



**Attribution-NonCommercial-
NoDerivatives 4.0 International
(CC BY-NC-ND 4.0)**



Este trabalho está licenciado com uma Licença [Creative Commons -
Atribuição-NãoComercial-SemDerivações 4.0 Internacional](https://creativecommons.org/licenses/by-nc-nd/4.0/).

Programação Orientada a Objetos - UTFPR Campus Apucarana



Programação Orientada a Objetos

BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

PROF. LUCIO AGOSTINHO ROCHA

**AULA 10:
INTERFACES E CLASSES ABSTRATAS**

2º.SEMESTRE 2022

Programação Orientada a Objetos - UTFPR Campus Apucarana

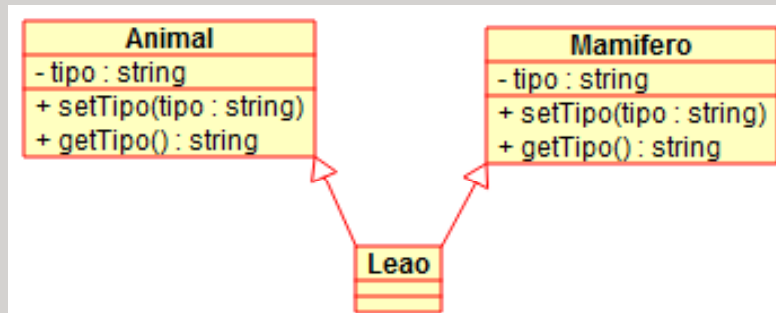
Interfaces

Interfaces

- **Interfaces:**
 - Java não suporta herança múltipla, mas admite múltiplas Interfaces.
 - Classes implementam Interfaces
 - ✦ A Interface garante que as classes implementem os métodos.
 - ✦ Métodos na interface devem ser declarados 'public abstract'
 - Interfaces permitem que métodos sejam implementados em Interfaces diferentes, e não todos em uma única classe.
 - Ao implementar uma interface a classe explicitamente deve definir qual método será implementado.

Interfaces

5



- Figura: Diagrama de Classes com Herança Múltipla.

Interfaces

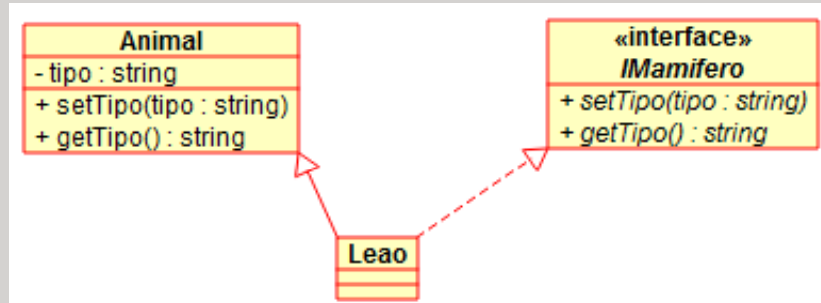
6

```
1#include <iostream>
2#include <string>
3
4#include "Leao.h"
5
6using namespace std;
7
8int main() {
9
10    Leao leao;
11    cout << leao.Animal::getTipo() << endl;
12    cout << leao.Mamifero::getTipo() << endl;
13
14
15    return 0;
16}
17
```

- C++ explicitamente informa a superclasse na chamada do método.

Interfaces

7



- Figura: Diagrama de Classes com Interface.

Interfaces

8

```
1
2 public interface IMamifero {
3
4     public final String tipo="mamifero";
5
6     public void setTipo(String tipo);
7
8     public String getTipo();
9 }
10
```

- Java: 1) Declaração dos métodos da interface.

Interfaces

9

```
1
2 public class Leao extends Animal implements IMamifero{
3
4     public Leao(){
5
6     }
7
8     public String toString(){
9         return this.getTipo();
10    }
11
12 }
13
```

- Java: 2) métodos da interface devem ser implementados ou sobrecarregados.

Interfaces

10

```
1
2 public class Principal {
3
4     public static void main(String[] args) {
5
6         Leao leao = new Leao();
7
8         System.out.println(leao);           //Animal
9         System.out.println(leao.tipo);      //Mamifero
10    }
11
12 }
13
```

- Java: 3) Declaração e Instanciação do objeto.

Classes Abstratas

Classes Abstratas

- **Classe Abstrata:**
 - Permitir que todas as classes herdem umas das outras é um risco de segurança.
 - Classe abstrata: fornece uma superclasse para a qual outras classes podem herdar.
 - Não instanciam objetos.
 - Subclasses devem implementar todos os métodos abstratos. Se não, a subclasse se torna abstrata.
- **Classe Concreta:**
 - Classes que permitem instanciar objetos.
 - Fornece modelo para instanciar objetos específicos.
 - ✦ Ex.: Circulo, Quadrado, Triangulo, Rosa, Margarida, Samambaia.

Classes Abstratas

13

Estudo de Caso com Classe Abstrata:

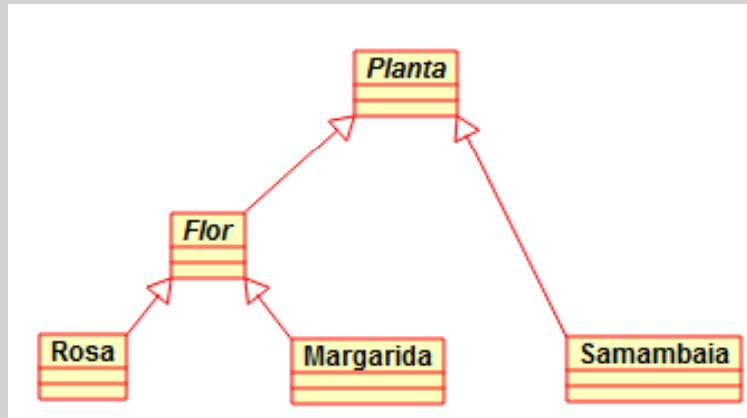


Figura: Classe Planta e Classe Flor são classes Abstract.
Subclasses folha não deveriam ser herdadas (final).

Classes Abstratas

14

Estudo de Caso com Classe Abstrata:

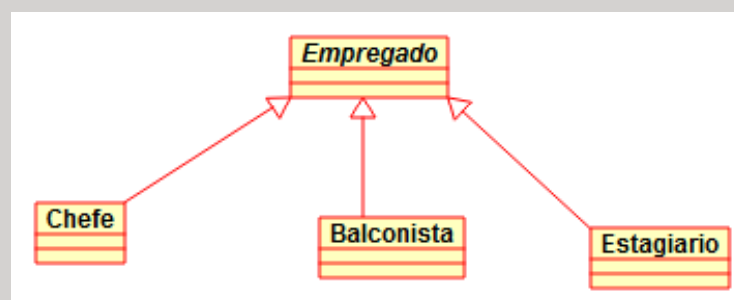


Figura: Classe Empregado é classe Abstract.
Subclasses folha não deveriam ser herdadas (final).

Classes Abstratas

15

- **Modificador de acesso 'final':**
 - Classes 'final' não podem ser herdadas.
 - Métodos 'final' não podem ser sobrecarregados.
 - Variáveis de instância 'final' são herdadas, mas não podem ser modificadas.

16



Revisão

Revisão

17

- Interfaces
- Classes Abstratas

Exercícios

18

<Ver conteúdo na plataforma de ensino>



Referências

19

- Referências bibliográficas da disciplina.