1. 说明

java及class文件在本文档当前目录的classloaderdemo工程中。

1. java文件内容如下

public class Student {  
 private int age;  
  
 public Student(int age) {  
 this.age = age;  
 }

//getter setter ...  
}

package com.yb;  
  
import java.util.ArrayList;  
  
public class Hello {  
 private static ArrayList<Student> students;  
  
 private static final int MIN\_AGE = 18;  
  
 static {  
 students = new ArrayList<Student>();  
 students.add(new Student(17));  
 students.add(new Student(17));  
 students.add(new Student(18));  
 }  
  
 public int checkAge (){  
 int young\_num = 0;  
 int man\_num = 0;  
 for (Student student : students) {  
 int age = student.getAge();  
 if (age < MIN\_AGE){  
 young\_num++;  
 } else {  
 man\_num =man\_num + 1;  
 }  
 }  
  
 int diff = man\_num - young\_num;  
  
 double man\_rate = (man\_num \* 1d)/students.size();  
  
 return young\_num;  
 }  
}

1. javac -c -verbose 反编汇的class文件内容如下

PS E:\designModo\classloaderdemo\target\classes\com\yb> javap -c -verbose .\Hello.class

Classfile /E:/designModo/classloaderdemo/target/classes/com/yb/Hello.class

Last modified 2020-10-18; size 1153 bytes // class文件创建时间

MD5 checksum dfbb9dcc8ee0142f0f3a9d697fbd9020 // MD5检查

Compiled from "Hello.java"

public class com.yb.Hello // 包名

minor version: 0

major version: 49 // jdk5编译

flags: ACC\_PUBLIC, ACC\_SUPER // 类使用public 修饰

Constant pool: // 常量池

#1 = Methodref #14.#43 // java/lang/Object."<init>":()V

#2 = Fieldref #8.#44 // com/yb/Hello.students:Ljava/util/ArrayList;

#3 = Methodref #10.#45 // java/util/ArrayList.iterator:()Ljava/util/Iterator;

#4 = InterfaceMethodref #46.#47 // java/util/Iterator.hasNext:()Z

#5 = InterfaceMethodref #46.#48 // java/util/Iterator.next:()Ljava/lang/Object;

#6 = Class #49 // com/yb/Student

#7 = Methodref #6.#50 // com/yb/Student.getAge:()I

#8 = Class #51 // com/yb/Hello

#9 = Methodref #10.#52 // java/util/ArrayList.size:()I

#10 = Class #53 // java/util/ArrayList

#11 = Methodref #10.#43 // java/util/ArrayList."<init>":()V

#12 = Methodref #6.#54 // com/yb/Student."<init>":(I)V

#13 = Methodref #10.#55 // java/util/ArrayList.add:(Ljava/lang/Object;)Z

#14 = Class #56 // java/lang/Object

#15 = Utf8 students // 所有的成员变量和局部变量的名称都在常量池中

#16 = Utf8 Ljava/util/ArrayList;

#17 = Utf8 Signature

#18 = Utf8 Ljava/util/ArrayList<Lcom/yb/Student;>;

#19 = Utf8 MIN\_AGE

#20 = Utf8 I

#21 = Utf8 ConstantValue

#22 = Integer 18

#23 = Utf8 <init>

#24 = Utf8 ()V

#25 = Utf8 Code

#26 = Utf8 LineNumberTable

#27 = Utf8 LocalVariableTable

#28 = Utf8 this

#29 = Utf8 Lcom/yb/Hello;

#30 = Utf8 checkAge

#31 = Utf8 ()I

#32 = Utf8 age

#33 = Utf8 student

#34 = Utf8 Lcom/yb/Student;

#35 = Utf8 young\_num

#36 = Utf8 man\_num

#37 = Utf8 diff

#38 = Utf8 man\_rate

#39 = Utf8 D

#40 = Utf8 <clinit>

#41 = Utf8 SourceFile

#42 = Utf8 Hello.java

#43 = NameAndType #23:#24 // "<init>":()V

#44 = NameAndType #15:#16 // students:Ljava/util/ArrayList;

#45 = NameAndType #57:#58 // iterator:()Ljava/util/Iterator;

#46 = Class #59 // java/util/Iterator

#47 = NameAndType #60:#61 // hasNext:()Z

#48 = NameAndType #62:#63 // next:()Ljava/lang/Object;

#49 = Utf8 com/yb/Student

#50 = NameAndType #64:#31 // getAge:()I

#51 = Utf8 com/yb/Hello

#52 = NameAndType #65:#31 // size:()I

#53 = Utf8 java/util/ArrayList

#54 = NameAndType #23:#66 // "<init>":(I)V

#55 = NameAndType #67:#68 // add:(Ljava/lang/Object;)Z

#56 = Utf8 java/lang/Object

#57 = Utf8 iterator

#58 = Utf8 ()Ljava/util/Iterator;

#59 = Utf8 java/util/Iterator

#60 = Utf8 hasNext

#61 = Utf8 ()Z

#62 = Utf8 next

#63 = Utf8 ()Ljava/lang/Object;

#64 = Utf8 getAge

#65 = Utf8 size

#66 = Utf8 (I)V

#67 = Utf8 add

#68 = Utf8 (Ljava/lang/Object;)Z

{

public com.yb.Hello(); // 默认的无参构造方法

descriptor: ()V // 无返回值

flags: ACC\_PUBLIC // 类使用public修饰

Code:

stack=1, locals=1, args\_size=1 // 执行该方法时需要的栈深度=1，局部变量表的槽位数=1，入参数=1

0: aload\_0 //

1: invokespecial #1 // Method java/lang/Object."<init>":()V

4: return

LineNumberTable:

line 5: 0

LocalVariableTable:

Start Length Slot Name Signature

0 5 0 this Lcom/yb/Hello;

public int checkAge();

descriptor: ()I // I表示函数返回值为int

flags: ACC\_PUBLIC // 函数使用public进行修饰

Code:

stack=4, locals=6, args\_size=1 // 执行该方法时需要的栈深度=4(栈深度怎么计算？)，局部变量表的槽位数=6，入参数=1(非静态方法默认必须传入this作为第一个参数)

0: iconst\_0 // 将字面值=0的int类型参数压栈

1: istore\_1 // 将栈中值pop并放到1号槽位(young\_num=0)

2: iconst\_0 // 将字面值=0的int类型参数压栈

3: istore\_2 // 将栈中值pop并放到2号槽位(man\_num =0)

4: getstatic #2 // Field students:Ljava/util/ArrayList;

7: invokevirtual #3 //invokevirtual 调用ArrayLit自己的iterator方法,即for循环为语法糖，编译后使用iterator

// Method java/util/ArrayList.iterator:()Ljava/util/Iterator;

10: astore\_3 // 将iterator的addr pop并放到3号槽位

11: aload\_3 // 将3号槽位压栈

12: invokeinterface #4, 1 // InterfaceMethod java/util/Iterator.hasNext:()Z

17: ifeq 58 //ifeq表示value=0，表示hasNext:()==false，跳转到58号指令，表示结束遍历

20: aload\_3 // 将3号槽位压栈

21: invokeinterface #5, 1 // 调用接口方法，即实现类的方法java/util/Iterator.next:()Ljava/lang/Object;

26: checkcast #6 //检查Object是否为Student类型 // class com/yb/Student

29: astore 4 // 将Student的addr出栈，由于此时槽位号>3，故不采用astore\_x的方式

31: aload 4 // 同理，只有aload123才会简写

33: invokevirtual #7 //调用本对象的private方法 // Method com/yb/Student.getAge:()I

36: istore 5

38: iload 5

40: bipush 18 // 将成员变量18入栈。当int值的范围不在-1~5,而在-128~127时(此时int=18)，使用bipush指令压栈

42: if\_icmpge 51 // 比较栈顶的2个int值，如果前者(先入栈的age)大于等于后者(后入栈的18)，则跳转到51序号的指令(51:iload\_2),否则进入分支，执行分支代码(young\_man++)

45: iinc 1, 1 // Increment local variable by constant；1,1=槽位号，增加的常量值。即将1号槽位(young\_num)的值增加1。即young\_num++

48: goto 55 // 跳转到55号指令

51: iload\_2 // 对应代码的else分支。将槽位2(man\_num)入栈

52: iconst\_1 //将常量int=1入栈

53: iadd // 将栈顶的2个int相加

54: istore\_2 //将计算后的结果放到槽位2(man\_num),即man\_num = man\_num +1

55: goto 11 // 跳转到11号指令(遍历下一个元素)，此时完成一个if else

58: iload\_2 // 将2(man\_num)入栈

59: iload\_1 // 将1(young\_num)入栈

60: isub // 计算 ..., value1, value2 →..., result，即操作数栈中用先入栈=先入栈-后入栈

61: istore\_3 // 将差值存放到3号槽位

62: iload\_2 // 将2(man\_num)入栈

63: i2d // 将栈顶的int 转成 double ,为了和1d相乘

64: dconst\_1 // 将常量1d入栈

65: dmul // 栈顶2个值相乘。..., value1, value2 → ..., result，即man\_num \* 1d

66: getstatic #2 // Field students:Ljava/util/ArrayList;

69: invokevirtual #9 // Method java/util/ArrayList.size:()I

72: i2d // 将size()的返回值int转成double,即 students.size()

73: ddiv // 两个double相除, 即 (man\_num \* 1d)/students.size()

74: dstore 4 // 将73的结果pop，并放到4号槽位(原本是student的引用，此处被覆盖)

76: iload\_1 // 将槽位1压栈，即1(young\_num)

77: ireturn // 将栈顶的int值return,即返回young\_num的值。注意return才表示不返回任何值

LineNumberTable: // 表示java原文件和class文件之间行号的映射关系

line 18: 0

line 19: 2

line 20: 4

line 21: 31

line 22: 38

line 23: 45

line 25: 51

line 27: 55

line 29: 58

line 31: 62

line 33: 76

LocalVariableTable:

Start Length Slot Name Signature

38 17 5 age I

31 24 4 student Lcom/yb/Student;

0 78 0 this Lcom/yb/Hello;

2 76 1 young\_num I

4 74 2 man\_num I

62 16 3 diff I

76 2 4 man\_rate D

static {}; // 静态代码块

descriptor: ()V // 无返回值

flags: ACC\_STATIC // static修饰

Code:

stack=4, locals=0, args\_size=0 // 类方法，无this入参

0: new #10 //创建对象(分配内存地址，并将地址压栈),尚未调用构造函数// class java/util/ArrayList

3: dup // 复制栈顶地址

4: invokespecial #11 //调用构造函数初始化对象，并将栈顶地址出栈(由于复制，故操作数栈中还保留一个地址引用)

// Method java/util/ArrayList."<init>":()V

7: putstatic #2 //将栈顶地址出栈赋值给静态变量 // Field students:Ljava/util/ArrayList;

10: getstatic #2 //将静态变量压栈即students = new ArrayList<Student>();

// Field students:Ljava/util/ArrayList;

13: new #6 //分配内存地址并压栈 // class com/yb/Student

16: dup // 复制栈顶地址

17: bipush 17 // 将int=17压栈

19: invokespecial #12 //执行 new Student(17) 入参是int,无返回值,栈顶引用出栈(还剩一个student的引用) // Method com/yb/Student."<init>":(I)V

22: invokevirtual #13 // 由于还有一个student的引用，才能作为该方法的入参

// Method java/util/ArrayList.add:(Ljava/lang/Object;)Z

25: pop // Pop the top operand stack value，将students 的引用出栈

26: getstatic #2 // Field students:Ljava/util/ArrayList;

29: new #6 // class com/yb/Student

32: dup

33: bipush 17

35: invokespecial #12 // Method com/yb/Student."<init>":(I)V

38: invokevirtual #13 // Method java/util/ArrayList.add:(Ljava/lang/Object;)Z

41: pop

42: getstatic #2 // Field students:Ljava/util/ArrayList;

45: new #6 // class com/yb/Student

48: dup

49: bipush 18

51: invokespecial #12 // Method com/yb/Student."<init>":(I)V

54: invokevirtual #13 // Method java/util/ArrayList.add:(Ljava/lang/Object;)Z

57: pop

58: return

LineNumberTable:

line 11: 0

line 12: 10

line 13: 26

line 14: 42

line 15: 58

}

SourceFile: "Hello.java"