# JVM的实现原理

## 1 类装载器（ClassLoader）

主要负责加载class文件，是否能执行主要取决于execution engine它是负责执行被加载类中包含的指令。有两种类加载器分别为启动类加载器和用户自定义类加载器，然而启动类加载器是JVM实现的一部分，用户自定义类加载器是Java程序一部分。

## 2，本地方法栈（native method stack）

主要作用是登记native方法，然后在execution engine执行的时候加载本地方法库。

## 3，栈有时我们又叫栈内存，

负责Java程序的运行，它是在线程创建时创建的，所以生命周期也是和线程生命周期一致，同时消亡，线程结束了栈也就释放，特别提醒的是栈不存在垃圾回收的问题，因为线程结束栈就是释放了。平时我们写的类变量、引用类型变量、实例方法等等都是在函数的栈内存分配好。

4，程序计数器，

是指方法区中的方法字节码由引擎读取下一条指令，它是一个非常小的内存空间。为什么有这种东西呢，大家都知道每个线程都是有一个程序计数器的，是线程私有的，相当一个指针。

## 5，方法区

它是指线程共享的，谁都可以共享使用，我们通常用来保存装载类的元结构信息。

## 6，堆（heap）

它是Java虚拟机用来存储对象实例的，比我们在开发过程使用的new对象，只要通过new创建的对象的内存的对象都在堆分配，注意一点的是堆中的对象内存需要等待垃圾器（GC）进行回收，也是Java虚拟机共享区。

7，本地接口（native interface）

作用是融合不同的编程语言为Java所用，注意底层是C、C++写的，学习JVM时了解C语言一些更好，最起码能看懂，这个方法的行为就是native method stack中登记native方法，然后在execution engine执行时加载native libraries的。

## GC垃圾回收

GC首先要知道GC的作用域是在方法区（method area）和堆（heap）的GC垃圾回收它是Java核心技术，它是怎么实现的呢，主要通过确定对象引用来判断是否收集该对象，我们平时常用的方法引用计数和对象引用便利两种方法的，GC收集的算法后续更新。