BSidesSF 2015 Workshop

Introduction to Reverse Engineering Android Applications

VerSprite

www.versprite.com
@VerSprite

#! whoami

- Lead Security Researcher @VerSprite Security
- Mobile & Embedded Security
- CTF'er, Reverse Engineering, Exploit Development
- @rotlogix

VerSprite

Training Overview

- Hour One (DECK) - Android Architecture, Components, and Inner Component Communication (ICC)
- Hour Two (LAB) - Introduction to Reverse Engineering with Androguard and Friends
- Hour Three: (LAB) - Bypass Emulator Detection in CrackME!
- Hour Four: (LAB) - Solve CrackME!

VerSprite

Environment

- Genymotion with ARM Translation
- Android 4.3 VM
- CrackME! Installed
- Apktool, Dex2Jar, Androguard, JD-GUI, Android SDK (Android Studio), Java

VerSprite

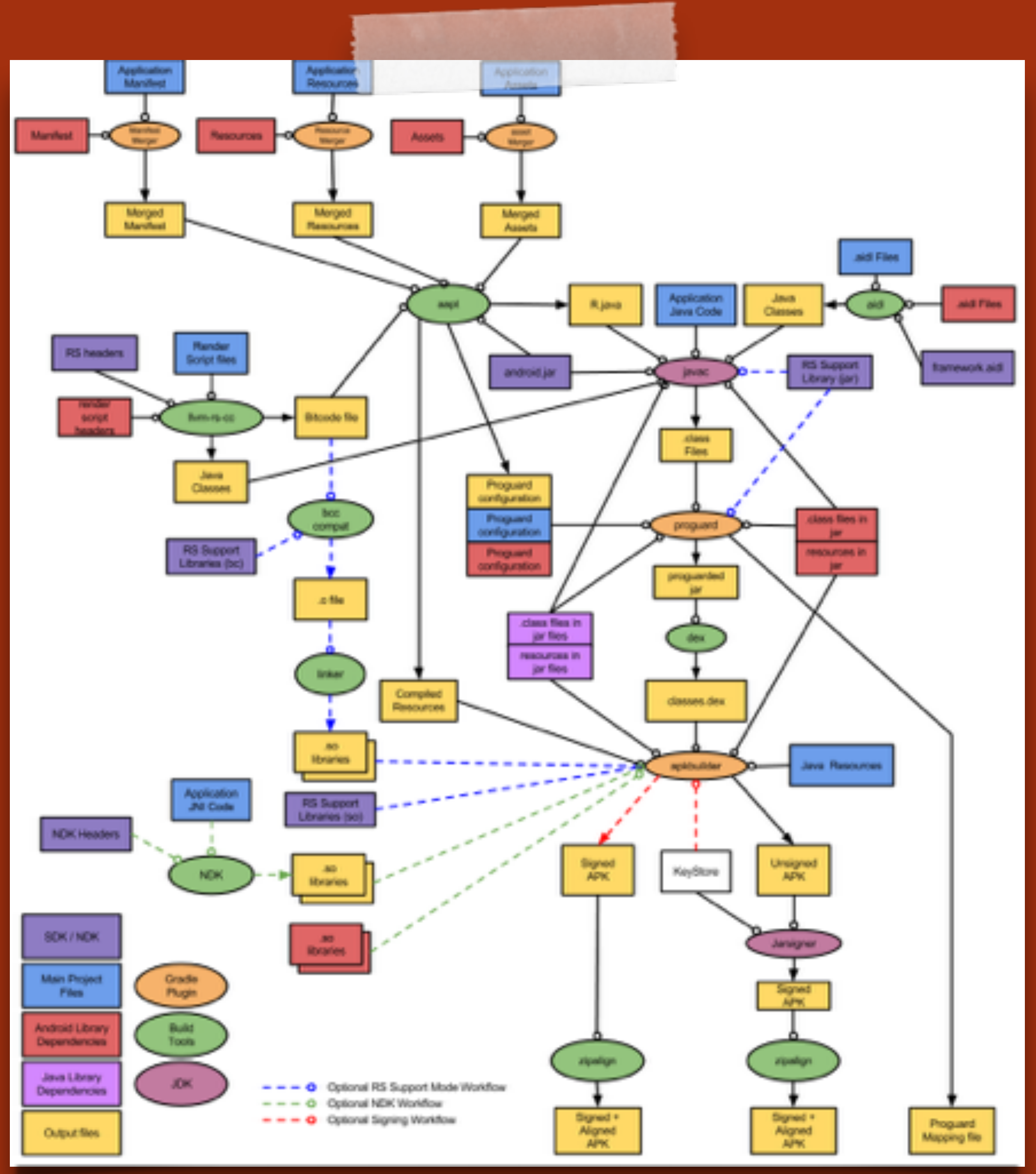
Android Architecture

- Applications are written in Java and compiled into Dalvik Executables (DEX)
- Application and Framework code executes within the Dalvik Virtual Machine
- Native Libraries and Services are written in C & C++

VerSprite

Android Architecture | Build System

- Java code is compiled by the Java compiler into .class files
- The dex tool coverts the .class files into Dalvik bytecode
- The apkbuilder builds everything to an Android Package (APK)
- APK is signed with a code signing key



Android Architecture | Components

- Application Components are declared in the AndroidManifest.xml
- System instantiates and tears down components by driven events in the system and UI
- Activities, Services, Broadcast Receivers, Content Providers
- Inner Component Communication (ICC)
- Component Communication is driven through Intents (IPC)

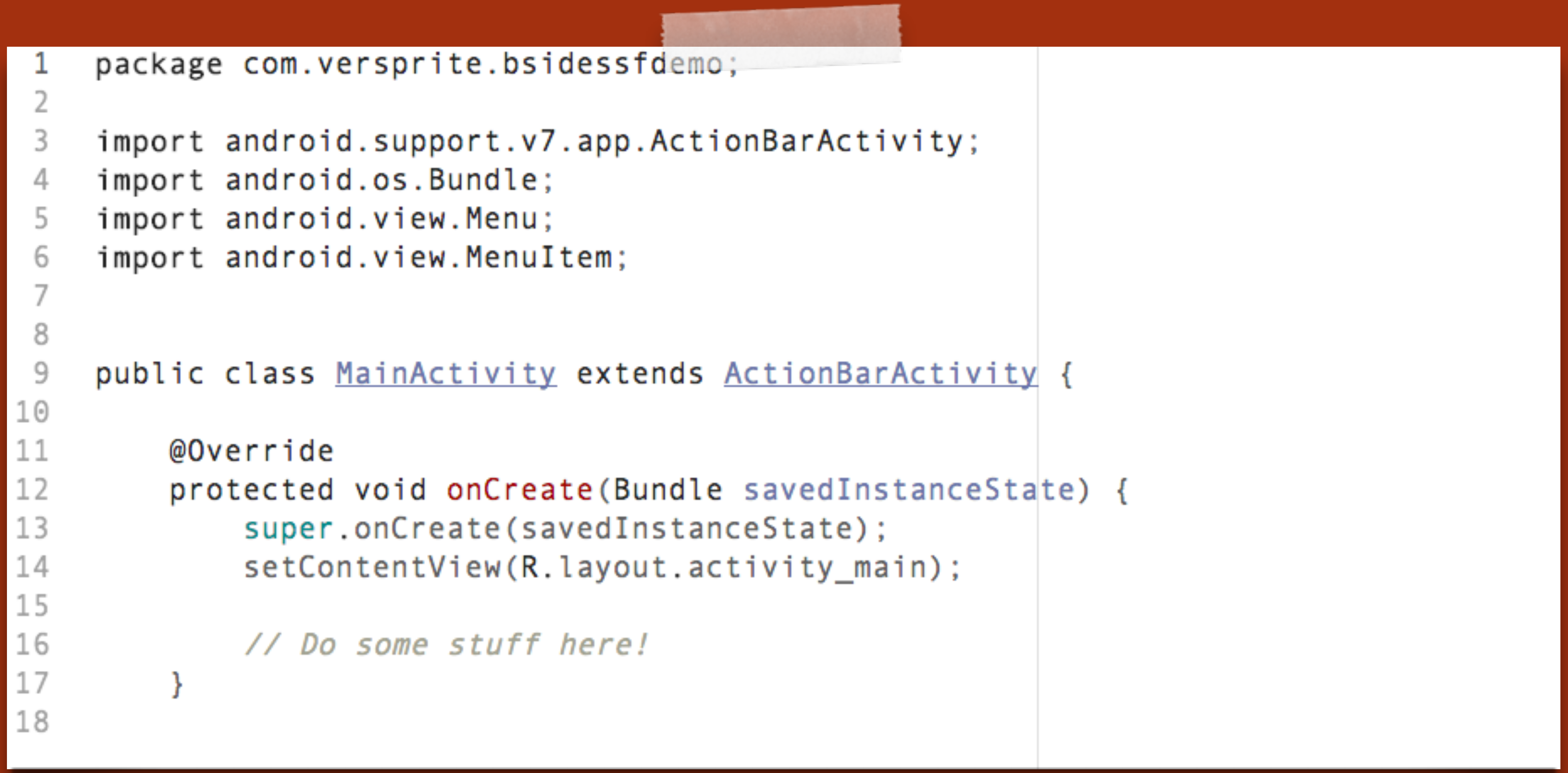
VerSprite

Android Architecture | Components | Activities

- Application Component that provides a screen which users can interact with
- Applications usually have multiple Activities
- onCreate() is the method called by the system when your Activity is first created
- Components can start Activities by pass Intents to them

VerSprite

Android Architecture | Components | Activities



```
package com.versprite.bsidessfdemo;

import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;


public class MainActivity extends ActionBarActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Do some stuff here!
    }

```

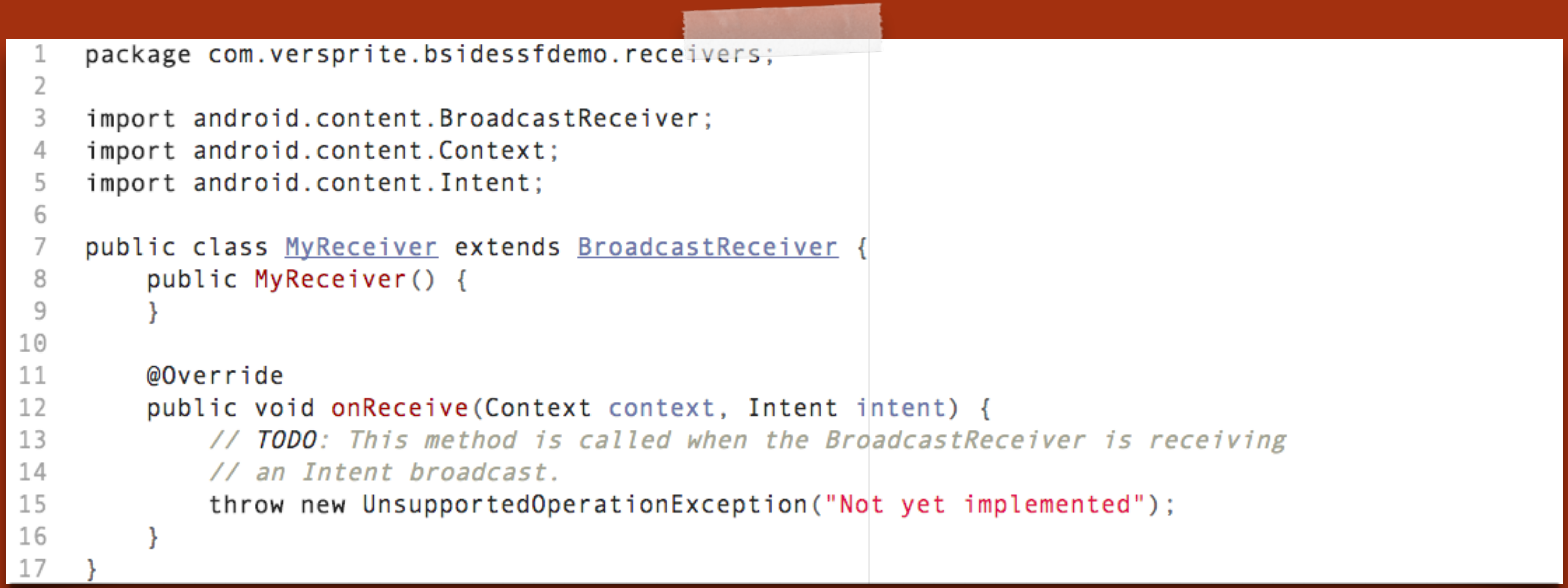
VerSprite

Android Architecture | Components | Broadcast Receivers

- IPC Endpoints
- Receive system wide broadcasts (Intents)
- onReceive() is called on registered Broadcast Receivers whenever the event occurs
- Broadcast events can either be generated by the system or applications

VerSprite

Android Architecture | Components | Broadcast Receivers



```java
package com.versprite.bsidessfdemo.receivers;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;

public class MyReceiver extends BroadcastReceiver {
    public MyReceiver() {
    }

    @Override
    public void onReceive(Context context, Intent intent) {
        // TODO: This method is called when the BroadcastReceiver is receiving
        // an Intent broadcast.
        throw new UnsupportedOperationException("Not yet implemented");
    }
}
```

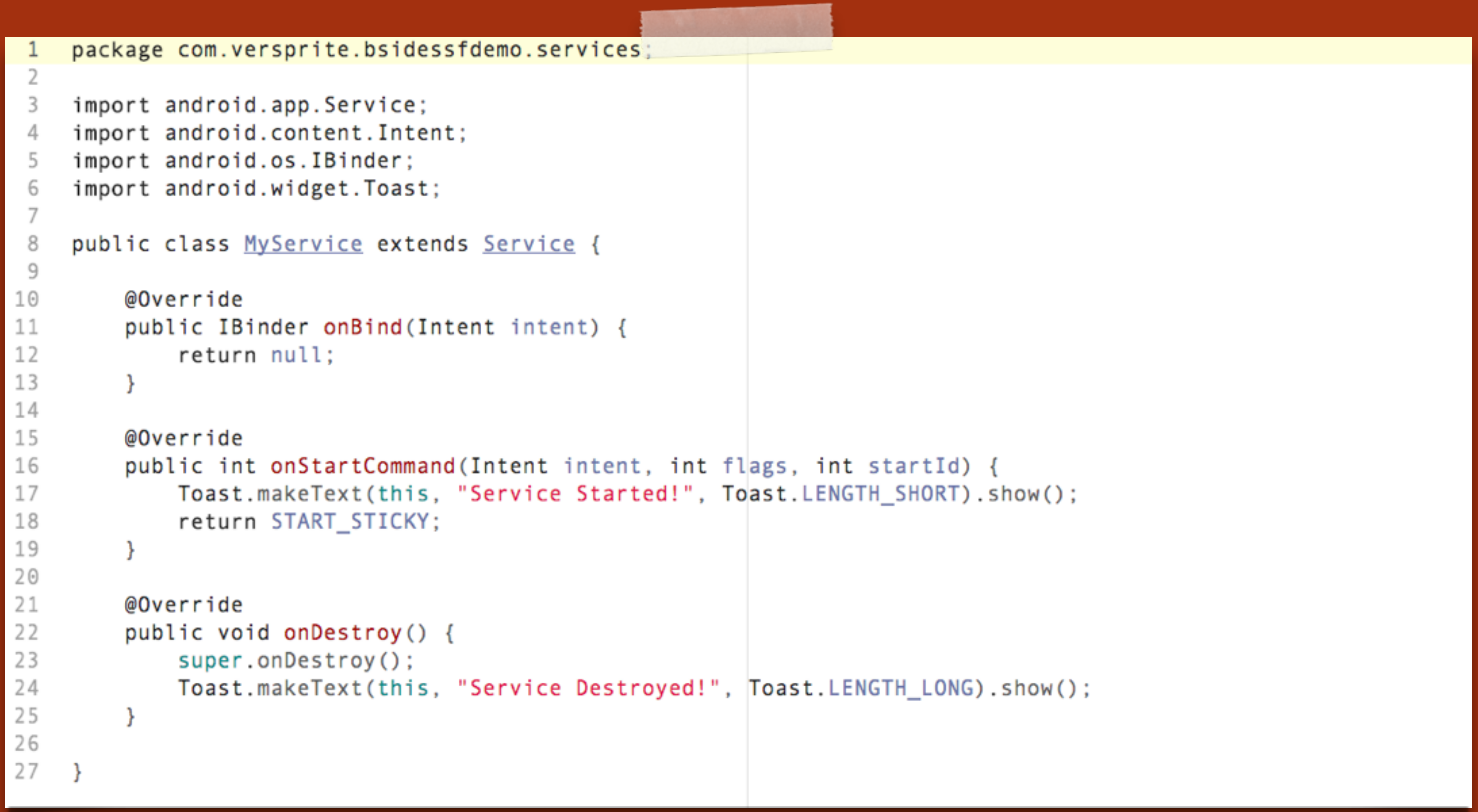
VerSprite

Android Architecture | Components | Services

- IPC Endpoints
- System calls onStartCommand() when another method wants to start the service
- System calls onBind() when another component wants to bind to the service

VerSprite

Android Architecture | Components | Services



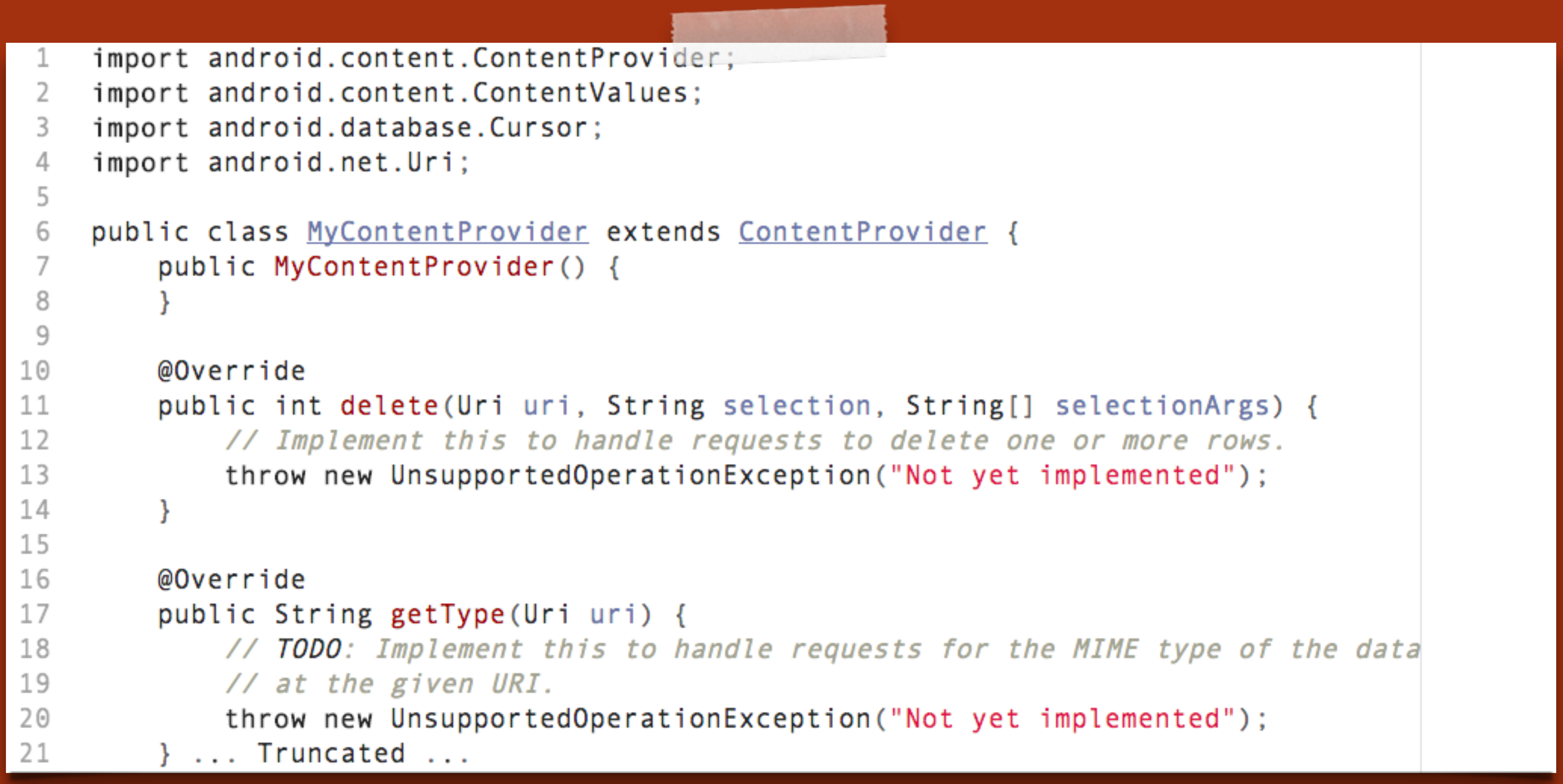
```java
package com.versprite.bsidessfdemo.services;

import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.widget.Toast;

public class MyService extends Service {

    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        Toast.makeText(this, "Service Started!", Toast.LENGTH_SHORT).show();
        return START_STICKY;
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        Toast.makeText(this, "Service Destroyed!", Toast.LENGTH_LONG).show();
    }

}
```

VerSprite

Android Architecture | Components | Content Providers

- Another form of IPC
- Standard interface that connects data between processes
- Can take the form of a SQLite database or file directory
- Content URIs define the data in the provider
- content://my_users/passwords

Android Architecture | Components | Content Providers



```java
import android.content.ContentProvider;
import android.content.ContentValues;
import android.database.Cursor;
import android.net.Uri;

public class MyContentProvider extends ContentProvider {
    public MyContentProvider() {
    }

    @Override
    public int delete(Uri uri, String selection, String[] selectionArgs) {
        // Implement this to handle requests to delete one or more rows.
        throw new UnsupportedOperationException("Not yet implemented");
    }

    @Override
    public String getType(Uri uri) {
        // TODO: Implement this to handle requests for the MIME type of the data
        // at the given URI.
        throw new UnsupportedOperationException("Not yet implemented");
    } ... Truncated ...
```

VerSprite

Android Architecture | Permissions

- Enforces restrictions on the specific operations that a process can perform
- Additional capabilities that are not provided by the Android sandbox
- Example - SEND_SMS, WRITE_EXTERNAL, INTERNET
- You are presented with accepting these permissions everytime you install a new application

VerSprite

Android Architecture | Intents

- High level abstraction layer of IPC in Android
- Communication objects that carry operations for components to act upon
- Commonly used to start Activities, send Broadcasts, and starting Services
- Supports the concept of Inner Component Communication (ICC)

VerSprite

Android Architecture | Intents

- Two types of Intents:
  - Explicit Intents – specify the component we want to communicate with
  - Implicit Intents – no specific component, only a general action
  - Implicit Intents usually result in vulnerabilities

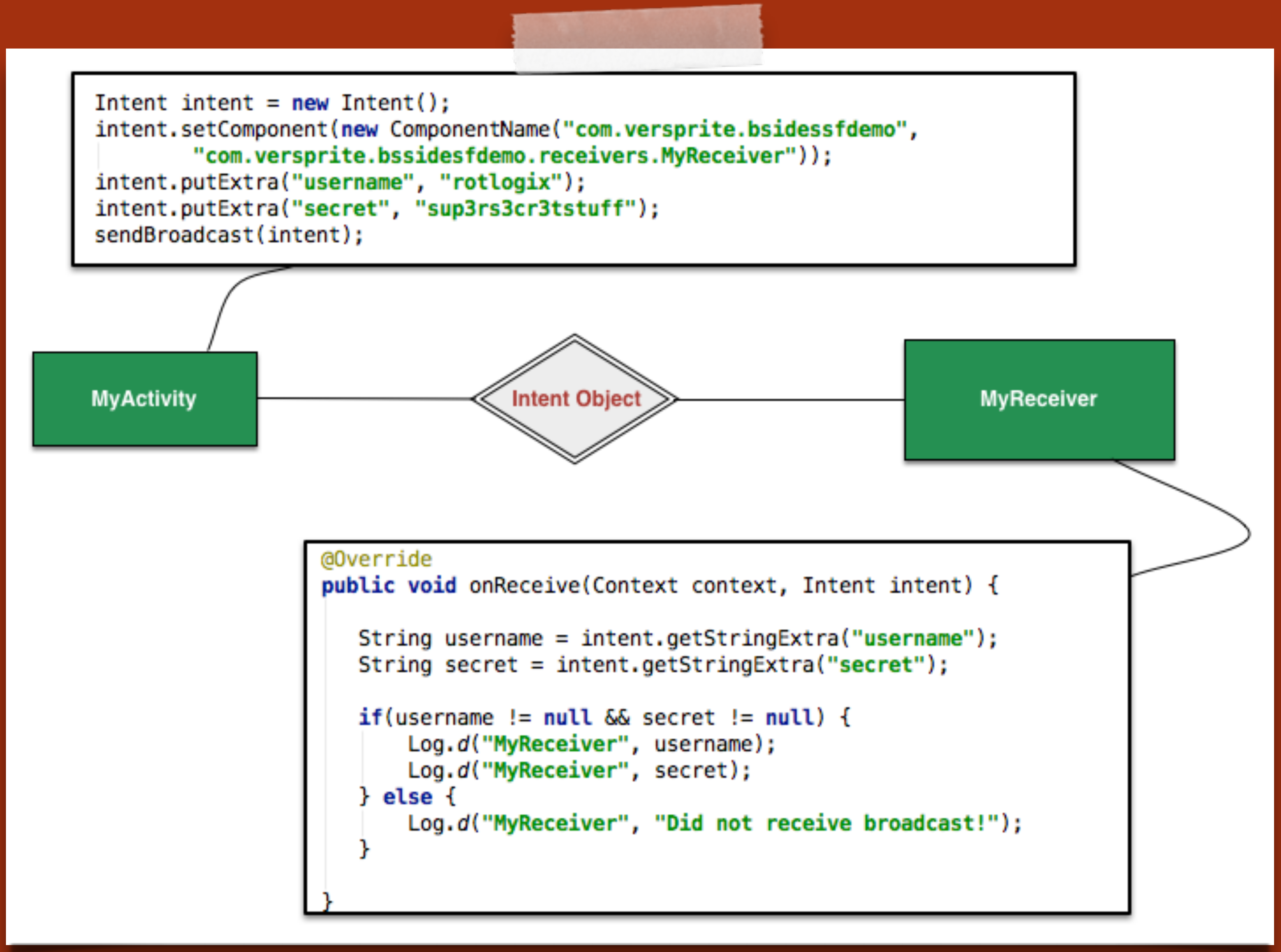
Android Architecture | Intents

- Intent Structure:
    - component - The name of component to start
    - action - General action to be performed, such as ACTION_VIEW, ACTION_EDIT
    - data - The data to perform an operation on

VerSprite

Android Architecture | Intents

Intent Communication Example



```java
Intent intent = new Intent();
intent.setComponent(new ComponentName("com.versprite.bsidessfdemo",
        "com.versprite.bssidesfdemo.receivers.MyReceiver"));
intent.putExtra("username", "rotlogix");
intent.putExtra("secret", "sup3rs3cr3tstuff");
sendBroadcast(intent);
```

```java
@Override
public void onReceive(Context context, Intent intent) {

   String username = intent.getStringExtra("username");
   String secret = intent.getStringExtra("secret");

   if(username != null && secret != null) {
       Log.d("MyReceiver", username);
       Log.d("MyReceiver", secret);
   } else {
       Log.d("MyReceiver", "Did not receive broadcast!");
   }

}
```

VerSprite

Android Architecture | Manifest

- Contains component and permission definitions for the application
- Will reveal fully qualified name for each component – com.application.package/.MainActivity
- Intent Filters are declared here for each component that needs to receive Intent communication
- Declares which components are exported (!)

VerSprite

Android Architecture | Component Exportation

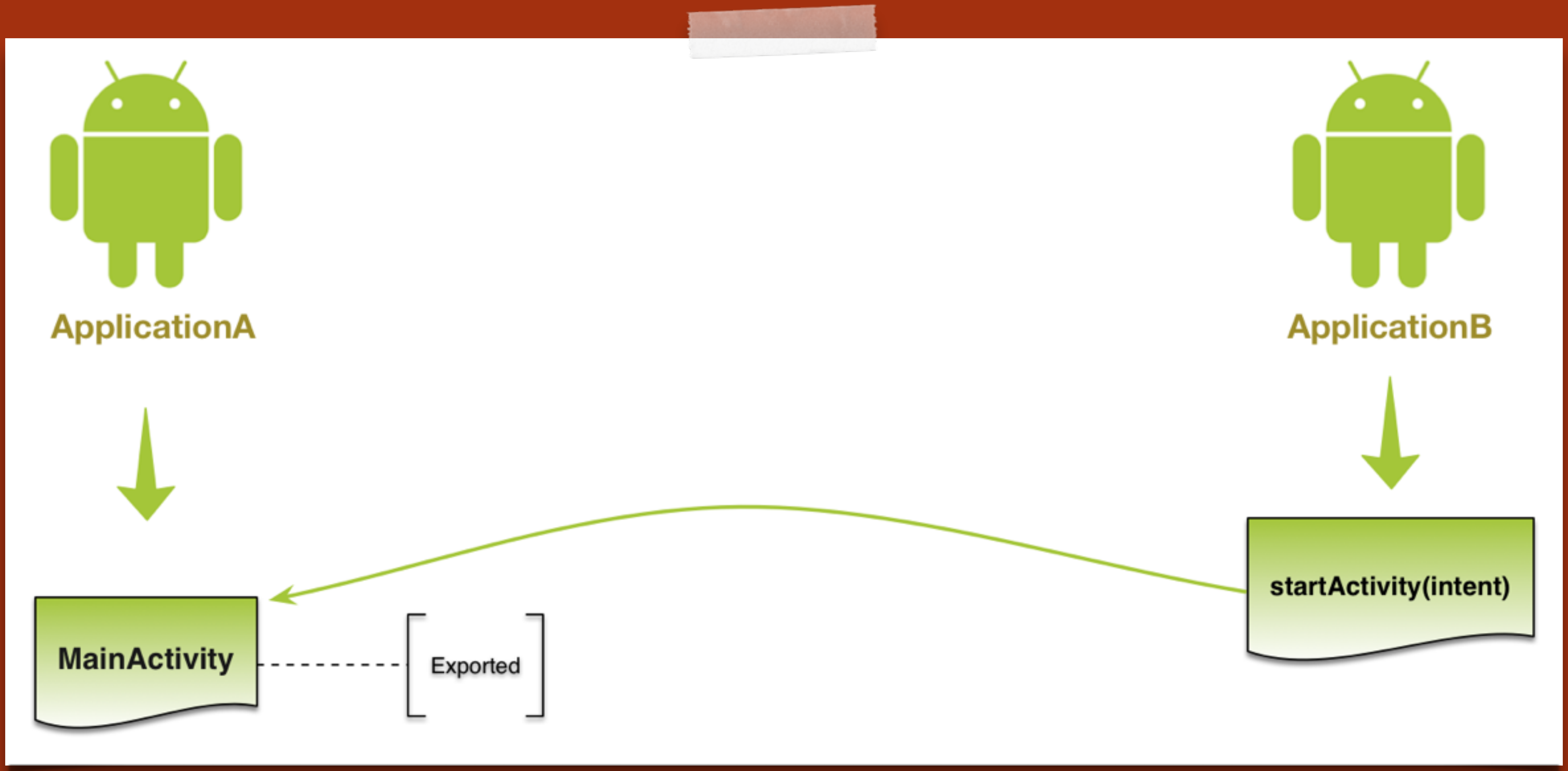
- Applies to Activities, Content Providers, Services, and Broadcast Receivers
- Tag: android:exported=(True|False)
- The presence of Intent Filters defaults component exportation to True
    - You must explicitly set the attribute android:exported=False in this case

VerSprite

Android Architecture | Component Exportation

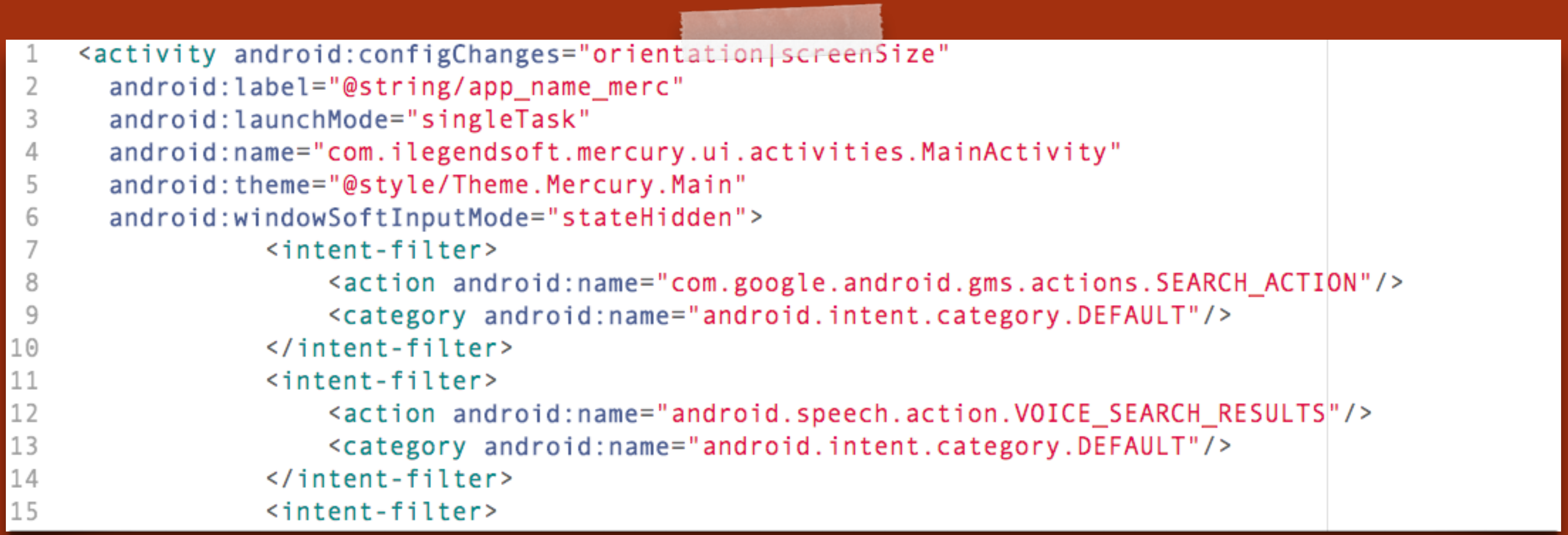
- Exported Components define our attack surface
- Supports component invocation and communication from external applications
- Exported Activities are common

Android Architecture | Component Exportation



VerSprite

Android Architecture | Manifest Example



```
<activity android:configChanges="orientation|screenSize"
  android:label="@string/app_name_merc"
  android:launchMode="singleTask"
  android:name="com.ilegendsoft.mercury.ui.activities.MainActivity"
  android:theme="@style/Theme.Mercury.Main"
  android:windowSoftInputMode="stateHidden">
          <intent-filter>
              <action android:name="com.google.android.gms.actions.SEARCH_ACTION"/>
              <category android:name="android.intent.category.DEFAULT"/>
          </intent-filter>
          <intent-filter>
              <action android:name="android.speech.action.VOICE_SEARCH_RESULTS"/>
              <category android:name="android.intent.category.DEFAULT"/>
          </intent-filter>
          <intent-filter>
```

VerSprite

Android Architecture | The Dalvik VM

- Java is compiled in .class files (Java bytecode)
- The Dalvik dx tool compiles your class files into one Dalvik Executable
    - `dx --dex --output=WhoDat.dex WhoDis.class`
- The Dalvik VM loads the DEX and begin interpreting the bytecode

VerSprite

Android Architecture | The Dalvik VM

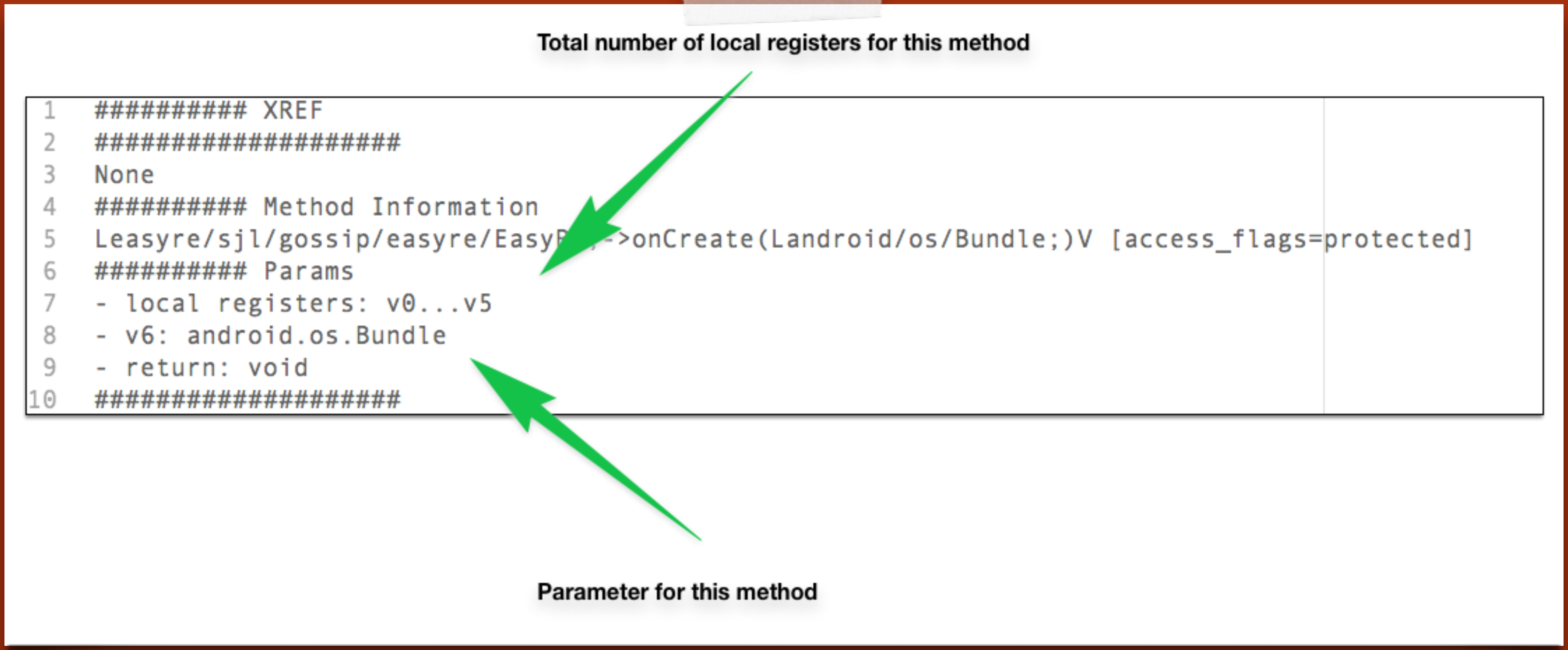
- Register-Based Virtual Machine
- Instead of using a stack structure for operating on data
    - Operands are stored on the registers of the DVM
    - There are no PUSH or POP operations (Stack)
    - The operands are addressed in the instruction
    - No Stack Pointer to reference the operands
    - Register to Register data transference and operations

Android Architecture | The Dalvik VM

- Registers are 32-bits wide
- Registers can hold any data type (String, Int, Float, Boolean)
- Instructions are 16-bits

VerSprite

Android Architecture | The Dalvik VM


Total number of local registers for this method
Parameter for this method

```
########## XREF
####################
None
########## Method Information
Leasyre/sjl/gossip/easyre/EasyR[illegible]->onCreate(Landroid/os/Bundle;)V [access_flags=protected]
########## Params
- local registers: v0...v5
- v6: android.os.Bundle
- return: void
####################
```

Android Architecture | Dalvik Bytecode

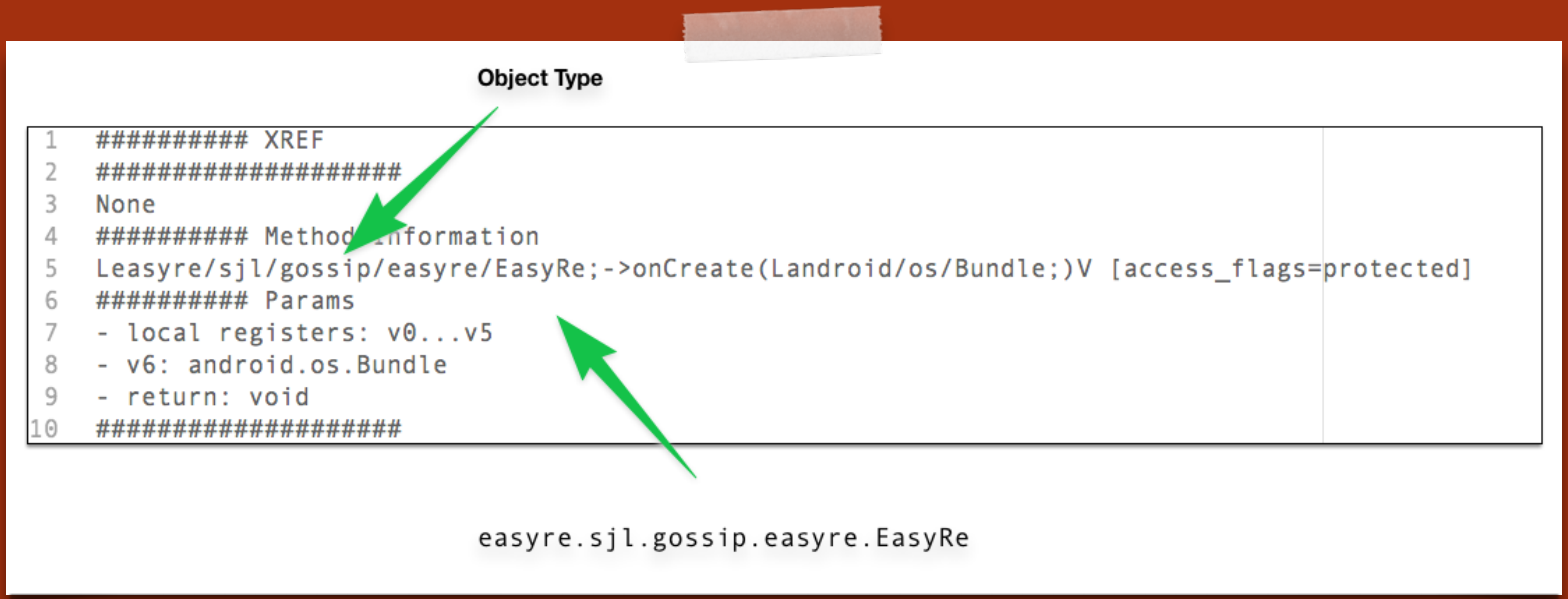
- Two major classes of type: primitive types and reference types
    - Reference types are any instantiable class as well as arrays
    - Primitive types are the basic data types (byte, short, integer, char)
- Primitives are represented by a single letter
    - V, Z, B, S, C, I, J, F, D
    - void, boolean, byte, short, char, integer, long, float, double

Android Architecture | Dalvik Bytecode | Objects

- Objects take the form: Lpackage/name/ObjectName;
- Leading ‘L’, lets you know it is of the object type

VerSprite

Android Architecture | Dalvik Bytecode | Objects



```
########## XREF
####################
None
########## Method Information
Leasyre/sjl/gossip/easyre/EasyRe;->onCreate(Landroid/os/Bundle;)V [access_flags=protected]
########## Params
- local registers: v0...v5
- v6: android.os.Bundle
- return: void
####################
```

easyre.sjl.gossip.easyre.EasyRe

VerSprite

Android Architecture | Dalvik Bytecode | Fields

- Declaration contains the field, the name of the field, and the type of the field
- Lpackage/name/MyObject;->MyField:Ljava/lang/String;

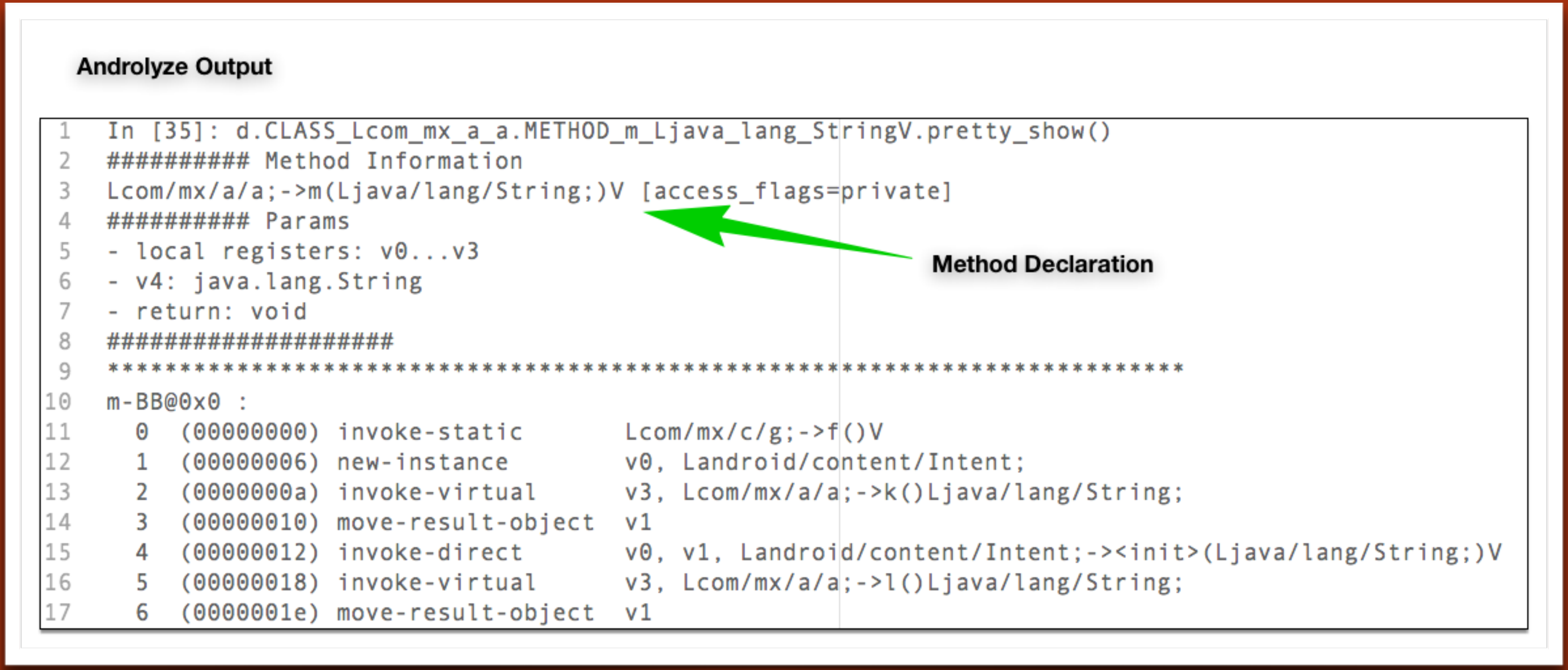
VerSprite

Android Architecture | Dalvik Bytecode | Methods

- Verbose Declaration
- Lpackage/name/MyObject;->MyMethod(III)Z
- Takes three arguments of the integer type, and returns a boolean type

VerSprite

Android Architecture | Dalvik Bytecode | Methods



```
In [35]: d.CLASS_Lcom_mx_a_a.METHOD_m_Ljava_lang_StringV.pretty_show()
########## Method Information
Lcom/mx/a/a;->m(Ljava/lang/String;)V [access_flags=private]
########## Params
- local registers: v0...v3
- v4: java.lang.String
- return: void
####################
***************************************************************************
m-BB@0x0 :
  0  (00000000) invoke-static       Lcom/mx/c/g;->f()V
  1  (00000006) new-instance        v0, Landroid/content/Intent;
  2  (0000000a) invoke-virtual      v3, Lcom/mx/a/a;->k()Ljava/lang/String;
  3  (00000010) move-result-object  v1
  4  (00000012) invoke-direct       v0, v1, Landroid/content/Intent;-><init>(Ljava/lang/String;)V
  5  (00000018) invoke-virtual      v3, Lcom/mx/a/a;->l()Ljava/lang/String;
  6  (0000001e) move-result-object  v1
```

VerSprite

Android Architecture | Dalvik Bytecode

- Smali is an assembler dissembler format to represent the DEX format

VerSprite

Break!

VerSprite

Lab Time!

- Lab-001
- Lab-002
- Lab-003

VerSprite