LRU Cache的LinkedHashMap实现

LRU是Least Recently Used 的缩写，翻译过来就是“**最近最少使用**”，**LRU缓存就是使用这种原理实现，简单的说就是缓存一定量的数据，当超过设定的阈值时就把一些过期的数据删除掉**，比如我们缓存10000条数据，当数据小于10000时可以随意添加，当超过10000时就需要把新的数据添加进来，同时要把过期数据删除，以确保我们最大缓存10000条，那怎么确定删除哪条过期数据呢，采用LRU算法实现的话就是将最老的数据删掉，废话不多说，下面来说下Java版的LRU缓存实现。

Java里面实现LRU缓存通常有两种选择，一种是使用**LinkedHashMap**，一种是自己设计数据结构，使用**链表+HashMap。**

# LRU缓存的LinkedHashMap实现

LinkedHashMap自身已经实现了**顺序存储**，默认情况下是**按照元素的添加顺序存储**，也可以启用**按照访问顺序存储**，**即最近读取的数据放在最前面，最早读取的数据放在最后面**，然后它还有一个判断是否删除最老数据的方法，默认是返回false，即不删除数据，我们使用LinkedHashMap实现LRU缓存的方法就是对LinkedHashMap实现简单的扩展，扩展方式有两种，**一种是inheritance，一种是delegation**，具体使用什么方式看个人喜好。