Class MyHashSet<T>

```
java.lang.Object
  └ MyHashSet<T>
```

All Implemented Interfaces:

java.lang.Iterable<T>

```
public class MyHashSet<T>
extends java.lang.Object
implements java.lang.Iterable<T>
```

The MyHashSet API is similar to the Java Set interface. This collection is backed by a hash table.

Field Summary

static int	**DEFAULT_INITIAL_CAPACITY** Unless otherwise specified, the table will start as an array of this length.
static double	**LOAD_FACTOR** When the ratio of size/capacity reaches or exceeds this value, it is time for the table to be expanded

Constructor Summary

MyHashSet() Initializes an empty table of length equal to DEFAULT_INITIAL_CAPACITY
MyHashSet(int initialCapacity) Initializes an empty table of the specified length (capacity).

Method Summary

void	**add**(T element) Adds the specified element to the collection.
boolean	**contains**(T element) Looks for the specified element in the table.
int	**getCapacity**() Returns the length of the table (the number of buckets).

int	**getSize**() Returns the number of elements stored in the table.
java.util.Iterator<T>	**iterator**() Returns an Iterator that can be used to iterate over all of the elements in the collection.
void	**remove**(T element) Removes the specified element from the collection.

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

DEFAULT_INITIAL_CAPACITY

public static final int **DEFAULT_INITIAL_CAPACITY**

Unless otherwise specified, the table will start as an array of this length.

See Also:

Constant Field Values

LOAD_FACTOR

public static final double **LOAD_FACTOR**

When the ratio of size/capacity reaches or exceeds this value, it is time for the table to be expanded

See Also:

Constant Field Values

Constructor Detail

MyHashSet

public **MyHashSet**(int initialCapacity)

Initializes an empty table of the specified length (capacity). Relies on the static makeArray method of the Node class.

Parameters:

initialCapacity - initial length (capacity) of table

MyHashSet

`public` **`MyHashSet`**`()`

Initializes an empty table of length equal to DEFAULT_INITIAL_CAPACITY

Method Detail

getSize

`public int` **`getSize`**`()`

Returns the number of elements stored in the table.

Returns:

number of elements in the table

---

getCapacity

`public int` **`getCapacity`**`()`

Returns the length of the table (the number of buckets).

Returns:

length of the table (capacity)

---

contains

`public boolean` **`contains`**`(T element)`

Looks for the specified element in the table.

Parameters:

`element` - to be found

Returns:

true if the element is in the table, false otherwise

---

add

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

Adds the specified element to the collection. If the element is already in the collection, then this method should do nothing. Important: After adding this element to the table, consider the ratio of size/capacity. If this ratio is greater than or equal to the LOAD_FACTOR then you must double the size of the table. This will require re-hashing of all of the data!

Parameters:

`element` - the element to be added to the collection

---

remove

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

Removes the specified element from the collection. If the element is not present then this method should do nothing.

Parameters:

`element` - the element to be removed

---

iterator

`public java.util.Iterator<T>` **`iterator`**`()`

Returns an Iterator that can be used to iterate over all of the elements in the collection.

Specified by:

`iterator` in interface `java.lang.Iterable<T>`