**HashMap**

**HashMap<key , value>**

**Key will not duplicate, if key is duplicate will be override (will chose the second one) , and no-order(random)**

h.put("city", "Niles1");

h.put("city", "Niles2");

city is a key and it’s duplicated , HashMap will chose the second one Niles2

Ex:

**public** **class** Main {

**public** **static** **void** main(String[] args) {

HashMap<String, String> h = **new** HashMap<>();

h.put("country", "USA");

h.put("city", "Niles1");

h.put("city", "Niles2");

h.put("code", "60714");

h.put("state", "ill");

h.put(**null**, "chicago1");

h.put(**null**, "chicago2");

System.***out***.println(h.get("country")); // USA

System.***out***.println(h.get("city")); // Niles2

System.***out***.println(h.get(**null**)); // chicago2

// iterator by using KeySet()

Iterator<String> it = h.keySet().iterator();

**while**(it.hasNext()) {

String key = it.next();

String value = h.get(key);

System.***out***.println("Key= "+key + " value= "+value);

/\*

Key= null value= chicago2

Key= country value= USA

Key= code value= 60714

Key= city value= Niles2

Key= state value= ill

\*/

}

System.***out***.println("...................................................");

// iterator by using entrySet

Iterator<Entry<String,String>> it1 = h.entrySet().iterator();

**while**(it1.hasNext()) {

Entry<String,String> entry = it1.next();

System.***out***.println("Key= "+entry.getKey()+ " Value= "+entry.getValue());

/\*

Key= null value= chicago2

Key= country value= USA

Key= code value= 60714

Key= city value= Niles2

Key= state value= ill

\*/

}

System.***out***.println("...................................................");

// iterator by using lambda

h.forEach( (k,v) -> System.***out***.println("Key= "+k+ " value= "+v) );

/\*

Key= null value= chicago2

Key= country value= USA

Key= code value= 60714

Key= city value= Niles2

Key= state value= ill

\*/

}

}

**Ex:**

**public** **class** Main {

**public** **static** **void** main(String[] args) {

// it will compare base on key and value, any change on key or value it will give us false

HashMap<Integer, String> map1 = **new** HashMap<>();

map1.put(1, "A");

map1.put(2, "B");

map1.put(3, "C");

HashMap<Integer, String> map2 = **new** HashMap<>();

map2.put(1, "A");

map2.put(3, "C");

map2.put(2, "B");

HashMap<Integer, String> map3 = **new** HashMap<>();

map3.put(1, "A");

map3.put(2, "B");

map3.put(3, "C");

map3.put(3, "D");

// 1-compare using equal method

System.***out***.println(map1.equals(map2)); // true

System.***out***.println(map2.equals(map3)); // false

// 2-compare using keySet() , it will compare base on key even the key is random doesn't matter

System.***out***.println(map1.keySet().equals(map2.keySet())); // true

System.***out***.println(map2.keySet().equals(map3.keySet())); // true

// 3-find out extra keys: we will compare map1 with map4 and find the different key(extra key)

HashMap<Integer, String> map4 = **new** HashMap<>();

map4.put(1, "A");

map4.put(2, "B");

map4.put(3, "C");

map4.put(4, "D");

HashSet<Integer> h = **new** HashSet<Integer>(map1.keySet());

h.addAll(map4.keySet());

h.removeAll(map1.keySet());

System.***out***.println(h); // [4]

Graphical user interface, application

Description automatically generated with medium confidence

}

}

how HashMap works internally?

**Diagram

Description automatically generated**

map.get("ninos");// hashCode of ninos will compare with hashCode in array and get the value 100

map.get("nahrain"); // 200

map.get("matthew"); // hashCode of matthew will compare with hashCode in array and it will go to index 4, first it will see same hashCode of ninos is the same hashCode of matthew so it will use equal method to see the name of key and it will see is not same,after will go to the next node and it will same hashCode and same name of the key

map.get("daniel"); // 400

map.get(**null**); // 700