Introdução ao PHP

PHP continua a ser uma ferramenta vital no arsenal de um desenvolvedor web devido à sua flexibilidade, poder e a enorme comunidade de suporte. Quer você esteja criando um simples site pessoal ou uma complexa aplicação empresarial, o PHP oferece as ferramentas e recursos necessários para realizar seu projeto com eficiência e eficácia.

Primeiro código

```php
<?php
// Comentário de uma linha

/*
   Comentário de múltiplas linhas
*/

echo "Olá, Mundo!"; // Exibe "Olá, Mundo!"
```

Variáveis.

Uma variável na programação é um nome simbólico associado a um valor e cuja principal função é armazenar dados que podem ser utilizados e manipulados ao longo da execução de um programa. Em termos simples, uma variável atua como um recipiente para guardar informações que podem ser alteradas ou referenciadas posteriormente.

Conceitos Fundamentais de Variáveis

Nome da Variável: É a identificação que damos à variável para referenciá-la no código. Geralmente, os nomes de variáveis devem ser descritivos para facilitar a leitura e manutenção do código.
Tipo de Dado: Refere-se ao tipo de valor que a variável pode armazenar, como números inteiros, números de ponto flutuante, strings, booleanos, entre outros. Algumas linguagens de programação exigem que o tipo de dado seja explicitamente declarado, enquanto outras determinam o tipo automaticamente.
Valor: É a informação ou dado que a variável contém. Este valor pode ser modificado durante a execução do programa.

Declaração de Variáveis no PHP

No PHP, uma variável é representada por um nome que começa com o símbolo **$**, seguido por uma combinação de letras, números e. Elas são usadas para armazenar valores que podem ser alterados e reutilizados ao longo do script.

Declaração de Variáveis no PHP

```php
<?php
$nome = "João"; // String
$idade = 30; // Inteiro
$altura = 1.75; // Float
$isAdult = true; // Booleano
```

Tipos de Dados Comuns

Inteiros (Integers): Números sem ponto decimal.

```
$idade = 30;
```

Ponto Flutuante (Floats): Números com ponto decimal.

```
$altura = 1.75;
```

Strings: Sequências de caracteres.

```
$nome = "João";
```

Tipos de Dados Comuns

Booleanos (Booleans): Valores verdadeiros ou falsos.

```php
$isAdulto = true;
```

Arrays: Coleções ordenadas de valores.

```php
$cores = array("vermelho", "verde", "azul");
```

Escopo de Variáveis

Escopo Global: Variáveis que podem ser acessadas em qualquer parte do programa.
Escopo Local: Variáveis que só podem ser acessadas dentro do bloco de código ou função onde foram declaradas.
Variáveis Estáticas: Em algumas linguagens, variáveis dentro de uma função podem manter seu valor entre diferentes chamadas da função.

Conclusão

Variáveis são um dos conceitos mais fundamentais na programação, permitindo que dados sejam armazenados, manipulados e reutilizados ao longo da execução de um programa. Compreender como declarar, nomear e utilizar variáveis é essencial para qualquer programador, independentemente da linguagem de programação utilizada.

O que é uma Função no PHP

Uma função no PHP é um bloco de código que pode ser chamado repetidamente em diferentes partes de um programa, permitindo a reutilização do código e a modularização do programa. Funções podem aceitar parâmetros de entrada, executar operações com esses parâmetros e retornar um valor de saída.

Declaração de Funções

Para declarar uma função em PHP, utiliza-se a palavra-chave `function`, seguida pelo nome da função, um conjunto de parênteses (que pode conter parâmetros) e um bloco de código entre chaves `{}`.

```php
<?php
function nomeDaFuncao() {
    // código da função
}
```

Parâmetros e Argumentos

Funções podem receber parâmetros, que são variáveis passadas para a função no momento de sua chamada. Esses parâmetros são definidos dentro dos parênteses na declaração da função.

```php
<?php
function saudacao($nome) {
    echo "Olá, $nome!";
}
```

Valores de Retorno

Funções podem retornar valores usando a palavra-chave `return`. Isso permite que o resultado da função seja armazenado em uma variável ou usado diretamente.

```php
<?php
function soma($a, $b) {
    return $a + $b;
}

$resultado = soma(5, 3);
echo $resultado; // Saída: 8
```

E “SE” ou IF

comando `if` em PHP é usado para executar um bloco de código condicionalmente, com base em uma expressão avaliada como verdadeira ou falsa. Ele é um dos componentes fundamentais do controle de fluxo em PHP, permitindo que o código tome decisões dinâmicas durante a execução.

Estrutura Básica do `if`

A estrutura básica de um comando `if` é a seguinte:

```php
<?php
if (condição) {
    // bloco de código a ser executado se a condição for verdadeira
}
```

Exemplo Básico

```php
<?php
$idade = 20;

if ($idade >= 18) {
    echo "Você é maior de idade.";
}
```

Comando `if-else`

O comando `if` pode ser expandido com `else` para especificar um bloco de código alternativo que será executado se a condição for falsa.

```php
<?php
$idade = 16;

if ($idade >= 18) {
    echo "Você é maior de idade.";
} else {
    echo "Você é menor de idade.";
}
```

Comando `if-elseif-else`

Para verificar múltiplas condições, você pode usar `elseif` (ou `else if`).

```php
<?php
$nota = 75;

if ($nota >= 90) {
    echo "A nota é A.";
} elseif ($nota >= 80) {
    echo "A nota é B.";
} elseif ($nota >= 70) {
    echo "A nota é C.";
} else {
    echo "A nota é D ou inferior.";
}
```

Comando `if-elseif-else`

Para verificar múltiplas condições, você pode usar `elseif` (ou `else if`).

```php
<?php
$nota = 75;

if ($nota >= 90) {
    echo "A nota é A.";
} elseif ($nota >= 80) {
    echo "A nota é B.";
} elseif ($nota >= 70) {
    echo "A nota é C.";
} else {
    echo "A nota é D ou inferior.";
}
```

Uso de Operadores Lógicos

Você pode usar operadores lógicos para combinar múltiplas condições em uma única expressão `if`.

Operador AND

```php
<?php
$idade = 25;
$renda = 3000;

if ($idade >= 18 && $renda >= 2000) {
    echo "Você atende aos critérios.";
}
```

Operador OR

```php
<?php
$idade = 16;
$renda = 3000;

if ($idade >= 18 || $renda >= 2000) {
    echo "Você atende a pelo menos um dos critérios.";
}
```

Operador NOT

```php
<?php
$idade = 16;

if (!($idade >= 18)) {
    echo "Você é menor de idade.";
}
```

Operador Ternário

O operador ternário é uma forma compacta de escrever uma declaração `if-else`.

```php
<?php
$idade = 20;
$mensagem = ($idade >= 18) ? "Você é maior de idade." : "Você é menor de idade.";
echo $mensagem;
```

XOR (`xor`)

Descrição: Retorna verdadeiro se uma, e apenas uma, das expressões for verdadeira.

```php
<?php
$a = true;
$b = false;

if ($a xor $b) {
    echo "Uma, e apenas uma, das expressões é verdadeira.";
}
```