ArrayList in Java: Complete Guide with Methods and Examples

An ArrayList in Java is a resizable array, part of the java.util package. Unlike a traditional array, it can dynamically grow and shrink in size when elements are added or removed.

Key Features of ArrayList:

1. Dynamic Resizing
2. Zero-based Indexing
3. Duplicate Elements Allowed
4. Non-Synchronized
5. Generic Support

Below is a detailed explanation of ArrayList methods along with examples.

Method: addAll(Collection<? extends E> c)

Description: Appends all elements from the specified collection to the end of the list.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange"));
ArrayList<String> newFruits = new ArrayList<>(Arrays.asList("Grapes", "Pineapple"));
fruits.addAll(newFruits);
System.out.println(fruits); // Output: [Apple, Orange, Grapes, Pineapple]
```

Method: addAll(int index, Collection<? extends E> c)

Description: Inserts all elements from the specified collection starting at the given index.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange"));
ArrayList<String> newFruits = new ArrayList<>(Arrays.asList("Grapes",
"Pineapple"));
fruits.addAll(1, newFruits);
System.out.println(fruits); // Output: [Apple, Grapes, Pineapple, Orange]
```

Method: removeAll(Collection<?> c)

Description: Removes all elements that are present in the specified collection.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange",
"Grapes"));
ArrayList<String> unwantedFruits = new ArrayList<>(Arrays.asList("Grapes",
"Banana"));
fruits.removeAll(unwantedFruits);
System.out.println(fruits); // Output: [Apple, Orange]
```

Method: retainAll(Collection<?> c)

Description: Retains only the elements that are present in both the list and the specified collection.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange",
"Grapes"));
ArrayList<String> commonFruits = new ArrayList<>(Arrays.asList("Apple",
```

```
"Banana"));
fruits.retainAll(commonFruits);
System.out.println(fruits); // Output: [Apple]
```

Method: subList(int fromIndex, int toIndex)

Description: Returns a view of the specified range of the list (inclusive of fromIndex, exclusive of toIndex).

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange",
"Grapes", "Pineapple"));
List<String> subList = fruits.subList(1, 3);
System.out.println(subList); // Output: [Orange, Grapes]
```

Method: toArray()

Description: Converts the list to an array of Object.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange"));
Object[] array = fruits.toArray();
System.out.println(Arrays.toString(array)); // Output: [Apple, Orange]
```

Method: toArray(T[] a)

Description: Converts the list to a typed array.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange"));
String[] array = fruits.toArray(new String[0]);
System.out.println(Arrays.toString(array)); // Output: [Apple, Orange]
```

Method: sort(Comparator<? super E> c)

Description: Sorts the list according to the specified comparator or natural order.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Banana", "Apple",
"Orange"));
fruits.sort(Comparator.naturalOrder());
System.out.println(fruits); // Output: [Apple, Banana, Orange]
```

Method: indexOf(Object o)

Description: Returns the index of the first occurrence of the element, or -1 if not found.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange",
"Apple"));
int index = fruits.indexOf("Apple");
System.out.println(index); // Output: 0
```

Method: lastIndexOf(Object o)

Description: Returns the index of the last occurrence of the element, or -1 if not found.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange",
"Apple"));
int lastIndex = fruits.lastIndexOf("Apple");
System.out.println(lastIndex); // Output: 2
```

Method: forEach(Consumer<? super E> action)

Description: Applies the given action to each element of the list.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange"));
fruits.forEach(System.out::println);
// Output:
// Apple
// Orange
```

Method: removeIf(Predicate<? super E> filter)

Description: Removes all elements that satisfy the given predicate.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange",
"Banana"));
fruits.removeIf(fruit -> fruit.startsWith("B"));
System.out.println(fruits); // Output: [Apple, Orange]
```

Method: clone()

Description: Creates a shallow copy of the ArrayList.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange"));
ArrayList<String> clonedList = (ArrayList<String>) fruits.clone();
System.out.println(clonedList); // Output: [Apple, Orange]
```

Method: spliterator()

Description: Creates a Spliterator for the list to support parallel streams.

Example:

```
ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Orange",
"Banana"));
Spliterator<String> spliterator = fruits.spliterator();
spliterator.forEachRemaining(System.out::println);
// Output:
// Apple
// Orange
// Banana
```