# JVM-垃圾收集器与内存分配策略

## *标记对象是生存还是死亡

### 1，引用计数法

- 给对象添加一个计数器，每当有一个地方引用它时，计数器值就加1，当引用失效的时候，计数器就减1.当任何时刻为0的时候对象就不在使用。
- 优点：实现简单，判定效率也高。
- 缺点：难以解决对象之家相互循环引用的问题。

### 2，可达性分析算法

- 该算法的基本思路就是从一系列成为GC Root的对象为起始点，从这些节点开始向下搜索，搜索走过的路径成为引用链，当一个对象不在引用链上的时候，就说明该对象是不可用的。
- GC Roots对象包含以下几种：
    - 虚拟机栈中引用的对象
    - 方法区中类静态属性引用的对象
    - 方法区中常量引用的对象
    - 本地方法栈中引用的对象
- 回收方法区中类的情况(下面的条件必须同时满足)
    - 该类所有的实例已经被回收，不存在类的实例对象
    - 加载该类的ClassLoader已经被回收
    - 该类对应的java.lang.Class对象没有任何地方被引用，无法在任何地方通过反射访问该类。

### 3，垃圾回收算法

标记清除法，复制算法，标记整理法

- 标记清除法
    - 首先标记要回收的对象，然后再统一回收被标记的对象。
    - 缺点
        - 效率问题：标记和清除两者的效率都不是的很高
        - 空间问题：标记清楚之后会产生内存碎片，当分配的对象所占内存空间较大的时候，会造成空间的浪费。
- 复制算法(适用于新生代)
    - 将内存按照容量划分为大小相等的两块，每次只使用其中的一块，当当前内存块用完之后，就将还存活的对象复制到另一个内存块中，然后再将当前内存块清理掉。
    - 实际上是将内存分成了一个较大的Eden区，和两块较小Survivor区，每次使用Eden区和一块Survivor区，当回收的时候，就将Eden区和Survivor区中的东西复制到另一个Survivor区中。
- 标记整理法(适用于老年代)
    - 将存户的对象都想一端移动，然后清理掉边界以外的内存。
- 分代收集算法

    - 只是将内存按照对象的存活周期分成了几个区域，对于新生代(对象的存活率较低)来说：一般采用复制算法，对于老年代(对象的存活率较高)来说，一般采用标记-整理法或者标记-清除算法。
- 长期存活的对象将进入老年代，每次GC之后，对象的年龄就增加一岁，一般来说，当年龄达到15岁(默认值)之后，就将晋升为老年代。
- 大对象直接进入老年代

## *垃圾收集器

- Serial收集器(新生代收集器，采用复制算法来进行GC)
    - 最基本，发展历史最悠久的收集器。
    - 单线程收集器，它在进行垃圾收集时，必须暂停其他所有的工作线程，直到它收集完成结束。
    - 特点：简单而高效，对于限定单个CPU环境来说，Serial收集器由于没有线程交互的开销，专心做垃圾收集自然可以获得最高的单线程收集效率。
- ParNew收集器(新生代收集器，采用复制算法来进行GC)
    - 本质上就是Seria收集器的多线程版本.
    - 在单CPU的环境中绝对不会有比Serial收集器更好的效果，由于ParNew收集器要进行线程之间的交互的开销。
- Paraller Scavenge收集器(新生代收集器，采用复制算法来进行GC)
    - 多线程收集器，和其他收集器关注的方向不太一样，CMS等收集器的关注点是尽可能地缩短垃圾收集时用户线程地停顿的时间，而Paraller Scavenge收集器关注的是达到一个可控制的吞吐量，所谓吞吐量就是CPU用于运行用户代码地时间与CPU总消耗地时间的比值。
    - 三个参数，一个是用来控制最大垃圾收集停顿时间，另一个是设置吞吐量的大小的参数，最后一个就是开关参数，当该参数打开之后，就不要手动设置新生代中Eden区，survivor区地比例关系，虚拟机会根据当前的系统自动调节最合适地停顿时间以及吞吐量。
- Serial Old收集器
    - 同样也是一个单线程收集器，使用标记-整理算法去回收老年代中的内存空间。
- Parallel Old收集器
    - 使用多线程对老年代进行垃圾回收，采用标记-整理算法。在JDK1.6之后才有的。
- CMS收集器
    - 是一种以获取最短回收停顿时间为目标的收集器。采用标记-清除算法实现
    - 运作过程
        - 初始标记：仅仅只是标记一下GC Roots能直接关联到的对象。
        - 并发标记：进行GC Roots 追踪的过程。
        - 重新标记：为了修正并发标记期间因用户程序继续运作而导致标记产生变动的那一部分对象的标记记录。
        - 并发清除
- G1收集器
    - 当今收集器技术发展的最前沿成果之一。是一款面向服务器端应用的垃圾处理器。
    - 在G1收集时，它将整个Java堆划分为多个大小相等独立的独立区域，新生代和老年代不再是物理隔离的了。
    - 之所以能建立可预测的停顿时间模型，是因为它可以有计划避免在整个Java堆中进行全区域的垃圾收集，G1跟踪各个区域里面的垃圾堆积的价值大小，在后台维护一个优先列表，每次根据允许的收集时间，优先收集价值最大的区域。