CSCI 2120:
Software Design & Development II

UNIT 2: Collections Framework & Generics

Iterate on Maps

Overview

1. Introduction
2. Iterate Map using Iterator over Map.Entry<K,V>
3. Example 1: Iterator on Map
4. Example 2: Remove Entry with Iterator on Map
5. Example 3: Iterate over keys or values using keySet() and value()
6. Example 4: Iterate Map using Map.Entry<K,V> method
7. Example 5: Iterate Map using forEach() method

Introduction

We learned in the previous lecture that a map in java is a container object that stores elements in the form of key and value pairs. If the key is provided, its associated value can be retrieved easily. Keys must be unique.

In this lecture we will explore **how to iterate maps in Java**. There are multiple convenient ways to iterate over a map. We can iterate over keys, values, or both. We can also remove an element from a map while iterating over it.

Iterate Map using Iterator over Map.Entry<K, V>

An iterator is an interface that contains methods to retrieve the entries of a map one by one. It provides three methods:

- **boolean hasNext ()**: It returns true if the iterator has more elements.
- **element next ()**: It returns the next element in the iterator.
- **void remove ()**: It removes the last element returned by the iterator.

Note that we cannot iterate over a map directly using iterator because Map interface is not the part of the Collection interface. To iterate map using iterator, you must familiar with Map.Entry<K, V> interface.

Since all maps in Java implement `Map` interface, this technique can be used with any map implementations such as `HashMap`, `TreeMap`, `LinkedHashMap`, `Hashtable`, etc.

Example 1: Iterator on Map

Let’s take an example program where we will iterate map in java using iterator concept.

Example 1: Iterator on Map

```
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
import java.util.Map.Entry;
public class IterateMapTester1 {
   public static void main(String[] args) {
       Map<String, String> map = new HashMap<>(); // Creating a map.

       map.put("V ", " Violet");
       map.put("I ", " Indigo");
       map.put("B ", " Blue");
       map.put("G ", " Green");
       map.put("Y ", " Yellow");
       map.put("O ", " Orange");
       map.put("R ", " Red");

       Iterator<Entry<String, String>> itr = map.entrySet().iterator(); // entrySet is used to get view of entries of a map.
       System.out.println("Iterating Entries of Map");
       while(itr.hasNext()) {
           Object key = itr.next();
           System.out.println(key);
       }
       Iterator<String> itr2 = map.keySet().iterator(); // keySet is a method that is used to get view of keys of a map.
       System.out.println("Iterating Keys of Map");
       while(itr2.hasNext()) {
           Object keyView = itr2.next();
           System.out.println(keyView);
       }
       Iterator<String> itr3 = map.values().iterator(); // values is a method that is used to get values of keys of a map.
       System.out.println("Iterating Values of Map");
       while(itr3.hasNext()) {
           Object valuesView = itr3.next();
           System.out.println(valuesView);
       }
   }
}
```

Example 1: Iterator on Map

Output:

```
V
B
R
Iterating Values of Map
 Orange
 Indigo
 Yellow
 Green
 Violet
 Blue
 Red
```

Example 2: Remove Entry with Iterator on Map

Let’s create another program where we will remove the last entry of a map returned by the Iterator.

Example 2: Remove Entry with Iterator on Map

```
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
import java.util.Map.Entry;
public class IterateMapTester2 {
   public static void main(String[] args) {
       Map<String, String> map = new HashMap<>();

       map.put("V ", " Violet");
       map.put("I ", " Indigo");
       map.put("B ", " Blue");
       map.put("G ", " Green");
       map.put("Y ", " Yellow");
       map.put("O ", " Orange");
       map.put("R ", " Red");

       // entrySet is used to get view of entries of a map.
       Iterator<Entry<String, String>> itr = map.entrySet().iterator();
       System.out.println("Iterating Entries of Map");
       while(itr.hasNext())
       {
           Object key = itr.next();
           System.out.println(key);
       }
       // Removing last entry returned by Iterator
       itr.remove(); // removes last entry of a map while iterating.
       System.out.println("Entries of Map after removing: " +map.entrySet());
   }
}
```

Example 2: Remove Entry with Iterator on Map

Output:

```
Iterating Entries of Map
O = Orange
I = Indigo
Y = Yellow
G = Green
V = Violet
B = Blue
R = Red
Entries of Map after removing: [O = Orange, I = Indigo, Y = Yellow, G = Green, V = Violet, B = Blue]
```

Example 3: Iterate over keys or values using keySet() and value()

This technique is useful when you want to get a set view of keys or values of a map. Using `keySet()`, `values()`, and for-each loop, you can iterate over keys or values of a map.

Let's take an example program where we will iterate over keys or values of a map using `keySet()` and `values()` methods. `keySet()` method returns a set view of the keys from a map and `values()` method returns a collection-view of values from a map.

Example 3: Iterate over keys or values using keySet() and value()

```
import java.util.HashMap;
import java.util.Map;
public class IterateMapTester3 {
   public static void main(String[] args) {
       Map<Integer, String> map = new HashMap<>();

       map.put(101, " John");
       map.put(202, " Ricky");
       map.put(303, " Deep");
       map.put(404, " Mark");
       map.put(505, " Maya");

       // Iterating over keys of a map using keySet() method.
       for (Integer rollNo : map.keySet())
           System.out.println("Roll No: " + rollNo);
       System.out.println();

       // Iterating over values of a map using values() method.
       for (String name : map.values())
           System.out.println("Name: " + name);
   }
}
```

Example 3: Iterate over keys or values using keySet() and value()

Output:

```
Roll No: 404
Roll No: 101
Roll No: 505
Roll No: 202
Roll No: 303

Name:  Mark
Name:  John
Name:  Maya
Name:  Ricky
Name:  Deep
```

Example 4: Iterate Map using Map.Entry<K,V> method

`Map.Entry<K,V>` is an interface that is used to work on an entry in the map. It returns a collection view of the map. Each `Map.Entry` object contains one key/value pair.

Let's create a program where we will iterate entry of a map using `Map.Entry<K,V>` method.

We will use the following methods for iteration.

1. **getKey()**: It is used to retrieve the key for a map entry. Its return type is key.
2. **getValue()**: It is used to get the value for a map entry. Its return type is value.
3. **entrySet()**: It returns a set view of entries of a map.

Example 4: Iterate Map using Map.Entry<K,V> method

```
import java.util.HashMap;
import java.util.Map;
public class IterateMapTester4 {
   public static void main(String[] args) {
       Map<Integer, String> map = new HashMap<>();

       map.put(101, " John");
       map.put(202, " Ricky");
       map.put(303, " Deep");
       map.put(404, " Mark");
       map.put(505, " Maya");

       // Iterating over entries of a map using entrySet() method
       for (Map.Entry<Integer,String> entry : map.entrySet()) {
           int key = entry.getKey();
           String value = entry.getValue();
           System.out.println("Roll No: " + key + ", Name: " + value);
       }
   }
}
```

Example 4: Iterate Map using Map.Entry<K,V> method

Output:

```
Roll No: 404, Name:  Mark
Roll No: 101, Name:  John
Roll No: 505, Name:  Maya
Roll No: 202, Name:  Ricky
Roll No: 303, Name:  Deep
```

Example 5: Iterate Map using forEach() method

The `forEach()` method was added to the `iterable` interface with the release of JDK 1.8. It is a default method defined in the `java.lang.Iterable` interface.

This method performs the specified action for each element within `Iterable` interface until all the elements have been processed. The collection classes which extend the `Iterable` interface can use `forEach()` method to iterate elements of collection objects.

The `forEach()` method takes action to be performed for each element as a parameter. If the specified action is null, it will throw `NullPointerException`. It does not return anything.

Let's take an example program based on the `forEach()` method. In this program, we will use lambda expression inside the `forEach()` method to display each entry of the map.

Example 5: Iterate Map using forEach() method

```
import java.util.HashMap;
import java.util.Map;
public class IterateMapTester5 {
   public static void main(String[] args) {
       Map<String, String> map = new HashMap<>();

       map.put("India", " Delhi");
       map.put("USA", " Washington, D.C.");
       map.put("Australia", " Canberra");
       map.put("New zealand", " Wellington");
       map.put("Switzerland", " Bern");

       // Iteration over map using forEach() method.
       map.forEach( (k,v) -> System.out.println("Country: "+ k + ", Capital: " + v) );
   }
}
```

Example 5: Iterate Map using forEach() method

Output:

```
Country: USA, Capital:  Washington, D.C.
Country: New zealand, Capital:  Wellington
Country: Australia, Capital:  Canberra
Country: Switzerland, Capital:  Bern
Country: India, Capital:  Delhi
```

END