<?php

    //MÉTODOS ACESSORES:

        class Forma {

            private string $sabor; //Perceba que nossos atributos estão privados...

            private string $formato;

            public function setSabor($sabor):string { //Temos aqui setters para atribuição de valores, por convenção

                return $this->sabor = $sabor;           //escrevemos "set" no método...

            }

            public function setFormato($formato):string {

                return $this->formato = $formato;

            }

            public function getSabor():string { //Temos aqui getters para visualizar os atributos, por convenção escrevemos

                return $this->sabor;            //get no método...

            }

            public function getFormato():string {

                return $this->formato;

            }

        }

        //Note que nas 2 instâncias abaixo nós precimos de atribuir valores aos atributos por através de métodos acessores...

        $chocolate = new Forma();

        $chocolate->setSabor('chocolate');

        $chocolate->setFormato('quadrado');

        echo $chocolate->getSabor() . PHP\_EOL; //Só conseguimos ver os valores se usarmos um get...

        echo PHP\_EOL;

        echo PHP\_EOL;

        $morango = new Forma();

        $morango->setSabor('morango');

        $morango->setFormato('redondo');

        var\_dump($morango);

        // $morango->sabor; //Veja que quando tentamos visualizar o valor de um atributo privado, teremos um erro...

[Running] php "c:\Users\Almoxarifado\Documents\php\arquivos\_das\_aulas\30-metodos\_acessores.php"

Chocolate

object(Forma)#2 (2) {

  ["sabor":"Forma":private]=>

  string(7) "morango"

  ["formato":"Forma":private]=>

  string(7) "redondo"

}

        class Animal {

            private $nome;

            private $idade;

            public function \_\_construct(string $nome, string $idade){

                $this->nome = $nome;

                $this->idade = $idade;

            }

            protected function getNome(){ //Perceba que deixamos os nossos métodos protegidos...

                return $this->nome;

            }

            protected function getIdade(){ //Perceba que deixamos os nossos métodos protegidos...

                return $this->idade;

            }

        }

        class Cachorro extends Animal {

            private $falar;

            public function \_\_construct(string $nome, string $idade, string $falar){ //Na classe filha nós empregamos todos

                                                                                        //os atributos de ambas as classes

                parent:: \_\_construct($nome, $idade);//Mas note que para referenciar     //mãe e filhas...

                $this->falar = $falar;              //os atributos da classe mãe nós

                                                    //usamos a palavra reservada "parent"

            }                                       //para referenciar os atributos da

                                                    //classe mãe...

            public function getFalar():string

            {

                return $this->falar;

            }

            public function getDados():string

            {

                return $this->getNome() . PHP\_EOL . $this->getIdade() . ' ano(s)'; //Note que métodos protegidos só podem

            }                                                                      //ser usados pelas classes filhas se

                                                                                    //chamarmos o método dentro de outro

                                                                                    //método e usando o "$this" para chamar

                                                                                    //o método, existe outra forma de chamá-lo

                                                                                    //usando o "parent", veja na classe "Gato"

        }

        class Gato extends Animal {

            private $falar;

            public function \_\_construct(string $nome, string $idade, string $falar){

                parent:: \_\_construct($nome, $idade);

                $this->falar = $falar;

            }

            public function getFalar():string

            {

                return $this->falar;

            }

            public function getDados():string

            {

                return parent::getNome() . PHP\_EOL . parent::getIdade() . ' ano(s)'; //Essa é outra forma de referenciar

            }                                                                           //métodos privados usando o "parent"

        }

        $pluto = new Cachorro('Pluto', '7', 'Au! Au!');

        echo $pluto->getDados();echo PHP\_EOL;

        echo $pluto->getFalar();echo PHP\_EOL;

        echo PHP\_EOL;

        $gato\_de\_botas = new Gato('Gato de Botas', '5', 'Meow!');

        echo $gato\_de\_botas->getDados();echo PHP\_EOL;

        echo $gato\_de\_botas->getFalar();echo PHP\_EOL;

        echo PHP\_EOL;

        var\_dump($pluto);echo PHP\_EOL;

        var\_dump($gato\_de\_botas);

?>

[Running] php "c:\Users\Almoxarifado\Documents\php\arquivos\_das\_aulas\30-metodos\_acessores.php"

Pluto

7 ano(s)

Au! Au!

Gato de Botas

5 ano(s)

Meow!

object(Cachorro)#3 (3) {

  ["nome":"Animal":private]=>

  string(5) "Pluto"

  ["idade":"Animal":private]=>

  string(1) "7"

  ["falar":"Cachorro":private]=>

  string(7) "Au! Au!"

}

object(Gato)#4 (3) {

  ["nome":"Animal":private]=>

  string(13) "Gato de Botas"

  ["idade":"Animal":private]=>

  string(1) "5"

  ["falar":"Gato":private]=>

  string(5) "Meow!"

}

[Done] exited with code=0 in 0.088 seconds