How to convert List<E> to Set<E>?

At times needs arise to convert one collection to another. Same is for this question. Conversion to list to set is one such thing to do.

Now remember when we learned about root [Java Collections Framework](http://data-structure-learning.blogspot.com/2015/05/java-collections-part-1.html) interface called [Collection<E>](http://data-structure-learning.blogspot.com/2015/05/java-collections-part-4collection.html) interface, we learnt about addAll(Collection<? extends E> c).

**boolean** addAll(Collection<? **extends** E> c); - This method is used for bulk operation. It takes another collection of type <? extends E> i.e. any type that extends E. We use <? extends E> if we need to get elements from Collection. So here we use this bounded wildcard and get all elements from collection c specified in parameter and insert into another collection. It returns true if collection is changed because of call to this method.

Below is code for using addAll(Collection<? extends E> c) method

Set<String> set2=**new** HashSet<String>();

set2.addAll(list);

We can do this by iterating list and inserting each element into Set using add(E e) method.

Set<String> set3=**new** HashSet<>(list);

**for**(String str:list){

set3.add(str);

}

Now as we are using class HashSet<E> we can use its parameterized constructor

**public** HashSet(Collection<? **extends** E> c)

Now we just put list as parameter

Set<String> set1=**new** HashSet<>(list);

Below is the code for all 3 ways:

**package** org.collections;

**import** java.util.ArrayList;

**import** java.util.HashSet;

**import** java.util.List;

**import** java.util.Set;

**public** **class** ListToSet {

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

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

list.add("Adam");

list.add("Cain");

list.add("Eve");

list.add("Adam");

list.add("Sam");

list.add("Dean");

Set<String> set1 = **new** HashSet<>(list);

System.***out***.println("By parameterized constructor "+set1);

Set<String> set2 = **new** HashSet<>();

set2.addAll(list);

System.***out***.println("Using addAll(..) method "+set1);

Set<String> set3 = **new** HashSet<>(list);

**for** (String str : list) {

set3.add(str);

}

System.***out***.println("Using add(e) method "+set1);

}

}

Output is as follows;

By parameterized constructor [Adam, Eve, Cain, Sam, Dean]

Using addAll(..) method [Adam, Eve, Cain, Sam, Dean]

Using add(e) method [Adam, Eve, Cain, Sam, Dean]

As you see the size of [List<E>](http://data-structure-learning.blogspot.com/2015/05/java-collections-part-5list-interface.html) and Set<E> is different. The reason is List<E> contains duplicates but Set<E> does not accept duplicates.

Set<E> is unordered collection.

The better ways to solve this question is to **either use parameterized constructor or use addAll(Collection<? extends E> c) method**.