Structural - Proxy Design Pattern (İkinci örneğe mutlaka bakılmalı)

Açıklama : Gerçek nesne ile isteyen yani client arasına ara bir katman olarak proxy konulur ve client'a proxy sunulur.Client real subject ile erişim kurmamış olur.Mesela online izlenen bir filmde Film real subject'dir araya bir Proxy eklenerek kısım kısım çekilen kısmını bize izletmeyi hedefler.Dolayısıyla oluşturulması zaman alan bir nesne yaratılması gerektiğinde, uzaktaki bir sunucuya erişilerek bir nesne yaratılması gerektiğinde, real subject'e ulaşmadan önce bazı kontrollerin yapılması gerektiğinde yararlı olacak bir desendir.Aslında basit bir cache'leme mekanizmasıdır

Açıklama : Proxy design pattern Client tarafından erişilecek nesneye vekalet eden bir tasarım desenidir. Burada vekaletten kasıt ilgili nesneyi kontrol edecek bir Proxy nesnesinin kullanılmasıdır.

Üç farklı durumda Proxy Design Pattern kullanılır.

Remote(Uzak) Proxy

Remote(uzak) bir nesne kullanılacağı durumlarda kullanılabilir. Uzaktaki nesneye local bir temsilci sağlar ve gerekli kontrolleri yapmamıza olanak tanır.

Virtual Proxy

Üretimi yahut kullanımı maliyetli nesnelerin oluşturulması veya kullanılması için tercih edilir. Buna örnek olarak genelde herkesin dillendirdiği resim yükleme işlevini verebiliriz. Yüksek boyutlu bir resmin boyutundan dolayı geç yüklenmesi durumunda verilen -yükleniyor- mesajı ve ardından yükleme işlemi bittiği anda resmin gösterilmesinde kullanılabilir.

Protection Proxy

Yetkilendirme yahut login durumlarında kullanılabilir.

```
//Subject
interface IBank {
   //ödeme yap
   boolean pay(double price);
}

//Real Subject
class Bank implements IBank {
   @Override
   public boolean pay(double price) {
      if (price < 100)
         System.out.println("tutar 100 den az olamaz");
      else if (price > 100)
         System.out.println("tutar 100 den fazla olamaz");
      else {
         System.out.println("ödeme gerçekleşti");
         return true;
      }
      return false;
   }
}
```

```
//Proxy nesnesi
class ProxyBank implements IBank {
    private Bank bank;
    private boolean login;
    private String userName;
    private String password;
    public ProxyBank(String userName, String password) {
        this.userName = userName;
        this.password = password;
    }
    boolean enterLogin(){
        if (userName.equals("oky") && password.equals("oky")){
            bank = new Bank();
            login = true;
            System.out.println("Hesaba login yapıldı");
            return true;
        }
        else
            System.out.println("Kullanıcı adı şifre hatalı");
        login = false;
        return false;
    }

    @Override
    public boolean pay(double price) {
        enterLogin();
        if (!login){
            System.out.println("hesaba giriş yapılmadığından dolayı ödeme yapılamıyor");
            return false;
        }
        bank.pay(price);
        return true;
    }
}

//Main
public static void main(String[] args) {
    IBank bank = new ProxyBank("oky","2");
    bank.pay(0);
}
```

Structural - Proxy Design Pattern (Örnek 2)

Genelde verilen örneğe değinelim. Disk'ten bir image'in yüklendiği bir senaryo düşünelim;

```
interface Image{
   void display();
}
```

Image interface'ini implemente edecek RealSubject class'ımız;

```
class RealImage implements Image{
   private String fileName;
   public RealImage(String fileName){
      this.fileName = fileName;
      loadFromDisc(fileName);
   }
   @Override
   public void display() {
      System.out.println("Displaying : " + fileName);
   }

   private void loadFromDisc(String fileName){
      System.out.println("Loading from disc " + fileName);
   }
}
```

Proxy class'ımıza geldi sıra;

```
class ProxyImage implements Image{
   private RealImage realImage;
   private String fileName;
   public ProxyImage(String fileName){
      this.fileName = fileName;
   }
   @Override
   public void display() {
      if (realImage == null){
         realImage = new RealImage(fileName);
      }
      realImage.display();
   }
}
```

Proxy'imizi kullanalım;

```
public static void main(String[] args) {
   Image image = new ProxyImage("test.jpg");
   image.display();
   image.display();
}
```

```
"C:\Program Files\Java\jdk1.8.0_231\bin\java.exe" ...
Loading from disc test.jpg
Displaying : test.jpg
Displaying : test.jpg
```

Görüldüğü gibi image.display'in ikinci çağırımın da disk'e gitmedi