



Iterable

Rawlabs Academy

What is **Iterable**?

The "**Iterable**" was introduced to be able to use in the *"foreach"* loop.
A class implementing the iterable can be iterated over.

Java **Iterable**

Hierarchy Interface



General-purpose **Implementation**

General-purpose implementations are the most commonly used implementations, designed for everyday use.

Interface	Hash Table	Resizable Array	Tree	Linked List	Hash Table + Linked List
Set	HashSet		TreeSet		LinkedHashSet
List		ArrayList		LinkedList	
Deque		ArrayDeque		LinkedList	

Iterable **Methods**



Example Iterable

```
public static void main(String[] args) {  
    Iterable<String> names = List.of("rawlabs", "academy");  
    for (String name : names) {  
        System.out.println(name);  
    }  
}
```

Iterator

The iterable interface has one `iterator()` method.

Iterator is class that manages iteration over an `Iterable`. It maintains a state of where we are in the current iteration, and knows what the next element is and how to get it.



Example **Iterator**

```
import java.util.Iterable;
import java.util.Iterator;
import java.util.List;

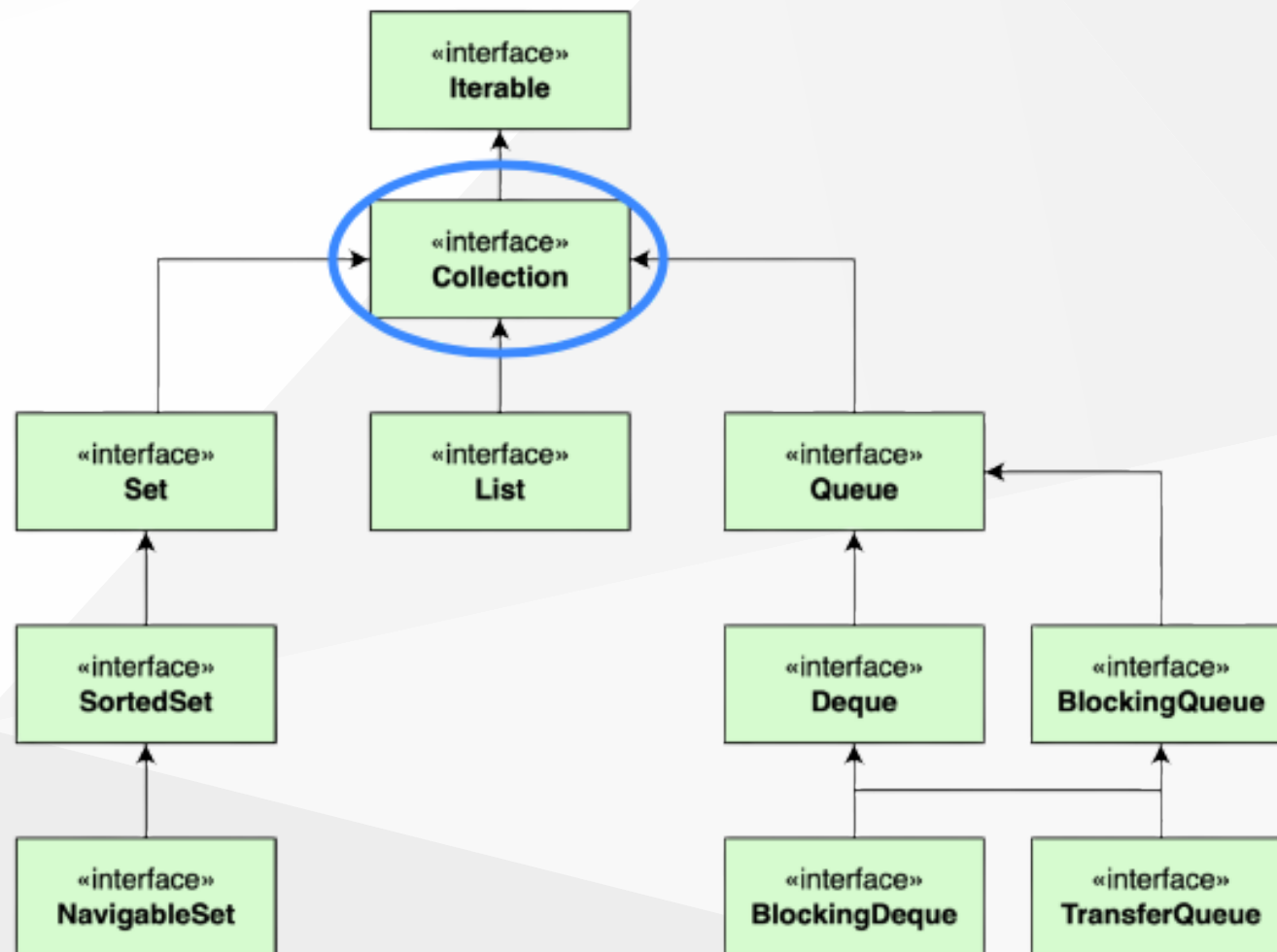
public class Main {
    public static void main(String[] args) {
        Iterable<String> names = List.of("rawlabs", "academy");
        Iterator<String> iter = names.iterator();
        while(iter.hasNext()) {
            String name = iter.next();
            System.out.println(name);
        }
    }
}
```


Java Collection

Any group of individual objects which are represented as a single unit is known as the collection of the objects.






















Java Collection

Hierarchy Interface



Collection

Methods

```
Collection.java
☐ Inherited members (⌘F12) ☐ Anonymous Classes (⌘I) ☐ Lambdas (⌘L) 
▼   Collection
  (m)   add(E): boolean
  (m)   addAll(Collection<? extends E>): boolean
  (m)   clear(): void
  (m)   contains(Object): boolean
  (m)   containsAll(Collection<?>): boolean
  (m)   equals(Object): boolean ↑Object
  (m)   hashCode(): int ↑Object
  (m)   isEmpty(): boolean
  (m)   iterator(): Iterator<E> ↑Iterable
  (m)   parallelStream(): Stream<E>
  (m)   remove(Object): boolean
  (m)   removeAll(Collection<?>): boolean
  (m)   removeIf(Predicate<? super E>): boolean
  (m)   retainAll(Collection<?>): boolean
  (m)   size(): int
  (m)   spliterator(): Spliterator<E> ↑Iterable
  (m)   stream(): Stream<E>
  (m)   toArray(): Object[]
  (m)   toArray(IntFunction<T[]>): T[]
  (m)   toArray(T[]): T[]
```

Collection Example

```
public class Main {  
    public static void main(String[] args) {  
        Collection<String> names = new ArrayList<>();  
        names.add("Rawlabs");  
        names.add("Academy");  
        names.add("Bootcamp");  
  
        names.remove("Bootcamp");  
        System.out.println(names.contains("Rawlabs"));  
    }  
}
```

Task

Make a summary for `Iterable` and `Collection` material and provide examples of implementation other than those contained in the material.

Implementation includes :

- While loop
- Foreach loop
- Implementation methods of `Iterable` and `Collection`