# IFRN

## PROGRAMAÇÃO ORIENTADA A OBJETOS EM C#

---

Classe Abstrata

Prof. Gilbert Azevedo

# Objetivos

- Utilizar classes abstratas na modelagem de aplicações
- Conceituar método abstrato
- Estudar a relação entre herança e classe abstrata
- Representar classes abstratas na UML

# Classes Abstratas

- Classes são modelos que descrevem entidades, definindo as características e comportamento dos objetos
  - Ex: Moto, Carro, Ônibus
- Classes abstratas definem modelos incompletos, precisando serem herdadas e complementadas
  - Ex: Veículo
- Uma classe abstrata não pode ser instanciada, sendo necessário, antes, que o modelo seja finalizado em numa classe derivada.
  - Ex: Não existem objetos da classe Veículo; existem objetos das classes Moto, Carro ou Ônibus, ....

# Classes Abstratas

- Classes abstratas possuem métodos não implementados (abstratos)
  - Ex: GetNumPneus – Não é possível obter o nº de pneus de um veículo
- Classes derivadas de uma classe abstrata devem implementar os métodos abstratos que são herdados
  - Ex: GetNumPneus – É possível obter o nº de pneus de uma moto
- Classes – Definem um modelo completo e todos os métodos são implementados
- Classes Abstratas – Definem um modelo incompleto e alguns métodos são abstratos (sem implementação)
- Interfaces – Definem um comportamento, nenhuma implementação

# Métodos Abstratos

- Os métodos abstratos em uma classe devem ser escritos com a palavra reservada *abstract* e não tem corpo (instruções) – *GetArea*
- A classe é abstrata se possui um método abstrato

```
abstract class Figura {
  private string cor;
  public Figura(string cor) { this.cor = cor; }
  public string GetCor() { return cor; }
  public abstract double GetArea();
}
```

# Herança e Classes Abstratas

- As classes derivadas, que herdam de uma classe abstrata, devem implementar todos os métodos abstratos
- Todo método abstrato é também virtual

```
class Circulo : Figura {
  public override double GetArea() {
    return Math.PI * r * r; }
}

class Triangulo : Figura {
  public override double GetArea() {
    return b * h / 2; }
}
```

# Herança e Classes Abstratas

- Classe Círculo derivada da classe abstrata Figura

```
class Circulo : Figura {
  private double r;
  public Circulo(string cor, double r) : base(cor) {
    this.r = r;
  }
  public override double GetArea() {
    return Math.PI * r * r;
  }
}
```

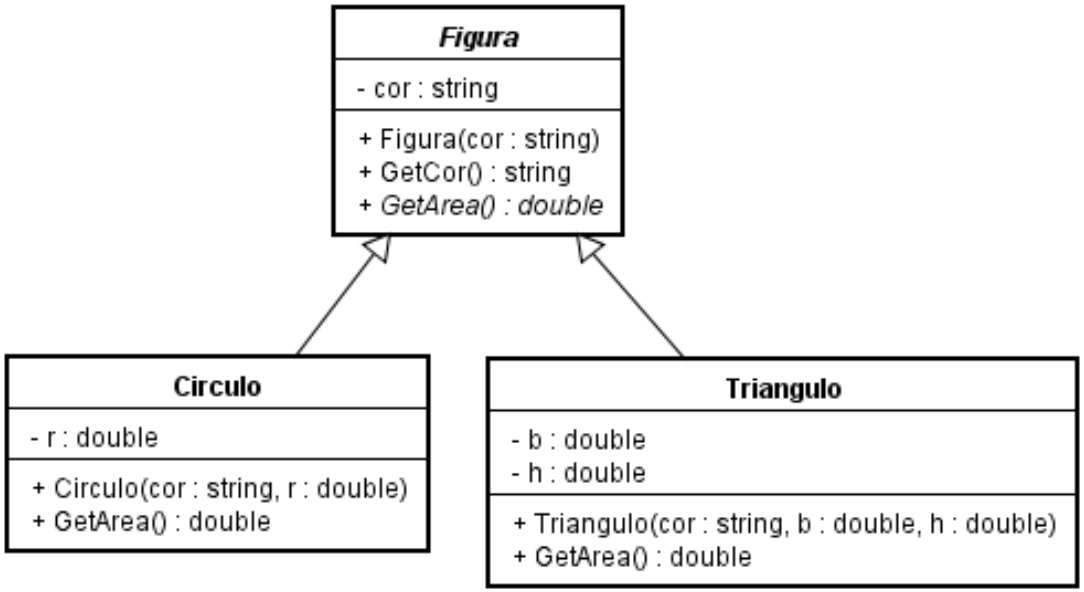
# Herança e Classes Abstratas

- Classe Círculo derivada da classe abstrata Figura

```
class Triangulo : Figura {
  private double b, h;
  public Triangulo(string cor, double b, double h) :
    base(cor) {
    this.b = b; this.h = h;
  }
  public override double GetArea() {
    return b * h / 2;
  }
}
```

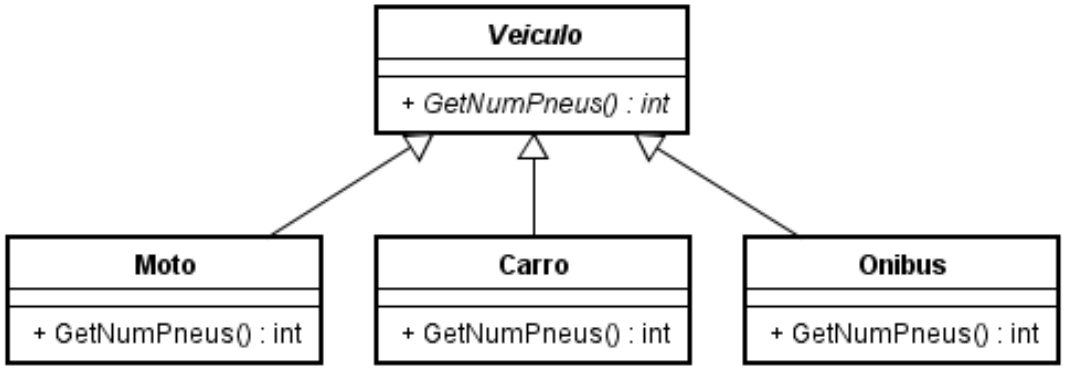
# Classes Abstratas e UML

- Classes e métodos abstratos são representados em itálico no diagrama de classes da UML



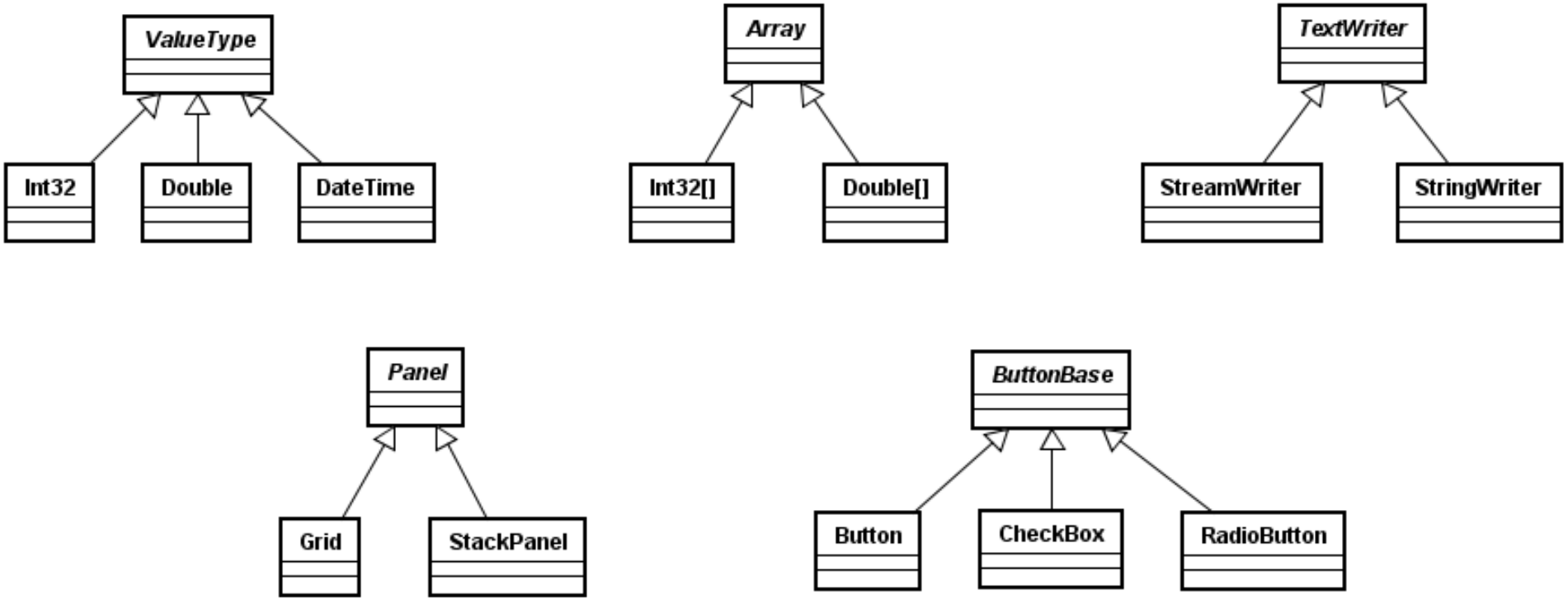
# Classes Abstratas e UML

- Classes e métodos abstratos são representados em itálico no diagrama de classes da UML



# Classes Abstratas e UML

- Frameworks utilizam frequentemente classes abstratas na definição de suas classes



# Referências


- Microsoft Visual C# 2010 – Passo a passo, John Sharp, Bookman, 2010
- UML – Uma Abordagem Prática, Gilleanes T. A. Guedes, Novatec, 2004
- Classes Abstratas (Guia de Programação C#)
  - https://docs.microsoft.com/pt-br/dotnet/csharp/programming-guide/classes-and-structs/abstract-and-sealed-classes-and-class-members
- Abstract (Referência de C#)
  - https://docs.microsoft.com/pt-br/dotnet/csharp/language-reference/keywords/abstract