JVM

JVM是运行在操作系统之上的，与硬件没有直接的交互。

JVM体系结构：

1. Class files
2. 类装载器
3. 运行时数据区
4. 执行引擎
5. 本地方法接口

**类加载器**

类装载器：负责加载class文件。

第一类是虚拟机自带的类加载器，有三种：

1：启动类加载器（Bootstrap）C++

2：扩展类加载器（Extension）Java

3：应用程序类加载器（App）Java ，也叫系统类加载器，加载当前应用的classpath的所有类

第二类是用户自定义加载器，Java.lang.ClassLoader的子类，用户可以定制类的加载方式

**类加载器的双亲委派机制**

**沙箱机制（防止恶意代码对java的破坏）**

比如自己写了一个java.lang.String

主要是为了保证java的沙箱机制

双亲委派:某个特定的类加载器在接到加载类请求的时候，首先将加载任务委托给父类加载器，依次递归，如果父类加载器可以完成加载任务，就返回成功，只有父类加载器无法完成加载任务的时候，才自己去加载。

**执行引擎**

负责解释命令，提交操作系统执行。

**Native Interface本地方法接口**

Java语言本身不能对操作系统底层进行访问和操作，但是可以通过JNI接口调用其他语言来实现对操作系统的访问。本地接口的作用是融合不同编程语言为Java所用，它的初衷是融合C/C++程序，Java在内存中专门开辟了一块区域处理标记为Native的代码，它的具体做法是Native Method Stack中等级Native方法，在Execution Engine执行的时候加载Native libraries（JNI）

目前该方法使用的越来越少，除非是与硬件有关的应用，比如通过Java程序驱动打印机或者Java系统管理生产设备，在企业级应用中已经比较少见。业务现在的异构领域间通信技术很发达，比如可以使用Socker通信，也可以使用WebService等。

**Native Method Stack 本地方法栈**

他的具体做法是在Native Method Stack 中等级Native方法，在Execution Engine执行的时候加载本地方法库。

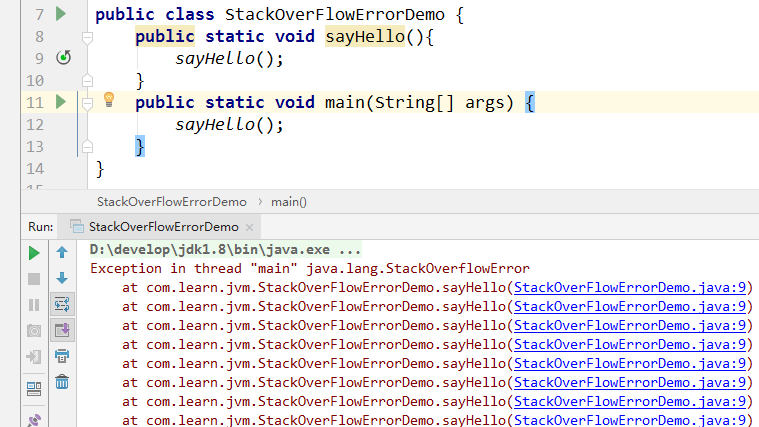
**PC寄存器**

每一个线程都有一个程序计数器，是线程私有的，就是一个指针，指向方法区中的方法字节码（用来存储指向下一条指令代码），由执行引擎读取下一条指令，是一个非常小的内存空间，几乎可以忽略不计。

**栈**

也叫栈内存，主管Java程序的运行，在创建线程的时候创建，它的生命周期是跟随着线程的生命周期，线程结束栈内存也就释放，对于栈来说不存在垃圾回收的问题，只要线程一结束，该栈就Over，生命周期和线程一致，是线程私有的。基本类型中的变量、实例方法、引用类型变量都是在函数的栈内存中分配。

**StackOverFlowError**



**方法区**

1、方法区是线程共享的，通常用来保存装载的类的元结构信息

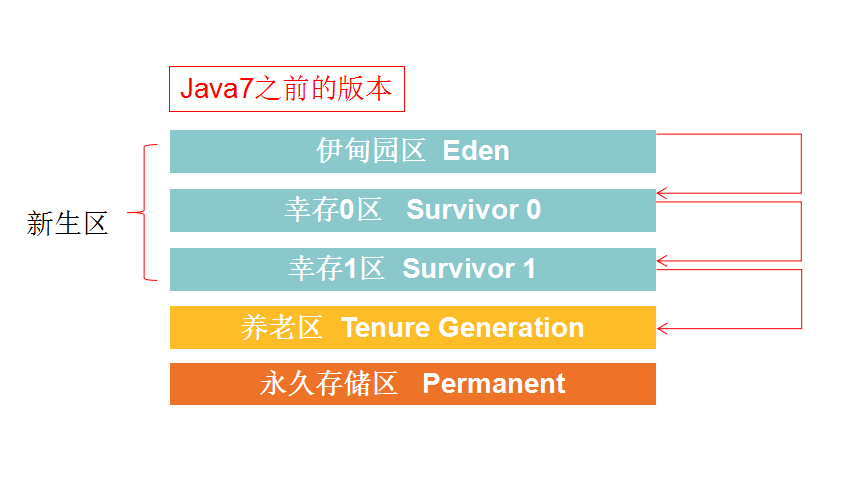
比如：运行时常量池+静态变量+常量+字段+方法字节码+在类/实例/接口 初始化用到的特殊方法。

1. 通常和永久区管理在一起（Java7之前），但是具体的跟JVM的实现和版本有关。

**Heap堆**

一个JVM实例只存在一个堆内存，堆内存的大小是可以调节的。类加载器读取了类文件之后，需要把类、方法、常变量放到堆内存中，保存所有类型的真实信息，以方便执行器执行。

堆内存在逻辑上分为三种：新生区，养老区，永久区



**新生区**

新生区是类的诞生、成长、消亡的区域，一个类在这里产生，应用，最后被垃圾回收器收集，结束生命。新生区又分为两部分： 伊甸区（Eden space）和幸存者区（Survivor pace） ，所有的类都是在伊甸区被new出来的。幸存区有两个： 0区（Survivor 0 space）和1区（Survivor 1 space）。当伊甸园的空间用完时，程序又需要创建对象，JVM的垃圾回收器将对伊甸园区进行垃圾回收(Minor GC)，将伊甸园区中的不再被其他对象所引用的对象进行销毁。然后将伊甸园中的剩余对象移动到幸存0区.若幸存0区也满了，再对该区进行垃圾回收，然后移动到1区。那如果1区也满了呢？再移动到养老区。若养老区也满了，那么这个时候将产生MajorGC（FullGC），进行养老区的内存清理。若养老区执行了Full GC之后发现依然无法进行对象的保存，就会产生OOM异常“OutOfMemoryError”。

如果出现java.lang.OutOfMemoryError: Java heap space异常，说明Java虚拟机的堆内存不够。原因有：

（1）Java虚拟机的堆内存设置不够，可以通过参数-Xms、-Xmx来调整。

（2）代码中创建了大量大对象，并且长时间不能被垃圾收集器收集（存在被引用）

