Modificadores de Acesso

Public

Public - Faz com que os métodos e propriedades de uma Classe sejam visíveis em todo o código sendo o mais acessível.

Protected

Protected - Apenas as Classes Herdeiras conseguem acessar os métodos e propriedades que tem esse modificador de acesso.

Private

Private - Métodos e propriedades só são visíveis pela Classe que os declarou.

```php
<?php

class Veiculo {
    protected $modelo;
}

class Carro extends Veiculo{
    public function ligarLimpador(){
        echo "Limpando em 3,2,1";
    }
}
```

```php
$veiculo = new Veiculo();
$veiculo->modelo = "Gol";
echo $veiculo->modelo;
```

Saída

Fatal error: Uncaught Error: Cannot access protected property Veiculo::$modelo in C:\xampp\htdocs\PHP-OO\PHP-OO\Aula5-Modificadores_de_acesso\index.php:34 Stack trace: #0 {main} thrown in **C:\xampp\htdocs\PHP-OO\PHP-OO\Aula5-Modificadores_de_acesso\index.php** on line **34**

> Essa saída ocorre pois os atributos e métodos são protegidos pelo Modificador Protected só podendo ser acessado através de Getters e Setters, como no exemplo abaixo:

```php
class Veiculo {
    protected $modelo;

    public function setModelo($modelo){
        $this->modelo = $modelo;
    }

    public function getModelo(){
        return $this->modelo;
    }
}

class Carro extends Veiculo{
}

$veiculo = new Veiculo();
```

```php
$veiculo->setModelo("Gol");
echo $veiculo->getModelo();
```

Saída

Gol

```php
class Veiculo {
    protected $modelo;
}

class Carro extends Veiculo{
    public function setModelo($modelo){
        $this->modelo = $modelo;
    }

    public function getModelo(){
        return $this->modelo;
    }
}

$carro = new Carro();
$carro->setModelo("Gol");
echo $carro->getModelo();
```

Saída

Gol

> Isso ocorre porque Carro é um Herdeiro ou Filho de Veículo e assim por meio dos Getters e Setters como dito na definição do Modificador Protected podemos utilizar os atríbutos e métodos da SuperClasse ou Classe Pai que é Veículo

```php
class Veiculo {
    private $modelo;
}

class Carro extends Veiculo{
    public function setModelo($modelo){
        $this->modelo = $modelo;
    }

    public function getModelo(){
        return $this->modelo;
    }
}

$carro = new Carro();
$carro->setModelo("Gol");
echo $carro->getModelo();

echo '<pre>';
var_dump($carro);
echo '</pre>';
```

Saída

```
Gol
object(Carro)#1 (4) {
["modelo":"Veiculo":private]=>
NULL
["cor"]=>
```

```
NULL
["ano"]=>
NULL
["modelo"]=>
string(3) "Gol"
}
```

> A saída é a mesma porém o var_dump nos mostra que o que foi criado é na verdade apenas um novo atributo modelo dentro de carro e o modelo de Veículo ainda continua NULL

Método Private, como usar?

```php
<?php

class Veiculo {

    public $modelo;
    public $cor;
    public $ano;

    private function Andar(){
        echo "Andou";
    }

    public function Parar(){
        echo "Parou";
    }
}

class Carro extends Veiculo{
}
```

```php
$veiculo = new Veiculo();
$veiculo->Andar();
```

Saída

Fatal error: Uncaught Error: Call to private method Veiculo::Andar() from global scope in C:\xampp\htdocs\PHP-OO\PHP-OO\Aula6-Modificadores-de-acesso-parte2\index.php:20 Stack trace: #0 {main} thrown in **C:\xampp\htdocs\PHP-OO\PHP-OO\Aula6-Modificadores-de-acesso-parte2\index.php** on line **20**

> Como utilizar então esse método?
>
> Podemos criar então um método novo chamado mostrarAcao() e chamar o método Andar() dentro dele com o $this já que estamos na mesma Classe

```php
<?php

class Veiculo {

    public $modelo;
    public $cor;
    public $ano;

    private function Andar(){
        echo "Andou";
    }

    public function Parar(){
        echo "Parou";
    }
```

```
    public function mostrarAcao(){
        $this->Andar();
    }
}

class Carro extends Veiculo{
}

$veiculo = new Veiculo();
$veiculo->mostrarAcao();
```

Saída

Andou

> Agora se fizermos isso na classe Filha ou Herdeira não obteremos o mesmo resultado pois estamos utilizando o private que não dá acesso de atributos e métodos aos Filhos ou Herdeiros

```
<?php

class Veiculo {
    public $modelo;
    public $cor;
    public $ano;

    private function Andar(){
        echo "Andou";
    }

    public function Parar(){
        echo "Parou";
```

```
    }

}

class Carro extends Veiculo{
    public function mostrarAcao(){
        $this->Andar();
    }
}



$carro = new Carro();

$carro->mostrarAcao();
```

Saída

Fatal error: Uncaught Error: Call to private method Veiculo::Andar() from scope Carro in C:\xampp\htdocs\PHP-OO\PHP-OO\Aula6-Modificadores-de-acesso-parte2\index.php:15 Stack trace: #0 C:\xampp\htdocs\PHP-OO\PHP-OO\Aula6-Modificadores-de-acesso-parte2\index.php(20): Carro->mostrarAcao() #1 {main} thrown in **C:\xampp\htdocs\PHP-OO\PHP-OO\Aula6-Modificadores-de-acesso-parte2\index.php** on line **15**

E se o método fosse protected, o filho conseguiria acessar?

```
<?php

class Veiculo {
    public $modelo;
    public $cor;
    public $ano;
```

```php
    protected function Andar(){
        echo "Andou";
    }

    public function Parar(){
        echo "Parou";
    }

}

class Carro extends Veiculo{
    public function mostrarAcao(){
        $this->Andar();
    }
}



$carro = new Carro();
$carro->mostrarAcao();
```

Saída

Andou

> Como podemos ver a resposta é sim!