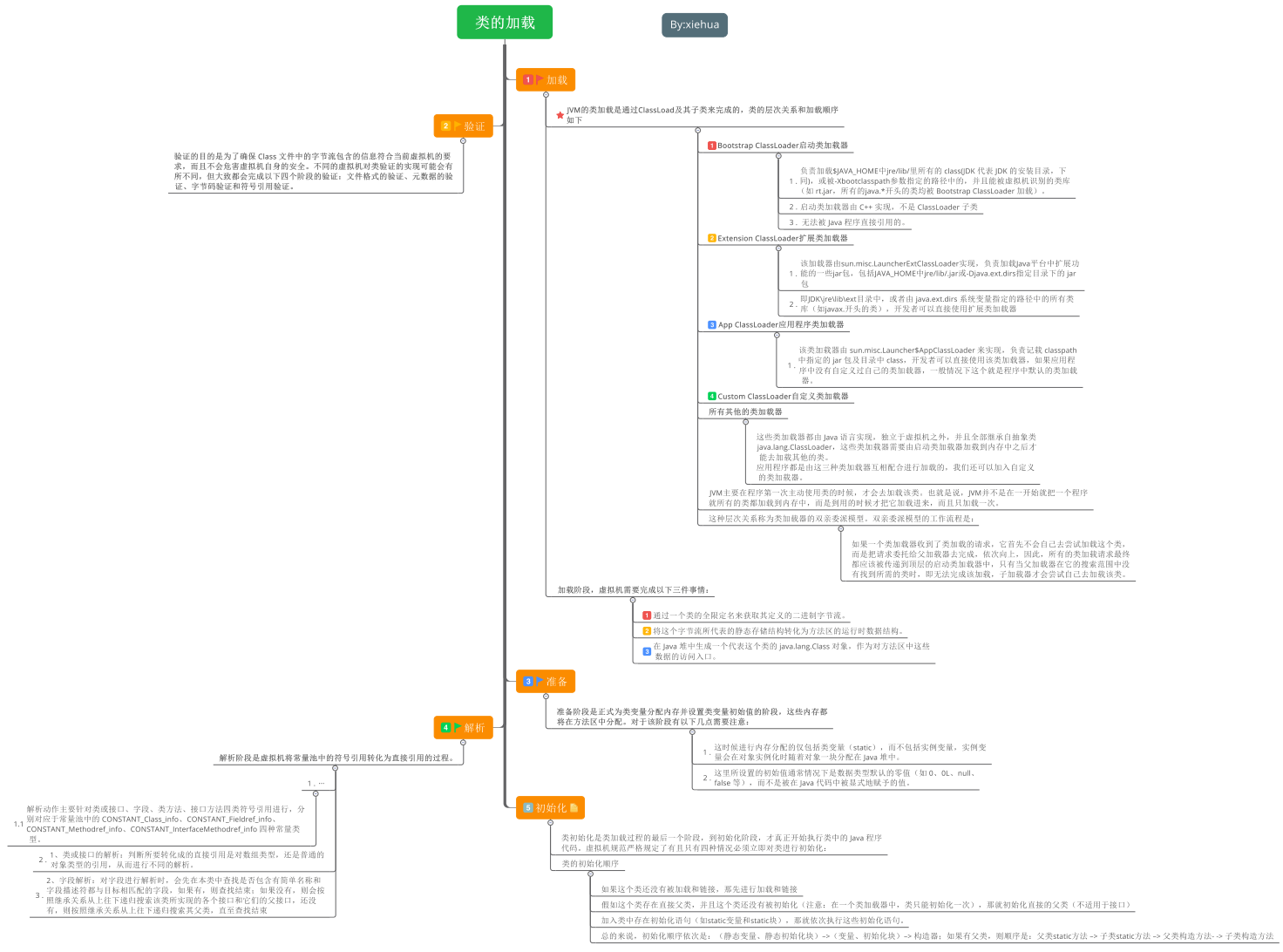
类的加载
By:xiehua
1 加载
JVM的类加载是通过ClassLoad及其子类来完成的，类的层次关系和加载顺序如下
1 Bootstrap ClassLoader启动类加载器
1、同），或被-Xbootclasspath参数指定的路径中的，并且能被虚拟机识别的类库（如 rt.jar，所有的java.*开头的类均被 Bootstrap ClassLoader 加载）。
2、启动类加载器由 C++ 实现，不是 ClassLoader 子类
3、无法被 Java 程序直接引用的。
2 Extension ClassLoader扩展类加载器
1、能的一些jar包，包括JAVA_HOME中jre/lib/.jar或-Djava.ext.dirs指定目录下的 jar 包
2、即JDK\jre\lib\ext目录中，或者由 java.ext.dirs 系统变量指定的路径中的所有类库（如javax.开头的类），开发者可以直接使用扩展类加载器
3 App ClassLoader应用程序类加载器
1、该类加载器由 sun.misc.Launcher$AppClassLoader 来实现，负责记载 classpath 中指定的 jar 包及目录中 class，开发者可以直接使用该类加载器，如果应用程序中没有自定义过自己的类加载器，一般情况下这个就是程序中默认的类加载器。
4 Custom ClassLoader自定义类加载器
所有其他的类加载器
这些类加载器都由 Java 语言实现，独立于虚拟机之外，并且全部继承自抽象类 java.lang.ClassLoader，这些类加载器需要由启动类加载器加载到内存中之后才能去加载其他的类。
应用程序都是由这三种类加载器互相配合进行加载的，我们还可以加入自定义的类加载器。
JVM主要在程序第一次主动使用类的时候，才会去加载该类。也就是说，JVM并不是在一开始就把一个程序就所有的类都加载到内存中，而是到用的时候才把它加载进来，而且只加载一次。
这种层次关系称为类加载器的双亲委派模型。双亲委派模型的工作流程是：
如果一个类加载器收到了类加载的请求，它首先不会自己去尝试加载这个类，而是把请求委托给父加载器去完成，依次向上，因此，所有的类加载请求最终都应该被传递到顶层的启动类加载器中，只有当父加载器在它的搜索范围中没有找到所需的类时，即无法完成该加载，子加载器才会尝试自己去加载该类。
2 验证
验证的目的是为了确保 Class 文件中的字节流包含的信息符合当前虚拟机的要求，而且不会危害虚拟机自身的安全。不同的虚拟机对类验证的实现可能会有所不同，但大致都会完成以下四个阶段的验证：文件格式的验证、元数据的验证、字节码验证和符号引用验证。
3 准备
加载阶段，虚拟机需要完成以下三件事情：
1 通过一个类的全限定名来获取其定义的二进制字节流。
2 将这个字节流所代表的静态存储结构转化为方法区的运行时数据结构。
3 在 Java 堆中生成一个代表这个类的 java.lang.Class 对象，作为对方法区中这些数据的访问入口。
准备阶段是正式为类变量分配内存并设置类变量初始值的阶段，这些内存都将在方法区中分配。对于该阶段有以下几点需要注意：
1、这时候进行内存分配的仅包括类变量（static），而不包括实例变量，实例变量会在对象实例化时随着对象一块分配在 Java 堆中。
2、这里所设置的初始值通常情况下是数据类型默认的零值（如 0、0L、null、false 等），而不是被在 Java 代码中被显式地赋予的值。
4 解析
解析阶段是虚拟机将常量池中的符号引用转化为直接引用的过程。
1、...
1.1 解析动作主要针对类或接口、字段、类方法、接口方法四类符号引用进行，分别对应于常量池中的 CONSTANT_Class_info、CONSTANT_Fieldref_info、CONSTANT_Methodref_info、CONSTANT_InterfaceMethodref_info 四种常量类型。
2、1、类或接口的解析：判断所要转化成的直接引用是对数组类型，还是普通的对象类型的引用，从而进行不同的解析。
3、2、字段解析：对字段进行解析时，会先在本类中查找是否包含有简单名称和字段描述符都与目标相匹配的字段，如果有，则查找结束；如果没有，则会按照继承关系从上往下递归搜索该类所实现的各个接口和它们的父接口，还没有，则按照继承关系从上往下递归搜索其父类，直至查找结束
5 初始化
类初始化是类加载过程的最后一个阶段，到初始化阶段，才真正开始执行类中的 Java 程序代码，虚拟机规范严格规定了有且只有四种情况必须立即对类进行初始化：
类的初始化顺序
如果这个类还没有被加载和链接，那先进行加载和链接
假如这个类存在直接父类，并且这个类还没有被初始化（注意：在一个类加载器中，类只能初始化一次），那就初始化直接的父类（不适用于接口）
加入类中存在初始化语句（如static变量和static块），那就依次执行这些初始化语句。
总的来说，初始化顺序依次是：（静态变量、静态初始化块）->（变量、初始化块）-> 构造器；如果有父类，则顺序是：父类static方法 -> 子类static方法 -> 父类构造方法-> 子类构造方法