GUIA BASICA DE PHP

Manual de PHP Básico

Introducción

PHP (Hypertext Preprocessor) es un lenguaje de programación de código abierto y de alto nivel, diseñado especialmente para generar contenido dinámico para la web. En este manual, te presentaremos los conceptos básicos de PHP, incluyendo variables, tipos de datos, operadores, control de flujo, funciones, estructuras de datos, entrada y salida, condicionales, bucles, errores y excepciones, comentarios y identificadores.

Variables

En PHP, las variables se declaran con un nombre y se inicializan con un valor. Las variables se utilizan para almacenar y manipular datos.

Ejemplo:

```
<?php
  $nombre = 'Juan';
  $edad = 30;
?>
```

Tipos de datos

PHP admite los siguientes tipos de datos:

- Enteros (int): números enteros, como 1, 2, 3, etc.
- Flotantes (float): números decimales, como 3.14 o -0.5.
- Cadena (string): secuencias de caracteres, como 'Hola' o "Adiós".
- Booleano (bool): valores lógicos, como true o false.
- Arreglo (array): colecciones de valores, como un conjunto de números o cadenas.
- Objetos (object): instancias de clases, que pueden contener propiedades y métodos.

Operadores

PHP admite los siguientes operadores:

- Aritméticos: `+`, `-`, `*`, `/`, `%`, etc.
- Comparativos: `==`, `!=`, `===`, `!==`, `<`, `>`, `<=`, `>=`, etc.
- Lógicos: `&&`, `||`, `!`, etc.
- Asignación: `=`, `+=`, `-=`, `*=`, `/=`, etc.

Ejemplo:

```
<?php
  $a = 5;
  $b = 3;
  echo $a + $b; // Output: 8
?>
```

Control de flujo

PHP admite las siguientes estructuras de control de flujo:

- If-else: `if (condición) { código } else { código }`
- Switch: `switch (valor) { case valor1: código; break; case valor2: código; break; ... }`
- For: `for (inicialización; condición; incremento) { código }`
- While: `while (condición) { código }`
- Do-while: `do { código } while (condición);`

Ejemplo:

```
<?php
  $edad = 25;
  if ($edad >= 18) {
    echo 'Eres mayor de edad.';
  } else {
    echo 'Eres menor de edad.';
  }
?>
```

Funciones

PHP admite la definición de funciones, que son bloques de código que se pueden llamar varias veces desde diferentes partes del programa.

Ejemplo:

```
<?php
  function saludar($nombre) {
    echo 'Hola, ' . $nombre . '!';
  }
  saludar('Juan'); // Output: Hola, Juan!
?>
```

Estructuras de datos

PHP admite las siguientes estructuras de datos:

- Arreglos (arrays): colecciones de valores, como un conjunto de números o cadenas.
- Objetos (objects): instancias de clases, que pueden contener propiedades y métodos.

Ejemplo:

```
<?php
  $arreglo = array('Juan', 'Maria', 'Pedro');
  echo $arreglo[0]; // Output: Juan
?>
```

Entrada y salida

PHP admite las siguientes funciones de entrada y salida:

- `echo`: imprime un valor en la salida estándar.
- `print`: imprime un valor en la salida estándar.
- `printf`: imprime un valor en la salida estándar con formato.
- `scanf`: lee un valor desde la entrada estándar.
- `input`: lee un valor desde la entrada estándar.

Ejemplo:

```
<?php
  echo 'Hola, mundo!';
  $nombre = input('¿Cuál es tu nombre?');
  echo 'Hola, ' . $nombre . '!';
?>
```

Condicionales

PHP admite las siguientes estructuras condicionales:

- If-else: `if (condición) { código } else { código }`
- Switch: `switch (valor) { case valor1: código; break; case valor2: código; break; ... }`

Ejemplo:

```
<?php
  $edad = 25;
  if ($edad >= 18) {
    echo 'Eres mayor de edad.';
  } else {
    echo 'Eres menor de edad.';
  }
?>
```

Bucles

PHP admite las siguientes estructuras de bucles:

- For: `for (inicialización; condición; incremento) { código }`
- While: `while (condición) { código }`
- Do-while: `do { código } while (condición);`

Ejemplo:

```
<?php
  for ($i = 0; $i < 5; $i++) {
    echo $i . '<br>';
  }
?>
```

Errores y excepciones

PHP admite los siguientes mecanismos de errores y excepciones:

- `error_reporting`: establece el nivel de reporte de errores.
- `trigger_error`: activa un error personalizado.

- `try-catch`: maneja errores y excepciones.

Ejemplo:

```
<?php
  try {
    $a = 5 / 0;
  } catch (Exception $e) {
    echo 'Error: ' . $e->getMessage();
  }
?>
```

Comentarios

PHP admite los siguientes tipos de comentarios:

- `//`: comentarios de una línea.
- `/* */`: comentarios de múltiples líneas.

Ejemplo:

```
<?php
  // Este es un comentario de una línea.
  /* Este es un comentario de múltiples líneas.
     Puedes escribir aquí lo que desees.
  */
?>
```

Identificadores

PHP admite los siguientes identificadores:

- Variables: se utilizan para nombrar variables.
- Funciones: se utilizan para nombrar funciones.
- Etiquetas: se utilizan para nombrar etiquetas.

Ejemplo:

```
<?php
  $nombre = 'Juan';
  function saludar($nombre) {
    echo 'Hola, ' . $nombre . '!';
  }
?>
```

Funciones

En PHP, una función es un bloque de código que se puede llamar varias veces desde diferentes partes del programa. Las funciones pueden recibir parámetros y devolver valores.

Ejemplo de función simple

```
<?php
  function saludar($nombre) {
    echo 'Hola, ' . $nombre . '!';
  }
  saludar('Juan'); // Output: Hola, Juan!
?>
```

Ejemplo de función con parámetros y devolución de valor

```
<?php
  function suma($a, $b) {
    return $a + $b;
  }
  $resultado = suma(2, 3);
  echo $resultado; // Output: 5
?>
```

Clases

En PHP, una clase es un conjunto de atributos y métodos que se pueden utilizar para crear objetos. Las clases se definen utilizando la palabra clave `class` y se pueden heredar de otras clases.

Ejemplo de clase simple

```
<?php
  class Persona {
    public $nombre;
    public $edad;

    public function __construct($nombre, $edad) {
      $this->nombre = $nombre;
      $this->edad = $edad;
    }

    public function saludar() {
      echo 'Hola, soy ' . $this->nombre . ' y tengo ' .
$this->edad . ' años.';
    }
  }
  $persona = new Persona('Juan', 30);
  $persona->saludar(); // Output: Hola, soy Juan y
tengo 30 años.
?>
```

Objetos

En PHP, un objeto es una instancia de una clase. Los objetos tienen atributos y métodos que se pueden utilizar para interactuar con ellos.

Ejemplo de objeto

```
<?php
```

```
  $persona = new Persona('Juan', 30);
  echo $persona->nombre; // Output: Juan
  echo $persona->edad; // Output: 30
  $persona->saludar(); // Output: Hola, soy Juan y
tengo 30 años.
?>
```

Herencia

En PHP, la herencia es la capacidad de una clase de heredar atributos y métodos de otra clase. La herencia se define utilizando la palabra clave `extends`.

Ejemplo de herencia

```
<?php
  class Persona {
    public $nombre;
    public $edad;

    public function __construct($nombre, $edad) {
      $this->nombre = $nombre;
      $this->edad = $edad;
    }

    public function saludar() {
      echo 'Hola, soy ' . $this->nombre . ' y tengo ' .
$this->edad . ' años.';
```

```
  }
 }

 class Estudiante extends Persona {
  public $materia;

  public function __construct($nombre, $edad,
$materia) {
    parent::__construct($nombre, $edad);
    $this->materia = $materia;
  }

  public function mostrarMateria() {
    echo 'Estoy estudiando ' . $this->materia . '.';
  }
 }

 $estudiante = new Estudiante('Juan', 20,
'Matemáticas');
 $estudiante->saludar(); // Output: Hola, soy Juan y
tengo 20 años.
 $estudiante->mostrarMateria(); // Output: Estoy
estudiando Matemáticas.
?>
```