TECHIN

ArrayList

Jaroslav Grablevski

ArrayList

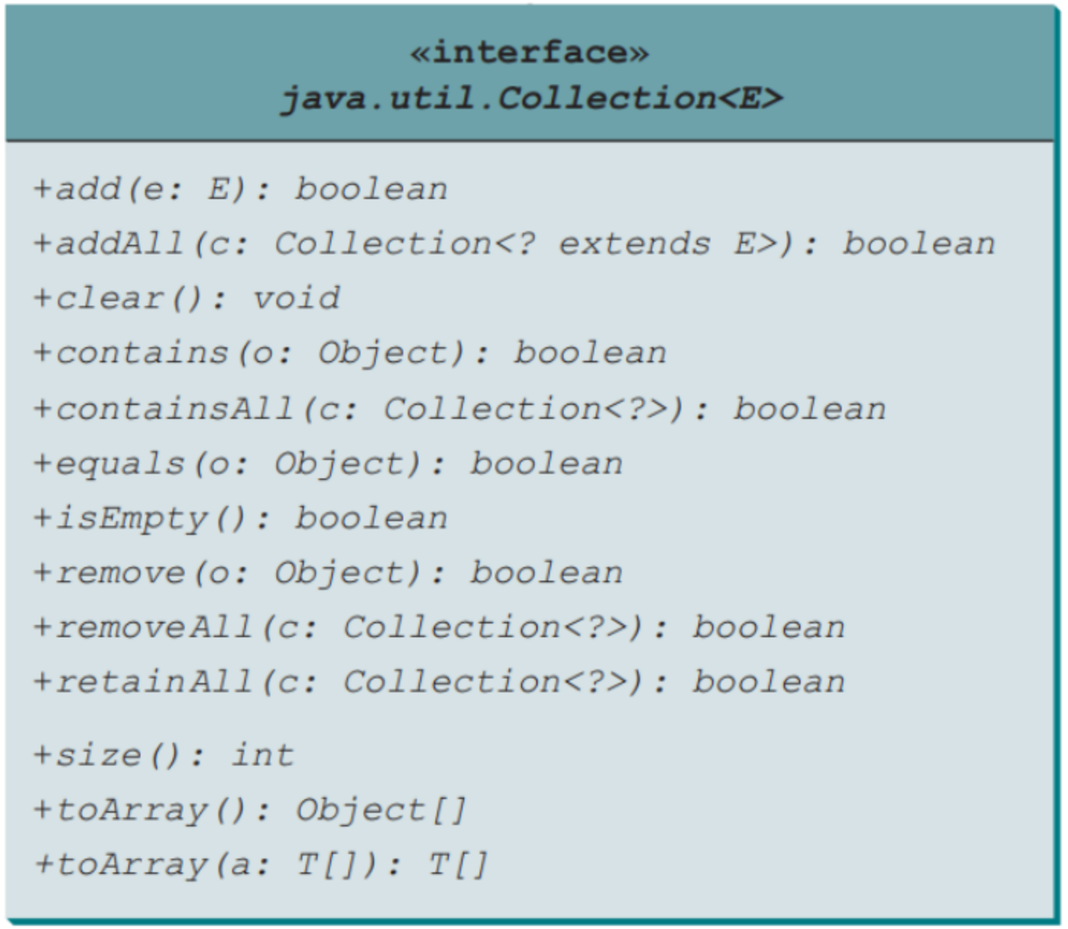
- one of the most widely used classes of Java Class Library
- dynamically grow after the addition and shrink after the removal of its elements
- class is built on top of a standard Java array
- being a generic class, it cannot store primitive types
- can store any reference types (including String‘s), wrapper classes (like Integer‘s)

```
ArrayList<String> names = new ArrayList<>();
ArrayList<Integer> ints = new ArrayList<>();
```

ArrayList

- `int size()` return the number of elements of the list;
- `Object get(int index)` returns the object of the list which is present at the specified index;
- `add(Object o)` adds a passed element to the last position of the collection;
- `add(int index, Object o)` adds a passed element to the specified position of the collection;
- `set(int index, Object o)` replaces the element present at the specified index with the object;
- `remove(Object o)` removes the object from the array;
- `remove(int index)` removes element from a given index;
- `clear()` removes all elements from the collection.

Collection

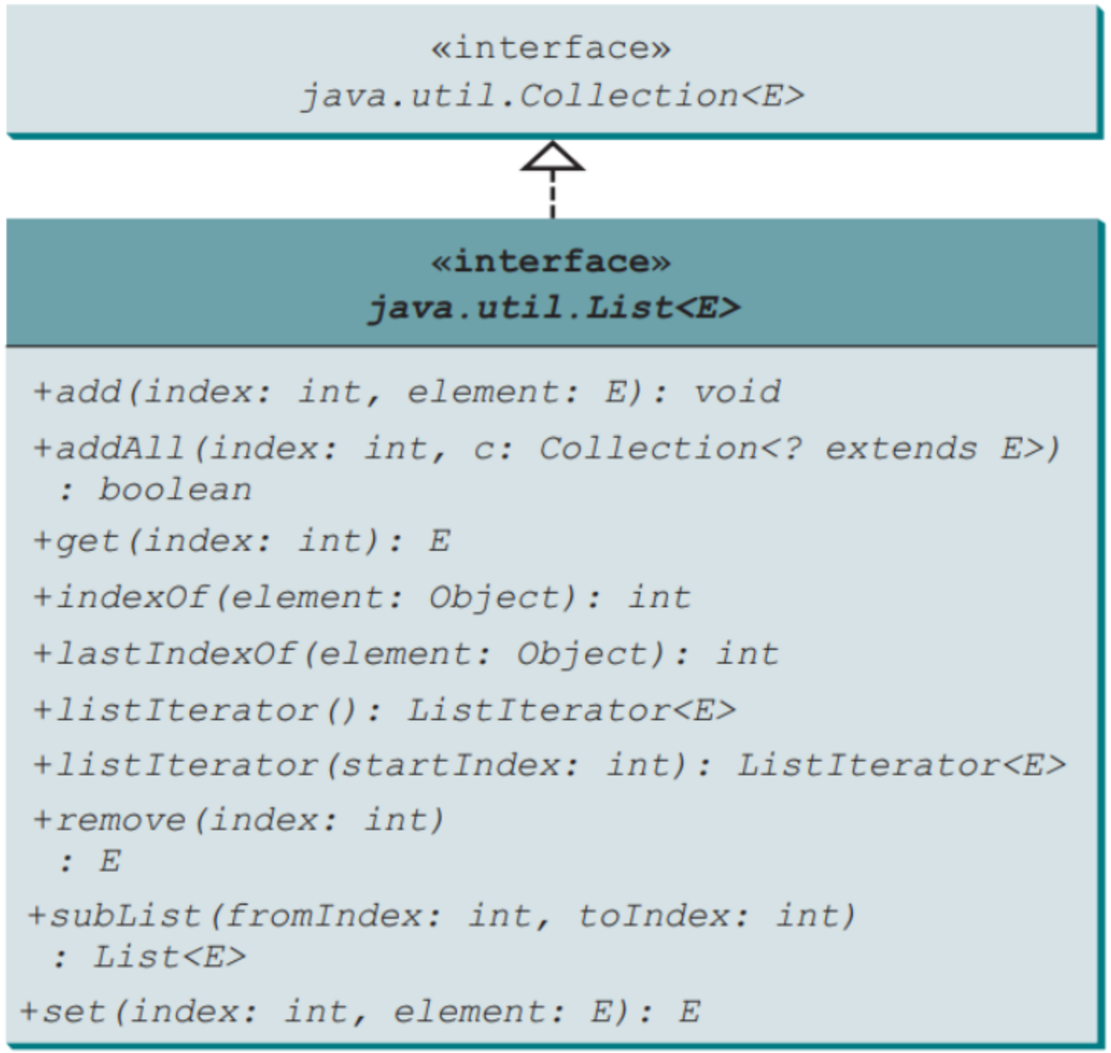


Adds a new element `e` to this collection.
Adds all the elements in the collection `c` to this collection.
Removes all the elements from this collection.
Returns true if this collection contains the element `o`.
Returns true if this collection contains all the elements in `c`.

Returns true if this collection contains no elements.
Removes the element `o` from this collection.
Removes all the elements in `c` from this collection.
Retains the elements that are both in `c` and in this collection.

Returns the number of elements in this collection.
Returns an array of `Object` for the elements in this collection.
Returns an array of the T[] type.

List



Adds a new element at the specified index.

Adds all the elements in `c` to this list at the specified index.

Returns the element in this list at the specified index.

Returns the index of the first matching element.

Returns the index of the last matching element.

Returns the list iterator for the elements in this list.

Returns the iterator for the elements from `startIndex`.

Removes the element at the specified index and returns the removed element.

Sets the element at the specified index and returns the old element.

Returns a sublist from `fromIndex` to `toIndex-1`.

Collections static methods

	java.util.Collections	
List	+sort(list: List): void	Sorts the specified list.
	+sort(list: List, c: Comparator): void	Sorts the specified list with the comparator.
	+binarySearch(list: List, key: Object): int	Searches the key in the sorted list using binary search.
	+binarySearch(list: List, key: Object, c: Comparator): int	Searches the key in the sorted list using binary search with the comparator.
	+reverse(list: List): void	Reverses the specified list.
	+reverseOrder(): Comparator	Returns a comparator with the reverse ordering.
	+shuffle(list: List): void	Shuffles the specified list randomly.
	+shuffle(list: List, rmd: Random): void	Shuffles the specified list with a random object.
	+copy(des: List, src: List): void	Copies from the source list to the destination list.
	+nCopies(n: int, o: Object): List	Returns a list consisting of *n* copies of the object.
	+fill(list: List, o: Object): void	Fills the list with the object.
Collection	+max(c: Collection): Object	Returns the `max` object in the collection.
	+max(c: Collection, c: Comparator): Object	Returns the `max` object using the comparator.
	+min(c: Collection): Object	Returns the `min` object in the collection.
	+min(c: Collection, c: Comparator): Object	Returns the `min` object using the comparator.
	+disjoint(c1: Collection, c2: Collection): boolean	Returns true if `c1` and `c2` have no elements in common.
	+frequency(c: Collection, o: Object): int	Returns the number of occurrences of the specified element in the collection.

Collections static methods (sort)

```
ArrayList<String> cities = new ArrayList<>();
cities.add("Vilnius");
cities.add("Kaunas");
cities.add("Klaipeda");

Collections.sort(cities);

for (String city : cities) {
    System.out.print(city + " ");
}
```

```
Kaunas Klaipeda Vilnius
```

ArrayList

- `int size()` return the number of elements of the list;
- `Object get(int index)` returns the object of the list which is present at the specified index;
- `add(Object o)` adds a passed element to the last position of the collection;
- `add(int index, Object o)` adds a passed element to the specified position of the collection;
- `set(int index, Object o)` replaces the element present at the specified index with the object;
- `remove(Object o)` removes the object from the array;
- `remove(int index)` removes element from a given index;
- `clear()` removes all elements from the collection.

ArrayList

```
ArrayList<String> names = new ArrayList<>(); // empty collection of strings

System.out.println(names.size()); // 0

names.add("Justin");    // now: [Justin]
names.add("Helen");     // now: [Justin, Helen]
names.add("Joshua");    // now: [Justin, Helen, Joshua]

System.out.println(names); // [Justin, Helen, Joshua]
System.out.println(names.size()); // 3

System.out.println(names.get(0)); // the first element is "Justin"

names.add(0, "Teresa"); // now: [Teresa, Justin, Helen, Joshua]

names.remove("Helen"); // now: [Teresa, Justin, Joshua]
names.clear(); // now: []
```

ArrayList

```java
/* an ArrayList of Integers, not ints */
ArrayList<Integer> numbers = new ArrayList<>();

numbers.add(1);
numbers.add(2);
numbers.add(3);
numbers.add(1);

System.out.println(numbers.contains(2)); // true
System.out.println(numbers.contains(4)); // false
System.out.println(numbers.indexOf(1)); // 0
System.out.println(numbers.lastIndexOf(1)); // 3
System.out.println(numbers.lastIndexOf(4)); // -1
```

Iterating over ArrayList

```
ArrayList<Long> powersOfTen = new ArrayList<>();

int count = 5;
for (int i = 0; i < count; i++) {
    long power = (long) Math.pow(10, i);
    powersOfTen.add(power);
}

for (Long value : powersOfTen) {
    System.out.print(value + " ");
}
```

```
1 10 100 1000 10000
```

Iterating over ArrayList (forEach method)

```
ArrayList<String> collection = new ArrayList<>();
collection.add("Vilnius");
collection.add("Kaunas");
collection.add("Klaipeda");

collection.forEach(e -> System.out.print(e.toUpperCase() + " "));
```

```
VILNIUS KAUNAS KLAIPEDA
```

```
for (String e : collection) {
    System.out.print(e.toUpperCase() + " ");
}
```