

Konu Başlıkları



- ▶ OOP Nedir ?
- ▶ Neden OOP ?
- ▶ Class(Sınıf) Nedir ?
 - ▶ Access Modifiers (Erişim Belirteçleri)
 - ▶ Field
 - ▶ Property
 - ▶ Constructors (Yapıcı Metotlar)
- ▶ OOP'nin Temel Kavramları

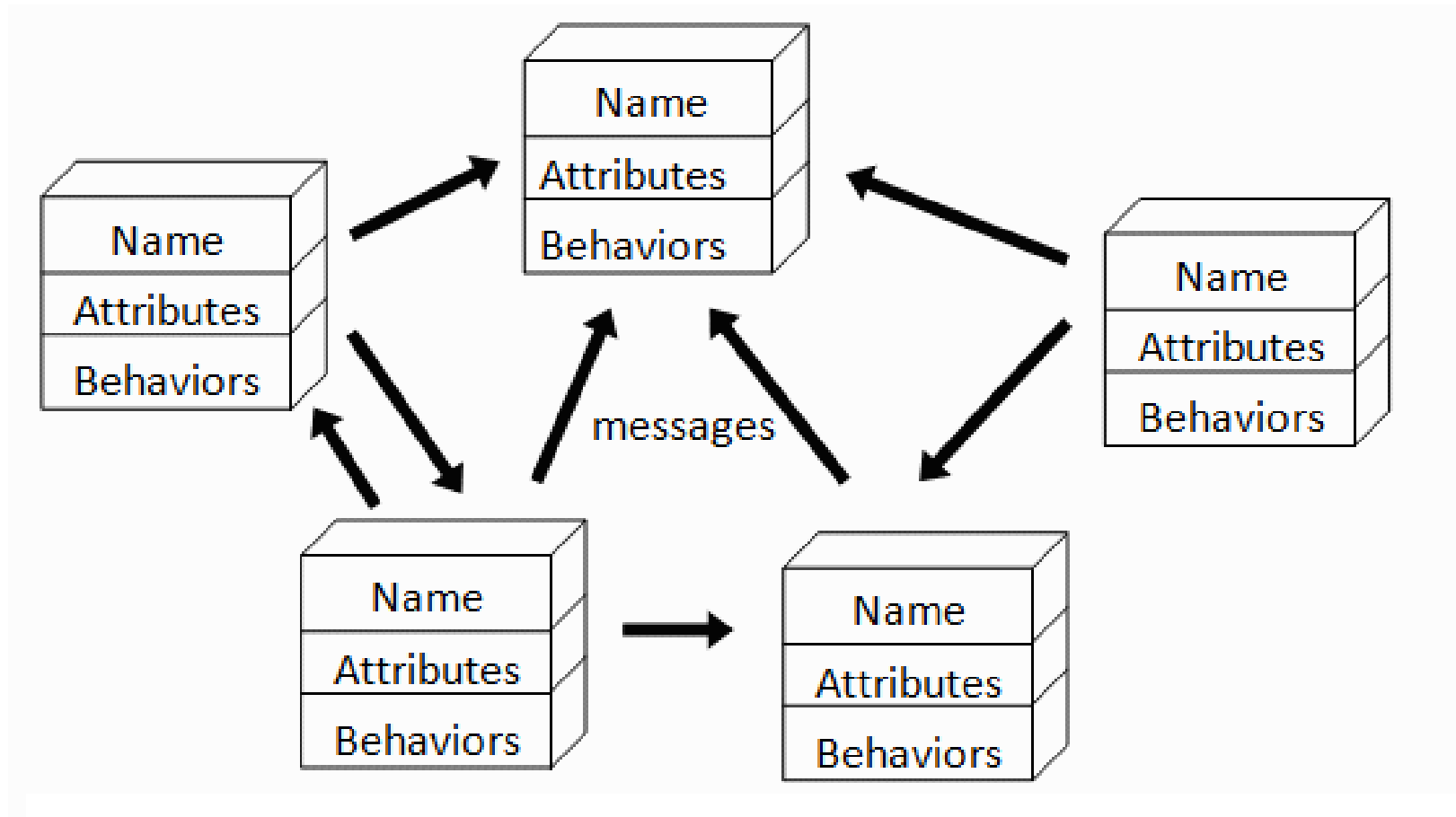


OOP Nedir?

Nesne yönelimli programlama olarak da ifade edebileceğimiz OOP, yapılması istenen işi küçük parçalara bölüp her bir parça arasındaki ilişkiyi kurarak büyük çapta uygulamalar yazmaya yarayan bir metodolojidir.



OOP Nedir?



Neden OOP?

OOP kullanılmadan da uygulamalar yazılabilir. Fakat aşağıda OOP' un sağladığı bazı kolaylıklar sıralanmıştır. Seçim sizin 😊.

- ▶ Çok daha az kod yazmak
- ▶ Hatalarda merkezi kontrol
- ▶ Daha anlamlı kod yazmak
- ▶ Belirli bir hiyerarşiye göre kod yazmak
- ▶ Gerçek hayattaki her şeyi programınızda simule edebilmek



Class(Sınıf)

OOP' un temel yapı taşı sınıflardır. Sınıflar, verileri modellemeye yarar ve nesneler oluşturabilmemizi sağlar.

Array, Random, Convert, Int32, String, MessageBox gibi sıkça kullandığımız komutlar da aslında birer sınıftır. Sınıflar sayesinde kendi tiplerimizi tanımlayacağız.



Class Tanımlaması

```
class <Class İsmi>
```

```
{
```

```
// Bir class özellikler, metotlar ve eventlar içerebilir.
```

```
}
```

Örnek:Aşağıda bir şirkette çalışan personelleri programatik olarak anlamlandırmak ve tanımlamak için Personel sınıfı oluşturulacaktır.

```
class Personel
```

```
{
```

```
/* Personellere ait özellikler ve işlemler yer alacaktır.
```

```
ad, soy ad, yas, sigorta numarası gibi özellikleri ve maaş hesaplama gibi işlemleri bulunacaktır. */
```

```
}
```



Nesne

Class'lar birer şablondur, nesne ise bu şablondan oluşturulmuş bir örnektir(instance).

Class (Şablon)

Personel Formu

TC Kimlik :

Ad :

Soy ad :

Sigorta No :

...

...

...

...

Nesne (Instance)

Personel Formu

TC Kimlik : 17231550122

Ad : ...Abdülkadir.....

Soy ad : ...BARIK.....

Sigorta No : ...123123131

...

...

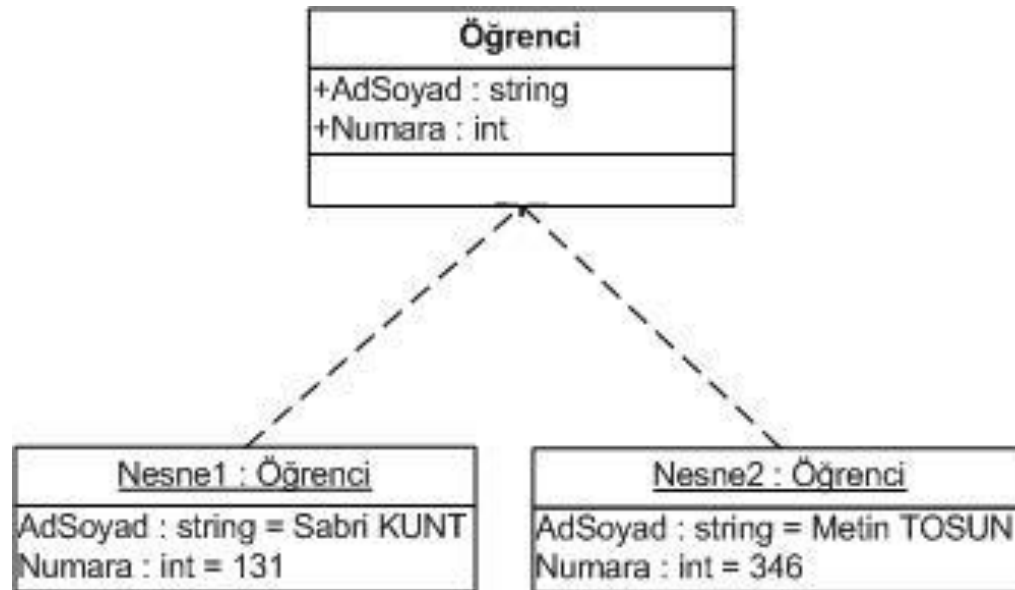
...

...



Nesne

Nesneler referans tipi değişkenler grubuna girerler yani hafızada heap bölümünde oluşturulurlar.



Class ve Nesne

Class Tanımlaması

```
class Personel
{
    public string _ad, _soyad;
    public int _yas;
    public decimal _tckimlikno;
    private decimal _primorani;
    public decimal MaasHesapla()
    {
        //Hesaplama Komutları
    }
}
```

Nesne Oluşturulması

```
Personel pers1 = new Personel();
pers1._ad = "Abdülkadir";
pers1._soyad = "BARLIK";
pers1._tckimlikno = 17232423454;
pers1._yas = 25;
Pers1._primorani = 1.12; // hatalı
```

Not: Private olan değişkenlere başka yerden erişilemez sadece o class içerisinde erişilir.



Access Modifiers (Erişim Belirteçleri)

Erişim belirteçleri class içerisindeki özellik ve metotlar için belirlenir. Bu erişim belirteçleri sayesinde bir özellik veya metodun diğer classlardan erişilip erişilemeyeceğini belirtir. Aşağıda erişim belirteçleri sıralanmıştır;

- ▶ public
- ▶ private
- ▶ internal
- ▶ protected
- ▶ protectedinternal

Not: Eğer bir özellik veya metodun erişim belirteci belirtilmemişse default olarak "private" değerini alır.



Access Modifiers (Eriřim Belirteçleri)

Access Modifier	Açıklama
Private	Sadece bu üyenin bulunduğu sınıf içerisinde erişilebilir.
Internal	Sadece bu üyenin bulunduğu proje içerisinde erişilebilir.
Protected	Sadece bu üyenin bulunduğu sınıf ve bu sınıftan türemiş alt sınıflardan erişilebilir.
Protected Internal	Bu üyeye aynı assembly içerisinde erişilmeye çalışıldığında “internal” gibi, başka assembly içerisinde erişilmeye çalışıldığında da “protected” gibi davranır.
Public	Public üyelerin erişiminde herhangi bir kısıtlama yapılmaz. İstenilen her yerden üyeye erişim açıktır.



Field(Alan)

- ❑ Field'lar sınıflarımız içerisinde veri barınmamızı sağlayan yapılardır. Yani sınıfa ait değişkenlerdir.
- ❑ Field'lar değer veya referans tipinden olabilir.
- ❑ Eğer field'lara değer atanmazsa default değerleri verilir.

Veri Tipi	Default Değeri
int	0
long	0
float	0.0
double	0.0
bool	false
char	\0' (null karakteri)
string	"" (boş metin)
Object	null



Field(Alan)

```
6 namespace OOP
7 {
8     public class Telefon
9     {
10         public string _model;
11         public string _marka;
12         public int _fiyat = 0;
13     }
14 }
15
```

```
private void Form1_Load(object sender, EventArgs e)
{
    Telefon tel = new Telefon();
    tel.
}
```

- ◆ _fiyat
- ◆ _marka
- ◆ _model
- ◆ Equals
- ◆ GetHashCode
- ◆ GetType
- ◆ ToString

int Telefon._fiyat

Property(Özellik)

- ❑ Field'lara yani verilerin tutulduğu alana doğrudan erişim iznini kısıtlamak ve geçerli veriler sağlamak adına "Property" kullanılır.
- ❑ Property sayesinde Field'lara koşullu erişim sağlanır.
- ❑ Property'ler asla veri tutmaz. Sadece field'ı "kapsüller".



Property(Özellik)

```
//Field
private string _model;

//Property
public string Model
{
    get { return _model; }
    set { _model = value; }
}

//Field
private string _marka;

//Property
public string Marka
{
    get { return _marka; }
    set { _marka = value; }
}
```

- ❑ Property, “get” ve “set” olmak üzere iki bloktan oluşur.
- ❑ Get bloğu değişkeninin değeri okunmak istendiğinde çalışır.
- ❑ Set bloğu değişkene değer atanmak istendiğinde çalışır.



Property(Özellik)

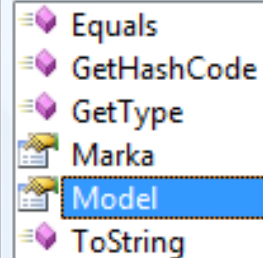
```
//Field
private string _model;

//Property
public string Model
{
    get { return _model; }
    set { _model = value; }
}

//Field
private string _marka;

//Property
public string Marka
{
    get { return _marka; }
    set { _marka = value; }
}
```

```
Telefon tel = new Telefon();
tel.
```



A dropdown menu showing the following options: Equals, GetHashCode, GetType, Marka, Model, and ToString. The 'Model' option is highlighted with a blue background.

string Telefon.Model

Property(Özellik)

- ❑ Eğer değer atanırken veya okunurken bir kısıtlama getirmek istiyorsak koşulları get ve set blokları içersinde yazmalıyız.
- ❑ “Value” değişkene atanmak istenen değeri belirtir.

```
public class Personel
{
    //Field
    private int _yil;

    //Property
    public int Yil
    {
        get { return _yil; }
        set
        {
            if (value >= 1900 && value <= 2100)
            {
                _yil = value;
            }
            else
            {
                throw new Exception("Tarih aralığını aştınız");
            }
        }
    }
}
```

```
private void Form1_Load(object sender, EventArgs e)
{
    Personel pers = new Personel();
    pers.Yil = 1200; //hata oluşacaktır.
}
```

Constructor (Yapıcı Metot)

Nesneler için classların örneğidir demiştik. Constructor her nesne tanımlandığında devreye giren o nesneye ait özelliklerinin ilk değerlerini atamak için kullanılan metottur.

Constructor Tanımlama Kuralları :

- Metodun ismi class ismi ile aynı olmak zorundadır.
- Geri dönüş tipi olmaz.
- Eğer tanımlanmazsa bile mutlaka default constructor vardır.
- Overload edilebilir.



Constructor (Yapıcı Metot)

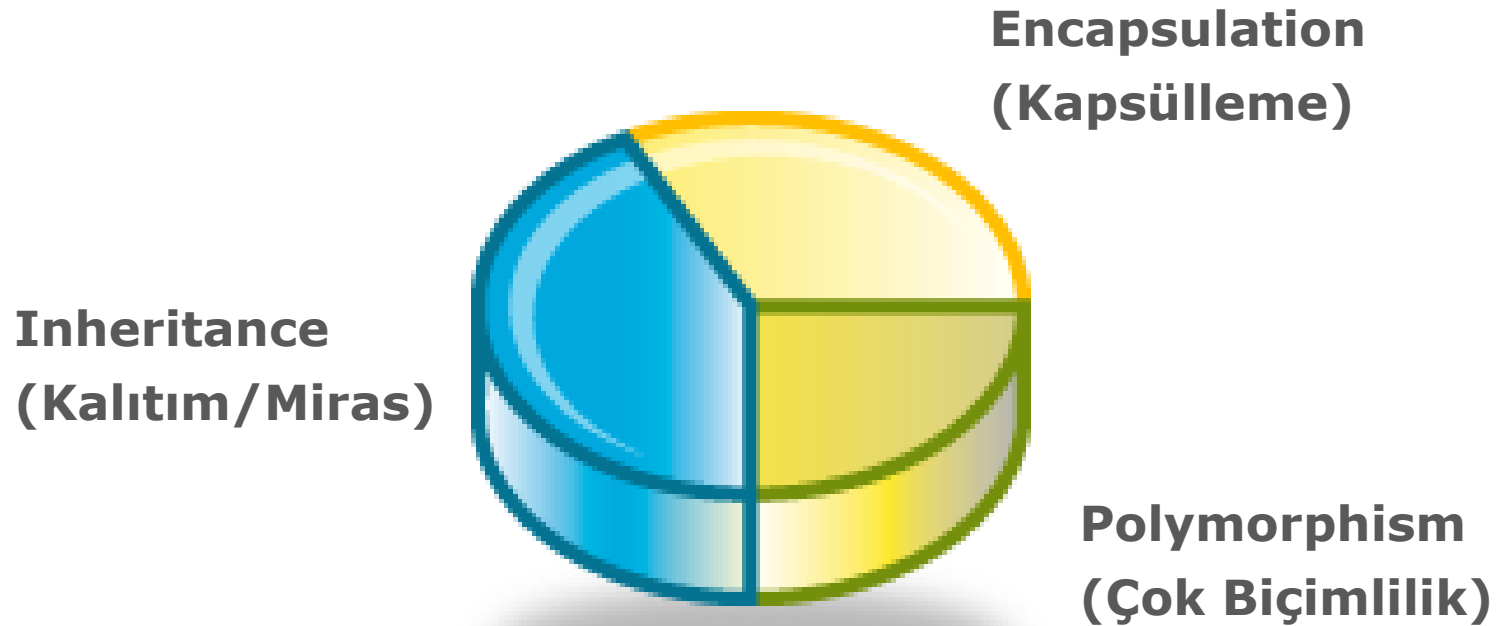
```
class Personel
{
    public string _ad,_soyad;
    public int _yas;

    public Personel() //Constructor
    {
        _ad = "";
        _soyad = "";
        _yas = 0;
    }
}
```

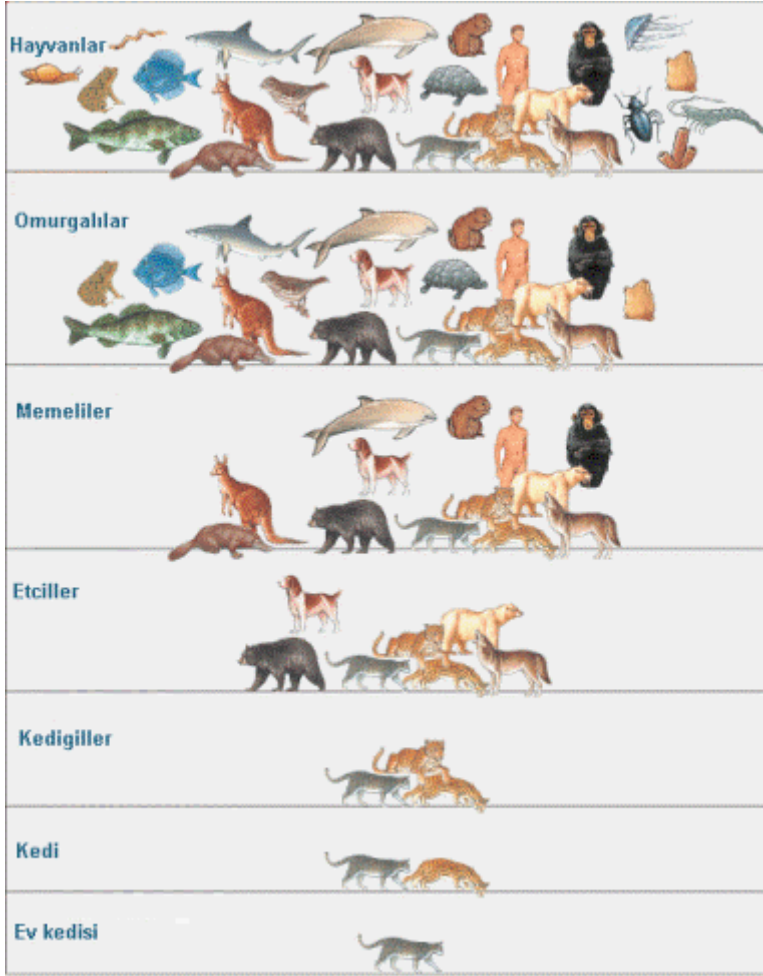
```
Personel pers1 = new Personel();
```



OOP'nin Temel Kavramları



Inheritance(Kalıtım)



- ❖ Hayvanlardan ev kedisine doğru gidildikçe sahip olunan özellikler spesifikleşir.
- ❖ Hayvanlarda tüm hayvanlara ait genel özellikler bulunmalıdır.

Inheritance(Kalıtım)

- ❖ OOP'un en temel kavramlarından biri Inheritance'dır.
- ❖ Bir class'ın üyelerini(field, property, metot gibi) başka bir class'a ancak kalıtım yoluyla aktarılabiliriz.
- ❖ Reusability(Tekrar Kullanılabilirlik) sağlar.
- ❖ Inheritance yoluyla classlar birbirlerinden türetilirler.



Inheritance(Kalıtım)

Base Class

İnsan
Ad;
Soyad;
Yaş;

- ❖ Öğrenci ve Öğretmen sınıfları "ad, soyad ve yaş" özelliklerini İnsan sınıfından miras alacaktır.
- ❖ Türetilen sınıfa "Base Class" denir.
- ❖ Türeyen sınıfa da "Derived Class" denir.

Öğrenci
Ad;
Soyad;
Yaş;
Öğrenci Numarası;

Derived Class

Öğretmen
Ad;
Soyad;
Yaş;
Maaş;
Ders Saati;

Derived Class



Inheritance(Kalıtım)

```
class İnsan
{
    private string ad;

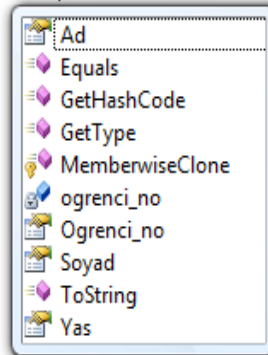
    public string Ad
    {
        get { return ad; }
        set { ad = value; }
    }
    private string soyad;

    public string Soyad
    {
        get { return soyad; }
        set { soyad = value; }
    }
    private int yas;

    public int Yas
    {
        get { return yas; }
        set { yas = value; }
    }
}
```

```
class Ogrenci:İnsan
{
    int ogrenci_no;

    public int Ogrenci_no
    {
        get { return ogrenci_no; }
        set { ogrenci_no = value; }
    }
    public Ogrenci()
    {
        this.
    }
}
```



```
class Ogretmen:İnsan
{
    private int maas;

    public int Maas
    {
        get { return maas; }
        set { maas = value; }
    }
    private int ders_saati;

    public int Ders_saati
    {
        get { return ders_saati; }
        set { ders_saati = value; }
    }
    public Ogretmen()
    {
        this.
    }
}
```

