Pitifulvm 源码分析

2020年01月06日 乌鑫龙

目录

Pitifulvm 是什么

- Pitifulvm：java虚拟机的简单实现
- C + struct 实现
- ‘简单实现’ -> 侧重体现JVM的运行原理

Github 代码仓库 lazyparser/**pitifulvm**

02

Pitifulvm 的基本结构

Pitifulvm 运行流程

精简字节码结构体
class_file_t

精简字节码指令解释器
execute()

Fopen()

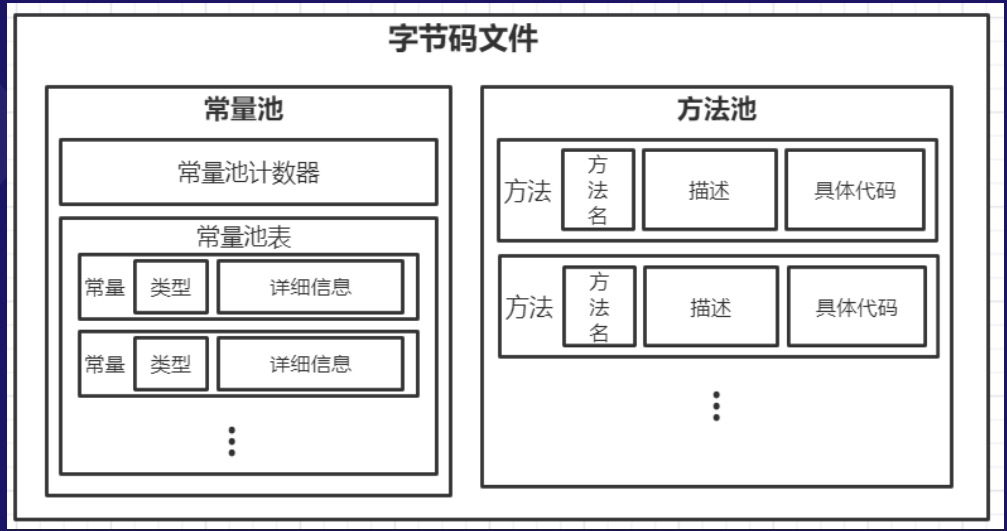
数据类型

class_file_t

```
typedef struct {
    constant_pool_t constant_pool;
    method_t *methods;      // 方法表
} class_file_t;
```

```
typedef struct {
    u2 constant_pool_count;         //uint16_t
    const_pool_info *constant_pool;         //
} constant_pool_t;
```

```
typedef struct {
    const_pool_tag_t tag;
    u1 *info;
    s, 具体内容等
} const_pool_info;
```



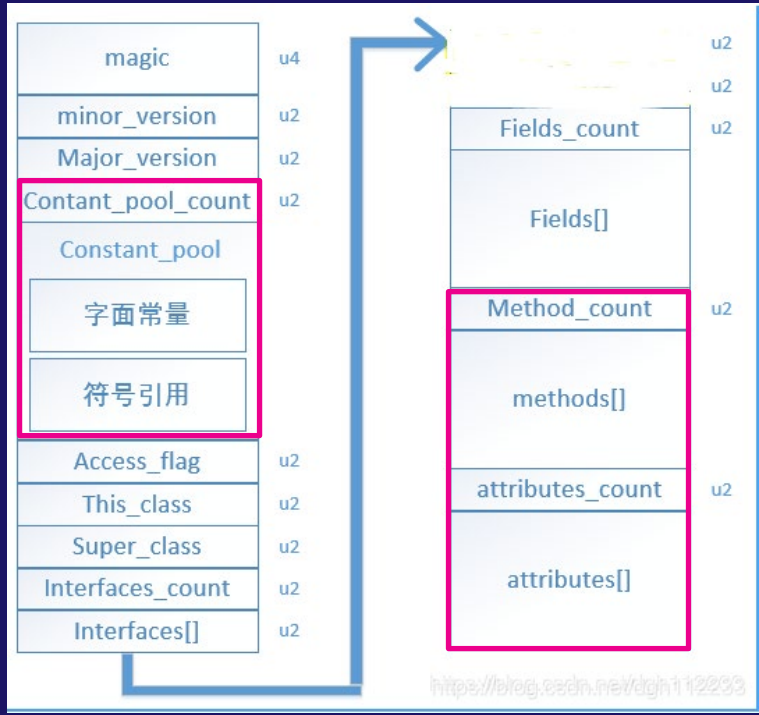
方法 get_class()

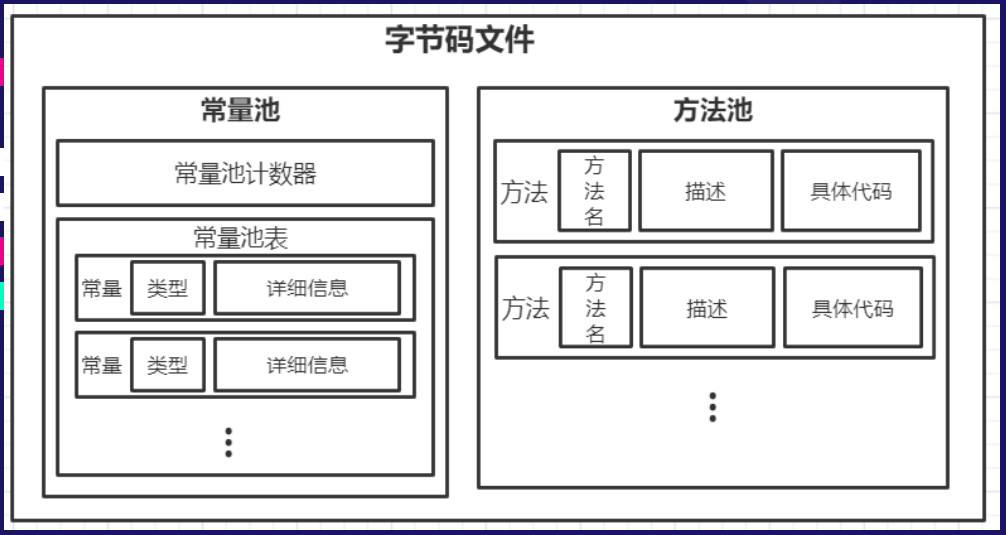
```
class_file_t get_class(FILE *class_file)
{
    /* Read the leading header of the class file */
    get_class_header(class_file);

    /* Read the constant pool */
    class_file_t clazz = {.constant_pool = get_constant_pool(class_file)};

    /* Read information about the class that was compiled. */
    get_class_info(class_file);

    /* Read the list of static methods */
    clazz.methods = get_methods(class_file, &clazz.constant_pool);
    return clazz;
}
```





方法 execute()

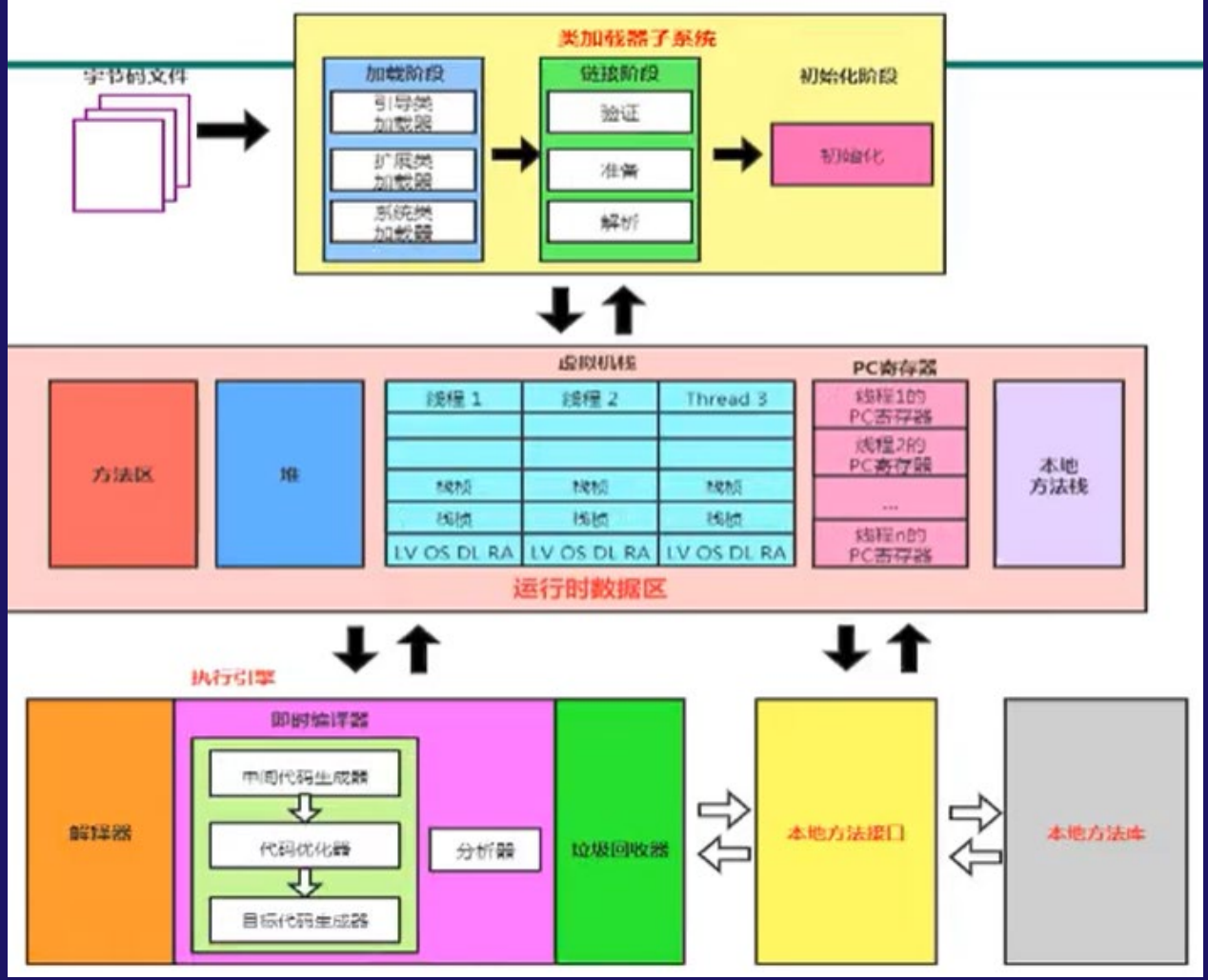
1. 从传入参数中拿到该方法的信息，方法名，参数返回值，最大栈深度等

1. 获取字节码指令，匹配，并且执行相应操作

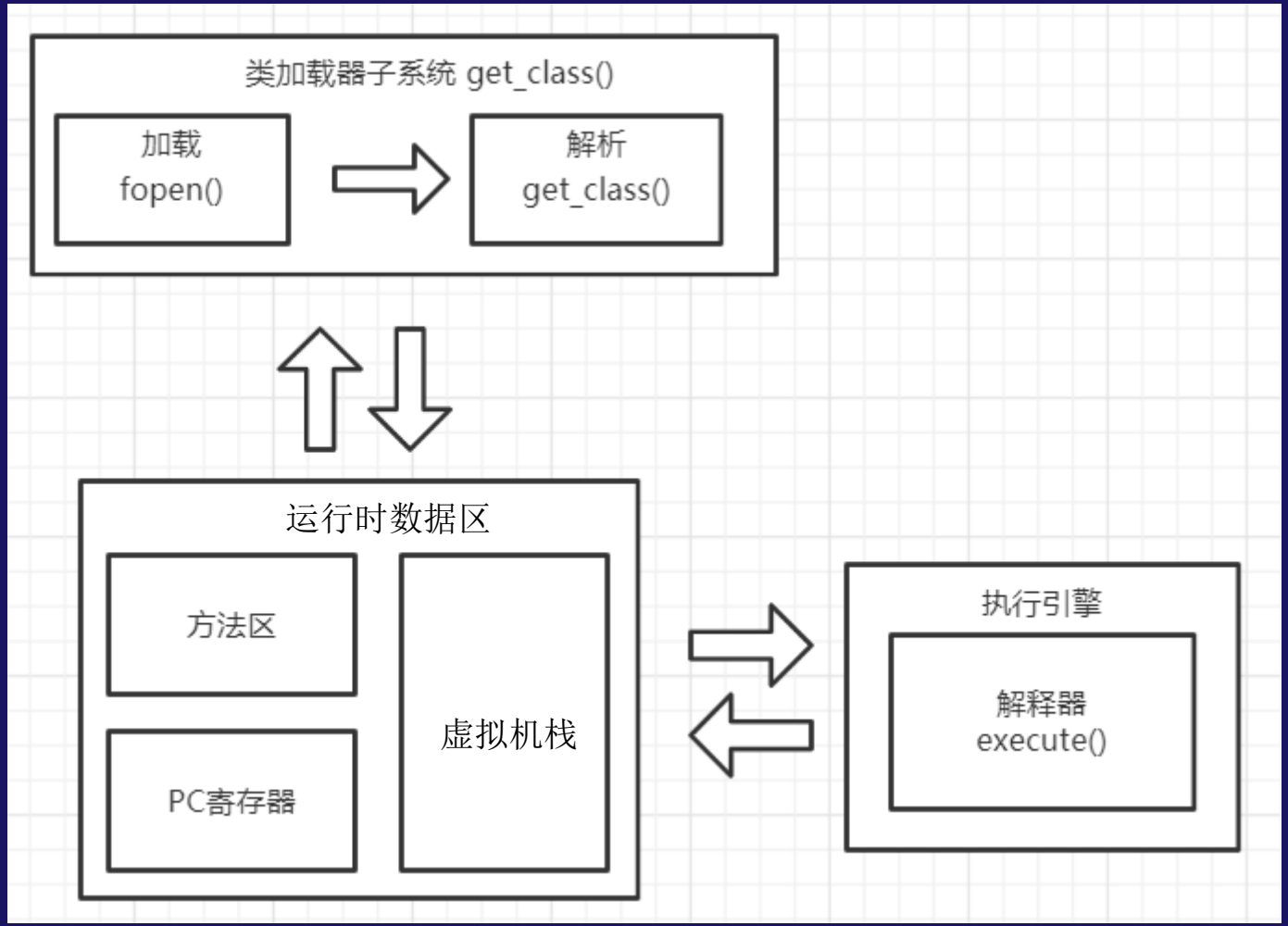
总共实现45条字节码指令，涵盖基本运算加减乘除，流程控制以及方法的调用返回

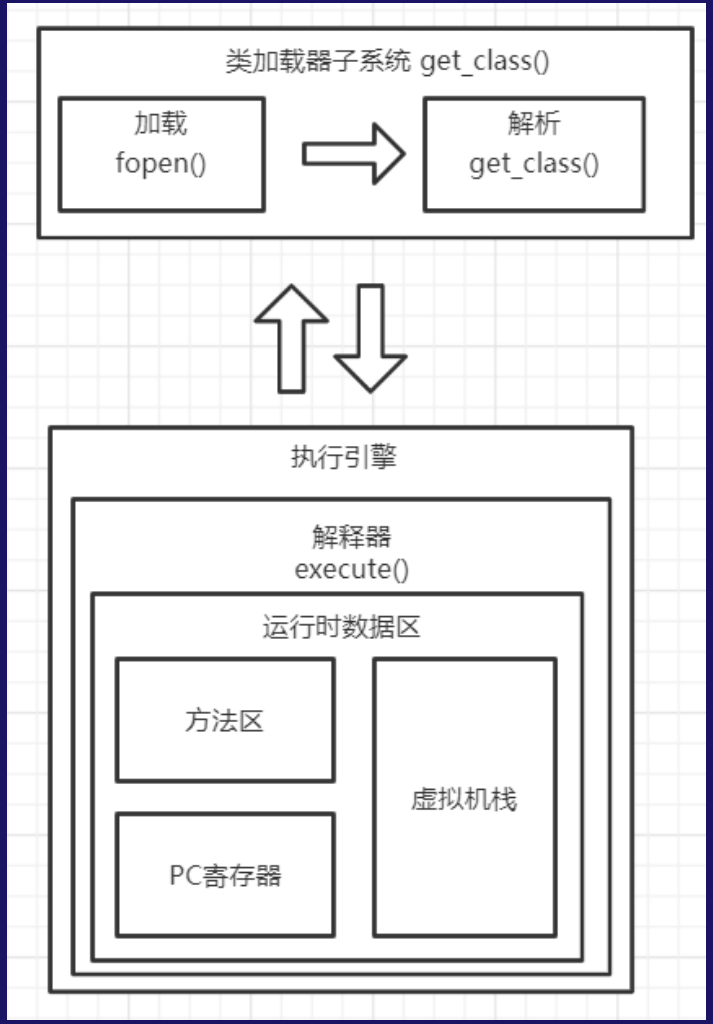
03

Pitifulvm和主流虚拟机的区别


类加载器子系统
字节码文件
加载阶段
引导类加载器
扩展类加载器
系统类加载器
链接阶段
验证
准备
解析
初始化阶段
初始化
虚拟机栈
PC寄存器
方法区
堆
线程 1
线程 2
Thread 3
栈帧
LV OS DL RA
线程1的PC寄存器
线程2的PC寄存器
...
线程n的PC寄存器
本地方法栈
运行时数据区
执行引擎
即时编译器
中间代码生成器
代码优化器
目标代码生成器
分析器
解释器
垃圾回收器
本地方法接口
本地方法库

JVM体系结构


类加载器子系统 get_class()
加载
fopen()
解析
get_class()
运行时数据区
方法区
虚拟机栈
PC寄存器
执行引擎
解释器
execute()




```c
int32_t *execute(method_t *method, int32_t *locals, class_file_t *clazz)
{
    code_t code = method->code;      // 获取方法
    int32_t op_stack[code.max_stack];   // 创建操作数栈

    uint32_t op_count = 0;       // 操作数栈的深度计数器，也就是栈顶指针

    /* position at the program to be run */
    uint32_t pc = 0;         // pc计数器
    uint8_t *code_buf = code.code;       // 方法的jvm代码
```

THINKS

2021年01月06日 乌鑫龙