Funções e Escopos

Academia Indiana

Funções em PHP

Funções podem ser definidas blocos de código com um objetivo específico, identificados por um nome através do qual pode ser referenciado á partir de várias partes do código. Essa é uma das principais técnicas utilizadas para garantir a reutilização de código, tornando a programação mais prática e o código mais “limpo” e organizado.

A declaração de funções no PHP é feita a partir da palavra reservada function seguida do nome da função e de sua lista de argumentos, enquanto o corpo da função é delimitado por chaves ({ e }), entre as quais deve ficar todo o conjunto de instruções a ser executado quando a função for invocada.

```
function nome_funcao($par1, $par2, $par3...$parN)
{
    //instruções
}
```

Argumentos de funções

Argumentos ou parâmetros são informações que podem ser passadas para funções. Podemos definir valores padrão para argumentos, assim, caso não sejam informados, o php irá assumir o valor padrão para o argumento. Alguns exemplos:

```
function exibir_mensagem()
{
    echo "Olá, Seja Bem Vindo(a)!";
}

function exibir_mensagem($nome)
{
    echo "Olá, $nome";
}
```

Argumentos com valor padrão

Abaixo um exemplo de função onde o argumento $nome possui um valor padrão. Caso seja chamada sem o argumento nome, irá exibir na tela Olá, Plinio

```
function exibir_mensagem($nome = “Plinio”)
{
    echo "Olá, $nome";
}
```

Funções retornando valores

As funções que mostramos até agora não retornam nenhum valor. Vamos ver agora como fazer uma função retornar um valor ao ser chamada.

A palavra reservada **return** é utilizada para definir o resultado da função e sinaliza também o fim da execução. Qualquer código que venha a ser colocado após o **return** será desconsiderado.

```
function somar($numA, $numB)
{
    return $numA + $numB;
}
```

Escopo de variáveis

O escopo de variável é o contexto no qual a variável foi criada, e no qual ela pode ser acessada.

Essencialmente, o PHP tem dois escopos:

- **Global:** A variável fica acessível em qualquer parte do script
- **Local:** A variável é acessível só dentro da função (ou método) que ele foi criado.

O escopo de variáveis – especialmente o escopo local – ajuda a tornar o código mais fácil de gerenciar. Se todas as suas variáveis são globais, elas podem ser lidas e alteradas em qualquer lugar do seu código. Isso pode causar um grande caos no seu script, com muitas partes tentando acessar e trabalhar com a mesma variável. Ao restringir uma variável ao escopo local, você limita a quantidade de código que pode acessar essa variável, tornando o código mais robusto, mais modular e mais fácil de depurar.

Escopo de variáveis (exemplo)

Aqui temos um exemplo de uma variável local e uma global:

```
<?php
     $globalName = "Plinio";
     function Hello() {
          $localName = "Cesar";
          echo "Olá, $localName!<br>";
     }
     Hello();
     echo "globalName: $globalname <br>";
     echo "localName: $localName <br>";
?>
```

Variáveis Globais

Para acessar uma variável global fora de uma função, basta escrever o nome da variável. Para acessar uma variável global dentro de uma função, no entanto, primeiro você precisa declarar a variável como global dentro da função usando a palavra-chave **global**:

```
<?php
	$globalName = "Plinio";

	function Hello() {
		$localName = "Cesar";
		echo "Olá, $localName!<br>";
		global $globalName;
		echo "Olá, $globalName!<br>";
	}

	Hello();
?>
```

Superglobals

O PHP fornece um conjunto especial de arrays globais contendo várias informações úteis. Esses arrays são conhecidos como superglobais, porque eles são acessíveis em qualquer lugar no seu código – inclusive dentro de funções – e você nem precisa declará-los como global usando a palavra-chave global. Estas variáveis superglobais são:

- *$GLOBALS*
- *$_SERVER*
- *$_GET*
- *$_POST*
- *$_FILES*
- *$_COOKIE*
- *$_SESSION*
- *$_REQUEST*
- *$_ENV*

Variáveis Static

Quando você cria uma variável local dentro de uma função, essa variável só existe enquanto a função está sendo executada. Quando a função acaba, a variável local desaparece. Quando a função é chamada novamente, uma nova variável local é criada.

No entanto, existem situações onde é útil criar uma variável local que mantém o mesmo valor entre cada chamada da função. É exatamente assim que funcionam as variáveis estáticas.

Para criar uma variável estática, você deve escrever a palavra-chave **static** antes do nome da variável e atribuir um valor inicial para a variável.

Variáveis Static (exemplo)

```php
function Teste()
{
    static $a = 0;
    echo $a;
    $a++;
}

teste();
teste();
teste();
```

Exercícios

1- Crie uma função que receba 2 valores inteiros como argumentos e retorne a sua soma. Se o valor da soma for negativo a função deverá o valor 0.

Exercícios

2- Crie uma função que receba 2 notas ($n1 e $n2) de um aluno. Essa função deve retornar um booleano indicando se o aluno foi aprovado. Para ser aprovado, a soma das notas deve ser igual ou superior a 19 e ambas devem ser superiores a 7.

...

Exercícios

3- Crie uma função em PHP que calcule e imprima o salário reajustado de um funcionário de acordo com a seguinte regra:

- salários até R$ 3.000,00 , reajuste de 50%
- salários de R$ 3.000,00 ou mais, reajuste de 30%

Exercícios

4- **Crie uma função em PHP que receba a altura e o peso de uma pessoa, calcule seu IMC de acordo com a fórmula abaixo:**

IMC = Peso dividido pela altura ao quadrado **(peso ÷ (altura²))**.

A função deve retornar uma string (magreza, normal. sobrepeso ou obesidade) de acordo com a tabela abaixo:

	IMC
Magreza	< 18.5
Normal	18.5 a 24.9
Sobrepeso	24.9 a 30
Obesidade	> 30