Nesne Tabanlı Programlama
Bölüm 5
Kalıtım (inheritance)

Doç. Dr. Murat TAŞYÜREK

13 Ekim 2025

Kayseri Üniversitesi, Bilgisayar Mühendisliği Bölümü

Kalıtım ve Kompozisyon (Composition)

- Yazılım geliştirirken önceden yazılmış sınıflar **kalıtım (inheritance) ve kompozisyon (composition)** olmarak üzere iki yöntem ile kullanılır.
- Önceden yazılmış sınıfların yeni yazılacak sınıflar içerisinde direk kullanılması **kompozisyon (composition)** olarak adlandırılır.
- Kompozisyon, bir sınıfın diğerini içermesine izin veren sınıflar arasındaki bir ilişki türüdür.
- Kompozisyona genellikle **Has-A (B sınıfı A sınıfını içeririyor manasına gelen)** ilişkisi denir. B sınıfın araba A sınıfında motor olduğunu düşünürkesek, bir arabanın motoru vardır.
- **Tıpkı kalıtım gibi, kompozisyon da kodun yeniden kullanılmasına izin verir.**

Kompozisyon (Composition)

- Sınıflar arasında bir ilişki oluşturmak (kompozisyon ilişkisi) için, kullanmak istediğimiz sınıfın içinde yeni bir sınıf nesnesi başlatırız.

```
2 references
class ClassA
{
    // class body
}

0 references
class ClassB
{
    ClassA nesne = new ClassA();

    //nesne.... ( ClassA sınıfdan metotların çağrılması);
}
```

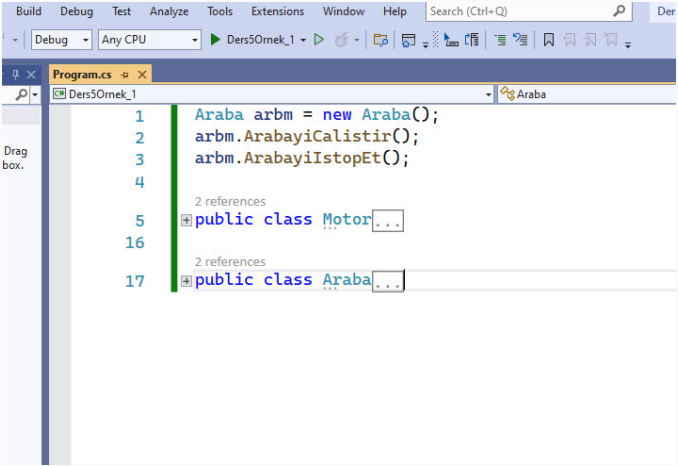
Örnek 1

- Bir arabamızın olduğunu düşünelim.
- Arabamızın motoru vardır ancak başka arabanın da aynı motoru olabilir.
- Nesne tababanlı programlama mantığında düşündüğümüzde az kod yazmak ve yazılan kodu tekrar kullanmak için Araba ve Motor için ayrı ayrı sınıf oluştururuz.
- Araba çalıştığına aslında önce motor çalışmış olur.
- Arabayı durdurduğumuzda önce motor durmuş olur.
- Araba ve motor arasındaki ilişkiyi içeren kodlamayı yapalım.

Araba ve Motor Class

```
public class Motor
{
    1 reference
    public void MotoruCalistir()
    {
        Console.WriteLine("Motor çalıştı");
    }
    1 reference
    public void MotoruKapat()
    {
        Console.WriteLine("Motor kapatıldı");
    }
}
0 references
public class Araba
{
    Motor arabaninMotoru = new Motor();//Compoustion, Araba has Motor
    0 references
    public void ArabayiCalistir()
    {
        arabaninMotoru.MotoruCalistir();
        Console.WriteLine("Araba çalıştı");
    }
    0 references
    public void ArabayiIstopEt()
    {
        arabaninMotoru.MotoruKapat();
        Console.WriteLine("Araba istop edildi.");
    }
}
```

Kaynak Kodu


Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
Debug
Any CPU
Ders5Ornek_1
Program.cs
Ders5Ornek_1
Araba
Drag box.
1 Araba arbm = new Araba();
2 arbm.ArabayiCalistir();
3 arbm.ArabayiIstopEt();
4
2 references
5 public class Motor...
16
2 references
17 public class Araba...

Çıktı

```
1   Araba arbm = new Araba();
2   arbm.ArabayiCalistir();
3   arbm.ArabayiIstopEt();
4
    2 references
5   public class Motor...
16
    2 references
17  public class Araba...
```

```
Microsoft Visual Studio Debug Console
Motor çalıştı
Araba çalıştı
Motor kapatıldı
Araba istop edildi.

D:\DropBox\Dropbox\Dersler\2024-2025_EgitimOgretim\NesneYönelimliProgramla
5Ornek_1\bin\Debug\net6.0\Ders5Ornek_1.exe (process 27752) exited with coo
To automatically close the console when debugging stops, enable Tools->Opt
le when debugging stops.
Press any key to close this window . . .
```

Kalıtım (Inheritance)

- Nesne Yönelimli Programlama dillerinde **kalıtım (inheritance)**, bir sınıfta (class) tanımlanmış değişkenlerin ve/veya metotların (fonksiyon, procedure) yeniden tanımlanmasına gerek olmaksızın yeni bir sınıfa taşınabilmesidir.
- Bunun için yapılan iş, bir sınıftan **bir alt-sınıf (subclass)** türetmektir.
- **Miras yoluyla (kalıtım)**, bir kez oluşturulmuş olan sınıfın tekrar tekrar kullanılabilir.
- Böylece, programlar daha kısa olur, programın yazılma zamanı azalır ve gerektiğinde değiştirilmesi ve onarılması kolay olur.

Kalıtım (Inheritance)

- **Kalıtım yoluyla yeni bir sınıf, önceden yazılmış başka bir sınıftan türetilebilir.**
- **Üretilen yeni sınıf, türetildiği sınıfın kalıtım yoluyla gelebilen özelliklerine sahip olur.**
- **Yeni üretilen bir sınıf türetildiği sınıfın özelliklerini değiştirerek kullanabilir.**
- **Yeni üretilen sınıfın kendisine ait yeni özellikleri de tanımlanabilir.**

Kalıtım (Inheritance)

- C#'ta kalıtım, mevcut bir sınıftan yeni bir sınıf oluşturmamıza olanak tanır.
- **Kalıtılım Nesneye Yönelik Programlamanın (OOP) en temel özelliklerinden birisidir.**
- **Yeni bir sınıfın oluşturulduğu sınıf, temel sınıf (parent or superclass) olarak bilinir. Yeni sınıfa türetilmiş sınıf (derived, child or subclass) denir.**
- **Türetilmiş sınıf, temel sınıfın özelliklerini, öznitelikleri ve yöntemlerini devralır. Bu, C#'daki kodun yeniden kullanılabilirliğine yardımcı olur.**

Kalıtım (Inheritance)

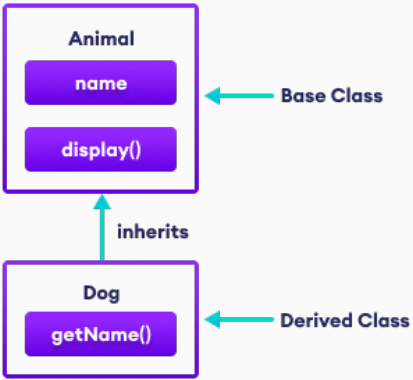
- C#'ta kalıtımı gerçekleştirmek için : sembolünü kullanırız.

```
1 reference
class Animal
{
    // fields and methods
}

// Dog inherits from Animal
0 references
class Dog : Animal
{
    // fields and methods of Animal
    // fields and methods of Dog
}
```

Kalıtım (Inheritance)

- Burada, Animal temel sınıf. Dog sınıfı miras alan yani türetilmiş (derived class) sınıf. Dog sınıfı artık Animal sınıfının alanlarına ve yöntemlerine erişebilir.



Animal ve Dog Class

```
// base class
1 reference
class Animal
{
    public string name;

    0 references
    public void display()
    {
        Console.WriteLine("I am an animal");
    }
}
```

```
// derived class of Animal
0 references
class Dog : Animal
{
    0 references
    public void getName()
    {
        Console.WriteLine("My name is " + name);
    }
}
```

Kaynak Kodu

```
1   // object of derived class
2   Dog dg = new Dog();
3
4   // access field and method of base class
5   dg.name = "Kangal";
6   dg.display();
7
8   // access method from own class
9   dg.getName();
10
11
12  // base class
    1 reference
13  class Animal...
22
23  // derived class of Animal
    2 references
24  class Dog ...
```

Kalıtım yoluyla gelen özellik ve metot

Çıktı

```
Animal
// object of derived class
Dog dg = new Dog();

// access field and method of base class
dg.name = "Kangal";
dg.display();

// access method from own class
dg.getName();


// base class
1 reference
class Animal...

// derived class of Animal
2 references
class Dog ...
```

```
Microsoft Visual Studio Debug Console
I am an animal
My name is Kangal

D:\DropBox\Dropbox\Dersler\2024-2025_EgitimOgretim\NesneYönelimliProgramlama\Der
5_Ornek2\bin\Debug\net6.0\Ders5_Ornek2.exe (process 24952) exited with code 0.
Press any key to close this window . . .
```

Kalıtımın önemi için örnek

- Kalıtımın önemini daha iyi anlamak için bir örnek yapalım.
- Kare, diktörgen ve benzeri normal çok genler ile çalışıyoruz.
- Girdiye göre bu çok genlerin alanların ve çevrelerinin değerlerini bulmak istiyoruz.
- Alan hesaplama formülü tüm normal çokgenler için ortaktır.
- Bu nedenle Normal Çokgen sınıfı ve çevreyi hesaplamak için **AlanHesapla()** yöntemini oluşturabiliriz.

Classes

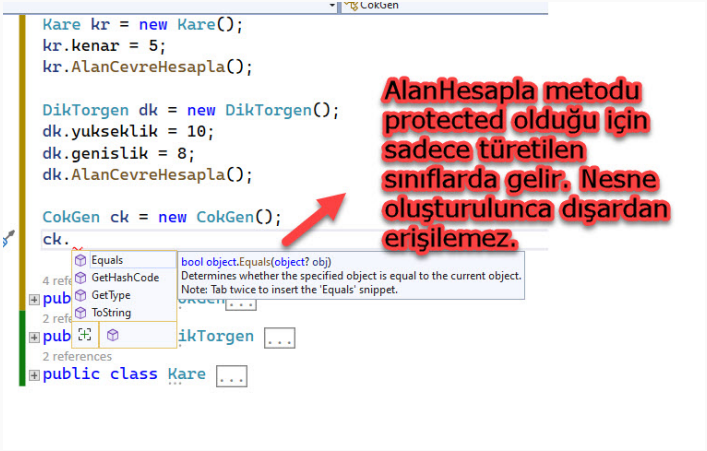
```
public class CokGen
{
    2 references
    protected void AlanHesapla(int width, int lenght)
    {
        Console.WriteLine("Çok genin alanı:" + (width * lenght));
    }
}
2 references
public class DikTorgen : CokGen
{
    public int genislik, yukseklik;
    1 reference
    public void AlanCevreHesapla()
    {
        int cevre = 2 * (genislik + yukseklik);
        Console.WriteLine("Çok genin çevresi :" + cevre);
        AlanHesapla(genislik, yukseklik);
    }
}
2 references
public class Kare : CokGen
{
    public int kenar;
    1 reference
    public void AlanCevreHesapla()
    {
        int cevre = 4 * kenar;
        Console.WriteLine("Çok genin çevresi :" + cevre);
        AlanHesapla(kenar, kenar);
    }
}
```

protected olduğundan sadece alt sınıflardan erişilebilir.

kalıtım

kalıtım

Kaynak Kodu


Kare kr = new Kare();
kr.kenar = 5;
kr.AlanCevreHesapla();
DikTorgen dk = new DikTorgen();
dk.yukseklik = 10;
dk.genislik = 8;
dk.AlanCevreHesapla();
CokGen ck = new CokGen();
ck.
Equals
GetHashCode
GetType
ToString
bool object.Equals(object? obj)
Determines whether the specified object is equal to the current object.
Note: Tab twice to insert the 'Equals' snippet.
2 references
public class Kare
AlanHesapla metodu protected olduğu için sadece türetilen sınıflarda gelir. Nesne oluşturulunca dışardan erişilemez.

Çıktı

```
Kare kr = new Kare();
kr.kenar = 5;
kr.AlanCevreHesapla();

DikTorgen dk = new DikTorgen();
dk.yukseklik = 10;
dk.genislik = 8;
dk.AlanCevreHesapla();

CokGen ck = new CokGen();
//ck.

4 references
public class CokGen...
2 references
public class DikTorgen ...
2 references
public class Kare ...
```

```
Microsoft Visual Studio Debug Console
Çok genin çevresi :20
Çok genin alanı:25
Çok genin çevresi :36
Çok genin alanı:80

D:\DropBox\Dropbox\Dersler\2024-2025_EgitimOgretim\NesneYönelimliP
5_Ornek3\bin\Debug\net6.0\Ders5_Ornek3.exe (process 3056) exited w
Press any key to close this window . . .
```

Kalıtımın önemi

- Alan hesabının yapıldığı yeri değiştirelim.
- Sonucun bütün sınıfları etkilediğini görelim.
- Program çalıştığından sonuçların değiştiğini göreceksiniz.
- Nesne yönelimli programlama bunun gibi durumlar için önemlidir.
- **Sadece bir yerde değişiklik yaparsınız o sınıftan üretilen sınıflarda veya nesnelerin tamamında güncelleme yapmış olursunuz.**

Kalıtım ve ilk değer alma sırası

- Kalıtım, yeni oluşturulan sınıfın türetildiği sınıfa ait özellikleri alması ve ayrıca kendisine ait özellikleri tanımlayabilmesidir.
- **Kalıtım yoluyla sınıflar oluşturulup daha sonra da nesne oluşturulurken öncelikle türetildiği sınıfın nesnesi oluşturulmaya çalışır.**
- **Bu işlem en son ilk kalıtım alınan sınıfın kurucu fonskiyonuna kadar gidilir.**
- **İlk önce o tetiklenir daha sonra ondan türeyen sınıfların metotları tetiklenir.**

Kalıtım ve ilk değer alma sırası

- Bengal kaplanı için bir sınıf olduğunu farz edelim.
- Bengal kaplanı aslında bir kaplandır ve kaplan da aslında bir hayvandır.
- Önce hayvan sınıfını tanımlayalım, daha sonra hayvan sınıfından kalıtım yoluyla kaplan sınıfını tanımlayalım ve daha sonra kaplan sınıfından kalıtım yoluyla türeyen bir bengal kaplanı sınıfını tanımlayalım.
- **Her üç sınıfında constructor metodu olsun.**

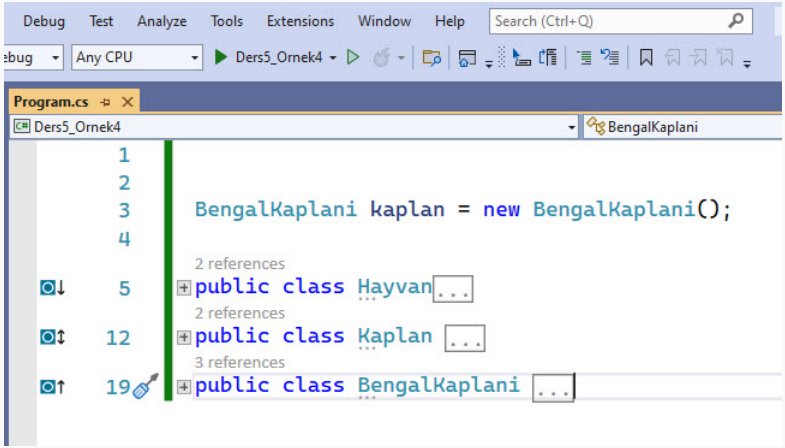
Classes

```
public class Hayvan
{
    0 references
    public Hayvan()
    {
        Console.WriteLine("Hayvan sınıfı için tetiklenen yapılandırıcı.");
    }
}
2 references
public class Kaplan : Hayvan
{
    0 references
    public Kaplan()
    {
        Console.WriteLine("Kaplan sınıfı için tetiklenen yapılandırıcı.");
    }
}
3 references
public class BengalKaplani : Kaplan
{
    1 reference
    public BengalKaplani()
    {
        Console.WriteLine("Bengal kaplanı sınıfı için tetiklenen yapılandırıcı.");
    }
}
```

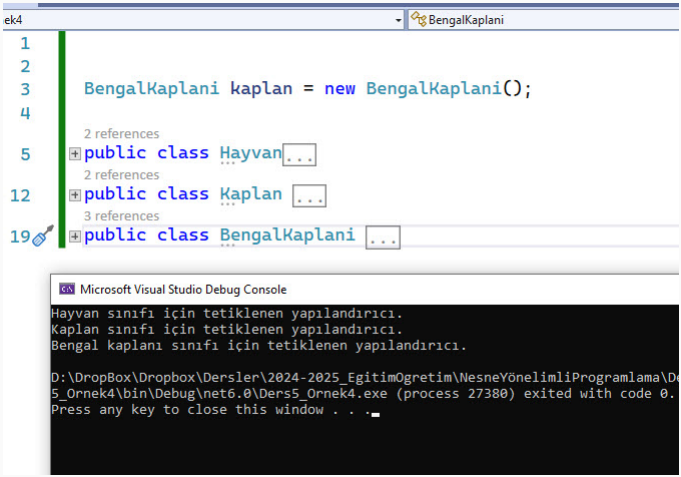
Kaynak Kodu

```
public class Hayvan
{
    0 references
    public Hayvan()
    {
        Console.WriteLine("Hayvan sınıfı için tetiklenen yapılandırıcı.");
    }
}
2 references
public class Kaplan : Hayvan
{
    0 references
    public Kaplan()
    {
        Console.WriteLine("Kaplan sınıfı için tetiklenen yapılandırıcı.");
    }
}
3 references
public class BengalKaplani : Kaplan
{
    1 reference
    public BengalKaplani()
    {
        Console.WriteLine("Bengal kaplanı sınıfı için tetiklenen yapılandırıcı.");
    }
}
```

Kurucu fonksiyonların çalışma sırası



Çıktı


ek4
BengalKaplani
1
2
3
BengalKaplani kaplan = new BengalKaplani();
4
2 references
5
public class Hayvan...
2 references
12
public class Kaplan ...
3 references
19
public class BengalKaplani ...
Microsoft Visual Studio Debug Console
Hayvan sınıfı için tetiklenen yapılandırıcı.
Kaplan sınıfı için tetiklenen yapılandırıcı.
Bengal kaplanı sınıfı için tetiklenen yapılandırıcı.
D:\DropBox\Dropbox\Dersler\2024-2025_EgitimOgretim\NesneYönelimliProgramlama\D
5_Ornek4\bin\Debug\net6.0\Ders5_Ornek4.exe (process 27380) exited with code 0.
Press any key to close this window . . .

Ödev Konusu: Film Akış Platformu i

- Bir **film kütüphanesi** oluşturulacaktır. Her film için *ad, yönetmen, süre, yaş sınırı, puan, tür* bilgileri tutulacaktır.
- Kullanıcılar **izleme listeleri** oluşturabilecek, filmleri *ekleyip/çıkarabilecek*, listeyi *sıralayabilecek* ve *karışık oynatma (shuffle)* yapabilecektir.
- **Kalıtım zorunlu**: Üst sınıf *Film*, alt sınıflar *Aksiyon, Komedi, Drama, Belgesel*. Her tür kuralı alt sınıfta doğrulanacaktır.
- **Tür kuralları**: Aksiyon $\geq$ 90 dk ve dublör bilgisi, Komedi $\leq$ 120 dk ve mizah yoğunluğu (1–5), Drama $\geq$ 90 dk ve yaş sınırı $\geq$ 13, Belgesel 45–180 dk ve anlatıcı boş olamaz.
- **Fiyatlandırma**: Aksiyon 2.0$, Komedi 1.2$, Drama 1.8$, Belgesel 1.5$. İzlenen filmler üzerinden toplam maliyet hesaplanacaktır.

Ödev Konusu: Film Akış Platformu ii

- Program akışı: Kullanıcıdan menüyle işlem alınacak → işlem uygulanacak → sonuç ve toplam maliyet ekrana yazdırılacaktır.
- Hatalı tür/süre/puan girişleri reddedilecek ve anlamlı uyarı verilecektir.
- **Teslim**: **1 sayfa tasarım notu** (sınıflar ve metotlar), **kaynak kod** ve **örnek çalışma çıktısı**.
- 20 Ekim 07.59'a kadar gönderilenler 8+2 puan, vize tarihine kadar gönderenler 8 puan.
- `https://classroom.google.com/c/ODE0NzE5OTg3MzE0?cjc=ftva6cnv` adresindeki **ilgili** başlığa yükleyiniz.