# Package myContainerPackage

## Interface Summary

| Interface | Description |
|---|---|
| **JavaContainer**<T> | JavaContainer interface |

## Class Summary

| Class | Description |
|---|---|
| **JavaSet**<T extends java.lang.Comparable<T>> | JavaSet class |
| **JavaVector**<T> | JavaVector class |

**Package** myContainerPackage

# Interface JavaContainer<T>

**Type Parameters:**

T - type of data It is a generic interface for JavaSet and JavaVector It has 4 methods: 1. Add(T n) : add an element to the container 2. Remove(T n) : remove an element from the container 3. Size() : return the size of the container 4. getIterator() : return an iterator of the container

**All Known Implementing Classes:**

JavaSet, JavaVector

---

```
public interface JavaContainer<T>
```

JavaContainer interface

## *Method Summary*

**All Methods** | **Instance Methods** | **Abstract Methods**

| Modifier and Type | Method | Description |
|---|---|---|
| boolean | Add(T n) | Add method |
| java.util.Iterator<T> | getIterator() | getIterator method |
| boolean | Remove(T n) | Remove method |
| int | Size() | Size method |

## *Method Detail*

### Add

`boolean Add(T n)`

Add method

**Parameters:**

`n` - element to be added It adds the given element to the container

### Remove

`boolean Remove(T n)`

Remove method

**Parameters:**

`n` - element to be removed It removes the given element from the container

### Size

`int Size()`

Size method

**Returns:**

size of the container It returns the size of the container

### getIterator

`java.util.Iterator<T> getIterator()`

getIterator method

**Returns:**

iterator of the container It returns an iterator of the container

**Package** myContainerPackage

# Class JavaSet<T extends java.lang.Comparable<T>>

java.lang.Object
    myContainerPackage.JavaSet<T>

**Type Parameters:**

T - type of data It is a generic class for JavaSet. It has 15 methods: 1. Add(T n) : add an element to the container 2. Remove(T n) : remove an element from the container 3. Size() : return the size of the container 4. getCapacity() : return the capacity of the container 5. getData(int index) : return the data at the given index 6. isIn(T element) : return true if the element is in the container 7. getIterator() : return an iterator of the container 8. toString() : return a string representation of the container 9. equals(Object obj) : return true if the given object is equal to the container 10. JavaSet() : default constructor 11. JavaSet(int n) : constructor with capacity 12. JavaSet(JavaSet other) : copy constructor 13. IteratorImpl : private class for iterator 14. hasNext() : return true if the iterator has next element 15. next() : return the next element of the iterator

**All Implemented Interfaces:**

JavaContainer<T>

---

```
public class JavaSet<T extends java.lang.Comparable<T>>
extends java.lang.Object
implements JavaContainer<T>
```

JavaSet class

## *Constructor Summary*

### Constructors

| Constructor | Description |
|---|---|
| JavaSet() | JavaSet constructor It creates a JavaSet object with default capacity 2 |
| JavaSet(int n) | JavaSet constructor |

| | |
|---|---|
| **JavaSet**(**JavaSet**<**T**> other) | JavaSet constructor |

## *Method Summary*

**All Methods** | **Instance Methods** | **Concrete Methods**

| Modifier and Type | Method | Description |
|---|---|---|
| boolean | **Add**(**T** n) | Add(T n) method |
| boolean | **equals**(java.lang.Object obj) | equals(Object obj) method |
| int | **getCapacity**() | getCapacity() method |
| **T** | **getData**(int index) | getData() method |
| java.util.Iterator<**T**> | **getIterator**() | getIterator() method |
| boolean | **isIn**(**T** element) | isIn(T element) method |
| boolean | **Remove**(**T** n) | Remove(T n) method |
| int | **Size**() | Size() method |
| java.lang.String | **toString**() | toString() method |

### Methods inherited from class java.lang.Object

clone, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

## *Constructor Detail*

### JavaSet

```
public JavaSet()
```

JavaSet constructor It creates a JavaSet object with default capacity 2

### JavaSet

```
public JavaSet(int n)
```

JavaSet constructor

**Parameters:**

`n` - capacity of the JavaSet

**Throws:**

`java.security.InvalidParameterException` - if the given capacity is invalid It creates a JavaSet object with given capacity

### JavaSet

```
public JavaSet(JavaSet<T> other)
```

JavaSet constructor

**Parameters:**

`other` - JavaSet object It creates a JavaSet object with given JavaSet object

## *Method Detail*

### getData

```
public T getData(int index)
```

getData() method

**Parameters:**

`index` - index of the data

**Returns:**

data at the given index

**Throws:**

`java.security.InvalidParameterException` - if the given index is invalid It returns the data at the given index

### Size

```
public int Size()
```

Size() method

**Specified by:**

`Size` in interface `JavaContainer<T extends java.lang.Comparable<T>>`

**Returns:**

size of the container It returns the size of the container

### getCapacity

```
public int getCapacity()
```

getCapacity() method

**Returns:**

capacity of the container It returns the capacity of the container

### getIterator

```
public java.util.Iterator<T> getIterator()
```

getIterator() method

**Specified by:**

`getIterator` in interface `JavaContainer<T extends java.lang.Comparable<T>>`

**Returns:**

iterator of the container It returns the iterator of the container

## Add

```
public boolean Add(T n)
```

Add(T n) method

**Specified by:**

`Add` in interface `JavaContainer<T extends java.lang.Comparable<T>>`

**Parameters:**

n - element to be added It adds the given element to the container It doubles the capacity if the size is equal to the capacity It adds the element to the container in order It shifts the elements after the given element It adds the element to the end if the element is the largest It increases the size by 1

**Returns:**

true if the element is added to the container

**Throws:**

`java.security.InvalidParameterException` - if the element is already in the container

## Remove

```
public boolean Remove(T n)
```

Remove(T n) method

**Specified by:**

`Remove` in interface `JavaContainer<T extends java.lang.Comparable<T>>`

**Parameters:**

n - element to be removed It removes the given element from the container It halves the capacity if the size is equal to half of the capacity It removes the element from the container It shifts the elements after the given element It decreases the size by 1

**Returns:**

true if the element is removed from the container

**Throws:**

`java.security.InvalidParameterException` - if the given element is not in the container

`java.lang.ArithmeticException` - if the container is empty

### isIn

```
public boolean isIn(T element)
```

isIn(T element) method

**Parameters:**

`element` - element to be checked

**Returns:**

true if the element is in the container It returns true if the element is in the container

### toString

```
public java.lang.String toString()
```

toString() method

**Overrides:**

`toString` in class `java.lang.Object`

**Returns:**

string representation of the container It returns a string representation of the container

### equals

```
public boolean equals(java.lang.Object obj)
```

equals(Object obj) method

**Overrides:**

`equals` in class `java.lang.Object`

**Parameters:**

`obj` - object to be compared

**Returns:**

true if the given object is equal to the container It returns false if the given object is null or the given object is not a JavaSet object It returns false if the size of the given object is not equal to the size of the container It returns false if the elements of the given object are not equal to the elements of the container

**Package** myContainerPackage

# Class JavaVector<T>

java.lang.Object
    myContainerPackage.JavaVector<T>

**Type Parameters:**

T - type of data It is a generic class for JavaVector. It has 15 methods: 1. Add(T n) : add an element to the container 2. Remove(T n) : remove an element from the container 3. Size() : return the size of the container 4. getCapacity() : return the capacity of the container 5. getData(int index) : return the data at the given index 6. isIn(T element) : return true if the element is in the container 7. getIterator() : return an iterator of the container 8. toString() : return a string representation of the container 9. equals(Object obj) : return true if the given object is equal to the container 10. JavaVector() : default constructor 11. JavaVector(int n) : constructor with capacity 12. JavaVector(JavaVector other) : copy constructor 13. IteratorImpl : private class for iterator 14. hasNext() : return true if the iterator has next element 15. next() : return the next element of the iterator

**All Implemented Interfaces:**

JavaContainer<T>

---

```
public class JavaVector<T>
extends java.lang.Object
implements JavaContainer<T>
```

JavaVector class

## *Constructor Summary*

### Constructors

| Constructor | Description |
|---|---|
| JavaVector() | JavaVector constructor It creates a JavaVector object with default capacity 2 |
| JavaVector(int n) | JavaVector constructor |

| | |
|---|---|
| JavaVector(JavaVector<T> other) | JavaVector constructor |

## *Method Summary*

**All Methods** | **Instance Methods** | **Concrete Methods**

| Modifier and Type | Method | Description |
|---|---|---|
| boolean | Add(T element) | Add method. |
| boolean | equals(java.lang.Object obj) | hasNext method |
| int | getCapacity() | getCapacity method |
| T | getData(int index) | getData method |
| java.util.Iterator<T> | getIterator() | getIterator method |
| boolean | isIn(T element) | isIn method. |
| boolean | Remove(T element) | Remove method. |
| void | setExactData(int index, T newData) | getExactData method |
| int | Size() | Size method. |
| java.lang.String | toString() | next method |

### Methods inherited from class java.lang.Object

clone, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

## *Constructor Detail*

### JavaVector

```
public JavaVector()
```

JavaVector constructor It creates a JavaVector object with default capacity 2

### JavaVector

```
public JavaVector(int n)
```

JavaVector constructor

**Parameters:**

`n` - capacity of the JavaVector

**Throws:**

`java.security.InvalidParameterException` - if the given capacity is invalid It creates a JavaVector object with given capacity

### JavaVector

```
public JavaVector(JavaVector<T> other)
```

JavaVector constructor

**Parameters:**

`other` - JavaVector object It creates a JavaVector object with given JavaVector object

## *Method Detail*

### getData

```
public T getData(int index)
```

getData method

**Parameters:**

`index` - index of the data

**Returns:**

data at the given index

**Throws:**

`java.security.InvalidParameterException` - if the given index is invalid It returns the data at the given index

### setExactData

```
public void setExactData(int index, T newData)
```

getExactData method

**Parameters:**

`index` - index of the data

`newData` - new data of the given index

**Throws:**

`java.lang.RuntimeException` - if the given index is invalid It returns the data at the given index

### getCapacity

```
public int getCapacity()
```

getCapacity method

**Returns:**

capacity of the container It returns the capacity of the container

### getIterator

```
public java.util.Iterator<T> getIterator()
```

getIterator method

**Specified by:**

`getIterator` in interface `JavaContainer<T>`

**Returns:**

iterator of the container It returns an iterator of the container

### Add

```
public boolean Add(T element)
```

Add method.

**Specified by:**

`Add` in interface `JavaContainer<T>`

**Parameters:**

`element` - element to be added. It adds the given element to the container. If the size is equal to capacity, it doubles the capacity. If the size is equal to 0, it adds the element to the first index. Otherwise, it adds the element to the end of the container. It increases the size by 1. It gives a warning if the element is already in the container.

**Returns:**

true if the element is added to the container.

### Remove

```
public boolean Remove(T element)
```

Remove method.

**Specified by:**

`Remove` in interface `JavaContainer<T>`

**Parameters:**

`element` - element to be removed. It removes the given element from the container. If the size is less than or equal to capacity / 2, it halves the capacity. It creates a new array with the new capacity. It copies the elements except the given element to the new array. It assigns the new array to the data. It decreases the size by 1.

**Returns:**

true if the element is removed from the container.

**Throws:**

`java.security.InvalidParameterException` - if the element is not in the container.

`java.lang.ArithmeticException` - if the container is empty.

### isIn

```
public boolean isIn(T element)
```

isIn method.

**Parameters:**

`element` - element to be checked.

**Returns:**

true if the element is in the container. It returns true if the element is in the container. Otherwise, it returns false.

### Size

```
public int Size()
```

Size method.

**Specified by:**

`Size` in interface `JavaContainer<T>`

**Returns:**

size of the container. It returns the size of the container.

### equals

```
public boolean equals(java.lang.Object obj)
```

hasNext method

**Overrides:**

`equals` in class `java.lang.Object`

**Returns:**

true if the iterator has next element It returns true if the iterator has next element. Otherwise, it returns false.

### toString

```
public java.lang.String toString()
```

next method

**Overrides:**

`toString` in class `java.lang.Object`

**Returns:**

the next element of the iterator. It returns the next element of the iterator.

A E G I J M R S T
All Classes All Packages

## A



## E



## G

## I



## J



## M

## R

**Remove(T)** - Method in interface myContainerPackage.JavaContainer
Remove method
**Remove(T)** - Method in class myContainerPackage.JavaSet
Remove(T n) method
**Remove(T)** - Method in class myContainerPackage.JavaVector
Remove method.

## S

**setExactData(int, T)** - Method in class myContainerPackage.JavaVector
getExactData method
**Size()** - Method in interface myContainerPackage.JavaContainer
Size method
**Size()** - Method in class myContainerPackage.JavaSet
Size() method
**Size()** - Method in class myContainerPackage.JavaVector
Size method.

## T

**toString()** - Method in class myContainerPackage.JavaSet
toString() method
**toString()** - Method in class myContainerPackage.JavaVector
next method