HashSet Examples

Below example shows how to read all elements from the HashSet objects. You can iterate through HashSet by getting Iterator object. By calling iterator() method, you can get Iterator object.

Code:

```
import java.util.HashSet;
import java.util.Iterator;

public class MyHashSetRead {

	public static void main(String a[]){
		HashSet<String> hs = new HashSet<String>();
		//add elements to HashSet
		hs.add("first");
		hs.add("second");
		hs.add("third");
		Iterator<String> itr = hs.iterator();
		while(itr.hasNext()){
			System.out.println(itr.next());
		}
	}
}
```

Output:
second
third
first

Below example shows how to copy another collection object to HashSet object. By calling addAll() method you can copy another collection to HashSet object.

Code:

```
import java.util.HashSet;
public class MyHashSetCopy {

   public static void main(String a[]){
      HashSet<String> hs = new HashSet<String>();
      //add elements to HashSet
```

```
      hs.add("first");
      hs.add("second");
      hs.add("third");
      System.out.println(hs);
      HashSet<String> subSet = new HashSet<String>();
      subSet.add("s1");
      subSet.add("s2");
      hs.addAll(subSet);
      System.out.println("HashSet content after adding another collection:");
      System.out.println(hs);
   }
}
```

Output:
[second, third, first]
HashSet content after adding another collection:
[s2, s1, second, third, first]

Below example shows how to copy all elements from HashSet to an array. By calling toArray() method and passing existing array object to this method, we can copy all elements of HashSet to an array.

```
import java.util.HashSet;
public class MyHashSetToArray {

   public static void main(String a[]){
      HashSet<String> hs = new HashSet<String>();
      //add elements to HashSet
      hs.add("first");
      hs.add("second");
      hs.add("third");
      System.out.println("HashSet content: ");
      System.out.println(hs);
      String[] strArr = new String[hs.size()];
      hs.toArray(strArr);
      System.out.println("Copied array content:");
      for(String str:strArr){
         System.out.println(str);
      }
   }
}
```

Output:
HashSet content:

[second, third, first]
Copied array content:
second
third
first

Below example shows how to compare two sets, and retain the values which are common on both set objects. By calling retainAll() method you can do this operation.

```
import java.util.HashSet;

public class MyHashSetRetain {

   public static void main(String a[]){
      HashSet<String> hs = new HashSet<String>();
      //add elements to HashSet
      hs.add("first");
      hs.add("second");
      hs.add("third");
      hs.add("apple");
      hs.add("rat");
      System.out.println(hs);
      HashSet<String> subSet = new HashSet<String>();
      subSet.add("rat");
      subSet.add("second");
      subSet.add("first");
      hs.retainAll(subSet);
      System.out.println("HashSet content:");
      System.out.println(hs);
   }
}
```

Output:
[second, apple, rat, third, first]
HashSet content:
[second, rat, first]

_____

Below example shows how to avoid duplicate user defined objects from HashSet. You can achieve this by implementing equals and hashcode methods at the user defined objects.

```
import java.util.HashSet;
```

```
public class MyDistElementEx {

   public static void main(String a[]){

      HashSet<Price> lhm = new HashSet<Price>();
      lhm.add(new Price("Banana", 20));
      lhm.add(new Price("Apple", 40));
      lhm.add(new Price("Orange", 30));
      for(Price pr:lhm){
         System.out.println(pr);
      }
      Price duplicate = new Price("Banana", 20);
      System.out.println("inserting duplicate object...");
      lhm.add(duplicate);
      System.out.println("After insertion:");
      for(Price pr:lhm){
         System.out.println(pr);
      }
   }
}

class Price{

   private String item;
   private int price;

   public Price(String itm, int pr){
      this.item = itm;
      this.price = pr;
   }

   public int hashCode(){
      System.out.println("In hashcode");
      int hashcode = 0;
      hashcode = price*20;
      hashcode += item.hashCode();
      return hashcode;
   }

   public boolean equals(Object obj){
      System.out.println("In equals");
      if (obj instanceof Price) {
         Price pp = (Price) obj;
         return (pp.item.equals(this.item) && pp.price == this.price);
```

```
        } else {
            return false;
        }
    }

    public String getItem() {
        return item;
    }
    public void setItem(String item) {
        this.item = item;
    }
    public int getPrice() {
        return price;
    }
    public void setPrice(int price) {
        this.price = price;
    }

    public String toString(){
        return "item: "+item+"  price: "+price;
    }
}
```

Output:

```
In hashcode
In hashcode
In hashcode
item: Apple  price: 40
item: Orange  price: 30
item: Banana  price: 20
inserting duplicate object...
In hashcode
In equals
After insertion:
item: Apple  price: 40
item: Orange  price: 30
item: Banana  price: 20
```

Below example shows how to search user defined objects from HashSet. You can achieve this by implementing equals and hashcode methods at the user defined objects.

```
import java.util.HashSet;
```

```
public class MyHashSetSearchObject {

   public static void main(String a[]){

      HashSet<Price> lhs = new HashSet<Price>();
      lhs.add(new Price("Banana", 20));
      lhs.add(new Price("Apple", 40));
      lhs.add(new Price("Orange", 30));
      for(Price pr:lhs){
         System.out.println(pr);
      }
      Price key = new Price("Banana", 20);
      System.out.println("Does set contains key? "+lhs.contains(key));
   }
}

class Price{

   private String item;
   private int price;

   public Price(String itm, int pr){
      this.item = itm;
      this.price = pr;
   }

   public int hashCode(){
      System.out.println("In hashcode");
      int hashcode = 0;
      hashcode = price*20;
      hashcode += item.hashCode();
      return hashcode;
   }

   public boolean equals(Object obj){
      System.out.println("In equals");
      if (obj instanceof Price) {
         Price pp = (Price) obj;
         return (pp.item.equals(this.item) && pp.price == this.price);
      } else {
         return false;
      }
   }
```

```
    public String getItem() {
        return item;
    }
    public void setItem(String item) {
        this.item = item;
    }
    public int getPrice() {
        return price;
    }
    public void setPrice(int price) {
        this.price = price;
    }

    public String toString(){
        return "item: "+item+"  price: "+price;
    }
}
```

Output:

```
In hashcode
In hashcode
In hashcode
item: Apple  price: 40
item: Orange  price: 30
item: Banana  price: 20
In hashcode
In equals
Does set contains key? True
```

Below example shows how to delete user defined objects from HashSet. You can achieve this by implementing equals and hashcode methods at the user defined objects.

```
import java.util.HashSet;

public class MylhsDeleteObject {

    public static void main(String a[]){

        HashSet<Price> lhs = new HashSet<Price>();
        lhs.add(new Price("Banana", 20));
        lhs.add(new Price("Apple", 40));
        lhs.add(new Price("Orange", 30));
        for(Price pr:lhs){
```

```
            System.out.println(pr);
        }
        Price key = new Price("Banana", 20);
        System.out.println("deleting key from set...");
        lhs.remove(key);
        System.out.println("Elements after delete:");
        for(Price pr:lhs){
            System.out.println(pr);
        }
    }
}

class Price{

    private String item;
    private int price;

    public Price(String itm, int pr){
        this.item = itm;
        this.price = pr;
    }

    public int hashCode(){
        System.out.println("In hashcode");
        int hashcode = 0;
        hashcode = price*20;
        hashcode += item.hashCode();
        return hashcode;
    }

    public boolean equals(Object obj){
        System.out.println("In equals");
        if (obj instanceof Price) {
            Price pp = (Price) obj;
            return (pp.item.equals(this.item) && pp.price == this.price);
        } else {
            return false;
        }
    }

    public String getItem() {
        return item;
    }
    public void setItem(String item) {
```

```
        this.item = item;
    }
    public int getPrice() {
        return price;
    }
    public void setPrice(int price) {
        this.price = price;
    }

    public String toString(){
        return "item: "+item+"  price: "+price;
    }
}
```

Output:

```
In hashcode
In hashcode
In hashcode
item: Banana  price: 20
item: Apple  price: 40
item: Orange  price: 30
deleting key from set...
In hashcode
In equals
Elements after delete:
item: Apple  price: 40
item: Orange  price: 30
```