Aggregation, Association & Composition

This article talks about Association, Aggregation and Composition Relationships between classes with some C++ examples.

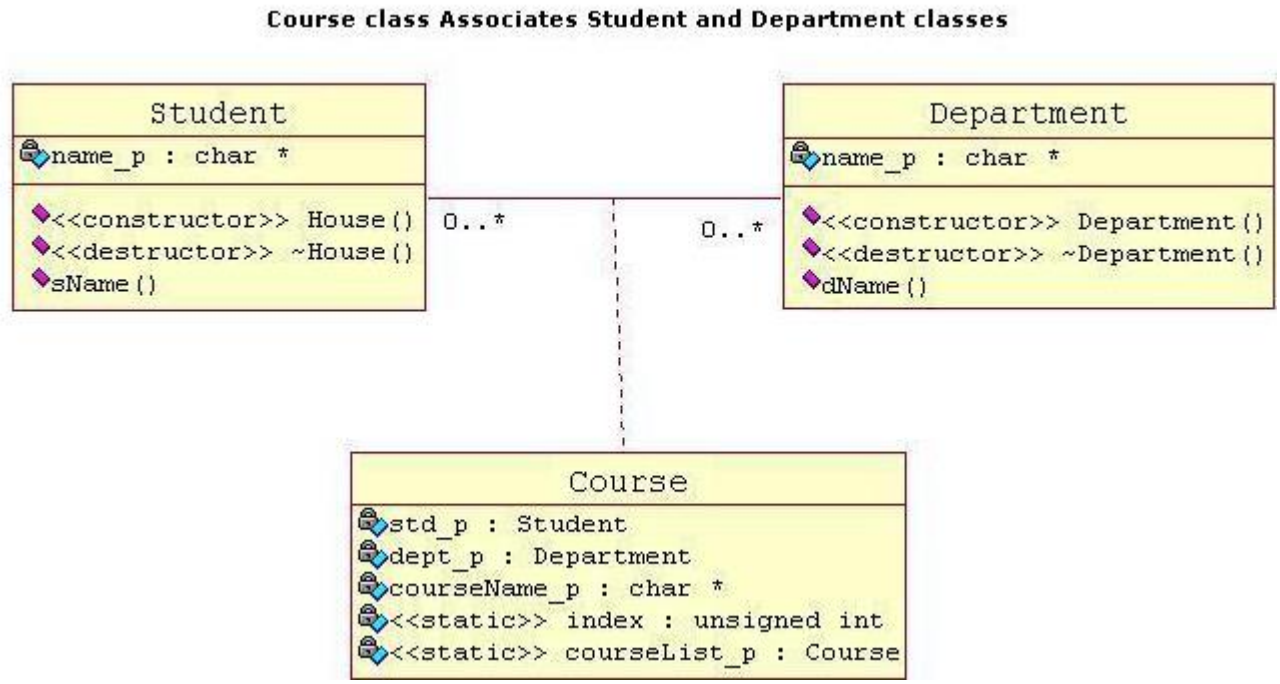
Background

Association is a simple structural connection or channel between classes and is a relationship where all objects have their own lifecycle and there is no owner.

Lets take an example of Department and Student.

Multiple students can associate with a single Department and single student can associate with multiple Departments, but there is no ownership between the objects and both have their own lifecycle. Both can create and delete independently.

Here is respective Model and Code for the above example.



Aggregation, Association & Composition

Aggregation is a specialize form of Association where all object has their own lifecycle but there is a ownership like parent and child. Child object cannot belong to another parent object at the same time. We can think of it as "has-a" relationship.

Implementation details:

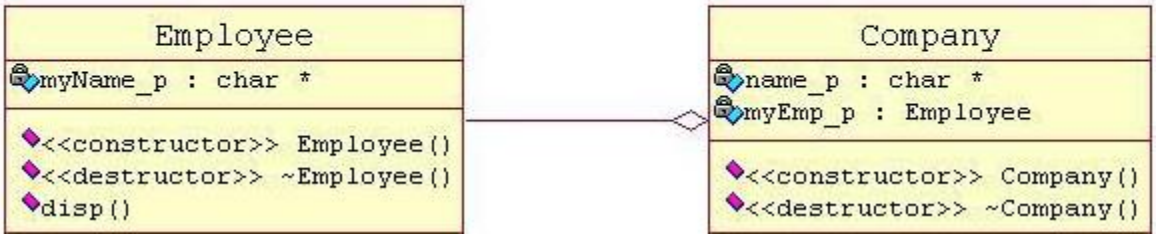
1. Typically, we use pointer variables that point to an object that lives outside the scope of the aggregate class
2. Can use reference values that point to an object that lives outside the scope of the aggregate class
3. Not responsible for creating/destroying subclasses

Let's take an example of Employee and Company.

A single Employee can not belong to multiple Companies (legally!!), but if we delete the Company, Employee object will not destroy.

Here is respective Model and Code for the above example.

Employee class has Agregation Relatioship with Company class



Aggregation, Association & Composition

Composition is again specialize form of Aggregation. It is a strong type of Aggregation. Here the Parent and Child objects have coincident lifetimes. Child object dose not have it's own lifecycle and if parent object gets deleted, then all of it's child objects will also be deleted.

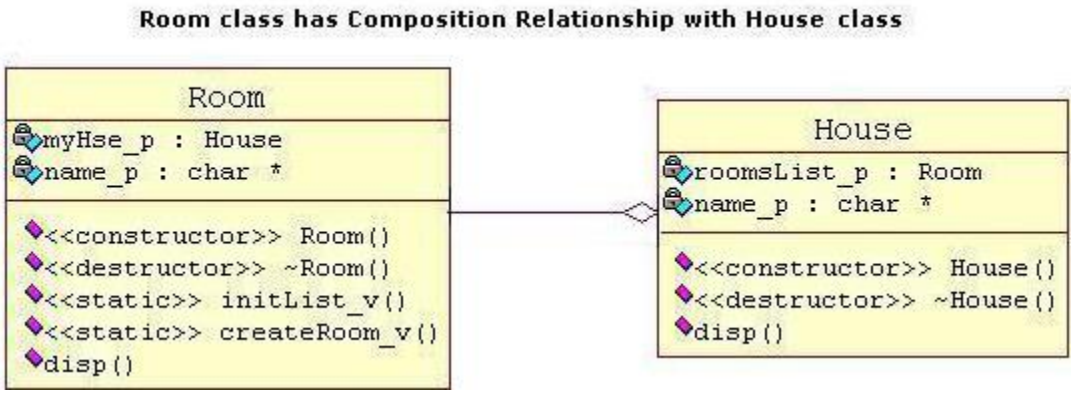
Implantation details:

1. Typically we use normal member variables
2. Can use pointer values if the composition class automatically handles allocation/deallocation
3. Responsible for creation/destruction of subclasses

Lets take an example of a relationship between House and it's Rooms.

House can contain multiple rooms there is no independent life for room and any room can not belong to two different house. If we delete the house room will also be automatically deleted.

Here is respective Model and Code for the above example.



Aggregation, Association & Composition

These relationships can also be explained with simple examples as below (these lines I have read in some blog):

Association:

1. Create a folder called "Links"
2. Create a shortcut/link inside this folder and link it to www.go4expert.com
3. Create another shortcut/link instide this folder and link it to www.google.com
4. Ask your friend to do the same on another machine using same links (www.go4expert.com and www.google.com)
5. Delete the "Links" folder, and open your browser to check if www.go4expert.com and www.google.com still exist or not

Briefly, Association is a relationship where all the objects have different lifecycles. there is no owner.

Aggregation:

1. Create a file called file.txt
2. Make a simple Application to open the File.txt (rw mode), but don't program it close the connection.
3. Run an instance of this application (it should work ok and can open the file for rw)
4. Keep the first instance, and run another instance of this application (In theory it should complain that it can't open the file in rw mode because it is already used by other application).
5. Close the 2 instances (make sure you close the connection).

From the above application, we understand that the Application and the File has a separate lifecycles, however this file can be opened only by one application simuletanously (there is only one parent at the same time, however, this parent can move the child to another parent or can make it orphan).

Composition:

1. Open a new Document name it as test.txt
2. Write this sentence inside this document "This is a composition".
3. Save the document.
4. Now, delete this document.

This is what is called composition, you can't move the sentence "This is a omposition" from the document because its lifecycle is linked to the parent (i.e. the document here !!)

Download Code:
https://www.go4expert.com/articles/association-aggregation-composition-t17264/