

# Java面试中的问题

[TOC]

## 常量池问题

常量池在Java中分两种：

第一种：静态常量池，class文件内的常量池，是字节码的一部分，用于保存编译时确定的数据。

第二种：运行时常量池（类似于缓存）

注释：

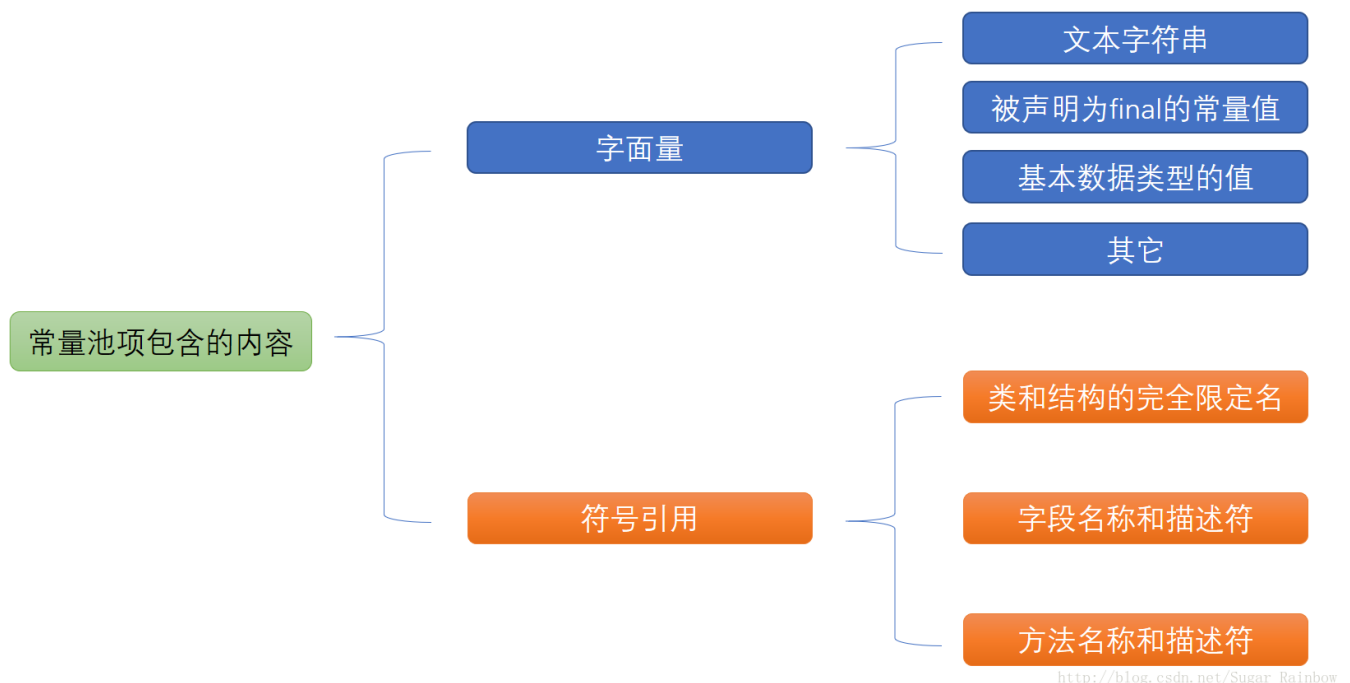
如果需要查看字节码可以通过jclasslib bytecode viewer插件。

### 静态常量池

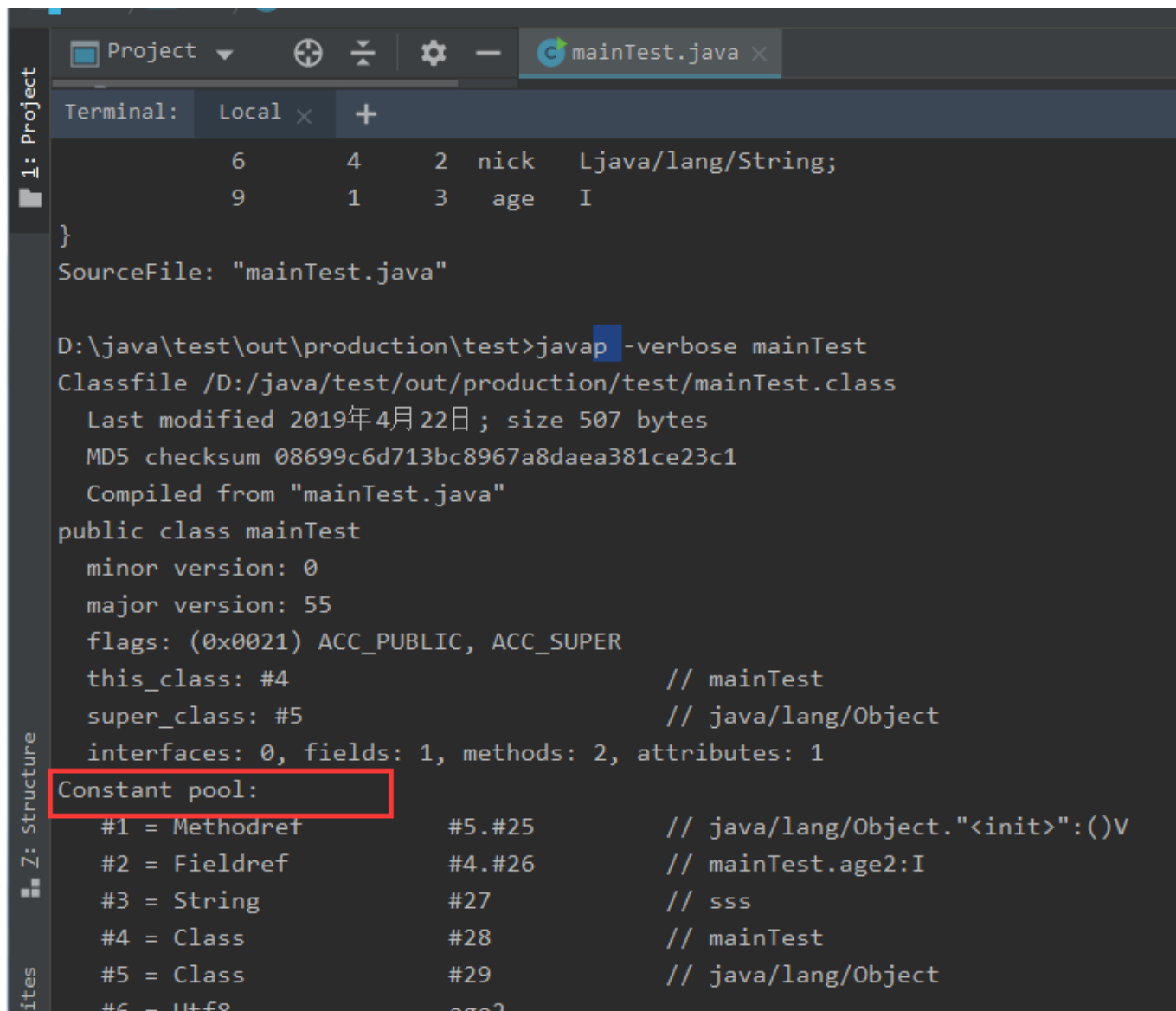
Java源码经过编译后，形成class文件（字节码文件）。

class文件包括MagicNumber, Version, Constant\_pool【常量池】，Access\_flag, This\_class, Super\_class, Interfaces, Fields, Methods 和Attributes这十个部分。

常量池是class文件中的一项非常重要的数据。常量池中存放了文字字符串，常量值，当前类的类名，字段名，方法名，各个字段和方法的描述符，对当前类的字段和方法的引用信息，当前类中对其他类的引用信息等等。



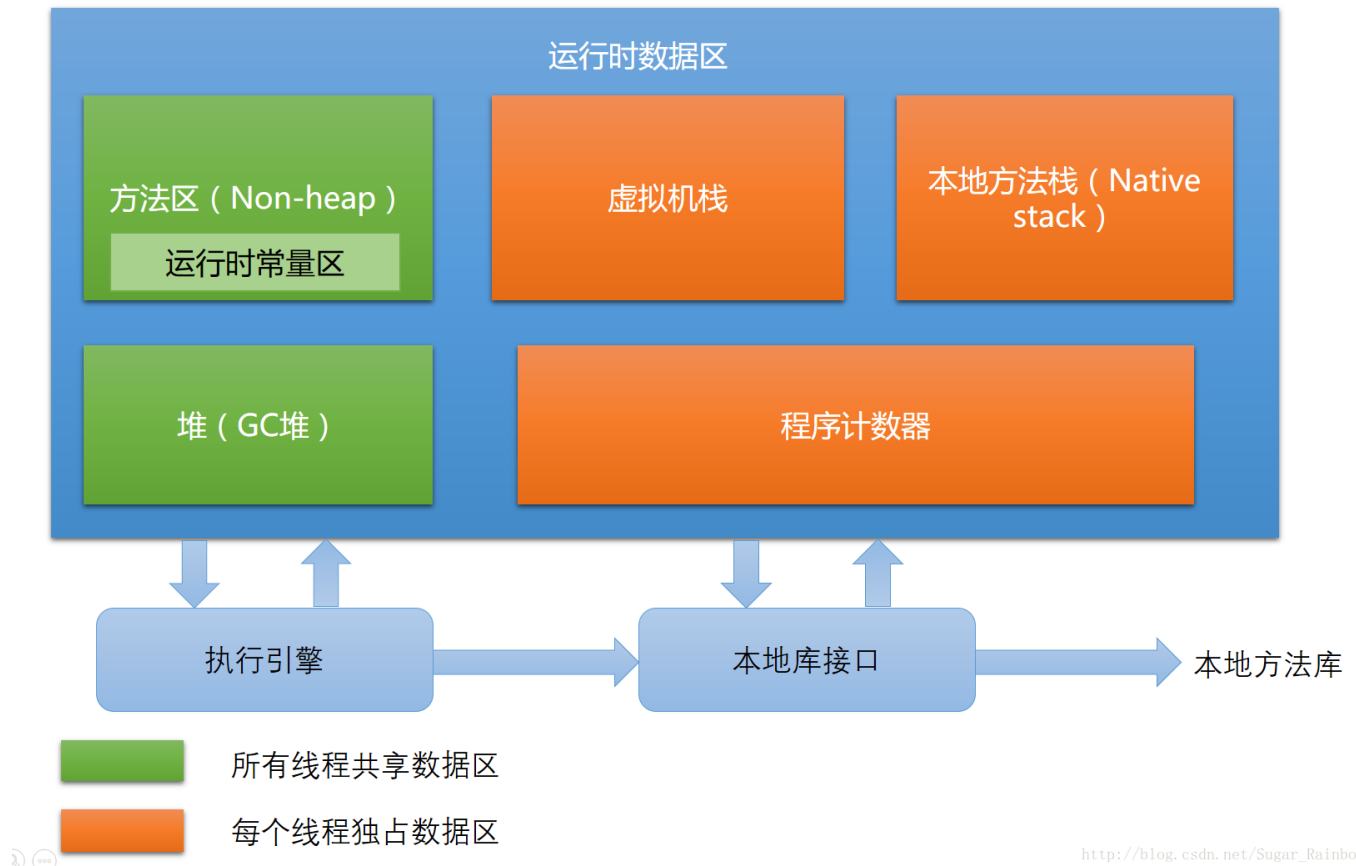
常量池中几乎包含类中的所有信息的描述，class文件中的很多其他部分都是对常量池中的数据项的引用。



```
Project
Terminal: Local x +
mainTest.java x
6      4      2  nick  Ljava/lang/String;
9      1      3   age  I
}
SourceFile: "mainTest.java"

D:\java\test\out\production\test>javap -verbose mainTest
Classfile /D:/java/test/out/production/test/mainTest.class
  Last modified 2019年4月22日; size 507 bytes
  MD5 checksum 08699c6d713bc8967a8daea381ce23c1
  Compiled from "mainTest.java"
public class mainTest
  minor version: 0
  major version: 55
  flags: (0x0021) ACC_PUBLIC, ACC_SUPER
  this_class: #4                // mainTest
  super_class: #5                // java/lang/Object
  interfaces: 0, fields: 1, methods: 2, attributes: 1
Constant pool:
  #1 = Methodref                #5.#25    // java/lang/Object."<init>":()V
  #2 = Fieldref                 #4.#26    // mainTest.age2:I
  #3 = String                   #27        // sss
  #4 = Class                    #28        // mainTest
  #5 = Class                    #29        // java/lang/Object
  #6 = Utf8                     age2
```

当字节码通过classloader加载到虚拟机后，常量池存储到JVM虚拟机的方法区中。此时静态常量池变成静态常量池。



## 运行时常量池

HotSpot VM里，记录interned string的一个全局表叫做StringTable，它本质上就是个HashSet。注意它只存储对java.lang.String实例的引用，而不存储String对象的内容。

```
String s1 = "Hello";
String s2 = "Hello";
String s3 = "Hel" + "lo";
String s4 = "Hel" + new String("lo");
String s5 = new String("Hello");
String s6 = s5.intern();
String s7 = "H";
String s8 = "ello";
String s9 = s7 + s8;

System.out.println(("s1 == s2 : ") + (s1 == s2)); // true
System.out.println(("s1 == s3 : ") + (s1 == s3)); // true
System.out.println(("s1 == s4 : ") + (s1 == s4)); // false
System.out.println(("s1 == s9 : ") + (s1 == s9)); // false
System.out.println(("s4 == s5 : ") + (s4 == s5)); // false
System.out.println(("s1 == s6 : ") + (s1 == s6)); // true
```