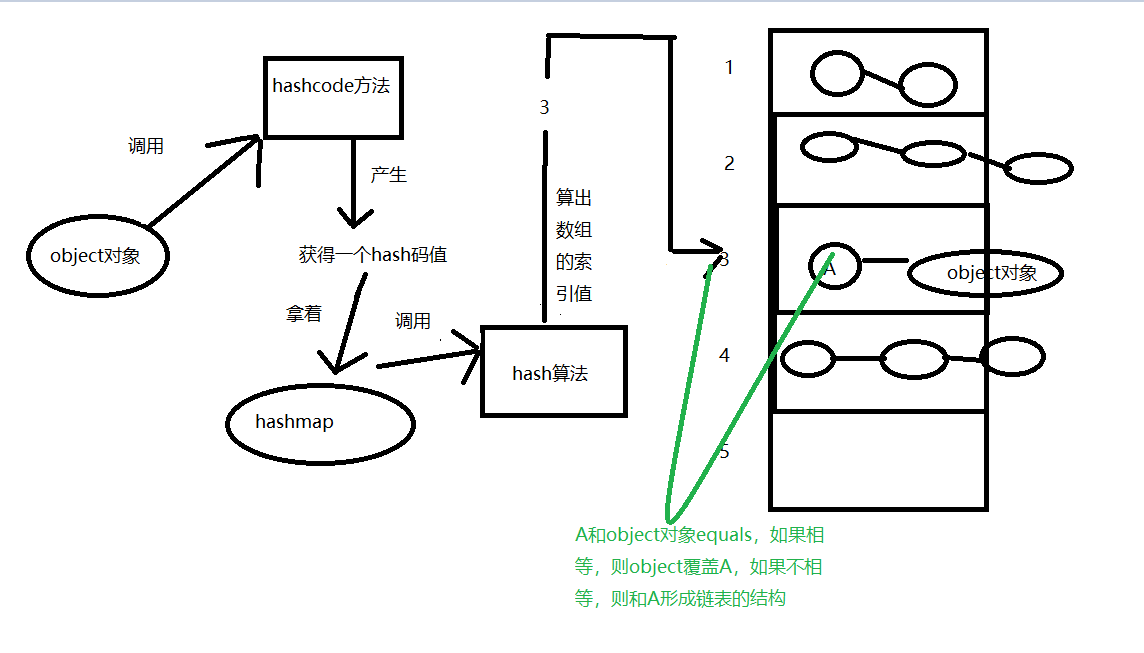
Java8的4个特性：

**1、速度更快：**主要体现在对hashmap更改和内存的更改

**对hashmap的更改**

Java8之前object存入hashmap的流程为：object调用hashcode方法产生一个hash码值，然后hashmap拿着hash码值调用hash算法计算出数组的索引值，如果这个索引值下有内容，则改object对象和改内容一个一个equals过去，如果都不想等，则形成链表，如果相等，则覆盖原来的。



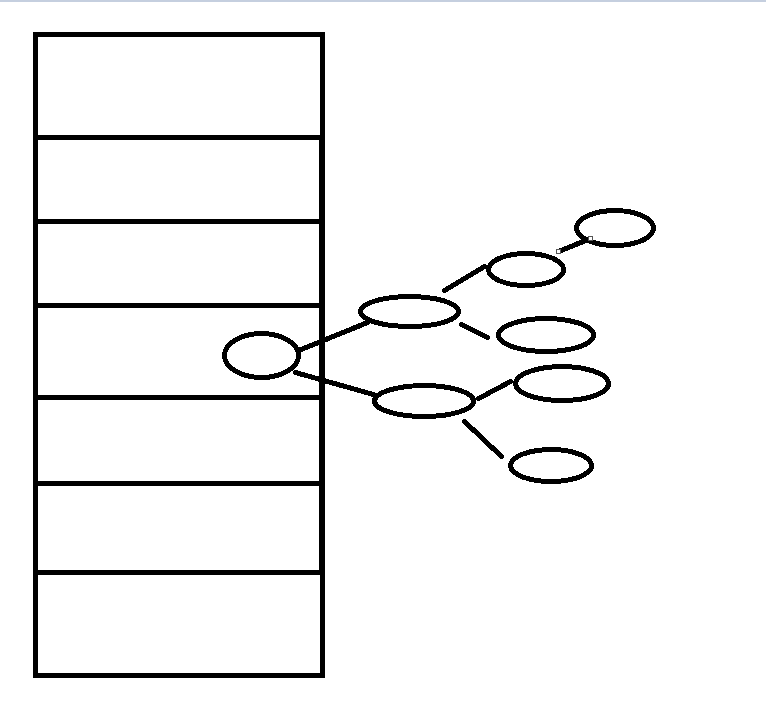
但是存着的问题是，无论hashcode和equals方法重写的有多好，数组的索引始终只有那么几个（默认为16），这样一来和不采用hash算法的结果一样，每一个都要equals的话，碰撞个数（有多少个对象同时在一个索引下面就有多少个碰撞）多就严重影响速度，于是hashmap提供了一个加载因子（默认为0.75），即索引有百分之75使用了就进行扩容，但是就算有扩容还是避免不了碰撞（比如说，要查询一个值，先找到索引值为1，但是1中有很多元素，那就需要一个一个进行判断过去）从而导致效率地下。

于是，java8改变了数组中对象存放的方式：

Java8之前是：数组-链表

Java8之后是：数组-链表或红黑树（二叉树的一种）

当链表中的个数大于8，并且haxmap总个数大于64，java8将会把链表变成红黑树

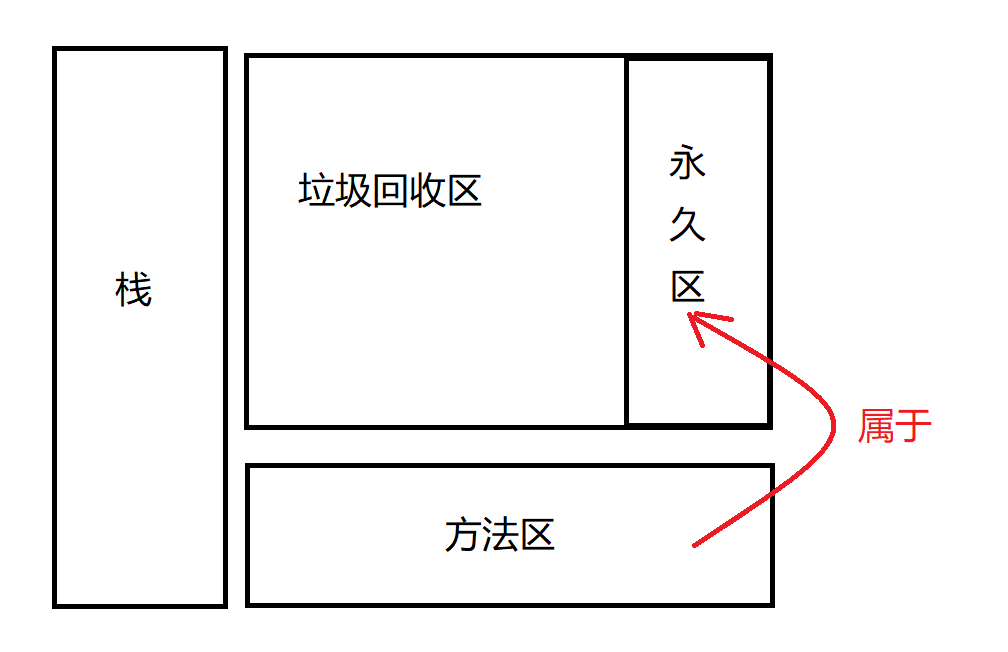


好处：除了添加以外，其他效率都要比链表快

注意：java8之前，添加链表是添加在链表最前面，java8之后是添加到链表的最后面，红黑树添加时，需要比较hashcode，根据大小决定位置，这样就导致查询删除效率高。

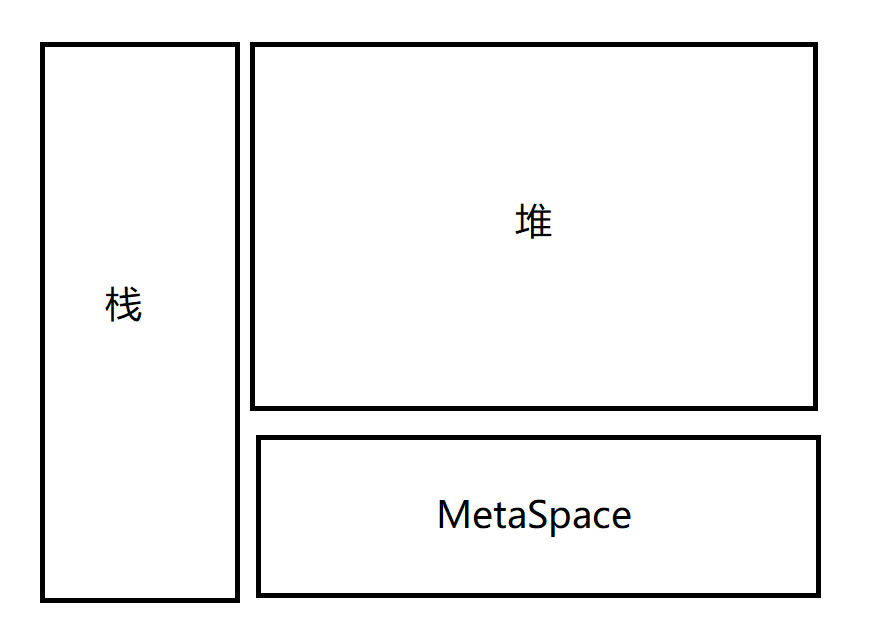
**内存的更改**

Java8之前：



方法区是堆的永久区域的一部分，堆中除了永久区就是垃圾回收区，永久区中存的是类信息，类加载信息，永久区的内容几乎不会被垃圾回收机制回收，注意是几乎不会不是不会，只是回收的条件比较苛刻。

Java8之后：



java8之后没有永久区了，取而代之的是MetaSpace（元空间），他与原来的方法区不同的是，MetaSpace使用的是物理内存，默认是物理内存（8G）多大，元空间就有多大,当物理内存用完或者将近塞满的时候，元空间才调用垃圾回收机制清理内存，降低了垃圾回收机制运行的概率

premGebSize和MaxpremGenSize这两个参数将变得无效，取而代之的是Metaspace和MaxMetaSpace来配置元空间的大小。

**2、代码更少：**引入了lambda表达式。

**3、Stream-API：**引入了Stream，能让java像sql一样查询筛选。

**4、optional：**最大化的减少了空指针异常。