

1. JVM 运行时数据区域有哪些？分别是有什么作用？

程序计数器：程序计数器（Program Counter Register）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。因此，为了线程切换后能恢复到正确的执行位置，每条线程都需要有一个独立的程序计数器，各条线程之间计数器互不影响，独立存储，我们称这类内存区域为“线程私有”的内存。

Java 虚拟机栈：Java 虚拟机栈（Java Virtual Machine Stacks）也是线程私有的，它的生命周期与线程相同。虚拟机栈描述的是 Java 方法执行的内存模型：每个方法在执行的同时都会创建一个栈帧（Stack Frame）用于存储局部变量表、操作数栈、动态链接、方法出口等信息。每一个方法从调用直至执行完成的过程，就对应着一个栈帧在虚拟机栈中入栈到出栈的过程。

本地方法栈：本地方法栈（Native Method Stack）与虚拟机栈所发挥的作用是非常相似的，它们之间的区别不过是虚拟机栈为虚拟机执行 Java 方法（也就是字节码）服务，而本地方法栈则为虚拟机使用到的 Native 方法服务。

Java 堆：Java 堆（Java Heap）是 Java 虚拟机所管理的内存中最大的一块。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，几乎所有的对象实例都在这里分配内存。

方法区：方法区（Method Area）与 Java 堆一样，是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信息、常量、静态变量、即时编译器编译后的代码等数据。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，一般来说，这个区域的回收“成绩”比较难以令人满意，尤其是类型的卸载，条件相当苛刻，但是这部分区域的回收确实是必要的。

2. 什么样的代码会造成 StackOverflowError？

如果线程请求的栈深度大于虚拟机所允许的深度，会抛出 StackOverflowError 异常。

3. 什么是 TLAB?对应 java 中的哪个 API 类？

为解决内存分配的并发问题，有了本地线程分配缓冲（Thread Local Allocation Buffer, TLAB），哪个线程要分配内存，就在哪个线程的 TLAB 上分配，只有 TLAB 用完并分配新的 TLAB 时，才需要同步锁定。

虚拟机是否使用 TLAB，可以通过 -XX: +/-UseTLAB 参数来设定。

对应 Java 的 ThreadLocal 类。

4. 直接内存(Direct Memory)是在 java 堆中吗？直接内存的大小是由什么决定

的？如何申请直接内存？java 中什么 API 用到了直接内存？

直接内存（Direct Memory）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。

本机直接内存的分配不会受到 Java 堆大小的限制，但是，既然是内存，肯定还是会受到本机总内存（包括 RAM 以及 SWAP 区或者分页文件）大小以及处理器寻址空间的限制。

通过 -XX:MaxDirectMemorySize 申请直接内存大小。

从 JDK1.4 开始，加入 NIO (new Input/Output)，可以通过 Native 函数库直接分配堆外内存，

然后通过一个存储在 Java 堆里面的 `DirectByteBuffer` 对象作为这块内存的引用进行操作。

5. 说说一个对象的内存布局。

在 HotSpot 虚拟机中，对象在内存中存储的布局可以分为 3 块区域：对象头（Header）、实例数据（Instance Data）和对齐填充（Padding）。

HotSpot 虚拟机的对象头包括两部分信息，第一部分用于存储对象自身的运行时数据。对象头的另外一部分是类型指针，即对象指向它的类元数据的指针，虚拟机通过这个指针来确定这个对象是哪个类的实例。

接下来的实例数据部分是对象真正存储的有效信息，也是在程序代码中所定义的各种类型的字段内容。无论是从父类继承下来的，还是在子类中定义的，都需要记录起来。

第三部分对齐填充并不是必然存在的，也没有特别的含义，它仅仅起着占位符的作用。当对象实例数据部分没有对齐时，就需要通过对齐填充来补全。

6. 运行时常量池是什么时候生成的？如何在运行时往常量池插入一个常量值？

运行时常量池（Runtime Constant Pool）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（Constant Pool Table），用于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。

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，Java 语言并不要求常量一定只有编译期才能产生，也就是并非预置入 Class 文件中常量池的内容才能进入方法区运行时常量池，运行期间也可能将新的常量放入池中，这种特性被开发人员利用得比较多的便是 String 类的 `intern()` 方法。

7. 对象的创建

虚拟机遇到一条 `new` 指令时，首先将去检查这个指令的参数是否能在常量池中定位到一个类的符号引用，并且检查这个符号引用代表的类是否已被加载、解析和初始化过。如果没有，那必须先执行相应的类加载过程。在类加载检查通过后，接下来虚拟机将为新生对象分配内存。对象所需内存的大小在类加载完成后便可完全确定，为对象分配空间的任务等同于把一块确定大小的内存从 Java 堆中划分出来。