什么是组合模式？

组合模式允许你将对象组合成树形结构来表现”部分-整体“的层次结构，使得客户以一致的方式处理单个对象以及对象的组合。

组合模式实现的最关键的地方是——简单对象和复合对象必须实现相同的接口。这就是组合模式能够将组合对象和简单对象进行一致处理的原因。

•组合部件（Component）：它是一个抽象角色，为要组合的对象提供统一的接口。

•叶子（Leaf）：在组合中表示子节点对象，叶子节点不能有子节点。

•合成部件（Composite）：定义有枝节点的行为，用来存储部件，实现在Component接口中的有关操作，如增加（Add）和删除（Remove）。

透明式的组合模式（见demo03）

在Component中声明所有来管理子对象的方法，其中包括Add，Remove等。这样实现Component接口的所有子类都具备了Add和Remove方法。这样做的好处是叶节点和枝节点对于外界没有区别，它们具备完全一致的接口。

弊端：客户端对叶节点和枝节点是一致的，但叶节点并不具备Add和Remove的功能，因而对它们的实现是没有意义的。

安全式组合模式（见demo02）

在Component中不去声明Add和Remove方法，那么子类的Leaf就不需要实现它，而是在Composit声明所有用来管理子类对象的方法。

弊端：叶节点无需在实现Add与Remove这样的方法，但是对于客户端来说，必须对叶节点和枝节点进行判定，为客户端的使用带来不便。

组合模式的优缺点

优点：

•组合模式使得客户端代码可以一致地处理对象和对象容器，无需关系处理的单个对象，还是组合的对象容器。

•将客户代码与复杂的对象容器结构解耦。

•可以更容易地往组合对象中加入新的构件。

缺点：

使得设计更加复杂。客户端需要花更多时间理清类之间的层次关系。（这个是几乎所有设计模式所面临的问题）。

注意的问题：

•有时候系统需要遍历一个树枝结构的子构件很多次，这时候可以考虑把遍历子构件的结构存储在父构件里面作为缓存。

•客户端尽量不要直接调用树叶类中的方法，而是借用其父类的多态性完成调用，这样可以增加代码的复用性。

组合模式的使用场景

在以下情况下应该考虑使用组合模式：

1.当想表达对象的部分-整体的层次结构时。

2.希望用户忽略组合对象与单个对象的不同，用户将统一地使用组合结构中的所有对象时。