**Difference between HashMap and HashTable**

|  |  |
| --- | --- |
| **HashMap** | **HashTable** |
| HashMap is non synchronized and not thread safe | Hashtable is thread safe and synchronized. |
| If your application do not require any multi-threading task, in other words HashMap is better for non-threading applications. | Hashtable should be used in multithreading applications. |
| HashMap allows one null key and any number of null values | Hashtable do not allow null keys and null values in the Hashtable object. |
| HashMap object values are iterated by using iterator | Hashtable is the only class other than vector which uses enumerator to iterate the values of Hashtable object. |
| The iterator in HashMap is fail-fast iterator | The enumerator for Hashtable is not fail-fast. |
| It is better off externally synchronizing a HashMap or using a ConcurrentMap implementation (e.g ConcurrentHashMap).HashMap is the subclass of the AbstractMap class. Although Hashtable and HashMap has different superclasses but they both are implementations of the *"Map"*  abstract data type. | Hashtable is a subclass of Dictionary class which is now obsolete in Jdk 1.7 |

**Performance :**  HashMap is much faster and uses less memory than Hashtable as former is unsynchronized . Unsynchronized objects are often much better in performance in compare to synchronized  object like Hashtable in single threaded environment.

**Similarities Between HashMap and Hashtable**  
  
**1. Insertion Order :**   Both HashMap and Hashtable  does not guarantee that  the order of the map will remain constant over time. Instead use LinkedHashMap, as the order remains constant over time.  
  
**2. Map interface :**   Both HashMap and Hashtable implements Map interface .  
  
**3. Put and get method :**  Both HashMap and Hashtable provides constant time performance for put and get methods assuming that the objects are distributed uniformly across the bucket.  
  
**4. Internal working :**  Both HashMap and Hashtable works on the Principle of Hashing.



