Programação Orientada a Objetos

Aula #07

- Associação entre classes: agregação & composição

Profª Luciene de Oliveira Marin
lucienemarin@utfpr.edu.br

Associação entre classes: agregação & composição

Associação entre classes

O que é?

- Na modelagem orientada a objetos a **associação** entre classes representa um tipo de relacionamento entre objetos destas classes
- Criar uma classe baseada em outras classes. **Como?**
  - usando instâncias de classes base como **campos (atributos)** da nova classe.

Para que serve?

Facilitar a reutilização de código:

- aproveitar classes já escritas, com funcionamento testado e comprovado,
- reutilizar código economiza trabalho do programador e diminui a possibilidade de erros.

Associação entre classes

O que é?

- Na modelagem orientada a objetos a **associação** entre classes representa um tipo de relacionamento entre objetos destas classes
- Criar uma classe baseada em outras classes. **Como?**
  - usando instâncias de classes base como **campos (atributos)** da nova classe.

Para que serve?

Facilitar a reutilização de código:

- aproveitar classes já escritas, com funcionamento testado e comprovado,
- reutilizar código economiza trabalho do programador e diminui a possibilidade de erros.

Associação entre classes

O que é?

- Na modelagem orientada a objetos a **associação** entre classes representa um tipo de relacionamento entre objetos destas classes
- Criar uma classe baseada em outras classes. **Como?**
  - usando instâncias de classes base como **campos (atributos)** da nova classe.

Para que serve?

Facilitar a reutilização de código:

- aproveitar classes já escritas, com funcionamento testado e comprovado,
- reutilizar código economiza trabalho do programador e diminui a possibilidade de erros.

Tipos de associações entre classes

Agregação

- Trata-se de um relacionamento "**é parte de**"
- Quando o objeto contido faz sentido mesmo sem ser parte do objeto que o contém

Tipos de associações entre classes

Agregação

- Trata-se de um relacionamento "**é parte de**"
- Quando o objeto contido faz sentido mesmo sem ser parte do objeto que o contém

Exemplo

A classe **Carro** possui um relacionamento com a classe **Motor**, pois um objeto da classe Carro contém 1 objeto da classe Motor

```
public class Carro{
    private String marca;
    private Motor propulsor;

    public Carro(String m, Motor mo){
        this.marca = m;
        this.propulsor = mo;
    }
}
```

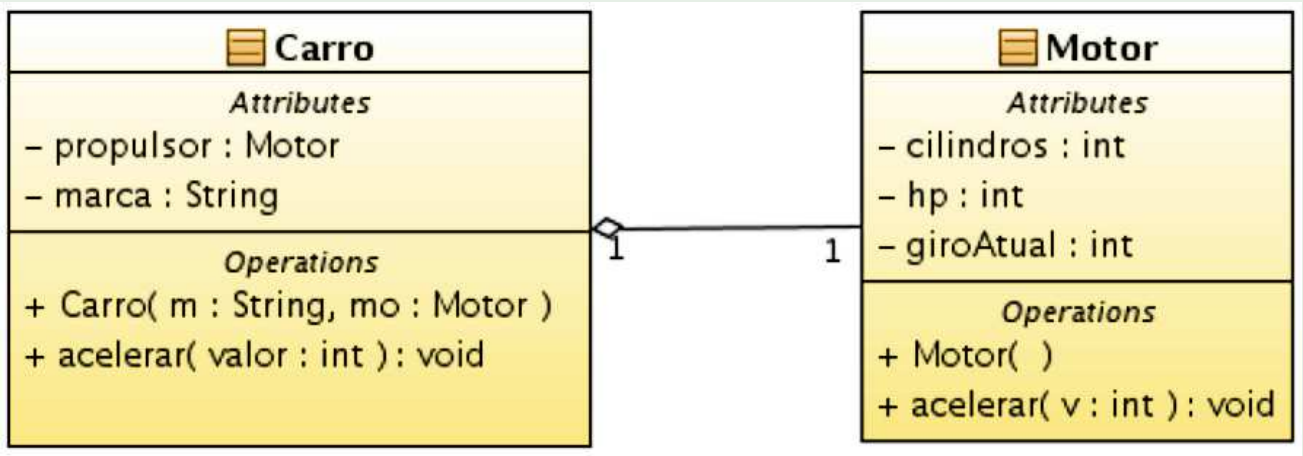
Tipos de associações entre classes

Agregação

- Trata-se de um relacionamento "**é parte de**"
- Quando o objeto contido faz sentido mesmo sem ser parte do objeto que o contém

Exemplo:

Um **carro** possui um **motor**. Se o carro deixar de existir, o motor poderia ser colocado em outro carro



Agregação - exemplo

Classes Carro e Motor

Detalhando a classe Carro, atenção para o método `acelerar`:

```
public class Carro{
    private String marca;
    private Motor propulsor;

    public Carro(String m, Motor mo){
        this.marca = m;
        this.propulsor = mo;
    }

    public void acelerar(int valor){
        this.propulsor.acelerar(valor);
    }
}
```

Tipos de associações entre classes

Composição

- Trata-se de um relacionamento "**faz parte de**" (mais restritivo que a Agregação)
- Quando o objeto contido não faz sentido sem o objeto que o contém

Exemplo:

Um **livro** é composto por diversos **capítulos**. Se destruirmos um **livro**, não faria mais sentido os capítulos existirem

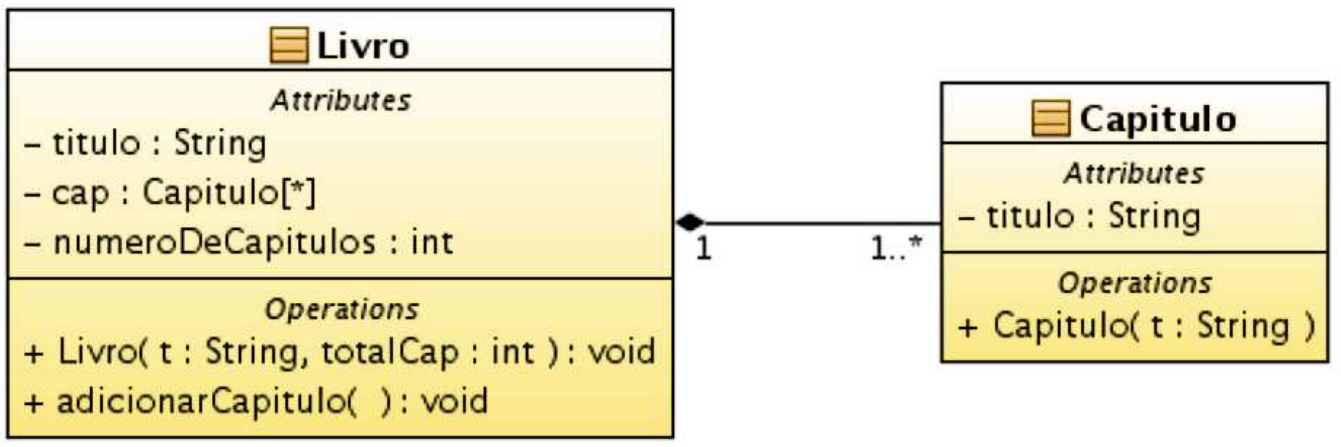
Tipos de associações entre classes

Composição

- Trata-se de um relacionamento "**faz parte de**" (mais restritivo que a Agregação)
- Quando o objeto contido não faz sentido sem o objeto que o contém

Exemplo:

Um **livro** é composto por diversos **capítulos**. Se destruirmos um **livro**, não faria mais sentido os capítulos existirem



Composição - exemplo

Classes Livro e Capítulo

Atenção para os métodos construtor e `adicionarCapitulo`

```
import java.util.Scanner;
public class Livro{
    private String titulo;
    private Capitulo[] cap;
    private int numeroDeCapitulos;

    public Livro{String t, int totalDeCapitulos){
        this.titulo = t;
        this.cap = new Capitulo[totalDeCapitulos];
        this.numeroDeCapitulos = 0;
    }
    public void adicionarCapitulo(){
        Scanner ler = new Scanner(System.in);
        String t = ler.nextLine();
        this.cap[numeroDeCapitulos] = new Capitulo(t);
        numeroDeCapitulos++;
    }
}
```