Tasarım Örüntüleri
Decorator

Örüntülerin Temel Prensipleri

- GoF tasarım örüntülerinin altında yatan temel prensipler
  - Encapsulation
  - Composition
  - Abstract Data Types

Decorator Örüntüsü

- Decorator
  - Nesnelere inheritance yerine **composition ile ilave davranış** eklemeyi sağlar
  - **Proxy** örüntüsüne çok **benzer**
  - **Davrannışsal bir örüntü** olarak kabul edilir
  - **Single responsibility prensi**bini öne çıkarır

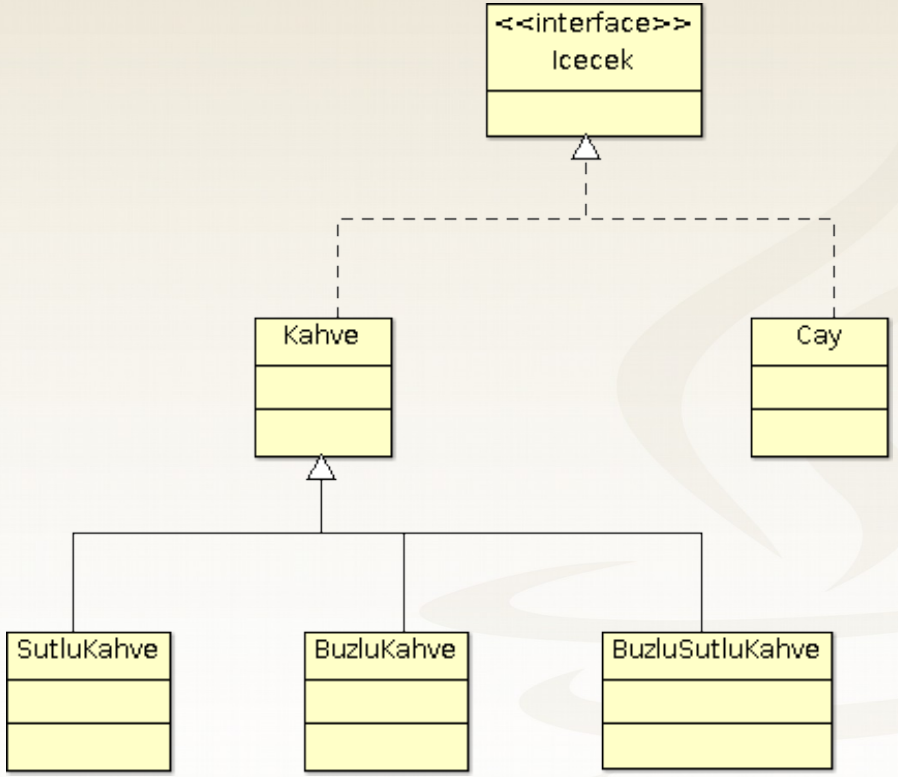
Örnek Problem

- Yeni açılan bir kafenin menüsünün içecekler kısmında çay ve kahve şeklinde iki farklı içecek seçeneği mevcuttur
- Müşteriler çay ve kahve siparişlerini sade verebildikleri gibi, sütlü veya buzlu olarak da verebilirler
- Tercihler bu karışımların birbirleri ile kombinasyonu şeklinde de olabilmektedir. Örneğin buzlu ve sütlü kahve siparişi gibi

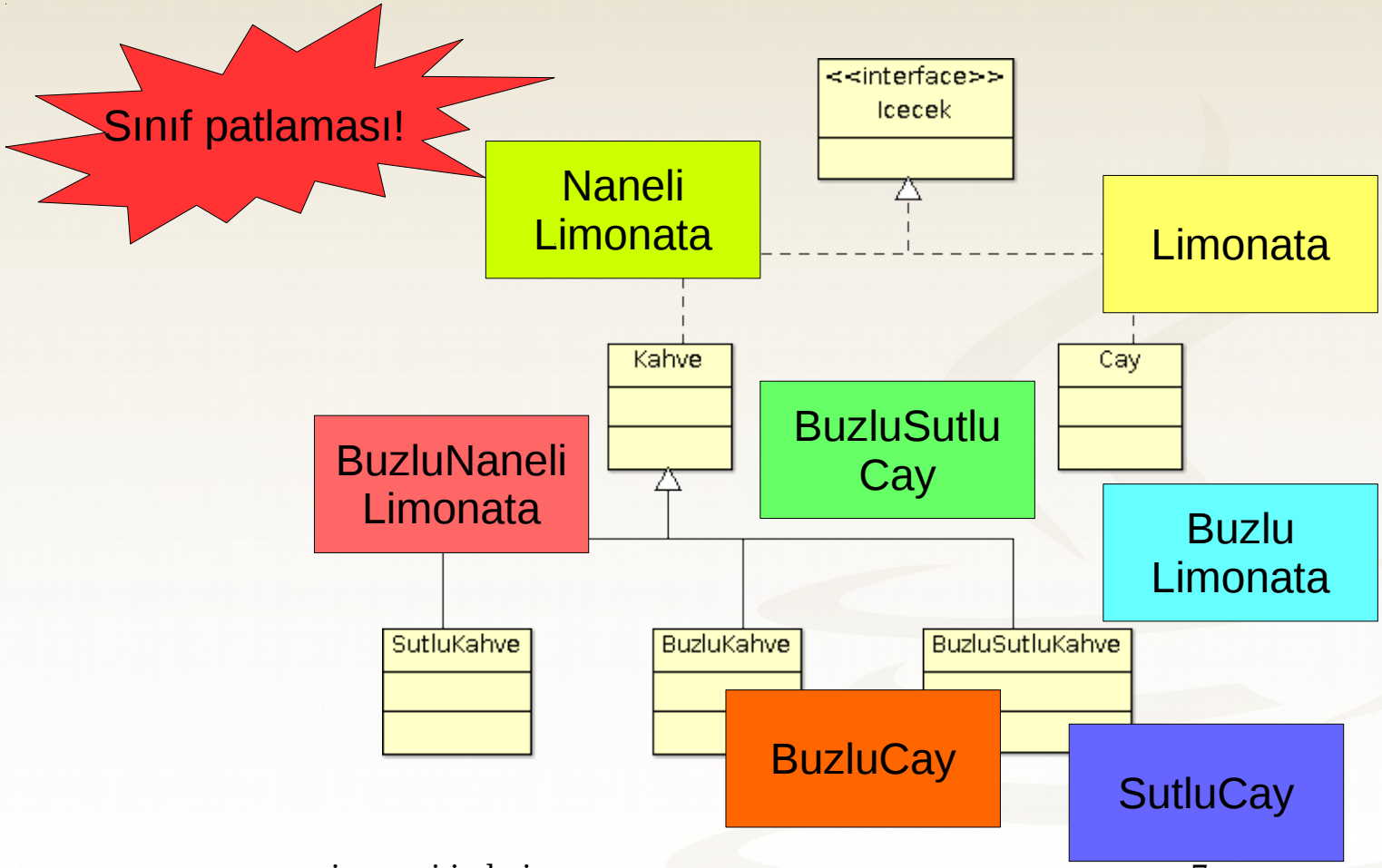
Nesnelere İlave Davranış Eklenmesi

- Nesnelere ilave davranış eklemek için ilk akla gelen yöntem **inheritance**
- Ancak kalıtım ile eklenen ilave davranışlar **yeterince esnek değil**
- Mevcut diğer özelliklerle yeni özelliğin kombinasyonu sonucu **sınıf hiyerarşilerinde geometrik artış!**

Inheritance ile İlave Davranış Eklenmesi



Inheritance ile İlave Davranış Eklenmesi



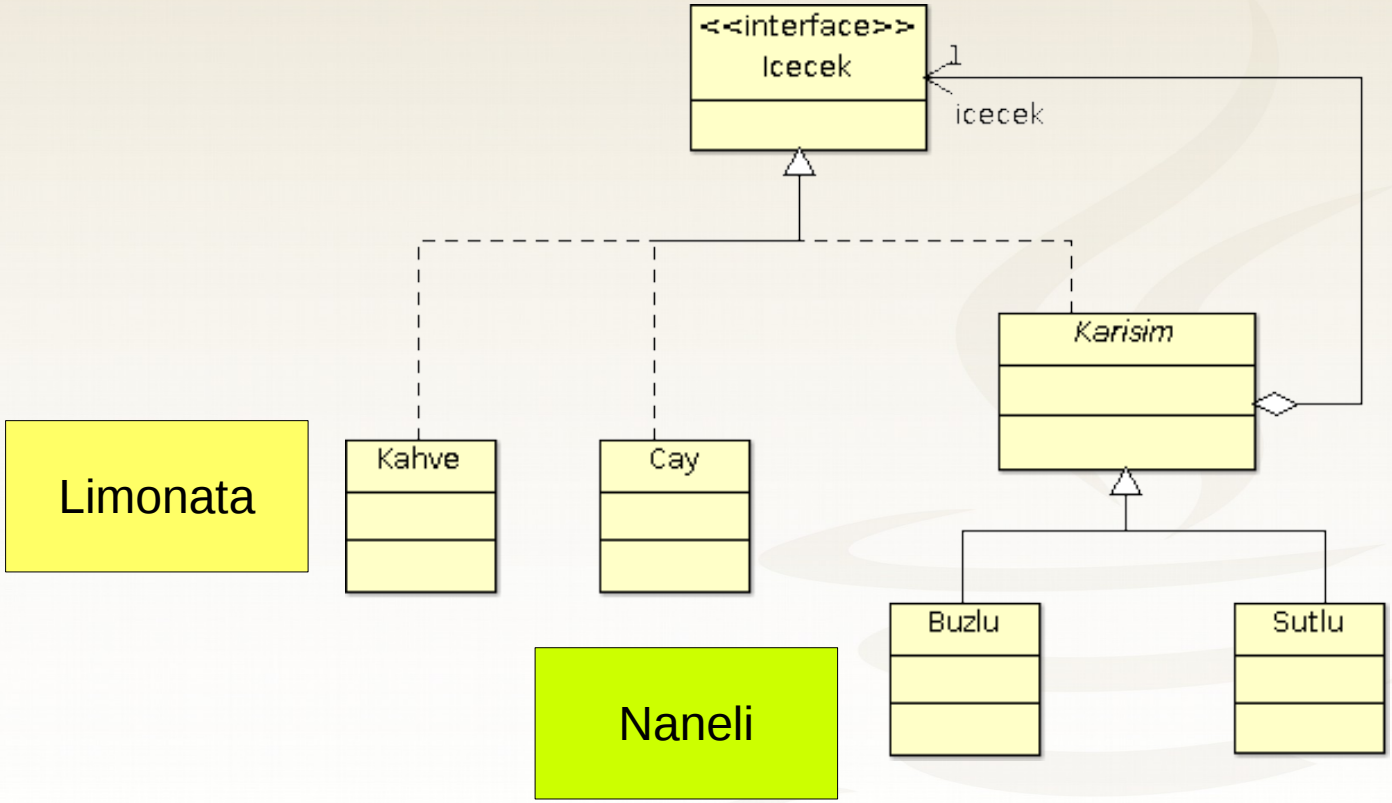
Nesnelere İlave Davranış Eklenmesi

- Object oriented çözümlerde **taksonomik sınıflandırma yaklaşımı**ndan uzak durun
- Bunun yerine nesneler arasındaki **işbirliği ve sorumluluk paylaşımı**na daha çok odaklanın

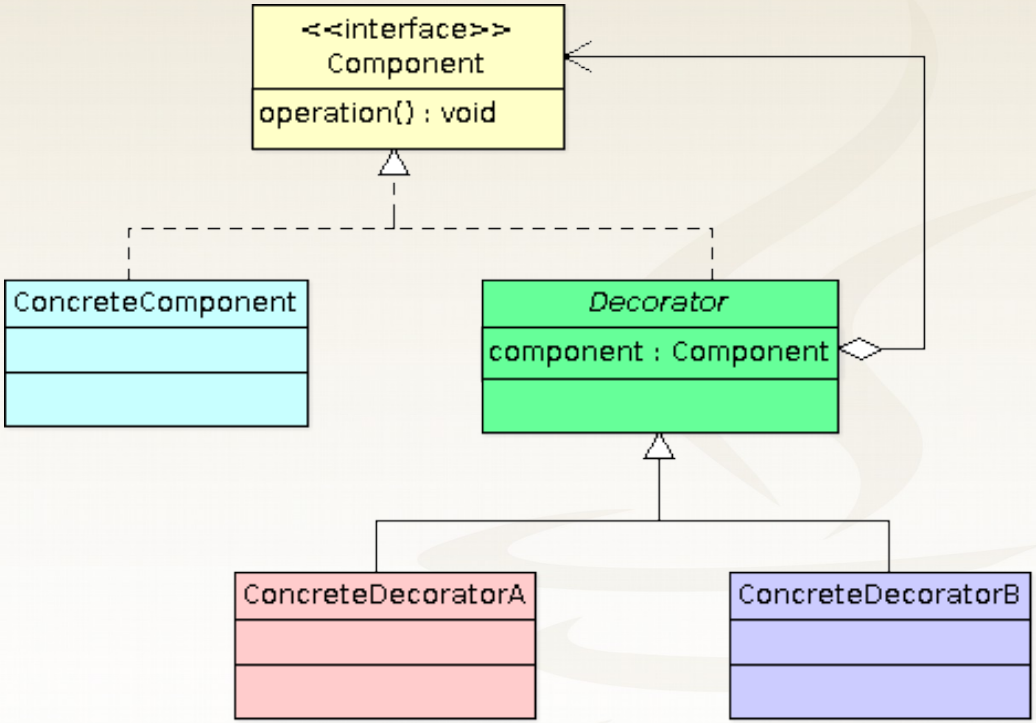
Inheritance Yerine Composition

- İlave davranışı kazandırmanın diğer bir yolu **composition**
- Mevcut sınıflarda ve metotlarda herhangi bir **değişikliğe gerek duyulmaz!**
- Sınıf hiyerarşilerinde **geometrik artışa neden olmaz!**

Composition ile İlave Davranış Eklenmesi



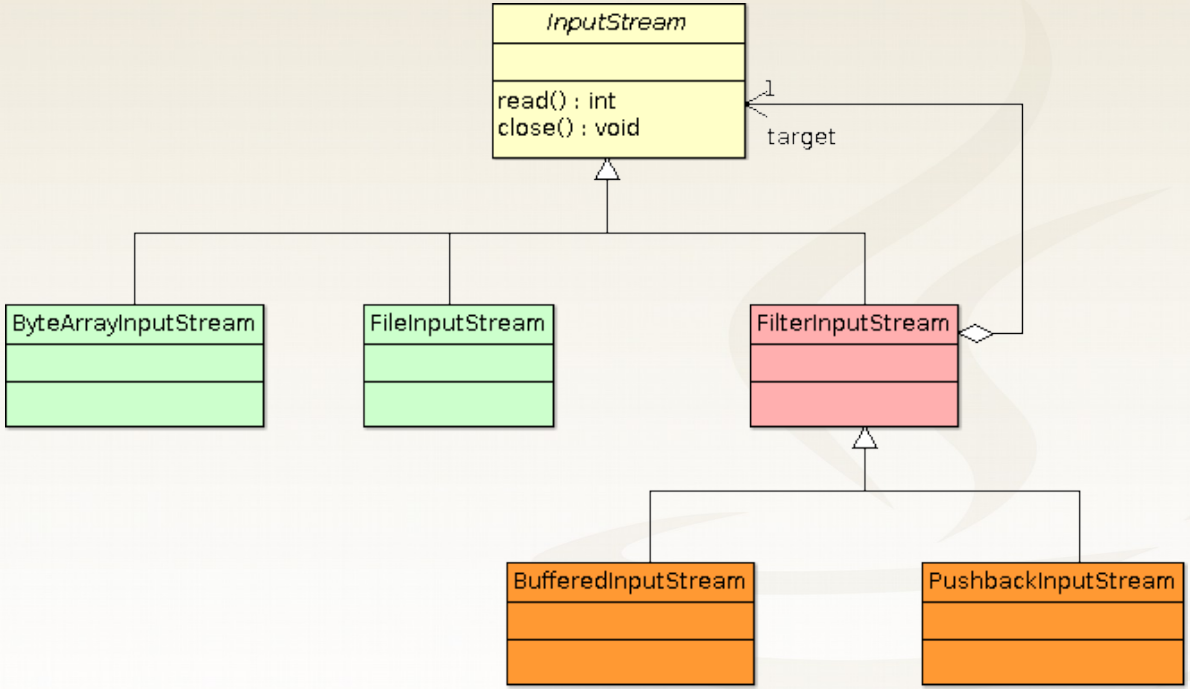
Decorator Sınıf Diagramı



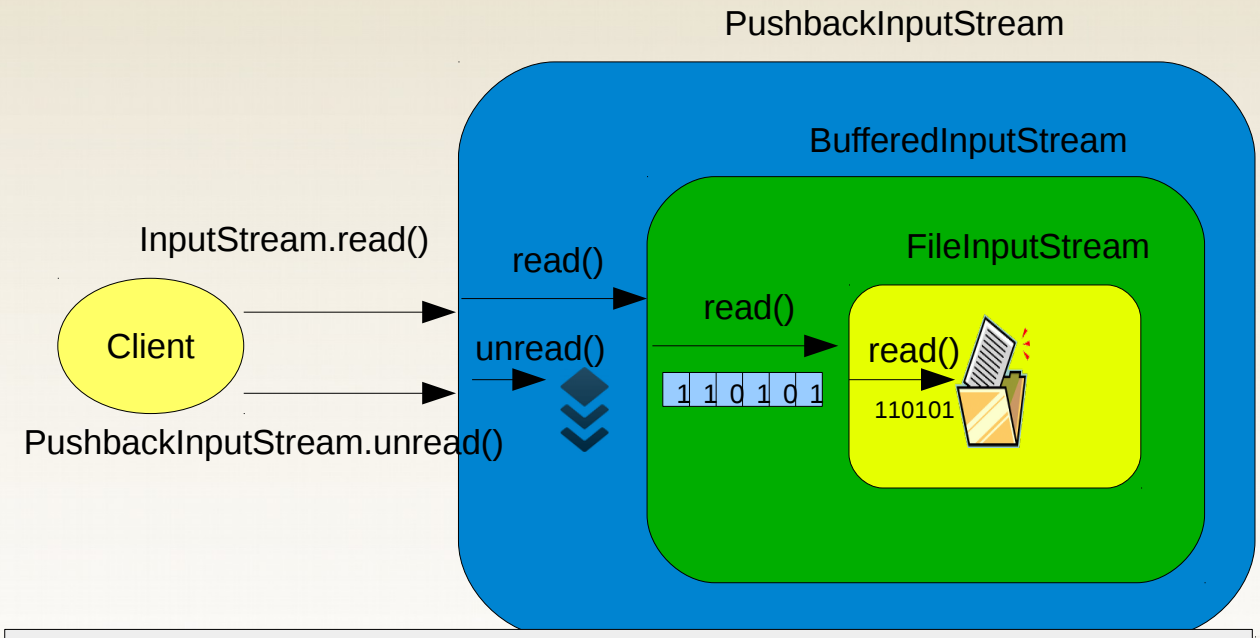
Java IO API ve Decorator

- InputStream/OutputStream
- Reader/Writer
- FileInputStream, ByteArrayInputStream, SocketInputStream
- İlave kabiliyetler: buffer kabiliyeti, pushback kabiliyeti, her ikisi birlikte...

Java IO API ve Decorator

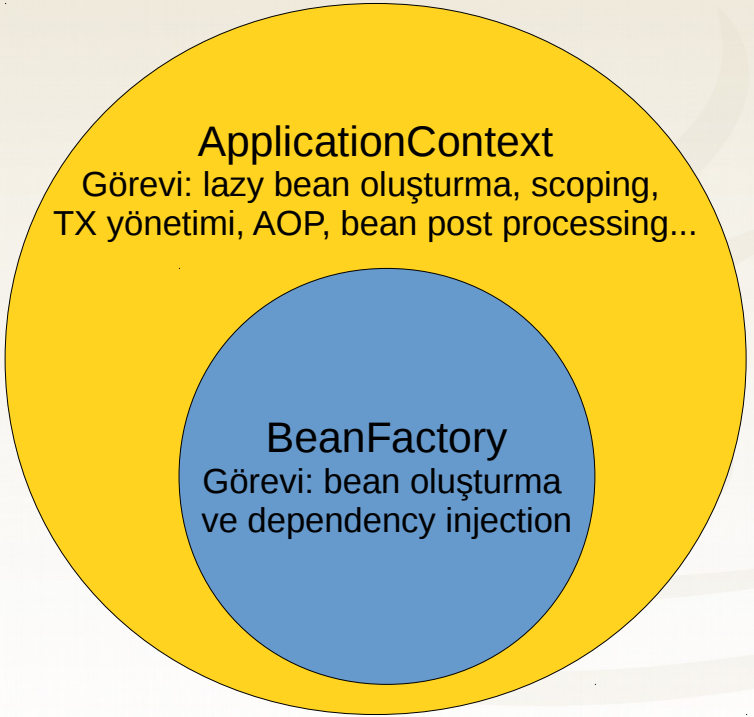


Java IO API ve Decorator

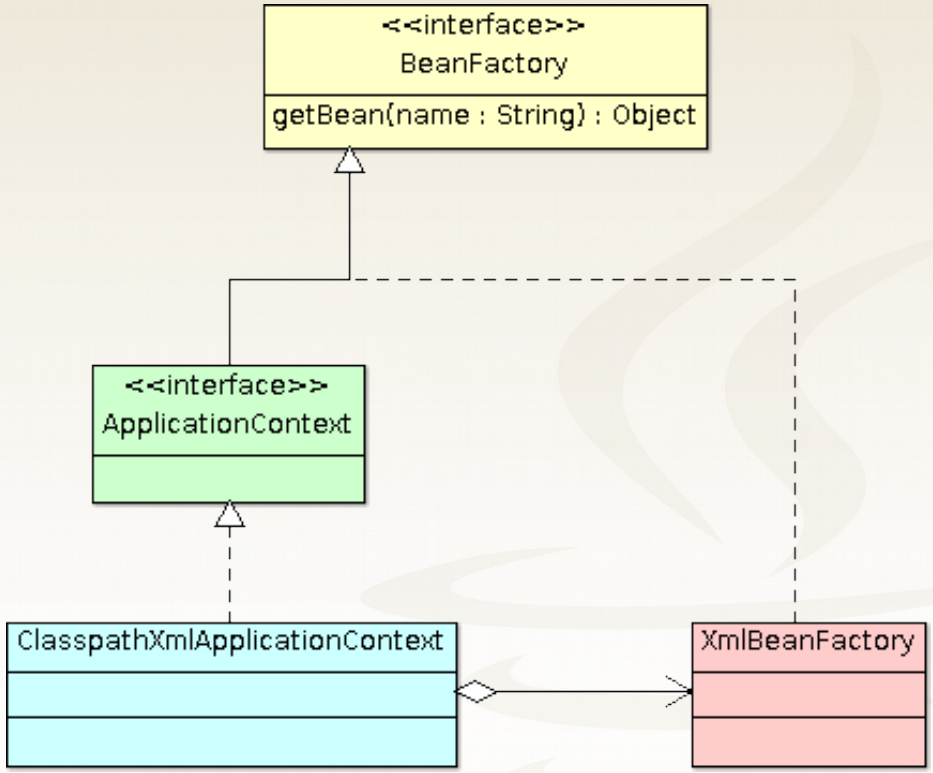



```
InputStream is = new PushbackInputStream(
                     new BufferedInputStream(
                          new FileInputStream ("/myfile.txt")));

is.read();

...

((PushbackInputStream)is).unread();
```

Spring ve Decorator



Spring ve Decorator



Decorator Örüntüsünün Sonuçları

- Yeni özellikler eklenirken sistemdeki sınıf sayısının **geometrik olarak artmasın**ın önüne geçilir
- Bir sınıfı tasarlarken sınıfın sahip olması gereken **ana sorumluluk ile** bu davranışın üzerine eklenebilecek **ilave kabiliyetler**in daha net ayrımına varılabilir
- Birden fazla kabiliyetin **bir sınıf içerisinde birikmesi**nin önüne geçmeye yardımcı olur

İletişim

JAVA Eğitimleri

Kurumsal Java Eğitimleri


www.java-egitimleri.com


info@java-egitimleri.com


@javaegitimleri

youtube.com/c/
KurumsalJavaEğitimleri