

- 关系
  - 实现关系
  - 泛化关系-generalization
  - 关联关系-association
- +表示public;
- -表示private;
- #表示protected;
- 不带符号表示default;

## 具体类

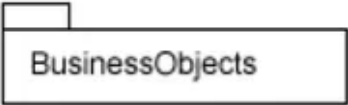
Java	UML
<pre>public class Hourly {     private double hourlyRate;     private double hoursWorked;     public double computePay() {         .....     } }</pre>	

## 抽象类 类名 / 方法——斜体

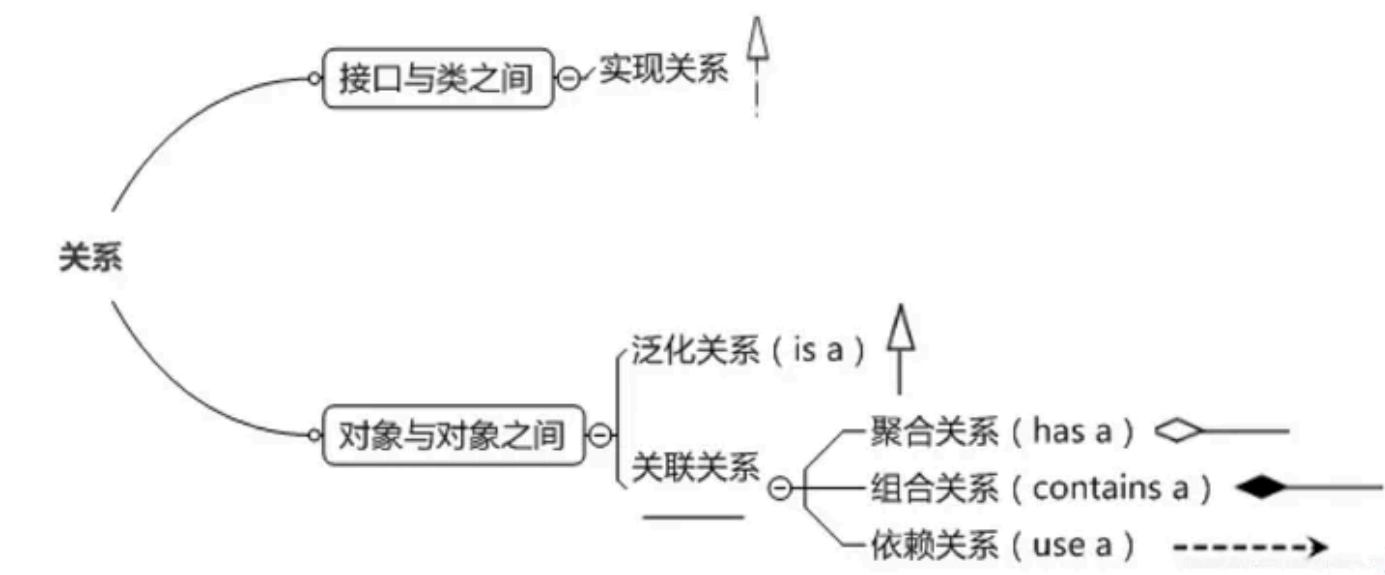
Java	UML
<pre>public abstract class Employee {     private String address;     private String name;     private int number;     public abstract double computePay() {         ...     }     public void mailCheck() {} }</pre>	

## 接口

Java	UML
<pre>public interface Shape {     public double computeArea();     public double computePerimeter(); }</pre>	

Java	UML
<pre>package BusinessObjects;  public class Employee { }</pre>	

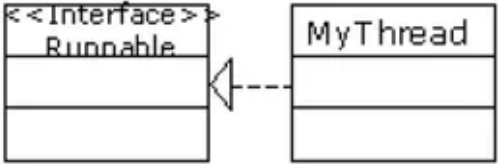
关系



实现关系

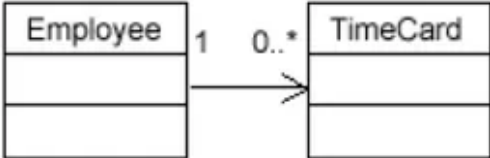
泛化关系-generalization

对象间继承关系

Java	UML
<pre>public interface Runnable { }  public class MyThread implements Runnable { }</pre>	

# 关联关系-association

- 单向关联 只有一个对象知道另一个对象（即可以调用）另一个对象的公共属性和操作。
- 双向关联

Java	UML
<pre>public class Employee {     private TimeCard _tc;     public void maintainTimeCard() {         ...     } }</pre>	 <pre>classDiagram     Employee "1" --&gt; "0..*" TimeCard</pre>

- 数字：精确的数量
- 或者0..：表示0到多个
- 0..1：表示0或者1个，在Java中经常用一个空引用来实现
- 1..\*：表示1到多个

关联关系分为依赖关联、聚合关联和组合关联

- 依赖关系dependency--弱关联