# JVM

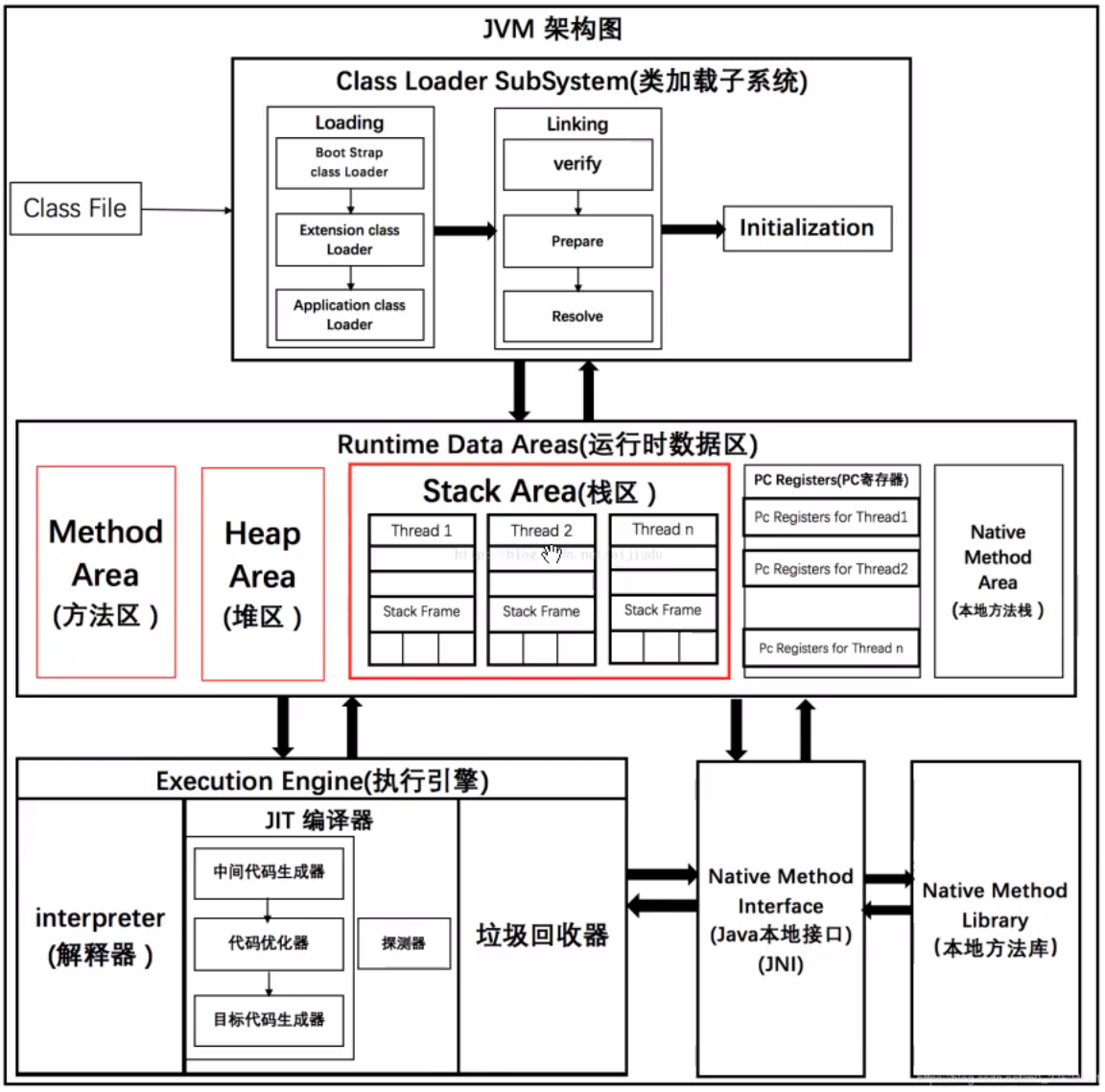
## 狂神

* 请你谈谈你对VM的理解？java8虚拟机和之前的变化更新？
* 什么是OOM，什么是栈溢出StackOverFlowError？怎么分析？
* JVM的常用调优参数有哪些？
* 内存快照如何抓取，怎么分析Dump文件？知道吗？
* 谈谈VM中，类加载器你的认识？

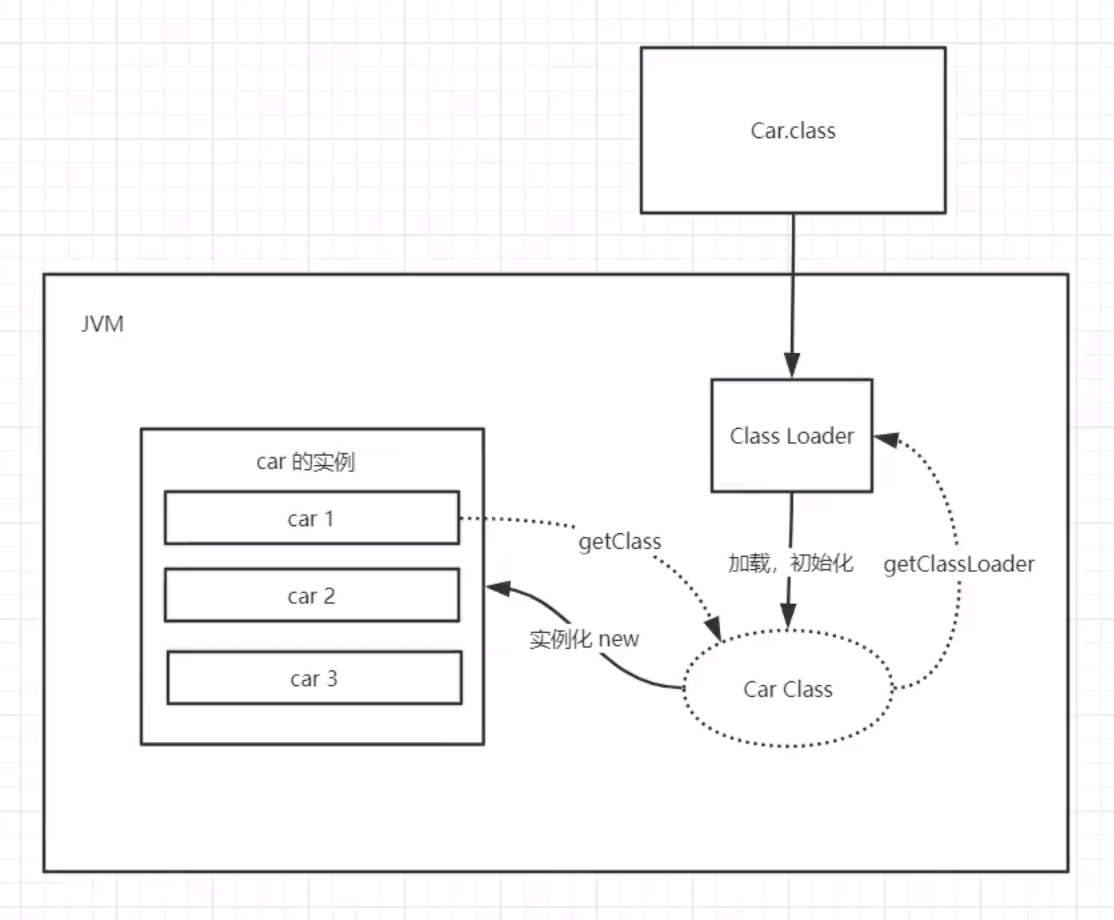
### 1.JVM的位置

JVM∈(win,linux,mac)操作系统∈硬件系统，相当于一个软件

### 2.JVM的体系结构



### 3.类加载器



### 4.双亲委派机制

类加载开始在不同加载器包下找类，

包下的类由该加载器加载，找不到抛出ClassNotFound

按照以下优先级加载：

Bootstrap classLoader 根加载器（同名类该加载器拥有最高优先级）

ExtClassLoader 扩展加载器

AppClassLoader 加载器

UserClasLoader 用户自定义加载器

### 5.沙箱安全机制

域的概念，不同块有不同权限，部分代码有危险所以没这么高运行权限。

### 6.Native

private native void start0();

关键字代表这个方法在本地方法栈中，并且在最终执行时通过JNI去调用例如C、C++、各种硬件等对应方法。

现在通过接口调用，socket，webservice，http

### 7.PC寄存器

给线程计数的，，占用内存空间非常小

### 8.方法区

静态变量，常量，类信息（构造方法，接口定义）运行时的常量池存在方法区中

static final Class 常量池

所有线程共享，

### 9.栈

8大基本类型+对象引用+实例的方法

队列：先进先出（FIFO:First Input First Output）

栈：先进后出，后进先出（羽毛球桶，单口队列。）

栈内存主管程序的运行，生命周期和线程同步，

线程结束，栈内存也就释放了，不存在垃圾回收问题

栈帧：

方法索引

输入输出参数

本地变量

ClasFile：引用

父帧

子帧

### 10.三种利VM

### 11.堆

### 12.新生区、老年区

### 13.永久区

### 14.堆内存调优

### 15.GC

1) 常用算法

### 16.JMM

### 17总结

## 马士兵

### 基础

stop the wolrd (stw): gc会把部分内存挂起扫描垃圾，此时会暂停应用程序访问

G1: 新的回收算法，

what is G1:把内存分成小块range进行管理，以控制stw现象

why G1：最小响应时间，控制好stw

how G1 works:初始标记，并发标记，最终标记，并行复制回收

CMS：旧的回收算法

### 你是不是垃圾

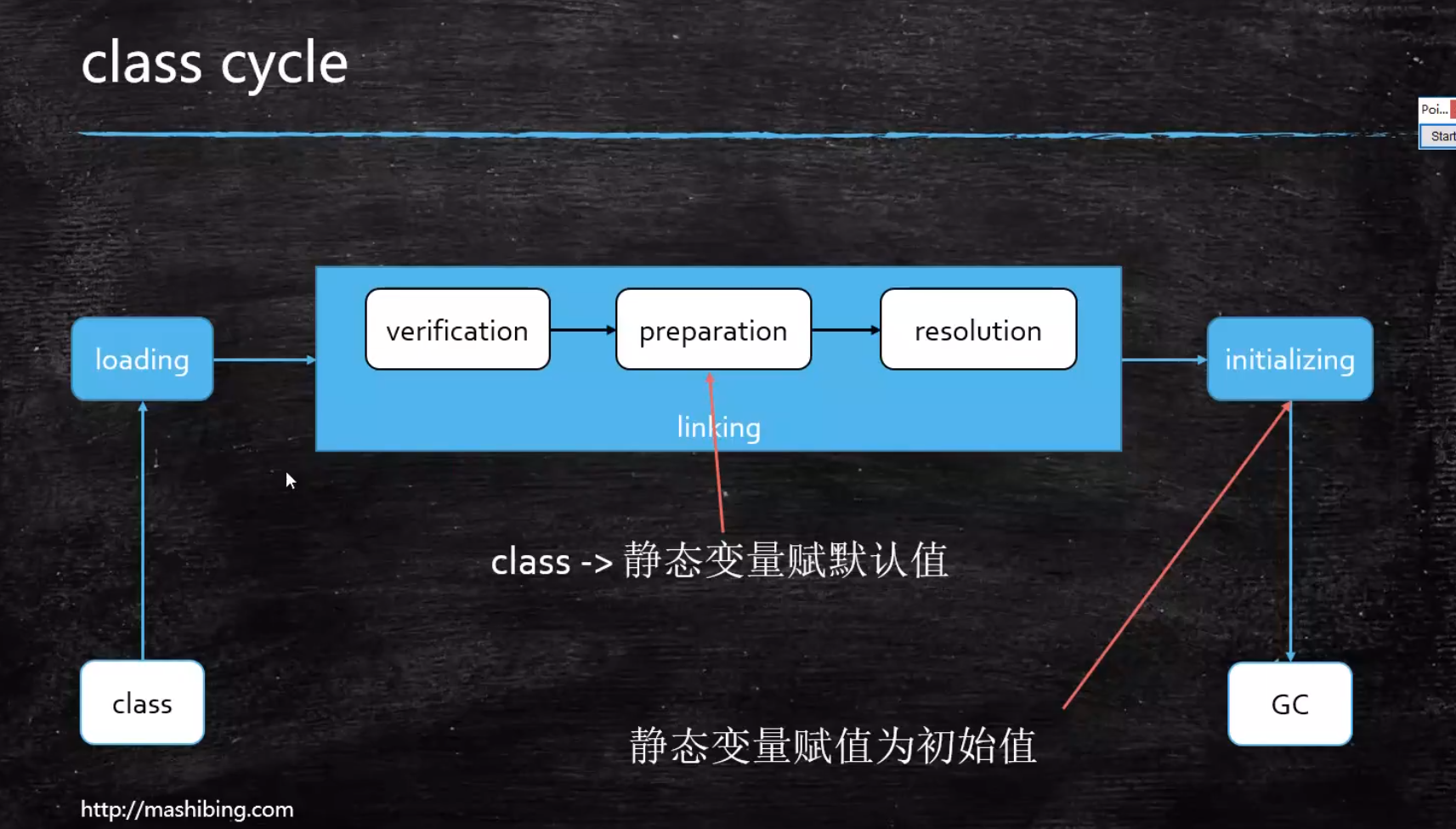
引用计数法：查找你是否有引用

GC Roots根可达算法：处理相互引用的对象

### ·JVM Introduction

### ·Class File format

### ·loading linking initializing



class→内存流程

class文件在硬盘上

用classloader进行loading到内存

linking链接

verification 对class格式校验

preparation 准备，静态变量赋默认值

resolution 解析，方法

initalizing类初始化

gc回收

### ·Java Memory Model

-thread specific

-thread share

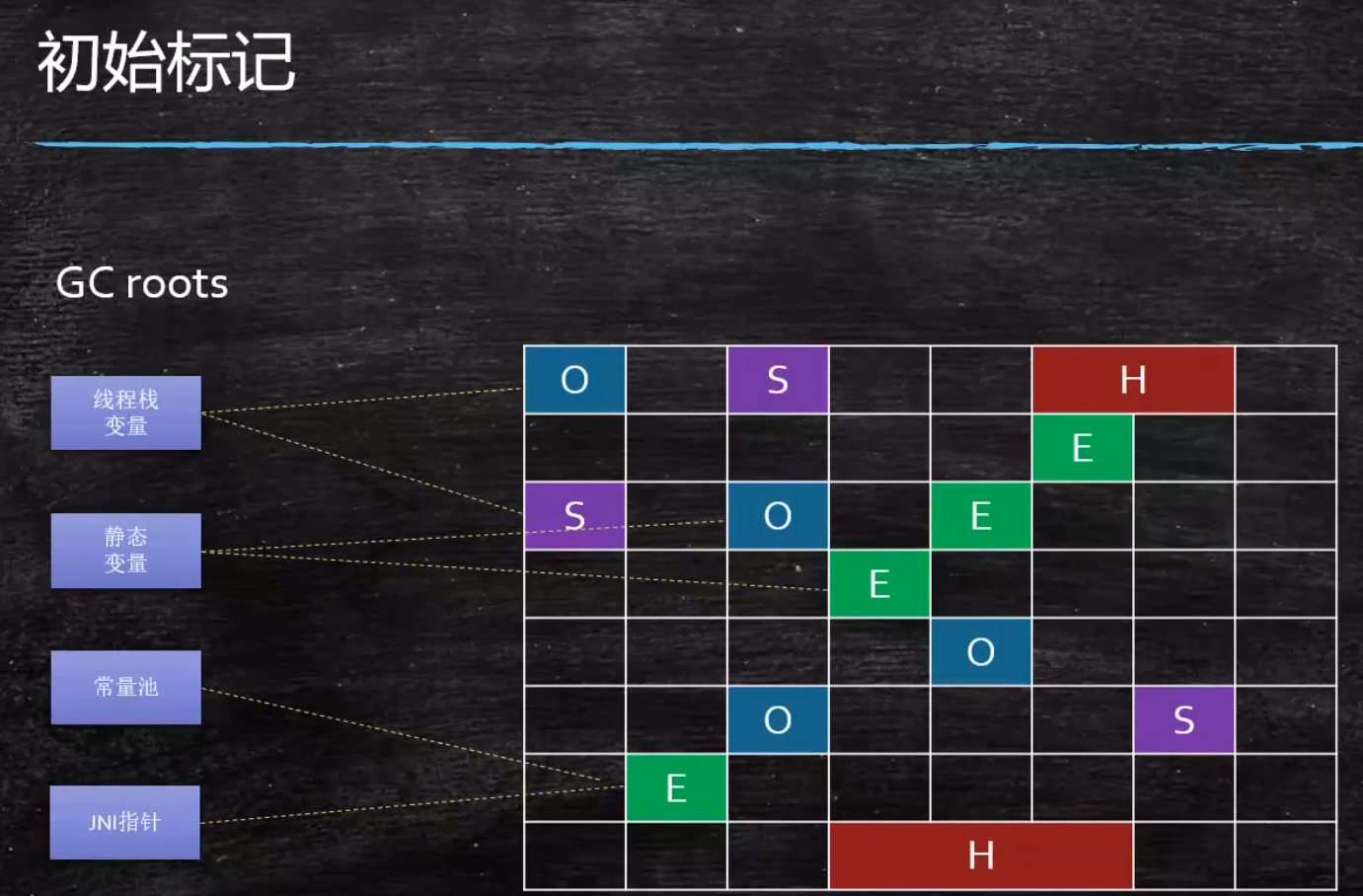
### ·JVM Instruction Set

### ·GC tuning

### MixedGC过程

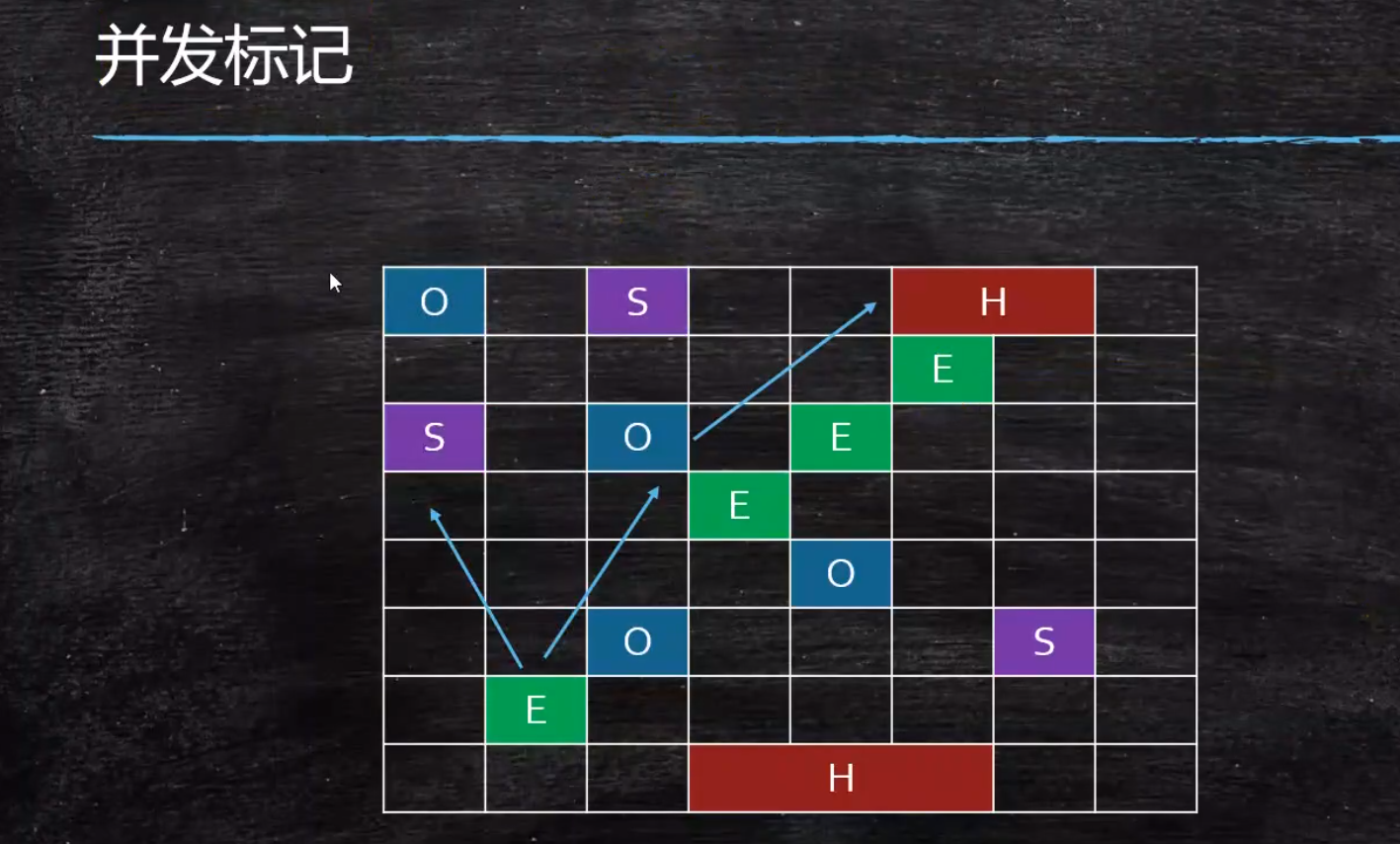
#### 初始标记 STW

把根对象直接引用的对象进行标记



#### 并发标记

监视应用是否会再引用标记对象



#### 最终标记STW

再标记一遍，具体怎么标记不知道

#### 并行回收 STW

节省空间，一小块range完全复制到另一块range，浪费比较少的内存

