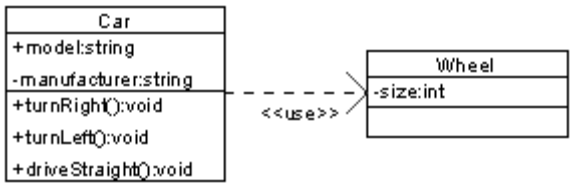
# Object Orientated Class Relationships

**Generalisation & Inheritance**
Generalisation is the process of extracting two characteristics from two different classes into a superclass that can then be used within the program.

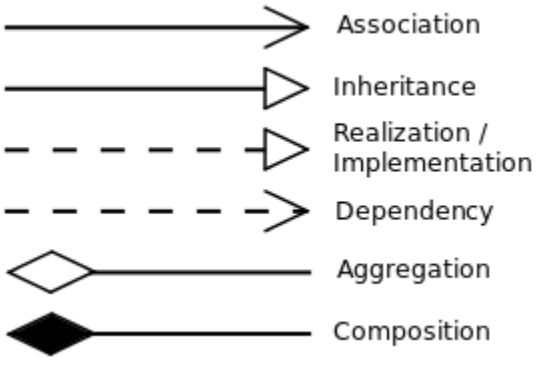
Inheritance is when a object takes characteristics and attributes from parent object to use within their own objects. The object that the characteristics are being taken from is called the parent object and the object that is taking the characteristics is called the child class which is why is it called inheritance.

**Dependency**
Dependency is a type of relationship between different classes that make sure that certain objects can not be implemented without the required elements being present from one class or another. Ana example of this would be a car and a wheel. The wheel is dependant on the car itself so the wheel can not work without the car.




**Loose & Tight Coupling**
Loose coupling is when the number of dependencies that a class uses will be reduced to increase the reusability and to make it easier to maintain if something was to go wrong. Tight coupling is the opposite such as having multiple objects depend on each other to the point where when one class has been edited the other class will cease to function and will have to be edited too.

**Realisation**
Realisation is the connection between classes within a UML diagram. The different type of relationship between two or more classes will determine how the information is shared or transferred between those classes. An example of this would be the image below:




**Aggregation/Composition**
Aggregation and composition are similar because aggregation and composition both have a relationship where there is a child and a parent but aggregation allows the child to be independent from the parent class. Where as composition must exist with the parent class.

**Low & High Cohesion**
Cohesion is how much of the elements of a system are related to each other. A high cohesion means that the elements in a program are very well connected to one another meaning each one has something in common with each other. A low cohesion means that the elements in a program and very loosely connected which can lead to poor maintenance and execution of the program.

*IBM® IBM Knowledge Center* [online]. (2016). Available from: <https://www.ibm.com/support/knowledgecenter/SS5JSH_9.1.2/com.ibm.xtools.modeler.doc/topics/caggreg.html>. [Accessed 10/12/2018].

*Net Stuff* [online]. (2015). Available from: <http://www.dotnet-stuff.com/tutorials/c-sharp/understanding-loose-coupling-and-tight-coupling>. [Accessed 10/12/2018].