jvm学习笔记

4.工具

4.1 jps：虚拟机进程状况工具

```
参数
1.-q 输出LVMID(本地虚拟机唯一ID)，省略主类的名称
2.-m 输出传递到主类main()函数的参数
3.输出主类的全名，若进程执行的是jar的，输出jar路径
4.-v输出jvm启动时的参数
```

4.2 jstat 虚拟机统计信息监视工具

jstat -gcutil 369 1000 3

> 查看进程369的gc情况3次，每个1000ms查看一次

4.3 jinfo：Java配置信息工具

实时查看和调整虚拟机各项参数

4.4 jmap:Java内存映像工具

5.类文件结构

5.1 概述

编译器将代码编译成存储字节码的class文件，虚拟机载入和执行，实现了程序的”一次编写，到处运行“。这些字节码是平台无关性的，可以由各种语言编译而来，比如Scala、Groovy、Jython等，都可以在jvm运行

5.2 class类文件结构

根据Java虚拟机规范的规定，Class文件格式采用一种类似于C语言结构体来存储数据，这种伪数据只有两种数据类型：无符号数和表

> 无符号数：以u1、u2、u4、u8来分别代表1、2、4、8个字节的无符号数，无符号数可以用来描述数字、索引引用、数量值或者按照UTF-8编码构成字符串值
>
> 表是由多个无符号数或者其他表作为数据项构成的复合数据类型

5.2.1 魔数

每个Class文件的头4个字节成为魔数，它的唯一作用是确定这个文件是否为一个能被虚拟机接受的Class文件

紧接着的4个字节存储的是Class文件的版本号，第5和第6个字节是此版本号，第7和第8个是主版本号

5.2.2 常量池

紧接着版本号之后的常量池入口。常量池是占用Class文件空间最大的数据项目之一，也是Class第一个出现的表数据类型项目

常量池中主要存放两大类常量：字面量和符号引用。

> 字面量是比较接近云Java语言层的常量，如文本字符串、声明为final的常量值等
>
> 符号引用：1.类和接口的全限定名；2.字段的名称和描述符；3.方法的名称和描述符

常量池中每一项常量都是一个表，JDK1.7中共有14种表,如下图5-1所示

类　型	标　志	描　述
CONSTANT_Utf8_info	1	UTF-8 编码的字符串
CONSTANT_Integer_info	3	整型字面量
CONSTANT_Float_info	4	浮点型字面量
CONSTANT_Long_info	5	长整型字面量
CONSTANT_Double_info	6	双精度浮点型字面量
CONSTANT_Class_info	7	类或接口的符号引用
CONSTANT_String_info	8	字符串类型字面量
CONSTANT_Fieldref_info	9	字段的符号引用
CONSTANT_Methodref_info	10	类中方法的符号引用
CONSTANT_InterfaceMethodref_info	11	接口中方法的符号引用
CONSTANT_NameAndType_info	12	字段或方法的部分符号引用
CONSTANT_MethodHandle_info	15	表示方法句柄
CONSTANT_MethodType_info	16	标识方法类型
CONSTANT_InvokeDynamic_info	18	表示一个动态方法调用点

图5-1 常量池的项目类型

5.2.3 访问标志

常量池结束后，紧接着的两个字节代表访问标志

标志名称	标 志 值	含　义
ACC_PUBLIC	0x0001	是否为 public 类型
ACC_FINAL	0x0010	是否被声明为 final，只有类可设置
ACC_SUPER	0x0020	是否允许使用 invokespecial 字节码指令的新语意，invokespecial 指令的语意在 JDK 1.0.2 发生过改变，为了区别这条指令使用哪种语意，JDK 1.0.2 之后编译出来的类的这个标志都必须为真
ACC_INTERFACE	0x0200	标识这是一个接口
ACC_ABSTRACT	0x0400	是否为 abstract 类型，对于接口或者抽象类来说，此标志值为真，其他类值为假
ACC_SYNTHETIC	0x1000	标识这个类并非由用户代码产生的
ACC_ANNOTATION	0x2000	标识这是一个注解
ACC_ENUM	0x4000	标识这是一个枚举

图5-2 访问标志

5.2.4 字段表集合

标志名称	标志值	含　义
ACC_PUBLIC	0x0001	字段是否 public
ACC_PRIVATE	0x0002	字段是否 private
ACC_PROTECTED	0x0004	字段是否 protected
ACC_STATIC	0x0008	字段是否 static
ACC_FINAL	0x0010	字段是否 final
ACC_VOLATILE	0x0040	字段是否 volatile
ACC_TRANSIENT	0x0080	字段是否 transient
ACC_SYNTHETIC	0x1000	字段是否由编译器自动产生的
ACC_ENUM	0x4000	字段是否 enum

图5-3 字段访问标志

标识字符	含　义	标识字符	含　义
B	基本类型 byte	J	基本类型 long
C	基本类型 char	S	基本类型 short
D	基本类型 double	Z	基本类型 boolean
F	基本类型 float	V	特殊类型 void
I	基本类型 int	L	对象类型，如 Ljava/lang/Object

图5-4 描述符标识字符含义

对于数组类型，每一维度都会使用一个前置的"["字符来描述，比如定义一个java.lang.Stirng[][]类型的二维数组，会被记录为"[[Ljava/lang/String;"，用'/'替代'.',并在结尾添加';'

用描述符来描述方法时，按照先参数列表，后返回值的顺序描述，参数列表按严格顺序放在一组小括号里面，如

> 方法void inc()的描述符为"()V"
>
> 方法int indexOf(char[] source, int index)的描述符为'([CI)I'

5.3 字节码

字节码详细可见《Java虚拟机规范（Java SE7版）》的第6章

5.3.1 加载和存储

1.将一个局部变量加载到操作栈：iload,fload 2.将数值从操作数站存储到局部变量表：istore 3.将常量加载到操作数栈：bipush,ldc,iconst_ *4.扩充局部变量表的访问索引指令：wide*

5.3.2 运算指令

1. *加法：iadd*
2. *减法：isub*
3. *乘法：imul*
4. *除法：idiv*
5. *求余：irem*
6. *取反：ineg*
7. *位移：ishl、ishr*
8. *按位或：ior、lor*
9. *按位与：iand*
10. *按位异或：ixor*
11. *局部变量自增：iinc*
12. *比较：dcmpg、dcmpl #### 5.3.3 类型转换 可以直接支持小范围类型向大范围类型的安全转换：int->long*

窄话类型转换必须使用显示转换：i2l;窄化转换仅仅丢弃除最低位N个字节以外的内容

5.3.4 对象创建与访问指令

1. *创建实例：new*
2. *创建数组：newarray、anewarray、multianewarray*
3. *访问类字段和实例字段：getfield、getstatic*
4. *将数组元素加载到操作数栈：baload、caload*
5. *取数组长度：arraylength*
6. *检查实例类型：instanceof*

5.3.5 控制转移指令

1. *条件分支：ifeq、iflt、ifle、igt、ige、ifnull、icomeq*
2. *符合条件分支：tableswitch*
3. *无条件分支：goto*

5.3.6 方法调用和返回指令

1. *invokevirtual：用于调用对象的实例方法*
2. *invokeinterface：用于调用接口方法*
3. *invokestatic：调用类方法（static方法）*

参考

1. *深入理解java虚拟机 第二版*
2. *https://www.jianshu.com/p/c69f9f7c273b*