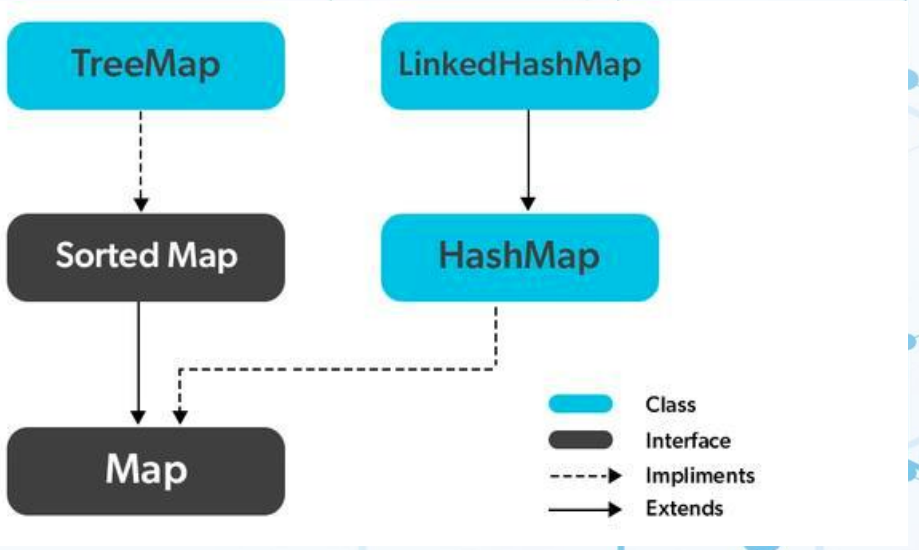
Map và TreeMap

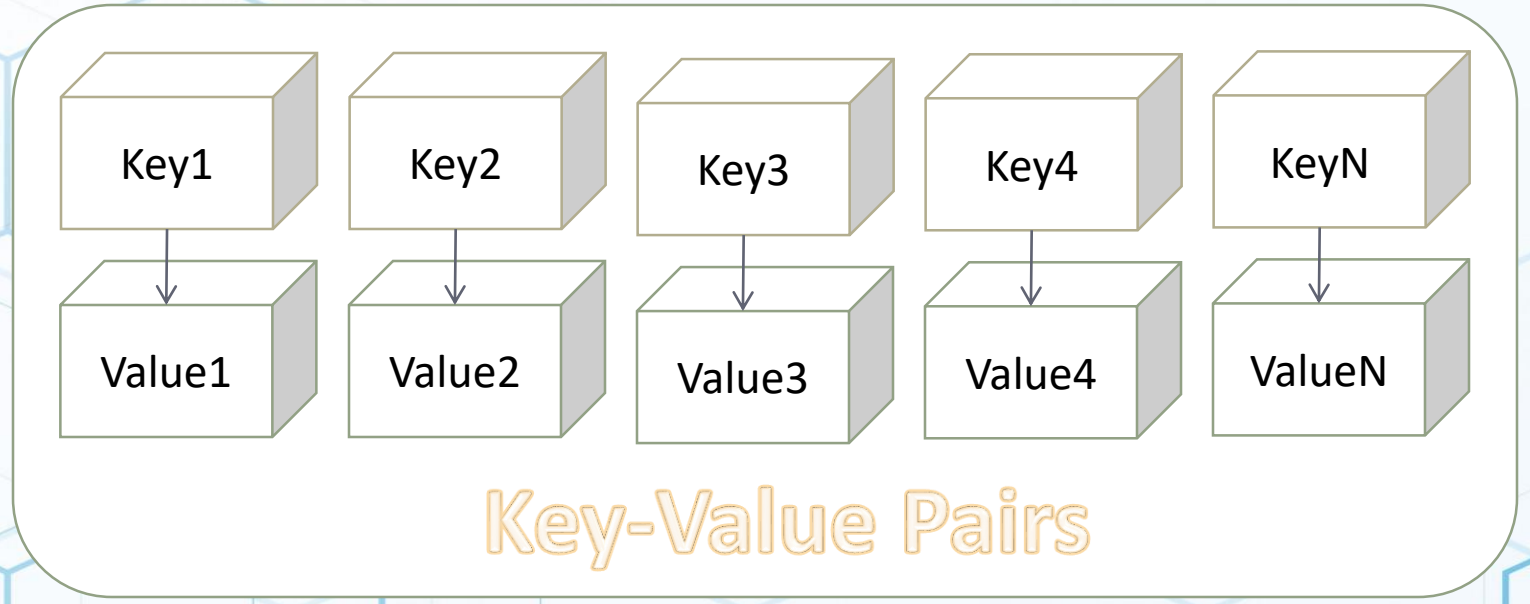
Map interface

- Là interface có trong java.util biểu thị một tham chiếu giữa một key (khóa) và một value (giá trị).
- Map interface không phải là kiểu con (subtype) của Collection interface. Do đó Map khác một chút so với phần còn lại của các kiểu collection.



Map

- Map là tập hợp các entry.
- Mỗi entry gồm key và value
- Sử dụng key để truy xuất giá trị của mỗi phần tử



Ví dụ Map


```
// Khai báo tập hợp các ánh xạ giữa chuỗi và số thực
Map<String, Double> map = new HashMap<String, Double>();
// bổ sung 4 cặp vào tập hợp
map.put("Nokia", 500.0);
map.put("Samsung", 600.99);
map.put("Motorola", 399.99);
map.put("Sony Ericson", 400.50);
// cập nhật giá trị của phần tử có khóa là Samsung
map.put("Samsung", 555.55);
// chuyển sang chuỗi và xuất ra
System.out.print(map.toString());
```

{Motorola=399.990, Nokia=500.000, Sony Ericson=400.500,Samsung=555.550}

Map API

Phương thức	Mô tả
Object put(Object key, Object value)	Bổ sung hoặc cập nhật một entry
Object get(Object key)	Lấy value theo key
Object remove(Object key)	Xóa một phần tử theo key
boolean containsKey(Object key)	Kiểm tra sự tồn tại entry theo key
int size()	Lấy số lượng entry
boolean isEmpty()	Kiểm tra có rỗng hay không
void clear()	Xoá sạch các entry.
Set keySet()	Lấy tập key
Collection values()	Lấy tập value
Set entrySet()	Lấy tập entry

Duyệt Map


```
Set<String> keys = map.keySet();
for(String key: keys){
    Double diem = map.get(key);
}
```

```
for(Entry<String, Double> entry : map.entrySet()){
     String ten = entry.getKey();
   double diem = entry.getValue();
}
```

TreeMap

- Là lớp được sử dụng để triển khai (implement) Map interface và NavigableMap. Do đó lưu trữ dữ liệu dưới dạng key - value
- Chứa các key duy nhất (không trùng lặp)
- Không cho phép key nào NULL nhưng có thể có nhiều value NULL
- Dữ liệu được sắp xếp dựa vào thứ tự tự nhiên của các key, hoặc bằng Comparator (bộ so sánh) cung cấp tại thời điểm tạo Map.

Khác biệt giữa HashMap và TreeMap

HashMap	TreeMap
HashMap có thể chứa một key là NULL	TreeMap không thể chứa bất kì key NULL nào
HashMap duy trì các phần tử không theo thứ tự	TreeMap duy trì các phần tử theo thứ tự key tăng dần.

Các constructor của lớp TreeMap

Constructor	Mô tả
TreeMap()	Khởi tạo một TreeMap trống
TreeMap(Comparator comparator)	Khởi tạo một TreeMap trống, được sắp xếp theo comparator đã cho
TreeMap(Map m)	Khởi tạo một TreeMap chứa các phần tử được copy từ map đã cho, được sắp xếp theo key một cách tự nhiên.
TreeMap(SortedMap m)	Khởi tạo một TreeMap chứa các phần tử được copy từ map đã cho, được sắp xếp theo SortedMap đã chỉ định.

Một số phương thức của lớp TreeMap

Phương thức	Mô tả
void clear()	Xóa tất cả các phần tử của TreeMap.
Object clone()	Trả về một bản copy của TreeMap.
boolean containsKey(Object key)	Trả về true nếu TreeMap chứa một phần tử có key được chỉ định.
boolean containsValue(Object value)	Trả về true nếu TreeMap chứa một phần tử có giá trị (value) được chỉ định.
boolean isEmpty()	Trả về true nếu TreeMap trống.
Object put(Object key, Object value)	Thêm một cặp key-value vào TreeMap.

Một số phương thức của lớp TreeMap

Phương thức	Mô tả
void putAll(Map t)	Sao chép các phần tử của Map được chỉ định vào TreeMap.
Collection values()	Trả về Collection của các giá trị có trong TreeMap.
Object firstKey()	Trả về key đầu tiên của map đã được sắp xếp.
Object lastKey()	Trả về key cuối cùng của map đã được sắp xếp.
Collection values()	Trả về Collection của các giá trị có trong TreeMap.
SortedMap subMap((K startKey, K endKey)	Trả về một phần của TreeMap bắt đầu từ phần tử có key startKey đến phần tử có key endKey.

Fan page: http://facebook.com/Proit4All

DEMO

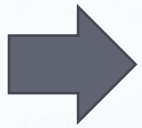
TreeMap

Java

Demo1: TreeMap

Khởi tạo và thêm phần tử vào map:

```
Map<String,String> map1= new TreeMap<String,String>();
        map1.put("J", "Java");
        map1.put("C", "C++");
        map1.put("P", "PHP");
        //hien thi map
        System.out.println(map1);
```

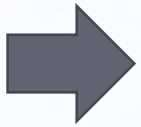


```
{C=C++, J=Java, P=PHP}
```

Demo2: TreeMap

Duyệt các phần tử trong một Map sử dụng For

```
public static void main(String[] args) {
        Map<String,String> map1= new TreeMap<String,String>();
        map1.put("J", "Java");
        map1.put("C", "C++");
        map1.put("P", "PHP");
        //hien thi map
        System.out.println(map1);
        show(map1);
    }
    public static void  show(Map<String, String> map){
        Set<String> keySet= map.keySet();
        for(String key: keySet){
            System.out.println(key +"-"+map.get(key));
        }
    }
```



```
{C=C++, J=Java, P=PHP}
C-C++
J-Java
P-PHP
```

Demo3: TreeMap

Duyệt các phần tử trong Map sử dụng Map.Entry interface:

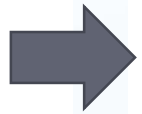
```
public static void main(String[] args) {
    Map<String,String> map1= new TreeMap<String,String>();
    map1.put("J", "Java");
    map1.put("C", "C++");
    map1.put("P", "PHP");
    //hien thi map
    System.out.println(map1);
    show(map1);
    for(Map.Entry<String, String> entry: map1.entrySet()){
        System.out.println(entry.getKey()+"-"+entry.getValue());
    }
}
public static void  show(Map<String, String> map){
    Set<String> keySet= map.keySet();
    for(String key: keySet){
        System.out.println(key +"-"+map.get(key));
    }
}
```

```
{C=C++, J=Java, P=PHP}
C-C++
J-Java
P-PHP
C-C++
J-Java
P-PHP
```

Demo 4: TreeMap

Duyệt các phần tử trong Map Iterator

```
public static void main(String[] args) {
    Map<String,String> map1= new TreeMap<String,String>();
    map1.put("J", "Java");
    map1.put("C", "C++");
    map1.put("P", "PHP");
    //hien thi map
    System.out.println(map1);
    Iterator<String> itr = map1.keySet().iterator();
    while(itr.hasNext()){
        System.out.println(map1.get(itr.next()));
    }
}
```

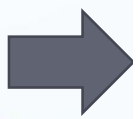


```
{C=C++, J=Java, P=PHP}
C++
Java
PHP
```

Demo 5: TreeMap

Truy cập phần tử của TreeMap

```
public static void main(String[] args) {
    Map<String, String> map1 = new TreeMap<String, String>();
    map1.put("J", "Java");
    map1.put("C", "C++");
    map1.put("P", "PHP");
    System.out.println("Phan tu thu nhat: "+map1.get("J"));
}
```

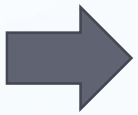


```
Phan tu thu nhat: Java
```

Demo 6: TreeMap

Cập nhật phần tử của TreeMap

```
public static void main(String[] args) {
    Map<String, String> map = new TreeMap<String, String>();
    map.put("J", "Java");
    map.put("C", "C++");
    map.put("P", "PHP");
    System.out.println(map);
    map.put("P", "Angular");
    System.out.println(map);
}
```

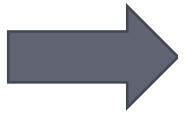


```
{C=C++, J=Java, P=PHP}
{C=C++, J=Java, P=Angular}
```

Demo 7: TreeMap

Xóa phần tử của TreeMap

```
public static void main(String[] args) {
    Map<String, String> map = new TreeMap<String, String>();
    map.put("J", "Java");
    map.put("C", "C++");
    map.put("P", "PHP");
    System.out.println(map);
    map.put("P", "Angular");
    System.out.println(map);
    //xoa phan tu theo key
    map.remove("P");
    System.out.println("map sau khi xoa"+map);
    //xoa toan bo map
    map.clear();
    System.out.println("Map sau khi clear:"+map);
}
```



```
{C=C++, J=Java, P=PHP}
{C=C++, J=Java, P=Angular}
map sau khi xoa{C=C++, J=Java}
Map sau khi clear:{}
```

thuydung ProIT4All

Fan page: http://facebook.com/Proit4All

THANK YOU

http://youtube.com/@AnhNguyenNgoc