Java设计模式之Composite模式Java认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/644/2021_2022_Java_E8_AE_BE_E8_AE_A1_c104_644808.htm Composite定义: 将对象以树形结构组织起来,以达成“ 部分－整体 ”的层次结构，使得客户端对单个对象和组合对象的使用具有一致性. Composite比较容易理解，想到Composite就应该想到树形结构图。组合体内这些对象都有共同接口,当组合体一个对象的方法被调用执行时，Composite将遍历(Iterator)整个树形结构,寻找同样包含这个方法的对象并实现调用执行。可以用牵一动百来形容。 所以Composite模式使用到Iterator模式，和Chain of Responsibility模式类似。 Composite好处: 1.使客户端调用简单，客户端可以一致的使用组合结构或其中单个对象，用户就不必关系自己处理的是单个对象还是整个组合结构，这就简化了客户端代码。 2.更容易在组合体内加入对象部件. 客户端不必因为加入了新的对象部件而更改代码。 如何使用Composite? 首先定义一个接口或抽象类，这是设计模式通用方式了，其他设计模式对接口内部定义限制不多，Composite却有个规定，那就是要在接口内部定义一个用于访问和管理Composite组合体的对象们（或称部件Component）. 下面的代码是以抽象类定义，一般尽量用接口interface, public abstract class Equipment { private String name. //实价 public abstract double netPrice(). //折扣价格 public abstract double discountPrice(). //增加部件方法 public boolean add(Equipment equipment) { return false. } //删除部件方法 public boolean remove(Equipment equipment) { return false. } //注意这里，这里就提供一种用于访问组合体类的部件

方法。 public Iterator iter() { return null. } public Equipment(final String name) { this.name=name. } } 抽象类Equipment就是Component定义，代表着组合体类的对象们,Equipment中定义几个共同的方法。 public class Disk extends Equipment { public Disk(String name) { super(name). } //定义Disk实价为1 public double netPrice() { return 1. } //定义了disk折扣价格是0.5 对折。 public double discountPrice() { return 0.5. } } Disk是组合体内的一个对象，或称一个部件，这个部件是个单独元素(Primitive)。