INSTITUTO KHIPU

Semestre V

Sesión 04

PROGRAMA DE ESTUDIOS

DESARROLLO DE SISTEMAS DE INFORMACIÓN

DESARROLLO DE BACKEND

Tema

MANEJO DE OPERADORES (Y PRECEDENCIA) EN EL LENGUAJE PHP. EJEMPLOS Y PRÁCTICA.

MANEJO DE OPERADORES Y PRECEDENCIA EN PHP

¿QUÉ ES UN OPERADOR?

Los operadores son los símbolos que nos permiten expresar las operaciones entre los datos en PHP al crear una tienda o página web. Con los operadores podemos realizar operaciones entre uno o más valores. Toman unos valores de entrada, los relaciona entre sí y muestran un resultado. Los valores que relacionamos pueden estar dentro de variables o escritos dentro del propio código que estemos programando. En ocasiones, las operaciones serán matemáticas y en otras, hablamos de operaciones lógicas o de asignación.

Tipos de Operadores en PHP

PHP ofrece una amplia gama de operadores, que se pueden clasificar en diferentes categorías:

1. Operadores Aritméticos

Se usan para realizar operaciones matemáticas.

Operador	Descripción	Ejemplo	Resultado
`+`	Suma	`5 + 3`	8
`-`	Resta	`5 - 3`	2
`*`	Multiplicación	`5 * 3`	15
`/`	División	`10 / 2`	5
`%`	Módulo (resto)	`10 % 3`	1
`**`	Potencia	`2 ** 3`	8

```php
<?php
$a = 10; $b = 3;
echo $a + $b; // 13
echo $a % $b; // 1
?>
```

2. Operadores de asignación

Sirven para asignar valores a variables.

```php
<?php
$x = 5;   // Asignación
$x += 3;  // $x = $x + 3 → 8
$x *= 2;  // $x = $x * 2 → 16
?>
```

3. Operadores de comparación

Comparan dos valores y devuelven true o false.

Operador	Descripción
`==`	Igualdad
`===`	Igualdad estricta (mismo valor y tipo)
`!=` o `<>`	Diferente
`!==`	Diferente estricta
`>` , `<`	Mayor, menor
`>=` , `<=`	Mayor o igual, menor o igual

```php
<?php
var_dump(5 == "5");   // true
var_dump(5 === "5");  // false
?>
```

4. Operadores lógicos

Se usan para combinar condiciones.

Operador	Significado
`&&` o `and`	Y lógico
`	
`!`	Negación

```php
<?php
$edad = 20;
$licencia = true;
if ($edad >= 18 && $licencia) {
    echo "Puede conducir";
}
?>
```

5. Operadores de concatenación

Unen cadenas de texto.

```php
<?php
$nombre = "Ana";
$apellido = "López";
echo $nombre . " " . $apellido; // Ana López
?>
```

6. Operadores de incremento y decremento

```php
<?php
$x = 5;
echo ++$x; // pre-incremento: 6
echo $x--; // post-decremento: muestra 6, luego pasa a 5
?>
```

PRECEDENCIA DE OPERADORES

La precedencia de operadores determina el orden en que se evalúan las operaciones en una expresión. En PHP, como en matemáticas, los operadores de mayor precedencia se evalúan antes que los de menor.

Aquí tienes una tabla simplificada de precedencia (de mayor a menor)

Ejemplo Práctico de Precedencia:

Considera la siguiente expresión:

```php
$resultado = 5 + 3 * 2;
```

Si no tuvieras en cuenta la precedencia, podrías pensar que el resultado es 16 (porque 5 + 3 es 8, y 8 * 2 es 16). Sin embargo, PHP sigue las reglas:

1. Primero, evalúa la multiplicación (`*`).
2. `3 * 2` es igual a `6` .
3. Luego, evalúa la suma (`+`).
4. `5 + 6` es igual a `11` .

Por lo tanto, `$resultado` será `11` .

Práctica: Controlando el Orden con Paréntesis

Cuando necesitas forzar un orden de evaluación diferente al predeterminado, usa paréntesis (). Los paréntesis son la herramienta más poderosa para controlar la precedencia.

Ejemplo de Práctica:

¿Qué pasaría si realmente quisieras que la suma se hiciera antes que la multiplicación en el ejemplo anterior?

```php
// Con paréntesis, se fuerza la suma a evaluarse primero
$resultado = (5 + 3) * 2;

// 1. (5 + 3) se evalúa primero, dando 8
// 2. 8 * 2 se evalúa después, dando 16

echo $resultado; // Salida: 16
```

Aquí, $resultado es 16, porque los paréntesis obligaron a PHP a realizar la suma antes de la multiplicación.

Usa paréntesis (): Aunque las reglas de precedencia son claras, usar paréntesis en expresiones complejas mejora enormemente la legibilidad del código. Es mejor ser explícito que dejarlo a la interpretación.

Conoce los operadores: Familiarízate con los operadores que usas con más frecuencia. Un buen manejo de los operadores te permite escribir código más conciso y efectivo.

Precaución con la igualdad: Recuerda la diferencia entre el operador de igualdad == (compara solo el valor) y el de identidad === (compara el valor y el tipo de dato). Usar === es una buena práctica en la mayoría de los casos para evitar errores inesperados.

INSTITUTO

KHIPU