GC 日志

- 代码案例

```
public class MyTest1 {

    public static void main(String[] args) {
        int size = 1024 * 1024;
        byte[] myAlloc1 = new byte[2 * size];
        byte[] myAlloc2 = new byte[2 * size];
        byte[] myAlloc3 = new byte[3 * size];
        //byte[] myAlloc4 = new byte[2 * size];

        System.out.println("hello world");
    }
}
```

- 运行时JVM参数

```
-verbose:gc  //报告每个垃圾收集事件，输出虚拟机中垃圾回收的详细日志
-Xms20M      //初始化堆空间大小
-Xmx20M      //最大堆空间大小，-Xms和-Xmx设置成一样可以避免垃圾回收造成的抖动问题
-Xmn10M      //新生代内存大小
-XX:+PrintGCDetails //打印GC回收详细日志
-XX:SurvivorRatio=8 //新生代中Eden和Survivior的比例默认为8那么 Eden(对象发源地):Form Survivor:To Survivor = 8:1:1
```

- GC日志

```
[GC (Allocation Failure) [PSYoungGen: 6195K->624K(9216K)] 6195K->4728K(19456K), 0.0083276 secs] [Times: user=0.01 sys=0.00, real=0.01 secs]
hello world
Heap
 PSYoungGen      total 9216K, used 4019K [0x00000007bf600000, 0x00000007c0000000, 0x00000007c0000000)
  eden space 8192K, 41% used [0x00000007bf600000,0x00000007bf950dd8,0x00000007bfe00000)
  from space 1024K, 60% used [0x00000007bfe00000,0x00000007bfe9c010,0x00000007bff00000)
  to   space 1024K, 0% used [0x00000007bff00000,0x00000007bff00000,0x00000007c0000000)
 ParOldGen       total 10240K, used 4104K [0x00000007bec00000, 0x00000007bf600000, 0x00000007bf600000)
  object space 10240K, 40% used [0x00000007bec00000,0x00000007bf002020,0x00000007bf600000)
 Metaspace       used 3067K, capacity 4496K, committed 4864K, reserved 1056768K
  class space    used 336K, capacity 388K, committed 512K, reserved 1048576K
```

- GC日志解析

- GC 标记表示触发了一次GC回收， 前面没有Full修饰，标明这是一个Minor GC，注意它不代表只是GC新生代，并且现有的不管是新生代还是老年代都会STW
- Allocation Failure 表示年轻代没有足够的内存能够存储新的数据
- PSYoungGen 表示本次GC发生在年轻代使用的是 Parallel Scavenge 收集器
- 6195K->624K(9216K) 表示新生代 6195 表示当前新生代已使用容量, 624 表示GC后的使用容量， 9216 表示该区域的总容量，单位:KB
- 6195K->4728K(19456K) 表示总的堆内存 三个参数分别表示: 堆回收之前大小，堆回收后的大小，堆的总大小
- 0.0083276 secs 表示GC耗时，单位是秒
- [Times: user=0.01 sys=0.00, real=0.01 secs] 分别表示用户耗时，内核耗时，总耗时
- PSYoungGen total 9216K, used 4019K 表示新生代总内存 9216K, 已使用 4019K
    - eden space 8192K, 41% used , eden 区内存大小 8192K，已使用 41%
    - from space 1024K, 60% used , from Survivor 区内存大小 1024K，已使用 60%
    - to space 1024K, 0% used , eden Survivor 区内存大小 1024K，已使用 0%
- ParOldGen total 10240K, used 4104K 老年代的总内存大小10240K， 已使用 4104K

- 新生代回收内存：6195-624 = 5571k //执行完gc后，新生代执行垃圾回收后堆空间回收内存
- 实际的回收内存：6195-4728 = 1467k //执行完gc后，总的堆空间释放的容量
- 年老代使用内存：5571-1467 = 4104k (新生代的释放内存：会分为1.新生代->老年代，2.真实被回收)

Full GC

GC 日志

```
[GC (Allocation Failure) [PSYoungGen: 6195K->614K(9216K)] 6195K->4710K(19456K), 0.0127704 secs] [Times: user=0.01 sys=0.01, real=0.01 secs]
[GC (Allocation Failure) --[PSYoungGen: 6999K->6999K(9216K)] 11095K->15199K(19456K), 0.0106521 secs] [Times: user=0.00 sys=0.00, real=0.01 secs]
[Full GC (Ergonomics) [PSYoungGen: 6999K->2489K(9216K)] [ParOldGen: 8200K->8192K(10240K)] 15199K->10681K(19456K), [Metaspace: 3070K->3070K(1056768K)], 0.0075955 secs] [Times: user=0.01 sys=0.00, real=0.00 secs]
hello world
Heap
 PSYoungGen      total 9216K, used 4977K [0x00000007bf600000, 0x00000007c0000000, 0x00000007c0000000)
  eden space 8192K, 60% used [0x00000007bf600000,0x00000007bfadc7a8,0x00000007bfe00000)
  from space 1024K, 0% used [0x00000007bfe00000,0x00000007bfe00000,0x00000007bff00000)
  to   space 1024K, 0% used [0x00000007bff00000,0x00000007bff00000,0x00000007c0000000)
 ParOldGen       total 10240K, used 8192K [0x00000007bec00000, 0x00000007bf600000, 0x00000007bf600000)
  object space 10240K, 80% used [0x00000007bec00000,0x00000007bf4000d8,0x00000007bf600000)
 Metaspace       used 3150K, capacity 4496K, committed 4864K, reserved 1056768K
  class space    used 346K, capacity 388K, committed 512K, reserved 1048576K
```

- Full GC 一般是年老代触发的GC
- Ergonomics 是以种标识，
- PSYoungGen: 6933K->3507K(9216K)，采用Parallel Scavenge 新生代垃圾收集器
- ParOldGen: 7176K->7168K(10240K)，采用Parallel Old 老年代垃圾收集器，老年代的对象增多，是由于部分来自新生代
- Metaspace: 元空间

注意

- 在新生代内存不足的时候，但是遇到了大对象会直接存储到老年代中。

-verbose:gc //报告每个垃圾收集事件，输出虚拟机中垃圾回收的详细日志
-Xms20M //初始化堆空间大小
-Xmx20M //最大堆空间大小，-Xms和-Xmx设置成一样可以避免垃圾回收造成的抖动问题
-Xmn10M //新生代内存大小
-XX:+PrintGCDetails //打印GC回收详细日志
-XX:PretenureSizeThreshold=4194304 //新创建的对象大小大于这个阈值直接在老年代创建，只有在Serial GC中才能起作用
-XX:+UseSerialGC //使用Serial GC