Nicholas Kamper
Sean Halpin
Travis Baumbaugh

Sets

Set is a sub interface of the Collection interface.

Set contains a list of objects without any objects that are equal to each other.

Only one null object is allowed in a set.

Sets contain a method, toArray(), that will return an array of the elements of the set.

When adding objects to a set, a boolean is returned describing whether or not the added object was successfully added(true) or if was ignored(false).

The addAll() method takes a Collection as a parameter and adds all of the elements within the collection to the set, as well as returns a boolean like the add method.

The remove() and removeAll() methods remove the specified object/collection and returns a boolean similar to add() and addAll().

AbstractSet, HashSet, LinkedHashSet, and TreeSet are all classes that implement Set.

When making a class that implements a Set or Map, and in which order is not important, you should implement a Hash version , and thus need to implement methods .equals() and .hashCode() in order to keep runtime down. (Trust us, runtime skyrockets if you don't)

There is a retainAll() method that takes a collection and will remove all of the elements within the set that aren't within the given collection and ignore the elements of the collection that aren't within the set.

Map is an interface that allows you to map some sort of keys to some sort of values. The keys are like a set; you cannot have duplicate keys. Keys may map to at most one value. There are three collection views that can be used with a map interface: the keys can be viewed as a set, the values can be viewed as a collection, or the mapping can be viewed as a set of keys mapped to the values.

The containsKey(Object key) and containsValue(Object value) methods return whether or not the given value or key is contained in the mapping.

The entrySet(), keyset(), and values() methods return a set view of the mapped entries, a set view of the keys, and a collection view of the values, respectively.

The methods get(Object key) and remove(Object key) respectively get or remove the value mapped to by the given key.

There are also methods like put(K key, V value) and putAll(Map m) that allow the adding of mappings.

Like with set, there are equals() and hashCode() methods.

There are also clear(), isEmpty(), and size() methods, which (rather obviously) clear the

mapping, test for emptiness, and return the number of mappings respectively.