Fetching element from a LinkedList

Let's discuss the different methods to fetch an element from LinkedList.

Fetching the first element

We can use the `getFirst()` method to fetch the first element in the list. If the **LinkedList** is empty, then `NoSuchElementException` is thrown.

Fetching the last element

We can use the `getLast()` method to fetch the last element in the list. If the **LinkedList** is empty, then `NoSuchElementException` is thrown.

Fetching an element at a particular index

We can fetch an element at a particular index by using the `get(int index)` method. The index should be more than zero and less than the size of the **LinkedList**; otherwise, `IndexOutOfBoundsException` is thrown.

```
import java.util.LinkedList;

public class LinkedListDemo {

    public static void main(String args[]) {
        LinkedList<Integer> linkedList = new LinkedList<>();
        linkedList.add(1);
        linkedList.add(2);
        linkedList.add(3);
        linkedList.add(4);
        linkedList.add(5);
        linkedList.add(6);

        System.out.println(linkedList.getFirst()); //Fetching the first element.

        System.out.println(linkedList.getLast()); //Fetching the last element.

        System.out.println(linkedList.get(2)); //Fetching the element at second index.
    }
}
```

Run Save Reset

Removing element from a LinkedList

Let's discuss the different methods to remove an element from LinkedList.

Removing the first element

We can use the `removeFirst()` method to remove the first element in the list. If the **LinkedList** is empty, then `NoSuchElementException` is thrown.

Removing the last element

We can use the `removeLast()` method to remove the last element in the list. If the **LinkedList** is empty, then `NoSuchElementException` is thrown.

Removing an element at a particular index

We can remove an element at a particular index by using the `remove(int index)` method. The index should be more than zero and less than the size of the **LinkedList**; otherwise, `IndexOutOfBoundsException` is thrown.

Removing a particular element

We can use the `remove(Object o)` method to remove a particular element from the LinkedList. If there is more than one occurrence of a particular element, then the first occurrence is removed. If we want to remove the last occurrence of an element, the `removeLastOccurrence()` method can be used.

```
import java.util.LinkedList;

public class LinkedListDemo {

    public static void main(String args[]) {
        LinkedList<Integer> linkedList = new LinkedList<>();

        linkedList.add(1);
        linkedList.add(2);
        linkedList.add(3);
        linkedList.add(4);
        linkedList.add(2);
        linkedList.add(4);
        linkedList.add(5);
        System.out.println("LinkedList before removing any element " + linkedList);

        linkedList.remove(); //Removes the first element.
        System.out.println("LinkedList after removing first element " + linkedList);

        linkedList.removeLast(); //Removes the last element.
        System.out.println("LinkedList after removing last element " + linkedList);

        linkedList.remove(new Integer(2)); //Removes the first occurrence of 2.
        System.out.println("LinkedList after removing first occurrence of 2. " + linkedList);

        linkedList.removeLastOccurrence(new Integer(4)); //Removes the last occurrence of 4.
        System.out.println("LinkedList after removing the last occurrence of 4. " + linkedList);
    }
```

Run Save Reset

Sorting a LinkedList

To sort a LinkedList, we can use the `sort()` method of the **Collections** class as shown in the example below.

```
import java.util.Collections;
import java.util.LinkedList;

public class LinkedListDemo {

    public static void main(String args[]) {
        LinkedList<Integer> linkedList = new LinkedList<>();

        linkedList.add(20);
        linkedList.add(2);
        linkedList.add(12);
        linkedList.add(40);
        linkedList.add(76);
        linkedList.add(41);
        linkedList.add(53);

        Collections.sort(linkedList);

        System.out.println(linkedList);
    }
}
```

Run Save Reset