JAC444 - BTP400 Course Object-Oriented Software Development II - Java

Collections

Segment 4

The Map Interface

The <u>Map<K,V></u> Interface

A `Map` is an object that maps keys to values.

<u>A map cannot contain duplicate keys</u>. - The collection of keys is a set

<u>Each key can map to at most one value</u> – Mathematical *function abstraction*

<u>Implementations:</u>

**<u>HashMap</u>** – Hash table and the constant-time implementation

<u>TreeMap</u> – The map is sorting according to the natural ordering of its keys

<u>LinkedHashMap</u> – Hash table with linked list implementation of `Map` interface

The Map<K,V> Interface

```java
public interface Map<K,V> {
    // Basic Operations
    Object put(K key, V value);
    V get(Object key);
    Object remove(Object key);
    boolean containsKey(Object key);
    boolean containsValue(Object value);
    int size();
    boolean isEmpty();

    // Bulk Operations
    void putAll(Map<? extends K, ? extends V> m);
    void clear();

    // Collection Views
    public Set<K> keySet();
    public Collection<V> values();
    public Set<Map.Entry<K,VL> entrySet();

    // Interface for entrySet elements
    public interface Entry<K,V> {
        K getKey();
        V getValue();
        V setValue(V value);
    }
}
```

1 Basic

2 Bulk

3 View

4 Entry Interface
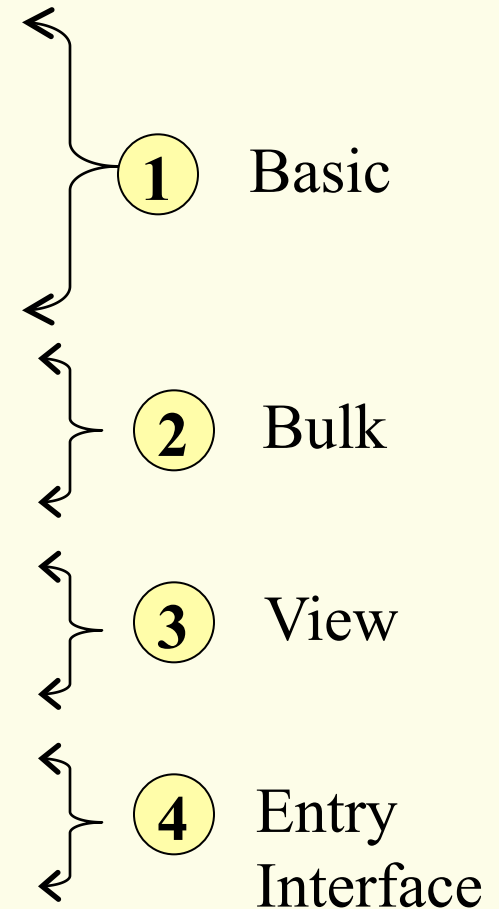
Collection Views of Map<K,V>

The `Collection` view methods allow a Map to be viewed as a `Collection` in these three ways:

`keySet` — the `Set` of keys contained in the `Map`

`values` — the `Collection` of values contained in the `Map`

`entrySet` — the `Set` of key-value pairs contained in the `Map`.
The `Map` interface has a nested interface called `Map.Entry`

The standard `Map` conversion constructor. If there is an object m of type `Map`

```
Map<K, V> copy = new HashMap<K, V>(m)
```

creates an object `copy` of type `HashMap` that contains the same key-value as `m`

Basic Map Operations

```java
import java.util.*;

public class Rate {

  public static void main(String[] args) {
    Map<String, Integer> m = new HashMap<>();

    for (String key : args) {
      Integer val = m.get(key);
      Integer newVal = (val == null) ? 1 : val + 1;
      m.put(key, newVal);
    }

    for (Map.Entry<String, Integer> e : m.entrySet())
      System.out.println(e.getKey() + "---> " + e.getValue());
  }
}
```

Common Idioms for <u>Map<K,V></u>

How to check if two maps objects `m1` and `m2` have the same keys:
Answer: `m1.keySet().equals(m2.keySet()`

How to find the common keys of two maps objects `m1` and `m2`
Answer: `Set<K> commonKeys = new HashSet<K>(m1.keySet());`
`commonKeys.retainAll(m2.keySet());`

Iterating over key-value pairs:

```
for (Map.Entry<K, V> e : m.entrySet())
  System.out.println(e.getKey() + ": " + e.getValue());
```