Java Set
Java HashSet class is used to create a collection that uses a hash table for storage. It inherits the AbstractSet class and implements Set interface.
The important points about Java HashSet class are:

- HashSet stores the elements by using a mechanism called hashing.
- HashSet contains unique elements only.

Difference between List and Set

List can contain duplicate elements whereas Set contains unique elements only.

The HashSet class extends AbstractSet class which implements Set interface. The Set interface inherits Collection and Iterable interfaces in hierarchical order.

1. public class HashSet<E> extends AbstractSet<E> implements Set<E>, Cloneable, Serializable

void clear()	It is used to remove all of the elements from this set.
boolean contains(Object o)	It is used to return true if this set contains the specified element.
boolean add(Object o)	It is used to adds the specified element to this set if it is not already present.
boolean isEmpty()	It is used to return true if this set contains no elements.
boolean remove(Object o)	It is used to remove the specified element from this set if it is present.
Object clone()	It is used to return a shallow copy of this HashSet instance: the elements themselves are not cloned.
Iterator iterator()	It is used to return an iterator over the elements in this set.
int size()	It is used to return the number of elements in this set.

Java HashMap class

Java HashMap class implements the map interface by using a hashtable. It inherits AbstractMap class and implements Map interface.

The important points about Java HashMap class are:

- A HashMap contains values based on the key.
- It contains only unique elements.
- It may have one null key and multiple null values.
- It maintains no order.

As shown in the above figure, HashMap class extends AbstractMap class and implements Map interface.

```
public class HashMap<K,V> extends AbstractMap<K,V> implements Map<K,V>, Cloneable, Serializable
```

Method	**Description**
void clear()	It is used to remove all of the mappings from this map.
boolean containsKey(Object key)	It is used to return true if this map contains a mapping for the specified key.
boolean containsValue(Object value)	It is used to return true if this map maps one or more keys to the specified value.
boolean isEmpty()	It is used to return true if this map contains no key-value mappings.
Object clone()	It is used to return a shallow copy of this HashMap instance: the keys and values themselves are not cloned.
Set entrySet()	It is used to return a collection view of the mappings contained in this map.
Set keySet()	It is used to return a set view of the keys contained in this map.
Object put(Object key, Object value)	It is used to associate the specified value with the specified key in this map.

int size()	It is used to return the number of key-value mappings in this map.
Collection values()	It is used to return a collection view of the values contained in this map.

EXAMPLES

```
package com.company;


import javax.swing.text.html.HTMLDocument;
import java.util.Collection;
import java.util.Map;
import java.util.HashMap;
import java.util.Properties;
import java.io.InputStream;
import java.io.FileInputStream;
import java.io.IOException;
import java.util.Set;
import java.util.*;
import java.io.*;
import java.lang.Object;

public class Main {
  private static String topicName = "arn:aws:sns:ap-southeast-1:480410955647:c237e19c-7da0-11e5-8bcf-feff819cdc9f";
    public static void main(String[] args) throws IOException
  {
    System.out.println(extractRegion(topicName));
    getProperties();
    Main main = new Main();
    main.getSystemProp();
    getMap();
    main.getconfigProp();
  }
 public static Properties getProperties() throws IOException
 {
  Properties prop = new Properties();
  FileInputStream fis = new FileInputStream("/home/prasanna/erpanderp/TestSNS/src/com/company/credentials");
  prop.loadFromXML(fis);
  String email = prop.getProperty("email.support");
  String userId = prop.getProperty("user.support");
  System.out.println(email+ " \n"+userId);
   return prop;
 }
 private static void getMap()
 {
   Map<String, String> map = new HashMap<String, String>();
```

```
    map.put("email","prasanna635@gmail.com");
    map.put("userId","prasanna635");
    map.put("company","Halcyontek");
    for(Map.Entry<String, String> entry:map.entrySet())
    {
        String key = entry.getKey();
        String value = entry.getValue();
        System.out.println("key is "+key+",  value is "+value);
    }
    Set set = map.entrySet();
    Iterator itr = set.iterator();
   while(itr.hasNext()){
       Map.Entry<String, String> entry = (Map.Entry)itr.next();
       System.out.println(entry.getKey()+" = "+entry.getValue());
}
}
private  void getSystemProp()
{
    Properties p=System.getProperties();
    Set set=p.entrySet();

    Iterator itr=set.iterator();
    while(itr.hasNext()) {
        Map.Entry entry = (Map.Entry) itr.next();
        System.out.println(entry.getKey() + " = " + entry.getValue());
    }
    }
 private static String extractRegion(String testTopicArn) {
     String[] strings = testTopicArn.split(":");
     if (strings.length > 3) {
         return strings[3];
     }
     return "us-east-1";
 }
 private  void getconfigProp() throws IOException
 {
     Properties prop = new Properties();
     String propFileName = "com/company/config.properties";
     InputStream is = getClass().getClassLoader().getResourceAsStream(propFileName);
     if (is != null) {
         prop.load(is);
     } else {
         throw new FileNotFoundException("property file '" + propFileName + "' not found in the classpath");
     }
     String accessKey =  prop.getProperty("s3.accesskey");
     String secretKey =  prop.getProperty("s3.secretkey");
     String bucketName =  prop.getProperty("s3.bucketName");
     System.out.println(accessKey+"\n"+secretKey+"\n"+bucketName);
 }
 }
```