

### Programação Orientada à Objetos

Aula 13 Classes Abstratas e Interfaces

Henrique Poyatos henrique.poyatos bandtec.com.br

## Classes Abstratas O que são?



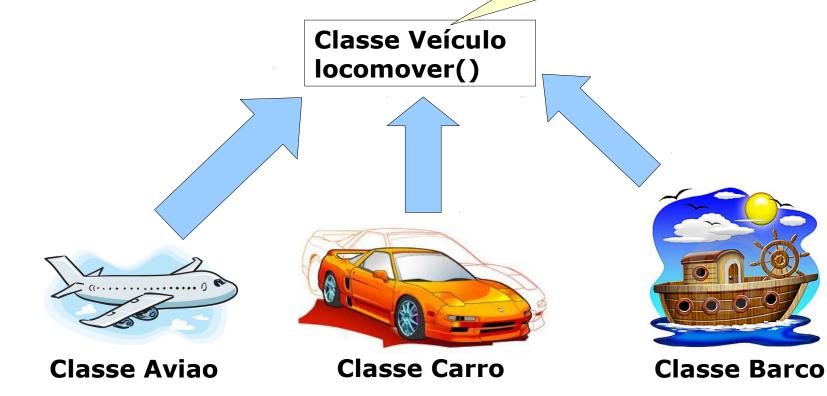
• São consideradas classes abstratas as classes que são tão generalizadas que não fariam sentindo algum se instanciadas como objeto.

- **PENSE:** O objeto "Veiculo" faria algum sentido, quando você tem objetos como "Aviao", "Carro" e "Barco" como opções?
- E "Pessoa"? Se houverem classes como "Aluno" ou "Professor"?

#### **Classe Abstrata**



Classe Veículo é ABSTRATA →
Proíbo instanciação de objetos
Ela só serve para reuso de código



#### Classe Abstrata Exemplo



```
Pode possuir métodos
                                 normalmente
abstract class Veiculo
       public virtual string locomover()
             Console.WriteLine("Veículo em movimento");
 Palavra-chave abstract →
                                 Subclasses herdam a partir dela
 Identifica a classe como abstrata.
                                 normalmente
class Carro : Veiculo
      public override string locomover()
             Console.WriteLine("Carro em movimento");
```

# **Métodos Abstratos** O que são?



- Classes Abstratas podem possuir métodos abstratos.
  - \* São métodos da qual declaramos apenas a assinatura;
  - Não possuem implementação;
  - \* Servem apenas para OBRIGAR a implementação do método;

**PERGUNTA:** Se colocamos em "Veiculo" um método abstrato Ligar(), qual é a certeza que teremos?

**RESPOSTA:** Que todos os seus filhos (Carro, Aviao, Barco) possuem um método Ligar() implementado.

#### **Métodos Abstratos** Exemplo



**Método abstrato** → A "intenção" em se ter um método Ligar().

```
abstract class Veiculo
      public abstract void Buzinar();
                                             Implementação forçada
                                             Os filhos são obrigados
                                             a implementar
class Carro : Veiculo
                                             Buzinar()
                                             (senão dá erro!)
      public override void Buzinar()
             Console.WriteLine("Carro buzinando!");
```

#### **Métodos Abstratos** Exemplo



```
Métodos abstratos e concretos
abstract class Veiculo
                                     convivem harmoniosamente.
      public abstract void Buzinar();
      public void DarSeta(string sentido) {
             Console.WriteLine("Virando para {0}", sentido);
class Carro : Veiculo
      public override void Buzinar()
             Console.WriteLine("Carro buzinando!");
```

## Interfaces x Classes Abstratas Diferenças



 Ambas não podem ser instanciadas → Servem apenas de "molde" para outras classes.. são generalizadas demais!

#### Interfaces

- \* Não são classes, são simplesmente "interfaces".
- Começam com a letra I maiúscula (convenção)
- \* As classes não herdam interfaces, elas as implementam;
- \* As classes podem implementar uma ou várias interfaces.
- Não possuem métodos declarados, só assinaturas, forçando quem implementa a declará-las. (forçando a implementação !)
- Os métodos não precisam das palavras-chave "abstract" ou "virtual".
- Os métodos não terão modificadores de acesso (public/protected/private/internal)
- ★ SERVE PARA ESTABELECER PADRÕES (Caso contrário Carro teria um Ligar(), Barco teria um Liga(), Moto um comecar() e por aí vai...)

## Interfaces x Classes Abstratas Diferenças



#### Classes Abstratas

- \* São classes não instanciáveis.
- As classes herdam da abstrata (uma só! - herança simples)
- Podem possuir métodos implementados e métodos abstratos.

## Interface Exemplo



```
public interface IEletrico
{
     void Ligar();
     void Desligar();
}

public interface IMovel
{
     void Acelerar();
     void Frear();
}
```

# Interface lEletrico Molde que obriga Equipamentos elétricos A possuir um liga e desliga.

#### Interface IMovel

Molde que obriga Equipamentos móveis a acelerar e frear.

## Interface Exemplo

Mais de uma interface pode ser implementada



```
public class Carro: IEletrico, IMovel
      public void Ligar() {
            Console.WriteLine("Liga");
      public void Desligar()
            Console.WriteLine("Desliga");
      public void Acelerar() {
            Console.WriteLine("Acelera");
      public void Frear() {
            Console.WriteLine("Frear");
```

## Implementação Forçada

A garantia que todos as classes que implementarem a interface terão estes métodos **EXATAMENTE** com estes nomes, estabelecendo **PADRÕES**.

## Interface Exemplo

1 classe abstrata + 2 interfaces!



```
public class Carro: Veiculo, IEletrico, IMovel
      public override void Buzinar() {
             Console.WriteLine("Carro buzinando!");
      public void Ligar() {
             Console.WriteLine("Liga");
      public void Desligar() {
             Console.WriteLine("Desliga");
      public void Acelerar() {
             Console.WriteLine("Acelera");
      public void Frear() {
             Console.WriteLine("Frear");
```