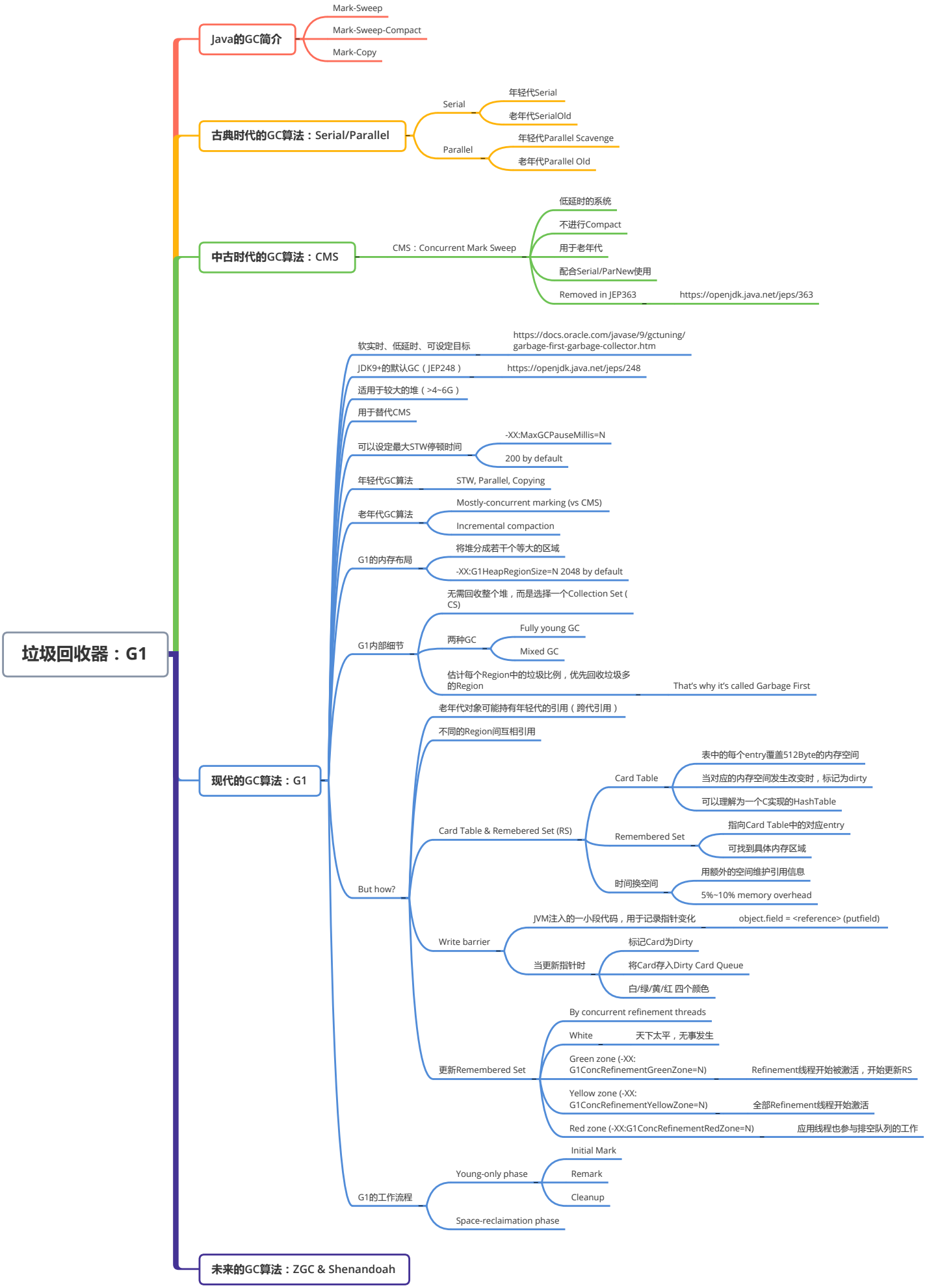
垃圾回收器：G1
Java的GC简介
Mark-Sweep
Mark-Sweep-Compact
Mark-Copy
古典时代的GC算法：Serial/Parallel
Serial
年轻代Serial
老年代SerialOld
Parallel
年轻代Parallel Scavenge
老年代Parallel Old
中古时代的GC算法：CMS
CMS：Concurrent Mark Sweep
低延时的系统
不进行Compact
用于老年代
配合Serial/ParNew使用
Removed in JEP363
https://openjdk.java.net/jeps/363
现代的GC算法：G1
软实时、低延时、可设定目标
https://docs.oracle.com/javase/9/gctuning/garbage-first-garbage-collector.htm
JDK9+的默认GC（JEP248）
https://openjdk.java.net/jeps/248
适用于较大的堆（>4~6G）
用于替代CMS
可以设定最大STW停顿时间
-XX:MaxGCPauseMillis=N
200 by default
年轻代GC算法
STW, Parallel, Copying
老年代GC算法
Mostly-concurrent marking (vs CMS)
Incremental compaction
G1的内存布局
将堆分成若干个等大的区域
-XX:G1HeapRegionSize=N 2048 by default
G1内部细节
无需回收整个堆，而是选择一个Collection Set (CS)
两种GC
Fully young GC
Mixed GC
估计每个Region中的垃圾比例，优先回收垃圾多的Region
That's why it's called Garbage First
But how?
老年代对象可能持有年轻代的引用（跨代引用）
不同的Region间互相引用
Card Table & Remebered Set (RS)
Card Table
表中的每个entry覆盖512Byte的内存空间
当对应的内存空间发生改变时，标记为dirty
可以理解为一个C实现的HashTable
Remembered Set
指向Card Table中的对应entry
可找到具体内存区域
时间换空间
用额外的空间维护引用信息
5%~10% memory overhead
Write barrier
JVM注入的一小段代码，用于记录指针变化
object.field = <reference> (putfield)
当更新指针时
标记Card为Dirty
将Card存入Dirty Card Queue
白/绿/黄/红 四个颜色
更新Remembered Set
By concurrent refinement threads
White
天下太平，无事发生
Green zone (-XX: G1ConcRefinementGreenZone=N)
Refinement线程开始被激活，开始更新RS
Yellow zone (-XX: G1ConcRefinementYellowZone=N)
全部Refinement线程开始激活
Red zone (-XX:G1ConcRefinementRedZone=N)
应用线程也参与排空队列的工作
G1的工作流程
Young-only phase
Initial Mark
Remark
Cleanup
Space-reclaimation phase
未来的GC算法：ZGC & Shenandoah