Lớp LinkedList trong Java

Lớp LinkedList trong Java kế thừa lớp AbstractSequentialList và triển khai List Interface. Nó cung cấp một cấu trúc dữ liệu linked-list (dạng danh sách được liên kết).

Để hiểu sâu hơn các khái niệm được trình bày trong chương này, mời bạn tham khảo loạt bài: **Ví dụ về Collection trong Java**.

Lớp LinkedList hỗ trợ hai constructor. Constructor đầu tiên xây dựng một linked-list trống:

```
LinkedList( )
```

Constructor sau xây dựng một linked-list mà được khởi tạo với các phần tử trong collection c:

```
LinkedList(Collection c)
```

Ngoài các phương thức được kế thừa từ các lớp cha, lớp LinkedList trong Java định nghĩa các phương thức sau:

STT	Phương thức và Miêu tả
1	**void add(int index, Object element)** Chèn element đã xác định tại index đã cho. Ném một IndexOutOfBoundsException nếu index đã cho là ở bên ngoài dãy (index < 0 || index > size())
2	**boolean add(Object o)** Phụ thêm phần tử đã cho tới cuối của List này
3	**boolean addAll(Collection c)** Phụ thêm tất cả phần tử trong collection đã cho tới cuối của list này, theo thứ tự mà chúng được trả về bởi Iterator của collection đã cho. Ném một NullPointerException nếu collection đã cho là null
4	**boolean addAll(int index, Collection c)** Chèn tất cả phần tử trong collection đã cho vào trong List này, bắt đầu từ vị trí đã cho.

	Ném NullPointerException nếu collection đã cho là null
5	**void addFirst(Object o)** Chèn phần tử đã cho vào phần đầu của list này
6	**void addLast(Object o)** Phụ thêm phần tử đã cho vào phần cuối của list này
7	**void clear()** Gỡ bỏ tất cả phần tử từ list này
8	**Object clone()** Trả về một shallow copy của LinkedList này
9	**boolean contains(Object o)** Trả về true nếu list này chứa phần tử đã cho. Chính thức hơn, trả về true nếu và chỉ nếu list này chứa ít nhất một phần tử e để mà (o==null ? e==null : o.equals(e))
10	**Object get(int index)** Trả về phần tử tại vị trí đã cho. Ném IndexOutOfBoundsException nếu index ở bên ngoài dãy (index < 0 || index >= size())
11	**Object getFirst()** Trả về phần tử đầu tiên trong list này. Ném NoSuchElementException nếu list này là trống
12	**Object getLast()** Trả về phần tử cuối trong list này. Ném NoSuchElementException nếu list này là trống

13	**int indexOf(Object o)** Trả về index trong list này cho sự xuất hiện đầu tiên của phần tử đã cho, hoặc -1 nếu List này không chứa phần tử này
14	**int lastIndexOf(Object o)** Trả về index trong list này cho sự xuất hiện cuối của phần tử đã cho, hoặc -1 nếu List này không chứa phần tử này
15	**ListIterator listIterator(int index)** Trả về một list-iterator của phần tử trong list này (trong dãy chính xác), bắt đầu từ vị trí đã cho trong list. Ném IndexOutOfBoundsException nếu index đã cho ở bên ngoài dãy (index < 0 || index >= size())
16	**Object remove(int index)** Gỡ bỏ phần tử tại vị trí đã cho. Ném NoSuchElementException nếu list này là trống
17	**boolean remove(Object o)** Gỡ bỏ sự xuất hiện đầu tiên của phần tử đã cho. Ném NoSuchElementException nếu list này trống. Ném IndexOutOfBoundsException nếu index ở bên ngoài dãy (index < 0 || index >= size())
18	**Object removeFirst()** Gỡ bỏ và trả về phần tử đầu tiên từ list này. Ném NoSuchElementException nếu list là trống
19	**Object removeLast()** Gỡ bỏ và trả về phần tử cuối từ list này. Ném NoSuchElementException nếu list là trống
20	**Object set(int index, Object element)**

	Thay thế phần tử tại vị trí đã cho trong list này với phần tử đã cho. Ném IndexOutOfBoundsException nếu index đã cho ở ngoài dãy (index < 0 || index >= size())
21	**int size()** Trả về số phần tử trong list này
22	**Object[] toArray()** Trả về một mảng chứa tất cả phần tử trong list này trong đúng thứ tự. Ném NullPointerException nếu mảng đã xác định là null
23	**Object[] toArray(Object[] a)** Trả về một mảng chứa tất cả phần tử trong list này trong đúng thứ tự; kiểu runtime của mảng trả về là như của mảng đã xác định

Ví dụ

Chương trình sau minh họa các phương thức được hỗ trợ bởi lớp LinkedList trong Java:

```
import java.util.*;


public class LinkedListDemo {


   public static void main(String args[]) {
      // create a linked list
      LinkedList ll = new LinkedList();
      // add elements to the linked list
      ll.add("F");
      ll.add("B");
      ll.add("D");
      ll.add("E");
      ll.add("C");
      ll.addLast("Z");
```

```
        ll.addFirst("A");
        ll.add(1, "A2");
        System.out.println("Original contents of ll: " + ll);

        // remove elements from the linked list
        ll.remove("F");
        ll.remove(2);
        System.out.println("Contents of ll after deletion: "
         + ll);

        // remove first and last elements
        ll.removeFirst();
        ll.removeLast();
        System.out.println("ll after deleting first and last: "
         + ll);

        // get and set a value
        Object val = ll.get(2);
        ll.set(2, (String) val + " Changed");
        System.out.println("ll after change: " + ll);
    }
}
```

Nó sẽ cho kết quả sau:

```
Original contents of ll: [A, A2, F, B, D, E, C, Z]
Contents of ll after deletion: [A, A2, D, E, C, Z]
ll after deleting first and last: [A2, D, E, C]
ll after change: [A2, D, E Changed, C]
```