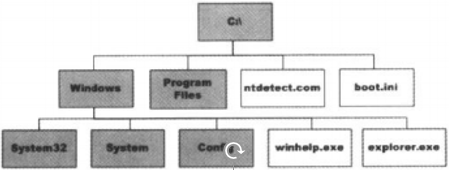
* 设计模式 – 合成模式（Composite）：

将对象组织到树结构中，可以用来描述整体与部分的关系。合成模式就是一个处理对象的树结构的模式。

合成模式把部分与整体的关系用树结构表示出来。

合成模式使得客户端把一个个单独的成分对象和由他们复合而成的合成对象同等看待。

一个文件系统就是一个典型的合成模式系统。如下图：

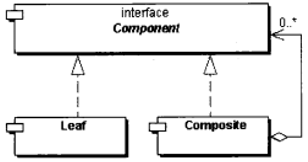


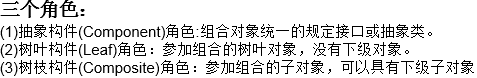
文件系统是一个树结构，树上长有节点。节点分为树枝和树叶两种。

树枝：即目录，有内部树结构，图中带阴影的部分。

树叶：即文件，没有内部结构，图中不带阴影的部分。

显然，可以把目录和文件当做同一种对象看待和处理，就是合成模式。



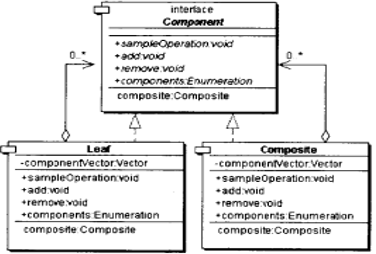


* 设计模式 – 合成模式分类：

合成模式可以不提供父对象的管理方法，但需要在合适的地方提供子对象的管理方法，诸如add(..)、remove(..)以及getChild(..)等方法。

基于这方面的考虑，合成模式分为**透明方式**和**安全方式**两类。

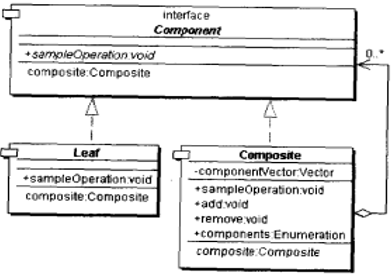
**1透明方式**：在Component里面声明所有用来管理子类对象的方法，包括add()、remove()、getChild()等方法。



**优点**：所有构件类都有相同的接口，树枝和树叶在功能上都是相同透明的。

**缺点**：不安全，树叶类对象应该没有下级对象，这些管理方法也没有意义，但在编译时不错，运行时可能会出错。

**2安全方式**：在Composite类里面声明管理子类对象的方法。这样做法比较安全，树叶类型的对象没有管理子类对象的方法，客户端使用叶子节点对象时，编译就会检查管理方法使用的错误。



* 设计模式 – 合成模式Demo：

