MAP

Map is an interface of java Collections framework.

Map are stored in Key/value pairs.

* Key - a unique identifier used to associate each element (value) in a map.( A map cannot contain duplicate keys, it should be unique)
* Value - elements associated by keys in a map(Value can be duplicate)
* Key+Value=Entry (Key🡪index ;Value🡪 Actual data)

IMPLEMENTATION

**HashMap: Based on hashing algorithm(random)**

**TreeMap: Based on ASCII order (alphabetic order)**

**LinkedHashMap: Based on insertion order(First in-first)**

**Methods of Map**

The Map interface includes all the methods of the Collection interface.

* **put(K, V)** - Inserts the association of a key K and a value V into the map. If the key is already present, the new value replaces the old value.
* **putAll()** - Inserts all the entries from the specified map to this map.
* **get(K)** - Returns the value associated with the specified key K. If the key is not found, it returns null.
* **getOrDefault(K, defaultValue)** - Returns the value associated with the specified key K. If the key is not found, it returns the defaultValue.
* **containsKey(K)** - Checks if the specified key K is present in the map or not.
* **containsValue(V)** - Checks if the specified value V is present in the map or not.
* **replace(K, V)** - Replace the value of the key K with the new specified value V.
* **remove(K)** - Removes the entry from the map represented by the key K.
* **keySet() -** Returns a set of all the keys present in a map.
* **values() -** Returns a set of all the values present in a map.
* **entrySet() -** Returns a set of all the key/value mapping present in a map.

SYNTAX TO CREATE MAP

Map<KeyDatatype,ValueDatatype> ref\_mapname = new HashMap< KeyDatatype,ValueDatatype>();

Example

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

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

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

/ Insert elements to the map

numbers.put("One", 1);

numbers.put("Two", 2);

System.out.println("Map: " + numbers);

// Access keys of the map

System.out.println("Keys: " + numbers.keySet());

// Access values of the map

System.out.println("Values: " + numbers.values());

// Access entries of the map

System.out.println("Entries: " + numbers.entrySet());

// Remove Elements from the map

int value = numbers.remove("Two");

System.out.println("Removed Value: " + value);

Map: {One=1, Two=2}

Keys: [One, Two]

Values: [1, 2]

Entries: [One=1, Two=2]

Removed Value: 2

## Iterate through HashMap using the forEach loop

import java.util.HashMap;

import java.util.Map.Entry;

class Main {

public static void main(String[] args) {

// Creating a HashMap

HashMap<String, String> languages = new HashMap<>();

languages.put("Java", "Enterprise");

languages.put("Python", "ML/AI");

languages.put("JavaScript", "Frontend");

System.out.println("HashMap: " + languages);

// iterating through key/value mappings

System.out.print("Entries: ");

for(Entry<String, String> entry: languages.entrySet()) {

System.out.print(entry);

System.out.print(", ");

}

// iterating through keys

System.out.print("\nKeys: ");

for(String key: languages.keySet()) {

System.out.print(key);

System.out.print(", ");

}

// iterating through values

System.out.print("\nValues: ");

for(String value: languages.values()) {

System.out.print(value);

System.out.print(", ");

}

}

}