Desarrollo web en entorno servidor

UT 1.1 – Arquitectura y herramientas web
6 – PHP – Procesando la petición y la respuesta HTTP

IES Clara del Rey – Madrid

PHP – Petición HTTP

Existen arrays asociativos superglobales (accesibles en cualquier punto de una aplicación) para acceder a los datos de una petición HTTP.

- $_GET – Parámetros enviados en la URL. Tanto en peticiones GET como POST. Son los que aparecen en la query string, separados por &.
- $_POST – Datos enviados en una petición POST, en el cuerpo del mensaje HTTP.
- $_FILES – Ficheros enviados en una petición post con mime "multipart-form-data" que proviene de un formulario con inputs tipo file.
- $_SERVER – Información del servidor y de la petición HTTP.

Además, se dispone de la función "getAllHeaders()", que devuelve un array con la información de las cabeceras HTTP recibidas.

PHP – $_GET

Supongamos la siguiente petición HTTP:

/prueba.php?nombre=Jose&apellido=Lopez&cumple=08-13

Podríamos acceder a los parámetros en la URL de la siguiente forma:

- Nombre: *$_GET['nombre']*
- Apellido: *$_GET['apellido']*
- Cumpleaños: *$_GET['cumple']*

En $_GET (y también en $_POST) los datos recibidos son siempre String.

Aunque PHP sea de tipado dinámico, los datos recibidos siempre se interpretan como String, y si es necesario convertirls a otro tipo se debe hacer manualmente.

PHP – $_GET – Múltiples valores

Supongamos esta petición:

/prueba.php?opcion=A&opcion=B&opcion=G&opcion=X

Puede generarse, por ejemplo, en un formulario en el que el usuario ha elegido cuatro opciones de un conjunto de checkboxes, todos con el mismo nombre, en este caso "opcion".

Si accedemos al valor recibido, y ejecutamos un var_dump:

```
var_dump($_GET['opcion']);
```

El resultado será:

```
string(1) "X"
```

¿Dónde han ido a parar los otros tres valores?

PHP – $_GET – Múltiples valores

Se usa un nombre de parámetro especial para procesarlo como array.

Si cambiamos la petición original:

/prueba.php?opcion=A&opcion=B&opcion=G&opcion=X

Por esta:

/prueba.php?opcion[]=A&opcion[]=B&opcion[]=G&opcion[]=X

Estamos indicando a PHP, usando los corchetes, que hay múltiples valores, y el resultado de var_dump lo refleja:

array(4) { [0]=> string(1) "A" [1]=> string(1) "B"
[2]=> string(1) "G" [3]=> string(1) "X" }

Para acceder al parámetro no se tienen que escribir los corchetes.

PHP – $_GET – Múltiples valores

Si en la URL tenemos únicamente un valor:

/prueba.php?opcion[]=A

El resultado seguirá siendo un array, con un único elemento.

array(1) { [0]=> string(1) }

Si no hay nada en la URL:

/prueba.php

Al intentar acceder al dato se producirá un warning:

Warning: Undefined array key "opcion" in ...

Y el valor devuelto por var_dump será NULL

PHP – $_GET – Validando entrada

Para evitar warnings y posteriores errores al procesar datos de entrada, podemos usar algunas funciones de PHP:

- isset($_GET['nombre_parametro']) – Devuelve true si está definido.
- array_key_exists('nombre-parametro', $_GET) – Devuelve true si en el array asociativo hay un elemento con la clave especificada.
- is_array($_GET['nombre-parametro']) – Devuelve true si el parámetro recibido es un array

Esta validación se debe aplicar también en datos recibidos en $_POST, en datos recibidos en $_SERVER o ficheros recibidos en $_FILES.

También en cabeceras recibidas y leídas por getAllHeaders.

PHP – $_POST

Supongamos este formulario HTML:

```
<form method="post" action="form-process.php">
	<p><label>Nombre y apellidos<input type="text" name="nombre"></label></p>
	<p><label>Edad <input type="text" name="edad"></label></p>
	<fieldset><legend>Aficiones</legend>
		<ul>
			<li><label><input type="checkbox" name="aficion[]"
					value="L">Leer</label></li>
			<li><label><input type="checkbox" name="aficion[]"
					value="D">Deporte</label></li>
			<li><label><input type="checkbox" name="aficion[]"
					value="B">Bricolaje</label></li>
		</ul>
	</fieldset>
	<p><input type="submit" value="Enviar"></p>
</form>
```

PHP – $_POST

El envío al servidor genera una petición HTTP, que podemos inspeccionar en las herramientas de desarrollador de Chrome o Edge, y vemos algunas cosas:

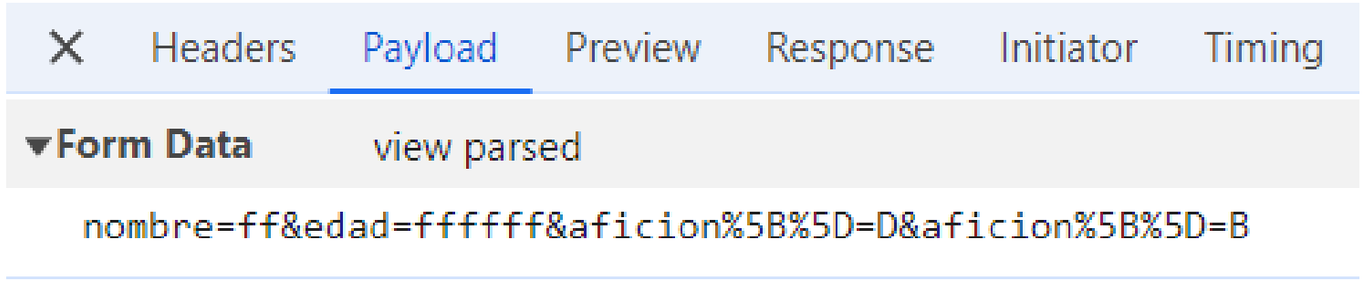
- URL solicitada, y método:

Request URL:	http://localhost:8000/form-process.php
Request Method:	POST

- Cabeceras de la petición:

Content-Length:	21
Content-Type:	application/x-www-form-urlencoded

- Cuerpo de la petición (%5D y %5B son [y]):



PHP – $_POST

Para acceder a los datos recibidos, se usa $_POST, de la misma forma que se usa $_GET para los valores recibidos en la Query String de la URL.

El comportamiento de POST respecto a parámetros que no se envían es exactamente igual, por lo que habrá que comprobar con isset o con otras funciones si se han recibido valores.

En el formulario de ejemplo, siempre se recibirá un valor para nombre y edad. Si no se escribe nada en los campos de formulario, llegará en blanco, como una cadena vacía.

Pero si no se marca ninguna afición no se recibirá el parámetro en absoluto, no se recibe nada (podríamos pensar que se recibe array vacío, pero no es así).

PHP – $_FILES

Recordemos que, en general, para enviar ficheros, se necesita un formulario HTML con los siguientes requisitos:

- Debe realizar el envío de la petición con método POST.
- Su atributo enctype debe ser multipart/form-data.
- Deben usarse campos input de tipo "file" para los ficheros.
- Debe tener un botón para poder enviar el formulario.

PHP – $_FILES

En el siguiente formulario:

```
<form action="form-files-process.php"
      method="post" enctype="multipart/form-data">
      <p><label>Nombre: <input type="text" name="nombre"></label></p>
      <p><label>Fichero:<input type="file" name="documento"></label></p>
      <p><input type="submit" value="Enviar"></p>
</form>
```

Si rellenamos el campo "nombre" con "Antonio", y seleccionamos un fichero "datos.txt" en el campo, file, al pulsar "Enviar", se realiza una petición al servidor, al fichero "form-files-process.php".

En la petición se incluirán tanto los campos "normales" como el contenido del fichero "datos.txt" con alguna información adicional.

PHP – $_FILES

Inspeccionando la petición en las herramientas de desarrollador...

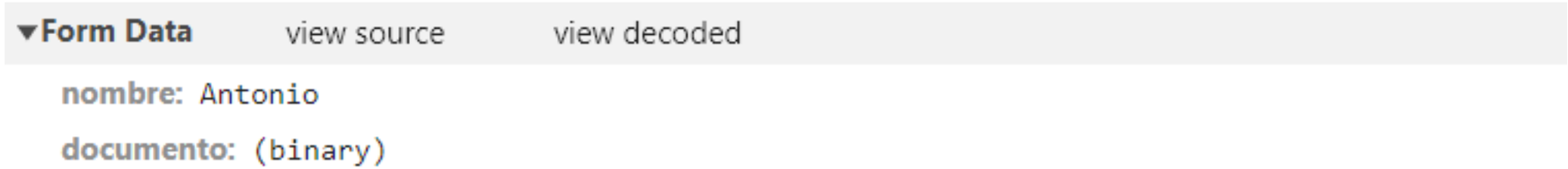
Datos generales de la petición: método POST y path de la petición:

Request URL:	http://localhost:8000/form-files-process.php
Request Method:	POST

Tamaño de la petición y codificación (entre otras cabeceras):

Content-Length:	370
Content-Type:	multipart/form-data; boundary=----WebKitFormBoundary98SVRGRXRpkKyaL6

Datos de la petición:



El documento aparece como binario porque se codifica, aunque sea un fichero de texto.

PHP – $_FILES

Cada fichero subido genera un elemento en el array $_FILES, a la que se puede acceder usando el nombre del campo HTML.

En el ejemplo, accederíamos a los datos del fichero subido con:

```
$datosFichero = $_FILES['documento'];
```

Los datos son otro array asociativo con los siguientes elementos:

- name: nombre original, el nombre en la máquina del cliente.
- type: tipo MIME del fichero, si el navegador envió esta información.
- size: tamaño en bytes del fichero.
- tmp_name: ruta temporal, en el servidor, del fichero subido.
- error: resultado de la operación de subida del fichero.

PHP – $_FILES – Códigos de error

- UPLOAD_ERR_OK: 0 – Sin errores.
- UPLOAD_ERR_INI_SIZE: 1 – El tamaño excede el límite establecido por la directive upload_max_filesize en php.ini.
- UPLOAD_ERR_FORM_SIZE: 2 – El tamaño excede el límite establecido (en bytes) por un campo "MAX_FILE_SIZE", oculto en el formulario.
- UPLOAD_ERR_PARTIAL: 3 – El fichero se subió incompleto.
- UPLOAD_ERR_NO_FILE: 4 – No se subió el fichero.
- UPLOAD_ERR_NO_TMP_DIR: 6 – No hay carpeta temporal.
- UPLOAD_ERR_CANT_WRITE: 7 – No se pudo escribir el fichero.
- UPLOAD_ERR_EXTENSION: 8 – Una extensión de PHP impidió la subida.

PHP – $_FILES – Funciones interesantes

Algunas funciones de PHP pueden ayudarnos a procesar los ficheros que se hayan subido en una petición POST:

- move_uploaded_file – Mueve un archivo subido a una nueva ubicación. Hace comprobaciones para asegurarse de que el fichero que va a moverse es realmente un fichero subido por un usuario. Para evitar acceso a ficheros del sistema.
- is_uploaded_file – Básicamente, la comprobación que realiza move_uploaded_file, pero aislada, sin mover el fichero.
- pathinfo – Devuelve información de la ruta de un fichero.
- basename – Devuelve la parte del nombre de una ruta de fichero.

PHP – $_SERVER

Array asociativo superglobal que contiene información sobre cabeceras HTTP, rutas, y ubicaciones de script.

Para acceder a las cabeceras HTTP, se utilizan las claves que comienzan con HTTP_, seguidas del nombre de la cabecera en mayúsculas y con guiones bajos en lugar de guiones normales.

Por ejemplo, para obtener la cabecera "user-agent", se usaría:

$_SERVER['HTTP_USER_AGENT']

Una cabecera puede no existir. Se debe comprobar con isset.

Algunas no empiezan por HTTP, como content-type y content-length.

Referencia: https://www.php.net/manual/es/reserved.variables.server.php

PHP – Función getallheaders

Función que devuelve un array asociativo con todas las cabeceras HTTP de la petición actual.

Esta función solo funciona en entornos Apache u otros que cumplan con la misma especificación. En concreto con los que cumplan FastCGI, CLI, y FPM (FastCGI Process Manager).

Esto quiere decir que estará disponible en la mayoría de los entornos de ejecución, pero puede que no sea así.

Si no se sabe el entorno de ejecución final, quizá sea más lógico usar $_SERVER para acceder a las cabeceras de la petición.

PHP – Respuesta HTTP

Una respuesta HTTP típica está formada por:

- Línea de estado – Indica el resultado de la operación con un código
- Cabeceras HTTP – Dan información adicional sobre la respuesta.
- Cuerpo de la respuesta – Contenido que se envía al cliente. Puede ser HTML, XML, JSON, datos binarios, etc.

Desde PHP se puede manipular estos elementos con:

- Estado: Función http_response_code
- Cabeceras: Función header
- Cuerpo de la respuesta: instrucciones o métodos de salida como echo, print, printf, var_dump, etc.

PHP – Establecer código de estado

Los códigos de estado HTTP informan al cliente sobre el resultado de la solicitud. PHP ofrece varias formas de establecer estos códigos.

- Usando http_response_code(). Esta función permite establecer o recuperar el código de estado actual:

```
http_response_code(404);          // Establece 404 (no encontrado)
$codigo = http_response_code();   // Recupera el estado actual
```

- Usando la Función header(). Enviando una línea completa con estado.

```
header("HTTP/1.1 500 Internal Server Error");
```

Hay que llamar a header() antes de enviar cualquier salida al navegador (como echo o print). Si no, Se produce un error de encabezados ya enviados.

PHP – Enviar cabeceras HTTP

Las cabeceras HTTP añaden información a una respuesta.

Para establecer una cabecera HTTP se usa la función header().

```
header("Content-Type: text/html; charset=UTF-8"); // Tipo de contenido
```

Por ejemplo, se puede usar header para redirigir la salida a otra página:

```
header("Location: http://www.example.com/", true, 302);
```

Los parámetros adicionales están indicando que la cabecera reemplaza el anterior valor de la cabecera (Location), y el segundo indica que es una redirección temporal (302 o 308). Una definitiva sería 301 o 307.

Como se ha dicho, las llamadas a header deben hacerse ANTES de cualquier salida de contenido. Esto es, código HTML, llamadas a echo, o a funciones como printf o var_dump.

PHP – Cabeceras para descarga de ficheros

Una llamada a una página PHP no tiene por qué devolver siempre HTML.

En ocasiones nos interesa devolver un fichero, por ejemplo, un PDF. Para que el navegador detecte el fichero como PDF y fuerce su descarga, hay que fijar ciertas cabeceras:

```
header('Content-Description: Informe de estado');     // Descripción
header('Content-Type: application/pdf');          // Tipo de contenido
// Indicar que es un adjunto y nombre del fichero a descargar
header('Content-Disposition: attachment; filename="datos.pdf"');
// TODO: Escribir el fichero en la salida
```

Esto fuerza a que el navegador pregunte al usuario dónde descargar el fichero, aunque el mismo navegador tenga soporte para visualizar el fichero.