

# Orientação a Objetos

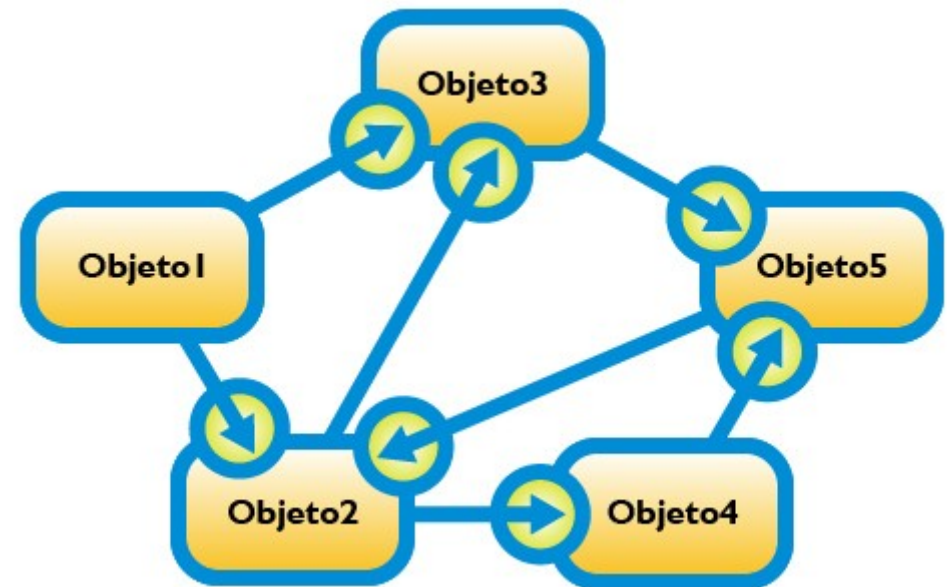
Prof. Daniel Di Domenico

## Interfaces



# O que já sabemos?

- Orientação a objetos
  - Abstração
  - **Classes e Objetos**
    - Atributos e Métodos
  - **Encapsulamento**
  - **Associação**



# Objetivos da aula

- Conhecer o conceito de **interfaces**
- Implementar e utilizar interfaces na linguagem PHP

# Interfaces

- **Interfaces** são estruturas que obrigam classes a implementar um ou mais métodos
  - É um contrato que define regras e restrições a serem seguidas
- **ATENÇÃO:** interfaces não são classes
  - Porém, as interfaces determinam como parte da classe deverá ser implementada

# Interfaces

- **Declaração de interfaces**
  - Uma interface é declarada com métodos sem implementação
  - Restrições:
    - A interface **não pode ter atributos**
    - Os **métodos não podem ser implementados**
      - Os métodos devem ser declarados apenas com a sua abstração
      - Os métodos devem ser declarados como públicos

# Interfaces

- **Utilização** de interfaces
  - Classes podem implementar uma ou mais interfaces
  - Condições:
    - A classe deve, **obrigatoriamente**, implementar todos os métodos da interface

# Interfaces: implementação

- **Declaração da interface**
  - Implementação em PHP:

**Padrão:** declarar com a inicial "I"

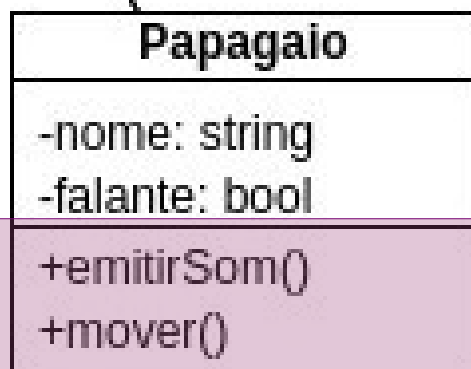
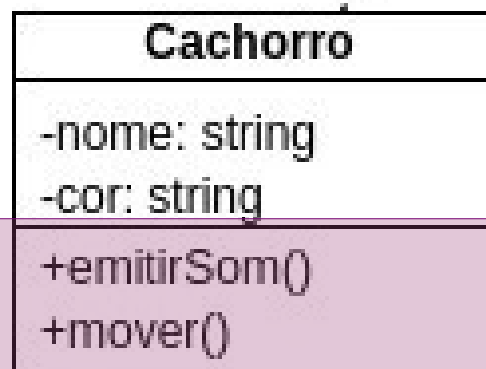
```
interface IAnimal {  
    public function emitirSom();  
    public function mover();  
}
```

**<<Interface>>  
IAnimal**

+emitirSom()  
+mover()

# Interfaces: implementação

- Utilização da interface



Métodos devem ser **redeclarados** nas classes que implementam a interface



# Interfaces: implementação

- **Utilização da interface**
  - Implementação em PHP:

```
class Cachorro implements IAnimal {  
  
    private string $nome;  
    private string $cor;  
  
    public function emitirSom() {  
        echo "Auauauauaua\n";  
    }  
  
    public function mover() {  
        echo "Andando\n";  
    }  
}
```

<<Interface>> IAnimal
+emitirSom() +mover()

Métodos da interface  
implementados na classe

# Interfaces: implementação

```
class Papagaio implements IAnimal {  
  
    private string $nome;  
    private bool $falante;  
  
    public function emitirSom() {  
        if($this->falante)  
            echo "Olá sou um papagaio\n";  
        else  
            echo "Rico rico rico\n";  
    }  
  
    public function mover() {  
        echo "Voando\n";  
    }  
}
```

<b>&lt;&lt;Interface&gt;&gt;</b> <b>IAnimal</b>
+emitirSom() +mover()

Métodos da interface  
implementados na classe

# Interfaces: implementação

- **Objetos**

- Implementação em PHP:

```
$tipo = "P"; //Poderia setar "C"
$animal = null;
if($tipo == "P")
    $animal = new Papagaio();
else if($tipo == "C")
    $animal = new Cachorro();

$animal->emitirSom();
$animal->mover();
```

Como ambas as classes implementam a interface *IAnimal*, o objeto **obrigatoriamente** possui os métodos *emitirSom()* e *mover()*.

# Exercícios

- **1-** Faça um programa que declare a interface *IFormaGeometrica* com os métodos:
  - *getArea()*: retorna a área da forma geométrica.
  - *getDesenho()*: retorna uma string com o desenho da forma geométrica utilizando apenas caracteres (<https://asciiflow.com/>)

Em seguida, declare as seguintes classes que implementam *IFormaGeometrica*:

- Circulo (atributo raio)
- Retangulo (atributos base e altura)
- Quadrado (atributo lado)

Por fim, implemente um programa que:

- Crie um objeto de uma das 3 classes de acordo com a escolha do usuário;
- Leia os atributos do objeto criado;
- Chame os métodos implementados pela interface *IFormaGeometrica*, imprimindo o seu retorno.

# Exercícios

- **2-** As concessionárias de energia calculam a fatura de luz com base no tipo de consumidor. De acordo com seu papel na sociedade, o custo do KWh é diferente. Neste sentido, implemente um programa com uma interface *IConsumidorEnergia* que definirá o método *getValorFatura()*. Após, declare as classes representando os tipos de consumidores para implementar a interface *IConsumidorEnergia*:
  - *Residencial*: deve cobrar R\$ 1,05 por KWh consumido.
  - *Comercial*: deve cobrar R\$ 1,45 por KWh consumido até 100 KWh, e R\$ 1,60 ao que exceder 100 Kwh.
  - *Industrial*: deve cobrar R\$ 1,80 por Kwh consumido até 500 KWh e R\$ 2,30 ao que exceder 500 KWh.

Todas as classes devem possuir um atributo denominado *consumo*.

Por fim, o programa a ser executado deve:

- Criar um objeto de uma das 3 classes para calcular o valor da fatura de energia. O tipo de consumidor deve ser lido para criar o objeto da classe correspondente;
- Ler o atributo consumo;
- Exibir o resultado do método *getValorFatura()*.