Kapsülleme

Kalıtım

```
public class Lecturer {
    private Long id;
    private String name;
    private String surname;
    private String department;
    private String academicTitle;

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSurname() {
        return surname;
    }

    public String getDepartment() {
        return department;
    }

    public void setDepartment(String department) {
        this.department = department;
    }

    public String getAcademicTitle() {
        return academicTitle;
    }

    public void setAcademicTitle(String academicTitle) {
        this.academicTitle = academicTitle;
    }
}
```

```java
public class Student {
    private Long id;
    private String name;
    private String surname;
    private String studentId;
    private String department;

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public String getStudentId() {
        return studentId;
    }

    public void setStudentId(String studentId) {
        this.studentId = studentId;
    }

    public String getDepartment() {
        return department;
    public void setDepartment(String department) {
        this.department = department;
    }
}
```

```java
public class Person {

    private Long id;
    private String name;
    private String surname;

    public Person() {
        super();
    }

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

}
```

```java
public class Lecturer extends Person {
    private String department;
    private String academicTitle;

    public String getDepartment() {
        return department;
    }

    public void setDepartment(String department) {
        this.department = department;
    }

    public String getAcademicTitle() {
        return academicTitle;
    }

    public void setAcademicTitle(String academicTitle) {
        this.academicTitle = academicTitle;
    }
}
```

urer2.java hosted with ❤ by GitHub

```java
public class Student extends Person {
    private String studentId;
    private String department;

    public String getStudentId() {
        return studentId;
    }

    public void setStudentId(String studentId) {
        this.studentId = studentId;
    }

    public String getDepartment() {
        return department;
    }

    public void setDepartment(String department) {
        this.department = department;
    }
}
```

Java ve C# çoklu kalıtımı desteklemez. Extends kelimesinden sonra sadece bir sınıf adı yazılabilir. Interface' ler ile bir nevi çoklu kalıtım gerçekleştirilebilir. Interface davranışsal kod barındırmaz, sadece metod imzaları bulunur. Bir interface' den türeyen sınıflar, o interface içindeki tüm metodları gerçekleştirmek zorundadır.

```
interface Rapor{
        public void hazirla(Fatura fatura);
}

class PdfRapor implements Rapor{
        public void hazirla(Fatura fatura) {
                //Faturayı pdf formatında hazırlayan kodlar
        }
}
```

Interface' ler soyut sınıflardan farklılık gösterir.

- Interface'ler de Abstract sınıflar gibi new ile oluşturulamazlar
- İçi dolu metod bulunduramazlar
- public static final değişkenler dışında herhangi bir değişken bulunduramazlar
- Bir sınıf birden fazla interface'den türeyebilir

Abstract sınıflar içerisinde normal yani içi dolu metodların, değişkenlerin ve interface'lerdeki gibi abstract (boş) metodların tanımlanabildiği yapılardır. Bu sınıflar new kelimesi ile oluşturulamazlar.

```
class Araba{
        private int agirlik;
        private String renk;
        private String model;

        public int getAgirlik() {
                return agirlik;
        }
        public void setAgirlik(int agirlik) {
                this.agirlik = agirlik;
        }
        public String getRenk() {
                return renk;
        }
        public void setRenk(String renk) {
                this.renk = renk;
        }
        public String getModel() {
                return model;
        }
        public void setModel(String model) {
                this.model = model;
        }
}
```

```
class Mercedes extends Araba{
	private int cantKalinligi;

	public int getCantKalinligi() {
		return cantKalinligi;
	}
	public void setCantKalinligi(int cantKalinligi) {
		this.cantKalinligi = cantKalinligi;
	}
}

class Ford extends Araba{
	private int devirSayisi;

	public int getDevirSayisi() {
		return devirSayisi;
	}
	public void setDevirSayisi(int devirSayisi) {
		this.devirSayisi = devirSayisi;
	}
}

class KullaniciEkrani{
	public void goster(Araba[] arabalar){
		for (int i = 0; i < arabalar.length; i++) {
			Araba araba = arabalar[i];
			System.out.println("Agirlik : "+araba.getAgirlik());
			System.out.println("Model : "+araba.getModel());
			System.out.println("Renk : "+araba.getRenk());
		}
	}
}

class AnaProgram{
	public static void main(String[] args) {
		Araba ford =new Ford();
		ford.setAgirlik(1000);
		ford.setModel("Fiesta");
		ford.setRenk("Gri");
		Araba mercedes=new Mercedes();
		mercedes.setAgirlik(2000);
		mercedes.setModel("E200");
		mercedes.setRenk("Siyah");

		Araba arabalar[]=new Araba[]{mercedes,ford};
		KullaniciEkrani ekran =new KullaniciEkrani();
		ekran.goster(arabalar);
	}
}
```

Yukarıdaki kodda gördüğünüz gibi Polymorphism sayesinde KullaniciEkrani sınıfı arabaların markalarından habersiz hepsini gösterebiliyor. Araba sınıfından türeyen her sınıf KullaniciEkrani sınıfında gösterilebiliyor. Buraya kadar gayet güzel. Şimdi müşteri bizden bu ekranda araçların saatte kaç litre benzin yaktıklarını da göstermemizi istedi. Fakat burada şöyle bir problem var her marka arabanın kaç litre benzin yaktığı kendi ağırlığına göre farklı hesaplanıyor. Araba sınıfına saatte yaktığı litre diye değişken eklesek olmayacak çünkü Mercedes bunu hesaplamak için farklı katsayı ile çarpıyor Ford farklı sayı ile çarpıyor. İşte bu noktada yardımımıza Abstract kavramı yetişiyor ve kodumuzu şöyle değiştiriyoruz.

```
abstract class Araba{
	private int agirlik;
	private String renk;
	private String model;

	public int getAgirlik() {
		return agirlik;
	}
	public void setAgirlik(int agirlik) {
		this.agirlik = agirlik;
	}
	public String getRenk() {
		return renk;
	}
	public void setRenk(String renk) {
		this.renk = renk;
	}
	public String getModel() {
		return model;
	}
	public void setModel(String model) {
		this.model = model;
	}

	public abstract int saateYaktigiBenzinLitresi();
}

class Mercedes extends Araba{
	private int cantKalinligi;

	public int getCantKalinligi() {
		return cantKalinligi;
	}
	public void setCantKalinligi(int cantKalinligi) {
		this.cantKalinligi = cantKalinligi;
	}

	public int saateYaktigiBenzinLitresi() {
		return getAgirlik()*2;
	}
}
```

```
class Ford extends Araba{
	private int devirSayisi;

	public int getDevirSayisi() {
		return devirSayisi;
	}
	public void setDevirSayisi(int devirSayisi) {
		this.devirSayisi = devirSayisi;
	}

	public int saateYaktigiBenzinLitresi() {
		return getAgirlik()*1;
	}
}

class KullaniciEkrani{
	public void goster(Araba[] arabalar){
		for (int i = 0; i < arabalar.length; i++) {
			Araba araba = arabalar[i];
			System.out.println("Agirlik : "+araba.getAgirlik());
			System.out.println("Model : "+araba.getModel());
			System.out.println("Renk : "+araba.getRenk());
			System.out.println("Yaktigi Lt. Benzin "+araba.saateYaktigiBenzinLitresi());
		}
	}
}

class AnaProgram{
	public static void main(String[] args) {
		Araba ford =new Ford();
		ford.setAgirlik(1000);
		ford.setModel("Fiesta");
		ford.setRenk("Gri");
		Araba mercedes=new Mercedes();
		mercedes.setAgirlik(2000);
		mercedes.setModel("E200");
		mercedes.setRenk("Siyah");

		Araba arabalar[]=new Araba[]{mercedes,ford};
		KullaniciEkrani ekran =new KullaniciEkrani();
		ekran.goster(arabalar);
	}
}
```

Polimorfizm
Bir printer PDF ve Word dosyalarını basabiliyor

```
public class PDF {
	public String getPrintableText() {
		return "This is a PDF document\n";
	}
}

public class Word {
	public String getPrintableText() {
		return "This is a Word document\n";
	}
}

public class Printer {
	public void printPDF(PDF pdf) {
		System.out.print(pdf.getPrintableText());
	}
	public void printWord(Word word) {
		System.out.print(pdf.getPrintableText());
	}
}
```

Yeni yöntem

```
public class Document {
	public String getPrintableText() {
		return "This is a document\n";
	}
}

public class PDF extends Document{
	@Override
	public String getPrintableText() {
		return "This is a PDF document\n";
	}
}



public class Word extends Document {
	@Override
	public String getPrintableText() {
		return "This is a Word document\n";
	}
}
```

```
public class Printer {
	public void print( Document document) {
		System.out.print(document.getPrintableText());
	}
}


public class PrinterTest {
	public static void main(String[] args) {
		Printer printer = new Printer();
		printer.print(new Document());
		printer.print(new PDF());
		printer.print(new Word());
	}
}
```

Document sınıfından türetilmiş (Document' i extend eden) bir nesneyi Document referansı yerine kullanabiliyoruz.

Document document = new PDF(); kullanımı da polimorfizmin bize sağladığı bir kullanımdır.

Super kelimesi ebeveyn sınıfa referans eder

```
public class Word extends Document {
	@Override
	public String getPrintableText() {
		return super. getPrintableText + "This is a Word document\n";
	}
}
```

Kaynaklar:

http://www.cihataltuntas.com/

https://omerozkan.net/kategori/java/