**Array Java Documentation**

static int: [**getLength**](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/reflect/Array.html#getLength(java.lang.Object))([**Object**](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/Object.html) array) This property is used to get the size of the array. Returns the length as an int (integer).

static void: [**setChar**](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/reflect/Array.html#setChar(java.lang.Object,int,char)) ([**Object**](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/Object.html) array, int index, char c) Sets the value of the indexed component of the specified array object to the specified char value.

static Boolean: [**getBoolean**](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/reflect/Array.html#getBoolean(java.lang.Object,int)) ([**Object**](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/Object.html) array, int index) Returns the value of the indexed component in the specified array object, as a Boolean. It is useful for retrieving Boolean values for system properties.

static long: [**getLong**](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/reflect/Array.html#getLong(java.lang.Object,int)) ([**Object**](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/Object.html) array, int index) Returns the long value of the component in the specified array object.

static void: [**setShort**](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/reflect/Array.html#setShort(java.lang.Object,int,short))([**Object**](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/Object.html) array, int index, short s)Sets the value of the indexed component of the specified array object to the specified short value

**Difference between List, Set and Map in Java**

Collection interface, is broken into three sub-interfaces, which is a core member of Java. Three sub-interfaces are List, Set and map. Each is briefly discussed below.

List represents an ordered collection of objects which duplicates can be used. You can add, update, remove and search with List properties. You use it with an ArrayList, LinkedList and Vector. The list function allows elements to be used by the index position.

public interface List<E> extends Collection<E>

Set is an unordered collection of objects where duplicates cannot be stored. There is only one null value in the set. You can use the set with a variety of class like HashSet, TreeSet, LinkedHashSet, and others.

public interface Set<E> extends Collection<E>

Map Interface is used to represent mapping between a key and a value. Duplicate elements is not allowed. Each key is linked to a specific The Java Map is not a subtype of the collections interface acts different from the other collections. Null implementation is excepted, only one, in HashMap and LinkedHashMap. It allows multiple null values in most

[Collection Interface in Java | GeeksforGeeks](https://www.geeksforgeeks.org/collection-interface-in-java-with-examples/)

[List vs Set vs Map in Java - Tpoint Tech](https://www.tpointtech.com/list-vs-set-vs-map-in-java)