

UML (Unified Modeling Language)

Notes by rajeeshvengalapurath@gmail.com

Class Notation

MyClass
+attribute1: int -attribute2: string #attribute1: Circle
+operation1(in p1: boolean, in p2): String -operation2(inout p3: int): float #operation3(out p6): Circle* (* denotes a pointer)

Class Visibility

- + denotes public attributes or operations
- - denotes private attributes or operations
- # denotes protected attributes or operations

Classifier

A classifier is a category of Unified Modeling Language (UML) elements that have some common features, such as attributes or methods.

Types of UML classifiers

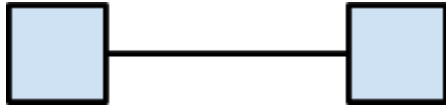
- Class
- Component
- Datatype
- Interface
- Node
- Signal
- Subsystem
- Use Case

Parameter Directionality

in, out or inout which specifies its direction with respect to the caller.

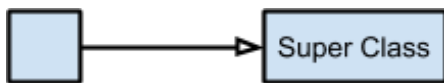
Relationships between classes

Association



Associations are relationships between classes in a UML Class Diagram. They are represented by a solid line between classes. Associations are typically named using a verb or verb phrase which reflects the real world problem domain.

Inheritance (Generalization)



It indicates that one of the two related classes (the subclass) is considered to be a specialized form of the other (the super type) and the superclass is considered a Generalization of the subclass.

Realization

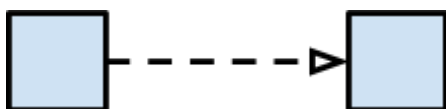
Client (one model element) implements the responsibility specified by the supplier (another model element). It is mostly found in the interfaces.

Interface Realization

In interface realization relationship, realizing classifiers conforms to the contract defined by the interface.

Types of realization

Canonical form



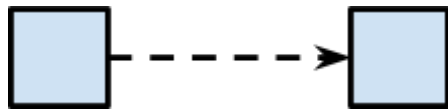
In this, the realization relationship is represented by a dashed line with a hollow arrowhead, and the interface is implemented using an object.

Elided form



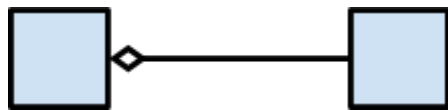
It is that kind of realization relationship in which the interface is represented by a circle, also known as a lollipop notation.

Dependency



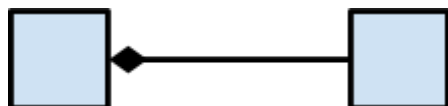
Dependency is a weaker form of bond that indicates that one class depends on another because it uses it at some point in time. One class depends on another if the independent class is a parameter variable or local variable of a method of the dependent class

Aggregation



Aggregation can occur when a class is a collection or container of other classes, but the contained classes do not have a strong lifecycle dependency on the container. The contents of the container still exist when the container is destroyed.

Composition



Composition can occur when a class is a collection or container of other classes, and the contained classes have a strong lifecycle dependency on the container. The contents of the container cannot exist when the container is destroyed.

Aggregation vs Composition

Aggregation	Composition
When the container is destroyed, the contents are usually not destroyed	When the container is destroyed, the contents are also destroyed

e.g. a professor has students; when the professor dies the students do not die along with them.

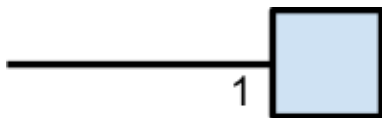
e.g. a university and its departments

Cardinality

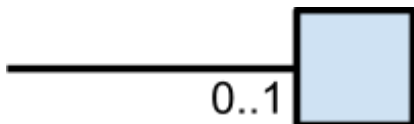
Cardinality is expressed in terms of:

- one to one
- one to many
- many to many

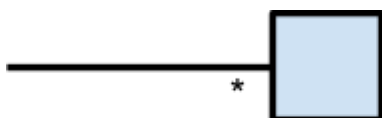
Exactly one



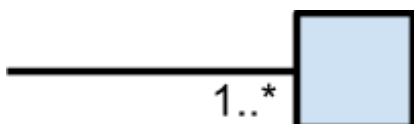
Zero or one



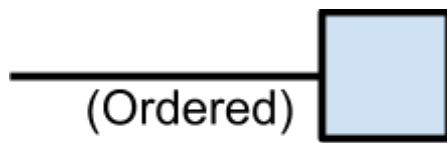
Zero or more



One or more



Ordered



Association vs Dependency

Association	Dependency
One object has the other object as a field/property/attribute.	An object accepts another object as a method parameter, instantiates, or uses another object.
A has a C object (as a member variable)	A references B (as a method parameter or return type)

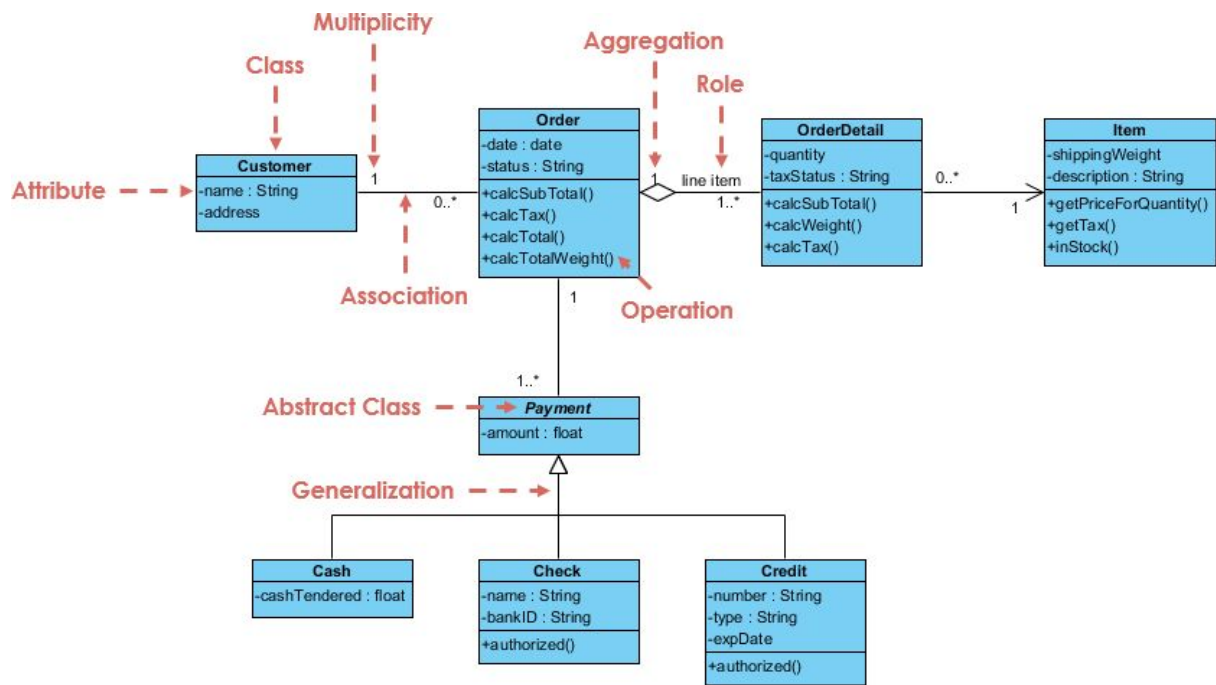
Abstract Classes

The abstract class has the same notation as that of the class. The only difference between a class and an abstract class is that the class name is strictly written in an italic font.

Class Diagram Example: Order System

Credits:

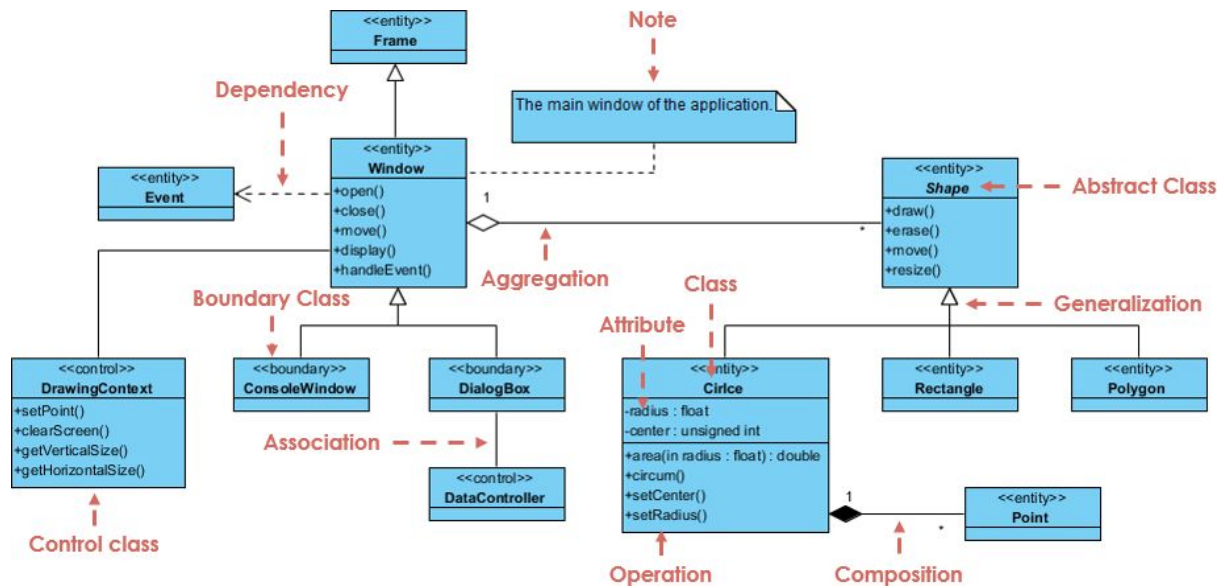
<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>



Class Diagram Example: GUI

Credits:

<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>



EoF