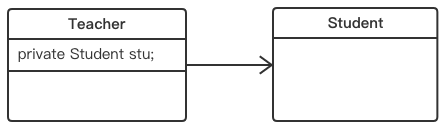
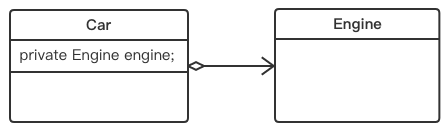
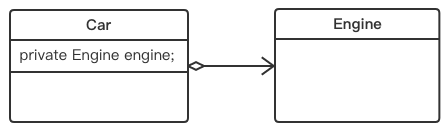
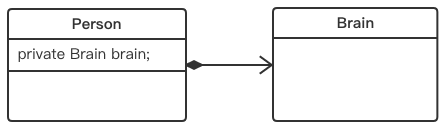
1. 设计模式的原则
   1. 总原则：开闭原则 对拓展开放，对修改关闭
   2. 单一职责原则： 每个类有单一的职责，如若不然，就把类拆分
   3. 里氏替换原则： 任何基类可以出现的地方，子类一定也可以出现（子类对基类实现的函数尽量不要重写和重载）
   4. 依赖倒转原则： 这个是开闭原则的基础，具体内容：面向接口编程，依赖于抽象而不依赖于具体。写代码时用到具体类时，不与具体类交互，而与具体类的上层接口交互。
   5. 接口隔离原则：每个接口中不存在子类用不到却必须实现的方法，如果不然，就要将接口拆分。使用多个隔离的接口，比使用单个接口（多个接口方法集合到一个的接口）要好。
   6. 迪米特法则（最少知道原则） ：一个类对自己依赖的类知道的越少越好。也就是说无论被依赖的类多么复杂，都应该将逻辑封装在方法的内部，通过public方法提供给外部。这样当被依赖的类变化时，才能最小的影响该类。
   7. 合成复用原则： 原则是尽量首先使用合成/聚合的方式，而不是使用继承。
2. 类的关系http://blog.51cto.com/sihai/2071497



* 1. 依赖：一个人(Person)可以买车(car)和房子(House)，那么就可以称：Person类依赖于Car类和House类。Person类里并没有使用Car和House类型的属性，Car和House的实例是以参量的方式传入到buy()方法中。



* 1. 关联：它使一个类知道另一个类的属性和方法。  
        关联可以是双向的，也可以是单向的。  
        在Java语言中，关联关系一般使用成员变量来实现。
  2. 
  3. 聚合： 聚合是关联关系的一种，是强的关联关系。  
        聚合是整体和个体之间的关系，但个体可以脱离整体而存在。
  4. 
  5. 组合： 组合是关联关系的一种，是比聚合关系强的关系，也以成员变量的形式出现。  
        在某一个时刻，部分对象只能和一个整体对象发生组合关系，由后者排他地负责生命周期。  
        部分和整体的生命周期一样。