The java.util.concurrent.ConcurrentMap interface represents a [**Map**](http://tutorials.jenkov.com/java-collections/map.html) which is capable of handling concurrent access (puts and gets) to it.

The ConcurrentMap has a few extra atomic methods in addition to the methods it inherits from its superinterface, [**java.util.Map**](http://tutorials.jenkov.com/java-collections/map.html).

**ConcurrentMap Implementations**

Since ConcurrentMap is an interface, you need to use one of its implementations in order to use it. The java.util.concurrent package contains the following implementations of the ConcurrentMap interface:

* ConcurrentHashMap

**ConcurrentHashMap**

The ConcurrentHashMap is very similar to the java.util.HashTable class, except that ConcurrentHashMap offers better concurrency than HashTable does. ConcurrentHashMap does not lock the Map while you are reading from it. Additionally, ConcurrentHashMap does not lock the entire Map when writing to it. It only locks the part of the Map that is being written to, internally.

Another difference is that ConcurrentHashMap does not throw ConcurrentModificationException if the ConcurrentHashMap is changed while being iterated. The Iterator is not designed to be used by more than one thread though.

**ConcurrentMap Example**

Here is an example of how to use the ConcurrentMap interface. The example uses a ConcurrentHashMap implementation:

ConcurrentMap c = new ConcurrentHashMap();

c.put("key", "value");

Object value = c.get("key");