

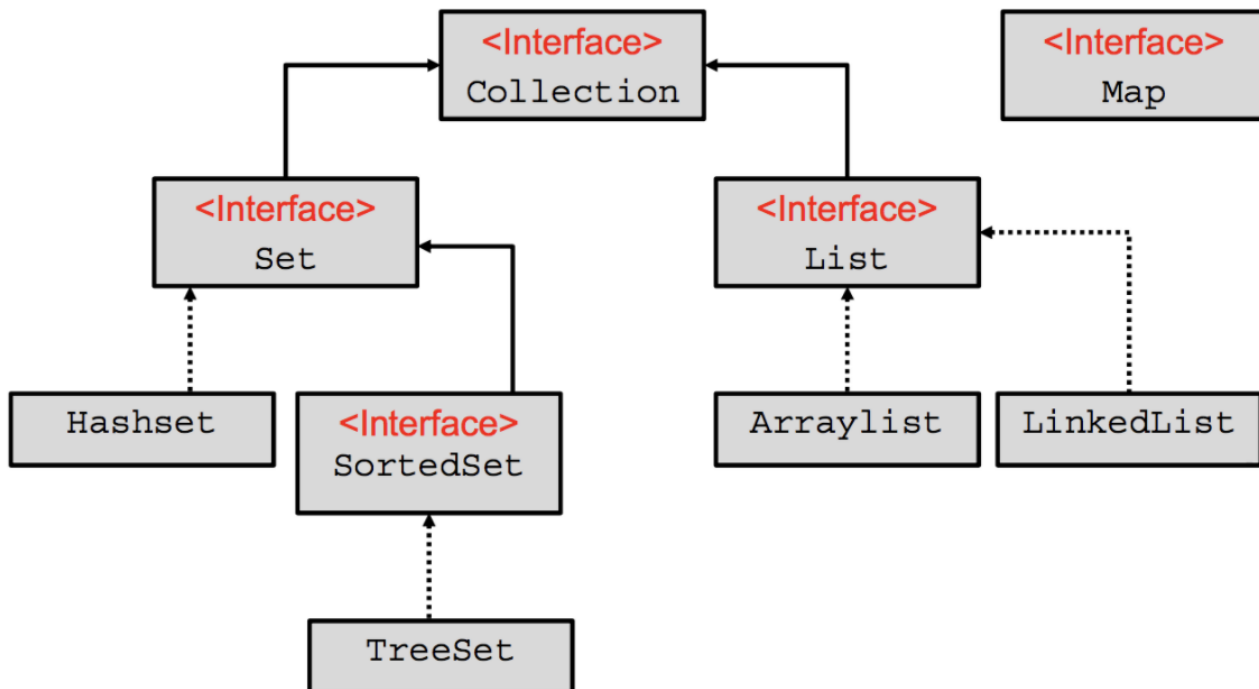
Map, HashMap, TreeMap y LinkedHashMap



anncode

1 de Junio de 2018

En el Curso Básico de Java vimos esta estructura de Colecciones interfaces




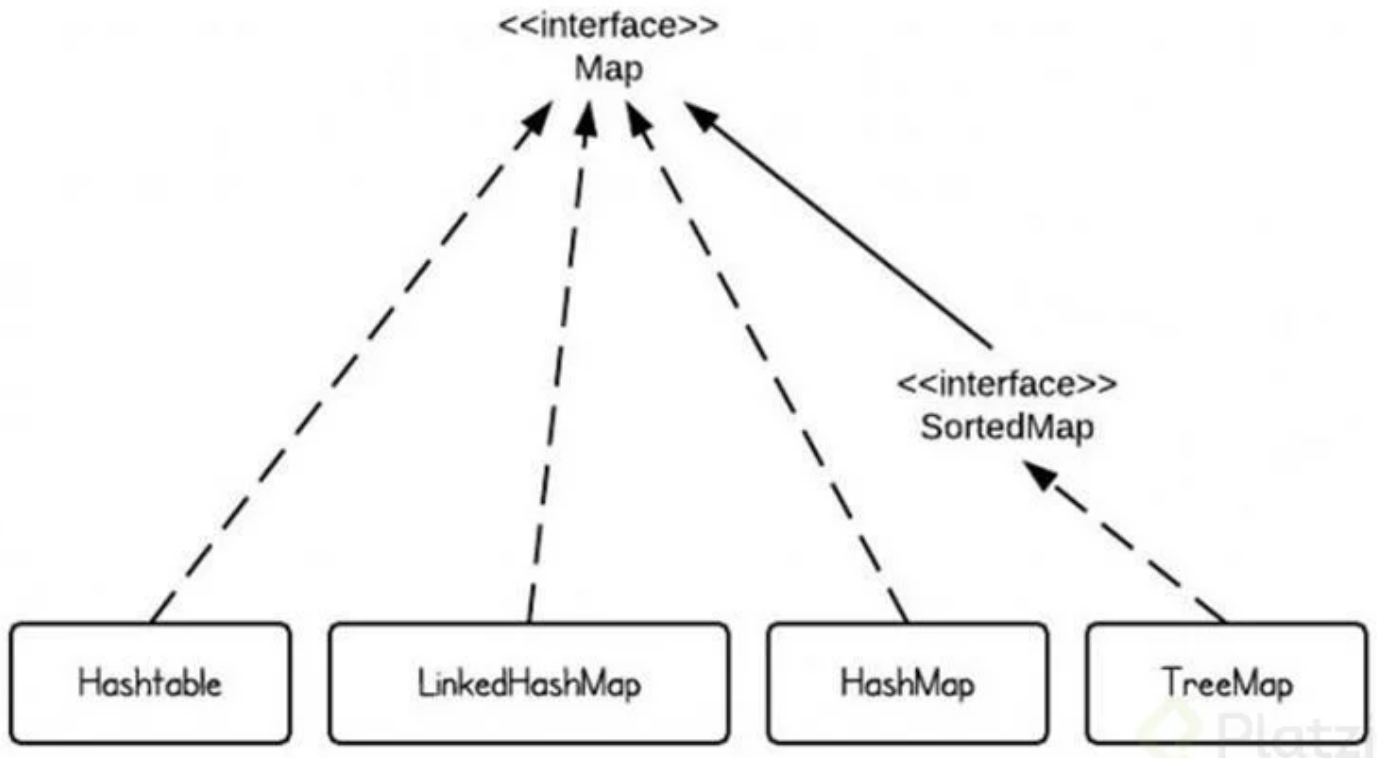
Te las expliqué todas menos la interfaz **Map** la cual te la dejé como reto (Si quieres repasarlo [aquí está el material](#)). Existe otra que no aparece en ese árbol se llama **Deque**

Hoy te platicaré sobre Map

Lo primero que debes saber es que tiene tres implementaciones:

- **HashTable**
- **LinkedHashMap**

- **HashMap**
- **SortedMap**  **TreeMap**



La interfaz **Map** no hereda de la interfaz `Collection` porque representa una estructura de datos de Mapeo y no de colección simple de objetos. Esta estructura es más compleja, pues cada elemento deberá venir en pareja con otro dato que funcionará como la llave del elemento.

`Map<K,V>`

- Donde **K** es el key o clave
- Donde **V** es el value o valor

Podemos declarar un map de la siguiente forma:

```
Map<Integer, String> map = new HashMap<Integer, String>();  
Map<Integer, String> treeMap = new TreeMap<Integer, String>();
```

```
Map<Integer, String> linkedHashMap = new LinkedHashMap<Integer, String>();
```

Como observas solo se puede construir el objeto con tres elementos que implementan de ella: **HashMap**, **TreeMap** y **LinkedHashMap** dejando fuera HashTable y SortedMap. SortedMap estará fuera pues es una interfaz y HashTable ha quedado deprecada pues tiene métodos redundantes en otras clases. Mira la funcionalidad de cada uno.

Como te conté hace un momento Map tiene implementaciones:

- **HashMap:** Los elementos no se ordenan. No aceptan claves duplicadas ni valores nulos.
- **LinkedHashMap** Ordena los elementos conforme se van insertando; provocando que las búsquedas sean más lentas que las demás clases.
- **TreeMap** El Mapa lo ordena de forma “natural”. Por ejemplo, si la clave son valores enteros (como luego veremos), los ordena de menos a mayor.

Para iterar alguno de estos será necesario utilizar la interface **Iterator** y para recorrerlo lo haremos un bucle **while** así como se muestra:

Para HashMap

```
// Imprimimos el Map con un Iterator
Iterator it = map.keySet().iterator();
while(it.hasNext()){
    Integer key = it.next();
    System.out.println("Clave: " + key + " -> Valor: " + map.get(key));
}
```

Para LinkedHashMap

```
// Imprimimos el Map con un Iterator
Iterator it = linkedHashMap.keySet().iterator();
```

```
while(it.hasNext()){
    Integer key = it.next();
    System.out.println("Clave: " + key + " -> Valor: " +
    linkedHashMap.get(key));
}
```

Para TreeMap

```
// Imprimimos el Map con un Iterador
Iterator it = treeMap.keySet().iterator();
while(it.hasNext()){
    Integer key = it.next();
    System.out.println("Clave: " + key + " -> Valor: " + treeMap.get(key));
}
```