类（Class）是指具有相同属性、方法和关系的对象的抽象，它封装了数据和行为，是面向对象程序设计（OOP）的基础，具有封装性、继承性和多态性等三大特性。在 UML 中，类使用包含类名、属性和操作且带有分隔线的矩形来表示。

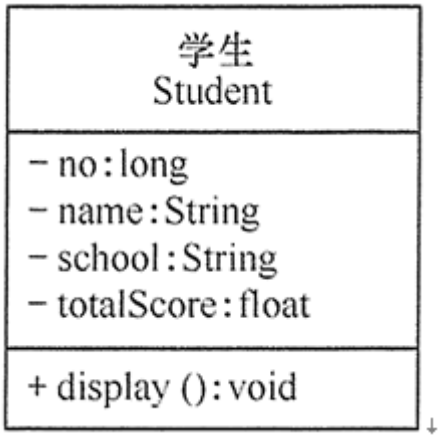
(1) 类名（Name）是一个字符串，例如，Student。

(2) 属性（Attribute）是指类的特性，即类的成员变量。UML 按以下格式表示：

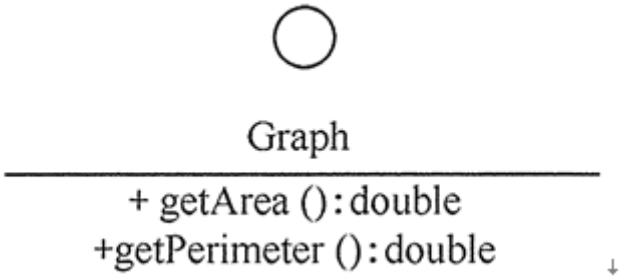
注意：“可见性”表示该属性对类外的元素是否可见，包括公有（Public）、私有（Private）、受保护（Protected）和朋友（Friendly）4 种，在类图中分别用符号+、-、#、~表示。

(3) 操作（Operations）是类的任意一个实例对象都可以使用的行为，是类的成员方法。UML 按以下格式表示：

例如：+display():void。

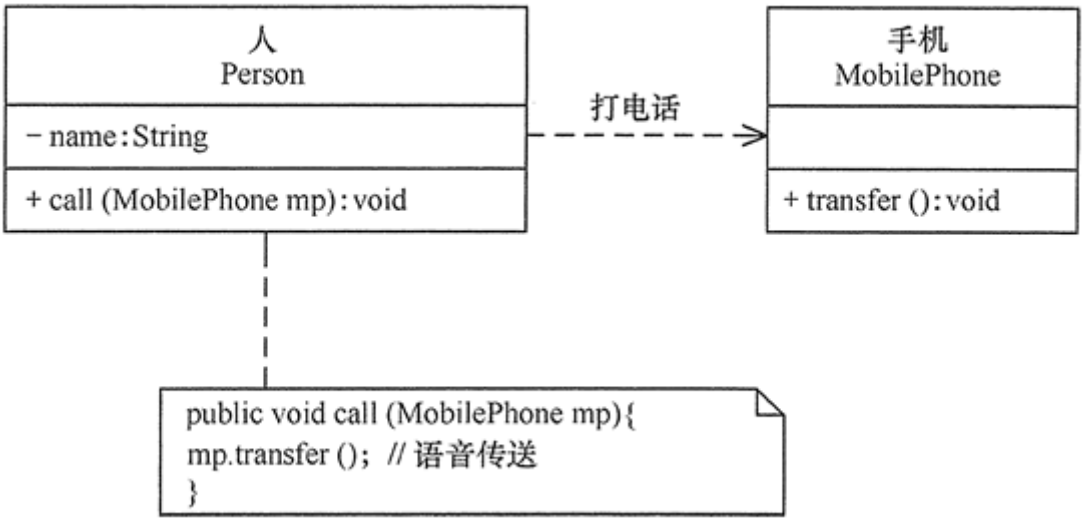


接口（Interface）是一种特殊的类，它具有类的结构但不可被实例化，只可以被子类实现。它包含抽象操作，但不包含属性。它描述了类或组件对外可见的动作。在 UML 中，接口使用一个带有名称的小圆圈来进行表示。



依赖关系 使用关系 对象之间耦合度最低的一种关联方式

类方法通过**局部变量,方法的参数或者对静态方法的调用来访问另一个类**

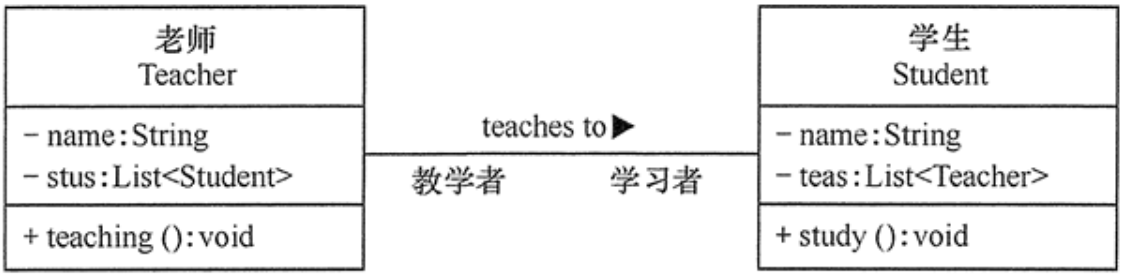


关联关系

对象之间的一种引用关系 用于表示一类对象与另一类对象之间的联系，如老师和学生、师傅和徒弟、丈夫和妻子等。分为一般关联关系、聚合关系和组合关系

关联可以是双向的，也可以是单向的。在 UML 类图中，双向的关联可以用**带两个箭头或者没有箭头的实线**来表示，单向的关联用**带一个箭头的实线**来表示，箭头从使用类指向被关联的类。

在代码中通常将一个类的对象作为另一个类的成员变量来实现关联关系。图 5 所示是老师和学生的关系图，每个老师可以教多个学生，每个学生也可向多个老师学，他们是双向关联。

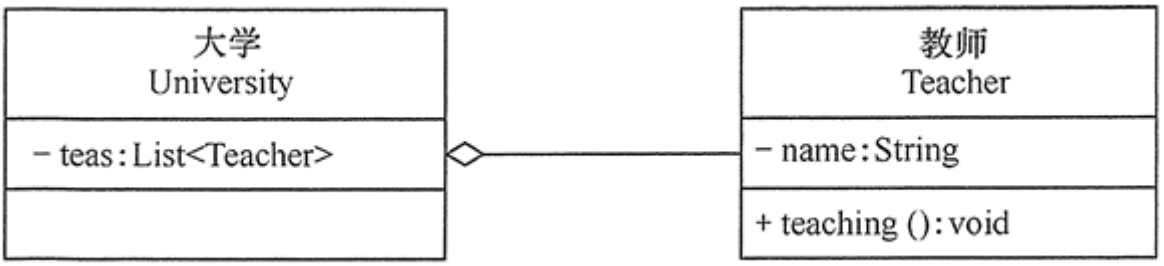


聚合（Aggregation）关系

聚合（Aggregation）关系是关联关系的一种，是强关联关系，是整体和部分之间的关系，是 has-a 的关系。

通过成员对象来实现的，其中成员对象是整体对象的一部分，但是成员对象可以脱离整体对象而独立存在

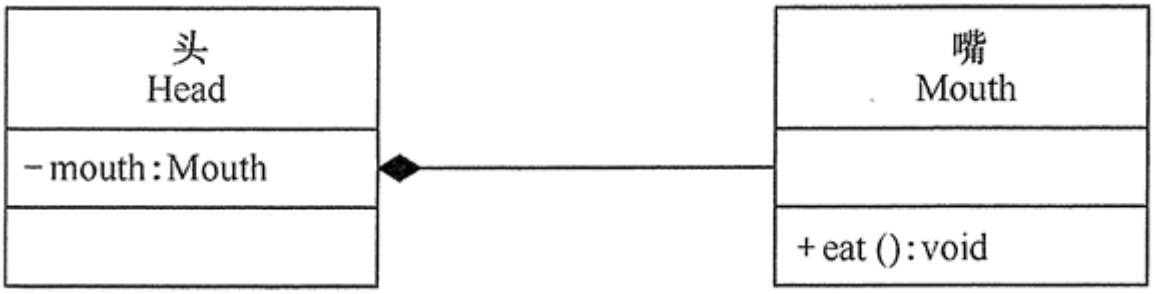
带空心菱形的实线来表示



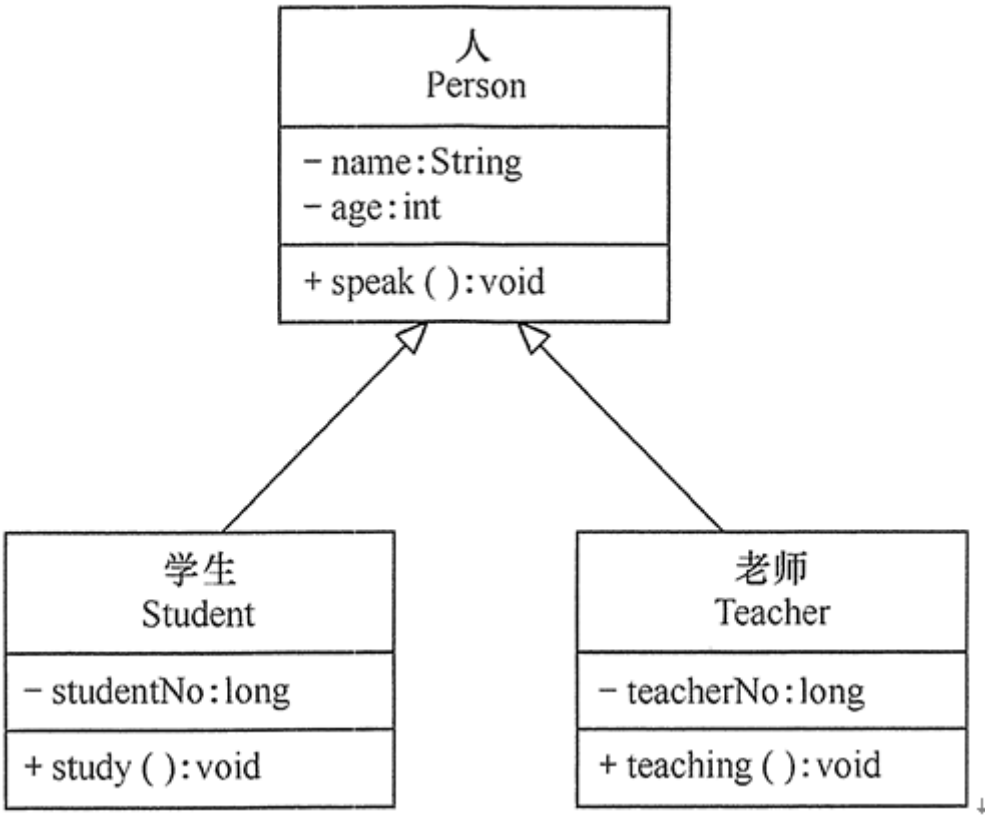
组合关系

组合（Composition）关系也是关联关系的一种，也表示类之间的整体与部分的关系，但它是一种**更强烈**的聚合关系

用带实心菱形的实线



泛化（Generalization）关系是对象之间耦合度最大的一种关系 表示一般与特殊的关系，是父类与子类之间的关系，是一种继承关系，是 is-a 的关系



实现（Realization）关系是接口与实现类之间的关系。在这种关系中，类实现了接口，类中的操作实现了接口中所声明的所有的抽象操作。