深入理解Java虚拟机笔记

# 自动内存管理

## Java内存区域与内存溢出异常

### 运行时数据区域

#### 程序计数器

概述：程序计数器是一块较小的内存空间，它可以看作是当前线程所执行的字节码的行号指示器,字节码解释器工作时就是通过改变这个计数器的值来选取下一条需要执行的字节码指令.是每个线程独有的,因为cpu时间片切换线程时,要知道这个线程的下一条指令是什么.不存在OutOfMemoryError情况的区域。

#### Java虚拟机栈

##### 概念

线程私有的,生命周期与线程相同,每个方法被执行的时候，Java虚拟机都会同步创建一个

栈帧用于存储局部变量表、操作数栈、动态连接、方法出口等信息。每一个方法被调用直至

执行完毕的过程，就对应着一个栈帧在虚拟机栈中从入栈到出栈的过程。

##### 局部变量表

1. 定义为一个数字数组，主要用于存储方法参数和定义在方法体内的局部变量,这些数据类型包括各类基本数据类型、对象引用(reference),以及returnAddress类型。
2. 由于局部变量表是建立在线程的栈上，是线程的私有数据，因此不存在数据安全问题。
3. 局部变量表所需的容量大小是在编译期确定下来的。
4. 局部变量表中的变量只在当前函数调用中有效，当函数调用结束后，随着函数栈帧的销毁，局部变量表也会随之销毁。
5. 由于局部变量表在栈帧中，因此，如果函数的参数和局部变量较多，会使得局部变量表膨胀，从而每一次函数调用就会占用更多的栈空间，最终导致函数的嵌套调用次数减少。
6. 数据类型在局部变量表中的存储空间以局部变量槽（Slot）来表示,一般64位的类型占用两个槽.

##### 异常类型

1. StackOverflowError:如果线程请求的栈深度大于虚拟机所允许的深度,如:方法中入参太多,局部变量太多的情况.
2. OutOfMemoryError

①：如果Java虚拟机栈容量可以动态扩展，当栈扩展时无法申请到足够的内存会抛出OutOfMemoryError异常。

②：HotSpot虚拟机的栈容量是不可以动态扩展的,但若一开始申请内存就失败也会抛出oom异常.

#### 本地方法栈

##### 概念

与虚拟机栈执行java方法不同的是,本地方法栈是为虚拟机使用到的本地（Native）方法服务。

##### 异常类型

同虚拟机栈一样,本地方法栈也会在栈深度溢出或者栈扩展失败时分别抛出StackOverflowError和OutOfMemoryError异常。

#### 堆

##### 概念

1. Java堆是被所有线程共享的一块内存区域，在虚拟机启动时创建。此内存区域的唯一目的就是存放对象实例，Java世界里“几乎”所有的对象实例都在这里分配内存
2. Java堆是垃圾收集器管理的内存区域，因此一些资料中它也被称作“GC堆”
3. 根据分代回收的思想,又细分为很多区域（老年代、年轻代等）

##### 异常类型

如果在Java堆中没有内存完成实例分配，并且堆也无法再扩展时,会发生OutOfMemoryError异常。

##### 可扩展

1. Java堆既可以被实现成固定大小的，也可以是可扩展的，不过当前主流的Java虚拟机都是按照可扩展来实现的（通过参数-Xmx和-Xms设定）
2. -Xms为JVM启动时申请的初始Heap值，默认为操作系统物理内存的1/64但小于1G
3. -Xmx为JVM运行时可申请的最大Heap值，默认值为物理内存的1/4但小于1G

#### 方法区

##### 概念

1. 各个线程共享的内存区域,它用于存储已被虚拟机加载的类型信息、常量、静态变量、即时编译器编译后的代码缓存等数据
2. 这区域的内存回收目标主要是针对常量池的回收和对类型的卸载,由于这部分垃圾的判断条件苛刻,回收率比较高.

##### 永久代与元空间

1. JDK8前,很多人都更愿意把方法区称呼为“永久代”,因为当时的HotSpot虚拟机设计团队选择把收集器的分代设计扩展至方法区，或者说使用永久代来实现方法区而已，这样使得HotSpot的垃圾收集器能够像管理Java堆一样管理这部分内存.其他虚拟机不含这个概念.
2. 在JDK6的时候HotSpot开发团队就有放弃永久代，逐步改为采用本地内存来实现方法区的计

划了，到了JDK7的HotSpot，已经把原本放在永久代的字符串常量池、静态变量等移出

1. 到了JDK8，终于完全废弃了永久代的概念，改用在本地内存中实现的元空间来代替，把JDK 7中永久代还剩余的内容（主要是类型信息）全部移到元空间中

##### 运行时常量池

是方法区的一部分。Class文件中除了有类的版本、字段、方法、接口等描述信息外，还有一项信息是常量池表（ConstantPoolTable），用于存放编译期生成的各种字面量与符号引用，这部分内容将在类加载后存放到方法区的运行时常量池中。

##### 异常类型

如果方法区无法满足新的内存分配需求时，将抛出OutOfMemoryError异常

#### 直接内存

##### 概念

直接内存（DirectMemory）并不是虚拟机运行时数据区的一部分，也不是《Java虚拟机规范》中

定义的内存区域。但是这部分内存也被频繁地使用，而且也可能导致OutOfMemoryError异常出现

##### 异常类型

Java中某些方法直接调用Native函数库,直接分配堆外内存,当内存大于物理机内存限制时会报OOM