**SortedMap Interface trong Java**

**1. Đặc điểm**

SortedMap Interface là 1 dạng riêng của Map Interface nên nó có những đặc điểm của Map đó là SortedMap cũng bao gồm một tập các cặp khóa - giá trị (key - value). Giá trị của các phần tử trong SortedMap có thể giống nhau, nhưng khóa thì không được giống nhau,và dựa vào khóa chúng ta có thể xác định được các giá trị value tương ứng với khóa đó. Ngoài ra,SortedMap có điểm vượt trội hơn so với Map là các entry có trong SortedMap được sắp xếp tăng dần theo khóa**.**

**2. Các phương thức phổ biến**

Vì SortedMap là 1 dạng riêng của Map nên những phương thức của nó sẽ tương tự như những phương thức có trong Map. Trong bài này, tôi sẽ không trình bày lại những phương thức đó mà tôi sẽ giới thiệu những phương thức riêng của SortedMap để cho các bạn tìm hiểu.

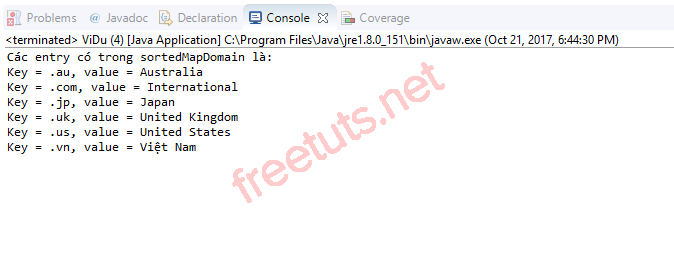
### **2.1. Tạo mới và hiển thị các phần tử của 1 SortedMap**

Để khai báo một SortedMap, chúng ta cần phải dùng đến Class để triển khai nó, trong phần này chúng ta sẽ sử dụng Class là TreeMap bởi vì các entry trong TreeMap được sắp xếp tăng dần theo khóa.

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | public static void main(String[] args) {      // khai báo SortedMap Interface tên sortedMapDomain      // và sử dụng Class là TreeMap để triển khai      // TreeMap là 1 Class Collection      SortedMap<String, String> sortedMapDomain = new TreeMap<>();        // thêm các entry vào trong sortedMapDomain      sortedMapDomain.put(".com", "International");      sortedMapDomain.put(".us", "United States");      sortedMapDomain.put(".uk", "United Kingdom");      sortedMapDomain.put(".jp", "Japan");      sortedMapDomain.put(".au", "Australia");      sortedMapDomain.put(".vn", "Việt Nam");        System.out.println("Các entry có trong sortedMapDomain là: ");      sortedMapDomain.forEach((keyChar, valueInt) -> System.out.println(          "Key = " + keyChar + ", value = " + valueInt));  } |

Sau khi biên dịch chương trình thì các entry có trong sortedMapDomain sẽ được sắp xếp theo chiều tăng dần của khóa. Kết quả như sau:



**Lưu ý:** Để khai báo SortedMap chúng ta cần phải import gói thư viện java.util.SortedMap. Cú pháp import như sau:

**Cú pháp**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | // Khai báo SortedMap  // thì import gói thư viện java.util.SortedMap  import java.util.SortedMap;  public class TênClass {      // ...  } |

### **2.2. Trích xuất một phần trong SortedMap**

Đối với SortedMap, Java cung cấp cho chúng ta các phương thức để tạo 1 SortedMap con từ 1 SortedMap đã tồn tại đó là subMap(), headMap() và tailMap(). Sau đây chúng ta sẽ cùng nhau tìm hiểu về 3 phương thức này.

**Phương thức subMap().**

**Cú pháp**

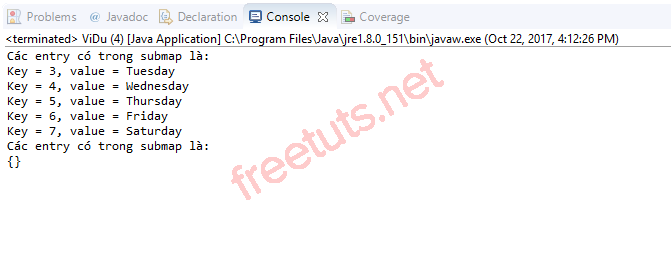
|  |  |
| --- | --- |
| 1 | subMap(K fromKey, K toKey); |

Công dụng: phương thức subMap() sẽ trả về một SortedMap con mà các giá trị key của nó có phạm vi từ fromKey đến toKey (không bao gồm toKey) của một SortedMap cho trước.

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | public static void main(String[] args) {      // khai báo SortedMap Interface tên sortedMapDomain      // và sử dụng Class là TreeMap để triển khai      // TreeMap là 1 Class Collection      SortedMap<Integer, String> sortedMap = new TreeMap<>();        // thêm các entry vào trong sortedMap      sortedMap.put(4, "Wednesday");      sortedMap.put(2, "Monday");      sortedMap.put(3, "Tuesday");      sortedMap.put(8, "Sunday");      sortedMap.put(6, "Friday");      sortedMap.put(5, "Thursday");      sortedMap.put(7, "Saturday");        // khai báo 1 SortedMap có tên là submap      // có key nằm trong đoạn [3, 7) của sortedMap      Map<Integer, String> submap = sortedMap.subMap(3, 8);      System.out.println("Các entry có trong submap là: ");      submap.forEach((keyChar, valueInt) -> System.out.println(          "Key = " + keyChar + ", value = " + valueInt));        // nếu key đầu và key cuối trùng nhau      // thì kết quả của phương thức subMap()      // sẽ trả về submap không có entry nào      submap = sortedMap.subMap(3, 3);      System.out.println("Các entry có trong submap là: ");      System.out.println(submap);  } |

Kết quả sau khi biên dịch chương trình:



**Phương thức headMap().**

**Cú pháp**

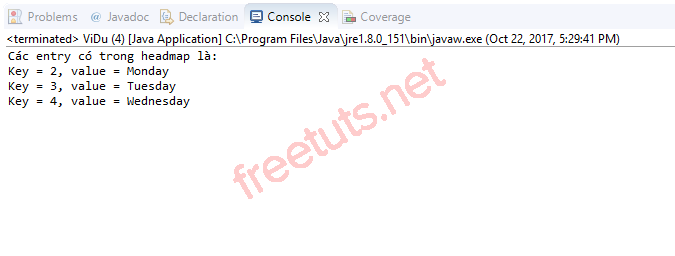
|  |  |
| --- | --- |
| 1 | headMap(K toKey); |

Công dụng: Phương thức headMap() sẽ trả về một SortedMap con mà các key của nó có giá trị nhỏ hơn toKey của một SortedMap cho trước.

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | public static void main(String[] args) {      SortedMap<Integer, String> sortedMap = new TreeMap<>();        // thêm các entry vào trong sortedMap      sortedMap.put(4, "Wednesday");      sortedMap.put(2, "Monday");      sortedMap.put(3, "Tuesday");      sortedMap.put(8, "Sunday");      sortedMap.put(6, "Friday");      sortedMap.put(5, "Thursday");      sortedMap.put(7, "Saturday");        // khai báo 1 SortedMap có tên là headmap      // có giá trị của key nhỏ hơn 5 trong sortedMap      Map<Integer, String> headmap = sortedMap.headMap(5);      System.out.println("Các entry có trong headmap là: ");      headmap.forEach((keyChar, valueInt) -> System.out.println(          "Key = " + keyChar + ", value = " + valueInt));  } |

Kết quả sau khi biên dịch chương trình:



**Phương thức tailMap().**

**Cú pháp**

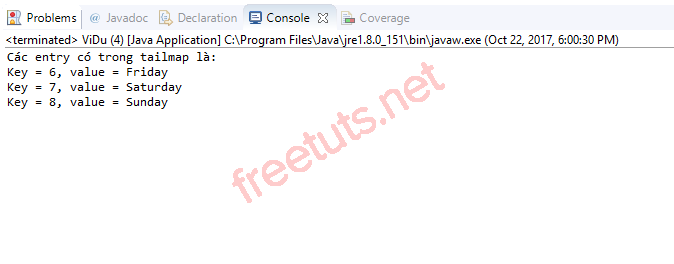
|  |  |
| --- | --- |
| 1 | tailMap(K fromKey); |

Công dụng: Phương thức tailMap() sẽ trả về một SortedMap con mà các key của nó có giá trị lớn hơn hoặc bằng fromKey của một SortedMap cho trước.

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | public static void main(String[] args) {      SortedMap<Integer, String> sortedMap = new TreeMap<>();        // thêm các entry vào trong sortedMap      sortedMap.put(4, "Wednesday");      sortedMap.put(2, "Monday");      sortedMap.put(3, "Tuesday");      sortedMap.put(8, "Sunday");      sortedMap.put(6, "Friday");      sortedMap.put(5, "Thursday");      sortedMap.put(7, "Saturday");        // khai báo 1 SortedMap có tên là tailmap      // có giá trị của key lớn hơn hoặc bằng 6 trong sortedMap      Map<Integer, String> tailmap = sortedMap.tailMap(6);      System.out.println("Các entry có trong tailmap là: ");      tailmap.forEach((keyChar, valueInt) -> System.out.println(          "Key = " + keyChar + ", value = " + valueInt));  } |

Kết quả sau khi biên dịch chương trình:



### **2.3. Tìm giá trị khóa (key) nhỏ nhất và lớn nhất [trong](https://freetuts.net/sortedmap-interface-trong-java-1088.html) SortedMap**

Để tìm giá trị khóa (key) nhỏ nhất (đầu tiên) và lớn nhất (cuối cùng) trong SortedMap, Java cung cấp cho chúng ta 2 phương thức đó là firstKey() (tìm giá trị nhỏ nhất) và lastKey() (tìm giá trị lớn nhất).

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | public static void main(String[] args) {      SortedMap<Double, Float> sortedMap = new TreeMap<>();        // thêm các entry vào trong sortedMap      sortedMap.put(0.2d, 0.1f);      sortedMap.put(6.06d, 10.22f);      sortedMap.put(1.2d, 9.8f);      sortedMap.put(1d, 2f);        // tìm giá trị khóa (key) nhỏ nhất và lớn nhất      // trong sortedMap      // vì key trong SortedMap có kiểu dữ liệu là Double      // nên 2 biến first và last cũng phải có kiểu là Double      Double first = sortedMap.firstKey();      Double last = sortedMap.lastKey();        System.out.println("Key có giá trị nhỏ nhất và lớn nhất"          + " trong sortedMap là " + first + " và " + last);  } |

Kết quả sau khi biên dịch chương trình:

