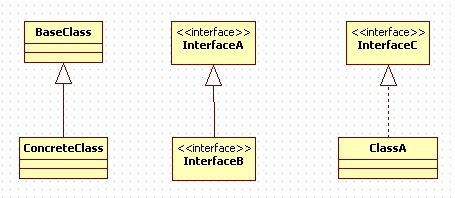
类图使用详解

本篇会讲解在UML类图中，常见几种关系: 泛化（Generalization），依赖(Dependency)，关联（Association)，聚合（Aggregation），组合(Composition)。

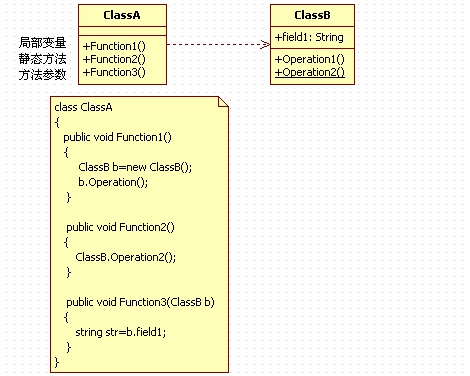
1.泛化关系

泛化关系是继承或实现的关系，是is a关系，具体表现为类与类的继承，接口与接口的继承，类对接口的实现关系。



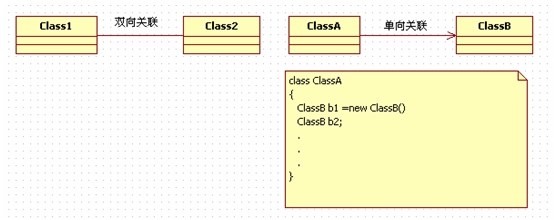
2.依赖关系

依赖关系表示为一个类使用另一个类，这种使用关系是具有偶然性的、临时性的、非常弱的，一个类的变化会影响到另一个类，是use a关系，如果类A依赖于类B,那么类B可以是类A的局部变量，或类A方法的参数，或静态方法的调用。



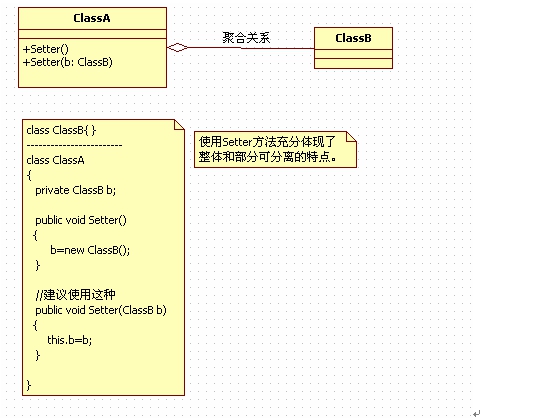
3.关联关系

关联关系是一种强依赖关系，这种关系不存在依赖关系的偶然性，关系也不是临时的，是长期的，稳定的。双方的关系是平等的，可以单向关联也可以是双向关联。假如类A关联了类B,则类B是类A的全局变量（注意是全局变量，再看看上面的依赖关系），大多数关联都是单向关联，这比较容易维护，关于关联，在生活中我们常会说，类A持有类B的引用。



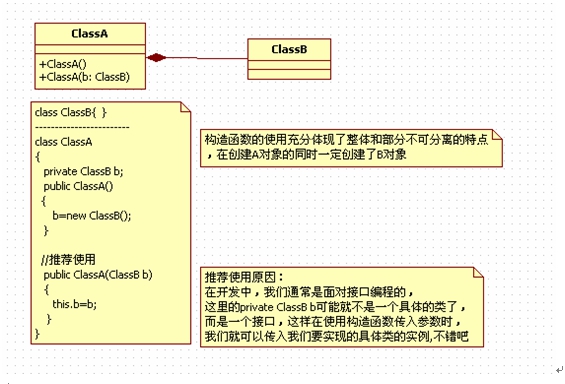
4.聚合关系

聚合关系是特殊的关联关系，是一种强的关联关系，他体现的是整体与部分关系，即has-a的关系，但是整体和部分是可以分离的，注意，是可以分离的。普通关联关系的两个类处于同一层次上，是平级的，而聚合关系的两个类处于不同的层次，一个是整体，一个是部分。同时，是一种弱的“拥有”关系。体现的是A对象可以包含B对象，但B对象不是A对象的必要的组成部分。具体表现为，如果A由B聚合成，表现为A包含有B的全局对象，但是B对象可以不在A创建的时刻创建，这句话非常有意义，它在代码中通常体现成依赖注入的setter方法，即A对象可以随时创建B对象，再想想这不就体现了整体和部分是可以分离了吗？创建整体的时候可以不创建部分。



5.组合关系

组合关系也是特殊的关联关系，它体现一种contains a（拥有）关系，这种关系是比聚合还要强，也称为强聚合。体现了严格的整体和部分关系，两者是不可分割的，它们的生命周期是一致的。如果A由B组成，那么A就包含B的全局变量，并在创建A的同时创建B，在代码上我们通常是使用构造函数进行实现，也是依赖注入中构造函数的实现。



最后，我们来总结一下，泛化就不用多少了，大家都懂的，就是继承和实现接口，重点说下其它的吧。

依赖，Class A依赖于Class B,则Class B体现为Class A的局部变量，我想用就用，用了就有关系，不用就没关系；

关联，Class A关联了Class B，则Class B体现为Class A的全局变量，不管你用不用，反正你知道我的存在了，持有了我的引用；

聚合，Class A由Class B聚合，则Class B体现为Class A的全局变量，Class B对象的创建是可以不用随Class A对象创建而创建了。用了就加强了关系，不用还是我只知道你的存在。聚合可以方便的持有多个类的引用，如使用List<>，所以当你发现有List<>等集合是可以使用聚合来表示，比如观察者模式的结构。

组合，Class A由Class B组成，则Class B体现为Class A的全局变量，并在创建Class A对象的同时必须创建Class Bx的对象，体现最强的关系。比如人出身了，必定也有头部吧，不然我真无法想象这个世界了。

说下最基本的依赖注入方式吧，一种是setter方法或属性注入，如聚合关系中体现的，使用一个方法或属性来完成注入，另一种是构造函数注入，如组合关系中体现的，还有一种是接口注入，通过接口来实现信息的注入,而其它的类要实现该接口时,就可以实现了注入。以后会在以后详细讲解。