer accesibles variables de la sesión creada anteriormente.

Es importante remarcar que tanto la función session_start() como session_register tienen que invocarse antes de enviar ningún dato al cliente. Es decir, tienen que ir antes de la marca <HTML> e, incluso, antes de ningún retorno de carro, espacio o tabulador.

Para eliminar una variable de una sesión se podría hacer: unset($_SESSION['variable1']);

Y para eliminar la sesión session_destroy();

Por ejemplo, si queremos tener una página privada a la que sólo se puede acceder si existe una sesion con la variable "acceso" con valor 1, sería necesario el siguiente código:

index.html

```
<FORM METHOD=POST ACTION=acceso.php>
<INPUT TYPE=TEXT NAME=nombre>
<INPUT TYPE=PASSWORD NAME=clave>
<INPUT TYPE=SUBMIT>
</FORM>
```

acceso.php

```
<?php
import_request_variables("P","f_");

if (!strcmp($f_nombre,"root") && !strcmp($f_clave,"super"))
{
 session_start();
 $_SESSION['acceso']=1;
}
echo "<a href=privada.php>Acceder a la página privada</a>";
```

privada.php

```
<?php
session_start();
if ($_SESSION['acceso']!="1")
{
 header("Location:index.html");
 exit;
}
echo "Bienvenido a la página privada";
echo "<a href=salir.php>Salir del sistema</a>";
?>
```

salir.php

```
<¿php
session_destroy();
echo "Ha salido del sistema";
echo "<a href=index.html>Volver al inicio</a>";
?>
```

A la hora de trabajar se puede simplificar el manejo creando varios ficheros que comprueban los permisos de acceso de diferentes niveles. Por ejemplo, se puede crear un fichero que autentique dos tipos de acceso (1 o 2):

index.php

```
<?php
import_request_variables("P","f_");

if($var=tipo_acceso($f_nombre,$f_clave))
{
 session_start();
 $_SESSION['acceso']=$var;
 header("Location:inicio.php");
}
else
{
 echo "<h2>Clave incorrecta</h2>";
}
?>
<FORM METHOD=POST ACTION=index.php>
<INPUT TYPE=TEXT NAME=nombre>
<INPUT TYPE=PASSWORD NAME=clave>
<INPUT TYPE=SUBMIT>
</FORM>
```

seguridad1.php

```
<?php
session_start();
```

```
if ($_SESSION['acceso']!="1")
{
 header("Location:index.php");
 exit;
}
?>
```

seguridad2.php

```
<?php
session_start();
if ($_SESSION['acceso']!="2")
{
 header("Location:index.php");
 exit;
}
?>
```

salir.php

```
<¿php
session_destroy();
echo "Ha salido del sistema";
echo "<a href=index.php>Volver al inicio</a>";
?>
```

Y el resto de páginas del sistema tendrían como primera línea una de las dos siguientes según el nivel de seguridad al que pertenezcan:

```
<?php include("seguridad1.php") ?>
<?php include("seguridad2.php") ?>
```

Orientación a objetos en PHP

PHP se ha centrado siempre en el paradigma de la programación estructurada, pero eran muchas las voces que pedian que este lenguaje de programación fuese poco a poco dando paso a otros paradigmas de programación, de entre los cuales el más demandado era sin lugar a dudas el relativo a la orientación de objetos.

Así pues paso a paso en sucesivas versiones, comenzo muy discretamente en la version 3 incrementandose el número de funcionalidades posibles en cada revisión, se empezaron a incluir diversos mecanismos con los que poder llevar a cabo aplicaciones basadas en el paradigma anteriormente citado y conocido por todos.

Pero es que, conscientes de la potencia de otras estructuras no propias de la orientación de objetos a la que nos acostumbran lenguajes tales como C++, como pueden ser las interfaces propias del mundo Java, PHP ha ido introduciendo todas estos artefactos para que la facilidad de adaptación de programadores procedentes de estos lenguajes sea representada con una curva de aprendizaje suave.

Vamos a ver las estructuras que PHP nos proporciona para llevar a cabo la orientación a objetos.

Clases

Se aplica el concepto de clase que todos conocemos, entidad contenedora de información basada en atributos y en métodos de construcción, modificación y consulta de dichos atributos necesarios y suficientes para representar un objeto con el que trabajar o procesar información relativa al dominio del problema en el que nos estamos moviendo.

Para esto se define la palabra reservada **`class`**. Vamos a ver un ejemplo de definición de una clase en PHP.

usuario.class.php

```
1 //-------------------------------------------------------------
2 // Creación de la clase Usuario
3 // Autor: Ildefonso Montero Pérez - monteroperez{arroba}us.es
4 //-------------------------------------------------------------
5 <?php
6  class Usuario{
7
8 // ATRIBUTOS
9   private $nombreDeUsuario;
10  private $palabraClave;
11
12 // CONSTRUCTOR DE LA CLASE
13  public function Usuario() { }
14
15 // FUNCIONES CONSULTORAS
16  public function getNombreDeUsuario() { return $this->nombreDeUsuario; }
17  public function getPalabraClave() { return $this->palabraClave; }
18
19  // FUNCIONES MODIFICADORAS
20  public function setNombreDeUsuario($nu) { $this->nombreDeUsuario = nu; }
21  public function setPalabraClave($pc) { $this->palabraClave = $pc; }
```

```
22
23 }
24 ?>
```

Vamos a comentar detalladamente el ejemplo que se adjunta. En él estamos definiendo una clase que representará a un usuario de una aplicación. Es un ejemplo muy básico en el que se introducirá unicamente el nombre de usuario y su contraseña, típico ejemplo de un sistema de autenticación. Para ello definiremos dos atributos denominados `nombredeUsuario` y `palabraClave` los cuales podremos consultar y modificar mediante sus respectivas funciones consultoras y modificadoras, más comunmente conocidas como funciones *getters* y *setters.*

Podemos comprobar en las cuatro primeras lineas que esta permitido el tipo de comentario propio de C++ (//) y vemos que a partir de la sexta linea comenzamos con la definicion de la clase mediante la palabra reservada `class` seguida del nombre de la clase, en este caso Usuario.

Los atributos son definidos como variables con visibilidad privada mediante la palabra clave `private`, tambien podriamos optar por hacer uso de `var` pero de esta forma estas variables serian publicas y no nos interesa Esto se define en las lineas nueve y diez.

A continuación definimos el constructor en la linea trece como una función definida como publica mediante `public function` y donde como vemos no es necesario introducir el tipo de objeto que se devuelve tras la llamada a la función o metodo como ocurre en Java o C++, es decir, nos olvidamos de indicar si la función devuelve un entero, un char o un tipo determinado, o si ni siquiera devuelve nada (void). No es necesario en PHP.

Así pues ya tenemos definida toda la clase. Merece especial atención la variable `$this` la cual corresponde como equivalente a `this` en C++ o Java, es decir, una instancia del propio objeto que estamos modelando.

A continuación vamos a ver un ejemplo de uso de esta clase en el que crearemos una instancia de un usuario e introduciremos unos datos concretos.

veranoazul.php

```
1 <?php
2  require_once('usuario.class.php');
3  $usuario = new Usuario();
4  $usuario->setNombreDeUsuario('chanquete');
5  $usuario->setPalabraClave('hamuerto');
6
7  echo 'Nombre de Usuario: '.$usuario->getNombreDeUsuario().'<br>';
8  echo 'Palabra Clave: '.$usuario->getPalabraClave().'<br>';
9
10 $usuario->setPalabraClave('nonosmoveran');
11
12 echo 'Nombre de Usuario: '.$usuario->getNombreDeUsuario().'<br>';
13 echo 'Palabra Clave: '.$usuario->getPalabraClave().'<br>';
14 ?>
```

En este ejemplo vemos como hacer uso de la clase que hemos definido en el fichero

usuario.class.php con lo que necesitamos que se incluya la definicion contenida en este fichero de la clase Usuario. Para ello hacemos uso de `require_once`, esta es la opción más óptima para la carga de definiciones ya que forzamos que solo se incluya una única vez dentro de todo el código.

A partir de la linea tres empezamos a trabajar con la clase anteriormente definida. Para ello creamos una variable denominada `$usuario` y mediante `new` construimos una nueva instancia de la clase Usuario, a la que vamos a inicializar sus atributos en las lineas cuatro y cinco y posteriormente en la diez. Veamos la salida en el navegador de este código:

```
Nombre de Usuario: chanquete
Palabra Clave: hamuerto
Nombre de Usuario: chanquete
Palabra Clave: nonosmoveran
```

Herencia

PHP nos permite hacer uso de herencia entre clases, para ello se introduce la palabra clave **`extends`**. A continuación vamos a ver un ejemplo de como aplicar herencia entre dos clases PHP y como referenciar a la superclase desde una clase hija.

El ejemplo se centrará en una clase que heredará de la clase anteriormente comentada Usuario para crear la clase MiUsuario.

miusuario.class.php

```
1 <?php
 require_once('usuario.class.php');
 class MiUsuario extends Usuario {
 ...
 public function unaFuncionCualquiera( ... ) {
 ...
    echo parent::getNombreDeUsuario();
 }
 ...
}
?>
```

Como se puede observar se hace uso de **`extends`** para heredar de la clase padre y mediante **`parent::`** acceder a la misma.

Interfaces

El concepto de interfaz es procedente del mundo Java. Una interfaz es una entidad que encierra funcionalidades que pueden ser implementadas por diversos objetos. Son muy utiles para poder permitir identificar rapidamente las funcionalidades que un sistema software nos proporciona sin necesidad de acceder a ver los métodos de las clases que lo conformen.

Esto se lleva a cabo mediante la palabra reservada **`interface`**. Veamos un ejemplo de interfaz

escrita en PHP:

idataaccess.class.php

```
1 //-------------------------------------------------------------
2 // Implementación de la interfaz de acceso a base de datos
3 // Autor: Ildefonso Montero Pérez - monteroperez@us.es
4 //-------------------------------------------------------------
5 <?php
6  interface IDataAccess{
7   public function execute($sqlquery);
8   public function debugQuery($sqlquery);
9 }
10 ?>
```

Como podemos observar la definición es identica. En este caso dado que estamos definiendo un contenedor de funcionalidades, iremos introduciendo funciones dentro de esta definición.

No tiene sentido que una interfaz sea definida si no va a existir por lo menos un objeto que sea capaz de implementarla. Esto quiere decir que necesitaremos definir una clase que implemente los metodos contenidos en esta interfaz. Veamoslo para el ejemplo anteriormente citado en el que la interfaz encierra funcionalidades típicas de una fachada de confrontacion de queries a una base de datos.

mysqldataaccess.class.php

```
1 //-------------------------------------------------------------
2 // Clase que confronta queries contra bbdd de tipo MySQL
3 // Autor: Ildefonso Montero Pérez - monteroperez@us.es
4 //-------------------------------------------------------------
5 <?php
6  class MySQLDataAccess implements IDataAccess{
   ...
   public function execute($sqlquery) { ... }
   public function debugQuery($sqlquery) { ... }
   ...
 }
?>
```

En este ejemplo hemos introducido una clase que lleva a cabo la implementación de las funcionalidades introducidas en la interfaz definida anteriormente. Para ello podemos observar que se añade la palabra reservada **`implements.`**