All Classes

A

E

G

J

M

The main method that serves as the entry point of the program.
myInterface - package myInterface

mySource - package mySource

R

remove(T) - Method in interface myInterface.JavaContainer
Removes an element from the container.
remove(T) - Method in class mySource.JavaSet
Removes an element from the set.
remove(T) - Method in class mySource.JavaVector
Removes the specified element from the vector.

S

Size() - Method in interface myInterface.JavaContainer
Returns the number of elements in the container.
Size() - Method in class mySource.JavaSet
Returns the size (number of elements) of the set.
Size() - Method in class mySource.JavaVector
Returns the size of the vector.

T

toString() - Method in class mySource.JavaSet
Returns a string representation of the set.
toString() - Method in class mySource.JavaVector
Returns a string representation of the vector.

Packages

Package	Description
myInterface	
mySource	

Hierarchy For All Packages

Package Hierarchies:
myInterface, mySource

Class Hierarchy

- java.lang.Object
  - **JavaContainerTest**
  - mySource.**JavaSet**<T> (implements myInterface.JavaContainer<T>)
  - mySource.**JavaVector**<T> (implements myInterface.JavaContainer<T>)

Interface Hierarchy

- myInterface.**JavaContainer**<T>

Package <Unnamed>

Class Summary

Class	Description
JavaContainerTest	JavaContainerTest class demonstrates the usage of JavaSet and JavaVector classes.

Package myInterface

Interface JavaContainer<T>

Type Parameters:

T - The type of elements that the container holds.

All Known Implementing Classes:

JavaSet, JavaVector

`public interface JavaContainer<T>`

The JavaContainer interface defines a basic contract for classes that represent a container for elements of type T. Containers allow elements to be added, removed, and iterated over.

Method Summary

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

Modifier and Type	Method	Description
void	**add**(T element)	Adds an element to the container.
java.util.Iterator<T>	**getIterator**()	Returns an iterator over the elements in the container.
void	**remove**(T element)	Removes an element from the container.
int	**Size**()	Returns the number of elements in the container.

Method Detail

add

`void add(T element)`

Adds an element to the container.

Parameters:

`element` - The element to be added to the container.

remove

`void remove(T element)`

Removes an element from the container.

Parameters:

`element` - The element to be removed from the container.

Size

`int Size()`

Returns the number of elements in the container.

Returns:
The size of the container.

getIterator

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

Returns an iterator over the elements in the container.

Returns:
An iterator for iterating over the elements in the container.

Package myInterface

Interface Summary

Interface	Description
JavaContainer<T>	The JavaContainer interface defines a basic contract for classes that represent a container for elements of type T.

Hierarchy For Package myInterface

Package Hierarchies:
All Packages

Interface Hierarchy

- myInterface.**JavaContainer**<T>

Package mySource

Class JavaSet<T>

java.lang.Object
    mySource.JavaSet<T>

Type Parameters:

T - The type of elements in the set.

All Implemented Interfaces:

JavaContainer<T>

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

The JavaSet class implements the JavaContainer interface, providing a set data structure that holds elements of a specified type T. It supports addition, removal, and iteration over elements.

Constructor Summary

Constructors

Constructor	Description
JavaSet()	Constructs a JavaSet with a default initial capacity of 10.
JavaSet(int _capacity)	Constructs a JavaSet with the specified initial capacity.

Method Summary

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

Modifier and Type	Method	Description
void	add(T element)	Adds an element to the set if it does not already exist.
boolean	equals (java.lang.Object obj)	Checks if the current set is equal to another object.
int	getCapacity()	Returns the current capacity of the set.
java.util.Iterator<T>	getIterator()	Returns an iterator over the elements in the set.
void	remove(T element)	Removes an element from the set.
int	Size()	Returns the size (number of elements) of the set.
java.lang.String	toString()	Returns a string representation of the set.

Methods inherited from class java.lang.Object

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

Constructor Detail

JavaSet

```
public JavaSet()
```

Constructs a JavaSet with a default initial capacity of 10.

JavaSet

```
public JavaSet(int _capacity)
```

Constructs a JavaSet with the specified initial capacity.

Parameters:

`_capacity` - The initial capacity of the set.

Method Detail

add

```
public void add(T element)
```

Adds an element to the set if it does not already exist.

Specified by:

`add` in interface `JavaContainer<T>`

Parameters:

`element` - The element to be added to the set.

Throws:

`java.lang.IllegalStateException` - If the element is already present in the set.

remove

```
public void remove(T element)
```

Removes an element from the set.

Specified by:

`remove` in interface `JavaContainer<T>`

Parameters:

`element` - The element to be removed from the set.

Throws:

`java.lang.IllegalStateException` - If the element isnt present in the set.

Size

```
public int Size()
```

Returns the size (number of elements) of the set.

Specified by:
`Size` in interface `JavaContainer<T>`

Returns:
The size of the set.

getCapacity

```
public int getCapacity()
```

Returns the current capacity of the set.

Returns:
The capacity of the set.

getIterator

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

Returns an iterator over the elements in the set.

Specified by:
`getIterator` in interface `JavaContainer<T>`

Returns:
An iterator for iterating over the elements in the set.

toString

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

Returns a string representation of the set.

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

Returns:
A string representation of the set.

equals

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

Checks if the current set is equal to another object.

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

Parameters:
`obj` - The object to compare with the current set.

Returns:
`true` if the sets are equal, `false` otherwise.

Package mySource

Class JavaVector<T>

java.lang.Object
    mySource.JavaVector<T>

Type Parameters:

T - Type of elements in the vector.

All Implemented Interfaces:

JavaContainer<T>

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

JavaVector class implementing JavaContainer interface.

Constructor Summary

Constructors

Constructor	Description
JavaVector()	Constructs a JavaVector with a default initial capacity of 10.
JavaVector(int capacity)	Constructs a JavaVector with the specified initial capacity.

Method Summary

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

Modifier and Type	Method	Description
void	add(T element)	Adds an element to the vector.
boolean	equals(java.lang.Object obj)	Checks if the vector is equal to another object.
int	getCapacity()	Returns the capacity of the vector.
java.util.Iterator<T>	getIterator()	Returns an iterator over the elements in the vector.
void	remove(T element)	Removes the specified element from the vector.
int	Size()	Returns the size of the vector.
java.lang.String	toString()	Returns a string representation of the vector.

Methods inherited from class java.lang.Object

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

Constructor Detail

JavaVector

`public JavaVector()`

Constructs a JavaVector with a default initial capacity of 10.

JavaVector

`public JavaVector(int capacity)`

Constructs a JavaVector with the specified initial capacity.

Parameters:
`capacity` - The initial capacity of the vector.

Method Detail

add

`public void add(T element)`

Adds an element to the vector.

Specified by:
`add` in interface `JavaContainer<T>`

Parameters:
`element` - The element to be added.

remove

`public void remove(T element)`

Removes the specified element from the vector.

Specified by:
`remove` in interface `JavaContainer<T>`

Parameters:
`element` - The element to be removed.

Throws:
`java.lang.IllegalStateException` - If the element isnt present in the set.

Size

`public int Size()`

Returns the size of the vector.

Specified by:
`Size` in interface `JavaContainer<T>`

Returns:
The size of the vector.

getCapacity

`public int getCapacity()`

Returns the capacity of the vector.

Returns:
The capacity of the vector.

getIterator

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

Returns an iterator over the elements in the vector.

Specified by:
`getIterator` in interface `JavaContainer<T>`

Returns:
An iterator.

toString

`public java.lang.String toString()`

Returns a string representation of the vector.

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

Returns:
A string representation of the vector.

equals

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

Checks if the vector is equal to another object.

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

Parameters:
`obj` - The object to compare with the vector.

Returns:
`true` if the vector is equal to the object, `false` otherwise.

Hierarchy For Package mySource

Package Hierarchies:
All Packages

Class Hierarchy

- java.lang.Object
  - mySource.**JavaSet**<T> (implements myInterface.JavaContainer<T>)
  - mySource.**JavaVector**<T> (implements myInterface.JavaContainer<T>)