Linguagem de Programação II

Prof.ª Ma. Jessica Oliveira

FBr Faculdade Brasília

Revisão

Resolvendo os desafios da última aula.

Clique **AQUI** para visualizar o gabarito.

Aula 02 – 19/08/2024

Fundamentos de PHP e Tipos de dados, variáveis e constantes.

Introdução ao PHP

Conceitos iniciais.

Desafio do dia

Valendo até 0,50 do ponto EXTRA.

Resenha de, no mínimo, 300 palavras, abordando os seguintes assuntos:

- História e evolução do PHP;
- Principais usos dessa linguagem;
- Uso do PHP nos dias de hoje.

Enviar em **.docx** para o e-mail: **jessica.oliveira@fbr.edu.br** em até 20 minutos.

Assunto do e-mail: Ponto extra – LPII – 19/08/2024.

Colocar **nome completo e matrícula** no arquivo anexado.

ATENÇÃO:
RESENHA NÃO É RESUMO!

Resenha

- **Objetivo:** Analisar, criticar e avaliar um texto, livro, etc.
- **Características:**
  - É mais longa que um resumo.
  - Permite que o autor apresente sua opinião sobre a obra.
  - Pode fazer comparações com outras obras.
  - Pode analisar a relevância da obra para determinada área do conhecimento.

Resumo

- **Objetivo:** Apresentar de forma concisa e objetiva as ideias principais de um texto.
- **Características:**
  - É mais curto que o texto original.
  - Mantém a ordem das ideias do texto original.
  - Utiliza as mesmas palavras do autor, evitando paráfrases.
  - Não inclui informações que não estejam presentes no texto original.

O que é PHP?

- *Hypertext Preprocessor* (pré-processador de hipertexto);
- Linguagem de programação de *script*, ou seja, é interpretada em vez de compilada;
- É usada principalmente para desenvolvimento web;
- Executada no lado do servidor, o que significa que o código PHP é processado antes de ser enviado ao navegador do cliente;

Vantagens

- **Open-source e gratuito:** é totalmente gratuito, o que o torna uma escolha econômica para desenvolvedores e empresas. Além disso, sendo *open-source*, ele possui uma grande comunidade que contribui com ferramentas, *frameworks*, e bibliotecas.
- **Grande suporte de hospedagem:** a maioria dos servidores de hospedagem suporta PHP, tornando-o amplamente disponível e fácil de implementar. Ele é compatível com quase todos os tipos de servidores (Apache, Nginx, etc.).

Vantagens

- **Fácil de aprender e usar:** é relativamente fácil de aprender para iniciantes, especialmente para aqueles que já têm algum conhecimento de programação. Sua sintaxe é intuitiva, e há vasta documentação disponível.
- **Suporte extensivo a bancos de dados:** pode se conectar a uma ampla gama de bancos de dados, tanto relacionais (como MySQL, PostgreSQL) quanto NoSQL (como MongoDB).

Vantagens

- **Comunidade ampla e ativa:** a comunidade PHP é uma das maiores do mundo, o que significa que há muitos recursos disponíveis, incluindo tutoriais, fóruns, e bibliotecas de código.
- **Rápido desenvolvimento e prototipagem:** com PHP, é possível desenvolver e testar aplicações rapidamente, o que é ideal para *startups* e projetos que exigem iterações rápidas.

Desvantagens

- **Segurança:** PHP é conhecido por ter algumas vulnerabilidades de segurança, especialmente quando mal utilizado. Embora as versões mais recentes tenham abordado muitas dessas questões, desenvolvedores inexperientes podem ainda introduzir brechas de segurança em suas aplicações.
- **Desempenho em grandes aplicações:** embora seja eficiente para pequenas e médias aplicações, ele pode enfrentar desafios de desempenho em grandes aplicações de alta escala. Para projetos de alta carga, linguagens como Python podem oferecer melhor desempenho.

Sintaxe básica

- **Tags PHP:** o código PHP é embutido dentro de um arquivo HTML e é delimitado por tags específicas. As tags padrão são:

```
<?php
echo “Turma ADS, 2º Período";
?>
```

Sintaxe básica

- **Comentários de uma linha:** utiliza-se “//” ou “#”.

```
// Este é um comentário de uma linha
# Este também é um comentário de uma linha
```

Sintaxe básica

- **Comentários de múltiplas linhas:** utiliza-se “/* ... */”.

```
/*
Este é um comentário
de múltiplas linhas
*/
```

Sintaxe básica

- **Instruções:** são as menores unidades de execução do código. Elas representam uma ação completa, como atribuir um valor a uma variável, exibir algo na tela, ou realizar um cálculo.
- **Declarações:** são tipos específicos de instruções que definem como variáveis e funções são tratadas.
- **Ponto e Vírgula:** cada instrução no PHP **DEVE** ser finalizada com um ponto e vírgula (;), pois ele indica ao interpretador que aquela linha de código chegou ao fim e que ele pode passar para a próxima instrução.

Tipos de dados em PHP

Inteiros, flutuantes, *strings*, *arrays*, booleano e *null*.

Inteiros (“int”)

- Números sem casas decimais, que podem ser positivos ou negativos.
- **Exemplo:**

```
$numero_pos = 32;
$numero_neg = -32;
```

Flutuantes (“float” ou “double”)

- Números com casas decimais, também conhecidos como números de ponto flutuante. Usados para representar valores como preços ou medidas.
- **Exemplo:**

```
$preco = 29.99;
$altura = 1.65;
```

Strings (“string”)

- Sequências de caracteres, usadas para representar texto. *Strings* em PHP podem ser delimitadas por aspas simples (') ou duplas (").
- **Exemplo:**

```
$nome = “Pedro";
$saudacao = 'Olá, humano!';
```

Importante!

- **Aspas simples:** o conteúdo é tratado literalmente, sem interpretação de variáveis.
- **Exemplo:**

```
$nome = ‘Pedro’;
echo 'Olá, $nome!'; // Saída: Olá, $nome!
```

Importante!

- **Aspas duplas:** variáveis dentro da string são interpretadas.
- **Exemplo:**

```
$nome = “Pedro";
echo “Olá, $nome!”; // Saída: Olá, Pedro!
```

Arrays (“array”)

- Estruturas que permitem armazenar múltiplos valores em uma única variável. Podem ser indexados por números (*arrays* numéricos) ou por *strings* (*arrays* associativos).
- **Exemplo:**

```
$pessoa = array("nome" => “Pedro", "idade" => 22, "cidade" => “Santa Maria");
```

Booleanos (“bool”)

- Representam dois estados: verdadeiro (true) ou falso (false). Muito usados em testes condicionais.
- **Exemplo:**

```
$ligado = true;
$desligado = false;
```

Null

- Representa uma variável sem valor. “null” é usado para indicar que uma variável não foi inicializada ou que foi explicitamente esvaziada.
- **Exemplo:**

```
$variavel = null;
```

Variáveis e Constantes

Declaração, escopo e tipos.

Declaração de Variáveis

- Em PHP, variáveis são precedidas pelo símbolo “$”, seguido do nome da variável, e são inicializadas atribuindo-lhes um valor usando o operador “=”.
- Não é necessário declarar o tipo de dado da variável, pois ele será inferido automaticamente com base no valor atribuído.

Declaração de Variáveis

- **Exemplo de declaração e atribuição:**

```
$nome = “Michele”;
$idade = 25;
$altura = 1.65;
$matricula = true;
```

Escopo de Variáveis

- Refere-se à área do programa onde a variável é acessível;
- É importante, pois ajuda a manter o código mais organizado e fácil de entender, evitando conflitos de nomes de variáveis;
- Evita que o valor de uma variável seja modificado acidentalmente em um lugar onde não deveria;
- Permite criar funções que podem ser reutilizadas em diferentes partes do código sem interferir umas nas outras;
- Existem três tipos de escopo em PHP: **local**, **global** e **estático**.

Escopo Local

- **Onde:** dentro de uma função.
- **Por que:** variáveis declaradas aqui só "vivem" dentro dessa função. Quando a função termina, essas variáveis são "esquecidas".
- **Quando usar:** para variáveis que só são necessárias dentro de uma função específica.
- **Exemplo:**

```php
function minhaFuncao() {
    $variavelLocal = "Oi, Ana!";
    echo $variavelLocal; // Funciona
}
minhaFuncao();
echo $variavelLocal;
/* Não funciona, pois $variavelLocal só existe dentro da função */
```

Escopo Global

- **Onde:** fora de qualquer função.
- **Por que:** variáveis declaradas aqui podem ser acessadas de qualquer lugar do seu script.
- **Quando usar:** usar com cautela, pois variáveis globais podem tornar o código mais difícil de entender e manter. Usar apenas quando for estritamente necessário.
- **Exemplo:**

```php
$variavelGlobal = "Eu sou global!";

function outraFuncao() {
    global $variavelGlobal;
    /* Precisa usar 'global' para acessar a variável global dentro da função */
    echo $variavelGlobal;
}
outraFuncao();
```

Escopo Estático

- **Onde:** dentro de uma função, mas com a palavra-chave *static*.
- **Por que:** variáveis declaradas como *static* mantém seu valor entre as chamadas da função. É como se a variável "lembrasse" seu valor da última vez que a função foi chamada.
- **Quando usar:** usar quando precisar de uma variável que mantenha seu valor entre as chamadas de uma função, como um contador ou um cache.
- **Exemplo:**

```php
function contarVisitas() {
    static $visitas = 0;
    $visitas++;
    echo "Esta página foi visitada " . $visitas . " vezes.<br>";
}
contarVisitas(); // Imprime: Esta página foi visitada 1 vez.
contarVisitas(); // Imprime: Esta página foi visitada 2 vezes.
contarVisitas(); // Imprime: Esta página foi visitada 3 vezes.
```

Constantes

- É um identificador que se refere a um valor que não pode ser alterado durante a execução do *script*.
- Uma vez definida, sua atribuição permanece a mesma e isso as torna ideais para armazenar valores que nunca mudarão;
- Não utilizam o símbolo “$” e são de escopo global, ou seja, uma constante definida em qualquer lugar do *script* pode ser acessada em qualquer outro lugar.
- Podem conter valores de tipos simples como *int*, *float*, *string*, ou *booleano*.

Declaração de constantes

- Exemplo com “**define**”:

```
define("NOME_DO_SITE", "Meu Site");
echo NOME_DO_SITE; // Saída: Meu Site
```

- Exemplo com “**const**” (geralmente usado dentro de classes):

```
const PI = 3.14;
echo PI; // Saída: 3.14
```

Na próxima aula...

Aula 04 (26/08/2024) - Estruturas de controle, funções e início das práticas e dos relatórios.

Dúvidas?

jessica.oliveira@fbr.edu.br