

APUNTES DE PHP

Parte 2: Arrays y Funciones

Contenido:

- Arrays (normales, asociativos, bidimensionales)
- Operaciones con Arrays
- Funciones
- Parámetros y Alcance
- Funciones Predefinidas

8. Arrays

¿Qué son?

Estructuras de datos que almacenan múltiples valores en una sola variable.

En PHP, todos los arrays son mapas ordenados (pares clave-valor).

■■ **IMPORTANTE:** Aunque el tipado es dinámico, NO mezcles tipos en un mismo array

8.1 Arrays Normales (Secuenciales)

Índices numéricos empezando en 0 (0-index como Java)

```
<?php
// Tres formas de declarar arrays
$frutas = array("naranja", "pera", "manzana");
$frutas2 = ["naranja", "pera", "manzana"]; // PHP >= 5.4

$frutas3 = [];
$frutas3[0] = "naranja";
$frutas3[1] = "pera";
$frutas3[] = "manzana"; // Añade al final

// Acceder a elementos
echo $frutas[0]; // "naranja"
echo $frutas[2]; // "manzana"
?>
```

Tamaño del array: count(\$array)

```
<?php
$frutas = ["naranja", "pera", "manzana"];
$tam = count($frutas); // 3

// Recorrer con FOR
for ($i = 0; $i < count($frutas); $i++) {
    echo "Elemento $i: $frutas[$i]<br>";
}

// Recorrer con FOREACH (más elegante)
foreach ($frutas as $fruta) {
    echo "$fruta<br>";
}
?>
```

8.2 Arrays Asociativos

Cada elemento es un par **clave => valor**

Se accede por la clave, NO por posición

```
<?php
$capitales = [
    "Italia" => "Roma",
    "Francia" => "París",
    "Portugal" => "Lisboa"
];

// Acceso por clave
echo $capitales["Francia"]; // "París"

// Añadir elemento
$capitales["Alemania"] = "Berlín";

// Mostrar en string
echo "La capital es {$capitales['Francia']}";

// Recorrer con foreach
foreach ($capitales as $pais => $ciudad) {
    echo "$pais: $ciudad<br>";
}
?>
```

■ ■ ¡NO HAGAS ESTO! \$capitales[] = 'Madrid'; // Añade con clave 0 ■

8.3 Operaciones con Arrays

Función	Descripción	Ejemplo
print_r(\$array)	Muestra contenido del array	print_r(\$frutas);
var_dump(\$var)	Muestra tipo y contenido	var_dump(\$frutas);
count(\$array)	Devuelve el tamaño	\$tam = count(\$frutas);
in_array(\$elem, \$array)	Verifica si existe	in_array("pera", \$frutas)
array_push(\$array, \$elem)	Añade al final	array_push(\$frutas, "piña");
array_pop(\$array)	Elimina y devuelve último	\$ult = array_pop(\$frutas);
array_keys(\$array)	Devuelve array de claves	\$claves = array_keys(\$caps);
sort(\$array)	Ordena el array	sort(\$frutas);
isset(\$array[\$elem])	Verifica si existe elemento	isset(\$frutas[2])
unset(\$array[\$elem])	Elimina elemento (deja hueco)	unset(\$frutas[1]);

```
<?php
$frutas = ["naranja", "pera", "manzana"];

array_push($frutas, "piña");
print_r($frutas); // Array([0]=>naranja [1]=>pera [2]=>manzana [3]=>piña)

$ultima = array_pop($frutas);
if (in_array("piña", $frutas)) {
    echo "Queda piña";
} else {
    echo "No queda piña"; // Esto se imprime
}
?>
```

8.4 Arrays Bidimensionales (Matrices)

Arrays de arrays - útiles para tablas y estructuras complejas

```
<?php
// Matriz 3x3
$matriz = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
];

// Acceso: $matriz[fila][columna]
echo $matriz[0][0]; // 1
echo $matriz[1][2]; // 6

// Recorrer con foreach anidado
foreach ($matriz as $fila) {
    foreach ($fila as $valor) {
        echo "$valor ";
    }
    echo "<br>";
}

// Recorrer con índices
foreach ($matriz as $i => $fila) {
    foreach ($fila as $j => $valor) {
        echo "[$i][$j] = $valor<br>";
    }
}
?>
```

Arrays Asociativos Multidimensionales:

```
<?php
$alumnos = [
    ["nombre" => "Juan", "edad" => 20, "nota" => 8.5],
    ["nombre" => "Ana", "edad" => 22, "nota" => 9.0],
    ["nombre" => "Pedro", "edad" => 21, "nota" => 7.5]
];

// Acceso
echo $alumnos[0]["nombre"]; // "Juan"
echo $alumnos[1]["nota"]; // 9.0

// Recorrer
foreach ($alumnos as $alumno) {
    echo "{$alumno['nombre']}: {$alumno['nota']}<br>";
}
?>
```

■ ■ **MEJOR PRÁCTICA:** Para estructuras complejas, usa objetos en lugar de arrays multidimensionales

9. Funciones

Sintaxis básica:

```
<?php
function nombreFuncion($par1, $par2) {
    // código
    return $valor;
}

$resultado = nombreFuncion($arg1, $arg2);
?>
```

Características:

- No se declaran tipos de parámetros (en PHP básico)
- Paso de parámetros por VALOR (se hace copia)
- No se indica el tipo de retorno (en PHP básico)

```
<?php
function diaSemana() {
    $semana = ["lunes", "martes", "miércoles", "jueves",
               "viernes", "sábado", "domingo"];
    $dia = $semana[rand(0, 6)];
    return $dia;
}

$diaCine = diaSemana();
echo "El próximo $diaCine voy al cine";
?>
```

9.1 Parámetros por Referencia

Usar **&** en la declaración para modificar la variable original

```
<?php
function duplicarPorValor($arg) {
    $arg = $arg * 2;
    echo "Dentro: $arg<br>";
}

function duplicarPorReferencia(&$arg) {
    $arg = $arg * 2;
    echo "Dentro: $arg<br>";
}

$num1 = 5;
duplicarPorValor($num1);
echo "Fuera: $num1<br>"; // 5 (no cambió)

$num2 = 7;
duplicarPorReferencia($num2);
echo "Fuera: $num2<br>"; // 14 (sí cambió)
?>
```

9.2 Parámetros Opcionales/Por Defecto

Asignar valores por defecto en la declaración

```
<?php
function saluda($nombre, $prefijo = "Sr") {
    echo "Hola $prefijo $nombre";
}

saluda("Alejandro", "Mr"); // "Hola Mr Alejandro"
saluda("Alejandro");       // "Hola Sr Alejandro"
?>
```

■ ■ **REGLA:** Los parámetros con valor por defecto SIEMPRE al final

9.3 Parámetros Variables (número variable de argumentos)

Forma clásica: func_num_args(), func_get_arg()

```
<?php
function sumaParametros() {
    if (func_num_args() == 0) {
        return false;
    }
    $suma = 0;
    for ($i = 0; $i < func_num_args(); $i++) {
        $suma += func_get_arg($i);
    }
    return $suma;
}

echo sumaParametros(1, 5, 9); // 15
?>
```

Forma moderna (PHP >= 5.6): Operador ... (variadics)

```
<?php
function sumaParametros(...$numeros) {
    if (count($numeros) == 0) {
        return false;
    }
    $suma = 0;
    foreach ($numeros as $num) {
        $suma += $num;
    }
    return $suma;
}

echo sumaParametros(1, 5, 9); // 15

// También para dividir arrays
function suma($a, $b) {
    return $a + $b;
}
echo suma(...[10, 20]); // 30
?>
```

9.4 Argumentos con Nombre (PHP 8.0+)

Pasar argumentos indicando explícitamente el nombre del parámetro

```
<?php
function calcular($a, $b = 2, $c = 4) {
    return $a + $b + $c;
}

// Argumentos posicionales
echo calcular(1, 3, 5); // 9

// Argumentos con nombre
echo calcular(c: 3, a: 1); // 1 + 2 + 3 = 6

// Mezclar ambos (posicionales primero)
echo calcular(1, c: 3); // 1 + 2 + 3 = 6
?>
```

9.5 Funciones Tipadas (PHP 7+)

Declarar tipos de parámetros y retorno para mayor seguridad

```
<?php
declare(strict_types=1); // Primera línea del archivo

function suma(int $a, int $b): int {
    return $a + $b;
}

echo suma(10, 30); // 40
echo suma(10, "5"); // ERROR por tipado estricto

// Función que no devuelve nada
function mostrar(string $texto): void {
    echo $texto;
}
?>
```

Tipos disponibles: int, float, string, bool, array, object, void

■ **IMPORTANTE:** declare(strict_types=1); debe ser la primera línea del archivo

9.6 Alcance de Variables

Tipo	Dónde se declara	Dónde es accesible
Global	Fuera de funciones	En todo el script
Local/Función	Dentro de función	Solo dentro de la función
Parámetro	En declaración función	Solo dentro de la función

```
<?php
// Variables locales tienen prioridad
function miCiudad() {
    $ciudad = "Elche"; // Variable local
    echo "Dentro: $ciudad<br>";
}

$ciudad = "Alicante"; // Variable global
echo "Antes: $ciudad<br>"; // Alicante
miCiudad(); // Elche
echo "Después: $ciudad<br>"; // Alicante (no cambió)

// Para usar variable global dentro de función
function miCiudad2() {
    global $ciudad; // Declarar como global
    $ciudad = "Murcia";
}

miCiudad2();
echo $ciudad; // Murcia (sí cambió)
?>
```

■ ■ EVITA USAR GLOBAL: Es mejor pasar valores como parámetros

9.7 Funciones Variable

Asignar una función a una variable y ejecutarla dinámicamente

```
<?php
function suma($a, $b) {
    return $a + $b;
}

$miFuncion = "suma"; // Nombre de la función en string
echo $miFuncion(3, 4); // 7

// Funciones anónimas (PHP 5.3+)
$saluda = function($nombre) {
    echo "Hola $nombre";
};
$saluda("Alejandro"); // "Hola Alejandro"
?>
```


9.8 Biblioteca de Funciones

Agrupar funciones en archivos para reutilización

Formas de incluir:

Instrucción	Comportamiento
include(archivo)	Incluye, si no existe continúa con warning
include_once(archivo)	Incluye solo una vez (evita duplicados)
require(archivo)	Incluye, si no existe ERROR FATAL
require_once(archivo)	Requiere solo una vez (recomendado)

```
<?php
// Archivo: biblioteca.php
function suma($a, $b) {
    return $a + $b;
}
function resta($a, $b) {
    return $a - $b;
}
?>

<?php
// Archivo: principal.php
require_once("biblioteca.php");

echo suma(10, 20); // 30
echo resta(40, 20); // 20
?>
```

■ ■ **RECOMENDACIÓN:** Usa `require_once` para archivos críticos

9.9 Plantillas con include

Separar HTML en archivos reutilizables (cabecera, pie, etc.)

```
<?php
// encabezado.php
?>
<!DOCTYPE html>
<html>
<head>
    <title><?= $titulo ?></title>
</head>
<body>

<?php
// pie.php
?>
<footer>Pie de página</footer>
</body>
</html>

<?php
// pagina.php
$titulo = "Mi Página";
include("encabezado.php");
?>
<h1><?= $titulo ?></h1>
<p>Contenido de la página</p>
<?php
include("pie.php");
?>
```

10. Funciones Predefinidas

PHP incluye miles de funciones organizadas por categorías

10.1 Funciones de Cadenas

Comillas:

- Simples (") → sin interpretación
- Dobles (") → interpretan variables y secuencias de escape (\n, \t, \\$)

```
<?php
$nombre = "Alejandro";
echo "Hola $nombre"; // Interpreta $nombre
echo 'Hola $nombre'; // Muestra literal "$nombre"

$moneda = "euro";
echo "Son 30 {$moneda}s"; // Llaves para separar
?>
```

Operaciones básicas con cadenas:

Función	Descripción	Ejemplo
strlen(\$str)	Longitud de cadena	strlen("Hola") → 4
substr(\$str, inicio, long)	Subcadena	substr("Hola", 0, 2) → "Ho"
str_replace(buscar, reemplazar, str)	Reemplazar texto	str_replace("a", "e", "Hala")
strtolower(\$str)	A minúsculas	strtolower("HOLA") → "hola"
strtoupper(\$str)	A mayúsculas	strtoupper("hola") → "HOLA"
trim(\$str)	Quitar espacios inicio/fin	trim(" hola ") → "hola"
explode(separador, \$str)	Cadena a array	explode(",", "a,b,c")
implode(separador, \$array)	Array a cadena	implode("-", [1,2,3])
strpos(\$str, buscar)	Posición de subcadena	strpos("hola", "o") → 1
strcmp(\$str1, \$str2)	Comparar cadenas	strcmp("a", "b") → -1

```
<?php
$cadena = "El caballero oscuro";
$tam = strlen($cadena); // 19

$oscuro = substr($cadena, 13); // "oscuro"
$caba = substr($cadena, 3, 4); // "caba"

$nueva = str_replace("c", "k", $cadena); // "El kaballero oskuro"

echo strtoupper($cadena); // "EL CABALLERO OSCURO"

$frase = "Hola,mundo,PHP";
$palabras = explode(",", $frase); // ["Hola", "mundo", "PHP"]

$array = ["uno", "dos", "tres"];
$texto = implode("-", $array); // "uno-dos-tres"
?>
```

Funciones de comprobación (ctype):

Función	Comprueba si...
ctype_alpha(\$str)	Solo contiene letras
ctype_alnum(\$str)	Solo alfanuméricos (letras y números)
ctype_digit(\$str)	Solo dígitos
ctype_space(\$str)	Solo espacios/tabuladores/saltos
ctype_punct(\$str)	Solo puntuación (sin espacios)

```
<?php
echo ctype_alpha("hola"); // true
echo ctype_alnum("hola123"); // true
echo ctype_digit("12345"); // true
echo ctype_space(" "); // true
?>
```

10.2 Funciones Matemáticas

Función/Constante	Descripción	Ejemplo
M_PI	Constante π	echo M_PI; \rightarrow 3.14159...
M_E	Constante e	echo M_E; \rightarrow 2.71828...
abs(\$num)	Valor absoluto	abs(-5) \rightarrow 5
pow(\$base, \$exp)	Potencia	pow(2, 3) \rightarrow 8
sqrt(\$num)	Raíz cuadrada	sqrt(16) \rightarrow 4
round(\$num)	Redondear	round(4.6) \rightarrow 5
ceil(\$num)	Redondear arriba	ceil(4.1) \rightarrow 5
floor(\$num)	Redondear abajo	floor(4.9) \rightarrow 4
rand(\$min, \$max)	Número aleatorio	rand(1, 10)
max(\$a, \$b, ...)	Valor máximo	max(1, 5, 3) \rightarrow 5
min(\$a, \$b, ...)	Valor mínimo	min(1, 5, 3) \rightarrow 1

```
<?php
echo M_PI; // 3.1415926535898

$num = 7.7;
echo floor($num); // 7
echo ceil($num); // 8
echo round($num); // 8

$saleatorio = rand(1, 100); // Número entre 1 y 100

// Formatear números
$precio = 1234.5678;
echo number_format($precio, 2); // "1,234.57"
echo number_format($precio, 2, ",", "."); // "1.234,57"
?>
```

10.3 Funciones de Tipos de Datos

Función	Descripción
gettype(\$var)	Devuelve el tipo como string
is_int(\$var)	Verifica si es entero
is_float(\$var)	Verifica si es decimal
is_string(\$var)	Verifica si es cadena
is_bool(\$var)	Verifica si es booleano
is_array(\$var)	Verifica si es array
is_object(\$var)	Verifica si es objeto
intval(\$var)	Convierte a entero
floatval(\$var)	Convierte a decimal
strval(\$var)	Convierte a cadena
settype(\$var, tipo)	Fuerza conversión de tipo

```
<?php
$numero = 123;
echo gettype($numero); // "integer"
echo is_int($numero);  // true

$texto = "456";
echo is_string($texto); // true

// Conversiones
$entero = intval("123"); // 123 (int)
$decimal = floatval("3.14"); // 3.14 (float)
$cadena = strval(456); // "456" (string)

// Forzar tipo
$var = "123";
settype($var, "integer");
echo is_int($var); // true
?>
```



FIN DE LA PARTE 2

■ Has completado todos los apuntes del Lenguaje PHP

Contenido total cubierto:

- PHP Básico • Variables y Constantes • Operadores
- Formularios • Condiciones • Bucles
- Arrays • Funciones • Funciones Predefinidas

¡Mucha suerte en tu examen! ■

