# 设计模式简介

## 设计模式的目的

提高代码的

1. 代码重用性(相同的代码不多次编写)
2. 可读性
3. 可扩展性(需要添加新功能时很方便)
4. 可靠性(当新增功能时，对原有的功能不会有影响)
5. 使程序高内聚低耦合

## 设计模式的7大原则

1. 单一职责原则（一个类应该只负责一项职责）
2. 接口隔离原则（一个类对另一个类的依赖应当建立在最小接口上的）
3. 依赖倒转原则
4. 高层模块不应依赖于低层模块
5. 抽象不应该依赖细节，细节不应该依赖抽象
6. 中心思想就是面向接口编程，抽象指的是接口与实现类，细节指的是实现类
7. 使用接口抽象类定义规范
8. 里氏替换原则

如何正确使用接口？

里氏替换原则强调尽量不要重写父类的方法。

1. 开闭原则

当软件需求发生变化时，尽量使用软件实体行为来实现变化，而不是通过修改原有代码。

1. 迪米特法则

一个对象应对另一个对象有最少了解，也就是最少的public原则。

也叫最少知道原则，其他类只需要调用方法就行，不需要关注具体实现

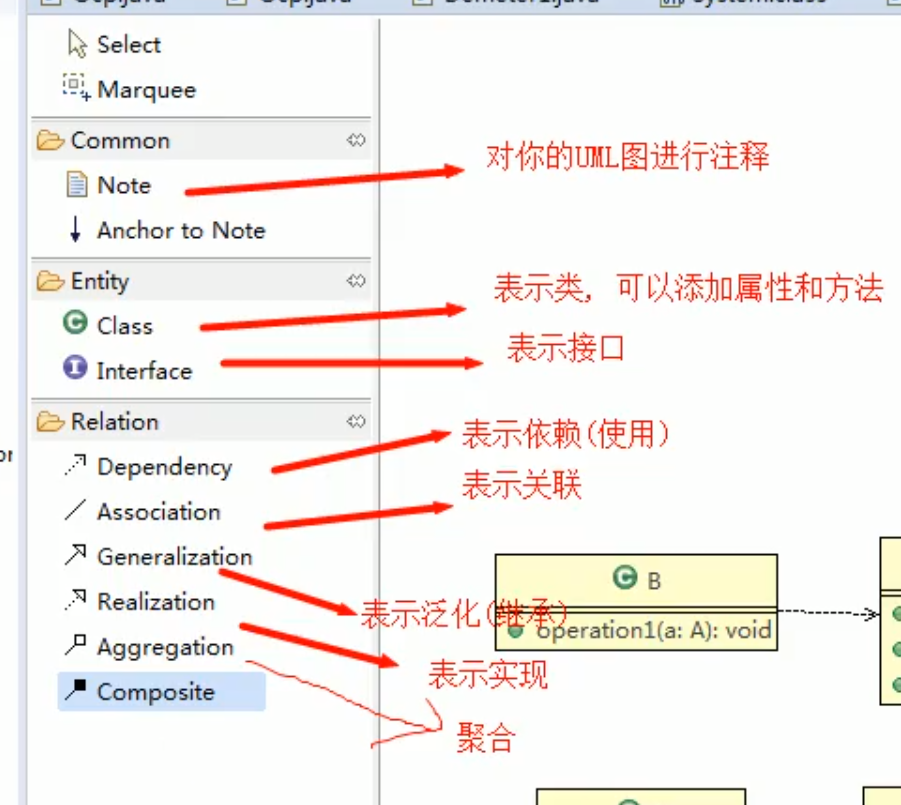
1. 合成复用原则

核心思想：

1. 独立出变化的部分
2. 针对接口编程而不是针对实现编程
3. 为松耦合原则

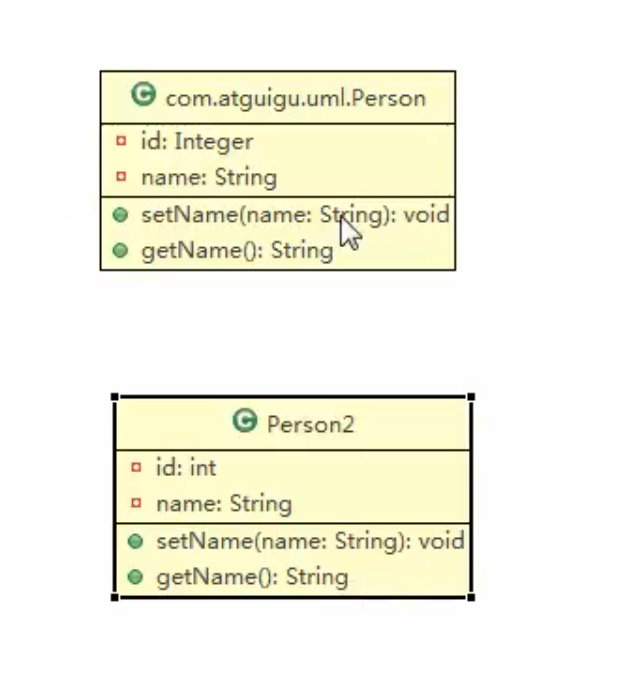
# UML图

## 类图



描述对象与对象之间的关系，包括依赖，泛化，实现，关联聚合和组合

### 基本画法



### 依赖



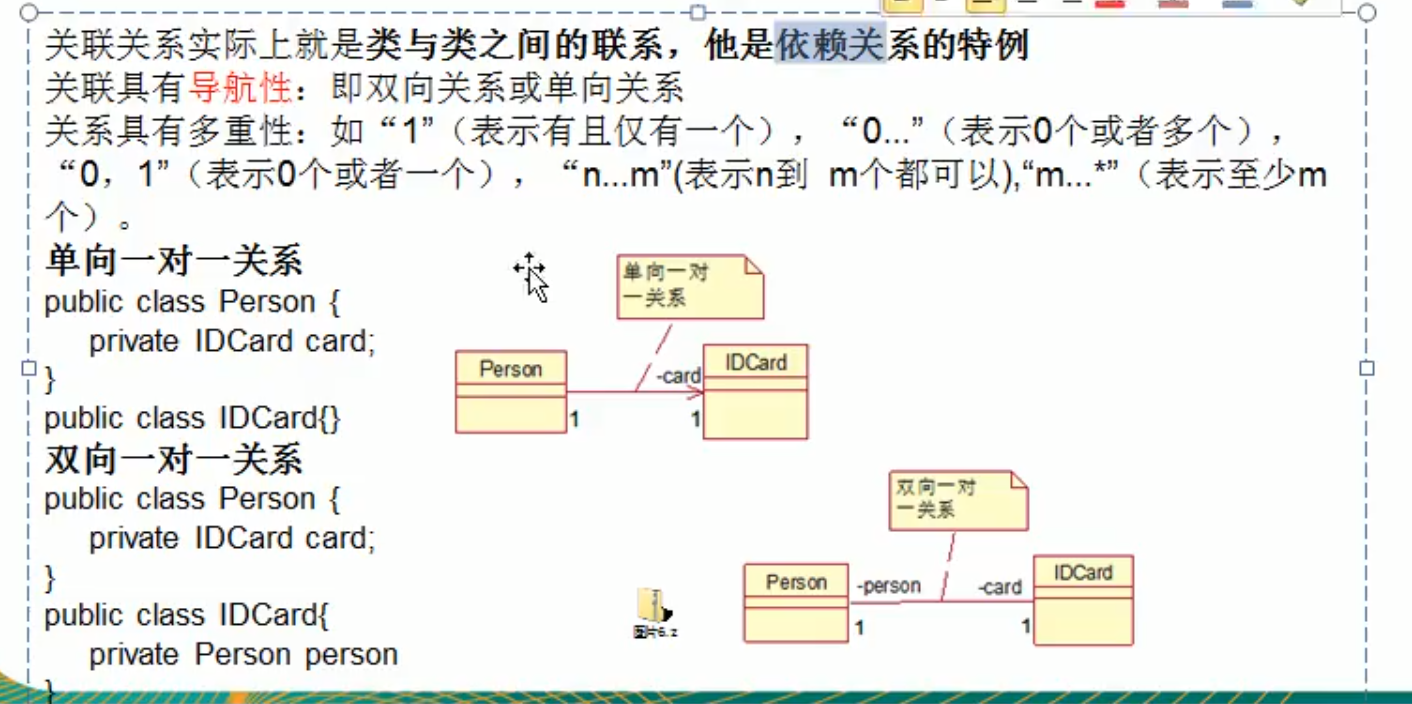
### 泛化

泛化其实就是继承关系

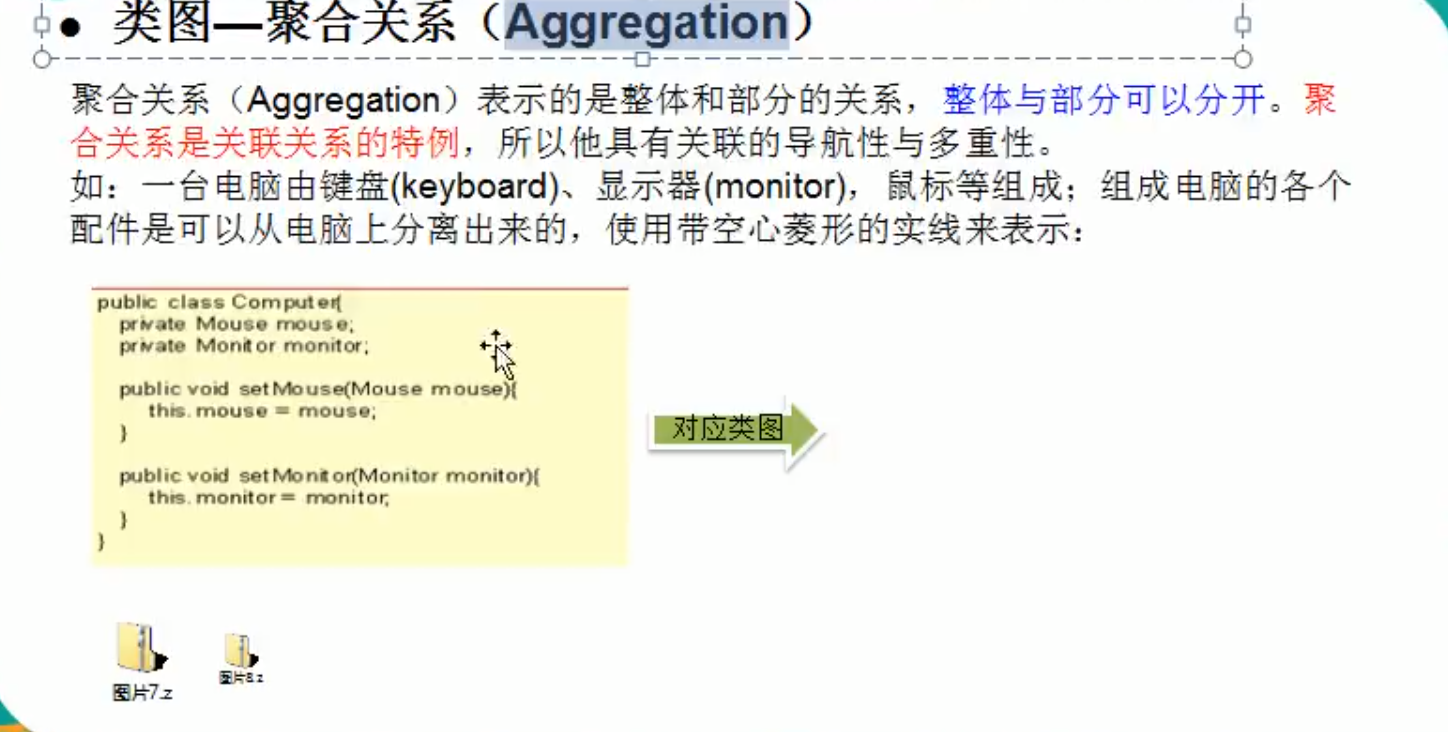
### 实现

其实就是接口实现的关系

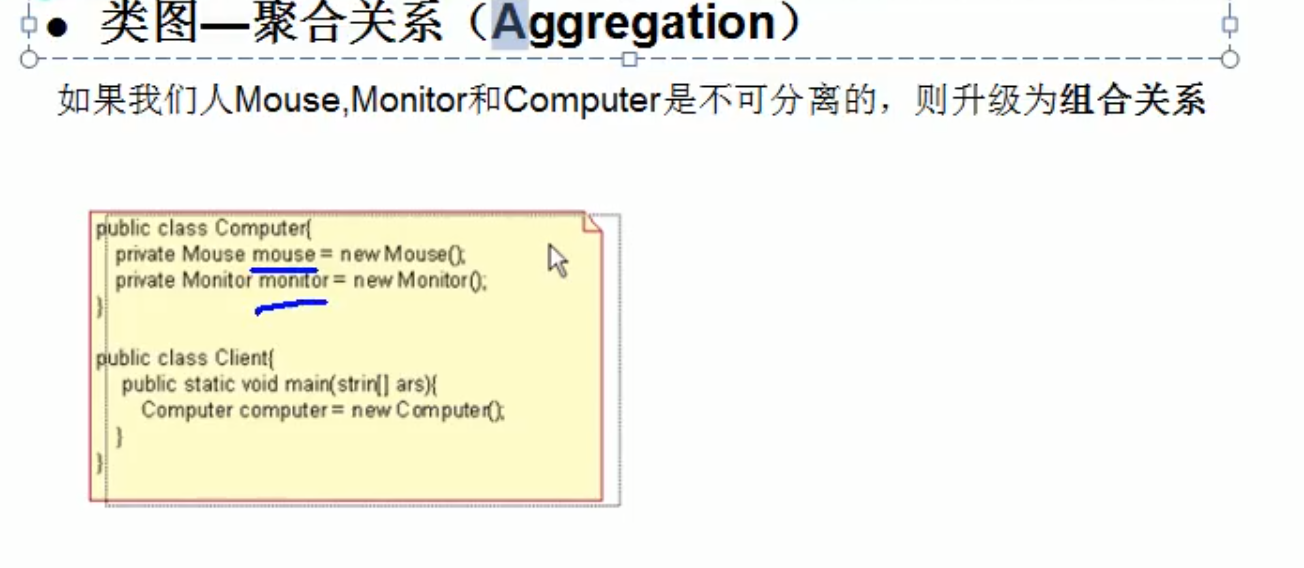
### 关联关系



### 聚合关系



### 组合关系



# 设计模式

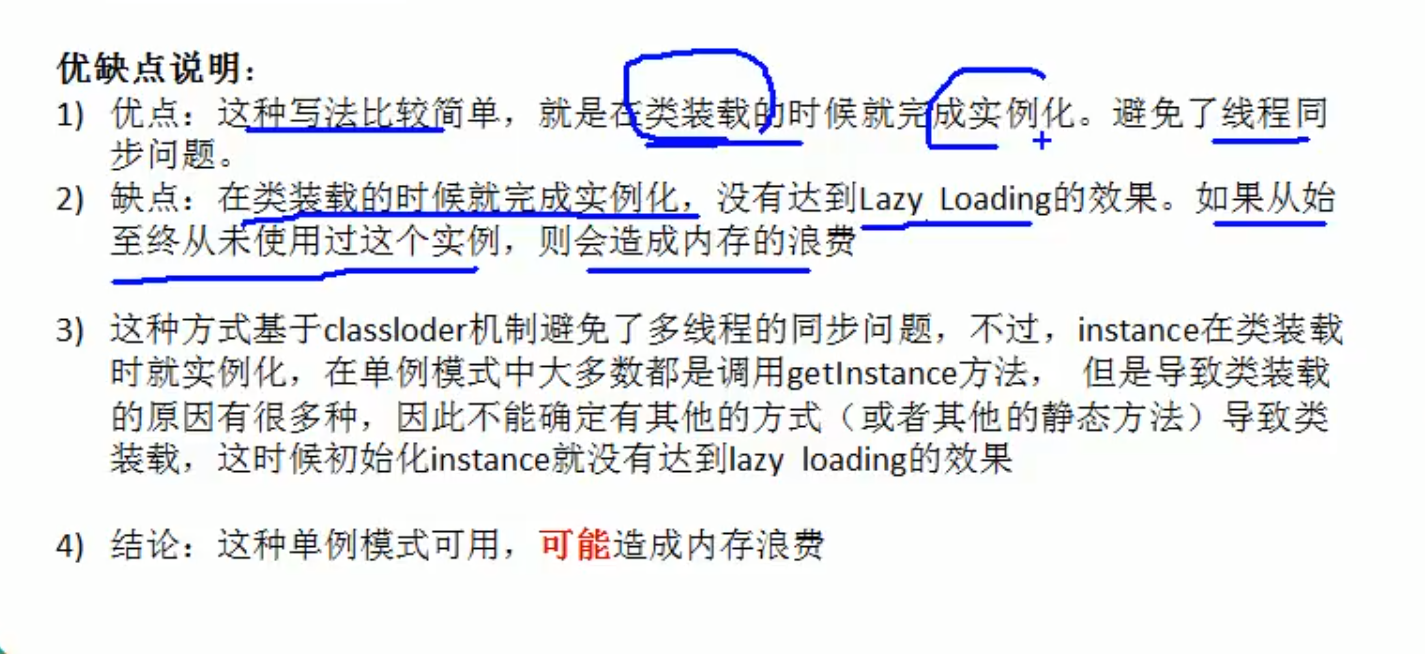
分类：



### 单例模式

饿汉式





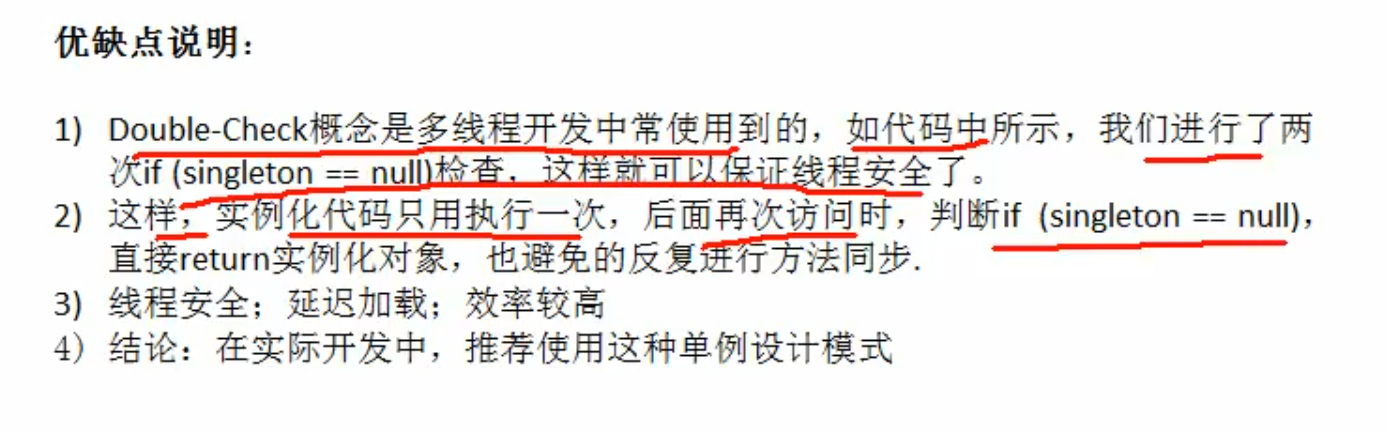
懒汉式



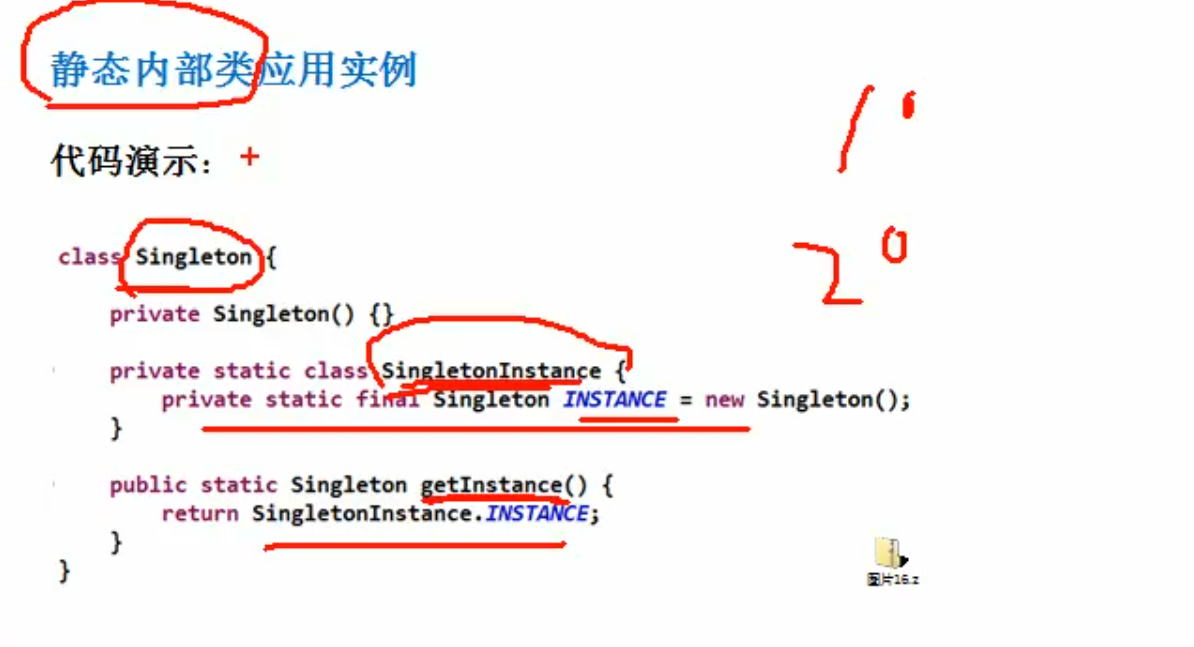


双重检查





静态内部类



# 工厂模式

简单工厂模式