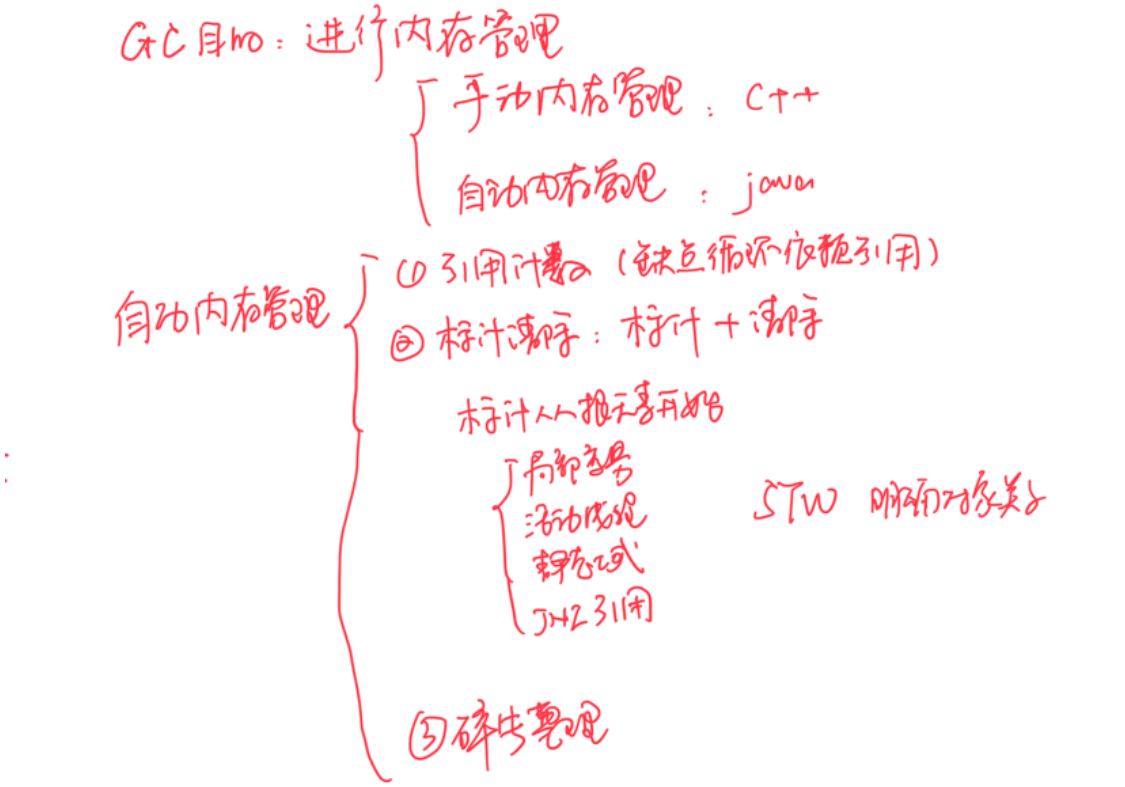
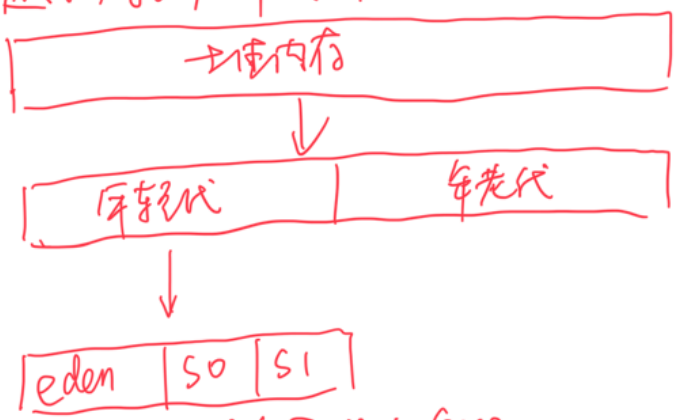
GC背景原理



因为在内存管理过程中需要STW所有用对象关闭，对象越多，停顿时间越长，把新生命周期提出了分代



Java 8 删除了永久代采用 MetaSpace

-XX：MaxMetaspaceSize=256m

垃圾回收

①标记：标记出可达对象

②删除不可达对象

a、清除：需要空闲列表记录区域大小

b. 整理：对象进行移动整理，[illegible]

c. 复制：将对象从一个区移动复制到另一个区

常见的GC算法

1、串行GC算法 -XX:+UseSerialGC

年轻代：标记 复制 mark-copy

老年代：标记 清除 整理 mark-sweep-compact

2、并行GC算法 -XX:+UseParallelGC

-XX:ParallelGCThreads=N 指定GC线程数

默认为cpu核心数

年轻代：mark-copy

老年代：mark-sweep-compact

优点可以并行

3、CMS算法：+XX:+UseConcMarkSweepGC

Mostly Concurrent Mark and Sweep GC

最大并发标记清除垃圾收集器

目的是避免老年代GC时间卡顿

年轻代：mark-copy（并行）

老年代：mark-sweep（通过free-list管理内存回收）

处理过程：

- initial mark 初始标记 先标记[illegible] (STW)
- concurrent mark 并发标记 从这个开始遍历标记
- concurrent preclean 并发预清理 待清理的先清理
- concurrent Abortable Preclean
- final remark 最终标记 (STW)
- concurrent sweep 并发清除
- concurrent reset 并发重置为下一次做准备

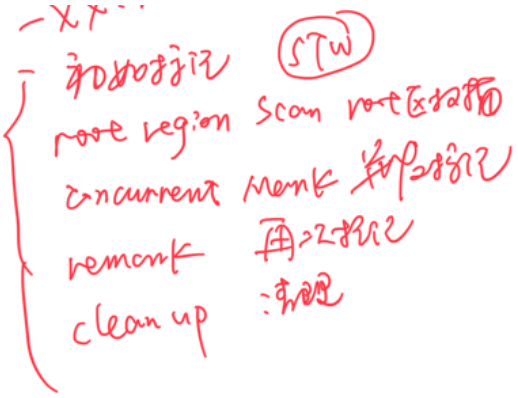
G1回收算法 Garbage-First

设计目标：将STW停顿时间和分布可预期可配置

G1算法不再按年代划分，而是按照小区域划分(2048个)

+XX:+UseG1GC

-XX:MaxGCPauseMills=50


STW
root region scan
concurrent mark
remark
clean up