ArrayList class removeIf method

This is 17th post in series of ArrayList class. Previously we have seen [ArrayList introduction](http://data-structure-learning.blogspot.com/2015/08/arraylist-class-introduction-and-how-it.html), ArrayList class [constructors](http://data-structure-learning.blogspot.com/2015/08/arraylist-class-constructors.html), [add](http://data-structure-learning.blogspot.com/2015/09/arraylist-class-add-methods.html)() method, [addAll](http://data-structure-learning.blogspot.com/2015/09/arraylist-class-addall-methods.html)() method, [clear](http://data-structure-learning.blogspot.com/2015/09/arraylist-class-clear-method.html)() method, [indexOf](http://data-structure-learning.blogspot.com/2015/09/arraylist-class-indexof-method.html)() method, [contains](http://data-structure-learning.blogspot.com/2015/09/arraylist-class-contains-method.html)() method, forEach() method, get(), isEmpty(), iterator(), lastIndexOf() method, listIterator(), remove(int index), remove(Object o) and removeAll(Collection<?> c) method.

In this post we will see removeIf(Predicate<? super E> filter) method. removeIf() method is new method added in Java 8. To understand this method you need to know about [lambda operator](http://data-structure-learning.blogspot.com/2015/06/using-functional-interface.html) and [Predicate](http://data-structure-learning.blogspot.com/p/1.html) Interface. You also need to know what [Functional Interface](http://data-structure-learning.blogspot.com/2015/06/functional-interfaces-java-8.html) is. I have attached links so it would be useful for you to read about it.

If you are reading this line that I assume that you have gone through above links.

Below is the program that uses Predicate as anonymous inner class and as Lambda Operator.

**package** org.example.collections.list.arraylist;

**import** java.util.ArrayList;

**import** java.util.List;

**import** java.util.function.Predicate;

**public** **class** ArrayListRemoveIf {

**public** List<String> names() {

List<String> names = **new** ArrayList<String>();

names.add("Ned");

names.add("Catelyn");

names.add("Rob");

names.add("Arya");

names.add("Sansa");

**return** names;

}

/\*\*

\* removeIf() method using Anonymous Inner Class.

\* \*/

**public** **void** removeIfDemoAnonymousInnerClass(List<String> names) {

System.***out***.println("names before removeIf " + names);

names.removeIf(**new** Predicate<String>() {

@Override

**public** **boolean** test(String name) {

**return** name.length() > 3;

}

});

System.***out***.println("names after removeIf " + names);

}

/\*\*

\* removeIf() method using Lambda Operator.

\* \*/

**public** **void** removeIfDemoLambdaOperator(List<String> names){

System.***out***.println("names before removeIf " + names);

names.removeIf(name -> name.length() > 3);

System.***out***.println("names after removeIf " + names);

}

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

ArrayListRemoveIf removeIfDemo = **new** ArrayListRemoveIf();

System.***out***.println("Call by Anonymous Inner Class");

List<String> names = removeIfDemo.names();

removeIfDemo.removeIfDemoAnonymousInnerClass(names);

System.***out***.println("\nCall by Lambda Operator");

names = removeIfDemo.names();

removeIfDemo.removeIfDemoLambdaOperator(names);

}

}

Output

Call by Anonymous Inner Class

names before removeIf [Ned, Catelyn, Rob, Arya, Sansa]

names after removeIf [Ned, Rob]

Call by Lambda Operator

names before removeIf [Ned, Catelyn, Rob, Arya, Sansa]

names after removeIf [Ned, Rob]

That’s all on removeIf() method . In next post we will see method replaceAll(UnaryOperator<E> operator) method. It is new method added in Java 8.