Programação Orientada a Objetos

Aula 4 – Professor Diogo Orlando Nunes de Almeida

Conteúdo

- Conceitos do paradigma da Programação orientada a objetos;
- Diagrama de classes (UML);
- Introdução a classes e objetos;
- Atributos, métodos e interação entre objetos;
- Construtores e destrutores;
- Agregação e composição de objetos;
- Encapsulamento;
- Herança e polimorfismo;
- Tratamento de Exceções.

UML – Unified Modeling Language

- Trata-se de uma linguagem com uma especificação semântica semiformal, que inclui sintaxe abstrata, regras bem-definidas e semântica dinâmica. (Page-Jones, p.80)
- Diagrama de Classe: Retrata as construções de herança, associação semântica, composição e agregação. (Page-Jones, p.80)
- A linguagem provê uma estrutura gráfica para a organização de construções de projeto. (Page-Jones, p.80)

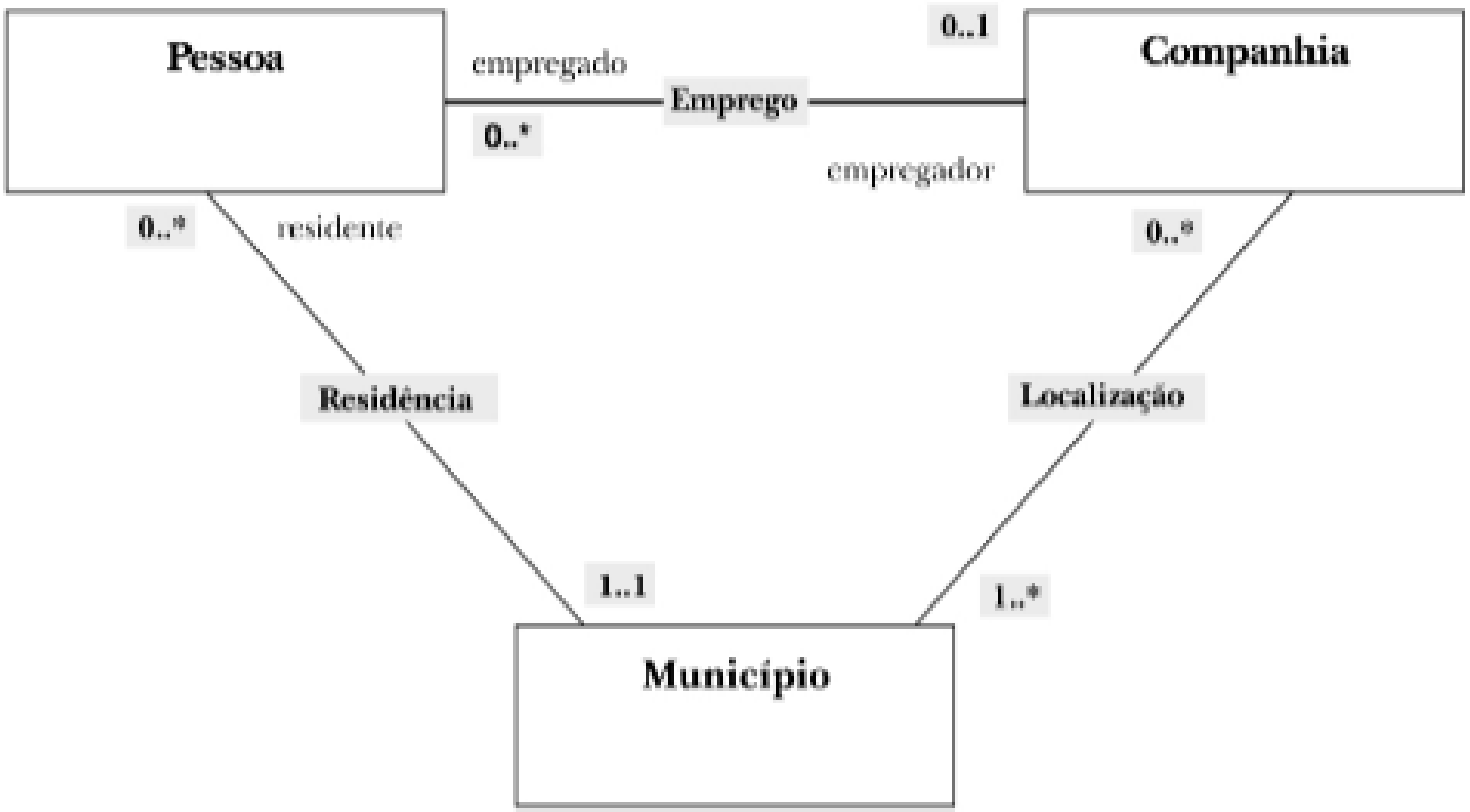
Associação

- Uma associação na UML representa uma população variada de vínculos (links) de relacionamento entre instâncias de classe. (Page-Jones, p.116).
- Pessoa e livro são os objetos e tomou emprestado é a **relação**.
- A associação pode ser dividida em agregação e composição.

Pessoa	Tomou emprestado	Livro
Fred	Tomou emprestado	Construção de Iglus para iniciante
Sally	Tomou emprestado	O Frêmito da Contabilidade de Custos
Mary	Tomou emprestado	Mecânica Quântica para Surrealistas
Stan	Tomou emprestado	Três Gerações de Wellingtons no Campo
Stan	Tomou emprestado	Pensamentos Confusos

Fonte: Adapt. Page-Jones, p.116

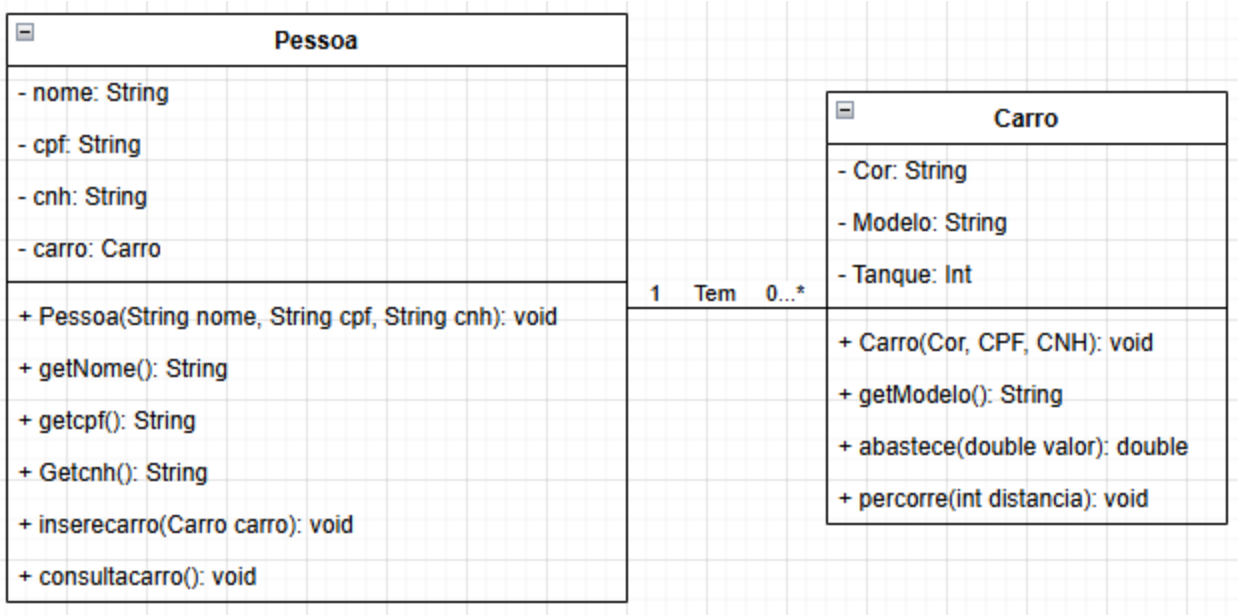
Associação



Fonte: Adapt. Page-Jones, p.118

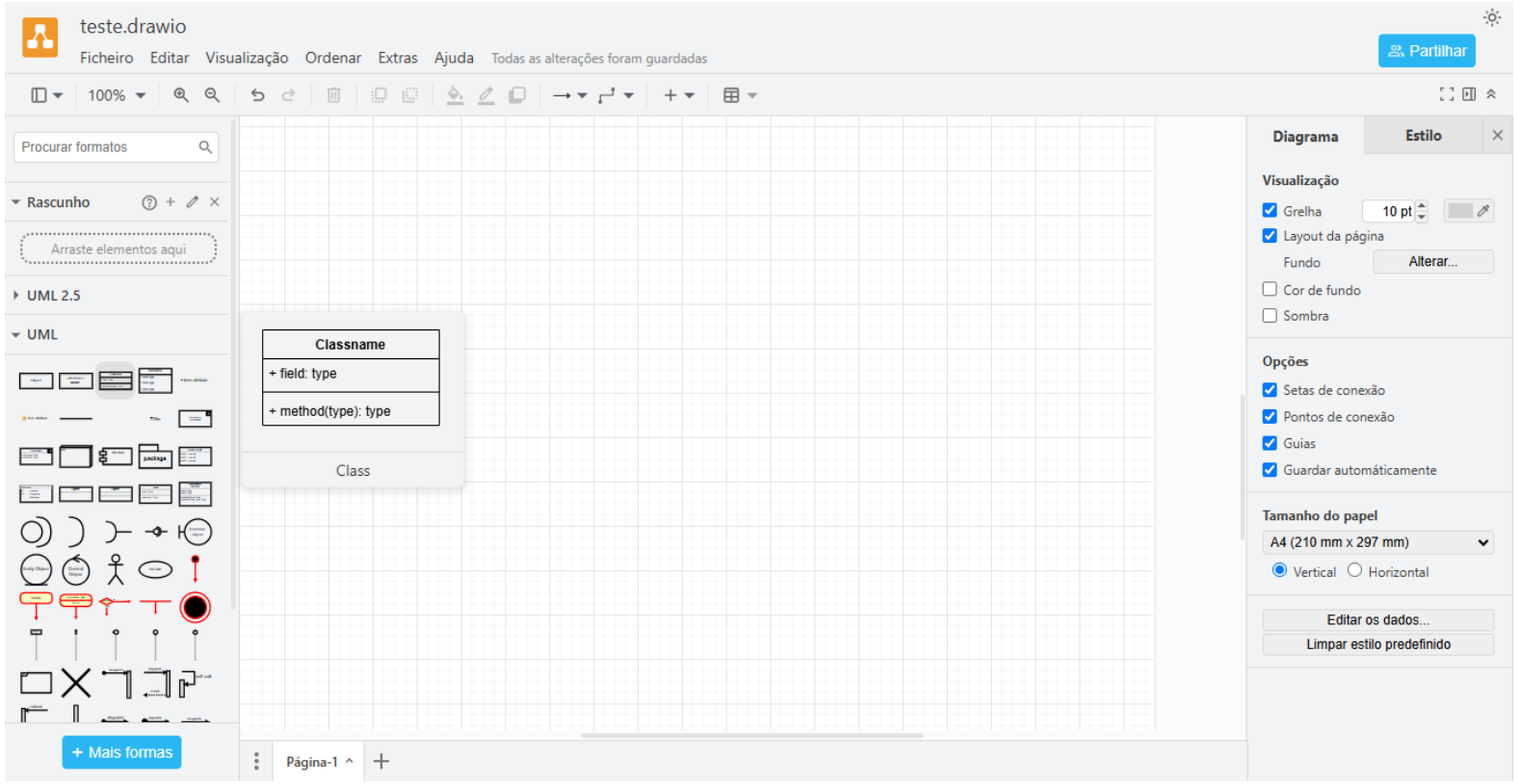
Associação

- Usando o site https://app.diagrams.net/, pode-se fazer os diagramas. Nesse exemplo a classe pessoa tem uma classe carro e cada pessoa pode ter 0 ou mais carros.



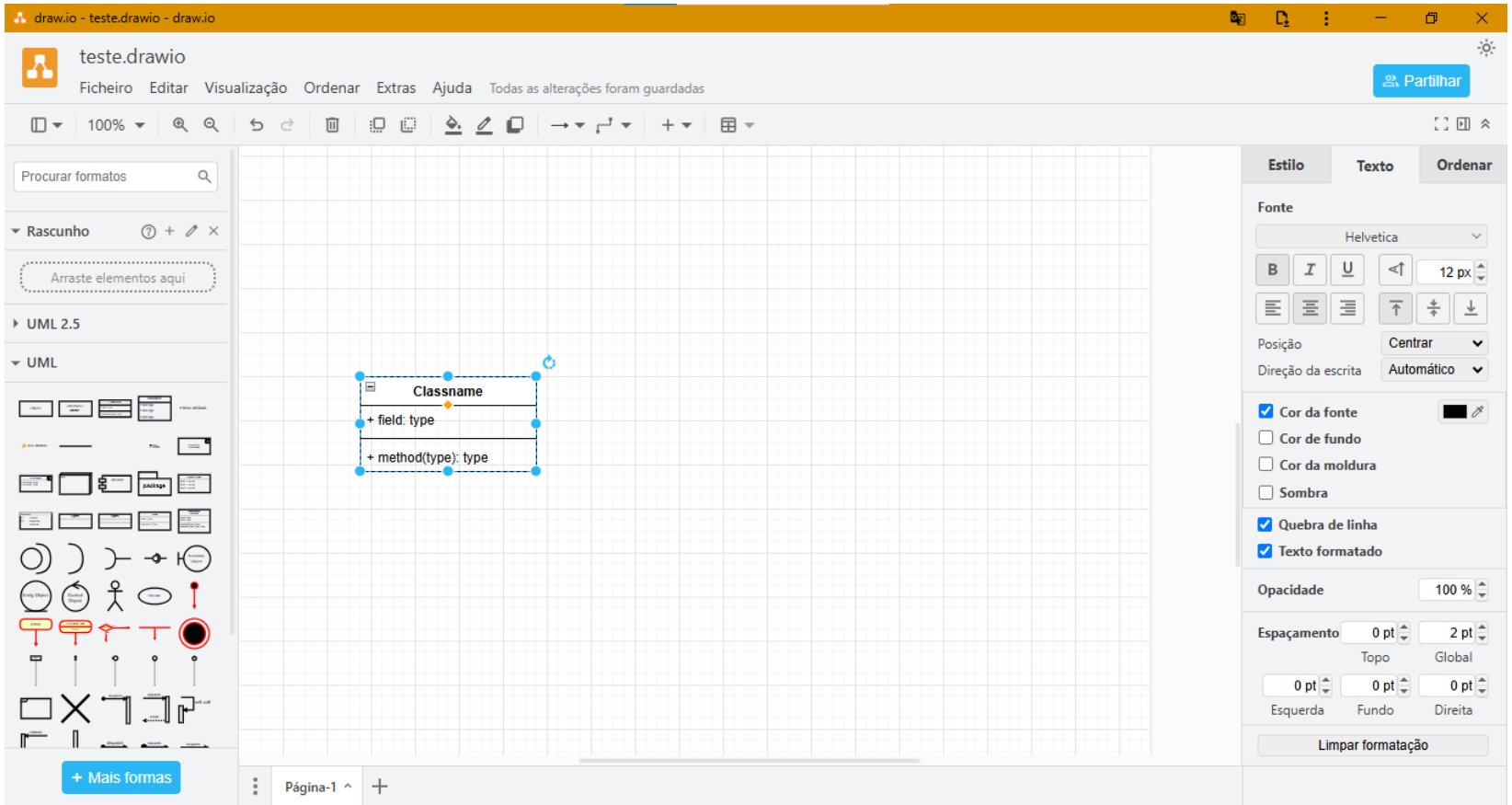
Associação - montando o diagrama

- Ao acessar o site para montar o diagrama, procure UML na esquerda e procure a caixa Class.



Associação - montando o diagrama

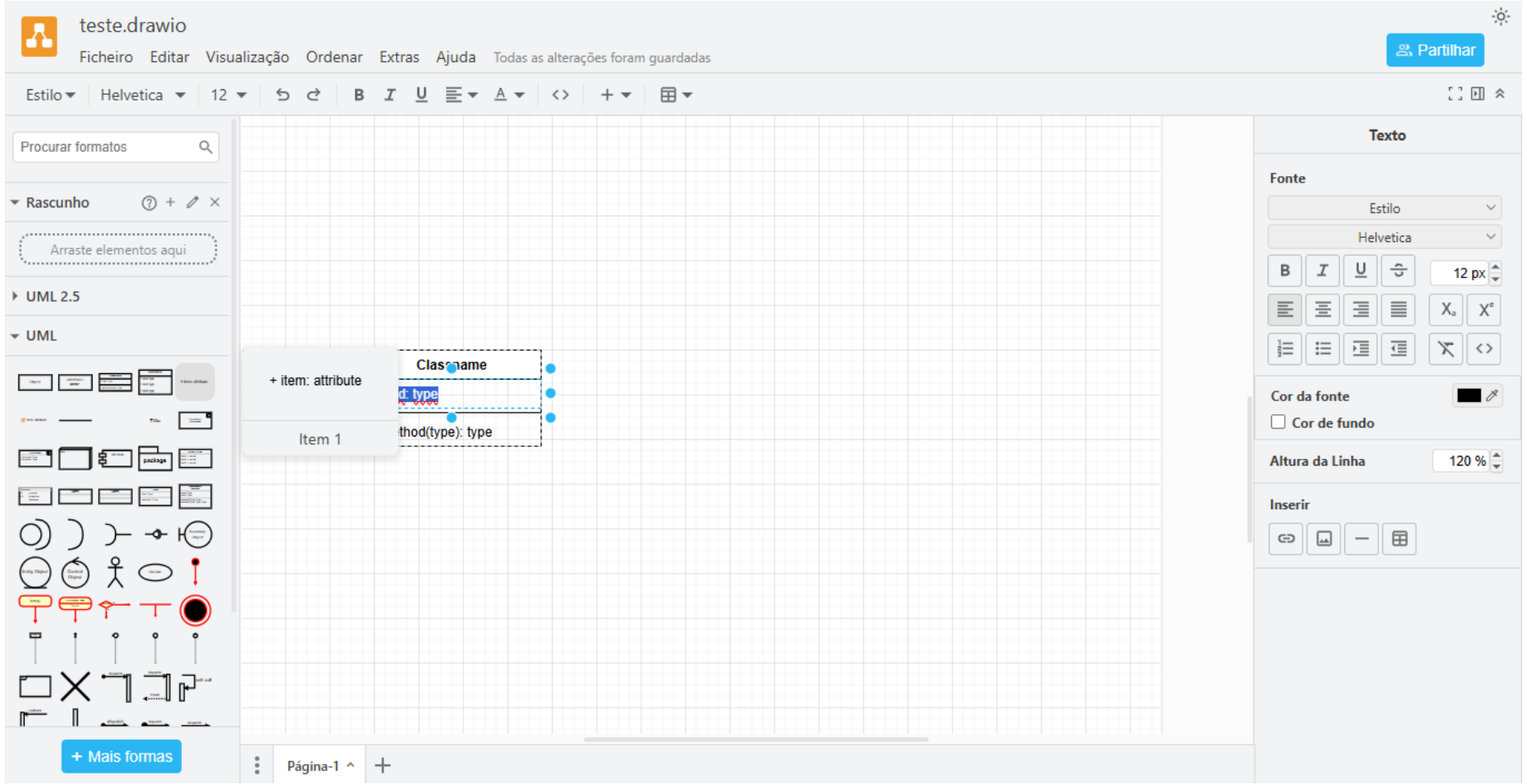
- Insira a sua Classe em qualquer lugar e clique duas vezes em Classname para mudar o nome da classe. Duas vezes em field e method para alterar os nomes dos métodos e campos.



Ao nomear os atributos e métodos usamos + para público e o – para privado.

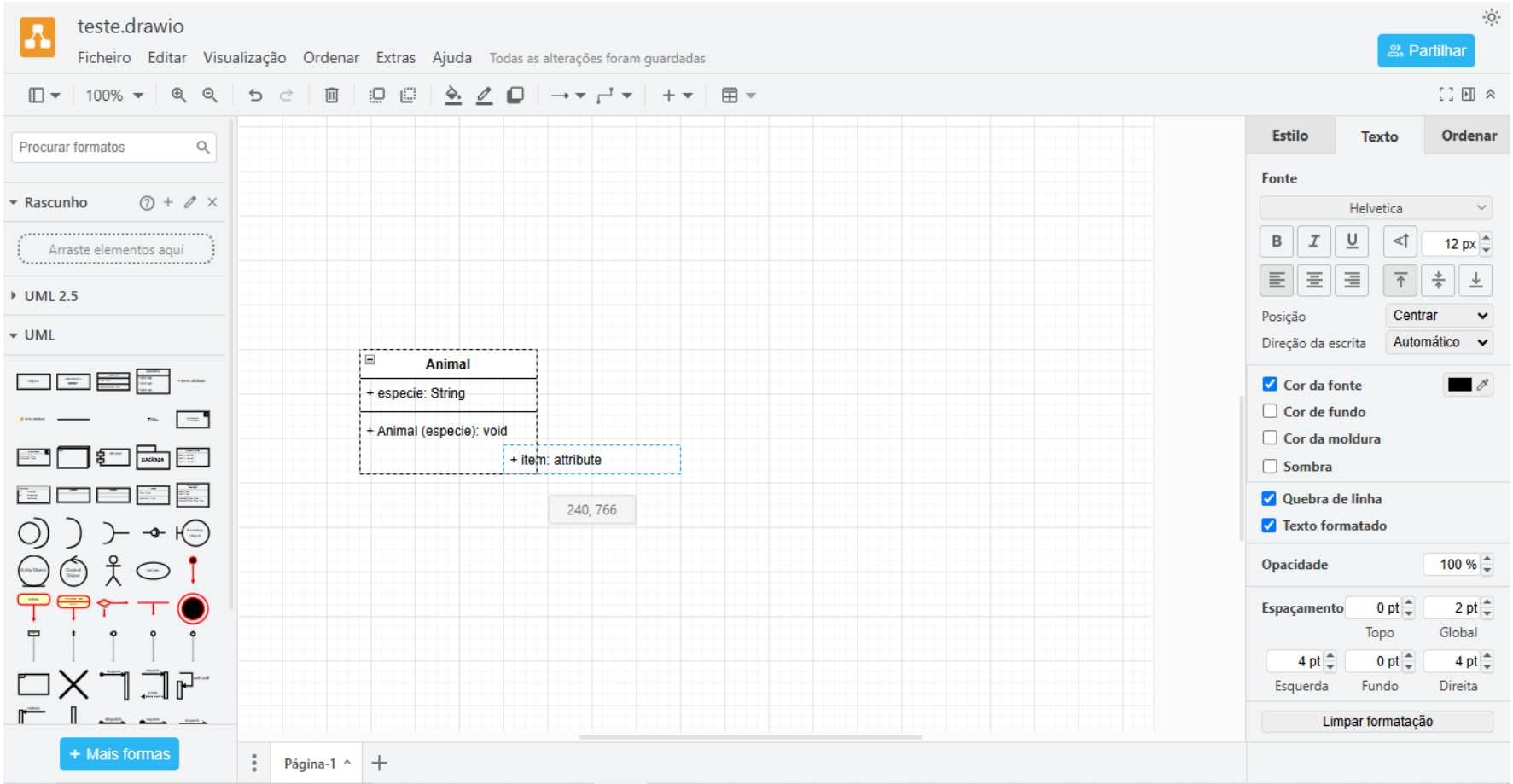
Associação - montando o diagrama

- Para criar um método ou atributo, pode-se criar um item que pode se tornar um atributo ou método.



Associação - montando o diagrama

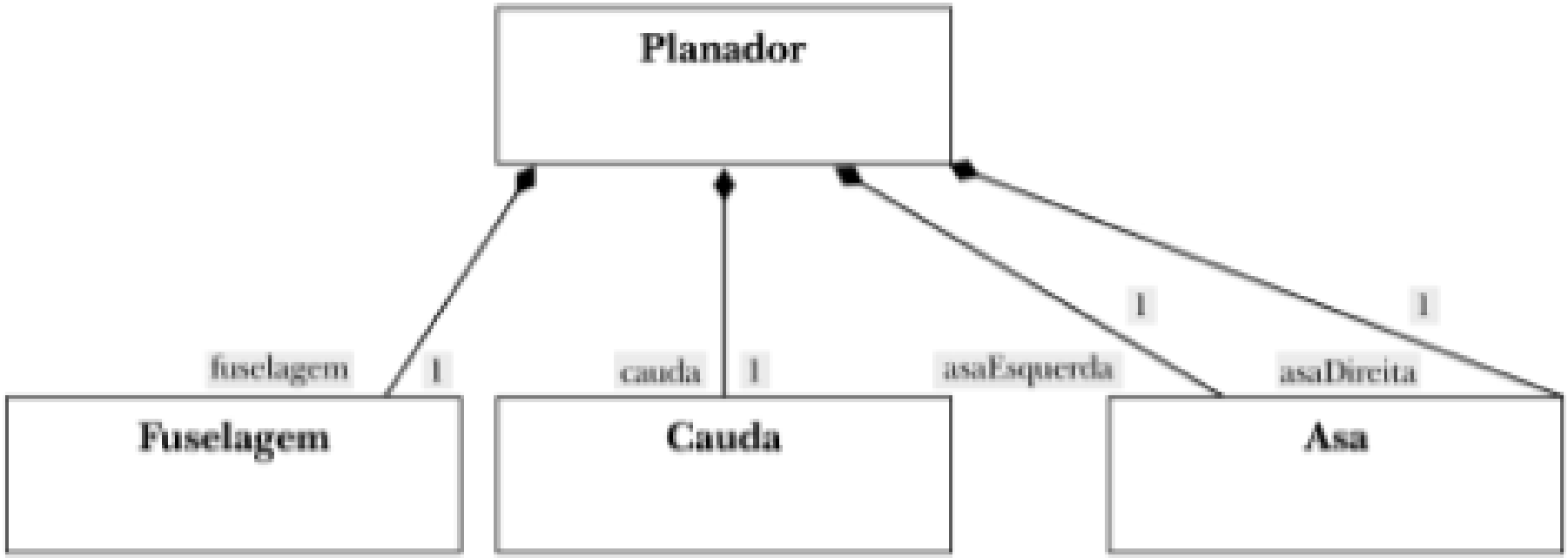
- Ao criar o item, pode se arrastar para dentro da classe junto com atributo ou método..



Composição

- A associação todo/parte é chamada de composição; o todo é chamado de objeto composto e a parte de objeto componente. (Page-Jones, p.125).
- As três características da composição são: (Page-Jones, p.125).
  - O objeto composto não existe sem os seus componentes.
  - A qualquer hora, cada dado objeto componente pode ser parte de somente um objeto composto.
  - A composição é tipicamente heterômera (cuja as partes não são semelhantes).

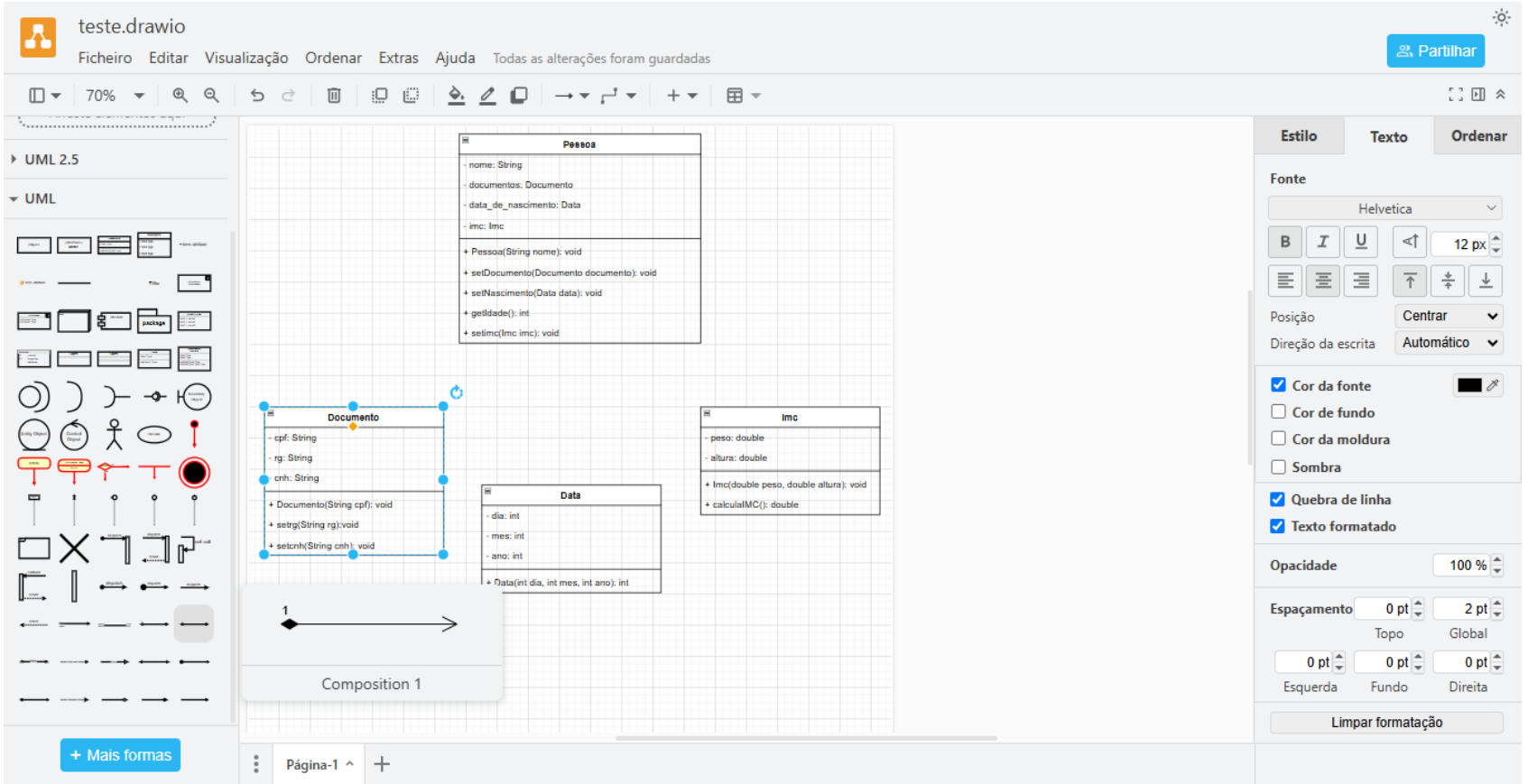
Composição



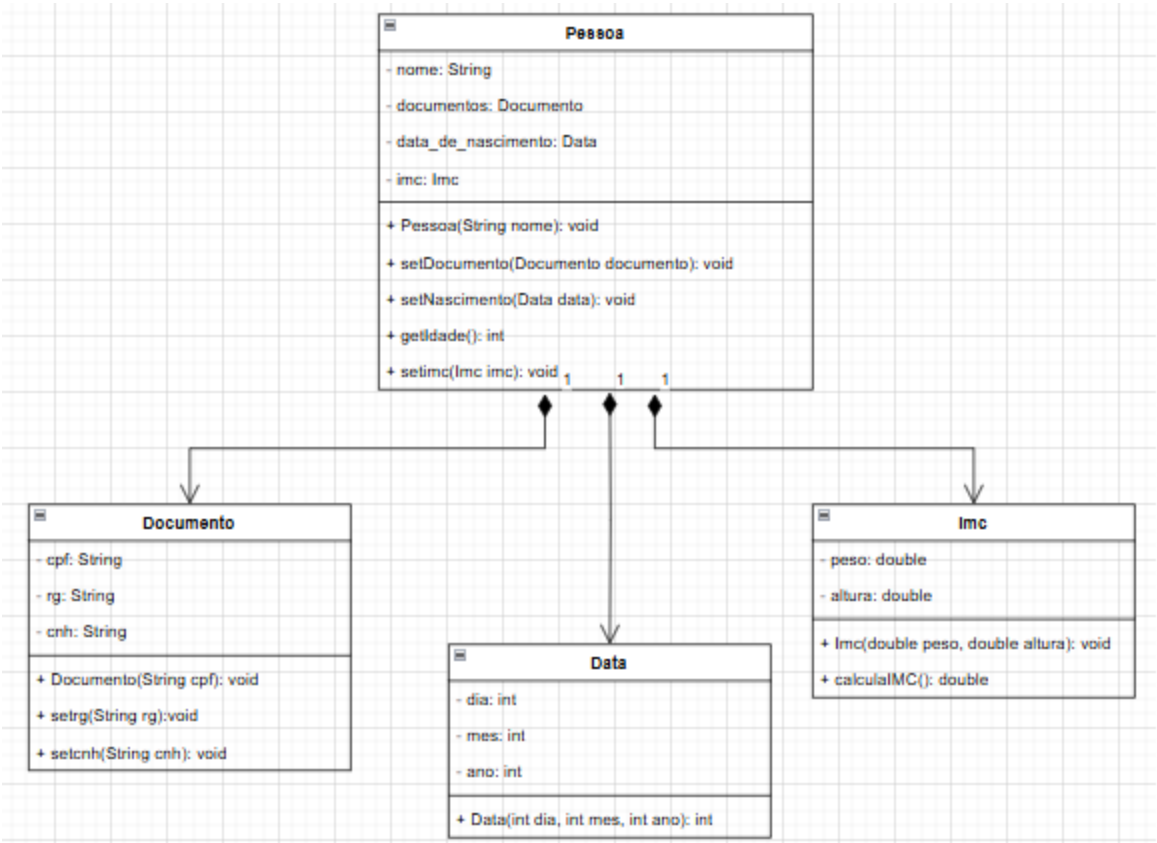
Fonte: Adapt. Page-Jones, p.126

Composição

- Para fazer uma composição no drawio, vá em UML e escolha "composition 1". Com a flecha na tela pode-se manipular livremente.



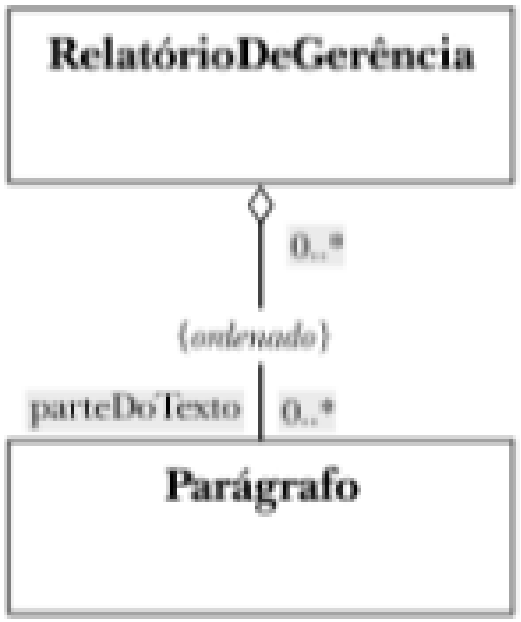
Composição

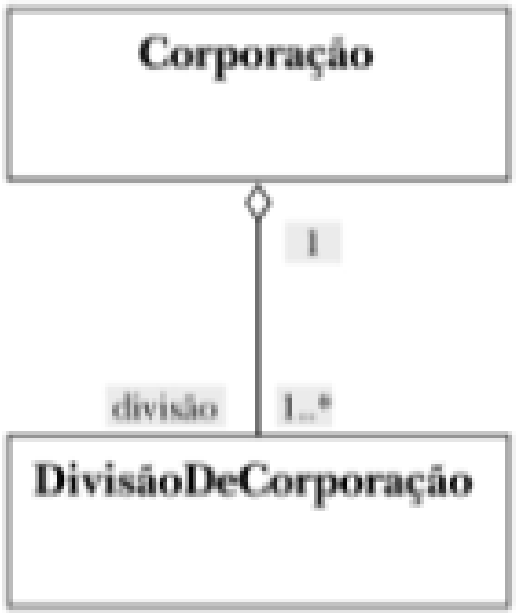

Pessoa
- nome: String
- documentos: Documento
- data_de_nascimento: Data
- imc: Imc
+ Pessoa(String nome): void
+ setDocumento(Documento documento): void
+ setNascimento(Data data): void
+ getIdade(): int
+ setImc(Imc imc): void
1
1
1
Documento
- cpf: String
- rg: String
- cnh: String
+ Documento(String cpf): void
+ setrg(String rg):void
+ setcnh(String cnh): void
Data
- dia: int
- mes: int
- ano: int
+ Data(int dia, int mes, int ano): int
Imc
- peso: double
- altura: double
+ Imc(double peso, double altura): void
+ calculaIMC(): double

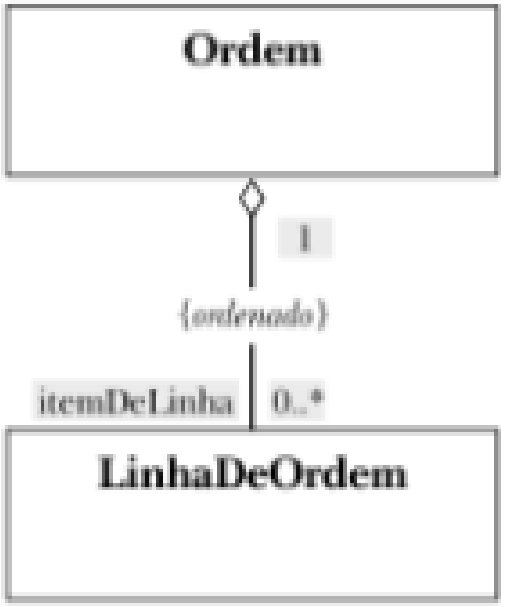
Agregação

- A associação é chamada agregação; o todo é denominado [objeto] agregado; a parte é determinada [objeto] constituinte. (Page-Jones, p.128).
- As três características mais importantes da agregação(Page-Jones, p.128, 129):
  - O objeto agregado pode potencialmente existir sem os seus objetos constituintes.
  - A qualquer hora, cada objeto pode ser um constituinte com mais de um agregado.
  - A agregação tende a ser homeômera, ou seja, cuja parte são todas semelhantes.

Agregação

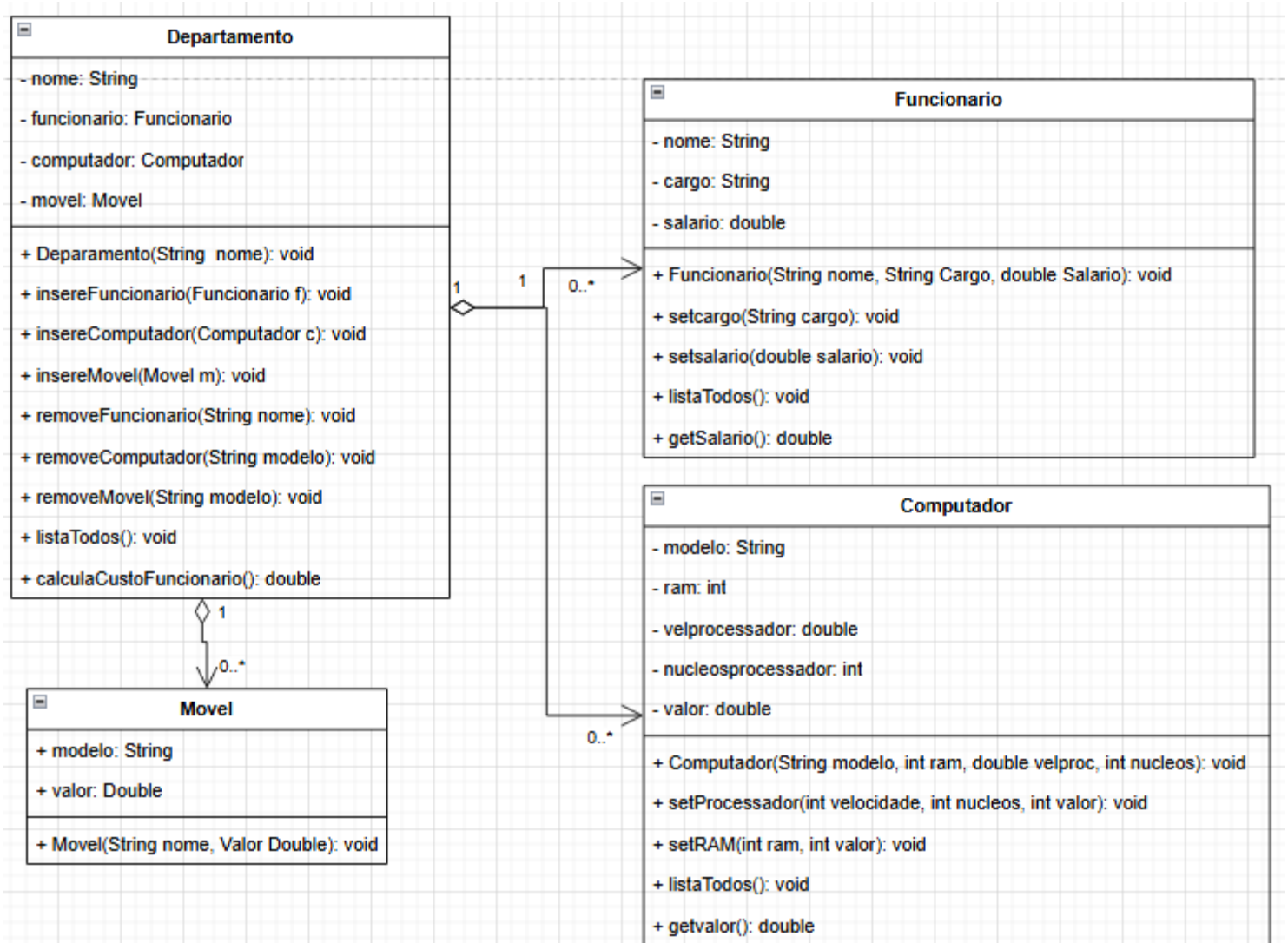




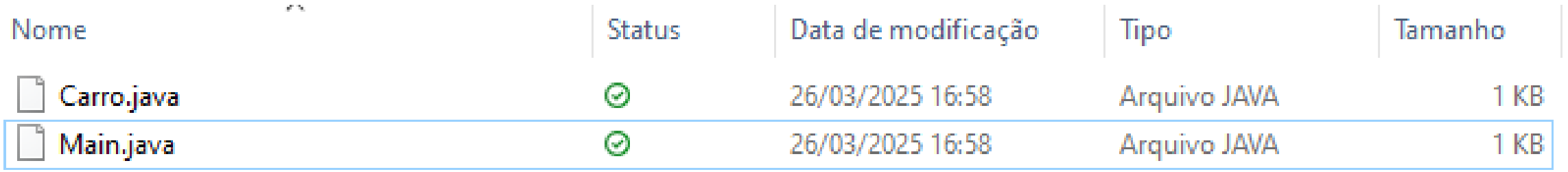


Fonte: Page-Jones, p.129 -131

Agregação


Departamento
- nome: String
- funcionario: Funcionario
- computador: Computador
- movel: Movel
+ Deparamento(String nome): void
+ insereFuncionario(Funcionario f): void
+ insereComputador(Computador c): void
+ insereMovel(Movel m): void
+ removeFuncionario(String nome): void
+ removeComputador(String modelo): void
+ removeMovel(String modelo): void
+ listaTodos(): void
+ calculaCustoFuncionario(): double
1
0..*
Movel
+ modelo: String
+ valor: Double
+ Movel(String nome, Valor Double): void
1
1
0..*
Funcionario
- nome: String
- cargo: String
- salario: double
+ Funcionario(String nome, String Cargo, double Salario): void
+ setcargo(String cargo): void
+ setsalario(double salario): void
+ listaTodos(): void
+ getSalario(): double
0..*
Computador
- modelo: String
- ram: int
- velprocessador: double
- nucleosprocessador: int
- valor: double
+ Computador(String modelo, int ram, double velproc, int nucleos): void
+ setProcessador(int velocidade, int nucleos, int valor): void
+ setRAM(int ram, int valor): void
+ listaTodos(): void
+ getvalor(): double

Trabalhando com várias classes



Nome	Status	Data de modificação	Tipo	Tamanho
Carro.java	✓	26/03/2025 16:58	Arquivo JAVA	1 KB
Main.java	✓	26/03/2025 16:58	Arquivo JAVA	1 KB

- Todas as classes devem estar em arquivos separados.
- O arquivo Carro.java **só e somente** pode ter a classe Carro
- O import da classe Carro vai estar citado no Main.java.

Trabalhando com várias classes

Main.java | Carro.java

```java
public class Carro {
    private double velocidade;
    private double distancia;

    public Carro(double velocidade, double distancia){
        this.velocidade = velocidade;
        this.distancia = distancia;
        System.out.println("Objeto criado");
    }

    public double velocidade_media(){
        double media = this.velocidade/this.distancia;
        return media;
    }

}
```

Código da classe Carro no Carro.Java

Main.java | Carro.java

```java
import Carro;

public class Main {
    static public void main(String args[]){
        Carro novo = new Carro(80, 2);
        double media = novo.velocidade_media();
        System.out.println(media);
    }
}
```

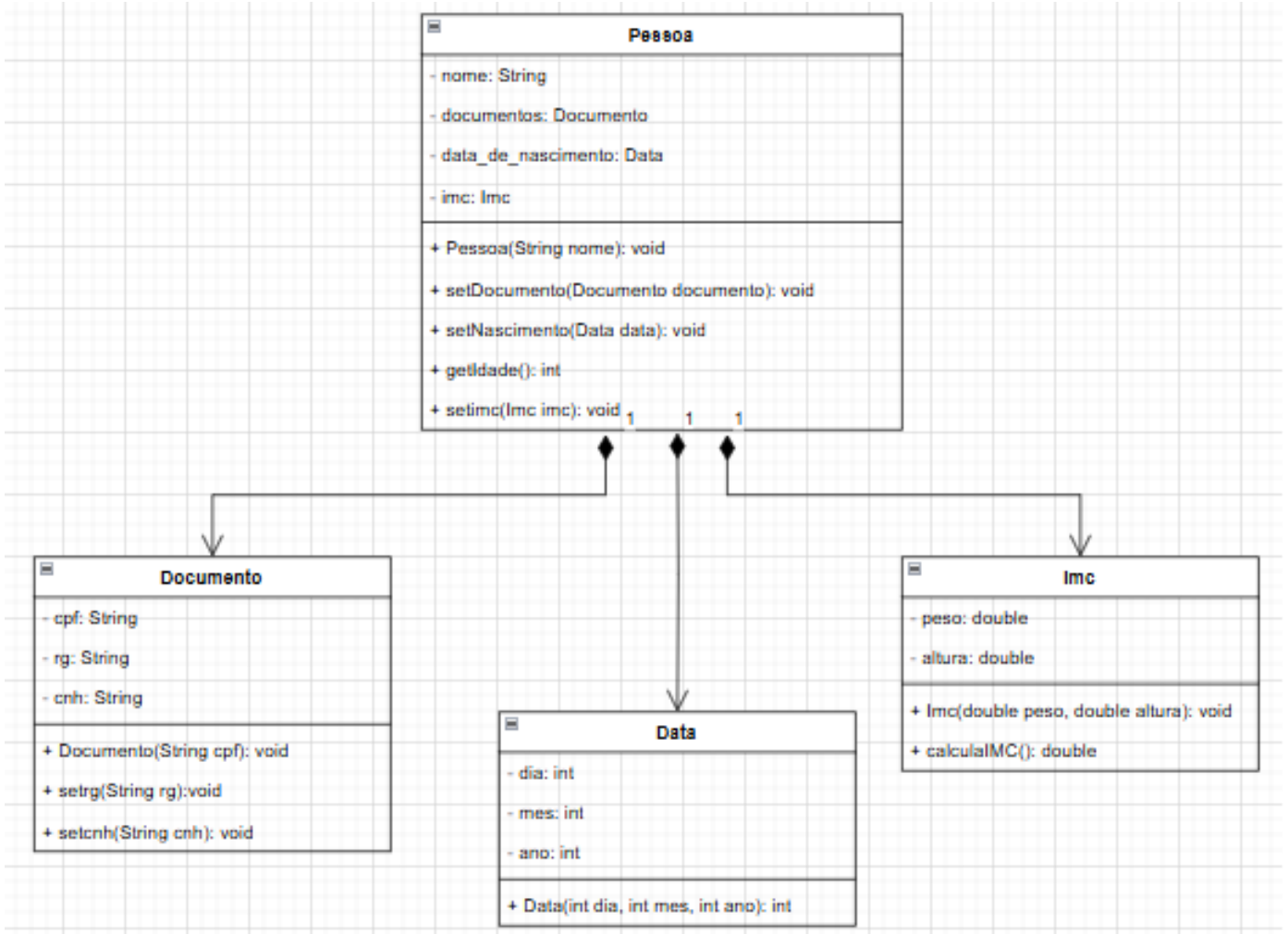
Problems | Javadoc | Declaration | Console

```
<terminated> Main [Java Application] C:\Program Files\Java\jdk1.8.0_2
Objeto criado
40.0
```

Código da classe Main que teve o import da classe Carro

*Evite usar palavras reservadas para nomear classe.

Criando o código com os alunos



Exercícios

- Crie um diagrama UML para um banco onde terá as classes conta, cliente e agência.
- Crie um diagrama UML para interface gráfica onde terá tela, circulo, quadrado e triângulo.
- Use a criatividade e crie um diagrama UML para um caso específico e que seja importante para a sociedade.

Referências

- PAGE-JONES, Meilir. **Fundamentos do desenho orientado a objeto com UML**. São Paulo: Pearson, 2001. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 25 mar. 2025.