|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. No. | **Property** | ***HashMap*** | ***Hashtable*** | ***LinkedHashMap*** | ***TreeMap*** |
| 1 | Insertion order | HashMap **does not maintains insertion order** in java. | Hashtable **does not maintains insertion** order in java. | LinkedHashMap  **maintains insertion order** in java. | TreeMap is **sorted by natural order of keys** in java. |
| 2 | Performance | HashMap is **not synchronized**, hence its operations are **faster** as compared to Hashtable. | Hashtable is **synchronized**, hence its operations are **slower** as compared HashMap.  If we are working not working in multithreading environment jdk recommends us to use HashMap. | LinkedHashMap must be used only when we want to maintain insertion order. **Time and space overhead** is there because for maintaining order it internally uses **Doubly Linked list**. | TreeMap must be used only when we want sorting based on natural order. Otherwise **sorting operations cost performance.** (Comparator is called for sorting purpose) |
| 3 | Null keys and values | HashMap allows to store **one null key** and **many null values** i.e. many keys can have null value in java. | HashTable does **not allow to store null key or null value**.  Any attempt to store null key or value throws runtimeException (NullPointerException) in java. | LinkedHashMap allows to store **one null key** and **many null values** i.e. many key can have null value in java. | TreeMap does **not allow to store null key but allow many null values**.  Any attempt to store null key throws runtimeException (NullPointerException) in java. |
| 4 | Implements which interface | HashMap implements **java.util.**[**Map**](http://www.javamadesoeasy.com/2015/04/map-hierarchy-in-java-detailed-hashmap.html) | Hashtable implements **java.util.Map** | LinkedHashMap implements **java.util.Map** | TreeMap implements  **java.util.Map**  **java.util.SortedMap**  **java.util.NavigableMap** |
| 5 | Implementation uses? | HashMap use [**buckets**](http://javamadesoeasy.com/2015/02/hashmap-custom-implementation.html) | Hashtable use **buckets** | LinkedHashMap uses [**doubly linked lists**](http://www.javamadesoeasy.com/2015/02/linkedhashmap-custom-implementation.html) | TreeMap uses **Red black tree** |
| 6 | Complexity of put, get and remove methods | O(1) | O(1) | O(1)  **overhead** of updating **Doubly Linked list** for maintaining order it internally uses. | O(log(n)) |
| 7 | Extends java.util.**Dictionary** (Abstract class, which is obsolete) | HashMap **doesn’t** extends Dictionary. | Hashtable **extends** Dictionary (which maps non-null keys to values. In a given Dictionary we can look up value corresponding to key) | LinkedHashMap **doesn’t** extends Dictionary. | TreeMap **doesn’t** extends Dictionary. |
| 8 | Introduced in which java version? | HashMap was introduced in second version of java i.e. **JDK 2.0** | Hashtable was introduced in first version of java i.e. **JDK 1.0**  But it was refactored in java 2 i.e. JDK 1.2 to implement the Map interface, hence making it a member of member of the [Java Collections Framework](http://download.oracle.com/javase/7/docs/technotes/guides/collections/index.html). | LinkedHashMap was introduced in fourth version of java i.e. **JDK 4.0** | TreeMap was introduced in second version of java i.e. **JDK 2.0** |