

Flutter ile Uygulama Geliştirme Kursu | Android & IOS

Collections

Kasım ADALAN

Elektronik ve Haberleşme Mühendisi

Android - IOS Developer and Trainer

Eğitim İçeriği

1. List
2. HashSet
3. HashMap

Sabit Boyutlu List İşlemleri

```
var dizi = List<int>.filled(5, 0); //Döngü ile veri okuma

dizi[0] = 10 ;//Veri ekleme
dizi[1] = 20 ;
dizi[2] = 30 ;
dizi[3] = 40 ;
dizi[4] = 50 ;

print(dizi);

//Veri okuma
print(dizi[1]); //20

dizi[3] = 99 ; //Güncelleme
print(dizi);

for(var d in dizi){
    print("Sonuç : $d");
}

for(var i=0;i<dizi.length;i++){
    print("$i. indeksteki veri : ${dizi[i]}");
}
```

Collection Types

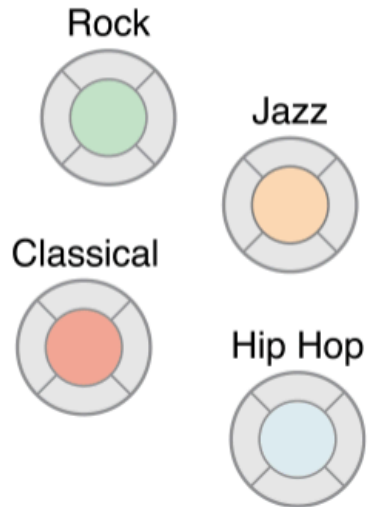
List

Indexes Values

0	Six Eggs
1	Milk
2	Flour
3	Baking Powder
4	Bananas

HashSet

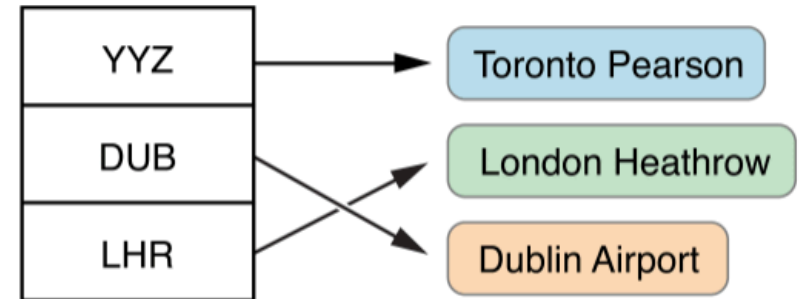
Values



HashMap

Keys

Values



List

List

- Aynı türde verileri bir arada tutar.
- İndeks numaraları 0 dan başlar.
- Veri eklendikçe otomatik olarak boyutu artar.

```
var meyveler = <String>[];
```

```
meyveler.add("Çilek");
```

```
meyveler.add("Muz");
```

```
meyveler.add("Elma");
```

```
meyveler.add("Kivi");
```

```
meyveler.add("Kiraz");
```

meyveler	
İndeks	Değer
0	Çilek
1	Muz
2	Elma
3	Kivi
4	Kiraz

List Tanımlama Yöntemleri

```
var iller = <String>[];
```

```
var plakalar = [16,34,6];
```

List'e Veri Ekleme ve Okuma

```
var meyveler = <String>[];
```

```
meyveler.add("Çilek");  
meyveler.add("Muz");  
meyveler.add("Elma");  
meyveler.add("Kivi");  
meyveler.add("Kiraz");
```

```
meyveler.add("Mandalina");//Sonuna ekleme  
print(meyveler);//[Çilek, Muz, Elma, Kivi, Kiraz, Mandalina]
```

```
meyveler[2] = "Ananas"; //Güncelleme  
print(meyveler);//[Çilek, Muz, Ananas, Kivi, Kiraz, Mandalina]
```

```
meyveler.insert(3, "Portakal");//List içine veriyi ekleme ve diğer verileri öteleme  
print(meyveler);//[Çilek, Muz, Ananas, Portakal, Kivi, Kiraz, Mandalina]
```

```
//Veri Okuma
```

```
String str = meyveler[0] ; //Çilek  
print(str);
```

```
print(meyveler[2]);//Ananas
```

meyveler	
İndeks	Değer
0	Çilek
1	Muz
2	Elma
3	Kivi
4	Kiraz

List İşlemleri

```

var meyveler = <String>[];

meyveler.add("Çilek");
meyveler.add("Muz");
meyveler.add("Elma");
meyveler.add("Kivi");
meyveler.add("Kiraz");

print(meyveler.isEmpty); //Dolu mu Boş mu kontrolü : false
print(meyveler.length); //Boyutu : 5
print(meyveler.first); //Listenin ilk elemanı : Çilek
print(meyveler.last); //Listenin son elemanı : Kiraz

print(meyveler.contains("Kiraz")); //İçeriği listede arar : true

var liste = meyveler.reversed; //Listeyi terse çevirir.
print(liste); // (Kiraz, Kivi, Elma, Muz, Çilek)

meyveler.sort(); //Listeyi sıralar
print(meyveler); // [Elma, Kiraz, Kivi, Muz, Çilek]

meyveler.removeAt(2); //İstenilen indeksteki veriyi siler.
print(meyveler); // [Elma, Kiraz, Muz, Çilek]

meyveler.remove("Kiraz"); //İstenilen elemanı siler.
print(meyveler); // [Elma, Muz, Çilek]

meyveler.clear(); //Bütün listeyi siler.
print(meyveler); // []

```

List Iterating – Döngüler ile Veri Çekme

```
var meyveler = <String>[];
```

```
meyveler.add("Çilek");  
meyveler.add("Muz");  
meyveler.add("Elma");  
meyveler.add("Kivi");  
meyveler.add("Kiraz");
```

```
for(var m in meyveler){  
    print("Sonuç : $m");  
}
```

Sonuç : Çilek
Sonuç : Muz
Sonuç : Elma
Sonuç : Kivi
Sonuç : Kiraz

```
for(var i=0;i<meyveler.length;i++){  
    print("$i. indeksteki veri : ${meyveler[i]}");  
}
```

0. indeksteki veri : Çilek
1. indeksteki veri : Muz
2. indeksteki veri : Elma
3. indeksteki veri : Kivi
4. indeksteki veri : Kiraz

Nesne Tabanlı - List

```

class Öğrenciler{
    int no;
    String ad;
    String sınıf;

    Öğrenciler(this.no, this.ad, this.sınıf);
}

```

```

var o1 = Öğrenciler(100,"Ahmet","10F");
var o2 = Öğrenciler(200,"Mehmet","12A");
var o3 = Öğrenciler(300,"Zeynep","9C");

var öğrenciler = <Öğrenciler>[];

öğrenciler.add(o1);
öğrenciler.add(o2);
öğrenciler.add(o3);

for(var o in öğrenciler){
    print("*****");
    print("Öğrenci no : ${o.no}");
    print("Öğrenci ad : ${o.ad}");
    print("Öğrenci sınıf : ${o.sınıf}");
}

```

List Sıralama

List Sıralama

Küçükten Büyüğe Sıralama

```
Comparator<Ogrenciler> siralama1 = (x,y) => x.no.compareTo(y.no);  
ogrenciler.sort(siralama1);
```

Büyükten Küçüğe Sıralama

```
Comparator<Ogrenciler> siralama2 = (y,x) => x.no.compareTo(y.no);  
ogrenciler.sort(siralama2);
```

Sıralamayı terse çevirmek için
parametrelerin yerini ters çeviriyoruz.

Örnek

```
class Öğrenciler{  
    int no;  
    String ad;  
    String sınıf;  
  
    Öğrenciler(this.no, this.ad, this.sınıf);  
}
```

```
İlk Hali  
*****  
Öğrenci no : 100  
Öğrenci ad : Ahmet  
Öğrenci sınıf : 10F  
*****  
Öğrenci no : 200  
Öğrenci ad : Mehmet  
Öğrenci sınıf : 12A  
*****  
Öğrenci no : 300  
Öğrenci ad : Zeynep  
Öğrenci sınıf : 9C
```

```
var o1 = Öğrenciler(100, "Ahmet", "10F");  
var o2 = Öğrenciler(200, "Mehmet", "12A");  
var o3 = Öğrenciler(300, "Zeynep", "9C");  
  
var öğrenciler = <Öğrenciler>[];  
  
öğrenciler.add(o1);  
öğrenciler.add(o2);  
öğrenciler.add(o3);  
  
for(var o in öğrenciler){  
    print("*****");  
    print("Öğrenci no : ${o.no}");  
    print("Öğrenci ad : ${o.ad}");  
    print("Öğrenci sınıf : ${o.sınıf}");  
}
```



```
Comparator<Ogrenciler> siralama1 = (x,y) => x.no.compareTo(y.no);  
ogrenciler.sort(siralama1);
```

Sayısal Küçükten Büyüğe `print("Sayısal Küçükten Büyüğe");`

Öğrenci no : 100

Öğrenci ad : Ahmet

Öğrenci sınıf : 10F

Öğrenci no : 200

Öğrenci ad : Mehmet

Öğrenci sınıf : 12A

Öğrenci no : 300

Öğrenci ad : Zeynep

Öğrenci sınıf : 9C

```
for(var o in ogrenciler){  
    print("*****");  
    print("Öğrenci no : ${o.no}");  
    print("Öğrenci ad : ${o.ad}");  
    print("Öğrenci sınıf : ${o.sinif}");  
}
```

```
Comparator<Ogrenciler> siralama2 = (y,x) => x.no.compareTo(y.no);  
ogrenciler.sort(siralama2);
```

`print("Sayısal Büyükten Küçüğe");`

```
for(var o in ogrenciler){  
    print("*****");  
    print("Öğrenci no : ${o.no}");  
    print("Öğrenci ad : ${o.ad}");  
    print("Öğrenci sınıf : ${o.sinif}");  
}
```

Sayısal Büyükten Küçüğe

Öğrenci no : 300

Öğrenci ad : Zeynep

Öğrenci sınıf : 9C

Öğrenci no : 200

Öğrenci ad : Mehmet

Öğrenci sınıf : 12A

Öğrenci no : 100

Öğrenci ad : Ahmet

Öğrenci sınıf : 10F

Metinsel Küçükten Büyüğe

Öğrenci no : 100

Öğrenci ad : Ahmet

Öğrenci sınıf : 10F

Öğrenci no : 200

Öğrenci ad : Mehmet

Öğrenci sınıf : 12A

Öğrenci no : 300

Öğrenci ad : Zeynep

Öğrenci sınıf : 9C

```
Comparator<Ogrenciler> siralama3 = (x,y) => x.ad.compareTo(y.ad);  
ogrenciler.sort(siralama3);
```

```
print("Metinsel Küçükten Büyüğe");
```

```
for(var o in ogrenciler){  
    print("*****");  
    print("Öğrenci no : ${o.no}");  
    print("Öğrenci ad : ${o.ad}");  
    print("Öğrenci sınıf : ${o.sinif}");  
}
```

```
Comparator<Ogrenciler> siralama4 = (y,x) => x.ad.compareTo(y.ad);  
ogrenciler.sort(siralama4);
```

```
print("Metinsel Büyükten Küçüğe");
```

```
for(var o in ogrenciler){  
    print("*****");  
    print("Öğrenci no : ${o.no}");  
    print("Öğrenci ad : ${o.ad}");  
    print("Öğrenci sınıf : ${o.sinif}");  
}
```

Metinsel Büyükten Küçüğe

Öğrenci no : 300

Öğrenci ad : Zeynep

Öğrenci sınıf : 9C

Öğrenci no : 200

Öğrenci ad : Mehmet

Öğrenci sınıf : 12A

Öğrenci no : 100

Öğrenci ad : Ahmet

Öğrenci sınıf : 10F

List Filtrerelemente

Filtreleme

- Listeyi filtrelemek için koşul yazılmalıdır.
- **Iterable** sınıfından yararlanılır.

```
Iterable<ogrenciler> filtrelenListe = ogrenciler.where((ogrenci) {  
    return ogrenci.no > 100; //Öğrenci no'su 100 den büyük olanları al.  
});
```

Şart

```
ogrenciler = filtrelenListe.toList();
```

Filtrelenen sonuç listeye aktarılır.

```
for(var o in ogrenciler){  
    print("*****");  
    print("Öğrenci no : ${o.no}");  
    print("Öğrenci ad : ${o.ad}");  
    print("Öğrenci sınıf : ${o.sinif}");  
}
```

Filtreleme Örnek

```
class Ogrenciler{  
    int no;  
    String ad;  
    String sinif;  
  
    Ogrenciler(this.no, this.ad, this.sinif);  
}
```

```
var o1 = Ogrenciler(100,"Ahmet","10F");  
var o2 = Ogrenciler(200,"Mehmet","12A");  
var o3 = Ogrenciler(300,"Zeynep","9C");
```

```
var ogrenciler = <Ogrenciler>[];
```

```
ogrenciler.add(o1);  
ogrenciler.add(o2);  
ogrenciler.add(o3);
```

```
Iterable<Ogrenciler> filtrelenListe = ogrenciler.where((ogrenci) {  
    return ogrenci.ad.contains("t") ;//Öğrenci isimlerinde t harfi içerenleri al.  
});
```

```
ogrenciler = filtrelenListe.toList();
```

```
for(var o in ogrenciler){  
    print("*****");  
    print("Öğrenci no : ${o.no}");  
    print("Öğrenci ad : ${o.ad}");  
    print("Öğrenci sınıf : ${o.sinif}");  
}
```

Öğrenci no : 100
Öğrenci ad : Ahmet
Öğrenci sınıf : 10F

Öğrenci no : 200
Öğrenci ad : Mehmet
Öğrenci sınıf : 12A

HashSet

HashSet

- List ile aynı özelliklere sahiptir.
- İçine eklenen veriler düzensiz rasgele yerleştirilir.
- Aynı veriden tekrar kayıt edilemez.

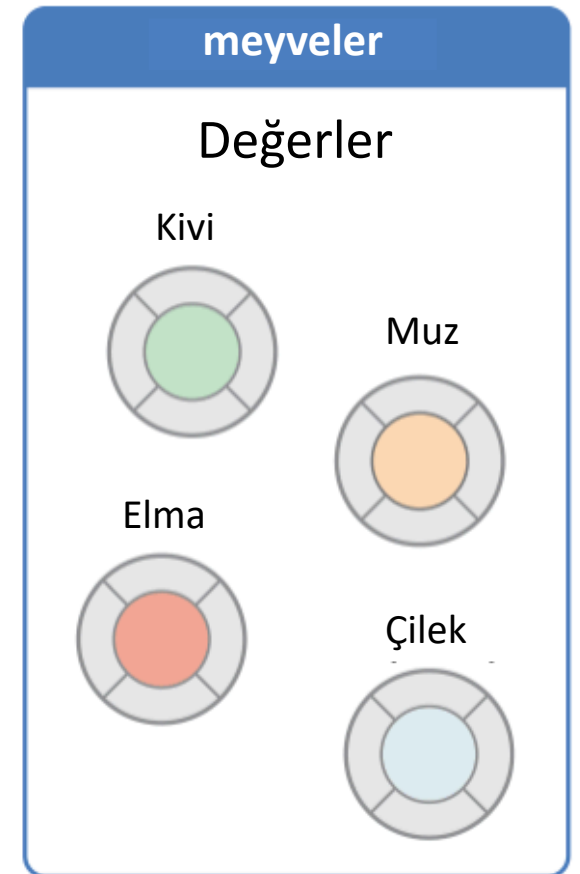
```
var meyveler = Set<String>();
```

```
meyveler.add("Çilek");
```

```
meyveler.add("Muz");
```

```
meyveler.add("Elma");
```

```
meyveler.add("Kivi");
```



HashSet Tanımlama Yöntemleri

```
var sayilar = HashSet<int>();
```

```
var isimler = HashSet.from(["Ahmet", "Mehmet"]);
```


HashSet İşlemleri

```
var sayilar = HashSet<int>();
```

```
var isimler = HashSet.from(["Ahmet", "Mehmet"]);
```

```
var meyveler = HashSet<String>();
```

```
meyveler.add("Çilek");
```

```
meyveler.add("Muz");
```

```
meyveler.add("Elma");
```

```
meyveler.add("Kivi");
```

```
print(meyveler); // {Kivi, Muz, Elma, Çilek}
```

```
meyveler.add("Elma"); // Aynı veri tekrar kayıt edilemez.
```

```
print(meyveler); // {Kivi, Muz, Elma, Çilek}
```

```
print(meyveler.elementAt(1)); // İstenilen indeksteki elemanı verir.
```

```
print(meyveler.length); // Boyutu verir : 4
```

```
print(meyveler.isEmpty); // Dolu mu boş mu kontrolü yapar : false
```

```
print(meyveler.contains("Muz")); // Liste içinde belirtilen eleman var mı : true
```

```
for(var m in meyveler){
```

```
  print("Sonuç : $m");
```

```
}
```

```
for(var i=0; i<meyveler.length; i++){
```

```
  print("$i. indeksteki veri : ${meyveler.elementAt(i)}");
```

```
}
```

```
meyveler.remove("Kivi"); // İstenilen elemanı siler
```

```
print(meyveler); // {Muz, Elma, Çilek}
```

```
meyveler.clear(); // Bütün verileri siler
```

```
print(meyveler); // {}
```

Nesne Tabanlı - HashSet

Nesne Tabanlı - HashSet

- Set yapı itibari ile içine insert edilen verileri rasgele sıralamaktadır.
- Bu rasgele sıralama int,string içeren set gibi ifadelerde kolaylıkla yapılabilir.
- Fakat set içine nesne yerleştirildiğinde nesne içindeki hangi değişkene göre bu rasgele sıralamayı yapacağını belirtmemiz gerekiyor.
- Örn : Öğrencinin nosuna göre mi ? adına göre mi ? sınıfına göre mi ? sıralama yapılacak

Örnek

```
class Ogrenciler{
    int no;
    String ad;
    String sinif;

    Ogrenciler(this.no, this.ad, this.sinif);

    @override
    int get hashCode => this.no;

    @override
    bool operator ==(other) {
        if(no == (other as Ogrenciler).no){
            return true;
        }else{
            return false;
        }
    }
}
```

```
var o1 = Ogrenciler(100, "Ahmet", "10F");
var o2 = Ogrenciler(200, "Mehmet", "12A");
var o3 = Ogrenciler(300, "Zeynep", "9C");
var o4 = Ogrenciler(300, "Zeynep", "9C");
//Aynı no daha önce kullanıldığı için hashsete eklenmez.

var ogrenciler = HashSet<Ogrenciler>();

ogrenciler.add(o1);
ogrenciler.add(o2);
ogrenciler.add(o3);
ogrenciler.add(o4);

for(var o in ogrenciler){
    print("*****");
    print("Öğrenci no : ${o.no}");
    print("Öğrenci ad : ${o.ad}");
    print("Öğrenci sınıf : ${o.sinif}");
}
```

Hash Map

HashMap

- Key ve value ilişkisi ile çalışır.
- Key ile verilere erişiriz.

```
var sayılar = HashMap<int,String>();
```

```
var oranlar = HashMap<double,String>();
```

HashMap İşlemleri

```
var iller = HashMap<int,String>();  
iller[16] = "BURSA";  
iller[34] = "İSTANBUL";  
print(iller);
```

//Güncelleme

```
iller[16] = "YENİ BURSA";  
print(iller);
```

//Veri okuma

```
String il = iller[34]!;  
print("34 key : $il");
```

```
print(iller.length);  
print(iller.isEmpty);  
print(iller.containsKey(17));  
print(iller.containsValue("İSTANBUL"));
```

```
var anahtarlar = iller.keys;
```

```
for(var a in anahtarlar){  
    print("Sonuç : ${iller[a]}");  
}
```

```
iller.remove(16);  
print(iller);
```

```
iller.clear();  
print(iller);
```

Nesne Tabanlı - HashMap

Örnek

```
class Öğrenciler{  
    int no;  
    String ad;  
    String sınıf;  
  
    Öğrenciler(this.no, this.ad, this.sınıf);  
}
```

```
var o1 = Öğrenciler(100,"Ahmet","10F");  
var o2 = Öğrenciler(200,"Mehmet","12A");  
var o3 = Öğrenciler(300,"Zeynep","9C");  
  
var öğrenciler = HashMap<int,Öğrenciler>();  
  
öğrenciler[o1.no] = o1;  
öğrenciler[o2.no] = o2;  
öğrenciler[o3.no] = o3;  
  
var anahtarlar = öğrenciler.keys;  
  
for(var a in anahtarlar){  
  
    var o = öğrenciler[a];  
  
    if (o != null){  
        print("*****");  
        print("Öğrenci no : ${o.no}");  
        print("Öğrenci ad : ${o.ad}");  
        print("Öğrenci sınıf : ${o.sınıf}");  
    }  
  
}
```

Teşekkürler...



kasim-adalan



kasimadalan@gmail.com



kasimadalan