

Unified Modeling Language

CSC207 Summer 2018



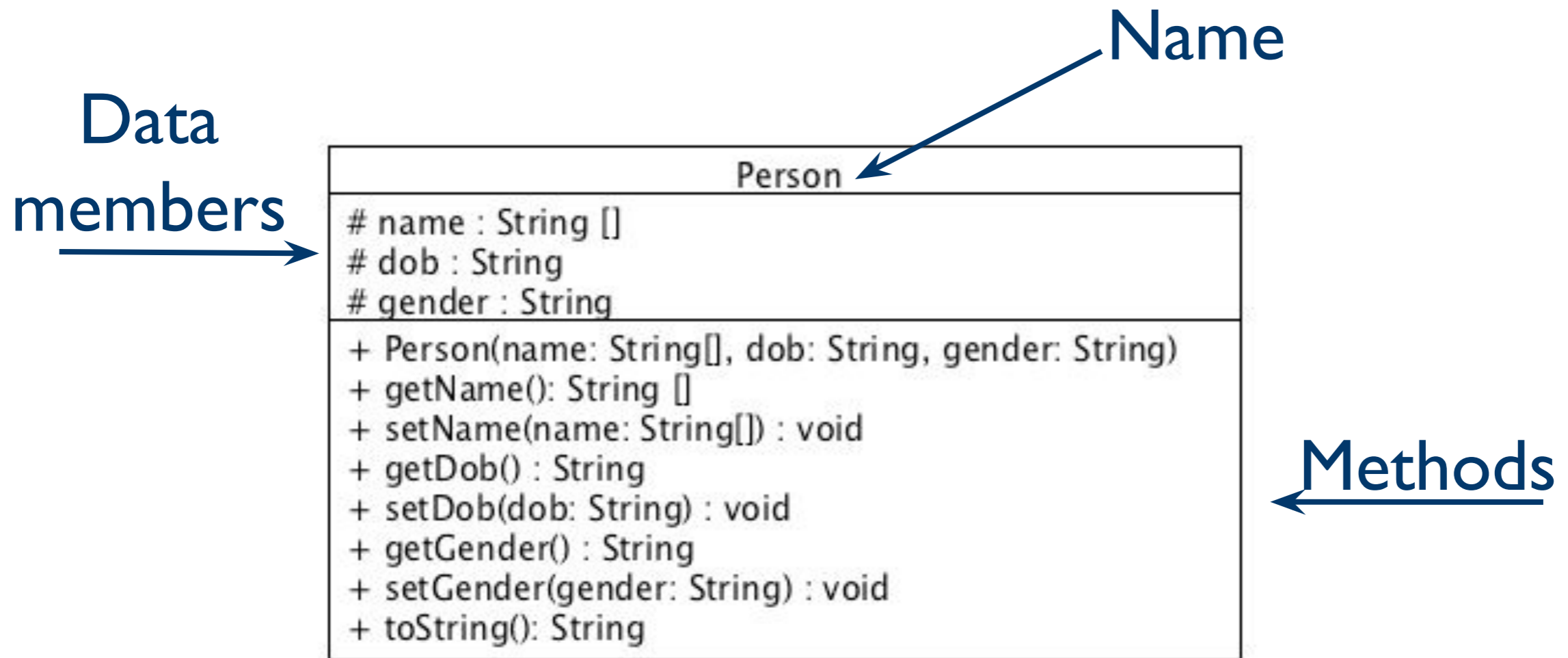
UML

Unified Modeling Language (UML)

An extremely expressive language.

We'll use only a small part of the language, Class Diagrams, to represent basic OO design.

Example: Class Person



Notation

Data members:

`name: type`

Methods:

`methodName (param1: type1, param2: type2, ...): returnType`

Visibility:

- private

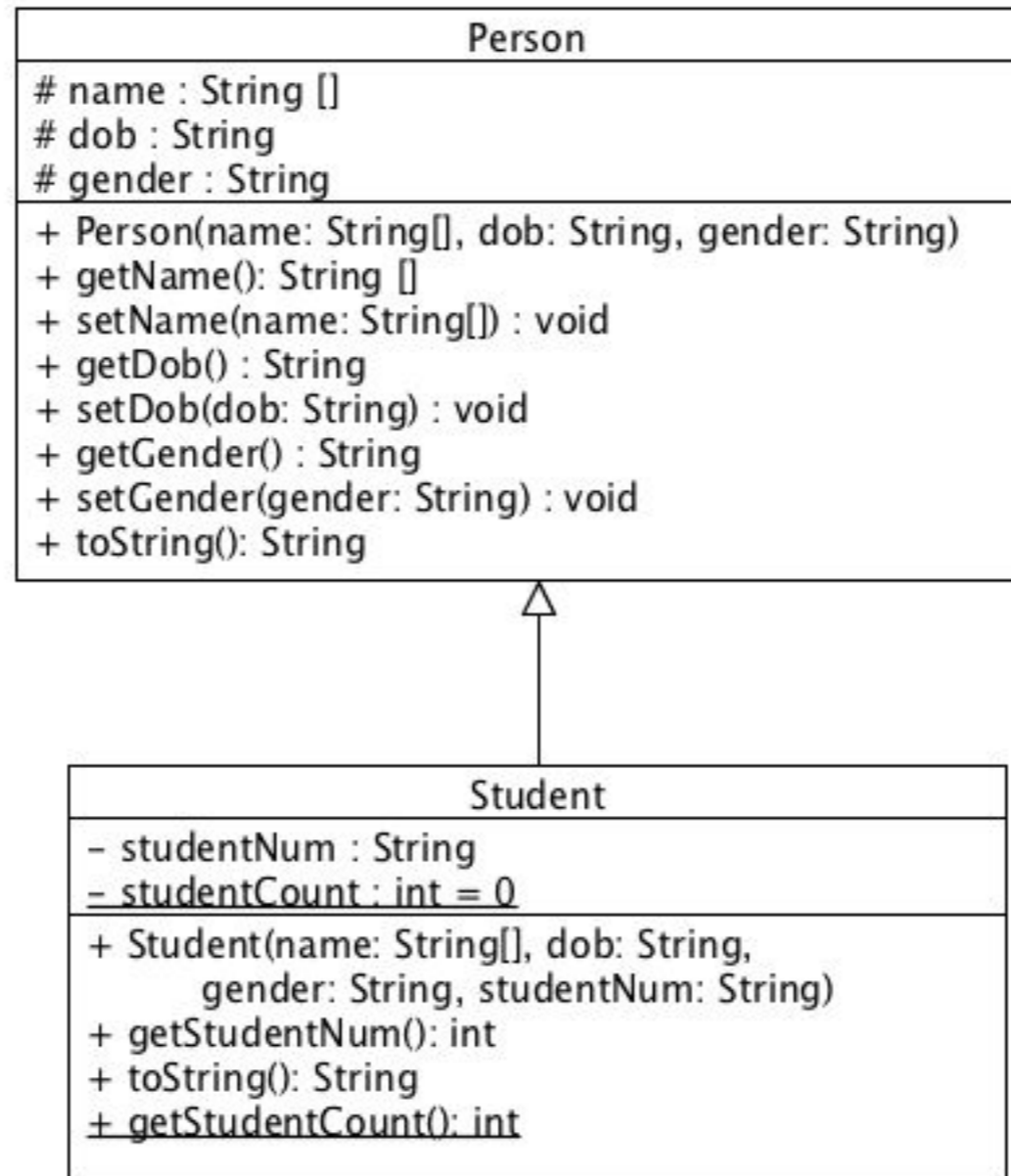
+ public

protected

~ package

Static: underline

Example: Inheritance



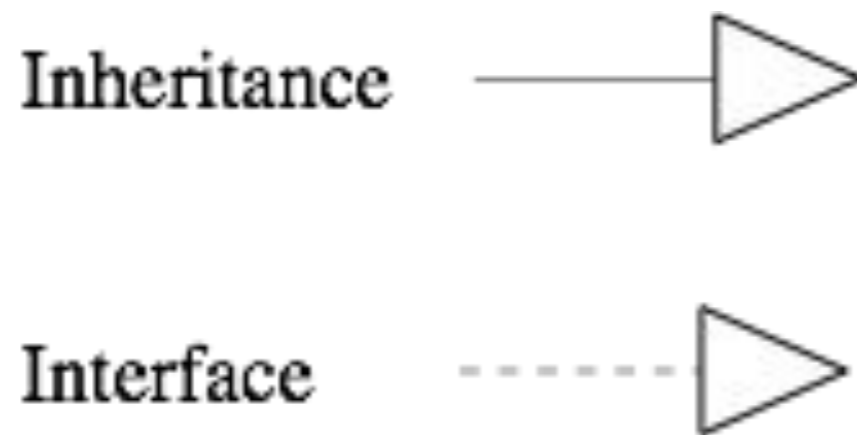
Notation (cont'd)

Abstract method: *italic*

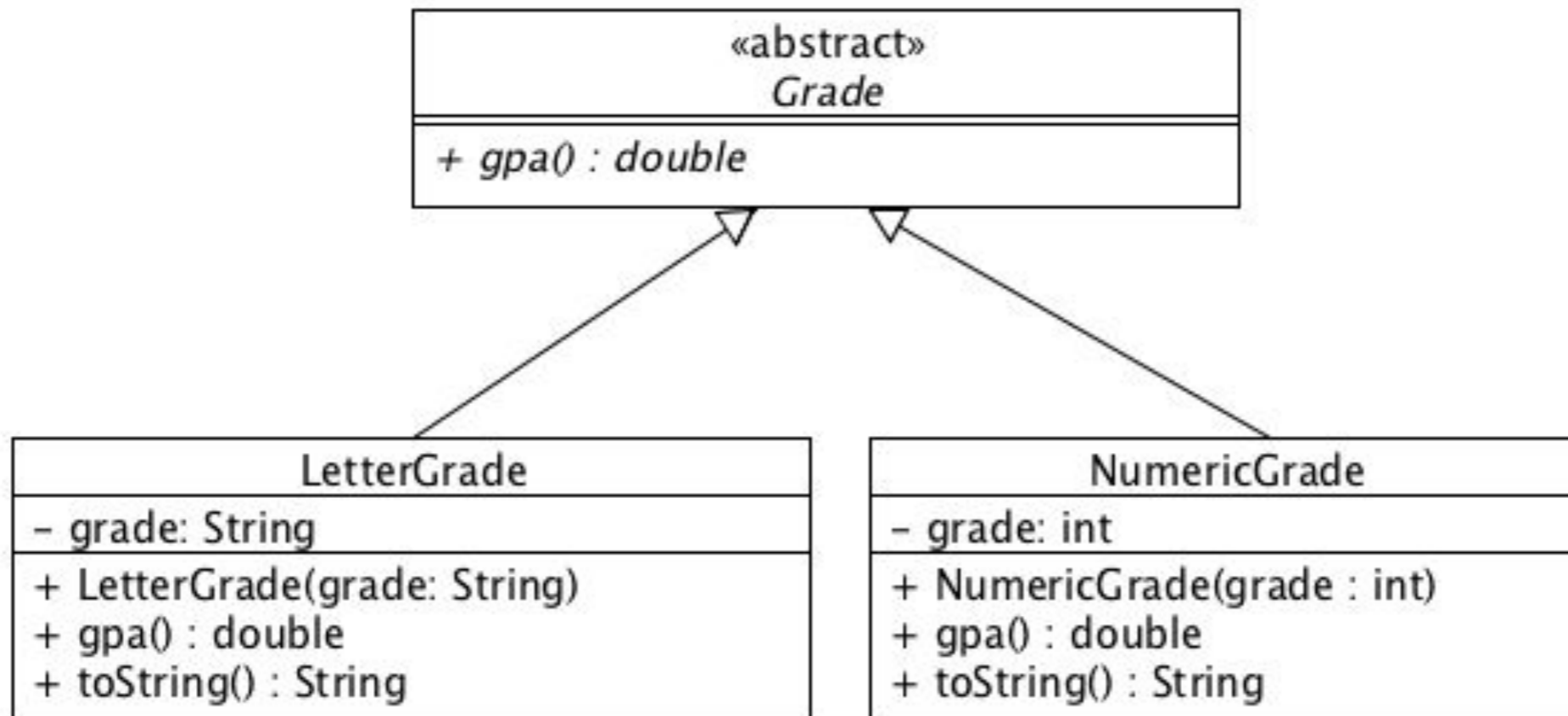
Abstract class: *italic* or <<abstract>>

Interface: <<interface>>

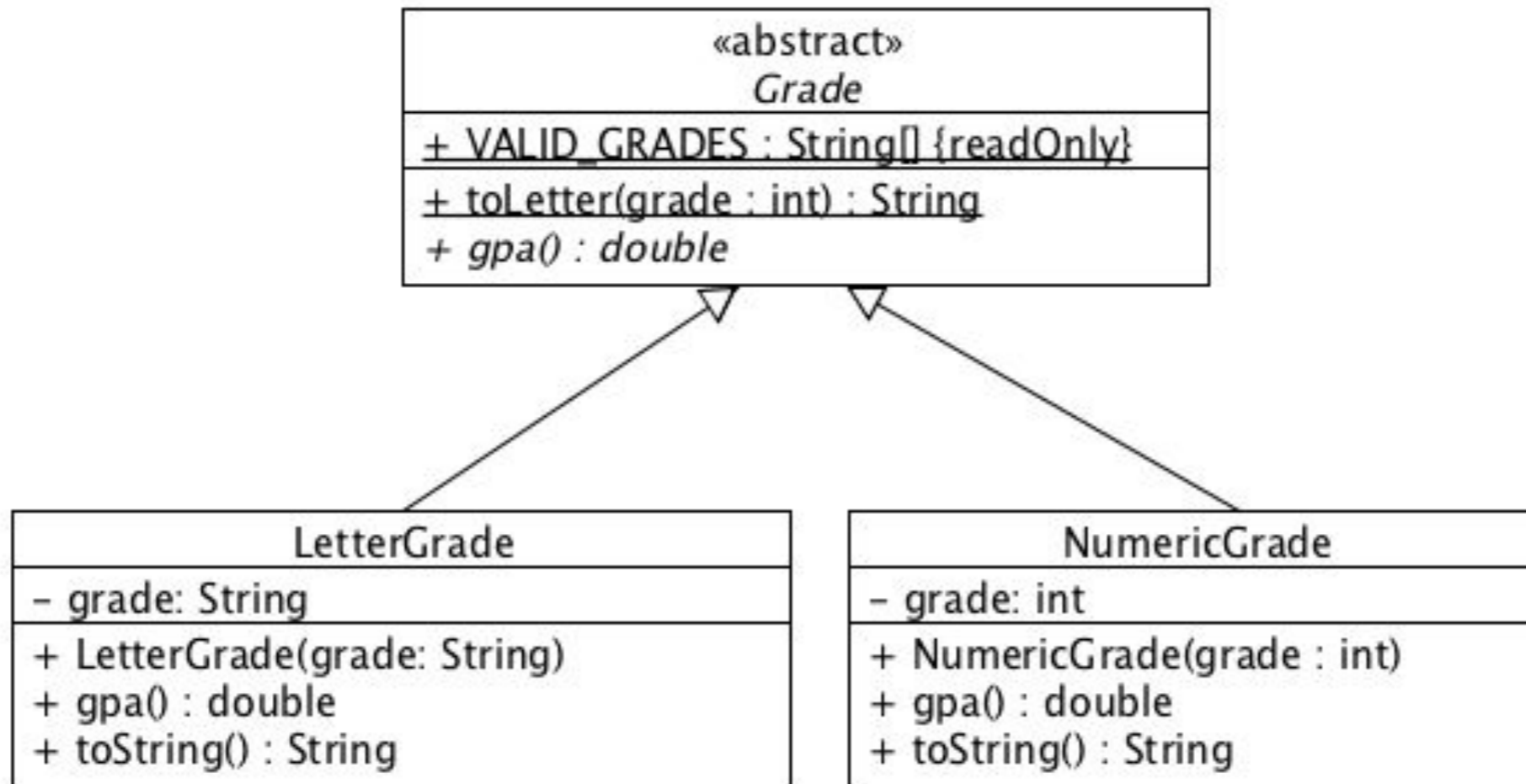
Relationship between classes:



Example: Abstract Class



Example: Abstract Class



Design Patterns

A **design pattern** is a general description of the solution to a well-established problem using an arrangement of classes and objects.

Patterns describe the shape of code rather than the details.

They're a means of communicating design ideas.

They are not specific to any one programming language.

You'll learn about lots of patterns in CSC301 (Introduction to Software Engineering) and CSC302 (Engineering Large Software Systems).

Iterator Design Pattern

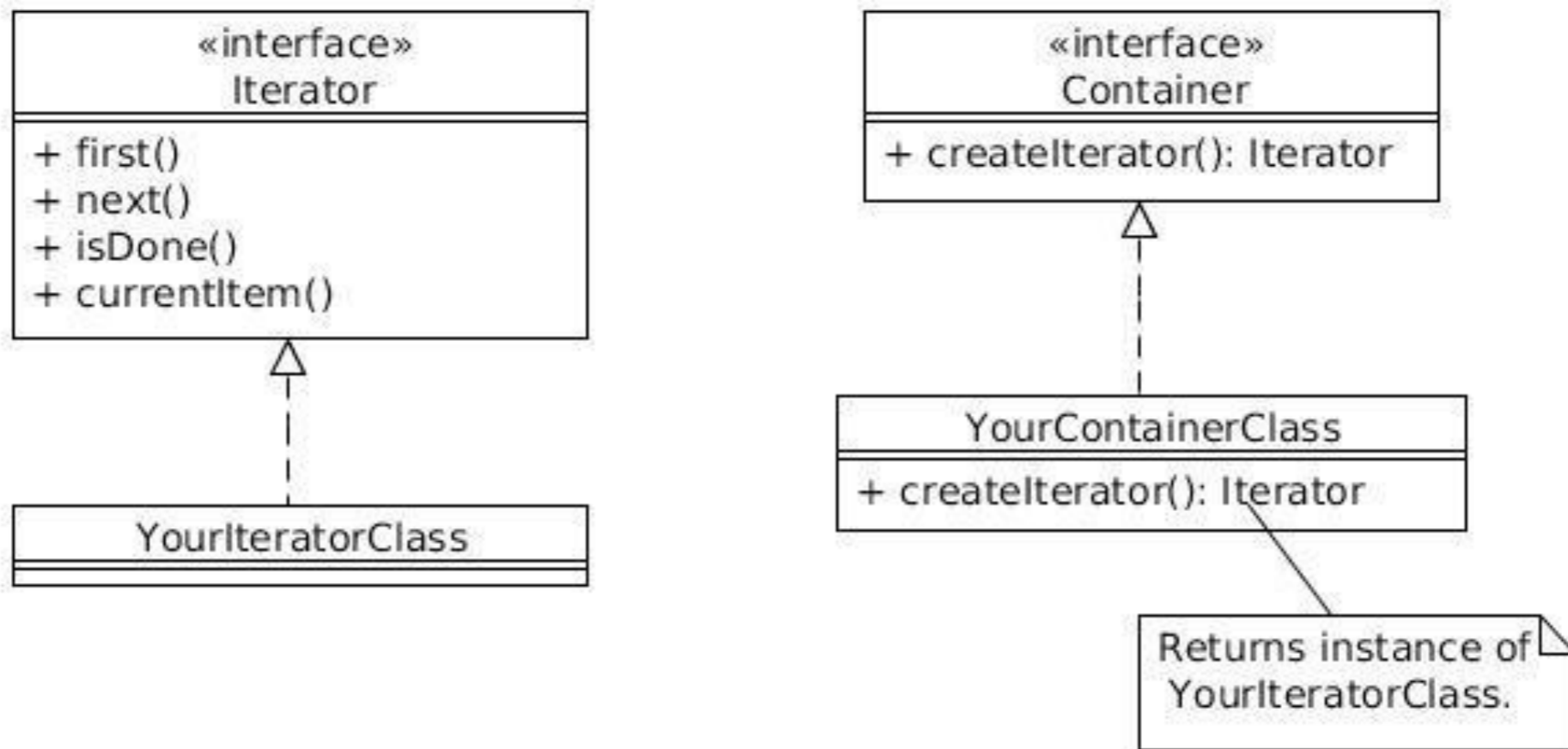
Context

- A container/collection object.

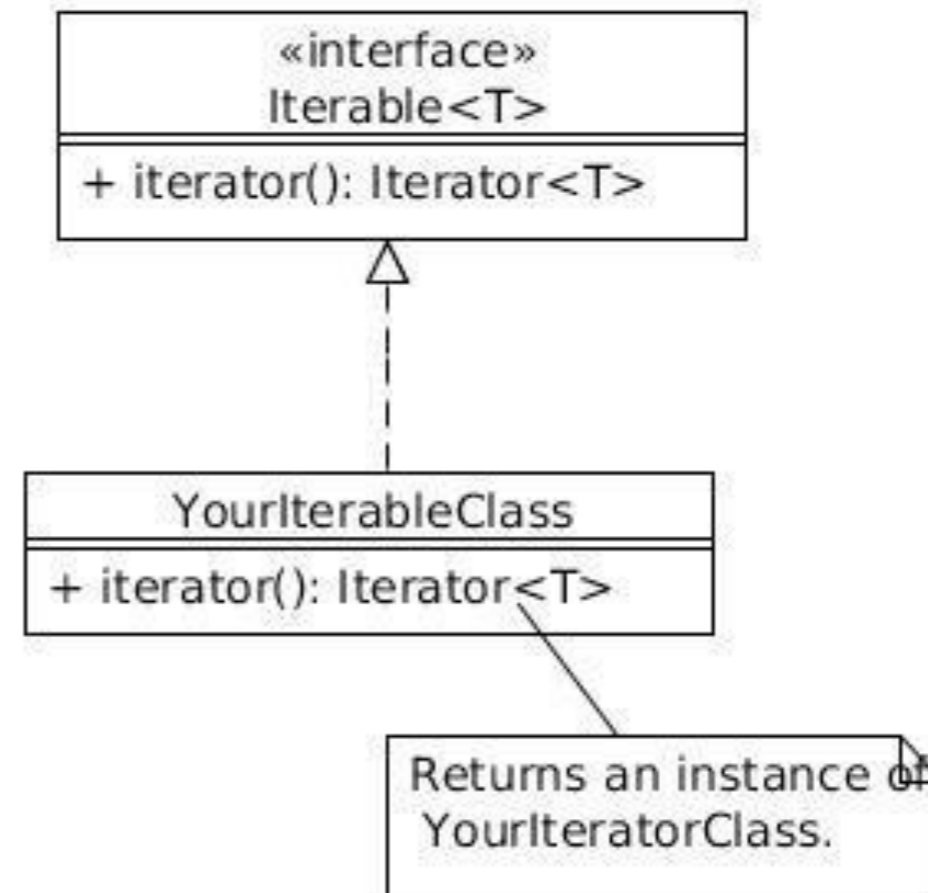
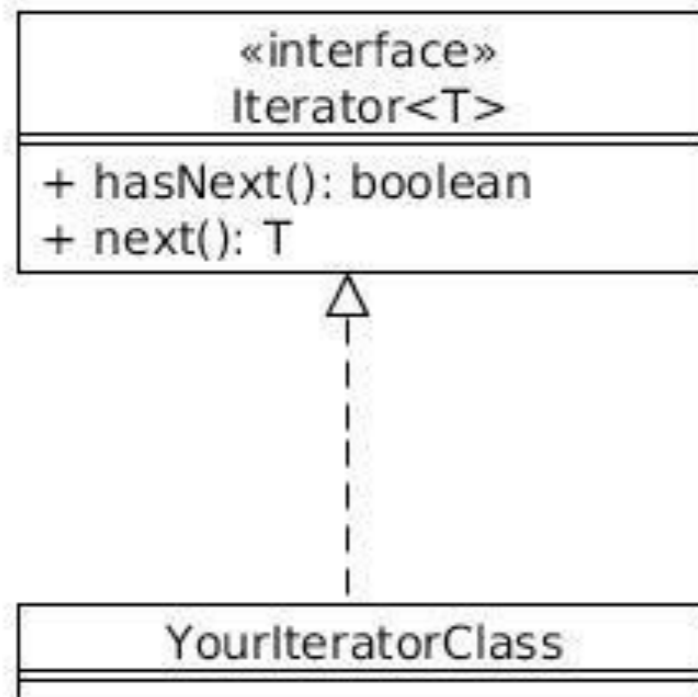
Problem

- Want a way to iterate over the elements of the container.
- Want to have multiple, independent iterators over the elements of the container.
- Do not want to expose the underlying representation: should not reveal how the elements are stored.

Iterator Design Pattern: Solution



Iterator Design Pattern: Java



Iterator: Example in Java

