Tipos de datos y operadores

Tipos de datos

Entender los tipos de datos en PHP es fundamental para declarar variables, asignarles valores y utilizarlos correctamente en nuestras aplicaciones. Si bien ya hemos aprendido cómo declarar variables, asignarles valores e imprimirlos, es crucial comprender qué tipo de dato estamos estableciendo en cada variable y cuándo debemos utilizar cada uno de ellos.

PHP proporciona una amplia gama de tipos de datos, pero en esta sección nos centraremos principalmente en los tipos de datos simples, destacados en color, dejando los tipos más complejos para futuras secciones.

Tipo de dato	Descripción	Ejemplo
boolean	Representa un valor verdadero o falso.	$activo = true;
integer	Representa un número entero.	$edad = 25;
float	Representa un número de punto flotante.	$precio = 19.99;
string	Representa una cadena de caracteres.	$nombre = "Nuria";
array	Representa una colección de elementos.	$numeros = [1, 2, 3, 4, 5];
object	Representa una instancia de una clase.	$usuario = new Usuario();
null	Representa un valor nulo.	$dato = null;
resource	Representa un recurso externo.	$archivo = fopen("archivo.txt");
callable	Representa un tipo de dato invocable.	$funcion = function() {};
iterable	Representa una estructura iterativa.	$lista = new ArrayIterator();

- **boolean**: Representa un valor verdadero o falso. Por ejemplo, podemos usarlo para indicar si un usuario ha iniciado sesión ($conectado = true;).
- **integer**: Representa un número entero. Podemos utilizarlo para almacenar edades, cantidades, entre otros ($edad = 25;).

- **float**: Representa un número decimal o de punto flotante. Es útil para valores que requieren precisión decimal, como precios o coordenadas ($precio = 19.99;).
- **string**: Representa una cadena de caracteres. Usamos este tipo de dato para almacenar texto, nombres, mensajes, entre otros ($nombre = "Nuria";).
- **null**: Representa un valor nulo o la ausencia de valor. Puede ser utilizado para indicar que una variable no tiene un valor asignado ($dato = null;).

Operadores

os operadores en PHP son herramientas clave que nos permiten manipular y realizar diversas operaciones en nuestros programas. Son caracteres especiales que indican al intérprete de PHP qué tipo de operación debe realizar entre los operandos.

Aritméticos

Estos operadores aritméticos nos permiten realizar diversas operaciones matemáticas en PHP. Puedes utilizar estos operadores junto con variables o valores numéricos para realizar cálculos y asignar los resultados a variables.

Operador	Descripción	Ejemplo
`+`	Suma	`$resultado = $a + $b;`
`-`	Resta	`$resultado = $a - $b;`
`*`	Multiplicación	`$resultado = $a * $b;`
`/`	División	`$resultado = $a / $b;`
`%`	Módulo (resto de la división)	`$resultado = $a % $b;`
`**`	Exponenciación	`$resultado = $a ** $b;`
`++`	Incremento	`$a++;`
`--`	Decremento	`$a--;`

Asignación

Los operadores de asignación nos permiten asignar y modificar valores en variables de manera más eficiente.

Operador	Descripción	Ejemplo
`=`	Asignación	`$a = 5;`
`+=`	Suma y asignación	`$a += 3; // Equivale a $a = $a + 3;`
`-=`	Resta y asignación	`$a -= 2; // Equivale a $a = $a - 2;`
`*=`	Multiplicación y asignación	`$a *= 4; // Equivale a $a = $a * 4;`
`/=`	División y asignación	`$a /= 2; // Equivale a $a = $a / 2;`
`%=`	Módulo y asignación	`$a %= 3; // Equivale a $a = $a % 3;`

Operador	Descripción	Ejemplo
`.=`	Concatenación y asignación	`$texto .= " mundo"; // Equivale a $texto = $texto . " mundo";`
`<<=`	Desplazamiento a la izquierda y asignación	`$a <<= 2; // Equivale a $a = $a << 2;`
`>>=`	Desplazamiento a la derecha y asignación	`$a >>= 1; // Equivale a $a = $a >> 1;`
`&=`	AND a nivel de bits y asignación	`$a &= $b; // Equivale a $a = $a & $b;`
`^=`	XOR a nivel de bits y asignación	`$a ^= $b; // Equivale a $a = $a ^ $b;`
`|=`	OR a nivel de bits y asignación	`$a |= $b; // Equivale a $a = $a | $b;`

Comparación

Los operadores de comparación nos permiten evaluar y comparar valores en PHP. Puedes utilizar estos operadores en combinación con variables y valores para realizar comparaciones y tomar decisiones basadas en el resultado.

Operador	Descripción	Ejemplo
`==`	Igual a	`$a == $b`
`!=`	Diferente de	`$a != $b`
`<`	Menor que	`$a < $b`
`>`	Mayor que	`$a > $b`
`<=`	Menor o igual que	`$a <= $b`
`>=`	Mayor o igual que	`$a >= $b`
`===`	Identidad	`$a === $b`
`!==`	No identidad	`$a !== $b`
`<=>`	Comparación combinada	`$a <=> $b`

Incremento / Decremento

Los operadores de preincremento y predecremento se utilizan para incrementar o decrementar el valor de una variable en 1 antes de utilizarla en una expresión. Por otro lado, los operadores de postincremento y postdecremento incrementan o decrementan el valor de una variable en 1 después de utilizarla en una expresión.

Operador	Descripción	Ejemplo
`++`	Postincremento (Incremento)	`$a++;`
`--`	Postdecremento (Decremento)	`$a--;`

Operador	Descripción	Ejemplo
`++`	Preincremento (Incremento)	`++$a;`
`--`	Predecremento (Decremento)	`--$a;`

Lógicos

Estos operadores lógicos se utilizan para combinar y evaluar condiciones lógicas en expresiones condicionales. Puedes utilizar estos operadores lógicos para construir expresiones más complejas y realizar evaluaciones condicionales en tu código PHP.

Operador	Descripción	Ejemplo
`&&`	AND lógico	`$a && $b`
`\\`	OR lógico	`$a || $b`
`and`	AND lógico (alternativo)	`$a and $b`
`or`	OR lógico (alternativo)	`$a or $b`
`xor`	XOR lógico	`$a xor $b`
`!`	NOT lógico	`!$a`

de String

Operador	Descripción	Ejemplo
`.`	Concatenación	`$a . $b`
`.=`	Concatenación y asignación	`$a .= $b`

de Array

Condicionales

Operador	Descripción	Ejemplo
`?:`	Ternario true/false	`$x = expresion1 ? expresion2 : expresion3`
`.=`	Ternario is Null	`$x = expresion1 ?? expresion2`