📚 Glossário Completo de PHP para Iniciantes

Este glossário reúne os principais conceitos, termos e práticas da linguagem PHP, explicados de forma clara e objetiva, com exemplos práticos para facilitar o seu aprendizado.

📌 Variáveis

Armazenam dados temporários na memória. Sempre começam com `$`.

```
$nome = "Maria";
$idade = 30;
$ativo = true;
```

📌 Constantes

São valores que não podem ser alterados depois de definidos.

```
define("PI", 3.1415);
const TAXA = 0.05;
```

📌 Tipos de Dados

Principais tipos:

- `int` — números inteiros
- `float` — números decimais
- `bool` — verdadeiro ou falso
- `string` — texto
- `array` — coleção de valores
- `object` — objetos

```
$numero = 10;
$preco = 29.99;
$flag = false;
$nome = "Ana";
```

📌 Operadores

Realizam operações matemáticas, lógicas ou relacionais.

```
$soma = 5 + 10;
$logico = ($idade > 18 && $ativo);
```

📌 Condicionais

Controlam o fluxo do programa.

```
if ($idade >= 18) {
    echo "Maior de idade";
} else {
    echo "Menor de idade";
}
```

📌 Loops

Permitem a repetição de blocos de código.

```
for ($i = 0; $i < 5; $i++) {
    echo $i;
}
```

📌 Arrays

Coleções indexadas ou associativas.

```
$frutas = ["Maçã", "Banana", "Uva"];
$dados = ["nome" => "Carlos", "idade" => 40];
```

📌 Funções

Blocos de código reutilizáveis.

```
function saudacao($nome) {
    return "Olá, $nome!";
}
```

📌 Classes

Definem estruturas para organizar código orientado a objetos.

```
class Pessoa {
    public $nome;

    public function falar() {
        echo "Meu nome é " . $this->nome;
```

```
    }
}
```

📌 Objetos

Instâncias de uma classe.

```
$p = new Pessoa();
$p->nome = "Lucas";
$p->falar();
```

📌 Herança

Permite que uma classe herde de outra.

```
class Estudante extends Pessoa {
    public $curso;
}
```

📌 Encapsulamento

Protege dados internos com modificadores de acesso.

```
class Conta {
    private $saldo = 0;

    public function getSaldo() {
        return $this->saldo;
    }

    public function depositar($valor) {
        $this->saldo += $valor;
    }
}
```

📌 Interfaces

Definem métodos obrigatórios para uma classe implementar.

```
interface Animal {
    public function emitirSom();
}
```

📌 Try-Catch (Tratamento de Exceções)

Gerencia erros no código.

```
try {
    throw new Exception("Erro!");
} catch (Exception $e) {
    echo $e->getMessage();
}
```

📌 Arrow Functions (Funções Anônimas)

Declaração curta de funções.

```
$soma = fn($a, $b) => $a + $b;
```

📌 Arrays Associativos

Mapeiam chaves a valores.

```
$pessoa = [
    "nome" => "João",
    "idade" => 25
];
```

📌 Foreach

Itera sobre arrays.

```
foreach ($frutas as $fruta) {
    echo $fruta;
}
```

📌 Superglobais

Variáveis globais disponíveis em todo o script.

- `$_GET`
- `$_POST`
- `$_SESSION`
- `$_FILES`
- `$_SERVER`

```
echo $_SERVER['HTTP_HOST'];
```

📌 Namespaces

Organizam o código e evitam conflitos.

```
namespace MeuProjeto;
```

📌 Composer

Gerenciador de dependências do PHP.

```
composer require nome/pacote
```