1.2. Variables, tipos de datos y operadores

LENGUAJES DE PROGRAMACIÓN NIVEL 4. UNIDAD 1

PHP - LENGUAJE DE PROGRAMACION PARA EL BACKEND

Financiado por la Unión Europea

Plan de Recuperación, Transformación y Resiliencia

Contenido

Introducción

En el módulo anterior, exploramos los cimientos de PHP, sumergiéndonos en su sintaxis básica y estructura de un archivo PHP. Aprendimos cómo abrir y cerrar bloques de código PHP y la importancia de los comentarios para mantener nuestro código comprensible.

Ese conocimiento nos proporcionó la base necesaria para comenzar a escribir scripts simples en PHP y nos preparó para los conceptos más avanzados que abordaremos en este módulo.

La piedra angular de la programación en PHP

Al adentrarnos en el mundo de PHP, rápidamente nos encontramos con tres conceptos fundamentales que son esenciales para casi todos los programas que escribiremos: variables, tipos de datos y operadores.

- Variables: Piensa en las variables como contenedores que almacenan información. En PHP, una variable puede contener diferentes tipos de datos en diferentes momentos durante la ejecución de tu programa, lo que las hace increíblemente versátiles y poderosas.
- Tipos de Datos: Los tipos de datos definen la naturaleza de la información que podemos almacenar en una variable. PHP es un lenguaje de tipado dinámico, lo que significa que las variables no están ligadas a un único tipo de dato. Sin embargo, entender los diferentes tipos de datos (como strings, números y booleanos) es crucial para el manejo efectivo de la información.
- Operadores: Los operadores nos permiten realizar operaciones sobre nuestras variables y valores. Desde realizar cálculos matemáticos hasta comparar valores y tomar decisiones lógicas, los operadores son las herramientas que utilizamos para manipular datos y controlar el flujo de nuestros programas.

Bloques de construcción de la lógica del programa

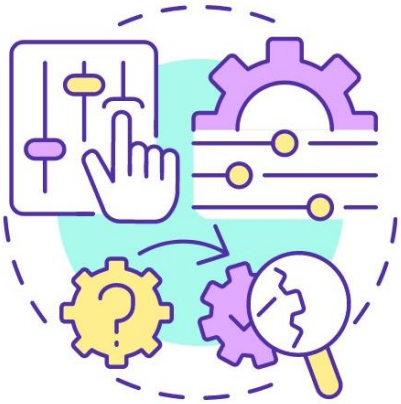
Las variables, tipos de datos y operadores son mucho más que meras características sintácticas de PHP; son los componentes esenciales que utilizamos para construir la lógica y el flujo de nuestros programas. Cada script PHP que escribimos, cada aplicación web que desarrollamos, se basa

en estos elementos fundamentales para almacenar, procesar y tomar decisiones sobre la información.

- Almacenamiento y Recuperación de Datos: Usamos variables para capturar y retener datos que nuestro programa puede necesitar en varios puntos durante su ejecución.
- Procesamiento de Datos: A través de una combinación de tipos de datos y operadores, transformamos y manipulamos estos datos para lograr los resultados deseados.
- Control de Flujo: Los operadores, especialmente los de comparación y lógicos, son cruciales para controlar cómo y cuándo se ejecutan diferentes partes de nuestro código en respuesta a ciertas condiciones.

Al dominar estos conceptos fundamentales, no solo estarás equipado para abordar una amplia gama de problemas de programación en PHP, sino que también establecerás una base sólida sobre la cual podrás construir a medida que te adentres en aspectos más complejos del desarrollo web. ***Te animo a experimentar con variables, tipos de datos y operadores en tus propios scripts PHP, explorando las posibilidades que ofrecen y cómo pueden servir como los bloques de construcción para tus ideas y proyectos.***

Variables en PHP



En cualquier lenguaje de programación, las variables son fundamentales para almacenar y trabajar con datos. En PHP, las variables son particularmente poderosas y flexibles, dada la naturaleza dinámica del lenguaje.

Este módulo profundiza en qué son las variables en PHP, cómo se declaran, las reglas para nombrarlas y su importancia en la construcción de aplicaciones web dinámicas.

¿Qué son las variables?

Una variable en PHP es un contenedor para almacenar información que puede ser modificada y utilizada a lo largo de tu programa. Piensa en una variable como una caja donde puedes guardar cosas, con la ventaja de que puedes cambiar el contenido en cualquier momento.

Las variables permiten a los programadores almacenar datos temporales, como resultados de cálculos, cadenas de texto, o información del usuario, que pueden ser accedidos y manipulados por el programa según sea necesario.

Reglas para nombrar variables en PHP

Todas las variables en PHP comienzan con un signo de dólar ($), seguido por el nombre de la variable. Por ejemplo, $miVariable.

Reglas de Nomenclatura:

- El nombre de una variable debe comenzar con una letra o un guion bajo (_), no un número.
- Los nombres de variables pueden contener letras, números y guiones bajos, pero no espacios ni caracteres especiales.
- PHP distingue entre mayúsculas y minúsculas, por lo que $variable y $Variable serían consideradas variables diferentes.

Importancia de la Nomenclatura:

- Elegir nombres significativos y consistentes para tus variables es crucial para la legibilidad y mantenibilidad del código. Un buen nombre de variable debe indicar claramente para qué se usa la variable.

Declaración de variables y asignación de valores

Aprender a declarar variables y asignarles valores de manera efectiva es, por lo tanto, un paso crucial en la escritura de código PHP claro, eficiente y funcional.

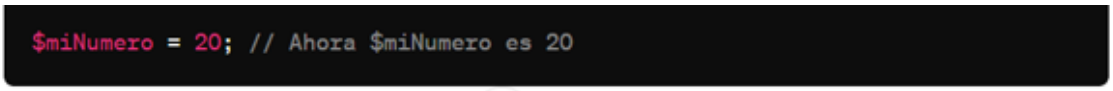
- Declaración:
  En PHP, declaras una variable simplemente asignándole un valor. No necesitas declarar explícitamente el tipo de dato de la variable debido al tipado dinámico de PHP.

```php
$miNumero = 10;
$miCadena = "Hola, mundo";
```

[1]

[1] Accesibilidad
$miNumero = 10;
miCadena = "Hola, mundo";

- Asignación de Valores:
  Puedes cambiar el valor de una variable simplemente asignándole un nuevo valor. PHP sobrescribirá el valor anterior con el nuevo.



2

Uso de variables

- Almacenamiento de Información: Las variables son utilizadas para almacenar información que tu programa puede necesitar utilizar en varios puntos, como entradas de usuario, configuraciones, o resultados de cálculos intermedios.
- Actualización de Datos: Una de las grandes ventajas de las variables es que su contenido puede ser actualizado o modificado basado en la lógica de tu programa, permitiendo una gran flexibilidad en el manejo de datos.

[2] Accesibilidad
$miNumezo = 20; //Ahora $miNumezo es 20

Tipos de datos en PHP

Al adentrarnos más en PHP, es crucial entender los tipos de datos que el lenguaje nos ofrece para representar y trabajar con información de diversas maneras. Cada tipo de dato en PHP tiene su propósito y se utiliza en situaciones específicas, lo que afecta directamente cómo declaramos nuestras variables y cómo operamos con ellas.

Este módulo te guiará a través de los tipos de datos fundamentales en PHP, desde simples cadenas y números hasta estructuras más complejas como arrays y objetos, y te mostrará cómo el tipado dinámico de PHP añade flexibilidad a tu código.

Tipos de datos

En general, distinguimos tres tipos de datos: Básicos, Compuestos y Especiales

Veamos cada uno de ellos con más detalle:

Tipos de datos básicos

- String:
  Representa secuencias de caracteres y se utiliza para almacenar texto. Las cadenas se pueden definir usando comillas simples o dobles.

```
$saludo = "Hola, mundo!";
```

3

- Integer:
  Se utiliza para representar números enteros, sin decimales. Útil para contar o realizar cálculos matemáticos.

```
$edad = 25;
```

4

- Float (o double):
  Representa números con decimales. Esencial para cálculos que requieren precisión más allá de los enteros.

```
$precio = 9.99;
```

5

[3] Accesibilidad:
$saludo = ¡Hola, mundo!";
[4] Accesibilidad
$edad = 25;
[5] Accesiilidad
$precio = 9.99;

- Boolean:
  Este tipo tiene solo dos valores posibles: true (verdadero) o false (falso). Los booleanos son fundamentales para las condiciones y decisiones lógicas en el código.

```php
$esCliente = true;
```

6

Tipos de datos compuestos

- Array:
  Un array almacena múltiples valores en una sola variable. Es útil para listas de elementos o cuando deseas almacenar datos relacionados juntos.

```php
$frutas = array("Manzana", "Naranja", "Plátano");
```

7

- Object:
  Los objetos representan instancias de clases[8], permitiendo la programación orientada a objetos (OOP) en PHP. Los objetos son fundamentales para estructurar código complejo y funcionalidades.

```php
$miCoche = new Coche();
```

9

[6] Accesibilidad
$esCliente = True;
[7] Accesibilidad
$frutas = array(“Manzana”, “Naranja”, “Plátano”);
[8] Una instancia de clase en programación orientada a objetos (OOP) se refiere a un objeto concreto creado a partir de una clase. La clase actúa como una especie de plano o plantilla que define las propiedades (atributos) y comportamientos (métodos) que los objetos basados en esa clase tendrán. Cuando se crea una instancia de una clase, estás creando un objeto único que tiene las características y capacidades definidas en la clase.
[9] Accesibilidad
$miCoche = new Coche();

Tipos especiales

- NULL:
  Indica que una variable no tiene valor. Se utiliza para representar una variable "vacía" o "sin definir".

```php
$nombre = NULL;
```

10

Tipos de datos dinámicos en PHP

PHP es un lenguaje de programación de tipado dinámico, lo que significa que las variables no están vinculadas a un tipo de datos específico. Esto permite que una variable que inicialmente almacena, por ejemplo, una cadena de texto pueda luego reasignarse para almacenar un número, un booleano, o cualquier otro tipo de dato.

Esta flexibilidad es una poderosa característica de PHP, pero también requiere que los desarrolladores sean cuidadosos para evitar errores y comportamientos inesperados en su código.

[10] Accesibilidad
$nombre = NULL;

Operadores en PHP

Tras haber explorado las variables y tipos de datos en PHP, es esencial entender cómo podemos manipular esos datos y tomar decisiones lógicas en nuestros programas. Aquí es donde entran en juego los operadores. Los operadores nos permiten realizar operaciones matemáticas, comparar valores, asignar datos a variables y mucho más.

Este módulo desglosará los tipos de operadores en PHP y te mostrará cómo y cuándo usarlos para construir la lógica de tus aplicaciones web.

Operadores aritméticos

Los operadores aritméticos son utilizados para realizar operaciones matemáticas comunes.

Los principales son los siguientes:

- Suma (+): Suma dos operandos.

```php
$suma = 5 + 3;  // Resultado: 8
```

11

- Resta (-): Resta el segundo operando del primero.
- Multiplicación (*): Multiplica dos operandos.
- División (/): Divide el primer operando entre el segundo.
- Módulo (%): Obtiene el residuo de la división del primer operando entre el segundo.

[11] Accesibilidad
$suma = 5 + 3; // Resultado: 8

Operadores de asignación

Los operadores de asignación se utilizan para asignar valores a las variables.

Los principales son:

- Asignación (=): Asigna el valor del operando derecho al operando izquierdo.

```
$numero = 5;
```

12

- Asignación con operación (+=, -=, *=, /=, %=): Realiza una operación y asigna el resultado.

```
$numero += 3;  // Equivale a $numero = $numero + 3;
```

13

[12] Accesibilidad
$numero = 5;
[13] Accesibilidad
$numero += 3; // Equivale a $numero = $numero + 3

Operadores de comparación y lógicos

Estos operadores son cruciales para tomar decisiones y controlar el flujo del programa.

- Igual (==): Verifica si dos valores son iguales.
- Idéntico (===): Verifica si dos valores son iguales y del mismo tipo.
- Diferente (!= o <>): Verifica si dos valores no son iguales.
- No idéntico (!==): Verifica si dos valores no son iguales o no son del mismo tipo.
- Mayor que (>), Menor que (<), Mayor o igual que (>=), Menor o igual que (<=).
- AND lógico (&& o and): Verdadero si ambos operandos son verdaderos.
- OR lógico (|| o or): Verdadero si al menos uno de los operandos es verdadero.
- NOT lógico (!): Invierte el valor de verdad del operando.

Ejemplos prácticos

```php
// Uso de operadores de asignación
$contador = 1;
$contador += 5;  // Incrementa $contador en 5

// Uso de operadores de comparación y lógicos para control de flujo
if ($edad >= 18 && $tieneLicencia) {
    echo "Puedes conducir.";
} else {
    echo "No cumples con los requisitos para conducir.";
}
```

14

[14] Accesibilidad

```
// Uso de operadores de asignación
$contador = 1;
$contador += 5; // Incrementa $contador en 5

// Uso de operadores de comparación y lógicos para control de flujo
if ($edad >= 18 88 &&  $tieneLicencia) {
    echo "Puedes conducir.";
} else {
    echo "No cumples con los requisitos para conducir.";
}
```

Buenas prácticas

A medida que avanzamos en nuestro viaje de aprendizaje en PHP, es esencial no solo comprender cómo escribir código, sino también cómo escribirlo bien.

Este módulo se centra en algunas buenas prácticas clave que te ayudarán a aprovechar al máximo las variables, los tipos de datos y los operadores en PHP. Adoptar estas prácticas no solo mejorará la calidad y legibilidad de tu código, sino que también te preparará para trabajar en proyectos más grandes y colaborativos.

Uso eficiente de variables y tipos de datos

- Nomenclatura Clara y Descriptiva:
  Elige nombres de variables que sean claros y descriptivos para que cualquier persona (incluido tú en el futuro) pueda entender fácilmente qué almacenan.
  Por ejemplo, usa $numeroDeUsuarios en lugar de $n.
- Selección Adecuada de Tipos de Datos:
  Selecciona el tipo de dato más adecuado para la información que necesitas almacenar. Esto no solo hace que tu código sea más comprensible, sino que también puede tener implicaciones en el rendimiento.
  Por ejemplo, si una variable solo va a almacenar valores verdadero o falso, úsala como booleano.

- Inicialización de Variables:
  Siempre inicializa tus variables antes de usarlas. Esto ayuda a prevenir errores y comportamientos inesperados en tu código.

Optimización del uso de operadores

- Simplicidad y Claridad:
  Utiliza operadores de manera que tu código sea fácil de leer y entender.
  Por ejemplo, usa operadores de incremento/decremento (++, --) cuando sea apropiado para simplificar tu código.
- Operadores de Comparación Estricta:
  Prefiere los operadores de comparación estricta (===, !==) a los operadores de comparación suelta (==, !=). Esto asegura que tanto el valor como el tipo de dato coincidan, evitando errores comunes relacionados con la conversión automática de tipos.
- Uso Consciente de Operadores Lógicos:
  En las expresiones lógicas, organiza las condiciones de manera que las más probables de fallar se evalúen primero. Esto puede aprovechar el cortocircuito en operadores como && y || para mejorar la eficiencia.

Adherirse a buenas prácticas al trabajar con variables, tipos de datos y operadores no solo mejora la calidad de tu código, sino que también facilita la colaboración con otros desarrolladores y hace que tu código sea más fácil de mantener y escalar. A medida que continúes explorando PHP, estas prácticas se convertirán en hábitos naturales que enriquecerán tu conjunto de habilidades de programación y te prepararán para enfrentar desafíos más complejos.

Conclusión

A lo largo de este módulo, hemos desentrañado los fundamentos críticos de PHP: las variables, tipos de datos y operadores. Estos conceptos forman el tejido conectivo de casi todos los programas PHP que escribirás, sirviendo como las herramientas básicas que necesitas para almacenar, manipular y evaluar datos dentro de tus aplicaciones. Entender cómo y cuándo utilizar cada uno de estos elementos es crucial para cualquier programador PHP, no solo para construir aplicaciones que funcionen como se espera, sino también para asegurar que tu código sea claro, eficiente y fácil de mantener.

La verdadera comprensión de estos conceptos fundamentales viene con la práctica. Te animo a que experimentes con variables, explores los diferentes tipos de datos y apliques operadores en una variedad de contextos. Intenta crear pequeños proyectos o ejercicios prácticos que te desafíen a utilizar lo que has aprendido de maneras nuevas e interesantes. Cada línea de código que escribes y cada error que encuentras y solucionas te acerca un paso más a convertirte en un desarrollador PHP competente y seguro.

Los conceptos que hemos cubierto son la base sobre la cual se construyen temas más avanzados en PHP. A medida que avanzamos en el curso, verás cómo estas herramientas básicas se aplican y extienden en estructuras de control como condicionales y bucles, cómo se utilizan dentro de funciones para crear código modular y reutilizable, y cómo interactúan con bases de datos y otras tecnologías para crear aplicaciones web completas. Cada nuevo tema que explores se basará en estos fundamentos, ampliando tu capacidad para crear soluciones más complejas y efectivas.

Al cerrar este capítulo de tu viaje de aprendizaje en PHP, espero que te sientas inspirado por las posibilidades que estos conocimientos fundamentales abren para ti. Recuerda que cada gran desarrollador comenzó dominando estos mismos conceptos básicos, y con tiempo, práctica y curiosidad, no hay límite para lo que puedes lograr. Continúa experimentando, construyendo y aprendiendo; el mundo de PHP está lleno de oportunidades para aquellos dispuestos a explorarlo.

¡Adelante y feliz codificación!

Recursos adicionales

El aprendizaje y la maestría en PHP, como en cualquier lenguaje de programación, se benefician enormemente de una variedad de recursos y perspectivas. Este módulo te guiará hacia recursos adicionales que te permitirán profundizar en tu comprensión de las variables, los tipos de datos y los operadores en PHP. Desde la documentación oficial hasta tutoriales interactivos y comunidades en línea, estos recursos serán invaluables en tu viaje de aprendizaje.

Documentación oficial de PHP

- PHP.net: La documentación oficial de PHP es el recurso definitivo para todos los programadores de PHP. Para temas específicos:
    - Variables (https://www.php.net/manual/en/language.variables.variable.php)
    - Tipos de Datos (https://www.php.net/manual/es/language.types.php)
    - Operadores (https://www.php.net/manual/es/language.operators.php)

Recursos educativos para principiantes

- W3Schools PHP Tutorial: W3Schools ofrece tutoriales introductorios que son fáciles de seguir y cubren una amplia gama de temas en PHP, perfectos para principiantes.
  https://www.w3schools.com/php/
- Codecademy's PHP Course: Codecademy ofrece un curso interactivo de PHP que te guía a través de los conceptos básicos con ejercicios prácticos, ideal para consolidar lo aprendido.
  https://www.codecademy.com/catalog/language/php4
- PHP: The Right Way: PHP: The Right Way es una guía de buenas prácticas popular que ofrece una visión general de PHP, incluyendo recomendaciones sobre estilos de codificación, prácticas recomendadas y enlaces a recursos de calidad.
  https://phptherightway.com/

Comunidades y foros

- Reddit r/PHP: El subreddit r/PHP es un buen lugar para mantenerse al día con las noticias de PHP, hacer preguntas y participar en discusiones con otros desarrolladores PHP.
  https://www.reddit.com/r/PHP/?rdt=45690

- PHP Freaks: PHP Freaks es un foro donde puedes encontrar respuestas a tus preguntas, participar en discusiones y conectarte con otros desarrolladores PHP.
  http://www.phpfreaks.com/

Expandir tu aprendizaje más allá del aula es crucial para convertirte en un desarrollador PHP competente y versátil. La documentación oficial te proporcionará una base sólida, mientras que los tutoriales interactivos y los ejercicios prácticos reforzarán tu comprensión y habilidades. Participar en comunidades te conectará con otros aprendices y profesionales, ofreciéndote apoyo y perspectivas adicionales. ***Aprovecha estos recursos para profundizar en tu conocimiento de PHP y llevar tus habilidades de programación al siguiente nivel.***