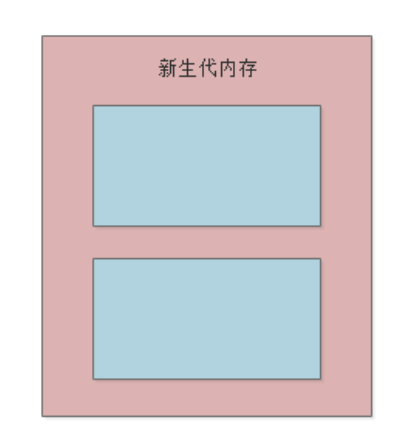
复制算法

1.针对新生代的垃圾回收算法🡪复制算法

简单来说，就是如下图所示



把新生代内存划分为两块内存区域，然后只使用其中一块内存

待那块内存快满的时候，就把里面的存活对象一次性转移到另外一块内存区域，保证没有内存碎片

接着一次性回收原来那块内存区域的垃圾对象，再次空出来一块内存区域。两块内存区域就这么重复着循环使用。

复制算法有什么缺点？

复制算法的缺点其实非常的明显，如果按照上述的思路，大家会发现，假设我们给新生代1G的内存空间，那么只有

512MB的内存空间是可以用的

另外512MB的内存空间是一直要放在那里空着的，然后512MB内存空间满了，就把存活对象转移到另外一块512MB

的内存空间去

从始至终，就只有一半的内存可以用，这样的算法显然对内存的使用效率太低了。

复制算法的优化：Eden区和Survivor区

把新生代内存区域划分为三块：

Eden区 80%

Survivor区 10%

Survivor区 10%

1个Eden区，2个Survivor区，其中Eden区占80%内存空间，每一块Survivor区各占10%内存空间，比如说Eden区有800MB内存，每一块Survivor区就100MB内存，

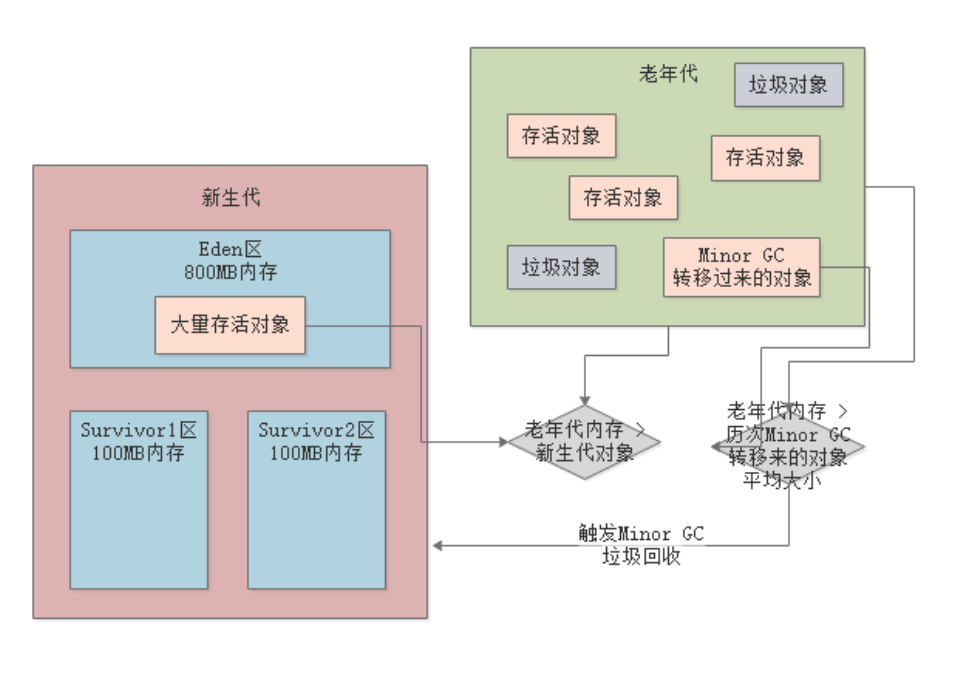
这么做最大的好处，就是只有10%的内存空间是被闲置的，90%的内存都被使用上了

无论是垃圾回收的性能，内存碎片的控制，还是说内存使用的效率，都非常的好。

老年代进行垃圾回收算法

老年代采取的是标记整理算法，这个过程说起来比较简单

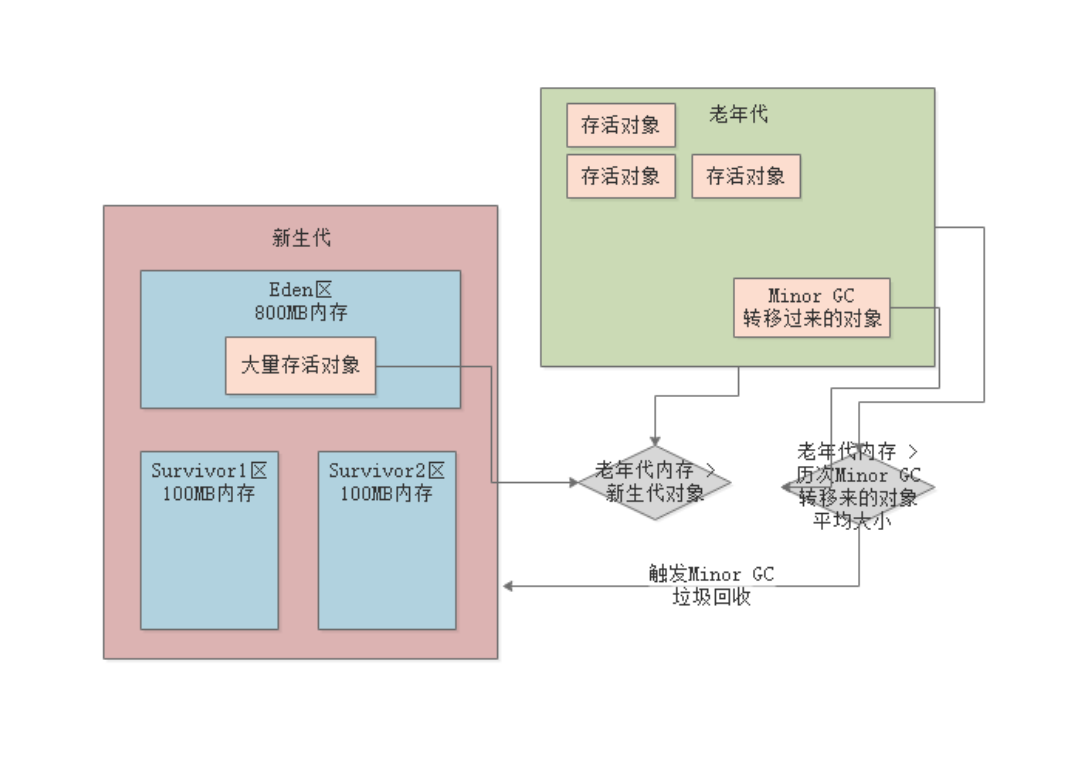
大家看下图，首先标记出来老年代当前存活的对象，这些对象可能是东一个西一个的。



接着会让这些存活对象在内存里进行移动，把存活对象尽量都挪动到一边去，让存活对象紧凑的靠在一起，避免垃圾

回收过后出现过多的内存碎片

然后再一次性把垃圾对象都回收掉，大家看下图。



大家一定要注意一点，这个老年代的垃圾回收算法的速度至少比新生代的垃圾回收算法的速度慢10倍。

如果系统频繁出现老年代的Full GC垃圾回收，会导致系统性能被严重影响，出现频繁卡顿的情况。