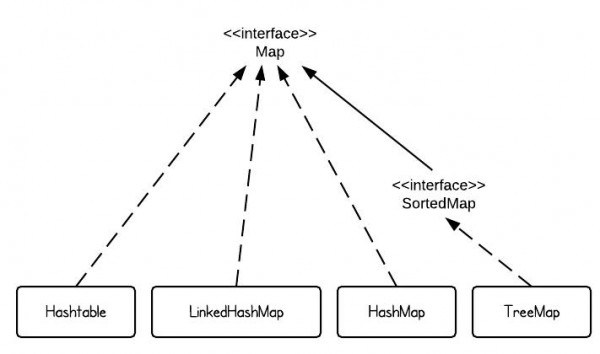
**Map và những thư viện trong Map thuộc gọi java.util**

Map Interface là dùng dể lưu trữ cấu trúc dữ liệu theo dạng key-value. Một đối tượng key sẽ thu nhận “một” đối tượng value.

Trong Map có 4 implementtation là HashMap, TreeMap và LinkedHashMap, HashTable.



Những phương thức chung của Map:

|  |  |
| --- | --- |
| **STT** | **Phương thức và Miêu tả** |
| 1 | **void clear( )**  Gỡ bỏ tất cả cặp key/value từ Map đang gọi |
| 2 | **boolean containsKey(Object k)**  Trả về true nếu Map đang gọi chứa k như là một key. Nếu không là false |
| 3 | **boolean containsValue(Object v)**  Trả về true nếu Map đang gọi chứa v như là một value. Nếu không là false |
| 4 | **Set entrySet( )**  Trả về một Set mà chứa các entry trong Map. Set này chứa các đối tượng của Map.Entry. Phương thức này cung cấp một set-view cho Map đang gọi |
| 5 | **boolean equals(Object obj)**  Trả về true nếu obj là một Map và chứa cùng các Entry. Nếu không là false |
| 6 | **Object get(Object k)**  Trả về value mà liên kết với key k |
| 7 | **int hashCode( )**  Trả về hash code cho Map đang gọi |
| 8 | **boolean isEmpty( )**  Trả về true nếu Map đang gọi là trống, nếu không là false |
| 9 | **Set keySet( )**  Trả về một Set mà chứa các key trong Map đang gọi. Phương thức này cung cấp một set-view của các key trong Map đang gọi |
| 10 | **Object put(Object k, Object v)**  Đặt một entry vào Map đang gọi, ghi đè bất kỳ value trước mà liên kết với key. Với key và value tương ứng là k và v. Trả về null nếu key đã không tồn tại. Nếu không thì, value trước mà liên kết với key được trả về |
| 11 | **void putAll(Map m)**  Đặt tất cả entry từ m vào trong Map này |
| 12 | **Object remove(Object k)**  Gỡ bỏ entry mà có khóa là k |
| 13 | **int size( )**  Trả về số các cặp key/value trong Map |
| 14 | **Collection values( )**  Trả về một tập hợp chứa các value trong Map. Phương thức này cung cấp một collection-view của các value trong Map |

\* Sự khác nhau giữa 3 implement này là :

* HashMap được thực hiện như một bảng băm, và không có thứ tự về các khóa hoặc giá trị.
* TreeMap được thực hiện dựa trên cấu trúc cây màu đỏ-đen và được sắp xếp theo khóa.
* LinkedHashMap giữ nguyên lệnh chèn
* Hashtable được đồng bộ, ngược với HashMap

\* HashMap được thực hiện như một bảng băm và không có thứ tự nên ngoài những phương thức được thừa kế từ MapInterface nó chỉ bổ xung thêm 1 phương thức nữa là clone().

Lưu ý: chúng ta có thể dùng phương thức size để biết số phần tử trong HashMap nhưng do HashMap không cung cấp cho ta phương thức để lấy key và value, cũng như để thuận tiền hơn ta nên mượn 1 Iterator để duyệt qua toàn bộ phần tử trong mảng. Nhưng để tạo được 1 Iterator từ HashMap mà Map lại không cung cấp phương thức để chuyển đổi nên ta phải thực hiện chuyển đổi từ HashMap sang Set thông qua phương thức entrySet(). Và khai bào 1 Map.entry để lấy ra những key và value tương ứng của nhau.

\* TreeMap: hiệu quả khi muốn lưu trữ những cặp key/value sắp xếp theo thứ tự và được thu hồi nhanh chóng.

- Những key trong TreeMap được sắp xếp theo thứ tự tang dần.

- TreeMap hỗ trợ 4 constructor là:

+ TreeMap(): xây dựng một TreeMap trống. key được sắp xếp theo thứ tự tự nhiên của key

+ TreeMap(Comparator comp): xây dựng một TreeMap trống. key đước sắp xếp theo thứ tự bởi sử dụng Comparator.

+ TreeMap(Map m): xây dụng một TreeMap từ Map định sẵn. key được sắp xếp theo thứ tự tự nhiên của key.

+ TreeMap(SortedMap sm): khỏi tạo treeMap từ các entry của sm và thứ tự theo sm.

- cung như HashMap, ngoài những phương thức được thừa kế từ MapInterface thì Map bổ xung thêm một số phương thức cho riêng nó như

+ clone(): tạo 1 bản sallow bản TreeMap này.

+ comparator(): trả về 1 Comparator từ TreeMap này.

+ firstKey(): trả về key đầu tiên trong sorted map này.

+ headMap(toKey); trả về 1 phần Map này có key nhỏ hơn toKey.

+ submap(fromKey, toKey): trả về 1 phần map này có key trong khoảng fromKey và toKey.

+ tailMap(fromKey): trả về 1 phần map này có key lớn hơn fromKey.

\* LinkedHashMap: kế thừa HashMap để cho phép lặp lại hay duy trì thứ tự chèn. ( không có phương thức mở rộng, nhược điểm là tốn bộ nhớ hơn HashMap do phải duy trì lưu trữ thứ tự chèn).

\* Hashtable: hiện lỗi thời nên không còn sử dụng nữa.

\* Tất cả nhưng demo đều có bước chuyển đổi sang Set để co thể lặp qua tất cả những phần tử trong Map và mỗi phần tử của một Set là 1 Map.Entry trong Map.Entry lại hỗ trợ những phương thức để lấy key và value của nó như getKey, getValue,…