AlexOne

四、代理模式

目录

1 概述

代理模式的优势在哪里?
在一定程度上可以优化调整构造的顺序。将Doc的全部构造，拆分成image、Text的分步构造。优化了Doc的构造函数。
为其他对象提供一种代理以控制对这个对象的访问
对一个对象访问控制的一个原因是只有在我们确实需要这个对象时才对它进行创建和初始化
[代理节省时间的本质就是把对象构造的时候加载资源花费的时间转移到使用的时候加载资源上去]
让容易加载的资源优先加载，加载慢的资源稍后加载。

2 实例

```
#include <iostream>
#include <unistd.h>

using namespace std;

class Text{                                     // 文本内容可以立即显示，速度很快！不需要代理
public:
    void showText(){
        cout<<"Wating for loading image."<<endl;
        cout<<"Wating for loading image."<<endl;
        cout<<"Wating for loading image."<<endl;
    }
};

#if 0
        ImageSubject
        /          \
   BigImage   <-  ProxyImage   <-  Doc
#endif

class ImageSubject{                             // 图片的加载需要时间，设置一个代理，设置代理就要提供接口
public:
    virtual void showImage()=0;
};

class BigImage:public ImageSubject{             // 缓慢加载图片
```

```
public:
    BigImage(){
        sleep(4);
        cout<<"已经加载完毕，可以显示图片"<<endl;
    }
    void showImage(){
        cout<<"长卷 清明上河图"<<endl;
    }
};

class ProxyImage:public ImageSubject{         // 使用代理加载图片
public:
    ProxyImage(){
        bi=NULL;
    }
    void showImage(){                         // 代理加载图片需要一个私有的图片数据指针成员
        if(bi==NULL)
            bi=new BigImage;                  // *********消耗时间*********
        bi->showImage();
    }
private:
    BigImage *bi;                             // 并不算子类对象，让代理生成以后，由代理去加载图片资
源
};

class Doc{
public:
    Doc(){
        t=new Text;
        pi=new ProxyImage;                    // 使用代理。代理生成的时候没有消耗时间
    }
    void show(){                              // 显示文本和使用代理去加载图片
        pi->showImage();                      // 本来该由BigImage去showImage，现在由代理去调用
子类对象的showImage
        t->showText();                        // 这两个调换位置还能不能得到想要的结果？明显不可
以，那么代理模式的优势在哪里?
    }
private:
    Text *t;
    ProxyImage *pi;
};

int main(){
    Doc d;       // 如果不使用代理，BigImage的构造(加载图片)需要6秒，【等到d构造结束才能显示出文本】
    d.show();    // 现在可以直接显示文本而无需等待图片加载结束
    return 0;
}
```