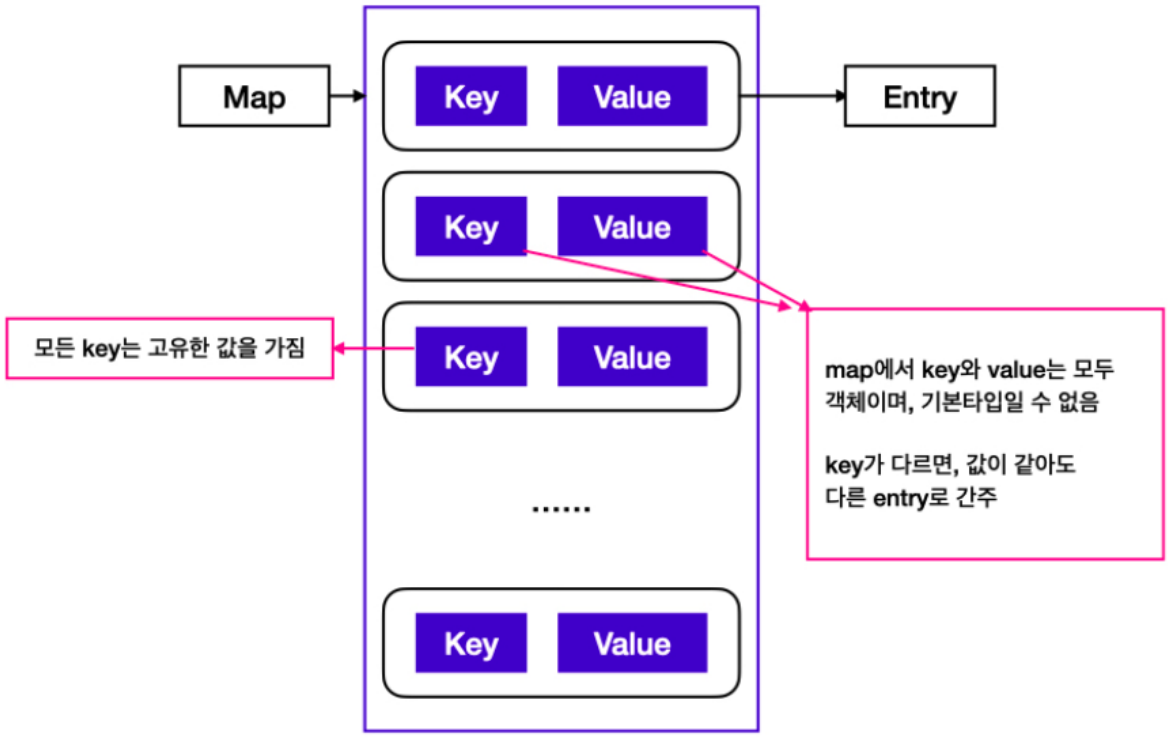
Map 이란**?**

Map 인터페이스는 **키(Key)와 값(Value)으로 구성된 객체를 저장하는 구조**를 가지고 있다. 여기서 이 객체를 Entry 객체라고 하는데, 이 Entry 객체는 키와 값을 각각 Key 객체와 Value 객체로 저장한다. Map 인터페이스를 구현한 클래스에는 HashMap, HashTable, TreeMap, SortedMap 등이 있다.



Map 은 **키는 중복 저장될 수 없지만, 값은 중복 저장이 가능하다.** 키의 역할이 값을 식별하는 것이다.

만약 저장된 키와 동일한 키로 값을 저장하면, 기존의 값이 새로운 값으로 대치된다.

Map 인터페이스를 구현한 클래스에서 공통적으로 사용 가능한 메서드

put(Object key, Object value)

주어진 키로 값을 저장한다. 해당 키가 새로운 키일 경우 null 을 리턴하지만, 동일한 키가 있을 경우에는 기존의 값을 대체하고 대체되기 이전의 값을 리턴한다. **Object** 타입을 리턴한다.

containsKey(Object key)

주어진 키가 있으면 true 없으면 false 를 리턴한다. **boolean** 타입을 리턴한다.

containsValue(Object value)

주어진 값이 있으면 true 없으면 false 를 리턴한다. **boolean** 타입을 리턴한다.

entrySet()

키와 값의 쌍으로 구성된 모든 Map.Entry 객체를 Set 에 담아서 리턴한다. **Set** 타입을 리턴한다.

get(Object key)

주어진 키에 해당하는 값을 리턴한다. **Object** 타입을 리턴한다.

isEmpty()

컬렉션이 비어 있는지 확인한다. **boolean** 타입을 리턴한다.

keySet()

모든 키를 Set 객체에 담아서 리턴한다. **Set** 타입을 리턴한다.

size()

저장된 Entry 객체의 총 갯수를 리턴한다. **int** 타입을 리턴한다.

values()

저장된 모든 값을 Collection 에 담아서 리턴한다. **Collection** 타입을 리턴한다.

clear()

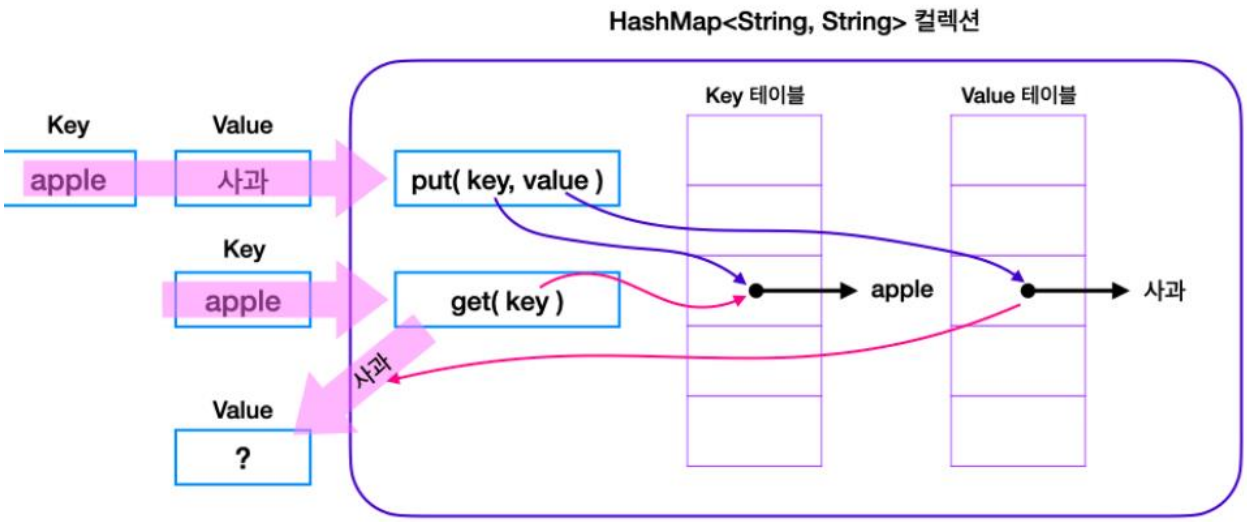
모든 Map.Entry(키와 값)을 삭제한다.

remove(Object key)

주어진 키와 일치하는 Map.Entry 를 삭제하고 값을 리턴한다. **Object** 타입을 리턴한다.

HashMap

HashMap 은 Map 인터페이스를 구현한 대표적인 클래스이다.



해시 함수를 통해 키와 값이 저장되는 위치를 결정하므로, 사용자는 그 위치를 알 수 없고, **삽입되는 순서와 위치 또한 관계가 없다. 많은 양의 데이터를 검색하는 데 있어서 뛰어난 성능을 보인다.**

Map.Entry 인터페이스에서 사용가능한 메서드

equals(Object o)

동일한 Entry 객체인지 비교합니다. **boolean** 타입을 리턴한다.

getKey()

Entry 객체의 Key 객체를 반환합니다. **Object** 타입을 리턴한다.

getValue()

Entry 객체의 Value 객체를 반환합니다. **Object** 타입을 리턴한다.

hashCode()

Entry 객체의 해시코드를 반환합니다. **int** 타입을 리턴한다.

setValue(Object value)

Entry 객체의 Value 객체를 인자로 전달한 value 객체로 바꿉니다. **Object** 타입을 리턴한다.

HashMap 예제

```
import java.util.*;

public class HashMapExample {
    public static void main(String[] args) {

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

        // Entry 객체 저장
        map.put("피카츄", 85);
        map.put("꼬부기", 95);
        map.put("야도란", 75);
        map.put("파이리", 65);
```

```
        map.put("피존투", 15);

        // 저장된 총 Entry 수 얻기
        System.out.println("총 entry 수: " + map.size());

        // 객체 찾기
        System.out.println("파이리 : " + map.get("파이리"));

        // key를 요소로 가지는 Set을 생성 -> 아래에서 순회하기 위해 필요하다
        Set<String> keySet = map.keySet();

        // keySet을 순회하면서 value를 읽어온다.
        Iterator<String> keyIterator = keySet.iterator();
        while(keyIterator.hasNext()) {
            String key = keyIterator.next();
            Integer value = map.get(key);
            System.out.println(key + " : " + value);
        }

        // 객체 삭제
        map.remove("피존투");

        System.out.println("총 entry 수: " + map.size());

        // Entry 객체를 요소로 가지는 Set을 생성 -> 아래에서 순회하기 위해 필요하다.
        Set<Map.Entry<String, Integer>> entrySet = map.entrySet();

        // entrySet을 순회하면서 value를 읽어온다.
        Iterator<Map.Entry<String, Integer>> entryIterator = entrySet.iterator();
        while(entryIterator.hasNext()) {
            Map.Entry<String, Integer> entry = entryIterator.next();
            // Map.Entry 인터페이스의 메서드
            String key = entry.getKey();
            // Map.Entry 인터페이스의 메서드
            Integer value = entry.getValue();
            System.out.println(key + " : " + value);
        }
```

```
        // 객체 전체 삭제
        map.clear();
    }
}

/* 출력
총 entry 수: 5
파이리 : 65
야도란 : 75
꼬부기 : 95
파이리 : 65
피카츄 : 85
피존투 : 15
총 entry 수: 4
야도란 : 75
꼬부기 : 95
파이리 : 65
피카츄 : 85
*/
```

Map 은 키와 값을 쌍으로 저장하기 때문에 iterator()를 직접 호출할 수 없다. 대신 **keySet()**이나 **entrySet()**메서드를 이용해 **Set** 형태로 반환된 컬렉션에 **iterator()**를 호출하여 반복자를 만든 후 순회할 수 있다**.**

HashTable

HashMap 과 내부 구조가 동일하며 사용 방법 또한 매우 유사하다.

HashTable 예제

```
import java.util.*;

public class HashTableExample {
```

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

        HashTable<String, String> map = new
Hashtable<String, String>();

        map.put("Spring", "345");
        map.put("Summer", "678");
        map.put("Fall", "91011");
        map.put("Winter", "1212");

        System.out.println(map);

        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("아이디와 비밀번호를 입력해
주세요");
            System.out.println("아이디");
            String id = scanner.nextLine();

            System.out.println("비밀번호");
            String password = scanner.nextLine();

            if (map.containsKey(id)) {
                if (map.get(id).equals(password)) {
                    System.out.println("로그인 되었습니다.");
                    break;
                }
                else System.out.println("비밀번호가 일치하지
않습니다. ");
            }
            else System.out.println("입력하신 아이디가
존재하지 않습니다.");
        }
    }
}
```