**Trabajar con Mapas**

Un mapa es un objeto que almacena asociaciones entre claves y valores o pares clave/valor. Tanto claves como valores son objetos.

A partir de una clave se puede encontrar su valor. Las claves han de ser únicas, pero los valores pueden estar duplicados. Algunos mapas pueden aceptar un valor null y una clave null, otro no.

Los mapas no implementan la interfaz Iterable por lo que no definen iterados sobre ellos. Sin embargo se puede obtener una vista de colección de un mapa, lo que permite el uso tanto del for-in como del iterador.

**Interfaz Map**

Esta interfaz traza el mapa de claves únicas a valores. Una clave es un objeto que se usa para recuperar un valor en una fecha posterior. Dados una clave y un valor se puede almacenar el valor en un objeto Map y se puede recuperar usando su clave.

Define métodos tales como:

* **void clear ():** este método elimina todos los pares clave/valor de la colección.
* **boolean containskey (Object key):** este método verifica si la colección contiene la clave key.
* **boolean containsValue (Object valor):** este método verifica si la colección contiene al menos una instancia de valor.
* **boolean equals(Object obj):** este método verifica si el mapa es igual al parámetro de entrada
* **V get(Object key):** este método obtiene el valor asociado a la clave
* **Set keySet ():** este método devuelve un conjunto que contiene todas las claves del mapa, y sobre el cual se puede iterar.
* **V put (Object key, Object valor):** este método permite agregar un nuevo par clave/valor al mapa, si ya existía lo sobrescribe.
* **V remove (Object key):** este método elimina el par clave/valor desde el mapa.
* **int size ():** este método retorna un valor entero que identifica la cantidad de pares clave/valor que tiene el mapa.

La clase HashMap implementa esta interfaz.

**Interfaz SortedMap**

Esta interfaz extiende de Map asegurando que las entradas al mapa se mantengan en orden ascendente según las claves.

Esta interfaz agrega los siguientes métodos:

* **Comparator comparator ():** este método devuelve el comparador de mapa ordenado que realiza la llamada. Si para el mapa que realiza la llamada se utiliza un orden natural, se devuelve null.
* **K firstKey():** este método devuelve la primera clave del mapa
* **SortedMap headMap (K fin):** este método devuelve un mapa ordenada con las entradas del mapa que tienen claves menores a fin.
* **K lastKey ():** este método devuelve la última clave del mapa.
* **SortedMap subMap (K inicio, K fin):** este método devuelve un mapa ordenado, que es un subconjunto del original, entre inicio y fin.
* **SortedMap tailMap (K inicio):** este método devuelve un mapa ordenada con las entradas del mapa que tienen claves mayores a inicio.

La clase TreeMap implementa esta interfaz.