

非贷款，0元入学，不1万就业不给1分钱学费，我们已于四年了！

笔记总链接：<http://bbs.itheima.com/thread-200600-1-1.html>

6、集合

6.2 集合类

6.2.5 Map、HashMap、TreeMap

Map：一次添加一对元素，Collection一次添加一个元素。

Map也称为双列集合，Collection集合称为单列集合。

其实Map集合中存储的就是键值对。

map集合中必须保证键的唯一性。

P.S. 关于泛型相关的知识请参看Java高级知识笔记：<http://bbs.itheima.com/thread-107670-1-1.html>.

常用方法：

1、添加

value put(key,value)返回的一个和key关联的值，如果没有返回null。

2、删除

void clear()清空map集合。

value remove(Object key)根据指定的key删除这个键值对。

3、判断

boolean containsKey(key);

boolean containsValue(value);

boolean isEmpty();

4、获取

value getKey()通过键获取值，如果没有该键返回null。

当然可以通过返回null，来判断是否包含指定键。

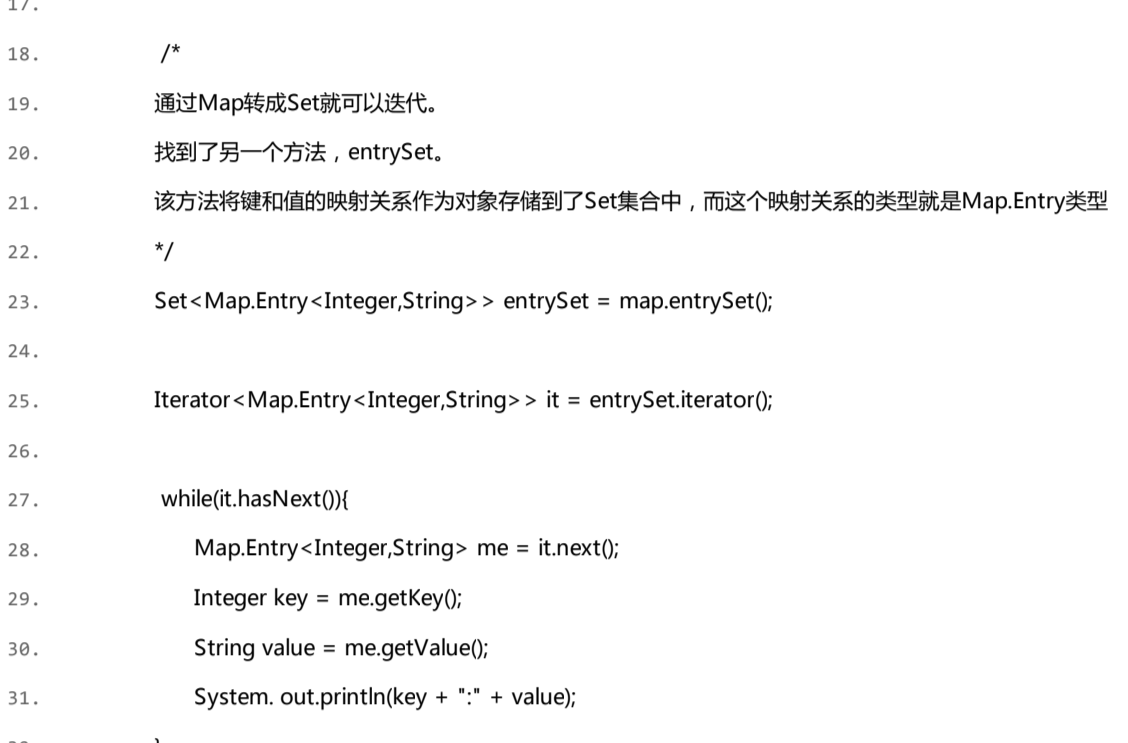
int size()获取键值对个数。

示例1：

```
01. import java.util.HashMap;
02. import java.util.Map;
03.
04. public class MapDemo{
05.     public static void main(String[] args){
06.         Map<Integer,String> map = new HashMap<Integer,String>();
07.         method(map);
08.     }
09.
10.     public static void method(Map<Integer,String> map){ //序号和姓名
11.         //添加元素
12.         System.out.println(map.put(8,"旺财"));
13.         System.out.println(map.put(8,"小强"));
14.         System.out.println(map);
15.
16.         map.put(2,"张三");
17.         map.put(7,"赵六");
18.         System.out.println(map);
19.
20.         //删除
21.         System.out.println("remove" + map.remove(2));
22.
23.         //判断
24.         System.out.println("containsKey" + map.containsKey(7));
25.
26.         //获取
27.         System.out.println("get" + map.get(7));
28.     }
29. }
```

复制代码

运行结果：



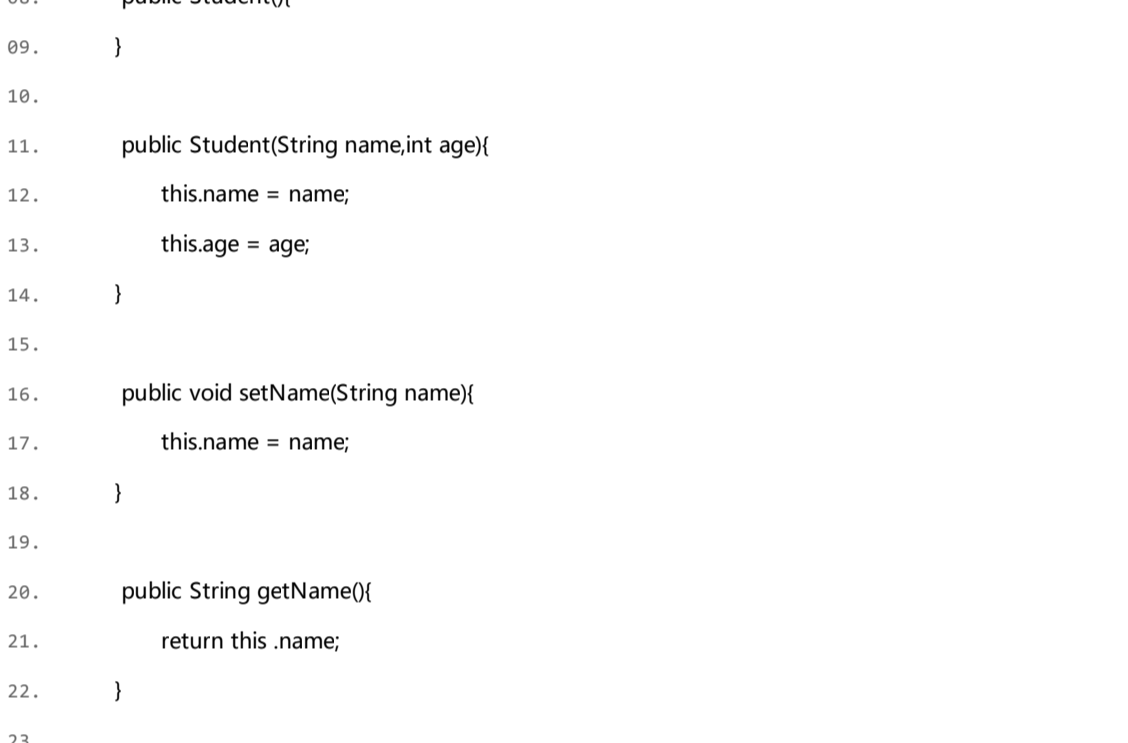
获取Map集合元素并打印方式一：

示例2：

```
01. import java.util.HashMap;
02. import java.util.Iterator;
03. import java.util.Map;
04. import java.util.Set;
05.
06. public class MapDemo{
07.     public static void main(String[] args){
08.         Map<Integer,String> map = new HashMap<Integer,String>();
09.         method(map);
10.     }
11.
12.     public static void method(Map<Integer,String> map){
13.         map.put(8,"旺财");
14.         map.put(2,"赵六");
15.         map.put(7,"小强");
16.         map.put(6,"旺财");
17.
18.         //取出map中的所有元素。
19.         //原理：通过keySet方法获取map中所有的键所在的set集合，在通过set的迭代器获取到每一个键。
20.         //再对每一个键通过map集合的get方法获取其对应的值即可。
21.
22.         Set<Integer> keySet = map.keySet();
23.         Iterator<Integer> it = keySet.iterator();
24.
25.         while(it.hasNext()){
26.             Integer key = it.next();
27.             String value = map.get(key);
28.             System.out.println(key + ":" + value);
29.         }
30.     }
31. }
```

复制代码

运行结果：



获取Map集合元素并打印方式二：

示例3：

```
01. import java.util.HashMap;
02. import java.util.Iterator;
03. import java.util.Map;
04. import java.util.Set;
05.
06. public class MapDemo{
07.     public static void main(String[] args){
08.         Map<Integer,String> map = new HashMap<Integer,String>();
09.         method(map);
10.     }
11.
12.     public static void method(Map<Integer,String> map){
13.         map.put(8,"王五");
14.         map.put(2,"赵六");
15.         map.put(7,"小强");
16.         map.put(6,"旺财");
17.
18.         /*
19.          * 通过Map转成Set就可以迭代。
20.          * 找到了另一个方法，entrySet,
21.          * 该方法将键和值的映射关系作为对象存储到了Set集合中，而这个映射关系的类型就是Map.Entry类型
22.          */
23.         Set<Map.Entry<Integer,String>> entrySet = map.entrySet();
24.
25.         Iterator<Map.Entry<Integer,String>> it = entrySet.iterator();
26.
27.         while(it.hasNext()){
28.             Map.Entry<Integer,String> me = it.next();
29.             Integer key = me.getKey();
30.             String value = me.getValue();
31.             System.out.println(key + ":" + value);
32.         }
33.     }
34. }
```

复制代码

运行结果：



获取Map集合元素并打印方式三：

示例4：

```
01. import java.util.Collection;
02. import java.util.Iterator;
03. import java.util.Map;
04. import java.util.Set;
05.
06. public class MapDemo{
07.     public static void main(String[] args){
08.         Map<Integer,String> map = new HashMap<Integer,String>();
09.         method(map);
10.     }
11.
12.     public static void method(Map<Integer,String> map){
13.         map.put(8,"王五");
14.         map.put(2,"赵六");
15.         map.put(7,"小强");
16.         map.put(6,"旺财");
17.
18.         Collection<String> values = map.values();
19.
20.         Iterator<String> it = values.iterator();
21.         while(it.hasNext()){
22.             System.out.println(it.next());
23.         }
24.     }
25. }
```

复制代码

运行结果：



Map常用的子类：

- [-]HashMap：内部结构是哈希表，是同步的。不允许null作为键，null作为值。
- [-]Properties：用来存储键值对类型的配置文件的信息，可以和IO技术相结合。
- [-]HashMap：内部结构是哈希表，是同步的。允许null作为键，null作为值。
- [-]TreeMap：内部结构是二叉树，不是同步的。可以对Map结合中的键进行排序。

hashSet实现Set接口，由哈希表（实际上是一个HashMap实例）支持。

示例5：

```
01. import java.util.HashMap;
02. import java.util.Iterator;
03.
04. class Student {
05.     private String name;
06.     private int age;
07.
08.     public Student(){
09.     }
10.
11.     public Student(String name,int age){
12.         this.name = name;
13.         this.age = age;
14.     }
15.
16.     public void setName(String name){
17.         this.name = name;
18.     }
19.
20.     public String getName(){
21.         return this.name;
22.     }
23.
24.     public void setAge(int age){
25.         this.age = age;
26.     }
27.
28.     public int getAge(){
29.         return this.age;
30.     }
31.
32.     public int hashCode() {
33.         final int prime = 31;
34.         int result = 1;
35.         result = prime * result + age;
36.         result = prime * result + ((name == null) ? 0 : name.hashCode());
37.         return result;
38.     }
39.
40.     public boolean equals(Object obj) {
41.         if (this == obj)
42.             return true;
43.         if (obj == null)
44.             return false;
45.         if (getClass() != obj.getClass())
46.             return false;
47.         Student other = (Student) obj;
48.         if (age != other.age)
49.             return false;
50.         if (name == null) {
51.             if (other.name != null)
52.                 return false;
53.         } else if (!name.equals(other.name))
54.             return false;
55.         return true;
56.     }
57. }
58.
59. public class HashMapDemo{
60.     public static void main(String[] args){
61.         //将学生和学生的归属地通过键与值存储到map集合中
62.         HashMap<Student,String> hm = new HashMap<Student,String>();
63.
64.         hm.put(new Student("lisi",38),"北京");
65.         hm.put(new Student("zhaoliu",24),"上海");
66.         hm.put(new Student("xiaoliang",31),"沈阳");
67.         hm.put(new Student("wangcai",28),"大连");
68.         hm.put(new Student("zhaoliu",24),"铁岭");
69.
70.         Iterator<Student> it = hm.keySet().iterator();
71.
72.         while(it.hasNext()){
73.             Student key = it.next();
74.             String value = hm.get(key);
75.             System.out.println(key.getName() + ":" + key.getAge() + "----" + value);
76.         }
77.     }
78. }
```

复制代码

运行结果：



~END~



~登上海，登黑马~

