How to add value to all types of Collections

http://javatutorialcorner.com/java/how-to-add-value-to-all-types-of-collections/

[COLLECTIONS](http://javatutorialcorner.com/category/collections/), [JAVA](http://javatutorialcorner.com/category/java/), [JAVA EXAMPLES](http://javatutorialcorner.com/category/java-examples/), [LIST](http://javatutorialcorner.com/category/list/), [MAP](http://javatutorialcorner.com/category/map/), [SET](http://javatutorialcorner.com/category/set/) *May 31, 2015, by*[*Annamalai Thangaraj*](http://javatutorialcorner.com/author/annamalai-thangaraj/) [1 Comment](http://javatutorialcorner.com/java/how-to-add-value-to-all-types-of-collections/#comments) 151

Following example shows how to add values in all types collections in Java.

**Sample Program**  
**AddValueToCollection.java**

package com.javatutorialcorner.collection;

import java.util.ArrayList;

import java.util.Collection;

import java.util.HashMap;

import java.util.HashSet;

import java.util.Iterator;

import java.util.LinkedHashMap;

import java.util.LinkedHashSet;

import java.util.LinkedList;

import java.util.List;

import java.util.Map;

import java.util.Set;

import java.util.SortedMap;

import java.util.SortedSet;

import java.util.TreeMap;

import java.util.TreeSet;

public class AddValueToCollection {

public static void main(String[] args) {

System.out.println("\*\*\*\*\*\*\*\*\*\* LinkedList \*\*\*\*\*\*\*\*\*\*");

List lnkLst = new LinkedList();

lnkLst.add("Java");

lnkLst.add("Spring");

lnkLst.add("Spring MVC");

lnkLst.add("Spring Security");

printCollection(lnkLst);

System.out.println("\*\*\*\*\*\*\*\*\*\* Array List \*\*\*\*\*\*\*\*\*\*");

List aryLst = new ArrayList();

aryLst.add("Spring Social");

aryLst.add("Spring Integration");

aryLst.add("Spring Batch");

aryLst.add("Hibernate");

printCollection(aryLst);

System.out.println("\*\*\*\*\*\*\*\*\*\* HashSet \*\*\*\*\*\*\*\*\*\*");

Set hashSet = new HashSet();

hashSet.add("RESTFul Webservice");

hashSet.add("SOAP Webservice");

hashSet.add("JSON");

hashSet.add("XML Parsing");

printCollection(hashSet);

System.out.println("\*\*\*\*\*\*\*\*\*\* TreeSet \*\*\*\*\*\*\*\*\*\*");

SortedSet treeSet = new TreeSet();

treeSet.add("Log4j");

treeSet.add("Java 7");

treeSet.add("Java Mail API");

treeSet.add("Jav Best practices");

printCollection(treeSet);

System.out.println("\*\*\*\*\*\*\*\*\*\* LinkedHashSet \*\*\*\*\*\*\*\*\*\*");

LinkedHashSet lnkHashset = new LinkedHashSet();

lnkHashset.add("AngularJS");

lnkHashset.add("JUnit");

lnkHashset.add("EasyMock");

lnkHashset.add("Big Data");

printCollection(lnkHashset);

System.out.println("\*\*\*\*\*\*\*\*\*\* HashMap \*\*\*\*\*\*\*\*\*\*");

Map map1 = new HashMap();

map1.put("xml1", "DOM Parser");

map1.put("xml2", "SAX Parser");

map1.put("xml3", "JDOM");

map1.put("xml4", "DOM4J");

map1.put("xml5", "XPATH");

map1.put("xml6", "JAXB");

printCollection(map1.keySet());

printCollection(map1.values());

System.out.println("\*\*\*\*\*\*\*\*\*\* TreeMap \*\*\*\*\*\*\*\*\*\*");

SortedMap map2 = new TreeMap();

map2.put("JSON1", "JSONSimple");

map2.put("JSON2", "Jackson");

map2.put("JSON3", "GSON");

printCollection(map2.keySet());

printCollection(map2.values());

System.out.println("\*\*\*\*\*\*\*\*\*\* LinkedHashMap \*\*\*\*\*\*\*\*\*\*");

LinkedHashMap map3 = new LinkedHashMap();

map3.put("db1", "MySQL");

map3.put("db2", "PostgreSQL");

map3.put("db3", "Oracle");

map3.put("db4", "MS SQL Server");

printCollection(map3.keySet());

printCollection(map3.values());

}

private static void printCollection(Collection collection) {

Iterator itr = collection.iterator();

while (itr.hasNext()) {

String str = (String) itr.next();

System.out.print(str + " ");

}

System.out.println();

}

}

**Result**  
The above code will produce the following output.

*\*\*\*\*\*\*\*\*\*\* LinkedList \*\*\*\*\*\*\*\*\*\*  
Java Spring Spring MVC Spring Security  
\*\*\*\*\*\*\*\*\*\* Array List \*\*\*\*\*\*\*\*\*\*  
Spring Social Spring Integration Spring Batch Hibernate  
\*\*\*\*\*\*\*\*\*\* HashSet \*\*\*\*\*\*\*\*\*\*  
JSON SOAP Webservice RESTFul Webservice XML Parsing  
\*\*\*\*\*\*\*\*\*\* TreeSet \*\*\*\*\*\*\*\*\*\*  
Jav Best practices Java 7 Java Mail API Log4j  
\*\*\*\*\*\*\*\*\*\* LinkedHashSet \*\*\*\*\*\*\*\*\*\*  
AngularJS JUnit EasyMock Big Data  
\*\*\*\*\*\*\*\*\*\* HashMap \*\*\*\*\*\*\*\*\*\*  
xml4 xml3 xml6 xml5 xml2 xml1  
DOM4J JDOM JAXB XPATH SAX Parser DOM Parser  
\*\*\*\*\*\*\*\*\*\* TreeMap \*\*\*\*\*\*\*\*\*\*  
JSON1 JSON2 JSON3  
JSONSimple Jackson GSON  
\*\*\*\*\*\*\*\*\*\* LinkedHashMap \*\*\*\*\*\*\*\*\*\*  
db1 db2 db3 db4  
MySQL PostgreSQL Oracle MS SQL Server*

Collection Shuffle

package com.javatutorialcorner.collection;

import java.util.Arrays;

import java.util.Collections;

import java.util.List;

public class ShuffleCollection {

public static void main(String[] args) {

String[] nameArray = new String[] { "Spring", "Struts", "Hibernate",

"Webservice", "SOAP Webservice", "RESTful Webservice" };

List<String> nameList = Arrays.asList(nameArray);

System.out.println("\*\*\*\*\*\*\*\*\*\*\* Before Shuffle \*\*\*\*\*\*\*\*\*\*\*\*");

for (String name : nameList) {

System.out.println(name);

}

Collections.shuffle(nameList);

System.out.println("\*\*\*\*\*\*\*\*\*\*\* After Shuffle \*\*\*\*\*\*\*\*\*\*\*\*");

for (String name : nameList) {

System.out.println(name);

}

}

}

**Result**  
The above code will produce the following output.

*\*\*\*\*\*\*\*\*\*\*\* Before Shuffle \*\*\*\*\*\*\*\*\*\*\*\*  
Spring  
Struts  
Hibernate  
Webservice  
SOAP Webservice  
RESTful Webservice  
\*\*\*\*\*\*\*\*\*\*\* After Shuffle \*\*\*\*\*\*\*\*\*\*\*\*  
RESTful Webservice  
Spring  
SOAP Webservice  
Struts  
Webservice  
Hibernate*

#### How to get the size of Collection

[COLLECTIONS](http://javatutorialcorner.com/category/collections/), [JAVA](http://javatutorialcorner.com/category/java/), [JAVA EXAMPLES](http://javatutorialcorner.com/category/java-examples/), [LIST](http://javatutorialcorner.com/category/list/), [MAP](http://javatutorialcorner.com/category/map/), [SET](http://javatutorialcorner.com/category/set/) *May 31, 2015, by*[*Annamalai Thangaraj*](http://javatutorialcorner.com/author/annamalai-thangaraj/) [0 Comment](http://javatutorialcorner.com/java/how-to-get-the-size-of-collection/#respond) 33

Following example shows how to get the size of collection using *collection.size()*method in Java.

**int java.util.List.size()**  
**size**  
*int size()*  
Returns the number of elements in this list. If this list contains more than Integer.MAX\_VALUE elements, returns Integer.MAX\_VALUE. Specified by:  
    *size in interface Collection<E>*  
**Returns:**  
    the number of elements in this list

**int java.util.Set.size()  
size**  
*int size()*  
Returns the number of elements in this set (its cardinality). If this set contains more than Integer.MAX\_VALUE elements, returns Integer.MAX\_VALUE.Specified by:*size in interface Collection<E>*  
**Returns:**  
    the number of elements in this set (its cardinality)

**int java.util.Map.size()  
size**  
*int size()*  
Returns the number of key-value mappings in this map. If the map contains more than Integer.MAX\_VALUE elements, returns Integer.MAX\_VALUE.   
**Returns:**  
    the number of key-value mappings in this map

**Sample Program**  
**SizeOfCollection.java**

package com.javatutorialcorner.collection;  
  
import java.util.Arrays;  
import java.util.HashMap;  
import java.util.HashSet;  
import java.util.List;  
import java.util.Map;  
import java.util.Set;  
  
public class SizeOfCollection {  
  
 public static void main(String[] args) {  
 String[] nameArray = new String[] { "Spring", "Struts", "Hibernate",  
 "Webservice", "SOAP Webservice", "RESTful Webservice" };  
 List<String> nameList = Arrays.asList(nameArray);  
 System.out.println("List size " + nameList.size());  
  
 Set<String> set = new HashSet<String>(nameList);  
 System.out.println("Set size " + set.size());  
  
 Map map = new HashMap();  
 System.out.println("Map size " + map.size());  
  
 }  
  
}

**Result**  
The above code will produce the following output.

*List size 6  
Set size 6  
Map size 0*

#### How to find min and max value from Collection

[COLLECTIONS](http://javatutorialcorner.com/category/collections/), [JAVA](http://javatutorialcorner.com/category/java/), [JAVA EXAMPLES](http://javatutorialcorner.com/category/java-examples/), [LIST](http://javatutorialcorner.com/category/list/), [SET](http://javatutorialcorner.com/category/set/) *May 31, 2015, by*[*Annamalai Thangaraj*](http://javatutorialcorner.com/author/annamalai-thangaraj/) [1 Comment](http://javatutorialcorner.com/java/how-to-find-min-and-max-value-from-collection/#comments) 52

Following example shows how to find minimum and maximum of Collection using *Collections.min(Collection<T> collection)*and*Collections.max(Collection<T> collection)* method in Java.

**<String> String java.util.Collections.min(Collection<? extends String> coll)  
min**  
*public static <T extends Object & Comparable<? super T>> T min(Collection<? extends T> coll)*  
Returns the minimum element of the given collection, according to the natural ordering of its elements. All elements in the collection must implement the Comparable interface. Furthermore, all elements in the collection must be mutually comparable (that is, e1.compareTo(e2) must not throw a ClassCastException for any elements e1 and e2 in the collection). This method iterates over the entire collection, hence it requires time proportional to the size of the collection.   
**Parameters:**  
    coll – the collection whose minimum element is to be determined.  
**Returns:**  
    the minimum element of the given collection, according to the natural ordering of its elements.  
**Throws:**  
*ClassCastException* – if the collection contains elements that are not mutually comparable (for example, strings and integers).  
*NoSuchElementException* – if the collection is empty.

**<String> String java.util.Collections.max(Collection<? extends String> coll, Comparator<? super String> comp)  
max**  
*public static <T> T max(Collection<? extends T> coll, Comparator<? super T> comp)*  
Returns the maximum element of the given collection, according to the order induced by the specified comparator. All elements in the collection must be mutually comparable by the specified comparator (that is, comp.compare(e1, e2) must not throw a ClassCastException for any elements e1 and e2 in the collection). This method iterates over the entire collection, hence it requires time proportional to the size of the collection.   
**Parameters:**  
    coll – the collection whose maximum element is to be determined.  
    comp – the comparator with which to determine the maximum element. A null value indicates that the elements’ natural ordering should be used.  
**Returns:**  
    the maximum element of the given collection, according to the specified comparator.  
**Throws:**  
*ClassCastException* – if the collection contains elements that are not mutually comparable using the specified comparator.  
*NoSuchElementException* – if the collection is empty.

**Sample Program**  
**MinMaxCollection.java**

package com.javatutorialcorner.collection;  
  
import java.util.Arrays;  
import java.util.Collections;  
import java.util.List;  
import java.util.Set;  
import java.util.TreeSet;  
  
public class MinMaxCollection {  
  
 public static void main(String[] args) {  
 String[] nameArray = new String[] { "Spring", "Struts", "Hibernate",  
 "Webservice", "SOAP Webservice", "RESTful Webservice" };  
 Set<String> set = new TreeSet<String>();  
  
 for (String name : nameArray) {  
 set.add(name);  
 }  
  
 System.out.println("\*\*\*\*\*\*\*\* Min and Max of Set \*\*\*\*\*\*\*\*");  
  
 System.out.println(Collections.min(set));  
 System.out.println(Collections.min(set, String.CASE\_INSENSITIVE\_ORDER));  
  
 System.out.println(Collections.max(set));  
 System.out.println(Collections.max(set, String.CASE\_INSENSITIVE\_ORDER));  
  
 System.out.println("\*\*\*\*\*\*\*\* Min and Max of List \*\*\*\*\*\*\*\*");  
 List<String> nameList = Arrays.asList(nameArray);  
 System.out.println(Collections.min(nameList));  
 System.out.println(Collections.min(nameList,  
 String.CASE\_INSENSITIVE\_ORDER));  
  
 System.out.println(Collections.max(nameList));  
 System.out.println(Collections.max(nameList,  
 String.CASE\_INSENSITIVE\_ORDER));  
 }  
  
}

**Result**  
The above code will produce the following output.

*\*\*\*\*\*\*\*\* Min and Max of Set \*\*\*\*\*\*\*\*  
Hibernate  
Hibernate  
Webservice  
Webservice  
\*\*\*\*\*\*\*\* Min and Max of List \*\*\*\*\*\*\*\*  
Hibernate  
Hibernate  
Webservice  
Webservice*

#### How to replace the element from List

[COLLECTIONS](http://javatutorialcorner.com/category/collections/), [JAVA](http://javatutorialcorner.com/category/java/), [JAVA EXAMPLES](http://javatutorialcorner.com/category/java-examples/), [LIST](http://javatutorialcorner.com/category/list/) *May 31, 2015, by*[*Annamalai Thangaraj*](http://javatutorialcorner.com/author/annamalai-thangaraj/) [1 Comment](http://javatutorialcorner.com/java/how-to-replace-the-element-from-list/#comments) 62

Following example shows how to replace all the occurrence of element with another element using*Collections.replaceAll(list, oldVal, newVal)* method in Java.

**<String> boolean java.util.Collections.replaceAll(List<String> list, String oldVal, String newVal)**  
**replaceAll**  
*public static <T> boolean replaceAll(List<T> list, T oldVal, T newVal)*  
Replaces all occurrences of one specified value in a list with another. More formally, replaces with newVal each element e in list such that (oldVal==null ? e==null : oldVal.equals(e)). (This method has no effect on the size of the list.)  
**Parameters:**  
list – the list in which replacement is to occur.  
oldVal – the old value to be replaced.  
newVal – the new value with which oldVal is to be replaced.  
**Returns:**  
true if list contained one or more elements e such that (oldVal==null ? e==null : oldVal.equals(e)).  
**Throws:**  
UnsupportedOperationException – if the specified list or its list-iterator does not support the set operation.

**Sample Program**  
**ReplaceElementInList.java**

package com.javatutorialcorner.collection;

import java.util.Arrays;

import java.util.Collections;

import java.util.List;

public class ReplaceElementInList {

public static void main(String[] args) {

String[] nameArray = new String[] { "Spring", "Struts", "Hibernate",

"Webservice", "SOAP Webservice", "RESTful Webservice" };

List<String> nameList = Arrays.asList(nameArray);

System.out.println("\*\*\*\*\*\*\*\*\*\*\* Before Replace \*\*\*\*\*\*\*\*\*\*\*\*");

for (String name : nameList) {

System.out.println(name);

}

System.out.println("\*\*\*\*\*\*\*\*\*\*\* After Replace \*\*\*\*\*\*\*\*\*\*\*\*");

Collections.replaceAll(nameList, "Spring", "Spring Framework");

for (String name : nameList) {

System.out.println(name);

}

}

}

**Result**  
The above code will produce the following output.

*\*\*\*\*\*\*\*\*\*\*\* Before Replace \*\*\*\*\*\*\*\*\*\*\*\*  
Spring  
Struts  
Hibernate  
Webservice  
SOAP Webservice  
RESTful Webservice  
\*\*\*\*\*\*\*\*\*\*\* After Replace \*\*\*\*\*\*\*\*\*\*\*\*  
Spring Framework  
Struts  
Hibernate  
Webservice  
SOAP Webservice  
RESTful Webservice*