[**深入理解JVM(四)——对象内存的分配策略**](https://blog.csdn.net/u010425776/article/details/51192448)

<https://blog.csdn.net/u010425776/article/details/51192448>

Java所承诺的自动内存管理主要是针对对象内存的回收和对象内存的分配。

在Java虚拟机的五块内存空间中，程序计数器、Java虚拟机栈、本地方法栈内存的分配和回收都具有确定性，一般在编译阶段就能确定需要分配的内存大小，并且由于都是线程私有，因此它们的内存空间都随着线程的创建而创建，线程的结束而回收。也就是这三个区域的内存分配和回收都具有确定性，垃圾回收器不需要在这里花费太大的精力。

而Java虚拟机中的方法区因为是用来存储类信息、常量、静态变量，这些数据的变动性较小，因此不是Java内存管理重点需要关注的区域。

而对于堆，所有线程共享，所有的对象都需要在堆中创建和回收。虽然每个对象的大小在类加载的时候就能确定，但对象的数量只有在程序运行期间才能确定，因此堆中内存的分配具有较大的不确定性。此外，对象的生命周期长短不一，因此需要针对不同生命周期的对象采用不同的内存回收算法，增加了内存回收的复杂性。

综上所述：Java自动内存管理最核心的功能是堆内存中对象的分配与回收。 

## 对象优先在Eden区中分配

目前主流的垃圾收集器都会采用分代回收算法，因此需要将堆内存分为新生代和老年代。

在新生代中为了防止内存碎片问题，因此垃圾收集器一般都选用“复制”算法。因此，堆内存的新生代被进一步分为：Eden区＋Survior1区＋Survior2区。

每次创建对象时，首先会在Eden区中分配。   
若Eden区已满，则在Survior1区中分配。   
若Eden区＋Survior1区剩余内存太少，导致对象无法放入该区域时，就会启用“分配担保”，将当前Eden区＋Survior1区中的对象转移到老年代中，然后再将新对象存入Eden区。 

## 大对象直接进入老年代

所谓“大对象”就是指一个占用大量连续存储空间的对象，如数组。

当发现一个大对象在Eden区＋Survior1区中存不下的时候就需要分配担保机制把当前Eden区＋Survior1区的所有对象都复制到老年代中去。   
我们知道，一个大对象能够存入Eden区＋Survior1区的概率比较小，发生分配担保的概率比较大，而分配担保需要涉及到大量的复制，就会造成效率低下。   
因此，对于大对象我们直接把他放到老年代中去，从而就能避免大量的复制操作。   
那么，什么样的对象才是“大对象”呢？

### 通过-XX:PretrnureSizeThreshold参数设置大对象

该参数用于设置大小超过该参数的对象被认为是“大对象”，直接进入老年代。   
**注意：**该参数只对Serial和ParNew收集器有效。 

## 生命周期较长的对象进入老年代

老年代用于存储生命周期较长的对象，那么我们如何判断一个对象的年龄呢？

新生代中的每个对象都有一个年龄计数器，当新生代发生一次MinorGC后，存活下来的对象的年龄就加一，当年龄超过一定值时，就将超过该值的所有对象转移到老年代中去。

### 使用-XXMaxTenuringThreshold设置新生代的最大年龄

设置该参数后，只要超过该参数的新生代对象都会被转移到老年代中去。 

## 相同年龄的对象内存超过Survior内存一半的对象进入老年代

如果当前新生代的Survior中，年龄相同的对象的内存空间总和超过了Survior内存空间的一半，那么所有年龄相同的对象和超过该年龄的对象都被转移到老年代中去。无需等到对象的年龄超过MaxTenuringThreshold才被转移到老年代中去。 

## “分配担保”策略详解

当垃圾收集器准备要在新生代发起一次MinorGC时，首先会检查“老年代中最大的连续空闲区域的大小 是否大于 新生代中所有对象的大小？”，也就是老年代中目前能够将新生代中所有对象全部装下？

若老年代能够装下新生代中所有的对象，那么此时进行MinorGC没有任何风险，然后就进行MinorGC。

若老年代无法装下新生代中所有的对象，那么此时进行MinorGC是有风险的，垃圾收集器会进行一次预测：根据以往MinorGC过后存活对象的平均数来预测这次MinorGC后存活对象的平均数。

如果以往存活对象的平均数小于当前老年代最大的连续空闲空间，那么就进行MinorGC，虽然此次MinorGC是有风险的。

如果以往存活对象的平均数大于当前老年代最大的连续空闲空间，那么就对老年代进行一次Full GC，通过清除老年代中废弃数据来扩大老年代空闲空间，以便给新生代作担保。

这个过程就是分配担保。

**注意：**   
1. 分配担保是老年代为新生代作担保；   
2. 新生代中使用“复制”算法实现垃圾回收，老年代中使用“标记-清除”或“标记-整理”算法实现垃圾回收，只有使用“复制”算法的区域才需要分配担保，因此新生代需要分配担保，而老年代不需要分配担保。