NESNE YÖNELİMLİ PROGRAMLAMA(Object Oriented Programming/OOP)

Öğr. Gör. Celil ÖZTÜRK
Marmara Üniversitesi
Teknik Bilimler Meslek Yüksekokulu

İçerik

- ✓Arayüzler(Interfaces)
- ✓Implement

Arayüzler(Interfaces)

- Arayüzler genelde programlama görevlerini tanımlamak için kullanılırlar.
- Arayüzler sınıf tasarımının bir özetini içerirler. Arayüzü uygulayan sınıflar, arayüzdeki tasarıma sadık bir şekilde uygulama gerçekleşir.
- Arayüzler, bir taslak olarak tanımlanır.
- Arayüzden(Interface) türeyen tüm sınıflar, interface metotlarını **implement** etmek **zorundadır**.
- **Türeyen sınıflar, interface'in içerisinde yer alan tüm metotları implement edeceğine dair söz verir !!!

Arayüzler(Interfaces)

- Çoklu kalıtım zor bir kavramdır. Çok fazla hata ile karşılaşılabilir (Constructor ?)
- Bu sebeple Java ve C# üzerinde çoklu kalıtıma izin verilmiyor.
- Çoklu kalıtım için Arayüz(Interface) adında bir alternatif sunulmaktadır.
- Soyut sınıfların içindeki bazı metotlar soyut olmayabiliyor. Ancak Interface'lerin içindeki tüm metotlar soyut olmalıdır.

Arayüzler(Interfaces)

- Arayüzler, bütün sınıfların gövdesiz olduğu soyut sınıflar gibi düşübülebilir.
- Bir arayüz içindeki tüm metotlar gövdesiz olmalıdır.
- Arayüz içindeki gövdesiz metotları abstract(soyut) olarak tanımlamak gerekmez, çünkü arayüzlerin içindeki metotların gövdesi olması zorunludur.

Arayüzler(Interfaces)

- Interface'ler metot imzaları içerirler.
- Interface'lerin içinde alan(field) tanımlanmaz.
- Interface'den new ile nesne örneklenemez.
- Constructor ve destructor tanımlanmaz.
- Metotlar için erişim belirleyicisi yazılmaz. Hepsi public olarak kabul edilir.
- Bir interface, başka bir sınıftan kalıtımla türeyemez.
- Bir sınıf bir Interface'den türetilmi ise Interface'den gelen bütün metotları uygulamak(implement) zorundadır.

Kaynak:MTA 98-361

Implements Anahtar Kelimesi

- Bir sınıf, bir arayüzdeki metotları uygulamak istiyorsa implements ifadesini kullanmak zorundadır.

```
class Kisi implements IKisi {

    @Override
    public void kisiBilgiGoster() {

        System.out.println("Ogrenci Bilgi goster metodu ....");

    }

}
```

```
public interface IKisi {
    public void kisiBilgiGoster();
}
```

Implements Anahtar Kelimesi

- Arayüzden(Interface) türeyen tüm sınıflar, interface metotlarını implement etmek zorundadır.
- **Türeyen sınıflar, interface'in içerisinde yer alan tüm metotları implement edeceğine dair söz verir !!!
- **Implement Etmek:Bir interface'i implement etmek, o interface'in şart koştuğu metotları yazmak....

Arayüz Kullanımı..

```
public interface IKisi {
    public void kisiBilgiGoster();
}

class Kisi implements IKisi {

    @Override
    public void kisiBilgiGoster() {

        System.out.println("Ogrenci Bilgi goster metodu ....");

    }

}
```

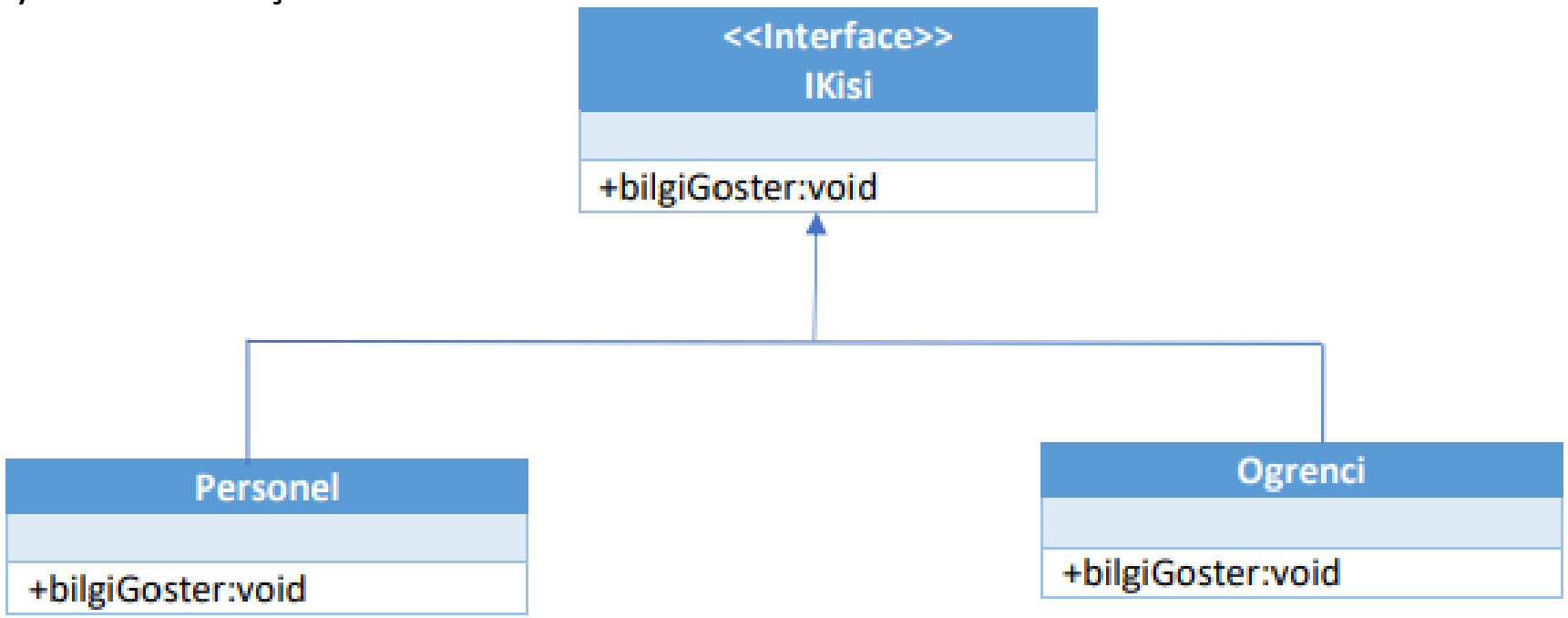
```
public class Arayuzler {


    public static void main(String[] args) {
        Kisi kisi1 = new Kisi();
        kisi1.kisiBilgiGoster();
    }

}
```

Arayüzler(Interfaces)

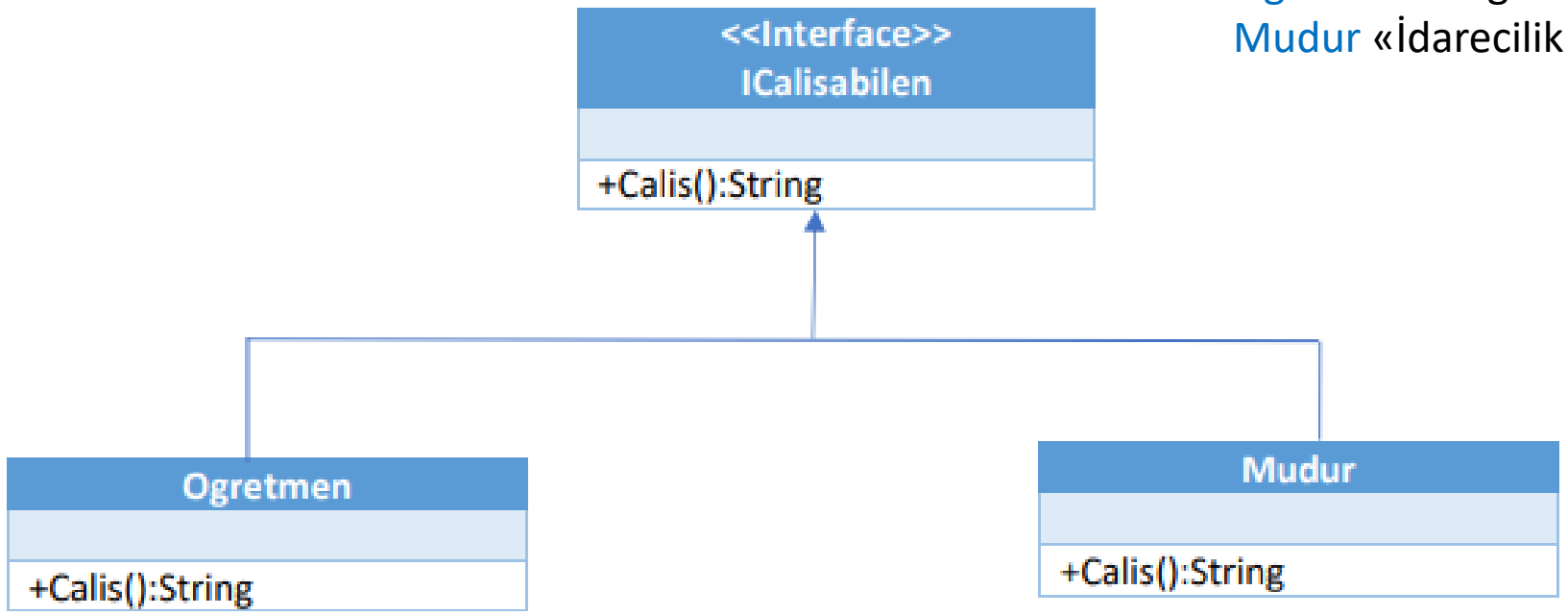
- Not:Soyut sınıftaki tüm metotlar soyut yapılırsa arayüz olarak kullanılabilir.
- Arayüzler **Sözleşme** yada **Kontrat** olarak kabul edilebilir. Bir sınıf arayüzü imlement ediyorsa sözleşmeye uymak zorundadır.

Arayüzler(Interfaces) UML Örnek

**** Örnek, Test adında bir arayüz eklenerek çoklu kalıtım kullanılabilir.



Arayüzler(Interfaces) UML Örnek

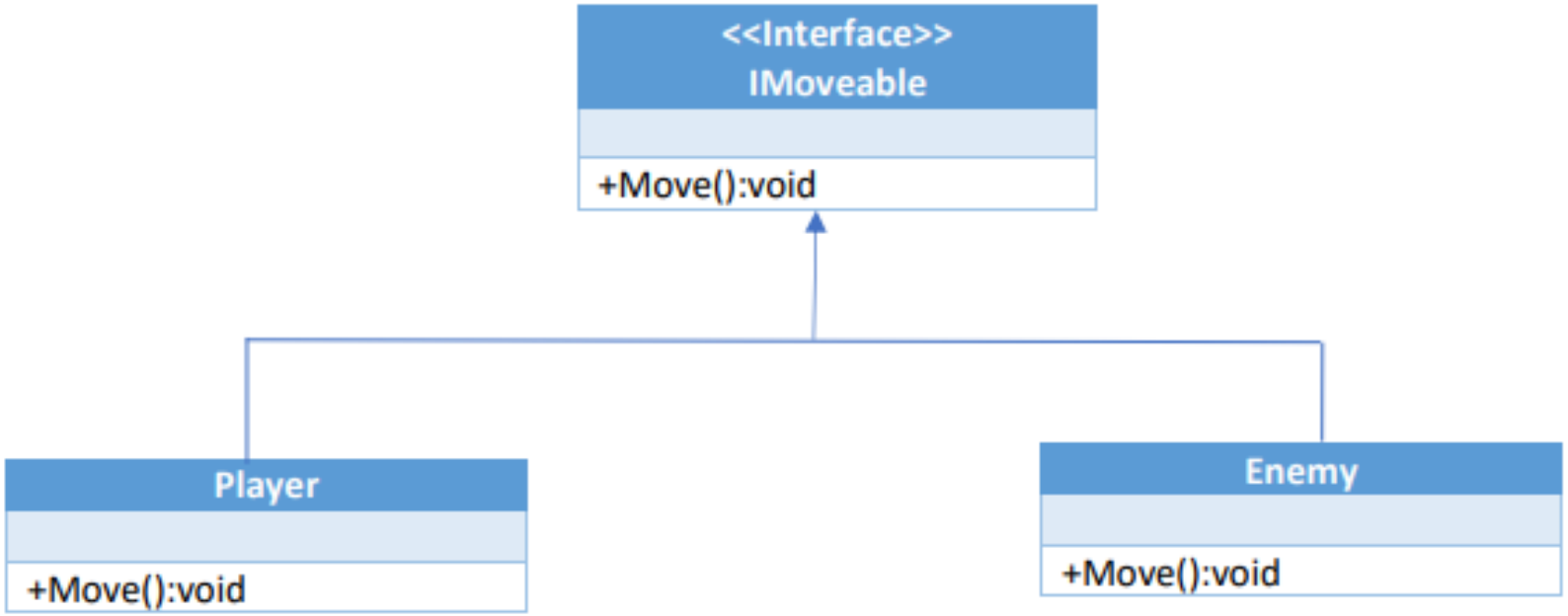


Sınıf Adı Calis() Metodunun Geriye Döneceği İfade

Ogretmen «Öğretirim…»

Mudur «İdarecilik yaparım…»

Arayüzler(Interfaces) UML Örnek



Interface(Arayüz) Kavramı

- Java'da interface, bir class'ta olması gereken method ve property'leri tanımlayan yapıdır.
- Kendisi normal bir class değildir, sadece neyin yapılacağını göstermekte, ancak nasıl yapılacağını göstermemektedir.

Kaynak:M.Ü TBMYO Aysun Zehra ALTIKARDEŞ NYP Notları.

Interface(Arayüz) Kavramı

```
public interface Matter{
public double getVolume();
public double getMass();
 }
```

- Bu ifade, 'bir maddenin yoğunluğu, hacmi ve kütlesi olur' demenin Java'daki yoludur

Kaynak:M.Ü TBMYO Aysun Zehra ALTIKARDEŞ NYP Notları.

INTERFACE IMPLEMENTASYONU

Bir class'ın interface'deki bütün method'ları içerdiğini, gerçekleştirdiğini belirtmesine implementation denir ve 'implements' keyword'üyle kullanılır.

Kaynak:M.Ü TBMYO Aysun Zehra ALTIKARDEŞ NYP Notları.

INTERFACE IMPLEMENTASYONU

```
public class CubeMatter implements Matter{
    public double density=1.0;
    public double edge=1.0;
    public double getDensity(){
    return density;
            }
    public double getVolume(){
    return edge*edge*edge;
    }
     public double getMass(){
    return density*edge*edge*edge;
    }
}
```

Kaynak:M.Ü TBMYO Aysun Zehra ALTIKARDEŞ NYP Notları.

INTERFACE IMPLEMENTASYONU

Bu örnekte "Küp diye bir nesnemiz var ve o bir maddedir, yani bir maddede olabilecek bütün nitelikler onda da bulunur." demiş olduk ve bunların nasıl hesaplandığını gösterdik.

INTERFACE IMPLEMENTASYONU

```
public class SphereMatter implements Matter{
    public double density=1.0;
    public double radius=1.0;
    public double getDensity(){
    return density;
    }
    public double getVolume(){
    return (3.14 * radius * radius * radius )/3;
    }
    public double getMass(){
    return density*(3.14 * radius * radius * radius )/3;
    }
}
```

Kaynak:M.Ü TBMYO Aysun Zehra ALTIKARDEŞ NYP Notları.

INTERFACE‘in KULLANIM ÖZELLİKLERİ

```
public interface MathConstants{
        double PI=3.14;
}

public class Circle implements MathConstants{
   private double radius=1.0;
   public getCircumference(){
      return 2*PI*radius;
   }
}
```

Kaynak:M.Ü TBMYO Aysun Zehra ALTIKARDEŞ NYP Notları.

Abstract(Soyut) Sınıf – Interface(Arayüz) Farkı

- Soyut sınıfların arayüzlerden farkı, soyut sınıflar soyut olmayan metotlar barındırabilmektedir. Arayüzlerin barındırdığı tüm metotlar soyut olmalıdır.
- Bir sınıf sadece bir sınıftan türetilebilir fakat, birden fazla arayüzü(Interface) imlement edebilir.
- Soyut sınıf veya Arayüz'ün hangi durumlarda kullanılacağını belirlemek zordur. Tüm metotlar ezilmek isteniyorsa arayüz, bazı metotların ezilmesi isteniyor ise soyut sınıf kullanılır.

Abstract(Soyut) Sınıf – Interface(Arayüz) Farkı

- Abstract sınıflar IS-A(dır,dir) –Kod tekrarını azaltmak,
- Interface ise CAN-DO(yapabilir) – Değişen kavramları uygulamadan soyutlamak,

şeklinde kullanılır.

**Nesnelerin daha spesifik bir örneğini oluşturmak istiyorsak soyut sınıflar, davranışlarının benzemesini istiyorsak arayüzler yaratılır. (CBU YZM-2105)

Kaynaklar

- Java ve Java Teknolojileri, *Tevfik KIZILÖREN* – Kodlab Yayınları
- Dr Öğr. Üyesi Aysun Zehra ALTIKARDEŞ- Nesne Yönelimli Programlama Notları
- Doç. Dr. Deniz KILINÇ CBÜ–Nesne Yönelimli Programlama Ders Notları
- Celal Bayar Üniversitesi - YZM 2105 Nesneye Dayalı Programlama Notları
- https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html#:~:text=An%20abstract%20class%20is%20a,but%20they%20can%20be%20subclassed.
- MTA 98-361 Software Development Fundamentals Notları
- IDE Danışmanlık MTA Notları
- https://amysgamedevelopmentblog.wordpress.com/2016/10/17/requirements-for-a-game-why-is-planning-important/