

PHP – Interacción con el cliente

Aplicaciones Web/Sistemas Web



Juan Pavón Mestras
Dep. Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense Madrid

Material bajo licencia Creative Commons

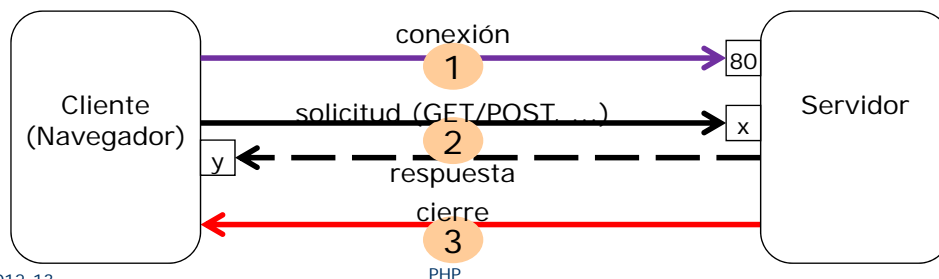


PHP - Interacción con el cliente

Formularios

Protocolo HTTP

- El navegador (cliente, *user agent*) solicita un recurso (página HTML, imagen, video, etc.) a un servidor
 - Solicitud: método que se utiliza GET, POST, PUT, HEAD, etc.
 - Campos de cabecera
 - Línea en blanco
 - Cuerpo del mensaje (texto): puede llevar parámetros del formulario
- El servidor responde enviando el recurso o con un mensaje de error
 - Línea de estado: código del estado (OK, Error) y texto asociado
 - Campos de cabecera
 - Línea en blanco
 - Cuerpo del mensaje: el recurso solicitado



Juan Pavón - UCM 2012-13

PHP

3

Paso de parámetros

- La petición del cliente puede llevar varios parámetros
 - Normalmente se obtienen de un formulario
- Cómo se pasan depende de la acción indicada en el formulario HTML en el que se recogen los datos
 - **GET:** petición de información (operación idempotente)
 - GET `consultatelefono.php?cliente=empresa1`
 - Los parámetros se pasan como pares nombre=valor
 - Se pueden pasar varios parámetros seguidos con &
 - **POST:** peticiones que cambian el estado del servidor
 - Guardar o actualizar datos
 - Enviar email
 - Ordenar datos
 - POST `modifica.php?cliente=empresa1&telefono=917892893`

Juan Pavón - UCM 2012-13

PHP

4

Escenario típico de interacción (con GET)

<http://localhost/CursoPHP/hola.html>

hola.html

```
<p>Por favor, indique su nombre:  
<form method="get" action="procesaform.php">  
Nombre:  
<input type="text" name="cliente" />  
<input type="submit" value="Enviar">  
</form>  
</p>
```

Por favor, indique su nombre:

Nombre:

procesaform.php

```
<?php  
$cliente=$_REQUEST["cliente"];  
echo "Hola $cliente";  
?>
```

localhost/CursoPHP/procesaform.php?cliente=Juan

Hola Juan

Escenario típico de interacción (con POST)

<http://localhost/CursoPHP/hola.html>

hola.html

```
<p>Por favor, indique su nombre:  
<form method="post" action="procesaform.php">  
Nombre:  
<input type="text" name="cliente" />  
<input type="submit" value="Enviar">  
</form>  
</p>
```

Por favor, indique su nombre:

Nombre:

procesaform.php

```
<?php  
$cliente=$_REQUEST["cliente"];  
echo "Hola $cliente";  
?>
```

localhost/CursoPHP/procesaform.php

Hola Juan

Formularios

- **<form>**
- Conjunto de controles que permiten al usuario interactuar
 - Generalmente para introducir datos y enviarlos al servidor web
 - El navegador envía únicamente los datos de los controles contenidos en el formulario
 - En una misma página puede haber varios formularios que envíen datos al mismo o a diferentes agentes

```
<form action="http://www.miweb.com/procesaform.php" method="post">
Escribe tu nombre:
<input type="text" name="nombre" value="" />
<br/>
<input type="submit" value="Enviar" />
</form>
```

Formulario muy sencillo

Escribe tu nombre:

POST "/procesaform.php"
nombre="valor"

Formularios

- Dentro de un formulario puede haber:
 - Cualquier elemento típico de una página web
 - Párrafos, imágenes, divisiones, listas, tablas, etc.
 - Controles de formularios
 - <input />
 - <button>
 - <select>
 - <optgroup>
 - <option>
 - <textarea>
 - Estructura de formularios
 - <fieldset>
 - <legend>
 - Información para accesibilidad
 - <label> permite mejorar la accesibilidad de los controles

Formularios

- Atributos de <form>
 - **action="URL"**: aplicación del servidor que procesará los datos remitidos (por ejemplo, un script de PHP)
 - **method**: método HTTP para enviar los datos al servidor
 - **GET**: como añadido a la dirección indicada en el atributo action
 - Limitado a 500 bytes
 - Los datos enviados se añaden al final de la URL de la página y por tanto se ven en la barra del navegador
 - Se suele usar cuando se envía información que no modifica el servidor (por ejemplo, términos para una búsqueda)
 - Si no se especifica, los navegadores suelen hacer GET
 - **POST**: en forma separada
 - Puede enviar más información
 - Permite enviar ficheros adjuntos
 - Los datos enviados no se ven en la barra del navegador
 - Se suele usar cuando se envía información que puede modificar el servidor
 - **enctype**: Tipo de codificación al enviar el formulario al servidor
 - "application/x-www-form-urlencoded" o "multipart/form-data"
 - Sólo se indica cuando se adjuntan archivos

Formularios

- <input />
 - **type** = "text | password | checkbox | radio | submit | reset | file | hidden | image | button" - Indica el tipo de control que se incluye en el formulario
 - **name** = "texto" - Nombre del control (para que el servidor pueda procesar el formulario)
 - **value** = "texto" - Valor inicial del control
 - **size** - Tamaño inicial del control (en píxeles, salvo para campos de texto y de password que se refiere al número de caracteres)
 - **maxlength** = "numero" - Máximo tamaño de texto y de password
 - **checked** = "checked" - Opción preseleccionada para los controles checkbox y radiobutton
 - **disabled** = "disabled" - El control aparece deshabilitado y su valor no se envía al servidor junto con el resto de datos
 - **readonly** = "readonly" - El contenido del control no se puede modificar
 - **src** = "url" - Para el control que permite crear botones con imágenes, indica la URL de la imagen que se emplea como botón de formulario
 - **alt** = "texto" - Descripción del control

Formularios

■ Cuadro de texto

Nombre


```
<input type="text" name="nombre" value="" />
```

Nombre

- Se enviará al servidor cuando se pulse un botón de enviar
- El nombre asignado en *name* tiene que concordar con el que se use en la aplicación en el servidor
 - No se deben utilizar caracteres problemáticos en programación (espacios en blanco, acentos y caracteres como ñ o ç)
- *value* permite establecer un valor inicial en el cuadro de texto

■ Contraseñas

Contraseña


```
<input type="password" name="contrasena" value="" />
```

Contraseña

- Igual que el cuadro de texto por el valor introducido no se ve

Formularios

■ Cuadro de texto de varias líneas

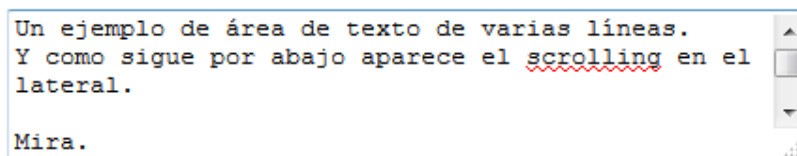
Nombre


```
<textarea name="nombre" rows="4" cols="50">
```

Contenido inicial del cuadro de texto

```
</textarea>
```

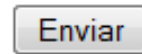
- filas: número de filas visibles (sale una barra de desplazamiento si se hay más)
- columnas: anchura en caracteres

A screenshot of a web browser showing a multi-line text area. The text area contains the text: "Un ejemplo de área de texto de varias líneas. Y como sigue por abajo aparece el scrolling en el lateral. Mira." The text "scrolling" is underlined with a red wavy line. The text area has a vertical scrollbar on the right side, indicating that the content is longer than the visible area.

Formularios

- Botón de envío de formulario

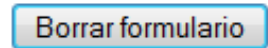
`<input type="submit" name="enviar" value="Enviar" />`



- El navegador se encarga de enviar automáticamente los datos cuando el usuario pincha el botón

- Botón de reseteo de formulario

`<input type="reset" name="borrar" value="Borrar formulario" />`

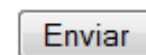


- El navegador borra toda la información introducida y muestra el formulario en su estado original

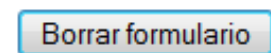
Formularios

- Botones en general: **<button>**

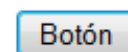
`<button type="submit">Enviar</button>`



`<button type="reset">Borrar formulario</button>`



`<button type="button">Botón</button>`



- El navegador se encarga de enviar automáticamente los datos cuando el usuario pincha el botón

Formularios

■ Casillas de verificación (*checkbox*)

Lenguajes de programación:

<input name="java" type="checkbox" value="on"/> Java

<input name="cplusplus" type="checkbox" value="on"/> C++

<input name="csharp" type="checkbox" value="on"/> C#

<input name="otros" type="checkbox" value="on"/> Otros

- value indica el tipo de casilla: on/off, yes/no, true/false

Lenguajes de programación:

☐ Java ☐ C++ ☐ C# ☐ Otros

■ *Radiobutton*

Sexo

<input type="radio" name="sexo" value="hombre" checked="checked" />

Hombre

<input type="radio" name="sexo" value="mujer" /> Mujer

Sexo

☒ Hombre

☐ Mujer

Ejercicio

■ Crear una página PHP que genere un formulario con los siguientes campos:

- Un campo de texto para preguntar el nombre
- Un campo radio button para seleccionar el sexo
- Un campo checkbox para seleccionar lenguajes de programación

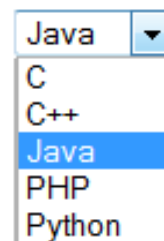
Al hacer submit se envían los datos al servidor con POST y el servidor devuelve una página que muestra los datos recopilados.

- Para probar lo que se envía al servidor, usar las herramientas de desarrollador del navegador
- También se puede probar con GET y se verán los parámetros en la URL resultante al hacer submit

Formularios

■ Listas de selección

```
<form action="">
<select name="lenguajes">
  <option value="c">C</option>
  <option value="cplusplus">C++</option>
  <option value="java" selected>Java</option>
  <option value="php">PHP</option>
  <option value="python">Python</option>
</select>
</form>
```



■ Atributos de option:

- *value* determina el valor que se envía al servidor
- *selected* permite definir la opción por defecto

Formularios

■ Agrupación de elementos

- Permite ver mejor las partes de un formulario agrupando elementos relacionados
- `<legend>` es el título que se visualiza con el grupo

```
<form action="">
<fieldset>
  <legend>Información personal:</legend>
  Nombre: <input type="text" size="50"><br>
  E-mail: <input type="text" size="50"><br>
  Ciudad: <input type="text" size="20">
</fieldset>
</form>
```

Información personal:

Nombre:	<input size="50" type="text"/>
E-mail:	<input size="50" type="text"/>
Ciudad:	<input size="20" type="text"/>

Información recibida con la solicitud del cliente

- El valor de los parámetros se guarda en **\$_REQUEST**
 - **\$_REQUEST** ["nombre-parámetro"]
 - nombre-parámetro es el que en el formulario se indica con el atributo name

<p>Nombre: <input type="text" name="nombre" /></p>

- Si se quiere depurar se puede ver toda la información recibida con `print_r($_REQUEST);`
- Se pueden usar igualmente las siguientes variables superglobales
 - **\$_GET** ["nombre-parámetro"]
 - **\$_POST** ["nombre-parámetro"]
 - Pero **\$_REQUEST** vale para ambos tipos de solicitudes

Ficheros en formularios

- Incluir un fichero
 - El atributo enctype en la etiqueta <form> del formulario tiene que ser multipart/form-data

```
<form name="fichero" action="procesa_fichero.php" method="post"
enctype="multipart/form-data">
Fichero: <input type="file" name="archivo" />
<input type="submit" value="Enviar">
</form>
```



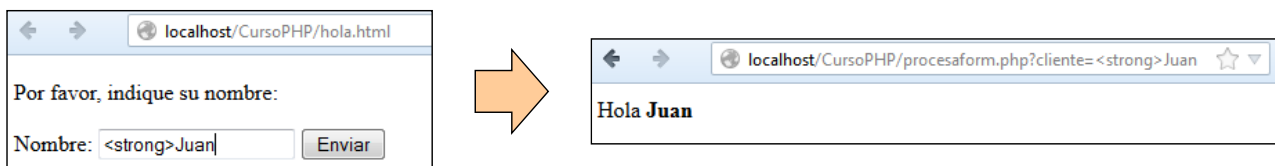
- Los ficheros recibidos se pueden acceder con **\$_FILE[]**
 - **\$_FILE['campoFile']['name']** Nombre del fichero en el cliente
 - **\$_FILE['campoFile']['type']** Tipo MIME del fichero
 - **\$_FILE['campoFile']['size']** Tamaño, en bytes, del fichero

Validación de la información recibida

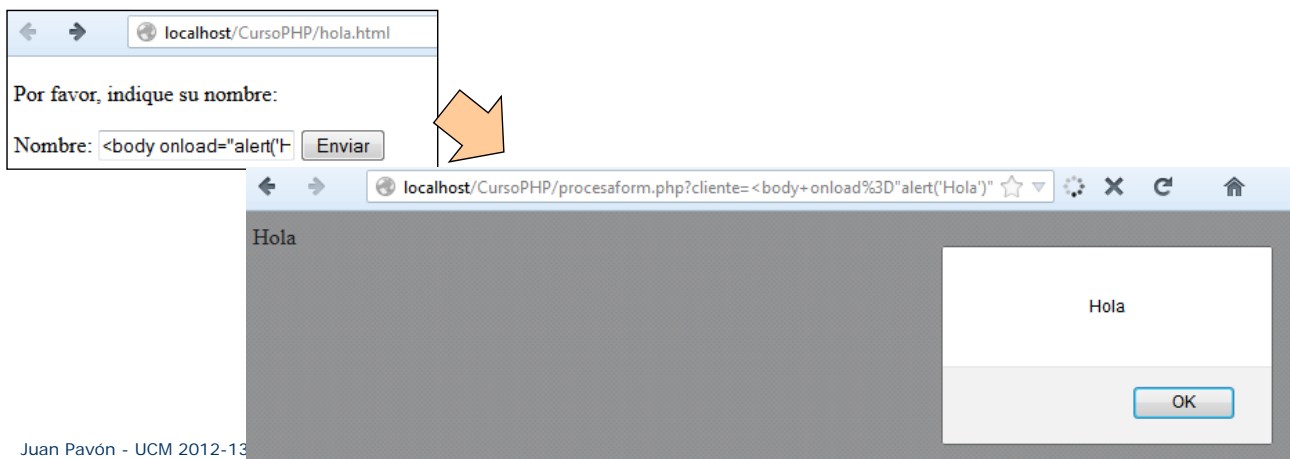
- Los campos de texto de los formularios siempre se reciben
 - Conviene comprobar que no estén vacíos
- Las casillas de verificación y los botones radio solamente están definidos en `$_REQUEST` si se han marcado en el formulario
 - Conviene comprobar que están definidos
- SIEMPRE hay que validar los datos recibidos
 - Texto correcto
 - No vacío (`strlen() > 0`)
 - Eliminar caracteres en blanco (`trim()`)
 - Cuidado con caracteres especiales
 - Números
 - Bien formados
 - Enteros: `intval()`
 - Reales: `floatval()`
 - Rango de valores
 - Dirección de correo electrónico
`Preg_match('/^[^@\s]+@([a-z0-9]+\.)+[a-z]{2,}$/i', $_POST['email'])`

Seguridad en las entradas

- Conviene comprobar que no llegue código con `< y >`



- Podría ocasionar efectos inesperados



Seguridad en las entradas

- Para evitarlo se usa una función que elimine < y >
 - **strip_tags(string)**
 - Retira las etiquetas HTML y PHP de un string
 - **htmlspecialchars(string)**
 - Convierte caracteres especiales en entidades HTML
 - & → &
 - " (comillas dobles) → "
 - ' (comilla simple) → '
 - < → <
 - > → >
- También conviene quitar los espacios al principio
 - **trim(string)**
 - Elimina los espacios en blanco iniciales y finales del string
- En resumen, se debería hacer algo así:
`$cliente=htmlspecialchars(trim(strip_tags($_REQUEST["cliente"])));`

Codificación de caracteres especiales

Carácter	Código
/	%2F
:	%3A
=	%3D
"	%22
'	%27
(espacio)	%20
?	%3F
@	%40
&	%26
\	%5C
~	%7E
	%7C

(también como +)

Funciones útiles para tratar strings

- **substr**(string, posición, [longitud])
 - Devuelve una subcadena de caracteres, a partir de la posición indicada y de longitud la especificada (o hasta el final si no se especifica)
- **strpos**(string1, string2, [posición])
 - Buscan en string1 la primera aparición de string2
 - Si se especifica, se empieza a buscar a partir de la posición indicada
- **htmlspecialchars**(string)
 - Reemplaza en el string aquellos caracteres que no son válidos en HTML y los convierte en sus equivalentes válidos (con &)
 - & → & " → " < → < > → >
- **nl2br**(string)
 - Cambia los saltos de línea '\n' por

Ejercicio

- Crear una página con un formulario que recoja información de un nuevo cliente, la valide y la almacene en la base de datos *tienda* como nuevo registro de la tabla *clientes*
 - Si todos los datos son correctos y se almacena bien en la base de datos se mostrará una página indicando que la operación se ha realizado con éxito, mostrando los campos del registro que se han guardado
 - Si hubiera campos con datos incorrectos, volver a mostrar el formulario resaltando dichos campos. Los datos que fueran correctos aparecerán en sus respectivos campos para que el usuario no tenga que volver a introducirlos

PHP - Interacción con el cliente

Cookies

Cookies

- HTTP es un protocolo SIN ESTADO
 - No se guarda información de la sesión/historia pasada
 - (Esto simplifica el protocolo)
- Uso de *cookies*
 - Un *cookie* es un *string* que se pasa en una cabecera HTTP y que el navegador puede guardar en un pequeño fichero de texto
 - En archivos temporales del navegador correspondiente
 - El *cookie* se reenvía luego al servidor HTTP con cada petición del cliente a ese servidor
 - Los *cookies* **no** pueden capturar información del cliente
 - Sólo recuerdan información proporcionada por el usuario al servidor (es el servidor quien los crea)
 - Usos
 - Guardar las preferencias del usuario
 - Reconocimiento de usuarios
 - El cookie puede guardar un identificador que permite al servidor acceder a todos los datos almacenados en su base de datos
 - Gestión de sesiones