Nesne Yönelimli Programlama

8.Hafta

Konu: Kalıtım Engelleme, Soyut Sınıflar

Dr. Öğr. Üyesi Güncel SARIMAN
E-posta: guncelsariman@mu.edu.tr

Kalıtımı Engellemek

Yazdığınız bir sınıftan kalıtım yoluyla başka sınıflar üretilmesini engellemek için sınıfınızı sealed anahtar kelimesi ile "mühürlemek" yeterlidir.

sealed class AnaSinif { // Sınıf üyeleri }</pre>

Tüm sınıfın kalıtım yoluyla aktarılmasını engellemek yerine, tek bir method'un override edilmesini engellemek için method'u sealed ile "mühürlemek" yeterlidir.

```
class AnaSinif
{
   public virtual void Goster()
   {
   }
}
```

```
class TuretilmisSinif : AnaSinif
{
public sealed override void Goster()
   {
   }
}
```

Soyut Sınıflar

Nesne tabanlı programlamada sınıf hiyerarşisi oluşturulurken, bazen en tepede bulunan sınıf türünden nesneler programcılar için anlamlı olmayabilir. Hiyerarşinin en tepesinde bulunan sınıfın kendisinden türetilecek olan alt sınıflar için ortak bir **arayüz (interface)** görevi yapması istenebilir.

Bunun için çözüm olarak oluşturulan metotlara ve sınıflara **soyut metot (abstract method)** ya da **soyut sınıf (abstract class)** denir. Soyut sınıflar büyük projelerde kullanılırlar ve kalıtım özelliğini kullanarak kod tekrarını azaltırlar. Soyut sınıflar diğer sınıflara taban olmak için kullanılırlar. Nesne türetemezler. Önlerine "abstract" sözcüğü yazılarak soyutlaştırılırlar. Önlerine **"virtual"** yazılmaz çünkü "abstract" sözcüğü uygulanan tüm sınıf ve metotlar zaten sanaldır.

Soyut Sınıflar

Kullanırken dikkat edilmesi gerekenler:

Soyut sınıflar "abstract" türünden nesneler tanımlamazlar.

Soyut sınıflar içerilerinde soyut olmayan metotlar da barındırabilir ancak soyut metotlar sadece soyut sınıflar içerisinde bildirilebilir.

Soyut metotlar türeyen sınıfta mutlaka bildirilmelidir.

Soyut metotlar override edilmek zorundadırlar, aksi takdirde derleyici hatası alınır.

"Static" metotlar soyut olarak tanımlanamazlar.

Soyut sınıflar "private" olarak tanımlanamazlar ama "public" ve "protected" olarak tanımlanabilirler.

Soyut sınıflar "sealed" anahtar sözcüğü ile ifade edilemezler.

Soyut Sınıflar

Abstract Class

Abstract Class, ortak özellikli Class'lara Base(taban) Class olma görevini üstlenir. Örneğin projenizde birden fazla Class oluşturup kullanmak istiyorsunuz. Ve oluşturacağınız Class'lar pek çok yönden ortak özellikleri içerisinde barındırıyor. İşte bu durumda Abstract Class kullanmanız kodun daha sağlıklı olmasını sağlayacaktır.

Abstract Class'lar, diğer sınıflara base Class olmak için yazılır. Bu nedenle Abstract Class'dan **nesne türetilemez**.

Soyut Fonksiyonlar

Soyut fonksiyonlar, sadece Abstract Class'ların içerisinde tanımlanır ve Abstract Class'ı kalıtan sınıf tarafından override edilmek zorundadır. Oluşturulan Abstract Class içerisine, Abstract Method yazılırken gövdesi yazılmaz ve daha sonra Abstract Class'ımızı kalıtacağımız sınıfta Abstract Method override edilir.

- ✓ Abstarct metotlar sadece tanımlanır. Herhangi bir işlemi yerine getirmezler. Yapacakları işlemler override edildikleri sınıfta kodlanmalıdır.

- ✓ Abstract Method'lar, Private olarak tanımlanamaz.

Kısaca

Abstract class'lar, oluşturulması istenen class'ların sadece base class olarak davranması ve üzerinden bir instance oluşturulmamasının istendiği durumlarda kullanılır.

Örnek

Ucak adında bir tane Abstract(taban) sınıf. Ve daha sonra bu Abstract Class'dan BuyukUcak ve KucukUcak sınıfları kalıtılacak. Ve Ucak Abstract Class'ın içerisine basit bir "UcakFiyatı" isimli method yazıp, bu method'u kalıttığımız Class'larda override ederek, UcakFiyatı isimli methodumuza gövdeyi implement edeceğiz.

```
abstract class Ucak
{
public int YolcuKapasitesi { get; set; }
public string UcakSirketi { get; set; }
//Ucak'ların ortak Özellikleri'leri

public abstract void UcakFiyati();
//override edilecek abstract methodumuz
}
```

Örnek

```
//Büyük Uçak Sınıfı
class BuyukUcak : Ucak //Ucak Abstract Class'ından kalıtılan Class
{
public override void UcakFiyati() //Ucak Abstract Class'ından, Class kalıtırken override ettiğimiz, gövdesini doldurduğumuz method.
{
Console.WriteLine("Büyük Uçağın Fiyatı 5m");
Console.ReadLine();
}
}
```

Örnek

```
//Küçük Uçak Sınıfı
class KucukUcak : Ucak
{
public override void UcakFiyati()
{
Console.WriteLine("Küçük Uçağın Fiyatı 3m");
Console.ReadLine();
}
}
```

Ucak Abstract Class'ından kalıttığımız BuyukUcak ve KucukUcak sınıflarında, UcakFiyati Abstract Method'unu override ederek gövde doldurulуyor. İçerisine ise Büyük Uçağın ve Küçük Uçağın temsili fiyatları yazdırılıyor.

Örnek

KucukUcak k = new KucukUcak();
k.UcakFiyati();

Eğer Ucak Abstract Class'ından bir nesne türetmeye çalışırsak hata verecektir.

Özetle Abstract Class, sonradan oluşturacağımız ortak özellik barındıran Class'lara base(temel) Class olur. Kodlarımızın Class'lar bazında düzenli olmasını sağlar. Abstract Method ise, Abstract Class'ların içerisine yazılabilen ve sonradan override edilmek üzere yazılır.

Örnek-2

```
abstract class mobilya
  {
    public string renk;
    abstract public void ozellikyaz();
  }

  class kanepe : mobilya
  {
    public string kumas;
    public override void ozellikyaz()
    {
      Console.WriteLine("Kanepenin Özellikleri");
      Console.WriteLine("Renk: {o}",renk);
      Console.WriteLine("Kumaş: {o}", kumas);
    }
  }
```

```
class masa : mobilya
  {
    public string malzeme;
    public override void ozellikyaz()
    {
      Console.WriteLine("Masanın
Özellikleri");
      Console.WriteLine("Renk:      {o}",
renk);
      Console.WriteLine("Malzeme:  {o}",
malzeme);
    }
  }
```

Örnek-2

```
kanepe knp1 = new kanepe();
masa calisma_masasi = new masa();
knp1.renk = "Siyah";
knp1.kumas = "Deri";
calisma_masasi.renk = "Sarı";
calisma_masasi.malzeme = "Ahşap";
knp1.ozellikyaz();
calisma_masasi.ozellikyaz();
```

Örnek-2

```
kanepe knp1 = new kanepe();
masa calisma_masasi = new masa();
knp1.renk = "Siyah";
knp1.kumas = "Deri";
calisma_masasi.renk = "Sarı";
calisma_masasi.malzeme = "Ahşap";
knp1.ozellikyaz();
calisma_masasi.ozellikyaz();
```

public class BananaCake : Cake { }

Örnek Çalışma: Laboratuvar Envanter Takip Sistemi

İş Tanımı

1-Üniversite bünyesindeki tüm laboratuvarlardaki malzemelerin sisteme kaydedilmesi, güncellenmesi, silinmesi ve okul bazında rapor alınabilmesini sağlayan bir sistem geliştirilecektir.
2-Sistem MSKÜ'deki fakülte, yüksekokul bölümlerinin lablarını sisteme kaydetmektedir.
3-Kaydedilen labların içlerine türlerine göre malzemeler kaydedilmektedir.
4-Lab türleri bölümlere göre değişmektedir. Bilgisayar, Kimya, Biyoloji, Elektrik vb.
5-Her bölümün malzemeleri farkı olsa da girilmesi gereken bilgiler ortaktır.
6-Malzeme Adı, Markası, Ne zaman Alındığı, Kaç adet olduğu, Tipi(Demirbaş veya Sarf Malzemesi vb.)
7-Sistem de okul bölüm ve sorumlu kullanıcı sisteme tanıtılabilecektir.
8-Kullanıcılar sistemi kullanırken malzemeler hakkında tüm bilgilere erişebilmektedir.

public class BananaCake : Cake { }

Örnek Çalışma: Laboratuvar Envanter Takip Sistemi

1-Okul

2-Bölüm

3-Program

4-Laboratuvarlar

5-Lab Türleri

6-Malzemeler

7-Kullanıcılar

public class BananaCake : Cake { }

Örnek Çalışma: Laboratuvar Envanter Takip Sistemi