- **UESC - Universidade Estadual de Santa Cruz**

Cap 4 – Classes Abstratas e Interfaces

Parte 1 – Classes Abstratas

Disciplina: Linguagem de Programação III
Professor: Otacílio José Pereira

Plano de Aula

- ***Objetivos***
  - Compreender classes abstratas e como codificá-las

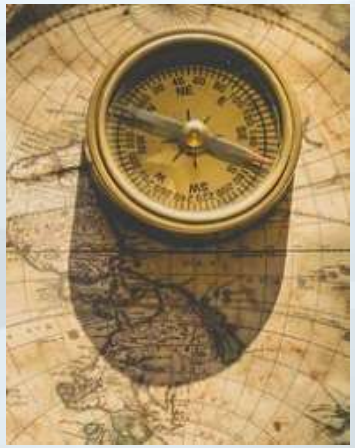
- ***Tópicos***
  - Contexto: Posicionamento na disciplina
  - Situações e discussões
  - Definição
  - Codificação
  - Aplicações

Contexto

- Onde estamos?
- Foco agora!
- Cenário de exemplo

Onde estamos?

- *Considerando nosso planejamento inicial*



- Capítulo 1 – Introdução
- Capítulo 2 - Conceitos básicos de Orientação a Objetos
- Capítulo 3 – Herança
  - Parte 1 – Herança
  - Parte 2 - Polimorfismo
- Capítulo 4 – Classes abstratas e interfaces
  - Parte 1 – Classes Abstratas
  - Parte 2 - Interfaces
- Capítulo 5 – Generics, collections e outros tópicos
- Capítulo 6 – Desenvolvimento de um projeto em OO

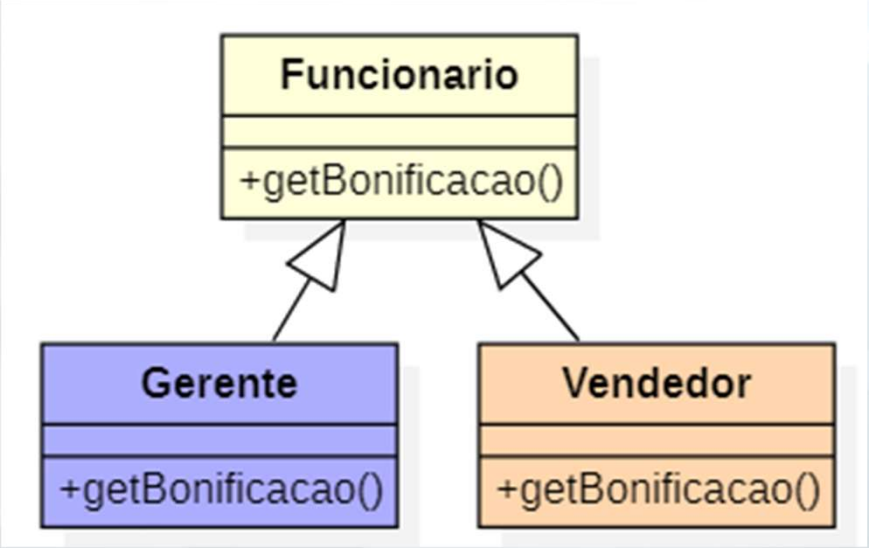
Foco

Situações e Discussões

- Funcionário
- Figuras Geométricas
- Conta Bancária

Cenário 1

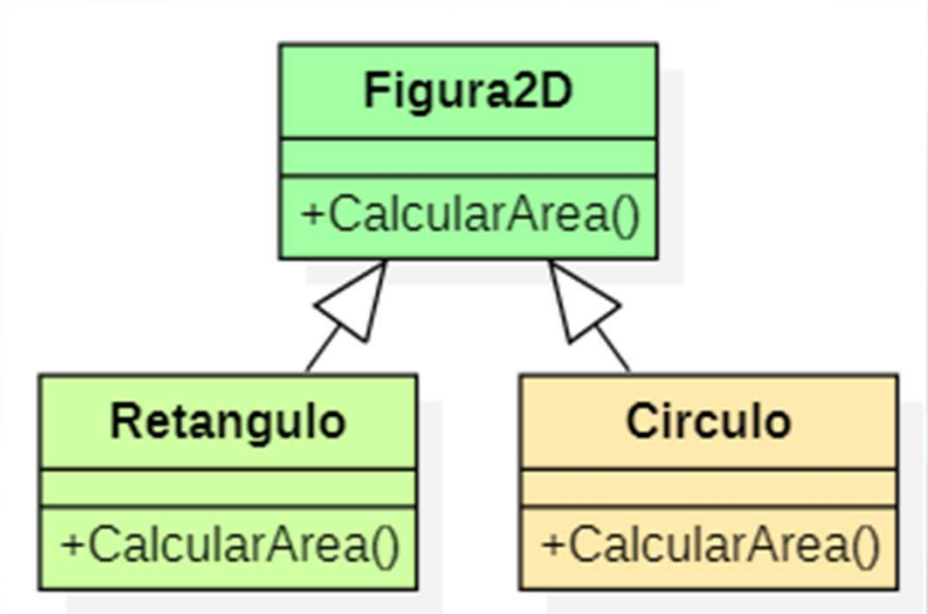
- Observe o cenário a seguir
  - Existe uma superclasse Funcionário
  - Existe as classes derivadas Gerente e Vendedor



- Imagine que o cálculo de bonificação só pode ser definido para Gerente e Vendedor, faz sentido codificar a lógica deste cálculo em Funcionário?
- E se na empresa existem apenas Gerentes e Vendedores fossem permitido, faz sentido ter um objeto do tipo funcionário?

Cenário 2

- Este cenário é ainda melhor para entender a ideia.
  - No exemplo anterior ainda fazia sentido ter um objeto de funcionário
- Perceba, faz sentido ter um objeto da classe Figura2D?
- Como seria calcular a área de um objeto Figura2D?
- Ou seja, fica bem claro que os objetos deve ser apenas das classes derivadas Retângulo ou Círculo.
- Mas apenas “pertubando” mais um pouco, porque então ter a classe Figura2D?

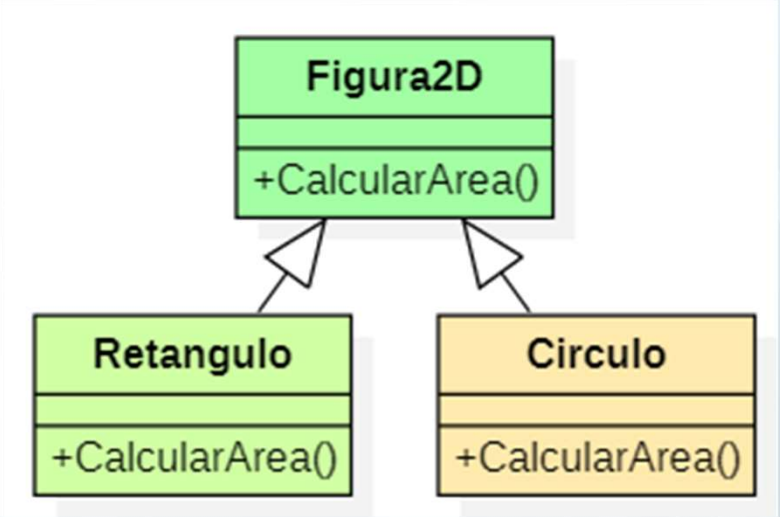


Classes abstratas

- Definição
- Exemplos
- Reforçando

Classes Abstratas

- Uma classe abstrata é uma classe que não permite que sejam instanciados objetos da classe
- Ou seja, não é possível criar instâncias de uma classe abstrata
- Ela serve apenas para acomodar atributos e métodos comuns às subclasses em geral



Classes Abstratas

- Por exemplo, caso sejam classes abstratas é possível executar os códigos a seguir

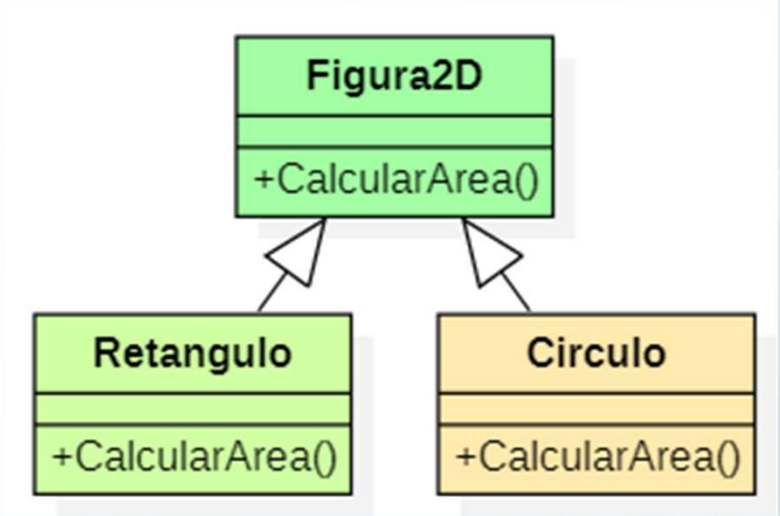
  Figura2D f = new Figura2D();

  Conta c = new Conta();

- Estes códigos não são permitidos, em vez disso faz-se

  Figura2D f = new Retangulo();

  Conta c = new Conta();



Classes Abstratas

- Mesmo não aceitando instâncias, perceba que elas ainda fazem sentido
- Isso porque estas classes além de permitir que o código comum a todas as subclasses sejam reusados
- Elas permitem também os mecanismos polimorfismo. Por exemplo, uma variável é de uma classe abstrata mas os objetos que terão os comportamentos concretos são das classes derivadas com suas características e comportamentos específicos.

```
Figura2d[] Desenho = new Figura2D[5];
Desenho[0] = new Retangulo();
Desenho[1] = new Criculo();
```

Classes Abstratas

- Perceba a codificação de classes abstratas
  - Na definição da classe usa-se o modificador **“abstract”**
  - As subclasses continuam herdando os métodos da mesma forma
  - Veremos a frente, mas os métodos que não fazem sentido na classe abstrata são definidos com “abstract” e sem código e são implementados nas subclasses

```
public abstract class Figura2D {

    protected double p1x, p1y;

    public abstract double CalcularArea();
```

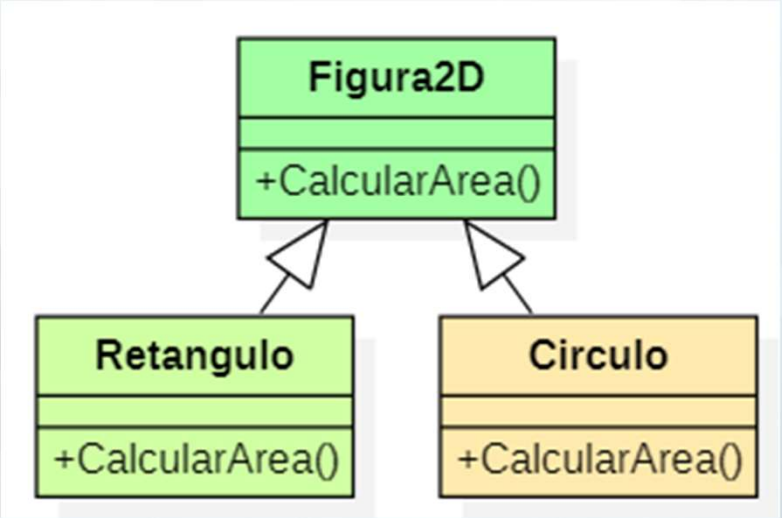
```
public class Retangulo extends Figura2D{
    private double p2x, p2y;

    @Override
    public double CalcularArea()
    {   double area;
        area = (this.p2x - this.p1x) *
               (this.p2y - this.p1y);

        return area;
    }
```

Métodos abstratos

- Situação
- Definição

Relembrando Cenário 1

- A classe Figura2D neste caso é abstrata, por exemplo, não sabemos como calcular a sua área.
- Mas sabemos também que Retângulo e Círculo precisam ter áreas calculadas
- Então, como fazemos para dizer que uma Figura2D precisa ter o cálculo de área mas que apenas será definido nas subclasses Retangulo e Círculo?



Métodos Abstratos

- A forma ou mecanismo de realizar esta situação da superclasse conter apenas a definição e as subclasses o código em si é por meio dos métodos abstratos
- Métodos abstratos apenas possuem assinatura, eles não contém código que explica como as coisas serão feitas/implementadas.
- A lógica de como processar o método fica delegado para as subclasses que vão herdar da superclasse

```
public abstract class Figura2D {

    protected double p1x, p1y;

    public abstract double CalcularArea();
```

```
public class Retangulo extends Figura2D{
    private double p2x, p2y;

    @Override
    public double CalcularArea()
    {   double area;
        area = (this.p2x - this.p1x) *
               (this.p2y - this.p1y);

        return area;
    }
```

Conclusões

- Retomando Plano de Aula
- Revisão
- Para saber mais

Plano de Aula

- ***Objetivos***
  - Compreender classes abstratas e como codificá-las

- ***Tópicos***
  - Contexto: Posicionamento na disciplina
  - Situações e discussões
  - Definição
  - Codificação
  - Aplicações