Class HashMap<K,V>

java.lang.Object
        HashMap<K,V>

```
public class HashMap<K,V>
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor and Description
HashMap() constructs an empty HashMap with the default initial capcity (7) and the default load factor (0.5)
HashMap(int cap) Constructs an empty HashMap with the specified inital capacity and the default load factor (0.5)
HashMap(int cap, double lf)

Method Summary

All Methods	Instance Methods	Concrete Methods

Modifier and Type	Method and Description
void	**clear**() removes all mappings from the map.
boolean	**containsKey**(K key) Using the same quadratic probing to determine if there is an entry with matching key
boolean	**containsValue**(V value) Return trues true if there is an entry with matching value
boolean	**isEmpty**() Returns true if the map is empty and false otherwise

V	**put**(**K** key, **V** value) Associate a value with a key in the map.
int	**size**() Returns the number of key-value pairs in this map
java.lang.String	**toString**() returns a string representation of the map

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

HashMap

public HashMap()

constructs an empty HashMap with the default initial capcity (7) and the default load
factor (0.5)

HashMap

public HashMap(int cap)
        throws java.lang.IllegalArgumentException

Constructs an empty HashMap with the specified inital capacity and the default load factor
(0.5)

Parameters:

cap - - the initial capacity

Throws:

java.lang.IllegalArgumentException - - if the inital capcity is negative

HashMap

public HashMap(int cap,
              double lf)

Parameters:

cap - - the initial capacity

lf - - the load factor

Throws:

```
java.lang.IllegalArgumentException - - if the inital capcity is negative or
load factor is nonpositive
```

Method Detail

put

```
public V put(K key,
             V value)
```

Associate a value with a key in the map. If the map previously contained a mapping for the key, the old value is replaced

Parameters:

```
key - - key with which the specified value is to be associated
```

```
value - - value to be associated with the specified key
```

Returns:

```
the previous value associated with key, or null if there was no mapping for
the key. (A null return can also indicate that the map previously associated
null with key).
```

containsKey

```
public boolean containsKey(K key)
```

Using the same quadratic probing to determine if there is an entry with matching key

Parameters:

```
key - - the key whose presence in the map is to be tested
```

Returns:

```
true if this map contains a mapping for the specified key, false otherwise
```

containsValue

```
public boolean containsValue(V value)
```

Return trues true if there is an entry with matching value

Parameters:

```
value - - the value whose presence in the map is to be tested
```

Returns:

```
true if this map maps one or more keys to the specified value
```

size

```
public int size()
```

Returns the number of key-value pairs in this map

Returns:
```
number of key-value pairs in this map
```

isEmpty

```
public boolean isEmpty()
```

Returns true if the map is empty and false otherwise

Returns:
```
true if this map contains no key-value pairs
```

clear

```
public void clear()
```

removes all mappings from the map. The map will be empty after this call returns

toString

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

returns a string representation of the map

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

Returns:
```
a string representation of the map
```