方法区(永久代)

线程共享内存区域

存储已被虚拟机加载的类信息、常量、静态变量(static变量属于类)、即时编译器编译后的代码等数据

运行时常量池(Runtime Constant Pool)

方法区的一部分。来源于Class文件中的常量池(存放编译期生成的各种字面量和符号引用)

这部分内容将在类加载后进入方法区的运行时常量池中存放。

JDK1.7及以上，放在永久代的字符串常量池被移出方法区。

运行时常量池相对于Class文件常量池的另外一个重要特征是具备动态性，

字面量

近似常量的概念，指的是数据的值

(1)文本字符串：public String s = "abc"; 中的 "abc"

(2)用final修饰的成员变量，包括静态变量、实例变量和局部变量：public final static int f = 0x101中的0x101

符号引用

(1)类和接口的全限定名

(2)字段的名称和描述符

(3)方法中的名称和描述符

https://www.jianshu.com/p/55f65dac1b4b

直接内存

Java堆溢出

虚拟机栈和本地方法栈溢出

(1)StackOverflowError异常：如果线程请求的栈深度大于虚拟机所允许的最大深度(单线程：eg：方法帧的本地变量表过长(递归))

如果使用虚拟机默认参数，栈深度在大多数情况下可达到1000-2000个方法帧是没有问题的，对于正常的递归是够用的。

(2)OutOfMemoryError异常：如果虚拟机在扩展栈时无法申请到足够的内存空间（建立过多线程）

内存分配：

对于每个线程：堆容量(最大堆容量Xmx) + 方法区容量(MaxPermSize最大方法区容量) + 程序计数器(内存很小可以忽略) + 虚拟机栈 + 本地方法栈

解决:(1)减少线程数;(2)更换为64位虚拟机;(3)减少最大堆和减少栈容量

方法区和运行时常量池溢出

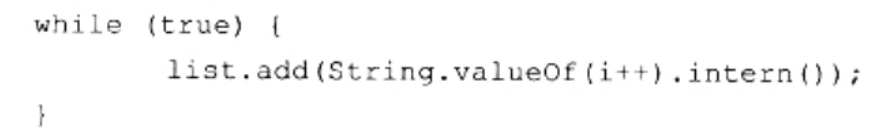
(1)JDK1.7之前: 运行时常量池是方法区的一部分，即“永久代”内

OutOfMemoryError: PermGen space

(2)JDK1.7及以上: 逐步去“永久代”，比如字符串常量池

String.intern()

作用: 如果字符串常量池中已经包含一个字符串内容相同的String对象，则返回池中该String对象；否则将此String对象复制后，添加到常量池中，并且返回池中该新对象的引用。（JDK1.6）



执行流程？？？

Class文件：类版本、字段、方法、接口、常量池