**Java – LinkedList ListIterator example**

[**JAVA COLLECTIONS**](http://beginnersbook.com/category/java-collections/)

In this example we will see how to [**iterate a LinkedList**](http://beginnersbook.com/2013/12/how-to-loop-linkedlist-in-java/) using [**ListIterator**](http://beginnersbook.com/2014/06/listiterator-in-java-with-examples/). Using Listterator we can iterate the list in both the directions(forward and backward). Along with traversing, we can also modify the list during iteration, and obtain the iterator’s current position in the list. Read more about it at [**ListIterator javadoc**](http://docs.oracle.com/javase/7/docs/api/java/util/ListIterator.html).

**Example**

Here we have a LinkedList of Strings and we are traversing it in both the directions using LitIterator.

import java.util.LinkedList;

import java.util.ListIterator;

public class ListIteratorExample {

public static void main(String[] args) {

// Create a LinkedList

LinkedList<String> linkedlist = new LinkedList<String>();

// Add elements to LinkedList

linkedlist.add("Delhi");

linkedlist.add("Agra");

linkedlist.add("Mysore");

linkedlist.add("Chennai");

linkedlist.add("Pune");

// Obtaining ListIterator

ListIterator listIt = linkedlist.listIterator();

// Iterating the list in forward direction

System.out.println("Forward iteration:");

while(listIt.hasNext()){

System.out.println(listIt.next());

}

// Iterating the list in backward direction

System.out.println("\nBackward iteration:");

while(listIt.hasPrevious()){

System.out.println(listIt.previous());

}

}

}

**Output:**

Forward iteration:

Delhi

Agra

Mysore

Chennai

Pune

Backward iteration:

Pune

Chennai

Mysore

Agra

Delhi