# Composite设计模式文档

**Composite**

将对象组合成树形结构以表示“部分-整体”的层次结构。组合模式使得用户对单个对象和组合对象的使用具有一致性。

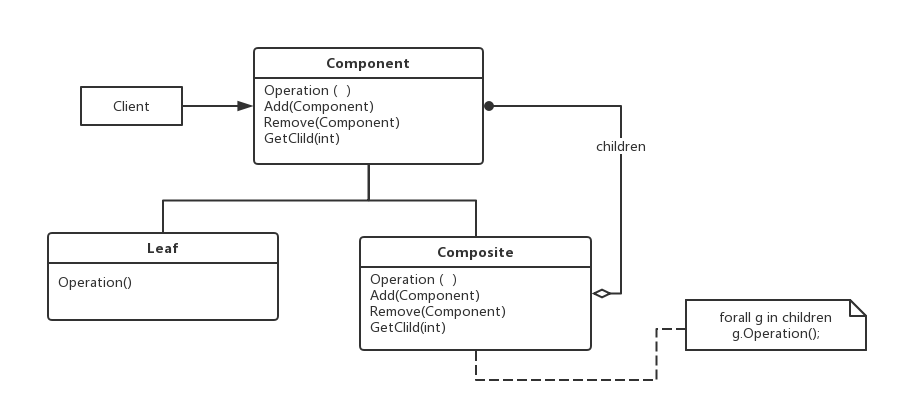
**透明方式**

组合模式的实现根据所实现接口的区别分为两种形式，分别称为安全模式和透明模式。组合模式可以不提供父对象的管理方法，但组合模式必须在合适的地方提供子对象的管理方法（诸如：add、remove、getChild等）。

在Component里面声明所有的用来管理子类对象的方法，包括add（）、remove（），以及getChild（）方法。这样做的好处是所有的构件类都有相同的接口。在客户端看来，树叶类对象与合成类对象的区别起码在接口层次上消失了，客户端可以同等同的对待所有的对象。这就是透明形式的组合模式。

**类图及样例实现：**

这里给出安全方式的组合模式的类图结构和样例实现，透明方式就是在叶子节点的add()/remove()/GetChild()均有实现，不过是无意义的实现。大部分应用都是基于透明模式的，因为这样代码可以重用。



这种形式涉及到三个角色：

**抽象构件(Component)角色**：这是一个抽象角色，它给参加组合的对象定义出公共的接口及其默认行为，可以用来管理所有的子对象。在安全式的合成模式里，构件角色并不是定义出管理子对象的方法，这一定义由树枝构件对象给出。

**树叶构件（Leaf）角色**：树叶对象是没有下级子对象的对象，定义出参加组合的原始对象的行为。

**树枝构件（Composite）角色**：代表参加组合的有下级子对象的对象。树枝对象给出所有的管理子对象的方法，如add（）、remove（）、getChild（）等。