* How HashSet works internally in java?

**HashSet** uses ***HashMap*** internally to store it’s objects. Whenever you create a HashSet object, one ***HashMap*** object associated with it is also created. This HashMap object is used to store the elements you enter in the HashSet. The elements you add into HashSet are stored as ***keys*** of this HashMap object. The value associated with those keys will be a ***constant***.

HashSet belongs to rt.jar.

**public** **class** **HashSet**<E>

**extends** AbstractSet<E>

**implements** Set<E>, Cloneable, java.io.Serializable

{

**private** **transient** HashMap<E,Object> map;

// Constant Dummy value to associate with an Object in the backing Map

**private** **static** **final** Object PRESENT = **new** Object();

**public** **HashSet**() {

*map =* ***new*** *HashMap<>();*

}

// SOME CODE ,i.e Other methods in Hash Set

**public** **boolean** **add**(E e) {

**return** map.put(e, PRESENT)==**null**;

}

// SOME CODE ,i.e Other methods in Hash Set

}

The main point to notice in above code is that put (key,value) will return  
1.  null , if key is unique and added to the map  
2.  Old Value of the key , if key is duplicate

So , in HashSet add() method ,  we check the return value of map.put(key,value) method with null value