PHP y HTTP

Álvaro González Sotillo

9 de enero de 2025

Índice

1. Repositorio de ejemplos **1**

2. Variables **1**

3. Estructuras de control **3**

4. Funciones **5**

5. Funciones predefinidas **6**

6. Entorno de desarrollo **6**

7. HTTP **7**

8. Variables predefinidas **9**

9. Formularios **9**

10. Sesiones **10**

11. AJAX / `fetch` **10**

12. Ejercicios **11**

13. Referencias **11**

1. Repositorio de ejemplos

https://alvarogonzalezsotillo@gitlab.com/alvarogonzalezsotillo/apuntes-clase-ejemplos-php.git

2. Variables

https://www.php.net/manual/es/language.variables.basics.php

- Precedidas de `$`
- Comienzan por una letra, o _
- Después, números, letras y _

2.1. Tipos de variables

https://www.php.net/manual/es/language.types.php

- Tipos simples (escalares):
  - bool type
  - int type
  - float type
  - string type: https://www.php.net/manual/es/language.operators.string.php
- Otros tipos (ya los veremos):
  - null type
  - array type
  - object type
  - never type
  - void type

2.1.1. Strings

https://www.php.net/manual/es/language.types.string.php

- Comillas dobles:
  - Los caracteres se escapan con \
  - Sustituyen las apariciones de variables por su valor
- Comillas simples:
  - Los caracteres \ y ' se escapan con \. Todos los demás no se escapan.
- *Heredoc* y *nowdoc*:
  - Cadenas delimitadas por una cadena arbitraria
  - *nowdoc* es como *heredoc* sin interpretar variables

Secuencias de escape de cadenas	
\n	avance de línea (LF o 0x0A (10) en ASCII)
\r	retorno de carro (CR o 0x0D (13) en ASCII)
\t	tabulador horizontal (HT o 0x09 (9) en ASCII)
\v	tabulador vertical (VT o 0x0B (11) en ASCII) (desde PHP 5.2.5)
\e	escape (ESC o 0x1B (27) en ASCII) (desde PHP 5.4.4)
\f	avance de página (FF o 0x0C (12) en ASCII) (desde PHP 5.2.5)
\\	barra invertida
\$	signo de dólar
\"	comillas dobles
\999	Caracter en octal
\xFF	caracter hexadecimal
\u{FFFF}	Caracter unicode

```
$nombre = "María";
echo <<<EOT
Mi nombre es "$nombre".
Esto debería mostrar una 'A' mayúscula: \x41
EOT;
```

Listado 1: *heredoc*

```
$nombre = "María";
echo <<<'EOT'
Mi nombre es "$nombre".
Esto debería mostrar una 'A' mayúscula: \x41
EOT;
```

Listado 2: *nowdoc*

2.2. Constantes

https://www.php.net/manual/es/language.constants.php
https://www.php.net/manual/es/reserved.constants.php

- Definición: `define(NOMBRE, valor)`
- No se puede cambiar posteriormente su valor
- Se utilizan como variables, sin `$`
- También con la función `constant(NOMBRE)`

3. Estructuras de control

https://www.php.net/manual/es/language.control-structures.php

3.1. If

```
if( CONDICION ){
    SI ES CIERTO
}
else{
    SI ES FALSO
}
```

Ejercicio: si la multiplicación de dos variables es mayor de 50, muestra el resultado con fondo rojo. Si no, el fondo será transparente

3.2. While

```
while(CONDICION){
    SENTENCIAS QUE SE REPITEN
    DEBERIA CAMBIARSE LA CONDICION
}
```

Ejercicio: muestra un tablero de ajedrez, con tantas filas y columnas como se diga en dos variables

3.3. for

```
for (EXPR1; EXPR2; EXPR3){
    SENTENCIA
}
```

Es equivalente a un bucle `while`

```
EXPR1;
while (EXPR2){
    SENTENCIA
    EXPR3
}
```

Ejercicio: Muestra todas las imágenes de https://www.glerl.noaa.gov/data/ice/max_anim/png, desde 1973 a 2024.

Ejercicio: Muestra todas las imágenes de https://jareksastro.org/astro/, desde m0 a m23.

3.4. foreach

- *Arrays*
  - Colección de valores con un nombre (clave o índice)
  - Cada posición del *array* tiene un valor
  - La posición puede ser numérica, o un *string*
  - Son parecidos a los objetos de Javascript

```php
$array = [
    "foo" => "bar",
    "bar" => "foo",
];
$alumnos = [
    "Pepe",
    "María",
    "Juan",
    "Susana"
];
```

- Se pueden recorrer los valores de un *array*
  - visitando solo los valores
  - con los índices (claves) y los valores

```php
$alumnos = [
    "Pepe",
    "María",
    "Juan",
    "Susana"
];
foreach($alumnos as $alumno){
    print( "Alumno: $alumno" )
}
foreach($alumnos as $posicion => $alumno){
    print( "Alumno $posicion: $alumno" )
}
```

Ejercicio: muestra las siguientes imágenes de https://astro.uni-bonn.de/~ithies/images/Planet_Formation_Illustrated/PNG/ en una página web:

annulus1.png	annulus2.png	ejection1.png	ejection2.png
ejection3.png	encounter1.png	encounter2.png	formation1.png
formation2.png	formation3.png	formation4.png	formation5.png
formation6.png	formation7.png	formation8.png	formation9.png
inclined1.png	inclined2.png	inclined3.png	inclined4.png
kozai1.png	kozai2.png	kozai3.png	rmleffect1.png
rmleffect2.png	system1.png	system2.png	

3.5. `continue`

- Dentro de un bucle, ignora el resto de órdenes y vuelve al principio

```php
for( $i = 0 ; $i < 10 ; $i += 1 ){
    if( $i % 3 != 0 ){
        continue;
    }
    echo( "$i es múltiplo de 3\n");
}
```

3.6. `include`, `require`

- Añaden a un fichero PHP el contenido de otro fichero PHP, antes de ejecutarlo
- Útil para
  - Funciones y variables comunes a varias páginas
  - Partes comunes entre páginas: Cabecera, pie, menús...
- `include`: Si el fichero no existe, provoca un *warning*
- `require`: Si el fichero no existe, provoca un error
- `include_once`, `require_once`: Si ya ha sido incluido, no se vuelve a incluir

3.7. try

https://www.php.net/manual/en/language.exceptions.php

- Solo funciona con `throw`, no con errores generales.
- Intentaremos no usarlo, excepto con `mysqli`

```php
<?php
function inverso($x) {
    if (!$x) {
        throw new Exception('División por cero.');
    }
    return 1/$x;
}

try {
    echo inverso(5) . "\n";
    echo inverso(0) . "\n";
}
catch (Exception $e) {
    echo 'Excepción capturada: ',  $e->getMessage(), "\n";
}
finally{
    echo 'Esto ocurre haya o no haya pasado por catch\n';
}
```

4. Funciones

4.1. Definición y uso

https://www.php.net/manual/es/language.functions.php

```php
<?php
function foo($arg_1, $arg_2, /* ..., */ $arg_n)
{
    echo "Función de ejemplo.\n";
    return $valor_devuelto;
}
?>
```

4.2. Ámbito de variables

https://www.php.net/manual/es/language.variables.scope.php

- Global: variables definidas fuera de las funciones
- Local: variables definidas en funciones

```php
$a = 3; // GLOBAL

function fun1(){
    echo $a; // ERROR, VARIABLE NO DEFINIDA
}

function fun2(){
    global $a;
    echo $a; // IMPRIME 3
}

function fun3(){
    $a = 4
    echo $a; // IMPRIME 4
    echo $GLOBALS["a""]; // IMPRIME 3
}
```

Nota: para `$GLOBALS` ver Variables predefinidas

4.3. Ejercicio

- `es_capicua`
- `es_primo`
- Encuentra un primo mayor de 1000000 utilizando las funciones anteriores

4.4. Parámetros por referencia/valor

- Un parámetro por valor **copia** su valor para la función, tanto para escalares como para *arrays*
- Un parámetro por referencia usa la misma variable dentro de la función

```php
function incrementa($variable, $valor){
    $variable += $valor;
}

$unavariable = 3;
incrementa($unavariable, 4);
echo $unavariable; // IMPRIME 3
```

```php
function incrementa(&$variable, $valor){
    $variable += $valor;
}

$unavariable = 3;
incrementa($unavariable, 4);
echo $unavariable; // IMPRIME 7
```

4.4.1. Objetos

- Los objetos se pasan siempre por referencia

```php
class Persona {
    public $nombre;

    public function __construct($nombre) {
        $this->nombre = $nombre;
    }
}

function cambiarNombre($persona) {
    $persona->nombre = "Carlos";
}

$persona1 = new Persona("Ana");
cambiarNombre($persona1);

echo $persona1->nombre; // Imprime "Carlos", no "Ana"
```

5. Funciones predefinidas

`phpinfo`
`htmlspecialchars`
`parse_url`
`explode, explode`
`strpos, substr, trim`
`strtolower, strtoupper`
`filter_var`

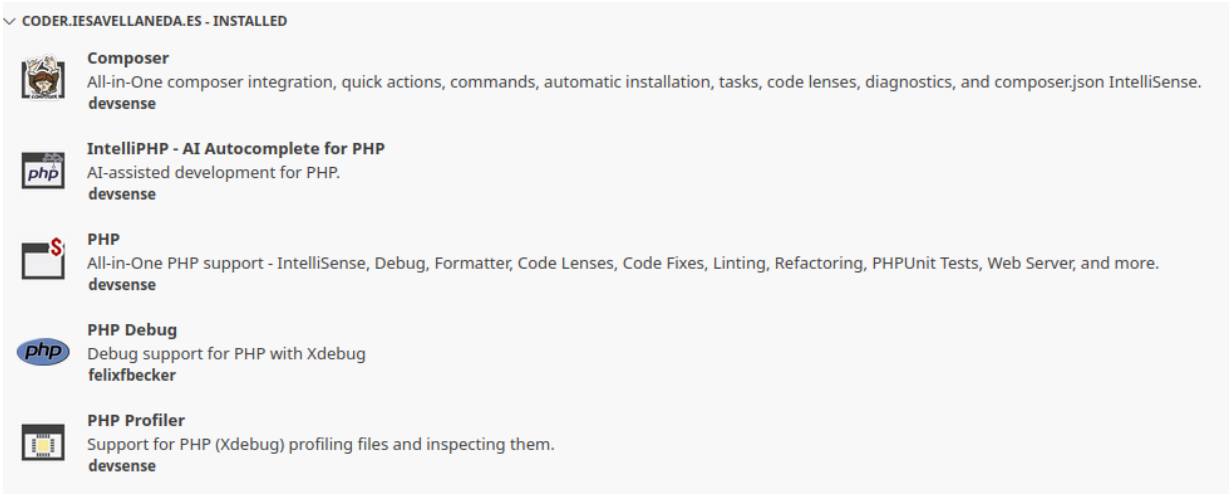
6. Entorno de desarrollo

- Depuración en VSCode:
  - Instalar `php-xdebug`

- Instalar alguna extensión en VSCode (F5)

- Trazas:
  - `var_dump`
  - `error_reporting`
  - `isset`
  - `is_numeric, is_array, is_nan`
  - Opciones `php.ini`:
    - `display_errors`
    - `display_startup_errors`
    - `log_errors`

6.1. Extensiones VSCode



7. HTTP

https://developer.mozilla.org/es/docs/Web/HTTP

7.1. URL

```php
<?php
$url = 'http://username:password@hostname:9090/path?arg=value#anchor';

var_dump(parse_url($url));
var_dump(parse_url($url, PHP_URL_SCHEME));
var_dump(parse_url($url, PHP_URL_USER));
var_dump(parse_url($url, PHP_URL_PASS));
var_dump(parse_url($url, PHP_URL_HOST));
var_dump(parse_url($url, PHP_URL_PORT));
var_dump(parse_url($url, PHP_URL_PATH));
var_dump(parse_url($url, PHP_URL_QUERY));
var_dump(parse_url($url, PHP_URL_FRAGMENT));
?>
```

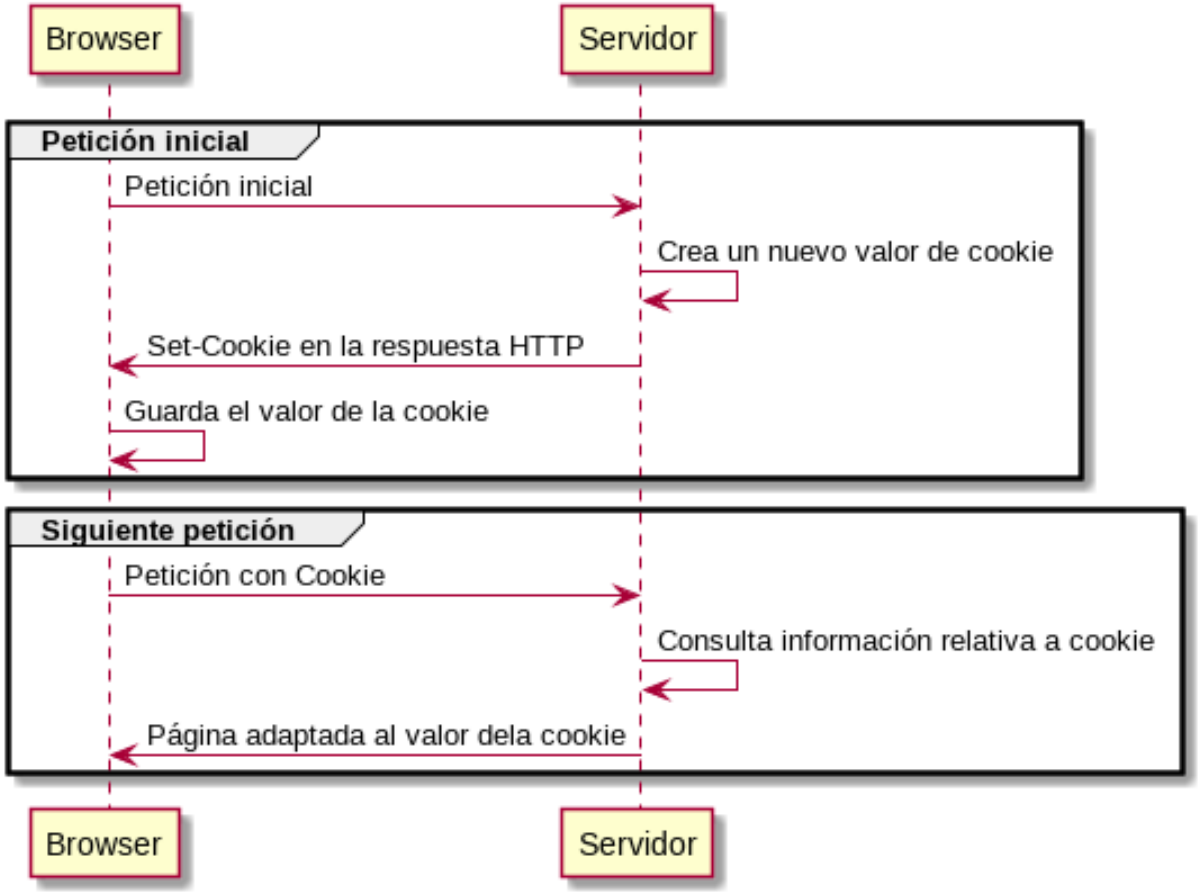
7.2. *Cookies*

- Las /cookies/ son cadenas que el servidor envía al navegador
  - Cabecera Set-Cookie
- El navegador guarda, y envía en las siguientes peticiones al servidor

- Cabecera Cookie

- El valor de la *cookie* no suelen ser datos
  - Es un identificador con el que el servidor accede a una base de datos
  - En esa base de datos están los datos reales asociados a la *cookie*

7.3. Diagrama



7.4. Uso de *cookies* en php

- Funcion `setcookie()`
- Variable `$_COOKIE`

7.4.1. Ejercicio

- Crea una página PHP que almacene una *cookie* en el navegador
  - Nombre: `ejercicio-cookies`
  - Valor: La marca de tiempo de la petición de la página (ver `getdate()` o `date()`)
- La página, si recibe la *cookie*, mostrará la marca de tiempo del acceso anterior, y la marca de tiempo actual

7.5. *headers*

- Tanto cliente como servidor pueden enviar cabeceras en la petición/respuesta
  - Por ejemplo, las *cookies* son una cabecera de la petición
- `getallheaders()`: consigue todas las cabeceras enviadas por el cliente

- Cabeceras comunes en las peticiones: `Accept`, `Cache-Control`, `Content-Enconding`...

- `header()`: Envía una cabecera en la respuesta HTTP
  - Cabeceras comunes en las respuestas: `Content-Type`, `Location`, `Transfer-Encoding`...

7.5.1. Ejercicio

- Crea una página web PHP que reenvíe a `https://www.google.com` utilizando una cabecera `Location`
- Crea una página web PHP que muestre código HTML, pero que se vea como texto, usando `Content-Type`
- Investiga qué cabeceras se utilizan para continuar un *download* de un fichero grande, después de una interrupción.

8. Variables predefinidas

https://www.php.net/manual/es/language.variables.predefined.php

```
$GLOBALS -- Hace referencia a todas las variables disponibles en el ámbito global
$_SERVER -- Información del entorno del servidor y de ejecución
$_GET -- Variables HTTP GET
$_POST -- Variables POST de HTTP
$_FILES -- Variables de subida de ficheros HTTP
$_REQUEST -- Variables HTTP Request
$_SESSION -- Variables de sesión
$_ENV -- Variables de entorno
$_COOKIE -- Cookies HTTP
```

9. Formularios

9.1. En HTML

- Vistos el año pasado en Lenguajes de marcas
- Referencia de w3schools

9.2. En PHP

https://www.php.net/manual/es/tutorial.forms.php https://www.php.net/manual/es/language.variables.external.php

- Los campos de un formulario llegan como valores en los *arrays*
  - `$_POST`
  - `$_GET`
  - `$_REQUEST`: reune `$_POST`, `$_GET` y `$_COOKIE`
  - `$_FILES`: Ficheros subidos por un formulario.

9.2.1. Ficheros en formularios

- El formulario debe tener `enctype="multipart/form-data"`
- Se pueden poner límites a los ficheros subidos con los parámetros de `php.ini`
  - `file_uploads`
  - `upload_max_filesize`
  - `upload_tmp_dir`
  - `post_max_size`
  - `max_input_time`
- La variable `$_FILES` contiene los ficheros subidos

10. Sesiones

https://www.php.net/manual/es/book.session.php https://www.php.net/manual/es/session.examples.basic.php

10.1. Uso básico

- La variable `$_SESSION` es un *array* con valores, que pueden pasarse de una página a otra
- `session_start()` se basa en *cookies* y ficheros en el servidor
  - La *cookie* es `PHPSESSID`
  - Se puede usar también el parámetro `SID` en vez de *cookies*
  - Se puede cambiar el almacenamiento de sesiones, por defecto en `session.save_path`

```php
session_start();
if (!isset($_SESSION['count'])) {
  $_SESSION['count'] = 0;
} else {
  $_SESSION['count']++;
}
```

```php
session_start();
unset($_SESSION['count']);
```

```php
session_start();
session_destroy(); // esto no destruye $_SESSION
```

10.2. Ejercicio: Contador de visitas

- Haz una página que cada vez que sea visitada, incremente el número de visitas
- Cada visitante tiene su propio número de visitas

10.3. Ejercicio: ventana de *login*

- Crea un formulario de *login*
- Si la contraseña es un número impar, es correcta
- Si el usuario ya ha entrado, no se muestra el formulario de *login*, sino su nombre y la contraseña que introdujo. También habrá un botón de *logout*, para salir de la sesión

11. AJAX / `fetch`

- El envío de un formulario implica la recarga de la página
- A veces, es necesario actualizar solo *parte* de la página
  1. Se envía una petición al servidor desde *javascript*. La página no se recarga
  2. *javascript* recibe la respuesta
  3. *javascript* utiliza el DOM o cualquier ora API para reflejar dicha respuesta

11.1. AJAX

- Basado en `XMLHttpRequest`
- Es la forma antigua
  - Basado en *callbacks*
  - En deshuso

11.2. `fetch`

- Basado en Promise
- Integrado con modernas API asíncronas

```
const response = await fetch("https://example.org/post", {
  method: "POST",
  headers: {
    "Content-Type": "application/json",
  },
  body: JSON.stringify({ username: "example" }),
});
```

11.2.1. Enviar un formulario

```
const form = document.getElementById("formulario");
const formData = new FormData(form);
const response = fetch( "./api.php", {
    method: "POST",
    body: formData
})
```

11.2.2. Promesas

```
fetch( "./api/endpoint", {
    method: "POST",
    body: JSON.stringify({ username:"example", action:"create"} )
}).
then( response =>{
    const json = =response.json();
    // PROCESAR EL JSON DE RESPUESTA
}).
catch( err => {
    alert("Error:" + err );
    console.err(err);
});
```

11.3. JSON en PHP

- `json_decode()`: Convierte una cadena de JSON a un *array* de PHP.
- `json_encode()`: Convierte un *array* de PHP a una cadena JSON.

```
<?php
$json = file_get_contents('php://input');
$request = json_decode($json, true);

// PROCESAR $request Y CREAR EL ARRAY DE RESPUESTA EN $response

header('Content-Type: application/json');
echo json_encode($response);
?>
```

12. Ejercicios

https://www.w3schools.com/PHP/php_exercises.asp

13. Referencias

- Formatos:
  - Transparencias
  - PDF

    - Página web
    - EPUB
- Creado con:
    - Emacs
    - org-re-reveal
    - Latex
- Alojado en Github