



Associação de Objetos

Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro
Prof. Edwar Saliba Júnior
Março de 2025

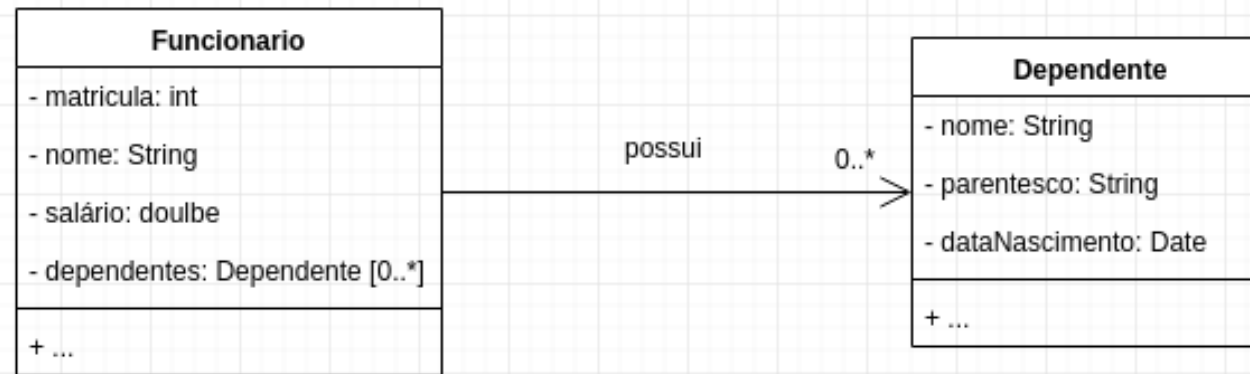


Tipos de Associação

- Uma classe pode possuir um ou mais atributos que são objetos de outras classes;
- Este tipo de relacionamento é chamado de ASSOCIAÇÃO e se subdivide em três subtipos:
 - Associação Simples;
 - Agregação e
 - Composição.



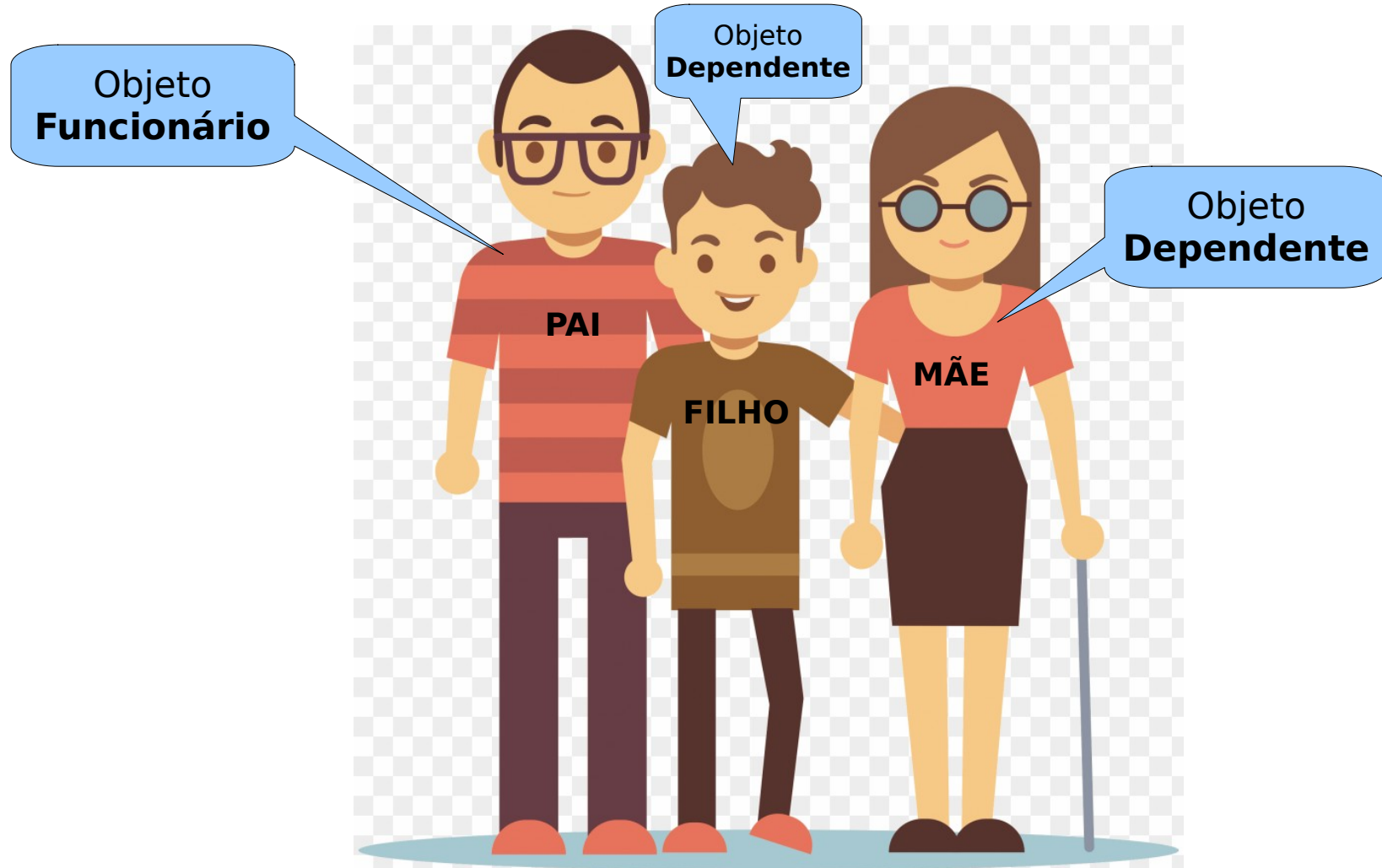
Associação Simples



- Estabelece uma relação simples entre as classes. Ou seja:
 - são 2 objetos independentes um do outro,
 - eles não estabelecem uma relação de todo-parte.
- Representada por um traço entre as classes;
- Exemplo:
 - um funcionário possui dependente(s).



Associação Simples

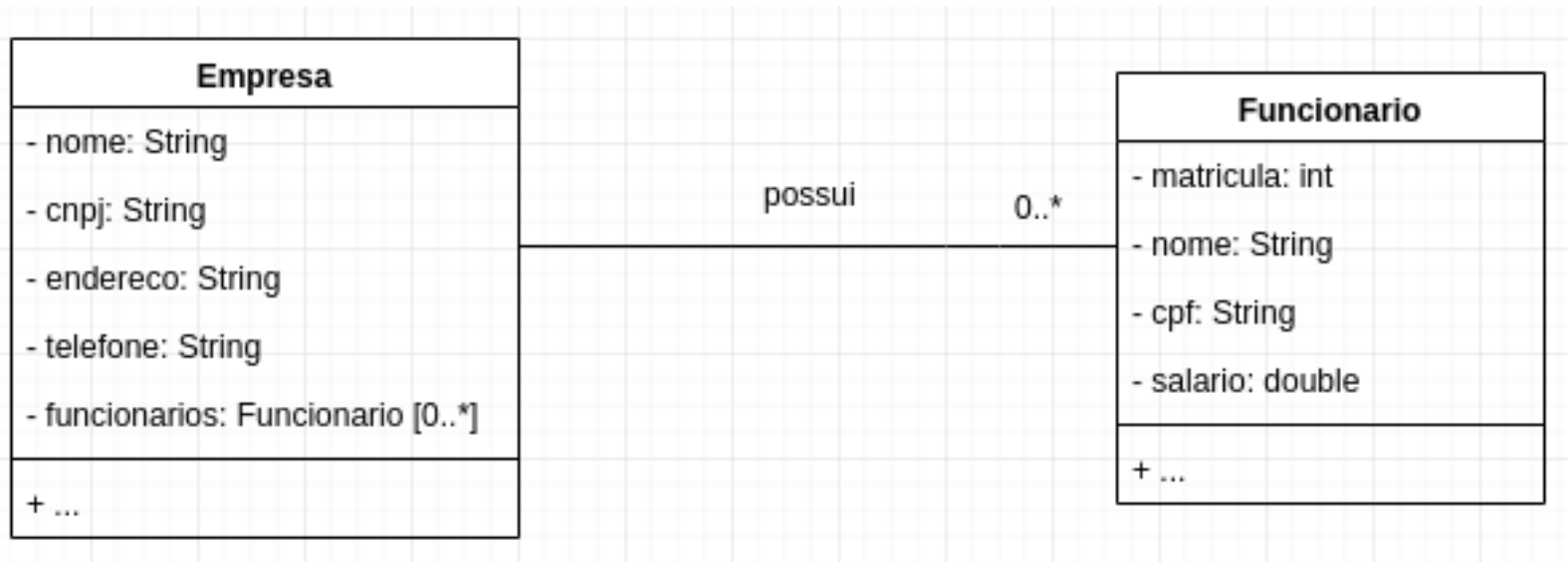


TOPPNG. **desenho pessoa png**. Disponível em: <<https://toppng.com/photo/213554/desenho-pessoa-png-pessoa-desenho>>. Acesso em: 15 jan. 2020.



Associação Simples

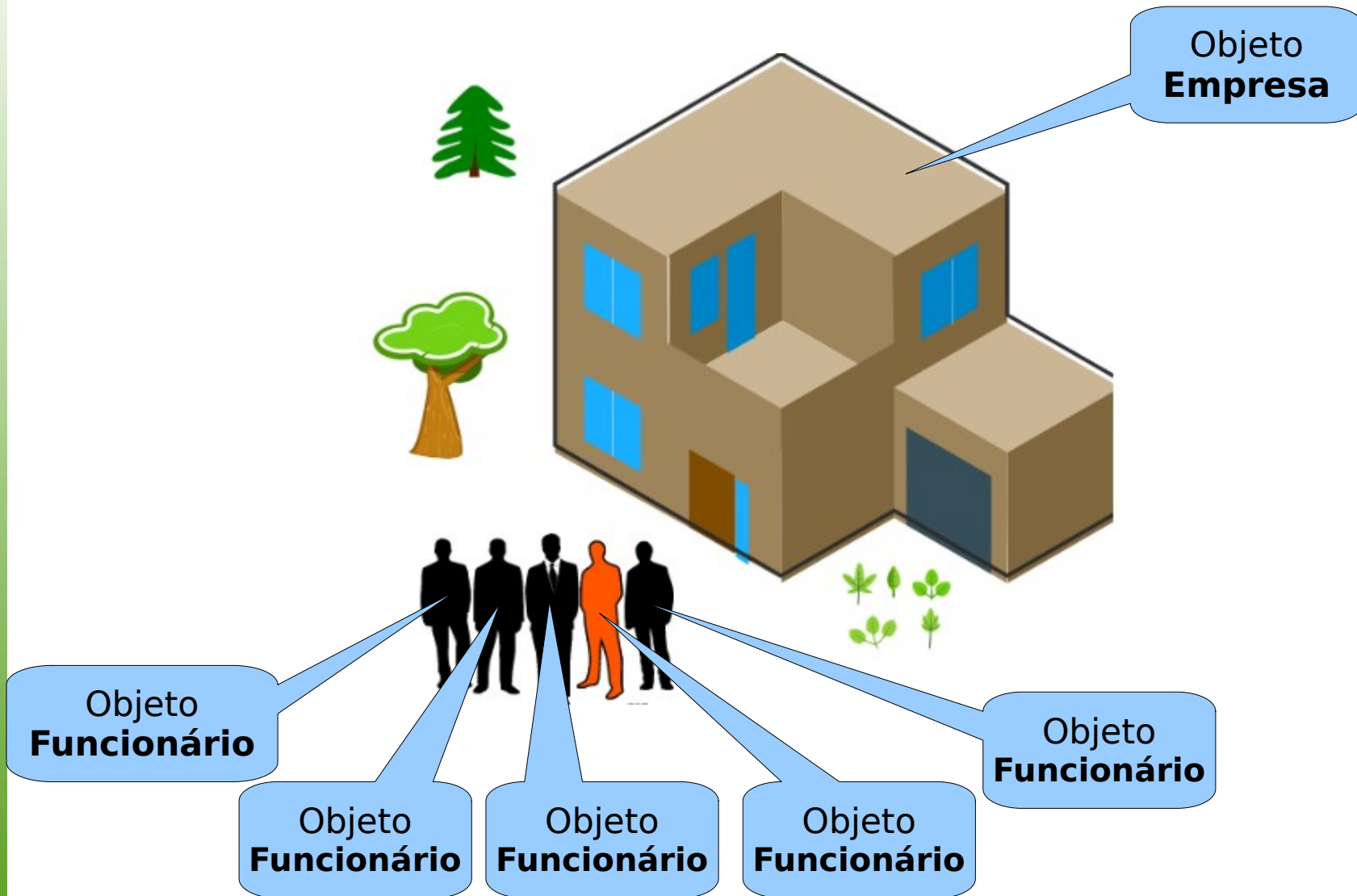
- Outro exemplo:
 - objetos independentes.



Dica: um objeto tem relação com o outro, porém, um não é parte do outro. Ou seja, não é uma agregação e tampouco uma composição.



Associação Simples





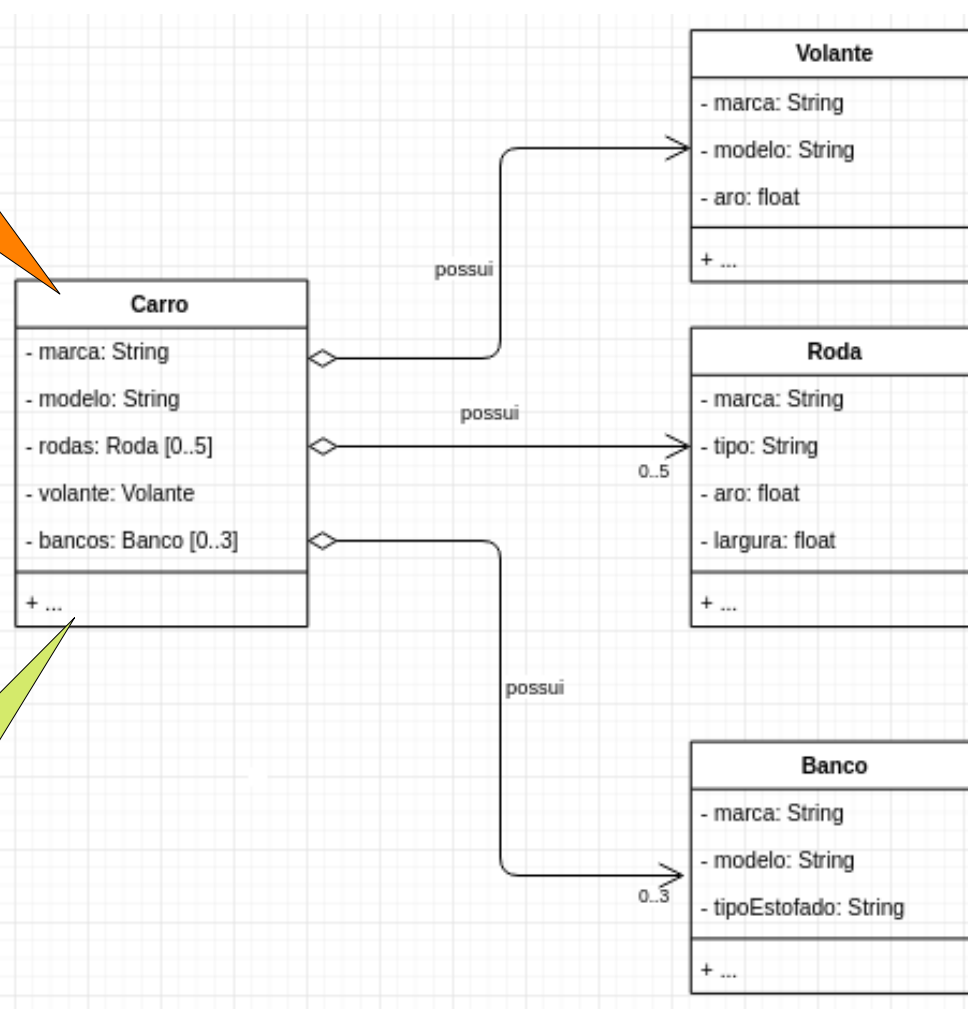
Aggregação

- Estabelece uma relação “todo-parte” entre as classes, sendo que a parte pode existir sem o todo;
- Representada por uma linha com um losango aberto no lado do todo;
- **Exemplo:** Carro e Roda. Roda é parte de um carro, porém ela é um objeto que pode existir sem o carro.



Aggregação - Exemplo

Objeto maior que é formado por diversos outros objetos menores.



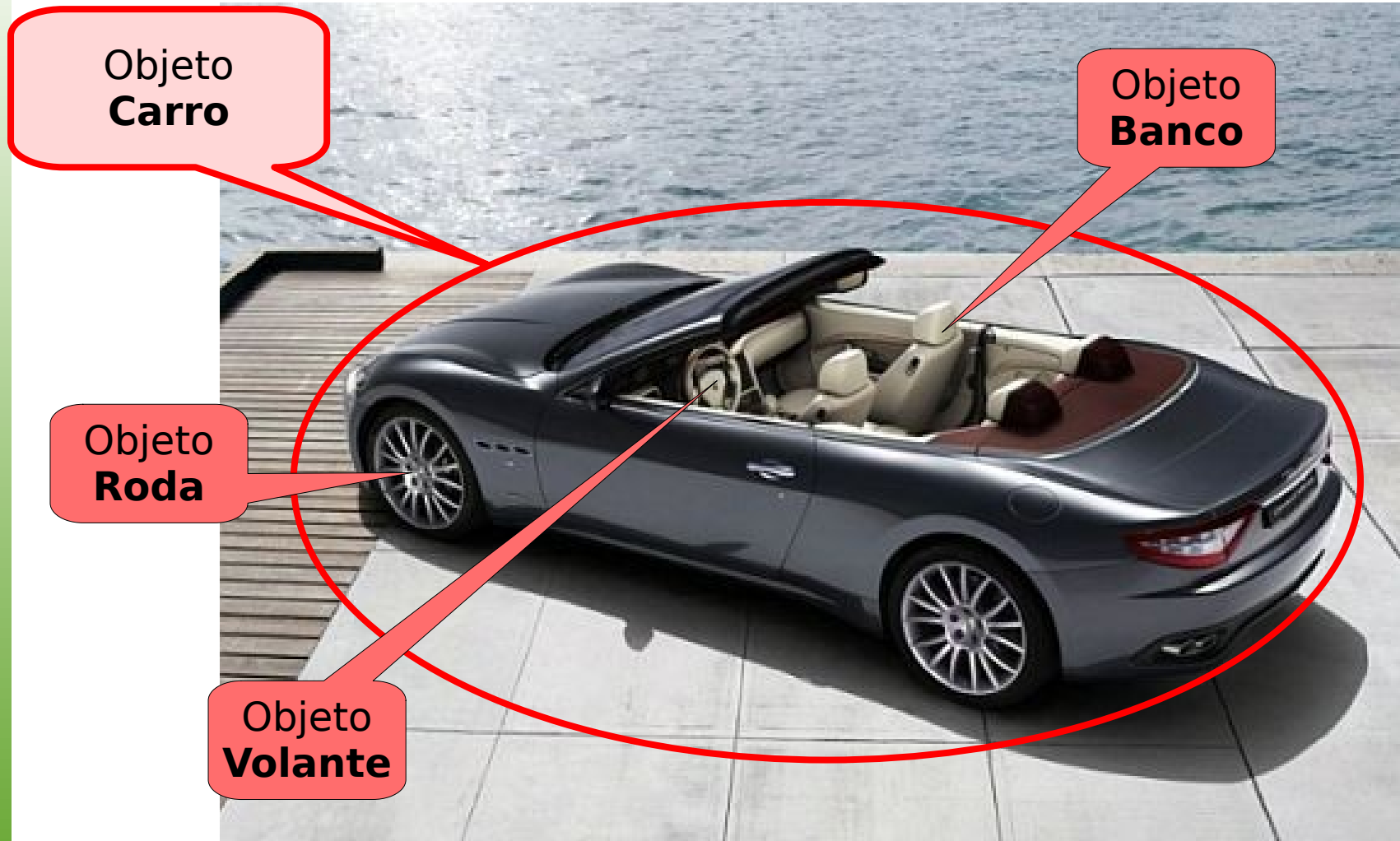
Objetos menores que existem independentemente do objeto maior existir ou não.

Objeto "todo".

Objetos "parte" existem independentemente do "todo" existir.



Agregação



CULTURA MIX. **Mini Roadster**. Disponível em: <<https://autos.culturamix.com/dicas/carros-conversiveis>>. Acesso em: 16 jan. 2020.



Composição

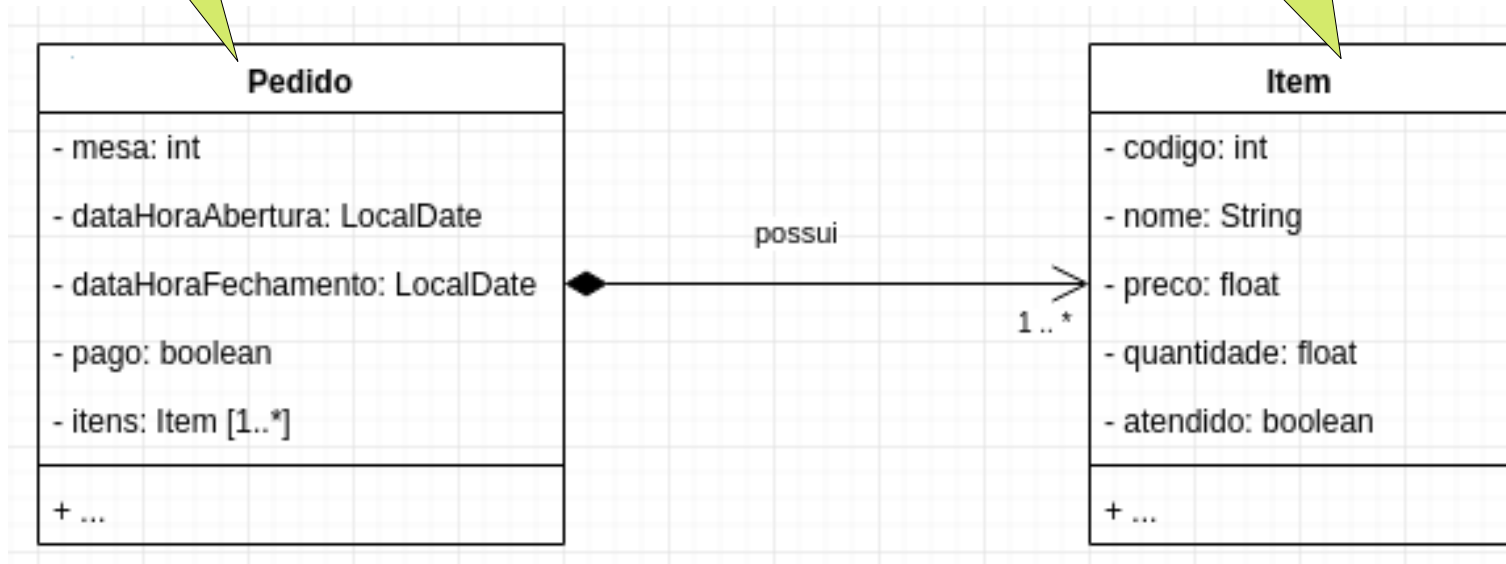
- Estabelece um relação “todo-parte” entre as classes. Sendo que a parte não existe sem o todo;
- Representada por uma linha com um losango cheio do lado do todo;
- **Exemplo:** Pedido e Itens de pedido. Se o pedido for destruído os itens que o compõem também deverão ser.



Composição - Exemplo 1

Objeto "todo".

Objeto "parte" só existe se o "todo" também existir.





Composição - Exemplo 1

Objeto Pedido

Objeto Item

Objeto Item

Objeto Item

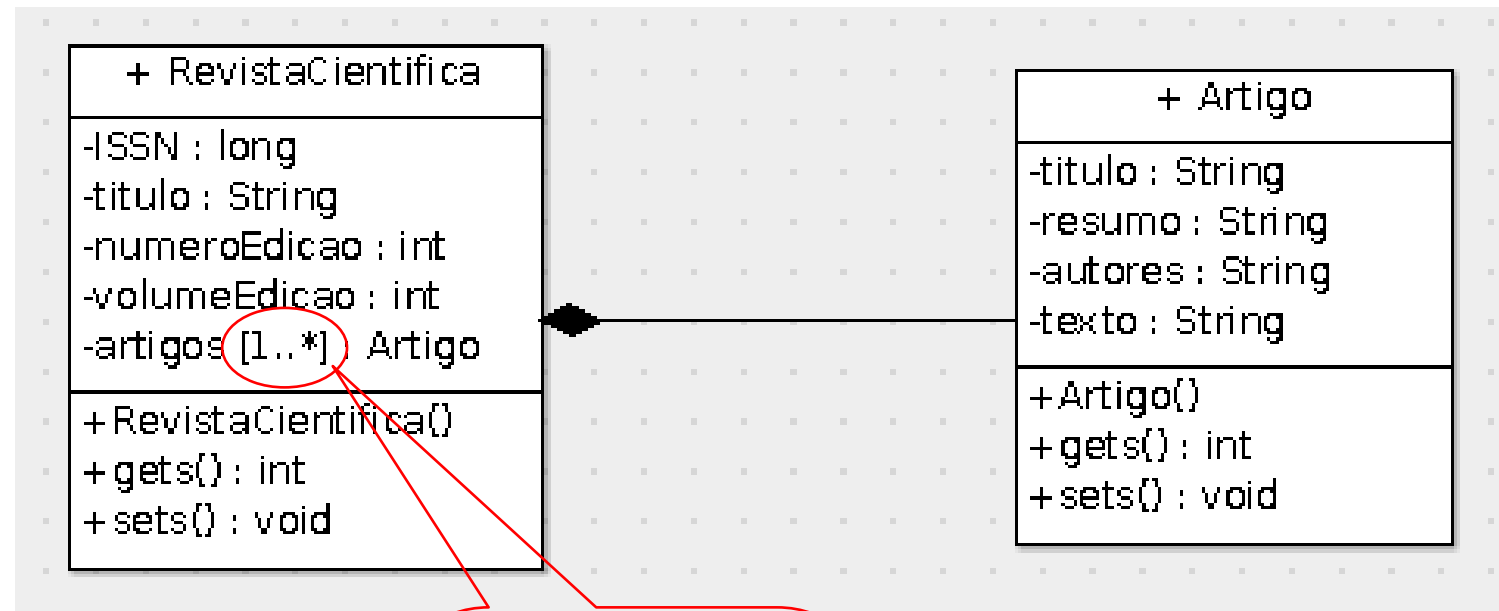
Objeto Item

-
-
-

The diagram illustrates a 'Pedido' (Order) form. It includes a header for 'Sua Empresa' and 'PEDIDO'. Below this is a table with columns for 'QUANT', 'UNID', 'POLPA DE FRUTAS', 'PREÇO UNITÁRIO', and 'TOTAL'. The table contains two columns of items, each with a list of products and their corresponding quantities and prices. Callouts labeled 'Objeto Item' point to various rows in the table, while a callout labeled 'Objeto Pedido' points to the form header.



Composição - Exemplo 2



O software ArgoUML representa a multiplicidade depois do nome do atributo ao invés de depois do tipo do atributo.



Composição - Exemplo 2



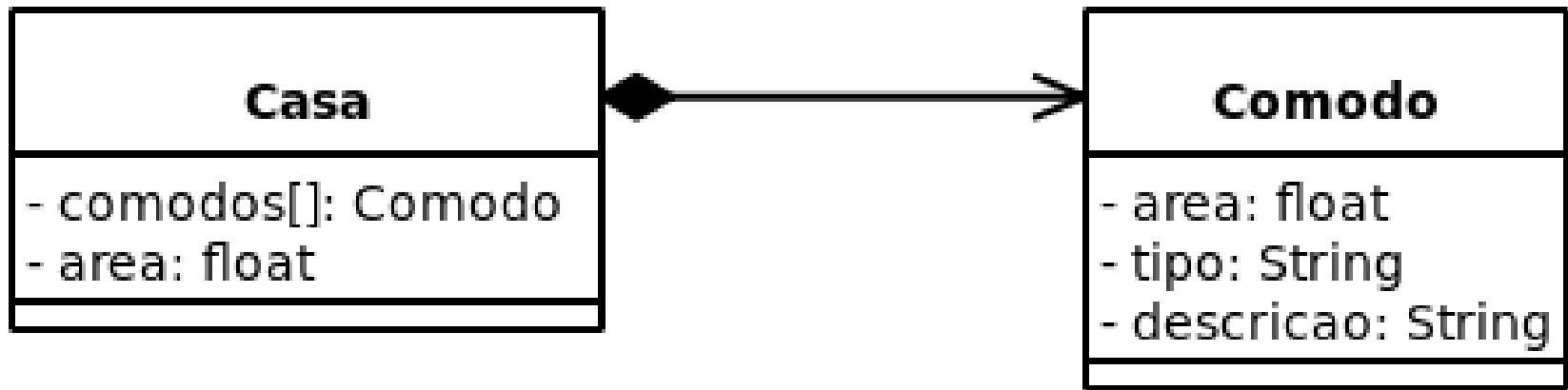
Objeto
RevistaCientífica

Objetos
Artigo
{ estão dentro
da revista }



Composição - Exemplo 3

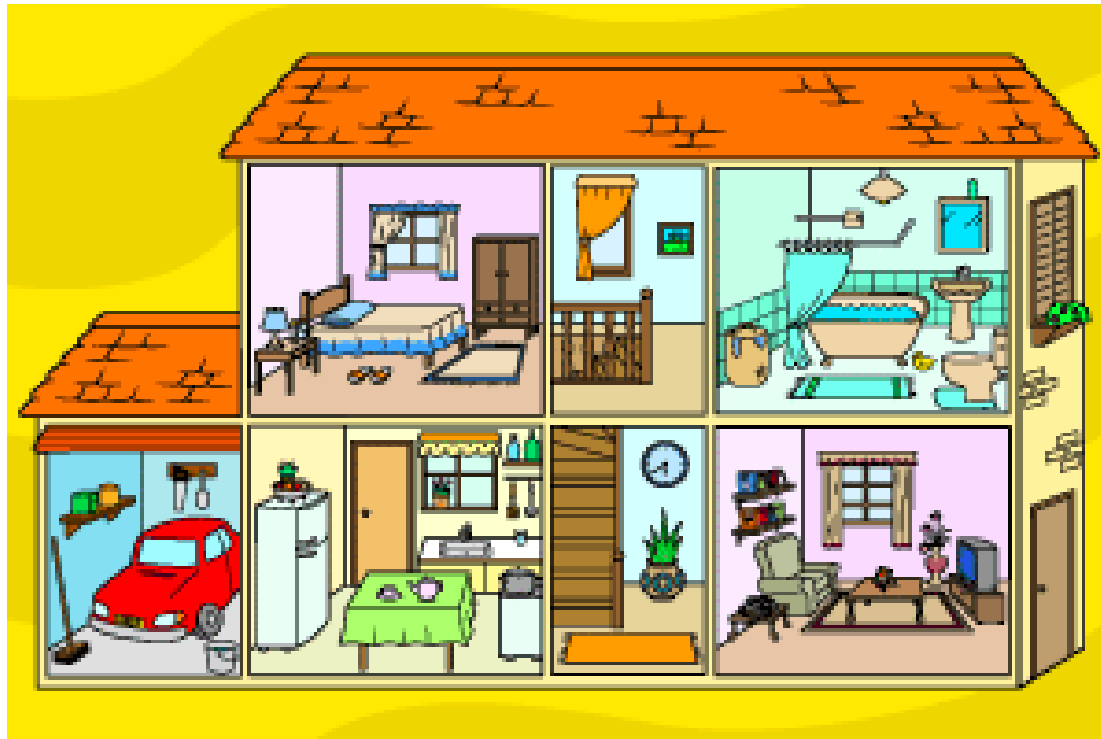
- Casa e cômodos





Composição - Exemplo 3

- Casa e cômodos



Fonte: <<https://www.smartkids.com.br/trabalho/partes-da-casa>>. Acesso em: 3 abr. 2022



Associação ou Relação de Dependência

- A relação de dependência em UML sinaliza uma relação de "uso" entre classes;
- uma classe precisa da outra para realizar alguma tarefa específica, mas não mantém uma ligação estrutural contínua com esta outra classe.

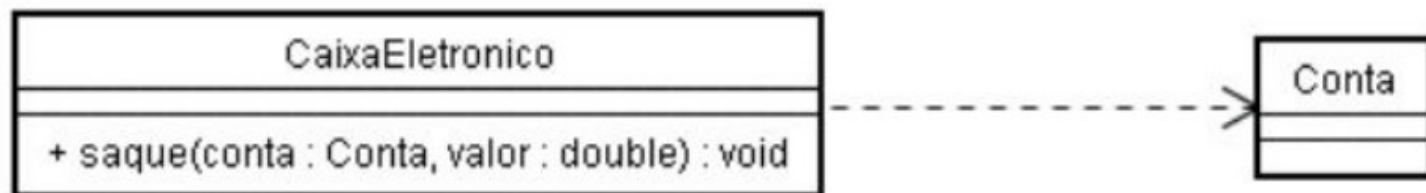
Associação ou Relação de Dependência

- O que define uma relação de dependência em UML são as seguintes características:
 - **uso temporário:** a classe dependente utiliza a classe da qual depende em algum momento durante sua execução, mas essa utilização geralmente é transitória;
 - **parâmetros, variáveis locais e retornos:** a dependência frequentemente se manifesta quando uma classe:
 - recebe objetos da outra classe como parâmetros de um método;
 - cria instâncias locais da outra classe dentro de um método;
 - utiliza objetos da outra classe como tipo de retorno de um método e
 - acessa métodos estáticos ou atributos estáticos da outra classe;
 - **sem propriedade estrutural:** diferentemente de associações, agregação ou composição, a classe dependente não possui a classe da qual depende como um de seus atributos. Não há uma ligação estrutural definida no nível da classe e
 - **direcional:** a dependência é uma relação direcional, representada por uma seta tracejada que aponta da classe dependente para a classe da qual se depende.

Associação ou Relação de Dependência

Dependência entre Classes

- Representada por uma linha tracejada, com uma seta partindo do cliente para o fornecedor.
- Método saque depende de um objeto da classe Contas

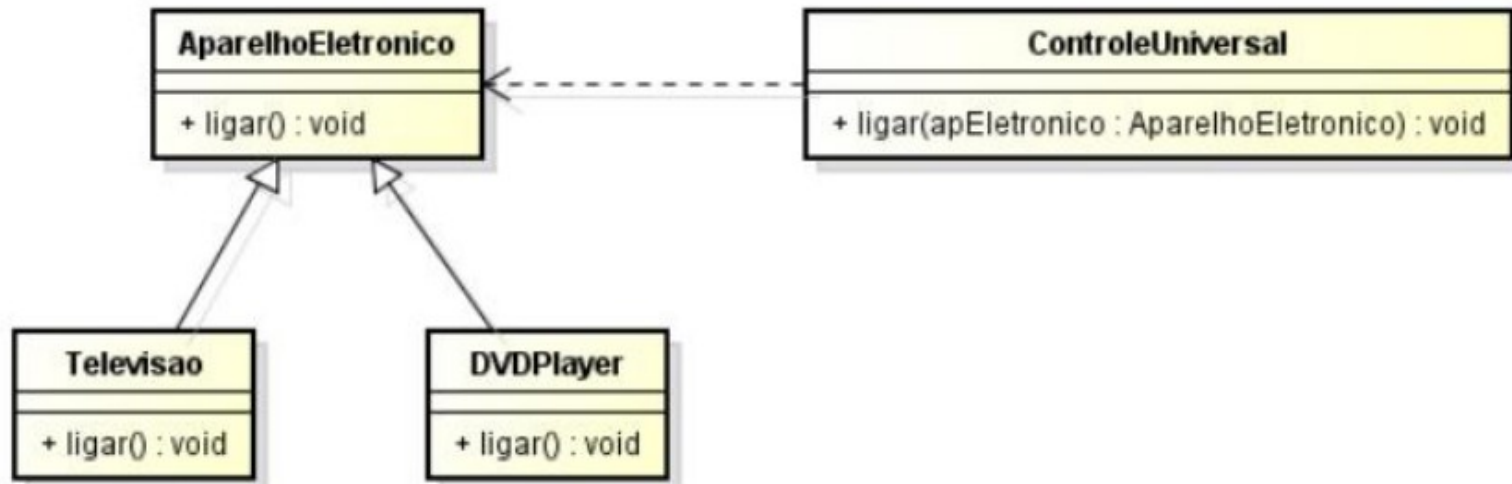


(SILVA, 2025)

Associação ou Relação de Dependência

Dependência entre Classes

- Método ligar da classe ControleUniversal depende de um objeto da classe da classe AparelhoEletronico



(SILVA, 2025)



Aggregação X Composição

- A linha que separa o conceito de agregação do de composição é tênue e muitos autores se embarassam na hora de explicá-la;
- em resumo:
 - composição seria um laço forte de dependência de um objeto (todo) com sua(s) parte(s) e
 - no caso de uma eventual destruição do objeto todo, as partes também serão destruídas;
 - já a agregação seria um laço fraco de dependência entre todo e parte(s) e
 - no caso de uma eventual destruição do objeto todo, as partes não precisam ser destruídas.
- Observação: uma associação de agregação em um *software* poderá ser representada como composição em outro, tudo dependerá do contexto.



Exemplo de Agregação Código-fonte



Classe Ponto

```
1 package circulo;
2
3 public class Ponto {
4
5     private float x, y;
6
7     public Ponto(float x, float y) {
8         this.x = x;
9         this.y = y;
10    }
11
12    public void alterarX(float x) {
13        this.x = x;
14    }
15
16    public void alterarY(float y) {
17        this.y = y;
18    }
19
20    public float obterX() {
21        return x;
22    }
23
24    public float obterY() {
25        return y;
26    }
27 }
28
```



Classe Círculo

```
1 package circulo;
2
3 public class Circulo {
4
5     private float raio;
6     private Ponto centro;
7
8     public Circulo(float x, float y, float r) {
9         raio = r;
10        centro = new Ponto(x, y);
11    }
12
13    public void alterarCentro(float x, float y) {
14        centro.alterarX(x);
15        centro.alterarY(y);
16    }
17
18    public void alterarRaio(float r) {
19        raio = r;
20    }
21
22    public float obterCentroX() {
23        return centro.obterX();
24    }
25
26    public float obterCentroY() {
27        return centro.obterY();
28    }
29
30    public float obterRaio() {
31        return raio;
32    }
33 }
34
```




Classe Principal

```
1 package circulo;
2
3 import java.util.Scanner;
4
5 public class Main {
6
7     public static void main(String[] args) {
8         float x, y, raio;
9         Circulo circ;
10        Scanner in = new Scanner(System.in);
11
12        System.out.println("Digite as coordenadas do centro: ");
13        System.out.println("Digite a coordenada X: ");
14        x = in.nextFloat();
15        System.out.println("Digite a coordenada Y: ");
16        y = in.nextFloat();
17        System.out.println("Digite o raio: ");
18        raio = in.nextFloat();
19
20        circ = new Circulo(x, y, raio);
21
22        System.out.println("Circulo criado: ");
23        System.out.println("Raio: " + circ.obterRaio());
24        System.out.println("Centro: (" + circ.obterCentroX() + ", "
25            + circ.obterCentroY() + ").");
26        System.out.println("");
27    }
28 }
29
```



Exercício

- Lista de exercícios 02.



Bibliografia

- BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML Guia do Usuário**. Rio de Janeiro: Campus, 2000.
- DEITEL, H. M.; DEITEL, P. J. **Java Como Programar**; tradução Edson Furmankiewicz; revisão técnica Fábio Lucchini. 6a. ed., São Paulo: Pearson, 2005.
- FERREIRA, Kecia Aline Marques. *Slides* da disciplina de Programação de Computadores II. CEFET-MG, 2009.
- KOLB, Juliana. **Composição**. Disponível em: <https://julianakolb.wordpress.com/2014/01/16/composicao/>. Acesso em: 31 Dez. 2019.
- SILVA, Wolley W. **Diagrama de classes: projeto orientado aos objetos**. Fonte: <https://www.slideserve.com/xyla-dotson/diagrama-de-classes>. Acesso em: 28 mar. 2025.