



clarusway



clarusway



clarusway



clarusway



clarusway

# ArrayList

Lesson:  
JAVA Chapter 12

# ArrayList

## Java ArrayList Nedir?

Algoritmalarımızda kullanmak için bir çok veriye ihtiyaç duyuyoruz, bu verileri elde etme yöntemi kadar nasıl depolayacağımızda önemli bir karar aşamasına sürüklüyoruz. Bu kararı kimi zaman veri tabanı, kimi zaman ise dizilerden yana kullanıyoruz.

Kararımız dizilerden tarafa olursa, bu dizinin esnek, çeşitli imkanlar sağlayan ve kontrol etmesi kolay bir yapıda olması gereklidir. Bu istekleri de karşılayacak kavram **Java Collections Framework** altında bulunan **List** kategorisinin altında yer alan **ArrayList** kavramından geçer.

**Java ArrayList**, dinamik olarak şekillenebilen, eklemeye, çıkartma, kontrol etmeye gibi çeşitli işlemleri kolay bir şekilde yapmamızı sağlayan metodlara ev sahipliği yapan, **List** yapısının altında bulunan tek boyutlu bir dizi sınıfıdır.

**ArrayList**, **Collections Framework**'unun bir parçasıdır ve **java.util** paketinde bulunur.



# ArrayList

- **ArrayList** sınıfı, birbirini tekrar eden ve boş değerleri içerebilir,
- **ArrayList** Veri eklenip silindiğçe **ArrayList** kendi uzunluğunu otomatik olarak ayarlar.
- **ArrayList** sınıfı, sıralı bir koleksiyondur ekleme sırasını korur,
- **ArrayList** sınıfı, senkronize bir yapıya sahip değildir. Birden fazla iş bloklarını aynı anda değiştiremez,
- **ArrayList** yavaş kalma sebeplerinden biri ise dizi listesinden herhangi bir öğe silindiğinde çok fazla kaydırma işlemi yapılması gerekebilir. (**ArrayList** listesine erişim işlemi  $O(1)$ , araya ekleme(insertion) işlemi  $O(n)$  ve silme (deletion) işlemi  $O(n)$  zaman karmaşasına sahiptir.)
- **ArrayList**, öğeleri depolamak için dahili olarak bir dizi kullanır. Tıpkı diziler gibi, elemanları indekslerine göre almamızı sağlar.
- int,char, double vb. gibi primitive data type'lardan **ArrayList** oluşturamayız.
- **ArrayList** elemanları arasında **foreach** loop, iteratörler ve **indexler** yardımıyla gezinilebilir.



# ArrayList

- Programcı, ne zaman **ArrayList** ve ne zaman **Array** kullanması gerekiği konusunda ikileme düşebilir. Eğer, depoya konulacak öğe sayısı belirli ve o sayı sık sık değişmeyecekse **Array** seçimi uygun olur.
- Depolanacak öğe sayısı baştan bilinemiyor ya da o sayı sık sık değişiyorsa **ArrayList** doğru bir seçimdir.
- **ArrayList object'lerin** depolanması içindir. **Primitive data type'lar** depolamak için **Array** seçilmesi uygun olur. Bütün bunların ötesinde ArrayList sınıfı List arayüzünün metodlarını kullanma yeteneğine sahiptir; dolayısıyla **Array** yapısına oranla programcıya daha çok kolaylık sağlar.



# ArrayList

## ArrayList Create:

```
ArrayList <DataType> list1 = new ArrayList < DataType >();
```

```
ArrayList < DataType > list2 = new ArrayList<>();
```

**List < DataType > list3 = new ArrayList<>(); -> Best Practice**

**ArrayList < DataType > list4 = new List<>(); -> CTE :constructor List değil ArrayList olmalı.**

**\* TRICK :)** **DataType** : String, Integer, Double, Long, Byte, Short, Boolean, Object, .... Wrapper veya Object class olmalıdır. **DataType** kesinlikle primitive data olamaz...

## ArrayList Print:

```
System.out.println(listName);
```



# ArrayList

**ArrayList Initialize (ilk Değer Atama):**

**arrayListName.add(value); -> ArrayList' eleman ekler...**

```
List<Integer> listNums = new ArrayList<>();  
listNums.add(7);  
listNums.add(20);  
listNums.add(34);  
listNums.add(17);
```

**System.out.println(listNums);// [7, 20, 34, 17]**



# ArrayList

## ArrayList Initialize (İlk Değer Atama):

Arrays.asList(value1,value2...) -> ArrayList' elemanları hepsini aynı anda ekler...

```
List<Integer> listNums = new ArrayList<>(Arrays.asList(7, 20, 34, 17));  
System.out.println(listNums); // [7, 20, 34, 17]
```

### \*TRICK :)

```
Integer sayı[] = {1, 2, 3, 4, 5};  
List<Integer> listNums = Arrays.asList(sayı);  
System.out.println(listNums); // [1, 2, 3, 4, 5]
```

```
listNums.add(101);
```

```
Exception in thread "main" java.lang.UnsupportedOperationException Create breakpoint  
at java.base/java.util.AbstractList.add(AbstractList.java:153)  
at java.base/java.util.AbstractList.add(AbstractList.java:111)  
at ClarusWay.main(ClarusWay.java:15)
```



# ArrayList

## ArrayList Initialize (İlk Değer Atama):

List.of(value1,value2...)-> ArrayList' elemanları hepsini aynı anda ekler...

```
List<Integer> listNums = new ArrayList<>(List.of(7, 20, 34, 17));  
System.out.println(listNums); // [7, 20, 34, 17]
```

### \*TRICK :)

```
Integer sayı[] = {1, 2, 3, 4, 5};  
List<Integer> listNums = List.of(sayı);  
System.out.println(listNums); // [1, 2, 3, 4, 5]  
  
listNums.add(101);
```

```
Exception in thread "main" java.lang.UnsupportedOperationException Create breakpoint  
at java.base/java.util.AbstractList.add(AbstractList.java:153)  
at java.base/java.util.AbstractList.add(AbstractList.java:111)  
at ClarusWay.main(ClarusWay.java:15)
```



# ArrayList Methods

**listName.add(eleman);-> Ekleme yapılan eleman list'in daima sonuna eklenir. List'te tekrarlayan eleman bulunabilir.**

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println(name); // [hardy, ashley, elly]
// A) add( ) method'a index girilmezse eleman listin sonuna eklenir
name.add("jack");
//B) add( ) method'a girilen index'e eleman eklenir.
System.out.println(name); // [hardy, ashley, elly, jack]
name.add(index: 1, element: "barry");
System.out.println(name); // [hardy, barry, ashley, elly, jack]
name.add(index: 0, element: "steve");
System.out.println(name); // [steve, hardy, barry, ashley, elly, jack]
name.add(index: 1, element: "marry");
System.out.println(name); // [steve, marry, hardy, barry, ashley, elly, jack]
```



# ArrayList Methods

**listName.addAll(list2);-> Ekleme yapılan eleman list in daima sonuna eklenir.**

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println(name); // [hardy, ashley, elly]
List<String> title = new ArrayList<>(Arrays.asList("manager", "qa", "developer"));
System.out.println(title); // [manager, qa, developer]
// A addAll() methoda index girilmezse eklenen list elemanları mevcut listin sonuna eklenir.
name.addAll(title);
System.out.println(name); // [hardy, ashley, elly, manager, qa, developer]
// B addAll() eklenen list elemanları mevcut listin girilen index'inden itibaren eklenir.
name.addAll( index: 1, title);
System.out.println(name); // [hardy, manager, qa, developer, ashley, elly, manager, qa, developer]
```



# ArrayList Methods

**listName.indexOf(eleman);** -> Girilen elemanın list'teki index nosunu return eder. List'te olmayan eleman index'i sorulduğunda CTE vermez ancak Run Time'da -1 return eder.

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println(name); // [hardy, ashley, elly]
System.out.println(name.indexOf("ashley")); // 1
System.out.println(name.indexOf("jack")); // -1
```

**listName.lastIndexOf(eleman);** -> Girilen elemanın list'teki sondan itibaren tarayarak index nosunu return eder. List'te olmayan eleman index'i sorulduğunda CTE vermez ancak Run Time'da -1 return eder.

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly", "marry", "jesus", "elly"));
System.out.println(name); // [hardy, ashley, elly, marry, jesus, elly]
System.out.println(name.lastIndexOf("elly")); // 5
System.out.println(name.indexOf("jack")); // -1
```



# ArrayList Methods

**listName.get(index);** -> List'in girilen index'deki elemanı return eder.

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));

System.out.println(name); // [hardy, ashley, elly]

System.out.println("name listinin 1.index elemani : " + name.get(1)); // name listinin 1.index elemani : ashley
```

**listName.size();** -> List'in eleman sayısını return eder.

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));

System.out.println(name); // [hardy, ashley, elly]

System.out.println("name listinin eleman sayısı : " + name.size()); // name listinin eleman sayısı : 3
```



# ArrayList Methods

**listName.isEmpty();**-> List'in elemanı olup olmadığını(bos olup olmadığı) kontrol eder true -false return eder.

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println("name listi boş mu : " + name.isEmpty()); // name listi boş mu : false
```

```
List<String> title = new ArrayList<>();
System.out.println("title listi boş mu : " + title.isEmpty()); // title listi boş mu : true
```

**listName.contains(eleman);**-> Girilen elemanın list'te var olup olmadığını kontrol eder true -false return eder.

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println("name listinde ashley var mı : " + name.contains("ashley")); // name listinde ashley var mı : true
System.out.println("name listinde jack var mı : " + name.contains("jack")); // name listinde jack var mı : false
```



# ArrayList Methods

**listName.clear();-> List'in tüm elemanlarını siler herhangi bir data return etmez.**

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println("name listi boş mu : " + name.isEmpty()); // name listi boş mu : false
System.out.println("name listinde kaç eleman var : " + name.size()); // name listinde kaç eleman var : 3

name.clear();
System.out.println("name listi boş mu : " + name.isEmpty()); // name listi boş mu : true
System.out.println("name listinde kaç eleman var : " + name.size()); // name listinde kaç eleman var : 0
```



# ArrayList Methods

**listName.remove(eleman);**-> List'in girilen elemanını siler ve elemanın silinip silinmediğini true-false return eder.

**listName.remove(index);**-> List'in girilen index'deki elemanı siler ve silinen elemani return eder.

\* TRICK :) **index < listSize**

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println(name); // [hardy, ashley, elly]
System.out.println("name listteki elly silindi mi : " + name.remove(0: "elly")); // name listteki elly silindi mi : true
System.out.println("name listteki 1. indexdeki silinen eleman : " + name.remove( index: 1)); // name listteki 1. index eleman silindi : ashley
System.out.println(name); // [hardy]
System.out.println("name listteki jack silindi mi : " + name.remove(0: "jack")); // name listteki jack silindi mi : false
System.out.println("name listteki 3. indexdeki silinen eleman : " + name.remove( index: 1)); // IndexOutOfBoundsException: Index 1 out of bounds for length 1 < 3
```



# ArrayList Methods

`listName.set(index,eleman);`-> List'in girilen index 'deki elemanı girilen eleman ile update eder.

\* TRICK :) `index < listSize` `set()` methodu `add()` methodu gibi eleman eklemez; sadece var olan elemani değiştirir.

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println(name); // [hardy, ashley, elly]
name.set(2, "jack");
System.out.println(name); // [hardy, ashley, jack]
name.set(5,"marry"); // IndexOutOfBoundsException: Index 5 out of bounds for length 3
```



# ArrayList Methods

**Collection.sort(listName);** -> Collection framework'den call edilen sort() methoda list'e parametre olarak girilen list'in elemanları NATUREL ORDER (k->b,alfabetik) olarak sıralanır.

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println(name); // [hardy, ashley, elly]
Collections.sort(name);
System.out.println(name); // [ashley, elly, hardy]
```

```
List<Integer> nums = new ArrayList<>(Arrays.asList(7, 20, 34, 17));
System.out.println(nums); // [7, 20, 34, 17]
Collections.sort(nums);
System.out.println(nums); // [7, 17, 20, 34]
```



# ArrayList Methods

**Collection.reverse(listName);**-> Collection framework'den call edilen reverse() methoda list'e parametre olarak girilen list'in elemanları **index'ine göre ters (index b->k)** olarak sıralanır.

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println(name); // [hardy, ashley, elly]
Collections.reverse(name);
System.out.println(name); // [elly, ashley, hardy]
```

```
List<Integer> nums = new ArrayList<>(Arrays.asList(7, 20, 34, 17));
System.out.println(nums); // [7, 20, 34, 17]
Collections.reverse(nums);
System.out.println(nums); // [17, 34, 20, 7]
```



# ArrayList Methods

`listName.equals(list2);`-> İki listen **eleman ve indexlerin** karşılaştırıp eşitliğini kontrol eder ve **true-false return** eder.

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println(name); // [hardy, ashley, elly]
```

```
List<String> isim = new ArrayList<>(Arrays.asList("ashley", "elly", "hardy"));
System.out.println(isim); // [ashley, elly, hardy]
```

```
System.out.println("name ve isim list eşit mi : " + name.equals(isim)); // name ve isim list eşit mi : false
```

```
List<String> bestIt = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly")); // [hardy, ashley, elly]
System.out.println("name ve bestIt list eşit mi : " + bestIt.equals(name)); // name ve bestIt list eşit mi : true
```



# Array'i ArrayList'e Çevirmek

```
List<String> name = new ArrayList<>(Arrays.asList("hardy", "ashley", "elly"));
System.out.println(name); // [hardy, ashley, elly]
// 1. yol...
String nameArr[] = name.toArray(new String[0]);
System.out.println(nameArr.length); // 3
System.out.println(Arrays.toString(nameArr)); // [hardy, ashley, elly]
// 2. yol...
Object nameArr1[] = name.toArray(new String[0]);
System.out.println(nameArr1.length); // 3
System.out.println(Arrays.toString(nameArr1)); // [hardy, ashley, elly]
```



# Array'i ArrayList'e Çevirmek

```
String nameArr[] = {"hardy", "ashley", "elly"};
List<String> name = Arrays.asList(nameArr);
System.out.println(name); // [hardy, ashley, elly]
name.add("jack"); // RTE-> Exception in thread "main" java.lang.UnsupportedOperationException
System.out.println(name); // [HARDY, ASHLEY, ELLY]
name.remove( index: 1); // RTE-> Exception in thread "main" java.lang.UnsupportedOperationException
System.out.println(name);
/*
Ahan da TRICK :)
Array'den list'e cevirma yapildiginda, elde edilen list uzunluk olarak esnek degildir.
Yani array'den olusturdugunuz list'e ekleme ve cikarma yapilamaz. code CTE vermez ancak
name.add("jack"); name.remove(1); methodlari RTE hata verir.
*/
```



# THANKS!

Any questions?

HalUk Bilgin | JAVA Backend Developer

BAŞARI GAYRETE AŞIKTIR ☺

