




---

---

---

---

---

---

---

### Arrays

- O que são
  - Arrays são coleções de objetos categorizados por índice numérico ou outro
- Como funciona
  - Funcionamento similar a estrutura interna de strings, com posições estruturadas

S	A	Softblue
[0]	[1]	[2]
- Definição própria de índices
  - É possível criar seu próprio índice de chaves, no lugar dos numéricos

S	A	Softblue
[alfa]	[nome]	[empresa]

---

---

---

---

---

---

---

### Arrays

- Criação
 

```
$arrayExemplo = array(5, "A", "Softblue");
```

S	A	Softblue
[0]	[1]	[2]
- Acesso
 

```
echo $arrayExemplo[2]; // Resultado: Softblue
$arrayExemplo[2] = "Web";
echo $arrayExemplo[2]; // Resultado: Web
```

---

---

---

---

---

---

---

## Arrays

- Criação e acesso com índices próprios

```
$arrayExemplo = array(alfa => 5, nome => "A", empresa => "Softblue");
```

5	A	Softblue
[alfa]	[nome]	[empresa]

```
echo $arrayExemplo[empresa]; // Resultado: Softblue
```

- Multidimensionais
  - Funcionalidade que permite armazenar arrays dentro de arrays

```
$arrayExemplo = array(5, array("andre", "milani"));
```

5	andre	milani
[0]	[0]	[1]

```
echo $arrayExemplo[0]; // Resultado: 5
echo $arrayExemplo[1][0]; // Resultado: andre
```

---

---

---

---

---

---

---

---

## Funções Especiais

- print\_r (impressão de valores)
- unset (excluindo posição)
- count e sizeof (tamanho de array)
- foreach (navegação em array)
- array\_push e array\_pop (trabalhando com pilhas)
- array\_shift e array\_unshift (trabalhando com filas)
- array\_map (executando função em todo array)
- array\_key\_exists e array\_keys (verificando ocorrência)
- array\_search e in\_array (localizando ocorrência)
- shuffle, sort e rsort (ordenação de array)
- parse\_str (transformação entre string e array)
- explode e implode (transformação entre string e array)

---

---

---

---

---

---

---

---

## Criando um array

- Inicializando

Sintaxe

```
(array) array();
(array) array(valores);
```

Exemplo

```
$meuArray = array();
echo $meuArray[1]; // Resultado: <vazio>

$meuArray = array("Maçã", "Melão", "Uva");
echo $meuArray[1]; // Resultado: Melão

$meuArray[1] = "Laranja";
echo $meuArray[1]; // Resultado: Laranja
```

---

---

---

---

---

---

---

---

## Impressão de valores



### • print\_r

Sintaxe

```
(void) print_r(array);
```

```
$arrayExemplo = array ("Php", "SQL", 100, "Assembler");
```

Php	SQL	100	Assembler
[0]	[1]	[2]	[3]

Exemplo

```
print_r($arrayExemplo);  
// Resultado: Array ( [0] => Php [1] => SQL [2] => 100 [3] => Assembler )
```

---

---

---

---

---

---

---

---

## Excluindo posição



### • unset

Sintaxe

```
(void) unset(array[posição]);
```

```
$arrayExemplo = array ("Php", "SQL", 100, "Assembler");
```

Php	SQL	100	Assembler
[0]	[1]	[2]	[3]

Exemplo

```
unset($arrayExemplo[1]);  
// Resultado:
```

Php	100	Assembler
[0]	[1]	[2]

---

---

---

---

---

---

---

---

## Tamanho de array



### • count / sizeof

Sintaxe

```
(int) count(array);  
(int) sizeof(array);
```

```
$arrayExemplo = array ("Php", "SQL", 100, "Assembler");
```

Php	SQL	100	Assembler
[0]	[1]	[2]	[3]

Exemplo

```
echo(count($arrayExemplo));  
// Resultado: 4  
  
echo(sizeof($arrayExemplo));  
// Resultado: 4
```

---

---

---

---

---

---

---

---

## Navegação em array



### • foreach

#### Sintaxe

```
(void) foreach(array as parte) { bloco de código };
```

```
$arrayExemplo = array ("Php", "SQL", 100, "Assembler");
```

Php	SQL	100	Assembler
[0]	[1]	[2]	[3]

#### Exemplo

```
foreach($arrayExemplo as $pedaco) {
    echo "Tem no array: ".$pedaco. " ";
}
// Resultado: Tem no array: Php, Tem no array: SQL, Tem no array: 100, Tem no array: Assembler,
```

---

---

---

---

---

---

---

---

## Inserindo elementos em fila e pilha



### • array\_push

#### Sintaxe

```
(int) array_push(array, elemento);
```

```
$arrayExemplo = array ("Php", "SQL", 100, "Assembler");
```

Php	SQL	100	Assembler
[0]	[1]	[2]	[3]

#### Exemplo

```
array_push($arrayExemplo, "André");
// Resultado:
```

Php	SQL	100	Assembler	André
[0]	[1]	[2]	[3]	[4]

---

---

---

---

---

---

---

---

## Removendo elemento de uma pilha



### • array\_pop

#### Sintaxe

```
(mixed) array_pop(array);
```

```
$arrayExemplo = array ("Php", "SQL", 100, "Assembler");
```

Php	SQL	100	Assembler
[0]	[1]	[2]	[3]

#### Exemplo

```
$x = array_pop($arrayExemplo);
echo ($x);
// Resultado: Assembler
```

Php	SQL	100
[0]	[1]	[2]

---

---

---

---

---

---

---

---

## Removendo elemento de uma fila

### • array\_shift

Sintaxe

(mixed) **array\_shift**(array);

\$arrayExemplo = array ("Php", "SQL", 100, "Assembler");

Php	SQL	100	Assembler
[0]	[1]	[2]	[3]

Exemplo

```
$x = array_shift($arrayExemplo);
echo ($x);
// Resultado: Php
```

SQL	100	Assembler
[0]	[1]	[2]

---

---

---

---

---

---

---

---

## Inserindo elemento na frente da fila

### • array\_unshift

Sintaxe

(mixed) **array\_unshift**(array, elemento);

\$arrayExemplo = array ("Php", "SQL", 100, "Assembler");

Php	SQL	100	Assembler
[0]	[1]	[2]	[3]

Exemplo

```
array_unshift($arrayExemplo, "Urgente");
// Resultado:
```

Urgente	Php	SQL	100	Assembler
[0]	[1]	[2]	[3]	[4]

---

---

---

---

---

---

---

---

## Executando função em todo array

### • array\_map

Sintaxe

(array) **array\_map**(função, array);

\$arrayExemplo = array ("Php", "SQL", 100, "Assembler");

Php	SQL	100	Assembler
[0]	[1]	[2]	[3]

Exemplo

```
function insereMoeda($valor) { return "R$ ".$valor; }
$arrayExemplo = array_map("insereMoeda", $arrayExemplo);
// Resultado:
```

R\$ Php	R\$ SQL	R\$ 100	R\$ Assembler
[1]	[2]	[3]	[4]

---

---

---

---

---

---

---

---

## Verificando ocorrência



- **array\_key\_exists**

Sintaxe

```
(bool) array_key_exists(chave, array);
```

\$arrayExemplo = array ( "Linguagem1" => "Php", "Linguagem2" => "SQL", "Linguagem3" => 100, "Linguagem4" => "Assembler");

Php	SQL	100	Assembler
["Linguagem1"]	["Linguagem2"]	["Linguagem3"]	["Linguagem4"]

Exemplo

```
echo array_key_exists("Linguagem2", $arrayExemplo);
// Resultado: 1 (TRUE)

echo array_key_exists("Linguagem7", $arrayExemplo);
// Resultado: (FALSE)
```

---

---

---

---

---

---

---

---

## Verificando ocorrência



- **array\_keys**

Sintaxe

```
(array) array_keys(array);
```

\$arrayExemplo = array ( "Linguagem1" => "Php", "Linguagem2" => "SQL", "Linguagem3" => 100, "Linguagem4" => "Assembler");

Php	SQL	100	Assembler
["Linguagem1"]	["Linguagem2"]	["Linguagem3"]	["Linguagem4"]

Exemplo

```
$keys = array_keys($arrayExemplo);
foreach($keys as $key) { echo $key." "; }
// Resultado: Linguagem1 Linguagem2 Linguagem3 Linguagem4
```

---

---

---

---

---

---

---

---

## Verificando ocorrência



- **array\_search**

Sintaxe

```
(mixed) array_search(elemento, array);
```

\$arrayExemplo = array ( "Linguagem1" => "Php", "Linguagem2" => "SQL", "Linguagem3" => 100, "Linguagem4" => "Assembler");

Php	SQL	100	Assembler
["Linguagem1"]	["Linguagem2"]	["Linguagem3"]	["Linguagem4"]

Exemplo

```
$key = array_search("Php", $arrayExemplo);
echo $key;
// Resultado: Linguagem1
```

---

---

---

---

---

---

---

---

## Verificando ocorrência



## • in\_array

Sintaxe

(mixed) **in\_array**(elemento, array);

```
$arrayExemplo = array (
    "Linguagem1" => "Php", "Linguagem2" => "SQL",
    "Linguagem3" => 100, "Linguagem4" => "Assembler");
```

Php	SQL	100	Assembler
["Linguagem1"]	["Linguagem2"]	["Linguagem3"]	["Linguagem4"]

Exemplo

```
$key = in_array("Php", $arrayExemplo);
echo $key;
// Resultado: 1 (TRUE)
```

---

---

---

---

---

---

---

---

## Ordenação de array



## • shuffle

Sintaxe

(bool) **shuffle**(array);

```
$arrayExemplo = array (
    "Linguagem1" => "Php", "Linguagem2" => "SQL",
    "Linguagem3" => 100, "Linguagem4" => "Assembler");
```

Php	SQL	100	Assembler
["Linguagem1"]	["Linguagem2"]	["Linguagem3"]	["Linguagem4"]

Exemplo

```
shuffle($arrayExemplo);
// Resultado:
```

100	Assembler	Php	SQL
[0]	[1]	[2]	[3]

---

---

---

---

---

---

---

---

## Ordenação de array



## • sort

Sintaxe

(bool) **sort**(array);

```
$arrayExemplo = array (
    "Linguagem1" => "Php", "Linguagem2" => "SQL",
    "Linguagem3" => 100, "Linguagem4" => "Assembler");
```

Php	SQL	100	Assembler
["Linguagem1"]	["Linguagem2"]	["Linguagem3"]	["Linguagem4"]

Exemplo

```
sort($arrayExemplo);
// Resultado:
```

Assembler	Php	SQL	100
[0]	[1]	[2]	[3]

---

---

---

---

---

---

---

---

## Ordenação de array



### • rsort

Sintaxe

```
(bool) rsort(array);
```

```
$arrayExemplo = array (
    "Linguagem1" => "Php", "Linguagem2" => "SQL",
    "Linguagem3" => 100, "Linguagem4" => "Assembler");
```

Php	SQL	100	Assembler
["Linguagem1"]	["Linguagem2"]	["Linguagem3"]	["Linguagem4"]

Exemplo

```
rsort($arrayExemplo);
// Resultado:
```

100	SQL	Php	Assembler
[0]	[1]	[2]	[3]

## Transformação entre string e array



### • parse\_str

Sintaxe

```
(void) parse_str(array);
```

```
$stringExemplo = "var1=natação&var2=basquete&teste=futebol";
```

Exemplo

```
parse_str($stringExemplo, $arrayResultado);
// Resultado:
```

natação	basquete	futebol
["var1"]	["var2"]	["teste"]

## Transformação entre string e array



### • explode

Sintaxe

```
(array) explode(divisor, string[, númeroMáximoDeDivisões]);
```

```
$stringExemplo = "Aluno do curso de PHP";
```

Exemplo

```
$arrayResultado = explode(" ", $stringExemplo);
// Resultado:
```

Aluno	do	curso	de	PHP
[0]	[1]	[2]	[3]	[4]



## Ordenação de array

- implode

Sintaxe

```
(string) implode([separador, ]array);
```

```
$arrayExemplo = array("Maçã", "Laranja", "Mamão", "Uva");
```

Maçã	Laranja	Mamão	Uva
[0]	[1]	[2]	[3]

Exemplo

```
$stringResultado = implode(" - ", $arrayExemplo);  
echo $stringResultado;  
// Resultado: Maçã - Laranja - Mamão - Uva
```

---

---

---


---

---

---

---

## Aulas práticas e manuais on-line



Assista agora as aulas práticas, que apresentam o uso dos comandos abordados nesta aula teórica.

[Clique aqui](#) para visualizar as aulas práticas disponíveis

---

---

---

---

---

---

---