



[New Guide] Download the 2018 Guide to Java: Features, Improvements, & Updates [Download Guide](#) ▶

Why are the keys of a Map in a jumbled order

by Peter Lawrey MVB · Jul. 31, 11 · Java Zone · Not set

Heads up...this article is old!

Technology moves quickly and this article was published 7 years ago. Some or all of its contents may be outdated.

Download Microservices for Java Developers: A hands-on introduction to frameworks and containers. Brought to you in partnership with Red Hat.

When you first try to printout a HashMap, the order doesn't make any sense and when you add entries, they appear to jump around. What is going on?

The order of Maps

The order keys should appear is not defined in many implementations.

Hash map uses a hash of the key to place the key/value in a pseudo random place in the underlying store (an array) How it does this is different in different implementations. For HashMap the size of the Map is always a power of 2. For Hashtable the size grows 11, 23, 47, 95.

For LinkedHashMap, the order will be the order the keys were added and for TreeMap & ConcurrentSkipList, the order is based of the Comparable.compareTo() result (asciibetical order for String)

For Hashtable and HashMap changing the initial size changes how the keys are hashed and thus their order

```
public static void main(String... args) {
```

```

        populate(new Hashtable());
        populate(new Hashtable(47), " (47)");
        populate(new Hashtable(95), " (95)");
        populate(new HashMap());
        populate(new HashMap(32), " (32)");
        populate(new HashMap(64), " (64)");
        populate(new LinkedHashMap());
        populate(new IdentityHashMap());
        populate(new WeakHashMap());
        populate(new ConcurrentHashMap());
        populate(new TreeMap());
        populate(new ConcurrentSkipListMap());
    }

    private static void populate(Map map) {
        populate(map, "");
    }

    private static void populate(Map map, String suffix) {
        for (String s : "one,two,three,four,five,six,seven,eight,nine,ten".split(","))
            map.put(s, s);
        System.out.println(map.getClass().getSimpleName() + suffix + " " + map.keySet());
    }

```

prints

```

Hashtable [three, six, ten, seven, nine, one, five, four, two, eight]
Hashtable (47) [ten, five, seven, two, three, one, nine, six, eight, four]
Hashtable (95) [nine, five, six, one, seven, eight, ten, two, four, three]
HashMap [ten, two, seven, five, nine, one, three, four, eight, six]
HashMap (32) [ten, five, nine, one, eight, six, two, seven, three, four]
HashMap (64) [ten, nine, one, eight, six, three, four, five, two, seven]
LinkedHashMap [one, two, three, four, five, six, seven, eight, nine, ten]
IdentityHashMap [four, two, five, nine, three, six, eight, one, seven, ten]
WeakHashMap [six, eight, four, three, nine, one, seven, five, ten, two]
ConcurrentHashMap [seven, two, one, nine, four, six, eight, three, ten, five]
TreeMap [eight, five, four, nine, one, seven, six, ten, three, two]
ConcurrentSkipListMap [eight, five, four, nine, one, seven, six, ten, three, two]

```

From <http://vanillajava.blogspot.com/2011/07/why-are-keys-of-map-in-jumbled-order.html>

Download Building Reactive Microservices in Java: Asynchronous and Event-Based Application Design. Brought to you in partnership with Red Hat.

Like This Article? Read More From DZone



Set Up and Integrate Prometheus With Grafana for Monitoring



How Conexus Credit Union Uses Azure Machine Learning to Improve Business Processes



Pseudonymizing Your Data With SQL Data Generator



**Free DZone Refcard
Getting Started With Kotlin**

Topics:

Opinions expressed by DZone contributors are their own.

Java Partner Resources

Designing Reactive Systems: The Role Of Actors In Distributed Architecture

Lightbend



Level up your code with a Pro IDE

JetBrains



Microservices for Java Developers: A Hands-On Introduction to Frameworks & Containers

Red Hat Developer Program



Predictive Analytics + Big Data Quality: A Love Story

Melissa Data

