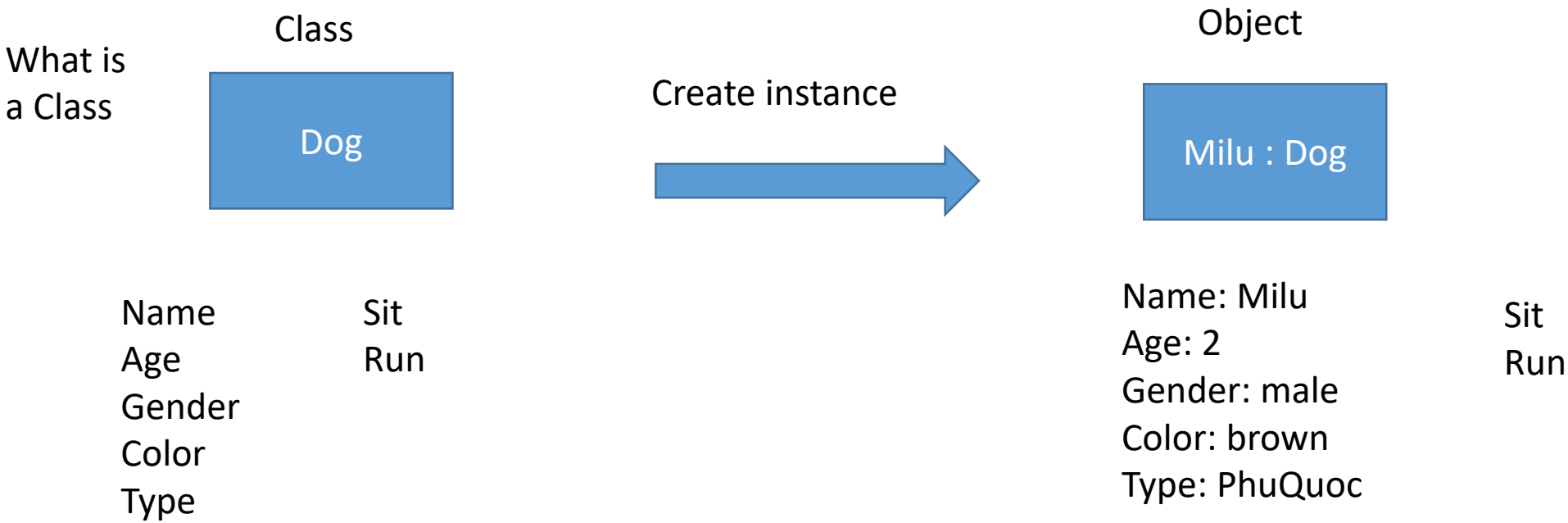
Class diagram guide

Notes for project #4

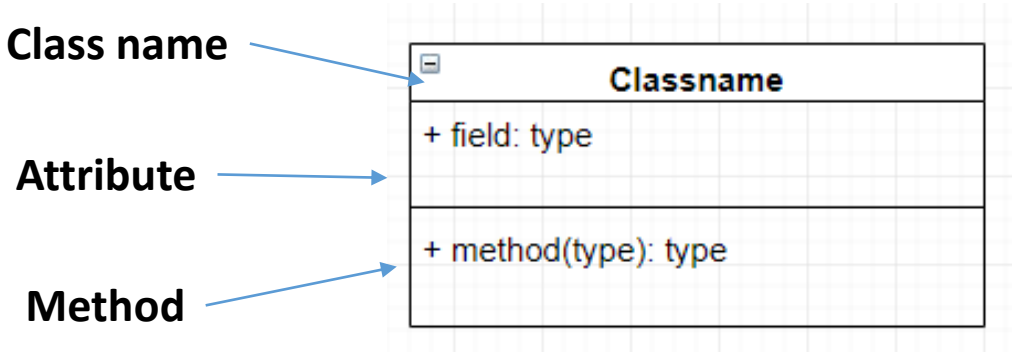
11/2019

What is a Class diagram

- Class diagrams, which show the object classes in the system and the associa-tions between these classes.



Class



https://www.uml-diagrams.org/visibility.html

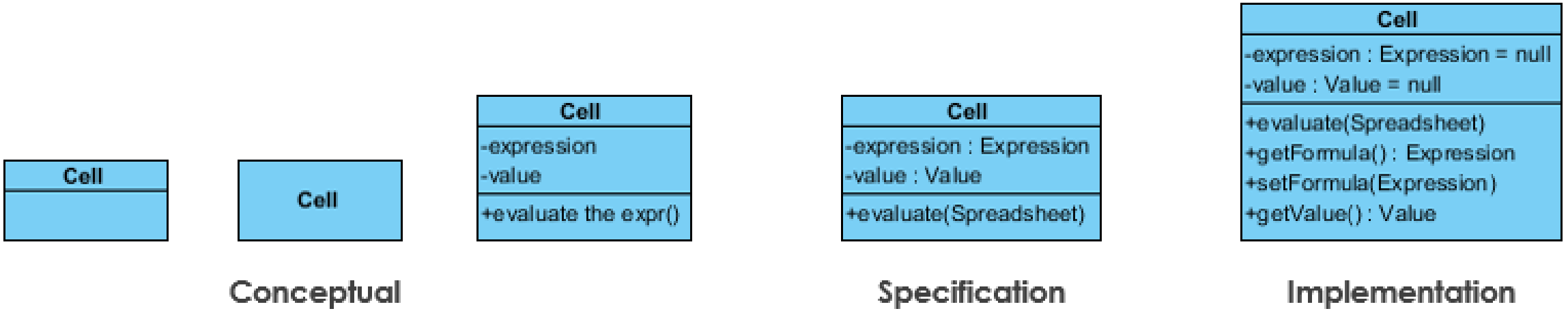
Visibility of Class Members
(attributes and methods)

\+ : public
\- : private
\# : protected
~ : package

Example:
Bank account class

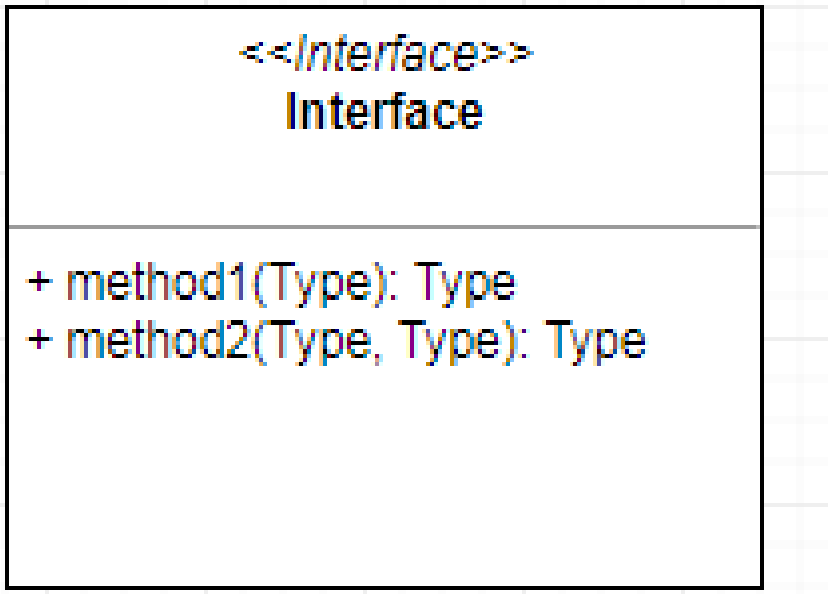
BankAccount
-owner : String -balance : Double = 0.0
+deposit (amount : Double) -withdraw (amount : Double)

Perspectives of Class Diagram



- **Conceptual**: represents the concepts in the domain
- **Specification**: focus is on the interfaces of Abstract Data Type (ADTs) in the software
- **Implementation**: describes how classes will implement their interfaces

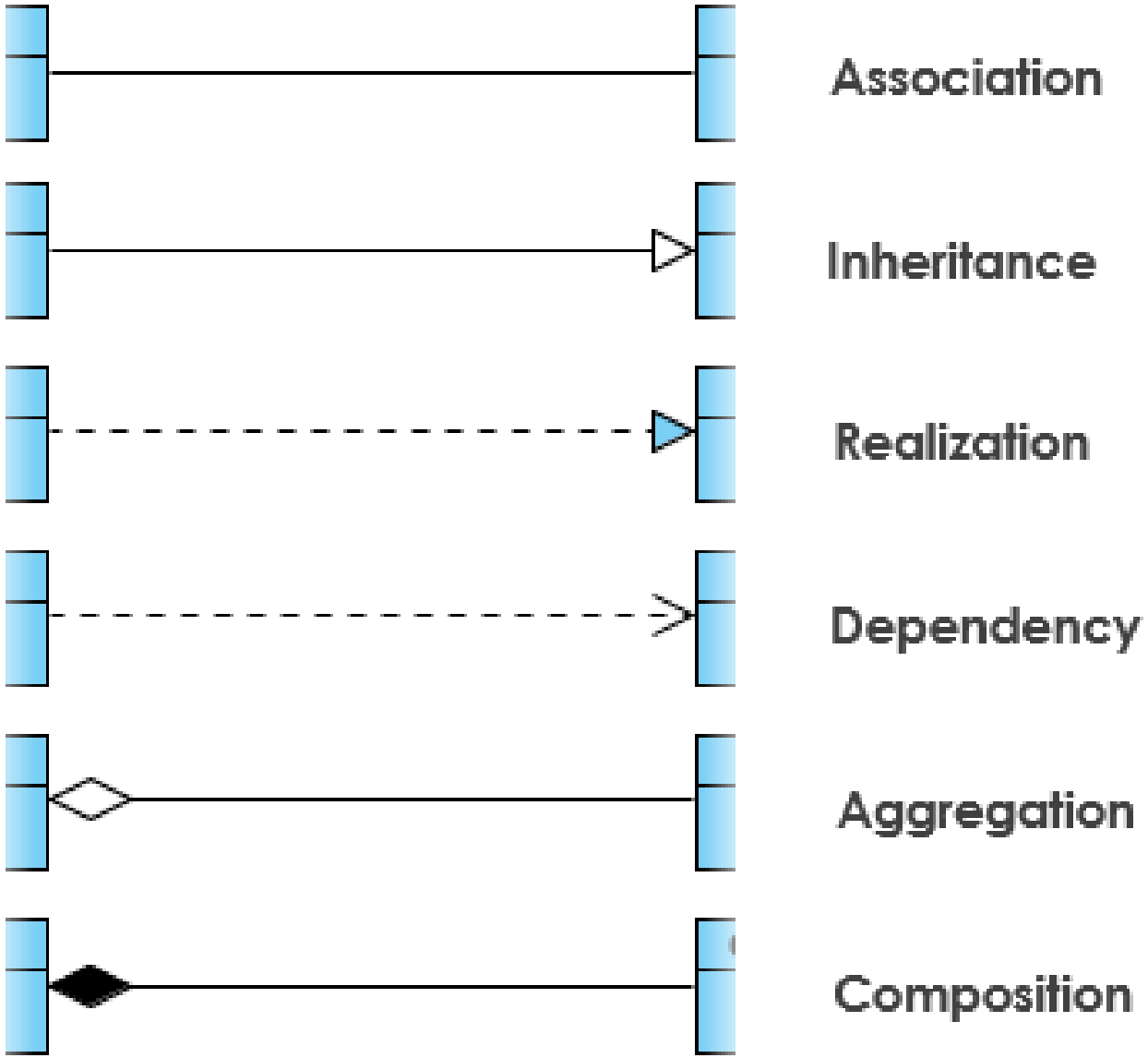
Interface



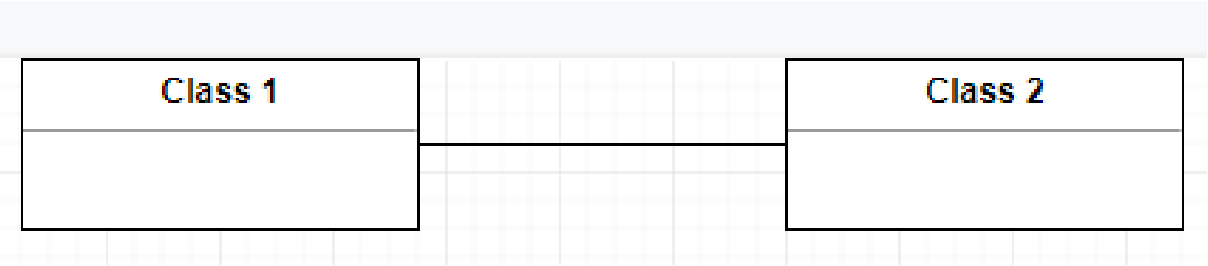
Abstract Class

Interface
vs
Abstract Class
???

Relationships between classes



Association

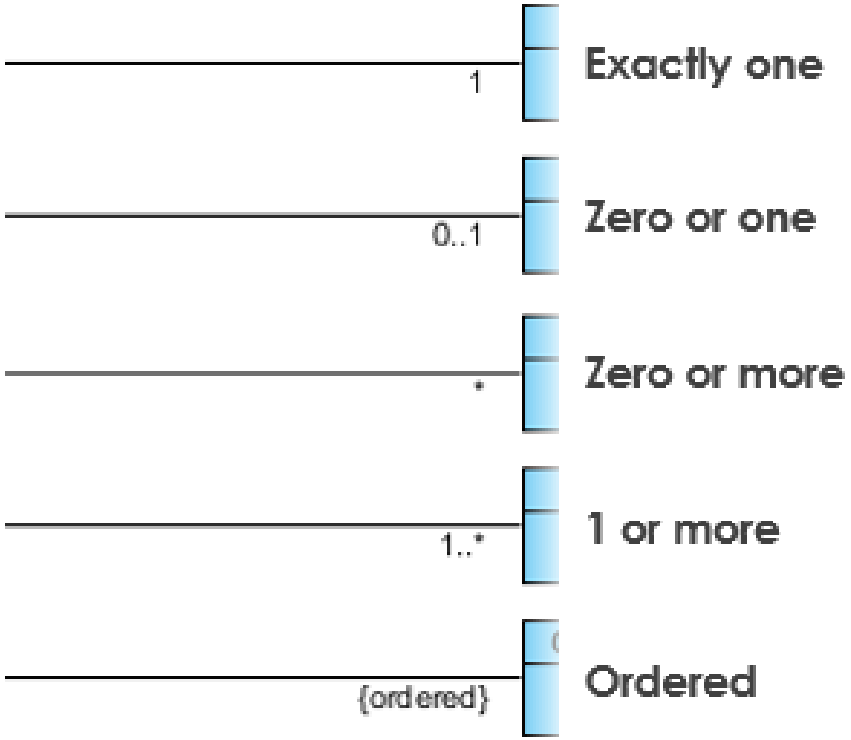


There is an association between Class1 and Class2

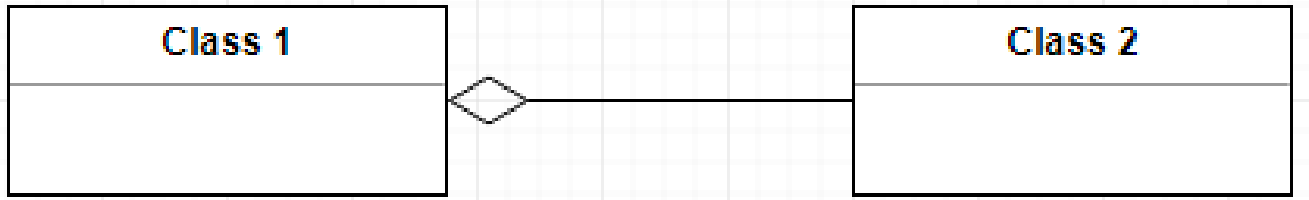
Simple Association

- A structural link between two peer classes.

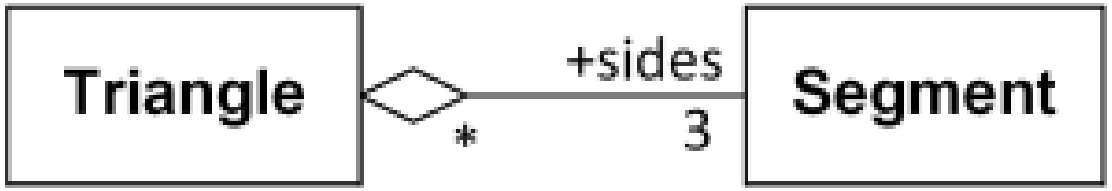
Cardinality



Aggregation



- A special type of association.
- It represents a "part of" relationship.
- Class2 is part of Class1.
- Many instances (denoted by the *) of Class2 can be associated with Class1.
- Objects of Class1 and Class2 have separate lifetimes.



Triangle has 'sides' collection of three line Segments. Each line Segment could be part of none, one, or several triangles.

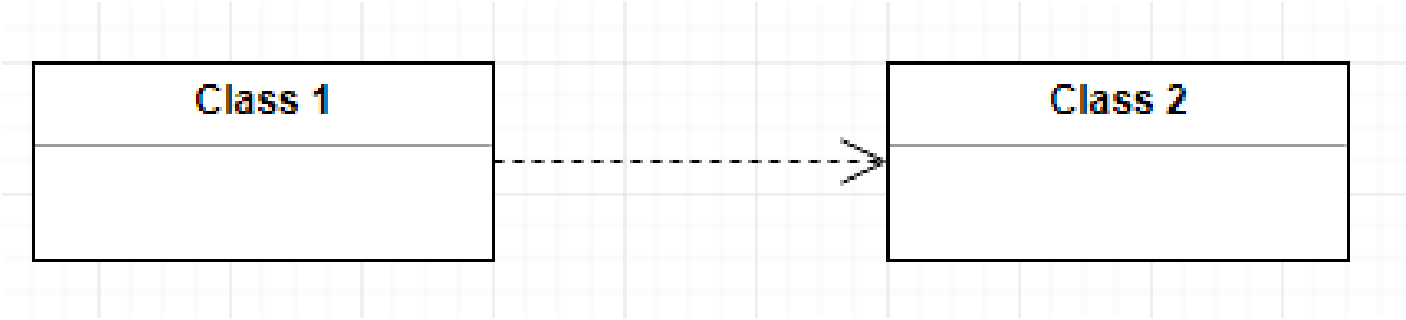
Composition

- A special type of aggregation where parts are destroyed when the whole is destroyed.
- Objects of Class2 live and die with Class1.
- Class2 cannot stand by itself.

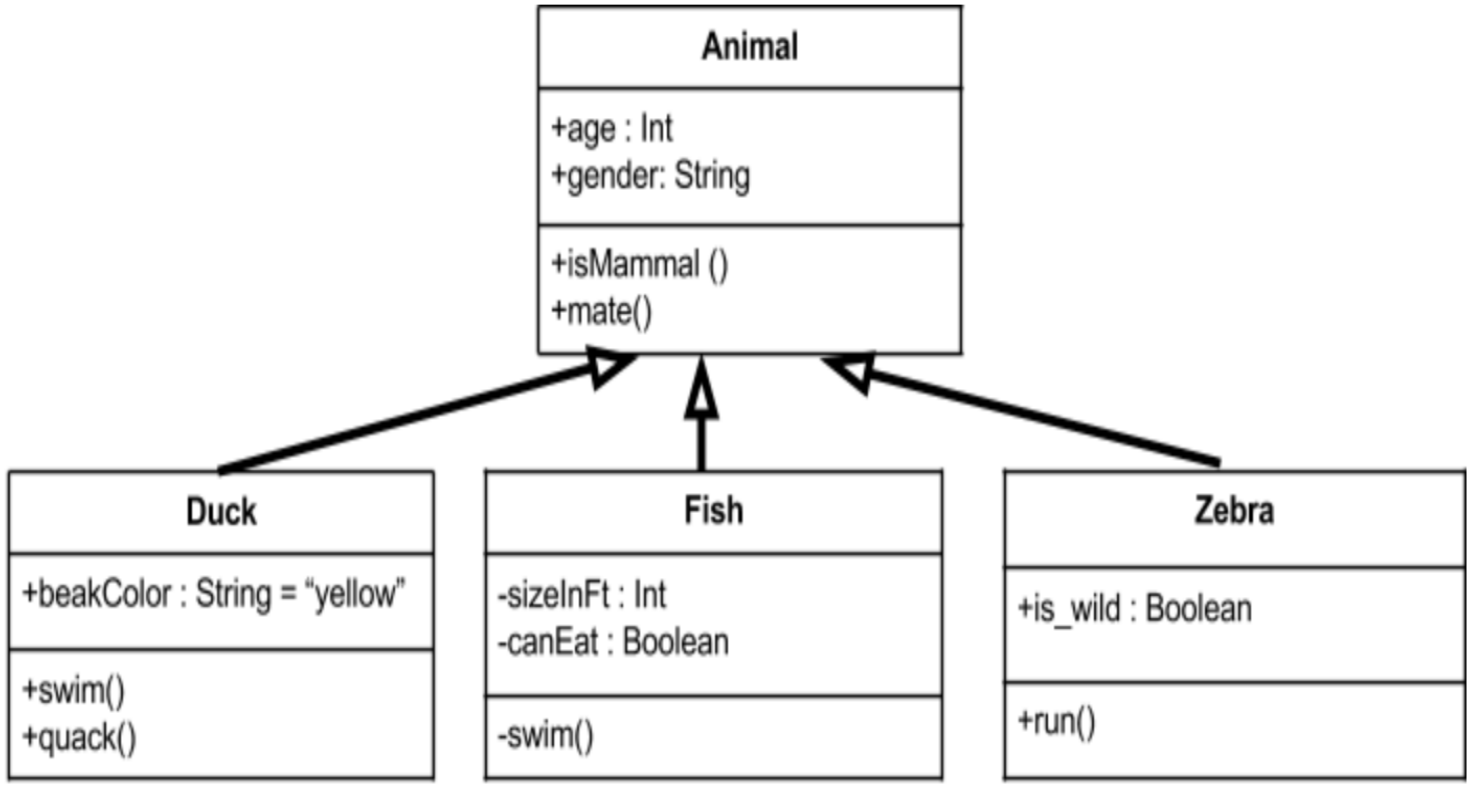
Folder could contain many files, while each File has exactly one Folder parent.
If Folder is deleted, all contained Files are deleted as well.

Dependency

- A special type of association.
- Exists between two classes if changes to the definition of one may cause changes to the other (but not the other way around).
- Class1 depends on Class2

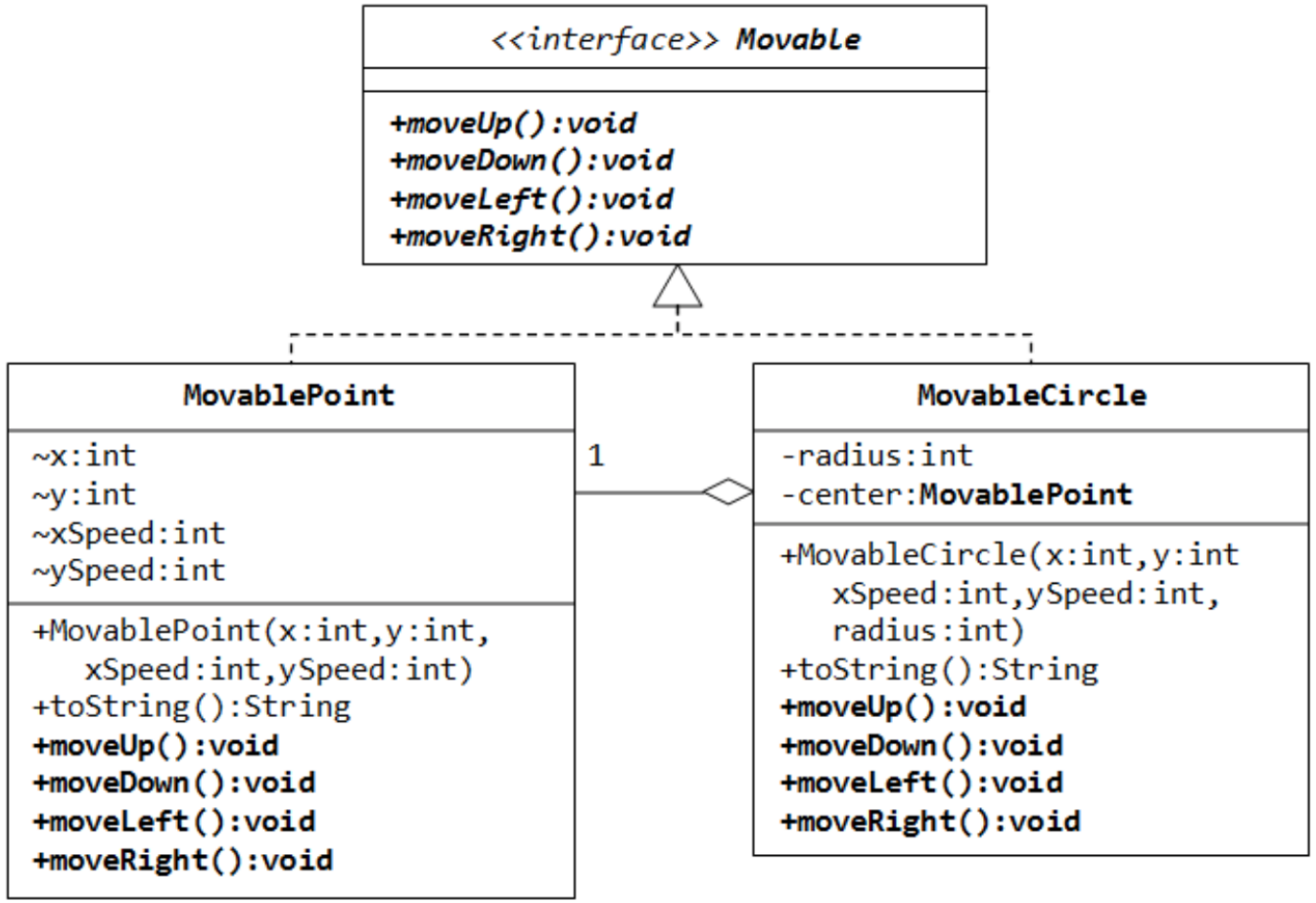


Inheritance (or Generalization)



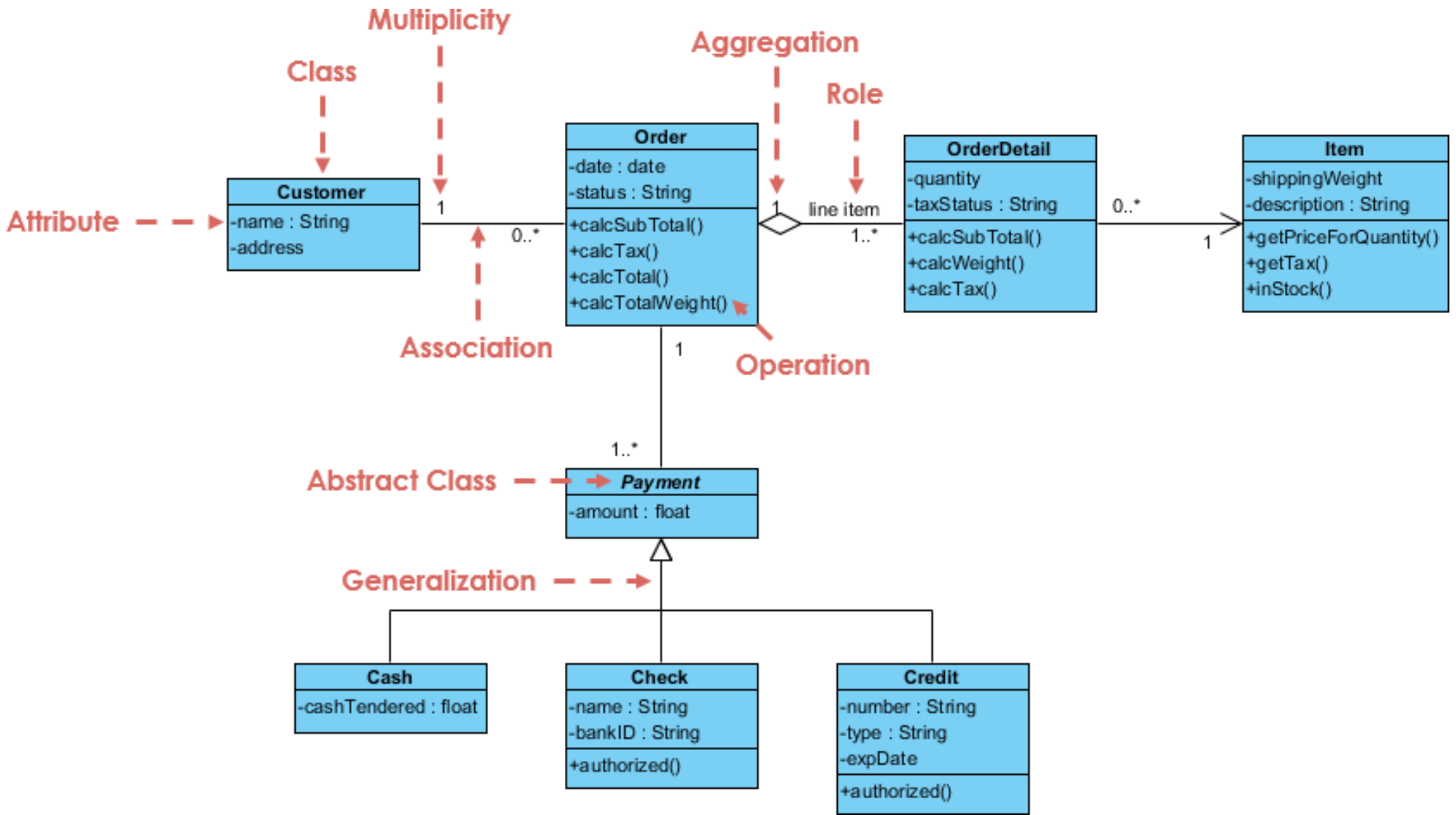
Realization/Implementation

Realization is a specialized abstraction relationship between two sets of model elements,
one representing a specification (the supplier) and
the other represents an implementation of the latter (the client).

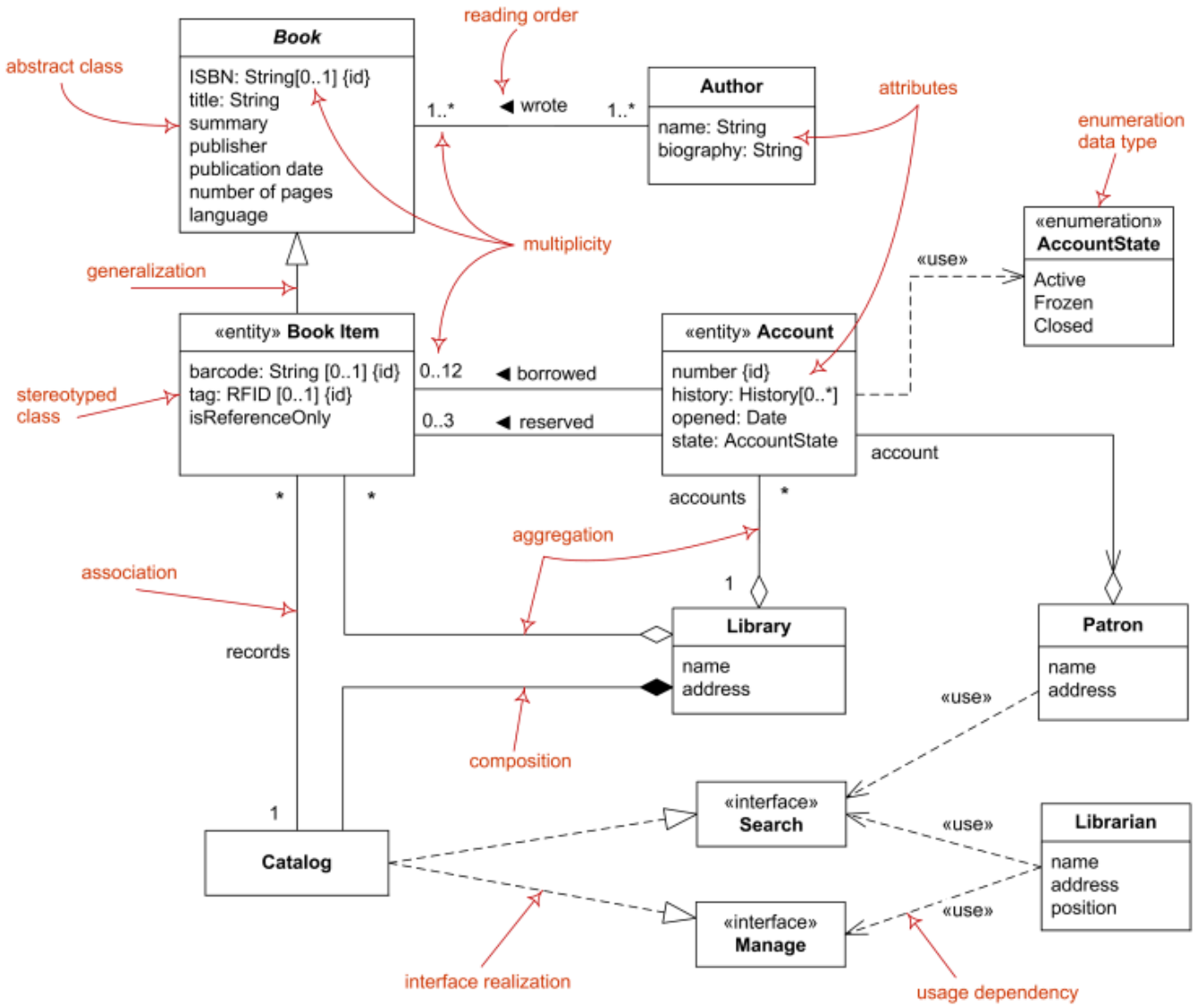


https://www.uml-diagrams.org/realization.html

Class Diagram Example: Order System

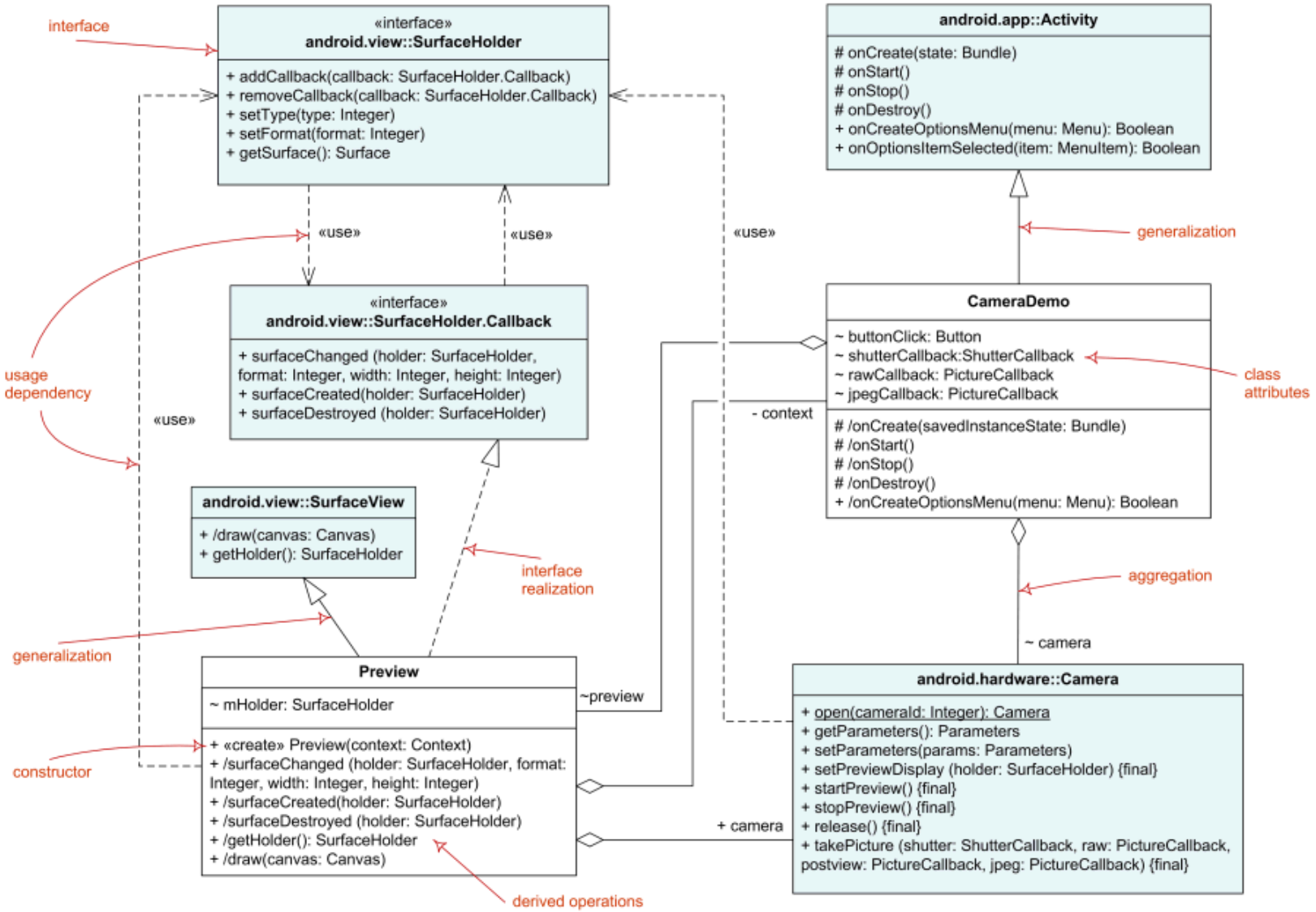


Domain Model Diagram



Domain diagram overview - classes, interfaces, associations, usage, realization, multiplicity.

Diagram of Implementation Classes



Elements of implementation class diagram - classes, interfaces, associations, usage, realization.

Class diagram ở cấp implementation

Design pattern

- Singleton pattern
- Proxy pattern
- Composite pattern
- Adapter pattern
- Facade pattern
- ….

http://blog.genmymodel.com/discover-five-design-patterns-in-one-uml-class-diagram-example.html

Note for Proj #4

- Module interface: dùng cách định nghĩa <<interface>> hoặc liệt kê danh sách các "method signature" cho từng module.
- Class diagram: cần phải có đủ entity/data class và control/business class và view/boundary class
- Method desc: cho tất cả method trong class diagram
- Sequence diagram, activity diagram & state-chart diagram : ở mức chi tiết (tên hàm)
- Design pattern: mô tả lại "chỗ nào"/"cái nào"/"cụm class nào" dùng và tại sao.
- A working demonstration: by sequence of screens