## UML中类图的总结

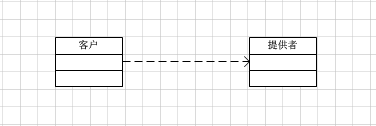
类图（Class Diagram）是描述类、接口、协作以及它们之间关系的图，用来显示系统中各个类的静态结构。

类图包含7个元素：类（Class）、接口（Interface）、协作（Collaboration）、依赖关系（Dependency）、泛化关系（Generalization）、关联关系（Association）以及实现关系（Realization）。

类之间的关系最常用的有4种，分别是表示类之间使用关系的**依赖关系**（Dependency）、表示类之间一般和特殊关系的**泛化关系**；表示对象之间结构关系的**关联关系**；表示类中规格说明和实现之间关系的**实现关系**。

1. **依赖关系（Dependency）：**

依赖表示两个或多个模型元素之间语意上的关系。它只将模型元素本省连接起来而不需要用一组实例来表达它的意思。它表示了这样一种情形，对于一个元素（提供者）的某些改变会影响或提供消息给其他元素（客户），即客户以某种形式依赖于其他类元。根据这个定义，关联、实现和泛化都是依赖关系，但是它们有更特别的语意，所以在UML中被分离出来作为独立关系。在UML中，依赖用一个从客户指向提供者的虚箭头表示。



UML定义了4种基本依赖类型，分别是使用（Usage）依赖、抽象（Abstraction）依赖、授权（Permission）依赖和绑定（Binding）依赖。

1).使用依赖：使用依赖都是非常直接的，通常表示客户使用提供者提供的服务以实现它的行为。

|  |  |  |
| --- | --- | --- |
| **依赖关系** | **功能** | **关键字** |
| 使用 | 声明使用一个模型元素需要用到已存在的另一个模型元素，这样才能正确实现使用者的功能（包括了调用、实例化、参数和发送），这是最常使用的依赖。 | use |
| 调用 | 声明使用一个类调用其他类的操作的方法 | call |
| 参数 | 声明一个操作和它的参数之间的关系 | parameter |
| 发送 | 声明信号发送者和信号接收者之间的关系 | send |
| 实例化 | 声明用一个类的方法创建了另一个类的实例 | instantiate |

2).抽象依赖：抽象依赖用来表示客户与提供者之间的关系，依赖于在不同抽象层次上的事物。

|  |  |  |
| --- | --- | --- |
| **依赖关系** | **功能** | **关键字** |
| 跟踪 | 声明不同模型中的元素之间存在一些连接，但不如映射精确 | trace |
| 精化 | 声明具有两个不同语意层次上的元素之间的映射 | refine |
| 派生 | 声明一个实例可以从另一个实例导出 | derive |

3).授权依赖：授权依赖表示一个事物访问另一个事务的能力。提供者通过规定客户的权限，可以控制和限制对其内容访的方法。

|  |  |  |
| --- | --- | --- |
| **依赖关系** | **功能** | **关键字** |
| 访问 | 允许一个包访问另一个包的内容 | access |
| 导入 | 允许一个包访问另一个包的内容并为被访问包的组成部分增加别名 | import |
| 友元 | 允许一个元素访问另一个元素，不管被访问的元素是否具有可见性 | friend |

4).绑定依赖：绑定依赖是较高级的依赖类型，用于绑定模板以创建新的模型元素。

|  |  |  |
| --- | --- | --- |
| **依赖关系** | **功能** | **关键字** |
| 绑定 | 为模板参数指定值，以生成一个新的模型元素 | bind |

1. **泛化关系（Generalization）：**

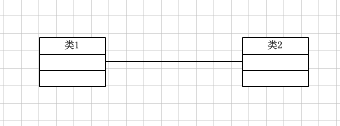
泛化关系是一种存在于一般元素和特殊元素之间的分类关系。其中，特殊元素与一般元素兼容，且还包含附加信息。泛化关系只使用在类型上，而不是实例上。泛化关系描述了“is a kind of”（是。。。。的一种）的关系。在类中，一半元素被称作超类或父类，而特殊元素被称作子类。

在UML中，泛化关系用一条从子类指向父类的空心三角箭头表示：



1. **关联关系（Association）：**

关联关系是一种结构关系，它指明一个事物的对象与另一个事务的对象之间的联系。也就是说，关联描述了系统中对象或实例之间的离散连接。关联的任何一个连接点都叫做关联端，与类有关的许多信息都附在它的端点上。在UML中，关联关系用一条连接两个类的实线表示：



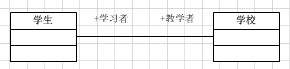
最普通的关联是二元关联。关联的实例之一是链，每个链由一组对象（一个有序列表）构成，每个对象来自于相应的类，其中二元链包含一对对象。有时同一个类在关联中出现不止一次，这时一个单独的对象就可以与自己关联。

除了关联的基本形式之外，还有6种应用于关联的修饰，分别是名称、角色、多重性、聚合、组合和导航性。

1).名称（Name）：关联可以有一个名称，用来描述关系的性质。名称以前缀或后缀一个指引阅读的方向指示符以消除名称含义上可能存在的歧义，方向指示符用一个实心的三角形箭头表示。

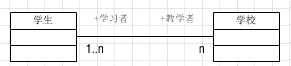


2).角色（Role）：角色是关联关系中一个类对另一个类表现出来的职责。

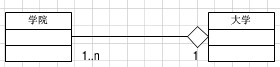


3).多重性（Multiplicity）：约束是UML三大扩展机制之一，多重性是其中的一种约束，也是使用最广泛的约束。关联的多重性是指有多少哥对象可以参与该关联，多重性可以哦也你过来表达一个取值范围、特定值、无限定的范围或一组离散值。在UML中，多重性表示为“..”分隔开的区间，其格式为“minimum..maximum”，其中minimum和maximum都是Int型的整数。赋给一个端点的多重性表示该端点可以有多少哥对象与另一个端点的一个对象关联。

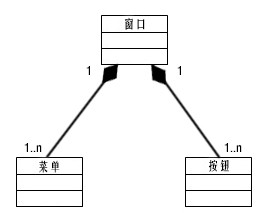
|  |  |  |  |
| --- | --- | --- | --- |
| **修饰** | **语义** | **修饰** | **语义** |
| 0 | 恰为0 | 1 | 恰为1 |
| 0..1 | 0或1 | 1..n | 1或更多 |
| 0..n | 0或更多 | n | 0或更多 |



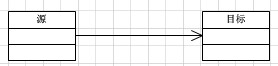
4).聚合（Aggregation）：聚合关系是一种特殊类型的关联，它表示整体与部分关系的关联。简单的说，关联关系中一组元素组成了一个更大、更复杂的单元。聚合关系描述了“has a”的关系。在UML中，聚合关系用带空心菱形头的实线来表示，其中头部指向整体。



5).组合关系（Composition）：组合关系是聚合关系中的一种特殊情况，是更强形式的聚合，又被称为强聚合。在组合中，成员对象的生命周期取决于聚合的生命周期，聚合不仅控制着成员对象的行为，而且控制着成员对象的创建和解构。在UML中，组合关系用带实心菱形头的实线来表示，其中头指向整体。

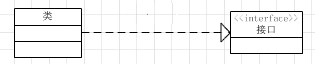


6).导航性（Navigation）：导航性描述的是一个对象通过链（关联的实例）进行导航访问另一个对象，即对一个关联端点设置导航属性意味这本端的对象可以被另一端的对象访问。只在一个方向上可以导航的关联称为单向关联，用一条带箭头的实线表示。在两个方向都可以导航的关联称为双向关联，用一条没有箭头的实线来表示。



1. **实现关系（Realization）：**

实现是规格说明和其实现之间的关系，它将一种模型元素与另一种模型元素连接起来，比如类和接口。泛化和实现关系都可以将一般描述与具体描述联系起来。泛化将同一语义层上的元素连接起来，并且通常在同一模型内。实现关系则将不同语义层内的元素连接起来，通常建立在不同的模型内。实现关系通常在两种情况下被使用：在接口与实现该接口的类之间；在用例以及实现该用例的协作之间。在UML中，实现关系的符号与泛化关系的符号类似，用一条带指向接口的空心三角箭头的虚线表示：



实现关系还有一种省略的表示方法，即将接口表示为一个小圆圈：

C:\Users\Manzhizhen\AppData\Roaming\Tencent\Users\835576511\QQ\WinTemp\RichOle\X6~JORUTV]}EXB{H5FI99I7.jpg