

СТРУКТУРА БАЙТ-КОДА ВИРТУАЛЬНОЙ МАШИНЫ JAVA

Основным преимуществом использования промежуточного представления кода в процессе компиляции с исходного языка программирования и его запуска на некоторой целевой машине является снижение сложности при проектировании компилятора. Полученные программы могут исполняться на различных архитектурах при условии существования соответствующей виртуальной машины.

Данный подход в разработке компиляторов здесь иллюстрируется на примере виртуальной JVM, которая является уровнем абстракции между кодом и оборудованием.

В состав JVM входит спецификация, реализация JVM и экземпляр JVM. JVM реализует внутреннюю среду исполнения байт-кода. Поддерживается динамическая объектно-ориентированная модель представления программы, как набора классов; при загрузке создается описатель класса с расширением *.class*. Описатель класса содержит ссылки на родительский класс, интерфейсы и т.п. Поддерживается динамическое связывание внешних методов, автоматическая сборка мусора, многопоточность, реализованная в Java на уровне языка.

Типы данных, допустимые в JVM, подразделяются на примитивные и ссылочные типы. Размеры примитивных типов в языке Java и в JVM являются постоянными, не зависящими от платформы.

Ссылочные типы в JVM – ссылки в плоском 32-разрядном адресном пространстве на объекты, экземпляры классов, массивы и интерфейсы.

Основные области памяти, с которыми работает JVM, куча и стек. Используются четыре 32-разрядных регистра. Класс является основной программной единицей платформы Java.

Виртуальная JVM – это программа-интерпретатор, реализуется как стековая машина, выполняющая байт-код в режиме интерпретации. Команды JVM состоят из 1-байтного кода операции, могут содержать операнды. Полный список команд приведен в «Спецификации JVM SE 8» [1]. Инструкция байт-кода обрабатывается, создается нативный код, который передается на исполнение процессору.

Каждый файл *.class* описывает один класс или интерфейс. Файл класса содержит поток байт, структурированный определенным образом. Основные компоненты файла класса:

- верификационная;

- флаг доступа и признак класса или интерфейса;
- пул констант – таблица имен классов и интерфейсов, полей, методов, константы, на которые есть ссылки в файле класса;
- ссылки на имена this-класса и суперкласса в пуле констант;
- интерфейсы, реализуемые классом (ссылки в пул констант);
- описание полей класса (имена, типы, модификаторы и т.д.);
- методы класса (имена, модификаторы, атрибуты и т.д.), главный атрибут метода – это его код: массив байт-кодов метода.

Пример простейшего приложения на Java (файл App.java):

```

package hello;
public class App {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}

```

Скомпилируем файл командой: `javac src/hello/App.java -d classes/`

Класс содержит два метода: конструктор по умолчанию и метод `main`:

	Method 1	Method 2
Constructor/main	00 01 - access_flags 00 07 - name_index 00 08 - descriptor_index 00 01 - attributes_count	00 09 - access_flags 00 0b - name_index 00 0c - descriptor_index 00 01 - attributes_count
Attribute 1	00 09 - name_index (Code) 00 00 00 1d -attribute_length 00 01 - max_stack 00 01 - max_locals 00 00 00 05 - code_length	00 09 - name_index (Code) 00 00 00 25 attribute_length 00 02 - max_stack 00 01 - max_locals 00 00 00 09 - code_length
code[code_length]	2a - aload_0 b7 00 01 - invokespecial b1 - return	b2 00 02 - getstatic 2 (java.lang.System) 12 03 - ldc 3 b6 00 04 - invokevirtual 4 b1 - return
Описание метода	00 00 - exception_table_length 00 01 - attributes_count 00 0a - attribute_name_index 00 00 00 06 -attribute_length 00 01 line_number_table_length 00 00 - start_pc 00 03 - line_number	00 00 -exception_table_length 00 01 - attributes_count 00 0a - attribute_name_index 00 00 00 0a attribute_length 00 02 -line_nuber_table_length 00 00 - start_pc 00 06 - line_number 00 08 - start_pc 00 07 - line_number

Далее следует описание атрибутов класса:

- 00 01 - attributes_count
- 00 0d - name_index (SourceFile)
- 00 00 00 02 - attributes_length
- 00 0e - sourcefile_index(App.java)

Содержимое файла App.class можно просмотреть в шестнадцатеричном редакторе, содержимое которого с комментариями приведено ниже:

The image shows a hex dump of a Java class file with several annotations in Russian explaining the fields and constants. The hex dump is organized into lines of 16 bytes each, with corresponding ASCII characters shown below. Annotations include:

- 4 байта - magic, который определяет формат файла**: Points to the magic number `ca fe ba be`.
- minor version major version**: Points to the version numbers `00 00 00 34`.
- размер constant_pool (пула констант)**: Points to the constant pool size `0a`.
- 1 эл.:** `0a - CONSTANT_Methodref`, `00 06 - class_index`, `указывает на`, `00 0f - name_and_type_index`. Points to the first constant pool entry.
- 3 эл.:** `08 - тэг для CONSTANT_String`, `указывает на`, `00 12 - string_index`. Points to the string constant entry.
- 11 эл.:** `01 - тэг CONSTANT_Utf8`, `00 04 - длина`, `6d 61 69 6e - строка «main»`. Points to the UTF-8 string constant entry.
- 5 эл.:** `07 - тэг для CONSTANT_Class`, `00 15 - название, которого в 21 эл.`. Points to the class constant entry.
- 28 эл.:** `01 - тэг CONSTANT_Utf8`, `00 15 - длина`, `строка (Ljava/lang/String;)V` в Utf8, `последний элемент в таблице constant_pool`. Points to the UTF-8 signature constant entry.
- this_class - индекс в таблице constant_pool, указывающий на структуру CONSTANT_Class_info**: Points to the `00 05` field.
- super_class 00 06 - индекс в таблице constant_pool**: Points to the `00 06` field.
- количество интерфейсов**, **количество полей**: Points to the `00 25` field.
- Байт-код метода:** `2a - aload_0`, `b7 00 01 - invokespecial`, `b1 - return`. Points to the method bytecode.
- 2 метода в классе: конструктор по умолчанию**, **и метод main**: Points to the `00 02` field.

Изложенный материал может быть полезен студентам и магистрантам, изучающим языки программирования и принципы компиляции.

ЛИТЕРАТУРА

1 The Java Virtual Machine Specification, Java SE 8 Edition [Электронный ресурс] -Режим доступа: <https://docs.oracle.com/javase/specs/jvms/se8/html/index.html> (дата обращения 21.01.2021).