

Industrial strength Java/JEE Career Companion to open more doors

search here ...

Go

Home

Java FAQs

600+ Java Q&amp;As

Career

Tutorials

Member

Why?

Can u Debug?

Java 8 ready?

Top X

Productivity Tools

Judging Experience?

[Home](#) › [Interview](#) › [Core Java Interview Q&A](#) › [Collection and Data structures](#) › ♥

Java Iterable Vs Iterator differences and know how

# ♥ Java Iterable Vs Iterator differences and know how

Posted on [July 8, 2015](#) by [Arulkumaran Kumaraswamipillai](#)

5

Like

Share

Tweet

3

51

G+1

Share

**Q1.** What are the differences between `Iterator<T>` Vs `Iterable<T>`?

**A1.** The “**Iterable**” was introduced to be able to use in the “foreach” loop. A class implementing the `Iterable` interface can be iterated over. For example,

```
1 public interface Collection<E> extends Iterable<E>
```

Hence, it can be used in the foreach loop

```
1 import java.util.Arrays;
2 import java.util.List;
3
4 public class IterableVsIterator {
5
```

## 600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

[open all](#) | [close all](#)

+ Ice Breaker Interview

+ Core Java Interview C

+ Java Overview (4)

+ Data types (6)

+ constructors-methc

+ Reserved Key Wor

+ Classes (3)

+ Objects (8)

+ OOP (10)

+ GC (2)

+ Generics (5)

+ FP (8)

+ IO (7)

+ Multithreading (12)

+ Algorithms (5)

+ Annotations (2)

+ Collection and Dat

+ ♦ Find the first n

+ ♦ Java Collector

+ ♥ Java Iterable \

+ ♥♦ HashMap &amp; H

```

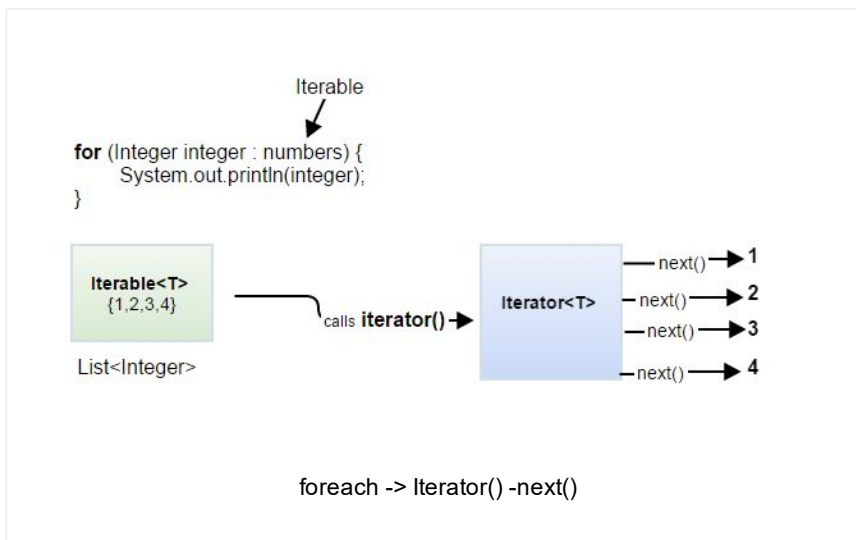
6      public static void main(String[] args) {
7          //numbers is iterable as List<E> extends
8          //Collection<E> extends Iterable<E>
9          List<Integer> numbers = Arrays.asList(new
10         for (Integer integer : numbers) {
11             System.out.println(integer);
12         }
13     }
14 }

```

The Iterable interface has one method iterator()

```
1 Iterator<T> iterator()
```

## So, what happens in the foreach loop?



## What is an Iterator()?

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.

Here is the **Iterator()** API methods.

| Method Summary    |   |
|-------------------|---|
| All Methods       | Instance Methods  |
| Modifier and Type | Method and Description  |
| default void      | <b>forEachRemaining(Consumer&lt;? super E&gt; action)</b><br>Performs the given action for each remaining element until all elements have been processed or the action throws an exception. |
| boolean           | <b>hasNext()</b><br>Returns true if the iteration has more elements.  |
| E                 | <b>next()</b><br>Returns the next element in the iteration.   |
| default void      | <b>remove()</b><br>Removes from the underlying collection the last element returned by this iterator (optional operation).  |

- ◆ Sorting objects
- 02: ◆ Java 8 Str
- 04: Understandir
- 4 Java Collection
- If Java did not ha
- Java 8: Different
- Part-3: Java Tre
- Sorting a Map by
- When to use whi
- + Differences Between
- + Event Driven Progr
- + Exceptions (2)
- + Java 7 (2)
- + Java 8 (24)
- + JVM (6)
- + Reactive Programrn
- + Swing & AWT (2)
- + JEE Interview Q&A (3
- + Pressed for time? Jav
- + SQL, XML, UML, JSC
- + Hadoop & BigData Int
- + Java Architecture Inte
- + Scala Interview Q&As
- + Spring, Hibernate, & I
- + Testing & Profiling/Sa
- + Other Interview Q&A 1
- + Free Java Interview

## 16 Technical Key Areas

open all | close all

- + Best Practice (6)
- + Coding (26)
- + Concurrency (6)
- + Design Concepts (7)
- + Design Patterns (11)
- + Exception Handling (3)
- + Java Debugging (21)
- + Judging Experience I

## Java Iterator methods

# Iterable can be rewound

You can **rewind** an Iterable by getting a new iterator() from the Iterable. But an Iterator only has the next() method. So, an Iterable is handy if you want to traverse the elements more than once.

**Q2.** How do you get an **Iterable** in Java 8 to be used in a foreach loop?

**A2.** In Java 8, when you call the double “::” notation as shown below, you get an Iterable from the Stream.

So, given a Stream s, the following results in an Iterable:

```
1 s::iterator
```

If you want to use this directly in an enhanced-for loop, you have to apply a cast in order to establish a target type for the method reference.

Here is the complete code:

```
1 package com.read.file;
2
3 import java.net.URL;
4 import java.nio.charset.StandardCharsets;
5 import java.nio.file.Files;
6 import java.nio.file.Path;
7 import java.nio.file.Paths;
8 import java.util.stream.Stream;
9
10 public class MyFileReader {
11
12     public static void main(String[] args) {
13         Path file = null;
14         try
15         {
16             //read from the classpath
17             URL url = MyFileReader.class.getResource(
18                 "file.txt");
19             file = Paths.get(url.toURI());
20
21             //Java 8: Stream class
22             Stream<String> lines = Files.lines(
```

- ⊞ [Low Latency \(7\)](#)
- ⊞ [Memory Management \(13\)](#)
- ⊞ [Performance \(13\)](#)
- ⊞ [QoS \(8\)](#)
- ⊞ [Scalability \(4\)](#)
- ⊞ [SDLC \(6\)](#)
- ⊞ [Security \(13\)](#)
- ⊞ [Transaction Management \(13\)](#)

## 80+ step by step Java Tutorials

[open all](#) | [close all](#)

- ⊞ [Setting up Tutorial \(6\)](#)
- ⊞ [Tutorial - Diagnosis \(2\)](#)
- ⊞ [Akka Tutorial \(9\)](#)
- ⊞ [Core Java Tutorials \(2\)](#)
- ⊞ [Hadoop & Spark Tutorial \(1\)](#)
- ⊞ [JEE Tutorials \(19\)](#)
- ⊞ [Scala Tutorials \(1\)](#)
- ⊞ [Spring & Hibernate Tutorials \(1\)](#)
- ⊞ [Tools Tutorials \(19\)](#)
- ⊞ [Other Tutorials \(45\)](#)

## 100+ Java pre-interview coding tests

[open all](#) | [close all](#)

- ⊞ [Can you write code? \(1\)](#)
- ⊞ [Complete the given code \(1\)](#)
- ⊞ [Converting from A to B \(1\)](#)
- ⊞ [Designing your class \(1\)](#)
- ⊞ [Java Data Structures \(1\)](#)
- ⊞ [Passing the unit tests \(1\)](#)
- ⊞ [What is wrong with this code? \(1\)](#)
- ⊞ [Writing Code Home Assignment \(1\)](#)
- ⊞ [Written Test Core Java \(1\)](#)

```

23         for( String line : (Iterable<String>
24             {
25                 System.out.println("read=" + lin
26             }
27
28         } catch (Exception e){
29             e.printStackTrace();
30         }
31     }
32 }

```

[Written Test JEE \(1\)](#)

## How good are your .....?

[open all](#) | [close all](#)
[Career Making Know-](#)
[Job Hunting & Resum](#)

## Here is Iterator and Iterable in action via a custom Iterable class

This is the **Iterator** design pattern in action.

```

1  import java.util.Iterator;
2
3  public class MyIterable<T> implements Iterable<T>
4
5      public T[] a= null;
6
7      public MyIterable(T[] array){
8          this.a = array;
9      }
10
11     @Override
12     public Iterator<T> iterator() {
13         return new Iterator<T>() {
14             private int count=0;
15
16             public boolean hasNext(){
17                 return count < a.length;
18             }
19             public T next(){
20                 System.out.println("Invoking nex
21                 return a[count++];
22             }
23
24             public void remove(){
25                 throw new UnsupportedOperationException
26             }
27         };
28     }
29 }

```

Now, using the above class

```

1  public class IterableVsIterator {
2
3      public static void main(String[] args) {
4          String[] technologies = {"Java", "JEE",
5          MyIterable<String> entries = new MyItera
6          for (String tech : entries) {

```

```
7         System.out.println(tech);  
8     }  
9 }  
10 }
```

## Output:

```
1 Invoking next()  
2 Java  
3 Invoking next()  
4 JEE  
5 Invoking next()  
6 XML
```

The “Invoking next()” is printed from the **iterator()** anonymous class’s **next()** method.

## Popular Posts

♦ 11 Spring boot interview questions & answers

823 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

765 views

18 Java scenarios based interview Questions and Answers

399 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

388 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

295 views

♦ 7 Java debugging interview questions & answers

293 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

285 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

279 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

239 views

## 001B: ♦ Java architecture &amp; design concepts

## interview questions &amp; answers

201 views

Bio

Latest Posts

**Arulkumaran  
Kumaraswamipillai**

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via [Amazon.com](https://www.amazon.com) in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site. **945+** paid members. [join my LinkedIn Group](#). [Reviews](#)

**About Arulkumaran Kumaraswamipillai**

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via [Amazon.com](https://www.amazon.com) in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site. **945+** paid members. [join my LinkedIn Group](#). [Reviews](#)

&lt; 06: ♥ Java 8 way of File reading and functionally processing the data

SSL in Java with Keytool to generate public-private key pair &gt;

**Posted in** Collection and Data structures, Differences Between X & Y**Tags:** Free Content

# Empowers you to open more doors, and fast-track

## Technical Know Hows

☀ [Java generics in no time](#) ☀ [Top 6 tips to transforming your thinking from OOP to FP](#) ☀ [How does a HashMap internally work? What is a hashing function?](#)  
☀ [10+ Java String class interview Q&As](#) ☀ [Java auto un/boxing benefits & caveats](#) ☀ [Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect](#)

## Non-Technical Know Hows

☀ [6 Aspects that can motivate you to fast-track your career & go places](#) ☀ [Are you reinventing yourself as a Java developer?](#) ☀ [8 tips to safeguard your Java career against offshoring](#) ☀ [My top 5 career mistakes](#)

## Prepare to succeed

☀ [Turn readers of your Java CV go from “Blah blah” to “Wow”?](#) ☀ [How to prepare for Java job interviews?](#) ☀ [16 Technical Key Areas](#) ☀ [How to choose from multiple Java job offers?](#)

Select Category ▼

## © Disclaimer

The contents in this Java-Success are copy righted. The author has the right to correct or enhance the current content without any prior notice.

These are general advice only, and one needs to take his/her own circumstances into consideration. The author will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. No guarantees are made regarding the accuracy or usefulness of content, though I do make an effort to be accurate. Links to external sites do not imply endorsement of the linked-to sites.