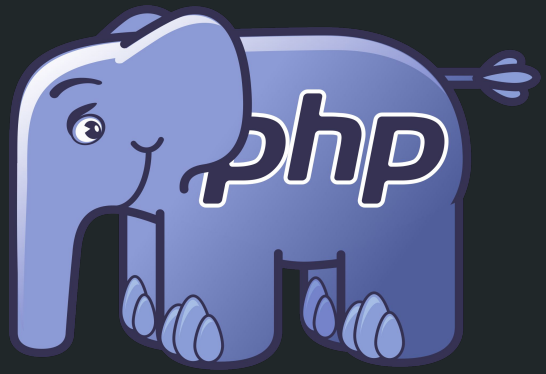
Curso PHP e MySQL



_Fundamentos da Linguagem PHP

> Tags, instruções no html, comentários, variáveis, operadores de atribuição, tipos primitivos, operadores aritméticos e precedência dos operadores.

Adriel Sales

Afinal,
O que é PHP?

- PHP: ***Hypertext Preprocessor***.

- É uma linguagem de script *open source* de uso geral, muito utilizada, e especialmente adequada para o desenvolvimento web.

Um pouco de história

Fonte: https://www.youtube.com/watch?v=YdS9f5IzIe8&t=148s

Currently Supported Versions

Branch	Initial Release		Active Support Until		Security Support Until	
7.4	28 Nov 2019	*2 years, 9 months ago*	28 Nov 2021	*9 months ago*	28 Nov 2022	*in 2 months*
8.0	26 Nov 2020	*1 year, 9 months ago*	26 Nov 2022	*in 2 months*	26 Nov 2023	*in 1 year, 2 months*
8.1	25 Nov 2021	*9 months ago*	25 Nov 2023	*in 1 year, 2 months*	25 Nov 2024	*in 2 years, 2 months*

Or, visualised as a calendar:

Key

Active support	A release that is being actively supported. Reported bugs and security issues are fixed and regular point releases are made.
Security fixes only	A release that is supported for critical security issues only. Releases are only made on an as-needed basis.
End of life	A release that is no longer supported. Users of this release should upgrade as soon as possible, as they may be exposed to unpatched security vulnerabilities.

Fonte: https://www.php.net/supported-versions.php

Por que o PHP?

- O PHP é gratuito;
- Os custos de manutenção de um servidor são muito reduzidos;
- Atualizações consistentes;
- Se integra a quase todos os Bancos de dados;
- Alta curva de aprendizado;
- Grande quantidade de ambientes de desenvolvimento profissionais disponíveis;
- Um grande número de pacotes/bibliotecas disponíveis e prontas para uso.

Universo PHP

Frameworks

Symfony

CakePHP

Code Igniter

Universo PHP

CMS mais usados no mercado

php

MySQL

Joomla!

WordPress

Drupal

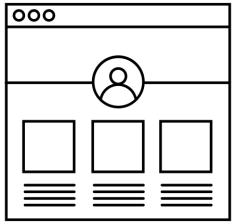
Magento

osCommerce
Open Source E-Commerce Solutions

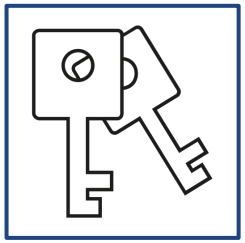
smarty
TEMPLATE ENGINE

zen cart
the art of e-commerce

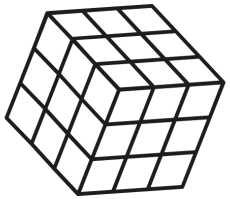
Básico de um sistema web



1. Páginas públicas.



2. Páginas privadas.



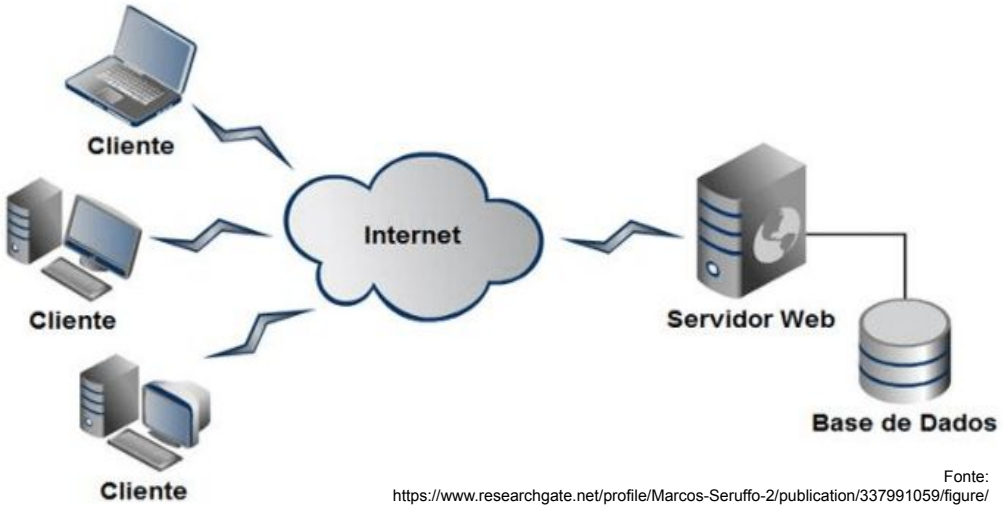
3. Conexão com um banco de dados.

Por onde começar ?

Arquitetura WEB - Cliente-Servidor

Do lado do cliente, os usuários utilizam um programa chamado de navegador (em inglês, browser) para manipular as páginas HTML.

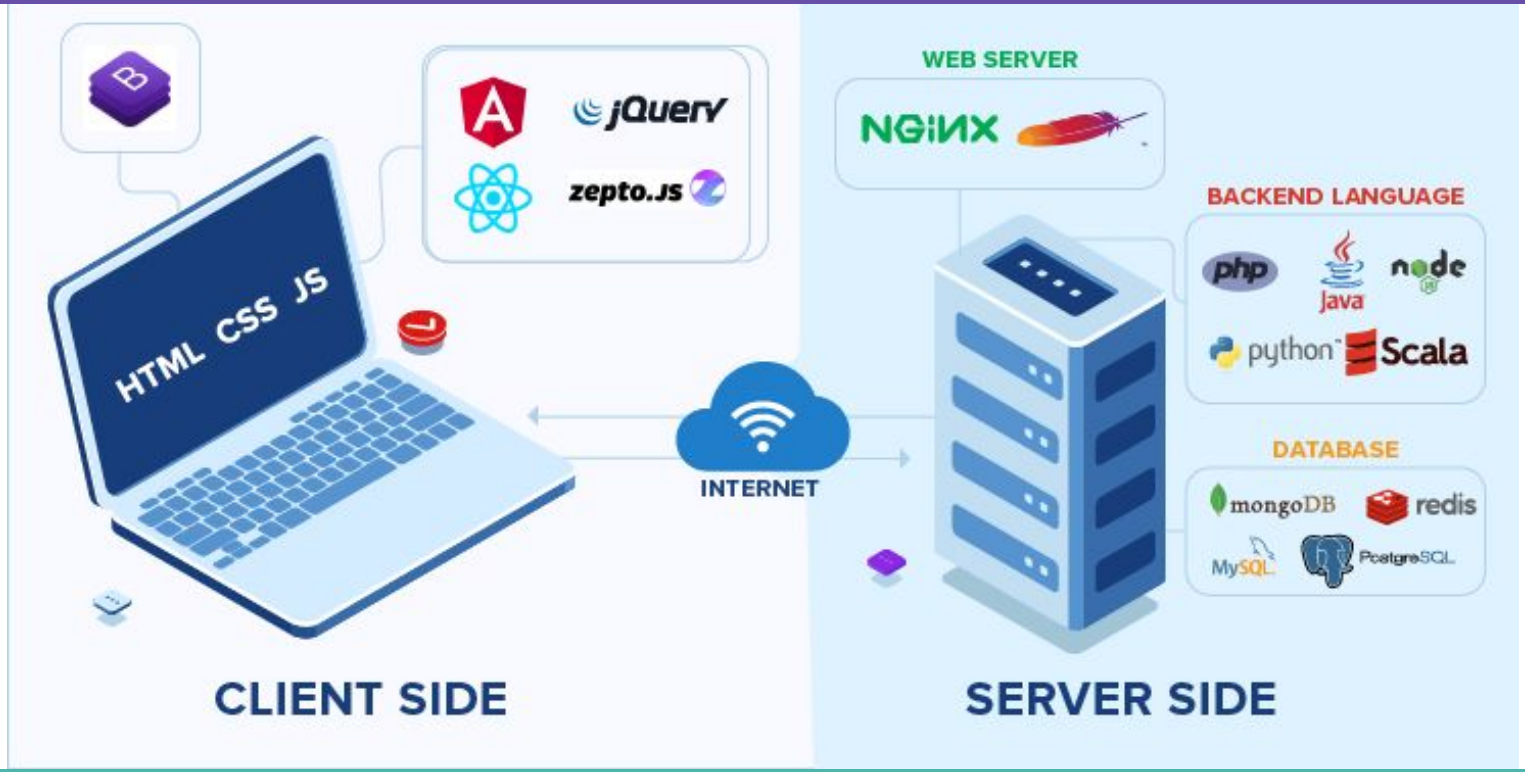
No lado do servidor, fica a camada de armazenamento dos dados, onde são executados um ou mais serviços ou programas que compartilham recursos com os clientes.



Fonte: https://www.researchgate.net/profile/Marcos-Seruffo-2/publication/337991059/figure/fig2/AS:886925120512001@1588470835998/Figura-2-Cenario-proposto-arquitetura-cliente-servidor.png

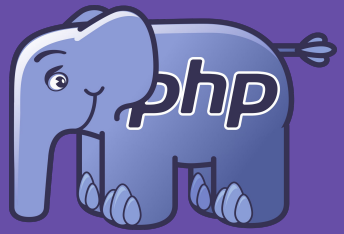
Neste modelo arquitetural, um cliente nunca se comunica diretamente com outro cliente, pois toda a comunicação ocorre diretamente com o servidor.

Front End vs Back End



Iniciaremos nossos estudos pelo BACK END com PHP

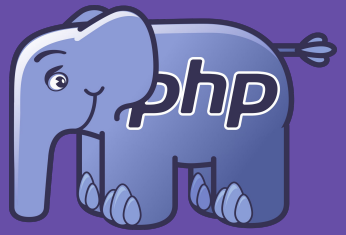
```
<?php
    echo “Vamos lá!”
```



_terminal interativo

O **terminal interativo** é um recurso em linha de comando onde podemos digitar código PHP e executá-lo em tempo real. Para iniciar o terminal digitamos "**php -a**". Quando desejarmos sair, basta executar o comando "**quit**".

```
adriel@work01 MINGW64 ~
$ php -a
Interactive shell

php > echo "Olá mundo!";
Olá mundo!
php > quit

adriel@work01 MINGW64 ~
$
```



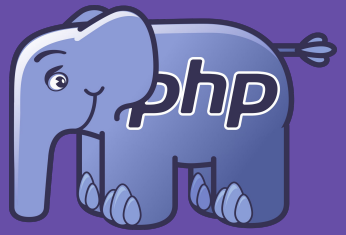
_tags

Quando o PHP interpreta um arquivo ele **procura pelas tags de abertura e fechamento**, **<?php e ?>**, que dizem ao PHP para iniciar ou parar a interpretação do código entre elas.

```
1.  <?php echo 'se você deseja rodar código PHP dentro de documentos XHTML ou XML,
                utilize essas tags'; ?>

2.  Você pode utilizar a tag curta <?= 'imprima essa string' ?>.
    É o equivalente de <?php echo 'print this string' ?>.

3.  <? echo 'Este código está entre tags curtas, mas somente funcionará '.
            'se short_open_tag estiver ativo'; ?>
```

Se um arquivo for código PHP puro, é preferível omitir a tag de fechamento no final do arquivo.

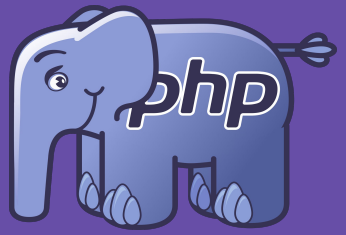
```
<?php
echo "Hello world";

// ... mais código

echo "última instrução";

// o script termina aqui, sem tag de fechamento PHP
```



_escapando HTML

Tudo o que estiver fora das tags PHP é ignorado pelo interpretador. Isso permite **arquivos PHP de conteúdo misto**. Permite que **o PHP seja incluído dentro de documentos HTML**, para, por exemplo, **a criação de templates**.

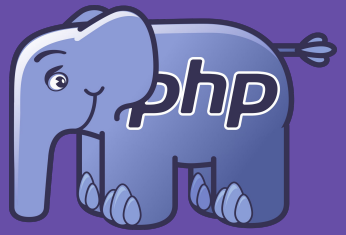
```
<p>Isto vai ser ignorado pelo PHP e exibido pelo navegador.</p>
<?php echo 'Enquanto isto vai ser interpretado.'; ?>
<p>Isto também vai ser ignorado pelo PHP e exibido no navegador.</p>
```



_separação das instruções

As instruções PHP devem ser terminadas com um **ponto-e-vírgula ";"** ao final de cada comando. **Se o comando for único**, você não precisa incluir o ponto-e-vírgula.

```php
<?php
    echo 'Isto é um teste';
?>

<?php echo 'Isto é um teste' ?>

<?php echo 'Nós omitimos a última tag de fechamento';
```



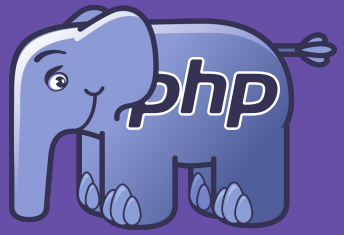
_comentários

O PHP suporta comentários no estilo 'C', 'C++' e do Unix shell (estilo Perl).

```
<?php
    echo 'Isto é um teste'; // Estilo de comentário de uma linha em  c++
    /* Este é um comentário de múltiplas linhas
       ainda outra linha de comentário */
    echo 'Isto é ainda outro teste';
    echo 'Um teste final'; # Este é um comentário de uma linha no estilo shell
?>

<h1>Isto é um <?php # echo 'simples';?> exemplo.</h1>
<p>O cabeçalho acima irá dizer 'Isto é um exemplo'.</p>
```

Comentários no estilo 'C' terminam ao primeiro */ encontrado. Tenha certeza de não aninhar comentários no estilo 'C'. É fácil fazer este equívoco se estiver tentando comentar grandes blocos de código.

```
<?php
 /*
    echo 'Isto é um teste'; /* Este comentário irá causar um problema */
 */
?>
```

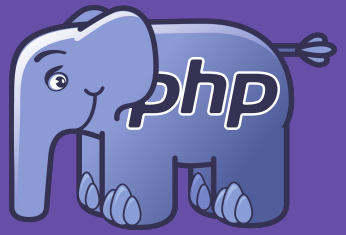


_variáveis

As variáveis no PHP **são representadas por um cifrão "$"** seguido pelo nome da variável. Os nomes de variável **são case-sensitive** (ou seja, há distinção entre maiúsculas e minúsculas).

Nomes de variável seguem as mesmas regras como outros rótulos no PHP. Um **nome de variável válido inicia-se com uma letra ou sublinhado**, seguido de qualquer número de letras, números ou sublinhados.

```php
<?php
$var = 'Bob';
$Var = 'Joe';
echo "$var, $Var";      // exibe "Bob, Joe"

$4site = 'not yet';     // inválido; começa com um número
$_4site = 'not yet';    // válido; começa com um sublinhado
$täyte = 'mansikka';    // válido; 'ä' é um caracter ASCII (extendido) 228
```



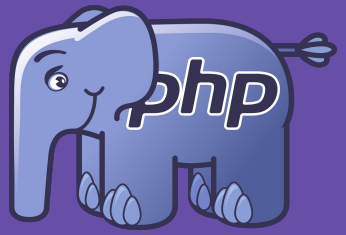
_operadores de atribuição

O operador básico de atribuição é "**=**". Pode-se dizer que **o operando da esquerda recebe o valor da expressão da direita**, ou seja, "é definido para ou recebe". Logo, o valor de "**$a = 3**" é **3**. Lê-se: $a recebe 3.

```php
<?php

$a = ($b = 4) + 5; // $a é igual a 9 agora e $b foi definido como 4.

?>
```

Além do operador básico de atribuição, há "operadores combinados" para todos os operadores aritméticos, que permitem a você pegar um valor de uma expressão e então usar seu próprio valor para o resultado daquela expressão.

```php
<?php

$a = 3;
$a += 5; // define $a para 8, como se disséssemos: $a = $a + 5;
$b = "Bom ";
$b .= "Dia!"; // define $b para "Bom Dia!", como em $b = $b . "Dia!";

?>
```

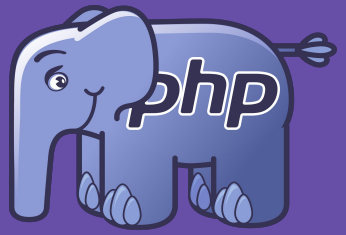


_tipos

O PHP suporta **dez tipos primitivos**.

- **Quatro tipos escalares:**
  - bool, int, float/double, string.
- **Quatro tipos compostos:**
  - array, object, callable, iterable.
- **E finalmente dois tipos especiais:**
  - resource e NULL.

```php
<?php
$a_bool = TRUE;   // um booleano
$a_str  = "foo";  // uma string
$a_str2 = 'foo';  // uma string
$an_int = 12;     // um inteiro
$a = 1.234;       // um float ou double
```

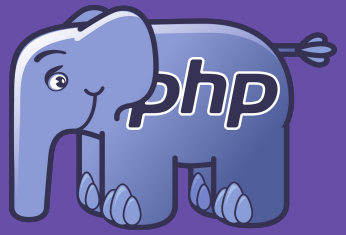


_operadores Aritméticos

São os mesmos utilizados na aritmética básica:

- Adição (+)
- Subtração (-)
- Multiplicação (*)
- Divisão (/)

Exemplo	Nome	Resultado
+$a	Identidade	Conversão de $a para int ou float conforme apropriado.
-$a	Negação	Oposto de $a.
$a + $b	Adição	Soma de $a e $b.
$a - $b	Subtração	Diferença entre $a e $b.
$a * $b	Multiplicação	Produto de $a e $b.
$a / $b	Divisão	Quociente de $a e $b.
$a % $b	Módulo	Resto de $a dividido por $b.
$a ** $b	Exponencial	Resultado de $a elevado a $b. Introduzido no PHP 5.6.



_precedência de operadores

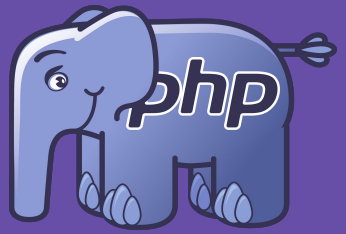
A precedência de um operador especifica quem tem mais prioridade quando há duas delas juntas. Por exemplo, na expressão **1 + 5 * 3**, a resposta é **16** e não **18** porque o operador de multiplicação ("*") tem prioridade de precedência que o operador de adição ("+"). Parênteses podem ser utilizados para forçar a precedência, se necessário. Assim, **(1 + 5) * 3** é avaliado como **18**.

```php
$resultado = 1 + 5 * 3;
echo $resultado; //resultado é 16

$resultado = (1 + 5) * 3;
echo $resultado; //resultado é 18
```

Quando operadores **têm precedência igual** a associatividade decide como os operadores são agrupados. Por exemplo "-" **é associado à esquerda**, de forma que **1 - 2 - 3** é agrupado como **(1 - 2) - 3** e resulta em **-4**. "=" por outro lado associa para a direita, de forma que **$a = $b = $**c é agrupado como **$a = ($b = $c)**.

```php
$resultado3 = 1 - (2 - 3);
echo $resultado3; //resultado é 2

$a = 2;
$b = 5;
$c = 1;
echo $a = $b = $c; //resultado é 1
```

O uso de parênteses, embora não estritamente necessário, **pode melhorar a leitura do código** ao deixar o agrupamento explícito em vez de depender da associatividade e precedências implícitas.



_atividades de fixação

Atividades: Tipos, variáveis e operadores aritméticos.

1. Escreva um programa PHP que armazene o valor 20 em uma variável A e o valor 5 em uma variável B. Em seguida, some as variáveis A com B e exiba o resultado.

2. Escreva um programa PHP que armazene o valor 10 em uma variável A, o valor 2 em uma variável B e o valor 4 em uma variável C. A seguir, some as variáveis C com B, multiplique por A e exiba o resultado.

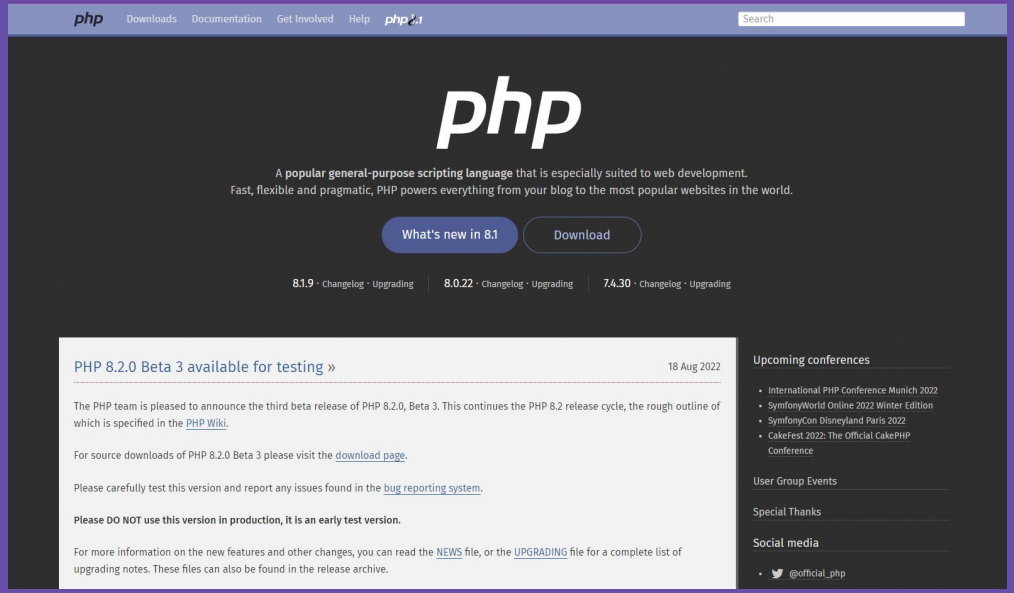
3. Escreva um programa PHP que armazene o valor 1000 em uma variável X e o valor 600 em uma variável Y. A seguir, subtraía Y de X e exiba o resultado.

4. Escreva um programa PHP que armazene o valor 937.0 em uma variável “salarioMinimo”e o valor 400.0 em uma variável “aumentoSalarial”. A seguir, some os dois e exiba o salário final como resultado.

5. Escreva um programa PHP que armazene o valor 100 em uma variável “laranjas” e o valor 5 em uma variável “cestas”. A seguir, exiba o resultado da divisão do total de laranjas pela quantidade de cestas.

6. Escreva um programa PHP que armazene o valor 10 em uma variável “A” e o valor 20 em uma variável “B”. A seguir (utilizando apenas atribuições entre variáveis) troque os seus conteúdos fazendo com que o valor que está em “A” passe para “B” e vice-versa. Ao final, escrever os valores que ficaram armazenados nas variáveis.

Fonte? Vá na fonte!



https://www.php.net

https://github.com/adrielacademico

Professor:
Adriel Sales