UML，全称 Unified Modeling Language，统一建模语言。而 UML 图分为用例图、类图、对象图、状态图、活动图、时序图、协作图、构件图、部署图等 9 种图。

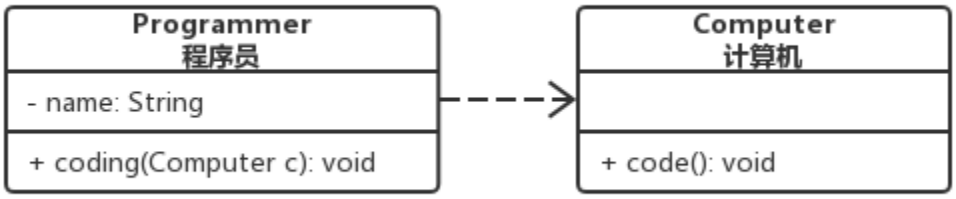
在面向对象语言中，我们经常看到有用 UML 类图去表示各种接口和类之间关系的。但是，每次看的都是云里雾里，搞不清楚那些虚线，箭头都是代表什么意思。今天，就让我们来一探究竟吧。

UML 类图中有六种关系，分别是依赖关系，关联关系，聚合关系，组合关系，实现关系，泛化关系。
经过我自己的理解，画出了六种关系的示例图。类的成员变量和方法前面的修饰符有 public, private, protected, default，在 UML 类图中分别用 +, -, #, ~表示。

一、依赖关系

依赖关系是一种使用关系，表示某个类依赖于另外一个类，通常表现为，某个类的方法的参数使用了另外一个类的对象。

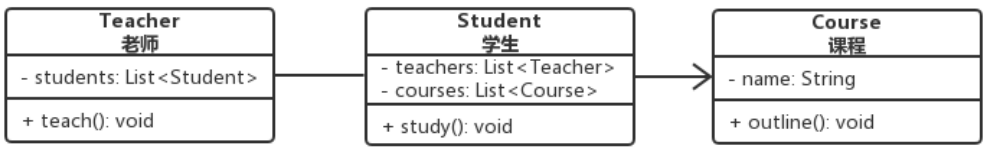
在 UML 类图中，依赖关系用带箭头的虚线表示，箭头从使用类指向被依赖的类。下图中表示，程序员依赖于计算机来编写代码。



二、关联关系

关联关系是对象之间的一种引用关系，表示一个类和另外一个类之间的联系，如老师和学生，丈夫和妻子等。

关联关系有单向和双向的。在 UML 类图中，单向关联用一个带箭头的实线表示，箭头从使用类指向被关联的类，双向关联用带箭头或者没有箭头的实线来表示。

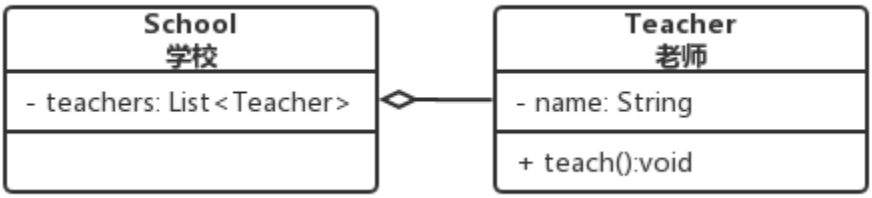


上图表示老师和学生之间的关系是双向的，一个老师可以有多个学生，一个学生也可以有多个老师。学生和课程之间是单向的，一个学生会学习多门课程，而课程是一个抽象的概念，它不拥有学生。

三、聚合关系

聚合关系是关联关系的一种，表示整体和部分之间的关系，如学校和老师，车子和轮胎。

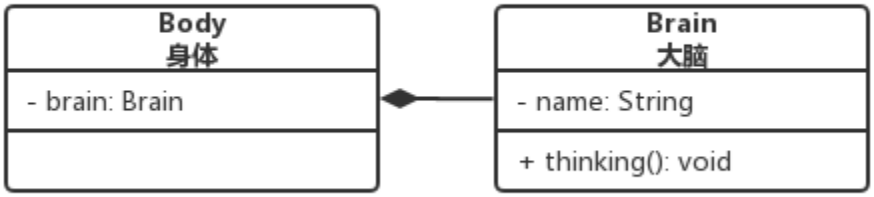
聚合关系在类中是通过成员对象来体现的，成员是整体的一部分，成员也可以脱离整体而存在。如老师是学校的一部分，同时老师也是独立的个体，可以单独存在。



在 UML 类图中，用带空心菱形的实线来表示聚合关系，菱形指向整体。

四、组合关系

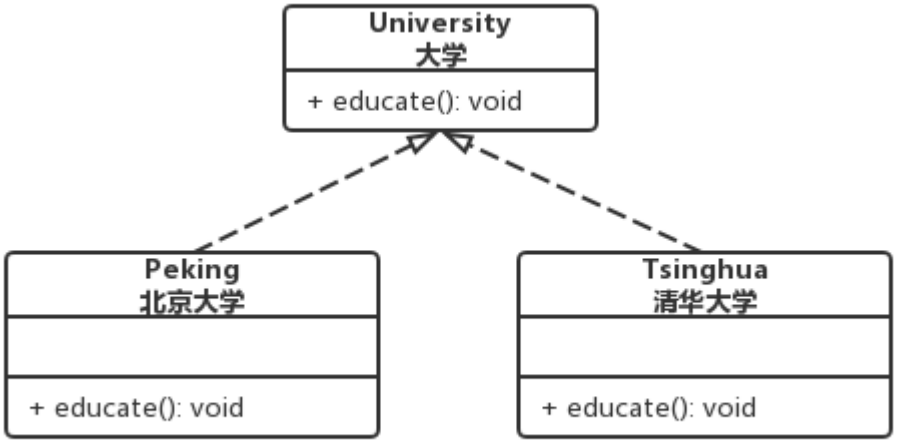
组合关系是整体和部分之间的关系，也是关联关系的一种，是一种比聚合关系还要强的关系。部分对象不能脱离整体对象而单独存在，如人的身体和大脑之间的关系，大脑不能脱离身体而单独存在。



在 UML 类图中，用带实心菱形的实线来表示组合关系，菱形指向整体。

五、实现关系

实现关系就是接口和实现类之间的关系。类实现了接口中的抽象方法。
在 UML 类图中，用带空心三角箭头的虚线来表示实现关系，箭头从实现类指向接口。

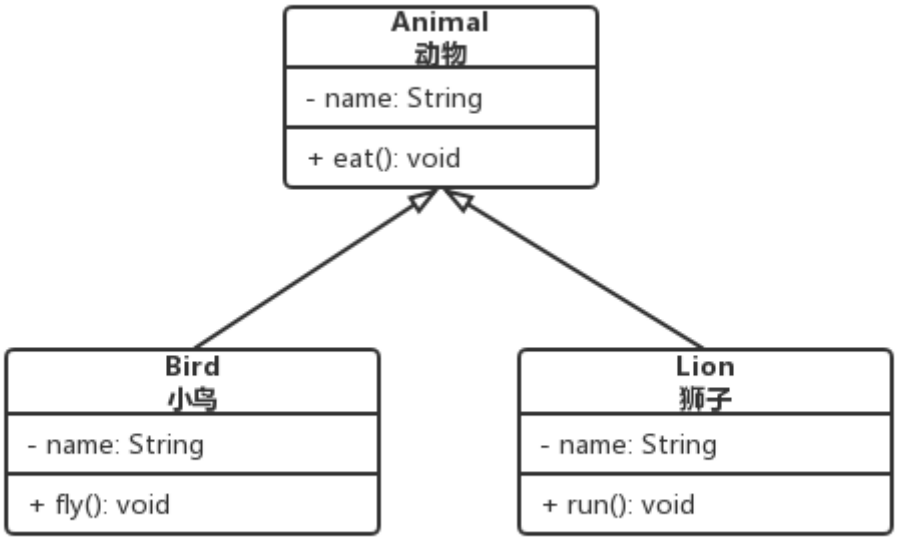


如上图，北京大学和清华大学分别实现了大学接口。

六、泛化关系

泛化关系其实就是父子类之间的继承关系，表示一般与特殊的关系，指定子类如何特殊化父类的特征和行为。

在 UML 类图中，用带空心三角箭头的实线来表示泛化关系，箭头从子类指向父类。



如上图，父类动物有一个吃的方法，小鸟和狮子都继承于动物类，小鸟有它特有的方法飞行，而狮子有特有的方法奔跑。
六种关系中，从弱到强依次是：
依赖关系 < 关联关系 < 聚合关系 < 组合关系 < 实现关系 = 泛化关系

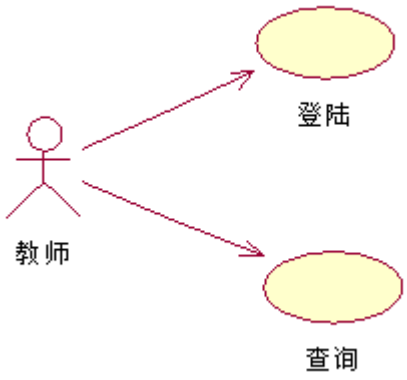
用例图中包括三种元素，参与者，用例，它们之间的关系。下面说说参与者与用例之间，用例与用例之间都有哪些关系。

1.关联关系

定义：参与者与用例之间通常用关联关系来描述。

表示方法：带箭头的实线，箭头指向用例。

如图所示：



2. 泛化关系

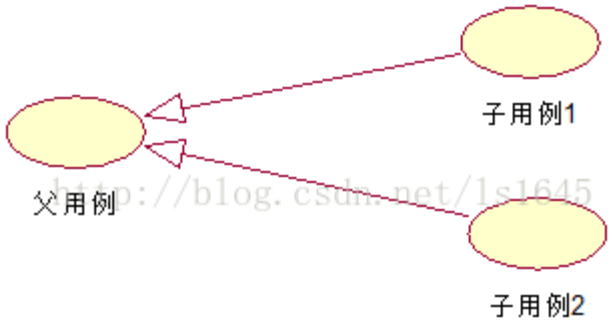
定义：一个用例可以被特别列举为一个或多个子用例，这被称为用例泛化。

泛化关系在类间也有。

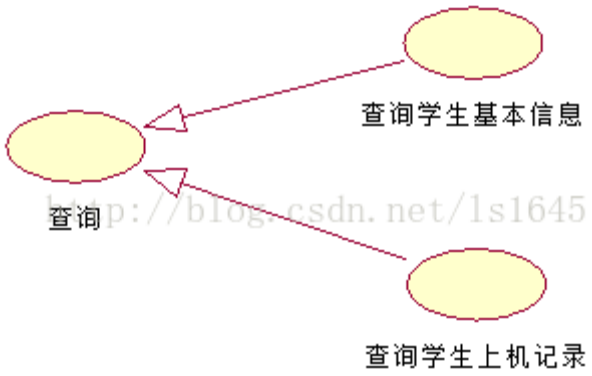
子用例从父用例处继承行为和属性，还可以添加行为或覆盖、改变已继承的行为。

表示方法：带空心箭头的实线，箭头指向泛化（被继承）的用例，即父用例。（**PS：**泛化关系的箭头不是指向被泛化，而是指向被继承。泛化和继承是不同的方向。泛化是从下到上的抽象过程，继承是从上到下，从一般到特殊的过程。）

如图所示：



机房收费系统中可以这样应用：



当系统中具有一个或多个用例是一般用例的特化时，就使用用例泛化。

3.包含关系

定义：其中一个用例（基础用例）的行为包含了另一个用例（包含用例）的行为。基础用例可以看到包含用例，并依赖于包含用例的执行结果。但是二者不能访问对方的属性。

表示方法：虚线箭头+<<include>>字样，箭头指向被包含的用例。

如图所示：

使用情况：

（1）如果两个以上用例有重复的功能，则可以将重复的功能分解到另一个用例中。其他用例可以和这个用例建立包含关系。

（2）一个用例的功能太多时，可以用包含关系创建多个子用例。

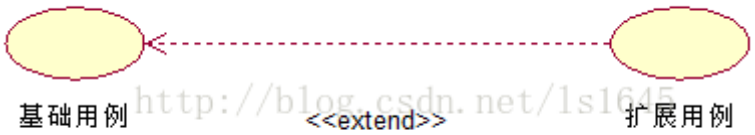
4.扩展关系（extend）

定义：是把新行为插入到已有用例的方法。

个人感觉可以叫做特殊情况处理。比如去食堂用饭卡打饭，绝大部分人是刷卡，拿饭，两个步骤就完成了。但是如果某个学生的饭卡里没钱了，假定不用现金或者借钱或者赊账等等其他的方式来打饭，而是必须用自己的饭卡来打饭。那么他就要先去给饭卡充值。“饭卡充值”就是“刷卡”的一个扩展用例。“饭卡充值”与“刷卡”就是扩展关系。

表示方法：虚线箭头+<<extend>>字样，箭头指向被扩展的用例（即基础用例）。

如图所示：



作用：为处理异常或构建灵活系统框架提供了一种有效的方法。

对比：

包含与扩展的区别。在扩展关系中，基础用例没有扩展也是完整的，而在包含关系中，基础用例依赖于包含用例的执行结果。

总结：

所有的箭头指向都是**“被”**的一端。

找关系，是一件挺复杂的事儿。从不同的角度看会有不同的结果。找到大前提，再理顺特定环境下的关系，会更加顺手。