Programação Orientada a Objetos

Aula 09 – Interfaces
INF31098/INF31030

Prof. Dr. Jonathan Ramos
jonathan@unir.br

Departamento Acadêmico de Ciências de Computação – DACC

Núcleo de Tecnologia – NT

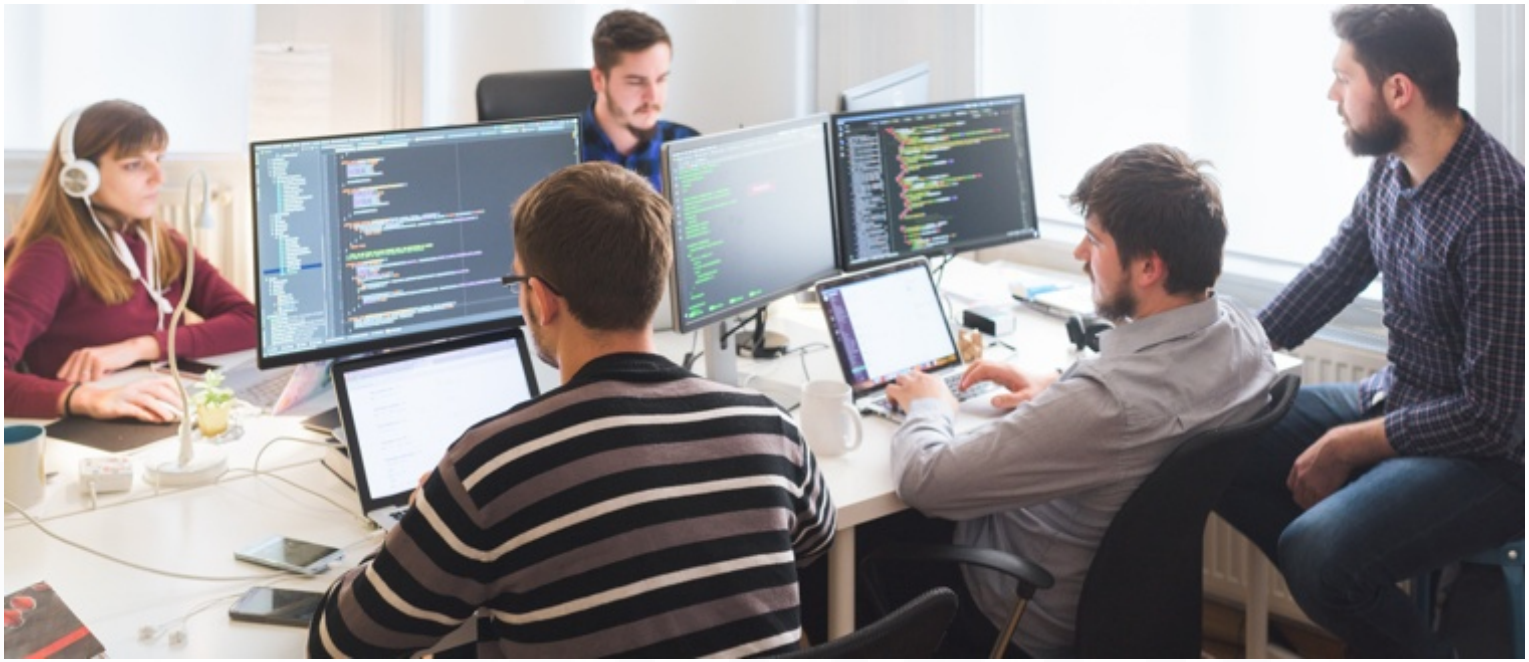
07/11/2022

Sumário

1 Definição de Interfaces

2 Exemplos

3 Exercícios Prático



Em geral, é comum que vários programadores trabalhe no mesmo projeto:

- É preciso **definir um "contrato"** entre os programadores;
- **Não importa como a implementação será feita**, desde que o contrato seja obedecido.

Em POO: as interfaces fornecem esse contrato

Definição de Interfaces: Exemplo USB

USB é um exemplo de interface:

Está presente em **diferentes dispositivos**, porém, **nem sempre tem o mesmo fim**!

- Hd Externo;
- Carregador de Celular;
- Mouse, teclado;
- **O que mais?**

As empresas que criam dispositivos que se conectam através de USB, só precisam conhecer o protocolo (mensagens trocadas) de uma conexão USB

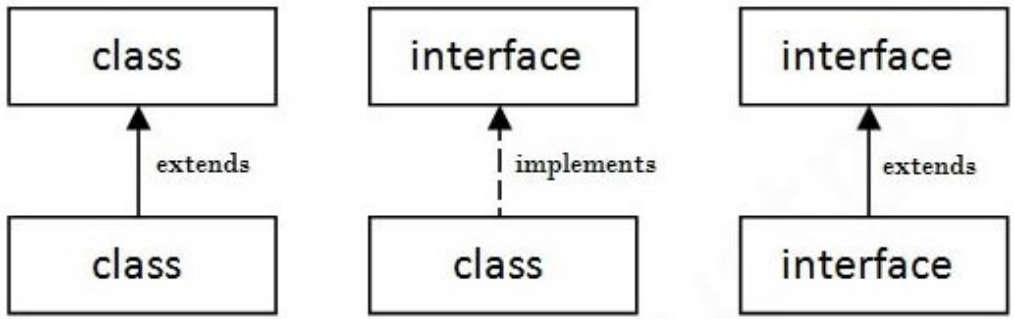
Definição de Interfaces: Em Java

interfaces são um tipo especial de referência, parecido com classes:

- **NÃO** podem ser instanciadas;
- Assim como classes, interfaces podem ser **public** ou **package-private**;
- Só podem conter **campos constantes**;
  - Implicitamente são **public**, **static** e **final**
- Os métodos são definidos **apenas pela sua assinatura**:
  - Implicitamente são **public** e **abstract**

Só podem ser implementadas por classes

Podem ser herdadas por outra interface:



Interfaces

Como declarar uma interface?

```
interface <nomeDaInterface>{
    // Declaração de campos constantes
    // Declaração de métodos abstratos
}

// Exemplo
interface imprimivel {
    // Antes de compilar:
    int min = 5;
    // Após compilar
    public static final int min = 5;

    // Antes de compilar:
    void imprimir();
    // Após compilar
    public abstract void imprimir();
}
```

- **static**: Quer dizer que a variável **pertence à classe, não ao objeto**.
  - Todas as instâncias de uma classe enxergam a mesma variável se uma delas modificar o valor vai refletir em todas as outras instâncias.
- **final**: Quer dizer que não pode atribuir valor duas vezes a variável.
- **static final**: o mesmo valor vai ser visto em todas instâncias da classe (**static**) e nunca vai poder ser modificado depois de inicializado (**final**)

Sumário

1 Definição de Interfaces

2 Exemplos

3 Exercícios Prático

Interface exemplo 01

Neste exemplo, a interface **Imprimivel** tem apenas um método abstrato – **print()** – que é implementado na classe A4:

```
interface Imprimivel {
    void imprimir();
}

class A4 implements Imprimivel {
    public void print() {
        System.out.println("Impressão na folha A4");
    }
}

public static void main(String args[]) {
    A4 objeto = new A4();
    objeto.imprimir();
}
```

Interface exemplo 02

Neste Exemplo, a interface **Desenho** tem apenas um método abstrato. A implementação dele é feita nas classes **Quadrado** e **Circulo**:

```
interface Desenho {  // Declaração da interface por um usuário
    void desenhar();
}
class Quadrado implements Desenho {  // Implementação
    void desenhar() { // por outro usuário
        System.out.println("Desenhando um Quadrado");
    }
}
class Circulo implements Desenho {  // Implementação diferente
    void desenhar() { // por um outro usuário
        System.out.println("Desenhando um círculo");
    }
}
public static void main(String args[]){
    Desenho desenho = new Circulo();
    desenho.desenhar();
}
```

Na prática:

A interface é **definida por um programador**, **implementadas por diversos outros programadores**. No final, acaba sendo usada por outra pessoa. Em geral, **a parte de implementação fica escondida de quem usa a interface**.

Interface exemplo 03

Banco fornece abstração para implementação do cálculo de moeda:

```
interface Banco {
    float valorMoeda();  // Com relação ao R$ (BRL)
}
// https://www.oanda.com/currency-converter/en/?from=BRL&to=USD&
    amount=1
class Dolar implements Banco {
    public float valorMoeda() { // USD
        return 0.19434f;
    }
}
// https://www.oanda.com/currency-converter/en/?from=BRL&to=JPY&
    amount=1
class Yen implements Banco {  // Yen Japonês
    public float valorMoeda() { // JPY
        return 28.6302f;
    }
}
public static void main(String[] args) {
    Banco banco = new Dolar();
    System.out.println("Valor de R$ 1 = " + banco.valorMoeda() + "
    Em Dolar");
}
```

Interface exemplo 04: Herança múltipla com interfaces

Herança múltipla:

É quando **uma classe implementa várias interfaces** ou **uma interface estende várias interfaces**:


```
interface Imprimivel{ void impirmir(); }
interface Desenho { void desenhar();  }
class A4 implements Imprimivel, Desenho {
    public void imprimir(){
        System.out.println("Imprimindo");
    }
    public void desenhar(){
        System.out.println("Desenho Qualquer");
    }
}
public static void main(String args[]){
    A4 obj = new A4();
    obj.desenhar();  obj.imprimir();
 }
```

Interface: Herança Múltipla

Porque herança múltipla não é suportada para classes??

- Devido **as ambiguidades que podem ocorrer**;
- **Isso não acontece nas interfaces**.

```
interface Imprimivel {
    void imprimir();
}
interface MostrarAlgo {
    void imprimir();
}
class TestInterfaces implements Imprimivel, MostrarAlgo {
    public void imprimir() {
        System.out.println("Olá mundo");
    }
}
public static void main(String args[]) {
    TestInterfaces obj = new TestInterfaces();
    obj.imprimir();
}
```

Imprimivel e **MostrarAlgo** possuem funções com nomes semelhantes:

Porém, a implementação somente é feita na classe **TestInterfaces**.

Interface: Herança Múltipla 02

Uma classe implementa uma interface:

Mas uma interface estende outra interface:

```
interface Imprimivel {
    void imprimir();
}
interface Mostrador extends Imprimivel {
    void mostrar();
}
class TestInterface Imprimivel Mostrador {
    public void imprimir(){
        System.out.println("Olá");
    }
    public void mostrar() {
        System.out.println("bem vindo");
    }
}

public static void main(String args[]){
    TestInterface obj = new TestInterface();
    obj.imprimir();
    obj.mostrar();
}
```

Interface: Métodos padrão (default)

Podemos implementar métodos padrões (default) no corpo da interface:

```
interface Desenho {
    void desenhar();
    default void mensagem() {
        System.out.println("Método padrão [default]");
    }
}
class Quadrado implements Desenho{
    public void desenhar(){
        System.out.println("Desenhando um quadrado");
    }
}

public static void main(String args[]){
    Desenho d = new Quadrado();
    d.desenhar();
    d.mensagem();
}
```

Isso permite:

Adicionar novos métodos a uma interface, tornando-os automaticamente disponíveis nas implementações.

Interface: Métodos estáticos (static)

Também podemos ter métodos estáticos (static) nas interfaces:

```
interface Desenho {
    void desenhar();
    static int cubo(int x) {
        return x*x*x;
    }
}
class Quadrado implements Desenho {
    public void desenhar() {
        System.out.println("Desenhando um quadrado");
    }
}
public static void main(String args[]){
    Desenho d = new Quadrado();
    d.desenhar();
    System.out.println(Desenho.cubo(3));
}
```

Métodos estáticos na interface:

- **Não pertencem a nenhuma classe de implementação.**
- **Deve ser chamado usando o nome da interface + Método:**

Desenho.cubo(valor)

Sumário

1 Definição de Interfaces

2 Exemplos

3 Exercícios Prático

Exercícios Prático

Implementar as seguintes interfaces da forma que achar mais adequada:

```
interface AnaliseData {
    // Calcula a idade de uma pessoa dado a data de
    nascimento e alguma outra data, pode ser hoje ou não,
    desde que a data para calcular seja maior que a data de
    nascimento
    int calcularIdade(LocalDate dataNascimento, LocalDate
    dataParaCalcular);
    int quantosDiasParaAniversario(LocalDate dataNascimento)
    ;
    int quantosDiasDesdeUltimoAniversario(LocalDate
    dataNascimento);
}
```

Exercícios Prático

Implementar as seguintes interfaces da forma que achar mais adequada:

```
interface NotasDisciplina {
    // Calcula a nota final na disciplina de POO (rs rs)
    considerando duas provas e um trabalho. A media das
    provas tem uma ponderação e a media dos trabalhos possui
     uma ponderação na nota final tbm: ((p1*wp1 + p2*wp2)/2
    )*pesoProvas + ( (t1*wt1+t2*wt2)/2) * pesoTrabalhos
    float mediaFinalAluno(float prova1, float pesop1, float
    prova2, float pesop2, float trabalho1, float pesot1,
    float trabalho2, float pesot2, float pesoProvas, float
    pesoTrabalhos);
    float mediaFinalAluno(List<Float> notasProvas, List<
    Float> pesoProvas, List<Float> notasTrabalhos, List<
    Float> pesoTrabalhos, float pesoProvas, float
    pesoTrabalhos);
}
```

Exercícios Prático

Implementar as seguintes interfaces da forma que achar mais adequada:

```
interface CalculaJuros {
    public static final double taxaJuros = 0.05; // 5%
    public abstract double calcularJurosSimples(double
    valorDevido);
    public abstract double calcularJurosSimples(double
    valorDevido, int qtMeses);
    // Pesquisar fórmula dos juros compostos
    public abstract double calcularJurosComposto(double
    valorDevido, int qtMeses);
}
```

Exercício Prático

Definir uma interface para o seguinte problema:

Uma empresa de transporte coletivo criará frotas em todos os países do mundo.

1. Na sua análise, quais métodos abstratos e/ou variáveis constantes deveriam existir nesse tipo de interface?
2. Escreva a interface com tudo que achar pertinente.

FIM!

jonathan@unir.br