Prácticas

Estilo: Normal ▾ **Tamaño letra:** Normal ▾ **Interlineado:** 1.2 ▾

Desarrollo de Aplicaciones Web

Práctica 11: PHP 5 (sistema de ficheros)

1. Objetivos

- Aprender a acceder al sistema de ficheros desde PHP.
- Aprender a subir y almacenar un fichero con PHP.

2. Recursos

¿Cómo se accede al sistema de ficheros desde PHP? ¿Cómo se crea un fichero? ¿Cómo se lee un fichero? ¿Cómo se crea un directorio?

- **PHP Filesystem** (1) : documentación oficial de las funciones de manejo del sistema de ficheros en PHP.
- **PHP File Handling** (2) : explicación en W3Schools de las funciones de manejo de ficheros en PHP.
- **PHP Filesystem Functions** (3) : referencia en W3Schools de las funciones de manejo del sistema de ficheros en PHP.

¿Cómo se maneja el envío de ficheros con PHP?

- **Form-based File Upload in HTML** (4) : definición oficial del protocolo de envío de ficheros desde un navegador web al servidor.
- **Manejo de envío de archivos** (5) : documentación oficial de PHP que explica cómo manejar en el servidor web el envío de ficheros desde un navegador web.
- **PHP File Upload** (6) : explicación en W3Schools de la subida de ficheros en PHP.

3. ¿Qué tengo que hacer?

En esta práctica tienes que incluir en la página principal un apartado llamado “foto escogida” (“foto favorita” o “foto del día” o como lo quieras llamar) donde se muestren, de entre todas las fotografías existentes en los álbumes, algunas fotografías escogidas por críticos de fotografía. Las fotografías se escogen a mano y se gestionan mediante un único fichero de texto en el que se almacena una referencia a la fotografía escogida, junto con el nombre de la persona (no son usuarios de la aplicación) que ha seleccionado cada fotografía y un comentario que explica las razones de dicha selección. El formato del fichero tiene que ser propio, no debes usar un formato estándar como JSON.

El fichero con las fotografías escogidas se edita directamente a mano, no se tiene que realizar una interfaz para su gestión desde la aplicación web. Cada vez que se cargue la página principal, se tiene que elegir una única fotografía, seleccionada de entre todas las fotografías del fichero de forma aleatoria: se tiene que mostrar la fotografía, la información disponible sobre ella (que se debe obtener de la base de datos), el nombre de la persona que la ha seleccionado y su comentario.

Además, en la página principal también tienes que incluir un apartado llamado “consejo fotográfico” donde se muestren consejos para tomar fotografías de calidad. Los consejos se almacenan en un fichero con formato JSON. Cada consejo pertenece a una categoría, tiene una dificultad y una descripción. Por ejemplo:

- Categoría: Material. Dificultad: Fácil. Consejo: Tener un trípode puede aportarte un montón de ventajas, tanto para las fotografías de interior como para las de exterior.
- Categoría: Ajustes. Dificultad: Media. Consejo: Experimenta con los ajustes de la cámara, lee el manual de instrucciones para descifrar todos esos pequeños símbolos que tiene tu cámara.
- Categoría: Ajustes. Dificultad: Alta. Consejo: La profundidad de campo se controla con los tres parámetros principales de una fotografía: diafragma, distancia al sujeto y distancia focal.

El fichero con los consejos también se edita directamente a mano, no se tiene que realizar una interfaz para su gestión desde la aplicación web. Cada vez que se cargue la página principal, se tiene que elegir un único consejo, seleccionado de entre todas las consejos de forma aleatoria.

Además, también tienes que manejar la subida de una fotografía de un usuario cuando se registra en la "página con el formulario de registro como nuevo usuario". La fotografía subida la debes mostrar en la respuesta de la "página con el formulario de registro como nuevo usuario", para que el usuario verifique que ha seleccionado la fotografía correcta y se ha subido sin problemas.

También tienes que permitir que la fotografía subida se pueda modificar o eliminar desde la opción "Modificar mis datos" del menú privado del usuario. **Importante:** todas las fotografías del perfil de los usuarios se tienen que almacenar en un mismo directorio, así que tienes que emplear un sistema de nombrado de ficheros que evite problemas cuando dos usuarios suban dos ficheros con el mismo nombre al servidor.

Por último, también tienes que manejar la subida de una fotografía desde la página "añadir foto a álbum" del menú privado del usuario. Se tiene que emplear alguna estrategia para evitar colisiones de nombres de ficheros cuando dos usuarios suban dos fotografías con el mismo nombre de fichero al servidor o cuando un mismo usuario suba dos fotografías con el mismo nombre de fichero a diferentes álbumes o incluso al mismo álbum. En la página "Ver álbum" también debes añadir un enlace a la página "añadir foto a álbum" y debe aparecer seleccionado por defecto el álbum desde el que se seleccionó el enlace "añadir foto a álbum". Si quieres, también puedes añadir un enlace a la página "añadir foto a álbum" desde otras partes de la aplicación (por ejemplo, puede existir una opción "subir foto" visible en todo momento en el menú principal de la aplicación).

4. ¿Cómo lo hago?

4.1. Sistema de ficheros

PHP proporciona un gran conjunto de funciones para acceder al sistema de ficheros y manejar los ficheros. La mayoría de las funciones permiten trabajar con ficheros que no están en local, sino que son accesibles a través de Internet mediante una URL.

Las principales funciones para trabajar con ficheros a nivel del sistema de ficheros son:

- `copy(origen, destino)`: realiza una copia del fichero `origen` a `destino`.
- `filemtime(fichero)`: devuelve la fecha y hora de la última modificación de un fichero.
- `filesize(fichero)`: devuelve el tamaño de un fichero.
- `is_file(fichero)`: devuelve `true` si es un fichero normal y `false` en caso contrario.
- `rename(nombreViejo, nombreNuevo)`: renombra o mueve de sitio un fichero.
- `unlink(fichero)`: elimina un fichero.

Las principales funciones para trabajar con directorios son:

- `chdir(directorio)`: cambia de directorio.
- `closedir(id_directorio)`: cierra un directorio.
- `is_dir(directorio)`: devuelve `true` si es un directorio y `false` en caso contrario.
- `mkdir(directorio)`: crea un directorio nuevo.

- `opendir(directorio)`: abre un directorio para su posterior procesamiento con las funciones de manejo de directorios `closedir()`, `readdir()` y `rewinddir()`; devuelve un identificador de recurso.
- `rename(nombreViejo, nombreNuevo)`: renombra o mueve de sitio un directorio.
- `readdir(id_directorio)`: devuelve el nombre del siguiente fichero en un directorio.
- `rewinddir(id_directorio)`: sitúa el manejador de directorio al principio del directorio.
- `rmdir(directorio)`: elimina un directorio.

El siguiente código muestra en una página web el contenido de un directorio, tal como lo hacen muchos servidores web. Para cada fichero se muestra su nombre, la fecha de la última modificación y el tamaño que ocupa. En la Figura 1 se muestra un ejemplo de la página que genera.

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Prueba de lectura de directorio</title>
</head>
<body>
<?php
 $nomdir = './';
 echo "<h1>Contenido de $nomdir</h1>\n";
 $dir = opendir($nomdir);

 echo "<pre>\n";
 echo "<b>";
 echo str_pad("Nombre", 30);
 echo str_pad("Fecha ult. mod.", 20);
 echo str_pad("Tamaño", 10);
 echo "</b></pre>\n";
 echo "<hr /><pre>\n";

 while(($fichero = readdir($dir)) != FALSE)
 {
   echo str_pad($fichero, 30);
   echo str_pad(date("d/m/Y H:i" , filemtime($nomdir . $fichero)), 20);
   if(is_dir($nomdir . $fichero))
   {
    echo "-";
   }
   else
   {
    echo str_pad(filesize($nomdir . $fichero), 10);
   }
   echo "<br />\n";
 }
 closedir($dir);

 echo "</pre><hr />\n";
?>
</body>
</html>
```

En el ejemplo anterior se emplea la función `str_pad(cadena, longitud)` para generar cadenas con una longitud definida que se rellenan con espacios en blanco. Además, la función `date(formato, tiempo)` se emplea para aplicar un formato al valor devuelto por la función `filemtime(fichero)`.

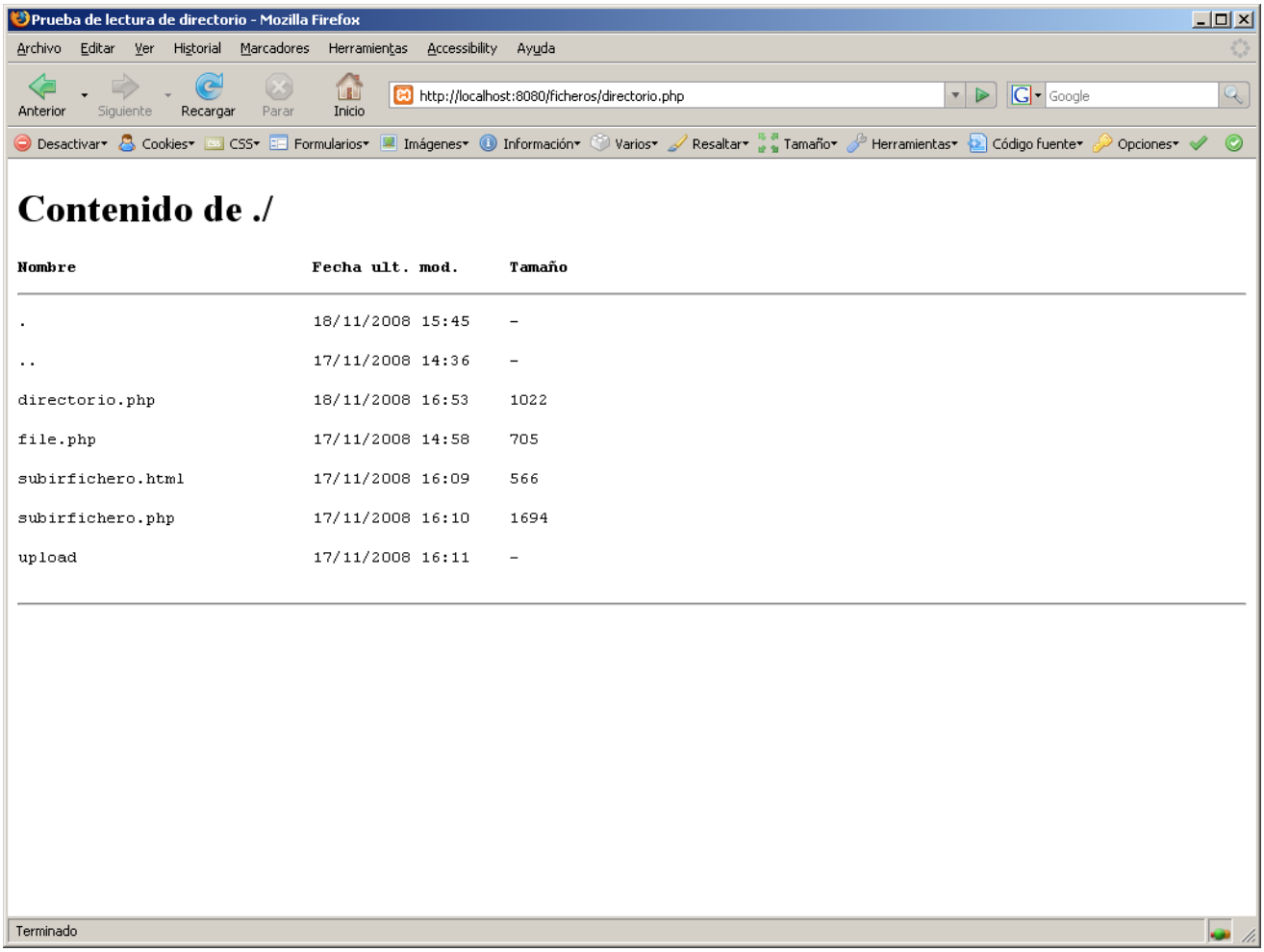

Figura 1: Visualización del contenido de un directorio

Las principales funciones para manejar los ficheros son:

- `fclose(id_fichero)`: cierra el fichero.
- `feof(id_fichero)`: devuelve `true` si se ha llegado al final del fichero y `false` en caso contrario.
- `fgetc(id_fichero)`: lee un carácter del fichero.
- `fgets(id_fichero)`: lee una línea del fichero.
- `file(nombre)`: lee todo el contenido de un fichero y lo almacena línea a línea en un array.
- `fopen(nombre, modo)`: abre un fichero según el modo indicado (lectura, escritura o lectura/escritura), devuelve un identificador de recurso que se emplea en otras funciones.
- `fscanf(id_fichero, formato, variable1, ...)`: lee datos de un fichero según el formato indicado.
- `fwrite(id_fichero, cadena)`: escribe la cadena en el fichero asociado al identificador de recurso.

El siguiente código muestra como leer todo el contenido de un fichero y visualizarlo línea a línea, indicando en cada línea el número de línea que es. En la Figura 2 se muestra como se visualiza el propio fichero.

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Prueba de file</title>
</head>
<body>
<?php
 if(($fichero = @file("file.php")) == false)
 {
```

```
    echo "No se ha podido abrir el fichero";
  }
  else
  {
    echo "<pre>\n";
    foreach($fichero as $numLinea => $linea)
    {
     echo "Línea #<b>" . sprintf("%03d", $numLinea) . "</b> : ";
     echo htmlspecialchars($linea);
    }
    echo "</pre>\n";
  }
?>
</body>
</html>
```

En el ejemplo anterior, se emplea el operador `@` para evitar que se muestren en la página web los mensajes de error que puede producir la función `file(nombre)`, como por ejemplo al no existir el fichero indicado (7). Además, se emplea la función `sprintf(formato, variable1, ...)` para generar una cadena con un formato establecido; en este ejemplo se emplea para generar una cadena de tres dígitos rellenada con ceros. Por último, se emplea la función `htmlspecialchars(cadena)` para convertir los caracteres especiales, como <, > y & a entidades de HTML: `<`, `>` y `&`.

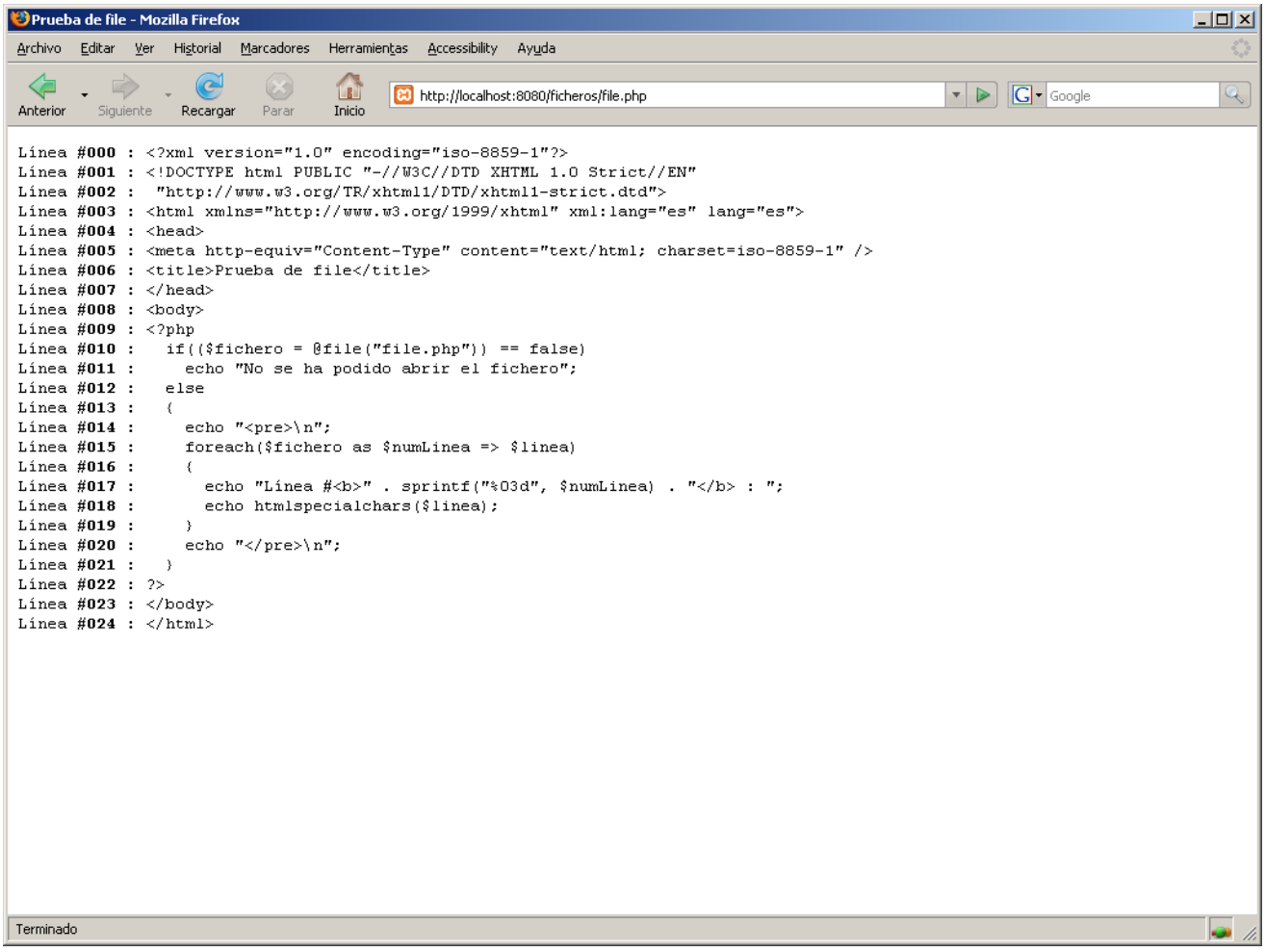

Figura 2: Lectura de un fichero con la función file

4.2. Manejo de envío de ficheros

Para manejar el envío de ficheros a través de una página web desde el cliente al servidor web se emplea la variable predefinida superglobal `$_FILES`. Esta variable es un array multidimensional donde cada posición representa un fichero que se ha enviado y que contiene la siguiente información:

- `$_FILES['fichero']['name']`: nombre original del fichero en el cliente.
- `$_FILES['fichero']['tmp_name']`: nombre del fichero temporal que se genera en el servidor para guardar el fichero subido.
- `$_FILES['fichero']['size']`: tamaño en bytes del fichero subido.
- `$_FILES['fichero']['type']`: tipo MIME (`image/gif`, `image/jpg`, `application/pdf`, etc.), siempre que el navegador lo proporcione.
- `$_FILES['fichero']['error']`: código de error asociado con la subida del fichero.

El manejo del envío de ficheros se puede configurar en el fichero `php.ini` con las directivas `file_uploads`, `upload_max_filesize`, `upload_tmp_dir`, `post_max_size` y `max_input_time`.

El siguiente ejemplo muestra cómo manejar el envío de ficheros. El primer fragmento de código es la página web que muestra un formulario que permite al usuario enviar un fichero. Para que se pueda realizar el envío se tiene que cumplir que:

- El formulario se envíe mediante el método `post`.
- Se indique como codificación de los datos `enctype="multipart/form-data"`, necesario para poder enviar datos binarios como los contenidos en un fichero.

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Prueba de subir fichero</title>
</head>
<body>
<form action="subirfichero.php" method="post" enctype="multipart/form-data">
Fichero: <input type="file" name="fichero" />
<br />
<input type="submit" value="Enviar" />
</form>
</body>
</html>
```

El segundo fragmento de código es la página PHP que recibe el fichero enviado en el servidor. En este código se emplea la función `file_exists(nombre)` para comprobar si ya existe el fichero que se intenta subir y la función `move_uploaded_file(origen, destino)` para mover el fichero temporal de la subida a su destino definitivo.

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Prueba de subir fichero</title>
</head>
<body>
<p>
<?php
 $msgError = array(0 => "No hay error, el fichero se subió con éxito",
              1 => "El tamaño del fichero supera la directiva
                  upload_max_filesize el php.ini",
              2 => "El tamaño del fichero supera la directiva
                  MAX_FILE_SIZE especificada en el formulario HTML",
              3 => "El fichero fue parcialmente subido",
              4 => "No se ha subido un fichero",
              6 => "No existe un directorio temporal",
```

```
                7 => "Fallo al escribir el fichero al disco",
                8 => "La subida del fichero fue detenida por la extensión");

 if($_FILES["fichero"]["error"] > 0)
 {
   echo "Error: " . $msgError[$_FILES["fichero"]["error"]] . "<br />";
 }
 else
 {
   echo "Nombre original: " . $_FILES["fichero"]["name"] . "<br />";
   echo "Tipo: " . $_FILES["fichero"]["type"] . "<br />";
   echo "Tamaño: " . ceil($_FILES["fichero"]["size"] / 1024) . " Kb<br />";
   echo "Nombre temporal: " . $_FILES["fichero"]["tmp_name"] . "<br />";

   if(file_exists("upload/" . $_FILES["fichero"]["name"]))
   {
    echo $_FILES["fichero"]["name"] . " ya existe";
   }
   else
   {
    move_uploaded_file($_FILES["fichero"]["tmp_name"],
                  "upload/" . $_FILES["fichero"]["name"]);
    echo "Almacenado en: " . "upload/" . $_FILES["fichero"]["name"];
   }
 }
?>
</p>
</body>
</html>
```

4.3. Generación de números aleatorios

La mayoría de los mecanismos de generación de números aleatorios que se utilizan en los sistemas informáticos son en realidad procesos pseudo-aleatorios, ya que los números se generan a partir de una fórmula y, por tanto, una secuencia de números pseudo-aleatorios se puede reproducir si se conoce el primer número y la fórmula.

En PHP, las principales funciones para generar números pseudo-aleatorios son:

- `rand(min, max)`: devuelve un número pseudo-aleatorio entre `min` y `max`; si no se indican `min` y `max`, se genera un número entre 0 y `getrandmax()`.
- `mt_rand(min, max)`: similar a `rand()`, pero es un generador mejorado y más rápido.
- `getrandmax()`: devuelve el máximo número pseudo-aleatorio que se puede generar con `rand()`.
- `mt_getrandmax()`: similar a `getrandmax()`.
- `srand()`: fija la semilla de inicio para `rand()`; desde PHP 4.2 no es necesario su uso ya que se hace de forma automática.
- `mt_srand()`: similar a `srand()`.

Por ejemplo, el siguiente código genera 10 números pseudo-aleatorios:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Prueba de números aleatorios</title>
</head>
<body>
<p>
<?php
 for($i = 1; $i <= 10; $i++)
 {
   echo rand() . "<br />";
```

```
 }
?>
</p>
</body>
</html>
```

En PHP también existen otras funciones que usan internamente la generación de números aleatorios y que te pueden ayudar en ciertas situaciones:

- `array_rand(array)`: selecciona aleatoriamente una o más entradas de un array.
- `shuffle(array)`: mezcla un array.
- `str_shuffle(array)`: reordena aleatoriamente una cadena.

4.4. JSON

JSON es el acrónimo en inglés de "JavaScript Object Notation", que en español, significa "Notación de Objetos en JavaScript". JSON es un formato textual que puede ser procesado por el ordenador pero que a su vez es sencillo de entender por las personas. Además, es un formato abierto, por lo que su uso se ha generalizado rápidamente.

JSON se emplea principalmente para el intercambio de datos, aunque también se puede emplear para otras cosas, como por ejemplo, almacenar información en un fichero. Se basa en un subconjunto de la notación literal de objetos en JavaScript, pero su popularidad ha hecho que se convierta en un formato independiente.

JSON nació a principios del año 2001, pero tardó unos años en estandarizarse. Su definición oficial la podemos encontrar en varias especificaciones. Una primera especificación fue el Request for Comments 4627 (8) de julio de 2006, que fue sustituido por varias versiones posteriores, hasta llegar a la última, el Request for Comments 8259 (9) de diciembre de 2017, que tiene el carácter de estándar de Internet.

En este documento podemos encontrar dos ejemplos sencillos de JSON. El primero muestra cómo se emplea JSON para representar un objeto que contiene el miembro Image que también es un objeto que a su vez contiene el miembro Thumbnail que también es un objeto y cuyo miembro IDs es un array de números:

```
{
	"Image": {
		"Width": 800,
		"Height": 600,
		"Title": "View from 15th Floor",
		"Thumbnail": {
			"Url":  "http://www.example.com/image/481989943",
			"Height": 125,
			"Width": 100
		},
		"Animated" : false,
		"IDs": [116, 943, 234, 38793]
	}
}
```

A partir de este ejemplo nos podemos dar cuenta de que los objetos se representan encerrados entre llaves y los arrays encerrados entre corchetes. También podemos ver que JSON emplea la recursividad: un valor JSON puede formar parte de otro valor JSON.

El segundo ejemplo es un array, por eso está encerrado entre corchetes, que contiene dos objetos encerrados entre llaves:

```
[
	{
		"precision": "zip",
		"Latitude": 37.7668,
		"Longitude": -122.3959,
		"Address": "",
		"City":    "SAN FRANCISCO",
		"State":   "CA",
		"Zip":     "94107",
		"Country": "US"
	},
	{
		"precision": "zip",
		"Latitude": 37.371991,
		"Longitude": -122.026020,
		"Address": "",
		"City":    "SUNNYVALE",
		"State":   "CA",
		"Zip":     "94085",
		"Country": "US"
	}
]
```

JSON también está estandarizado en la especificación ECMA-404 (10) de diciembre 2017. En este documento podemos encontrar unas figuras que representan la sintaxis de JSON. En este documento, los diagramas sintácticos emplean una notación llamada ?diagrama de ferrocarril?.

En la Figura 3 se explica que un valor JSON puede ser un objeto, un array, un número, una cadena, el valor booleano true o false y el valor nulo.

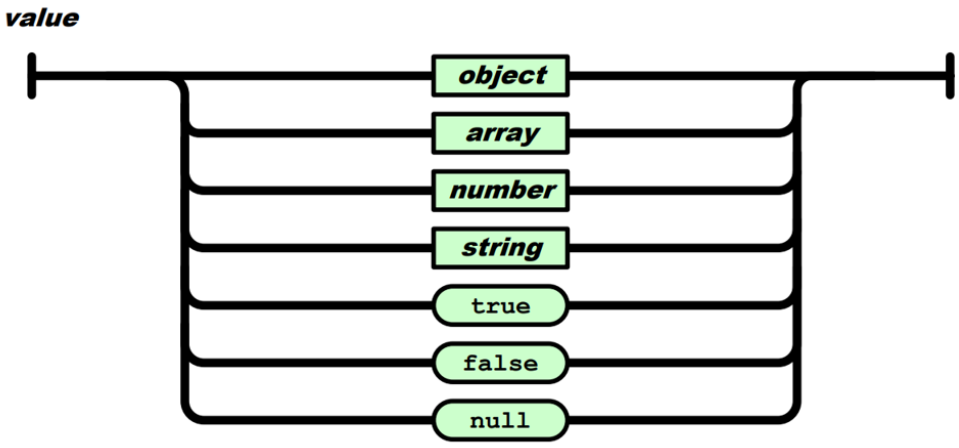

Figura 3: Valor JSON

En la Figura 4 se explica que un objeto JSON es una expresión encerrada entre llaves que está formada por la repetición de cero o más veces de una cadena, dos puntos, un valor JSON. Las repeticiones dentro del objeto se separan con la coma.

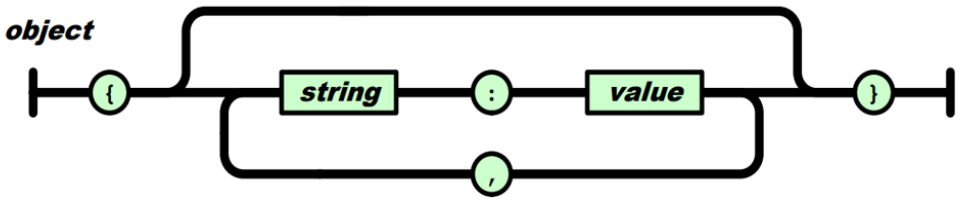

Figura 4: Objeto JSON

Por ejemplo, esto es un objeto que representa una persona que posee dos atributos, “nombre” y “apellidos”:

```
{
	"nombre": "Sergio",
	"apellidos": "Luján Mora"
}
```

La Figura 5 explica que un array JSON es una expresión encerrada entre corchetes que está formada por la repetición de cero o más veces de un valor JSON. Las repeticiones dentro del array se separan con la coma.

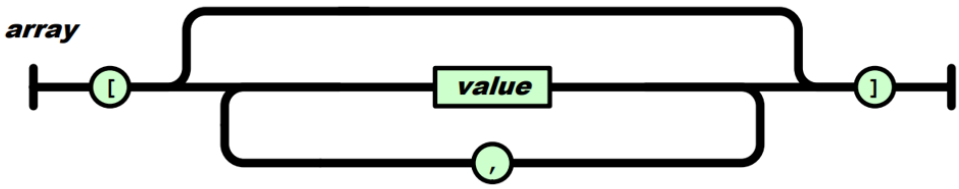

Figura 5: Array JSON

Por ejemplo, esto es un array que posee dos objetos que representan una persona que posee dos atributos, “nombre” y “apellidos”, cada una de ellas:

```
[
	{
		"nombre": "Sergio",
		"apellidos": "Luján Mora"
	},
	{
		"nombre": "Tim",
		"apellidos": "Berners-Lee"
	}
]
```

Normalmente, un valor JSON no se recibe “bonito”, bien formateado con espacios en blanco, tabuladores y saltos de línea como se puede ver a continuación:

```
{
	"kind": "youtube#searchListResponse",
	"etag": "\"m2yskBQFythfE4irbTIeOgYYfBU/PaiEDiVxOyCWelLPuuwa9LKz3Gk\"",
```

```
	"nextPageToken": "CAUQAA",
	"regionCode": "KE",
	"pageInfo": {
		"totalResults": 4249,
		"resultsPerPage": 5
	},
	"items": [
		{
			"kind": "youtube#searchResult",
			"etag": "\"m2yskBQFythfE4irbTIeOgYYfBU/QpOIr3QKlV5EUlzfFcVvDiJT0hw\"",
			"id": {
				"kind": "youtube#channel",
				"channelId": "UCJowOS1R0FnhipXVqEnYU1A"
			}
		},
		{
			"kind": "youtube#searchResult",
			"etag": "\"m2yskBQFythfE4irbTIeOgYYfBU/AWutzVOt_5p1iLVifyBdfoSTf9E\"",
			"id": {
				"kind": "youtube#video",
				"videoId": "Eqa2nAAhHN0"
			}
		},
		{
			"kind": "youtube#searchResult",
			"etag": "\"m2yskBQFythfE4irbTIeOgYYfBU/2dIR9BTfr7QphpBuY3hPU-h5u-4\"",
			"id": {
				"kind": "youtube#video",
				"videoId": "IirngItQuVs"
			}
		}
	]
}
```


Normalmente se recibe así, una línea de caracteres:


```
{"kind":"youtube#searchListResponse","etag":"\"m2yskBQFythfE4irbTIeOgYYfBU/PaiEDiVxOyCWelLPuuwa9LKz3Gk
```


¿Qué contiene esta cadena? Para saber qué hay aquí se puede usar algún visor de JSON [(11)](#).

En PHP existen dos funciones para trabajar con JSON:

- `json_decode(cadena)`: decodifica una cadena que representa un valor JSON.
- `json_encode(valor)`: devuelve la representación JSON del valor dado.

Por ejemplo, el siguiente código PHP:

```
$a = [
	["nombre" => "Sergio", "apellidos" => "Luján Mora"],
	["nombre" => "Tim", "apellidos" => "Berners-Lee"],
];

echo json_encode($a);
```

muestra la cadena:


```
[{"nombre":"Sergio","apellidos":"Luj\u00e1n Mora"},{"nombre":"Tim","apellidos":"Berners-Lee"}]
```

En el código anterior se puede observar que los dos arrays asociativos de PHP se han representado como objetos en PHP.

El segundo parámetro de la función `json_encode()` es una máscara de bits que permite controlar el comportamiento de la función. Por ejemplo, el siguiente código fuerza a que los arrays de PHP se conviertan siempre como objetos en PHP:

```
$a = [
      ["nombre" => "Sergio", "apellidos" => "Luján Mora"],
      ["nombre" => "Tim", "apellidos" => "Berners-Lee"],
];

echo json_encode($a, JSON_FORCE_OBJECT);
```

El código anterior muestra la cadena:

```
{"0":{"nombre":"Sergio","apellidos":"Luj\u00e1n Mora"},"1":{"nombre":"Tim","apellidos":"Berners-Lee"}}
```

Las posiciones del array se han convertido en las propiedades `"0"` y `"1"` de un objeto JSON.

5. Recomendaciones

Recuerda que las páginas que contengan código PHP tienen que tener la extensión `.php`. Si modificas alguna página web que ya tengas hecha de prácticas anteriores para añadirle código PHP, tendrás que cambiarle la extensión y corregir todos los enlaces que apunten a esa página.

Recuerda que para que el código PHP se ejecute, la página web tiene que pasar por el servidor web para que éste se la pase al intérprete de PHP. Para ello, tienes que cargar la página a través del servidor web: la URL de conexión tiene que tener la forma `http://localhost`.

El manual de PHP te lo puedes descargar en diferentes formatos de su sitio web (12) para tenerlo siempre a mano y poder hacer las búsquedas de información rápidamente. También puedes acceder a través de Internet a la ayuda de cualquier función de PHP escribiendo el nombre de la función a continuación de la URL `http://php.net/`. Por ejemplo, `http://php.net/header` muestra la ayuda de la función `header()`.

Cuando trabajes con ficheros debes realizar todas las comprobaciones que puedas, ya que los ficheros suelen ser origen de muchos problemas durante el tiempo de ejecución de un programa: un fichero puede ser borrado, puede cambiar su nombre o puede ser movido de sitio. Además, otro problema son los permisos del sistema de archivos: el fichero puede existir, pero puede ser que no tengas permisos para acceder a él. Por tanto, **siempre que trabajes con ficheros, el primer paso debería ser comprobar que el fichero realmente existe y tienes permisos de acceso a él.**

Permitir que los usuarios puedan subir ficheros al servidor web puede suponer un agujero de seguridad del sitio web, por lo que hay que tomar todas las medidas de protección que se pueda, como por ejemplo limitar el tamaño máximo de fichero que se puede subir o comprobar las extensiones de los ficheros que se suben.

Las funciones de números aleatorios no suelen controlar que no se repita el mismo número en peticiones sucesivas de números aleatorios. Este comportamiento es el esperado en un suceso con probabilidad no condicionada, como puede ser el lanzamiento de una moneda para elegir cara o cruz. Sin embargo, a veces interesa obtener una secuencia de números aleatorios, pero sin que se repita un número en toda la secuencia o que no se repita al menos en dos posiciones consecutivas. En estos casos, es el

programador el que debe controlar la generación de los números aleatorios para lograr el resultado esperado.

---

(1) http://es.php.net/manual/es/book.filesystem.php

(2) http://www.w3schools.com/php/php_file.asp

(3) http://www.w3schools.com/php/php_ref_filesystem.asp

(4) http://tools.ietf.org/html/rfc1867

(5) http://es.php.net/manual/es/features.file-upload.php

(6) http://www.w3schools.com/php/php_file_upload.asp

(7) No es una buena práctica utilizar `@` para "esconder" los errores, lo correcto es utilizar un manejador de errores.

(8) https://tools.ietf.org/html/rfc4627

(9) https://tools.ietf.org/html/rfc8259

(10) https://www.ecma-international.org/publications/standards/Ecma-404.htm

(11) https://codebeautify.org/jsonviewer

(12) http://www.php.net/download-docs.php