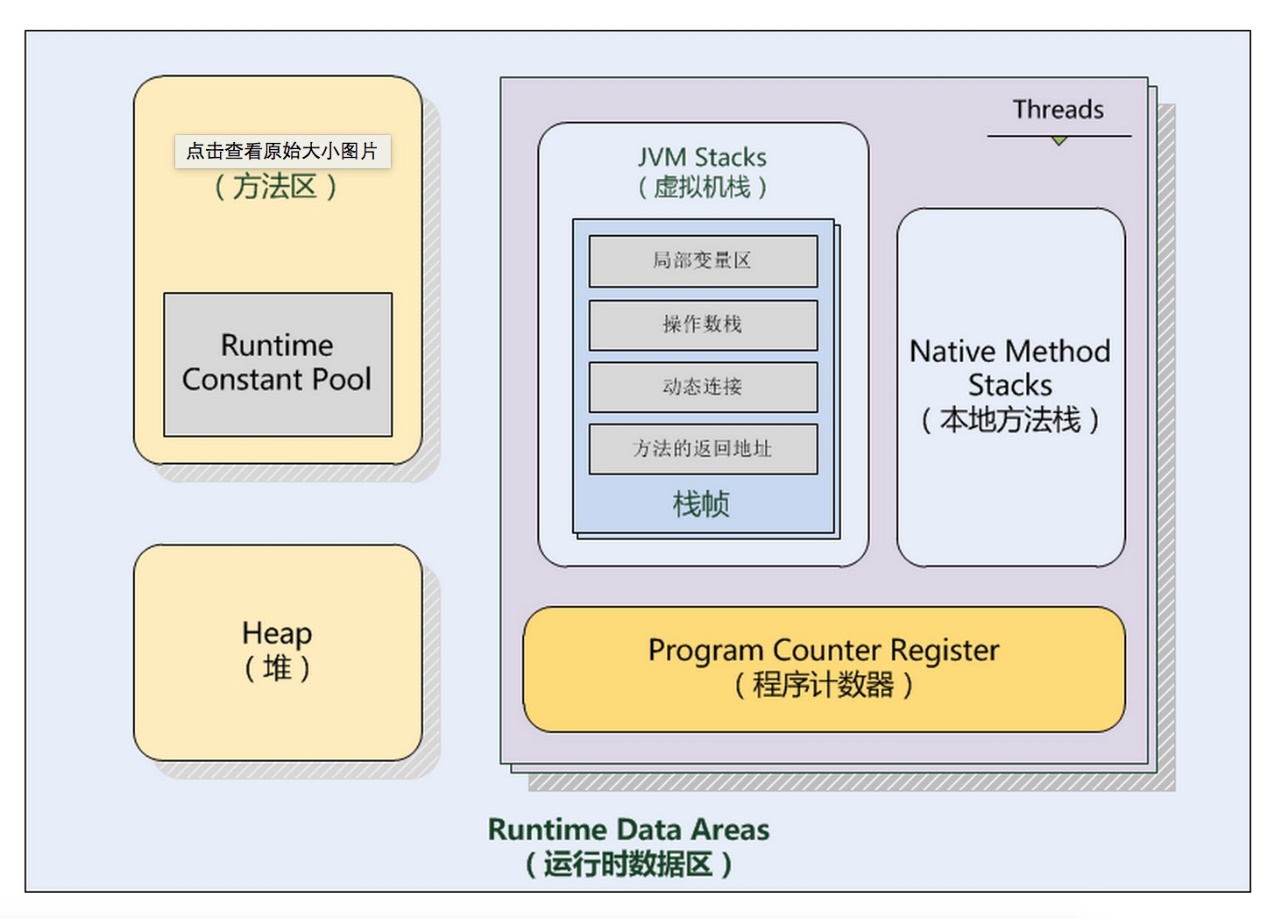
JVM运行时数据区域

**定义：**

JVM运行时数据区域（Run-Time Data Areas），JVM定义了在程序执行期间使用的各种运行时数据区域。其中一些数据区域是在java虚拟机启动时临时创建，在Java虚拟机退出时销毁。其他数据区域是单线程区域（ Per-thread data areas），单线程区域是随着线程创建、销毁时创建销毁的。



1. 程序计数器（The PC Register）

程序计数器是处理器中的寄存器，用于指示计算机在其程序序列中的位置。它包含正在执行指令的当前位置，当每个指令被获取，程序计数器的地址加一。在每个指令被获取之后，程序计数器指向顺序中的下一个指令。当计算机重启，程序计数器置0。Java虚拟机可以支持多线程一次运行，在任何时候，每个java虚拟机线程都在执行单个方法的代码，即该线程的当前方法。如果该方法不是本地方法，程序计数器就包含当前正在执行的java虚拟机指令的地址。如果当前线程正在执行的方法是本地的，程序计数器的值就是Undefined的。程序计数器的空间足够大，可以保存在特定平台的返回地址类型（returnAddress）或本地指针。

1. Java虚拟机栈

每个java虚拟机线程都是一个私有的java虚拟机栈，与线程同时创建，同时销毁。Java虚拟机栈类似于诸如传统C语言的堆栈，他保存了局部变量和部分结果，并在方法的调用和返回中起作用。除了压入和取出栈外，Java虚拟机栈不会被直接操作，栈的分配是由堆完成的。Java虚拟机栈的内存是不连续的。

在java虚拟机版本规范的第一版中，java虚拟机栈被称作java堆栈。此规范允许java虚拟机栈具有固定大小或根据计算的需要动态扩展和收缩。如果每个Java虚拟机堆栈都具有固定大小，则可以在创建时独立选择每个java虚拟机堆栈的大小。

Java虚拟机可以实现为程序员或用户提供对java虚拟机栈初始大小的控制，以及在动态扩展或收缩java虚拟机栈的情况下，控制最大值和最小值。

Java虚拟机栈有以下异常：

1. 如果在一个线程请求所需要的内存大于java虚拟机栈的最大值，则会抛出StackOverflowError。
2. 如果在动态扩展java虚拟机栈的过程中，可以使用的内存不足以支撑扩展，或者可使用的内存不足以为新的线程创建java虚拟机栈，则java虚拟机栈会抛出OutOfMemoryError。

**补充说明：**

以上来自JVM官方文档的翻译。与程序计数器一样，java虚拟机栈也是线程私有，是随线程创建销毁而创建销毁，java虚拟机栈描述的是java方法执行的内存模型。每个方法被执行时，都会创建一个栈帧，它是用于支撑java虚拟机对方法的调用和执行的数据结构。每个虚拟机栈的栈顶都存储了局部变量表、操作数栈、动态链接、方法返回地址等等信息，执行引擎只会对栈的栈顶进行操作。

3、堆（HEAP）

Java虚拟机具有在所有线程之间共享的区域，堆。堆为java虚拟机中所有的实例和数组分配内存。堆是在Java虚拟机启动的时候创建的。对象的堆存储由自动存储管理系统（垃圾回收器）回收，对象永远不会被显示释放。Java虚拟机假设没有特定类型的自动存储管理系统，可以根据实现者的系统要求选择存储管理技术。堆可以具有固定大小，也可以根据计算的需要进行扩展，并且如果不需要这么大的内存，可以进行收缩。堆得内存是不连续的。Java虚拟机可以为开发者或者用户提供初始大小的控制，以及可以动态扩展或者收缩堆，控制最大值和最小值（maximum and minimum heap size）。

常见的堆异常：

如果计算所需要的堆内存超过自动存储系统可用的堆内存，就会抛出OutOfMemoryError。

4、方法区（Method Area）

方法区是所有线程共享的内存区域，是在java虚拟机启动的时候创建的。方法区类似于传统语言的编译代码的存储区域，或类似于操作系统进程中的”文本段（text）”，它存储了每个类的结构，例如运行时常量池，字段和方法数据，以及方法和构造函数的代码，包括类和实例初始化中使用的特殊方法。

虽然方法区在逻辑上是属于堆的部分，但是垃圾收集器不会对它进行回收。本规范未规定方法区的位置，或用于管理管理编译代码的策略。方法区可以是固定大小的，也可以根据计算进行扩展，并且在不需要这么大的时候，进行收缩该区域内存。方法区的内存同样是非连续的。

常见的方法区异常：

如果计算所需要的方法区内存超过方法区可使用的内存，就会抛出OutOfMemoryError。

5、运行时常量池（Run-Time Constant Pool）

运行时常量池是类文件中constant\_pool表中类和运行时接口的表现形式，它包含几种常量，从编译时已知的数字到必须在运行时解析的方法和引用。运行时常量池提供类似于传统编程语言的符号表的功能，而且它比传统的符号表存储的数据类型更丰富。每个运行时常量池都是从java虚拟机的方法区分配出来的。当Java虚拟机创建类和接口时，将构造类和接口的运行时常量池。

常见的运行时常量池异常：

创建或接口时，如果常量池的构造所需要的内存大于Java虚拟机方法区可使用的内存，就会抛出OutOfMemoryError。

6、本地方法栈（Native Method Stacks）

Java虚拟机的实现可使用传统的堆栈，俗称为“C堆栈”，用支持本地方法（用Java编程语言以外语言写的方法）。一个不支持本地方法执行的JVM是不需要这个区域的。如果支持，则通常在创建线程时为每个线程分配本地方法栈。

此规范允许本机方法堆栈具有固定大小或根据计算的需要动态扩展和收缩。 如果本机方法堆栈具有固定大小，则可以在创建该堆栈时独立地选择每个本机方法堆栈的大小。

常见的关于本地方法栈的异常：

1. 如果线程中的计算需要的堆栈内存比本地方法栈的内存更大是，则Java虚拟机会抛出StackOverflowError。
2. 如果可以动态扩展本地方法堆栈并尝试本机方法堆栈扩展但可用内存不足，或者如果可用内存不足以为新线程创建初始本机方法堆栈，则Java虚拟机会抛出OutOfMemoryError。