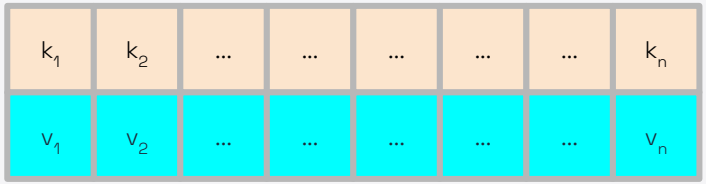
MAP

COLECCIONES

MAP<K,V>

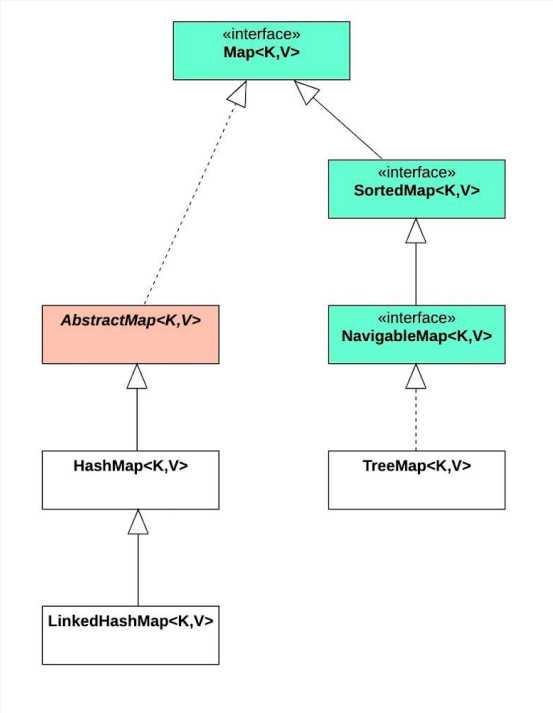
- Una colección que maneja pares *clave*, *valor*.
  - Cada clave existe una única vez en el *map*.
  - Cada clave tiene asociado un único valor.
  - Puede tener una clave nula, y múltiples valores nulos.
- No hereda de Collection<E>



ALGUNAS IMPLEMENTACIONES DE MAP<K,V>

Más rápido

Orden de inserción



Orden natural de las claves

CREACIÓN DE UN MAP

- *Map.of*
  - Crea un *Map* no modificable y que no permite nulos.
  - Recibe siempre un número par de argumentos (también 0)
- Map.*ofEntries*(...)
  - Recibe un varargs de *Map.Entry<K,V>* (par *clave,valor* de un Map)
  - Crea un Map conteniendo las claves y los valores, no los *entries* en sí.
- A través de sus implementaciones.

AÑADIR Y OBTENER ELEMENTOS

- Para añadir
  - *put(clave, valor)*: asocia la clave y el valor
  - *putIfAbsent(clave, valor)*: asocia la clave y el valor solamente si la clave no está insertada o está asociada a null.
  - *putAll(Map)* copia todos los pares de otro map.
- Para consultar
  - *get(clave)*: obtiene el valor asociado a una clave
  - *getOrDefault(clave, defecto)*: obtiene el valor asociado a una clave, y si no existe la clave, devuelve el valor por defecto.,

OBTENCIÓN DE LOS ELEMENTOS DE UN MAP

- *Map<K,V>* ofrece 3 métodos
  - *Set<K> keySet()*: devuelve una vista en forma de Set<K> con las claves del mapa.
  - *Collection<V> values()*: devuelve una vista en forma de Collection<V> con los valores del mapa.
  - *Set<Map.Entry<K,V>> entrySet()*: devuelve una vista en forma de Set que contiene los pares *clave,valor* (Map.Entry) del mapa

RECORRER UN MAP

- Lo más habitual es
  - Obtener el *keySet()*
  - Iterar sobre él como un Set (¿bucle for each?)
  - Para cada clave, obtener su valor con *get*

```
for(int n : map3.keySet()) {
    System.out.println("%d %s".formatted(n, map3.get(n)));
}
```

OTRAS OPERACIONES CON MAP<K,V>

- Consultar si una clave o un valor están contenidos
  - *containsKey(clave) / containsValue(valor)*
- Reemplazar el valor asociado a una clave
  - *replace(clave, valor) / replace(clave, valorAntiguo, valorNuevo)*
- Eliminar el par clave / valor
  - *remove(clave)*
  - *remove(clave, valor)*
- Vaciar el Map: *clear()*

HASHMAP<K,V>

- Implementación más “básica” de Map<K,V>.
  - Sin orden.
  - Permite una clave null y valores null.
  - Operaciones de acceso en tiempo constante (O(1))
- Similar a HashSet pero implementando Map.

LINKEDHASHMAP<K,V>

- Rendimiento muy similar a HashMap<K,V>
- Mantiene además una lista ordenada según la inserción.
- Al iterar sobre las claves, podemos obtener los pares en dicho orden.

TREEMAP<K,V>

- Implementación que mantiene el orden natural de sus claves.
- Similar a TreeSet
- Operaciones para
  - *keys*: firstKey, floorKey, ceilingKey, ... devuelven una clave
  - *entries*: firstEntry, floorEntry, ceilingEntry devuelven un par k,v
- Iteración en orden ascendente y descendente