PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
DEPARTAMENTO DE COMPUTAÇÃO
TÉCNICAS DE PROGRAMAÇÃO 1 – CMP 1046
PROF. MSC. ANIBAL SANTOS JUKEMURA

[Interfaces]

PUC GOIÁS

Agenda:

- Interfaces
  - Conceitos
  - ***Exemplo***
- Exercício1: implementação simples
- Exercício2: implementação com Herança

PUC GOIÁS

Interfaces - Introdução

- A interface é um recurso muito na maioria das linguagens orientadas a objeto, para “obrigar” a um determinado grupo de classes a ter métodos ou propriedades em comum para existir em um determinado contexto. Contudo, os métodos podem ser implementados em cada classe de uma maneira diferente.

- Pode-se dizer, a grosso modo, que uma interface é um contrato que quando assumido por uma classe deve ser implementado.

- Uma **interface** Java descreve um conjunto de métodos que pode ser chamado em um objeto para instruí-lo, por exemplo, a realizar alguma tarefa ou retornar algumas informações.

- Objetos de software também se comunicam por **interfaces**.

PUC GOIÁS

Interfaces

- Uma declaração de interface inicia com a **palavra-chave interface** e **contém somente constantes e métodos abstract**.

- Diferentemente das classes**, todos os membros de interface devem ser public** e as **interfaces não podem especificar nenhum detalhe de implementação**, como declarações de método concretas e variáveis de instância.

- Todos os métodos declarados em uma interface são implicitamente **métodos public abstract**, e todos os campos são implicitamente **public, static e final**.

PUC GOIÁS

Interfaces

- Para utilizar uma interface, uma **classe concreta deve especificar** que ela **implementa** a interface e declarar cada método na interface com a assinatura especificada na declaração de interface.

- Para especificar que uma classe implementa uma interface, adicione a palavra-chave **implements** e o nome da interface ao final da primeira linha da declaração de classe.

- Uma classe que não implementa todos os métodos da interface é uma classe abstrata **e deve ser declarada abstract**.

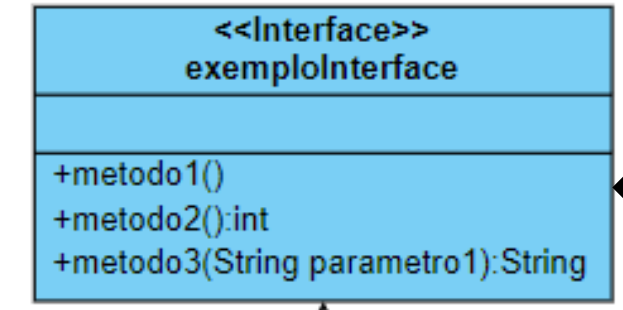
PUC GOIÁS

Interfaces

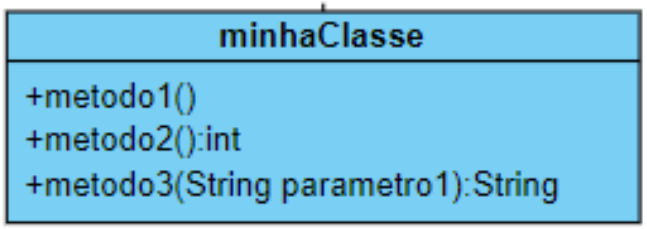
- Interfaces versus classes abstratas:
  - Uma interface costuma ser utilizada no lugar de uma classe abstract quando não há nenhuma implementação padrão a herdar — isto é, nenhum campo e nenhuma implementação padrão de método.
  - Como as classes public abstract, interfaces são tipicamente tipos public.
  - **Assim como uma classe public, uma interface public deve ser declarada em um arquivo com o mesmo nome que o da interface e a extensão de arquivo .java**.

PUC GOIÁS

Interfaces – Exemplo de uma interface simples




```
public interface exemploInterface {

    public void metodo1();
    public int metodo2();
    public String metodo3( String parametro1);

}
```

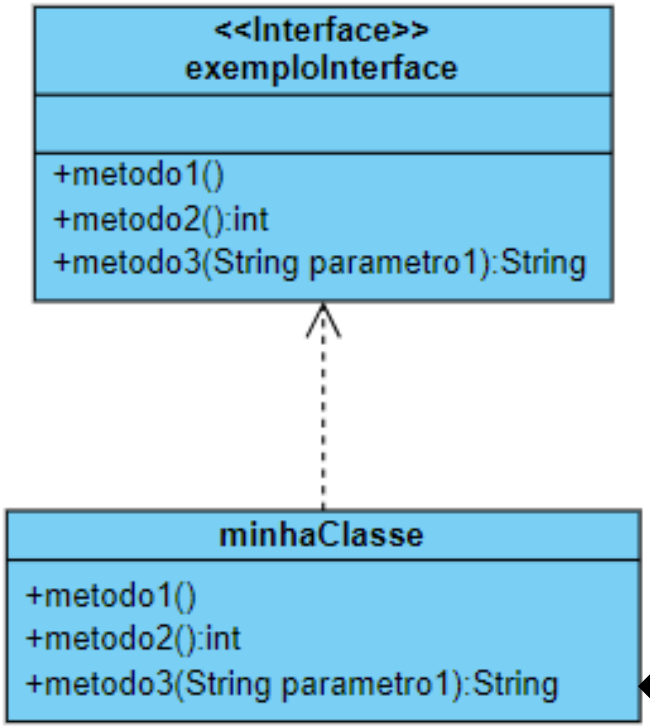


Esses métodos já estão implícitos na interface. Não precisam ser descritos aqui.

Perceba que os métodos na interface não têm corpo, apenas assinatura. Agora temos um "contrato" que deve ser seguido caso alguém a implemente.

PUC GOIÁS

Interfaces – Exemplo de uma interface simples

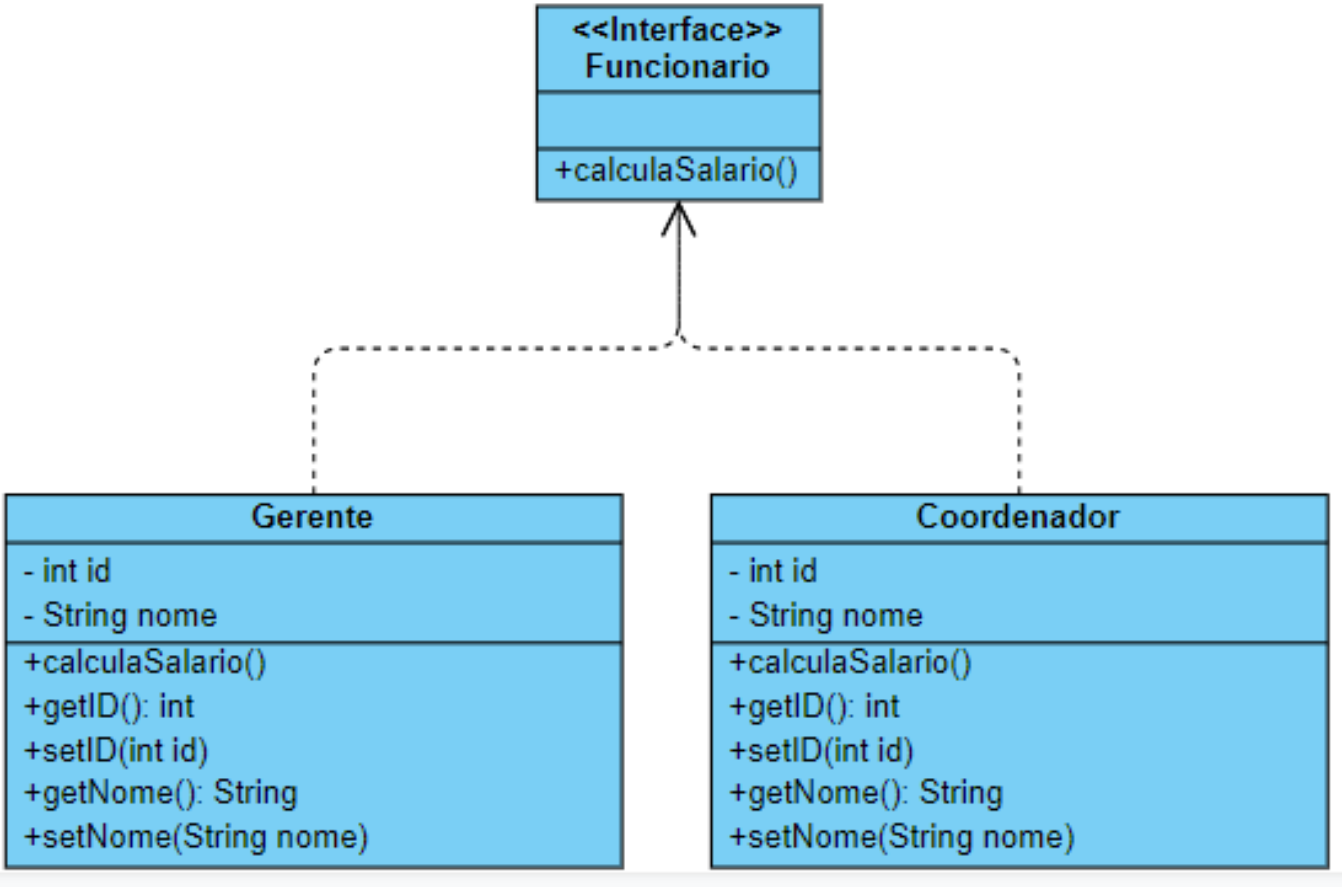



```
public class MinhaClasse implements exemploInterface {

    @Override
    public void metodo1 ( ) {
        // TODO Auto-generated method stub
    }

    @Override
    public int metodo2 ( ) {
        // TODO Auto-generated method stub
        return 0;
    }

    @Override
    public String metodo3 (String parametro1) {
        // TODO Auto-generated method stub
        return null;
    }
}
```

implements na MinhaClasse, faz a IDE (Eclipse, Netbeans ...) obrigar você a implementar os métodos descritos na Interface.

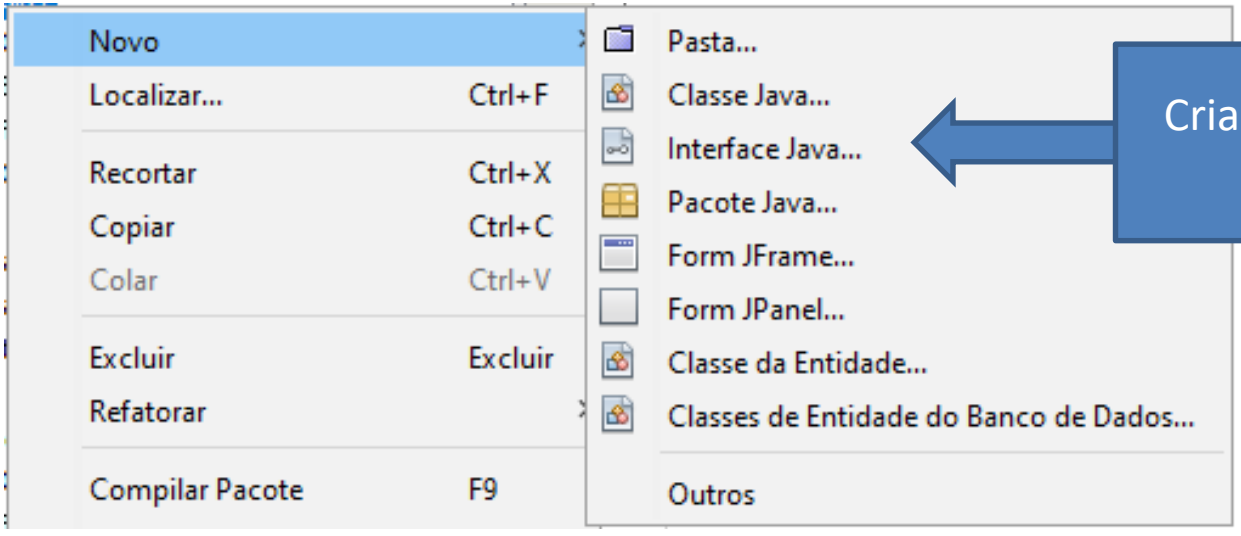
PUC GOIÁS

Interfaces – Exercício 1

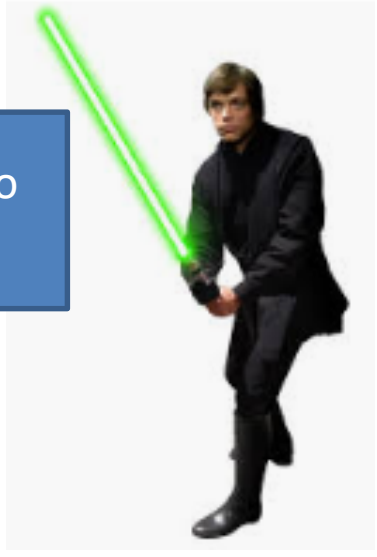


PUC GOIÁS

Interfaces – Exercício 1



Cria interface no projeto



PUC GOIÁS

Interfaces – Exercício 1

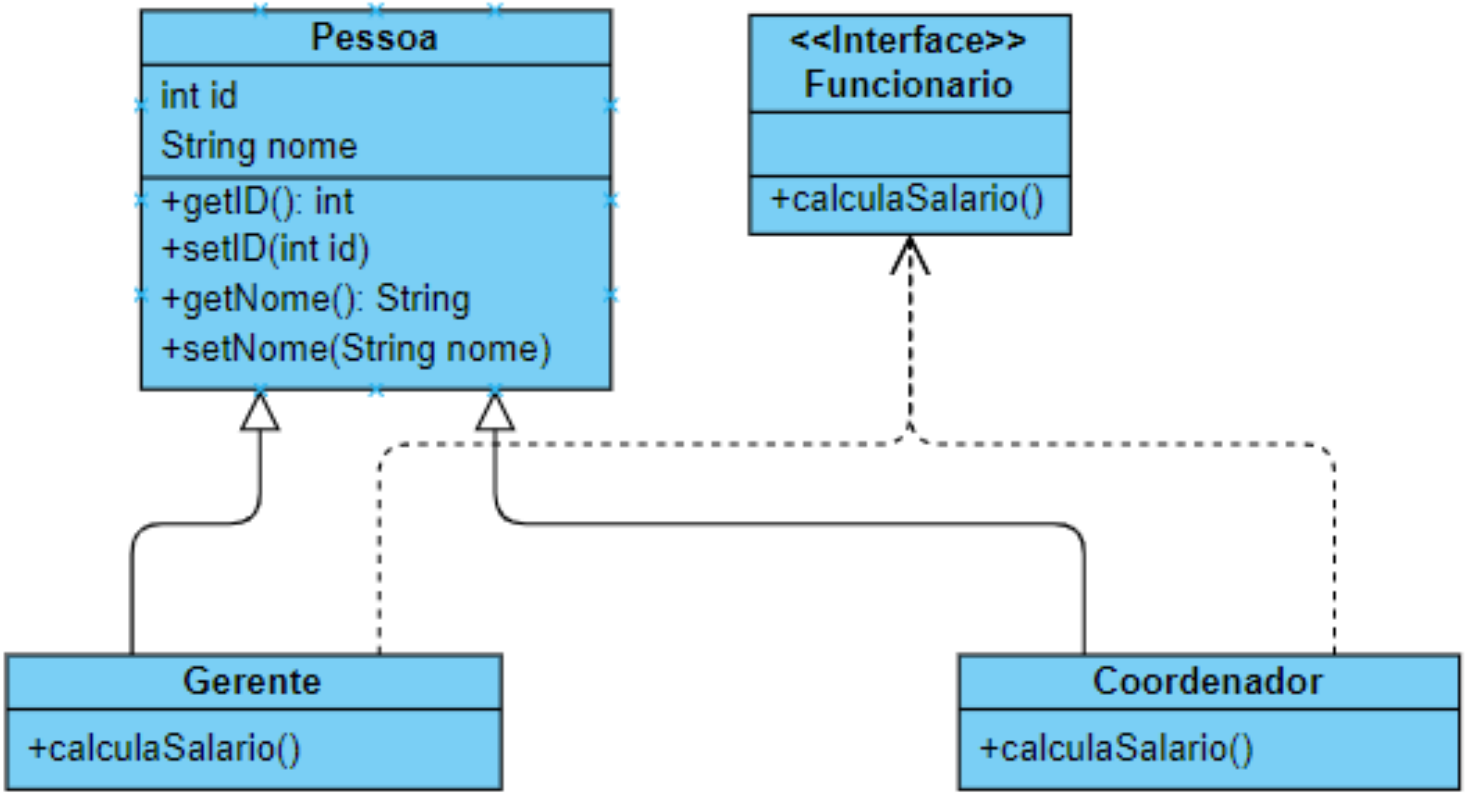
```
10      * @author japa
11      */
        public class Coordenador implements Funcionario {
13          int id;
14
15          public int getId() {
16              return id;
17          }
18
19          public void setId(int id) {
20              this.id = id;
21          }
22
23          public String getNome() {
24              return nome;
25          }
26
27          public void setNome(String nome) {
28              this.nome = nome;
29          }
30          String nome;
31
32
33
34      }
35
```

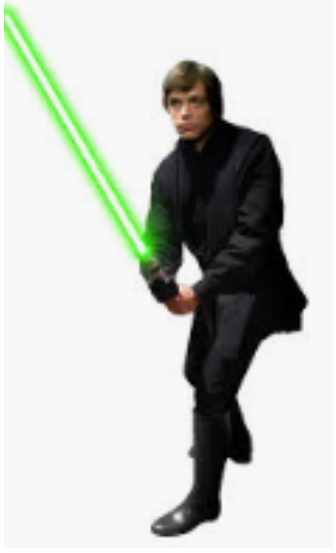
Gerar
- Construtor...
- Logger...
- equals() e hashCode()...
- toString()...
- Delegar Método...
- Implementar Método...
- Substituir Método...
- Adicionar Propriedade...

Implementa método da interface

PUC GOIÁS

Interfaces – Exercício 2

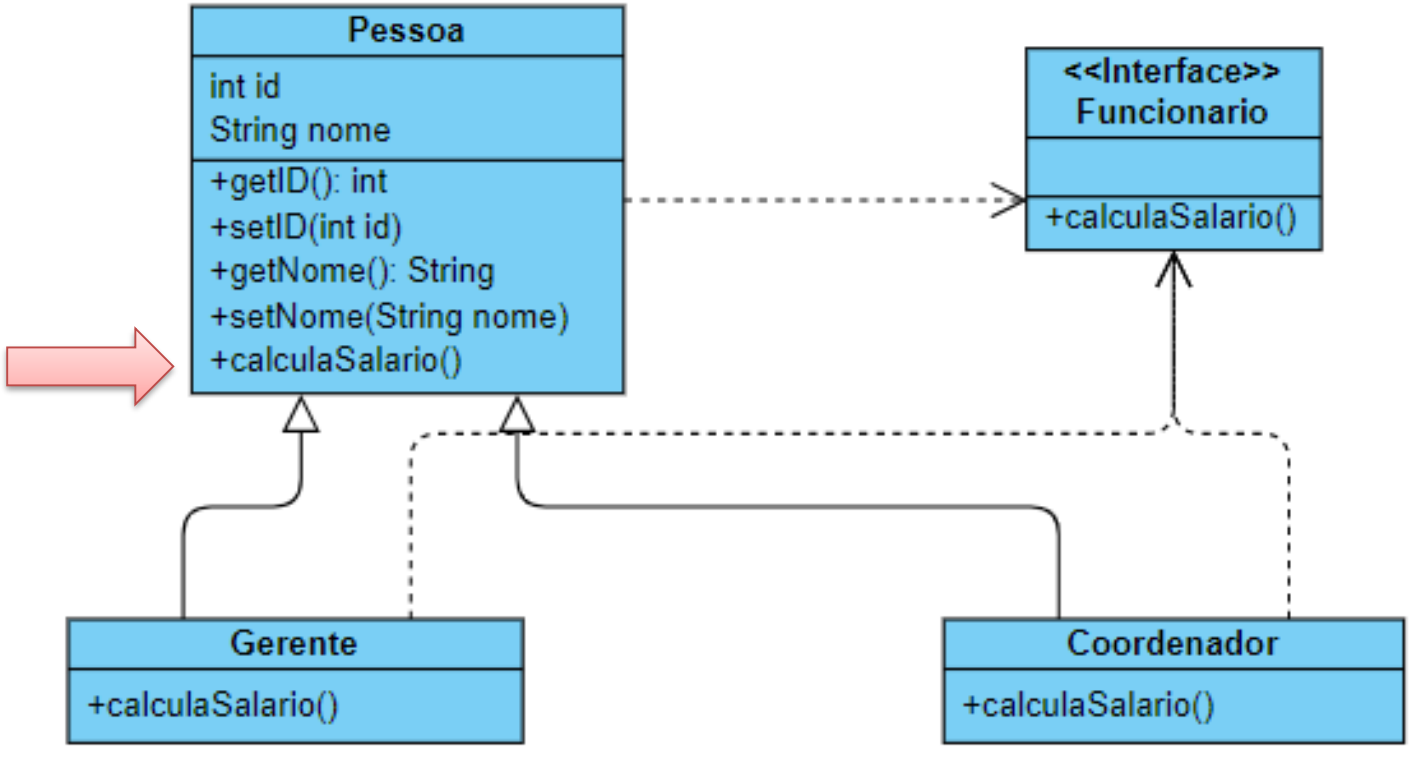




Não Declara Pessoa como Gerente ou Coordenador

PUC GOIÁS

Interfaces – Exercício 3

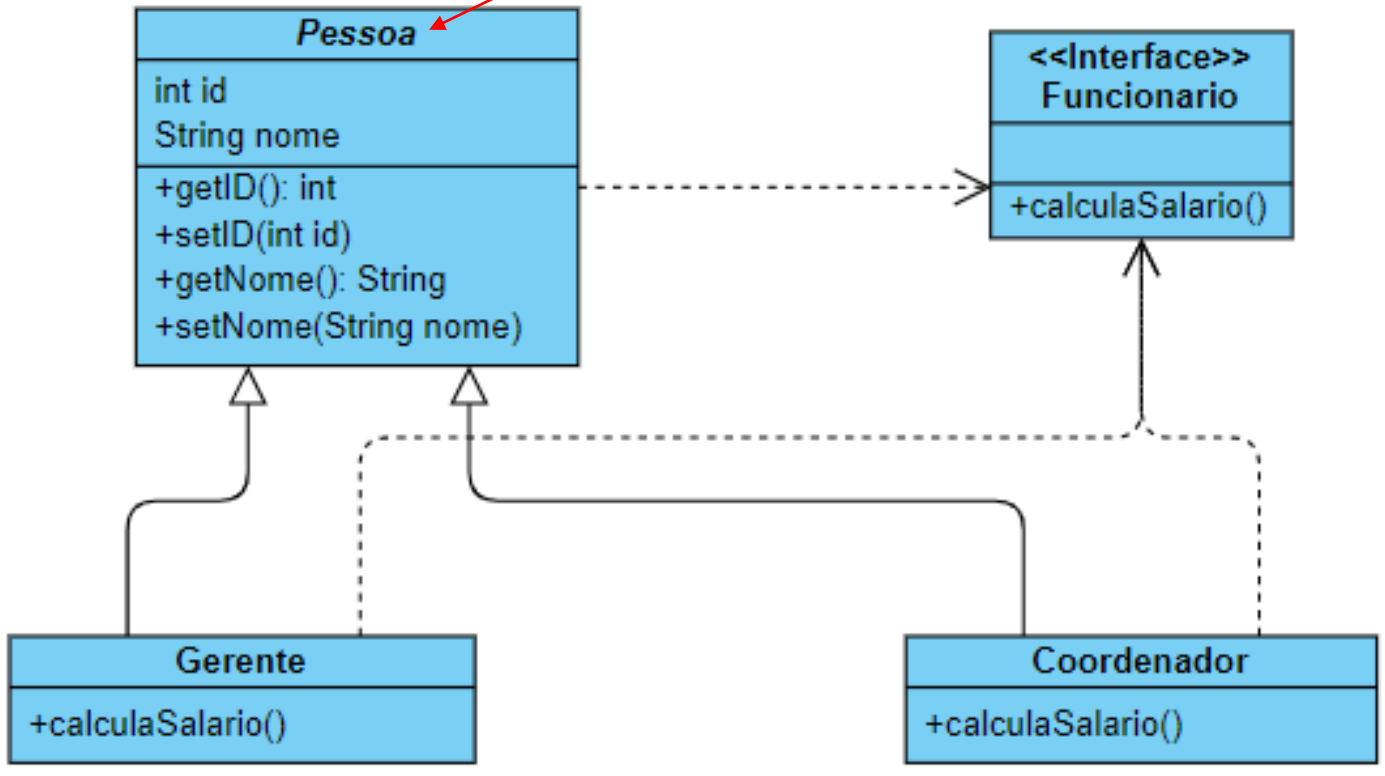


Declara Pessoa como Gerente ou Coordenador. Mas também instancia Pessoa

PUC GOIÁS

Interfaces – Exercício 4

Em itálico = abstract

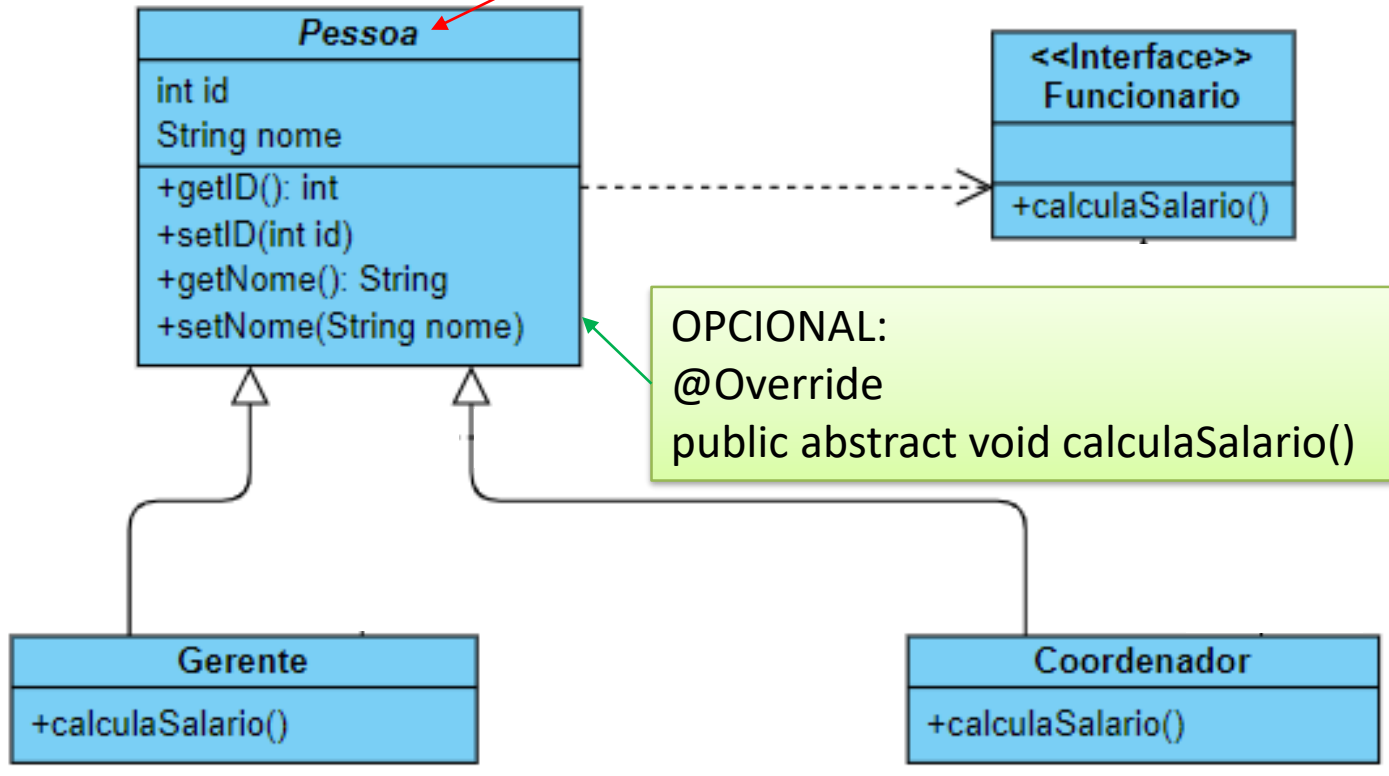


Declara Gerente ou Coordenador como Pessoa. Não instancia Pessoa

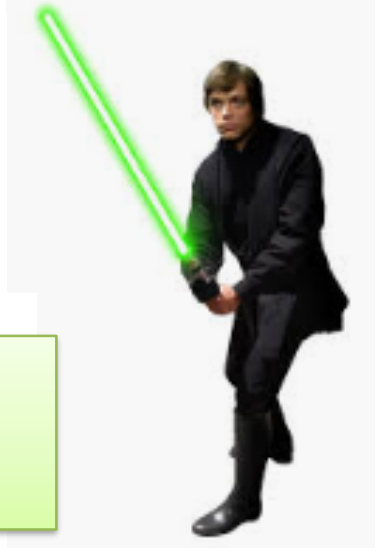
PUC GOIÁS

Interfaces – Exercício 5

Em itálico = abstract



OPCIONAL:
@Override
public abstract void calculaSalario()



Como Pessoa implementa a interface, Gerente e Coordenador não precisam implementar a interface (Herdam de Pessoa o calculaSalario – que também não precisa ser declarada em Pessoa).

PUC GOIÁS

BOAS PRÁTICAS DE PROGRAMAÇÃO

Boa prática de programação 10.1

De acordo com a especificação da linguagem Java, *o estilo adequado é declarar métodos* `abstract` *de uma interface sem as palavras-chave* `public` *e* `abstract`, *porque elas são redundantes nas declarações de método da interface. De maneira semelhante, as constantes da interface devem ser declaradas sem as palavras-chave* `public`, `static` *e* `final`, *porque elas também são redundantes.*

Erro comum de programação 10.6

Falhar em implementar qualquer método de uma interface em uma classe concreta que `implementa` *a interface resulta em um erro de compilação indicando que a classe deve ser declarada* `abstract`.

PUC GOIÁS

Referência Bibliográfica Principal

• DEVMEDIA. Disponível em https://www.devmedia.com.br. Acessado em Julho de 2019.

• DEITEL, Harvey M. Java: Como Programar – 10 ed. Cap 10. 2015.

• CAELUM. Disponível em https://www.caelum.com.br/apostila-java-orientacao-objetos/classes-abstratas/. Acessado em Agosto de 2019.