HASHMAP

// Declaración de un Map (un HashMap) con clave "Integer" y Valor "String". Las claves pueden ser de cualquier tipo de objetos, aunque los más utilizados como clave son los objetos predefinidos de Java como String, Integer, Double ... !!!!CUIDADO los Map no permiten datos atómicos

Map<Integer, String> nombreMap = new HashMap<Integer, String>();

nombreMap.size(); // Devuelve el numero de elementos del Map

nombreMap.isEmpty(); // Devuelve true si no hay elementos en el Map y false si si los hay

nombreMap.put(K clave, V valor); // Añade un elemento al Map

nombreMap.get(K clave); // Devuelve el valor de la clave que se le pasa como parámetro o 'null' si la clave no existe

nombreMap.clear(); // Borra todos los componentes del Map

nombreMap.remove(K clave); // Borra el par clave/valor de la clave que se le pasa como parámetro

nombreMap.containsKey(K clave); // Devuelve true si en el map hay una clave que coincide con K

nombreMap.containsValue(V valor); // Devuelve true si en el map hay un Valor que coincide con V

nombreMap.values(); // Devuelve una "Collection" con los valores del Map

Otro elemento importante a la hora de trabajar con los Maps (aunque no lo es tanto como a la hora de trabajar con los [ArrayList](https://jarroba.com/arraylist-en-java-ejemplos/" \t "_blank)) son los "*Iteradores*" ([Iterator](http://docs.oracle.com/javase/1.4.2/docs/api/java/util/Iterator.html" \t "_blank)). Los Iteradores sirven para recorrer los Map y poder trabajar con ellos. Los Iteradores solo tienen tres métodos que son el *“hasNext()”* para comprobar que siguen quedando elementos en el iterador, el*“next()”*  para que nos de el siguiente elemento del iterador; y el *“remove()”* que sirve para eliminar el elemento del Iterador. En realidad se puede prescindir de los iteradores para trabajar con los Map ya que la gran ventaja de los Map frente a los ArrayList, es que estos tienen una clave asociada al objeto y se les puede buscar por la clave, aunque nunca esta de más saber utilizar los iteradores para manejar los Map.

Antes de seguir con ejemplos, os habréis dado cuenta que hemos empezado la entrada diciendo que Map es una Interface y por tanto se deben de implementar los métodos de la interface. Java ya tiene implementadas varias "clases Map". No vamos a explicar todas, pero si las tres que consideramos más importantes y útiles para que veáis la diferencia como son la clase "[HashMap](http://docs.oracle.com/javase/7/docs/api/java/util/HashMap.html" \t "_blank)", "[TreeMap](http://docs.oracle.com/javase/7/docs/api/java/util/TreeMap.html" \t "_blank)" y "[LinkedHashMap](http://docs.oracle.com/javase/7/docs/api/java/util/LinkedHashMap.html" \t "_blank)". La diferencia principal de estas 3 clases es la forma o el orden en las que guardan los valores en el Map. Si disteis en la carrera de Ingeniería Informática (o derivados) toda esa teoría de arboles (que no vamos a recordar aquí), veríais que la gran diferencia entre los arboles AVL, B, B+ y B\* es la forma en la que guardan los datos para después optimizar las altas, bajas, modificaciones y consultas de los "valores". Pues bien toda esa teoría en esta entrada para nosotros es trasparente, pero esta bien que veamos estas tres clases para que en función de vuestro problema elijáis la mejor clase:

* ***HashMap:***Los elementos que inserta en el map no tendrán un orden específico. No aceptan claves duplicadas ni valores nulos.
* ***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.
* ***LinkedHashMap:*** Inserta en el Map los elementos en el orden en el que se van insertando; es decir, que no tiene una ordenación de los elementos como tal, por lo que esta clase realiza las búsquedas de los elementos de forma más lenta que las demás clases.

Visto esto vamos a poner un ejemplo de estas 3 clases y como ordena los elementos. Para este ejemplo vamos a tener un Map en el que la clave va a ser el dorsal de los jugadores de fútbol de la selección española que jugaron de titulares la final de la copa del mundo de 2010, y su valor va a ser su nombre.

Con un HashMap

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

map.put(1, "Casillas"); map.put(15, "Ramos");

map.put(3, "Pique"); map.put(5, "Puyol");

map.put(11, "Capdevila"); map.put(14, "Xabi Alonso");

map.put(16, "Busquets"); map.put(8, "Xavi Hernandez");

map.put(18, "Pedrito"); map.put(6, "Iniesta");

map.put(7, "Villa");

*// Imprimimos el Map con un Iterador*

Iterator it = map.keySet().iterator();

while(it.hasNext()){

Integer key = it.next();

System.out.println("Clave: " + key + " -> Valor: " + map.get(key));

}

El resultado que tenemos es el siguiente, en el que vemos que nos da los objetos sin un "orden lógico":

Clave: 16 -> Valor: Busquets

Clave: 1 -> Valor: Casillas

Clave: 18 -> Valor: Pedrito

Clave: 3 -> Valor: Pique

Clave: 5 -> Valor: Puyol

Clave: 6 -> Valor: Iniesta

Clave: 7 -> Valor: Villa

Clave: 8 -> Valor: Xavi Hernandez

Clave: 11 -> Valor: Capdevila

Clave: 14 -> Valor: Xabi Alonso

Clave: 15 -> Valor: Ramos

Con un TreeMap

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

treeMap.put(1, "Casillas"); treeMap.put(15, "Ramos");

treeMap.put(3, "Pique"); treeMap.put(5, "Puyol");

treeMap.put(11, "Capdevila"); treeMap.put(14, "Xabi Alonso");

treeMap.put(16, "Busquets"); treeMap.put(8, "Xavi Hernandez");

treeMap.put(18, "Pedrito"); treeMap.put(6, "Iniesta");

treeMap.put(7, "Villa");

*// Imprimimos el Map con un Iterador que ya hemos instanciado anteriormente*

it = treeMap.keySet().iterator();

while(it.hasNext()){

Integer key = it.next();

System.out.println("Clave: " + key + " -> Valor: " + treeMap.get(key));

}

El resultado que tenemos es el siguiente en el que vemos que nos ordena los objetos por clave en "orden natural":

Clave: 1 -> Valor: Casillas

Clave: 3 -> Valor: Pique

Clave: 5 -> Valor: Puyol

Clave: 6 -> Valor: Iniesta

Clave: 7 -> Valor: Villa

Clave: 8 -> Valor: Xavi Hernandez

Clave: 11 -> Valor: Capdevila

Clave: 14 -> Valor: Xabi Alonso

Clave: 15 -> Valor: Ramos

Clave: 16 -> Valor: Busquets

Clave: 18 -> Valor: Pedrito