NESNEYE YÖNELİK PROGRAMLAMA

Soyut sınıflar

Emir Öztürk

SOYUT SINIFLAR

- Sınıf tanımıyla aynı
- abstract anahtar kelimesi
- Örnek oluşturamama

Soyut sınıf tanımı sınıf tanımıyla aynı şekilde gerçekleştirilir.
Farklı olarak soyut sınıfın tanımlanmasında abstract anahtar kelimesi kullanılmaktadır.
Soyut sınıflardan örnek oluşturulamamaktadır.

SOYUT SINIFLAR

- Ortak özelliklerin toplanması
- Nesne oluşturamayacak genellikte olması

Belirli sınıflar ortak özellikleri toplayacak kadar genel olabilir fakat bu sınıflardan nesne oluşturulması anlamsız olabilmektedir. Sınıf hiyerarşisinde sınıfların genelleştirilmesi ile bu sınıflardan nesne oluşturulamama seviyesine ulaşılabileceğinden böyle durumlarda en uygun olan soyut sınıfların kullanımıdır.

SOYUT SINIFLAR – ARAYÜZLER

- Soyut Sınıf
  - Çok sınıf, ortaklık
  - Ortak metot ve alanlar, erişim düzeyleri
  - Non static, non final

Birbirlerine çok yakın sınıflar arası kod paylaşımı,
Soyut sınıfı kullanan diğer sınıflarda ortak olan metot ve alanların çokluğu / public dışındaki erişim belirleyicilerine sahip metotların olması,
Static veya final olmayan alanların olması durumlarında soyut sınıfların kullanılması tercih edilmektedir.

SOYUT SINIFLAR – ARAYÜZLER

- Arayüz
  - Çok sınıf, bağımsızlık
  - İmplementasyon bağımsızlığı
  - Çoklu kalıtım

Birbirleri ile alakalı olmayan fazla sayıda sınıf ortak özellikleri kullanacaksa (örneğin comparable gibi bir arayüz bir çok sınıf tarafından kullanılmaktadır.),
Türler belirli fakat implementasyon bağımsızsa,
Çoklu kalıtım ihtiyacı varsa arayüzlerin kullanılması tercih edilmektedir.

SOYUT SINIFLAR

```
class A{

}
class B extends A{

}
public class Main {

    public static void main(String[] args) {
        A a = new A();
        B b = new B();
    }
}
```

```
abstract class A{

}
class B extends A{

}
public class Main {

    public static void main(String[] args) {
        A a = new A();
        B b = ne
    }
}
```

'A' is abstract; cannot be instantiated

Make 'A' not abstract ⌥⏎ More actions... ⌥⏎

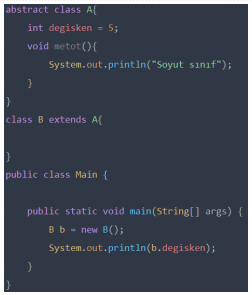
Soyut sınıflar tanımlanırken abstract kelmesi kullanılmaktadır.
Abstract olarak tanımlanmış bir sınıfın herhangi bir yerde örneği alınamaz. Bu sınıf yalnızca diğer sınıflardan türetilmek için kullanılır.

SOYUT SINIFLAR

- Özelliklere sahip olabilme
- Özelliklerinin değerinin atanabilmesi
- Özelliklere erişim

Soyut sınıflardan bir örnek alınamasa da soyut sınıflarda özellikler tanımlanabilmektedir.
Bu özelliklere varsayılan bir değer atanabilmesi de mümkündür.
Bu sınıfı kalıtan başka bir sınıf tanımlandığında bu özelliğe erişim sağlanabilmektedir.

SOYUT SINIFLAR



```
abstract class A{
    int degisken = 5;
    void metot(){
        System.out.println("Soyut sinif");
    }
}
class B extends A{

}
public class Main {

    public static void main(String[] args) {
        B b = new B();
        System.out.println(b.degisken);
    }
}
```

Örnekte görüldüğü gibi bir A soyut sınıfında degisken isimli 5 değerine sahip bir değişken tanımlanmıştır. B sınıfı bu sınıfı kalıttığında bu değere erişim sağlanabilmektedir. B sınıfından bir nesne oluşturulduğunda default olarak tanımlanan bu değişken ekrana 5 değerini gösterecektir.

SOYUT SINIFLAR

- Metotlara sahip olabilme
- Metotların gövdesinin tanımlanabilmesi
- Metotlara erişim

Soyut sınıflarda metotlar da tanımlanabilmektedir. Bu metotların gövdeleri yazılarak diğer sınıflar tarafından da kullanılabilmektedir. Soyut bir sınıftan kalıtılan başka bir sınıf tanımlanmış bir metodu çiğnemiyorsa (override) soyut sınıfta tanımlanmış metot implementasyonu kullanılacaktır.

SOYUT SINIFLAR

```java
abstract class A{
    void metot(){
        System.out.println("Soyut sınıf");
    }
}
class B extends A{

}
public class Main {

    public static void main(String[] args) {
        B b = new B();
        b.metot();
    }
}
```

Soyut sınıf

```java
abstract class A{
    void metot(){
        System.out.println("Soyut sınıf");
    }
}
class B extends A{
    void metot(){
        System.out.println("Aşırı yüklenmiş metot");
    }
}
public class Main {

    public static void main(String[] args) {
        B b = new B();
        b.metot();
    }
}
```

Çiğnenmiş Metot

Soyut sınıflar diğer sınıflarda olduğu gibi metotlar ve özellikler içerebilmektedirler. Aynı zamanda final kelimesi kullanılmadığı sürece bu metotlar override edilebilmektedirler.
Override edilmediği sürece metotlar normal sınıflarda olduğu gibi kalıtım sırasında bulunabilen en yakın sınıftan çağırılırlar.

SOYUT SINIFLAR

- Soyut metotlar
- Soyut metot olan sınıfın soyut sınıf olması
- abstract
- Metot gövdesi

Soyut sınıflarda soyut metotlar da tanımlanabilmektedir. Bu metotlar bir gövde içermemekte ve override edilmesi gerekmektedir.
Soyut metotlar yalnızca soyut sınıf içerisinde tanımlanabilirler.
Soyut bir metot yalnızca sınıfın içerisinde bu metodun bulunacağını ve bu metodun nasıl bir imzaya sahip olacağını belirler. İmplementasyon detayı kalıtımı gerçekleştiren sınıfa ait olacaktır.

SOYUT SINIFLAR

```java
abstract class A{
    abstract void metot(){
    }
}
```

Abstract methods cannot have a body

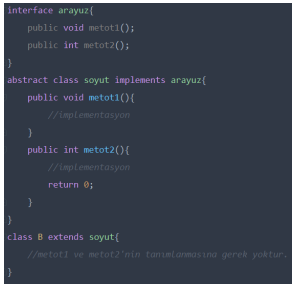
Delete method body ⌥⇧↩ More actions... ⌥↩

```java
abstract class A{
    abstract void metot();
}
class B extends A{
    void metot(){
        System.out.println("Aşırı yüklenmiş metot");
    }
}
public class Main {

    public static void main(String[] args) {
        B b = new B();
        b.metot();
    }
}
```

Soyut tanımlanan bir metot gövde içermemektedir.
Bu metodun kullanılabilmesi için diğer sınıflar içerisinde override edilmesi gerekmektedir.

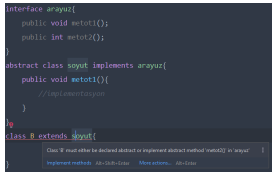
SOYUT SINIFLAR



```
interface arayuz{
    public void metot1();
    public int metot2();
}
abstract class soyut implements arayuz{
    public void metot1(){
        //implementasyon
    }
    public int metot2(){
        //implementasyon
        return 0;
    }
}
class B extends soyut{
    //metot1 ve metot2'nin tanımlanmasına gerek yoktur.
}
```

Soyut sınıflar aynı zamanda arayüzleri de içerebilirler. Soyut sınıfların bu arayüzleri içermesi durumunda arayüzün tanımlanması gereken metotlarının tamamı soyut sınıf üzerinde tanımlanabilir.
Böylece soyut sınıftan türetilmiş başka bir sınıf arayüzün metotlarını tekrar tanımlamadan kullanılabilir. B sınıfından oluşturulan bir nesne metot1 veya metot2 metotlarını çağırdığında bu metotlar temel sınıfta tanımlı oldukları için sorunsuz bir şekilde ulaşabileceklerdir.

SOYUT SINIFLAR



```
interface arayuz{
    public void metot1();
    public int metot2();
}
abstract class soyut implements arayuz{
    public void metot1(){
        //implementasyon
    }
}
class B extends soyut{
```

```
interface arayuz{
    public void metot1();
    public int metot2();
}
abstract class soyut implements arayuz{
    public void metot1(){
        //implementasyon
    }
}
class B extends soyut{
    public int metot2() {
        return 0;
    }
}
```

Soyut sınıflar arayüzlerin tüm metotlarını içermek zorunda değildirler. Herhangi bir soyut sınıfta eksik bir tanım yapıldığında kod hata vermeyecektir. Fakat bu soyut sınıftan kalıtılmış başka bir sınıf olması durumunda bu sınıfta eksik kalan metotların tamamlanması gerekmektedir. Örneğin arayuz interface'ini içeren bir soyut isimli abstract class, arayüzün yalnızca metot1() isimli metodunu tanımlamıştır. Bu durumda derleyici hata vermemektedir. Soyut sınıftan türetilmiş bir B sınıfı ise tanımlandığı anda metot2() metodunun eksik olduğu hatasını verecektir. Bu durumda kalan metotları (metot2()) bu sınıf içerisinde tanımlamak gerekmektedir.
