Fail-Safe Iterator

In previous post we saw Fail-Fast Iterator and [ConcurrentModificationException](http://data-structure-learning.blogspot.com/2015/05/concurrentmodificationexception.html). In Fail-Fast Iterator if Collection is modified structurally then ConcurrentModificationException is thrown.

In this post we will see what is Fail-Safe iterator.

Fail-Safe iterator iterates over the snapshot of the array. Operations like add and set are implemented on the fresh copy of underlying array. This is costly operations but is more efficient if we don’t want to synchronize traversals. remove(), set() and add() method of Iterator will throw UnsupportedOperationException.

Fail-Safe iterator does not guarantee that data currently been read is data currently in data structure.

**import** java.util.Arrays;

**import** java.util.Iterator;

**import** java.util.concurrent.CopyOnWriteArrayList;

**public** **class** Driver {

**public** **static** **void** main(String[] args) {

CopyOnWriteArrayList<Integer> l1 = **new** CopyOnWriteArrayList<Integer>(Arrays.*asList*(1, 2, 3, 4));

Iterator<Integer> iter=l1.iterator();

**while**(iter.hasNext()){

iter.next();

l1.set(1,44);

l1.remove(**new** Integer(3));

}

System.***out***.println(l1);

}

}

Output:

[1, 44, 4]