Android Smali

بسم الله الرحمن الرحيم اليوم الشرح بيكون عن android smali

What is android smali

```
chatgpt:
Smali is a low-level programming language used for developing Android applications. It is a human-readable representation of the Dalvik Virtual Machine (DVM) bytecode, which is the runtime environment used by Android to execute Android application code. Smali code is written in a text format that is designed to be both human-readable and writable, making it easier for developers to disassemble, modify, and reassemble Android apps
```

بمعنى انها مثل الاسمبلي بس خاصة للأندويد و تعتبر اسهل من الاسمبلي كرأيي شخصي

Why should i know about it?

كثير من التطبيقات يكون فيها anti decompile او حتى ال decompiler حقك تصير له مشاكل بال decompile و هذا شي طبيعي ف في هذي الحالة احد الحلول اللي ممكن تسويها انك تقرا smali و هو الحل اللي افضلة شخصيا لأنها سهلة جدا

```java
package jakhar.aseem.diva;

import android.os.Bundle;
import android.support.p003v7.app.AppCompatActivity;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

/* loaded from: classes.dex */
public class Hardcode2Activity extends AppCompatActivity {
    private DivaJni djni;

    /* JADX INFO: Access modifiers changed from: protected */
    @Override // android.support.p003v7.app.AppCompatActivity, android.support.p000v4.app.FragmentActivi
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(C0319R.layout.activity_hardcode2);
        this.djni = new DivaJni();
    }

    public void access(View view) {
        EditText hckey = (EditText) findViewById(C0319R.C0321id.hc2Key);
        if (this.djni.access(hckey.getText().toString()) != 0) {
            Toast.makeText(this, "Access granted! See you on the other side :)", 0).show();
        } else {
            Toast.makeText(this, "Access denied! See you in hell :D", 0).show();
        }
    }
}
```

هنا عندنا كود بسيط بالجافا و بعدها بحولة الى samli تقدر تلاقي الخيار في jadx تحت

```
.source "Hardcode2Activity.java"


# instance fields
.field private djni:Ljakhar/aseem/diva/DivaJni;


# direct methods
.method public constructor <init>()V
    .registers 1

    .prologue
    .line 9
    invoke-direct {p0}, Landroid/support/v7/app/AppCompatActivity;-><init>()V

    return-void
.end method


# virtual methods
.method public access(Landroid/view/View;)V
    .registers 6
    .param p1, "view"    # Landroid/view/View;

    .prologue
    const/4 v3, 0x0

    .line 22
    const v1, 0x7f0c007e

    invoke-virtual {p0, v1}, Ljakhar/aseem/diva/Hardcode2Activity;->findViewById(I)Landroid/view/V
```

هنا نفس الكود بس samli بناخذ الmethod الاولى و نفهم كيف شغال

```
.method public constructor <init>()V
```
هنا يقولك بالبداية انها method و نوعها public و اسم الmethod و في الاخير ()init و هذا يعني ان الmethod هذي ما تاخذ ال arg و ال V بالاخير يعني void بمعنى ان ال method ماراح تسوي return لأي قيمة

```
.registers 1
 .prologue
 .line 9
invoke-direct {p0}, Landroid/support/v7/app/AppCompatActivity;-><init>()V




    return-void
```
اول 3 سطور غير مهمة السطر الرابع بيسوي invoke يعني بيستخدم فنكشن و ال p0 يعني اول parameter

```
p0 = first parameter
p1 =  second parameter
p2 = third parameter
......
```
و راح يسوي invoke لفنكشن AppCompatActivity
اذا الشرح ماكان واضح جدا هذي فنكشن ثانية اسهل

```
move-result-object v1

const-string v2, "vendorsecretkey"

invoke-virtual {v1, v2}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z

move-result v1

if-eqz v1, :cond_24

.line 20
const-string v1, "Access granted! See you on the other side :)"

invoke-static {p0, v1, v3}, Landroid/widget/Toast;->makeText(Landroid/content/Co

move-result-object v1

invoke-virtual {v1}, Landroid/widget/Toast;->show()V

.line 25
:goto_23
return-void

.line 23
:cond_24
const-string v1, "Access denied! See you in hell :D"

invoke-static {p0, v1, v3}, Landroid/widget/Toast;->makeText(Landroid/content/Co

move-result-object v1
```

```
const-string v2, "vendorsecretkey"
```

السطر هذا بيعرف متغير و بيحفظ قيمتة في v2

```
let v2 = "vendorsecretkey" #javascript
v2 = "vendorsecretkey" #python
```

نفس الكود حق smali بس كتبته ب python و جافاسكربت

```
invoke-virtual {v1, v2}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z
```

الكود هذا بيسوي invoke لفنكشن equals لمتغير v1 و متغير v2 و راح يشيك هل هم متشابهين او لا

```
move-result v1
```

بعدين بيحفظ النتيجة في v1

```
if-eqz v1
```

و هنا بيسوي شرط تحقق اذا كانت v1 تساوي true او false

```
const-string v1, "Access granted! See you on the other side :)"
```

```
    invoke-static {p0, v1, v3}, Landroid/widget/Toast;-
>makeText(Landroid/content/Context;Ljava/lang/CharSequence;I)Landroid/widg
et/Toast;



    move-result-object v1



    invoke-virtual {v1}, Landroid/widget/Toast;->show()V
```

اذا true يتساوون بيشغل هذا الكود اللي بيسوي متغير v1 و بعدها بيسوي makeText بالمتغير هذا و راح يسوي له show بالاخير

```
 const-string v1, "Access denied! See you in hell :D"

    invoke-static {p0, v1, v3}, Landroid/widget/Toast;-
>makeText(Landroid/content/Context;Ljava/lang/CharSequence;I)Landroid/widg
et/Toast;

    move-result-object v1

    invoke-virtual {v1}, Landroid/widget/Toast;->show()V

    goto :goto_23
```

اذا النص ما يتشابة بيسوي نفس الشي بس النص اللي يطلع يختلف

Trick when analysis smali

```
const-string = set variable
invoke-static,invoke-virtual  = run function
if-*** =  if statement
```

بالغالب انت بتركز عالثلاث ذول و دايم بتلاحظ samli فيه تفاصيل اكثر ممكن تسحب عليها لأن ال decompiler يخفيها لك عشانك ما تحتاجها بالغالب

في تحليل ال smali انصحك تفتح هذي الصفحة معك بتساعدك لو نسيت شي او ما عرفت شي

GitHub - LaurieWired/SmaliReference: Smali reference for reverse engineering

Dalvik Bytecode

GitHub

الملخص

كثير من برامج ال apk مسوين حركات مثل anti decompile و وحدة من الحلول و افضلها انك تقرا ال smali كود و منها تفهم البرنامج كيف شغال و ممكن تكتشف ثغرات بسهولة و افضل طريقة عشان تتعلم عليها خذ ال apk و شف ال java decompile و شف ال smali و اربط بينهم و مع الوقت بتقرا تقرا smali بكل سهولة