**java.util.LinkedList<E> class**

Java LinkedList is a doubly linked list implementation of Java’s List and Deque interfaces. It is part of Java’s collections framework. Here is the class hierarchy of LinkedList:

Diagram

Description automatically generated

**Example of creating a LinkedList objects**

import java.util.LinkedList;

// . . .

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

LinkedList<Integer> integerList = new LinkedList<Integer>();

**Methods of LinkedList and those it inherits**

|  |  |  |
| --- | --- | --- |
| Type of method | method | description |
| insertion | **void addLast**(E e) | Appends the specified element to the end of this list. |
| **addFirst**(E e) | Inserts the specified element at the beginning of this list. |
| **void add**(int index, E element) | Inserts the specified element at the specified position in this list. The first element is at index 0 |
| deletion | **E removeFirst**() | Removes and returns the first element from this list. |
| **E removeLast**() | Removes and returns the last element from this list. |
| **E remove**(int index) | Removes the element at the specified position in this list. The first element is at index 0 |
| boolean removeFirstOccurrence(Object o) | Removes the first occurrence of the specified element in this list (when traversing the list from head to tail). |
| boolean removeLastOccurrence(Object o) | Removes the last occurrence of the specified element in this list (when traversing the list from head to tail). |
| search | **int indexOf(Object o)** | Returns the index of the first occurrence of the specified element in this list, or -1 if this list does not contain the element. The first element is at index 0 |
| **int lastIndexOf**(Object o) | Returns the index of the last occurrence of the specified element in this list, or -1 if this list does not contain the element. The first element is at index 0 |
| boolean contains(Object o) | Returns true if this list contains the specified element. |
| replacement | **E set**(int index, E element) | Replaces the element at the specified position in this list with the specified element. Returns the replaced element. |
| Iterator | **Iterator<E> iterator**() | Returns an iterator over the elements in this list. |
| size | int size() | Returns the number of elements in this list. |
| isEmpty | boolean isEmpty() | Returns true if this list contains no elements.  This implementation returns size() == 0. |
| clear | void clear() | removes all the elements in the list. However, the reference of the list created is still stored. |
| conversion to array | Object[] **toArray**() | Returns an array containing all of the elements in this list in proper sequence (from first to last element). |

**Iterating over a LinkedList**

One can iterate over a LinkedList using:

* iterator()
* simple for-each loop

|  |
| --- |
| **import** java.util.\*;  **public** **class** LinkedListDriver1 {  **public** **static** **void** main(String args[]){          LinkedList<String> list = **new** LinkedList<>();          list.add("Riyadh");list.add("Dammam"); list.add("Jubail");          Iterator<String> iterator = list.iterator();  **while** (iterator.hasNext()) {              System.out.print(iterator.next() + " ");          }      }  } |
| import java.util.LinkedList;  public class LinkedListDriver6 {  public static void main(String args[]){  LinkedList<String> list = new LinkedList<>();  list.add("Riyadh");  list.add("Dammam");  list.add("Jubail");    for(String cityName : list) {  System.out.print(cityName + " ");  }  } } |

Example:

**import java.util.LinkedList;  
import java.util.Iterator;  
  
class LinkedListDriver {  
 public static void main(String[] args) {   
 LinkedList<String> stringList = new LinkedList<>();  
  
 stringList.add("Madinah");  
 stringList.add("Dammam");  
 stringList.add("Riyadh");  
 stringList.add("Dhahraan");  
 int index = stringList.indexOf("Riyadh");  
 if(index == -1)  
 System.out.println("Riyadh not in list");  
 else{  
 stringList.add(index + 1, "Jubail"); // insert Jubail after Riyadh  
 stringList.addFirst("Najraan"); // insert Najraan at the beginning  
 }  
   
 System.out.println("List: " + stringList);  
   
 index = stringList.indexOf("Dhahraan");  
 if(index == -1)  
 System.out.println("Dhahraan is not in list");  
 else  
 stringList.set(index, "Abha"); // replace Dhahran with Abha  
  
  
 String removedString = stringList.** removeFirst**();  
 System.out.println("Removed Element: " + removedString);   
 stringList.addFirst("Taif");  
 System.out.println("Updated list: " + stringList);**

**// Add code that will insert “Tabouk” before “Dammam”, display error message if**

**// “Dammam” is not in list or if the list is empty. Your code must work for any**

**// two strings str1 and str2.**

**// Add code that will insert “Madinah” before the last node, your code must work**

**// for any non-empty list. Display an error message if the list is empty.**

**}  
}**

Output:

List: [Najraan, Madinah, Dammam, Riyadh, Jubail, Dhahraan]  
Accessed Element: Najraan  
Removed Element: Najraan  
Updated list: [Taif, Madinah, Dammam, Riyadh, Jubail, Abha]