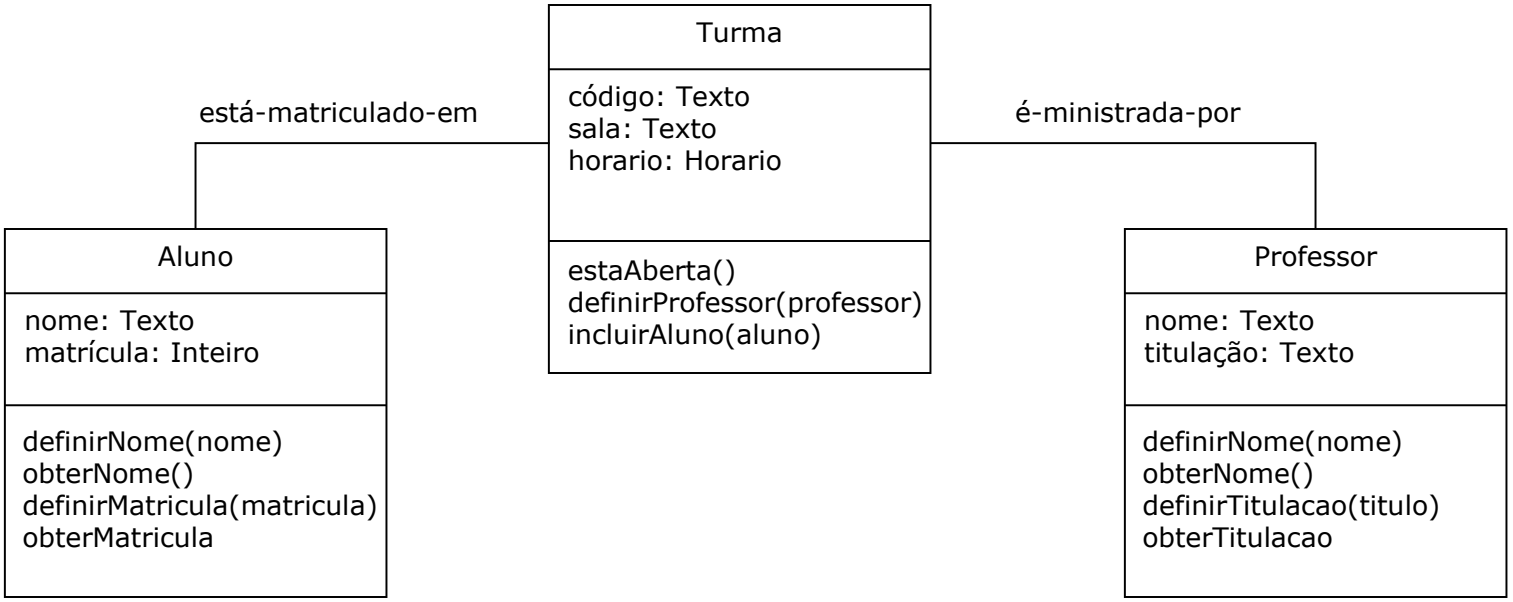
UML - Diagrama de Classes

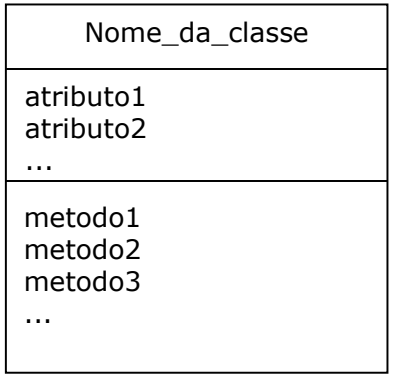
Diagrama de Classes

- Mostra um conjunto de classes e seus relacionamentos.
- É o diagrama central da modelagem orientada a objetos.



Classes

- Graficamente, as classes são representadas por retângulos incluindo seu nome, atributos e métodos.



- Devem receber nomes de acordo com o vocabulário do domínio do problema.
- É comum adotar-se padrões para nomeá-las.
  - Por exemplo, todos os nomes de classes são substantivos singulares com a primeira letra maiúscula.

Classes: atributos

- Atributos
  - Representam o estado (conjunto de características) que a classe modela
  - Visibilidade:
    - + público: visível em qualquer classe de qualquer pacote
    - # protegido: vísivel para classes do mesmo pacote
    - - privado: visível somente dentro da classe
  - Exemplo:
    - + rua : String

Classes: métodos

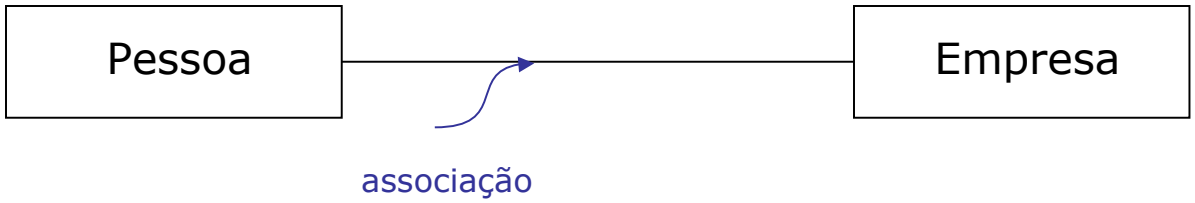
- Métodos
  - Representam o comportamento (conjunto de operações) que a classe fornece
  - Visibilidade:
    - + público: visível em qualquer classe de qualquer pacote
    - # protegido: vísivel para classes do mesmo pacote
    - - privado: visível somente dentro da classe
  - Exemplo:
    - + setEndereco(rua: String, numero: int, bairro: String): void

Relacionamentos

- Relacionamentos
  - nome: descrição dada ao relacionamento (faz, tem, possui,...)
  - sentido de leitura do relacionamento
  - navegabilidade: indicada por uma seta no fim do relacionamento
  - navegabilidade ≠ sentido de leitura
  - multiplicidade: 0..1, 0..*, 1, 1..*, 2, 3..7
  - tipos de relacionamentos: associação (agregação, composição), generalização e dependência
- Papéis
  - Desempenhados por classes em um relacionamento

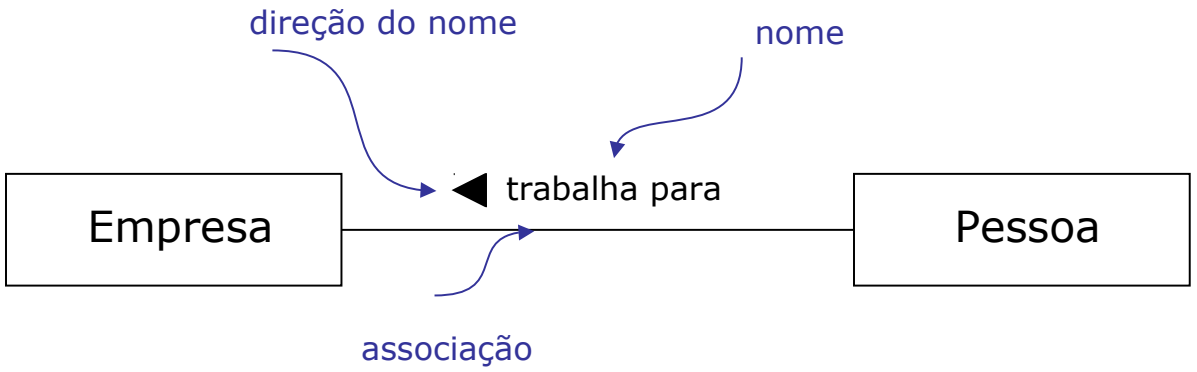
Relacionamentos: Associação

- Uma **associação** é um relacionamento estrutural que indica que os objetos de uma classe estão vinculados a objetos de outra classe.
- Uma associação é representada por uma linha sólida conectando duas classes.



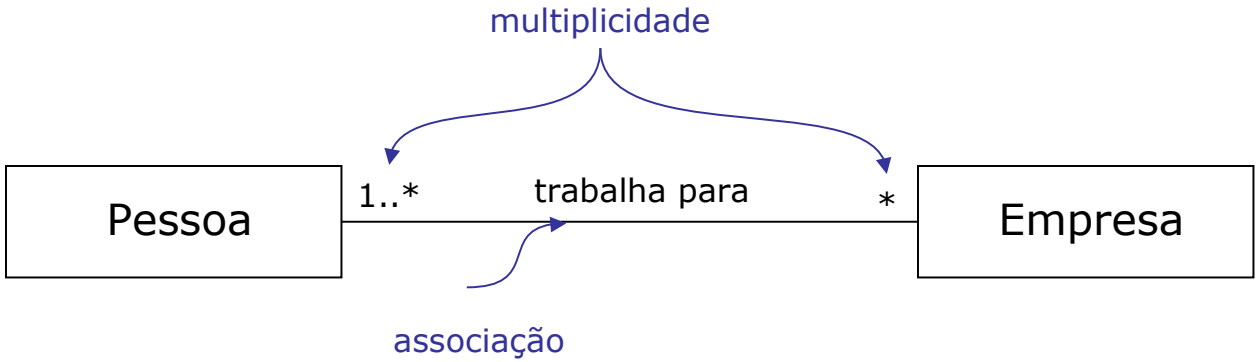
Relacionamentos: Associação

- Uma associação pode ter um **nome.**
  - Torna mais clara a natureza do relacionamento.
- Se necessário podemos incluir um triângulo para indicar a direção de leitura do nome.
  - Evita ambigüidades



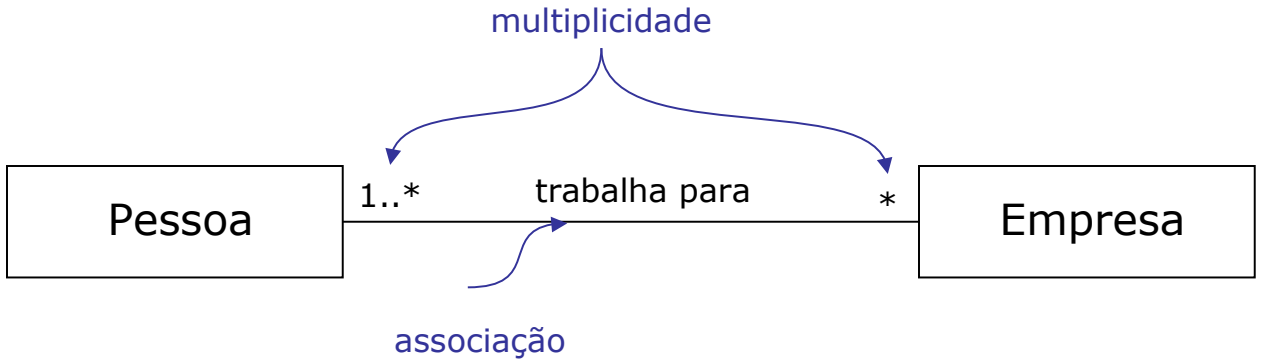
Relacionamentos: Associação

- **Multiplicidade** é o número de instâncias de uma classe que se relacionam com **uma** instância de outra classe.



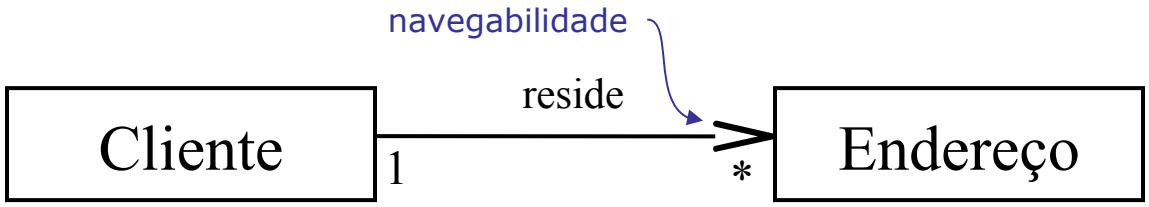
Relacionamentos: Associação

- Indicadores de multiplicidade mais comuns:
  - 1 Exatamente um
  - 1..* Um ou mais
  - 0..* Zero ou mais (muitos)
  - * Zero ou mais (muitos)
  - 0..1 Zero ou um
  - m..n Faixa de valores (por exemplo: 4..7)



Relacionamentos: Associação

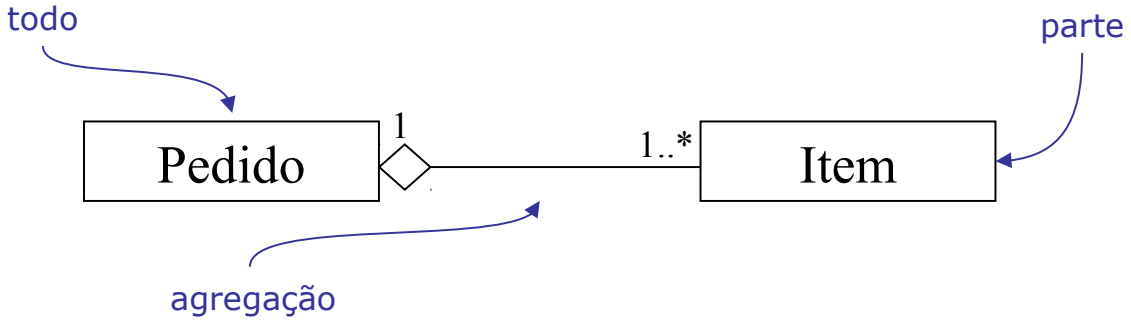
- Em geral, associações são bidirecionais.
  - Pode ser desejável limitar sua ”navegação” em uma única direção.
  - A **navegabilidade** é indicada por uma seta em uma das extremidades da associação.



- Um cliente sabe quais são os seus endereços, mas o endereço não sabe qual é o seu cliente.

Relacionamentos: Agregação

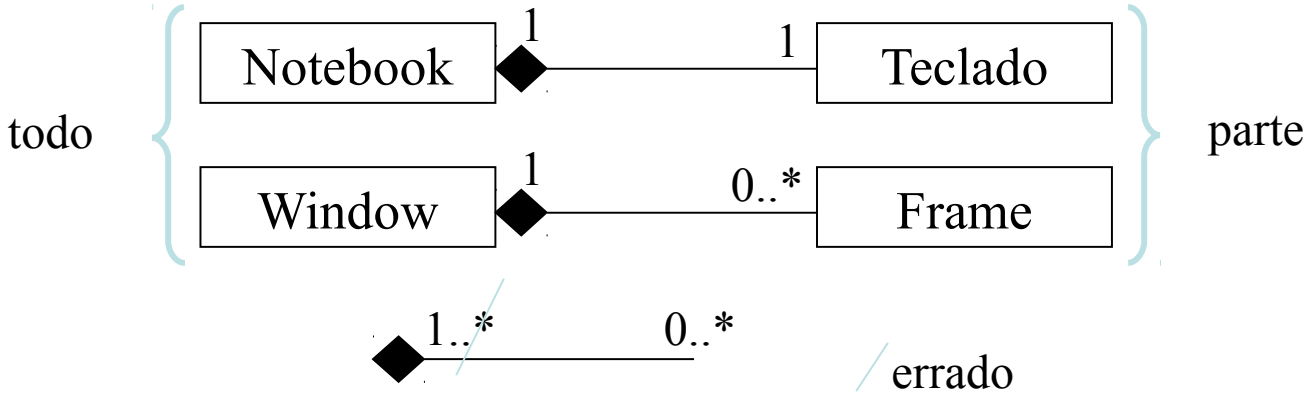
- Uma **agregação** é um tipo especial de associação utilizada para indicar um relacionamento ”todo-parte”.
- Uma agregação é representada por uma linha sólida com um losango vazado na extremidade referente ao todo.



- O significado da agregação é inteiramente conceitual, o losango simplesmente diferencia o todo da parte.
- Um objeto “parte” pode fazer parte de vários objetos “todo”.

Relacionamentos: Composição

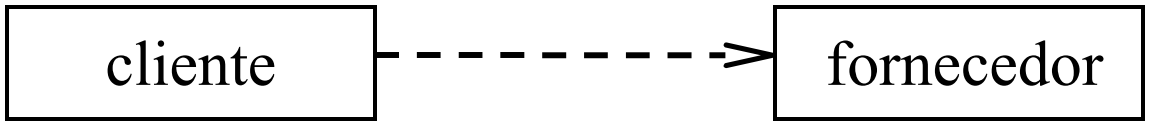
- Em uma **composição,** os objetos "parte" só podem pertencer a um único objeto "todo" e têm seu tempo de vida coincidente com o dele.
  - É uma variante semanticamente mais "forte" da agregação
  - É representada por uma linha contínua com um losango cheio na extremidade referente ao todo.



  - Quando o ***todo*** "morre" todas as suas ***partes*** também "morrem"
  - Apenas o ***todo*** pode criar e destruir as ***partes***

Relacionamentos: Dependência

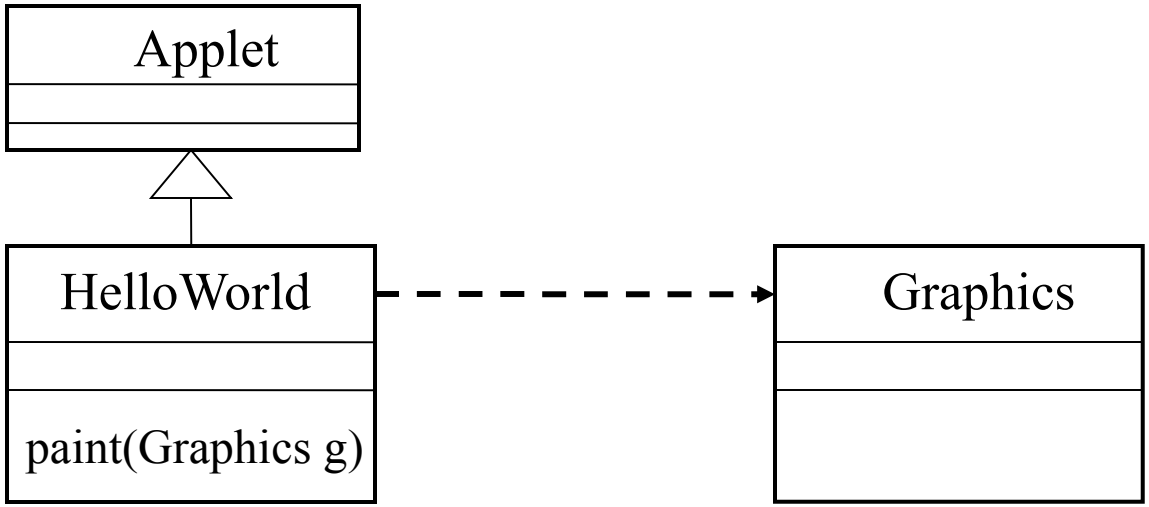
- Uma **dependência** é um relacionamento de utilização entre dois itens, no qual a alteração de um (**o item independente**) pode afetar o outro (**o item dependente**).
  - É representada por uma linha tracejada com uma seta apontando para o item independente.



- A classe cliente não declara nos seus atributos um objeto do tipo fornecedor.

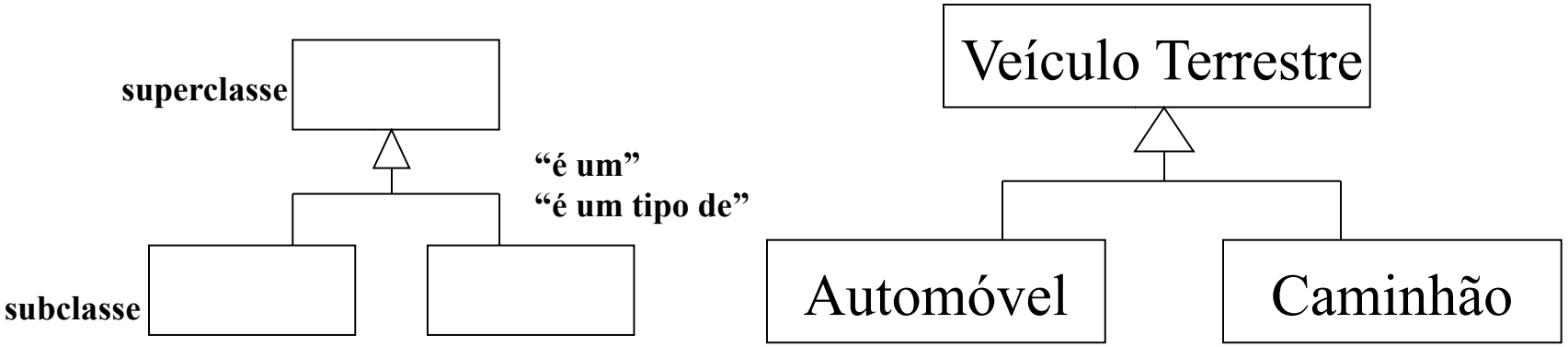
Relacionamentos: Dependência

```
Import java.awt.Graphics;
class HelloWorld extends java.applet.Applet
{
    public void paint (Graphics g)
            g.drawString("Hello, world!", 10, 10);
}
```



Relacionamentos: Generalização

- Uma **generalização** é um relacionamento entre itens gerais (superclasse) e itens mais específicos (subclasses).
  - É representada por uma linha sólida com um triângulo vazado apontando para o item mais geral.



O Blog (exemplo)

- Um *blog* tem um título e uma data de criação e além disso é um conjunto de conteúdos.
- Estes conteúdos podem ser notas ou comentários sobre as notas. Tanto notas quanto comentários têm características comuns como o texto e a data de sua criação.
- Todo usuário possui:
  - E-mail (deve ser único, ou seja, não há mais de um usuário com o mesmo e-mail)

Blog: o sistema deve...

- Permitir a criação de blogs
- Permitir a utilização de blogs
  - Qualquer usuário pode ler conteúdos
  - Somente o dono do blog pode criar notas
  - Qualquer usuário pode criar comentários. Para criar um comentário o usuários precisa ler as notas.
  - Somente o dono do blog pode remover conteúdos. Para remover um conteúdo ele precisará ler o conteúdo. Caso ele remova um comentário, o autor do comentário deve ser notificado por e-mail.