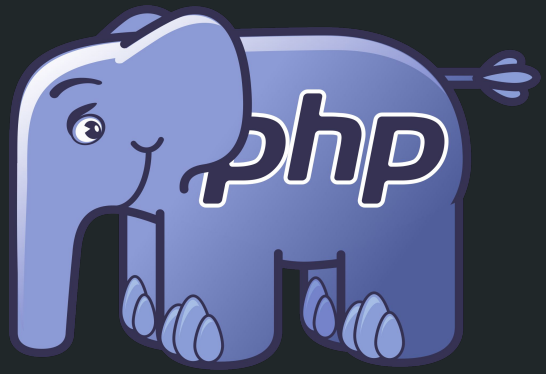
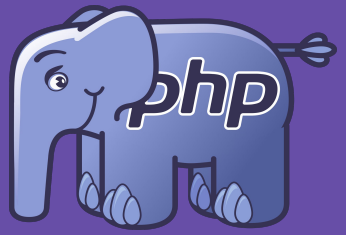
Curso PHP e MySQL



_Arrays em PHP

> Arrays númericos/indexados, arrays associativos, arrays multidimensionais e funções para inclusão de conteúdo (include, include_once, require e require_once).

Adriel Sales



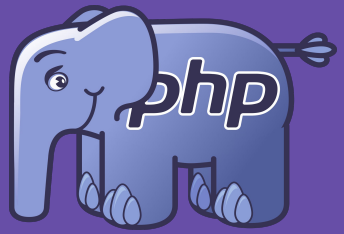
_arrays

Array é um tipo de dado, assim como integer, float, string ou boolean. Contudo, um array **pode armazenar mais de um valor**, relacionando-o a uma chave. No PHP, um mesmo array pode conter diferentes tipos de dados, incluindo novos arrays.

A sintaxe de um php array funciona da seguinte forma:

```
array (

  primeira_chave => primeiro_valor,

  segunda_chave => segundo_valor,

  terceira_chave => terceiro_valor,

  ...

);
```

Ou seja,
Um array é uma variável que nos permite armazenar mais de um valor nela.



_arrays

Tipos de arrays:

Arrays númericos/indexados:

São arrays que têm um índice numérico como chave.

```
$cars = array("Volvo", "BMW", "Toyota");
```

or the index can be assigned manually:

```
$cars[0] = "Volvo";
$cars[1] = "BMW";
$cars[2] = "Toyota";
```

Arrays associativos:

São arrays que têm chaves nomeadas.

```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
```

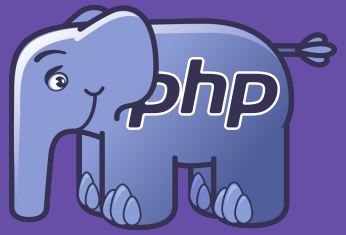
or:

```
$age['Peter'] = "35";
$age['Ben'] = "37";
$age['Joe'] = "43";
```

Arrays multidimensionais:

são arrays que têm um ou mais arrays dentro dele. Arrays multidimensionais também são chamados matrizes.

```
$cars = array (
  array("Volvo",22,18),
  array("BMW",15,13),
  array("Saab",5,2),
  array("Land Rover",17,15)
);
```

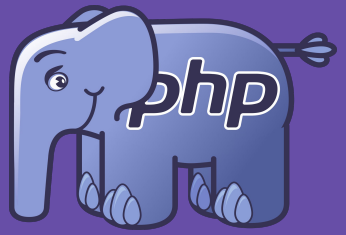


_arrays

Array é um tipo de dado, assim como integer, float, string ou boolean. Contudo, um array **pode armazenar mais de um valor**, relacionando-o a uma chave. No PHP, um mesmo array pode conter diferentes tipos de dados, incluindo novos arrays.

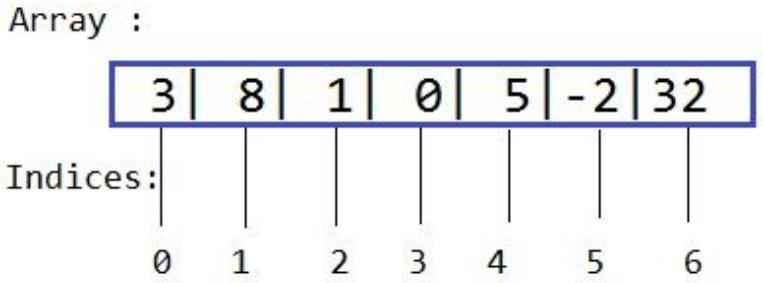
```
array(
    chave  => valor,
    chave2 => valor2,
    chave3 => valor3,
    ...
)
```

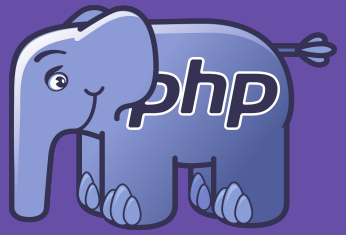
```php
<?php
$array = array(
    "foo" => "bar",
    "bar" => "foo",
);

// Utilizando a sintaxe curta
$array = [
    "foo" => "bar",
    "bar" => "foo",
];
?>
```



_arrays

Geralmente, estão estruturados com índices e seus respectivos valores:

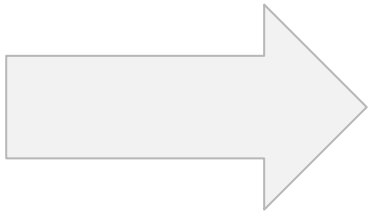




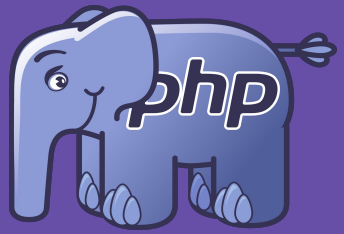
_arrays

A **chave** de um array **pode ser um int ou uma string**.
O **valor** pode ser de **qualquer tipo**.

```php
<?php
$array = array(
    "foo" => "bar",
    "bar" => "foo",
    100   => -100,
    -100  => 100,
);
var_dump($array);
?>
```



```
array(4) {
  ["foo"]=>
  string(3) "bar"
  ["bar"]=>
  string(3) "foo"
  [100]=>
  int(-100)
  [-100]=>
  int(100)
}
```

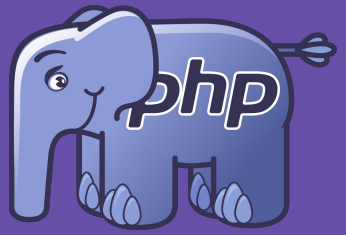


_arrays

```php
<?php

//Declaração de arrays:

$array_exemplo1 = array(); //assim

$array_exemplo2 = []; //ou assim

$cars = array("Volvo", "BMW", "Toyota");
echo "I like $cars[0], $cars[1] and $cars[2]" . PHP_EOL;

$cars = ["Volvo", "BMW", "Toyota"];
echo "Yes, I like $cars[0], $cars[1] and $cars[2]" . PHP_EOL;

var_dump($array_exemplo1, $array_exemplo2);

?>
```

Result for 8.1.10:

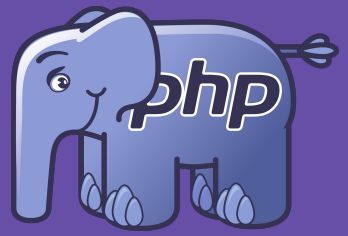
```
I like Volvo, BMW and Toyota
Yes, I like Volvo, BMW and Toyota
array(0) {
}
array(0) {
}
```



_arrays multidimensionais

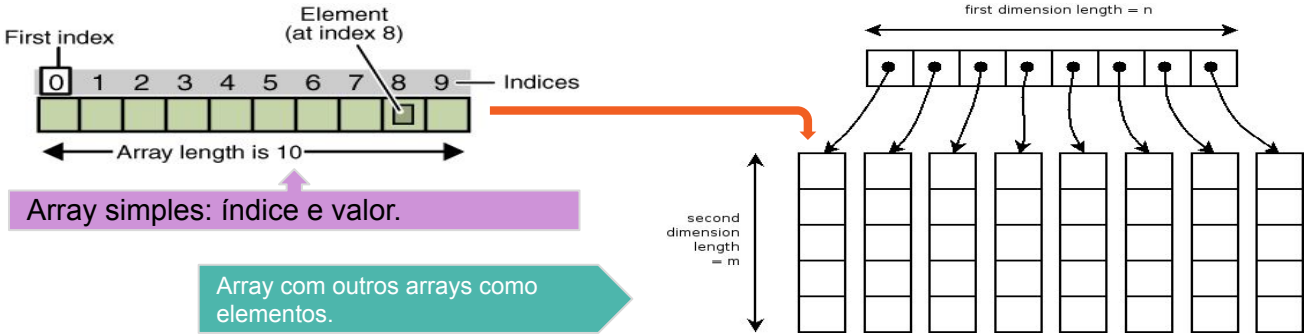
são arrays que têm um ou mais arrays dentro dele.
Arrays multidimensionais também são chamados matrizes.

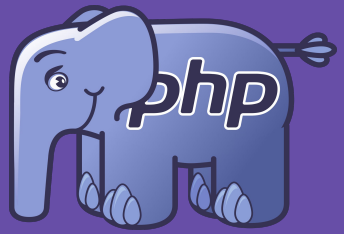
```php
<?php

$turma = [
    [
        'nome' => 'Joao da sIlva',
        'idade' => 18,
        'rg' => '5556666',
        'cpf' => '010.200.256-89',
    ]
];

var_dump($turma[0]['nome'], $turma[0]['idade'], $turma[0]['rg'], $turma[0]['cpf']);

/**
 * Resultado:
    string(13) "Joao da sIlva" //nome
    int(18) // idade
    string(7) "5556666" // rg
    string(14) "010.200.256-89" //cpf
 */
```



_arrays multidimensionais

são arrays que têm um ou mais arrays dentro dele.
Arrays multidimensionais também são chamados matrizes.





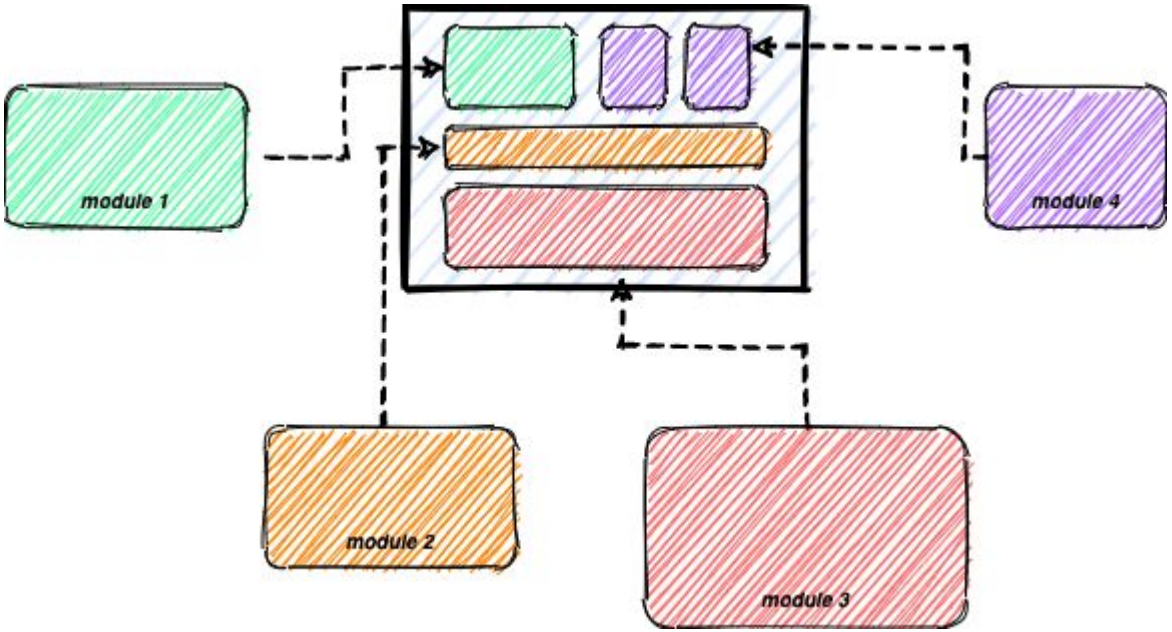
_arrays

Algumas funções

O PHP possui funções para arrays que facilitam bastante a manipulação dessas estruturas de uma forma simples. Vejamos alguns exemplos:

Função	Finalidade
`is_array()`	Verificar se uma variável é do tipo array.
`array_unshift()`	Adicionar um novo elemento no início do array.
`array_push()`	Adicionar um novo elemento no final do array.
`array_shift()`	Remover o primeiro elemento de um array.
`array_pop()`	Remover o último elemento de um array.
`rsort()`	Ordenar um array em ordem decrescente.
`asort()`	Ordenar um array em ordem crescente, ou seja, em ordem alfabética.
`sort()`	Ordenar um array. Essa função é mais comum, porque também organiza os índices do array.
`end()`	Recuperar o último elemento de um array.

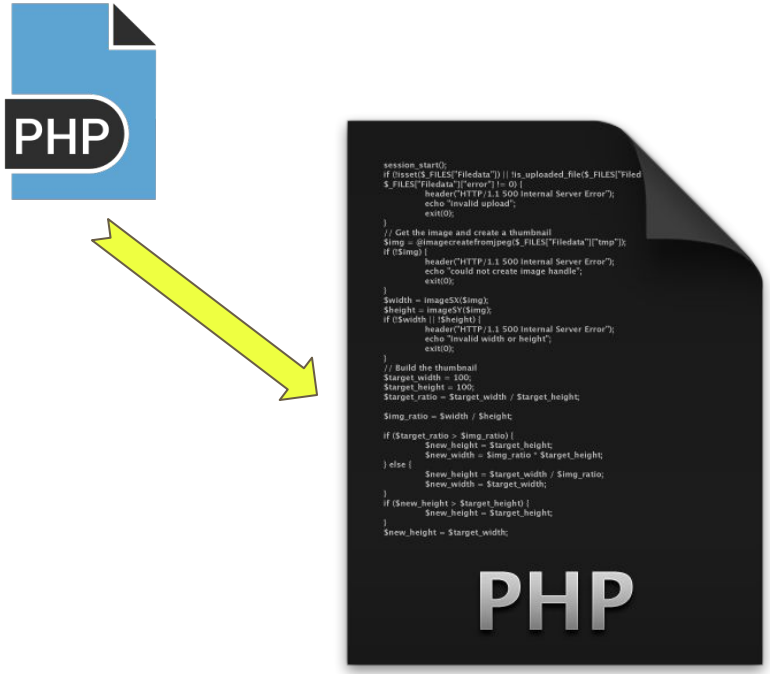
Incluindo conteúdo



Incluindo conteúdo

Há quatro funções básicas:

- include “meu-arquivo.php”;
- include_once “meu-arquivo.php”;
- require“meu-arquivo.php”;
- require_once “meu-arquivo.php”;



Incluindo conteúdo

Include

A declaração include inclui e avalia o arquivo informado. Se o arquivo não for encontrado no include_path, a declaração include checará no diretório do script que o executa e no diretório de trabalho corrente, antes de falhar.

```
include 'file.txt';  // Works.
include 'file.php';  // Works.
```

Incluindo conteúdo

include_once

Avalia o arquivo informado durante a execução do script. Este é um comportamento similar a declaração include, com a única diferença que, se o código do arquivo já foi incluído, não o fará novamente, e o include_once retornará TRUE.

```php
<?php
include_once "a.php"; // this will include a.php
include_once "A.php"; // this will include a.php again! (PHP 4 only)
?>
```

Incluindo conteúdo

require

É idêntica à include, exceto que em caso de falha também produzirá um erro fatal de nível E_COMPILE_ERROR. Em outras palavras, **ele parará** o script enquanto que o include apenas emitirá um alerta (E_WARNING) permitindo que o script continue.

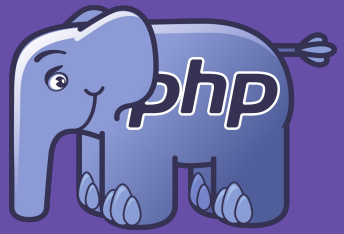
```php
<?php
require('somefile.php');
?>
```

Incluindo conteúdo

require_once

É idêntica à *require,* exceto que o PHP verificará se o arquivo já foi incluído, e em caso afirmativo, não o incluirá (exigirá) novamente.

```php
<?php
define('__ROOT__', dirname(dirname(__FILE__)));
require_once(__ROOT__.'/config.php');
?>
```



_atividades de fixação

Dica: utilize a função `fgets(STDIN) ou readline()` para capturar o valor digitado no terminal.

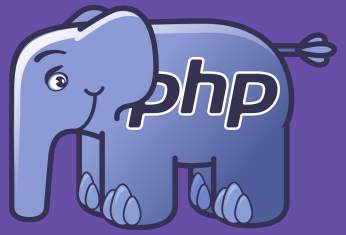
1. Utilizando o seu conhecimento em laços de repetição, crie dois arrays: **listaCrescente:** com uma sequência de inteiros de 0 a 100; **listaDecrescente:** com uma sequência de inteiros de 100 para 0; Ao final, faça a impressão de cada array no terminal;

2. Faça um programa PHP que devolve um array de números lidos do teclado.

3. Percorra o array a seguir e exiba a mensagem indicando o nome e a posição deste quando for localizado no array: [“João”, “José”, “Maria”, “Severino”, “Antônio”, “Marcos”, “Manoel”, “Filipe”, “André”, “Paulo”]; Use a linha de comando para digitar o nome;

4. Crie um array chamado “alunosDeMinhaTurma” e, com o recurso da função fgets(STDIN), adicione os nomes de todos os seus colegas de sala. Ao final, faça a impressão dos nomes. (Será necessário usar um laço de repetição).

5. Crie um programa PHP que recebe um array de inteiros predefinido e retorna as quantidades de elementos do array que são números negativos e números positivos.



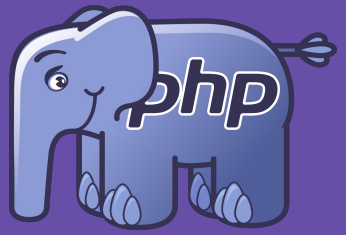
_funções

O que são funções em PHP?

- Função em PHP é um trecho de código que pode ser chamado (invocado) de qualquer parte do código para executar uma tarefa qualquer e retornar algum valor. Ou seja, você pode chamar quantas vezes forem necessárias a função.
- Você pode passar zero ou mais **parâmetros** para uma função, e poderá **retornar apenas um valor**. Porém este valor pode ser um array de valores. Mas continua sendo um valor apenas.
- A função quando chamada, faz com que o código PHP que a chama, seja parado. Ou seja, o código para, e espera o retorno da função para dar prosseguimento novamente ao seu invocador.

Fonte: https://www.oficinadanet.com.br

```php
function nomeDaMinhaFuncao(string $texto){
    return "retornar o $texto passado";
}
```



_funções

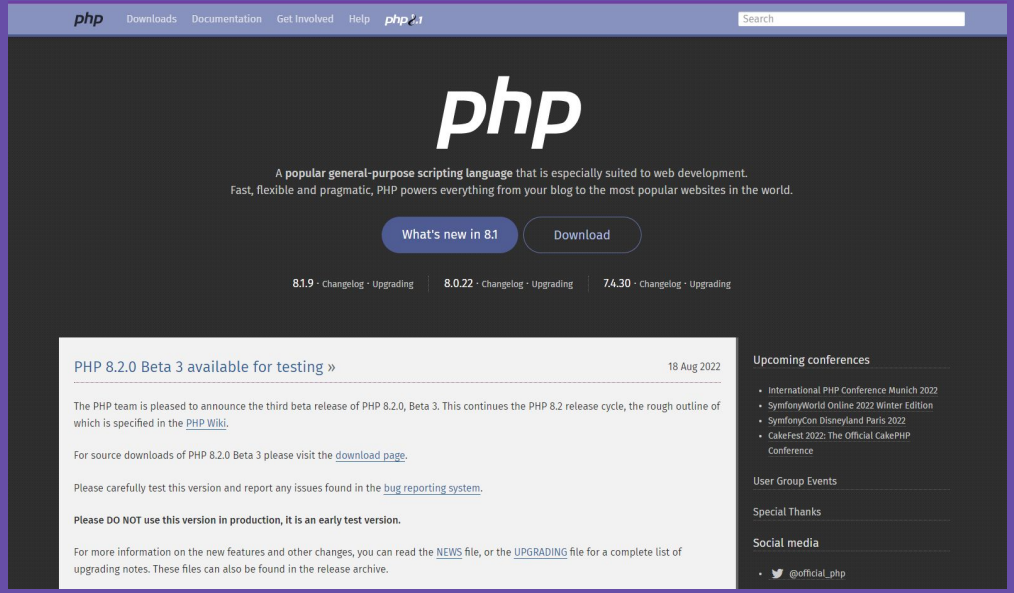
Parâmetros de funções PHP:

- A função pode ter mais de um parâmetro, assim separamos eles com uma vírgula.
- Assim podemos passar mais de uma instrução para a função. Estes parâmetros podem ainda ser arrays.
- Os parâmetros de uma função em PHP podem não ser obrigatórios. Para não obrigar um valor, precisamos deixar algo estado nesta variável.

Fonte: https://www.oficinadanet.com.br

```php
function nomeDaMinhaFuncao(string $texto, int $numero, ?bool $ativo){
    return "retornar o $texto passado com o $numero";
}

function nomeDaMinhaOutraFuncao($texto, $numero, $ativo = null){
    return "retornar o $texto passado com o $numero";
}
```

Fonte? Vá na fonte!



https://www.php.net

https://github.com/adrielacademico

Professor:
Adriel Sales