Nesne Yönelimli Programlama

Hazırlayan: M.Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Not: Bu dersin sunumları, “Java Programlama Dili ve Yazılım Tasarımı, Altuğ B. Altıntaş, Papatya Yayıncılık, 2016” kitabı kullanılarak hazırlanmıştır.

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

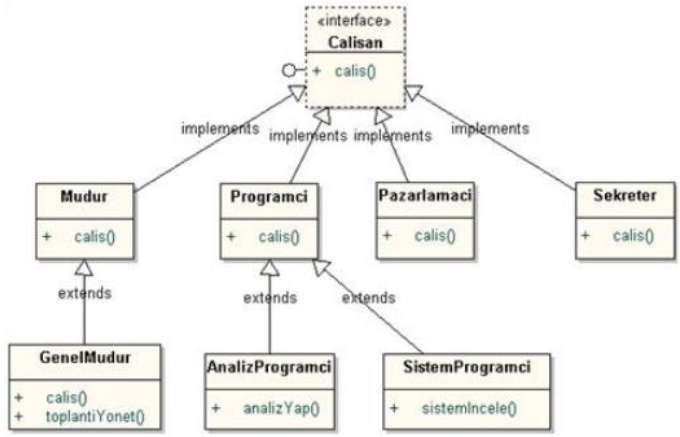
Arayüz ve Soyut Sınıflar

- Diğer programlama dillerinde olan **çoklu kalıtım (multiple inheritance)** Java programlama dilinde yoktur.
- Java programlama dilinde **çoklu kalıtım** için **arayüz (interface)** ve **dahili sınıflar (inner classes)** kullanılır.
- Arayüzler, soyut sınıfların bir üst modeli gibi düşünülebilir.
- **Soyut sınıflarda** hem **iş yapan** hem de **iş yapmayan** birleştirici yordamlar vardır.
- Birleştirici rol oynayan yordamlar, soyut sınıftan türetilmiş alt sınıflarda iptal edilmelidir (override).
- **Arayüzlerde** iş yapan herhangi bir yordam bulunamaz, **sadece gövdesiz (soyut) yordamlar bulunur.**
- Arayüzler, **birleştirici rol** oynamaları için tasarlanmıştır.
- Arayüzlerdeki **soyut yordamlar public** olmak zorundadır.
- Arayüzlerdeki **alanlar public**, **final** ve **statik** tanımlanır.

3

Arayüz ve Soyut Sınıflar

- Aşağıdaki Calisan arayüzü birleştiricidir.
- Calisan arayüzündeki soyut calis() yordamı, bu arayüze erişen tüm sınıflarda iptal edilmek zorundadır (override).



4

Arayüz ve Soyut Sınıflar

```
interface Calisan { // arayuz
    public void calis() ;
}

class Mudur implements Calisan {
    public void calis() { // iptal etti (override)
        System.out.println("Mudur Calisiyor");
    }
}

class GenelMudur extends Mudur {
    public void calis() { // iptal etti (override)
        System.out.println("GenelMudur Calisiyor");
    }
    public void toplantiYonet() {
        System.out.println("GenelMudur toplanti yonetiyor");
    }
}

class Programci implements Calisan {
    public void calis() { // iptal etti (override)
        System.out.println("Programci Calisiyor");
    }
}

class AnalizProgramci extends Programci {
    public void analizYap() {
        System.out.println("Analiz Yapiliyor");
    }
}
```

```
class SistemProgramci extends Programci {
    public void sistemIncele() {
        System.out.println("Sistem Inceleniyor");
    }
}

class Pazarlamaci implements Calisan {
    public void calis() { // iptal etti (override)
        System.out.println("Pazarlamaci Calisiyor");
    }
}

class Sekreter implements Calisan {
    public void calis() { // iptal etti (override)
        System.out.println("Sekreter Calisiyor");
    }
}

public class BuyukIsYeri {
    public static void mesaiBasla(Calisan[] c ) {
        for (int i = 0 ; i < c.length ; i++) {
            c[i].calis(); // ! Dikkat !
        }
    }

    public static void main(String args[]) {
        Calisan[] c = new Calisan[6];
        // c[0]=new Calisan(); ! Hata ! arayüz olusturulamaz
        c[0]=new Programci();     // yukari cevirim (upcasting)
        c[1]=new Pazarlamaci();  //yukari cevirim (upcasting)
        c[2]=new Mudur();        //yukari cevirim (upcasting)
        c[3]=new GenelMudur();   //yukari cevirim (upcasting)
        c[4]=new AnalizProgramci(); //yukari cevirim (upcasting)
        c[5]=new SistemProgramci(); //yukari cevirim  (upcasting)
        mesaiBasla(c);
    }
}
```

```
Programci Calisiyor
Pazarlamaci Calisiyor
Mudur Calisiyor
GenelMudur Calisiyor
Programci Calisiyor
Programci Calisiyor
```

5

Arayüz ve Soyut Sınıflar

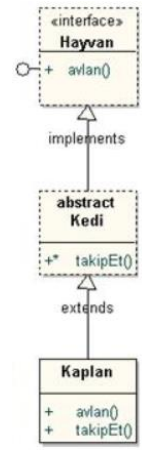
- Bir soyut sınıf, arayüz kullanılarak oluşturulabilir.
- Arayüz ile oluşturulan **soyut sınıf, soyut yordamları iptal etmek (override) zorunda değildir.**

```
interface Hayvan {
    public void avlan() ;
}

abstract class Kedi implements Hayvan {
    public abstract void takipEt() ;
}

class Kaplan extends Kedi {
    public void avlan() { // iptal etti (override)
        System.out.println("Kaplan avlaniyor...");
    }

    public void takipEt() { // iptal etti (override)
        System.out.println("Kaplan takip ediyor...");
    }
}
```



6

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

Arayüz ile Çoklu Kalıtım

- Bir sınıf, birden fazla arayüz ile oluşturulabilir.

```
interface BuzUstundeKayabilme {
    public void buzUstundeKay();
    public void sutAt();
}

interface SutAtabilme {
    public void sutAt();
}

class SportmenMehmet implements BuzUstundeKayabilme, SutAtabilme {
    public void buzUstundeKay() {
        System.out.println("SportmenMehmet buz ustunde kayiyor");
    }
    public void sutAt() {
        System.out.println("SportmenMehmet sut atiyor");
    }
    public static void main(String args[]) {
        SportmenMehmet sm = new SportmenMehmet();
        sm.sutAt();
    }
}
```

```
public void sutAt(int sutsayisi);
```

?

```
SportmenMehmet sut atiyor
```

8

Arayüz ile Çoklu Kalıtım

- Tüm arayüzlerdeki soyut yordamlar override yapılmalıdır.

```
interface BuzUstundeKayabilme {
    public void buzUstundeKay();
    public void sutAt(int sutsayisi);
}

interface SutAtabilme {
    public void sutAt();
}

class SportmenMehmet implements BuzUstundeKayabilme, SutAtabilme {
    public void buzUstundeKay() {
        System.out.println("SportmenMehmet buz ustunde kayiyor");
    }
    public void sutAt(int sutsayisi) {
        System.out.println("SportmenMehmet sut atiyor= " + sutsayisi);
    }
    public void sutAt() {
        System.out.println("SportmenMehmet sut atiyor");
    }
    public static void main(String args[]) {
        SportmenMehmet sm = new SportmenMehmet();
        sm.sutAt(5);
    }
}
```

```
SportmenMehmet sut atiyor
```

9

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

Arayüzlerin Kalıtım ile Genişletilmesi

- Arayüzler kalıtım ile türetilerek genişletilebilir.

```
interface Kosabilme {
    public void kos();
}
interface DahaHizliKosabilme extends Kosabilme {
    public void dahaHizliKos();
}
interface Avlanabilme extends DahaHizliKosabilme {
    public void avlan();
}
class RoadRunner implements DahaHizliKosabilme {
    public void kos() {
        System.out.println("RoadRunner kosuyor, bip ");
    }
    public void dahaHizliKos() {
        System.out.println("RoadRunner kosuyor, bip bippp");
    }
}
public class Jaguar implements Avlanabilme {
    public void avlan() {
        System.out.println("Juguar avlaniyor");
    }
    public void dahaHizliKos() {
        System.out.println("Juguar daha hizli kos");
    }
    public void kos() {
        System.out.println("Juguar Kosuyor");
    }
}
```

11

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

Çakışmalar

- Çoklu arayüz ile sınıf oluşturulduğunda, farklı arayüzlerdeki aynı başlığa sahip yordamlarda çakışma olmaz.

```
interface BuzUstundeKayabilme {
    public void buzUstundeKay();
    public void sutAt();
}

interface SutAtabilme {
    public void sutAt();
}

class SportmenMehmet implements BuzUstundeKayabilme, SutAtabilme {
    public void buzUstundeKay() {
        System.out.println("SportmenMehmet buz ustunde kayiyor");
    }
    public void sutAt() {
        System.out.println("SportmenMehmet sut atiyor");
    }
    public static void main(String args[]) {
        SportmenMehmet sm = new SportmenMehmet();
        sm.sutAt();
    }
}
```

```
SportmenMehmet sut atiyor
```

13

Çakışmalar

- Çoklu arayüz ile sınıf oluşturulduğunda, arayüzlerdeki farklı dönüş türüne sahip yordamlarda çakışma hatası olur.

```
interface A1 {
    public void hesapla();
}
interface A2 {
    public void hesapla(int d);
}
interface A3 {
    public int hesapla();
}
class S1 implements A1,A2 { // sorunsuz
    public void hesapla() { //adas yordamlar(overloaded)
        System.out.println("S1.hesapla");
    }
    public void hesapla(int d) { //adas yordamlar(overloaded)
        System.out.println("S1.hesapla " + d );
    }
}
/* ! Hatali !, adas yordamlar (overloaded)
    donus tiplerine gore ayirt edilemez. */

class S2 implements A1,A3 {
    public void hesapla() {
        System.out.println("S2.hesapla");
    }
    /* ! Hata !
      public int hesapla() {
          System.out.println("S2.hesapla");
          return 123;
      }
     */
}
```

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

Arayüzlerde Alanlar

- Arayüzlerde global alanların değeri atanmak zorundadır.
- Bu alanlar, otomatik olarak public, final ve static tir.

```
interface Aylar {
  int
  OCAK = 1, SUBAT = 2, MART = 3,
  NISAN = 4, MAYIS = 5, HAZIRAN = 6, TEMMUZ = 7,
  AGUSTOS = 8, EYLUL = 9, EKIM = 10,
  KASIM = 11, ARALIK = 12;
}

public class Deneme {
  public static void main(String args[]) {

      int ay = (int)(Math.random()*13) ;
      System.out.println("Gelen ay = " + ay);
      switch ( ay ) {
        case Aylar.OCAK : System.out.println("Ocak");break;
        case Aylar.SUBAT : System.out.println("Subat");break;
        case Aylar.MART : System.out.println("Mart");break;
        case Aylar.NISAN : System.out.println("Nisan");break;
        case Aylar.MAYIS : System.out.println("Mayis");break;
        case Aylar.HAZIRAN : System.out.println("Haziran");break;
        case Aylar.TEMMUZ : System.out.println("Temmuz");break;
        case Aylar.AGUSTOS : System.out.println("Agustos");break;
        case Aylar.EYLUL : System.out.println("Eylul");break;
        case Aylar.EKIM : System.out.println("Ekim");break;
        case Aylar.KASIM : System.out.println("Kasim");break;
        case Aylar.ARALIK : System.out.println("Aralik");break;
        default:System.out.println("Tanimsiz Ay");
      }
  }
}
```

```
Gelen ay = 2
Subat
```

Arayüzlerde Alanlar

- Arayüzlerdeki alanlara ilk değeri run-time'da verilebilir.

```
interface A7 {

    int devir_sayisi = (int) ( Math.random() *6 ) ;
    String isim = "A7" ;
    double t1 = ( Math.random() * 8 ) ;
}

public class Test {
    public static void main(String args[] ) {

        System.out.println("devir_sayisi = " + A7.devir_sayisi );
        System.out.println("isim = " + A7.isim );
        System.out.println("t1 = " + A7.t1 );
    }
}
```

```
devir_sayisi = 4
isim = A7
t1 = 2.819430819690039
```

17

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

Arayüzler ve Yukarı Çevrim

- Arayüzlerde çoklu yukarı çevrim yapılabilir.

```
interface Arayuz1 {
    public void a1() ;
}
interface Arayuz2 {
    public void a2() ;
}
abstract class Soyut1 {
    public abstract void s1();
}

class A extends Soyut1 implements Arayuz1, Arayuz2 {
    public void s1() { // iptal etti (override)
        System.out.println("A.s1()");
    }
    public void a1() { // iptal etti (override)
        System.out.println("A.a1()");
    }
    public void a2() { // iptal etti (override)
        System.out.println("A.a2()");
    }
}
```

19

Arayüzler ve Yukarı Çevrim

- Arayüzlerde çoklu yukarı

```
public class Deneme {
    public static void main(String args[]) {
        Soyut1 soyut_1 = new A();
        Arayuz1 arayuz_1 = (Arayuz1) soyut_1 ;
        Arayuz2 arayuz_2 = (Arayuz2) soyut_1 ;
        // Arayuz2 arayuz_2 = (Arayuz2) arayuz_1 ; // dogru

        soyut_1.s1();
        // soyut_1.a1(); // ! Hata !
        // soyut_1.a2(); // ! Hata !

        arayuz_1.a1();
        // arayuz_1.a2(); // ! Hata !
        // arayuz_1.s1(); // ! Hata !

        arayuz_2.a2();
        // arayuz_2.a1(); // ! Hata !
        // arayuz_2.s1(); // ! Hata !
    }
}
```

Sadece s1() yordamına erişebilir.

Sadece a1() yordamına erişebilir.

Sadece a2() yordamına erişebilir.

```
Soyut1 soyut_1 = new A();
Arayuz1 arayuz_1 = new A();
Arayuz2 arayuz_2 = new A();
```

?

```
A.s1()
A.a1()
A.a2()
```

20

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

Dahili Arayüzler

- Bir arayüz, başka bir arayüzün içinde tanımlanabilir.
- Bir arayüzün içerisinde tanımlanan dahili arayüzler, protected, friendly veya private erişim belirleyicisine sahip olamaz.

```
interface ArayuzA { //aslinda public erisim belirleyicisine sahip
    public interface DahiliArayuz1 {
        public void isYap1() ;
    }
    /* // ! Hata !
      protected interface DahiliArayuz2 {
          public void isYap2() ;
      }
     */
    interface DahiliArayuz3 { // aslinda public erisim belirleyicisine sahip
        public void isYap3() ;
    }
    /* // ! Hata !
      private interface DahiliArayuz4 {
          public void isYap4() ;
      }
     */
}
```

Dahili Arayüzler

```
class Erisim1 implements ArayuzA.DahiliArayuz1 {
    public void isYap1() {
        System.out.println("Erisim1.isYap1()");
    }
}
class Erisim2 implements ArayuzA.DahiliArayuz3 {
    public void isYap3() {
        System.out.println("Erisim1.isYap3()");
    }
}
public class Deneme {
    public static void main(String args[]) {
        Erisim1 e1 = new Erisim1();
        Erisim2 e2 = new Erisim2();
        e1.isYap1();
        e2.isYap3();
    }
}
```

```
Erisim1.isYap1()
Erisim1.isYap3()
```

23

Dahili Arayüzler

- Bir arayüz, başka bir sınıfın içinde tanımlanabilir.

```
public class SinifA {

    public interface A1 {
        public void ekranaBas();
    } //arayüz

    public class DahiliSinif1 implements A1 {
        public void ekranaBas() {
            System.out.println("DahiliSinif1.ekranaBas()");
        }
    } // class DahiliSinif1

    protected class DahiliSinif2 implements A1 {
        public void ekranaBas() {
            System.out.println("DahiliSinif2.ekranaBas()");
        }
    } // class DahiliSinif2

    class DahiliSinif3 implements A1 {
        public void ekranaBas() {
            System.out.println("DahiliSinif3.ekranaBas()");
        }
    } // class DahiliSinif3

    private class DahiliSinif4 implements A1 {
        public void ekranaBas() {
            System.out.println("DahiliSinif4.ekranaBas()");
        }
    } // class DahiliSinif4
```

Dahili Arayüzler

```
    public static void main(String args[]) {
        SinifA.DahiliSinif1 sad1 = new SinifA().new DahiliSinif1();
        SinifA.DahiliSinif2 sad2 = new SinifA().new DahiliSinif2();
        SinifA.DahiliSinif3 sad3 = new SinifA().new DahiliSinif3();
        SinifA.DahiliSinif4 sad4 = new SinifA().new DahiliSinif4();

        sad1.ekranaBas();
        sad2.ekranaBas();
        sad3.ekranaBas();
        sad4.ekranaBas();

        SinifB sb = new SinifB();
        sb.ekranaBas();
    }
}

class SinifB implements SinifA.A1{
    public void ekranaBas() {
        System.out.println("SinifB.ekranaBas()");
    }
}
```

```
DahiliSinif1.ekranaBas()
DahiliSinif2.ekranaBas()
DahiliSinif3.ekranaBas()
DahiliSinif4.ekranaBas()
SinifB.ekranaBas()
```

25

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

Dahili Sınıflar

- Dahili sınıf özelliği sayesinde bir sınıf diğer bir sınıfın içerisinde tanımlanabilir.
- Dahili sınıflar yapısal olarak 3 gruba ayrılır:
  - Dahili üye sınıflar
  - Yerel sınıflar
  - İsimsiz sınıflar

27

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

Dahili Üye Sınıflar

- Bir sınıfın içerisinde tanımlanan sınıfa dahili üye sınıf denir.
- Hesaplama sınıfının içerisinde tanımlanmış Toplama sınıfı bir dahili üye sınıfıdır.
- Hesaplama sınıfı ise çevreleyici sınıftır.
- Toplama sınıfına ait bir nesne oluşturmak için, önce Hesaplama sınıfına ait bir nesne oluşturmamız gerekir.

```
public class Hesaplama {

    public class Toplama { //Dahili uye sinif
        public int toplamaYap(int a, int b) {
            return a+b ;
        }
    } // class Toplama

    public static void main(String args[]) {
        Hesaplama.Toplama ht = new Hesaplama().new Toplama();
        int sonuc = ht.toplamaYap(3,5);
        System.out.println("Sonuc = " + sonuc );
    }
} // class Hesapla
```

29

Dahili Üye Sınıflar

- Dahili üye sınıflara, public, friendly, protected veya private erişim belirleyicileri atanabilir.

```
public class Hesaplama1 {
    public class Toplama { // Dahili uye sinif - public
        public int toplamaYap(int a, int b) {
            return a + b ;
        }
    } // class Toplama
    protected class Cikartma { // Dahili uye sinif - protected
        public int cikartmaYap(int a, int b) {
            return a - b ;
        }
    } // class Cikartma
    class Carpma { // Dahili uye sinif - friendly
        public int carpmaYap(int a, int b) {
            return a * b ;
        }
    } // class Carpma
    private class Bolme { // Dahili uye sinif - private
        public int bolmeYap(int a, int b) {
            return a / b ;
        }
    } // class Bolme
```

30

Dahili Üye Sınıflar

- Dahili üye sınıflara, public, friendly, protected veya private erişim belirleyicileri atanabilir.

```
    public static void main(String args[]) {
        Hesaplama1.Toplama ht = new Hesaplama1().new Toplama() ;
        Hesaplama1.Cikartma hck = new Hesaplama1().new Cikartma() ;
        Hesaplama1.Carpma hcp = new Hesaplama1().new Carpma() ;
        Hesaplama1.Bolme hb = new Hesaplama1().new Bolme() ;
        int sonuc1 = ht.toplamaYap(10,5);
        int sonuc2 = hck.cikartmaYap(10,5);
        int sonuc3 = hcp.carpmaYap(10,5);
        int sonuc4 = hb.bolmeYap(10,5);
        System.out.println("Toplama Sonuc = " + sonuc1 );
        System.out.println("Cikartma Sonuc = " + sonuc2 );
        System.out.println("Carpma Sonuc = " + sonuc3 );
        System.out.println("Bolme Sonuc = " + sonuc4 );
    }
} // class Hesaplama
```

```
Toplama Sonuc = 15
Cikartma Sonuc = 5
Carpma Sonuc = 50
Bolme Sonuc = 2
```

31

Dahili Üye Sınıflar

- private üyelere sınıf dışından erişim yapılamaz.

```
class Hesaplama2 {
    public class Toplama2 { // Dahili uye sinif - public
        public int toplamaYap(int a, int b) {
            return a + b ;
        }
    } // class Toplama2
    protected class Cikartma2 {  // Dahili uye sinif - protected
        public int cikartmaYap(int a, int b) {
            return a - b ;
        }
    } // class Cikartma2
    class Carpma2 { // Dahili uye sinif - friendly
        public int carpmaYap(int a, int b) {
            return a * b ;
        }
    } // class Carpma2
    private class Bolme2 { // Dahili uye sinif - private
        public int bolmeYap(int a, int b) {
            return a / b ;
        }
    } // class Bolme2
} // class Hesaplama2
```

32

Dahili Üye Sınıflar

- private üyelere sınıf dışından erişim yapılamaz.

```
public class Hesaplama2Kullan {
    public static void main(String args[]) {

        Hesaplama2.Toplama2 ht=new Hesaplama2().new Toplama2() ;
        Hesaplama2.Cikartma2 hck=new Hesaplama2().new Cikartma2() ;
        Hesaplama2.Carpma2 hcp = new Hesaplama2().new Carpma2() ;
        // Hesaplama2.Bolme2 hb = new Hesaplama2().new Bolme2() ;
        // ! Hata !

        int sonuc1 = ht.toplamaYap(10,5);
        int sonuc2 = hck.cikartmaYap(10,5);
        int sonuc3 = hcp.carpmaYap(10,5);
        // int sonuc4 = hb.bolmeYap(10,5); // ! Hata !

        System.out.println("Toplama Sonuc = " + sonuc1 );
        System.out.println("Cikartma Sonuc = " + sonuc2 );
        System.out.println("Carpma Sonuc = " + sonuc3 );
    }
}
```

```
Toplama Sonuc = 15
Cikartma Sonuc = 5
Carpma Sonuc = 50
```

33

Dahili Üye Sınıflar

- Statik dahili üye sınıfına ait nesne oluşturmak için, onu çevreleyen sınıfa ait bir nesne oluşturmak zorunlu değildir.
- Statik dahili üye sınıflar, kendilerini çevreleyen üst sınıfa ait bağlantıyı (this) kaybederler.
- Statik dahili üye sınıflar, kendilerini çevreleyen üst sınıfa ait global alanlara ve yordamlara direk ulaşımı kaybederler.

34

Dahili Üye Sınıflar

- Örnek

```
class Hesaplama4 {
    int sabit  = 2 ;
    private int ozelsabit = 1 ;
    public static class Toplama4 {   // Statik uye dahili sinif
        static int toplam ;       // dogru
        int sonuc ;    // dogru
        public  int toplamaYap(int a, int b) {
            // return  (a+b)  + sabit  ;  ! Hata !
            sonuc = toplam = a+b ;
            return sonuc    ;
        }
        public void dekontOlustur() {
            /*  -sabit- alanina  ve
                -ekranaBas() yordamına ulasabilmek icin
                Hesaplama4 sinifina ait nesne olusturmamiz gerekir.
             */
            Hesaplama4 hs4 = new Hesaplama4(); //dikkat
            int a =  hs4.ozelsabit ; // dogru
            hs4.ekranaBas() ; //dogru
            System.out.println("Dekont olusturuyor = " +
                    hs4.sabit + " - "  +a  );
        }
    }  // class Toplama4
```


Dahili Üye Sınıflar

- Örnek

```
    public class Cikartma4 { //Uye dahili sinif
        int sonuc ;
        // static int sonuc1 ; // ! hata !
        public  int cikartmaYap(int a, int b) {
            ekranaBas();  // dikkat
            sonuc = (a-b) -  ozelsabit ;
            return sonuc   ; // dikkat
        }
    }  // class Cikartma4
    private void ekranaBas() {
        System.out.println("Hesaplama4.ekranaBas()");
    }
    public static void main(String args[]) {

        // !  Hata  !
        // Hesaplama4.Toplama4 ht=new Hesaplama4().new Toplama4();
        Toplama4  tp4  =  new Toplama4();
        tp4.dekontOlustur();
        int sonuc = tp4.toplamaYap(10,5);
        System.out.println("Sonuc  =  10 + 5 =  " + sonuc );
    }

} // class Hesaplama4
public class Hesaplama4Kullan {
    public static void main(String args[]) {
        // !  Hata  !
        // Hesaplama4.Toplama4 ht=new Hesaplama4().new Toplama4() ;
        Hesaplama4.Toplama4  tp4  =  new Hesaplama4.Toplama4();
        int sonuc = tp4.toplamaYap(10,5);
        System.out.println("Sonuc  =  10 + 5   =  " + sonuc );
    }
}  // class  Hesaplama4Kullan
```

```
Sonuc  =  10 + 5   =  15
```

Dahili Üye Sınıflar

- statik dahili üye sınıf içerisindeki statik bir yordamı çağırmak için ne statik dahili üye sınıfına ne de onu çevreleyen sınıfa ait nesne oluşturmak gerekmez.

```
public class Hesaplama5 {
    private static int  x = 3 ;

    public static class Toplama5 { // Statik uye dahili sinif
        static int toplam ; // dogru
        int sonuc ;       // dogru
        public  static int toplamaYap(int a, int b) {
            // sonuc = a+b + x ;  // ! Hata !
            toplam = a + b + x ;
            return toplam    ;
        }
    } // class Toplama5

    public static void main(String args[]) {
        int sonuc = Hesaplama5.Toplama5.toplamaYap(16,8); // dikkat
        System.out.println("Sonuc   =  16 + 8  =  " + sonuc );
    }
} // class Hesaplama5
```

```
Sonuc   =  16 + 8  =  27
```


Dahili Üye Sınıflar

- Statik olmayan dahili üye sınıfların içinde, statik alan ve yordam tanımlanamaz, "statik ve final" alan tanımlanabilir.
- Bir alan hem statik hem de final ise, SABİT değere sahiptir.
- Statik olmayan dahili üye sınıfların içerisinde statik ve final alanlar kullanılabilir.

```
class CevreliyiciSinif1 {

    class DahiliSinif1 { // Dahili uye siniflar
        // static int x = 10 ; // ! Hata !
    }
}
// Dogru
class CevreliyiciSinif2 {

    class DahiliSinif2 {
        int x; // Dogru
    }
}
// Dogru
class CevreliyiciSinif3 {

    class DahiliSinif3 {
        static final int x = 0;  // Dogru
    }
}
```

Dahili Üye Sınıflar

- Dahili üye sınıfların yapılandırıcıları olabilir.
- Dahili üye sınıfın üst sınıfına ait bir nesne oluşturulduğunda, dahili üye sınıfına ait bir nesne otomatik oluşturulmaz.
- Örnekte sadece BuyukA sınıfına ait bir nesne oluşturulmuştur.
- Sadece BuyukA sınıfına ait yapılandırıcı çağrılır.
- Dahili üye B sınıfına ait yapılandırıcının çağrılması istenseydi, main() içerisinde " BuyukA.newB() " gerekirdi.

```
public class BuyukA {

    public class B {
        public B() { // yapilandirici
            System.out.println("Ben B sinifi ");
        }
    }  // class B

    public BuyukA() {
        System.out.println("Ben BuyukA sinifi ");
    }

    public static void main(String args[]) {
        BuyukA ba = new BuyukA();
    }
}
```

```
Ben BuyukA sinifi
```

39

Dahili Üye Sınıflar

- Bir sınıfın içerisinde içiçe dahili üye sınıflar tanımlanabilir.

```
public class Abc {

    public Abc() {       // Yapilandirici
        System.out.println("Abc nesnesi olusturuluyor");
    }

    public class Def {
        public Def() { // Yapilandirici
            System.out.println("Def nesnesi olusturuluyor");
        }

        public class Ghi {
            public Ghi() { // Yapilandirici
                System.out.println("Ghi nesnesi olusturuluyor");
            }

        }  // class Ghi

    } //class Def

    public static void main( String args[] ) {
        Abc.Def.Ghi  ici_ice = new Abc().new Def().new Ghi();
    }

} // class Abc
```

```
Abc nesnesi olusturuluyor
Def nesnesi olusturuluyor
Ghi nesnesi olusturuluyor
```

40

Dahili Üye Sınıflar

- Dahili üye sınıflar, soyut sınıf olarak tanımlanabilir.
- Soyut dahili üye sınıflardan türetilen sınıflar, soyut dahili üye sınıfların içerisindeki soyut yordamları iptal etmek zorundadır.

```
class Hayvan {
    abstract class Kus {
        public abstract void uc ();
        public abstract void kon();
    }
    public void avlan() {
        System.out.println("Hayvan avlaniyor...");
    }
}
public class Kartal extends Hayvan.Kus {
    public void uc() {
        System.out.println("Kartal Ucuyor...");
    }
    public void kon() {
        System.out.println("Kartal Konuyor...");
    }
    // public Kartal() {  } // ! Hata !
    public Kartal(Hayvan hv) {
        hv.super(); //Dikkat
    }
    public static void main(String args[]) {
        Hayvan h = new Hayvan();  //Dikkat
        Kartal k = new Kartal(h);
        k.uc();
        k.kon();
    }
}
```

```
Kartal Ucuyor...
Kartal Konuyor...
```

41

Dahili Üye Sınıflar

- Kus sınıfı statik dahili üye olursa, super() gerekmez.

```
class Hayvan1 {

    static abstract class Kus1 {
        public abstract void uc ();
        public abstract void kon();
    }

    public void avlan() {
        System.out.println("Hayvan avlaniyor...");
    }
}

class Kartal1 extends Hayvan1.Kus1 {
    public void uc() {
        System.out.println("Kartal1 Ucuyor...");
    }
    public void kon() {
        System.out.println("Kartal1 Konuyor...");
    }

    public Kartal1() {  } // dogru

    public static void main(String args[]) {
        Kartal1   k1 = new Kartal1();
        k1.uc();
        k1.kon();
    }
}
```

```
Kartal1 Ucuyor...
Kartal1 Konuyor...
```

42

Dahili Üye Sınıflar

- Dahili üye sınıflar, normal sınıflar gibi başka sınıflardan türetilebilirler.

```
class Motor {
    public void calis() {
        System.out.println("Motor Calisiyor");
    }
    public void dur() {
        System.out.println("Motor Durdu");
    }
}

public class YarisArabasi {
    public void hizYap() {
        System.out.println("YarisArabasi hiz yapiyor");
    }
    public class SuperMotor extends Motor {
        public void calis() { // iptal etti (override)
            System.out.println("SuperMotor Calisiyor");
        }
        public void dur() { // iptal etti (override)
            System.out.println("SuperMotor Durdu");
        }
    }
}
```

43

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

Yerel Sınıflar

- Yerel sınıflar;
  - Yapılandırıcıların
  - Sınıf yordamlarının
  - Nesne yordamlarının
  - Statik alanlara toplu değer vermek için kullanılan statik bloğun
  - Statik olmayan alanlara toplu değer vermek için kullanılan bloğun

  içerisinde tanımlanabilir.
- Yerel sınıflar, yalnızca içinde tanımlandıkları yordamın veya bloğun içerisinde geçerlidir.
- Yerel sınıflar başka sınıflardan türetilebilir veya arayüzlere erişebilir.
- Yerel sınıfların yapılandırıcıları olabilir.

45

Yerel Sınıflar

- Yerel sınıflar;
  - içinde bulundukları yordamın sadece final olan değişkenlerine ulaşabilir.
  - statik veya statik olmayan yordamların içinde tanımlanabilir.
  - private, protected ve public erişime sahip olamazlar, sadece friendly erişime sahip olabilirler.
  - statik olarak tanımlanamazlar.

```
public class Hesaplama7 {
    public static int topla(int a, final int b) {
        int a_yedek = a ;
        class Toplama7 {
            private int x ; // dogru
            public int y ;  // dogru
            // protected int z = a_yedek ;  // ! Hata !
            int p ;  // dogru
            public int degerDondur() {
                // int degera = a ;  // Hata
                int degerb = b ;
                return b ;
            }
        }  // class Toplama7
        Toplama7  t7 = new Toplama7();
        return t7.degerDondur();
    }
```

46

Yerel Sınıflar

```
    public  void ekranaBas()  {
        /*  yerel siniflar sadece friendly erisim
                              belirleyicisine sahip olabilirler
            public  class Toplama8 {
              public void  test() {}
            }  // class Toplama8
         */
    }  // ekranaBas
    public void hesaplamaYap()  {
        /*  yerel sinif sadece friendly erisim
                        belirleyicisine sahip olabilirler
            static class Toplama9 {
              public void abcd() {
              }
            }  // class Toplama9
         */
    }  // hesaplamaYap

    public static void main(String args[]) {
        int sonuc = Hesaplama7.topla(5,9);
        System.out.println("Sonuc    " + sonuc );
    }
} // class Hesaplama7
```

```
Sonuc    9
```

47

Konular

- Arayüz ve Soyut Sınıflar
- Arayüz ile Çoklu Kalıtım
- Arayüzlerin Kalıtım ile Genişletilmesi
- Çakışmalar
- Arayüzlerde Alanlar
- Arayüzler ve Yukarı Çevrim
- Dahili Arayüzler
- Dahili Sınıflar
- Dahili Üye Sınıflar
- Yerel Sınıflar
- İsimsiz Sınıflar

İsimsiz Sınıflar

- İsimsiz sınıflar, bir çok işlem için çok avantajlıdır.
- Özellikle olay dinleyicilerin (event listeners) yer aldığı uygulamalarda sıkça kullanılırlar.
- İsimsiz sınıflar extends ve implements anahtar kelimeleri kullanılarak diğer sınıflardan türetilemez ve arayüzlere erişemezler.
- İsimsiz sınıfların herhangi bir ismi olmadığı için yapılandırıcısı olamaz.

49

İsimsiz Sınıflar

- Arayüz kullanılarak isimsiz sınıf oluşturulabilir.

```
interface Toplayici {
    public int hesaplamaYap() ;
}

public class Hesaplama8 {

    public Toplayici topla(final int a, final int b) {
        return new Toplayici() {
            public int hesaplamaYap()  {

                // final olan yerel degiskenlere ulasabilir.
                return a + b ;
            }
        }; // noktali virgul sart

    } // topla, yordam sonu


    public static void main(String args[]) {

        Hesaplama8 h8 = new Hesaplama8();
        Toplayici t  = h8.topla(5,9);
        int sonuc = t.hesaplamaYap();
        System.out.println("Sonuc  =  5 + 9  =  " + sonuc );
    }
} // class Hesaplama8
```

```
Sonuc  =  5 + 9  =  14
```

50

İsimsiz Sınıflar

- Soyut sınıf kullanılarak isimsiz sınıf oluşturulabilir.

```
abstract  class  BuyukToplayici {
    private  int deger ;
    public BuyukToplayici(int x) {
        deger = x ;
    }
    public int degerDondur() {
        return deger ;
    }
    public abstract int hesaplamaYap() ; // iptal edilmesi gerek
}
public class Hesaplayici {
    public BuyukToplayici degerGoster( int gonderilen ) {
        return new BuyukToplayici( gonderilen ) {
            public int hesaplamaYap()  { //iptal etti (override)
                return  super.degerDondur() + 5  ;
            }
        }; // noktali virgul sart
    } // degerGoster , yordam sonu
    public static void main(String args[]) {

        Hesaplayici h8 = new Hesaplayici();
        BuyukToplayici t  = h8.degerGoster(9);
        int sonuc = t.hesaplamaYap();
        System.out.println("Sonuc  =  " + sonuc );
    }
} // class Hesaplama8
```

```
Sonuc  =  14
```

51