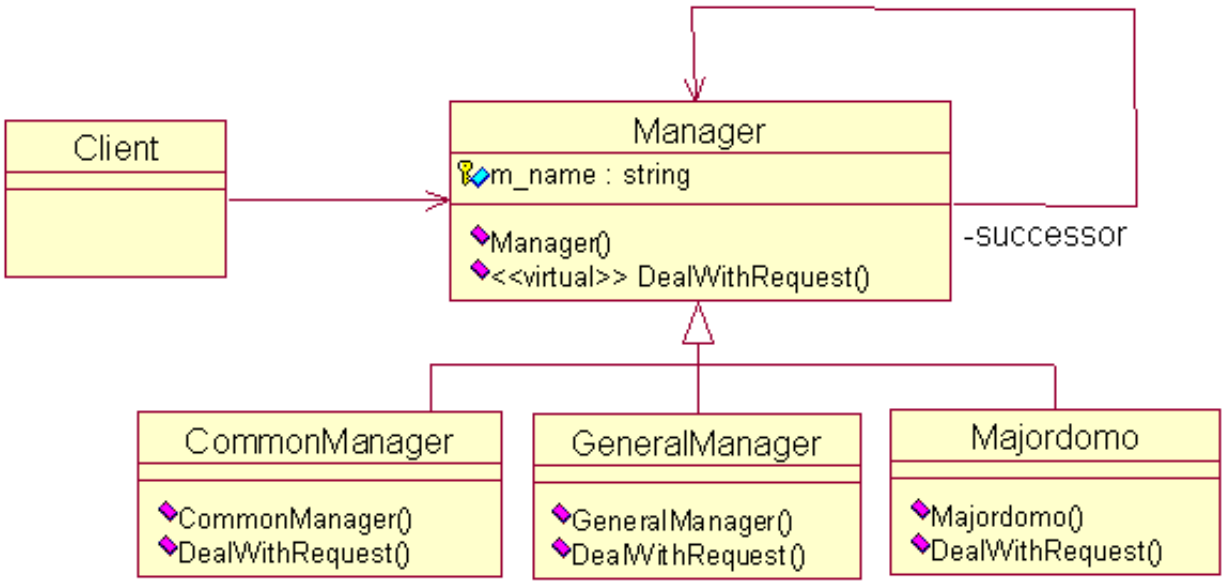
# 設計模式C++實現（14）——職責鏈模式

星期六, 2013 12月 14, 1:11 上午

軟件領域中的設計模式為開發人員提供了一種使用專家設計經驗的有效途徑。設計模式中運用了面向對象編程語言的重要特性：封裝、繼承、多態，真正領悟設計模式的精髓是可能一個漫長的過程，需要大量實踐經驗的積累。最近看設計模式的書，對於每個模式，用C++寫了個小例子，加深一下理解。主要參考《大話設計模式》和《設計模式:可復用面向對象軟件的基礎》兩本書。本文介紹裝飾模式的實現。

職責鏈模式：使多個對象都有機會處理請求，從而避免請求的發送者和接收者之間的耦合關係。將這些對象連成一條鏈，並沿著這條鏈傳遞該請求，直到有一個對象處理它為止。其思想很簡單，考慮員工要求加薪。公司的管理者一共有三級，總經理、總監、經理，如果一個員工要求加薪，應該向主管的經理申請，如果加薪的數量在經理的職權內，那麼經理可以直接批准，否則將申請上交給總監。總監的處理方式也一樣，總經理可以處理所有請求。這就是典型的職責鏈模式，請求的處理形成了一條鏈，直到有一個對象處理請求。給出這個例子的UML圖。




代碼的實現比較簡單，如下所示：

```
1. //抽象管理者
2. class Manager
3. {
4. protected:
5.     Manager *m_manager;
6.     string m_name;
7. public:
8.     Manager(Manager *manager, string name):m_manager(manager), m_name(name)
   {}
9.     virtual void DealWithRequest(string name, int num)  {}
10. };
11. //經理
12. class CommonManager: public Manager
13. {
14. public:
```

```
15.     CommonManager(Manager *manager, string name):Manager(manager,name) {}
16.     void DealWithRequest(string name, int num)
17.     {
18.         if(num < 500) //經理職權之內
19.         {
20.             cout<<"經理"<<m_name<<"批准"<<name<<"加薪"<<num<<"元"
    <<endl<<endl;
21.         }
22.         else
23.         {
24.             cout<<"經理"<<m_name<<"無法處理，交由總監處理"<<endl;
25.             m_manager->DealWithRequest(name, num);
26.         }
27.     }
28. };
29. //總監
30. class Majordomo: public Manager
31. {
32. public:
33.     Majordomo(Manager *manager, string name):Manager(manager,name) {}
34.     void DealWithRequest(string name, int num)
35.     {
36.         if(num < 1000) //總監職權之內
37.         {
38.             cout<<"總監"<<m_name<<"批准"<<name<<"加薪"<<num<<"元"
    <<endl<<endl;
39.         }
40.         else
41.         {
42.             cout<<"總監"<<m_name<<"無法處理，交由總經理處理"<<endl;
43.             m_manager->DealWithRequest(name, num);
44.         }
45.     }
46. };
47. //總經理
48. class GeneralManager: public Manager
49. {
50. public:
51.     GeneralManager(Manager *manager, string name):Manager(manager,name) {}
52.     void DealWithRequest(string name, int num)  //總經理可以處理所有請求
53.     {
54.         cout<<"總經理"<<m_name<<"批准"<<name<<"加薪"<<num<<"元"
    <<endl<<endl;
55.     }
56. };
```

客戶調用方式為：

```
1. //測試案例
2. int main()
3. {
4.     Manager *general = new GeneralManager(NULL, "A"); //設置上級，總經理沒有上
    級
5.     Manager *majordomo = new Majordomo(general, "B"); //設置上級
6.     Manager *common = new CommonManager(majordomo, "C"); //設置上級
7.     common->DealWithRequest("D",300);   //員工D要求加薪
```

```
    common->DealWithRequest("E", 600);
    common->DealWithRequest("F", 1000);
    delete common; delete majordomo; delete general;
    return 0;
}
```