Orientação a Objetos 1

Classes Abstratas

Prof. Dr. Vinícius Camargo Andrade

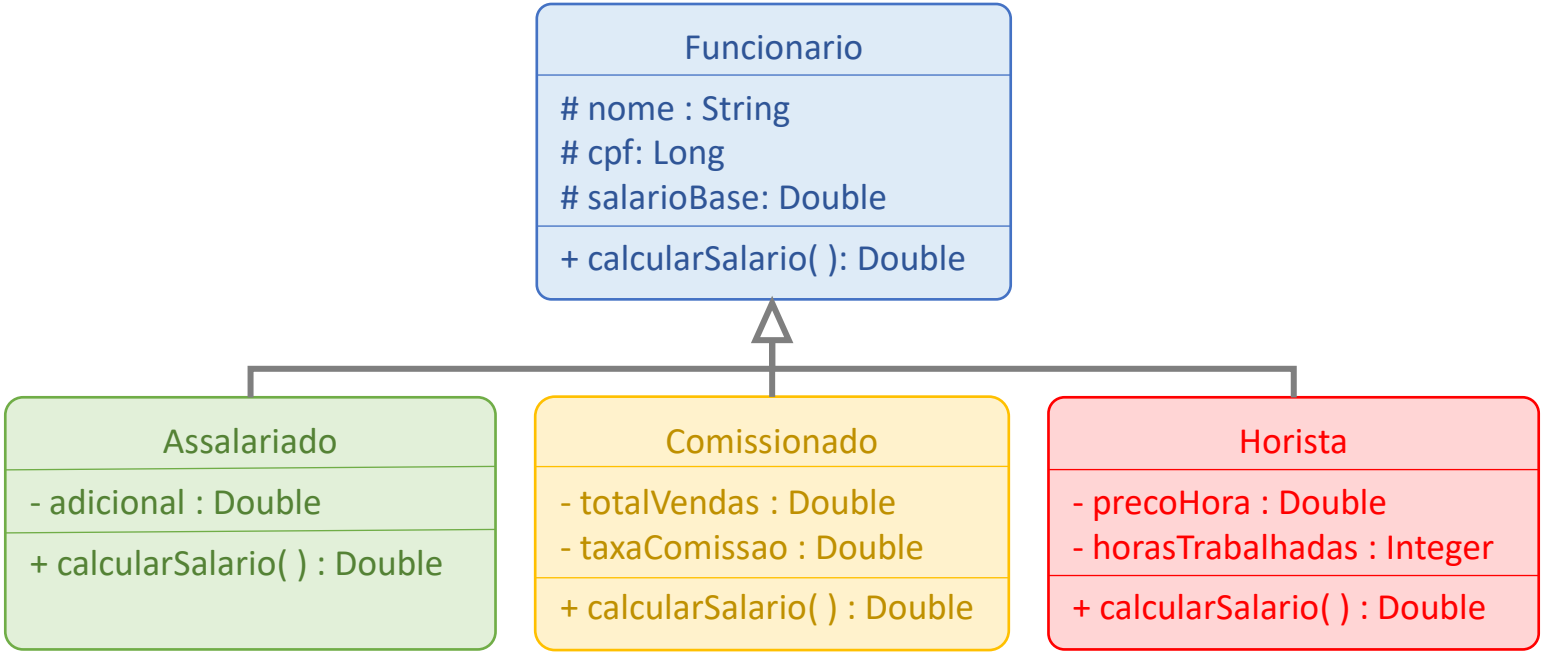
vcandrade@utfpr.edu.br

Departamento Acadêmico de Informática
Universidade Tecnológica Federal do Paraná

Cenário

Uma empresa possui três tipos diferentes de funcionários: Assalariado, Comissionado e Horista. Todos possuem um salário, porém o cálculo é diferente para cada funcionário, conforme o modelo.

Cenário


Funcionario
nome : String
cpf: Long
salarioBase: Double
+ calcularSalario( ): Double
Assalariado
- adicional : Double
+ calcularSalario( ) : Double
Comissionado
- totalVendas : Double
- taxaComissao : Double
+ calcularSalario( ) : Double
Horista
- precoHora : Double
- horasTrabalhadas : Integer
+ calcularSalario( ) : Double

```java
public class Funcionario {

    protected String nome;
    protected Long cpf;
    protected Double salarioBase;

    public Funcionario(String nome, Long cpf, Double salarioBase) {

        this.nome = nome;
        this.cpf = cpf;
        this.salarioBase = salarioBase;
    }

    public Double calcularSalario() {

        return null;
    }
```

```java
public class Assalariado extends Funcionario {

    private Double adicional;

    public Assalariado(String nome, Long cpf, Double salarioBase, Double adicional) {

        super(nome, cpf, salarioBase);
        this.adicional = adicional;
    }

    @Override
    public Double calcularSalario() {

        return super.getSalarioBase() + this.getAdicional();
    }
```

```java
public class Comissionado extends Funcionario {

    private Double totalVendas;
    private Double taxaComissao;

    public Comissionado(String nome, Long cpf, Double salarioBase, Double totalVendas, Double taxaComissao) {

        super(nome, cpf, salarioBase);
        this.totalVendas = totalVendas;
        this.taxaComissao = taxaComissao;
    }

    @Override
    public Double calcularSalario() {

        return super.getSalarioBase() + (this.getTotalVendas() * this.getTaxaComissao());
    }
```

```java
public class Horista extends Funcionario {

    private Double precoHora;
    private Double horasTrabalhadas;

    public Horista(String nome, Long cpf, Double salarioBase, Double precoHora, Double horasTrabalhadas) {

        super(nome, cpf, salarioBase);
        this.precoHora = precoHora;
        this.horasTrabalhadas = horasTrabalhadas;
    }

    @Override
    public Double calcularSalario() {

        return super.getSalarioBase() + (this.getHorasTrabalhadas() * this.getPrecoHora());
    }
```

```java
public class FuncionarioTeste {

    public static void main(String[] args) {

        Funcionario funcionario  = new Funcionario("João da Silva", 12345678910L, 5000.00);
        Funcionario assalariado  = new Assalariado("Maria de Oliveira", 98765432199L, 5000.00, 2000.00);
        Funcionario comissionado = new Comissionado("Carlos Santos", 15975398755L, 5000.00, 70000.00, 0.10);
        Funcionario horista      = new Horista("Bruna Rodrigues", 75375375398L, 5000.00, 25.00, 40.00);

        System.out.println("Funcionário: R$" + funcionario.calcularSalario());
        System.out.println("Assalariado: R$" + assalariado.calcularSalario());
        System.out.println("Comissionado: R$" + comissionado.calcularSalario());
        System.out.println("Horista: R$" + horista.calcularSalario());
    }
}
```

```
run:
Funcionário: R$null
Assalariado: R$7000.0
Comissionado: R$12000.0
Horista: R$6000.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

Problema

Problema

Para este sistema, a empresa não necessita de uma instância do tipo Funcionário, ou seja, não se pode permitir que a classe Funcionário seja instanciada.

Problema

```java
public class FuncionarioTeste {

    public static void main(String[] args) {

        Funcionario funcionario  = new Funcionario("João da Silva", 12345678910L, 5000.00);
        Funcionario assalariado  = new Assalariado("Maria de Oliveira", 98765432199L, 5000.00, 2000.00);
        Funcionario comissionado = new Comissionado("Carlos Santos", 15975398755L, 5000.00, 70000.00, 0.10);
        Funcionario horista      = new Horista("Bruna Rodrigues", 75375375398L, 5000.00, 25.00, 40.00);

        System.out.println("Funcionário: R$" + funcionario.calcularSalario());
        System.out.println("Assalariado: R$" + assalariado.calcularSalario());
        System.out.println("Comissionado: R$" + comissionado.calcularSalario());
        System.out.println("Horista: R$" + horista.calcularSalario());
    }
}
```

```
run:
Funcionário: R$null
Assalariado: R$7000.0
Comissionado: R$12000.0
Horista: R$6000.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

Problema

```java
public class FuncionarioTeste {

    public static void main(String[] args) {

        Funcionario funcionario  = new Funcionario("João da Silva", 12345678910L, 5000.00);
        Funcionario assalariado  = new Assalariado("Maria de Oliveira", 98765432199L, 3000.00, 2000.00);
        Funcionario comissionado = new Comissionado("Carlos Santos", 15975398755L, 5000.00, 70000.00, 0.10);
        Funcionario horista      = new Horista("Bruna Rodrigues", 75375375398L, 5000.00, 25.00, 40.00);

        System.out.println("Funcionário: R$" + funcionario.calcularSalario());
        System.out.println("Assalariado: R$" + assalariado.calcularSalario());
        System.out.println("Comissionado: R$" + comissionado.calcularSalario());
        System.out.println("Horista: R$" + horista.calcularSalario());
    }
}
```

```
run:
Funcionário: R$null
Assalariado: R$7000.0
Comissionado: R$12000.0
Horista: R$6000.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

Solução

Solução

Para casos como este, utiliza-se classes abstratas.

Classes Abstratas

Classes Abstratas

São classes definidas exclusivamente para servirem de classe base e não podem gerar objetos, ou seja, não podem ser instanciadas.

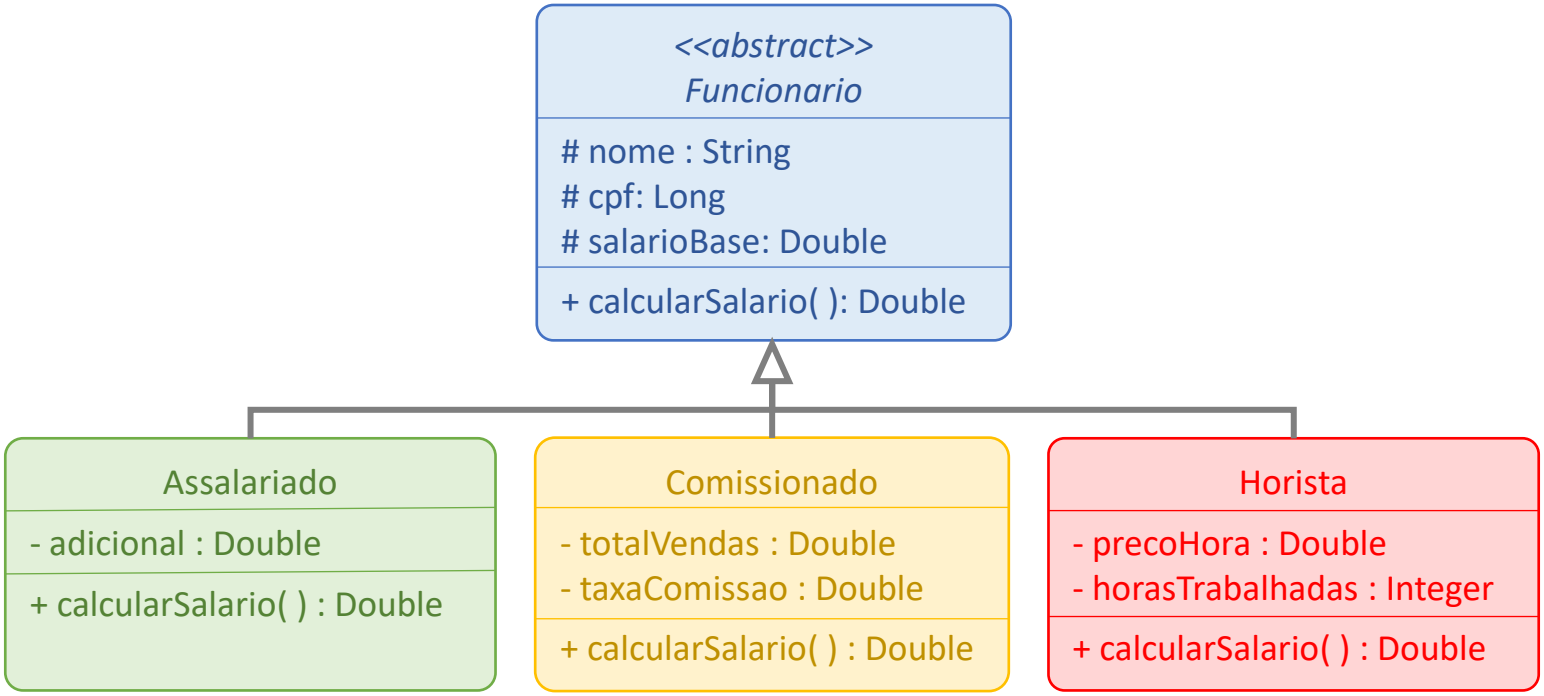
Classes Abstratas

É uma maneira de garantir herança total, ou seja, apenas subclasses não abstratas podem ser instanciadas.

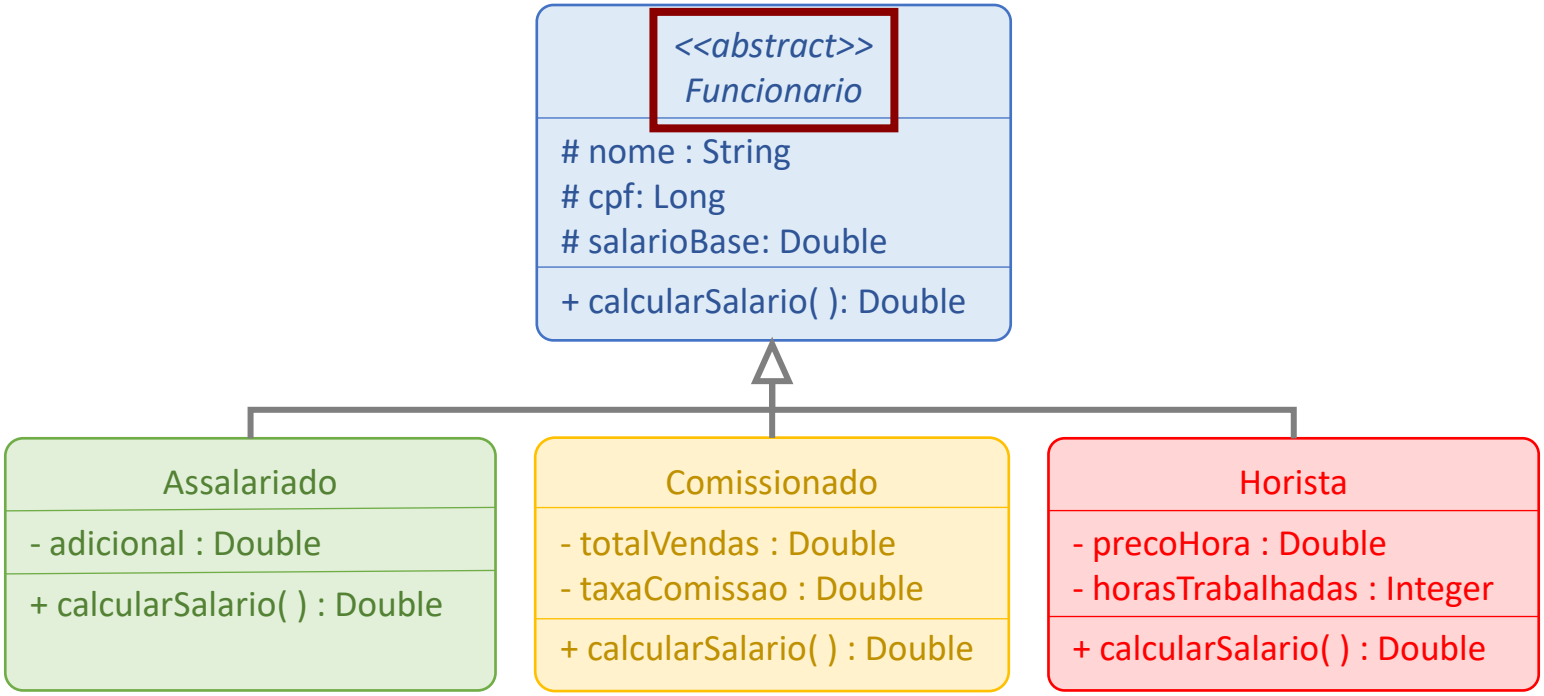
Modelo

Classes Abstratas

Modelo


<<abstract>>
Funcionario
nome : String
cpf: Long
salarioBase: Double
+ calcularSalario( ): Double
Assalariado
- adicional : Double
+ calcularSalario( ) : Double
Comissionado
- totalVendas : Double
- taxaComissao : Double
+ calcularSalario( ) : Double
Horista
- precoHora : Double
- horasTrabalhadas : Integer
+ calcularSalario( ) : Double

Modelo


<<abstract>>
Funcionario
nome : String
cpf: Long
salarioBase: Double
+ calcularSalario( ): Double
Assalariado
- adicional : Double
+ calcularSalario( ) : Double
Comissionado
- totalVendas : Double
- taxaComissao : Double
+ calcularSalario( ) : Double
Horista
- precoHora : Double
- horasTrabalhadas : Integer
+ calcularSalario( ) : Double

Implementação

Classes Abstratas

Classes Abstratas

Para se definir uma classe abstrata usa-se a palavra reservada abstract na declaração da classe.

Implementação

```java
public abstract class Funcionario {

    protected String nome;
    protected Long cpf;
    protected Double salarioBase;

    public Funcionario(String nome, Long cpf, Double salarioBase) {

        this.nome = nome;
        this.cpf = cpf;
        this.salarioBase = salarioBase;
    }

    public Double calcularSalario() {

        return null;
    }
```

Implementação

```java
public abstract class Funcionario {

    protected String nome;
    protected Long cpf;
    protected Double salarioBase;

    public Funcionario(String nome, Long cpf, Double salarioBase) {

        this.nome = nome;
        this.cpf = cpf;
        this.salarioBase = salarioBase;
    }

    public Double calcularSalario() {

        return null;
    }
```

Implementação

```
public class FuncionarioTeste {

    public static void main(String[] args) {

        Funcionario funcionario  = new Funcionario("João da Silva", 12345678910L, 5000.00);
        Funcionario assalariado  = new Assalariado("Maria de Oliveira", 98765432199L, 5000.00, 2000.00);
        Funcionario comissionado = new Comissionado("Carlos Santos", 15975398755L, 5000.00, 70000.00, 0.10);
        Funcionario horista      = new Horista("Bruna Rodrigues", 75375375398L, 5000.00, 25.00, 40.00);

        System.out.println("Funcionário: R$" + funcionario.calcularSalario());
        System.out.println("Assalariado: R$" + assalariado.calcularSalario());
        System.out.println("Comissionado: R$" + comissionado.calcularSalario());
        System.out.println("Horista: R$" + horista.calcularSalario());
    }
}
```

Funcionario is abstract; cannot be instantiated

(Alt-Enter shows hints)

Por que utilizar?

Por que utilizar?

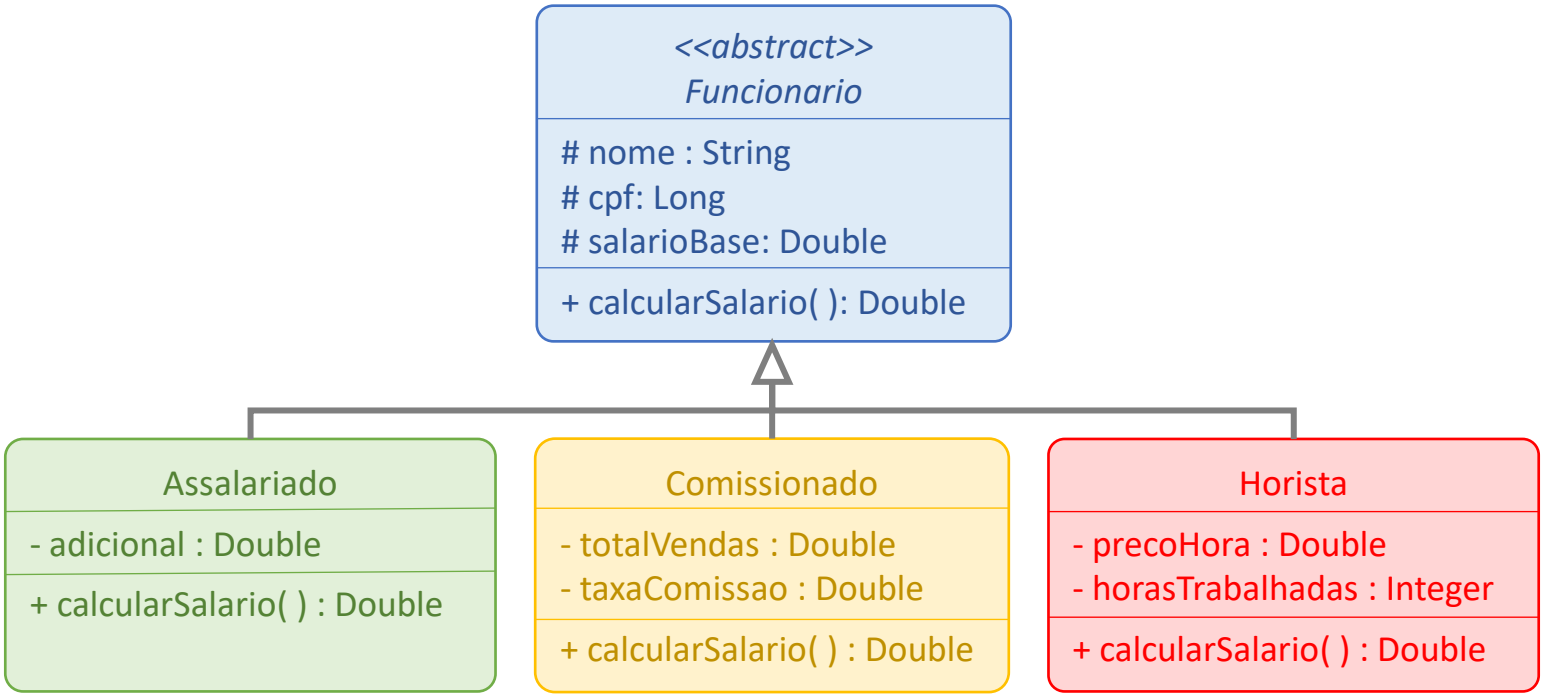
Se a classe Funcionário não será instanciada, por que então programá-la? Não seria mais fácil programar apenas as classes Assalariado, Comissionado e Horista?

Por que utilizar?

Com a generalização dos atributos e métodos,

evita-se a repetição desnecessária de código.

Por que utilizar?



Por que utilizar?

Pode-se utilizar o conceito de polimorfismo, uma vez que pode-se declarar um objeto do tipo genérico e atribuir a ele uma instância de uma das subclasses.

```java
public class FuncionarioTeste {

    public static void main(String[] args) {

        String opcao = "Comissionado";

        Funcionario funcionario = null;

        switch(opcao) {
            case "Assalariado":
                funcionario = new Assalariado("Maria de Oliveira", 98765432199L, 5000.00, 2000.00);
                break;

            case "Comissionado":
                funcionario = new Comissionado("Carlos Santos", 15975398755L, 5000.00, 70000.00, 0.10);
                break;

            case "Horista":
                funcionario = new Horista("Bruna Rodrigues", 75375375398L, 5000.00, 25.00, 40.00);
                break;

            default:
                System.out.println("Opção inválida.");
        }
        System.out.println("Funcionário: R$" + funcionario.calcularSalario());
    }
}
```