PROGRAMACIÓN EN PHP

Capítulo.1.- Instalación de Apache+MySQL+PHP+OpenLDAP+OpenSSL

Ver documento adjunto

Capítulo 2.- Conceptos básicos

El lenguaje PHP es un lenguaje de programación de estilo clásico, es decir que es un lenguaje de programación con variables, sentencias condicionales, bucles, funciones, etc. No es un lenguaje de etiquetas como podría ser HTML, XML o WML. Está mas cercano a JavaScript o a C, para aquellos que conocen estos lenguajes.

Pero a diferencia de Java o JavaScript que se ejecutan en el navegador, PHP se ejecuta en el servidor, por eso nos permite acceder a los recursos que tenga el servidor como por ejemplo podría ser una base de datos. El programa PHP es ejecutado en el servidor y el resultado enviado al navegador. El resultado es normalmente una página HTML pero igualmente podría ser una pagina WML.

Al ser PHP un lenguaje que se ejecuta en el servidor no es necesario que su navegador lo soporte, es independiente del navegador, sin embargo para que las páginas PHP funcionen, el servidor donde están alojadas debe soportar PHP.

2.1.- Nuestro primer PHP

La ventaja que tiene PHP sobre otros lenguajes de programación que se ejecutan en el servidor (como podrían ser los script CGI Perl), es que nos permite intercalar las sentencias PHP en las páginas HTML, es un concepto algo complicado de entender si no se ha visto nunca como funciona unas paginas PHP o ASP.

Vamos a ver un ejemplo sencillo para comprenderlo mejor. En azul está el código HTML y en rojo el código PHP. Seguiremos este criterio durante todo el manual.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>

<body>

Parte de HTML normal.
<BR><BR>

<?php
   echo "Parte de PHP<br>";

   for($i=0;$i<10;$i++)
   {
      echo "Linea ".$i."<br>";
   }
?>

</body>
</html>
```

El código PHP ejecutado tiene dos partes: la primera imprime "Parte de PHP" y la segunda es un bucle que se ejecuta 10 veces de 0 a 9, por cada vez que se ejecuta se escribe una línea, la variable $i contiene el número de línea que se está escribiendo.:

```
Parte de HTML normal.

Parte de PHP
Linea 0
Linea 1
Linea 2
Linea 3
Linea 4
Linea 5
Linea 6
Linea 7
Linea 8
Linea 9
```

2.2.- Variables

Una variable es un contenedor de información, en el que podemos meter números enteros, números decimales, carácteres, etc. El contenido de las variables se puede leer y se puede cambiar durante la ejecución de una página PHP.

En PHP todas las variables comienzan con el símbolo del dólar $ y no es necesario definir una variable antes de usarla. Tampoco tienen tipos, es decir que una misma variable puede contener un número y luego puede contener carácteres.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
   $a = 1;
   $b = 3.34;
   $c = "Hola Mundo";
   echo $a,"<br>",$b,"<br>",$c;
?>
</body>
</html>
```

En este ejemplo hemos definido tres variables, $a, $b y $c y con la instrucción echo hemos impreso el valor que contenían, insertando un salto de línea entre ellas.

```
1
3.34
Hola Mundo
```

Existen 2 tipos de variables, las variables locales que solo pueden ser usadas dentro de funciones y las variables globales que tienen su ámbito de uso fuera de las funciones, podemos acceder a una variable global desde una función con la instrucción `global nombre_variable;`

2.3.- Aritméticos

Los operadores de PHP son muy parecidos a los de C y JavaScript, si usted conoce estos lenguajes le resultaran familiares y fáciles de reconocer.

Estos son los operadores que se pueden aplicar a las variables y constantes numéricas.

Operador	Nombre	Ejemplo	Descripción
+	Suma	5 + 6	Suma dos números
-	Resta	7 - 9	Resta dos números
*	Multiplicación	6 * 3	Multiplica dos números
/	División	4 / 8	Divide dos números
%	Módulo	7 % 2	Devuelve el resto de dividir ambos números, en este ejemplo el resultado es 1
++	Suma 1	$a++	Suma 1 al contenido de una variable.
--	Resta 1	$a--	Resta 1 al contenido de una variable.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
   $a = 8;
   $b = 3;
   echo $a + $b,"<br>";
   echo $a - $b,"<br>";
   echo $a * $b,"<br>";
   echo $a / $b,"<br>";
   $a++;
   echo $a,"<br>";
   $b--;
   echo $b,"<br>";
?>
</body>
</html>
```

```
11
5
24
2.6666666666667
9
2
```

2.4.- Comparación

Los operadores de comparación son usados para comparar valores y así poder tomar decisiones.

Operador	Nombre	Ejemplo	Devuelve verdaderocuando:
==	Igual	$a == $b	$a es igual $b
===	Idéntico	$a === $b	$a es igual $b y ambos son del mismo tipo
!=	Distinto	$a != $b	$a es distinto $b
!==	No idéntico	$a !== $b	$a no es idéntico a $b
<	Menor que	$a < $b	$a es menor que $b
>	Mayor que	$a > $b	$a es mayor que $b
<=	Menor o igual	$a <= $b	$a es menor o igual que $b
>=	Mayor o igual	$a >= $b	$a es mayor o igual que $b

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
   $a = 8;
   $b = 3;
   $c = 3;
   echo $a == $b,"<br>";
   echo $a != $b,"<br>";
   echo $a < $b,"<br>";
   echo $a > $b,"<br>";
   echo $a >= $c,"<br>";
   echo $b <= $c,"<br>";
?>
</body>
</html>
```

```
1

1
1
1
```

```
$var1 = 1;          // Asignación$var2 = 1;$var3 = “1”;($var1
== $var2)    // Cierto, son iguales($var1 == $var3)  // Son
iguales (tras conversión)($var1 === $var2)     // Cierto, son
idénticas($var1 === $var3)// FALSO, el tipo no coincide
```

2.5.- Lógicos

Los operadores lógicos son usados para evaluar varias comparaciones, combinando los posibles valores de estas.

Operador	Nombre	Ejemplo	Devuelve verdadero cuando:
&&	Y	(7>2) && (2<4)	Devuelve verdadero cuando ambas condiciones son verdaderas.
and	Y	(7>2) and (2<4)	Devuelve verdadero cuando ambas condiciones son verdaderas.
||	O	(7>2) || (2<4)	Devuelve verdadero cuando al menos una de las dos es verdadera.
or	O	(7>2) or (2<4)	Devuelve verdadero cuando al menos una de las dos es verdadera.
!	No	! (7>2)	Niega el valor de la expresión.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
   $a = 8;
   $b = 3;
   $c = 3;
   echo ($a == $b) && ($c > $b),"<br>";
   echo ($a == $b) || ($b == $c),"<br>";
   echo !($b <= $c),"<br>";
?>
</body>
</html>
```

```
1
```

2.6.- Condicionales

Las sentencias condicionales nos permiten ejecutar o no unas ciertas instrucciones dependiendo del resultado de evaluar una condición. Las más frecuentes son la instrucción `if` y la instrucción `switch`.

Sentencia `if ... else`

```
<?php
   if (condición)
   {
      Sentencias a ejecutar cuando la
      condición es cierta.
   }
   else
   {
      Sentencias a ejecutar cuando la
      condición es falsa.
   }
?>
```

La sentencia `if` ejecuta una serie de instrucciones u otras dependiendo de la condición que le pongamos. Probablemente sea la instrucción más importante en cualquier lenguaje de programación.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
   $a = 8;
   $b = 3;
   if ($a < $b)
   {
      echo "a es menor que b";
   }
   else
   {
      echo "a no es menor que b";
   }
?>
</body>
</html>
```

```
a no es menor que b
```

En este ejemplo la condición no es verdadera por lo que se ejecuta la parte de código correspondiente al `else`.

Sentencia `switch ... case`

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
   $posicion = "arriba";

   switch($posicion) {
      case "arriba":   // Bloque 1
         echo "La variable contiene";
         echo " el valor arriba";
         break;
      case "abajo":   // Bloque 2
         echo "La variable contiene";
         echo " el valor abajo";
         break;
      default:   // Bloque 3
         echo "La variable contiene otro valor";
         echo " distinto de arriba y abajo";
   }
?>
</body>
</html>
```

```
La variable contiene el valor arriba
```

Con la sentencia `switch` podemos ejecutar unas u otras instrucciones dependiendo del valor de una variable, en el ejemplo anterior, dependiendo del valor de la variable `$posicion` se ejecuta el bloque 1 cuando el valor es "arriba", el bloque 2 cuando el valor es "abajo" y el bloque 3 si no es ninguno de los valores anteriores.

Este tipo de condicionales se ocupa muy frecuentemente para distinguir entre navegadores y luego ejecutar código especial para cada uno, i.e.: código para Netscape Communicator y código para Microsoft Internet Explorer.

2.7.- Bucles

Los bucles nos permiten iterar conjuntos de instrucciones, es decir repetir la ejecución de un conjunto de instrucciones mientras se cumpla una condición.

Sentencia `while`

```
<?php
   while (condición)
   {
      intrucciones a ejecutar.
   }
?>
```

Mientras la condición sea cierta se reiterará la ejecución de las instrucciones que están dentro del `while`.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
Inicio<BR>
<?php
   $i=0;
   while ($i<10)
   {
      echo "El valor de i es ", $i,"<br>";
      $i++;
   }
?>
Final<BR>
</body>
</html>
```

```
Inicio
El valor de i es 0
El valor de i es 1
El valor de i es 2
El valor de i es 3
El valor de i es 4
El valor de i es 5
El valor de i es 6
El valor de i es 7
El valor de i es 8
El valor de i es 9
Final
```

En el siguiente ejemplo, el valor de `$i` al comienzo es `0`, durante la ejecución del bucle, se va sumando `1` al valor de `$i` de manera que cuando `$i` vale `10` ya no se cumple la condición y se termina la ejecución del bucle.

Sentencia for

```
<?php
   for (inicial ; condición ; ejecutar en iteración)
   {
      intrucciones a ejecutar.
   }
?>
```

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
Inicio<BR>
<?php
   for($i=0 ; $i<10 ; $i++)
   {
      echo "El valor de i es ", $i,"<br>";
   }
?>
Final<BR>
</body>
</html>
```

```
Inicio
El valor de i es 0
El valor de i es 1
El valor de i es 2
El valor de i es 3
El valor de i es 4
El valor de i es 5
El valor de i es 6
El valor de i es 7
El valor de i es 8
El valor de i es 9
Final
```

La instrucción for es la instrucción de bucles más completa. En una sola instrucción nos permite controlar todo el funcionamiento del bucle.

El primer parámetro del for, es ejecutado la primera vez y sirve para inicializar la variable del bucle, el segundo parámetro indica la condición que se debe cumplir para que el bucle siga ejecutándose y el tercer parámetro es una instrucción que se ejecuta al final de cada iteración y sirve para modificar el valor de la variable de iteración.

2.8.- Salida

Hasta ahora hemos usado la instrucción `echo` para realizar salida a pantalla, esta instrucción es bastante limitada ya que no nos permite formatear la salida. En esta página veremos la instrucción `printf` que nos da mucha más potencia.

```
<?php
  printf(cadena formato, variable1, variable2...);
?>
```

La cadena de formateo indica cómo se han de representar las valores que posteriormente le indicaremos. La principal ventaja es que además de poder formatear los valores de salida, nos permite intercalar texto entre ellos.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
   printf("El numero dos con diferentes formatos: %d %f %.2f",2,2,2);
?>
</body>
</html>
```

```
El numero dos con diferentes formatos: 2 2.000000 2.00
```

La cadena de formato puede incluir una seria de caracteres especiales que indican como formatear las variables que se incluyen en la instrucción.

Elemento	Tipo de variable
%s	Cadena de carácteres.
%d	Número sin decimales.
%f	Número con decimales.
%c	Carácter ASCII.
Aunque existen otros tipos, estos son los más importantes.	

Las siguientes sentencias son todas válidas:

```
echo "Hola mundo"; echo
("Hola Mundo"); print "Hola
mundo"; print ("Hola
mundo"); print
(3.1415926);
```

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
   $var="texto";
   $num=3;
   printf("Puede fácilmente intercalar <b>%s</b> con números <b>%d</b>
<br>",$var,$num);

   printf("<TABLE BORDER=1 CELLPADDING=20>");
   for ($i=0;$i<5;$i++)
   {
      printf("<tr><td>%10.d</td></tr>",$i);
   }
   printf("</table>");
?>
</body>
</html>
```

Puede fácilmente intercalar **texto** con números **3**

0
1
2
3
4

2.9.- Manejo de cadenas (strings)

Dado el uso del lenguaje PHP el tratamiento de cadenas es muy importante, existen bastantes funciones para el manejo de cadenas, a continuación explicaremos las más usadas.

1. `strlen(cadena)`. Nos devuelve el número de carácteres de una cadena.
2. `split(separador,cadena)`. Divide una cadena en varias usando un carácter separador.
3. `sprintf(cadena de formato, var1, var2...)`. Formatea una cadena de texto al igual que printf pero el resultado es devuelto como una cadena.
4. `substr(cadena, inicio, longitud)`. Devuelve una subcadena de otra, empezando por `inicio` y de longitud `longitud`.
5. `chop(cadena)`. Elimina los saltos de línea y los espacios finales de una cadena.
6. `strpos(cadena1, cadena2)`. Busca la cadena2 dentro de cadena1 indicándonos la posición en la que se encuentra.
7. `str_replace(cadena1, cadena2, texto)`. Reemplaza la `cadena1` por la `cadena2` en el texto.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
   echo strlen("12345"),"<br>";

   $palabras=split(" ","Esto es una prueba");
   for($i=0;$palabras[$i];$i++)
      echo $palabras[$i],"<br>";

   $resultado=sprintf("8x5 = %d <br>",8*5);
   echo $resultado,"<br>";

   echo substr("Devuelve una subcadena de otra",9,3),"<br><br>";

   if (chop("Cadena \n\n ") == "Cadena")
      echo "Iguales<br><br>";

   echo strpos("Busca la palabra dentro de la frase", "palabra"),"<br><br>";

   echo str_replace("verde","rojo","Un pez de color verde, como verde es la
hierba."),"<br>";

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

```
5
Esto
es
una
prueba
8x5 = 40

una

Iguales

9

Un pez de color rojo, como rojo es la hierba.
```

2.10.- Los arreglos (arrays)

El PHP ofrece la posibilidad de agrupar un conjunto de valores para almacenarlos juntos y referenciarlos por un índice.

Probar la salida del siguiente código:

```
<?
print "Mi_array es $mi_array<BR>";
print "Mi_array[5] es $mi_array[5]<BR>";
$mi_array[5] = "Posición 6ta";
print "Mi_array[5] es $mi_array[5]<BR>";
print "Mi_array es $mi_array<BR>";
?>
```

que produce la siguiente salida:

```
Mi array es
Mi_array[5] es
Mi_array[5] es Posición 6ta
Mi_array es Array
```

Concatenar el mismostring:

```
$cad = 'A esta cadena';
$cad = $cad . 'le vamos a añadir más texto.';
```

Usar partes del mismo string:

```
$cad2 = "Tercer carácter de \$cad : '$cad[2]'";
```

2.11.- Strings como índices

Los índices puedes ser del tipo numérico (entero) o una cadena de forma indistinta.

```
$comida["Mallorca"] = "Sopas";
$comida["Valencia"] = "Paella";
$comida["Madrid"] = "Cocido";
```

2.12.- Constantes

Las constantes se definen con la función define():

```
define("SALUDO", "Hola, mundo!");

echo "La constante SALUDO vale " . SALUDO;
```

Las constantes en PHP se diferencian de las variables en que:

- no llevan el símbolo del dólar delante.
- puede accederse a ellas desde cualquier parte del código donde han sido definidas, sin restricciones de ámbito como en las variables.
- no pueden ser redefinidas o borradas una vez definidas.
- sólo pueden contener valores escalares, no vectores.

2.13.- Verificación de Tipos.

gettype(arg)
Retorna un string representando el tipo de argumento: integer, double, string, array, object o unknown type.

is_int(arg), is_integer(arg), is_long(arg)
Retorna verdadero si arg es de tipo entero, falso en caso contrario.

is_double(arg), is_float(arg), is_real(arg)
Retorna verdadero si arg es un double, falso en caso contrario.

is_bool(arg)
Retorna verdadero si arg es del tipo Boolean (TRUE o FALSE) y falso si no lo es.

is_string(arg)
Retorna verdadero si arg es un string.

is_array(arg)
Retorna verdadero si arg es un array.

is_object(arg)
Retorna verdadero si arg es un objeto.

2.14.- Funciones

El uso de funciones nos da la capacidad de agrupar varias instrucciones bajo un solo nombre y poder llamarlas a estas varias veces desde diferentes sitios, ahorrándonos la necesidad de escribirlas de nuevo.

```
<?php
   function Nombre(parametro1, parametro2...)
   {
      instrucción1;
      instrucción2;
      instrucción3;
      instrucción4;

      return valor_de_retorno;
   }
?>
```

Opcionalmente podemos pasarle parámetros a las funciones que se trataran como variable locales y así mismo podemos devolver un resultado con la instrucción return valor; Esto produce la terminación de la función retornando un valor.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php

   function media_aritmetica($a, $b)
   {
      $media=($a+$b)/2;
      return $media;
   }

   echo media_aritmetica(4,6),"<br>";
   echo media_aritmetica(3242,524543),"<br>";

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

```
5
263892.5
```

Capítulo 3.- Usos útiles

3.1.- Librerías

El uso de librerías es tremendamente útil, nos permiten agrupar varias funciones y variables en un mismo fichero, de manera que luego podemos incluir esta librería en distintas páginas y disponer de esas funciones fácilmente.

```
<!-- Manual de PHP -->

<?php
   function CabeceraPagina()
   {
?>
  <FONT SIZE="+1">Esta cabecera estará en todas sus páginas.</FONT><BR>
  <hr>
<?
   }

   function PiePagina()
   {
?>
  <hr>
  <FONT SIZE="-1">Este es el pie de página.</FONT><BR>
  Autor: John Doe
<?
   }
?>
```

Ahora vamos a crear 2 páginas que usan la librería definida anteriormente para conseguir que las dos paginas tengan la misma cabecera y pie de pagina.

La instrucción para incluir una librería en nuestra página es `include("nombre de librería")`

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php include("libreria01.phtml") ?>
<?php CabeceraPagina(); ?>

Página 1
<BR><BR><BR><BR><BR>

Contenido blalbl blalb alb<BR><BR>
más cosas...<BR><BR>

fin<BR><BR>

<?php PiePagina(); ?>
</body>
</html>
```

Esta cabecera estará en todas sus páginas.

Página 1

Contenido blalbl blalb alb

más cosas...

fin

Este es el pie de página.
Autor: John Doe

Ejemplo 2:

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php include("libreria01.phtml") ?>
<?php CabeceraPagina(); ?>

Esta es otra página<BR><BR>
completamente distinta<BR><BR>
pero comparte el pie y la cabecera con la otra.<BR><BR>

<?php PiePagina(); ?>
</body>
</html>
```

3.2.- Páginas con plantillas (usando librerías)

En este ejemplo vamos a usar el PHP y la capacidad de definir librerías para conseguir que todas nuestras páginas tengan el mismo formato de página, incluyendo las partes comunes en librerías. Así mismo modificando la librería modificaríamos tambien todas las páginas de una manera muy rápida.

libpagina.phtml

```
<!-- Manual de PHP -->

<?php
   function CabeceraPagina()
   {
?>  <FONT SIZE="+1">Esta cabecera estará en todas sus
páginas.</FONT><BR>
  <hr>
<?   }

   function PiePagina()
   {
?>    <hr>
  <FONT SIZE="-1">Este es el pie de página.</FONT><BR>
  Autor: John Doe
<?   }

   function Indice()
   {
?>  <A HREF="ejem06a.phtml">Pagina 1</A><BR>
  <A HREF="ejem06a2.phtml">Pagina 2</A><BR>
<?   } ?>
```

ejem06a.phtml

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php include("libpagina.phtml") ?>
<?php CabeceraPagina(); ?>
<TABLE>
<TR>
  <TD><?php Indice() ?></TD>
  <TD>
      Esta es otra página<BR><BR>
      completamente distinta<BR><BR>
      pero comparte el pie y la cabecera con la otra.<BR><BR>
  </TD>
</TR>
</TABLE>
<?php PiePagina(); ?>
</body>
</html>
```

Esta cabecera estará en todas sus páginas.

Esta es otra página

Pagina 1 completamente distinta
Pagina 2

pero comparte el pie y la cabecera con la otra.

Este es el pie de página.
Autor: John Doe

ejem06a2.phtml

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php include("libpagina.phtml") ?>
<?php CabeceraPagina(); ?>
<TABLE>
<TR>
  <TD><?php Indice() ?></TD>
  <TD>
Página 1
<BR><BR><BR><BR><BR>

Contenido blalbl blalb alb<BR><BR>
más cosas...<BR><BR>

fin<BR><BR>
  </TD>
</TR>
</TABLE>
<?php PiePagina(); ?>
</body>
</html>
```

Esta cabecera estará en todas sus páginas.

Página 1

Pagina 1
Pagina 2

Contenido blalbl blalb alb

más cosas...

fin

Este es el pie de página.
Autor: John Doe

3.3.- Enlace externo con frame

Con este ejemplo damos solución al problema de los enlaces externos y de forma que la web externa queda en la parte inferior del frame y así no se sale de nuestra web.

ejem06b.html

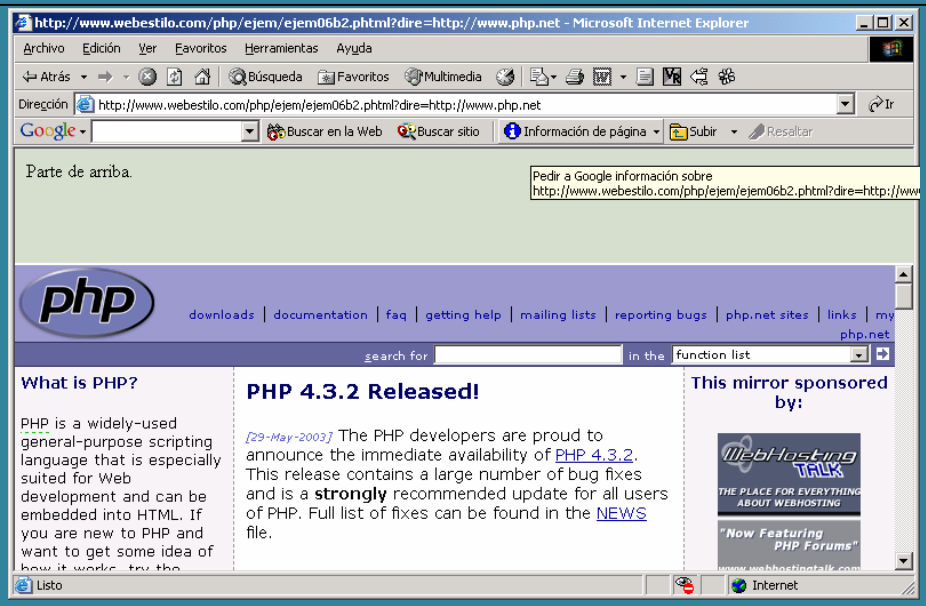
```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>

<A HREF="ejem06b2.phtml?dire=http://www.php.net">www.php.net</A><BR><BR>
<A HREF="ejem06b2.phtml?dire=http://www.abc.com">www.abc.com</A>

</body>
</html>
```

www.php.net

www.abc.com

ejem06b2.phtml

```
<!-- Manual de PHP -->
<!-- frames -->
<FRAMESET ROWS="100,*">
<FRAME NAME="arriba" SRC="ejem06b3.html" MARGINWIDTH="10" MARGINHEIGHT="10"
SCROLLING="auto" FRAMEBORDER="0">
<FRAME NAME="abajo" SRC="<?php echo $dire ?>" MARGINWIDTH="10"
MARGINHEIGHT="10" SCROLLING="auto" FRAMEBORDER="0">
</FRAMESET>
```



Capítulo 4.- La directiva register_globals en PHP 4.2.0

A partir de la versión de PHP 4.2.0 el valor por defecto de la directiva register_globals es off. El por qué de este cambio viene motivado por un aumento del nivel de seguridad en la configuración del PHP por defecto. Pero esto puede provocar que nuestras páginas dejen de funcionar.

La directiva register_globals cuando esta activada (estado que estaba por defecto antes de la versión 4.2.0 de PHP), provoca que automáticamente se generen variables globales para cookies y valores enviados por get y post entre otros.

Por ejemplo:

Si llamábamos a una página con http://www.internet.com/prueba.php?var1=4, esto provocaba que en la página prueba.php automáticamente se generase la variable `$var1` con el valor 4.

Por razones de seguridad este comportamiento automático se ha cambiado, estableciendo el valor por defecto de register_globals a off.

Este cambio puede producir que nuestras antiguas páginas dejen de funcionar, ante esto tenemos dos opciones:

- Activar el `register_globals` a `on`.
- Dejar `register_globals` a `off` y cambiar nuestras páginas por las referencias adecuadas a cada caso. Esta es la opción más recomendable.

¿Cómo debemos hacer este cambio?

Debemos buscar todas aquellas variables que son definidas automáticamente, variables del servidor, que provienen de get o post, cookies, files, variables de entorno o sesión.

Reemplazar esas variables por las referencias adecuadas en cada caso, en PHP se han definido unos arrays diferentes con valores, dependiendo del lugar de procedencia. Así tenemos los arrays `$_SERVER`, `$_GET`, `$_POST`, `$_COOKIE`, `$_FILES`, `$_ENV`, `$REQUEST` y `$_SESSION`.

Si por ejemplo teníamos el siguiente script:

```
print "Su edad: ".$edad; //cookie
print "Navegador: ".$HTTP_USER_AGENT;
print "Variable: ".$var; //variable de get
```

Deberíamos reemplazarlas por:

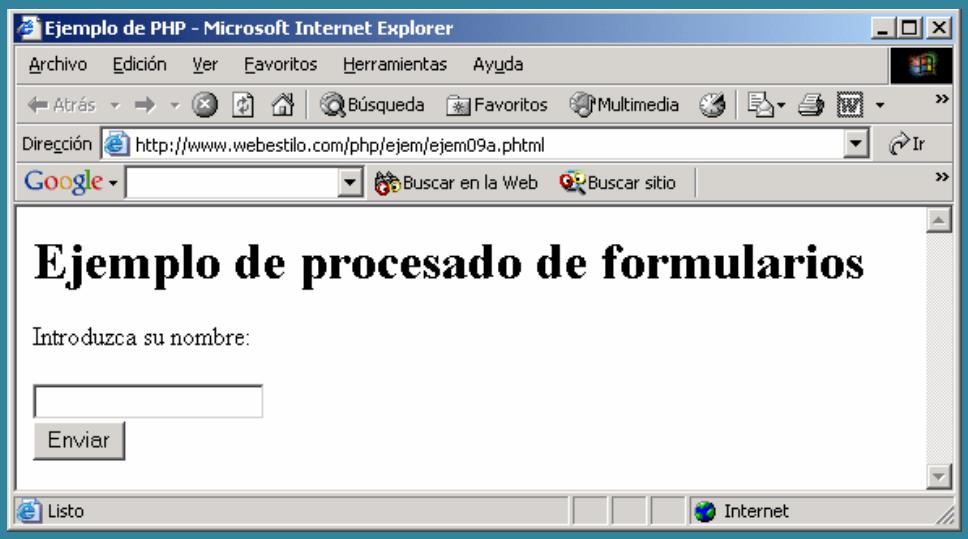
```
print "Su edad: ".$_COOKIE['edad']; //cookie
print "Navegador: ".$_SERVER['HTTP_USER_AGENT'];
print "Variable: ".$_GET['var']; //variable de get
```

Capítulo 5.- Envío y recepción de datos usando Formularios

El lenguaje PHP nos proporciona una manera sencilla de manejar formularios, permitiéndonos de esta manera procesar la información que el usuario ha introducido.

Al diseñar un formulario debemos indicar la página PHP que procesará el formulario, así como en método por el que se le pasará la información a la página.

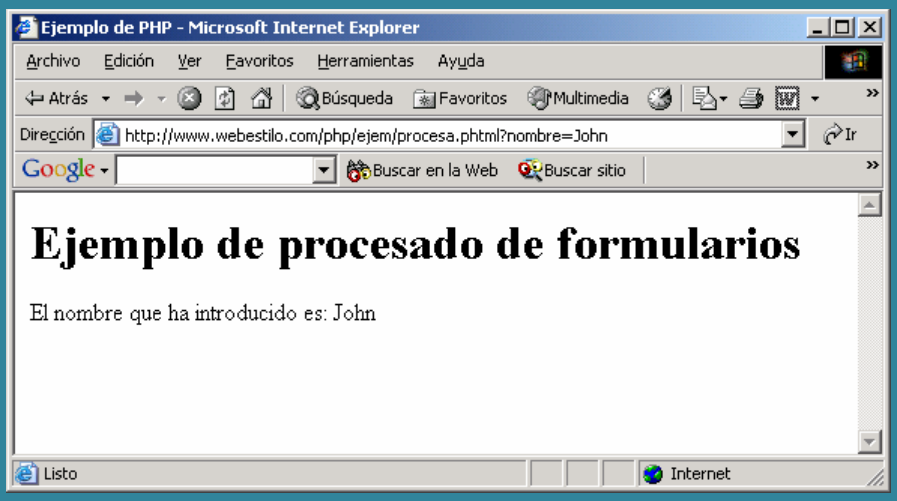
```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>
Introduzca su nombre:
<FORM ACTION="procesa.phtml" METHOD="GET">
<INPUT TYPE="text" NAME="nombre"><BR>
<INPUT TYPE="submit" VALUE="Enviar">
</FORM>
</body>
</html>
```



Al pulsar el botón Enviar el contenido de cuadro de texto es enviado a la página que indicamos en el atributo `ACTION` de la etiqueta `FORM`.

PHP crea una variable por cada elemento del `FORM`, esta variable creada tiene el mismo nombre que el cuadro de texto de la página anterior y el valor que hayamos introducido. En este ejemplo se ha creado una variable llamada `$nombre` con el valor que haya introducido el navegante.

```
<!-- Manual de PHP de WebEstilo.com -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>
El nombre que ha introducido es: <?php echo $nombre ?>
<br>
</FORM>
</body>
</html>
```

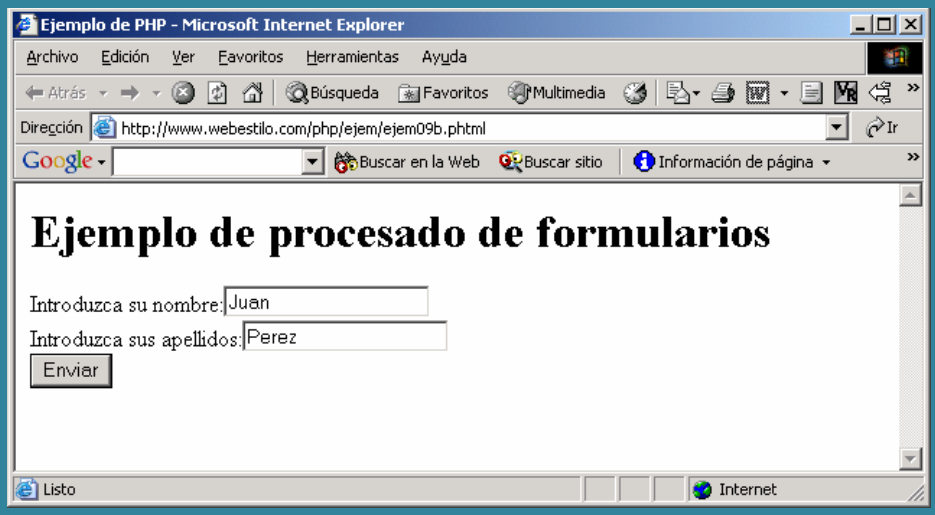


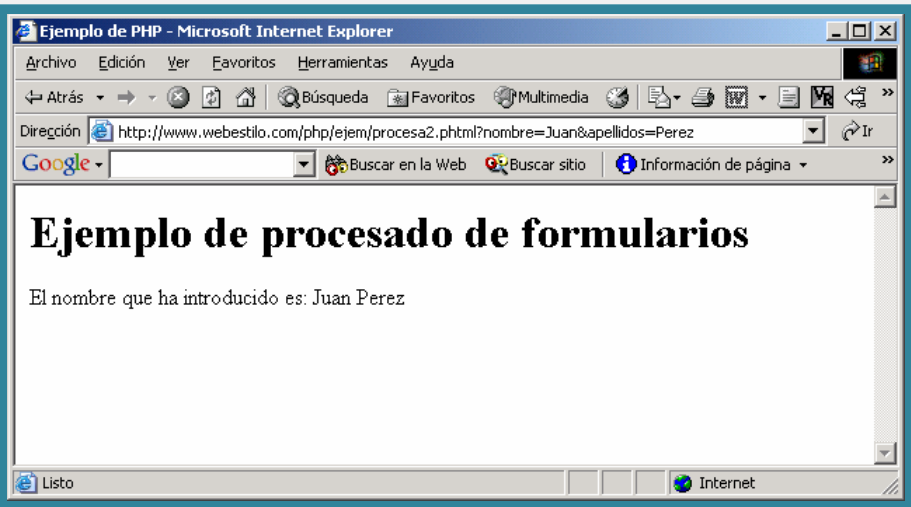
5.2.- Método GET y POST

En la página anterior hemos comentado que los datos de un formulario se envía mediante el método indicado en el atributo METHOD de la etiqueta FORM, los dos métodos posibles son GET y POST.

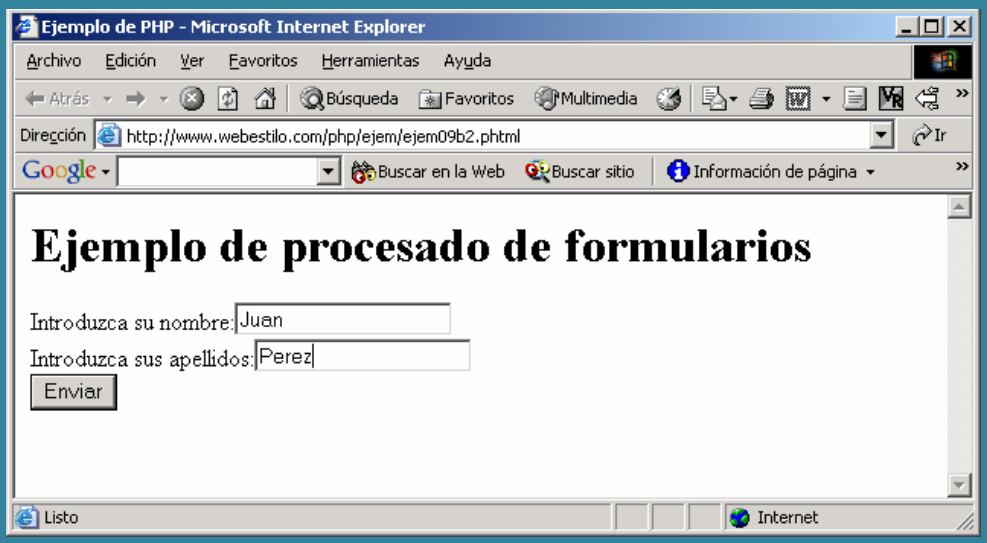
La diferencia entre estos dos métodos radica en la forma de enviar los datos a la página, mientras que el método GET envía los datos usando la URL, el método POST los envía por la entrada estándar STDIO.

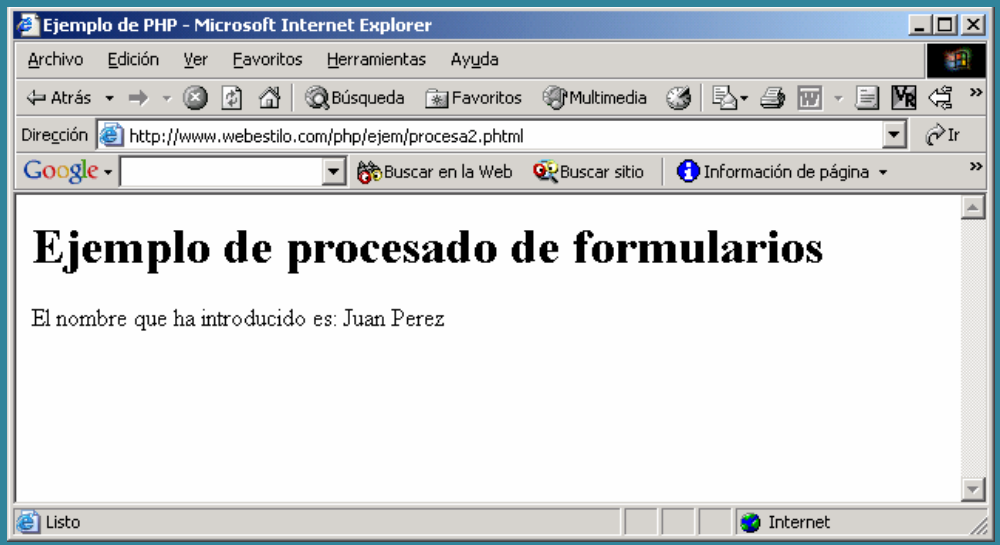
```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>

<FORM ACTION="procesa2.phtml" METHOD="GET">
Introduzca su nombre:<INPUT TYPE="text" NAME="nombre"><BR>
Introduzca sus apellidos:<INPUT TYPE="text" NAME="apellidos"><BR>
<INPUT TYPE="submit" VALUE="Enviar">
</FORM>
</body>
</html>
```




Ejemplo de PHP - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Búsqueda Favoritos Multimedia
Dirección http://www.webestilo.com/php/ejem/procesa2.phtml?nombre=Juan&apellidos=Perez
Ir
Google Buscar en la Web Buscar sitio Información de página
Ejemplo de procesado de formularios
El nombre que ha introducido es: Juan Perez
Listo
Internet

```
<!-- Manual de PHP de WebEstilo.com -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>
<FORM ACTION="procesa2.phtml" METHOD="POST">
Introduzca su nombre:<INPUT TYPE="text" NAME="nombre"><BR>
Introduzca sus apellidos:<INPUT TYPE="text" NAME="apellidos"><BR>
<INPUT TYPE="submit" VALUE="Enviar">
</FORM>
</body>
</html>
```





procesa2.phtml

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>
El nombre que ha introducido es: <?php echo $nombre," ",$apellidos ?>
<br>
</body>
</html>
```

El resultado final es el mismo, solo que con el método GET podemos ver los parámetros pasados ya que están codificados en laURL.

5.3.- Envío de emails

PHP nos ofrece la posibilidad de enviar emails de una manera sencilla y fácil, para ello el lenguaje nos proporciona la instrucción `mail( )`

```
<?php
   mail(destinatario, tema, texto del mensaje);
?>
```

En el parámetro `destinatario` pondremos la dirección de email a donde se enviará el mensaje, en el parámetro `tema` el tema o subject del mensaje y el parámetro `texto del mensaje` el cuerpo del mensaje en formato texto plano.

Existe una sintaxis extendida de la instrucción `mail( )` que nos permite añadir información adicional a la cabecera del mensaje.

```
<?php
   mail(destinatario, tema, texto del mensaje, información adicional de cabecera);
?>
```

En la información de cabecera podremos incluir parámetros adicionales al mensaje como `Reply-To:`, `From:`, `Content-type:`... que nos permiten tener un mayor control sobre el mensaje.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de envio de email</H1>
Introduzca su direccion de email:
<FORM ACTION="email.phtml" METHOD="GET">
<INPUT TYPE="text" NAME="direccion"><BR><BR>
Formato: <BR>
<INPUT TYPE="radio" NAME="tipo" VALUE="plano" CHECKED> Texto plano<BR>
<INPUT TYPE="radio" NAME="tipo" VALUE="html"> HTML<BR><BR>
<INPUT TYPE="submit" VALUE="Enviar">
</FORM>
</body>
</html>
```

email.phtml

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de envio de email</H1>
<? if ($direccion!=""){
   if ($tipo=="plano"){
      // Envio en formato texto plano

      mail($direccion,"Ejemplo de envio de email","Ejemplo de envio de email
de texto plano\n\nPHP.\nhttp://www.php.net/\n Manuales para desarrolladores
web.\n","FROM: Pruebas <webmaster@hotmail.com>\n");
   } else {
      // Envio en formato HTML
      mail($direccion,"Ejemplo de envio de email","<html><head><title>PHP.
Manual de PHP</title></head><body>Ejemplo de envio de email de
HTML<br><br>PHP.<br>http://www.php.net/<br> <u>Manuales</u> para
<b>desarrolladores</b> web.</body></html>","Content-type: text/html\n",
"FROM: Pruebas <webmaster@hotmail.com>\n");
   }
echo "Se ha enviado un email a la direccion: ",$direccion," en formato
<b>",$tipo,"</b>.";
}
?>
<br>
</FORM>
</body>
</html>
```

Capítulo 6.- PHP y base de datos

Para la realización de este curso sobre PHP con acceso a base de datos hemos elegido la base de datos MySQL por ser gratuita y por ser también la mas empleada en entornos UNIX, para lo cual el servidor donde tenemos alojadas las páginas nos tiene que proporcionar herramientas para crearla o acceso al Telnet para que la creemos por nosotros mismos.

El comando para crear una base de datos MySQL es el siguiente:

```
mysqladmin -u root create base_datos
```

Con este comando conseguimos crear la una base de datos en el servidor de bases de datos de nuestro servidor.

Una vez conseguido esto debemos crear las tablas en la base de datos, la descripción de las tablas contienen la estructura de la información que almacenaremos en ellas. Para lo cual usaremos en lenguaje de consultas SQL común para todas las bases de datos relacionales.

En este ejemplo creamos una tabla llamada prueba con 3 campos: un campo identificador, que nos servirá para identificar unívocamente una fila con el valor de dicho campo, otro campo con el nombre de una persona y por último un campo con el apellido de la persona.

Para crear la tabla puede usar la herramienta de administración de MySQL de su servidor web o puede escribir un fichero de texto con el contenido de la sentencia SQL equivalente y luego decirle al motor de base de datos que la ejecute con la siguiente instrucción:

```
mysql -u root base_datos <prueba.sql
```

prueba.sql

```
CREATE TABLE prueba (
ID_Prueba int(11) DEFAULT '0' NOT NULL auto_increment,
Nombre varchar(100),
Apellidos varchar(100),
PRIMARY KEY (ID_Prueba),
UNIQUE ID_Prueba (ID_Prueba)
);
```

6.1.- Conexión a la base de datos

Una vez que tenemos creada la base de datos en nuestro servidor, el siguiente paso es conectarnos a la misma desde una página PHP. Para ello PHP nos proporciona una serie de instrucciones para acceder a bases de datos MySQL.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
function Conectarse()
{
   if (!($link=mysql_connect("localhost","usuario","Password")))
   {
      echo "Error conectando a la base de datos.";
      exit();
   }
   if (!mysql_select_db("base_datos",$link))
   {
      echo "Error seleccionando la base de datos.";
      exit();
   }
   return $link;
}

$link=Conectarse();
echo "Conexión con la base de datos conseguida.<br>";

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

Al ejecutar la instrucción `mysql_connect` creamos un vínculo entre la base de datos y la pagina PHP, este vínculo será usado posteriormente en las consultas que hagamos a la base de datos.

Finalmente, una vez que hemos terminado de usar el vínculo con la base de datos, lo liberaremos con la instrucción `mysql_close` para que la conexión no quede ocupada.

Es necesario consultar con su administrador web para ver las variables por omisión (by defualt) que se tienen fijadas en el archivo “php.ini” en relación a las bases de datos, como por ejemplo la habilitación o prohibición de usar conexiones persistentes (mysql_pconnect).

6.2.- Consultas a la base de datos

Una vez que nos hemos conectado con el servidor de bases de datos, ya podemos realizar consultas a las tablas de la base de datos.

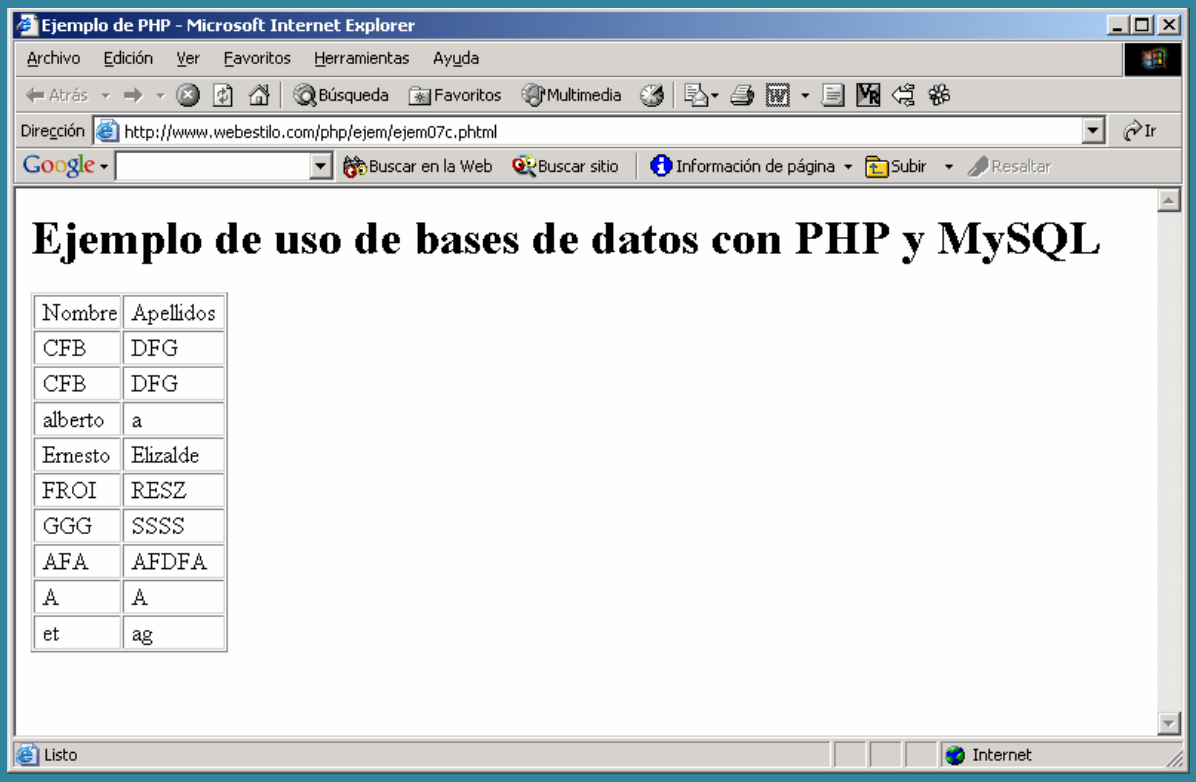
Para facilitar la programación hemos separado la función de conexión en una librería a parte, de tal manera que la incluiremos en todas las páginas que accedan a la base de datos.

conex.phtml

```
<!-- Manual de PHP -->
<?php
function Conectarse()
{
   if (!($link=mysql_connect("localhost","usuario","Password")))
   {
      echo "Error conectando a la base de datos.";
      exit();
   }
   if (!mysql_select_db("base_datos",$link))
   {
      echo "Error seleccionando la base de datos.";
      exit();
   }
   return $link;
}
?>
```

ejem07c.phtml

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de uso de bases de datos con PHP y MySQL</H1>
<?php
   include("conex.phtml");
   $link=Conectarse();
   $result=mysql_query("select * from prueba",$link);
?>
  <TABLE BORDER=1 CELLSPACING=1 CELLPADDING=1>
    <TR><TD> Nombre</TD><TD> Apellidos </TD></TR>
<?php

   while($row = mysql_fetch_array($result)) {
      printf("<tr><td> %s</td><td> %s </td></tr>",
$row["Nombre"],$row["Apellidos"]);
   }
   mysql_free_result($result);
   mysql_close($link);
?>
</table>
</body>
</html>
```



En este ejemplo hemos utilizado 3 instrucciones nuevas: `mysql_query`, `mysql_fetch_array` y `mysql_free_result`. Con la instrucción `mysql_query` hemos hecho una consulta a la base de datos en el lenguaje de consultas SQL, con la instrucción `mysql_fetch_array` extraemos los datos de la consulta a un array y con `mysql_free_result` liberamos la memoria usada en la consulta.

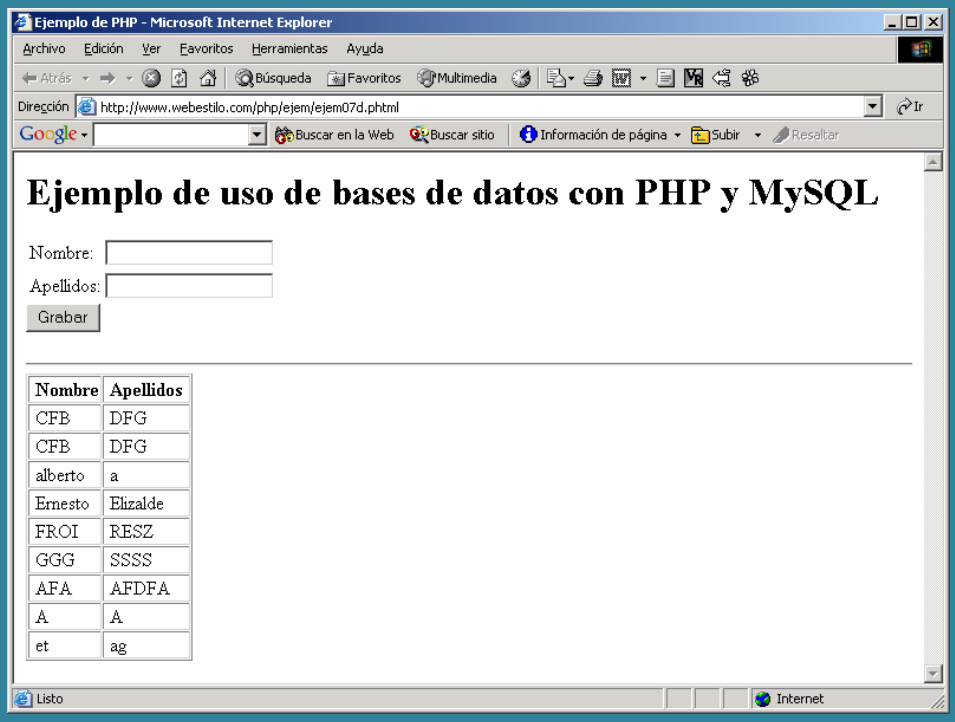
6.3.- Inserción de registros

Hasta ahora nos hemos conectado a una base de datos y hemos hecho consultas a la misma, ahora presentaremos como introducir nuevo registros en la base de datos.

Para ello usaremos un formulario y en el ACTION del FORM <FORM ACTION="programaPHP"> indicaremos que debe ser procesado una pagina PHP, esta página lo que hará será introducir los datos del formulario en la base de datos.

ejem07d.phtml

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de uso de bases de datos con PHP y MySQL</H1>
<FORM ACTION="procesar.phtml">
<TABLE>
<TR>
  <TD>Nombre:</TD>
  <TD><INPUT TYPE="text" NAME="nombre" SIZE="20" MAXLENGTH="30"></TD>
</TR>
<TR>
  <TD>Apellidos:</TD>
  <TD><INPUT TYPE="text" NAME="apellidos" SIZE="20" MAXLENGTH="30"></TD>
</TR>
</TABLE>
<INPUT TYPE="submit" NAME="accion" VALUE="Grabar">
</FORM>
<hr>
<?php
   include("conex.phtml");
   $link=Conectarse();
   $result=mysql_query("select * from prueba",$link);
?>
  <TABLE BORDER=1 CELLSPACING=1 CELLPADDING=1>
    <TR><TD> <B>Nombre</B></TD>
<TD> <B>Apellidos</B> </TD></TR>
<?php

   while($row = mysql_fetch_array($result)) {
      printf("<tr><td> %s</td> <td> %s </td></tr>",
$row["Nombre"], $row["Apellidos"]);
   }
   mysql_free_result($result);
   mysql_close($link);
?>
</table>
</body>
</html>
```



procesar.phtml

```
<?php
   include("conex.phtml");
   $link=Conectarse();
   mysql_query("insert into prueba (Nombre,Apellidos) values
('$nombre','$apellidos')",$link);

   header("Location: ejem07d.phtml");
?>
```

La primera página PHP `ejem07d.phtml` es un formulario que nos permite introducir nombre y apellido para añadirlo a la base de datos, seguido de una consulta que nos muestra el contenido de la tabla prueba. El formulario llama a la pagina `procesar.phtml` que añadirá los datos a la tabla.

La segunda página `procesar.phtml` se conecta a la base de datos y añade un nuevo registro con la instrucción `insert` del lenguaje de base de datos SQL. Una vez el registro se ha añadido se vuelve a cargar la página `ejem07d.phtml`

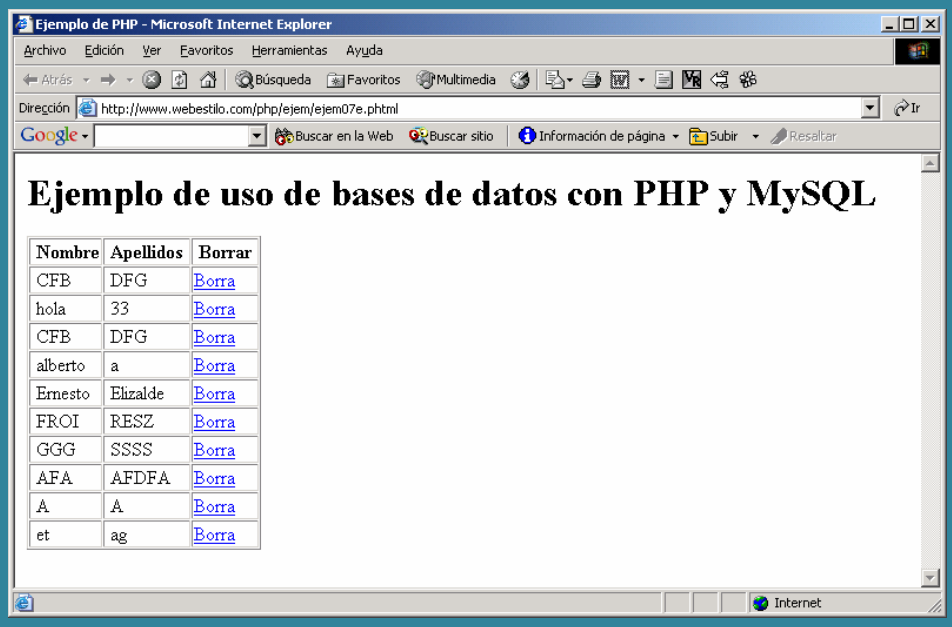
6.4.- Borrado de registros

Y finalmente, para cerrar el ciclo, nos queda el borrado de registros. El borrado de registros el uno de los procesos mássencillos.

Para indicar que elemento vamos a borrar hemos usado un enlace a la página `borra.phtml` pasándole el `ID_Prueba` de cada registro, de esta manera la página `borra.phtml` sabe que elemento de la tabla ha de borrar.

ejem07e.phtml

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de uso de bases de datos con PHP y MySQL</H1>

<?php
   include("conex.phtml");
   $link=Conectarse();
   $result=mysql_query("select * from prueba",$link);
?>
  <TABLE BORDER=1 CELLSPACING=1 CELLPADDING=1>
    <TR><TD> <B>Nombre</B></TD>
<TD> <B>Apellidos</B> </TD>
<TD> <B>Borrar</B> </TD></TR>
<?php

   while($row = mysql_fetch_array($result)) {
      printf("<tr><td> %s</td><td> %s </td><td><a
href=\"borra.phtml?id=%d\">Borra</a></td></tr>",
$row["Nombre"],$row["Apellidos"],$row["ID_Prueba"]);
   }
   mysql_free_result($result);
   mysql_close($link);
?>
</table>
</body>
</html>
```



borra.phtml

```
<?php
   include("conex.phtml");
   $link=Conectarse();
   mysql_query("delete from prueba where ID_Prueba = $id",$link);

   header("Location: ejem07e.phtml");
?>
```

La página `borra.phtml` se conecta a la base de datos y borra el registro indicado en la variable `$id` que ha sido pasado desde la página `ejem07e.phtml`. Una vez el registro se ha borrado se vuelve a cargar la página `ejem07e.phtml`

Recordar cambiar `$id` por `$_GET['id']` si el register_globals está en on.

Capítulo 7.- Restringir el acceso

En esta sección se explica cómo se puede restringir el acceso a las páginas, para que solo las personas autorizadas puedan acceder a ciertas partes del sitio web.

Atención: El acceso restringido a páginas usando las variables globales $PHP_AUTH_USER, $PHP_AUTH_PW y $PHP_AUTH_TYPE solo funciona si PHP ha sido instalado como un módulo de Apache, si ha sido instalado como un CGI los ejemplos de ésta sección no funcionarán.

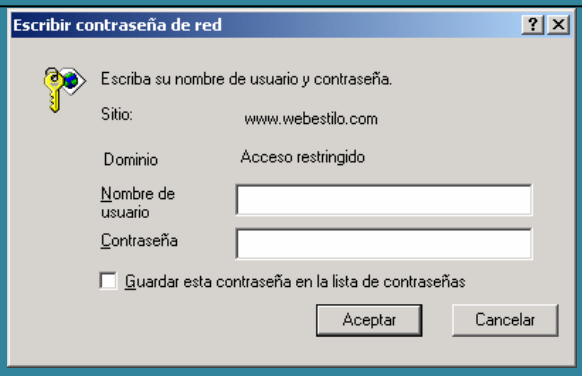
Atención: Si tiene activado "register globals = on" en su php.ini (que es ON por omisión en PHP 4.3.x en adelante), entonces deberá cambiar la variable $PHP_AUTH_USER por $_SERVER[´PHP_AUTH_USER´].

Para conseguir la autentificación en las páginas usaremos el sistema de autentificación del protocolo HTTP, este sistema se basa en las variables globales $_SERVER[´PHP_AUTH_USER´]y $_SERVER[´PHP_AUTH_PW´]..

1. $PHP_AUTH_USER. Nombre de usuario introducido.
2. $PHP_AUTH_PW. Contraseña introducida.

Para que el navegador nos muestre la ventana de petición de nombre de usuario y contraseña basta con enviar la siguiente cabecera (HEADER):

```
<?php // Manual de PHP
   if (!isset($_SERVER[´PHP_AUTH_USER´])) {
      header('WWW-Authenticate: Basic realm="Acceso restringido"');    //usado la primera vez
      header('HTTP/1.0 401 Unauthorized');
      echo 'Authorization Required.';              // en caso que usuario aprete CANCELAR
      exit;
   }
   else {
      echo "Ha introducido el nombre de usuario: $_SERVER[´PHP_AUTH_USER´] <br>";
      echo "Ha introducido la contraseña: $_SERVER[´PHP_AUTH_PW´]<br>";
   }
?>
```

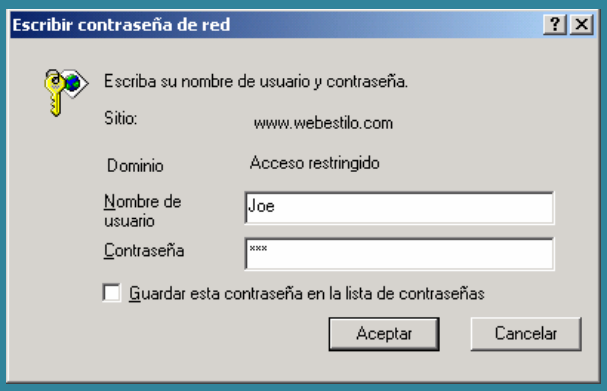


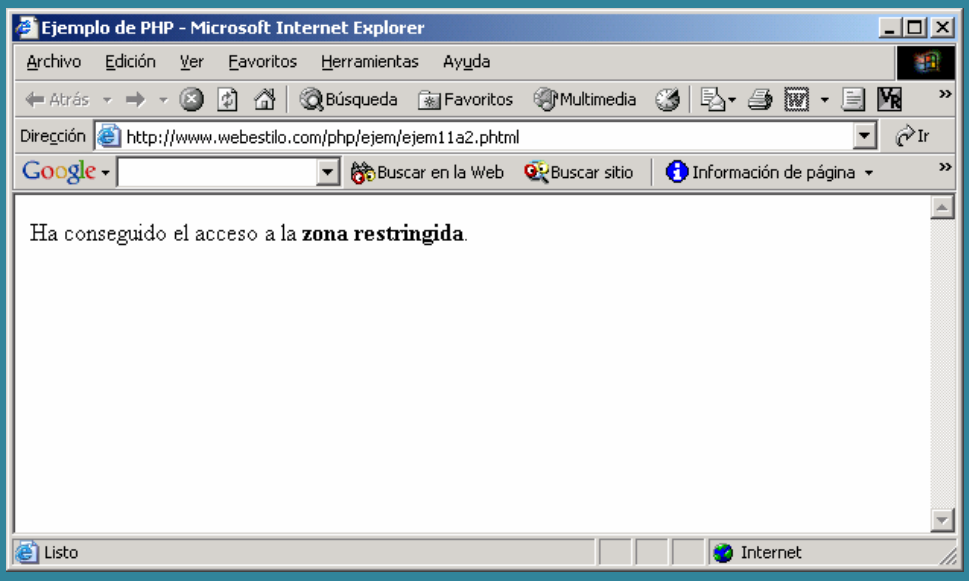
Esto provoca que se muestre la ventana de nombre de usuario y contraseña y los datos introducidos se asignen a las variables $_SERVER[´PHP_AUTH_USER´] y $_SERVER[´PHP_AUTH_PW´].

A partir de aquí realizaremos las comprobaciones necesarias para asegurarnos que los datos introducidos son los correctos (usuarios que deberían tener acceso a la información).

En el siguiente ejemplo pediremos autorización y comprobaremos si el nombre de usuario es Joe y la contraseña 123, si es así tendremos acceso al resto de la página.

```php
<?php // Manual de PHP
   if (($_SERVER[´PHP_AUTH_USER´]!="Joe") || ($_SERVER[´PHP_AUTH_PW´] !="123")) {
      header('WWW-Authenticate: Basic realm="Acceso restringido"');
      header('HTTP/1.0 401 Unauthorized');
      echo 'Authorization Required.';
      exit;
   }
?>
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
Ha conseguido el acceso a la <B>zona restringida</B>.
</body>
</html>
```





7.1.- Validación de usuarios usando un archivo de texto

En la anterior página todo el mundo que tenía acceso a la parte restringida entraba con el mismo nombre de usuario y contraseña, esto evidentemente no es una buena solución, es mejor que cada persona tenga un nombre de usuario y contraseña, ya que de esta forma podemos inhabilitar a un usuario sin ver comprometida la seguridad de nuestro sitio.

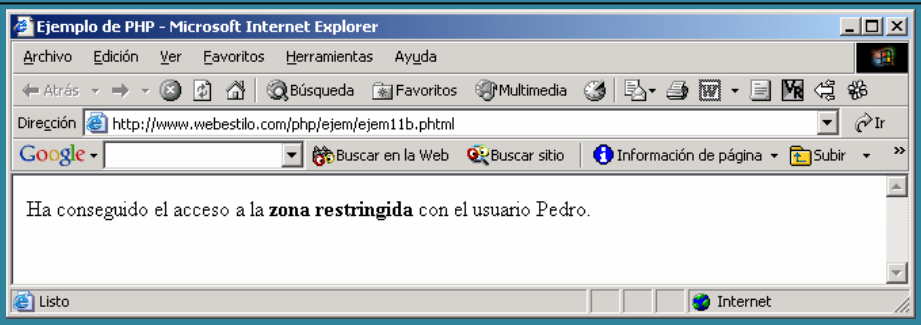
En esta página veremos la forma de realizar esto, teniendo un fichero separado con los nombres de usuario y las contraseñas válidas. Dicho fichero podría tener el siguiente formato: `nombre_de_usuario:contraseña`. Por ejemplo:

passwords.txt:

```
Joe:1235
Pedro:qwer
Noe:Gty45e
kermit:rwe4v
```

En este ejemplo se pide la autorización al comienzo de la página si no se ha establecido con anterioridad y se comprueba con el fichero de contraseñas que hemos llamado passwords.txt, si el nombre de usuario y contraseña coincide con alguna entrada del fichero se nos permite ver el resto de la página.

```
<?php // Manual de PHP
    if (!isset($_SERVER[´PHP_AUTH_USER´])) {               //código para pedir usuario/clave
      header('WWW-Authenticate: Basic realm="Acceso restringido"');
      header('HTTP/1.0 401 Unauthorized');
      echo 'Authorization Required.';
      exit;
   }

   $fich = file("passwords.txt");     //una vez obtenido usuario/clave se procede a verificar
   $i=0; $validado=false;
   while ($fich[$i] && !$validado) {
      $campo = explode(":",$fich[$i]);
      if (($_SERVER[´PHP_AUTH_USER´]==$campo[0]) && $_SERVER[´PHP_AUTH_PW´] ==chop($campo[1])))
            $validado=true;
      $i++;
   }

   if (!$validado) {          //si no coincidió con ningún usuario del archivo, entonces ....
      header('WWW-Authenticate: Basic realm="Acceso restringido"');
      header('HTTP/1.0 401 Unauthorized');
      echo 'Authorization Required.';
      exit;
   }
?>
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
Ha conseguido el acceso a la <B>zona restringida</B> con el usuario
<?php echo $_SERVER[´PHP_AUTH_USER´] ?>.
</body>
```



Otra forma de hacer lo mismo.....

```php
<?php

$auth = false; // Asume que el usuario no está autentificado aún...

if (isset( $PHP_AUTH_USER ) && isset($PHP_AUTH_PW)) {

    // Leer el archivo completo en la variable $file_contents

    $filename = '/path/to/file.txt';
    $fp = fopen( $filename, 'r' );
    $file_contents = fread( $fp, filesize( $filename ) );
    fclose( $fp );

    // Colocar cada línea del archivo en un arreglo.

    $lines = explode ( "\n", $file_contents );

    // Partir cada linea en usuario y clave
    // e intenta hacer coincidir los valores con $PHP_AUTH_USER and $PHP_AUTH_PW.

    foreach ( $lines as $line ) {

        list( $username, $password ) = explode( ':', $line );

        if ( ( $username == "$_SERVER[´PHP_AUTH_USER´]" ) &&
             ( $password == "$_SERVER[´PHP_AUTH_PW´]" ) ) {

            // Si se encuentra una coincidencia, entonces el usuario es autenticado
            // Detener la búsqueda.

            $auth = true;
            break;

        }
    }
}

if ( ! $auth ) {

    header( 'WWW-Authenticate: Basic realm="Privado"' );
    header( 'HTTP/1.0 401 Unauthorized' );
    echo 'Authorization Required.';
    exit;

} else {

    echo '<P>Ud. esta autenticado!</P>';
}

?>
```

7.2.- Validación de usuarios usando htaccess

Un archivo ".htaccess" es usado por el servidor Web (i.e. Apache) para autenticar usuarios. Se almacena el nombre del usuario y la clave de éste en el archivo ".htpasswd" en forma encriptada con el estándar DES usando la función "crypt()":

```
joe:WvzodahMR9USk jane:g3RYjX5evEvdM
julie:YzASzTGEo2VMA
```

```php
<?php

$auth = false; // Asume que el usuario no está autentificado aún...

if (isset( $PHP_AUTH_USER ) && isset($PHP_AUTH_PW)) {

    // Leer el archivo completo en la variable $file_contents

    $filename = '/path/to/.htpasswd';
    $fp = fopen( $filename, 'r' );
    $file_contents = fread( $fp, filesize( $filename ) );
    fclose( $fp );

    // Colocar cada línea del archivo en un arreglo.

    $lines = explode ( "\n", $file_contents );

    // Partir cada linea en usuario y clave
    // e intenta hacer coincidir los valores con $PHP_AUTH_USER and $PHP_AUTH_PW.

    foreach ( $lines as $line ) {

        list( $username, $password ) = explode( ':', $line );

        if ( $username == "$PHP_AUTH_USER" ) {

            // Obtener la semillade encriptación de $password.
            // Son siempre los 2 primeros
            // caracteres del string encriptado con DES.
            $salt = substr( $password , 0 , 2 );

            // encriptar $PHP_AUTH_PW basado en $salt
            $enc_pw = crypt( $PHP_AUTH_PW, $salt );

            if ( $password == "$enc_pw" ) {
            // Si se encuentra una coincidencia, entonces el usuario es autenticado
            // Detener la búsqueda.
                $auth = true;
                break;
            }
        }
    }
}

if ( ! $auth ) {

    header( 'WWW-Authenticate: Basic realm="Private"' );
    header( 'HTTP/1.0 401 Unauthorized' );
    echo 'Authorization Required.';
    exit;
} else {
    echo '<P> Ud. esta autenticado!</P>';
}

?>
```

7.3.- Validación de usuarios usando MySQL

Suponga que su tabla de usuarios se llama "users" y es como sigue:

real_name	username	password
Joe Smith	joe	ai890d
Jane Smith	jane	29hj0jk
Mary Smith	mary	fsSS92
Bob Smith	bob	2NNg8ed
Dilbert	dilbert	a76zFs

Una coincidencia para el nombre de usuario y su clave correspondiente se puede obtener con la siguiente sentencia SQL:

```
SELECT *
FROM users
WHERE username='$_SERVER[´PHP_AUTH_USER´]' and password='$_SERVER[´PHP_AUTH_PW´]'
```

```php
<?php
$auth = false; // Asume que el usuario no está autentificado aún...

if (isset( $PHP_AUTH_USER ) && isset($PHP_AUTH_PW)) {
    // Connexion a MySQL
    mysql_connect( 'hostname', 'username', 'password' )
        or die ( 'Unable to connect to server.' );

    // Seleccionar la BD en el servidor MySQL
    mysql_select_db( 'your_db' )
        or die ( 'Unable to select database.' );

    // Generar la consulta
    $sql = "SELECT * FROM users WHERE username = '$_SERVER[´PHP_AUTH_USER´]' AND
            password = '$_SERVER[´PHP_AUTH_PW´]'";

    // Ejecutar la consulta y colocar los resultados en $result
    $result = mysql_query( $sql )
        or die ( 'Unable to execute query.' );

    // Obtener el número de filas de $result.
    $num = mysql_numrows( $result );

    if ( $num != 0 ) {
        // Si se encuentra una coincidencia, entonces el usuario es autenticado.
        $auth = true;
    }
}
if ( ! $auth ) {
    header( 'WWW-Authenticate: Basic realm="Private"' );
    header( 'HTTP/1.0 401 Unauthorized' );
    echo 'Authorization Required.';
    exit;
} else {
    echo '<P> Ud. esta autenticado!</P>';
}
?>
```

Capítulo 8.- ¿Qué son las sesiones?

El uso de sesiones es un método ampliamente extendido en cualquier aplicación de cierta entidad. Básicamente una sesión es la secuencia de páginas que un usuario visita en un sitio web. Desde que entra en nuestro sitio, hasta que lo abandona.

El término sesión en PHP, session en inglés, se aplica a esta secuencia de navegación, para ello crearemos un identificador único que asignamos a cada una de estas sesiones de navegación. A este identificador de sesión se le denomina, comúnmente, como la sesión.

El proceso en cualquier lenguaje de programación podría ser algo así: Existe

una sesión?
Si existe la retomamos
Si no existe creamos una nueva
Generar un identificador único

Y para que no perdamos el hilo de la navegación del usuario deberemos asociar esta sesión a todas las URLs y acciones de formulario. Podemos también crear un cookie que incluya el identificador de sesión, pero es conveniente recordar que la disponibilidad o no de las cookies depende del usuario, y no es conveniente fiarse de lo que un usuario pueda o no tener habilitado.

Lo contado hasta ahora es teoría pura y es aplicable a cualquier lenguaje de programación C, Perl, etc. Los que programamos en PHP4 tenemos la suerte de que toda la gestión de sesiones la hace el mismo PHP.

Por lo tanto lo comentado a partir de aquí es **solo aplicable a PHP4**. Si aún desarrollas PHP3, tendrás que crear tus propias librerías de gestión de sesiones o recurrir a alguna de las existentes, como la de PHPLIB.

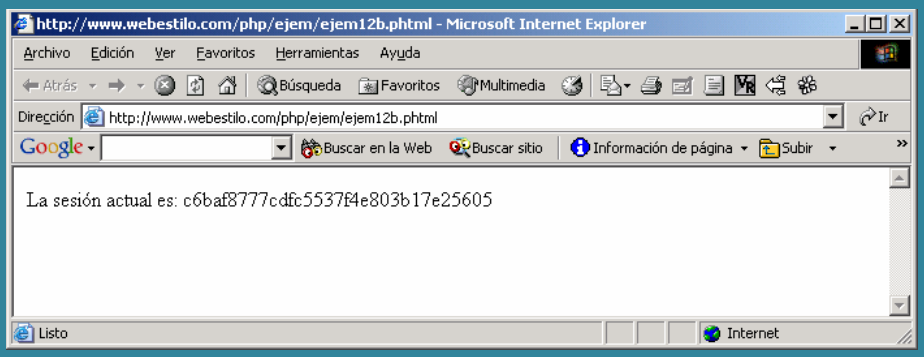
8.1.- Inicialización de la sesión

Para utilizar sesiones en PHP lo primero es inicializarlas. Podemos hacerlo explícitamente, mediante la función `session_start()`, o al registrar una variable en una sesión mediante `session_register('miVariable')`. En ambos casos se crea una nueva sesión, si no existe, o se retoma la sesión actual. Veamos un sencillo ejemplo:

```
<?php // Manual de PHP

session_start();
echo "He inicializado la sesión";
?>
```

Esta es la forma más básica, si el usuario tiene los cookies activados, PHP habrá insertado de forma automática la sesión y ésta será pasada de una página a otra sin hacer nada más. Desde un punto de vista práctico la sesión es operativa, pero no vemos nada. Podemos obtener la sesión en cualquier momento mediante la función `session_id()`. Inserta en las sucesivas páginas la siguiente línea para ver si la sesión está disponible:

```
<?php // Manual de PHP

session_start();
echo 'La sesión actual es: '.session_id();
?>
```

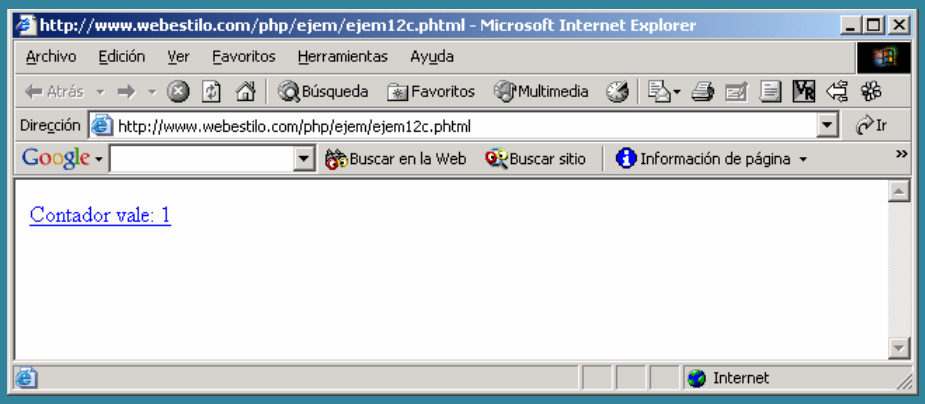


En este caso `session_start()` comprueba en los cookies que existe una sesión y continua con ella, `session_id()` devuelve el identificador actual.

8.2.- Ejemplo práctico

Veamos otro ejemplo que, tal vez, te lo aclare un poco más:

```
<?php // Manual de PHP

session_register('contador');
echo '<a href="'.$PHP_SELF.'?'.$SID.'">Contador vale: '.++$contador.'</a>';
?>
```



Como se dijo anteriormente la sesión se crea o recoge mediante `session_start()`, o también cuando se registra una variable de sesión mediante `session_register()`.

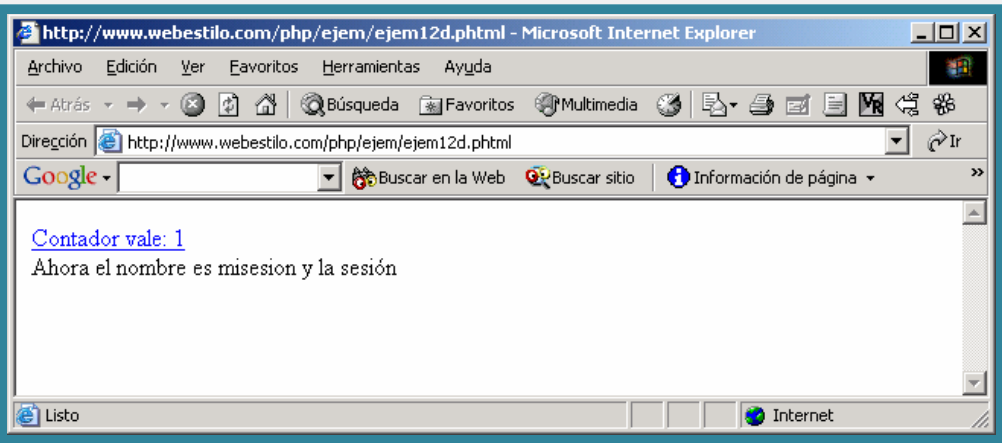
Si no has utilizado nunca las sesiones, el concepto de variable de sesión, puede resultar un poco abstracto. Básicamente es una variable, como cualquiera de las que gestiona PHP4, pero que reside en un espacio específico en el servidor, junto con el identificador de sesión, y que pertenece únicamente a un usuario.

En nuestro ejemplo anterior, registramos la variable `$contador` en la primera línea del script. En la segunda línea, entre otras cosas, cada vez que recarguemos la página o hagamos click sobre el enlace, el valor de `$contador` se incrementará en 1.

En esta línea hacemos uso de la variable reservada `$PHP_SELF`, que hace referencia al propio script en ejecución y una constante propia de PHP4, `$SID`, que contiene el nombre de la sesión y el identificador de la misma.

Podemos averiguar también el nombre de la sesión, o modificarlo, mediante la función `session_name()`. Veamos una pruebapráctica:

```
<?php // Manual de PHP
session_name('misesion');
session_register('contador');
echo '<a href="'.$PHP_SELF.'?'.SID.'">Contador vale: '.++$contador.'</a><br>';
echo 'Ahora el nombre es '.session_name().' y la sesión '.$misesion.'<br>';
?>
```



La asignación del nombre de sesión debe realizarse antes que ninguna otra función con sesiones, antes que `session_start()` o `session_register()`.

Error común

Uno de los errores más comunes cuando se utilizan sesiones es dejar líneas en blanco antes de la inicialización de PHP o enviar alguna salida a la pantalla. Para probarlo crea una línea en blanco o con cualquier cosa antes de `<?php`.

Si tienes los cookies activados, te encontrarás un error de este tipo:

```
Warning: Cannot send session cookie - headers already sent by (output started at
/home/session.php:2) in /home/session.php on line 4
```

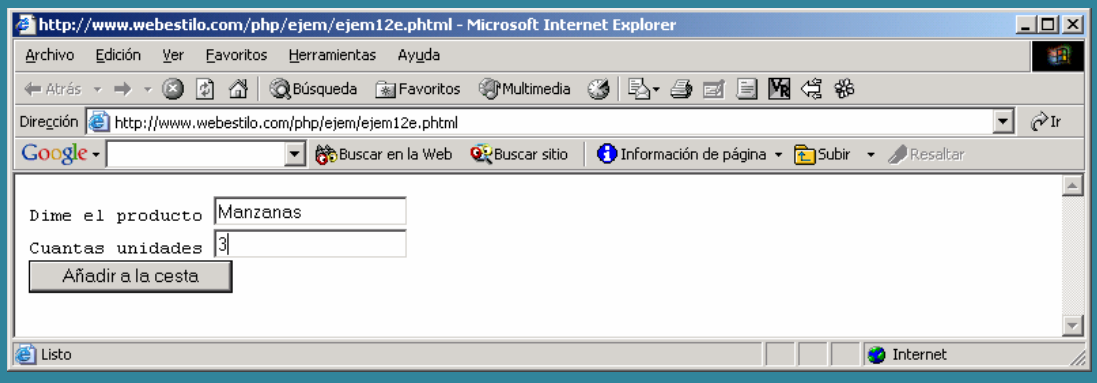
PHP está informando de que no puede activar los cookies en el navegador del usuario, porque las cabeceras ya han sido enviadas. Simplemente por la existencia de una línea en blanco. Como medida práctica, no dejes espacios ni antes del inicio del script, ni después de la finalización.

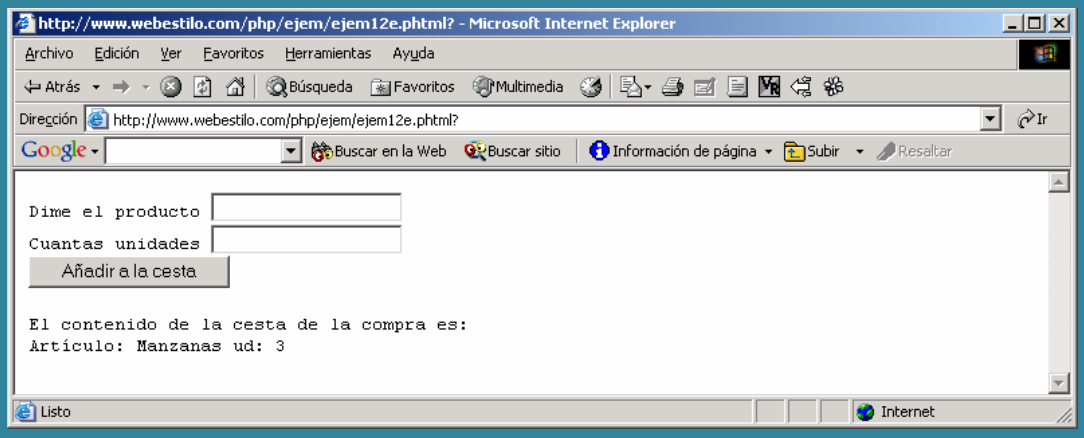
Si después de todo lo comentado aún no entiendes para que sirven las sesiones, veamos un ejemplo práctico. Imagina que quisieras crear un sistema de cesta de la compra...

8.3.- Carrito de compra

Si después de todo lo comentado aún no entiendes para que sirven las sesiones, veamos un ejemplo práctico. Imagina que quisieras crear un sistema de cesta de la compra, en su forma básica podría ser algo así:

```
<?php // Manual de PHP
session_start();
session_register('itemsEnCesta');
if ($item){
   if (!isset($itemsEnCesta)){
      $itemsEnCesta[$item]=$cantidad;
   }else{
      foreach($itemsEnCesta as $k => $v){
         if ($item==$k){
         $itemsEnCesta[$k]+=$cantidad;
         $encontrado=1;
         }
      }
      if (!$encontrado) $itemsEnCesta[$item]=$cantidad;
   }
}
?>
<html>
<body>
<tt>
<form action="<?=$PHP_SELF."?".SID?>" method="post">
Dime el producto <input type="text" name="item" size="20"><br>
Cuantas unidades <input type="text" name="cantidad" size="20"><br>
<input type="submit" value="Añadir a la cesta"><br>
</form>
<?
if (isset($itemsEnCesta)){
   echo'El contenido de la cesta de la compra es:<br>';
   foreach($itemsEnCesta as $k => $v){
      echo 'Artículo: '.$k.' ud: '.$v.'<br>';
   }
}
?>
</tt>
</body>
</html>
```





Una breve explicación. En la línea 4 comprobamos si el usuario ha pasado algún artículo, desde el formulario. En la 5 si el array `itemsEnCesta` no existe, lo creamos con el nuevo producto y la cantidad indicada. Si el array existe recorremos su contenido, entre las líneas 8 y 13, y si encontramos un artículo igual, añadimos la cantidad en la línea 10. Si no lo encontramos, es un nuevo artículo, por lo tanto, añadimos el nuevo producto con la correspondiente cantidad a `itemsEnCesta` en la línea 14.

Y a continuación imprimimos el formulario y los resultados, si los hubiera, a partir de la línea 18, donde empieza el HTML.

¿Te imaginas las posibilidades de un sistema de almacenamiento de información de estas características?. No necesitas ficheros, ni bases de datos, ni tienes que andar pasando valores de una página a otra. PHP va gestionando estos datos por nosotros, hasta el momento en que decidamos almacenar la información donde más nos interese.

Estas son las funcionalidades básicas de las sesiones, espero que te halla resultado ilustrativo y no olvides consultar el resto de funciones asociadas al uso de sesiones en el manual de PHP.

Capítulo 9.- ¿Qué son las cookies?

La principal utilidad de las cookies (galletas) es la de solventar el problema de la falta de estado en la navegación a través de las paginas web.

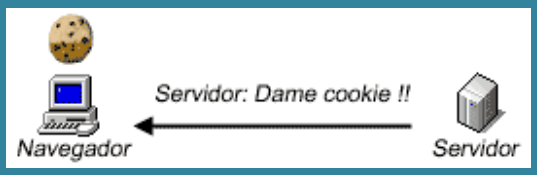
Con las cookies, pequeñas porciones de información se quedan registradas en el navegador permitiendo identificar a este a través de diferentes páginas de un mismo sitio e incluso durante visitas entre distintos días.

Realmente las cookies no son mas que cadenas de texto que son enviadas desde el servidor al cliente (navegador) y almacenadas en este, luego el navegador envía estas cookies al servidor permitiendo así la identificación del cliente en el servidor.

Funcionamiento

La cookie es enviada al navegador desde el servidor y si este la acepta permanece en él.

Las páginas piden la cookie al navegador…



El navegador las envía, permitiendo la identificación del usuario por parte del servidor.

El manejo de cookies en PHP se realiza mediante el uso de la función `setcookie`, esta función esta disponible a partir de la versión 3 de PHP.

```
int setcookie (string Nombre [, string Valor [, int Expire [, string Path
[, string Dominio [, int Secure]]]]])
```

`Setcookie()` define una cookie que es enviada junto con el resto de la información de la cabecera(header). Las cookies deben ser enviadas antes de cualquier tag de html, por lo tanto deberemos realizar la llamada a estas funciones antes de cualquier tag `<HTML>` o `<HEAD>`. Esta es una restricción de las cookies no de PHP.

Todos los argumentos excepto el nombre son opcionales.

- Nombre. Nombre de la cookie. Si creamos una cookie solamente con el nombre, en el cliente se eliminara la cookie que exista con ese nombre. También podemos reemplazar cualquier argumento con una cadena vacía ("").
- Value. Valor que almacenará la cookie en el cliente.
- Expire. El argumento expire es un argumento entero que indica la hora en que se eliminara la cookie en el formato de hora que devuelven las funciones UNIX time() y mktime(). Normalmente se usa `time() + N. segundos de duración`, para especificar la duración de una cookie.
- Path. Subdirectorio en donde tiene valor la cookie.
- Dominio. Dominio en donde tiene valor la cookie. Si ponemos como dominio `www.domain.com` la cookie no se transmite para `domain.com`, mientras que si ponemos `domain.com` la cookie se transmite tanto para `domain.com` como para `www.domain.com`
- Secure. El argumento secure indica que la cookie solo se transmitirá a través de una conexión segura HTTPS.

Ejemplo

```
setcookie("usuario", "Luis", time()+3600,"/","php.net");
```

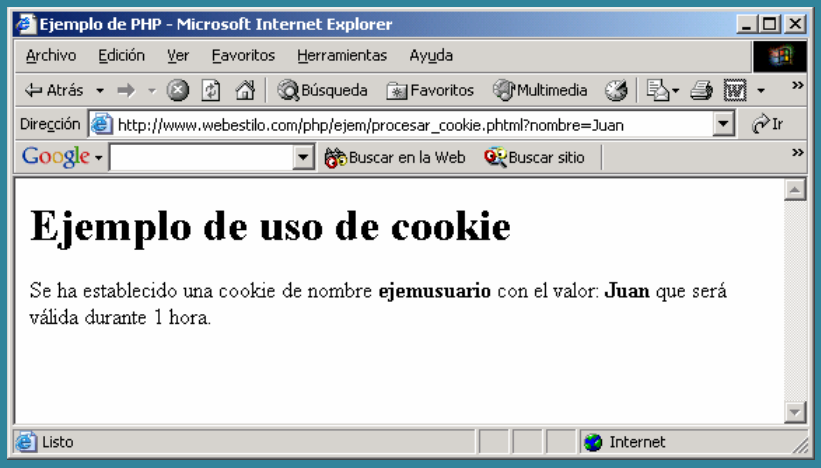
En este ejemplo establecemos una cookie de nombre `usuario` que contiene el valor `Luis`, que dura `1 hora` (3600 segundos) válida para todo el dominio `php.net`

9.1.- Ejemplo de uso de cookies

En este ejemplo vamos a ver como establecer una cookie y cómo se recupera el valor establecido. Para ello pediremos al usuario que introduzca su nombre, que guardaremos en una cookie.

Primero pedimos al usuario que introduzca el valor de su nombre, usamos un formulario que procesará la página `procesar_cookie.phtml`.

```
<!-- Manual de PHP -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de uso de cookie</H1>
Introduzca su nombre:
<FORM ACTION="procesar_cookie.phtml" METHOD="GET">
<INPUT TYPE="text" NAME="nombre"><BR>
<INPUT TYPE="submit" VALUE="Enviar">
</FORM>
</body>
</html>
```

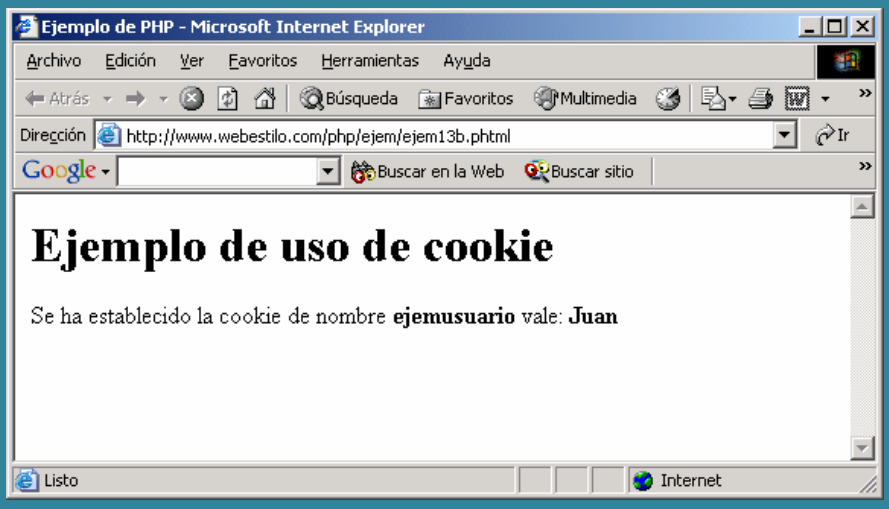


Se establece la cookie `ejemusuario` con el valor introducido anteriormente, y cuya duración es una hora.

```
<?php // Manual de PHP
   setcookie("ejemusuario", $nombre, time()+3600,"/","");
?>
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de uso de cookie</H1>

Se ha establecido una cookie de nombre <b>ejemusuario</b> con el valor: <b><? print
$nombre; ?></b> que será válida durante 1 hora.
</body>
</html>
```

En este ejemplo vemos lo fácil que es recuperar el valor de la cookie establecida anteriormente.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de uso de cookie</H1>

Se ha establecido la cookie de nombre <b>ejemusuario</b> vale: <b><? print $ejemusuario; ?></b>
</body>
</html>
```



Capítulo 10.- PHP y LDAP

PHP tiene varias funciones que permiten realizar consultas a un servidor LDAP. Para ver si su PHP posee soporte para LDAP, debe ejecutar la función "phpinfo()" y observar si existe un recuadro LDAP.

Para efectos de ejemplo se usará la siguiente base de LDAP:

```
dn: dc=my-domain, dc=com
objectClass: dcObject
objectClass: organization
o: MyOrganization
dc: my-domain.com

dn: mail=root@my-domain.com, dc=my-domain,
dc=com
objectClass: inetOrgPerson
cn: Keith
sn: Richards
mail: root@my-domain.com

dn: mail=joe@my-domain.com, dc=my-domain, dc=com
objectClass: inetOrgPerson
cn: Joe
sn: Somebody
mail: joe@my-domain.com

dn: mail=sarah@my-domain.com, dc=my-domain,
dc=com
objectClass: inetOrgPerson
cn: Sarah
sn: Nobody
mail: sarah@my-domain.com
telephoneNumber: 23 67 128 5639
```

A continuación se muestra un código PHP que se conecta al servidor LDAP y muestra el directorio completo.

leer_ldap.php

```
<html>
<head>
</head>

<body>

<?php

// specify the LDAP server to connect to
$conn = ldap_connect("localhost") or die("Could not connect to server");

// bind to the LDAP server specified above
$r = ldap_bind($conn) or die("Could not bind to server");

// start searching
// specify both the start location and the search criteria
// in this case, start at the top and return all entries
$result =ldap_search($conn,"dc=my-domain,dc=com", "(cn=*)") or die ("Error in search
query");

// get entry data as array
$info = ldap_get_entries($conn, $result);

// iterate over array and print data for each entry
for ($i=0; $i<$info["count"]; $i++)
{
    echo "dn is: ". $info[$i]["dn"] ."<br>";
    echo "first cn is: ". $info[$i]["cn"][0] ."<br>";
    echo "first email address is: ". $info[$i]["mail"][0] ."<p>"; }

// print number of entries found
echo "Number of entries found: " . ldap_count_entries($conn, $result) . "<p>";

// all done? clean up
ldap_close($conn);

?>

</body>
</html>
```

RESULTADO:

dn is: mail=root@my-domain.com, dc=my-domain, dc=com
first cn is: Keith
first email address is: root@my-domain.com

dn is: mail=joe@my-domain.com, dc=my-domain, dc=com
first cn is: Joe
first email address is: joe@my-domain.com

dn is: mail=sarah@my-domain.com, dc=my-domain, dc=com
first cn is: Sarah
first email address is: sarah@my-domain.com

Number of entries found: 3

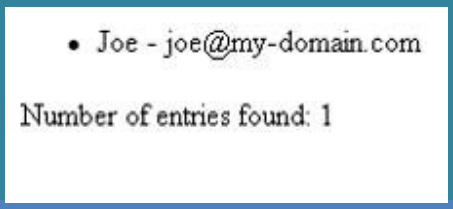
buscar_ldap.html

```
<html>
<head>
<title>Search</title>
</head>
<body>
<form action="search.php" method="POST">
<input type="text" name="name" length="30">
<input type="submit" name="submit" value = "Search">
</form>
</body>
</html>
```

search.php

```
<html>
<head>
</head>

<body>

<?php

// specify the LDAP server to connect to
$conn = ldap_connect("localhost") or die("Could not connect to server");

// bind to the LDAP server specified above
$r = ldap_bind($conn) or die("Could not bind to server");

// create the search string
$query = "(cn=" . $_POST['name'] . ")";

// start searching
// specify both the start location and the search criteria
// in this case, start at the top and return all entries
$result =ldap_search($conn,"dc=my-domain,dc=com", $query) or die ("Error in search query");

// get entry data as array
$info = ldap_get_entries($conn, $result);

// iterate over array and print data for each entry
echo "<ul>";
for ($i=0; $i<$info["count"]; $i++)
{
      echo "<li>" . $info[$i]["cn"][0] ." - ".$info[$i]["mail"][0] . "</li>";
} echo "</ul>";

// print number of entries found
echo "Number of entries found: " . ldap_count_entries($conn, $result) ."<p>";

// all done? clean up
ldap_close($conn);

?>

</body>
</html>
```

Al buscar por "joe"....



buscar2.html

```
<html>
<head>
<title>Search</title>
</head>
<body>
<form action="search2.php" method="POST">
First name
<br>
<input type="text" name="cn" length="30"><br>
Last name
<br>
<input type="text" name="sn" length="30"><br>
Email address
<br>
<input type="text" name="email" length="30"><br>
<input type="submit" name="submit"
value="Search">
</form>
</body>
</html>
```

search2.php

```
<html><head></head><body>
<?php
// specify the LDAP server to connect to
$conn = ldap_connect("localhost") or die("Could not connect to server");

// bind to the LDAP server specified above
$r = ldap_bind($conn) or die("Could not bind to server");

// create the search string
$query = "(&(cn=" . $_POST['cn'] . ")(sn=" . $_POST['sn'] . ")(mail=" .
$_POST['email'] . "))";

// start searching
// specify both the start location and the search criteria
// in this case, start at the top and return all entries
$result =ldap_search($conn,"dc=my-domain,dc=com", $query) or die("Error in
search query");

// get entry data as array
$info = ldap_get_entries($conn, $result);

// iterate over array and print data for each entry
echo "<ul>";
for ($i=0; $i<$info["count"]; $i++)
{
    echo "<li>".$info[$i]["sn"][0]." - ".$info[$i]["mail"][0]." -
".$info[$i]["dn"]."</li>"; } echo "</ul>";

// print number of entries found
echo "Number of entries found: " . ldap_count_entries($conn, $result) .
"<p>";

// all done? clean up
ldap_close($conn);
?>
</body>
</html>
```

Al buscar por "joe"....

- Somebody - joe@my-domain.com - mail=joe@my-domain.com, dc=my-domain, dc=com

Number of entries found: 1

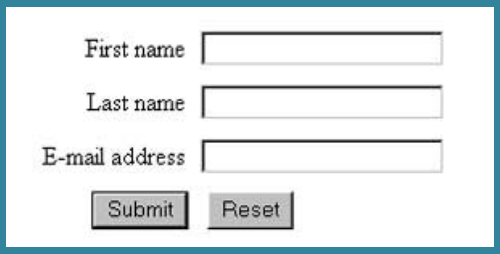
El siguiente código nos sirve para **listar** los usuarios en el directorio LDAP……

```
<html>
<head>
</head>
<body>
<table width="450" cellpadding="5" cellspacing="5" border="1">

<?php
// specify the LDAP server to connect to
$conn = ldap_connect("localhost") or die("Could not connect to server");

// bind to the LDAP server specified above
$r = ldap_bind($conn) or die("Could not bind to server");

// set base DN and required attribute list
$base_dn = "dc=my-domain, dc=com";              // base de la búsqueda
$params = array("mail", "cn", "sn");            //parámetros que se deben retornar

// list all entries from the base DN
$result = ldap_list($conn, $base_dn, "cn=*", $params);
?>
         <tr>
                  <td>First Name</td>
                  <td>Last Name</td>
                  <td colspan=2> </td>
         </tr>
<?php
// get entries
$info = ldap_get_entries($conn, $result);

// and print attribute values
for ($i=0; $i<$info["count"]; $i++)
{
    echo "<tr>";
    echo "<td>".$info[$i]["cn"][0]."</td>";
    echo "<td>".$info[$i]["sn"][0]."</td>";
    echo "<td><a href=\"edit.php?mail=" . urlencode($info[$i]["mail"][0]) . "\">Edit</a></td>";
    echo "<td><a href=\"delete.php?mail=" . urlencode($info[$i]["mail"][0]) . "\">Delete</a></td>";
    echo "</tr>";
}
// all done? clean up
ldap_close($conn);
?>

</table>
<p>
<a href="add.html">Add new entry</a>            // AGREGAR
</body>
</html>
```

First Name	Last Name		
Keith	Richards	Edit	Delete
Joe	Somebody	Edit	Delete
Sarah	Nobody	Edit	Delete
Harish	Kamath	Edit	Delete
Girish	Kamath	Edit	Delete
Tijjay	Majiyagbe	Edit	Delete

Add new entry

El siguiente código nos sirve para **agregar** un usuario al directorio LDAP.....



add.html

```
<html>
<head>
<title>Add Entry</title>
</head>
<body>
<form method="POST" action="add.php">
  <table border="0" cellpadding="0" cellspacing="10" width="500" >
    <tr>
      <td width="50%" align="right">First name</td>
      <td width="50%"><input type="text"  name="cn" size="20"></td>
    </tr>
    <tr>
      <td width="50%" align="right">Last  name</td>
      <td width="50%"><input type="text"  name="sn" size="20"></td>
    </tr>
    <tr>
      <td width="50%" align="right">E-mail address</td>
      <td width="50%"><input type="text" name="mail" size="20"></td>
    </tr>
    <tr>
      <td width="100%" colspan="2" align="center">
        <input type="submit" value="Submit" name="Submit">
        <input type="reset" value="Reset" name="Reset">
      </td>
    </tr>
  </table>

</form>
</body>
</html>
```

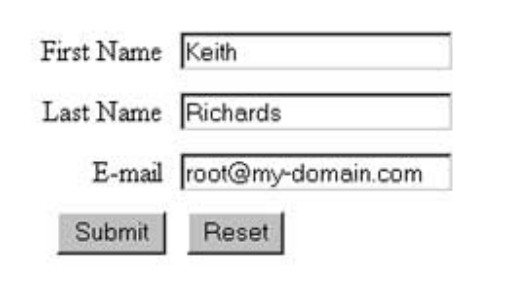
add.php

```
<html>
<head>
</head>
<body>
<?php
// configure privileged user
$root_dn = "cn=root, dc=my-domain, dc=com";
$root_pw = "secret";

// specify the LDAP server to connect to
$conn = ldap_connect("localhost") or die("Could not connect to server.
Error is " . ldap_error($conn));

// bind to the LDAP server
$r = ldap_bind($conn,$root_dn,$root_pw) or die("Could not bind to server.
Error is " . ldap_error($conn));

// prepare data
$info["cn"] =$_POST['cn'];
$info["sn"] =$_POST['sn'];
$info["mail"] = $_POST['mail'];
$info["objectClass"] = "inetOrgPerson";

// prepare DN for new entry
$dn = "mail=" . $_POST['mail'] . ", dc=my-domain, dc=com";

// add data to directory
$result = ldap_add($conn, $dn, $info);

// if successful, display success message if($result)
{
   echo "New entry with DN " . $dn . " added to LDAP directory."; }
// else display error else
{
   echo "An error occurred. Error number " . ldap_errno($conn) . ": " .
ldap_err2str(ldap_errno($conn));
}
// all done? clean up
ldap_close($conn);
?>
</body>
</html>
```

Resultado:

New entry with DN mail=tim@my-domain.com, dc=my-domain, dc=com" added to LDAP directory.

El siguiente código nos sirve para **Editar** un usuario del directorio LDAP……



edit.php

```
<html>
<head>
</head>
<body>

<table width="450" cellpadding="5" cellspacing="5" border="1">
<?php
// specify the LDAP server to connect to
$conn = ldap_connect("localhost") or die("Could not connect to server");
// bind to the LDAP server specified above
$r = ldap_bind($conn) or die("Could not bind to server");
// set base DN and return attribute list
$base_dn = "dc=my-domain,dc=com";
$params = array("mail", "cn", "sn");
// perform search using email address passed on URL
$result = ldap_list($conn, $base_dn, "mail=" . urldecode($_GET['mail']), $params);
// extract data into array
$info = ldap_get_entries($conn, $result);

// print and display as editable form
?>
<form method="POST" action="modify.php">                                    // MODIFICA
 <table border="0" cellpadding="0" cellspacing="10" width="500" >
   <tr>
    <td width="50%" align="right">First Name</td>
    <td width="50%"><input type="text" name="cn" size="20" value="<?php echo $info[0]["cn"][0]; ?>"></td>
   </tr>
   <tr>
    <td width="50%" align="right">Last Name</td>
    <td width="50%"><input type="text" name="sn" size="20" value="<?php echo $info[0]["sn"][0]; ?>"></td>
   </tr>
   <tr>
    <td width="50%" align="right">E-mail</td>
    <td width="50%"><input type="text" name="mail" size="20" value="<?php echo $info[0]["mail"][0]; ?>"></td>
   </tr>
   <tr>
    <td width="100%" colspan="2" align="center">
     <input type="submit" value="Submit" name="Submit">
        <input type="reset" value="Reset" name="Reset">
    </td>
   </tr>
 </table>
</form>

<?php
// all done? clean up
ldap_close($conn);
?>

</table>
</body>
</html>
```

modify.php

```
<html>
<head>
</head>
<body>

<?php

// specify the LDAP server to connect to
$conn = ldap_connect("localhost") or die("Could not connect to server. Error is " .
ldap_error($conn));

// user with privileges to add an entry to LDAP hierarchy
$root_dn = "cn=root, dc=my-domain, dc=com";
$root_pw = "secret";

// bind to the LDAP server specified above
$r = ldap_bind($conn, $root_dn, $root_pw) or die("Could not bind to server. Error is " .
ldap_error($conn));

// prepare data
$info["cn"] = $_POST['cn'];
$info["sn"] = $_POST['sn'];
$info["mail"] = $_POST['mail'];
$info["objectClass"] = "inetOrgPerson";

// prepare DN
$dn = "mail=" . $_POST['mail'] . ", dc=my-domain, dc=com";

// modify data in the directory
$result = ldap_modify($conn, $dn, $info);

// if successful, display success message
if($result)
{
   echo "Entry with DN " . $dn . " modified in LDAP directory."; } // else display error
else
{
   echo "An error occurred. Error number " . ldap_errno($conn) . ": " .
ldap_err2str(ldap_errno($conn)); }

// all done? clean up
ldap_close($conn);

?>

</body>
</html>
```

Resultado:

Entry with DN mail=tim@my-domain.com, dc=my-domain, dc=com" modified in LDAP directory.

El siguiente código nos sirve para **Borrar** un usuario del directorio LDAP……

delete.php

```
<html>
<head>
</head>
<body>
<?php

// specify the LDAP server to connect to
$conn = ldap_connect("localhost") or die("Could not connect to server. Error is " .
ldap_error($conn));

// set privileged user
$root_dn = "cn=root, dc=my-domain, dc=com";
$root_pw = "secret";

// bind to the LDAP server specified above
$r = ldap_bind($conn, $root_dn, $root_pw) or die("Could not bind to server. Error is " .
ldap_error($conn));

// prepare DN for entry to delete
$dn = "mail=".$_GET['mail']."", dc=my-domain, dc=com";

// delete the entry from the directory
$result=ldap_delete($conn, $dn) ;

// if successful, display success message
if($result)
{
   echo "Entry with DN " . $dn . " deleted from LDAP directory."; } // else display error
else
{
   echo "An error occurred. Error number " . ldap_errno($conn) . ": " .
ldap_err2str(ldap_errno($conn)); }

// all done? clean up
ldap_close($conn);

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

Resultado:

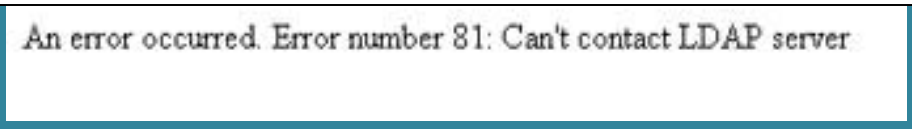
Entry with DN mail=tim@my-domain.com, dc=my-domain, dc=com" deleted from LDAP directory.

En los anteriores códigos se utilizó la función **ldap_errno()** que retorna un número en caso de que exista un error. Este número no nos indica mucha información. Para ello, se utiliza la función **ldap_err2str()** que retorna un MENSAJE asociado a dicho número.

En el siguiente ejemplo se intentará conectar a un servidor LDAP y veremos el error generado:

```
<html>
<head>
</head>
<body>
<?php
// specify the LDAP server to connect to
$conn = ldap_connect("www.somewhere.com") or die("Could not connect to server");

// bind to the LDAP server specified above
$r = ldap_bind($conn);

// if not successful, display error message
if(!$r)
{
   echo "An error occurred. Error number " . ldap_errno($conn) . ": " . ldap_err2str(ldap_errno($conn)); }

// further processing as required
// all done? clean up
ldap_close($conn);
?>
</body>
</html>
```

Resultado:



Equivalentemente se puede usar el siguiente código que utiliza la función **ldap_error** que retorna el último mensaje de error generado.

```
<html>
<head>
</head>
<body>
<?php

// specify the LDAP server to connect to
$conn = ldap_connect("www.somewhere.com") or die("Could not connect to server");;

// bind to the LDAP server specified above
$r = ldap_bind($conn);

// if not successful, display display error message
if(!$r)
{
   echo "An error occurred - " . ldap_error($conn);
}

// further processing as required
// all done? clean up
ldap_close($conn);

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

Más información sobre LDAP en: http://www.php.net/manual/en/ref.ldap.php

Anexo I.- Conexión entre PHP (en Linux) y DB2 (en AS400).

http://dns.celleweb.de/db2/db2howto.eng.html

Software requirements:
Linux:
gilbc 2.2, Kernel 2.2 (SuSE 7.0)
IBM DB2 Connect Personal Edition 7.1 für Linux Apache
1.3.19 or later
PHP 4.0.4pl1 or later

AS400: OS400>=
V4.2 DB2 >=V4r2

1. install DB2 Personal Connect Edition on Linux.
download the files and unpack them in a proper directory. install with db2setup all packages you need and create a db2-instance (db2inst1). Look also on SuSE Support database.

2. configure PHP
PHP has to be compiled with the option --enable-ibm-db=/usr/IBMdb2/V7.1/. On my system the lib /usr/lib/libdb2.so.1 was missing. My solution was to create a link to this lib, but you can also alter the ld.so.conf-file.

3. Apache
After compiling and installing apache mit php-support, you have to add environment variables to the apache startup-script. The easiest thing is to call the startupskrikpt of the db2instance: . /usr/lib/db2/db2inst1/sqllib/db2profile

4. DB2 Client connection
The installation of the client-connection is a little complicated. You have to login as db2-instance-user, i.e. db2inst1, and to call the db2 utility:
db2inst1@linux:~ >db2

# Catalog connection:
db2 => catalog tcpip node as400 remote ip-address or name server 446 or name of service from /etc/services
db2 => terminate
this seems to be important to update the directory.
# Call db2 again and catalog the database
db2 => catalog database as400 [as as400db] at node as400 authentication dcs db2
=> terminate
The authentication is very important, because otherwise you got the errormessage: SQL1400N Authentication not supported

# catalog database as DCS
db2 => catalog dcs database as400 as as400 db2
=> terminate

This was the easiest part, now you have to config the as400.

5. Configuration of the AS400

The settings on the AS400 are not well explained, because i have no experience on as400 You have to call WRKRDBDIRE on the as400 to put "*LOCAL" for AS400 Database.
Next is to start DRDA-Services on as400. If you allready have ODBC-Connections, it will be the case, but it is essential for linux.

If you test the connection no with:
connect as400 user test using test you
got the errormessage:

SQL30081N A communication error has been detected. Communication protocol being used: "TCP/IP". Communication API being used: "SOCKETS". Location where the error was detected: "". Communication function detecting the error: "connect". Protocol specific error

code(s): "111", "*", "*". SQLSTATE=08001

If this errormessage appears the DRDA Listener has to be startet: STRTCPSVR SERVER(*DDM).
The service can be startet automatically:
CHGDDMTCPA AUTOSTART(*YES)

If you try a connection now, you got the following messsage:
SQL0332N There is no avaiable conversion for the source code page "65535" to the target code page "819". Reason Code "1". SQLSTATE=57017

This means, that you have to set the codepageconversion (CCSID) for the specific user:
CHGUSRPRF USRPRF(test) CCSID(273)
I have testet the above setting, which works very fine.

6. Connecting as400 from PHP/linux

```
<?
$dbname="AS400";
$dbuser="test";
$dbpwd="test";
// Connect to database
$dbid=odbc_connect($dbname,$dbuser,$dbpwd);

$sql="select count(*) from as400.library.table";
$obid=odbc_exec ($dbid,$sql);
....
odbc_close($dbid);
?>
```

7. Known Bugs:
One known error is, that odbc_num_rows return -1 an not the number of resultrows. The driver doesn't support rownumber in odbc_fetch_row or odbc_fetch_into. But that isn't a problem.

8. Unknown bugs an problems
The best hints on connecting to the db2 can be found in the newsgroup-archive at google groups.google.com. i got all the information from there only put some significant words or the error messages in the searchfield and you got a lot of answers. The documentation from IBM is not very well in most cases. The explanation of error-messages is quiet bad.

9. Suggestions, Contact to the author
If you have any suggestions or other hints to solve problems to connect to As400 db2 get in contact with me: Email: s.dreyer@celleweb.de. I will add your suggestions to this howto.

Más referencias en:

http://www.php-faq.com/as400.php http://www.faqts.com/knowledge_base/view.phtml/aid/5613/fid/14

Anexo II.- Aplicación de Calendario Simple

Esta aplicación se puede agregar a una pagina web que ya exista y permite al usuario ver un calendario simple, pudiendo navegar con los meses y años usando las flechas de “>>” y “<<”.

NOTA: Para que se pueda realizar un llamado al comando **cal** de UNIX, la variable **“safe_mode”** debe estar en **off** en el archivo **php.ini**

cal.php

```
<?
$_PHP_LINK = $PHP_SELF;

$meses = array(1=>"Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio", "Julio", "Agosto", "Septiembre",
"Octubre", "Noviembre", "Diciembre");
$WeekDays = array ("Do", "Lu", "Ma", "Mi", "Ju", "Vi", "Sa");

$fontb = "<font color=#FFFFFF>" ;
$fonte = "</font>";

if (!isset($Day)) {
          $month = date ("n");
          $year = date ("Y");
          $Day = time();
 }
 else {
          $month = date ("n", $Day);
          $year = date ("Y", $Day);
}

$output = `cal $month $year`;
$starter = date ("w", mktime (0, 0, 0, $month,1, $year));
$strings = split ("\n", $output);
?>

<html>
<style type="text/css">
<!-- a:hover { color: #FF0000; text-decoration: none} a:link {
text-decoration: none}
a:visited { text-decoration: none}
.small_letter { font-size: xx-small; color: #FFFFFF} -->
</style>

<title>Pequeno Calendario</title>

<body>
<table border="0" cellspacing="0" cellpadding="0">
<tr bgcolor="#3366CC">
<td bgcolor="#3366CC">  </td>
<td align=center class="small_letter" valign="top">
 $NextMonth = mktime (0,0,0, date("n", $Day) + 1, date("j",$Day), date ("Y", $Day));
 $PreviousMonth = mktime(0,0,0 , date("n",$Day)-1, date("j",$Day), date ("Y", $Day)); print("<a
 href=\"$_PHP_LINK?Day=$PreviousMonth\">$fontb << $fonte</a> ".$meses[date ("n",
 $Day)]. "  <a href=\"$_PHP_LINK?Day=$NextMonth\"> $fontb >>$fonte</a>");

 print("<table bgcolor=\"#FFFFFF\" class=\"small_letter\" > <tr bgcolor=\"#3366CC\"
 class=\"small_letter\">");

 $splited = split(" ", $strings[1]);
 $i=0;

 for ($i=0; $i<7; $i++ )  {
                    print (" <td> ".$WeekDays[$i]." </td>");
```