

APUNTES DE PHP

Parte 1: Fundamentos del Lenguaje

Contenido:

- PHP Básico
- Variables y Constantes
- Operadores
- Formularios
- Condiciones
- Bucles

1. PHP - Introducción

¿Qué es PHP?

- Acrónimo de *Personal Home Page*
- Lenguaje de propósito general, especializado en desarrollo web
- Sintaxis similar a C/Java
- El código se ejecuta en el servidor (Apache mediante mod_php)
- El cliente recibe el resultado generado
- Archivos con extensión **.php**

■■ **NOTA:** Versión actual: PHP 8.x - Importante usar al menos PHP 7.0 por rendimiento

1.1 Código Embebido

El código PHP se escribe entre `<?php y ?>`

Las sentencias se separan con ;

```
<?php
echo "Hola mundo";
echo "Es muy fácil programar en PHP.";
?>
```

■■ **IMPORTANTE:** Si un archivo solo contiene PHP (clases, interfaces), usa solo la etiqueta de apertura `<?php` sin cerrarla

1.2 Generando Contenido

Tres formas de mostrar contenido:

Sintaxis	Uso
<code>echo expresión;</code>	Más común, sin paréntesis
<code>print(expresión);</code>	Con paréntesis, devuelve 1
<code>&lt;?=<expresión ?&gt;</code>	Atajo para echo, ideal en HTML

```
<?php
echo "Texto con echo";
print("Texto con print");
?>
&lt;p&gt;&lt;?<= "Texto con atajo" ?&gt;&lt;/p&gt;
```

1.3 Comentarios

```
<?php
// Comentario de una línea

/* Comentario
   de múltiples
   líneas */
?>
```

1.4 Errores

Los errores se muestran como **Fatal Error** y desde PHP 5 se lanzan como excepciones.

Más adelante veremos el uso de **try/catch**

2. Variables

Características:

- No es necesario declararlas previamente
- Comienzan por \$, ejemplo: **\$nombre**
- Tras el \$, el siguiente carácter debe ser letra minúscula o guión bajo _
- Son **case sensitive**: \$var ≠ \$Var
- **Tipado dinámico**: el tipo se asigna en tiempo de ejecución
- Conveniente inicializarlas para evitar errores

```
<?php
$nombre = "Alejandro";
$edad = 20;
$altura = 1.75;
$estudiante = true;

echo "Me llamo $nombre";
echo "Tengo $edad años";
?>
```

Tipos de datos en PHP:

Tipo	Descripción	Ejemplo
boolean	Verdadero/Falso	true, false
integer	Números enteros	123, -456
float	Números decimales	3.14, -0.5
string	Cadenas de texto	"Hola", 'Adiós'
array	Colecciones	[1, 2, 3]
object	Instancias de clase	new MiClase()
null	Sin valor	null

3. Constantes

Variables cuyo valor NO varía. Dos formas de declararlas:

```
<?php
define("PI", 3.1416); // Forma clásica
const IVA = 0.21;    // PHP >= 5.3

echo PI; // NO lleva $
echo IVA; // NO lleva $
?>
```

■■ **CONVENCIÓN:** Las constantes se declaran siempre en MAYÚSCULAS

Constantes predefinidas (Magic Constants):

- __LINE__, __FILE__, __DIR__, __FUNCTION__, __CLASS__, etc.

4. Operadores

4.1 Operadores Aritméticos

Operador	Nombre	Ejemplo
-\$a	Negación	Opuesto de \$a
\$a + \$b	Suma	Suma de \$a y \$b
\$a - \$b	Resta	Diferencia
\$a * \$b	Multiplicación	Producto
\$a / \$b	División	Cociente
\$a % \$b	Módulo	Resto de división
\$a ** \$b	Potencia	\$a elevado a \$b

Concatenación de cadenas: Se usa el operador . (punto)

```
<?php
$x = 33;
$y = 11;
echo "La suma de ".$x." y ".$y." es ".$( $x + $y );

// Las variables se expanden automáticamente:
echo "La suma de $x y $y es ".$( $x + $y );
?>
```

4.2 Operadores de Comparación

Operador	Nombre	Resultado
\$a == \$b	Igual	true si son iguales (con conversión)
\$a === \$b	Idéntico	true si iguales Y mismo tipo
\$a != \$b	Diferente	true si diferentes
\$a !== \$b	No idéntico	true si diferentes o distinto tipo
\$a < \$b	Menor que	true si \$a < \$b
\$a > \$b	Mayor que	true si \$a > \$b
\$a <= \$b	Menor o igual	true si \$a <= \$b
\$a >= \$b	Mayor o igual	true si \$a >= \$b
\$a <=> \$b	Nave espacial	-1, 0 o 1 (PHP 7+)

■■ DIFERENCIA IMPORTANTE: == compara valores, === compara valores Y tipos

4.3 Operadores Lógicos

Operador	Nombre	Resultado
\$a && \$b	AND (y)	true si ambos son true
\$a \$b	OR (o)	true si al menos uno es true

!\$a	NOT (no)	true si \$a es false
\$a xor \$b	XOR	true si solo uno es true

4.4 Operadores de Asignación

Operador	Equivalente	Descripción
\$a = \$b	-	Asignación básica
\$a += \$b	\$a = \$a + \$b	Suma y asigna
\$a -= \$b	\$a = \$a - \$b	Resta y asigna
\$a *= \$b	\$a = \$a * \$b	Multiplica y asigna
\$a /= \$b	\$a = \$a / \$b	Divide y asigna
\$a %= \$b	\$a = \$a % \$b	Módulo y asigna
\$a .= \$b	\$a = \$a . \$b	Concatena y asigna
\$a++	\$a = \$a + 1	Incrementa en 1
\$a--	\$a = \$a - 1	Decrementa en 1

■■ **PRIORIDAD:** Paréntesis → Negación → Multiplicación/División → Suma/Resta → Comparación → Lógicos → Asignación

5. Trabajando con Formularios

Los datos se envían via URL: **var1=valor1&var2=valor2**

Proceso en 2 pasos:

1. Generar formulario HTML con **action='archivo.php' method='GET'**
2. En archivo.php leer datos con **\$_GET['nombreVar']**

```
<!-- saluda.html -->
<form action="saluda.php" method="get">
  <label for="nombre">Nombre:</label>
  <input type="text" name="nombre" id="nombre">
  <input type="submit" value="enviar">
</form>
```

```
<?php
// saluda.php
$nombre = $_GET["nombre"];
$apellido = $_GET["apellido"];
echo "Hola $nombre $apellido";
?>
```

■■ **BUENA PRÁCTICA:** Separar siempre HTML de PHP. Más adelante veremos POST además de GET

6. Condiciones

6.1 Condición Simple - if

```
<?php
$hora = 8;
if ($hora === 8) {
    echo "Suenan el despertador";
}
?>
```

■ ■ **RECOMENDACIÓN:** Usar siempre llaves {} aunque sea una sola instrucción

6.2 Condición Compuesta - if-else

```
<?php
$hora = 17;
if ($hora <= 12) {
    echo "Buenos días";
} else {
    echo "Buenas tardes";
}
?>
```

6.3 Condiciones Múltiples - if-else if-else

```
<?php
$hora = 14;
if ($hora === 8) {
    echo "Hora de desayunar";
} else if ($hora === 14) {
    echo "Hora de comer";
} else if ($hora === 21) {
    echo "Hora de cenar";
} else {
    echo "Ahora no toca comer";
}
?>
```

6.4 Switch

```
<?php
$hora = 14;
switch ($hora) {
    case 8:
        echo "Hora de desayunar";
        break;
    case 14:
        echo "Hora de comer";
        break;
    case 21:
        echo "Hora de cenar";
        break;
    default:
        echo "Ahora no toca comer";
}
?>
```

■ ■ **¡NO OLVIDES EL BREAK!** Sin break, ejecutará el siguiente caso automáticamente

6.5 Operador Ternario

Sintaxis: **condición ? valorTrue : valorFalse**

```
<?php
```

```
$hora = 14;  
$formato = ($hora > 12) ? 24 : 12;  
echo "El formato es de $formato horas";  
?>
```

6.6 Operador Elvis (?:)

Para valores por defecto: **expresión ? : valorSiVacio**

```
<?php  
$nombre = $_GET['nombre'] ?: "desconocido";  
?>
```

7. Bucles

7.1 Bucle while

```
<?php
$i = 1;
while ($i <= 10) {
    echo "Línea $i<br>";
    $i++;
}
?>
```

7.2 Bucle do-while

Se ejecuta AL MENOS UNA VEZ antes de comprobar la condición

```
<?php
do {
    $dado = rand(1, 6);
    echo "Ha salido un $dado<br>";
} while ($dado != 5);
echo "¡Saco ficha!";
?>
```

7.3 Bucle for

```
<?php
// Ascendente
for ($i = 1; $i <= 10; $i++) {
    echo "Línea $i<br>";
}

// Descendente
for ($i = 10; $i >= 0; $i--) {
    echo "Línea $i<br>";
}
?>
```

7.4 Break y Continue

- **break:** Sale del bucle completamente
- **continue:** Salta a la siguiente iteración

```
<?php
for ($i = 1; $i <= 10; $i++) {
    if ($i == 5) {
        break; // Sale cuando i=5
    }
    echo "$i ";
}
// Imprime: 1 2 3 4

for ($i = 1; $i <= 5; $i++) {
    if ($i == 3) {
        continue; // Salta el 3
    }
    echo "$i ";
}
// Imprime: 1 2 4 5
?>
```

■ ■ **ALTERNATIVA:** Es preferible usar variables flag en lugar de break/continue para mejor legibilidad

7.5 Resumen de Bucles

Bucle	Cuándo usar
while	Cuando no sabes cuántas veces se repetirá
do-while	Cuando necesitas ejecutar AL MENOS una vez
for	Cuando sabes el número exacto de iteraciones
foreach	Para recorrer arrays (lo veremos después)



FIN DE LA PARTE 1

La Parte 2 incluirá: Arrays, Funciones y Funciones Predefinidas

