How to create Iterator in Java

This article will help to make people aware on the process to create Iterator in Java, how to use the same and be an introduction to List Iterator as well.

What is an Iterator in Java? This is basically but a traversing object that is made specifically for Collection objects like List and Set. There are lot of different kind of traversing methods such as for-loop ,while loop,do-while,for each lop etc. They all are considered to be index based traversing however for the reason that we are aware on Java being a purely object oriented language; there is always possible ways of doing things making use of objects. Hence this makes Iterator a way to traverse as well as access the data from the collection. As far as the traversing with object is concerned, we have Enumeration, Iterator and ListIterator as well in Java.

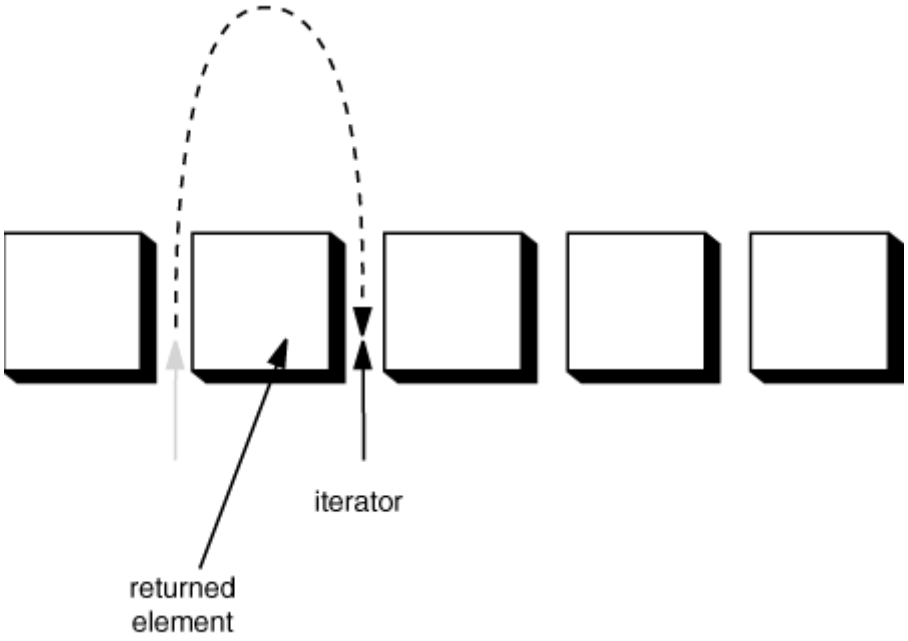

Figure 1: Iterators in Java

Introduction to Iterator in Java API

Java iterator is basically an interface that belongs to collection framework. This framework permits the user to navigate the collection and access the data element of collection without interrupting the user about particular implementation of that collection. Basically List and set collection can provide the iterator that can be obtained from ArrayList, LinkedList, and TreeSet etc. Talking about the Map implementation such as HashMap, it doesn't provide Iterator directory however one can get there keySet or Value Set and can iterator via that collection.

Syntax:

It is known to come inside java.util package. as public interface Iterator and comprise of three methods:

- boolean hasNext(): this method is known to return true in case this Iterator has more element to iterate.
- Object next(): This method return the next element in the collection till the time the hasNext() method return true. It is always recommended to call hasNext() method prior to calling next() method so as to avoid java.util.NoSuchElementException: Hashtable Iterator

- remove(): method is used to remove the last element that is being returned by the iterator and this method only calls once per call to next().

Creation of Iterator in Java:

Each of the collection classes is used to provide an iterator() method that returns an iterator to the beginning of the collection making use of which, one can access each element in the collection, one element at a time. One can follow the below steps in order to make use of an iterator to traverse through the contents of a collection:

1. The first step is to obtain an iterator to the beginning of the collection. This is done by calling the collection's iterator( )
2. Next would be to set up a loop that makes a call to hasNext( ) and then have the loop iterate as long as hasNext( ) returns true.
3. Finally, within that loop, obtain each element by calling next( ).

Process to use Iterator in Java

The below code lists the process of using Iterator in Java that is used to create Hashtable, populate it with dummy data. Post this, it use Iterator so as to traverse Map and prints key and value for each Entry. Also the below Java Program makes use of the Generic version of Iterator and this will ensure type-safety and avoid casting.

Listing 1: Using Iterator in Java

```
import java.util.Hashtable;
import java.util.Iterator;
import java.util.Map;
import java.util.Map.Entry;
import java.util.Set;

/**
 * Simple Java Program that displays the process to use Iterator in Java to loop through all
 * of Collection classes like ArrayList, LinkedList, HashSet or Map implementation like
 * HashMap,TreeMap and Hashtable.
 */
public class IteratorExample{

    public static void main(String[] args) {
```

```
        //Hashtable instance used for Iterator Example in Java
        Hashtable<Integer, String> stockTable=new Hashtable<Integer,String>();

        //Populating Hashtable instance with sample values
        stockTable.put(new Integer(1), "Two");
        stockTable.put(new Integer(2), "One");
        stockTable.put(new Integer(4), "Four");
        stockTable.put(new Integer(3), "Three");

        //Getting Set of keys for Iteration
        Set<Entry<Integer, String>> stockSet = stockTable.entrySet();

        // Making use of Iterator to loop Map in Java, here Map implementation is Hashtable
        Iterator<Entry<Integer, String>> i= stockSet.iterator();
        System.out.println("Iterating over Hashtable in Java");

        //Iterator begins
        while(i.hasNext()){
            Map.Entry<Integer,String> m=i.next();
            int key = m.getKey();
            String value=m.getValue();
            System.out.println("Key :"+key+"  value :"+value);

        }
        System.out.println("Iteration over Hashtable finished");
    }
}
```


Output:

Iterating over Hashtable in Java

Key: 4 value: Four

Key: 3 value: Three

Key: 2 value: One

Key: 1 value: Two

Iteration over Hashtable finished

Introduction to List Iterator in Java

ListIterator in Java is an Iterator that permits the user so as to traverse Collection like ArrayList and HashSet in both direction by making use of the method previous() and next (). One can also obtain ListIterator from all List implementation that comprises of ArrayList and LinkedList.

As far as current index is concerned, ListIterator doesn't keep the same and its current position is largely determined by call to next () or previous () based on direction of traversing.

Key Notes on Iterator in Java:

1. Iterator in Java basically is used to support generics thereby making it mandatory to make use of the Generic version of Iterator and not the Iterator with raw type.
2. In case the objects need to be removed from the Collection, in that case do not make use of for-each loop. Instead one can make use of Iterator's remove() method to avoid any ConcurrentModificationException.
3. As far as Iterating over collection with the help of Iterator concerned, it is subject to ConcurrentModificationException if Collection is modified after Iteration started. However this is meant to take place only in case of fail-fast Iterators.
4. We have fail-fast and fail-safe type of Iterators in Java and you need to check for the difference between these types.
5. List collection type is also known to supports ListIterator that comprises of add() method so as to incorporate elements in collection at the time of iteration. Difference between Iterator

Conclusion

We learnt how to create Iterator in Java, also looked around the process to use the same. We also had a glance over the introduction of list Iterator in Java and saw the difference between Iterator and List Iterator.

Varun Arora

I am a software developer from India with hands on experience on java, html for over 5 years.

What did you think of this post? Like (0) (0)

Be the first to comment

Services
Add a comment Add to favorites Mark as viewed Add a note

+Java

Related Posts

BOOL LOGIN

Post

First Steps in Java Persistence API (JPA)

Post

Handling Enumerations in Java

Post

Working with RESTful Web Services in Java

Post

Hibernate 4: Whats new?

Post

Working with JSON Data in Java

Post

Handling Exceptions in a Struts 2 Application

Post

Understanding Java 8 Lambda Expressions

Show more

Contact us | Publish your post

BOOL LOGIN

MrBool.com