

PHP 3.1

FORMULARIOS. MANEJO DE FICHEROS. DE VARIABLES. SERVIDOR

1. FORMULARIOS

1.1. ELEMENTOS DE ENTRADA A FORMULARIO. ACCESO DESDE PHP

1.2. PASO DE DATOS A UN FORMULARIO

1.2.1. Con variables globales

1.2.2. Con arrays asociativos

1.2.3. Por la URL

1.2.4. Paso de Matrices a PHP

1.2.5. Por QUERY STRING

1.3. PROCESAMIENTO DE FORMULARIOS EN PHP

1.3.1. En un solo script. FORMULARIOS REENTRANTES

1.3.2. Validación de formularios desde servidor

2. MANEJO DE FICHEROS

3. MANEJO DE DIRECTORIOS

4. EJECUCIÓN DE COMANDOS DEL SISTEMA OPERATIVO EN PHP

5. ENVÍO DE FICHEROS AL SERVIDOR

1. FORMULARIOS

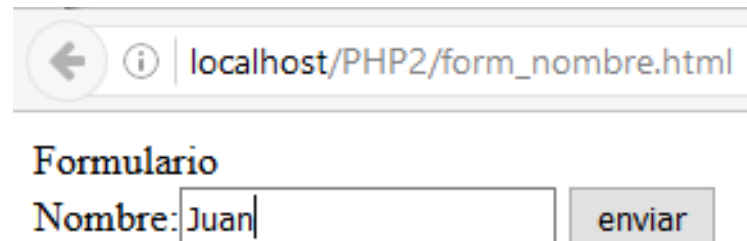
El lenguaje PHP nos proporciona una manera sencilla de manejar formularios, permitiendo procesar la información que el usuario ha introducido.

Un formulario hace de cada objeto (cajas de texto, botones de radio , etc.) una variable, que tendrá aparejado un dato.

Suponemos el siguiente formulario (archivo **form_nombre.html**):

```
<html>
  <body>
    <div>Formulario Nombre</div>
    <form action="form_nombre.php" method="get">
      nombre:<input type="text" name="nombre" value="" size="20">
      <input type="submit" name="enviar" value="enviar">
    </form>
  </body>
</html>
```

Al ejecutarlo tendremos:



Formulario

Nombre:

Cuando el usuario pulsa el botón de envío de un formulario (*SUBMIT*) se manda al servidor una cadena de la forma:

nombre=valor&nombre=valor&nombre=valor...

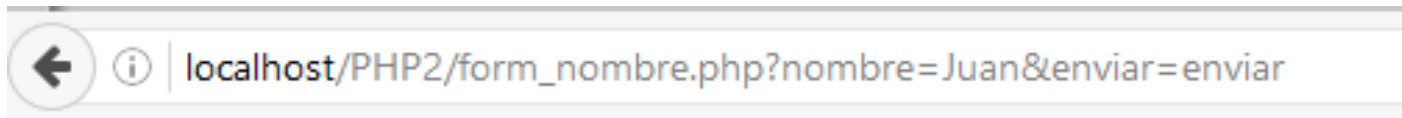
La manera de recibirla dependerá del atributo **METHOD** que hayamos puesto en el FORM:

Ejemplo:

Si enviamos los datos mediante **GET**, se procesarán por el siguiente script **procesa_formulario.php**: y podemos observar que los pares *nombre-valor* van en la **URL**:

http://localhost/PHP2/form_nombre.php?nombre=juan&enviar=enviar

Al pulsar *Enviar* veremos la salida:



hola Juan

Método GET:

- Los datos se envían en un mensaje **HTTP-GET** que contiene la cadena con los pares **nombre=valor** precedidos del carácter '?' a la **dirección URL** con la solicitud del script que tratará los datos.
- Esta cadena se denomina **QUERY_STRING** y si usamos un CGI se almacenará en una variable de entorno con ese nombre en el servidor.
- PHP almacenará estos valores en la matriz **superglobal \$_GET** utilizando como índice del valor el nombre **\$_GET['nombre']**.
- También podremos usar la matriz **\$_REQUEST** para acceder a un valor recibido intermediando HTTP-GET.

Método PUESTO:

- Los datos se envían en un mensaje **HTTP-POST** donde se pasan los pares **nombre-valor** en **el propio cuerpo del mensaje de solicitud** http y **no en la URL**.
- PHP almacenará estos valores en la matriz **superglobal \$_POST** utilizando como índice del valor el nombre **\$_POST['nombre']**.
- Tendremos que usar este estilo intermedio de referencia y no el estilo corto y largo, tal como comentábamos en el GET.
- También podremos usar la matriz **\$_REQUEST** para acceder a un valor recibido mediante HTTP-POST.

Si recordamos, la etiqueta HTML del formulario (<form>) tiene varios atributos que nos interesan:

- `name`: nombre del formulario.
- `method`: define la modalidad de envío de la información. Los valores posibles son **POST** o **GET**;
- `action`: indica la acción o el destino para los datos del formulario , en nuestro caso apuntará al fichero php que manipulará estos datos
- `enctype`: especifica el tipo de encriptación que se realizará con los datos que se enviaron . Este atributo solo se aplica si `method` es POST.

Nota:

Por defecto es "*application/x-www-form-urlencoded*".

Otro posible valor es "*multipart/form-data*" que utilizamos cuando tenemos en el formulario campos de tipos fichero.

- `target`: nombre del marco donde se visualizará la información devuelta por el servidor. Funciona igual que en el elemento ancla de hipertexto A.

La acción de pulsar el botón de envío del formulario desencadena la transmisión de ciertos datos a una página receptora (***action=página.php***), a una dirección de correo electrónico (***action=mailto:micorreo@correo.es***) o un programa o CGI para su tratamiento.

PHP entiende todos los datos de un formulario como un **array asociativo**, de forma que podremos utilizar con él funciones de arrays.

ejemplo: `count($_TABLA)`.

Funciones php relacionadas con formularios:

Uno de los problemas que podemos tener cuando usamos formularios es el tipo de datos con que los usuarios rellenarán los campos.

Se puede introducir caracteres extraños. En lo referente a esto hay varias funciones *de limpieza* que nos pueden ayudar:

- `htmlspecialchars($variable)` → convierte los signos `<`, `>`, `&` y `"` en sus correspondientes entidades HTML (`<`, `>`, `&Amp;`, `"`)
- `striplashes($variable)` → elimina todas las barras invertidas de una cadena
- `nl2br($variable)` → convierte todos los saltos de línea del editor en el equivalente HTML
- `isset($variable)` → Si no rellenamos todos los campos de un formulario y pulsamos el botón de Enviar, podemos tener errores. En este caso es útil la función `isset($variable)`, que comprueba si la variable se ha definido (*true*) o no (*false*).
- `empty($variable)` → sirve para comprobar si una variable no está asignada, si está vacía o si corresponde a valor 0.

Ejemplo completo:

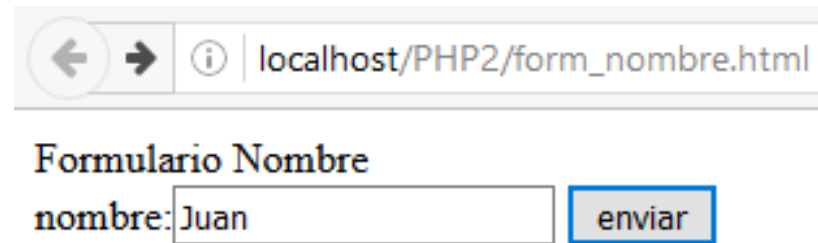
Form_nombre.HTML:

```
<body>
    <div>Formulario Nombre</div>
    <form action="form_nombre.php" method="get">
        nombre:<input type="text" name="nombre" value="" size="20">
        <input type="submit" name="enviar" value="enviar">
    </form>
</body>
```

Form_nombre.php:

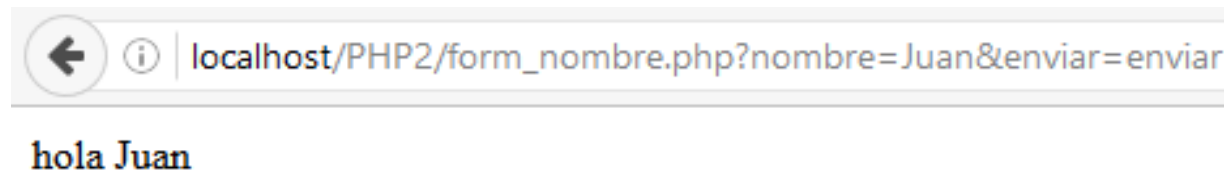
```
if (!empty ($_GET['nombre']))
    echo 'hola ' . $_GET['nombre'];
else
    echo 'no hay datos para visualizar';
```

Ejecución:



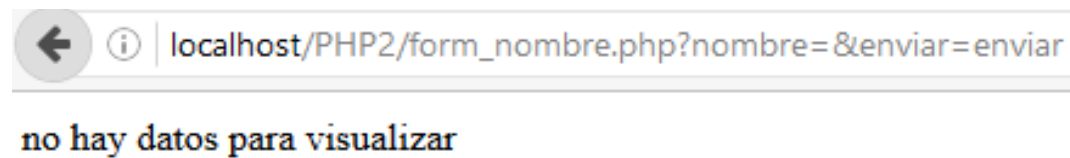
A screenshot of a web browser window. The address bar shows 'localhost/PHP2/form_nombre.html'. Below the address bar, the page title is 'Formulario Nombre'. The form consists of a label 'nombre:' followed by a text input field containing the text 'Juan'. To the right of the input field is a button labeled 'enviar'.

Introducimos Nombre y pulsamos Enviar:



A screenshot of a web browser window. The address bar shows 'localhost/PHP2/form_nombre.php?nombre=Juan&enviar=enviar'. Below the address bar, the text 'hola Juan' is displayed.

Si no Introducimos Nombre y pulsamos Enviar, veremos:



A screenshot of a web browser window. The address bar shows 'localhost/PHP2/form_nombre.php?nombre=&enviar=enviar'. Below the address bar, the text 'no hay datos para visualizar' is displayed.

Nota:

Para comprobar si una variable de un formulario tiene asignado valor también se puede hacer:

```
<?php
if (!empty($_POST["comentario"])) {
    $comment=htmlspecialchars($_POST["comentario"]);
    $comment=stripslashes($comment);
    $comment=nl2br($comment);
    echo "Has escrito:".$comment;
}
?>
```

1.1 ELEMENTOS DE ENTRADA A FORMULARIO. ACCESO DESDE PHP

Un formulario permite los siguientes elementos para la entrada de información :

- Elementos de tipo INPUT
 - TEXT
 - RADIO
 - CHECKBOX
 - BUTTON
 - FILE
 - HIDDEN
 - PASSWORD
 - SUBMIT
- Elemento SELECT
 - Simple / múltiple
- Elemento TEXTAREA

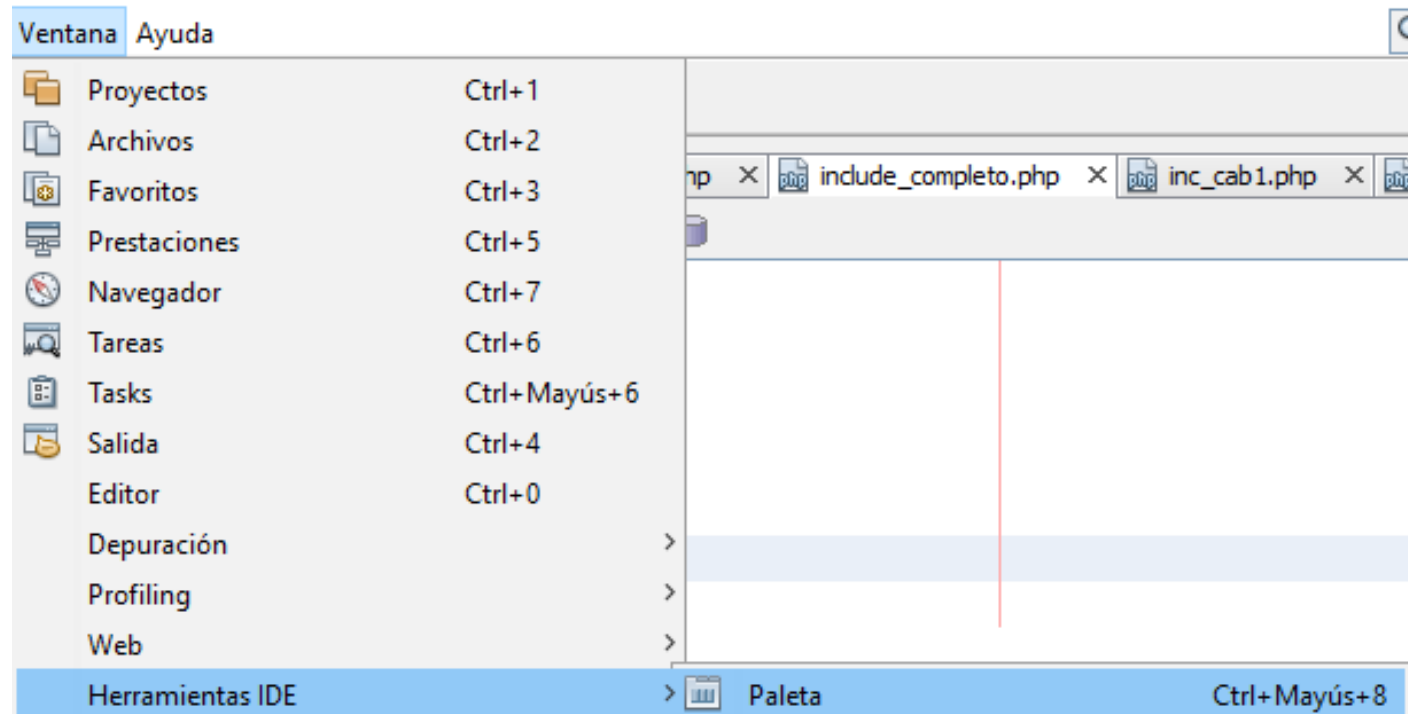
Nota:

Existen muchas herramientas para facilitarnos el diseño y generación de código HTML, pero nosotros usaremos:

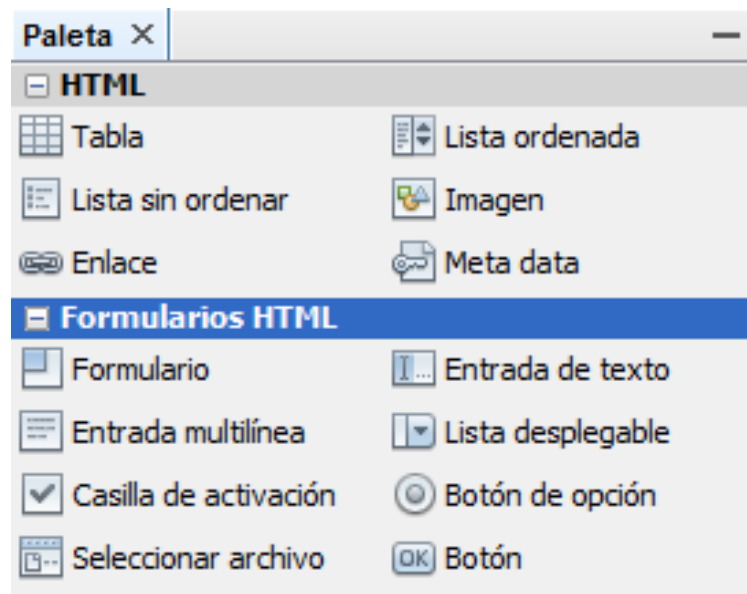
- NETBEANS
- DreamWeaver
- CKeditor4

COMO INCLUIR ELEMENTOS DE FORMULARIO DESDE NETBEANS:

Desde VENTANA - HERRAMIENTAS IDE – PALETA:



Nos muestra una ventana como la siguiente:



Ahora, mediante **Seleccionar- Arrastrar y soltar**, podemos incluir cada uno de esos elementos en nuestra aplicación

INPUT TEXTO

```
<h4>Formulario Texto</h4>
<form action="formulario_salida.html" method="get" >
  Introduce texto: <input type="text" maxlength="10" size="10" name="nombre">
</form>
```

Ejecución:

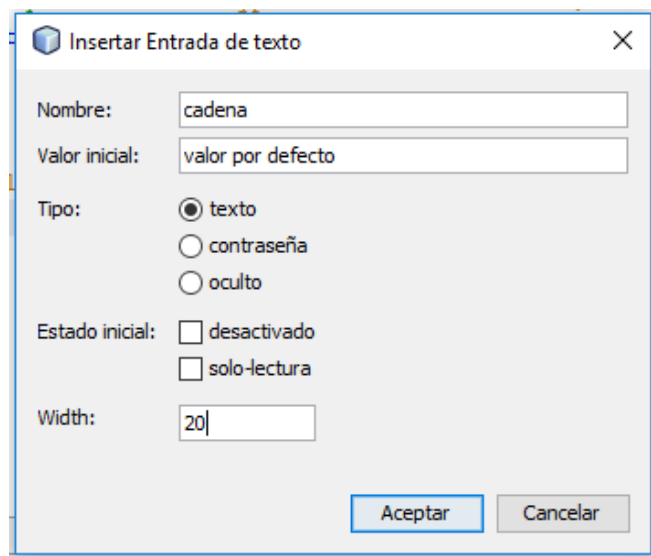
Formulario Texto

Introduce texto:

Desde NETBEANS:

Ver Archivo *formularioEjemplos.php* del PROYECTO del tema:

PALETA - ENTRADA DE TEXTO:



INPUT RADIO

HTML:

```
Matrícula:<br>  
<input type="radio" name="matricula" value="o" checked="checked" />Ordinaria  
<input type="radio" name="matricula" value="l" />Libre<br><br>
```

Php:

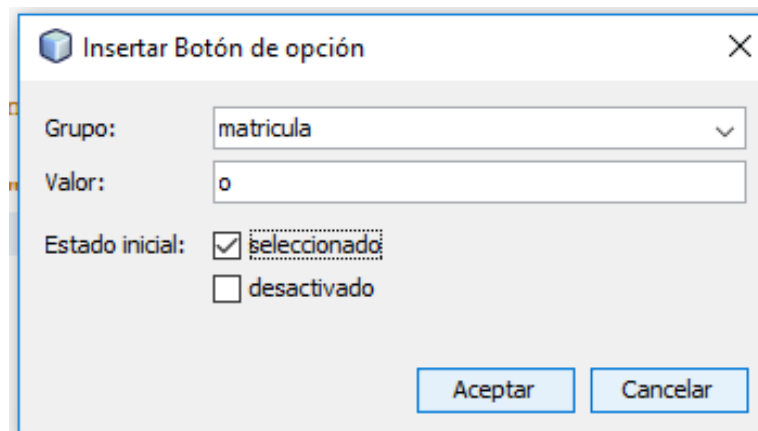
```
<?php  
$matricula = $_request['matricula'];  
echo $matricula; // visualizará o o l  
?>
```

Ejecución:

Matrícula:

☒ Ordinaria ☐ Libre

Desde NETBEANS: PALETA - BOTÓN DE OPCIÓN:



INPUT CASILLA DE SELECCIÓN

HTML:

```
<b>Idioma:</b><br>
<input type="checkbox" name="optativas[]" value="log" checked="checked" />Lógica<br>
<input type="checkbox" name="optativas[]" value="fra" />Francés<br>
<input type="checkbox" name="optativas[]" value="inf" />Informática<br><br>
```

Php:

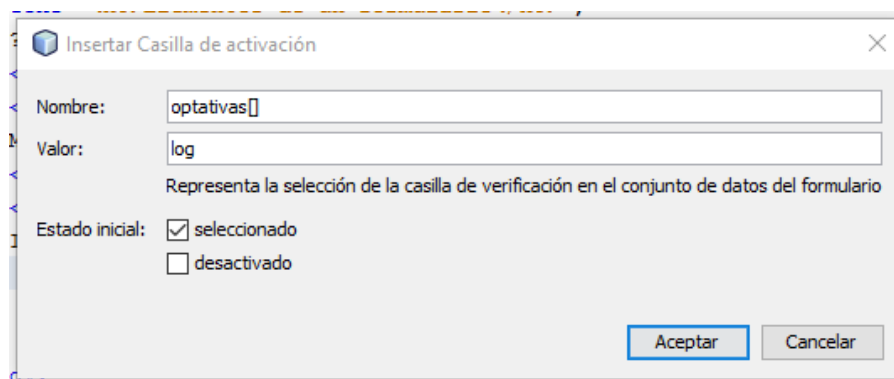
```
<?php
$optativas = $_request['optativas']; // devolverá una matriz.
foreach ($optativas as $optativa)
    echo $optativa.'<br>';
?>
```

Ejecución:

Idioma:

- ☒ Lógica
- ☐ Francés
- ☐ Informática

Desde NETBEANS: PALETA –CASILLA De ACTIVACIÓN:



INPUT BUTTON

HTML:

```
<input type="button" value="Actualizar Datos" name="actualizar" /><br>
```

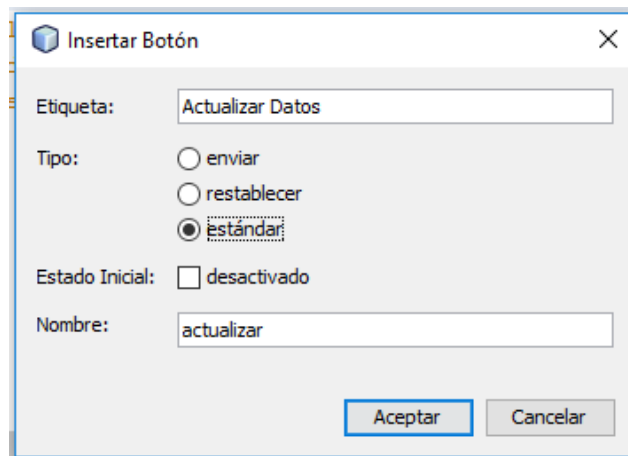
Php:

```
<?php  
$actualizar = $_request['actualizar'];  
if (isset($actualizar) == true)  
echo 'se han actualizado los datos';  
?>
```

Ejecución:

Actualizar Datos

Desde NETBEANS: PALETA – BOTÓN. Tipo estándar:



INPUT FILE multipart

HTML:

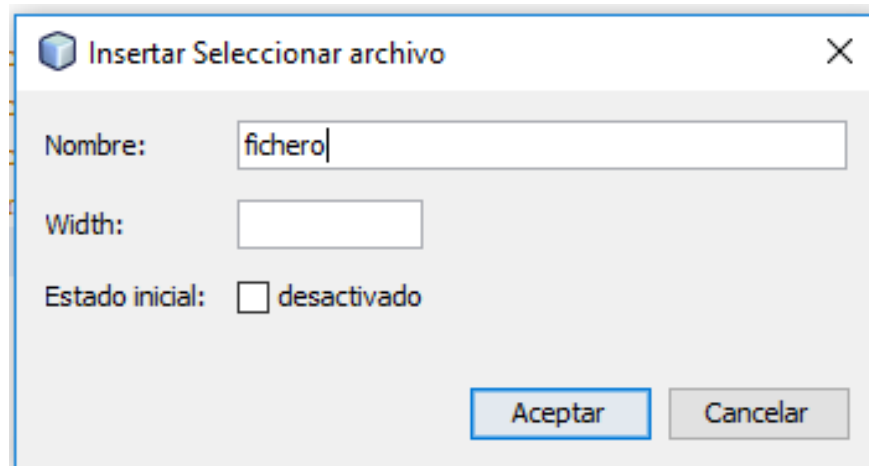
```
html:  
<form action="<?php echo $_SERVER['PHP_SELF'] ?>" method="post"  
  enctype="multipart/form-data">  
  <input type="file" name="fichero">  
</form>
```

Php:

Lo estudiaremos más adelante

Ejecución:

Desde NETBEANS: PALETA - SELECCIONAR ARCHIVO:



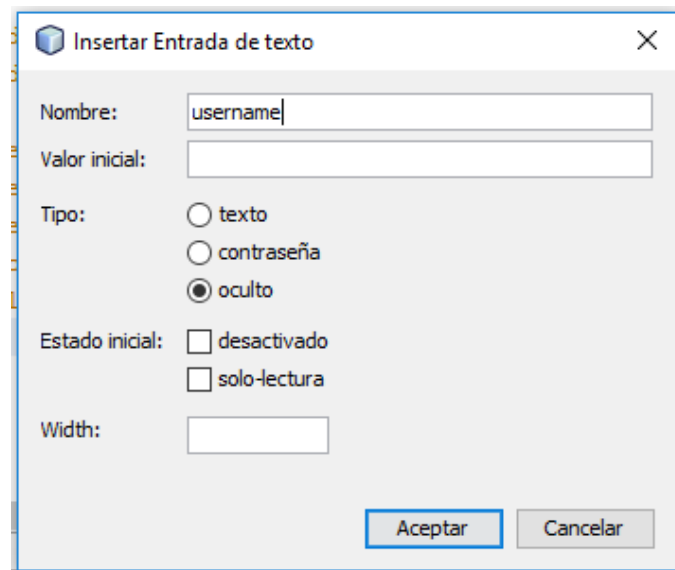
INPUT HIDDEN

```
html:  
<input type="hidden" name="username" value="<?php $usuario ?>">  
  
php:  
<?php  
$username = $_request['username'];  
echo $username;  
?>
```

Ejecución:

No se visualizará nada.

Desde NETBEANS: PALETA - ENTRADA DE TEXTO. Tipo oculto:

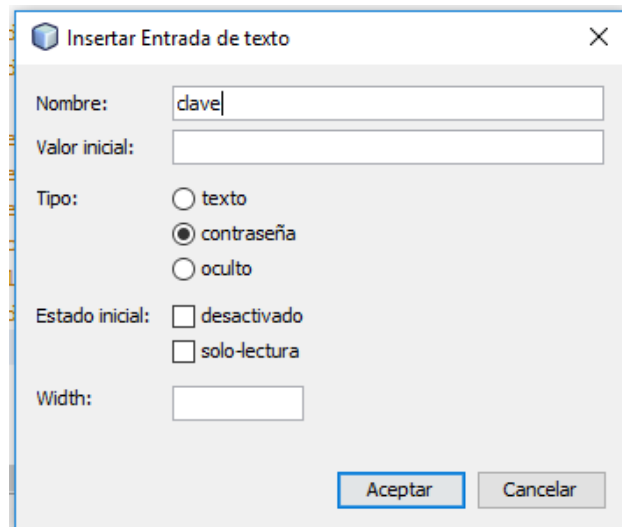


INPUT PASSWORD

```
html:  
contraseña: <input type="password" name="clave">  
  
php:  
<?php  
$clave = $_request['clave'];  
echo $clave;  
?>
```

Ejecución:

Desde NETBEANS: PALETA - ENTRADA DE TEXTO. Tipo Contraseña:



Insertar Entrada de texto

Nombre:

Valor inicial:

Tipo: ☐ texto ☒ contraseña ☐ oculto

Estado inicial: ☐ desactivado ☐ solo-lectura

Width:

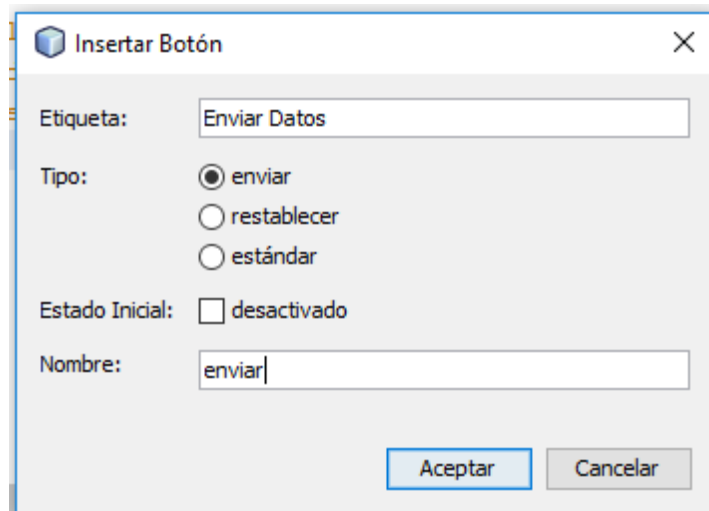
INPUT SUBMIT

```
html:  
<input type="submit" name="enviar" value="enviar datos">  
  
php:  
<?php  
$enviar = $_request['enviar'];  
if (isset($enviar) == true)  
    echo 'se ha pulsado el botón de enviar';  
?>
```

Ejecución:

Enviar

Desde NETBEANS: PALETA – BOTÓN. Tipo Enviar:



SELECT SIMPLE

```
html:
calificación:
<select name="calificación">
  <option value="nopresentado" selected>no presentado
  <option value="noapto">no apto
  <option value="apto">apto
</select>

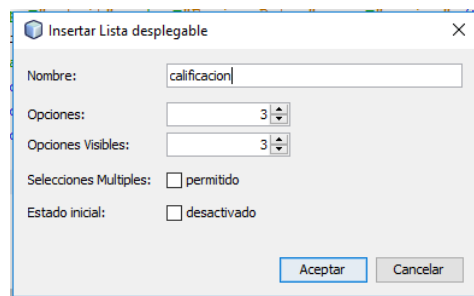
php:
<?php
$calificacion = $_request['calificación'];
echo $calificacion;
?>
```

Ejecución:

Calificación:

No Presentado
No Apto
Apto

Desde NETBEANS: PALETA – LISTA DESPLEGABLE:



- Una vez se han creado las opciones, se tiene que indicar nombre de cada opción, mediante el atributo *value* de option.
- También escribir el texto que deseamos que se vea en cada opción. Estará entre las etiquetas *<option>* y *</option>*

SELECT MÚLTIPLE

```
html:
<select multiple size="4" name="idiomas[]">
  <option value="valenciano" selected>valenciano
  <option value="frances">francés
  <option value="ingles">ingles
</select>

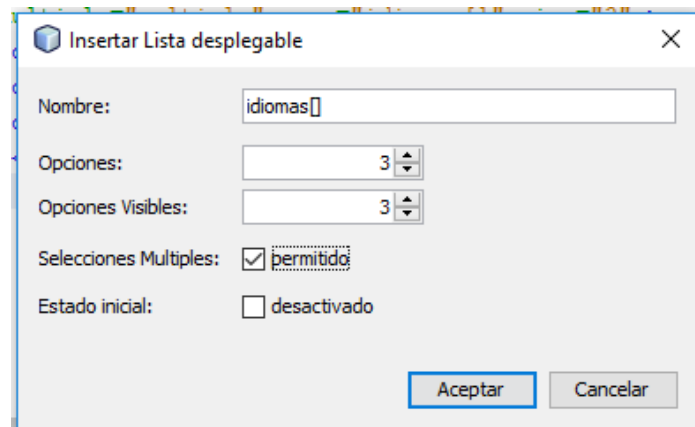
php:
<?php
$idiomas = $_request['idiomas'];
foreach ($idiomas as $idioma)
    echo '$idioma<br>';
?>
```

Ejecución:

Idioma:

Valenciano
Francés
Inglés

Desde NETBEANS: PALETA – LISTA DESPLEGABLE: Selecciones múltiples se marca Permitido :



TEXTAREA

```
html:
comentario:
<textarea cols="30" rows="3" name="comentario">
este libro me parece ...
</textarea>

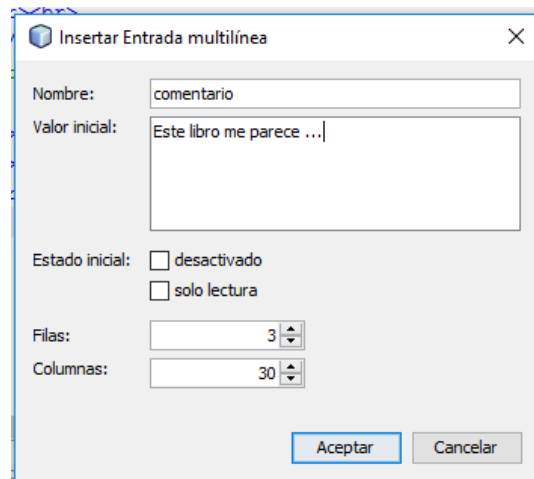
php:
<?php
$comentario = $_request['comentario'];
echo $comentario;
?>
```

Ejecución:

Comentario

Este libro me parece ...

Desde NETBEANS: PALETA- ENTRADA MULTILÍNEA:

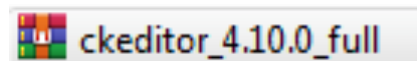


CÓMO INCLUIR ELEMENTOS DE FORMULARIO DESDE CKEditor4:

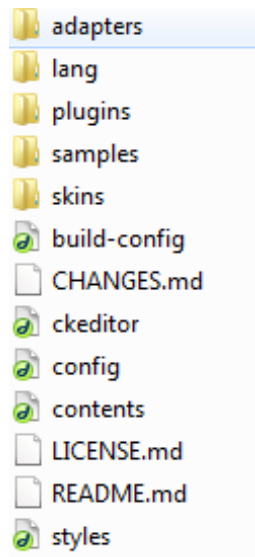
Descargar herramienta Desde:

<https://ckeditor.com/ckeditor-4/download/>

Descargamos archivo .zip:

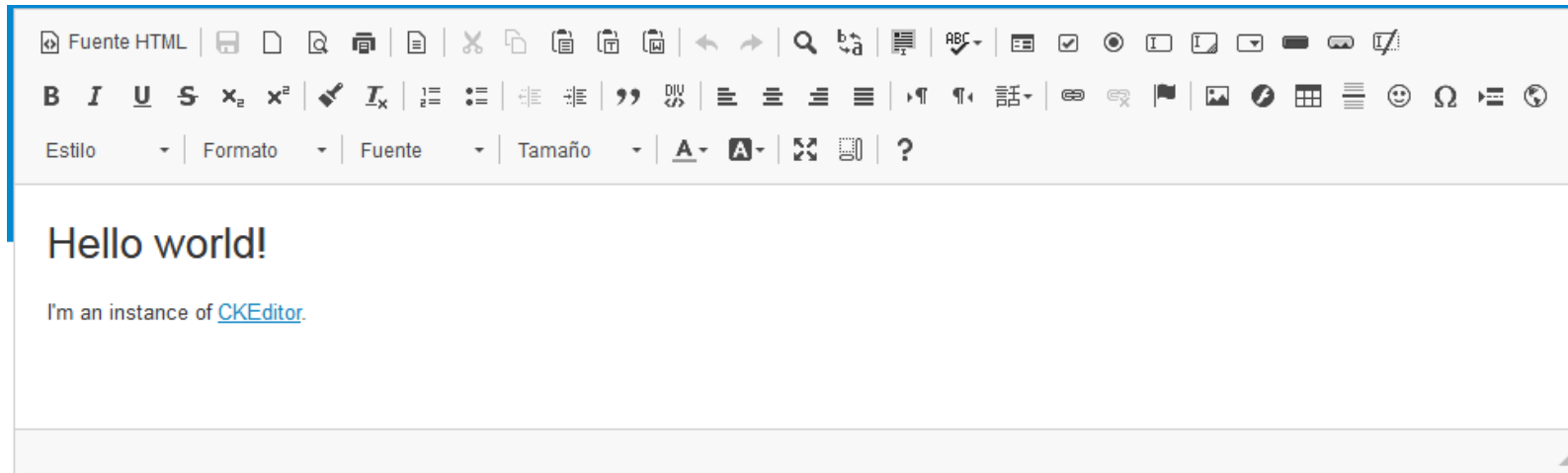


Lo descomprimos en cualquier carpeta del nuestro equipo. Nos generará la siguiente estructura de carpetas dentro de la carpeta **ckeditor**:

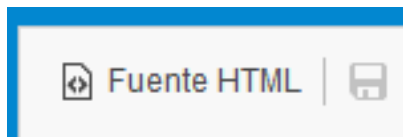


Para ejecutar entramos en **samples\index.HTML**.

Veremos esta pantalla:

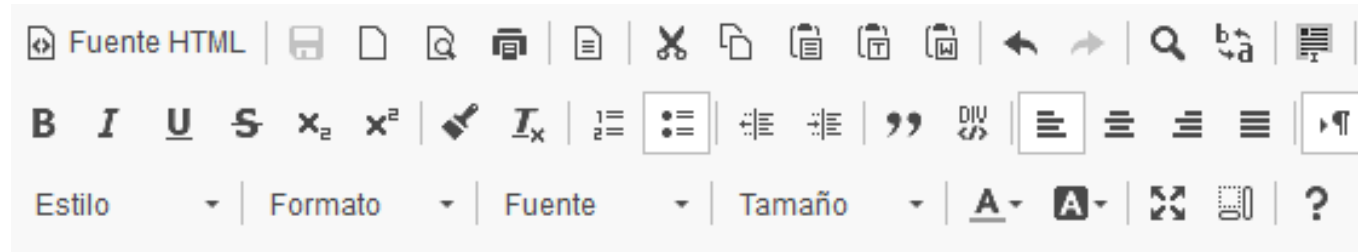


Ahora iremos seleccionando los iconos de cada tipo de elemento HTML de la Barra de Herramientas y, al pulsar la pestaña



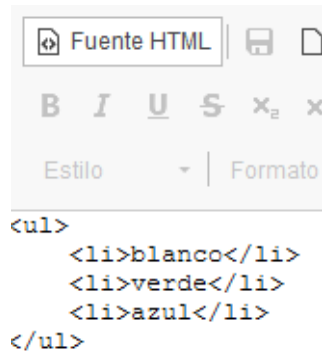
, veremos que código HTML que se ha generado.

Ejemplo: Escribo varios valores tipos viñeta y le doy formato de **lista no ordenada**:



- blanco
- verde
- azul

Veamos el HTML generado:



Buscar en internet más ejemplos, de Formularios, Tablas, etc.....

1.2 PASO DE DATOS EN FORMULARIOS

Las variables de un script tienen una validez exclusiva para el script y resulta imposible conservar su valor cuando ejecutamos otro archivo diferente, aunque los dos estén enlazados.

1.2.1 Con Variables Globales

Cada nombre de variable del formulario está disponible como variable **global (\$nombre)**.

Este es el sistema tradicional y el más simple de implementar, pero también más inseguro, por lo que se desaconseja.

Sólo funcionará si en el archivo *php.ini* la directiva ***register_globals*** está activada).

Ejemplo:

```
<input type="texto" name="nombre">
```

1.2.2 Con Arrays Asociativos

- **\$_POST:** Array asociativo que contiene las variables pasadas por el método POST
- **\$_GET:** Array asociativo que contiene las variables pasadas por el método GET
- **\$_REQUEST:** Array asociativo que contiene las variables pasadas por el método. REQUEST. Contiene toda la información de GET , POST y COOKIE.

Nota:

Como ya se dijo , \$_request[] es una variable GLOBAL y es accesible desde cualquier parte del script. A partir de PHP 5.4.0 solo se puede acceder a datos del formularios HTML usando \$_POST[] o \$_GET[].

La manera de acceder a los elementos del formulario , y que son elementos de estos arrays seria, **\$_POST['nombre']** donde "*nombre*" seria la una variable del formulario y, por tanto , equivaliendo a *\$nombre*, teniendo en ambos casos el valor introducido en el formulario.

Nota:

El elemento REQUEST_METHOD de la matriz global \$_SERVER indica el método de envío .

Ejemplo:

```
if($_SERVER["REQUEST_METHOD"]=="POST")
    print_r($_POST);
else
    print_r($_GET);
```

1.2.3 Paso de variables por la url

Para pasar los variables de una página a una otra lo podemos hacer introduciendo esta variable dentro del enlace hipertexto de la página destino.

La sintaxis sería la siguiente:

```
<a href="destino.php?variable1=valor1&variable2=valor2&..."> El mio enlace</a>
```

- Estas variables no poseen el símbolo \$ delante.
- Esta manera de pasar variables no es específico de PHP sino que es utilizado por otros lenguajes.
- Ahora nuestra variable pertenece también al ámbito de la página *destino.php* y está lista para su utilización.

Nota:

No siempre se definen automáticamente las variables recibidas por parámetro en las páginas web, depende de una variable de configuración de PHP : `register_globals`, que tiene que estar ***activada*** para que así sea.

Ejemplo:

Tenemos dos páginas, *origen.HTML* y *destino.php*:

origen.HTML

```
<html>
  <head>
    <title>origen.html</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  </head>
  <body>
    <a href="destino.php?saludo=hola&texto=Esto es una variable texto">
      Paso variables saludo y texto a la página destino.php</a>
  </body>
</html>
```

destino.php

```
<html>
<head>
<title>destino.php</title>
</head>
<body>
<?php
$saludo=$_GET["saludo"];
$texto=$_GET["texto"];

echo "contenido variable \$saludo: $saludo <br>\n";
echo "contenido variable \$texto: ".$texto." <br>\n"
?>
</body>
</html>
```

1.2.4 Paso de Matrices a PHP.

Ya hemos visto en los casos de selección múltiple (**SELECT MÚLTIPLE**) como enviar los datos en una **matriz**. Pero nos puede interesar enviar varios datos de diferentes campos de un formulario metidos en una matriz indexada.

Ejemplo:

Form_pasoMatriz.HTML:

```
<form action="form_pasoMatriz.php" method="post">
  <fieldset>
    <legend>datos propios</legend>
    <div>nombre:
    <input type="text" name="propio[nombre]" value="" size="20"></div>
    <div>correo:
    <input type="text" name="propio[correo]" value="" size="20"></div>
  </fieldset>
  <fieldset>
    <legend>datos conyuge</legend>
    <div>nombre:
    <input type="text" name="conyuge[nombre]" value="" size="20"></div>
    <div>correo:
    <input type="text" name="conyuge[correo]" value="" size="20"></div>
  </fieldset>
  <div>
    <input type="submit" name="enviar" value="enviar">
    <input type="reset" name="borrar" value="borrar formulario">
  </div>
</form>
```

Form_pasoMatriz.php

La función *extract* permite asociar una variable a una posición del array.

El consigue enlazando *nombreDeArray_nombrePosición*. En mi caso con *\$propio_nombre*, es decir usando guión bajo

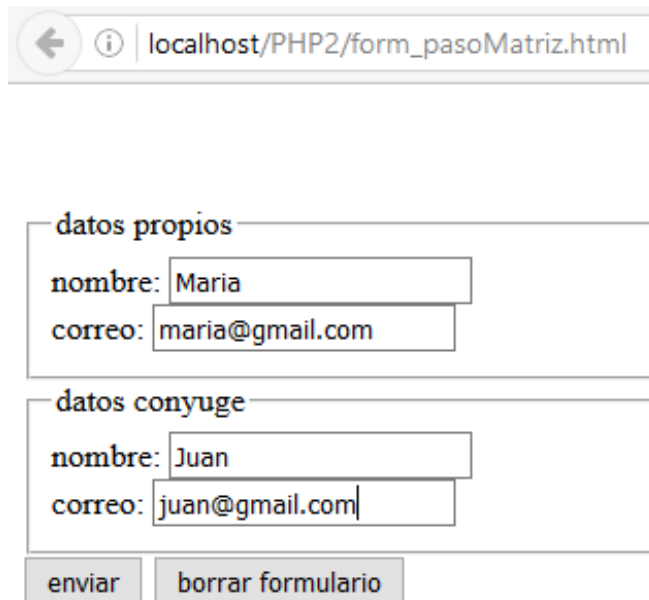
```
<?php
extract($_POST['propio'], EXTR_PREFIX_ALL, 'propio');
extract($_POST['conyuge'], EXTR_PREFIX_ALL, 'conyuge');
echo '<table>';
    echo '<tr>';
        echo '<td></td><td>nombre</td><td>correo</td>';
    echo '</tr>';
    echo '<tr>';
        echo '<td>propio</td>';
        echo '<td>'.$propio_nombre.'</td>';
        echo '<td>'.$propio_correo.'</td>';
    echo '</tr>';
    echo '<tr>';
        echo '<td>conyuge</td>';
        echo '<td>'.$conyuge_nombre.'</td>';
        echo '<td>'.$conyuge_correo.'</td>';
    echo '</tr>';
echo '</table>';
?>
```

Nota:

Ver información de función *extract()* en:

http://www.w3schools.com/php/func_array_extract.asp

Ejecución:



← ⓘ localhost/PHP2/form_pasoMatriz.html

datos propios

nombre:

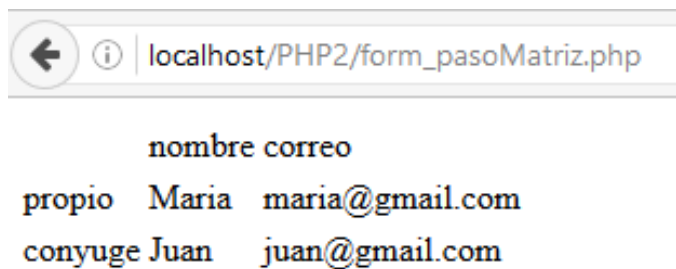
correo:

datos conyuge

nombre:

correo:

Al pulsar Enviar, se obtiene:



← ⓘ localhost/PHP2/form_pasoMatriz.php

	nombre	correo
propio	Maria	maria@gmail.com
conyuge	Juan	juan@gmail.com

1.2.5 Paso de datos por QUERY_STRING.

A veces interesará pasar datos a través de un enlace en lugar de un formulario, por ejemplo para pasar variables de JavaScript a PHP o para pasar un identificador de sesión. En esos casos seremos nosotros los que generaremos un QUERY_STRING con los variables JavaScript o variables de sesión para pasarlos a través de un mensaje HTTP-GET.

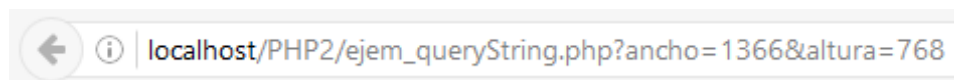
Ver ejemplos en : <http://www.devservice.es/blog/como-pasar-variables-javascript-a-php/>

Ejemplo:

ejem_QueryString.php:

```
<?php
// Como hemos pasado los datos en el QUERY_STRING usamos $_GET
if (isset($_GET['ancho'])===true && isset($_GET['altura'])===true){
    echo "El ancho de la pantalla es: ".$_GET['ancho']."<br/>";
    echo "La altura de la pantalla es: ".$_GET['altura']."<br/>";
}
else{
    echo "<script language='javascript'>\n";
    // Pasamos las variables al propio script.
    echo " location.href=\"${_SERVER['SCRIPT_NAME']}?\";
    // Concatenamos el QUERY_STRING que ya tuviera la pagina.
    echo "${_SERVER['QUERY_STRING']}";
    // Concatenamos vbles de Javascript alto y ancho de la resolución que está usando el usuario.
    echo "ancho=\" + screen.width + \"&\";
    echo "altura=\" + screen.height;\n";
    echo "</script>";
    echo '<a class="enlace" href="ejem_queryString.php?id='.$_SESSION['idSesion'].'">'.$texto_enlace.'</a>';
    exit();
}
```

Ejecución:



Ejemplo paso datos por Query String

El ancho de la pantalla es: 1366

La altura de la pantalla es: 768

1.3 PROCESAMIENTO DE FORMULARIOS EN PHP

1.3.1 Procesamiento en un solo Script. FORMULARIOS REENTRANTES

Hasta ahora se ha visto el esquema de envío de datos por formulario en que participan dos páginas, una que contiene el formulario y una otra que recibe los datos de este formulario.

Lo mismo ocurre cuando enviamos variables por una URL.

Tenemos una página que contendrá el enlace y una otra página que recibirá y tratará esos datos para mostrar unos resultados.

En el presente apartado veremos cómo se puede enviar y recibir datos de un formulario con una única página.

A este efecto se'l denomina **autollamamiento de páginas**, también se les suele llamar formularios **reentrantes**. Se suele usar este esquema para, formularios o envío de datos por la URL

La manera de funcionamiento es:

-Si no se reciben datos → Muestro el formulario o los enlaces que pasan variables.

-Si recibo datos → Entonces tengo que procesar el formulario o las variables de la URL

Ventajas:

- Disminuye el número de ficheros.
- Genera código ordenado
- Permite validar los datos del formulario en el propio formulario.

Ejemplo1:

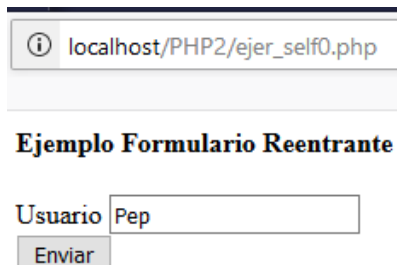
De formulario *reentrante* (utilizando *exit()*)

Ejer_self0.php:

```
<body>
<?php
if(isset($_POST['enviado']))
    if (!empty($_POST['name']))
    {
        $name = $_POST['name'];
        echo "Soy el usuario: <b> $name </b>";
        exit();
    }

?>
<h4>Ejemplo Formulario Reentrante</h4>
<form method="post" action="ejercicio_self0.php">
    <input type="text" name="name"><br>

    <input type="submit" name="enviado" value="Formulario"><br>
</form>
</body>
```

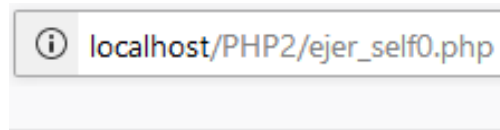
Ejecución


localhost/PHP2/ejer_self0.php

Ejemplo Formulario Reentrante

Usuario

Al pulsar Enviar veremos:



Soy el usuario: **Pep**

En el ejemplo , el primer paso es conocer si se están recibiendo o no datos por un formulario. Para esto se comprueba con una sentencia *if* si existe o no una variable `$_POST`.

Nota:

if (!\$_POST) querría decir algo como "*Si no existen datos venidos de un formulario*".

En caso de que no existan, muestro el formulario. En caso que sí que existan, recoge los datos y los imprime en la página.

Una otra forma seria usando valor de la variable interna **`$PHP_SELF`**.

`$PHP_SELF` té toda la información pertinente al servidor web donde se ejecuta el script PHP, la ruta de donde se ejecuta, el nombre del archivo que se ejecuta... los variables GET y POST que se le envían al archivo que se ejecuta.

`$_SERVER['PHP_SELF']`

Ver manuales para `$_SERVER` en:

<http://php.net/manual/en/reserved.variables.server.php>

Ejemplo2:

De formulario *reentrante* (utilizando *exit()* y *\$_SERVER['PHP_SELF']*)

Usar *\$_SERVER['PHP_SELF']* en el **action** del formulario:

```
action="<?php echo $_SERVER['PHP_SELF']; ?>"
```

Ejer_self1.php:

```
if(isset($_POST['enviado']))
    if (!empty($_POST['name']))
    {
        $name = $_POST['name'];
        echo "Soy el usuario: <b> $name </b>";
        exit();
    }

?>
<h4>Ejemplo Formulario Reentrante</h4>
<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
    <input type="text" name="name"><br>
    <input type="submit" name="enviado" value="Formulario"><br>
</form>
```

Ejemplo3:

De formulario *reentrante* (Sin usar `exit()`)

Ejer_self2.php:

```

<body>
<?php
//if (!$_POST){ // es decir, si no enviamos nada desde el formulario
    if (empty($_POST['nombre'])
        || empty($_POST['empresa']) ||
        empty($_POST['telefono']))
    {
        // nota: el teléfono no será necesario, pues le doy valor por defecto
    }
?>

<form action="ejercicio2.php" method="POST">
    Nombre: <input type="text" name="nombre" size="30"><br>
    Empresa: <input type="text" name="empresa" size="30"><br>
    Telefono: <input type="text" name="telefono" size="14" value="+34"><br>
    <input type="submit" value="Enviar">
</form>
<?php
} else
{
    echo "<br>Su nombre: " . $_POST["nombre"];
    echo "<br>Su empresa: " . $_POST["empresa"];
    echo "<br>Su Teléfono: " . $_POST["telefono"];
}
?>
</body>

```


1.3.2 Validación de formularios Desde el servidor

Buscar en internet

Como ya mencionamos, podremos validar los datos en el lado del cliente con algún tipo de script o Desde el servidor en un proceso de ida y vuelta .

Para validar los datos desde el servidor se pueden usar dos tipos de esquema:

- Mostrar el formulario más de una vez:

```

SI Se ha enviado el formulario
  SI Hay Errores
    Mostrar formulario con errores
  SINO
    Procesar formulario
  FINSI
SINO Mostrar formulario
FINSI
  
```

- Muestro el formulario solo una vez:

```

SI Se ha enviado el formulario
  Validar datos
  FINSI
SI Se ha enviado el formulario y no hay errores
  Procesar formulario
SINO
  Mostrar formulario con valores por defecto o ya enviados
  Mostrar algún aviso de error.
FINSI
  
```

Ejemplo del segundo esquema:

Suponemos que disponemos de la siguiente biblioteca de funciones de validación **valida_datos.inc**:

Ver funcion `ereg()` en :

<http://php.net/manual/es/function.ereg.php>

Ver expresiones regulares en:

<http://stackoverflow.com/questions/1653425/a-za-z-a-za-z0-9-regular-expression>

valida_datos.inc:

```
<?php
function valida_EsTexto($texto)
{
    $reg = "[A-Za-z]+[A-Za-z0-9]*$";
    return ereg($reg, $texto);
}
function valida_EsNumerico($texto)
{
    $reg = "[0-9]+$";
    return ereg($reg, $texto);
}
function valida_CamposCorrectos($datos, $tipos, &$errores)
{
    $numDatos = count($datos);
    assert($numDatos == count($tipos));
    $correctos = true;
    for ($i=0; $i < $numDatos; ++$i)
    {
        switch($tipos[$i])
        {
            case 'texto':
                $errores[$i] = !valida_EsTexto($datos[$i]);
                break;
            case 'entero':
                $errores[$i] = !valida_EsNumerico($datos[$i]);
                break;
            default: assert(false);
        }
        if ($errores[$i] === true)
            $correctos = false;
    }
    return $correctos;
}
?>
```

valida_datosServidor.php

```
html>
<head>
<title>ejemplo validacion</title>
<style>
<!--
.stynormal { font-family: arial; font-size: 10pt; }
.styerror { color: #ff0000; font-family: arial; font-size: 10pt; font-
style: italic }
-->
</style>
</head>
<body>
<?php
include_once('valida.inc');

$campos = array('nombre','nota');
$tipos = array('texto','entero');
$errores = array(false, false);
$enviadoformulario = isset($_POST['datosenviados']);

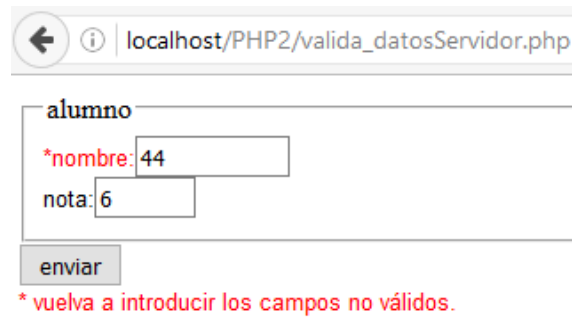
if ($enviadoformulario === true)
{
    $datos = $_POST['datos'];
    if (valida_camposcorrectos($datos, $tipos, $errores))
        $hayerroresenformulario = false;
    else
        $hayerroresenformulario = true;
}
else
{
    $datos = array("", "");
    $hayerroresenformulario = false;
}
```

```

if ($enviadoformulario === true && $hayerroresenformulario === false)
{
    $numdatos = count($datos);
    for ($i=0; $i < $numdatos; ++$i)
        echo '<div>'.$campos[$i].': '.$datos[$i].</div>';
}
else
{
    echo '<form action="'.$_SERVER['PHP_SELF'].'" method="post">';
    echo '<fieldset>';
    echo '<legend>alumno</legend>';
    echo '<div class="">';
    echo $errores[0]?'styerror">*: 'stynormal">';
    echo $campos[0].': ';
    echo '<input type="text" name="datos[]" value="'.$datos[0].'"
        size="10"></div>';
    echo '<div class="">';
    echo $errores[1]?'styerror">*: 'stynormal">';
    echo $campos[1].': ';
    echo '<input type="text" name="datos[]" value="'.$datos[1].'"
        size="5"></div>';
    echo '</fieldset>';
    echo '<input type="submit" name="datosenviados"
        value="enviar">';
    echo '</form>';
    if ($hayerroresenformulario === true)
        echo '<div class="styerror">* vuelva a introducir los
            campos no válidos.</div>';
}
?>
</body>
</html>

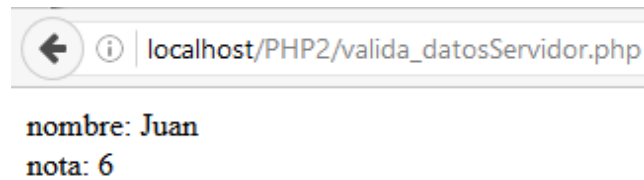
```

Ejecución con datos incorrectos:



A screenshot of a web browser window. The address bar shows 'localhost/PHP2/valida_datosServidor.php'. The page content includes a form with the label 'alumno'. Inside the form, there is a red asterisk followed by the text '*nombre:' and a text input field containing the number '44'. Below this, there is a label 'nota:' followed by a text input field containing the number '6'. At the bottom of the form is a button labeled 'enviar'. Below the form, a red error message reads '* vuelva a introducir los campos no válidos.'

Ejecución con datos correctos:



A screenshot of a web browser window. The address bar shows 'localhost/PHP2/valida_datosServidor.php'. The page content displays the result of a successful form submission: 'nombre: Juan' and 'nota: 6'.