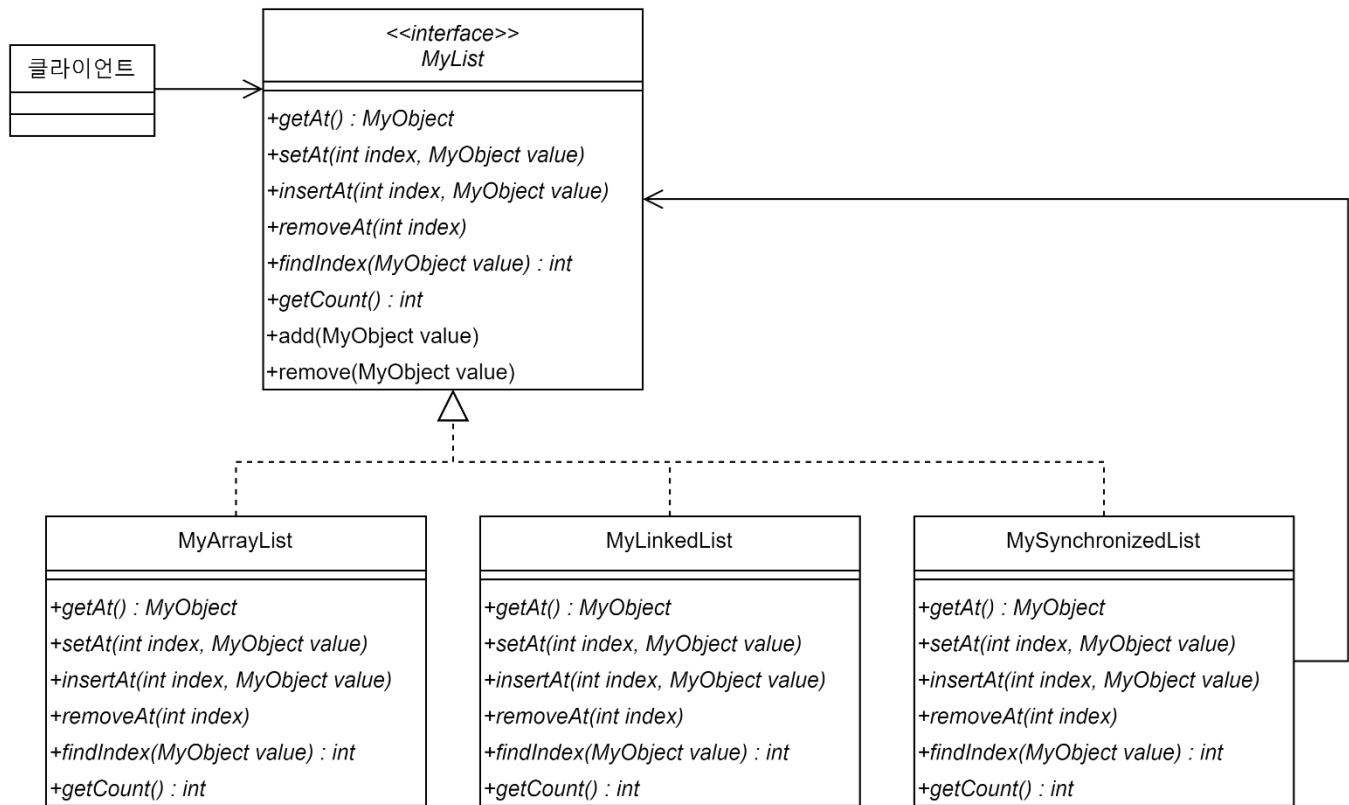
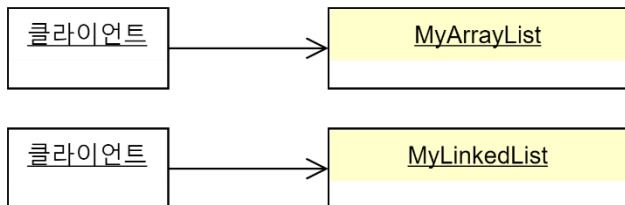


1) 개요

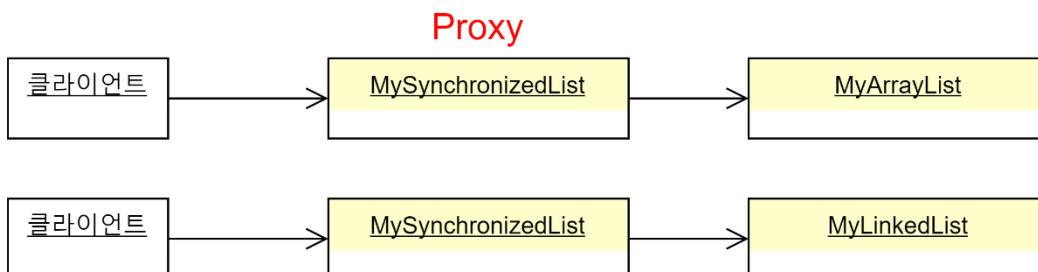


e1.Example



thread safe 하지 않다

e2.Example



thread safe 하다

2) MyList.java

```
1 package proxy.e2;
2
3 interface MyList {
4
5     MyObject getAt(int index);
6     void setAt(int index, MyObject value);
7     void insertAt(int index, MyObject value);
8     void removeAt(int index);
9     int findIndex(MyObject value);
10    int getCount();
11
12    default void add(MyObject value) {
13        insertAt(getCount(), value);
14    }
15
16    default void remove(MyObject value) {
17        int index = findIndex(value);
18        if (index == -1)
19            return;
20        removeAt(index);
21    }
22 }
```

3) MyArrayList.java

```
1 package proxy.e2;
2
3 import java.util.Arrays;
4
5 class MyArrayList implements MyList {
6     private MyObject[] data;
7     private int count;
8     private int size;
9
10    public MyArrayList() {
11        this(10);
12    }
13
14    public MyArrayList(int size) {
15        this.count = 0;
16        this.size = size;
17        this.data = new MyObject[size];
18    }
19
20    private void expand() {
21        size = data.length * 2;
22        data = Arrays.copyOf(data, size);
23    }
24
25    @Override
26    public MyObject getAt(int index) {
27        return data[index];
28    }
29
30    @Override
31    public void setAt(int index, MyObject value) {
32        data[index] = value;
33    }
34
35    @Override
36    public void insertAt(int index, MyObject value) {
37        if (count >= size)
38            expand();
39        for (int i = count - 1; i >= index; --i)
40            data[i + 1] = data[i];
41        data[index] = value;
42        count++;
43    }
44
45    @Override
46    public void removeAt(int index) {
47        for (int i = index; i < count - 1; ++i)
48            data[i] = data[i + 1];
49        count--;
50    }
51
52    @Override
53    public int findIndex(MyObject value) {
54        for (int i = 0; i < count; ++i)
55            if (value.equals(data[i]))
56                return i;
57        return -1;
58    }
59
60    @Override
61    public int getCount() {
62        return count;
63    }
64 }
```

4) MyLinkedList.java

```
1 package proxy.e2;
2
3 public class MyLinkedList implements MyList {
4     private static class Node {
5         private MyObject data;
6         private Node prev, next;
7
8         Node(MyObject data) {
9             this.data = data;
10        }
11    }
12
13    private Node dummy;
14    private int count;
15
16    public MyLinkedList() {
17        dummy = new Node(null);
18        dummy.prev = dummy.next = dummy;
19        count = 0;
20    }
21
22    private Node getNode(int index) {
23        Node node = dummy;
24        if (index < count / 2)
25            for (int i = 0; i <= index; ++i)
26                node = node.next;
27        else
28            for (int i = count-1; i >= index; --i)
29                node = node.prev;
30        return node;
31    }
32
33    @Override
34    public MyObject getAt(int index) {
35        return getNode(index).data;
36    }
37
38    @Override
39    public void setAt(int index, MyObject value) {
40        getNode(index).data = value;
41    }
42
43    @Override
44    public void insertAt(int index, MyObject value) {
45        Node newNode = new Node(value);
46        Node node = getNode(index);
47        newNode.next = node;
48        newNode.prev = node.prev;
49        node.prev.next = newNode;
50        node.prev = newNode;
51        ++count;
52    }
53
54    @Override
55    public void removeAt(int index) {
56        Node node = getNode(index);
57        node.prev.next = node.next;
58        node.next.prev = node.prev;
59        --count;
60    }
61
62    @Override
63    public int findIndex(MyObject value) {
64        int index;
65        Node node = dummy.next;
66        for (index = 0; index < count; ++index) {
67            if (value.equals(node.data)) break;
68            node = node.next;
```

```
69     }
70     return index;
71 }
72
73 @Override
74 public int getCount() {
75     return count;
76 }
77 }
```

5) MySynchronizedList.java

```
1 package proxy.e2;
2
3 public class MySynchronizedList implements MyList {
4     MyList list;
5
6     MySynchronizedList(MyList list) {
7         this.list = list;
8     }
9
10    @Override
11    public synchronized int getCount() {
12        return list.getCount();
13    }
14
15    @Override
16    public synchronized MyObject getAt(int index) {
17        return list.getAt(index);
18    }
19
20    @Override
21    public synchronized void setAt(int index, MyObject data) {
22        list.setAt(index, data);
23    }
24
25    @Override
26    public synchronized void insertAt(int index, MyObject data) {
27        list.insertAt(index, data);
28    }
29
30    @Override
31    public synchronized void removeAt(int index) {
32        list.removeAt(index);
33    }
34
35    @Override
36    public synchronized int findIndex(MyObject data) {
37        return list.findIndex(data);
38    }
39 }
```

6) Example2.java

```
1 package proxy.e2;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6 public class Example2 {
7
8     static void work(MyList list) {
9         for (int i=0; i < 1000; ++i) {
10             list.insertAt(0, new MyInt(999));
11             list.removeAt(0);
12         }
13     }
14
15     static void add(MyList list, int count) {
16         for (int i = 0; i < count; ++i)
17             list.add(new MyInt(i));
18     }
19
20     static void print(MyList list) {
21         System.out.printf("Count: %d\n", list.getCount());
22         for (int i = 0; i < list.getCount(); ++i)
23             System.out.printf("%s ", list.getAt(i));
24         System.out.println();
25     }
26
27     static void doSomething(MyList list) throws Exception {
28         List<Thread> threads = new ArrayList<>();
29         add(list, 100);
30         for (int i = 0; i < 100; ++i) {
31             Thread t = new Thread(() -> work(list));
32             t.start();
33             threads.add(t);
34         }
35         for (Thread t: threads)
36             t.join();
37         print(list);
38     }
39
40     public static void main(String[] args) throws Exception {
41         doSomething(new MySynchronizedList(new MyArrayList()));
42         doSomething(new MySynchronizedList(new MyLinkedList()));
43     }
44 }
```

proxy 패턴 적용하여 multi thread 충돌 해결함

proxy.e1.Example1B.java 소스코드와 비교하여, 수정된 부분을 확인하자.