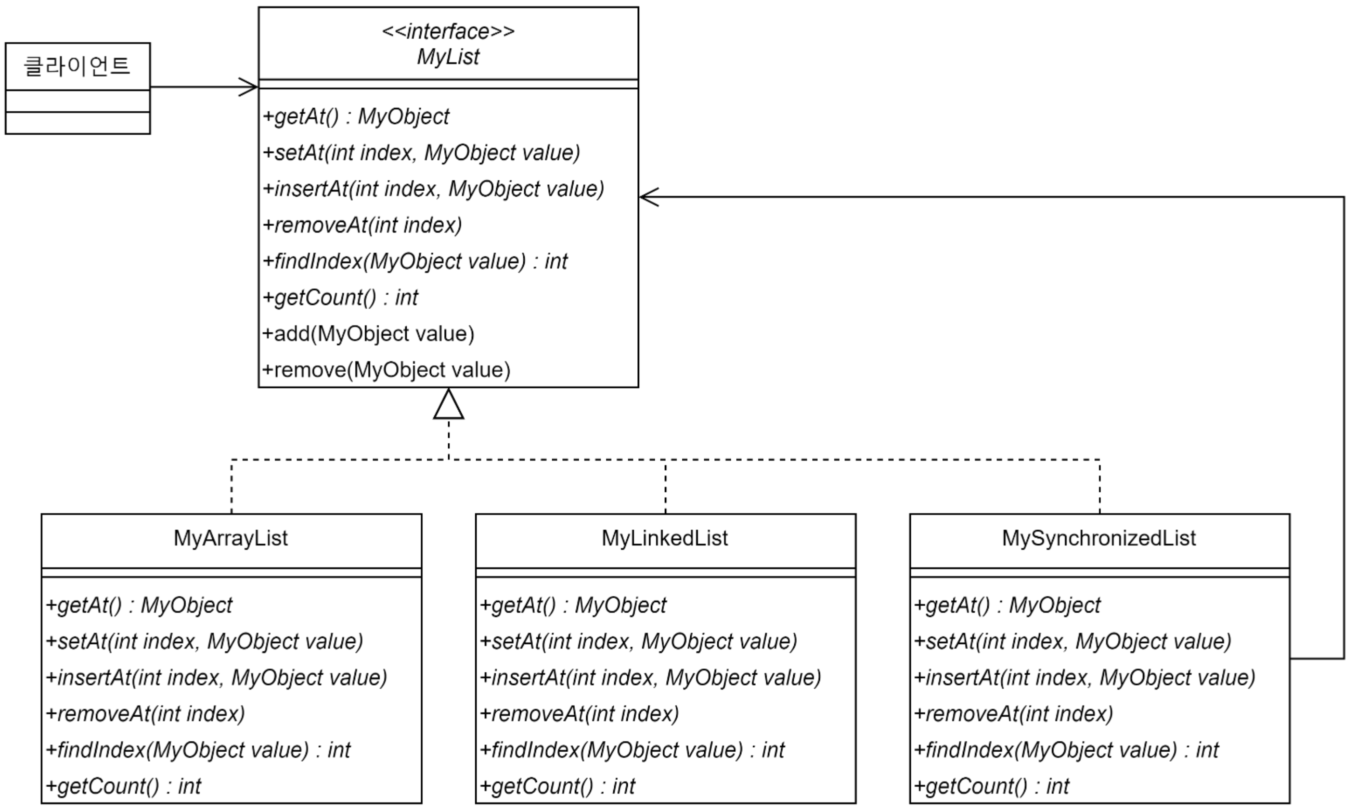
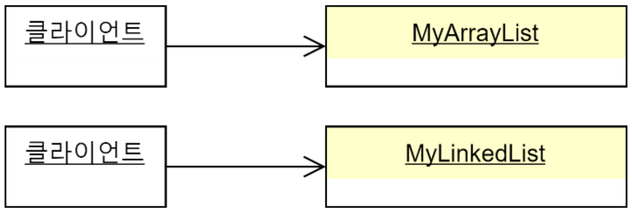
1) 개요

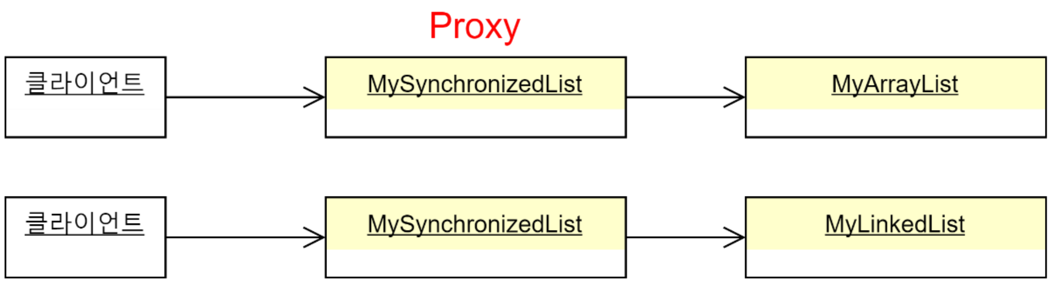


e1.Example



thread safe 하지 않다

e2.Example



thread safe 하다

2) MyList.java

```java
package proxy.e2;

interface MyList {

    MyObject getAt(int index);
    void setAt(int index, MyObject value);
    void insertAt(int index, MyObject value);
    void removeAt(int index);
    int findIndex(MyObject value);
    int getCount();

    default void add(MyObject value) {
        insertAt(getCount(), value);
    }

    default void remove(MyObject value) {
        int index = findIndex(value);
        if (index == -1)
            return;
        removeAt(index);
    }
}
```

3) MyArrayList.java

```java
package proxy.e2;

import java.util.Arrays;

class MyArrayList implements MyList {
    private MyObject[] data;
    private int count;
    private int size;

    public MyArrayList() {
        this(10);
    }

    public MyArrayList(int size) {
        this.count = 0;
        this.size = size;
        this.data = new MyObject[size];
    }

    private void expand() {
        size = data.length * 2;
        data = Arrays.copyOf(data, size);
    }

    @Override
    public MyObject getAt(int index) {
        return data[index];
    }

    @Override
    public void setAt(int index, MyObject value) {
        data[index] = value;
    }

    @Override
    public void insertAt(int index, MyObject value) {
        if (count >= size)
            expand();
        for (int i = count - 1; i >= index; --i)
            data[i + 1] = data[i];
        data[index] = value;
        count++;
    }

    @Override
    public void removeAt(int index) {
        for (int i = index; i < count - 1; ++i)
            data[i] = data[i + 1];
        count--;
    }

    @Override
    public int findIndex(MyObject value) {
        for (int i = 0; i < count; ++i)
            if (value.equals(data[i]))
                return i;
        return -1;
    }

    @Override
    public int getCount() {
        return count;
    }
}
```

4) MyLinkedList.java

```
package proxy.e2;

public class MyLinkedList implements MyList {
    private static class Node {
        private MyObject data;
        private Node prev, next;

        Node(MyObject data) {
            this.data = data;
        }
    }

    private Node dummy;
    private int count;

    public MyLinkedList() {
        dummy = new Node(null);
        dummy.prev = dummy.next = dummy;
        count = 0;
    }

    private Node getNode(int index) {
        Node node = dummy;
        if (index < count / 2)
            for (int i = 0; i <= index; ++i)
                node = node.next;
        else
            for (int i = count-1; i >= index; --i)
                node = node.prev;
        return node;
    }

    @Override
    public MyObject getAt(int index) {
        return getNode(index).data;
    }

    @Override
    public void setAt(int index, MyObject value) {
        getNode(index).data = value;
    }

    @Override
    public void insertAt(int index, MyObject value) {
        Node newNode = new Node(value);
        Node node = getNode(index);
        newNode.next = node;
        newNode.prev = node.prev;
        node.prev.next = newNode;
        node.prev = newNode;
        ++count;
    }

    @Override
    public void removeAt(int index) {
        Node node = getNode(index);
        node.prev.next = node.next;
        node.next.prev = node.prev;
        --count;
    }

    @Override
    public int findIndex(MyObject value) {
        int index;
        Node node = dummy.next;
        for (index = 0; index < count; ++index) {
            if (value.equals(node.data)) break;
            node = node.next;
```

```java
        }
        return index;
    }

    @Override
    public int getCount() {
        return count;
    }
}
```

5) MySynchronizedList.java

```java
package proxy.e2;

public class MySynchronizedList implements MyList {
    MyList list;

    MySynchronizedList(MyList list) {
        this.list = list;
    }

    @Override
    public synchronized int getCount() {
        return list.getCount();
    }

    @Override
    public synchronized MyObject getAt(int index) {
        return list.getAt(index);
    }

    @Override
    public synchronized void setAt(int index, MyObject data) {
        list.setAt(index, data);
    }

    @Override
    public synchronized void insertAt(int index, MyObject data) {
        list.insertAt(index, data);
    }

    @Override
    public synchronized void removeAt(int index) {
        list.removeAt(index);
    }

    @Override
    public synchronized int findIndex(MyObject data) {
        return list.findIndex(data);
    }
}
```

6) Example2.java

```java
package proxy.e2;

import java.util.ArrayList;
import java.util.List;

public class Example2 {

    static void work(MyList list) {
        for (int i=0; i < 1000; ++i) {
            list.insertAt(0, new MyInt(999));
            list.removeAt(0);
        }
    }

    static void add(MyList list, int count) {
        for (int i = 0; i < count; ++i)
            list.add(new MyInt(i));
    }

    static void print(MyList list) {
        System.out.printf("Count: %d\n", list.getCount());
        for (int i = 0; i < list.getCount(); ++i)
            System.out.printf("%s ", list.getAt(i));
        System.out.println();
    }

    static void doSomething(MyList list) throws Exception {
        List<Thread> threads = new ArrayList<>();
        add(list, 100);
        for (int i = 0; i < 100; ++i) {
            Thread t = new Thread(() -> work(list));
            t.start();
            threads.add(t);
        }
        for (Thread t: threads)
            t.join();
        print(list);
    }

    public static void main(String[] args) throws Exception {
        doSomething(new MySynchronizedList(new MyArrayList()));
        doSomething(new MySynchronizedList(new MyLinkedList()));
    }
}
```

proxy 패턴 적용하여 multi thead 충돌 해결함

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