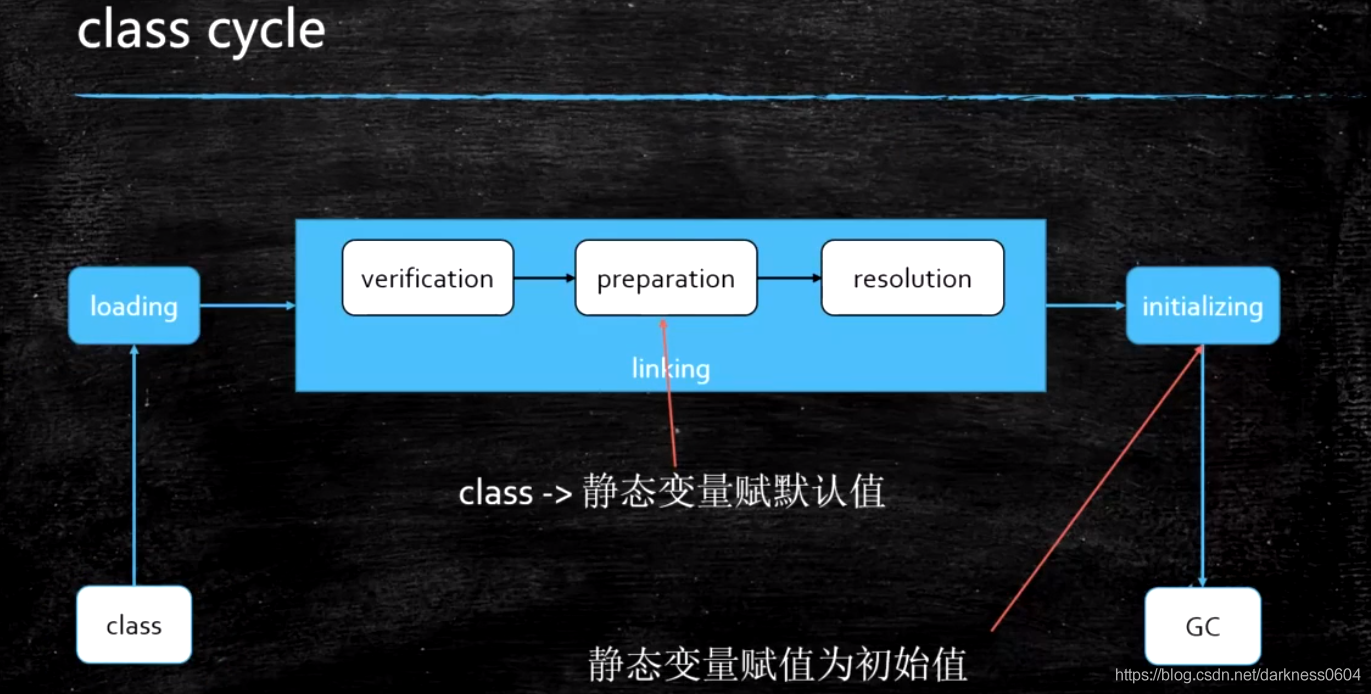
JVM第四天-Class三大声明周期之Linking和Initializing



一般来说，class的加载过程比较繁琐，因此我们之前比较大篇幅的进行了描述，但相对而言，class之后的2个步骤Linking和 Initializing就相对简单了一些，这次我们就来把剩下的两个讲完。

**Linking**

把class链接的过程，它又分为三小步：

**1、Verification**

验证文件是否符合JVM规定（例如是否包含cafe babe之类的规范内容）

**2、Preparation**

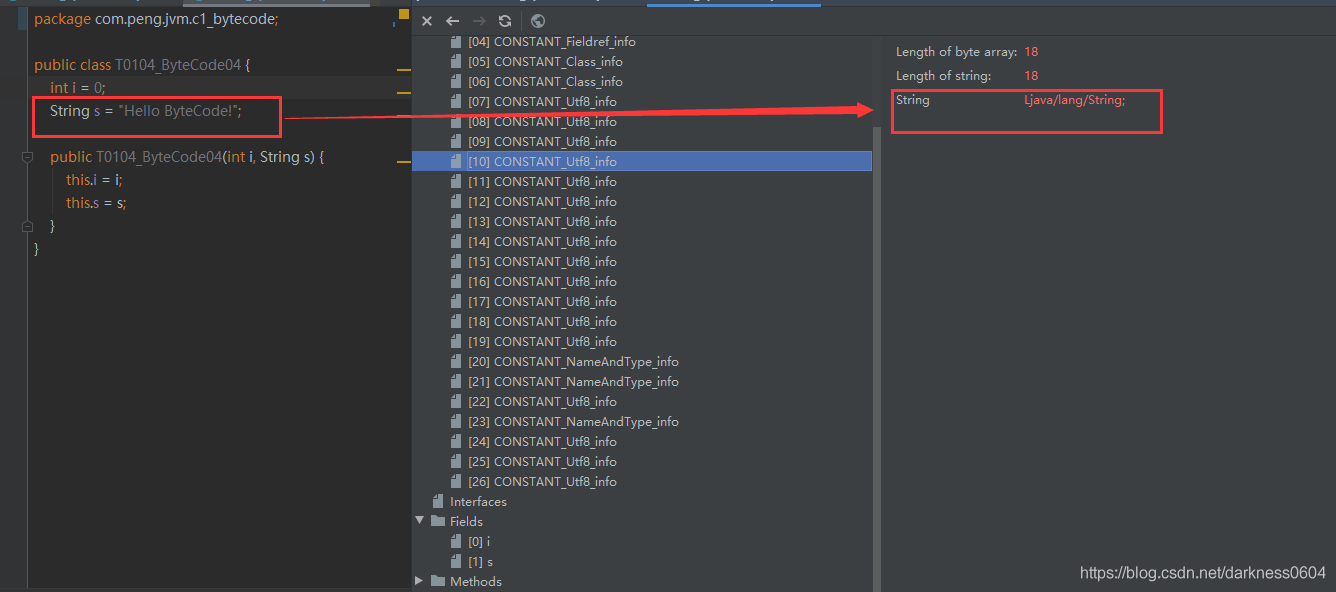
静态成员变量赋默认值

**3、Resolution**

将类、方法、属性等符号引用解析为直接引用。

常量池中的各种符号引用解析为指针、偏移量等内存地址的直接引用。

还记得我们在class文件格式分析的时候，看到的各种属性和常量值里各种的引用指向吗



可以认为，这个阶段，就是把这种字面量的值转换成内存里的实际地址。

在ClassLoader类中的loadClass方法中我们之前应该看到过一个方法：

|  |
| --- |
| /\*\*  \* Links the specified class. This (misleadingly named) method may be  \* used by a class loader to link a class. If the class <tt>c</tt> has  \* already been linked, then this method simply returns. Otherwise, the  \* class is linked as described in the "Execution" chapter of  \* <cite>The Java™ Language Specification</cite>.  \*  \* @param c  \* The class to link  \*  \* @throws NullPointerException  \* If <tt>c</tt> is <tt>null</tt>.  \*  \* @see #defineClass(String, byte[], int, int)  \*/  protected final void resolveClass(Class<?> c) {  resolveClass0(c);  } |

这个方法实际内部就完成了这个工作。

**Initializing**

调用类初始化代码 ，给静态成员变量赋初始值

**通过面试题理解静态变量赋值过程**

以上我们就把class的三大步骤讲述完了，接下来我们结合这个过程，来分析一道面试题。

|  |
| --- |
| package com.peng.jvm.c2\_classloader;  public class T001\_ClassLoadingProcedure {  public static void main(String[] args) {  System.out.println(T.count); //这个count是几？  }  }  class T {  public static T t = new T();  public static int count = 2;  private T() {  count ++;  }  } |

来看一下上面的代码，分析一下最后打印的count应该是多少呢？

结果是2，和你想的一样吗？ 我们来分析下为什么：

T.count这种调用静态属性在JVM规范中是一种必须要初始化类的行为，因此会触发T的初始化过程，而我们现在终于知道一个Class要经过三大步才算整体完成装载。

第一步，把T这个Class通过Classloader装载到了内存中。

第二步，Linking环节，校验class文件的格式，然后**对静态变量从上到下的赋予默认值（引用类型是null，基本类型是**0），最后完成符号引用的替换。于是这一步执行之后，T里的成员值应该是这样的：

|  |
| --- |
| public static T t = null;  public static int count = 0; |

第三步，进入Initializing环节，开始对静态变量从上到下的赋予初始值，调用静态方法。这里执行之后，T里的成员值是这样的：

|  |
| --- |
| public static T t = new T(); |

这里会需要调用到T的构造方法中，然后把count++，然后count一开始是0，此时就变成1了：

|  |
| --- |
| public static int count = 1; |

接下来，对count赋予初始值：

|  |
| --- |
| public static int count = 2; |

所以count最终的值是2。

那这道题换一下呢，把count和t的顺序变一下，这样的话呢：

|  |
| --- |
| package com.peng.jvm.c2\_classloader;  public class T001\_ClassLoadingProcedure {  public static void main(String[] args) {  System.out.println(T.count); //这个count是几？  }  }  class T {  public static int count = 2;  public static T t = new T();    private T() {  count ++;  }  } |

结果是几？ 是3

我们再走一遍流程：

第一步装载class不说了。

第二步，从上到下赋予静态变量默认值，变成：

|  |
| --- |
| public static int count = 0;  public static T t = null; |

第三步，从上到下赋予静态变量的初始值，变成：

|  |
| --- |
| public static int count = 2; |

|  |
| --- |
| public static T t = new T(); |

这里会去调用T的构造方法，count++，count变成3.

因此最后count的值是3.

**对象变量的赋值**

上面面试题剖析了静态变量的赋值过程，分为三部：load - 默认值 - 初始值

而其实对象变量的赋值，也不是一步完成的，也是分为了2步：

1、当一个对象通过new，首先需要向内存申请空间，申请完内存后会首先把对象变量都赋予默认值

2、接下来再给对象里的属性赋予初始值。