

GeekBand 极客班

互联网人才 + 油站!

C++ 设计模式

www.geekband.com

GeekBand 极客班 互联网人才+油站：

极客班携手 网易云课堂，针对热门IT互联网岗位，联合业内专家大牛，紧贴企业实际需求，量身打造精品实战课程。

专业课程 + 项目碾压 + 习题&辅导

- 顶尖大牛亲授
- 紧贴课程内容
- 学前导读
- 贴合企业实际需求
- 全程实战操练
- 周末直播答疑
- 找对重点深挖学习
- 作品就是最好的PASS卡
- 定期作业点评
- 多项专题辅导



www.geekband.com

C++设计模式

Decorator 装饰模式

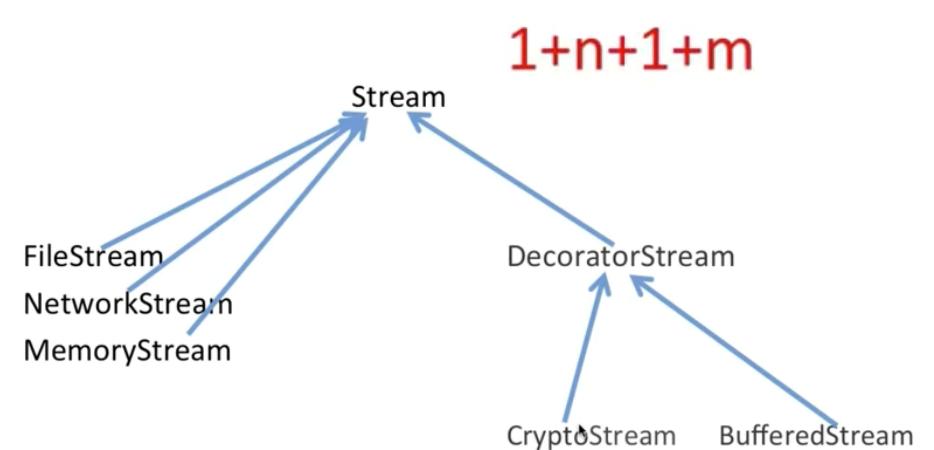
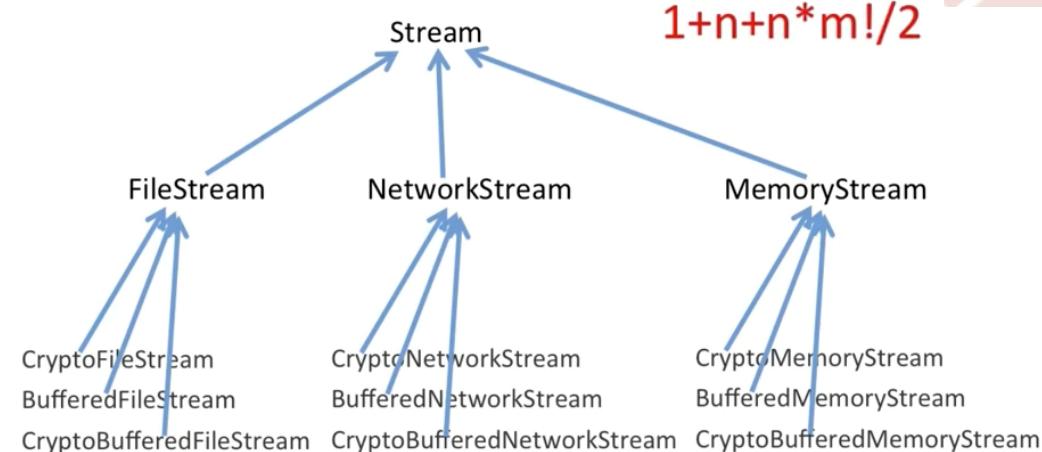
李建忠

GeekBar 板客班

“单一职责” 模式：

- 在软件组件的设计中，如果责任划分的不清晰，使用继承得到的结果往往是随着需求的变化，子类急剧膨胀，同时充斥着重复代码，这时候的关键是划清责任。
- 典型模式
 - Decorator
 - Bridge

Decorator 装饰模式



动机 (Motivation)

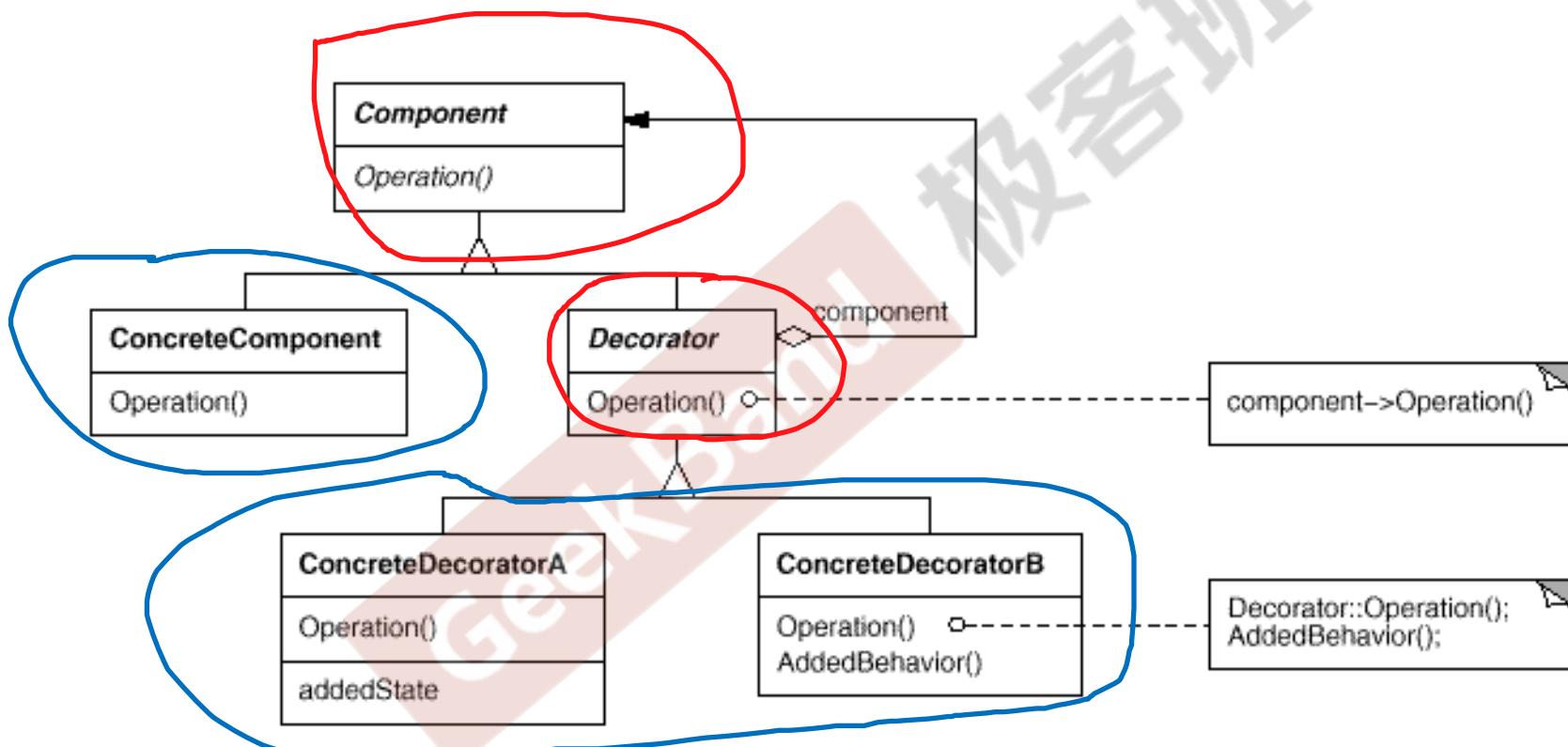
- 在某些情况下我们可能会 “过度地使用继承来扩展对象的功能” ,
由于继承为类型引入的静态特质 使得这种扩展方式缺乏灵活性 ;
并且随着子类的增多 (扩展功能的增多) , 各种子类的组合 (扩展
功能的组合) 会导致更多子类的膨胀。
- 如何使 “对象功能的扩展” 能够根据需要来动态地实现 ? 同时避
免 “扩展功能的增多” 带来的子类膨胀问题 ? 从而使得任何 “功能
扩展变化” 所导致的影响将为最低 ?

模式定义

动态（组合）地给一个对象增加一些额外的职责。就增加功能而言，Decorator模式比生成子类（继承）更为灵活（消除重复代码 & 减少子类个数）。

——《设计模式》 GoF

结构 (Structure)



要点总结

- 通过采用组合而非继承的手法，Decorator模式实现了在运行时动态扩展对象功能的能力，而且可以根据需要扩展多个功能。避免了使用继承带来的“灵活性差”和“多子类衍生问题”。
- Decorator类在接口上表现为is-a Component的继承关系，即Decorator类继承了Component类所具有的接口。但在实现上又表现为has-a Component的组合关系，即Decorator类又使用了另外一个Component类。
- Decorator模式的目的并非解决“多子类衍生的多继承”问题，Decorator模式应用的要点在于解决“主体类在多个方向上的扩展功能”——是为“装饰”的含义。