UML Class Diagrams

EECS 348 Lab 6 — 3/6/2025

Harlan Williams

Electrical Engineering and Computer Science
University of Kansas

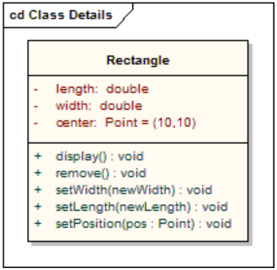
What are class diagrams for?

- Typically for object-oriented code
  - Straightforward in languages with syntax for explicitly designing classes
  - In languages such as C, what counts as a class is more up to the programmers
- Documents the *structure* of a program—specifically what classes there are and how they are intended to interact with each other
  - A class diagram should roughly correspond one-to-one with class definitions and interactions written in actual code
- Intended for communicating to other programmers

Classes

Classes are drawn as boxes with lines separating the **name** of the class, the names of its member **variables**, and the names of its member **functions**

- If a class is **abstract**, *i.e.,* it is never instantiated, but it defines an interface that other classes inherit from, its name is in italics
- If a class member is **public**, it is prefixed with a '+'
- If a class member is **private**, it is prefixed with a '-'



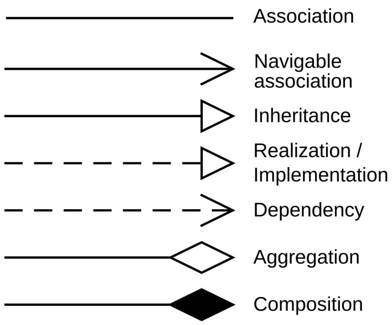
cd Class Details
Rectangle
- length: double
- width: double
- center: Point = (10,10)
+ display() : void
+ remove() : void
+ setWidth(newWidth) : void
+ setLength(newLength) : void
+ setPosition(pos : Point) : void

Source: https://sparxsystems.com/resources/tutorials/uml2/class-diagram.html

Relations

Lines are drawn between classes based on their **relations** to each other, and are read from **tail** to **tip**

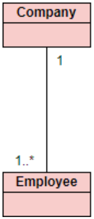
- An association means broadly that objects of the two classes interact
- Inheritance means that the tail specializes the class at the tip
- Aggregation and composition mean the tail class is contained within objects of the tip class
- A realization means that the tail class is a concrete implementation of the tip class



Source:
https://commons.wikimedia.org/wiki/File:Uml_classes_en.svg

Multiplicities

- A multiplicity gives how many occurrences of one class relate to another
- In composition or aggregation, this means how many of the tail class are contained in the tip class
- In the diagram below, a company has one or more employees **and** an employee has exactly one company

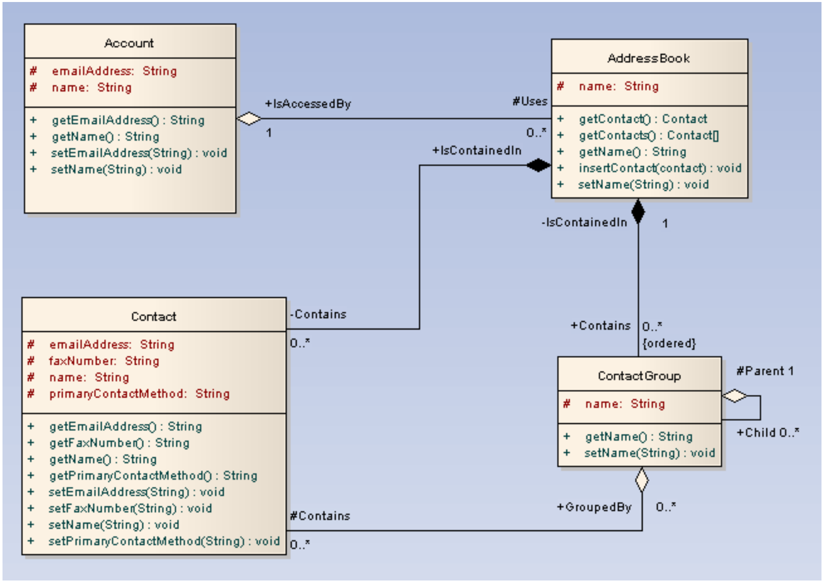


Multiplicities examples:

1	Exactly one, no more and no less
0..1	Zero or one
*	Many
0..*	Zero or many
1..*	One or many

Source: https://blog.visual-paradigm.com/what-is-multiplicity/

Composites and aggregates example



Source: https://sparxsystems.com/resources/tutorials/uml2/class-diagram.html

Assignment 4 hint—writing object-oriented C

Assignment 4 asks you to rewrite your max-heap email assignment in C and to provide a structural UML class or object diagram for it

Most C programmers write object-oriented code by treating source (`.c`) files as classes and defining their functions there, with data declarations in a corresponding header (`.h`) file. See lab 3 again for help with compiling code written this way

heap.h

```
 typedef struct Heap {
    Email *emails;
    size_t items;
} Heap;
Heap new_heap();
void push_heap();
void pop_heap();
```

heap.c

```
 Heap new_heap() {
    Heap heap;
    heap.items = 0;
    return heap;
}
```

more definitions...

Lost? See...

- For the lab
  - https://sparxsystems.com/resources/tutorials/uml2/class-diagram.html
  - https://online.visual-paradigm.com/
- For assignment 4
  - https://publications.gbdirect.co.uk/c_book/