OBJECT-ORIENTATED PROGRAMMING

Advantages

- programs written in modules, therefore it is easier to amend code and split it between members of a team
- inheritance allows the reuse of code throughout the program
- libraries can be created enabling code to be reused easily

Encapsulation

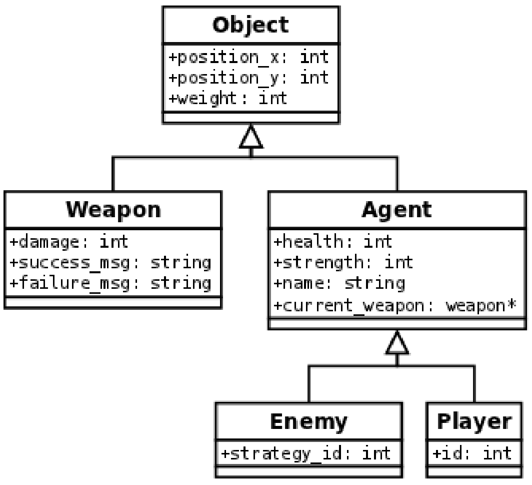
The concept of putting properties, methods and data in one object

- Method: the code or routine contained within a class
- Properties: the defining features of an object or class in terms of its data
- Class: blueprint or master copy that defines a related group of things. It contains properties and methods, but it does not store any data.
- Object: is an instance of a class so will have the same properties and methods from the class which it is built. It will also contain the data on which the methods will be run.
- Instantiation: the process of creating an object from a class

Inheritance

Concept that properties and methods in one class can be shares with a subclass

Object is the base class, super class or parent class. Weapon and agent are subclasses, derived classes or child classes



Class diagrams represent relationships and inheritance between classes. In class diagrams:

+ public properties and methods

- private properties and methods (only that class)

protected properties and methods (that class and subclasses)

→ pointing to the base class to show inheritance

data types of each variable are defined

- Polymorphism: ability of different data types to be manipulated with the same method (overriding of functions)
- Overriding: where a method described in a subclass takes precedence over a method with the same name in the base class

Abstract, virtual and static methods

- Static: the method can be used without an object of the class being instantiated
- Virtual: method is defined in the base class but can be overridden by the method in the subclass where it will be used.
- Abstract: the actual method is not supplied in the base class, which means that it must be provided in the subclass.

Aggregation: method of creating new objects that contain existing objects, based on the way in which objects are related.

- Composition aggregation: the objects contained will *cease to exist* if the containing object is destroyed
- Association aggregation: the objects contained will *continue to exist* even if the containing object is destroyed