
Decorator Design Pattern

1. Decorator Design Pattern

Bu paternde amaç dinamik olarak nesnelere özellik ekleyebilmemizin sağlanmasıdır. Biz bir interface veya sınıfa bir fonksiyon ekleyerek bunun kalıtım yapan tüm sınıflar ve üretilen tüm nesnelerce kalıtılmasını sağlayabiliyoruz.

Fakat bu sefer isteğimiz tüm sınıfların ve nesnelerin değil sadece üretilmiş bazı nesnelerin bazı ek özellikler alması. Bu yüzden bu patern ile dinamik nesnelerin ek özellikler alması sağlanıyor.

1.1. Projede kullanma amacımız

Projede paterni örnekler içinde bulunduğumuz korona virüsünden etkilenen eğitim sistemi ve üniversiteleri ele aldık. Normalde her sene için nesne üreten ve öğrencileri kaydeden öğrenci işleri sınıfları var iken 2019 yılındaki öğrencilere özel mezuniyet şartları sağlamak için o yıla ait öğrencilere özel dinamik kapsülleme yapacak bir sistem sağlamak istiyoruz.

Sınıflara veya tüm nesnelere değilde sadece belirli nesnelere dinamik olarak özellik eklemek istediğimiz için bu paterni kullanarak kapsülleme işlemini gerçekleştirirerek koronaya özel mezuniyet şartları sağlanmıştır. Bu sayede Daha az staj günü ve akts ilede 2019 yılındaki öğrenciler mezun olabilmektedir.

2018 yılındaki bir öğrenci ile 2019 yılındaki bir öğrenci aynı akts ve staj gününe sahiptir. Buna rağmen dinamik olarak kapsülleme sağlandığı

için 2019 yılındaki öğrenci mezun olabilir iken diğer yıllardaki öğrenciler mezun olamıyor.

2. Decorator UML

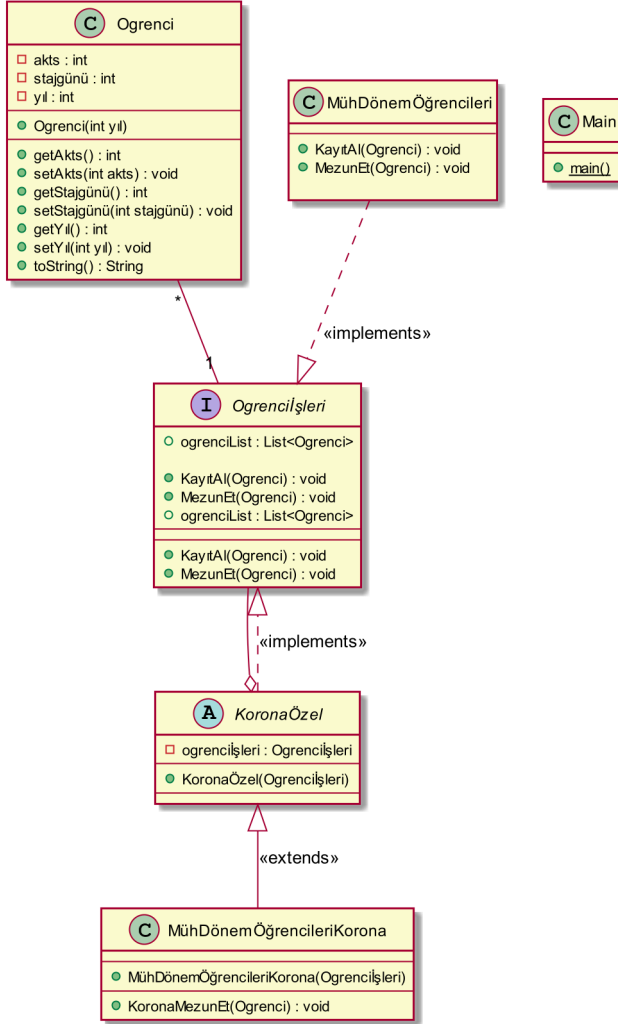


Figure 1. Decorator Uml

3. Decorator Tasarım Kalıbı Kod İncelemesi

Bu tasarım kalıbında asıl amaç bir nesnedeki fonksiyonları başka bir nesne aracılığıyla çalıştırmaktır. Örnek olarak vermek gerekir ise

donanım tarafında arka planda nasıl çalıştığını bilmememize rağmen arayüz yada bir buton aracılığı ile yaptığımız işlem classlar aracılığı ile işlenmektedir.

```
public class Ogrenci { ❶
    private int akts = 0;
    private int stajgünü = 0;
    private int yıl;

    public Ogrenci(int yıl){
        this.yıl = yıl;
    }
    //getter-setter
}

public interface Ogrenciİşleri { ❷

    public List<Ogrenci> ogrenciList = new ArrayList<>();

    public void KayıtAl(Ogrenci ogrenci);

    public void MezunEt(Ogrenci ogrenci);
}

public class MühDönemÖğrencileri implements Ogrenciİşleri { ❸

    @Override
    public void KayıtAl(Ogrenci ogrenci) {
        ogrenciList.add(ogrenci);
    }

    @Override
    public void MezunEt(Ogrenci ogrenci) {
        if (ogrenci.getAkts() >= 240 && ogrenci.getStajgünü() >= 50){
            System.out.println("Tebrikler mezun olabilirsiniz...");
        }else{
            System.out.println("Üzgünüz mezuniyet şartlarını
taşımıyorsunuz.");
        }
    }
}
```

```

    }
}

public abstract class KoronaÖzel implements Ogrenciİşleri {

    private Ogrenciİşleri ogrenciİşleri;

    public KoronaÖzel(Ogrenciİşleri ogrenciİşleri){ ④
        this.ogrenciİşleri = ogrenciİşleri;
    }

    public void KayıtAl(Ogrenci ogrenci) {
        ogrenciList.add(ogrenci);
    }

    public void MezunEt(Ogrenci ogrenci) {
        if (ogrenci.getAks() >= 240 && ogrenci.getStajgünü() >= 50){
            System.out.println("Tebrikler mezun olabilirsiniz....");
        }else{
            System.out.println("Üzgünüz mezuniyet şartlarınızı
taşımıyorsunuz.");
        }
    }
}

public class MühDönemÖğrencilerikorona extends KoronaÖzel {

    public MühDönemÖğrencilerikorona(Ogrenciİşleri ogrenciİşleri) {
        super(ogrenciİşleri);
    }

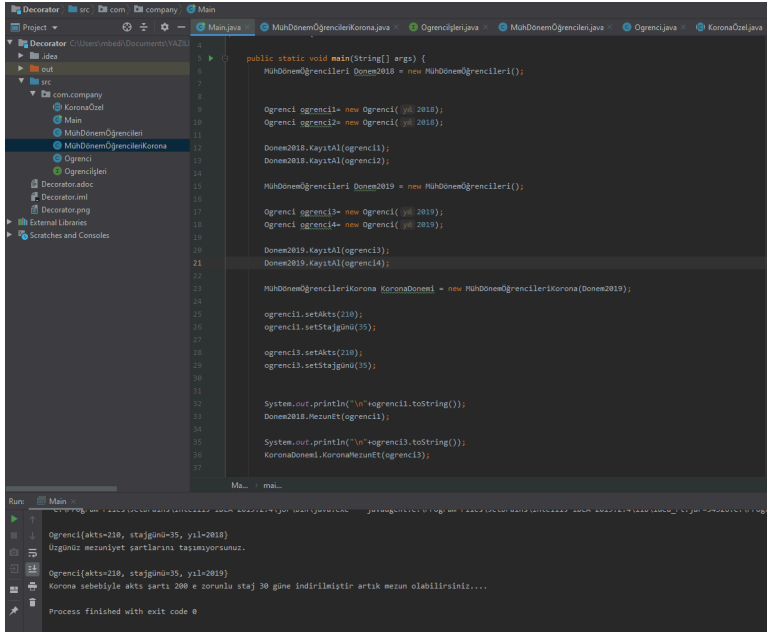
    public void KoronaMezunEt(Ogrenci ogrenci){ ⑤
        if (ogrenci.getAks() >= 200 && ogrenci.getStajgünü() >= 30){
            System.out.println("korona sebebiyle aks şartı 200 e
zorunlu staj 30 güne indirilmiştir artık mezun olabilirsiniz....");
        }
    }
}

```

```
        }else{
            System.out.println("Üzgünüz korona mezuniyet şartlarını
taşımıyorsunuz.");
        }
    }
}
```

- ❶ Proje genelinde kullanılmak üzere öğrenci akts staj günü ve mezuniyet yılını tutan bir sınıf yapısı
- ❷ Üniversitesdeki bölümlerin her yıl öğrencilerinin bilgilerini tutmak için oluşturulan sınıfların yapısını belirlemek için oluşturulan interface.
- ❸ Mühendislik öğrencilerinin işlemlerini yapmak için oluşturulan her yıl yenisi oluşturulan sınıf. Öğrenci işlerini implement ederek oradaki fonksiyonları override eder.
- ❹ Dinamik olarak kapsülleme yapmak için 3 numarada bahsettiğimiz sınıftan ürettiğimiz nesneleri vererek özel yapı oluşturacağımız nesneleri üretmek için extend edeceğimiz abstract classın contractor yapısını görüyoruz. Burada dinamik olan nesne olarak kapsülleme işlemi gerçekleştirilir ve istenilen ek özellikler fonksiyonlar bir sonraki kısımda eklenir.
- ❺ Dinamik olarak kapsülleme işlemini gerçekleştirdiğimiz kısımdan sonra eklenen fonksiyon burada tanımlanmıştır ve korona dönemine özel staj günü ve akts sayısı ile mezuniyet şartları denenir.

Decorator Design Pattern



```
1  public static void main(String[] args) {
2      MuhDönemÖğrencileri Dönem2018 = new MuhDönemÖğrencileri();
3
4      Öğrenci öğrenci1= new Öğrenci(100, 2018);
5      Öğrenci öğrenci2= new Öğrenci(100, 2018);
6
7      Dönem2018.KayıtAl(öğrenci1);
8      Dönem2018.KayıtAl(öğrenci2);
9
10     MuhDönemÖğrencileri Dönem2019 = new MuhDönemÖğrencileri();
11
12     Öğrenci öğrenci3= new Öğrenci(100, 2019);
13     Öğrenci öğrenci4= new Öğrenci(100, 2019);
14
15     Dönem2019.KayıtAl(öğrenci3);
16     Dönem2019.KayıtAl(öğrenci4);
17
18     MuhDönemÖğrencileriKorona KoronaDönemi = new MuhDönemÖğrencileriKorona(Dönem2019);
19
20     öğrenci1.setAktı(210);
21     öğrenci1.setStajGünü(35);
22
23     öğrenci3.setAktı(210);
24     öğrenci3.setStajGünü(35);
25
26     System.out.println("\n"+öğrenci1.toString());
27     Dönem2018.MezunEt(öğrenci1);
28
29     System.out.println("\n"+öğrenci3.toString());
30     KoronaDönemi.KoronaMezunEt(öğrenci3);
31
32 }
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Run: Main

Öğrenci(aktı=210, stajGünü=35, yıl=2018)
Üzgünüz mezuniyet şartlarına tapınıyorsunuz.

Öğrenci(aktı=210, stajGünü=35, yıl=2019)
Korona sebebiyle aktı şartı 200 e zorunlu staj 30 güne indirilmiştir artık mezun olabilirsiniz....

Process finished with exit code 0

Figure 2. Kod Ekran Çıktısı