Associação e Herança

Instituto Federal do Espirito Santo – Ifes
Campus Colatina – Programação Orientada a Objetos I
Professor: Giovany Frossard Teixeira

Associação

- Associação é uma ligação entre dois objetos.
- Associações podem ser implementadas como atributos e/ou como métodos.
- Suponhamos as classes Cliente e Prato, onde todo Cliente sabe o Prato que está comendo.
  - Dentro da classe Cliente deverá ser definido um atributo do tipo Prato, pois assim, será fácil extrair informações sobre o prato de um cliente com facilidade.
  - Nos próximos slides conheceremos as classes Cliente e Prato.

Exemplo

```
public class Prato {
    private String nome;
    private String inventor;
    public Prato(String Nome, String Inventor) {
        this.nome = Nome;
        this.inventor = Inventor;
    }
    public String getNome() {
        return nome;
    }
    public String getInventor() {
        return inventor;
    }
}
```

```
public class Cliente {
    private String nome;
    private Prato prato;
    public Cliente(String nome, Prato prato) {
        this.nome = nome;
        this.prato = prato;
     }
    public Prato getPrato() {
        return prato;
    }
    public String getNome() {
        return nome;
    }
}
```

Exemplo

```
public class UsaClientePrato {


    public static void main(String[] args) {
        Prato p = new Prato("Bife", "Romeu");
        Cliente c = new Cliente("Joao", p);
        System.out.println("O nome do prato de " + c.getNome() + " é " + c.getPrato().getNome());
    }
}
```

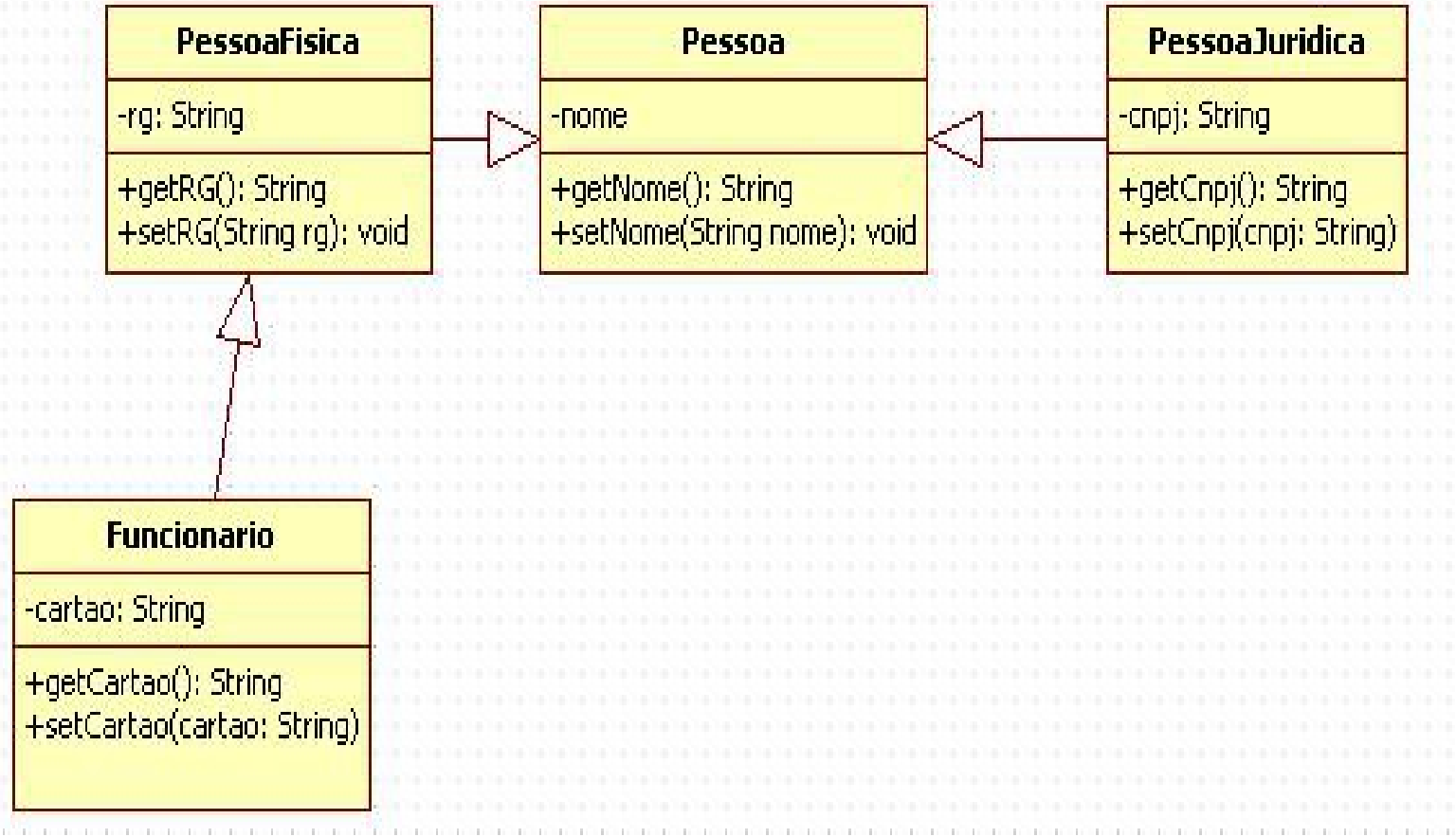
Herança - Conceito

- **Herança:** é um conjunto de características (atributos e métodos) que uma classe possui mas que foram passadas por outra classe, chamada classe ancestral ou superclasse.

- **Especialização:** quando uma classe herda de outra ela possui características da ancestral, mas ela pode implementar características específicas não implementadas na classe ancestral, tornando a classe descendente (classe que herda da ancestral) especializada em algum processo.

Herança – Ex:

- Implementaremos o seguinte diagrama de classes:



Herança – Ex:

```
package introducao_oo;

public class Pessoa {

    private String nome;

    public String getNome() {

        return nome;

    }


    public void setNome(String nome) {

        this.nome = nome;

    }

}
```

Herança – Ex:

```
package introducao_oo;

public class PessoaFisica extends Pessoa {

    private String rg;


    public String getRg() {

        return rg;

    }


    public void setRg(String rg) {

        this.rg = rg;

    }

}
```

```
package introducao_oo;

public class PessoaJuridica extends Pessoa {

    private String cnpj;

    public String getCnpj() {

        return cnpj;

    }


    public void setCnpj(String cnpj) {

        this.cnpj = cnpj;

    }

}
```

Herança – Ex:

```
package introducao_oo;

public class Funcionario extends PessoaFisica {

    private String cartao;

    public String getCartao() {

        return cartao;

    }


    public void setCartao(String cartao) {

        this.cartao = cartao;

    }

}
```

Herança – Ex:

```
package introducao_oo;

public class UsaFuncionario {

    public static void main(String[] args) {

        Funcionario funcionario = new
         Funcionario();

        funcionario.setNome("Lucas");

        funcionario.setRg("25.654.678-x");

        funcionario.setCartao("RH845");

        System.out.println(funcionario.getNome());

        System.out.println(funcionario.getRg());

        System.out.println(funcionario.getCartao());

    }

}
```

- Note que não é necessário implementar os métodos get e set para Nome e Rg no funcionário pois eles já foram definidos nas classes ancestrais.

Dúvidas?