Java EnumMap class

Java EnumMap class is the specialized Map implementation for enum keys. It inherits Enum and AbstractMap classes.

EnumMap class hierarchy

The hierarchy of EnumMap class is given in the figure given below.

EnumMap class declaration

Let's see the declaration for java.util.EnumMap class.

1. **public** **class** EnumMap<K **extends** Enum<K>,V> **extends** AbstractMap<K,V> **implements** Serializable, Cloneable

EnumMap class Parameters

Let's see the Parameters for java.util.EnumMap class.

* **K:** It is the type of keys maintained by this map.
* **V:** It is the type of mapped values.

Constructors of Java EnumMap class

|  |  |
| --- | --- |
| **Constructor** | **Description** |
| EnumMap(Class<K> keyType) | It is used to create an empty enum map with the specified key type. |
| EnumMap(EnumMap<K,? extends V> m) | It is used to create an enum map with the same key type as the  specified enum map. |
| EnumMap(Map<K,? extends V> m) | It is used to create an enum map initialized from the specified map. |

Methods of Java EnumMap class

|  |  |  |
| --- | --- | --- |
| **SN** | **Method** | **Description** |
| 1 | [clear()](https://www.javatpoint.com/post/java-enummap-clear-method) | It is used to clear all the mapping from the map. |
| 2 | [clone()](https://www.javatpoint.com/post/java-enummap-clone-method) | It is used to copy the mapped value of one map to another  map. |
| 3 | [containsKey()](https://www.javatpoint.com/post/java-enummap-containskey-method) | It is used to check whether a specified key is present in this map  or not. |
| 4 | [containsValue()](https://www.javatpoint.com/post/java-enummap-containsvalue-method) | It is used to check whether one or more key is associated  with  a given value or not. |
| 5 | [entrySet()](https://www.javatpoint.com/post/java-enummap-entryset-method) | It is used to create a set of elements contained in the  EnumMap. |
| 6 | [equals()](https://www.javatpoint.com/post/java-enummap-equals-method) | It is used to compare two maps for equality. |
| 7 | [get()](https://www.javatpoint.com/post/java-enummap-get-method) | It is used to get the mapped value of the specified key. |
| 8 | [hashCode()](https://www.javatpoint.com/post/java-enummap-hashcode-method) | It is used to get the hashcode value of the EnumMap. |
| 9 | [keySet()](https://www.javatpoint.com/post/java-enummap-keyset-method) | It is used to get the set view of the keys contained in the  map. |
| 10 | [size()](https://www.javatpoint.com/post/java-enummap-size-method) | It is used to get the size of the EnumMap. |
| 11 | [Values()](https://www.javatpoint.com/post/java-enummap-values-method) | It is used to create a collection view of the values contained in this  map. |
| 12 | [put()](https://www.javatpoint.com/post/java-enummap-put-method) | It is used to associate the given value with the given  key in this  EnumMap. |
| 13 | [putAll()](https://www.javatpoint.com/post/java-enummap-putall-method) | It is used to copy all the mappings from one EnumMap  to a new  EnumMap. |
| 14 | [remove()](https://www.javatpoint.com/post/java-enummap-remove-method) | It is used to remove the mapping for the given key from  EnumMap  if the given key is present. |

Java EnumMap Example

1. **import** java.util.\*;
2. **public** **class** EnumMapExample {
3. // create an enum
4. **public** **enum** Days {
5. Monday, Tuesday, Wednesday, Thursday
6. };
7. **public** **static** **void** main(String[] args) {
8. //create and populate enum map
9. EnumMap<Days, String> map = **new** EnumMap<Days, String>(Days.**class**);
10. map.put(Days.Monday, "1");
11. map.put(Days.Tuesday, "2");
12. map.put(Days.Wednesday, "3");
13. map.put(Days.Thursday, "4");
14. // print the map
15. **for**(Map.Entry m:map.entrySet()){
16. System.out.println(m.getKey()+" "+m.getValue());
17. }
18. }
19. }

Output:

Monday 1

Tuesday 2

Wednesday 3

Thursday 4

Java EnumMap Example: Book

1. **import** java.util.\*;
2. **class** Book {
3. **int** id;
4. String name,author,publisher;
5. **int** quantity;
6. **public** Book(**int** id, String name, String author, String publisher, **int** quantity) {
7. **this**.id = id;
8. **this**.name = name;
9. **this**.author = author;
10. **this**.publisher = publisher;
11. **this**.quantity = quantity;
12. }
13. }
14. **public** **class** EnumMapExample {
15. // Creating enum
16. **public** **enum** Key{
17. One, Two, Three
18. };
19. **public** **static** **void** main(String[] args) {
20. EnumMap<Key, Book> map = **new** EnumMap<Key, Book>(Key.**class**);
21. // Creating Books
22. Book b1=**new** Book(101,"Let us C","Yashwant Kanetkar","BPB",8);
23. Book b2=**new** Book(102,"Data Communications & Networking","Forouzan","Mc Graw Hill",4);
24. Book b3=**new** Book(103,"Operating System","Galvin","Wiley",6);
25. // Adding Books to Map
26. map.put(Key.One, b1);
27. map.put(Key.Two, b2);
28. map.put(Key.Three, b3);
29. // Traversing EnumMap
30. **for**(Map.Entry<Key, Book> entry:map.entrySet()){
31. Book b=entry.getValue();
32. System.out.println(b.id+" "+b.name+" "+b.author+" "+b.publisher+" "+b.quantity);
33. }
34. }
35. }

Output:

101 Let us C Yashwant Kanetkar BPB 8

102 Data Communications & Networking Forouzan Mc Graw Hill 4

103 Operating System Galvin Wiley 6