FORMULARIOS

RECOGIDA DE DATOS POR TECLADO MEDIANTE FORMULARIOS

En una página web, los datos se introducen mediante formularios. En la mayoría de las ocasiones tendremos dos páginas: una página con un formulario que recoge los datos y otra página que los procesa.

A continuación tenemos una página con nombre **index.php** (se podría llamar también index.html ya que no contiene código PHP):

```
<html>
  <head>
    <meta charset="UTF-8">
  </head>
  <body>
    Introduce tu nombre:
    <form action="saluda.php" method="get">
    <input type="text" name="nombre"><br>
    <input type="submit" value="Enviar">
    </form>
  </body>
</html>
```

Al pulsar el botón **Enviar**, el contenido del formulario se envía a la página que indicamos en el atributo **action**. La página saluda.php tendría el siguiente código:

```
<html>
  <head>
    <meta charset="UTF-8">
  </head>
  <body>
    Hola <?php $_GET['nombre']; ?>,
    que tengas un buen día.
  </body>
</html>
```

Observa que la información que se quiere enviar se introduce en un cuadro de texto dentro del formulario (página **index.php**). A esa información la llamamos, en este caso, **nombre**. Para recoger esa información se utiliza $_GET['**nombre**'].

1. Cuando los datos son números

Cuando los datos que se recogen y se manipulan son números en lugar de cadenas de caracteres, el procedimiento es el mismo.

En el siguiente ejemplo tenemos una aplicación que suma dos números. El programa **index.php** recoge los datos y el programa **suma.php** suma los dos números que recibe y muestra el resultado.

```
<!DOCTYPE html>
<html> <head>
<meta charset="UTF-8"> </head>
<body>
<h3>Calcula la suma de dos números</h3>
<form action="suma.php" method="get">
  Primer número: <input type="number" name="a"><br>
  Segundo número: <input type="number" name="b"><br>
  <input type="submit" value="Sumar">
</form>
</body>
</html>
```

Fíjate que los campos input del formulario son de **tipo number.** Se podrían declarar de tipo text y el programa funcionaría sin problemas pero se declaran number para que el formulario no deje pasar los datos que no sean numéricos.

Imagínate que el usuario introduce la palabra hola en un campo en lugar de un número. Si el input es de tipo text, la palabra hola llegaria a suma.php y se intentaría sumar, lo que provocaría un error. Sin embargo, si el input es de tipo number, el formulario no envía hola sino que muestra un mensaje diciendole al usuario que introduzca un número.

En la medida de lo posible, intenta utilizar siempre el tipo adecuado en los campos input; por ejemplo number, color, date, email, url, etc. de tal forma que los datos que envía el formulario tengan el formato correcto.

A continuación se muestra el fichero **suma.php.**

```
<!DOCTYPE html>
<html> <head>
<meta charset="UTF-8"> </head>
<body>
  <?php
    $a = $_GET['a'];
    $b = $_GET['b'];
    echo "La suma de $a mas $b es ", $a + $b;
  ?>
</body> </html>
```

2. Métodos GET y POST

Mediante el atributo **method** de la etiqueta **form** se debe especificar un método de envío; los dos métodos posibles son get y post. El resultado final es el mismo, la diferencia radica en que con el método **get** se pueden ver los parámetros que envía el formulario en la barra de direcciones del navegador.

GET agrega pares de name/value a la URL y de esta manera pasa los valores recolectados del formulario al archivo PHP de destino.

El método **POST** incrusta los pares name/value dentro del cuerpo de la solicitud HTTP y la información se envía al servidor.

Nota : lo veremos más adelante (Métodos de envío GET Y POST)

3. $_REQUEST

Cuando se envía un formulario, PHP almacena la información recibida en una matriz llamada $_REQUEST. El número de valores recibidos y los valores recibidos dependen tanto del formulario como de la acción del usuario.

Un array asociativo que por defecto contiene el contenido de $_GET, $_POST y $_COOKIE.

Para acceder al valor entregado por el formulario sería: **$_REQUEST[valor_del_atributo_name**]

Esta es una '**superglobal**' o una variable automatic global. Significa simplemente que es una variable que está disponible en cualquier parte del script. No hace falta hacer **global $variable;** para acceder a la misma desde funciones o métodos.

Las variables en **$_REQUEST** se proporcionan al script a través de los mecanismos de entrada GET, POST, y COOKIE y por lo tanto pueden ser manipulados por el usuario remoto y no debe confiar en el contenido.

```
<?php

$_GET['foo'] = 'a';
$_POST['bar'] = 'b';
var_dump($_GET); // Element 'foo' is string(1) "a"
var_dump($_POST); // Element 'bar' is string(1) "b"
var_dump($_REQUEST); // Does not contain elements 'foo' or 'bar'

?>
```

4. Cómo recoger datos para un array mediante un formulario

Imagina que quieres pedir diez números por teclado y quieres guardar esos números en un array. Se puede pedir un número mediante un formulario, a continuación el siguiente, luego el siguiente, etc. pero ¿cómo enviarlos? Hay un truco muy sencillo. Se pueden ir concatenando valores en una cadena de caracteres y el resultado de esa concatenación se puede reenviar una y otra vez en un formulario como campo oculto. Por último, se puede utilizar la **función explode()** para convertir la cadena de caracteres en un array.

Es importante señalar que los valores que se van concatenando deben tener algún tipo de separación dentro de la cadena, por ejemplo un espacio en blanco.

A continuación se muestra un ejemplo completo:

```
<!DOCTYPE html>
<html>
  <head>
<meta charset="UTF-8"> </head>
<body>
<?php
      $n = $_GET['n'];
      $contadorNumeros = $_GET['contadorNumeros'];
      $numeroTexto = $_GET['numeroTexto'];
      if (!isset($n)) {
        $contadorNumeros = 0;
        $numeroTexto = "";
      }
      // Muestra los números introducidos
       if ($contadorNumeros == 6) {
          $numeroTexto = $numeroTexto . " " . $n; // añade el último número
leído
```

```
            $numeroTexto = substr($numeroTexto, 2); // quita los dos primeros
            $numero = explode(" ", $numeroTexto);
            echo "Los números introducidos son: ";
            foreach ($numero as $n) {
            echo $n, " ";
            }
        }
        // Pide número y añade el actual a la cadena
        if (($contadorNumeros < 6) || (!isset($n))) {
        ?>

        <form action="#" method="get">
        Introduzca un número:
          <input type="number" name ="n" autofocus>
          <input type="hidden" name="contadorNumeros" value="<?=
++$contadorNumeros ?>">
          <input type="hidden" name="numeroTexto" value="<?= $numeroTexto . " "
. $n ?>">
          <input type="submit" value="OK">
        </form>
        <?php
        }
      ?>
    </body>
</html>
```

5. Controles en formularios

http://www.mclibre.org/consultar/php/lecciones/php-controles.html

6. Recogida de datos

http://www.mclibre.org/consultar/php/lecciones/php-recogida-datos.html