PHP

Herança e Elementos Estáticos

Softblue
cursos online

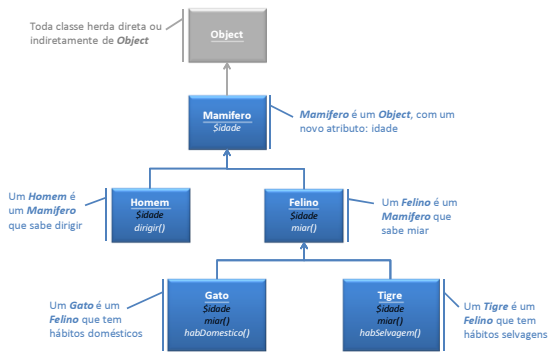
Herança

Softblue

- Permite que uma classe filha (subclasse) herde atributos e métodos não privados (***private***) de sua classe pai, também conhecida como superclasse
- Benefícios:
  - Reaproveitamento de código
  - Organização do código
  - Facilidade de manutenção
- Em algumas linguagens é possível utilizar herança múltipla
- Na omissão, herda-se da classe ***Object***

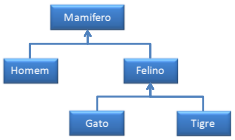
Hierarquia de classes

Softblue



Herdando uma classe

- Operador ***extends***




```php
class Mamifero {

    public $idade;
    public $peso;

    public function andar() {
        echo "andar";
    }
}
```

```php
class Homem {

    public $idade;
    public $peso;
    public $cpf;

    public function andar() {
        echo "andar";
    }

    public function dirigir($carro) {
        echo "dirigir";
    }
}
```

```php
class Homem extends Mamifero {

    public $idade;
    public $peso;
    public $cpf;

    public function andar() {
        echo "andar";
    }

    public function dirigir($carro) {
        echo "dirigir";
    }
}
```

Construtores hierárquicos

- Operador ***parent***

```php
class Mamifero {

    public $idade;
    public $peso;

    public function __construct($idade, $peso) {
        // Validações de idade e peso
        $this->idade = $idade;
        $this->peso = $peso;
    }

    public function andar() {
        echo "andar";
    }
}
```

```php
class Homem extends Mamifero {

    public $cpf;

    public function __construct($idade, $peso, $cpf) {
        parent::__construct($idade, $peso);

        // Validações cpf
        $this->cpf = $cpf;
    }

    public function dirigir($carro) {
        echo "dirigir";
    }
}
```

Sobrescrita de métodos

- Métodos herdados podem ser sobrescritos
- Subclasse se comporta de maneira diferente para a mesma invocação do método
- Métodos sobrescritos substituem para a classe em questão os métodos da superclasse
- Superclasse continua com seu comportamento habitual, não sofrendo alterações

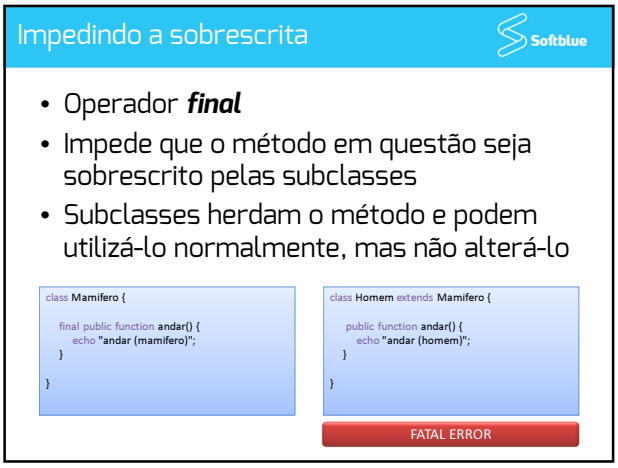
Sobrescrevendo métodos

```php
class Mamifero {

    public function andar() {
        echo "andar (mamifero)";
    }

}
```

```php
class Homem extends Mamifero {

    public function andar() {
        echo "andar (homem)";
    }

}
```

```php
$h = new Homem();
$h->andar(); // andar(homem)

$m = new Mamifero();
$m->andar(); // andar (mamifero)
```

Impedindo a sobrescrita



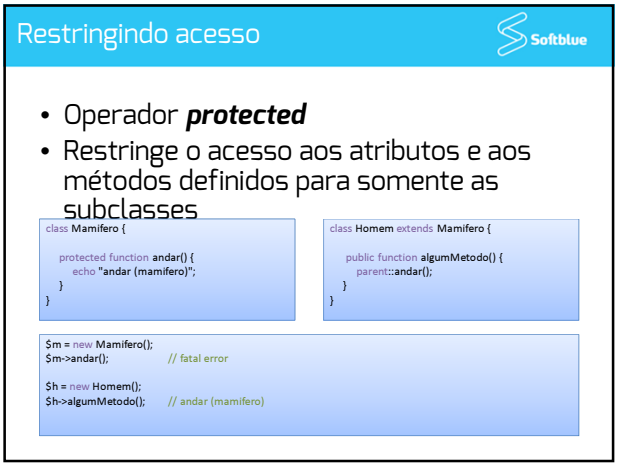
- Operador ***final***
- Impede que o método em questão seja sobrescrito pelas subclasses
- Subclasses herdam o método e podem utilizá-lo normalmente, mas não alterá-lo

```php
class Mamifero {

    final public function andar() {
        echo "andar (mamifero)";
    }

}
```

```php
class Homem extends Mamifero {

    public function andar() {
        echo "andar (homem)";
    }

}
```

FATAL ERROR

Restringindo acesso



- Operador ***protected***
- Restringe o acesso aos atributos e aos métodos definidos para somente as subclasses

```php
class Mamifero {

    protected function andar() {
        echo "andar (mamifero)";
    }
}
```

```php
class Homem extends Mamifero {

    public function algumMetodo() {
        parent::andar();
    }
}
```

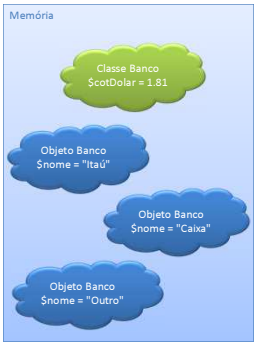
```php
$m = new Mamifero();
$m->andar();            // fatal error

$h = new Homem();
$h->algumMetodo();      // andar (mamifero)
```

Elementos estáticos

- Atributos e métodos que não dependem de valores específicos baseados na instância do objeto em questão
- Elementos que dependem da classe, e não dos objetos criados
- Operador ***static***

Como funcionam elementos estáticos

```
class Banco {

    private $nome;
    public static $cotDolar = 1.81;

    public function __construct($nome) {
        $this->nome = $nome;
    }

    public static function conversaoDeMoeda($reais) {
        return $reais / Banco::$cotDolar;
    }
}
echo Banco::$cotDolar;  // 1.81
echo Banco::conversaoDeMoeda(100);  // 55.24
Banco::$cotDolar = 1.83;
echo Banco::$cotDolar;  // 1.83
$b1 = new Banco("Itaú");
$b1 = new Banco("Caixa");
echo $b1->conversaoDeMoeda(100);  // 54.64
```



Classes estáticas

- Classes não podem ser estáticas
- Classes auxiliares, apenas elementos estáticos

```
class Geometria {

    public static $pi = 3.14;

    public static function calculaArea($raio) {
        return ($raio * $raio) * Geometria::$pi;
    }
}
echo Geometria::$pi;  // 3.14
echo Geometria::calculaArea(5);  // 78.5
```

Aulas práticas e manuais on-line

Assista agora as aulas práticas.

Clique aqui para visualizar as aulas práticas disponíveis