Programação Web

Professor Dr. Leonardo Souto.

PHP
Personal Home Page

Apresentação da Linguagem

By: Rasmus Lerdof

Principais Características do PHP

- PHP é um acrônimo para “**PHP: Hypertext Preprocessor**”;
- Trata-se de uma linguagem **script** (software livre);
- Muito utilizada para gerar conteúdo dinâmico na Web;
- Scripts PHP são executados em servidores;
- PHP dá suporte a vários servidores de bancos de dados (MySQL, Oracle, PostgreSQL, etc.).

Principais Características do PHP

- É livre para ser baixado e usado;
- Linguagem portável: Versões para vários sistemas operacionais;
- Pode ser mesclada com HTML;
- Pode gerar páginas HTML;
- Sintaxe similar a linguagens como C e Java;
- Permite o desenvolvimento de páginas que serão geradas dinamicamente.

Softwares

Computador que irá conter as páginas PHP

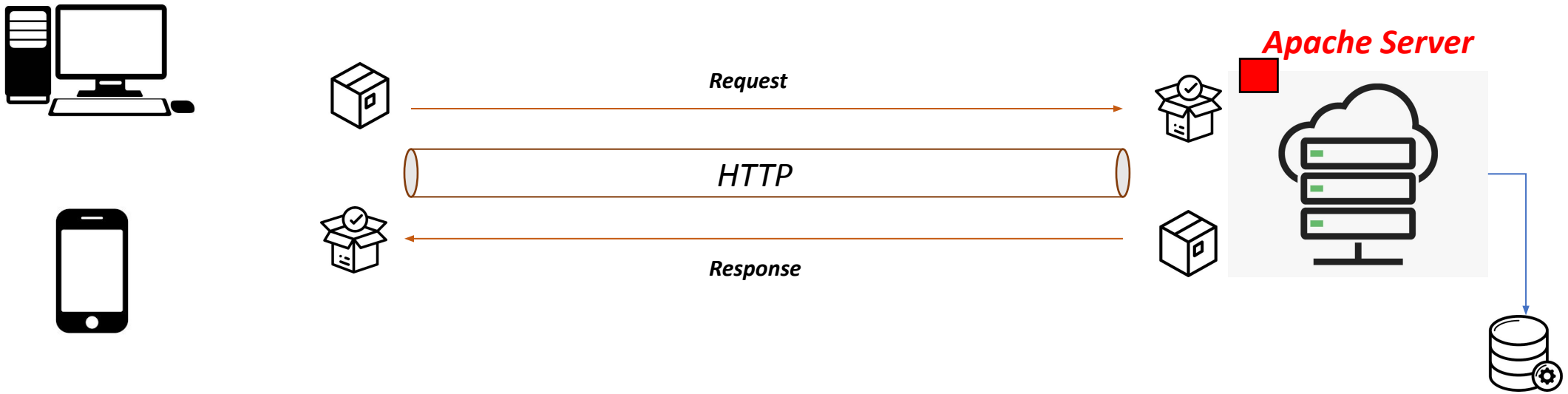
- O **servidor** que dará suporte a PHP (local ou remoto) deverá possuir os seguintes softwares:

1. Um **servidor** HTTP;
2. O PHP;
3. Um SGBD (Opcional);

Software que irá reconhecer o protocolo HTTP e as solicitações do cliente

WWW

http://localhost/Codigos/pagina.php



Cliente

Servidor

Servidor HTTP (ou servidor Web)

- Programa responsável por aceitar **requisições HTTP** de clientes, geralmente navegadores, e servi-los com respostas HTTP;
- Servidor HTTP padrão para scripts PHP: **Apache Server**
  - Software colaborativo (livre), portável e pouco exigente com recursos de Hardware;
  - Gerenciado por um grupo mundial de voluntários;
  - A fundação *Apache Software Foundation* provê o suporte financeiro, legal e organizacional para o Apache.

O PHP

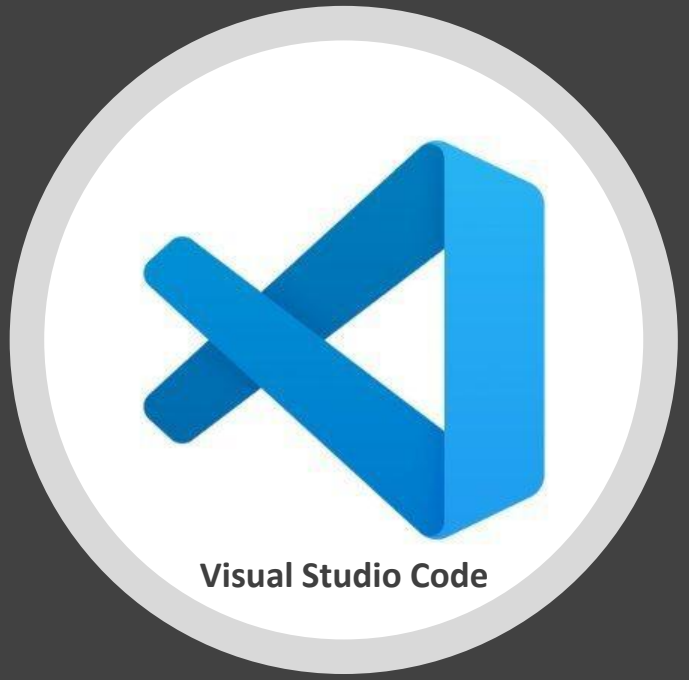
- O PHP precisa estar instalado em um servidor para que a linguagem seja reconhecida.

SGBD - Sistema de Gerenciamento de Banco de Dados

- Necessário para sites e aplicações Web que realizam as operações de manipulação dos dados (CRUD) em banco de dados;
- SGBD difundido para uso agregado com PHP: MySQL Server
  - Utiliza a linguagem SQL como interface;
  - Software Livre;
  - Portável e pouco exigente com recursos de Hardware.

Vou precisar instalar tudo isso pra começar a programar com PHP?

Sim... Mas tem programas que instalam tudo pra você!



Editores e IDE’s

Pasta htdocs

- Dentro da pasta onde foi instalado o **XAMPP**, adicione o script PHP dentro da pasta **htdocs**.
- Ou “apague o conteúdo de htdocs” ou “crie uma pasta para adicionar os exercícios". Lembre-se do nome da pasta;
- Em seguida é só acessar o endereço localhost no navegador;
- http://localhost
- Caso tenha criado uma pasta dentro do localhost, será preciso detalhar o caminho da pasta na url: http://localhost/suapasta

Sintaxe do PHP

Sintaxe do PHP

- Um script PHP sempre começa com **<?php** e termina com **?>**;
- Em alguns servidores, também é possível iniciar um script PHP com **<?** e terminar com **?>**;
- Um script PHP pode ser inserido em qualquer região de um documento HTML, ou ser totalmente codificado em PHP.

```
<?php

   // corpo do script

?>
```

Hello World!

Exemplo de script PHP que exibe o texto “Olá mundo!”:

```
<?php
   echo “Hello World”;
?>
```

Comandos de Saída

- **echo –** Imprime uma ou mais variáveis no console ou página
- **print –** Imprime uma string no console ou página
- **var_dump –** Imprime o conteúdo de uma variável de forma detalhada (debug)
- **print_r –** Imprime o conteúdo de uma variável de forma detalhada de forma mais legível

Sintaxe do PHP

- Cada linha de código de um script PHP **deve ser finalizado com ;** (ponto e vírgula). O ponto e vírgula é um separador e é utilizado para distinguir uma instrução da outra;
- No exemplo, utilizamos a declaração **echo**, usada para escrever texto em um documento HTML;
- O PHP fornece várias declarações para outros tipos de ações.

Sintaxe do PHP

- Toda página que possui um script PHP deve ser completamente codificada em PHP. Assim, a extensão do arquivo deve ser **.php**;
- Logo, salve a página como index.php

- Lembre-se: **PHP é case sensitive**!

Hello World

Exemplo de Página PHP que exibe o texto “Olá mundo!” em negrito:

```
<?php
  echo “<b>Hello World</b>”;
?>
```

Comentários

Comentários

- Em PHP, usamos **//** para digitar um comentário de linha;
- Utilizamos **/*** e ***/** para digitar um bloco de comentários;

Ex.:

```
<?php
    // Escreve algo na página
    echo “Texto”;
?>
```

Variáveis

Variáveis

- Variáveis são usadas para armazenar valores, como strings, números ou vetores;
- Uma variável declarada pode ser usada repetidamente em scripts;
- Uma variável PHP começa com o caractere **$**, seguido por seu nome;
- Nomes de variável não podem ter espaços;
- PHP não exige a definição de tipos de variáveis, logo, é uma linguagem **fracamente tipada**!

Tipos de Variáveis

Tipo	Exemplo
Booleano	True or False
Numérico	Inteiro ou Real
String	‘’ ou “”
Array	array()
Objeto	new Class
Recurso	resource
Misto	mixed
Callback	Função passada por parâmetro
Null	Variável não tem valor

Variáveis

Exemplos de variáveis e valores

```
<?php
    $texto = “Hello World”;
    $numero = 16;
    $salario = 20.50;
    $ligou = true;
?>
```

Variáveis

Exemplo 2

Escrevendo em uma página HTML o conteúdo de uma variável:

```
<?php
   $texto = “Tem alguém com sono?”;
   echo $texto;
?>
```

Concatenando Strings

- Usando o operador “.”

```
$numero = 100;
echo “O numero é: ” . $numero;
```

- Usando Template Strings

```
$numero = 100;
echo “O numero é: $numero”;
```

Constantes

- Para declarar uma constante em PHP é utilizado a função define.

```
<?php
   define(“texto”, ”Hello World!”);

   echo texto;
?>
```

Operadores

Operadores Aritméticos

Operador	Símbolo
Soma	+
Subtração	-
Multiplicação	*
Divisão	/
Resto	%
Incremento	++
Decremento	--
Concatenar	.

Operadores de Atribuição

Nome	Operador
Atribuição	=
Soma	+=
Subtração	-=
Multiplicação	*=
Divisão	/=
Concatenação	.=
Resto	%=

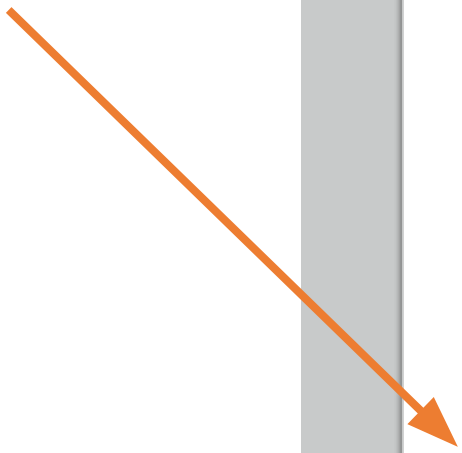
Operadores de Atribuição
Exemplo de Uso

```php
<?php
$a = 3;
$a += 5;
// define $a para 8, é interpretado como: $a = $a + 5;

$b = "Bom ";
$b .= "Dia!";
// define $b para "Bom Dia!", é o mesmo que $b = $b . "Dia!";
?>
```

Operadores de Comparação

- ***Idêntico: Igual com o mesmo tipo de dado***



Nome	Operador
Igualdade	==
Diferente	!= / <>
Maior que	>
Menor que	<
Maior ou Igual	>=
Menor ou Igual	<=
Idêntico	===
Não Idêntico	!===

Operadores Lógicos

Nome	Operador
E	&& / and
Ou	|| / or
Não	! / not

Capturando dados de formulários HTML

Tratando dados de formulários HTML

- A partir de formulários HTML, podemos fazer com que usuários submetam dados para scripts PHP;
- Os valores digitados no inputs HTML (caixa de texto, caixa de seleção, botões de seleção, etc.) poderão ser disponibilizados automaticamente para scripts PHP.

Exemplo – Testem!

Formulário HTML que contem dois campos de texto e um botão de submissão. Salve como **index.php**

```
<html>
<body>
    <form action="teste.php"
    method="POST">
        Nome: <input
        type="text"
        name="nome" /><br/>
        Idade: <input
        type="text"
        name="idade" /><br/>
        <input type="submit"
        value="Enviar" />
    </form>
</body>
</html>
```

Tratando dados de um formulário HTML

- Quando clicamos no botão de submissão em um formulário HTML, os dados são **submetidos** para o arquivo indicado no atributo **action** da tag **form**;
- No exemplo anterior, os dados contidos no formulário (nome e idade) serão enviados para o script **teste.php**;
- Portanto, precisamos criar o script que irá tratar os dados enviados pelo formulário HTML.

Script teste.php

Crie o arquivo teste.php e edite-o da seguinte forma:

```
<?php
  $nome = $_POST['nome'];
  $idade = $_POST['idade'];

  echo "Nome do usuário: " . $nome;
  echo "<br>Sua idade: " . $idade;
?>
```

index.php

```
<html><body>
       <form action="teste.php" method="POST">
          Nome: <input type="text" name="nome" /><br/>
          Idade: <input type="text" name="idade" /><br/>
          <input type="submit" value="Enviar" />
       </form>
</body></html>
```

teste.php

```
<?php
   $nome = $_POST['nome'];
   $idade = $_POST['idade'];
   echo "Nome do usuário: " . $nome;
   echo "<br>Sua idade: " . $idade;
 ?>
```

A variável $_POST

A variável **$_POST**

- É um array associativo de variáveis passados para o script atual via método HTTP POST;
- É usada para receber valores de formulário que foram submetidas através do método POST;
- O valor POST é indicado no atributo method da tag HTML form;
- Informações enviadas via formulários através do método POST são invisíveis para os usuários;
- Os limite em relação a quantidade de dados a serem enviados são definidos pelo tamanho (post_max_size).

A variável **$_POST** – Exemplo parte HTML

Formulário HTML (salve como index2.php)

```
<html>
<body>
   <form action='teste2.php'
 method='post'>
     Sua cor preferida: <input
 type='text' name='cor' />
     <input type='submit'
 value='Enviar' />
   </form>
</body>
</html>
```

A variável **$_POST** – Exemplo parte PHP

Script PHP (teste2.php):

```
<?php
        $cor = $_POST[‘cor’];
        echo ‘Minha cor
 preferida é ’ . $cor;
?>
```

A variável $_GET

A variável **$_GET**

- É um array associativo de variáveis passadas para o script atual via o método HTTP GET;
- É usada para receber valores de formulário que foram submetidas através do método **GET**;
- O valor GET é indicado no atributo **method** da tag HTML **form**;
- Informações enviadas via formulários através do método GET são visíveis para os usuários (visíveis na barra de endereços do navegador);

A variável **$_GET** – Exemplo parte HTML

Formulário HTML:

```
<html>
<body>
 <form action='teste2.php'
 method='get'>
  Sua cor preferida: <input
 type='text' name='cor' />
  <input type='submit'
 value='Enviar' />
 </form>
</body>
</html>
```

A variável **$_GET** – Exemplo parte PHP

Script PHP (teste2.php):

```
<?php
        $cor = $_GET[‘cor’];

        echo ‘Minha cor
 preferida é ’ . $cor;
?>
```

A variável $_REQUEST

A variável **$_REQUEST**

- É um array associativo que por padrão contém informações de $_GET, $_POST e $_COOKIE.
- É usada para receber valores de formulário que foram submetidas através do método **GET ou POST**;
- Não importa qual método de envio foi definido no formulário HTML, a função $_REQUEST obterá os dados enviados, reconhecendo automaticamente o padrão de envio de dados.

A variável **$_REQUEST** – Exemplo HTML

Formulário HTML:

```
<html>
<body>
   <form action='teste2.php' method='get'>
      Sua cor preferida: <input type='text' name='cor' />
      <input type='submit' value='Enviar' />
   </form>
</body>
</html>
```

A variável **$_REQUEST** – Exemplo PHP

Script PHP (teste2.php):

```
<?php
        $cor = $_REQUEST[‘cor’];

        echo ‘Minha cor
 preferida é ’ . $cor;
?>
```

Exercícios

Para Praticar…

Exercício 1

- Faça um documento HTML que permita ao usuário digitar o nome de um aluno e suas duas notas na disciplina de Programação Web;
- Em seguida faça um script PHP que receba os dados submetidos a partir do documento HTML anterior, calcula a média do aluno e exiba a seguinte mensagem: “O aluno XXXXX ficou com XX,X de média”;
- Use o método POST no formulário!

Exercício 2

Um funcionário recebe um salário fixo mais 4% de comissão sobre as vendas. Faça uma aplicação Web, seguindo os seguintes requisitos:

- Faça um documento HTML que permita ao usuário digitar o salário fixo de um funcionário e o valor total apurado em vendas;
- Faça um script PHP que receba os dados submetidos pelo documento HTML anterior (método GET). O script deverá calcular e mostrar a comissão e o salário final do funcionário";

Exercício 3

Faça um documento HTML que permita ao usuário digitar a base maior, a base menor e a altura de um trapézio. Em seguida, Faça um script PHP que receba os dados submetidos pelo documento HTML anterior (use a variável $_REQUEST no script e o método POST no HTML), calcula e exiba sua área;

$$área\,do\,trapézio = \frac{(base\,maior + base\,menor)}{2} * altura$$

Misturando Código HTML e PHP

- As páginas html podem conter código PHP.
  - Para isso é necessário que elas sejam executadas a partir do servidor PHP