编写的java文件最终被编译为class文件，在运行期间将class文件加载到方法区，对文件流进行校验解析初始化，最终成为可以使用的java对象。Class文件加载到内存到从内存卸载经历了这些阶段：加载、连接、初始化、使用、卸载。其中连接有分为验证、准备、解析三个阶段，如下图。



我们重点研究下加载、连接、初始化这三个过程中发生了什么。

一、使用类加载器加载class文件

先让我们自定义一个类加载器，并初始化这个class对象，来初步体验下类加载到使用的过程，首先实现一个简单的class类

public class MgDemoSample {  
  
 public void startSample()  
 {  
 System.*out*.println("MgDemoSample");  
 }  
}

使用javac MgDemoSample.java 编译出 class文件，将class文件放到E:/cltmp/目录下，然后创建一个URLClassLoader加载MgDemoSample，并实例化对象，通过反射执行startSample这个函数

URL url = new URL("file:/E:/cltmp/");  
URLClassLoader loader = new URLClassLoader(new URL[]{url});  
Class cl = loader.loadClass("MgDemoSample");  
*log*.info("....{}",cl.getName());  
Object instance = cl.newInstance();  
Method mainMethod = cl.getMethod("startSample");

注意：如果报错Can't initialize javac processor due to (most likely) a class loader problem: java.lang.NoClassDefFoundError: com/sun/tools/javac/processing/JavacProcessingEnvironment是SDK中缺少tools.jar，添加一下可以了

接下来一起研究下，加载、连接、初始化，这三个过程。

二、加载

在加载过程中主要做了什么呢

1、获取class的二进制字节流

2、将字节流存储到方法区，并转换为静态存储结构

3、生成jvm可用的class对象

注意：class的连接并不是在加载结束后才会开始，这两个阶段是并行的，例如需要验证阶段对文件格式验证通过以后，字节流才能存储到方法区中。

类加载器只需要获取到class的二进制字节流就可以进行类的加载，二进制字节流可以有一下来源：

1、从本地的class文件中读取，这是最常见的方式

2、从jar包中获取

3、程序中生成字节流加载到类加载器中，如动态代理

4、通过网络获取，如远程调试

可以通过url或者直接加载二进制流

二、连接

1、验证，主要是验证字节流是否符合class文件结构，以及保证不会危害虚拟机安全。验证阶段是类加载过程中很重要的阶段，但不是必须的，如果能保证字节流的决定安全可以通过jvm参数设置关闭验证流程，从而节省加载时间。

验证过程分为以下四个阶段：文件格式验证、元数据验证、字节码验证、符号引用验证

1）文件格式验证，验证字节流是否符合class文件结构，以及保证对jvm可用。如：验证是否以魔数0xCAFEBABE开头，版本号是否符合虚拟机型号，常量池中常量是否有不支持的常量类型等等。

2）元数据验证，对字节码描述的信息进行进行语义验证，保证符合java语言规范要求。这个过程主要针对的是数据类型。如：验证是否有父类，父类是否允许被继承（final修饰），是否实现了需要实现的所有方法（抽象）等等。

3）字节码验证，通过数据流和控制流分析，确定程序的语义是合法的。针对的是方法体进行验证，这是四种验证中最复杂的部分，保证方法体的内容不会危害虚拟机。如：类型转换的有效，跳转指令不会跳出方法体以外等。

4）符号引用，符号引用的验证一般和解析过程一起，即发生在符号引用转变为直接引用的过程中。是对类自身以外的信息进行匹配性验证。如：通过引用类的类名是否能找到对应类。引用的类、字段、方法是否能被当前类访问（public修饰）

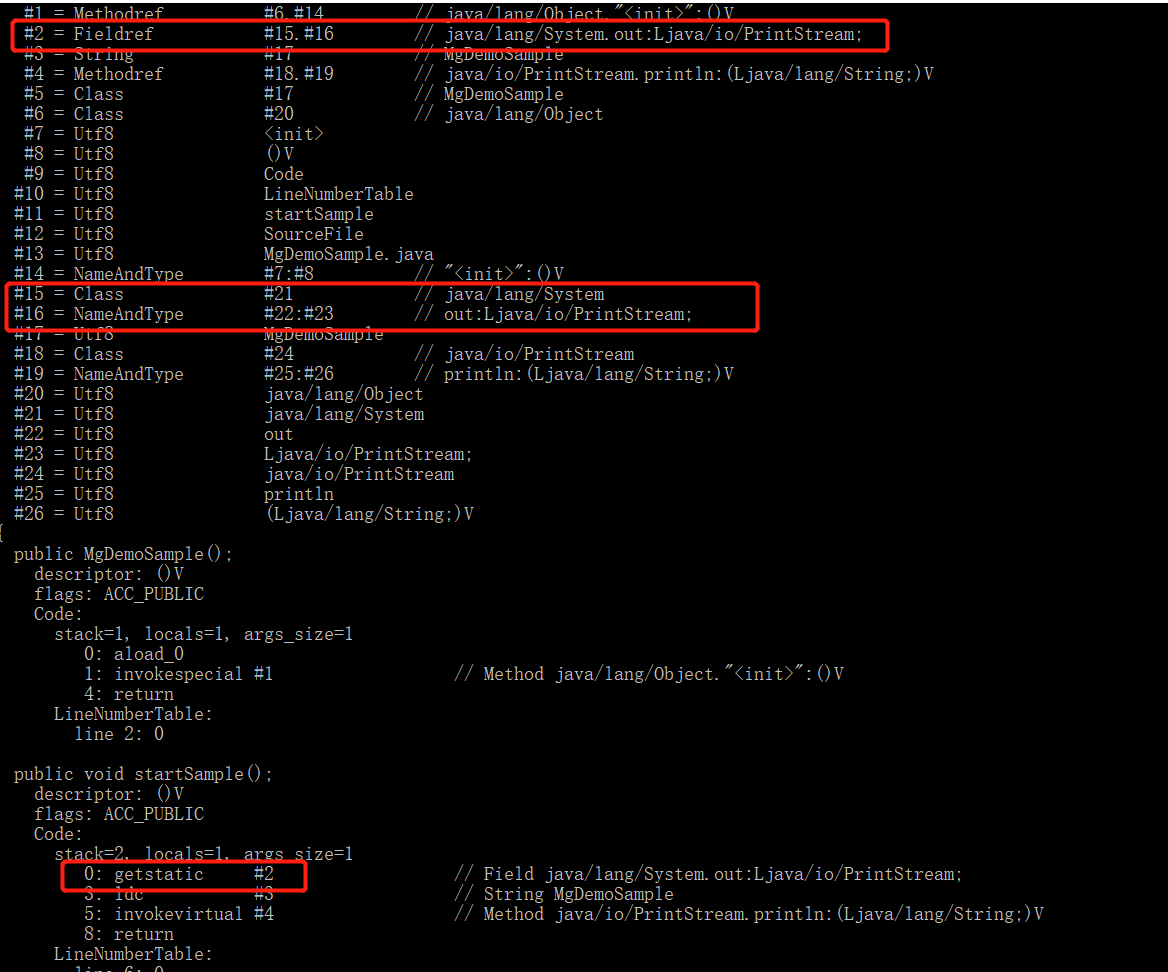
2、准备，这个阶段正式为类变量（static修饰的静态变量）分配内存并设置变量初始值。这里只针对类的静态变量，最终类变量存储来方法区。类变量设置的初始值只是变量的默认值。例如 public static int no =10；在这个阶段no被设置为int的默认值0，而不是10，设置为10是在初始化阶段。

注意：整形（int、short byte、long）默认值是0，泛型（double、float）是0.0，boolean默认是false,char默认是\u0000,引用类型默认是null

3、解析，是符号引用转换为直接引用的过程。

首先说一下什么是符号引用。

符号引用，即一组符号用来表示引用的目标。很抽象对吧，首先看下一组符号，也就说可以不是一个符号，可以是多个符号。我们javap一下MgDemoSample这个class文件得到下图结果



图中常量表#2，是由#15和#16组合出来的，所以要找到System.out的out流的符号引用是#2#15#16这一组符号引用描述的。class文件接口中的多个表描述了引用的指向。

直接引用这个都知道就是直接指向目标的引用，这样就明白了解析过程做了什么了。

解析过程针对类、接口、字段、方法、方法类型、方法句柄、调用点7中符号引用进行。

四、初始化

这是类加载的最后一步，这个阶段才会真正执行java中的程序代码，根据程序指定的目标去初始化类变量和其他资源。即，在准备阶段被初始化为0的no，在这个阶段会被赋值为10。

这个阶段执行<clinit>()这个类构造器方法。这个方法是是编译器收集所有类变量赋值操作和静态代码块中的语句合并生成的，收集顺序是源文件中出现顺序。

这样类的加载过程：加载、连接、初始化，就全部结束了，我们后边会一起研究下类加载器和双亲委派模型等机制，来进一步了解class在jvm的使用情况。