Departamento de Engenharia de Software
Centro de Educação Superior do Alto Vale do Itajaí - CEAVI
Universidade do Estado de Santa Catarina - UDESC

Programação orientada a objetos
Dependência

Relacionamentos entre classes

- Classes podem se relacionar entre si, definindo um vínculo entre os objetos dessas classes.

Exemplos

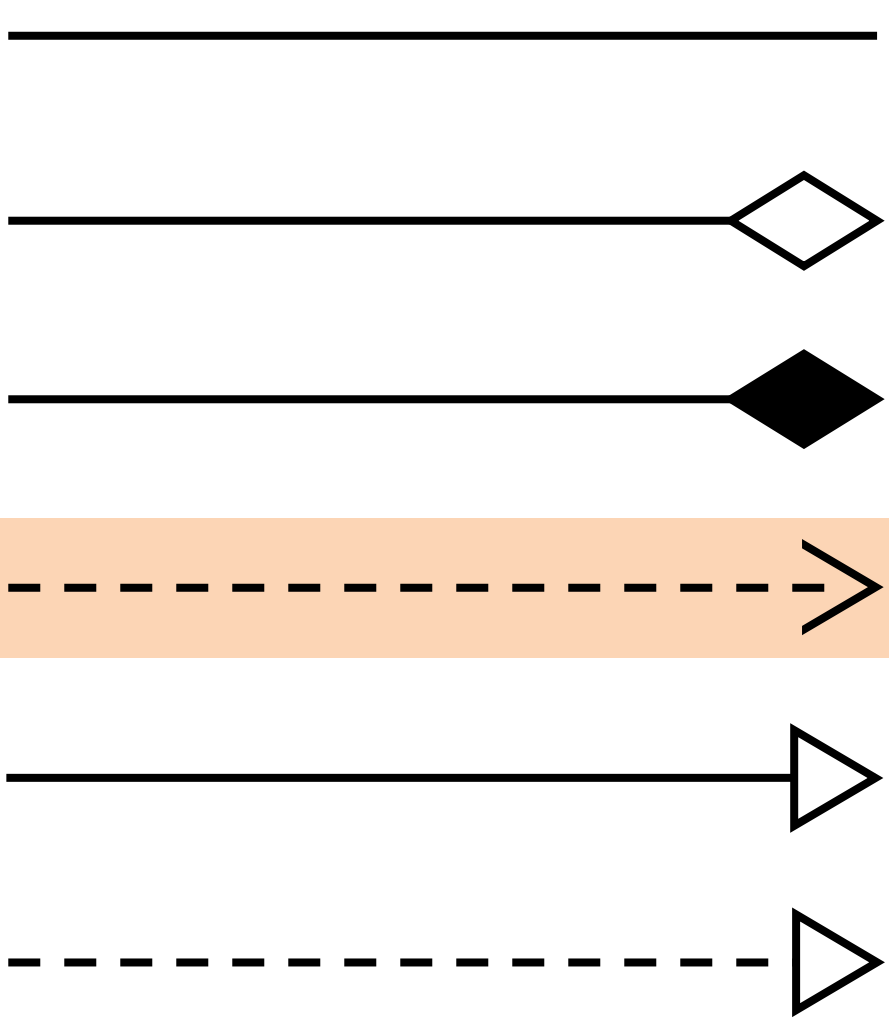
- Um **cliente** possui um **endereço**.
- Uma **empresa** é composta por **funcionários**.
- Uma **moto** é um tipo de **veículo**.
- Um **restaurante** possui **pratos**.
- Uma **correspondência** possui um **remetente** e um **destinatário**.

Relacionamentos entre classes

- **Associação:** conexão entre classes.
- **Agregação e composição:** especialização de uma associação onde um todo é relacionado com suas partes (relacionamento "parte-de").
- **Dependência:** um objeto depende de alguma forma de outro (relacionamento de utilização).
- **Herança (generalização):** um dos princípios da orientação a objetos, permite a reutilização, uma nova classe pode ser definida a partir de outra já existente.
- **Realização:** um contrato que a classe segue (obrigação).

Relacionamentos entre classes

- **Associação:**
- **Agregação**
- **Composição:**
- **Dependência:**
- **Herança (generalização):**
- **Realização:**



Dependência

- Uma classe **A** depende de uma classe **B** quando, no momento da compilação da classe **A**, o código da classe **B** também é compilado. Ou seja, para que a classe **A** funcione, é preciso existir (e funcionar) a classe **B**.
- Logo, classes que possuem entre si quaisquer relacionamentos (associação, agregação, composição, especialização) possuem uma dependência.
  - A dependência é dada pela navegabilidade do relacionamento.
  - **Exemplo:** em uma associação, a classe que possui um objeto da outra, possui uma dependência com a mesma.

Dependência

- Uma classe **A** depende de uma classe **B** quando, no momento da compilação da classe **A**, o código da classe **B** também é compilado. Ou seja, para que a classe **A** funcione, é preciso existir (e funcionar) a classe **B**.
- Logo, classes que possuem entre si quaisquer relacionamentos (associação, agregação, composição, especialização) possuem uma dependência.
  - A dependência é dada pela navegabilidade do relacionamento.
  - **Exemplo:** em uma associação, a classe que possui um objeto da outra, possui uma dependência com a mesma.
- Na orientação a objetos, relacionamentos de diferentes naturezas são representados com diferentes tipos (associação, agregação, composição, etc.).
- Para os casos onde o relacionamento não se encaixa nos tipos predefinidos, o relacionamento é chamado de **dependência**.
- Em geral, uma dependência ocorre quando um objeto da outra classe é utilizado como parâmetro, retorno ou no interior de um método.

Dependência

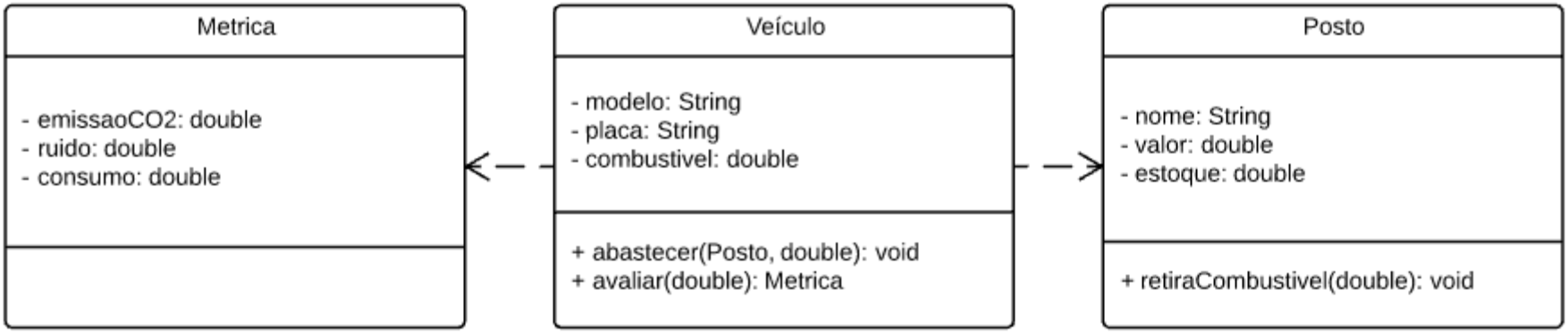
- Uma classe **A** depende de uma classe **B** quando, no momento da compilação da classe **A**, o código da classe **B** também é compilado. Ou seja, para que a classe **A** funcione, é preciso existir (e funcionar) a classe **B**.
- Logo, classes que possuem entre si quaisquer relacionamentos (associação, a

Se a classe independente for excluída, a classe dependente não compilará!

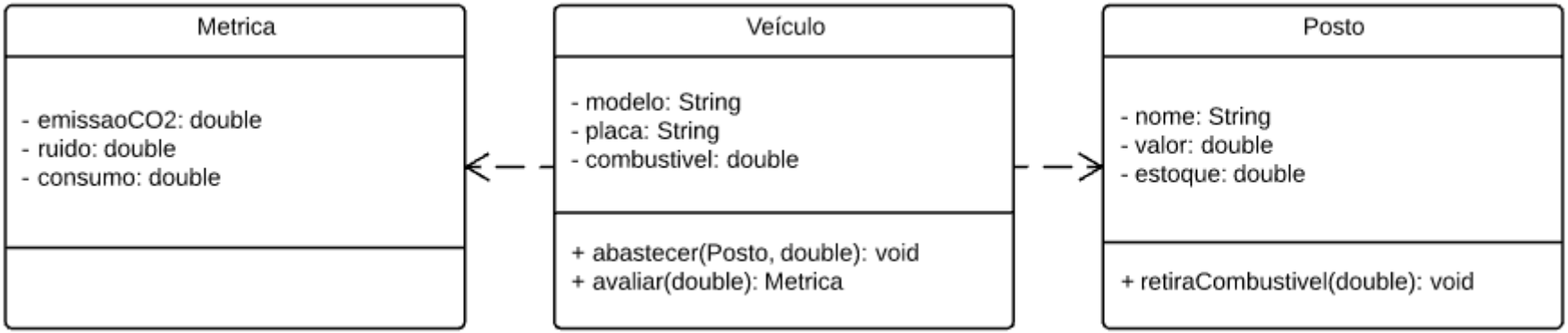
  - **Exemplo:** em uma associação, a classe que possui um objeto da outra, possui uma dependência com a mesma.
- Na orientação a objetos, relacionamentos de diferentes naturezas são representados com diferentes tipos (associação, agregação, composição, etc.).
- **Para os casos onde o relacionamento não se encaixa nos tipos predefinidos, o relacionamento é chamado de dependência.**
- **Em geral, uma dependência ocorre quando um objeto da outra classe é utilizado como parâmetro, retorno ou no interior de um método.**

Exemplo de dependência

- Uma entidade veículo possui um modelo, placa e a quantidade de combustível. O método abastecer recebe como parâmetros um posto de gasolina (modelado pela classe Posto) e uma quantidade de combustível. O método subtrai a quantidade de combustível do posto, inserindo no veículo. O método avaliar mede uma série de parâmetros (emissão de gás carbônico, ruído e consumo), dado um valor de aceleração. Este método devolve um objeto da classe Metrica, que armazena os referidos valores.
- **Perceba que a classe veículo não possui vínculo direto (associação, agregação ou composição) com as demais classes, caracterizando a dependência.**



Implementação – classes independentes

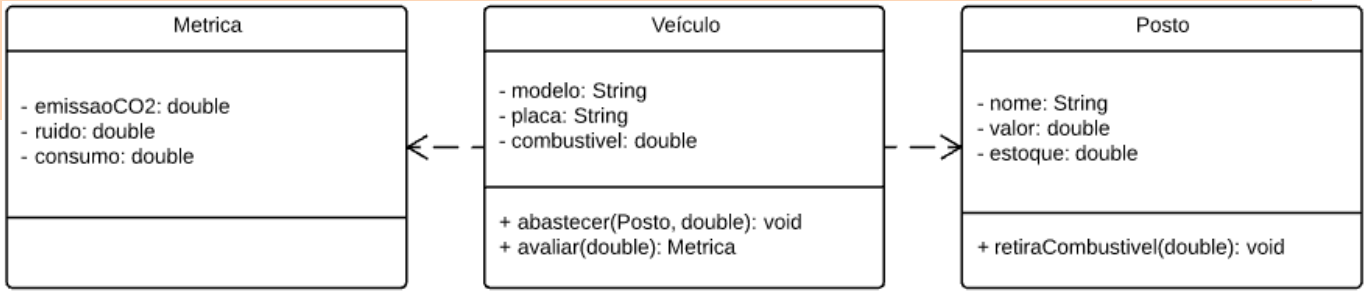



```java
public class Metrica {
    private double emissao;
    private double ruido;
    private double consumo;

    public double getEmissao() {
        return emissao;
    }

    public void setEmissao(double emissao) {
        this.emissao = emissao;
    }

    //demais setters e getters
}
```

```java
public class Posto {
    private String nome;
    private double valor;
    private double estoque;

    public boolean retiraComb(double qtd) {
        if(qtd <= estoque) {
            estoque -= qtd;
            return true;
        }
        return false;
    }

    //setters e getters
}
```

Implementação – classes dependentes

```
public class Veiculo {
    private String modelo;
    private String placa;
    private double combustivel;

    public void abastecer(Posto posto, double qtd) {
        if(posto.retiraCombustivel(qtd))
            this.combustivel += qtd;
    }

    public Metrica avaliar(double aceleracao) {
        Metrica m = new Metrica();
        if(aceleracao <= 10) {
            m.setConsumo(12);
            m.setRuido(41);
            m.setEmissao(340);
        } else {
            m.setConsumo(6);
            m.setRuido(70);
            m.setEmissao(510);
        }
        return m;
    }
}
```



Implementação – exemplo de uso

```
public class ExemploDependencia {
    public static void main(String[] args) {
        Posto p = new Posto();
        p.setNome("Posto XYZ");
        p.setValor(3.50);
        p.setEstoque(4500);

        Veiculo v = new Veiculo();
        v.setModelo("Gol");
        v.setPlaca("ABC-1234");
        v.setCombustivel(12.3);

        System.out.println("Combustível: " + v.getCombustivel());
        v.abastecer(p, 25);
        System.out.println("Combustível: " + v.getCombustivel());

        Metrica m = v.avaliar(22);
        System.out.println("Emissão: " + m.getEmissao() +
                           "\nRuído: " + m.getRuido() +
                           "\nConsumo: " + m.getConsumo());
    }
}
```

```
Combustível: 12.3
Combustível: 37.3
Emissão: 510.0
Ruído: 70.0
Consumo: 6.0
```

Referências

DEITEL, H. M. **Java: como programar**. H. M Deitel e P. J. Deitel - 8a ed. Porto Alegre: Prentice-Hall, 2010.

Leitura complementar

TutorialsPoint Java (http://www.tutorialspoint.com/java).