

# Java Flight Recorder & Mission Control

## 一个高效的性能分析工具

朱光宇(效山)

[guangyu.zhu@aliyun.com](mailto:guangyu.zhu@aliyun.com)

# Agenda

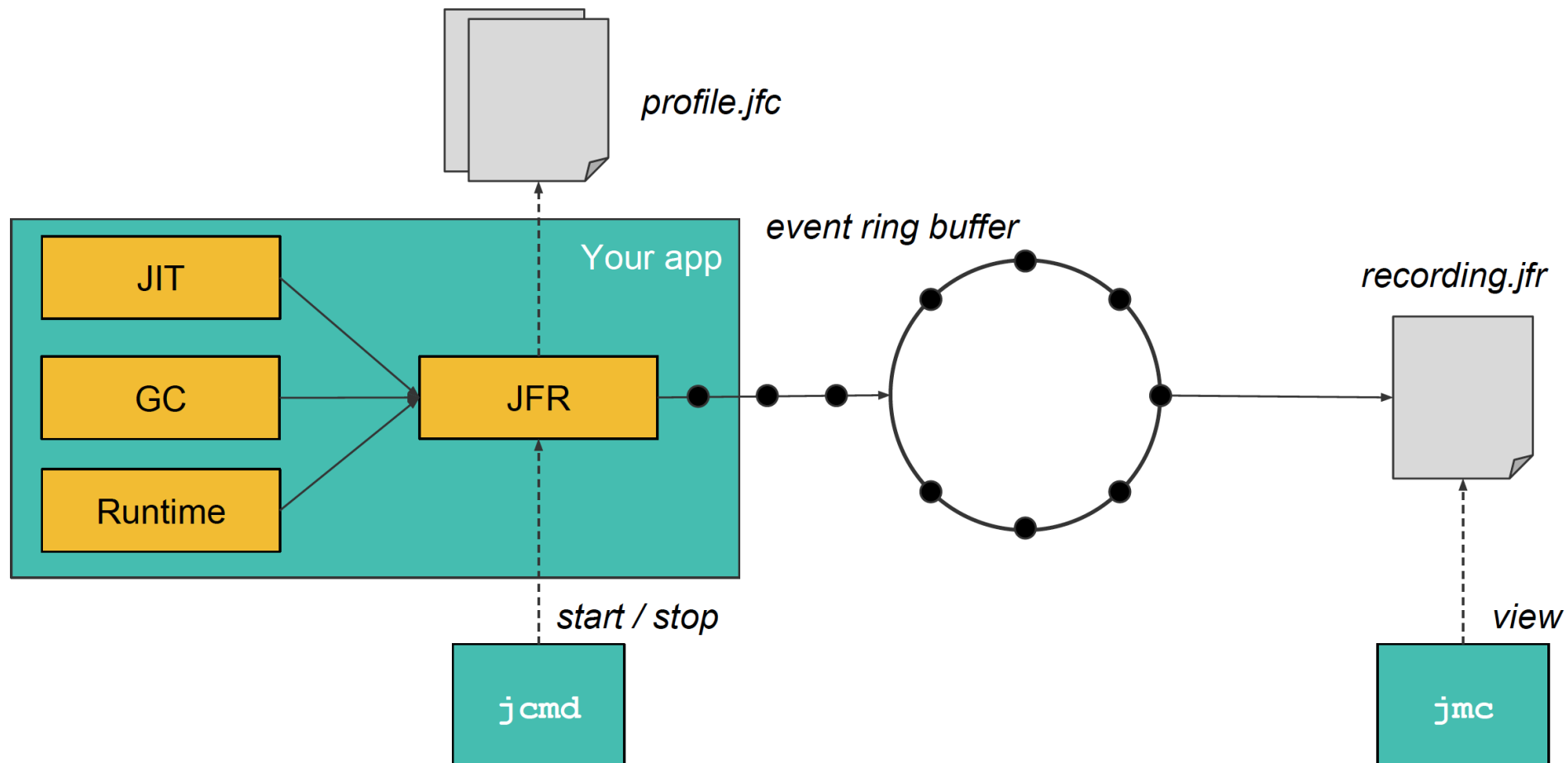
- 什么是Java飞行记录器
- 如何使用飞行记录器优化性能

# Java飞行记录器

- 历史
  - JRockit -> Oracle JDK 7u40 -> Open JDK 11
- 用途
  - 全方位收集Java应用运行时产生的各种事件
  - 性能分析、运行监控、故障排除
- 特点
  - 深入集成到JDK/JVM内部
  - 低开销，适用于生产环境
  - 可扩展，方便的API，可自定义事件



# JFR的整体架构



# JFR事件框架

- Everything is an Event!
- 支持120多种事件
  - 精准的方法采样
  - 详细的GC信息：暂停、引用处理
  - 深入JIT内部：内联、逆优化
  - 详细的Safepoint信息
  - TLAB对象分配

# 自定义事件

```
import jdk.jfr.*;

import java.nio.file.*;
import jdk.jfr.consumer.*;

Path p = Paths.get("recording.jfr");
for (RecordedEvent e : RecordingFile.readAllEvents(p)) {
    System.out.println(e.getStartTime() + " : " + e.getValue("message"));
}

public static void main(String... args) throws IOException {
    HelloWorld event = new HelloWorld();
    event.message = "hello, world!";
    event.commit();
}
```

# 为什么开销这么低

- 高效的采样机制
  - 不需要字节码操纵就可以Profile对象分配
  - 直接从vframe获取调用栈信息
- 高效的事件缓冲机制
  - TLB的使用
  - 内存访问尽量使用原子操作而避免使用锁
- 自描述二进制事件编码格式
  - 配合常量池

二进制编码: 98 80 80 00 87 02 95 ae e4 b2 92 03

a2 f7 ae 9a 94 02 02 01 8d 11 00 00

```
Event size [98 80 80 00]
Event ID [87 02]
Timestamp [95 ae e4 b2 92 03]
Duration [a2 f7 ae 9a 94 02]
Thread ID [02]
Stack trace ID [01]
Payload [fields]
    Loaded Class: [0x8d11]
    Defining ClassLoader: [0]
    Initiating ClassLoader: [0]
```

# Agenda

- 什么是Java飞行记录器
- **如何使用飞行记录器优化性能**



# Java应用性能分析概要

- 性能指标
  - Benchmark : Score
  - 真实应用 : QPS/TPS、RT
- 性能分析关注点
  - 整体性能 : CPU占用率、GC暂停时间、堆使用量、线程活动等
  - 代码执行性能 : 热点方法
  - GC性能 : GC次数、暂停时间、堆使用量
  - 同步性能 : Monitor阻塞时间
  - I/O性能 : 文件I/O、套接字I/O
  - ... ..

# 如何启动Java飞行记录器

- 配置文件
  - \$JAVA\_HOME/jre/lib/jfr/default.jfc, profile.jfc
- 在启动应用时启用飞行记录器
  - XX:+UnlockCommercialFeatures (jdk11不再需要)
  - XX:+FlightRecorder
- 开启记录
  - 通过JVM命令行参数
    - XX:StartFlightRecording=delay=10s,duration=60m,settings=profile,filename=rec.jfr (固定时间记录)
    - XX:StartFlightRecording=defaultrecording=true (持续记录)
    - XX:FlightRecorderOptions=defaultrecording=true,disk=true,repository=/tmp,maxage=8h,settings=default
  - 通过jcmd
    - jcmd \$PID JFR.start
    - jcmd \$PID JFR.dump recording=1 filename=/tmp/r.jfr
    - jcmd \$PID JFR.stop recording=1
  - 通过Mission Control的JVM浏览器

# Demo#1:通过Mission Control启动飞行记录

- Demo
  - 通过MBean建立连接、开启记录

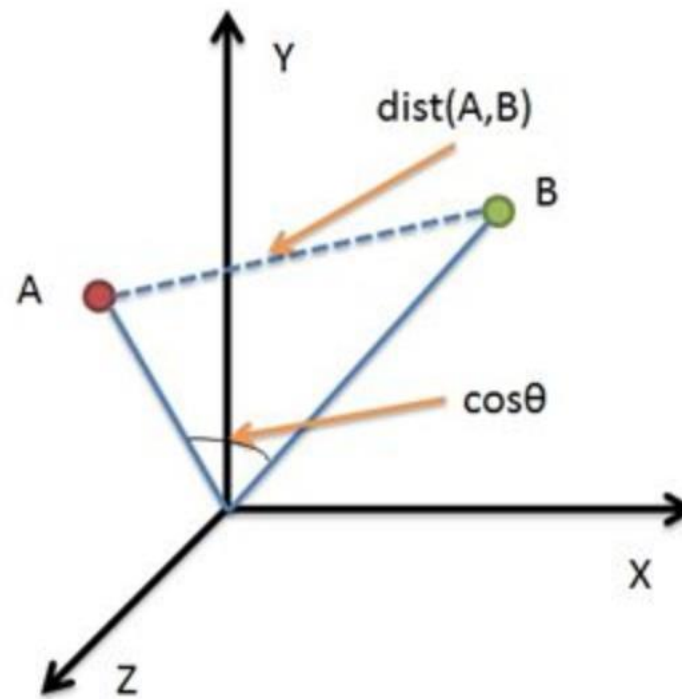
# Demo#2: Mission Control 基本功能

- Demo
  - Mission Control 基本功能



# Demo#3: 热点方法分析

- 应用场景
  - 新闻的相关性、照片的相似度
- Micro benchmark
  - 向量的余弦距离













# 热点方法分析 - 优化效果

优化前

vec 128 time used: **27471**











vec 512 time used: **15787**

堆栈跟踪	计数
▶  int jdk.internal.misc.Unsafe.getIntUnaligned(Object, long, boolean)	168
▶  float VectorCosineFloat.vectorDot512(byte[], byte[], FloatVector\$FloatSpecies, Unsafe)	144
▶  Buffer java.nio.ByteBuffer.limit(int)	56
▶  void java.nio.Buffer.<init>(int, int, int, int)	53
▶  float VectorCosineFloat.vectorDot128(byte[], byte[], FloatVector\$FloatSpecies, Unsafe)	26
▶  Buffer java.nio.Buffer.limit(int)	12
▶  int jdk.internal.misc.Unsafe.convEndian(boolean, int)	12
▶  Buffer java.nio.ByteBuffer.position(int)	10
▶  Buffer java.nio.Buffer.position(int)	6
▶  void java.lang.Object.<init>()	4

使用AVX指令优化后

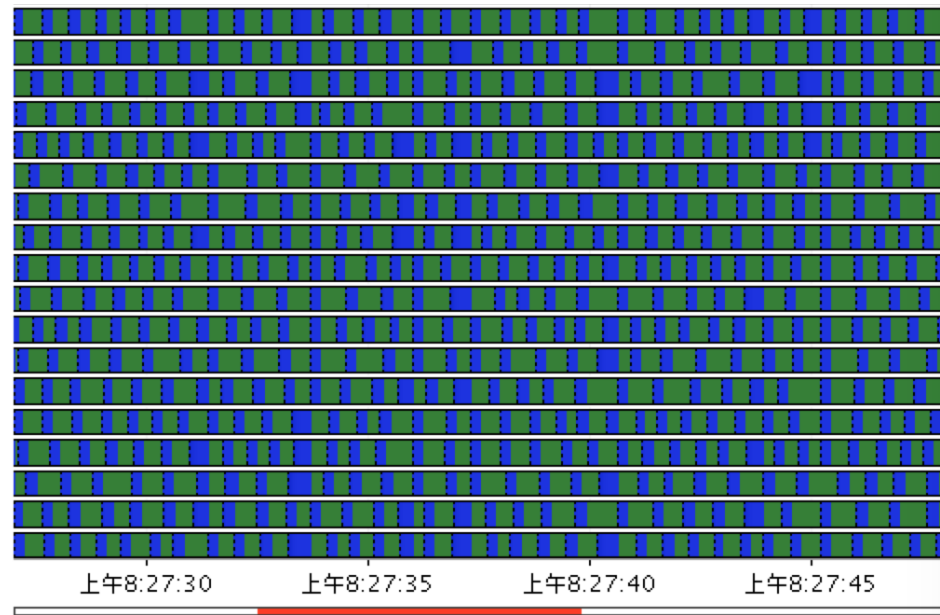
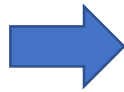
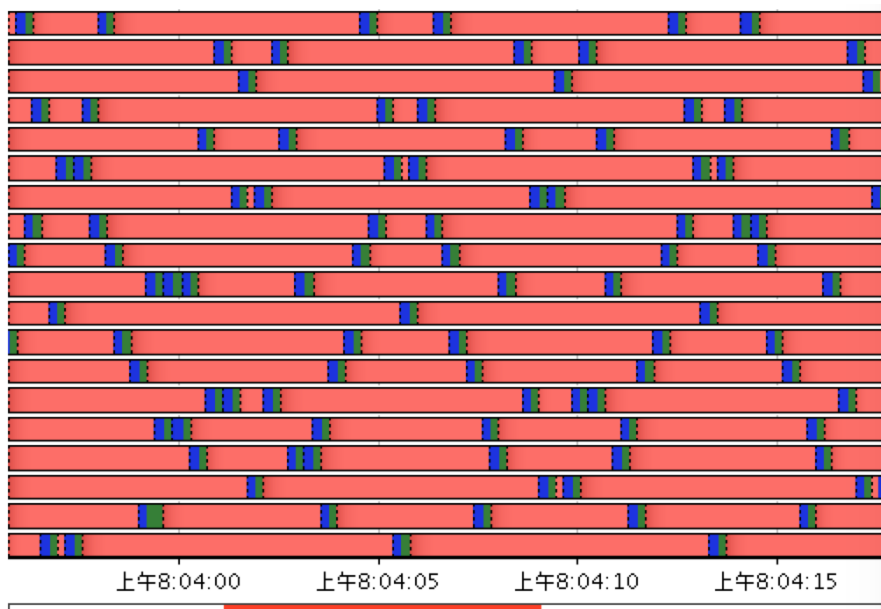
vec 128 time used: **2003**

vec 512 time used: **11536**

堆栈跟踪	计数
▶  ByteBuffer java.nio.ByteBuffer.wrap(byte[], int, int)	133
▶  Float512Vector jdk.incubator.vector.Float512Vector.bOp(Vector, FloatVector\$FBinOp)	112
▶  Long java.lang.Long.valueOf(long)	44
▶  Float512Vector jdk.incubator.vector.Float512Vector\$Float512Species.fromByteArray(b...	14
▶  Vector jdk.incubator.vector.VectorIntrinsics.load(Class, Class, int, Object, long, Object, i...	8
▶  float VectorCosineFloat.vectorDot512(byte[], byte[], FloatVector\$FloatSpecies, Unsafe)	6
▶  Object jdk.incubator.vector.VectorIntrinsics.binaryOp(int, Class, Class, int, Object, Objec...	4
▶  float jdk.incubator.vector.Float512Vector.addAll()	3
▶  Float512Vector jdk.incubator.vector.Float512Vector.sub(Vector)	2
▶  Float512Vector jdk.incubator.vector.Float512Vector.mul(Vector)	2

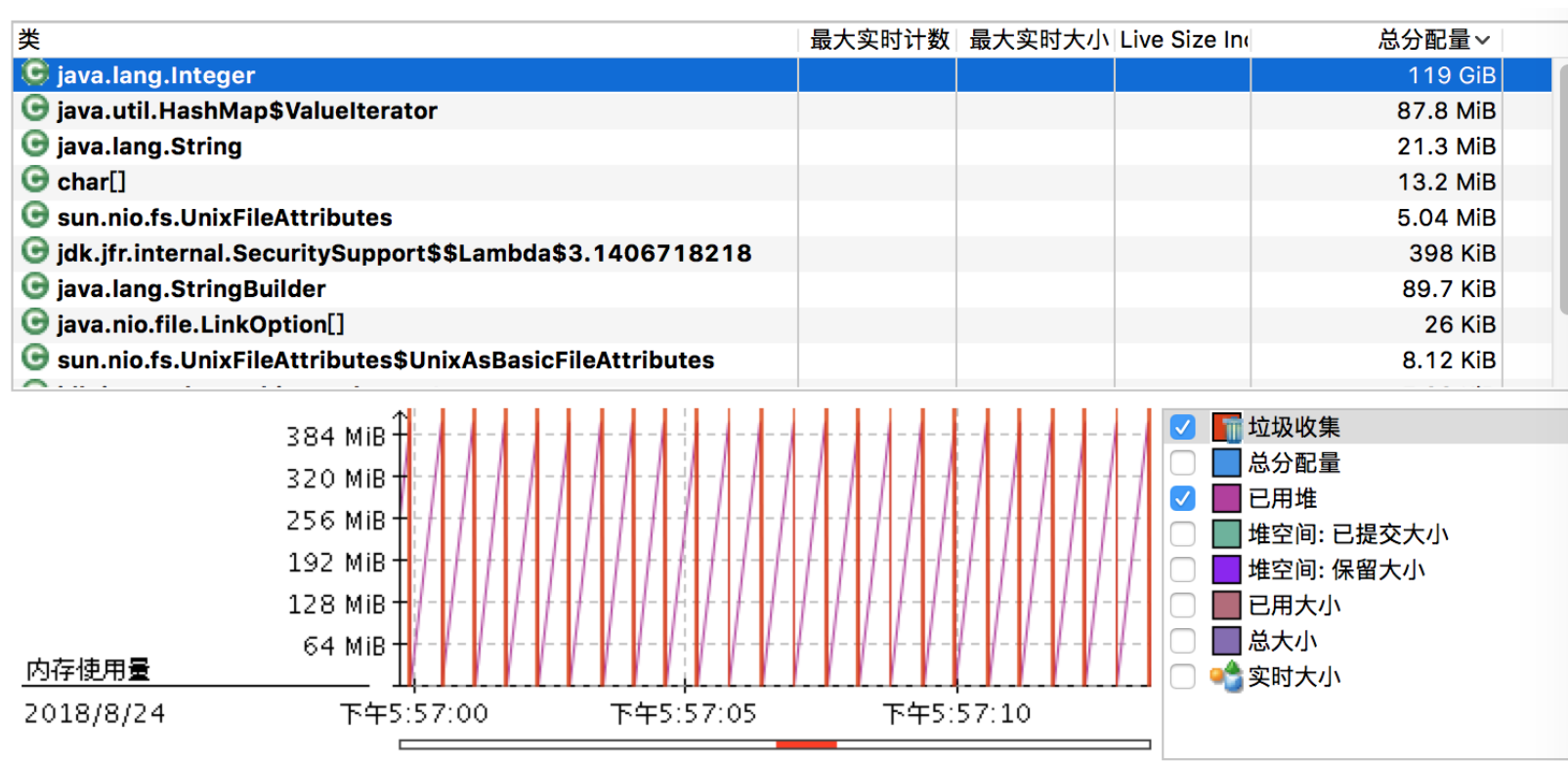
# Demo#4: 线程同步性能分析

- An example from Java Mission Control 6.0 Tutorial



# Demo#5: GC性能分析

- An example from Java Mission Control 6.0 Tutorial





# Reference

- Java Flight Recorder源码
- Java Mission Control 6.0 Tutorial - Marcus Hirt  
<http://hirt.se/downloads/oracle/JMC6Tutorial>

Thanks