Collections class reverse method.

As the name suggests reverse method reverses the elements in the List.

Below is the code taken from JDK and I have commented it so you will understand it properly.

**private** **static** **final** **int** ***REVERSE\_THRESHOLD*** = 18;

/\*\*

\* reverse method reverses the elements of the

\* List<?>. Wildcard <?> means <? extends Object>.

\* \*/

@SuppressWarnings({ "rawtypes", "unchecked" })

**public** **static** **void** reverse(List<?> list) {

//take the size of list.

**int** size = list.size();

/\*\*

\* If the size of List is less then threshold or

\* List is instance of RandomAccess then reverse

\* using index.

\* It uses swap method that will 2 swap elements

\* from 2 ends at a time.

\* \*/

**if** (**size < *REVERSE\_THRESHOLD* || list instanceof RandomAccess**) {

**for (int i = 0, mid = size >> 1, j = size - 1; i < mid; i++, j--)**

***swap***(list, i, j);

} **else** {

// instead of using a raw type here, it's possible to capture

// the wildcard but it will require a call to a supplementary

// private method

ListIterator fwd = list.listIterator();

/\*\*

\* list.listIterator(size) means listIterator will

\* start from the end of List.

\* \*/

ListIterator rev = list.listIterator(size);

**for** (**int** i = 0, mid = list.size() >> 1; i < mid; i++) {

Object tmp = fwd.next();

fwd.set(rev.previous());

rev.set(tmp);

}

}

}

/\*\*

\* swap method swaps the two elements in List given

\* index of those elements.

\* \*/

@SuppressWarnings({ "rawtypes", "unchecked" })

**public** **static** **void** swap(List<?> list, **int** i, **int** j) {

// instead of using a raw type here, it's possible to capture

// the wildcard but it will require a call to a supplementary

// private method

**final** List l = list;

l.set(i, l.set(j, l.get(i)));

}

RandomAccess interface plays a very important role here. RandomAccess is a marker interface. Marker interfaces does not have any methods or variables declared in it. It is used for tagging purpose. Implementing this interface denotes specific behavior on part of implementing class. Currently I know about 4 classes ArrayList, CopyOnWriteArrayList, Stack and Vector which implements this interface.

For the above method if there is any class which implements RandomAccess interface then it will enter the if condition which is in **bold font**.

Below is the program that uses reverse method.

**import** java.util.Arrays;

**import** java.util.Collections;

**import** java.util.List;

**public** **class** CollectionsReverse {

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

CollectionsReverse rev = **new** CollectionsReverse();

List<Integer> intList = Arrays.*asList*(1, 2, 3, 4);

rev.reverseDemo(intList);

List<String> stringList = Arrays.*asList*("s", "w", "q");

rev.reverseDemo(stringList);

}

**public** <E> **void** reverseDemo(**final** List<E> list) {

System.***out***.println("Before reverse: " + list);

**Collections.*reverse*(list);**

System.***out***.println("After reverse: " + list);

}

}

Output

Before reverse: [1, 2, 3, 4]

After reverse: [4, 3, 2, 1]

Before reverse: [s, w, q]

After reverse: [q, w, s]

That’s all on reverse() method of Collections class.

In next posts we will saw other methods for Collections class.