Modelação e Design 11: Diagrama de Classes: conceitos básicos

Leonor Melo

leonor@isec.pt

1

Conceitos básicos

- Visibilidade
- Sintaxe dos Atributos

2

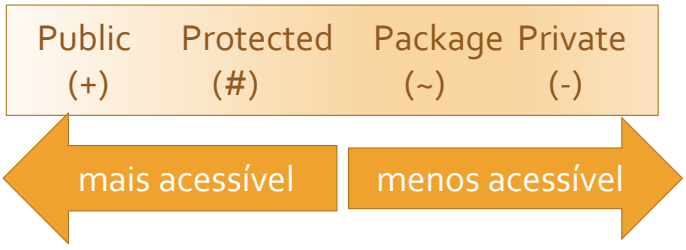
Visibilidade

- O encapsulamento é uma das vantagens mais unânimes da orientação a objetos:
  - Oferece robustez ao código
- O encapsulamento é conseguido controlando a visibilidade dos atributos e operações das classes

3

Visibilidade

- Indica aquilo que a classe esconde
  - e aquilo que disponibiliza
- UML permite 4 níveis de visibilidade:
  - visibilidade UML pode ser diferente da visibilidade de algumas linguagens de programação!



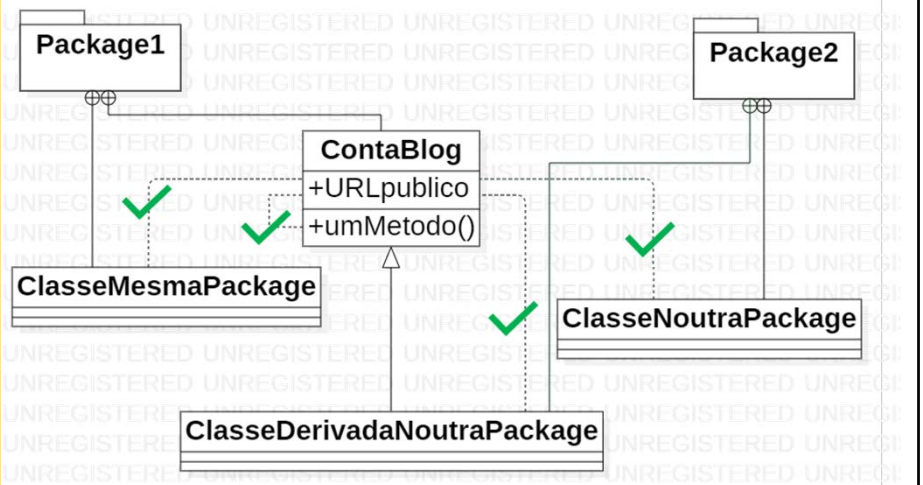
4

Visibilidade Public - 1

- Visibilidade public
  - a mais acessível
  - + antes do atributo ou operação
  - acessível para métodos de qualquer outra classe

5

Visibilidade Public - 2



6

Visibilidade Public - 3

- interface público de uma classe
  - coleção de atributos e operações públicos dessa classe
  - deve mudar o mínimo ao logo do tempo para não afetar o funcionamento das outras classes
- Normalmente evita-se usar atributos públicos
  - se um atributo for visível também é modificável
    - exceção: se atributo for uma constante usada por outras classes (ex. Pi)

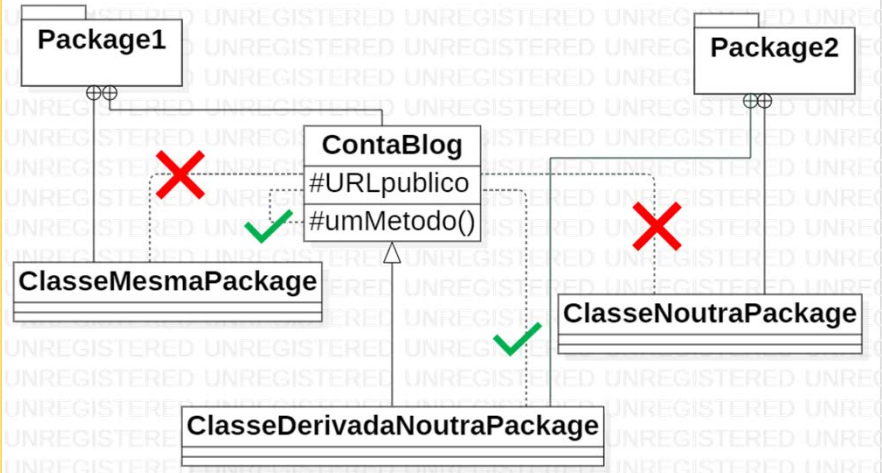
7

Visibilidade Protected - 1

- Visibilidade protected
  - menos acessível que a public
  - # antes do atributo ou operação
  - acessível para métodos da própria classe e para métodos das classes descendentes dela
    - inacessível para os métodos das restantes classes, estejam elas na mesma package ou não

8

Visibilidade Protected - 2



9

Visibilidade Protected - 3

- Crucial para deixar que as classes especializadas aceder aos atributos e operações da classe mais genérica
  - mas sem disponibilizar o atributos e operações a todo o sistema
  - operações e atributos úteis para o funcionamento interno da classe (e suas derivadas) mas que mais ninguém deve usar

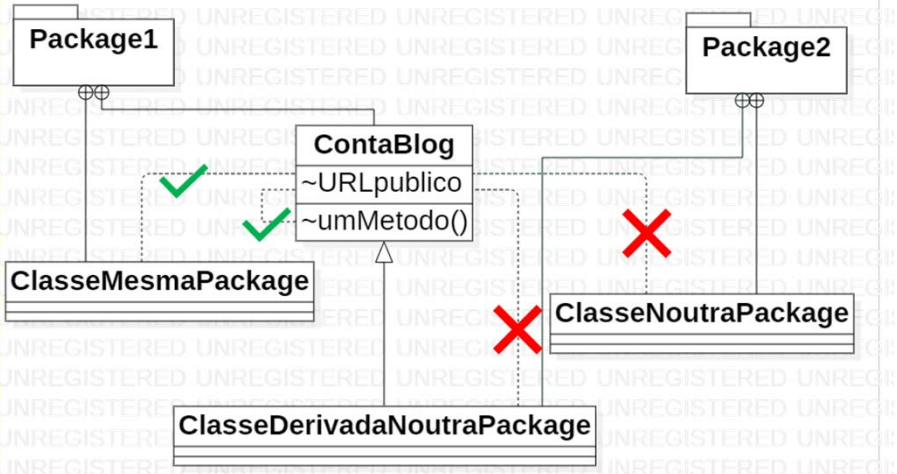
10

Visibilidade Package - 1

- Visibilidade package
  - mais acessível que a private
  - ~ antes do atributo ou operação
  - acessível para os métodos da própria classe e para os métodos das classes da mesma package
    - inacessível para os métodos das restantes classes, mesmo que sejam classes derivadas dessa

11

Visibilidade Package - 2



12

Visibilidade Package - 3

- Usada sobretudo quando temos uma coleção de método que queremos reutilizar dentro da mesma package
  - Ex. Numa package de "utility classes", queremos reutilizar os métodos dentro das classes dessa package, mas não queremos que fiquem expostos ao resto do sistema.
    - métodos para partilhar dentro da package: package
    - métodos para partilhar com o sistema: public

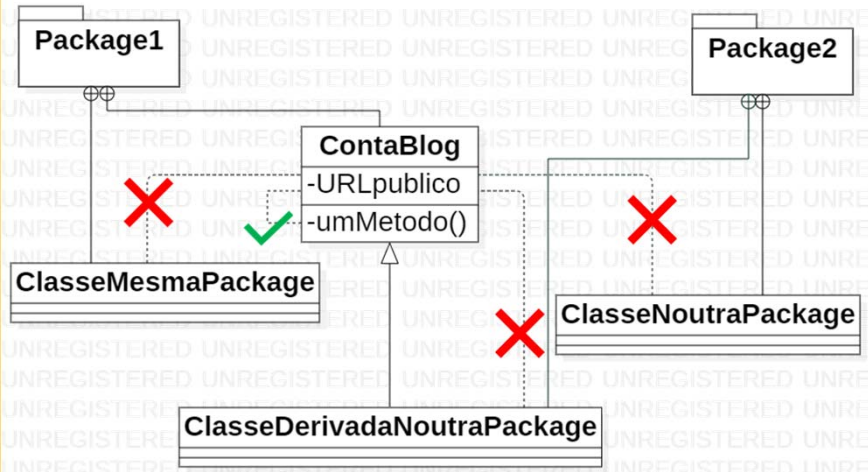
13

Visibilidade Private - 1

- Visibilidade private
  - a menos acessível
  - - antes do atributo ou operação
  - acessível para métodos da própria classe

14

Visibilidade Private - 2



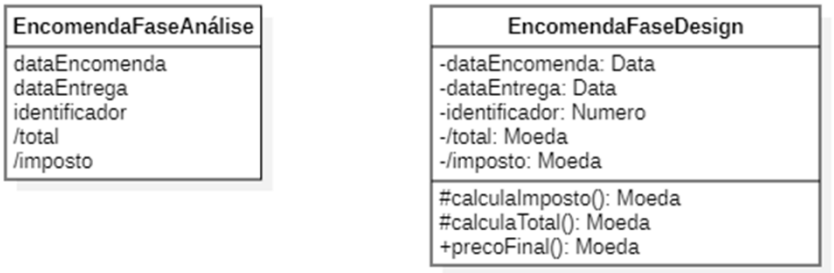
15

Visibilidade Private - 3

- Por regra os atributos são private
  - eventualmente protected se forem necessários nas classes descendentes
- Operações private são aquelas que dizem respeito ao funcionamento interno da classe
  - ou que queremos poder alterar mais tarde

16

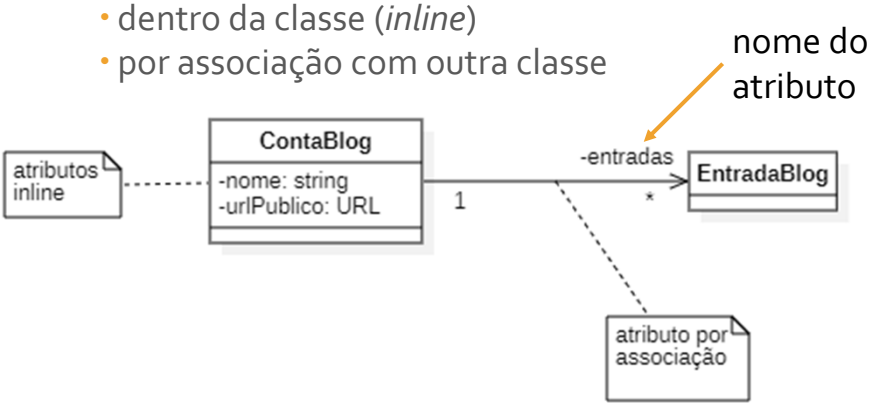
Exemplo visibilidade: (excerto de diagrama de classes sobre o processamento de encomendas)



17

Atributos - 1

- Atributos (propriedades) da classe
  - estado do objetos
- Podem ser representados
  - dentro da classe (*inline*)
  - por associação com outra classe



18

Atributos - 2

- Formato completo:
  - visibilidade nome : tipo multiplicidade = default {modificador-de-propriedade}
- No mínimo, a assinatura do atributo tem:
  - visibilidade
  - nome
  - tipo
- mas só nome é realmente obrigatório

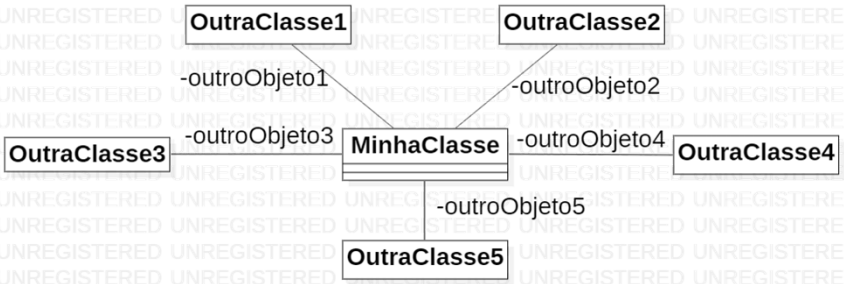
19

Atributos - Nome e tipo

- Nome:
  - conjunto de carateres
  - único dentro da classe
  - deve descrever a informação que o atributo representa
- Tipo:
  - tipo de dados primitivo
    - inteiro, booleano, ...
  - outra classe
- Podem adequar-se às convenções da linguagem usada na implementação

20

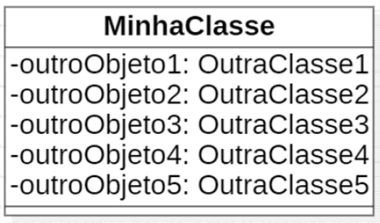
Atributos inline vs Atributos por associação - 1



- por associação
  - relações de associação entre classes evidente
  - diagrama rapidamente cheio
  - obscurece outras relações entre classes

21

Atributos inline vs Atributos por associação - 2



- inline
  - relações de associação entre classes menos claras
  - diagrama mais conciso
    - espaço para outras informações
  - outras relações entre classes em evidencia

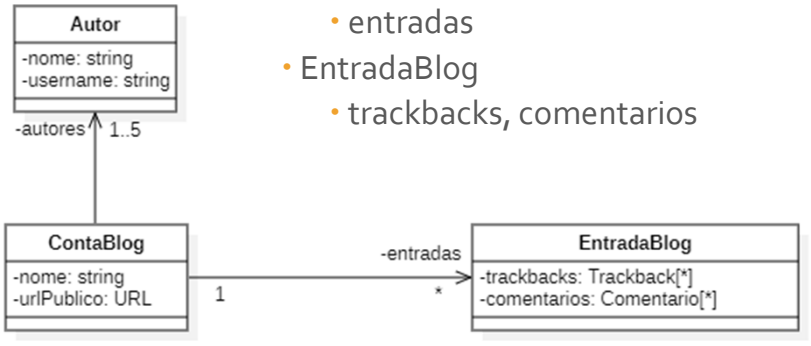
22

Atributos - Multiplicidade - 1

- Um atributo pode representar mais do que um objeto
  - pode representar todo um conjunto de objetos desse tipo
- Multiplicidade
  - indica que um atributo é na realidade uma coleção
  - aplica-se a atributos
    - inline
    - por associação

Atributos - Multiplicidade - 2

- Quantos objetos podem ser guardados?
  - ContaBlog:
    - nome, urlPublico
    - autores
    - entradas
  - EntradaBlog
    - trackbacks, comentarios



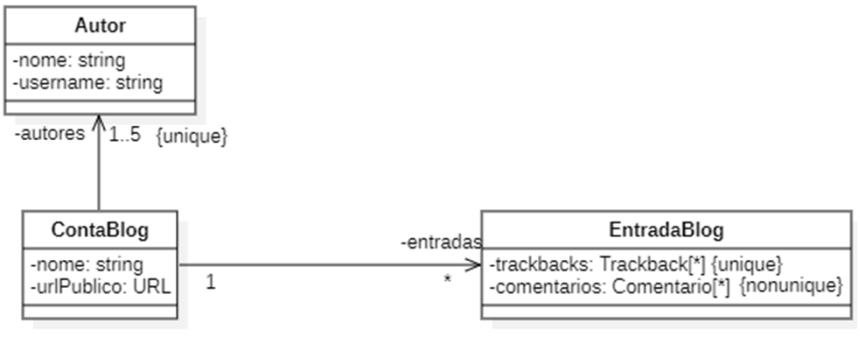
Modificadores de Propriedade quando multiplicidade > 1

- Podemos usar "modificadores de propriedade" para especificar ainda com mais detalhe a multiplicidade:
  - modificador de propriedade *unique*
    - não existem elementos repetidos no atributo múltiplo
  - modificador de propriedade *nonunique*
    - podem existir elementos repetidos no atributo múltiplo
- por omissão os atributos múltiplos são unique

25

Modificadores de Propriedade quando multiplicidade > 1

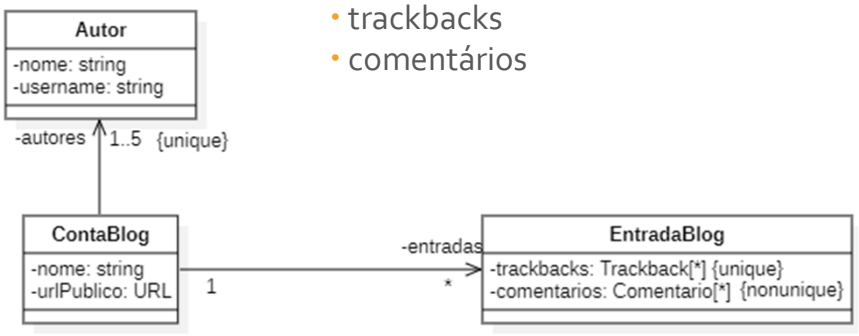
- Exemplo
  - Não faz sentido ter o mesmo autor repetido
  - Nem dois trackbacks iguais



26

Modificadores de Propriedade quando multiplicidade > 1

- Em quais dos seguintes atributos podem existir elementos repetidos?
  - autores
  - entradas
  - trackbacks
  - comentários



27

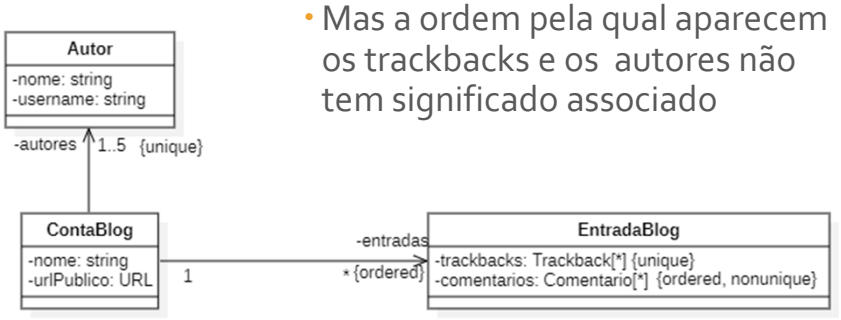
Modificadores de Propriedade quando multiplicidade > 1

- Podemos indicar se os elementos de determinado atributo múltiplo estão ordenados
  - modificador de propriedade *unique*
    - os elementos do atributo estão ordenados de acordo com algum critério
- por omissão os atributos múltiplos não têem nenhuma ordem associada

28

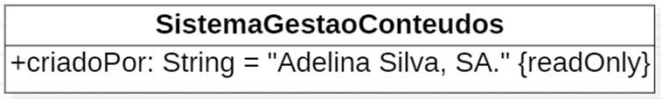
Modificadores de Propriedade quando multiplicidade > 1

- Faz sentido associar uma ordem às entradas do blog e aos comentários
  - por exemplo, a ordem pela qual foram adicionadas
- Mas a ordem pela qual aparecem os trackbacks e os autores não tem significado associado

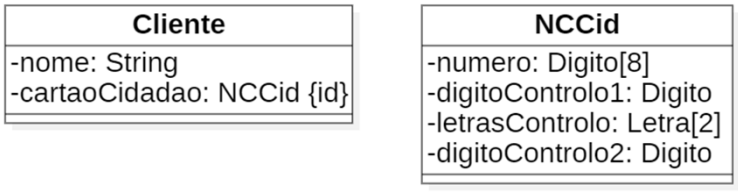


29

Outros modificadores de propriedade comuns



- Modificador de propriedade *readOnly*
  - valor não pode mudar depois da atribuição inicial
- Modificador de propriedade *id*
  - valor é/faz parte do identificador único desse objeto



30