Aggregation vs Composition

TAKE YOUR CODE INTO THE FUTURE
Find Modern Code tutorials and documentation from Intel.
intel
Software

I understand what composition is in OOP, but I am not able to get a clear idea of what Aggregation is. Can someone explain?

object-oriented

edited Aug 8 '14 at 17:08
gnat
21.8k 13 65 129

asked Mar 24 '11 at 4:25
vinoth
5,600 18 43 75

9 Answers

Simple rules:

1. A "owns" B = Composition : B has no meaning or purpose in the system without A
2. A "uses" B = Aggregation : B exists independently (conceptually) from A

Example 1:

A Company is an aggregation of People. A Company is a composition of Accounts. When a Company ceases to do business its Accounts cease to exist but its People continue to exist.

Example 2: (very simplified)

A Text Editor owns a Buffer (composition). A Text Editor uses a File (aggregation). When the Text Editor is closed, the Buffer is destroyed but the File itself is not destroyed.

edited Mar 25 '11 at 8:42
HorusKol
2,744 11 21

answered Mar 24 '11 at 14:57
Curtis Batt
2,546 1 7 13

6 So is a car an aggregate or a composition of its parts? – reinierpost Mar 31 '11 at 12:13

2 And how is aggregation different from any other relationship between two sorts of entities? – reinierpost Mar 31 '11 at 12:14

33 @reinierpost In *reality*, a car is an aggregation of parts, and parts are simply an aggregation of molecules... However, in a *model* it all depends on your requirements. Is it important to treat the engine as a separate entity so that you can track its lifetime independent of the car? Can you reuse the exact same engine in another car? If so, then you probably want aggregation. Otherwise you want a composition because you don't care about engines that aren't part of cars, nor do you care about reusing engines. – Curtis Batt Mar 31 '11 at 14:06

2 what is missing is an implementation example for a complete understanding... – Chesnokov Yuriy Oct 26 '13 at 10:04

1 What about employee when a company ceases to do business? Employee and people are different entities right? So can I say company is composition of employees? – arjun Jun 16 '15 at 2:13

From http://en.wikipedia.org/wiki/Object_composition

> Aggregation differs from ordinary composition in that it does not imply ownership. In composition, when the owning object is destroyed, so are the contained objects. In aggregation, this is not necessarily true. For example, a university owns various departments (e.g., chemistry), and each department has a number of professors. If the university closes, the departments will no longer exist, but the professors in those departments will continue to exist. Therefore, a University can be seen as a composition of departments, whereas departments have an aggregation of professors. In addition, a Professor could work in more than one department, but a department could not be part of more than one university.

So - while you have an ownership relationship with composition the owned object is also destroyed when the owner is - an aggregation (and the objects contained) can exist independently.

--

Update: Apologies - this answer is far too simplistic in hindsight.

c.batt provides an excellent definition in his answer: Aggregation vs Composition

edited Mar 24 '11 at 22:05

community wiki
3 revs, 2 users 94%
HorusKol

3 In the example you quote the composition is a one-to-many and the aggregation also has a one-to-many relationship implied, though here it could also be a many-to-many relationship for the aggregation (we can suppose possible that a teacher can teach in multiple departments). Whereas a department cannot be part of multiple universities. Composition implies ownership whereas aggregation does not go beyond relationship. The quote is correct but the comment is not. – Newtopian Mar 24 '11 at 5:20

1 it has nothing to do with destruction! UML does not define garbage collection system. – Display Name Mar 24 '11 at 10:01

1 i think the wikipedia link is getting reflexive upvotes, but this is a terrible definition - as @bold pointed out these relationships have nothing to do with GC. This also falls apart when an object is the component of two other objects, such as the ball in a ball-joint joining two artificial limbs. The Component relationship is about functional dependence. – Steven A. Lowe Mar 24 '11 at 14:13

1 I agree that my answer is severely lacking - but so is the WikiPedia article... – HorusKol Mar 24 '11 at 22:13

The difference between composition and aggregation is clear. The problem with aggregation is that it's unclear how it is different from ordinary association. – reinierpost Aug 16 at 12:20

There is no single explanation. Different authors mean different things by aggregation. Most don't really mean anything specific by it.

answered Mar 24 '11 at 9:58

reinierpost
469 5 8

3 This is the correct answer. I've read it in two books, one of them being Martin Fowler's *UML Distilled*. – davidhaskins Mar 25 '11 at 12:52

martinfowler.com/bliki/AggregationAndComposition.html – Jerry Andrews Jul 25 at 21:56

- Composition is an Association
- Aggregation is an Association
- Composition is a *strong* Association (If the life of contained object totally depends on the container object, it is called strong association)
- Aggregation is a *weak* Association (If the life of contained object doesn't depends on the container object, it is called weak association)

Example:

```
class Contained {
    public void disp() {
        System.out.println("disp() of Contained A");
    }
}

public class Container {
    private Contained c;

    //Composition
    Container() {
        c = new Contained();
    }
    //Association
    public Contained getC() {
        return c;
    }

    public void setC(Contained c) {
        this.c = c;
    }

    public static void main(String[] args) {
        Container container = new Container();
        Contained contained = new Contained();
        container.setC(contained);
    }
}
```

edited Jul 9 '13 at 8:33
Kilian Foth
65.6k 19 179 212

answered Jul 9 '13 at 7:24
Pawan
121 1 3

1 What is the difference between aggregation and association that is neither composition nor aggregation? – reinierpost Mar 24 '14 at 14:54

aggregation is a simple collection, like a bag of marbles

composition implies internal/functional dependencies, like the hinges on a box

cars aggregate passengers; they get in and out without breaking the car's functionality

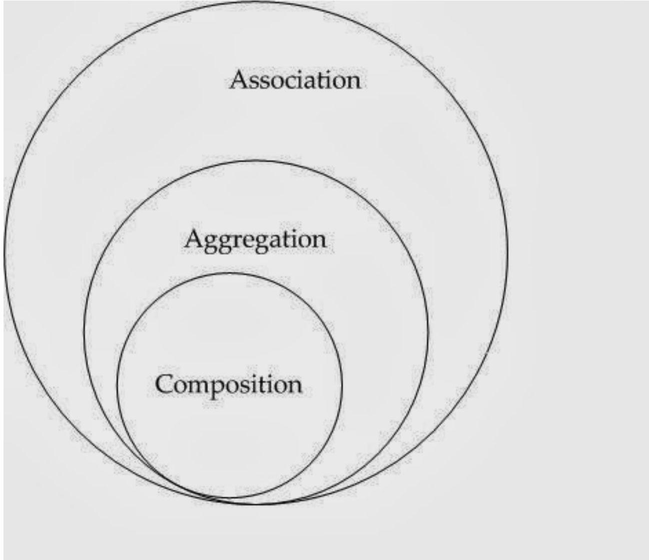
the tires are components; remove one and the car no longer functions correctly

[note: the spare tire is an aggregate!]

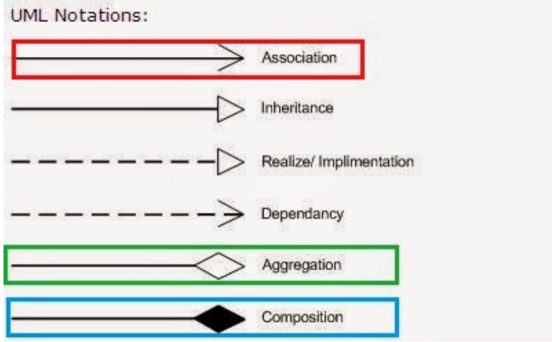
answered Mar 24 '11 at 14:09
Steven A. Lowe
31.8k 1 70 142

	Aggregation	Composition
Life time	Have their own lifetime	Owner's life time
Relation	Has	part-of
Example	Car has driver	Engine is part of Car

Both denotes relationship between object and only differ in their strength.



UML notations for different kind of dependency between two classes



Composition : Since Engine is part-of Car, relationship between them is Composition. Here is how they are implemented between Java classes.

```
public class Car {
    //final will make sure engine is initialized
    private final Engine engine;

    public Car(){
       engine  = new Engine();
    }
}

class Engine {
    private String type;
}
```

Aggregation : Since Organization has Person as employees, relationship between them is Aggregation. Here is how they look like in terms of Java classes

```
public class Organization {
    private List employees;
}


public class Person {
    private String name;
}
```

Source

edited Jan 19 at 8:37

answered Jul 1 '15 at 5:28

Premraj
360 2 12

this doesn't seem to offer anything substantial over points made and explained in prior 12 answers – gnat Jul 1 '15 at 7:05

Pretty sure books can exist without libraries. Bad example! – T Blank Jan 4 at 15:43

Here the List of Employees is the part of Organization Object. How can this be aggregation ? – Salman Muhammad Ayub Feb 12 at 9:59

How is association different from aggregation? – reinierpost Aug 16 at 12:22

I always look at composition as 'needs a', i.e. a car *needs an* engine, and I look at aggregation as 'things related for a purpose'. So staying with the car analogy, my aggregation may be to represent a journey which may involve bringing a car and passengers together. The journey does not own the car or the passengers, I'm aggregating data that is related for a specific scenario. When the journey is completed the car and the passengers go on. When a car is ended, the car and it's engine are normally destroyed together.

answered Mar 24 '11 at 9:52

Lazarus
371 2 6

Semantically, all sets are made of subsets, right? Therefore:

- The aggregation is when those subsets exists independently of the father set. As a monitor can be unplugged from the computer to be connected to another.
- The composition is when those subsets depends of the existence of the father set. As a leaf is a part of a tree or liver is a part of a body.

These concepts talks about the kind of dependency between two objects or classes, conceptually. Directly in a program, in an aggregation, when the father object disposes, the aggregate objects should be disposed too. In the same scenario for a composition, composite son objects will persist then the father object dispenses.

edited Aug 10 '14 at 1:31
gnat
21.8k 13 65 129

answered Aug 8 '14 at 18:55
Negarrak
21 2

How about this simple example:

An array of objects is a composition. An array of pointers to objects is an aggregation.

If I delete the first one, its contents vanish with it. The second one, on the other hand can vanish without affecting its members existence unless there is a specific method that deletes each object as its pointer is deleted.

answered Sep 5 '14 at 14:03
user2831074
11

3 this doesn't seem to add anything substantial over points made and explained in prior 11 answers – gnat Sep 5 '14 at 16:23

protected by gnat Sep 5 '14 at 16:22

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?