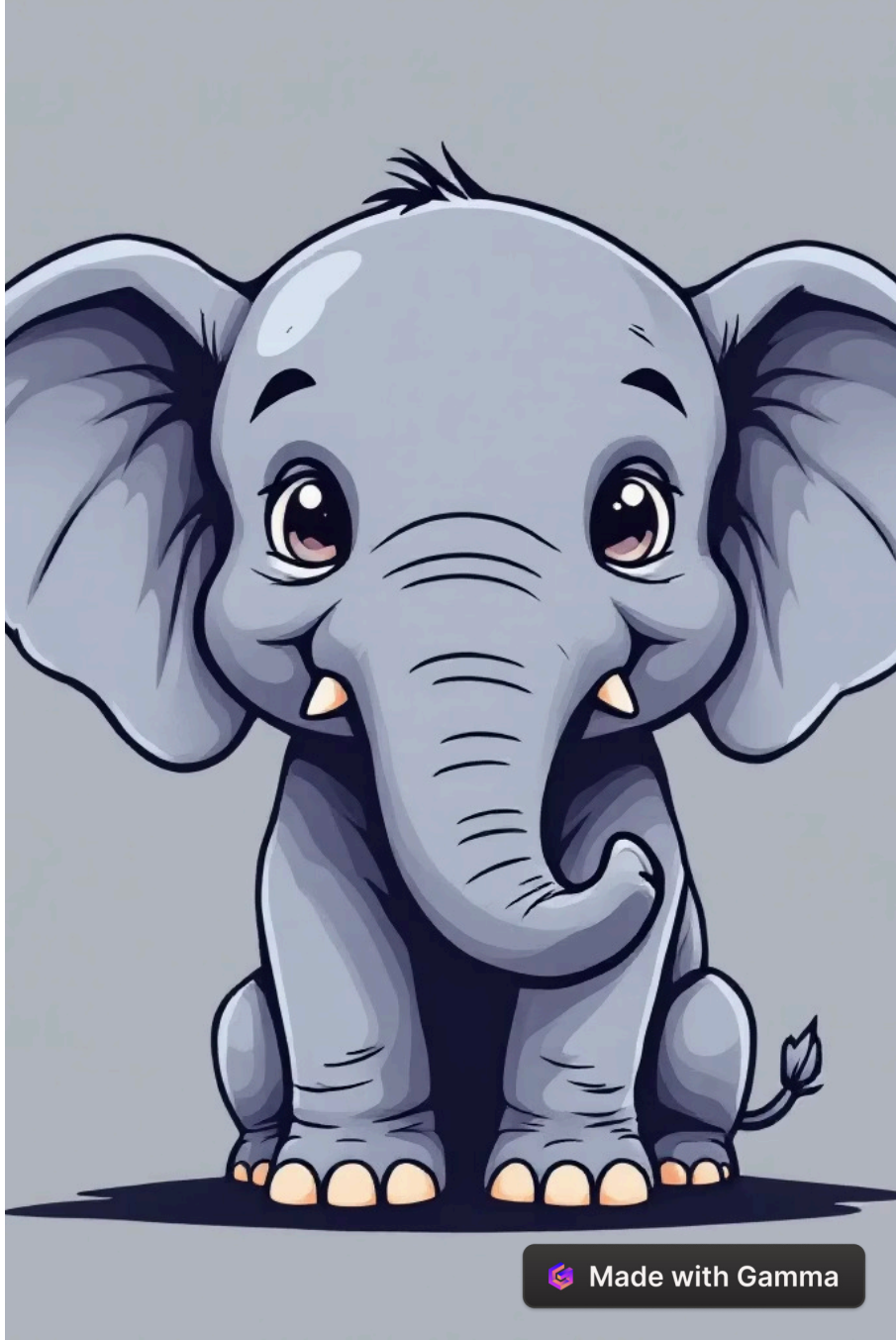
Funções PHP

por Rafael SP



Aula: Funções em PHP

1. O que são Funções em PHP?

Em PHP, uma **função** é um bloco de código que você pode definir e reutilizar sempre que necessário. Ao definir funções, você pode evitar a repetição de código, tornando o código mais organizado, modular e fácil de manter.

2. Funções Básicas em PHP

A sintaxe básica de uma função em PHP é a seguinte:

```
function nome_da_funcao() {
    // Corpo da função
    // Código que será executado quando a função for chamada
}
```

Exemplo de uma função simples:

```
<?php
function saudacao() {
    echo "Olá, seja bem-vindo ao PHP!";
}

// Chamando a função
saudacao(); // Saída: Olá, seja bem-vindo ao PHP!
?>
```

Neste exemplo, a função `saudacao()` é definida para imprimir uma mensagem. Quando chamamos a função `saudacao()`, o código dentro dela é executado.

3. Funções com Parâmetros

Você pode passar **parâmetros** para uma função para tornar seu comportamento dinâmico. Os parâmetros são valores que você passa para a função ao chamá-la.

Exemplo de função com parâmetros:

```
<?php
function saudacao_personalizada($nome) {
    echo "Olá, $nome! Seja bem-vindo ao PHP!";
}

// Chamando a função com o parâmetro
saudacao_personalizada("Carlos"); // Saída: Olá, Carlos! Seja bem-vindo ao PHP!
?>
```

Aqui, a função `saudacao_personalizada()` recebe um parâmetro chamado `$nome` e imprime uma saudação personalizada.

4. Funções com Valor de Retorno

Funções também podem **retornar valores** usando a palavra-chave `return`. O valor retornado pode ser utilizado em outra parte do código.

Exemplo de função com retorno:

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

// Chamando a função e armazenando o resultado
$resultado = soma(5, 3);
echo "O resultado da soma é: $resultado"; // Saída: O resultado da soma é: 8
?>
```

Neste exemplo, a função `soma()` retorna a soma de dois números. O valor retornado é armazenado na variável `$resultado` e exibido.

5. Funções com Parâmetros Opcionais

Você pode definir parâmetros **opcionais** em uma função, atribuindo um valor padrão a eles. Se o parâmetro não for passado ao chamar a função, o valor padrão será usado.

Exemplo de função com parâmetro opcional:

```
<?php
function saudacao($nome = "Visitante") {
    echo "Olá, $nome! Seja bem-vindo ao PHP!";
}

// Chamando a função sem passar o parâmetro
saudacao(); // Saída: Olá, Visitante! Seja bem-vindo ao PHP!

// Chamando a função passando o parâmetro
saudacao("Maria"); // Saída: Olá, Maria! Seja bem-vindo ao PHP!
?>
```

Neste exemplo, o parâmetro `$nome` tem um valor padrão de `"Visitante"`. Se o nome não for fornecido, o valor padrão será usado.

6. Funções Anônimas (Funções Lambda)

Em PHP, você pode criar **funções anônimas**, ou seja, funções sem nome. Elas podem ser atribuídas a variáveis e usadas como callbacks ou passadas para outras funções.

Exemplo de função anônima:

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

echo $soma(5, 10); // Saída: 15
?>
```

Aqui, a função anônima é atribuída à variável `$soma` e chamada diretamente.

7. Funções Recursivas

Uma função é chamada **recursiva** quando ela se chama dentro de seu próprio corpo. Isso é útil para problemas que podem ser quebrados em subproblemas menores.

Exemplo de função recursiva (Fatorial):

```
<?php
function fatorial($n) {
    if ($n == 0) {
        return 1;
    } else {
        return $n * fatorial($n - 1);
    }
}

echo fatorial(5); // Saída: 120
?>
```

Neste exemplo, a função `fatorial()` chama a si mesma até atingir o caso base, que é quando `$n == 0`.

8. Funções com `static` e `public function`

Agora, vamos entender o uso de `static` e `public function`.

Funções com `static`

A palavra-chave `static` em PHP é usada para declarar uma função (ou variável) que pertence à **classe**, em vez de pertencer a uma instância de objeto dessa classe. Ou seja, você pode chamar a função `static` sem criar um objeto da classe.

Exemplo de função estática:

```
<?php
class Calculadora {
    public static function somar($a, $b) {
        return $a + $b;
    }
}

// Chamando a função estática diretamente pela classe, sem precisar de uma instância
echo Calculadora::somar(5, 3); // Saída: 8
?>
```

A função `somar()` é declarada como `static` e pode ser chamada diretamente pela classe, sem a necessidade de instanciar um objeto.

Funções `public` e `public function`

Em PHP, o `public` é um modificador de acesso que determina a visibilidade de métodos e propriedades de uma classe. Um método `public` pode ser acessado de qualquer lugar, tanto dentro como fora da classe.

Exemplo de função `public`:

```
<?php
class Saudacao {
    public function exibirMensagem($nome) {
        echo "Olá, $nome! Seja bem-vindo ao PHP!";
    }
}

// Criando uma instância da classe
$saudacao = new Saudacao();

// Chamando o método público
$saudacao->exibirMensagem("Carlos"); // Saída: Olá, Carlos! Seja bem-vindo ao PHP!
?>
```

Neste exemplo, a função `exibirMensagem()` é `public`, o que significa que pode ser chamada diretamente fora da classe após criar uma instância dela.

9. Exemplo Completo com `static` e `public function`

```
<?php
class Produto {
    public $nome;
    public $preco;

    // Método público
    public function exibirDetalhes() {
        echo "Produto: $this->nome, Preço: $this->preco";
    }

    // Método estático
    public static function calcularDesconto($preco, $percentual) {
        return $preco - ($preco * $percentual / 100);
    }
}

// Criando uma instância e chamando o método público
$produto = new Produto();
$produto->nome = "Camiseta";
$produto->preco = 50;
$produto->exibirDetalhes();  // Saída: Produto: Camiseta, Preço: 50

// Chamando o método estático
$precoComDesconto = Produto::calcularDesconto(50, 10);
echo "Preço com desconto: $precoComDesconto"; // Saída: Preço com desconto: 45
?>
```

10. Exercícios para Treinamento

Agora que você aprendeu sobre funções em PHP, incluindo `static` e `public function`, aqui estão 5 exercícios para praticar!

Exercício 1: Função Estática

Crie uma classe chamada `OperacoesMatematicas` com uma função estática chamada `multiplicar()`. A função deve receber dois números e retornar o produto deles. Chame a função estática sem criar uma instância da classe.

Exercício 2: Função Recursiva

Crie uma função recursiva chamada `fibonacci()` que calcule o n-ésimo número da sequência de Fibonacci. Por exemplo, `fibonacci(5)` deve retornar `5`.

Exercício 3: Classe com Método Público

Crie uma classe chamada `Livro` com propriedades públicas para `titulo` e `autor`. Crie um método público chamado `exibirInformacoes()` que exibe o título e o autor do livro. Crie um objeto e exiba as informações do livro.

Exercício 4: Função com Parâmetro Opcional

Crie uma função chamada `desconto()` que recebe dois parâmetros: `preco` e `percentual`. O percentual deve ter um valor padrão de `10%`. A função deve retornar o preço com o desconto aplicado.

Exercício 5: Função Estática com Parâmetro

Crie uma classe chamada `Imposto` com uma com uma função estática chamada `calcularImposto()` que recebe o preço de um produto e retorna o valor do imposto (5% do preço). Chame a função estática e passe um valor para calcular o imposto.