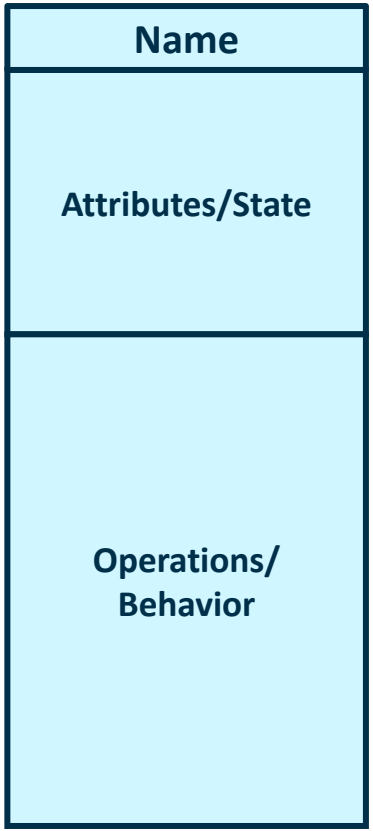
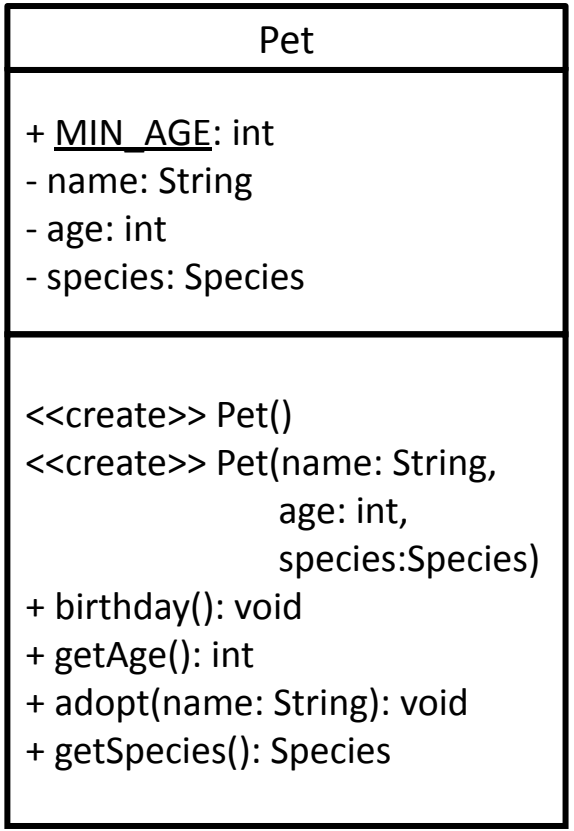
SWEN-262
Engineering of Software Subsystems

UML Class Diagrams

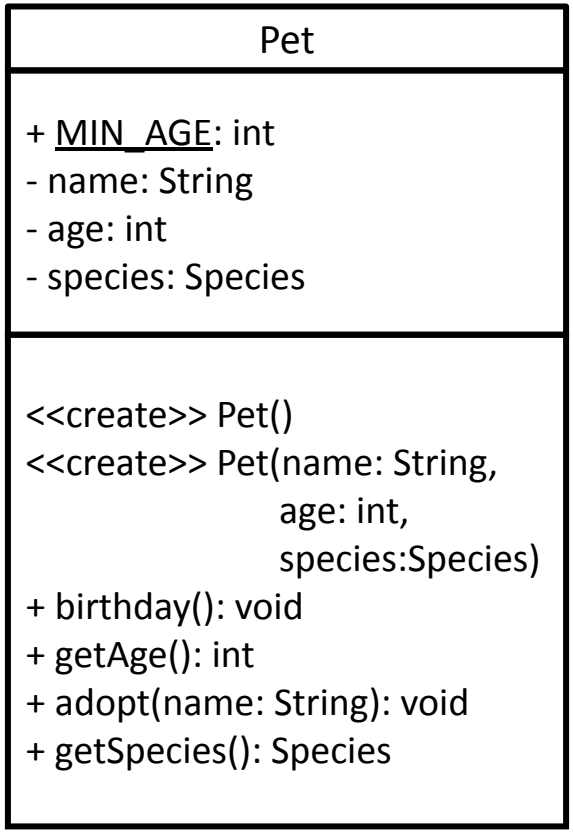
UML Class Diagrams

- The ***Unified Modeling Language*** (***UML***) is the standard used to ***diagram*** object-oriented software systems.
- While UML can be used to diagram many different aspects of a software system, the most common diagram is a ***class diagram***.
- Boxes partitioned into three parts are used to describe each class including:
  - The class ***name***, its ***attributes*** (fields), and its ***operations*** (methods).
- Visibility is specified for each attribute and operation:
  - **+** public
  - **-** private
  - **#** protected





A Closer Look at UML Class Diagrams



A Closer Look at UML Class Diagrams

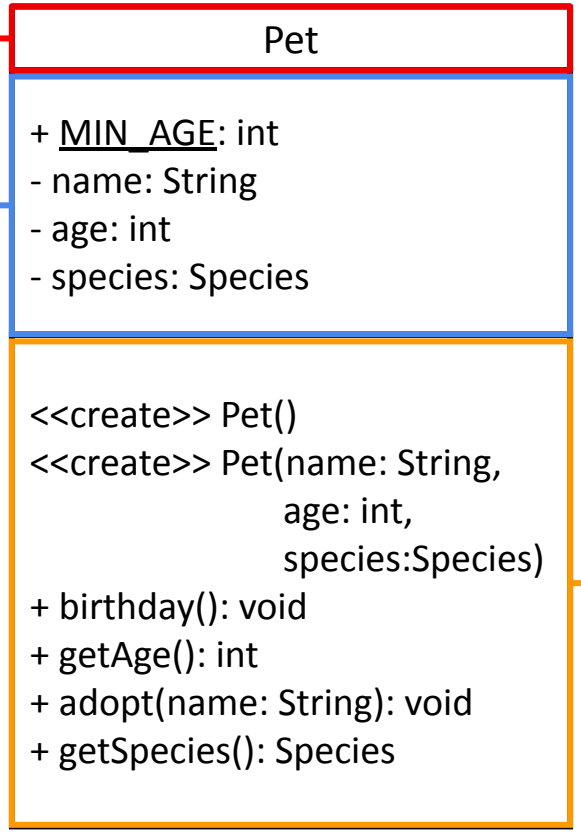
Classes in a UML class diagram are represented by boxes divided into three parts.

The class ***name*** is written in the topmost partition.

The ***attributes*** (fields) are listed in the middle partition.

Attributes are shown in with the ***name*** first followed by the ***type***, e.g.
`age: int`

Static attributes are *underlined* and by convention are named in UPPERCASE.



The ***operations*** (methods) re listed in the bottom partition.

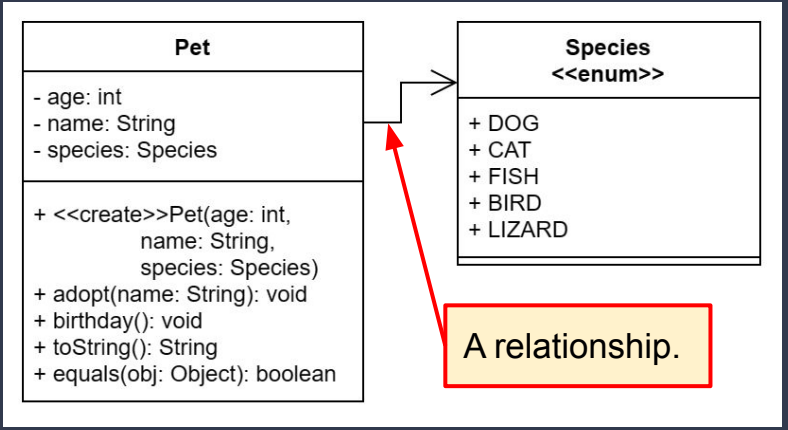
Any parameters are shown with the name first followed by the type, e.g.
`name: String`

The return value is listed after the method declaration, e.g.
`getAge(): int`

Constructors are marked with the `<<create>>` annotation.

Visibility is indicated using **+** (public), **–** (private), or **#** (protected).

UML – Relationships

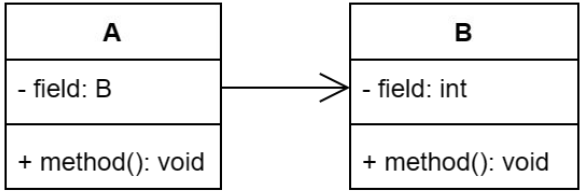


UML includes several different kinds of associations, each of which has a different meaning.

Professional software developers are expected to be able to implement code based on UML class diagrams.

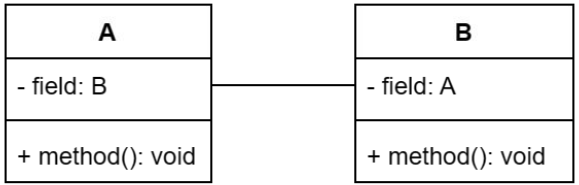
- In addition to describing individual classes, UML is used to describe the relationships that exist between classes.
  - A relationship between two classes indicates that the classes use each other in some way.
- Relationships are represented using a line that connects the two classes.
  - The lines may be solid or dashed.
  - A solid line indicates a stronger relationship.
- Relationships may also include an arrow that indicates ***directionality***.
  - The arrow points ***from*** a class ***to*** the class that it uses.
  - If there is no arrow, the relationship is ***undirected*** meaning that the classes ***use each other***.
- Class diagrams and relationships form a more complete picture of how all the components relate to each other.
  - The association syntax is very important, using the wrong ones is like using the wrong grammar or punctuation in a sentence; it totally changes the meaning, e.g. "Let's eat Grandma!" vs "Let's eat, Grandma."

Basic UML Relationships

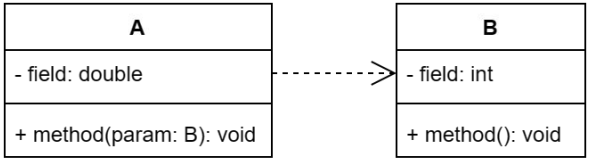


An ***association*** usually indicates that class ***A*** has a ***field*** of type ***B***.

Note the ***solid line*** and that the arrow points ***from*** A ***to*** B because A needs B (not the other way around).



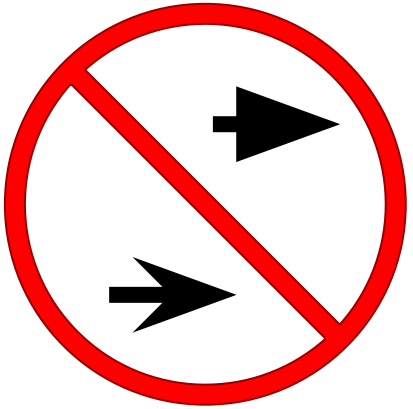
The absence of an arrow indicates that the association is ***undirected*** - both classes use ***each other***.



A ***dependency*** is weaker than an association and indicates that ***A uses B*** in some way, e.g. as a ***parameter*** or a ***return value*** but B is ***not*** part of the state of A.

Note the ***dashed line*** and that the arrow points in the direction of dependency: A depends on B (not the other way around).

A dependency may also be undirected if both classes use each other. In that case there is no arrow on the dashed line.



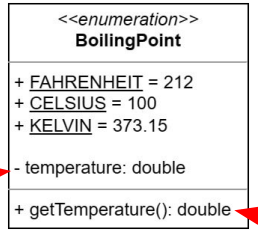
These kinds of arrows are ***never*** used in UML class diagrams.

More UML Notation

Simple ***enumerations*** are tagged using ***guillemets*** (***<< >>***).

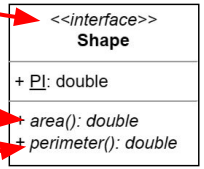
The enumerated values are listed below the name and the box for operations is omitted.

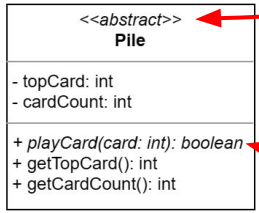
The values in an enum are ***constants***. Constants are underlined in UML class diagrams.



Complex enumerations typically declare fields and methods. These should be shown in the diagram.

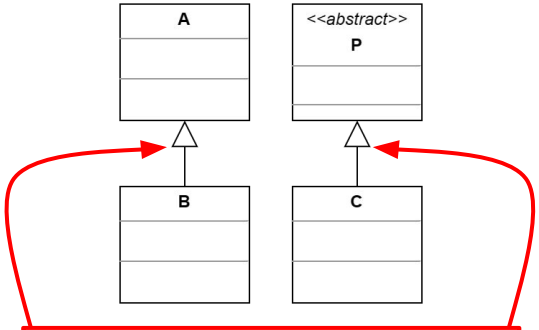
Interfaces and ***abstract classes*** are tagged using guillemets.





Abstract methods are shown in ***italics***.

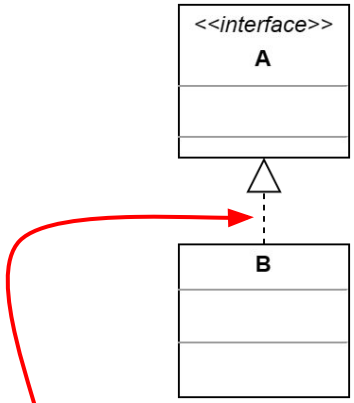
More UML Relationships



Inheritance is indicated with a ***solid line*** with a ***white arrowhead***.

The arrow points ***from*** the child class ***to*** the parent class. In this case ***child B extends parent A***.

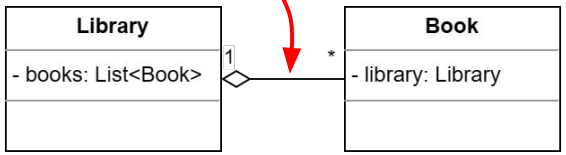
The formal name for this relationship is ***specialization*** because B is a more specialized class than A.



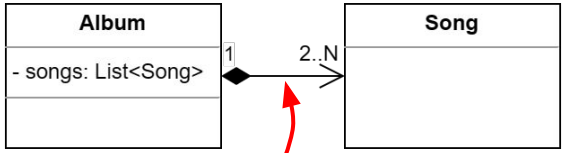
Interface inheritance is indicated with a ***dashed line*** with a ***white arrowhead***.

The formal name for this relationship is ***realization*** because class B is real (or concrete) implementation of interface A.

If one class ***contains*** another as part of its state, an ***aggregation*** relationship is indicated with a ***white diamond***.

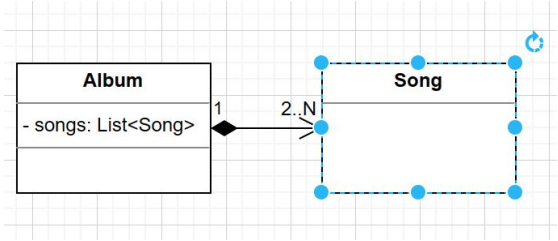


Multiplicity should be indicated if it is known. In this case, 1 ***Library*** contains zero or more ***Books***.



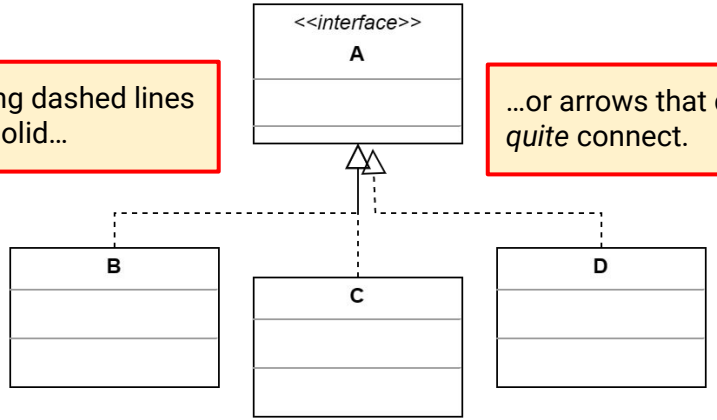
Composition is indicated with a ***black diamond***. It is like aggregation except, in this case, Song ***cannot exist*** without an Album to contain it.

Things to Avoid

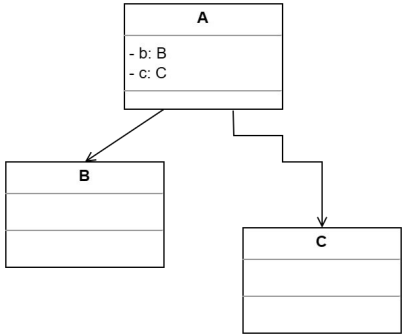


Submitting screenshots of your diagrams. Save them as an image file.

Overlapping dashed lines that turn solid…



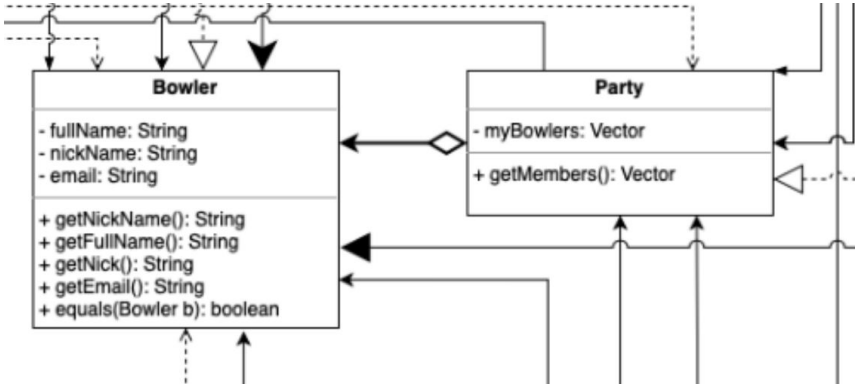
…or arrows that don't *quite* connect.



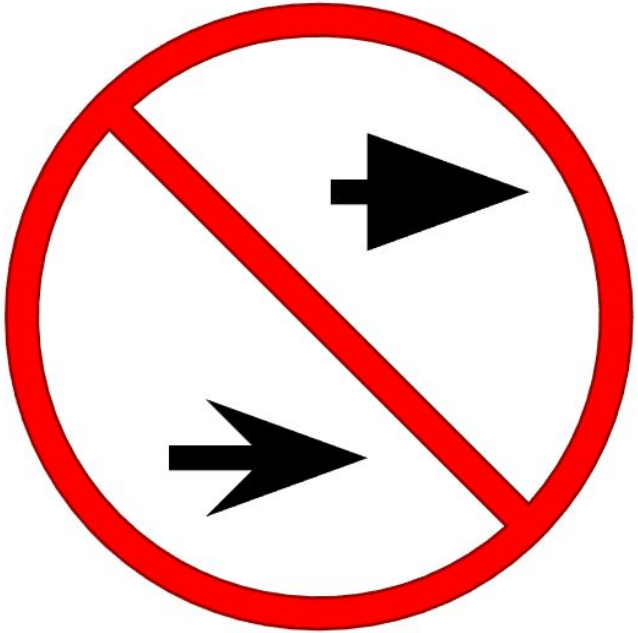
Angled lines or lines with unnecessary bends.

In general, anything that makes your diagrams look sloppy, rushed, or unprofessional.

More Things to Avoid



This...



These kinds of arrows are ***never*** used in UML class diagrams.