

# Iterator Arayüzü

---

`java.util`  
**Interface Iterator**

**Altarayüzleri:**

[ListIterator](#)

**Kılğılayan sınıflar:**

[BeanContextSupport.BCSIterator](#)

public interface **Iterator**

biçiminde tanımlı olan Iterator Java 1.2 sürümüyle gelmiştir. [Java Collections Framework](#) çatısına aittir. *Iterator*, koleksiyonlar üzerinde, *Enumerations* arayüzünün yerini alır. Ama ondan iki nedenle farklıdır:

- Iterator bir koleksiyon üzerinde tekrarlama eylemini yaparken koleksiyondan öğeler silinmesine izin veren [remove\(\)](#) metoduna sahiptir.
- Metot adları, yaptıkları eyleme uyacak biçimde değişmiştir.

**Ayrıca bakınız:**

[Collection](#), [ListIterator](#), [Enumeration](#)

## Iterator Methodları

boolean	<a href="#">hasNext()</a> Tekrarlamada hâlâ öğe varsa <code>true</code> değerini verir.
Object	<a href="#">next()</a> Tekrarlamada sonraki öğeyi verir.
void	<a href="#">remove()</a> Tekrarlamada iteratörün verdiği son öğeyi, istendiğinde siler.

## Iterator Kullanımı

Tekrarlayıcı (iterator) [List](#) ve [Set](#) koleksiyonları üzerinde, bütün öğeleri tarayacak biçimde tekrarlanan eylemleri gerçekleştirmek için kullanılır.

`java.util.Iterator<E>` arayüzü koleksiyonu baştan sona doğru tek yönlü tarar.

`java.util.ListIterator<E>` arayüzü ise, koleksiyonu baştan sona doğru ya da sondan başa doğru tarayabilir.

`Iterator<E>` arayüzü eski [Enumeration](#) arayüzü yerine konmuştur.

Java *collection*'a ait her koleksiyon sınıfının bir *iterator()* metodu vardır. Bu metot koleksiyonun ilk öğesinden başlar, son öğesine doğru bütün öğeleri tarar. Dolayısıyla, koleksiyondaki her öğeye erişim sağlanır. Metodu kullanabilmek için şunlar yapılmalıdır:

1. Hangi koleksiyon için gerekiyorsa, ona ait *iteraor()* metodu çağrılır.
2. *hasNext()* metodu yardımıyla koleksiyonu tarayacak döngü kurulur. Bu metot, koleksiyonda henüz döngü adımlarının erişmediği öğe varsa *true* değerini verir.
3. Sonraki öğeye erişmek için *next()* metodunu kullan.

*List* arayüzünü kılıglayan koleksiyonlar için *Iterator* yerine *ListIterator* kullanılır. *ListIterator*, listeyi baştan sona ya da sondan başa doğru tarama yeteneğine sahiptir; dolayısıyla sözkonusu listeler için tercih edilmelidir.

Aşağıdaki örnek iteratörün kullanımını göstermektedir.

### Örnek 1

```
import java.util.*;

public class Iterator01 {
    public static void main(String args[]) {
        // bir ArrayList yarat
        ArrayList al = new ArrayList();
        // arrayliste öğeler ekle
        al.add("C");
        al.add("A");
        al.add("E");
        al.add("B");
        al.add("D");
        al.add("F");
        // al ArrayList'ini yazdır
        System.out.print("Orijinal Listenin öğeleri      : ");
        Iterator itr = al.iterator();
        while (itr.hasNext()) {

            Object element = itr.next();
            System.out.print(element + " ");

        }
        System.out.println();
        // Değişmiş listeyi tarıyor
        ListIterator litr = al.listIterator();
        while (litr.hasNext()) {

            Object element = litr.next();
            litr.set(element + "#");

        }
        System.out.print("Değiştirilmiş Listenin öğeleri  : ");
        itr = al.iterator();
        while (itr.hasNext()) {

            Object element = itr.next();
            System.out.print(element + " ");

        }
        System.out.println();
        // ArrayListi ters sırada yaz
```

```

System.out.print("Değiştirilen Listenin ters sırası: ");
while (litr.hasPrevious()) {

    Object element = litr.previous();
    System.out.print(element + " ");

}
System.out.println();
}
}

```

### Örnek 1

```

import java.util.*;

public final class Iterator01 {

    private static void whileLoop(Collection<String> dondurma) {
        Iterator<String> seçimIter = dondurma.iterator();
        while (seçimIter.hasNext()) {
            System.out.println(seçimIter.next());
        }
    }

    /**
     * for-döngüsü tamsayı indis kullanmıyor
     */
    private static void forLoop(Collection<String> dondurma) {
        for (Iterator<String> seçimIter = dondurma.iterator();
        seçimIter
            .hasNext();) {
            System.out.println(seçimIter.next());
        }
    }

    public static void main(String[] args) {
        List<String> seçim = new ArrayList<String>();
        seçim.add("çukolatalı");
        seçim.add("çilekli");
        seçim.add("vanilyalı");

        whileLoop(seçim);

        forLoop(seçim);
    }
}

```