1. **Dalvik VM(DVM)和 Java VM(JVM)的区别：**

   1) JVM 是运行 Java 字节码，DVM 是运行 Dalvik 字节码

   2) Dalvik 可执行文件(.dex)的体积更小

   3) 虚拟机架构不同：JVM 是基于栈，DVM 是基于寄存器

   Java 代码：

```
public class Hello {
    public int foo(int a, int b) {
        return (a + b) * (a - b);
    }

    public static void main(String[] argc) {
        Hello hello = new Hello();
        System.out.println(hello.foo(5, 3));
    }
}
```

   反编译 int foo(int a, int b)函数的 Java 字节码：

```
public int foo(int, int);
  Code:
   0:   iload_1
   1:   iload_2
   2:   iadd
   3:   iload_1
   4:   iload_2
   5:   isub
   6:   imul
   7:   ireturn
```

   反编译 int foo(int a, int b)函数的 Dalvik 字节码：

```
      Hello.foo:(II)I
0000: add-int v0, v3, v4
0002: sub-int v1, v3, v4
0004: mul-int/2addr v0, v1
0005: return v0
```

2. **Dalvik 汇编语言介绍**

   v 命名法与 p 命名法：

V命名法	p命名法	寄存器含义
v0	v0	第1个局部变量寄存器
v1	v1	第2个局部变量寄存器
…	…	中间的局部变量寄存器
vM-N	p0	第1个参数寄存器（通常为调用对象）
…	…	中间的参数寄存器
vM-1	pN-1	第N个参数寄存器

   类型描述符：

语法	含义
V	void
Z	boolean
B	byte
S	short
C	char
I	int
J	long
F	float
D	double
L	java类
[	数组

寄存器：

1) DVM 寄存器都是 32bit 的，与名称无关

2) J，D 类型，需要相邻 2 个寄存器

3) 对象类型：Ljava/lang/String; = java.lang.String

4) 数组：[I = int []，[[I = int [][]

方法：

1) 格式：Lpackage/name/ObjectName;->MethodName(III)Z

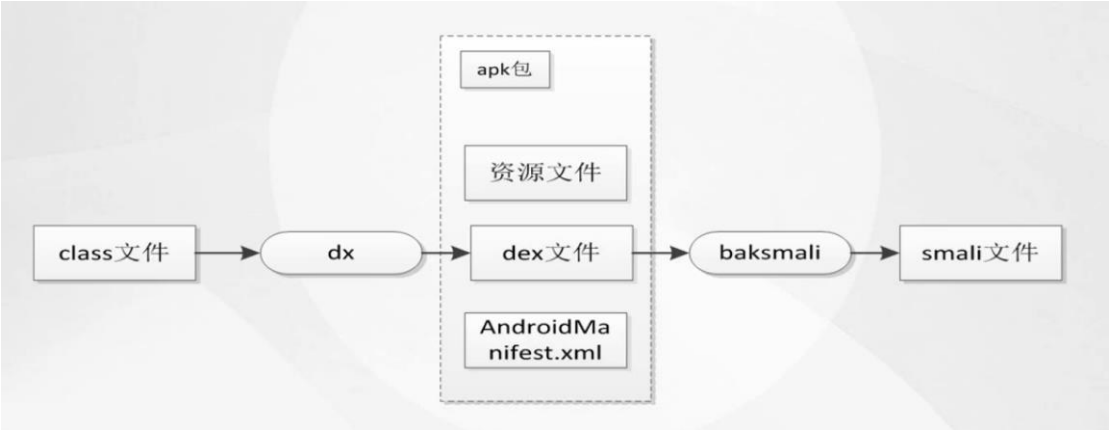
2) 例子：method(I[[IILjava/lang/String;[Ljava/lang/Object;)Ljava/lang/String

等价于：String method(int, int[][], int, String, Object[])

字段：

格式：Lpackage/name/ObjectName;->FieldName:Ljava/lang/String;

3. 程序编译与反编译



主要的反编译器：

1)jeb；2)AndroidKiller

Dalvik 指令集：

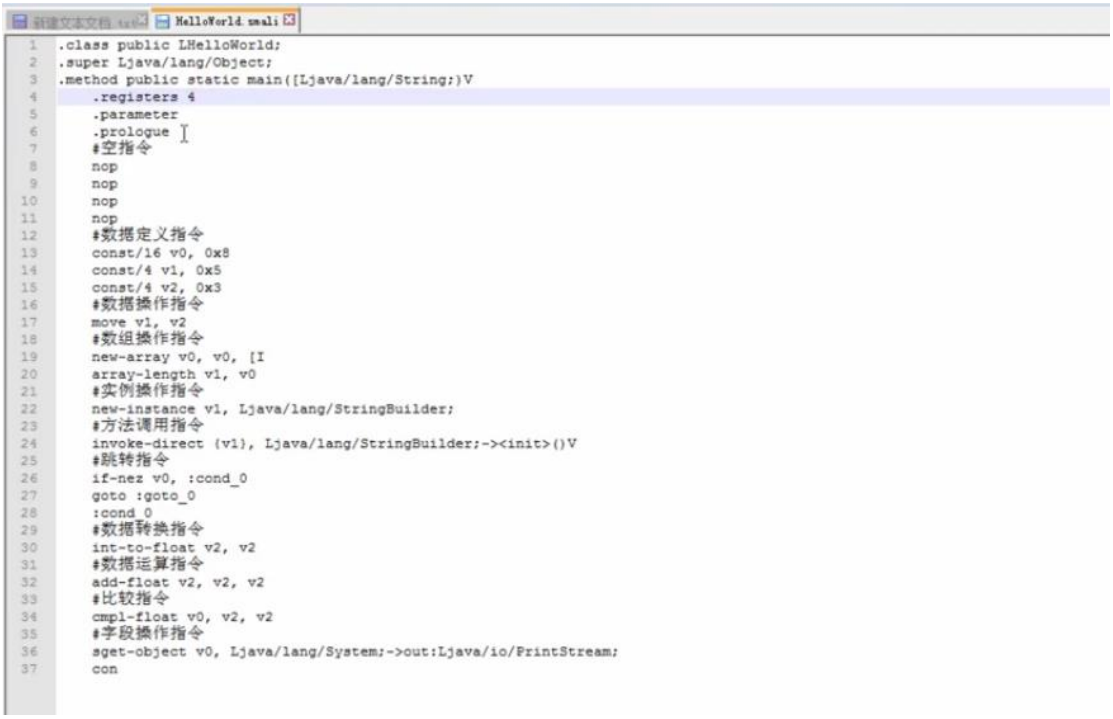
1) 空操作指令：nop

2) 数据操作指令：move

move vA, vB：将 vB 寄存器的值赋给 vA 寄存器，源寄存器与目的寄存器都是 4 位

move-object/from 16 vAA, vBBBB：为对象赋值，源寄存器为 8 位，目的寄存器为 16 位

4. 汇编语言实现



```
.class public LHelloWorld;
.super Ljava/lang/Object;
.method public static main([Ljava/lang/String;)V
    .registers 4
    .parameter
    .prologue
    #空指令
    nop
    nop
    nop
    nop
    #数据定义指令
    const/16 v0, 0x8
    const/4 v1, 0x5
    const/4 v2, 0x3
    #数据操作指令
    move v1, v2
    #数组操作指令
    new-array v0, v0, [I
    array-length v1, v0
    #实例操作指令
    new-instance v1, Ljava/lang/StringBuilder;
    #方法调用指令
    invoke-direct {v1}, Ljava/lang/StringBuilder;-><init>()V
    #跳转指令
    if-nez v0, :cond_0
    goto :goto_0
    :cond_0
    #数据转换指令
    int-to-float v2, v2
    #数据运算指令
    add-float v2, v2, v2
    #比较指令
    cmpl-float v0, v2, v2
    #字段操作指令
    sget-object v0, Ljava/lang/System;->out:Ljava/io/PrintStream;
    con
```

对应的 Java 代码：

```
HelloWorld.class
import java.io.PrintStream;

public class HelloWorld
{
  public static void main(String[] paramArrayOfString)
  {
    int[] arrayOfInt = new int[8];
    arrayOfInt.length;
    new StringBuilder();
    if (arrayOfInt == null)
      return;
    float f1 = 3;
    float f2 = f1 + f1;
    (f2 < f2);
    System.out.println("Hello World");
  }
}
```

编译 smali 文件：

java –jar smali.jar –o classes.dex HelloWorld.smali

执行程序：

上传到手机：adb push classes.dex /data/local/

执行程序：adb shell dalvikvm –cp /data/local/classes.dex HelloWorld