INSTITUTO FEDERAL
Ceará

Campus
Tianguá

DESENVOLVIMENTO WEB

CURSO BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

SEMESTRE: 7º

Prof.: Me. Rhyan Ximenes

Lattes: lattes.cnpq.br/2089613781353862

Ferramentas Utilizadas

- **Wamp server;**
- **Sublime text;**
- **MySQL.**

Ferramentas Utilizadas

- Wamp server;
- Xampp;
- PHP Triad;
- Easy PHP;
- Entre outros.

W – Windows
A – Apache
M – MySQL
P - PHP

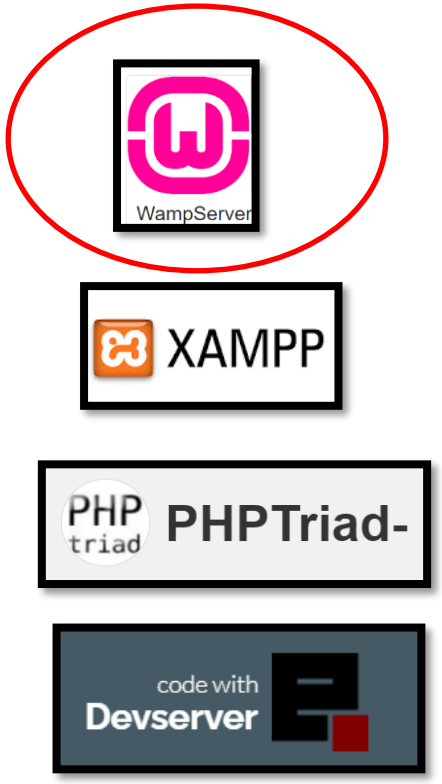


Figura 1. Exemplos de web server
Fonte: wikipedia.com/

Figura 2. phpMyAdmin para administração MySQL
Fonte: Próprio autor

Ferramentas Utilizadas

- Sublime text;

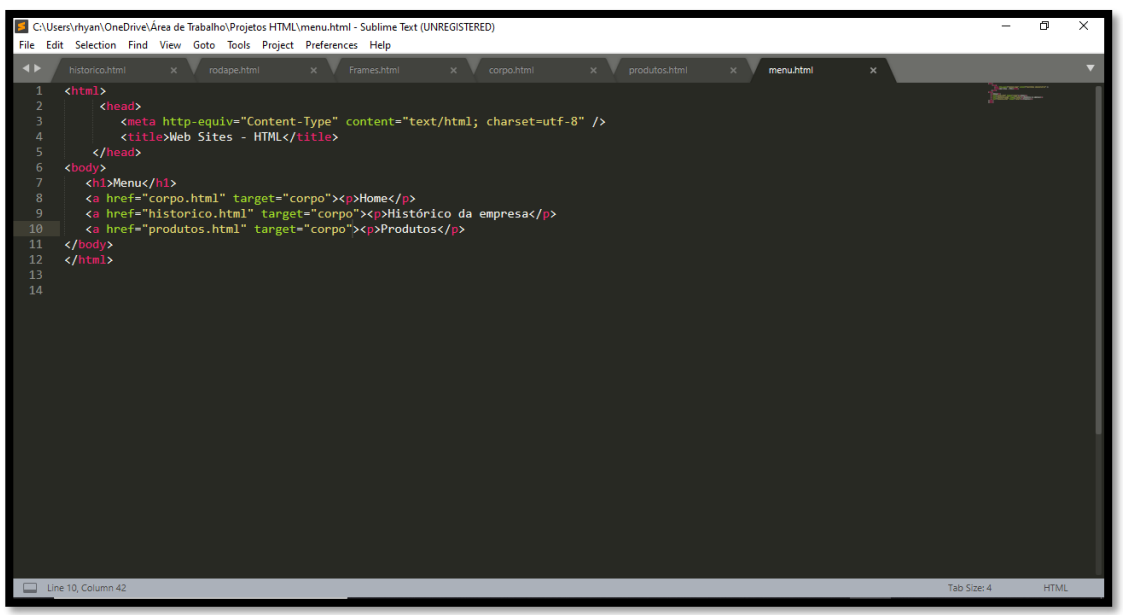


Figura 3. Sublime text 3
Fonte: Própria autor

Ferramentas Utilizadas

- MySQL

Figura 4. MySQL Workbench
Fonte: Próprio autor

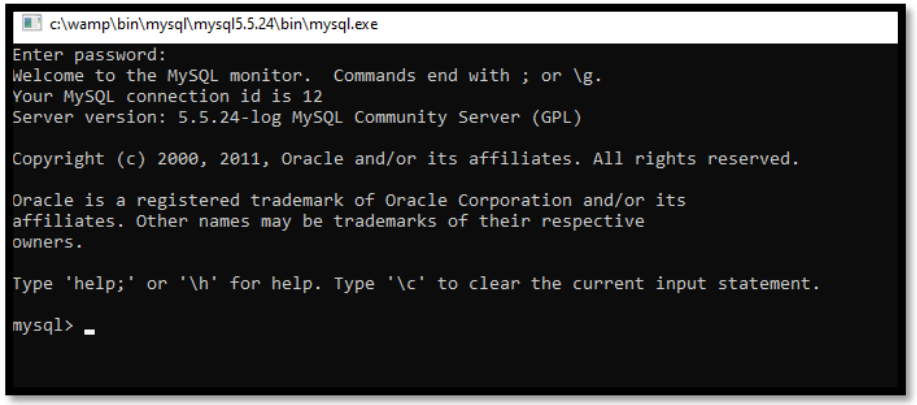
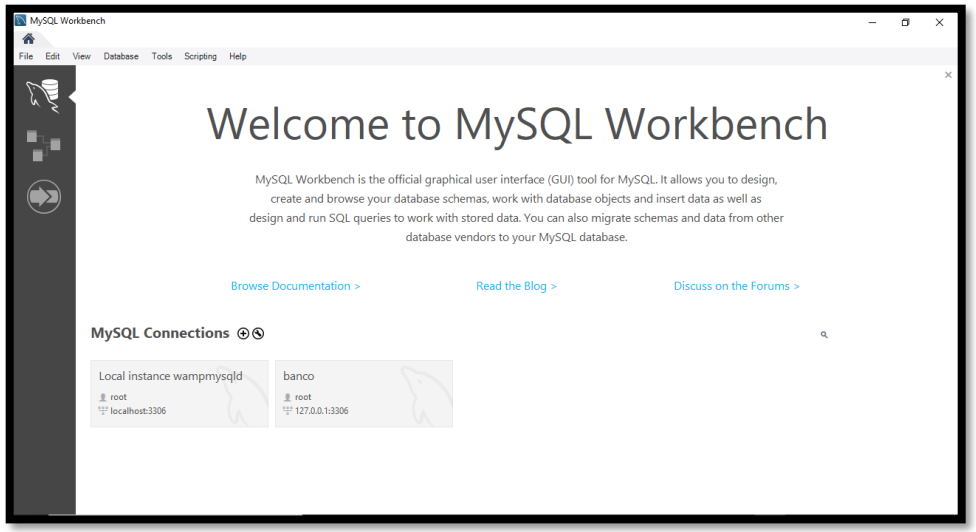


Figura 5. Console do MySQL
Fonte: Próprio autor



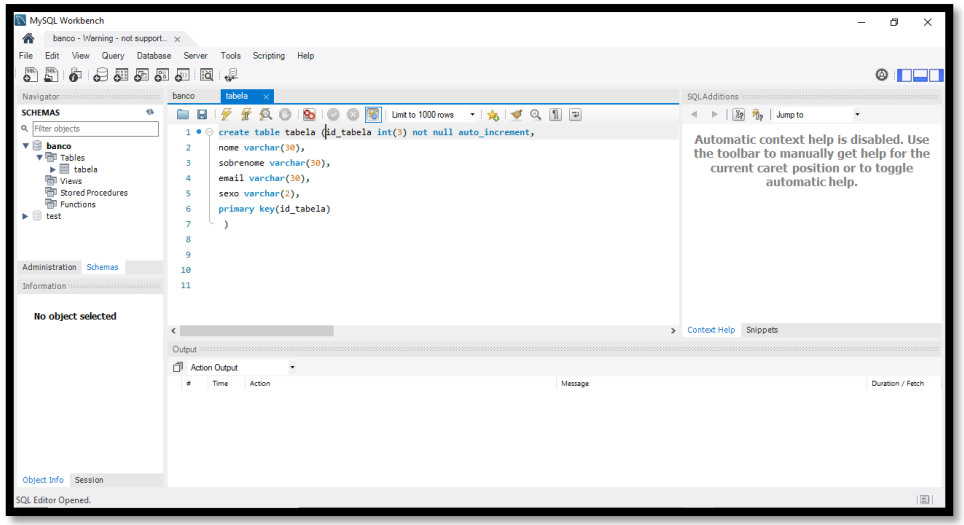


Figura 6. Workbench
Fonte: Próprio autor

O que é o PHP?

- PHP é uma linguagem de programação;
- Multiparadigma;
- Possui tipagem dinâmica;
- Voltada para o desenvolvimento de websites dinâmicos;

Figura 7. PHP (um acrônimo recursivo para **PHP**: *Hypertext Preprocessor*)
Fonte: https://blog.dankicode.com/

O que pode ser feito com PHP?

- Podem ser criados websites capazes de:
    - Fazerem diversas operações em banco de dados;
    - Enviar e-mail utilizando serviços de e-mail (*sendmail, postfix, protonmail*);
    - Trabalhar com dados enviados por formulário;

Como funciona o PHP?

- O funcionamento baseia-se na requisição de um cliente:
    - (Internet Explorer, Firefox, Safari, Opera, etc.)
- A um servidor:
    - (*Apache, Internet Information Service, Glassfish, Tomcat*, etc.)
- E na resposta de um servidor a um cliente;

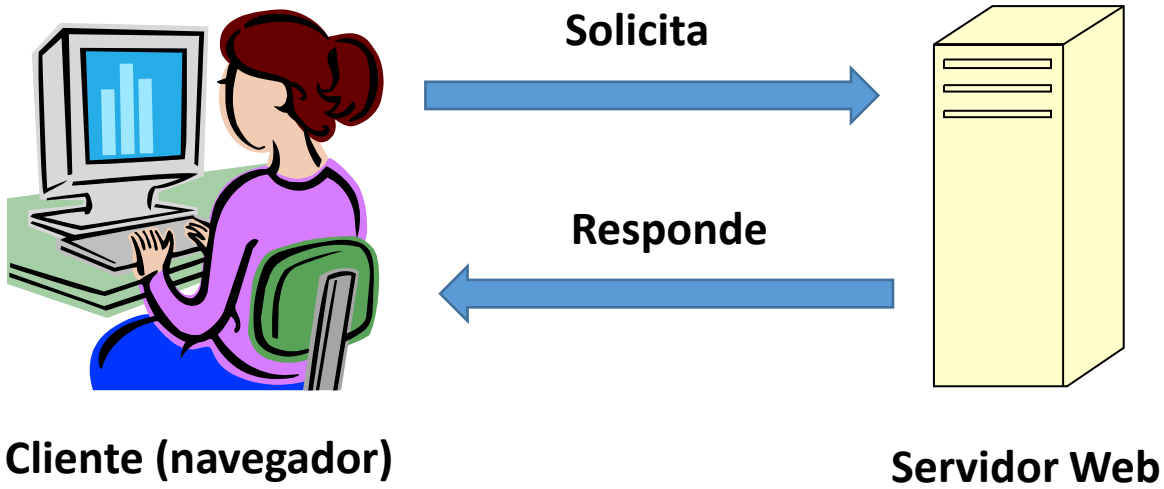


Figura 8. Cliente-servidor
Fonte: Próprio autor

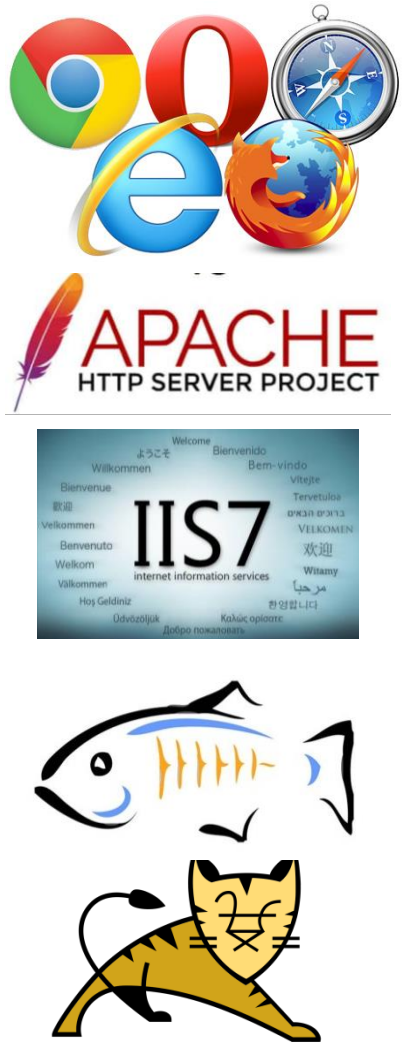


Figura 9. Navegador, e Servidores Web

Como funciona o PHP?

- O navegador funciona como um interpretador (Compilador x Interpretador);
- O navegador (cliente) é capaz de interpretar somente HTML, Javascript e CSS;
- PHP é interpretado no servidor que aciona o interpretador PHP, que faz as operações necessárias (como por exemplo acesso a banco de dados), retornando uma resposta ao servidor web;

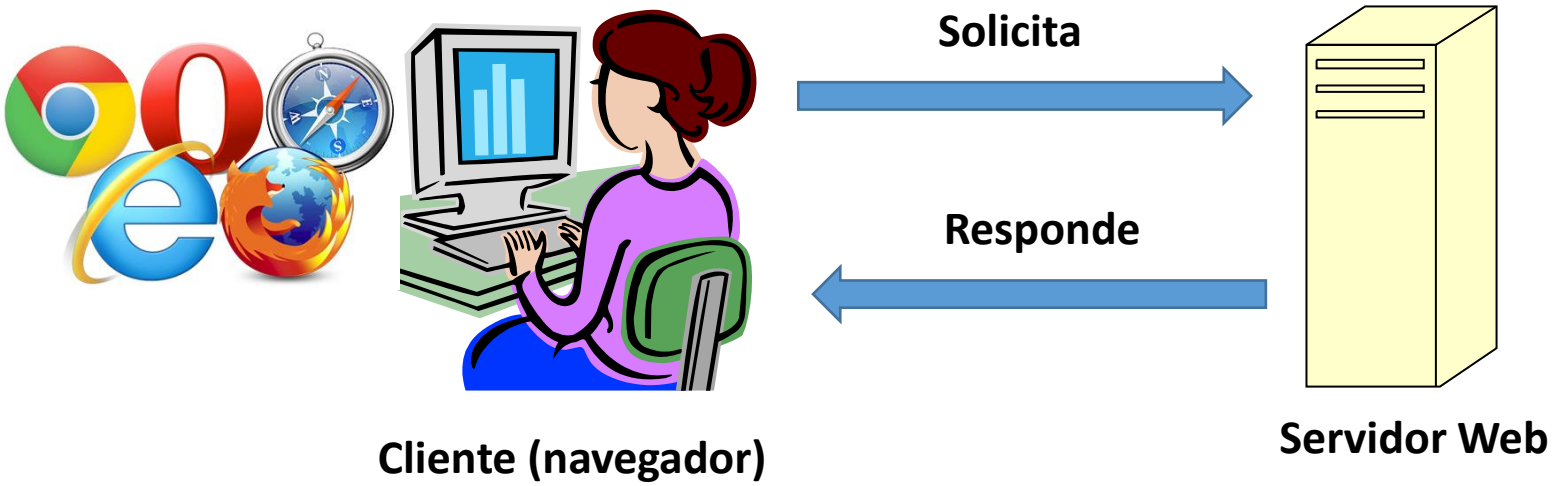


Figura 10. Cliente-servidor
Fonte: Próprio autor e https://www.showmetech.com.br/

Como funciona o PHP?

- O servidor web:
    - Interpreta a resposta, transforma a mesma em HTML e envia ao cliente;
- O cliente (navegador) :
    - Interpreta o HTML e exibe na tela o resultado da interpretação.

Server-side (lado do servidor);

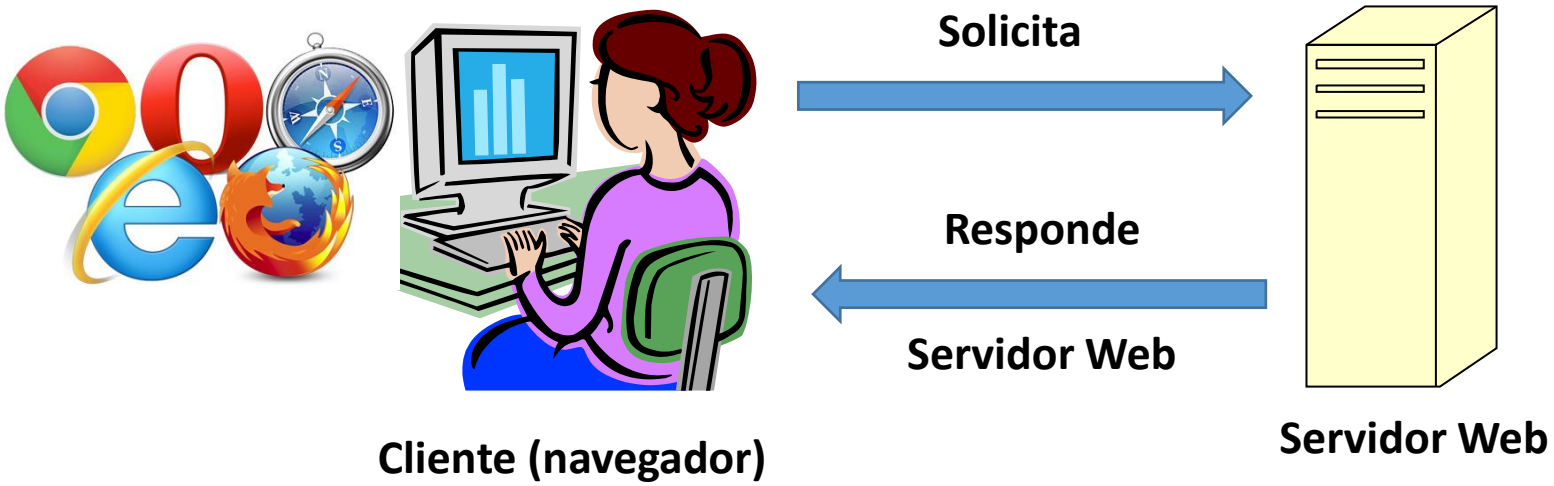


Figura 11. Cliente-servidor
Fonte: Próprio autor e https://www.showmetech.com.br/

Como funciona o PHP?

- O que o cliente não ver é a comunicação entre a aplicação de uma base de dados é feita pelo server-side (lado do servidor);
- Resumindo...
    - Tudo que acontece do lado do navegador (*front-end*) é *client-side*;
    - Tudo que acontece do lado do servidor (*back-end*) é *server-side*.

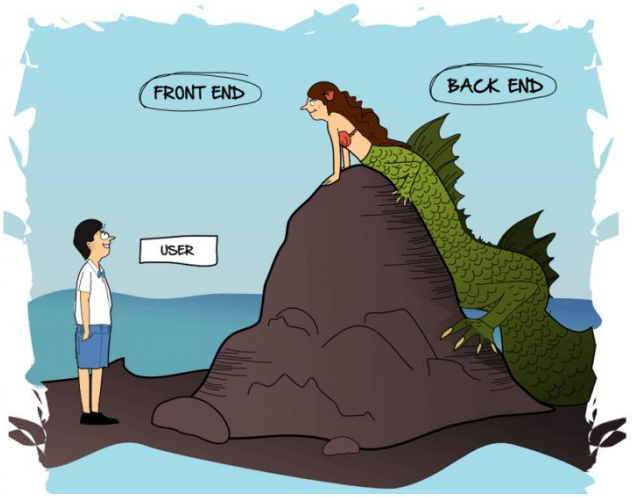


Figura 12 . Back-end X Front-end
Fonte: https://eufacoprogramas.com/programacao-back-end-o-que-e/

Como funciona o PHP?

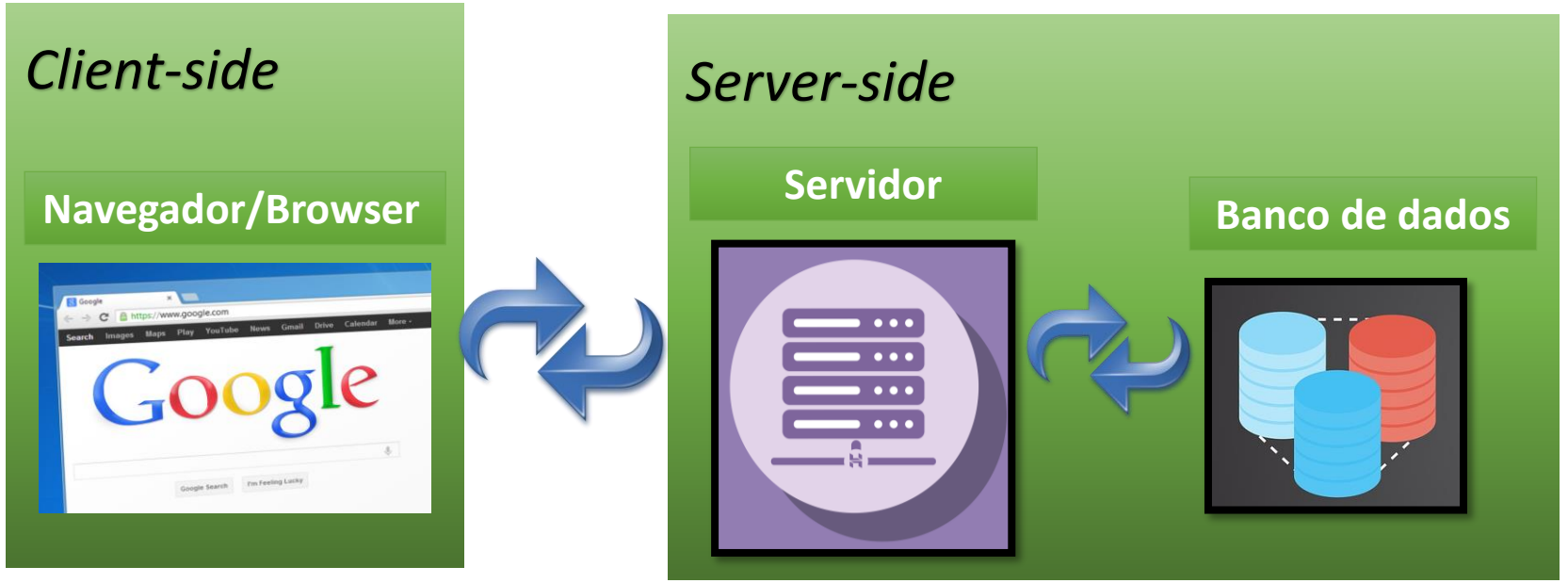


Figura 13 . Funcionamento do client-side e Funcionamento server-side
Fonte: Adaptado de https://bityli.com/hEHwHe

PHP é indicado para...

- Desenvolvimento de websites dinâmicos;
- E indicado para construção de sites simples;
- É porta de entrada para desenvolvimento web;
- Não é recomendado para grandes sistemas web;
- Resposta rápida para problemas pequenos;
- Contrário ao Java ou .NET que não se usa para sites dinâmicos que acessa um banco de dados com três tabelas;
- Para mais velocidade pode ser utilizado um dos frameworks MVC para PHP;

Mãos a Prática!

- No PHP todo o código fica entre **<?php //código ?>**.
- Porém dentro das tags do HTML.

Sintaxe:

```
<html>
        <head>
                <title>Título da Página</title>
        </head>
        <body>
                <?php
                  // Código PHP aqui!
                ?>
        </body>
</html>
```

Figura 14. Sintaxe do PHP
Fonte: Adaptado de Welling e Thomson (2003)

Olá mundo!

```
<html>
        <head>
                <title>Título da Página</title>
        </head>
        <body>
          <?php
           echo “<h1>Olá Mundo!</h1>”;
           print(‘Site criado em PHP!’);
          ?>
        </body>
</html>
```

Figura 15. Algoritmo olá mundo
Fonte: Adaptado de Welling e Thomson (2003)

echo → É um comando que imprime uma ou mais variáveis ou textos, onde os mesmos são colocados em aspas simples ' ' ou duplas “ ”.
print → É uma função que imprime uma String(texto) no console.

Variáveis em PHP

- Utiliza-se um cifrão ($) na declaração;
- Possuem tipagem dinâmica e fraca;
- Tipagem dinâmica significa que uma variável assume um tipo de acordo com o valor que lhe é atribuído;
- Tipagem fraca significa que você pode trabalhar de forma livre com as variáveis;
- Geralmente elas são declaradas e inicializadas e armazenadas na memória RAM do servidor web, motivo pelo qual os servidores precisam de grande quantidades de memória.

```
<?php
  $nome = “João”;
  $sobrenome = “Hermanoteu”;
  $idade = 23;
  echo “$nome   $sobrenome $idade “;
?>
```

Figura 16. Algoritmo imprime nome, sobrenome e idade
Fonte: Próprio autor

Tipos de Variáveis

Tabela 1. Tipos primitivos para variáveis

Tipo	Valor
Booleano	True, false
Inteiro	{0,1,2,3,4,...}
Ponto flutuante	{31.20,15.28,0.75,...}
Numérico	{inteiro, ponto flutuante,...}
String	{“j”,’endereco, ‘cep’,...}

Conversão de Variáveis

Tabela 2. Conversão de tipos

Type Casting	Descrição
(int),(integer)	Converte em inteiro.
(real),(float),(double)	Converte em ponto flutunte.
(string)	Converte em string.
(object)	Converte em objeto.

```
<?php
    $a = (int) (4.56 + 120+0.75-0.21);
    echo $a;
?>
```

Figura 17. Algoritmo com casting
Fonte: Próprio autor

Operadores

Tabela 3. Operadores e descrição

Operadores	Descrição
=	Atribuição simples.
+=	Soma, depois atribui.
-=	Subtrai, depois atribui.
*=	Multiplica, depois atribui.
/=	Divide, depois atribui.
%=	Modulo(resto) da divisão, depois atribui.
.=	Concatena, depois atribui.

Operadores de strings: Usa-se '.' ou '.='

Operadores

Operadores de decremento e incremento:

Tabela 4. Operadores e descrição

Operadores	Descrição
++$a	Pré-incremento. Incrementa $a em um e, então, retorna $a.
$a++	Pós-incremento. Retorna $a, então, incrementa $a em um.
- -$a	Pré-decremento. Decrementa $a em um e, então, retorna $a.
$a- -	Pós-decremento. Retorna $a, então, decrementa $a em um.

Operadores

Operadores de decremento e incremento:

```
<?php
    $num = 45;
    print(++$num);  // incrementa 1 , depois imprime 46
    print($num++);  // imprime 46 e depois incrementa 1
    print($num);     / / imprime $num com 47
    print(- -num);   //  decrementa 1 e imprime 46
    print(num- -);  // imprime 46, depois decrementa 1
    print($num);   // imprime 45
?>
```

Figura 18. Algoritmo com operadores
Fonte: Baseado em Welling e Thomson (2003)

Operadores Relacionais

Tabela 5. Operadores e descrição

Comparadores	Descrição
==	Igual. Resulta em TRUE se as expressões forem iguais.
===	Idêntico. Resulta em TRUE se as expressões forem iguais e do mesmo tipo de dados.
!= ou <>	Diferente. Resulta verdadeiro se as variáveis foram diferentes.
<	Menor ou menor que. Resulta TRUE se a primeira expressão for menor.
>	Maior ou maior que. Resulta TRUE se a primeira expressão for maior.
<=	Menor ou igual. Resulta TRUE se a primeira expressão for menor ou igual.
>=	Maior ou igual. Resulta TRUE se a primeira expressão for maior ou igual.

Entrada de Valores

- Existem três arrays associativos que podemos usar para receber dados de formulários HTML em PHP. São eles:
- **$_GET**, **$_POST** e **$_REQUEST**.

- **$_GET**, **$_POST** são usados de acordo com o método de envio de informações definido para o formulário;
- Já o array **$_REQUEST** recupera dados tanto de formulários que utilizam **GET** quanto **POST**.

- **action** - Caminho do arquivo que receberá os dados do formulário ao ser enviado.

Entrada de Valores

- **$_GET** → retorna um array com todos os valores enviados e seus supostos índices.

```
1  <form method="get" action="gravar.php">
2      Login: <input name="login" type="text">
3      Senha: <input name="senha" type="password">
4          <input type="submit" value="Logar">
5  </form>
```

Figura 19. Algoritmo com $_get
Fonte: Adaptado de Welling e Thomson (2003)

- **$_GET ['nome_da_campo']** → retorna o valor passado pelo campo.

```
1  <?php
2
3  echo $_GET['login']."<br>";
4  echo $_GET['senha']."<br>";
5
6  ?>
```

Figura 20. Algoritmo com $_get
Fonte: Adaptado de Welling e Thomson (2003)

Entrada de Valores

- **$_POST** → retorna um array com todos os valores enviados e seus supostos índices.

```
1  <form method="post" action="gravar.php">
2      Login: <input name="login" type="text">
3      Senha: <input name="senha" type="password">
4          <input type="submit" value="Logar">
5  </form>
```

Figura 21. Algoritmo com $_post
Fonte: Adaptado de Welling e Thomson (2003)

- **$_POST['nome_da_campo']** retorna o valor → passado pelo campo.

```
1  <?php
2
3  echo $_POST['login']."<br>";
4  echo $_POST['senha']."<br>";
5
6  ?>
```

Figura 22. Algoritmo com $_post
Fonte: Adaptado de Welling e Thomson (2003)

Entrada de Valores

- O método GET utiliza a URL para enviar dados ao servidor;
- Quando enviamos um formulário pelo método GET, o navegador pega as informações do formulário e coloca junto com a URL de onde o formulário vai ser enviado e envia, separando o endereço da URL dos dados do formulário por um “?” (ponto de interrogação) e “&”.

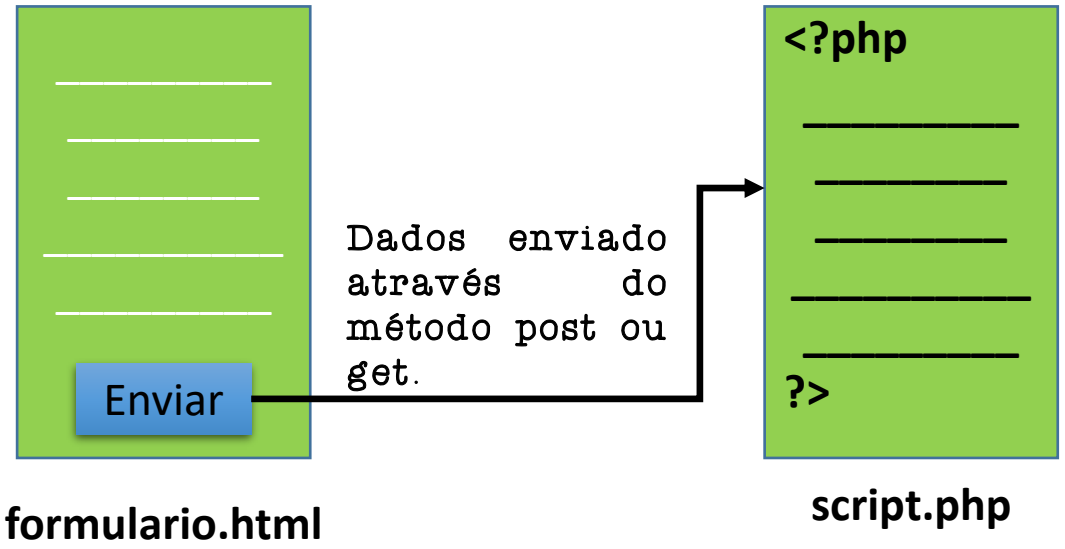


Figura 23. Funcionamento do get ou post
Fonte: Adaptado de Welling e Thomson (2003)

Entrada de Valores

- O método POST é muito semelhante ao método GET, porém a principal diferença está em enviar os dados encapsulado dentro do corpo da mensagem, sua utilização é mais viável quando trabalhamos com informações segura ou que poder ser alteradas somente por eventos do Browser.

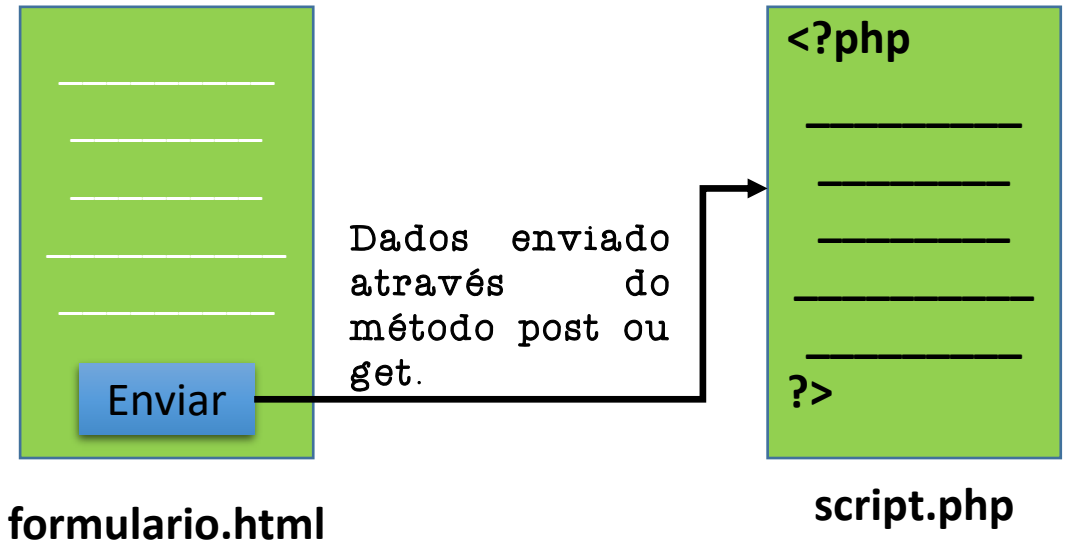


Figura 24. Funcionamento do get ou post
Fonte: Adaptado de Welling e Thomson (2003)

Exemplo de Entrada de Valores

```html
<html>
 <head>
  <title>Recebendo dados do Formulário</title>
 </head>
<body>
      <form method="post" action="receberValor.php">Digite seu nome:<br/>
         <input type="text" name="nome"/>
         <input type="submit" value="Enviar"/>
      </form>
</body>
</html>
```

Figura 25. formulario.html
Fonte: Próprio autor

```php
<?php
   $nome = $_POST['nome'];
   echo "Olá! " .$nome;
?>
```

Figura 26. receberValor.php
Fonte: Próprio autor

$nome = **$_REQUEST['nome']**;

action - Caminho do arquivo que receberá os dados do formulário ao ser enviado.

Operadores

Exemplo de operadores relacionais com uso de ternário:

```
<?php
    $num1 = 45;
    $num2 = 50;
    $num3 =  52.5;
    echo $num2 = = $num3 ? “verdadeiro” : “falso”;  // verdadeiro
    echo $num2 = = = $num3 ? “verdadeiro” : “falso”;  // falso
    echo $num1 <= $num3 ? “verdadeiro” : “falso”;  // verdadeiro
    echo $num2 > = $num3 ? “verdadeiro” : “falso”;  // falso
?>
```

Figura 27. Algoritmo com uso do operador ternário
Fonte: Próprio autor

Operadores Lógicos

Tabela 6. Operadores lógicos

Operador	Descrição
($a and $b)	E : Verdadeiro se tanto $a quanto $b forem verdadeiros.
($a or $b)	OU : Verdadeiro se $a ou $b forem verdadeiros.
($a xor $b)	XOR : Verdadeiro se $a ou $b forem verdadeiro, de forma exclusiva.
(! $a)	NOT : Verdadeiro se $a for falso, usado para inverter o resultado da condição.
($a && $b)	E : Verdadeiro se tanto $a quando $b forem verdadeiros.
($a || $b)	OU : Verdadeiro se $a ou $b forem verdadeiros.

Exemplos:

Exemplo 1:

```
<?php
    $num1 = 15>7;
    $num2 = 50>= 50;
   $num3 = False;
   echo ($num2 and $num1) ? “sim” : “não”; // sim
   echo ($num2 or $num3) ? “sim” : “não”;    // sim
?>
```

Figura 28. Algoritmo01 com exemplo de uso de operador lógico
Fonte: Próprio autor

Exemplo 2:

```
<?php
    $num1 = 15>7;
    $num2 = 50>= 50;
   $num3 = False;
   echo (!$num3) ? “sim” : “não”; // sim
   echo ($num2 && $num3) ? “sim” : “não”;    // não
   echo ($num1 || num2) ? ”sim ” : “não”; //sim
?>
```

Figura 29. Algoritmo02 com exemplo de uso de operador lógico
Fonte: Próprio autor

Referências Bibliográficas

- Apostila projeto e-jovem. Governo do Estado do Ceará.
- DEITEL, H. M. & DEITEL, P. J. Internet e World Wide Web Como Programar. Bookman, 2ª Edição, 2003.
- FREEMAN, Elisabeth. **Use a cabeça!: HTML com CSS e HTML**. Alta books, 2008.
- SIERRA, K., BATES, B., BASHAM, B. Use a Cabeça! Servlets & JSP. Rio de Janeiro: Alta Books, 2005.
- SILVA, Maurício Samy. **Criando sites com HTML: sites de alta qualidade com HTML e CSS**. Novatec Editora, 2008.
- _____. **Construindo sites com CSS e (X) HTML: sites controlados por folhas de estilo em cascata**. Novatec Editora, 2007.
- WELLING, Luke; THOMSON, Laura. **PHP and MySQL Web development**. Sams Publishing, 2003.