

# SW-I

# SISTEMAS WEB I

---

Prof. Anderson Vanin

AULA 12 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM PHP  
HERANÇA

# Introdução

Entender o conceito de herança em PHP, demonstrando como uma classe pode herdar atributos e métodos de outra, promovendo reuso de código e organização. Introduzir os modificadores de acesso ***protected*** e a palavra-chave ***extends***.

# O que é Herança?

- Herança permite que uma classe (classe filha) **herde atributos e métodos de outra classe** (classe pai), promovendo reuso de código.
- Exemplo do mundo real: *"Um carro é um tipo de veículo, então um carro pode herdar características gerais de um veículo."*

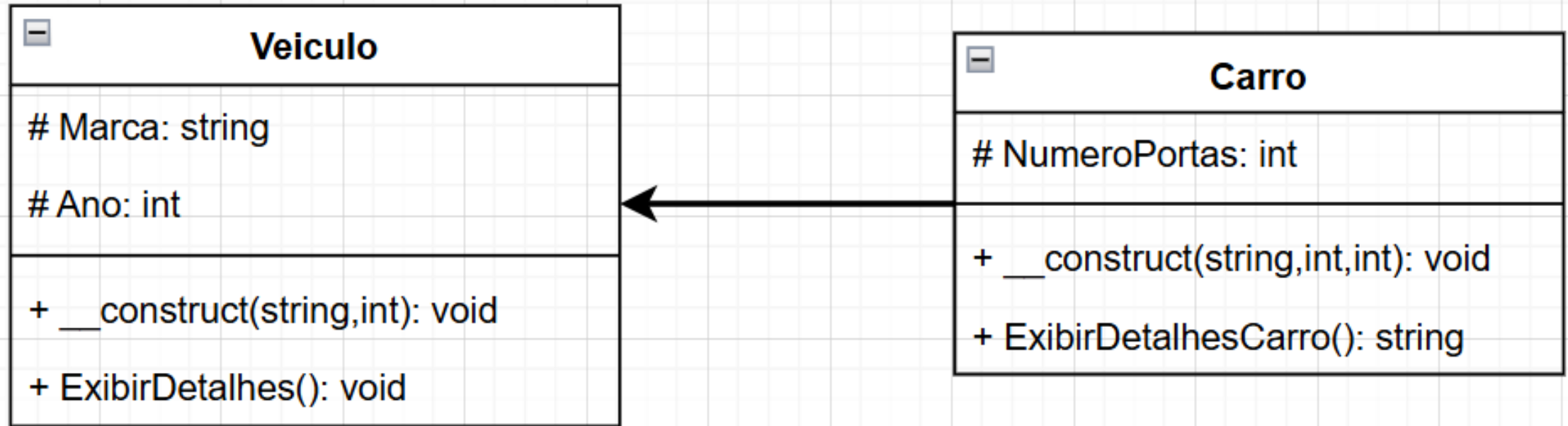
# Sintaxe básica

- Uso da palavra-chave ***extends*** para criar herança.
- Introdução ao modificador ***protected***: permite que atributos e métodos sejam acessados pela classe pai e suas classes filhas, mas não externamente.

# Exemplo Prático

- Vamos criar um exemplo simples de código, com uma classe pai (Veiculo) e uma classe filha (Carro).
- Mostrar como a classe filha herda atributos e métodos, e como pode adicionar ou modificar comportamentos.

# Exemplo Prático



# Exemplo Prático

## ▼ AULA12

 Carro.class.php

 principal.php


 **Veiculo.class.php**

 Veiculo.class.php


```
1  <?php
2      // Classe pai
3      class Veiculo {
4          protected $Marca;
5          protected $Ano;
6
7          public function __construct($marca, $ano) {
8              $this->Marca = $marca;
9              $this->Ano = $ano;
10         }
11
12         public function ExibirDetalhes() {
13             return "<br> Marca: $this->marca <br> Ano: $this->ano";
14         }
15     }
16  ?>
```

# Exemplo Prático

## ▼ AULA12

 Carro.class.php

 principal.php

 Veiculo.class.php

```
Carro.class.php
1  <?php
2      // Classe filha de Veiculo
3      require_once 'Veiculo.class.php';
4
5      class Carro extends Veiculo {
6          private $NumeroPortas;
7
8          public function __construct($marca, $ano, $numeroPortas) {
9              parent::__construct($marca, $ano);
10             $this->NumeroPortas = $numeroPortas;
11         }
12
13         public function ExibirDetalhesCarro() {
14             return $this->ExibirDetalhes() . "<br> Número de portas: $this->numeroPortas";
15         }
16     }
17  ?>
```

parent::\_\_construct() para inicializar os atributos da classe pai.




# Exemplo Prático

## ▼ AULA12

 Carro.class.php

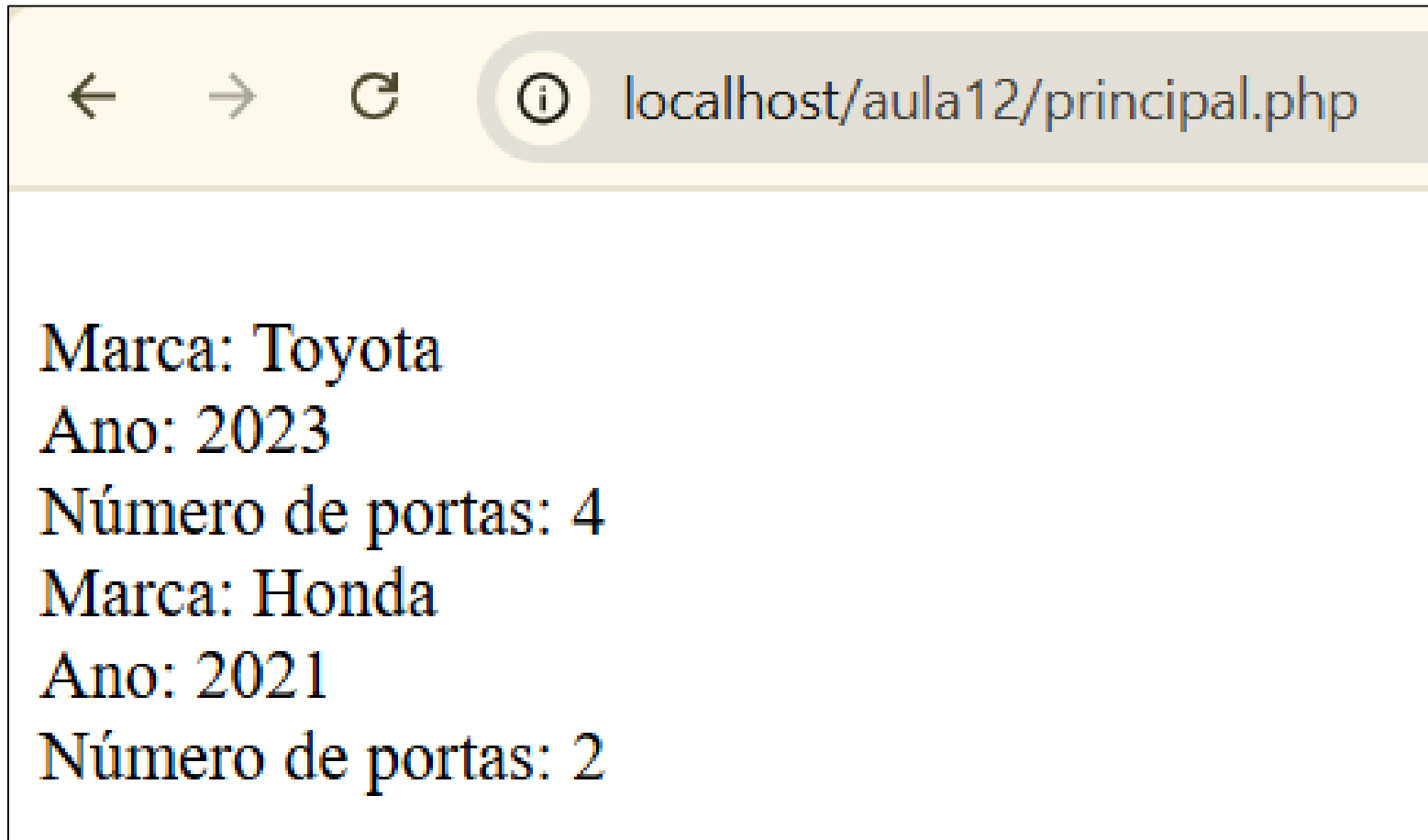
 principal.php

 Veiculo.class.php

 principal.php

```
1  <?php
2      require_once 'Carro.class.php';
3
4      $meuCarro = new Carro("Toyota", 2023, 4);
5      echo $meuCarro->ExibirDetalhesCarro();
6
7      // Exemplo adicional: criando outro carro
8      $outroCarro = new Carro("Honda", 2021, 2);
9      echo $outroCarro->ExibirDetalhesCarro();
10  ?>
```

# Exemplo Prático



Tente acessar diretamente `$meuCarro->Marca` em `principal.php` para demonstrar que o atributo *protected* não é acessível fora da classe ou suas filhas (resultará em erro).

# Notas Importantes

- **Modularização:**

Separar as classes em arquivos diferentes melhora a organização e reutilização do código.

O uso de ***require\_once*** evita inclusões duplicadas.

- **Herança:**

A classe Carro **herda tudo de Veiculo**, mas pode adicionar ou modificar comportamentos.

O encapsulamento ***protected*** permite que Carro acesse marca e ano, mas **impede acesso externo**.

*Modifiquem o código, adicionando um novo atributo à classe Carro (ex.: cor) e atualizem o método ExibirDetalhesCarro().*

*Crie uma classe Moto em um arquivo separado (Moto.class.php) que herda de Veiculo, com um atributo privado ***cilindradas*** e um método específico.*

# Exercício 01

## Classe Animal e Cachorro

- Crie uma classe pai Animal com atributos *protected* (nome e idade) e um método EmitirSom().
- Crie uma classe filha Cachorro que herda de Animal, adicionando um atributo privado *raca* e um método que combine o som do animal com a raça.
- Exemplo de saída: ***"O cachorro Rex, da raça Pastor Alemão, faz: Au au!"***.

# Exercício 02

## Classe Funcionario e Gerente

- Crie uma classe Funcionario com atributos *protected* (nome e salario) e um método para exibir informações.
- Crie uma classe Gerente que herda de Funcionario, adicionando um atributo privado departamento e um método para exibir informações específicas do gerente.
- Exemplo de saída: ***"Gerente João, salário: 5000, departamento: Vendas"***.

# Exercício 03

## Classe Produto e Livro

- Crie uma classe Produto com atributos *protected* (nome e preco) e um método para calcular desconto.
- Crie uma classe Livro que herda de Produto, adicionando um atributo privado autor e um método para exibir detalhes do livro.
- Exemplo de saída: ***"Livro: Dom Casmurro, Autor: Machado de Assis, Preço com desconto: 36.00".***

# Exercício 03

## Desafio com Múltiplas Classes Filhas

- Crie uma classe pai `FiguraGeometrica` com atributos *protected* (como largura e altura) e um método para calcular área.
- Crie duas classes filhas: `Retangulo` e `Triangulo`, cada uma implementando o cálculo de área de forma diferente.
- Instancie objetos de ambas as classes e mostre os resultados.