Programação III

Programação III - Herança

Prof. Vinicius F. Caridá

UEMG FRUTAL

C# Herança

Roteiro

- Herança
  - Conceituação
  - SuperClasse e subclasse
  - Herança Única x Herança Múltipla
  - O que pode ser feito na subclasse?
  - Implementação de uma herança simples
- Override
- Dynamic Binding

Herança

- A idéia de herança é simples, **mas poderosa**:

  "Quando você deseja criar uma **nova classe** e já existe uma classe que inclui alguns dos códigos que você quer, você pode **derivar** sua nova classe a partir da atual classe. Ao fazer isso, os atributos e métodos são **reutilizados** sem ter a necessidade de escrevê-los e depurá-los novamente" (The java Tutorials).

Herança

- Permite a criação de **classes idênticas** às já existentes, fazendo **reuso** de código.

- Uma classe que é derivada de uma outra classe é chamada uma **subclasse** (também chamada de classe derivada, classe estendida, ou classe filha).

- A classe a partir do qual a subclasse é derivada é chamada de **superclasse** (também chamada de classe base ou classe pai).

Herança

- Uma classe é uma subclasse de outra se a Pergunta **"É um (a)"** é **respondida afirmativamente**. Por exemplo: Cliente **é uma** Pessoa?

- Em Java, exceto a classe **Object**, que não tem superclasse, cada classe tem uma e apenas uma superclasse (Herança Única), ou seja, uma classe filha pode ter somente uma pai.

- O conceito de **Herança Múltipla,** que uma subclasse pode ter mais do que uma superclasse, em Java, não é possível sua implementação.

Herança

- O que pode ser feito em uma subclasse?
  - Os atributos herdados podem ser usados diretamente na subclasse, desde que eles tenham o modificador ***protected*** e as classes estejam no mesmo pacote.
  - A subclasse pode ter atributos com o mesmo nome da superclasse, mas isso **não é recomendado**.
  - Um método pode ser escrito na subclasse com a **mesma assinatura da superclasse**, ou seja, o método é sobrescrito. Esse conceito é denominado de **Override**.
  - Para fazer referência a superclasse a palavra-chave **super** é utilizada.

Herança

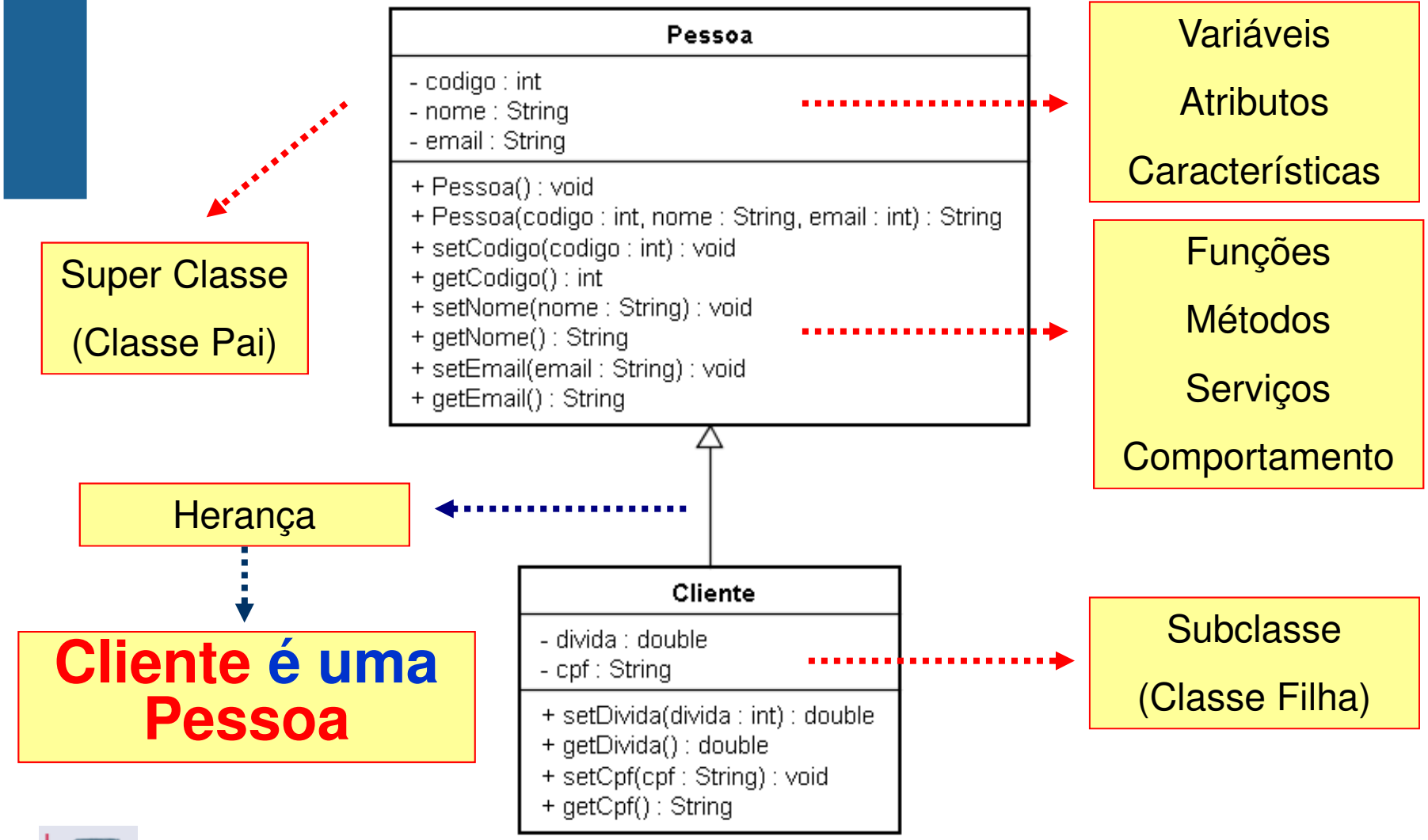
- Na ausência de qualquer superclasse escrita de maneira explícita, cada classe é implicitamente uma subclasse de **Object**.

- O conceito de Herança também é conhecido como: Generalização-Especialização (**Gen-espec**).

- Mas, o que é **herdado?**

  Uma subclasse herda todos **os membros** (atributos, métodos e classes aninhadas) de sua superclasse. **Construtores não são membros**, então eles não são herdados por subclasses, mas o construtor da superclasse pode ser invocado (chamado) a partir da subclasse.

Herança em UML



Implementação de Herança em Java

```
public class Cliente extends Pessoa {
    private double divida;
    private String cpf;

    public Cliente ( ) {
        super();
    } // fim do construtor

    public Cliente(int id, String n, String e,
                   int c, double d){
        super(id,n,e);
        cpf = c;
        divida = d;
    } // fim do construtor

    public int getCpf() { return cpf;   }
    public void setCpf(int c){ cpf = c; }

    public double getDivida() {return divida; }
    public void setDivida(double d) {divida = d;}
} // fim da classe
```

```
// Super Classe (classe pai)
public class Pessoa {
    private int codigo;
    private String nome;
    private String email;

    public Pessoa ( ){  }

    public Pessoa (int id, String n, String e){
        codigo = id;
        nome = n;
        email = e;
    }
} // fim da classe
```

Os dois construtores da **subclasse** invocam os construtores da **superclasse**

Utilizando a Classe Cliente

```
public class Principal {
    public static void main(String arg[]){
        Pessoa p1 = new Pessoa();
        Cliente c1 = new Cliente(10, "Juca", "ensinalegal@gmail.com", 100, "1010");

        p1 = c1;
        c1 = (Cliente) p1;

    }
}
```

Instanciação de um objeto da **superclasse**

Instanciação de um objeto da **subclasse**

Um objeto da superclasse recebe um objeto da subclasse

Para atribuir um objeto da superclasse para a subclasse é necessário realizar um conversão de tipo, denominada de **Cast**

Cast: conversão de tipo

Override

- **Um método de instância** em uma subclasse com a mesma assinatura e tipo de retorno **sobrepõe (override)** o método da superclasse.

- Na sobreposição de uma método pode ser utilizado a anotação **@override** para indicar ao compilador que o método deverá sobrepor ao método da superclasse.

- **Método de Classe** em uma subclasse com a mesma assinatura da superclasse não é sobreposto, ou seja, **não ocorrer Override para métodos de classe**.

Override

```
public class Animal {
    public Animal(){}
    public static void testeMetodoClasse() {
        System.out.println("Método de Classe: Animal");
    }

    public void testeMetodoInstancia() {
        System.out.println("Métod de Instância: Animal");
    }
} // fim da classe Animal

public class Cat extends Animal {
    public Cat(){super();}

    public static void testeMetodoClasse() {
        System.out.println("Método de Classe: Cat");
    }
    @Override
    public void testeMetodoInstancia() {
        System.out.println("Método de Instância: Cat");
    }
} // fim da classe Cat
```

Método de Classe

Método de Instância, que foi sobreposto (**Overrride**) na subclasse Cat

Método de Classe, que não é sobreposto

Override

Override e Dynamic Binding

```
public class Principal {

    public static void main(String[] args) {
        Cat myCat = new Cat();
        Animal myAnimal = myCat;
     (1) Animal.testeMetodoClasse();
     (2) myAnimal.testeMetodoInstancia();
    }
} // fim da classe Principal
```

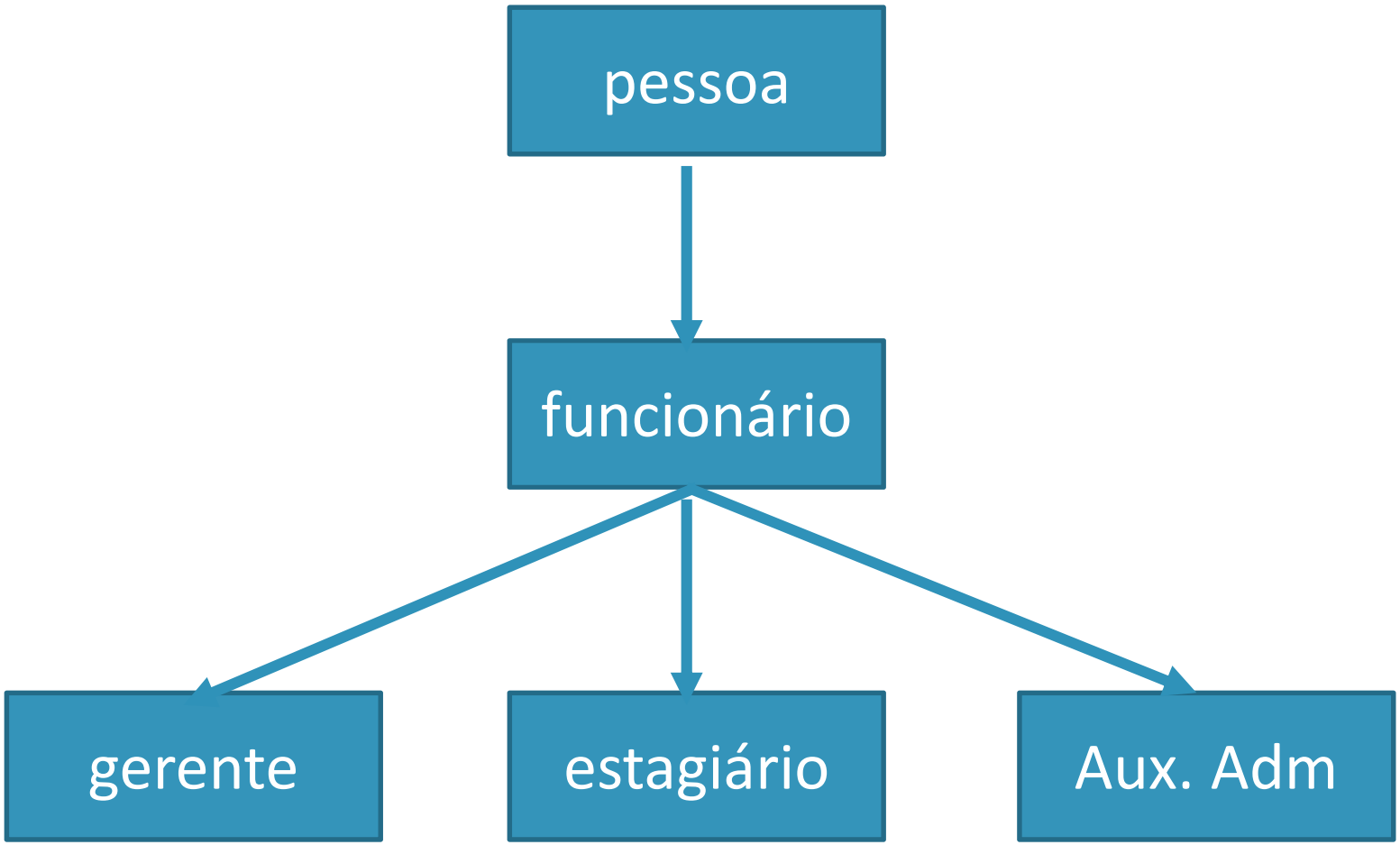
Saída:

(1) Método de Classe: Animal
(2) Método de Instância: Cat

Atenção:

Mesmo sendo executado um método da superclasse (Animal), em tempo de execução,foi chamado o método da subclasse, ou seja, **Dynamic Binding**.

Herança



Estudo de Caso: Herança – 3 Níveis de Hierarquia

- Hierarquia de 3 níveis, ponto/círculo/cilindro
  - Ponto
    - Par coordenado (x,y)
  - Círculo
    - Par coordenado (x,y)
    - Raio
  - Cilindro
    - Par coordenado (x,y)
    - Raio
    - Altura

UEMG FRUTAL

Estudo de Caso - Herança

- Aula Prática

Criar o sistema do estudo de caso

Vinicius Fernandes Caridá

vfcarida@gmail.com