WHAT IS

# ANDROID

Stanislaw Baranski
https://stan.bar

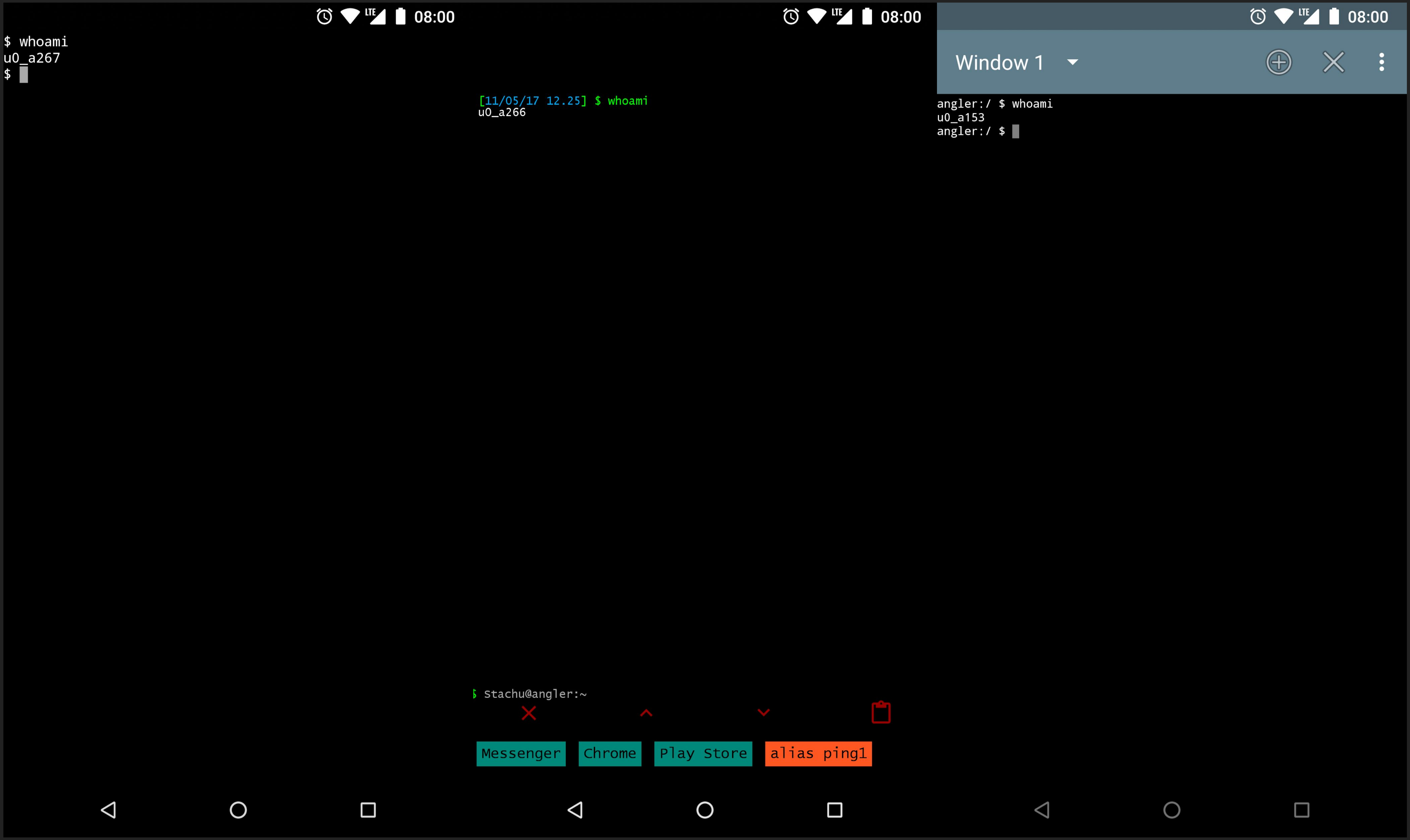
09.11.2017

## GOALS

- What is Android ?
- Android Architecture Layers
- Runtime Walkthrough

## LINUX KERNEL

- Memory and process management
- Permission-based security model
- Driver model
- Open-source



```
$ whoami
u0_a267
$
```

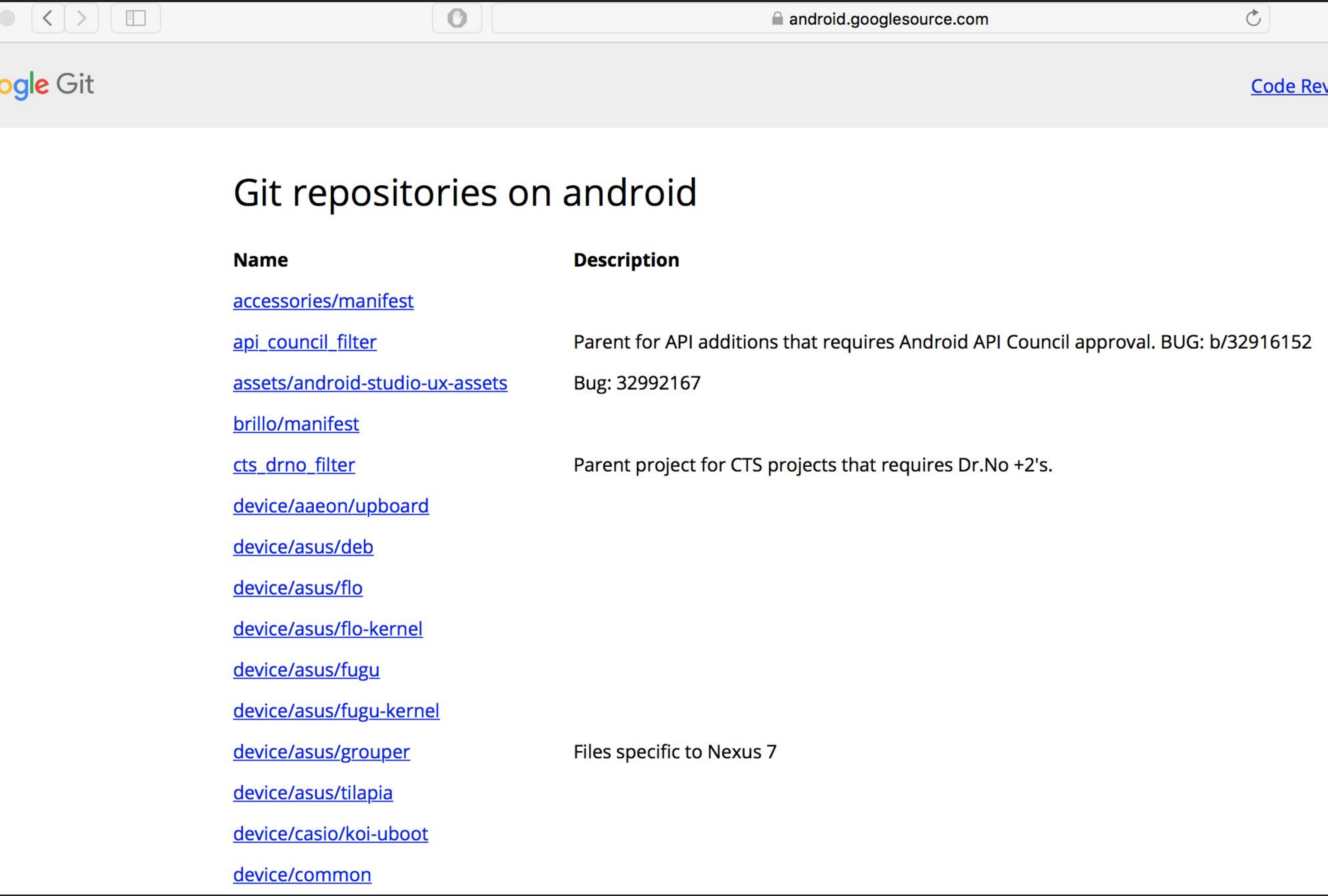
```
[11/05/17 12.25] $ whoami
u0_a266
```

```
$ Stachu@angler:~
```

```
angler:/ $ whoami
u0_a153
angler:/ $
```

## LINUX KERNEL

- Memory and process management
- Permission-based security model
- Driver model
- Open-source

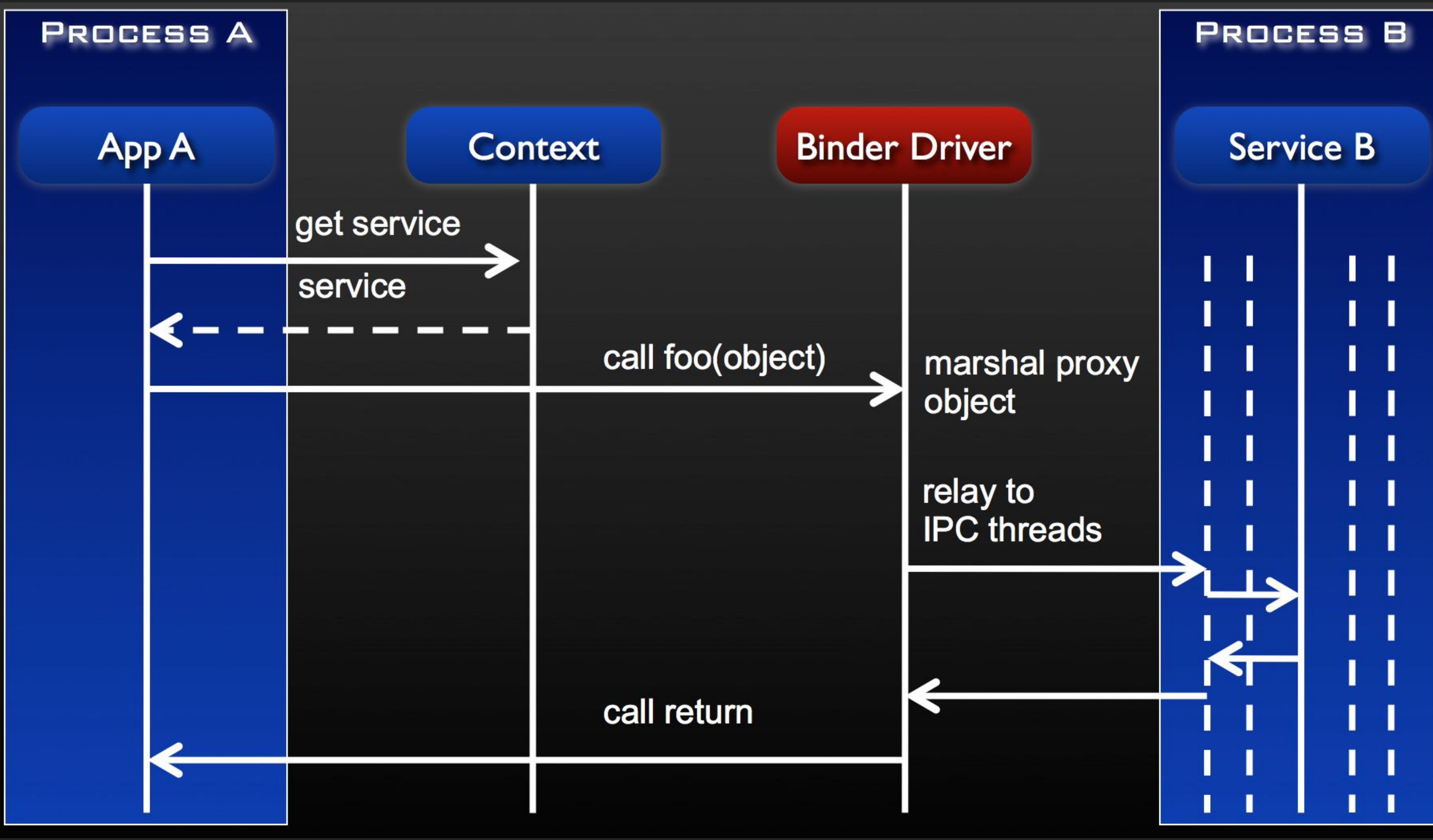


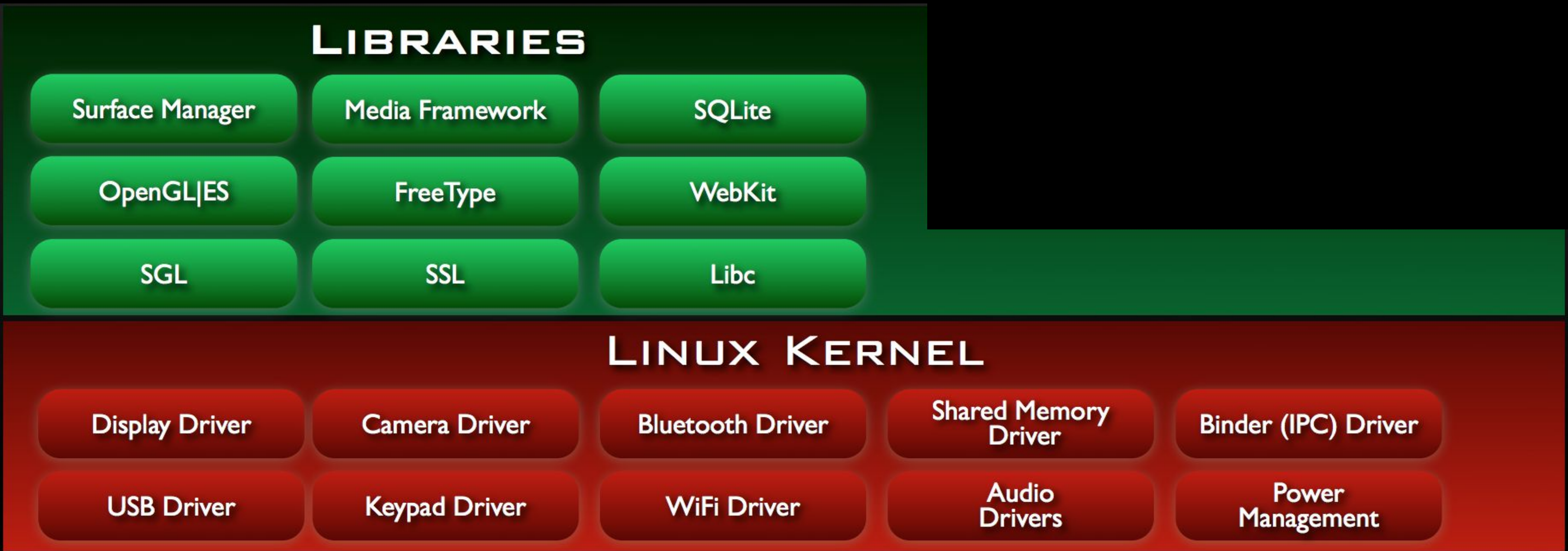
# Git repositories on android

| Name | Description |
|---|---|
| accessories/manifest | |
| api_council_filter | Parent for API additions that requires Android API Council approval. BUG: b/32916152 |
| assets/android-studio-ux-assets | Bug: 32992167 |
| brillo/manifest | |
| cts_drno_filter | Parent project for CTS projects that requires Dr.No +2's. |
| device/aaeon/upboard | |
| device/asus/deb | |
| device/asus/flo | |
| device/asus/flo-kernel | |
| device/asus/fugu | |
| device/asus/fugu-kernel | |
| device/asus/grouper | Files specific to Nexus 7 |
| device/asus/tilapia | |
| device/casio/koi-uboot | |
| device/common | |

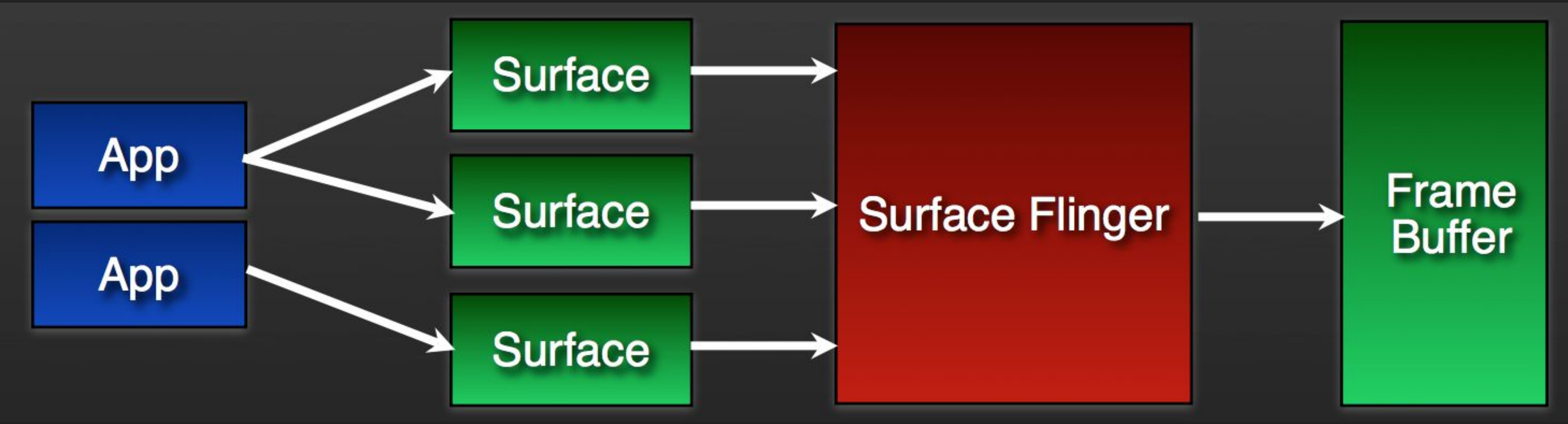
## Linux Kernel

| | | | | |
|---|---|---|---|---|
| Display Driver | Camera Driver | Bluetooth Driver | Shared Memory Driver | Binder (IPC) Driver |
| USB Driver | Keypad Driver | WiFi Driver | Audio Drivers | Power Management |

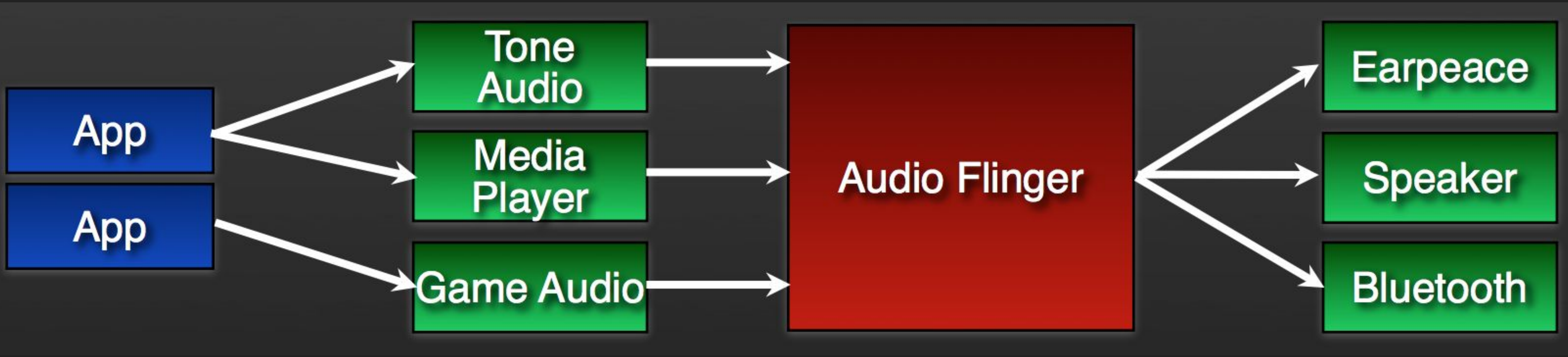

Process A
Process B
App A
Context
Binder Driver
Service B
get service
service
call foo(object)
marshal proxy object
relay to IPC threads
call return

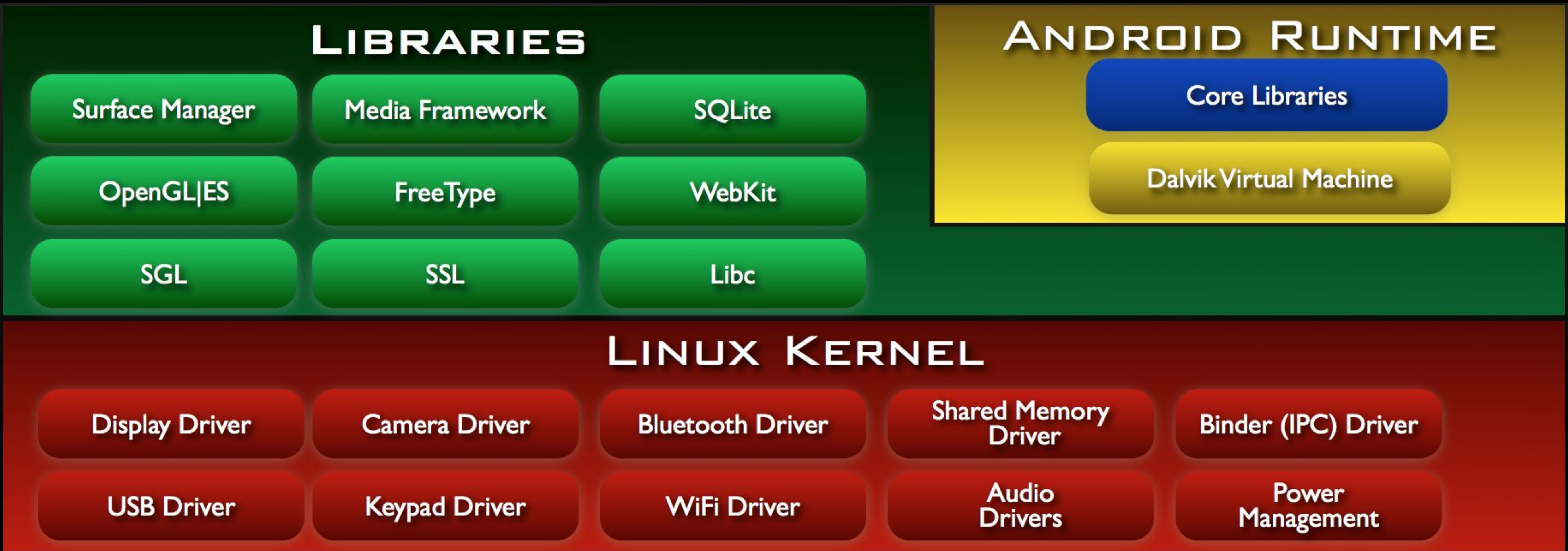

Libraries
Surface Manager
Media Framework
SQLite
OpenGL|ES
FreeType
WebKit
SGL
SSL
Libc
Linux Kernel
Display Driver
Camera Driver
Bluetooth Driver
Shared Memory Driver
Binder (IPC) Driver
USB Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

## SURFACE MANAGER



## AUDIO MANAGER




Libraries
Surface Manager
Media Framework
SQLite
OpenGL|ES
FreeType
WebKit
SGL
SSL
Libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Bluetooth Driver
Shared Memory Driver
Binder (IPC) Driver
USB Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

Kópasker
Þórshöfn
Siglufjörður
Árnes
Húsavík
Dalvík
mavík
Sauðárkrókur
Laugar
Akureyri
Reykjahlíð
Blönduós
Varmahlíð
Hvammstangi
Laugarbakki
Islandia
Reykholt
Skeiða- og
Gnúpverjahreppur
Dalvík

```
public static long sumArray(int[] arr) {
    long sum = 0;
    for (int i : arr) {
        sum += i;
    }
    return sum;
}
```

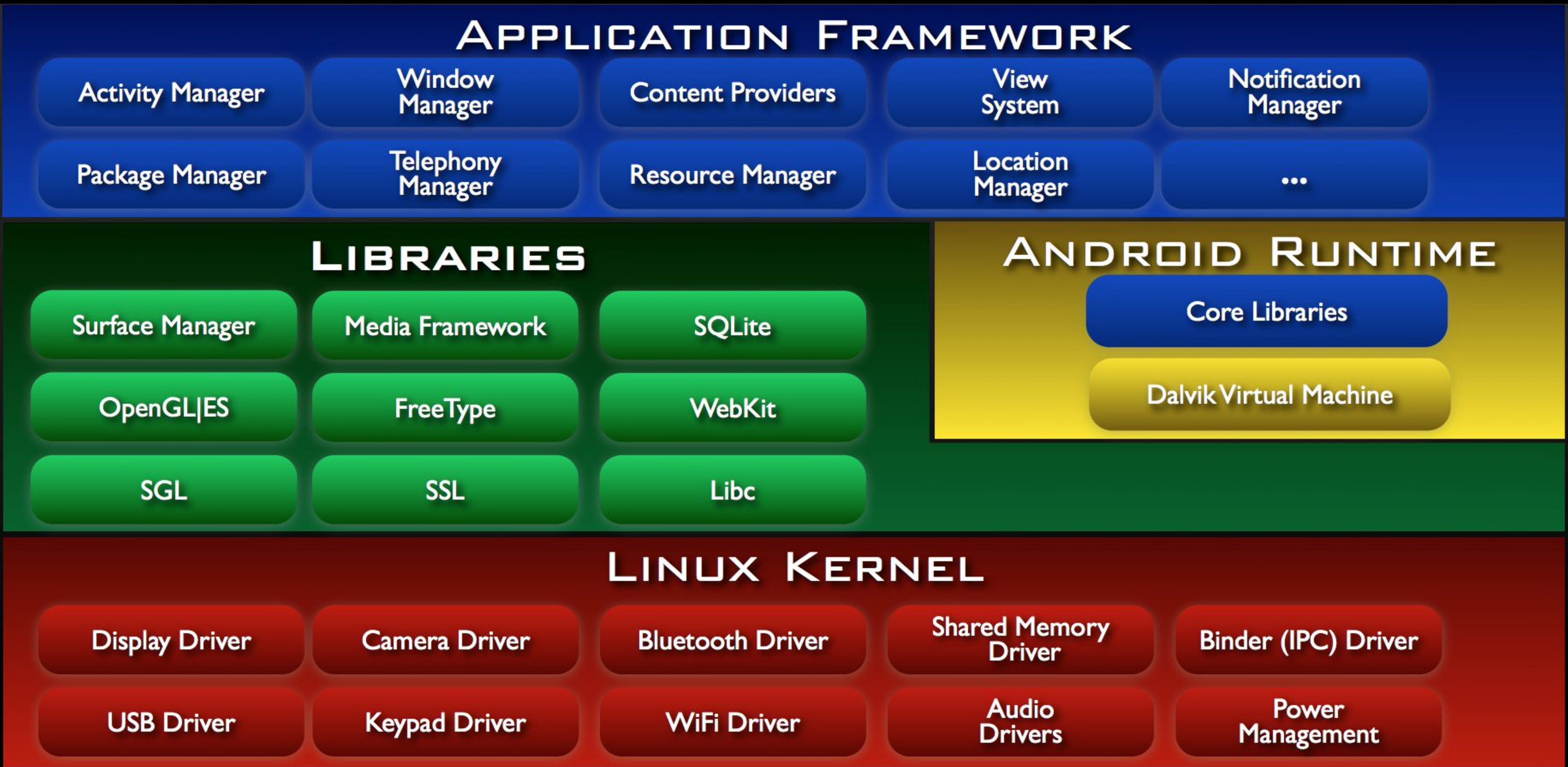
```
    0000: lconst_0
    0001: lstore_1
    0002: aload_0
    0003: astore_3
    0004: aload_3
    0005: arraylength
    0006: istore 04
    0008: iconst_0
    0009: istore 05
    000b: iload 05               // rl ws
    000d: iload 04               // rl ws
    000f: if_icmpge 0024         // rs rs
    0012: aload_3                // rl ws
    0013: iload 05               // rl ws
    0015: iaload                 // rs rs ws
    0016: istore 06              // rs wl
    0018: lload_1                // rl rl ws ws
    0019: iload 06               // rl ws
    001b: i2l                    // rs ws ws     read stack
    001c: ladd                   // rs rs rs rs ws ws
    001d: lstore_1               // rs rs wl wl       write stack
    001e: iinc 05, #+01          // rl wl
    0021: goto 000b     read local          write local
    0024: lload_1
    0025: lreturn
```

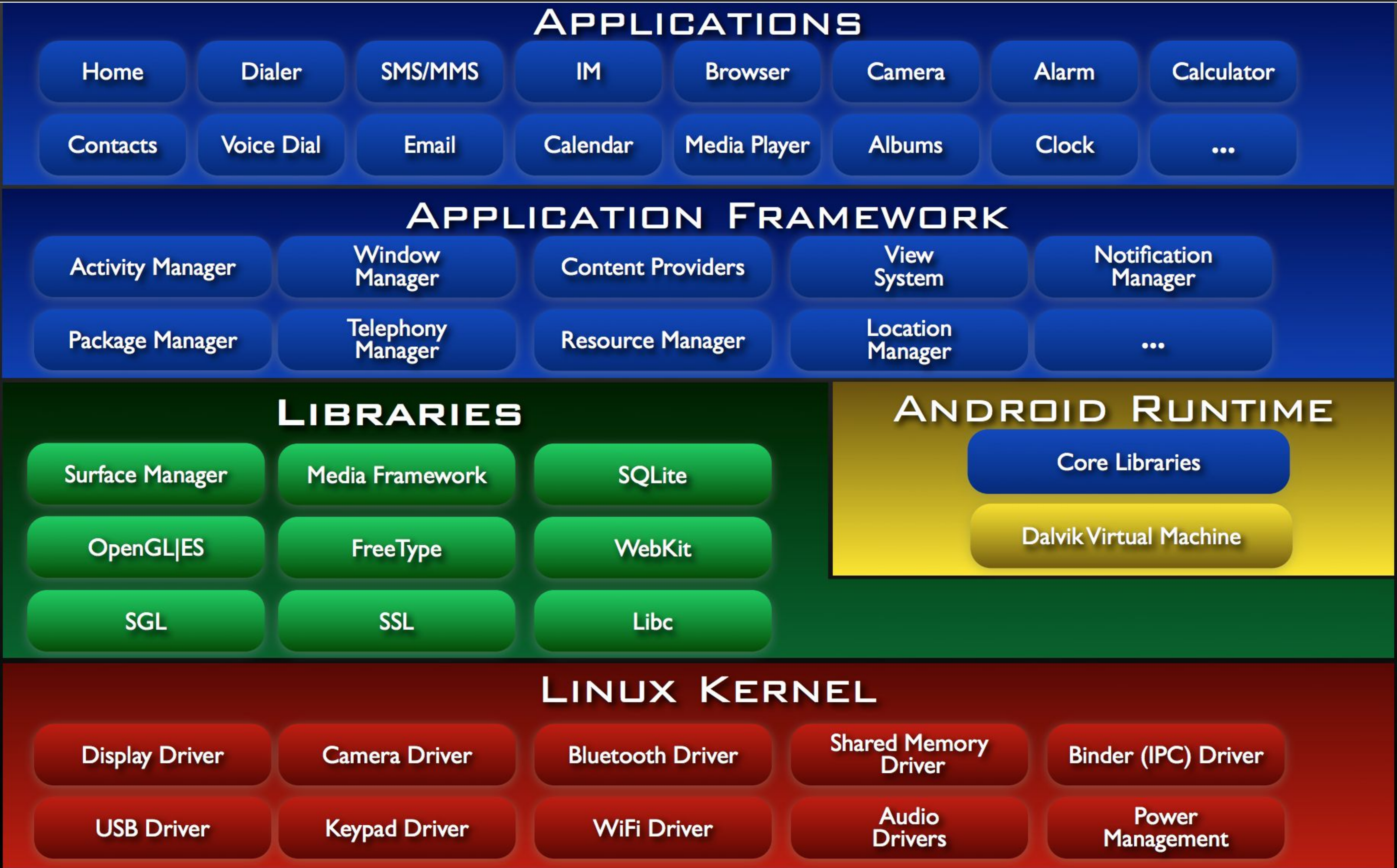
- 25 bytes
- 14 dispatches
- 45 reads
- 16 writes

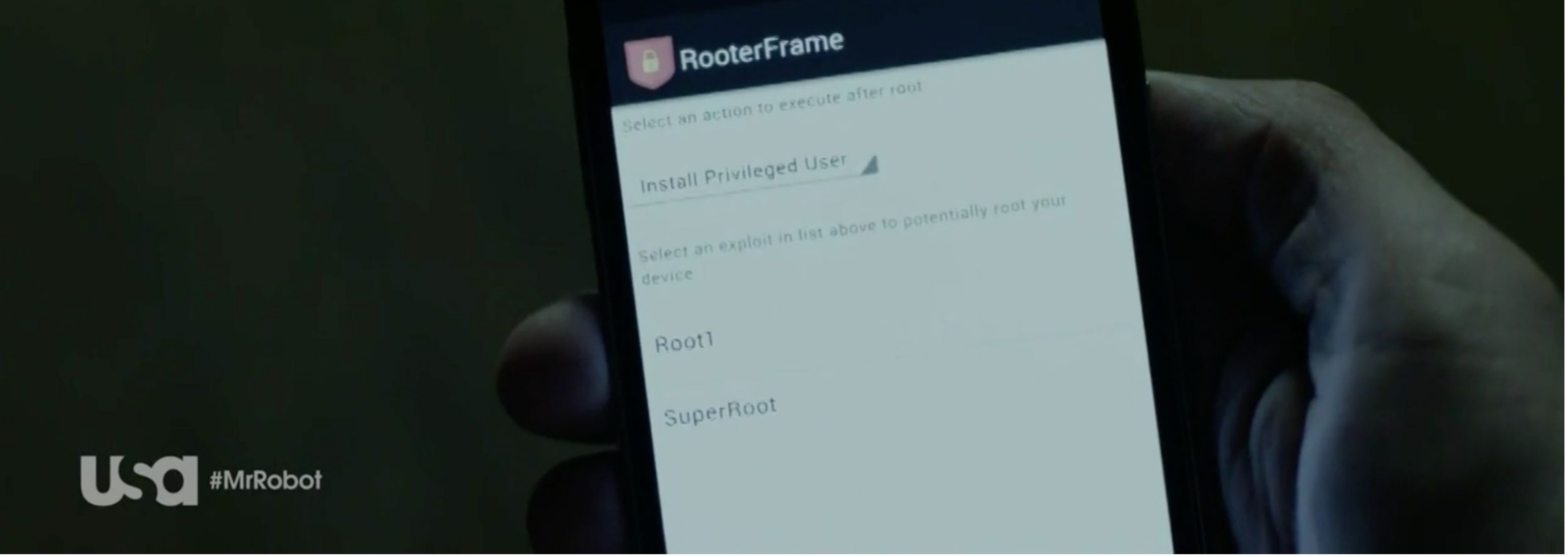
```
public static long sumArray(int[] arr) {
    long sum = 0;
    for (int i : arr) {
        sum += i;
    }
    return sum;
}
```

- 18 bytes
- 6 dispatches
- 19 reads
- 6 writes

```
0000: const-wide/16 v0, #long 0
0002: array-length v2, v8
0003: const/4 v3, #int 0
0004: move v7, v3
0005: move-wide v3, v0
0006: move v0, v7
0007: if-ge v0, v2, 0010              // r r
0009: aget v1, v8, v0                 // r r w
000b: int-to-long v5, v1              // r w w
000c: add-long/2addr v3, v5           // r r r r w w
000d: add-int/lit8 v0, v0, #int 1     // r w
000f: goto 0007
0010: return-wide v3
```


Application Framework
Activity Manager
Window Manager
Content Providers
View System
Notification Manager
Package Manager
Telephony Manager
Resource Manager
Location Manager
...
Libraries
Surface Manager
Media Framework
SQLite
OpenGL|ES
FreeType
WebKit
SGL
SSL
Libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Bluetooth Driver
Shared Memory Driver
Binder (IPC) Driver
USB Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management


Applications
Home
Dialer
SMS/MMS
IM
Browser
Camera
Alarm
Calculator
Contacts
Voice Dial
Email
Calendar
Media Player
Albums
Clock
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Notification Manager
Package Manager
Telephony Manager
Resource Manager
Location Manager
...
Libraries
Surface Manager
Media Framework
SQLite
OpenGL|ES
FreeType
WebKit
SGL
SSL
Libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Bluetooth Driver
Shared Memory Driver
Binder (IPC) Driver
USB Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management



RooterFrame

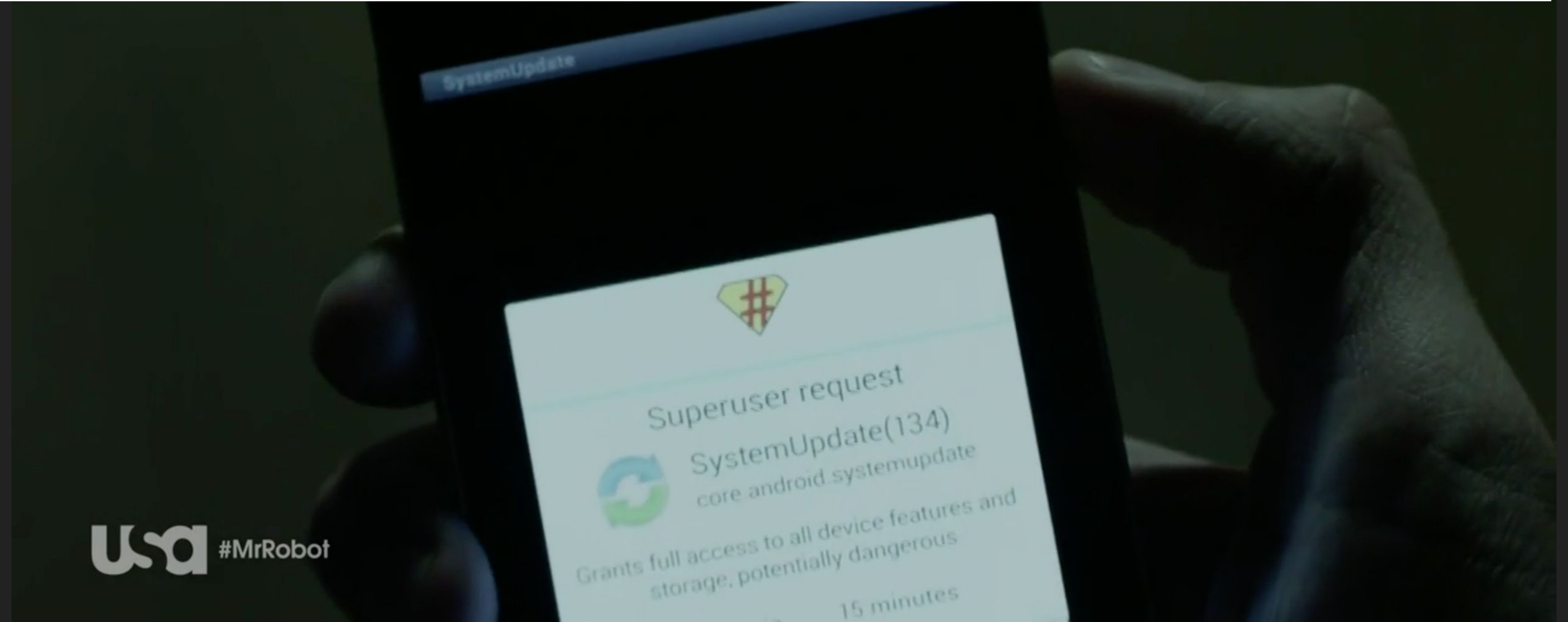
Select an action to execute after root

Install Privileged User

Select an exploit in list above to potentially root your device

Root1

SuperRoot



SystemUpdate

Superuser request

SystemUpdate(134)

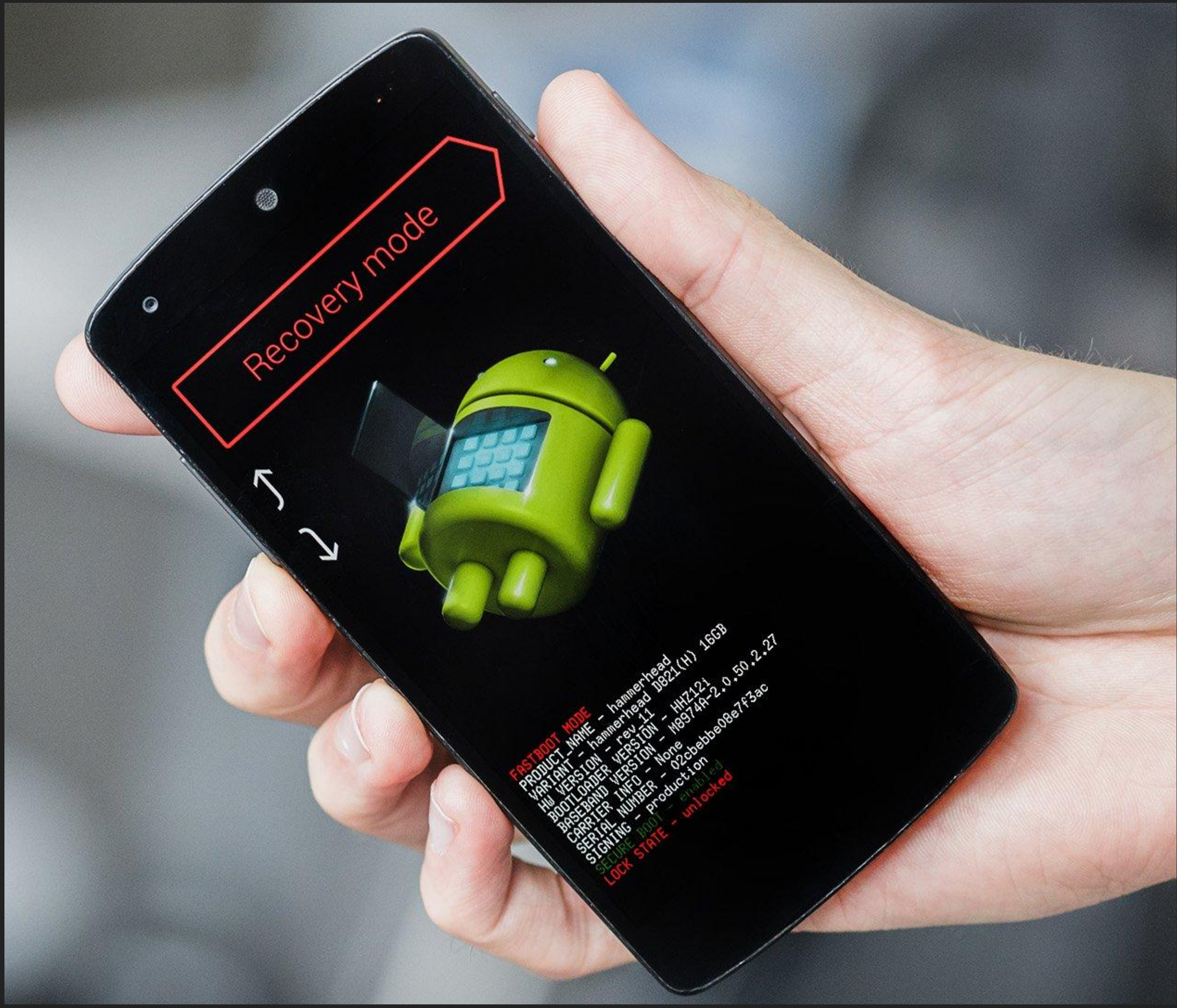
core.android.systemupdate

Grants full access to all device features and storage, potentially dangerous

15 minutes

## RUNTIME WALKTHROUGH

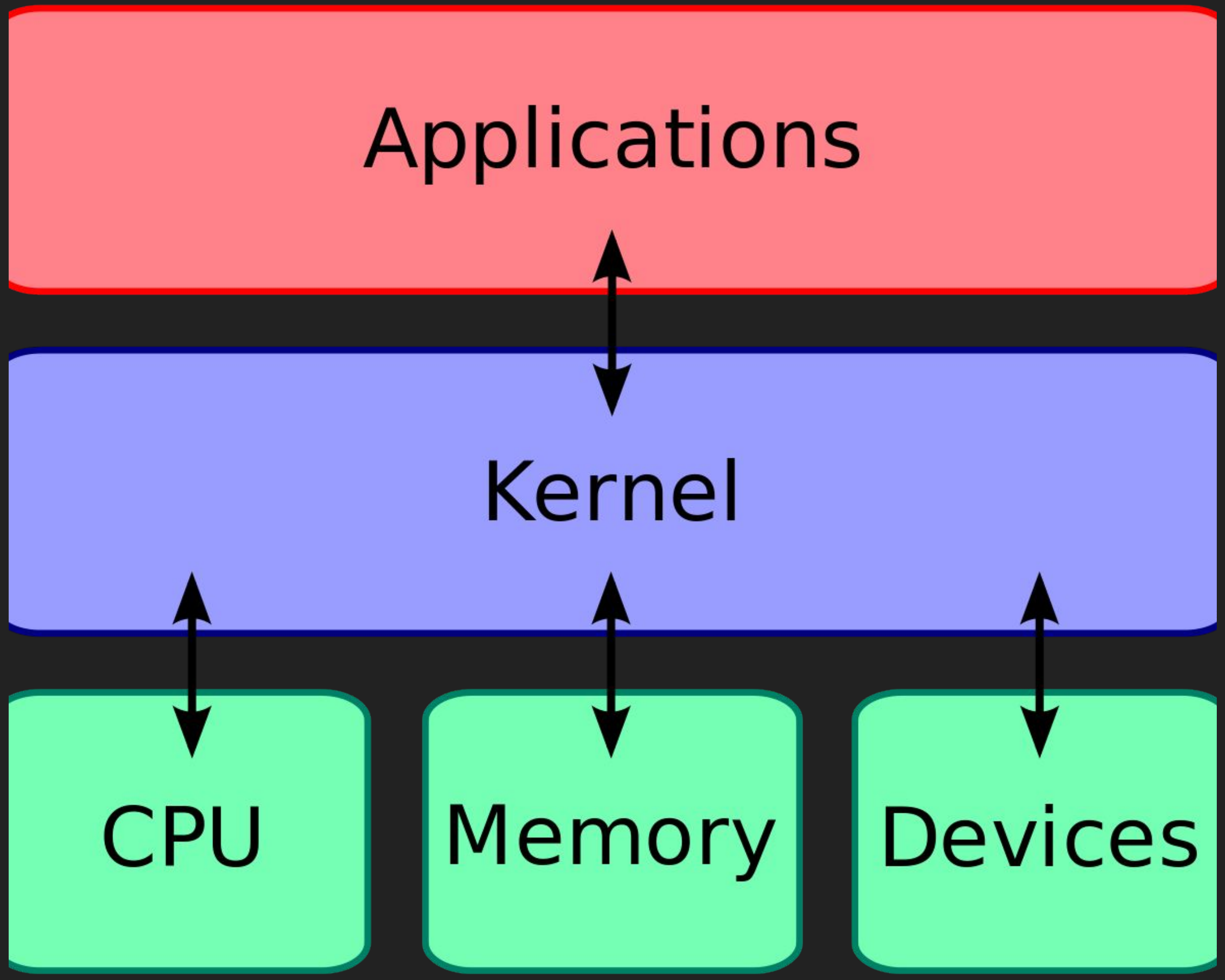
- Bootloader



```
FASTBOOT MODE
PRODUCT_NAME - hammerhead
VARIANT - hammerhead D821(H) 16GB
HW VERSION - rev_11
BOOTLOADER VERSION - HHZ12i
BASEBAND VERSION - M8974A-2.0.50.2.27
CARRIER INFO - None
SERIAL NUMBER - 02cbebbe08e7f3ac
SIGNING - production
SECURE BOOT - enabled
LOCK STATE - unlocked
```

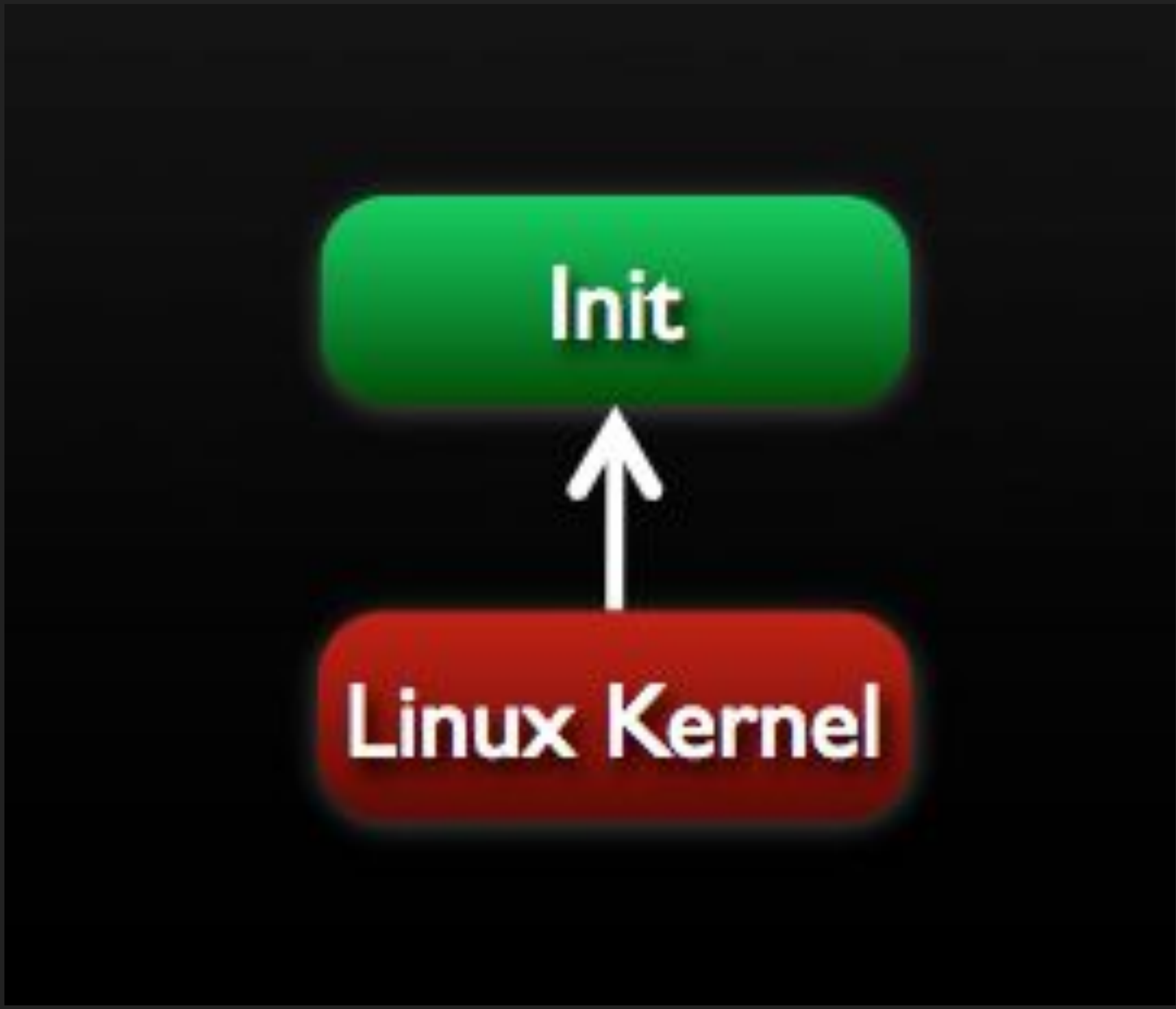
## RUNTIME WALKTHROUGH

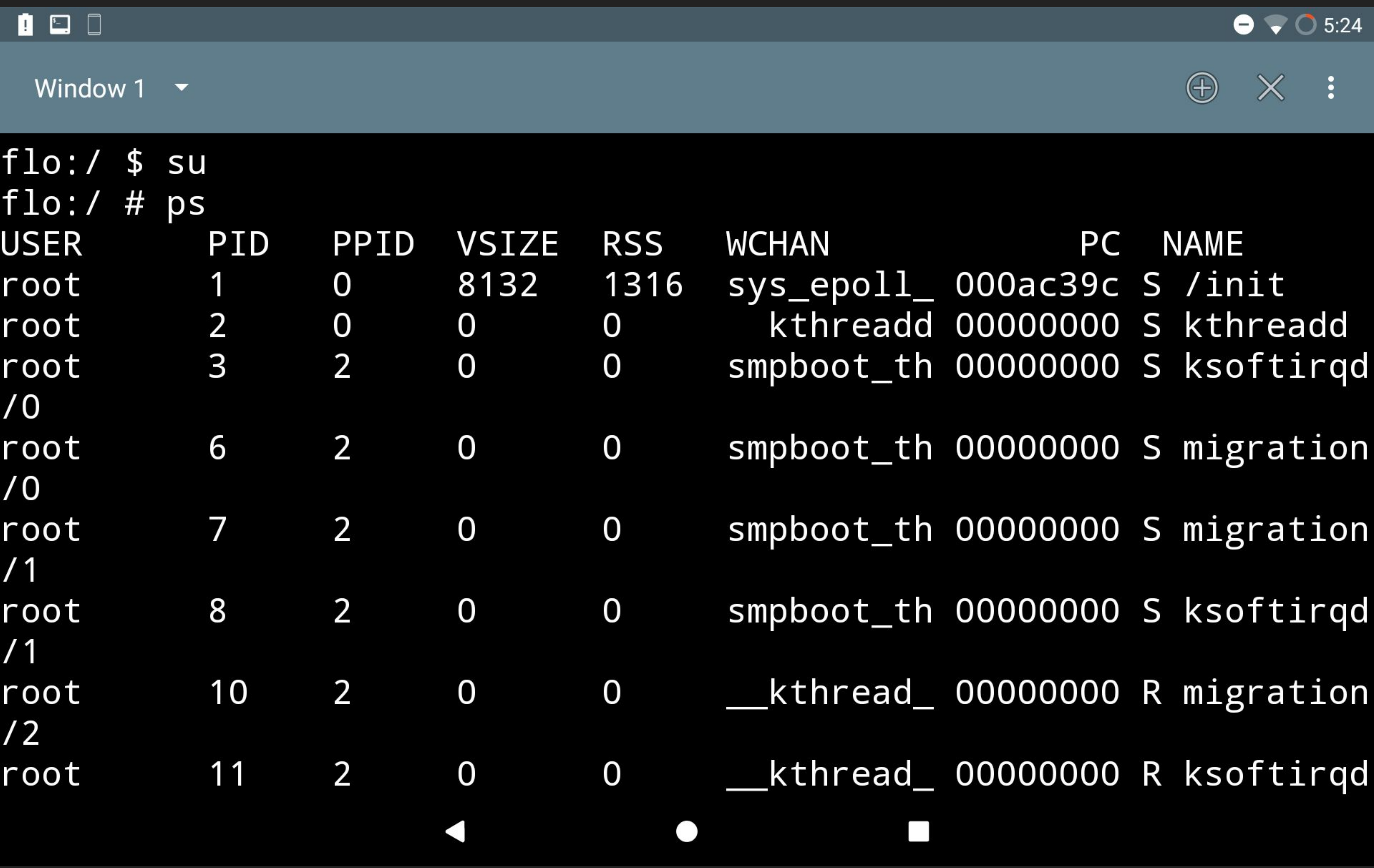
- Bootloader
- Linux Kernel



## RUNTIME WALKTHROUGH

- Bootloader
- Linux Kernel
- init



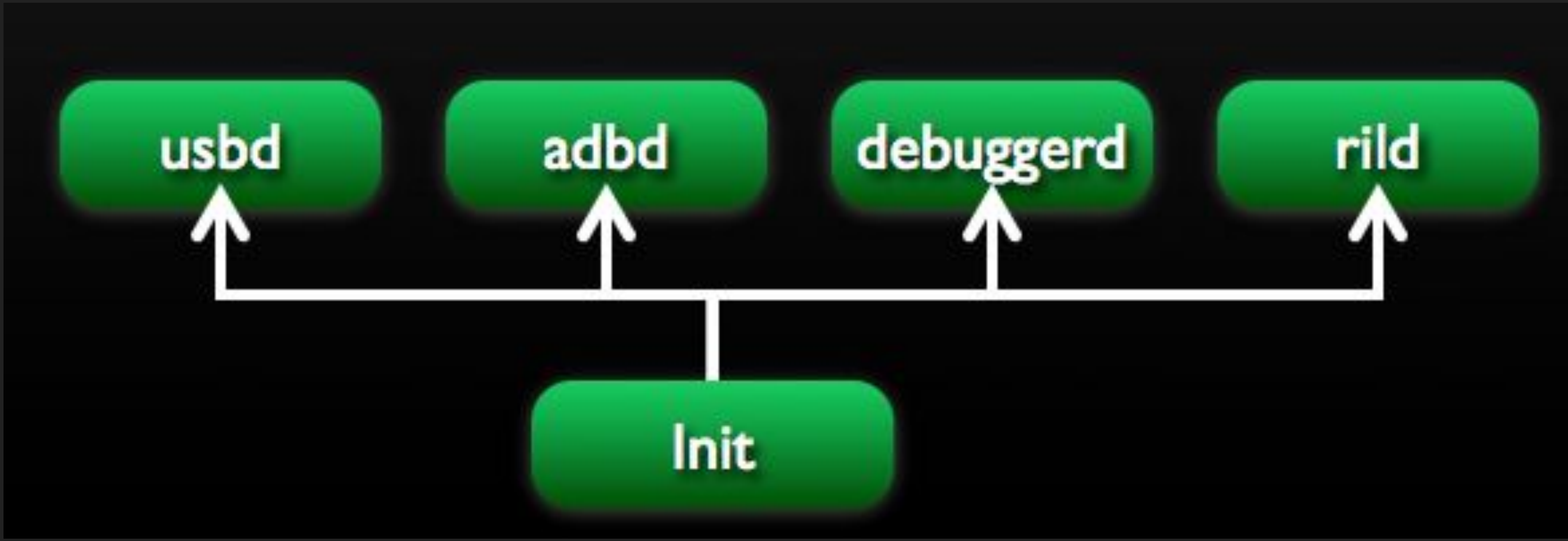


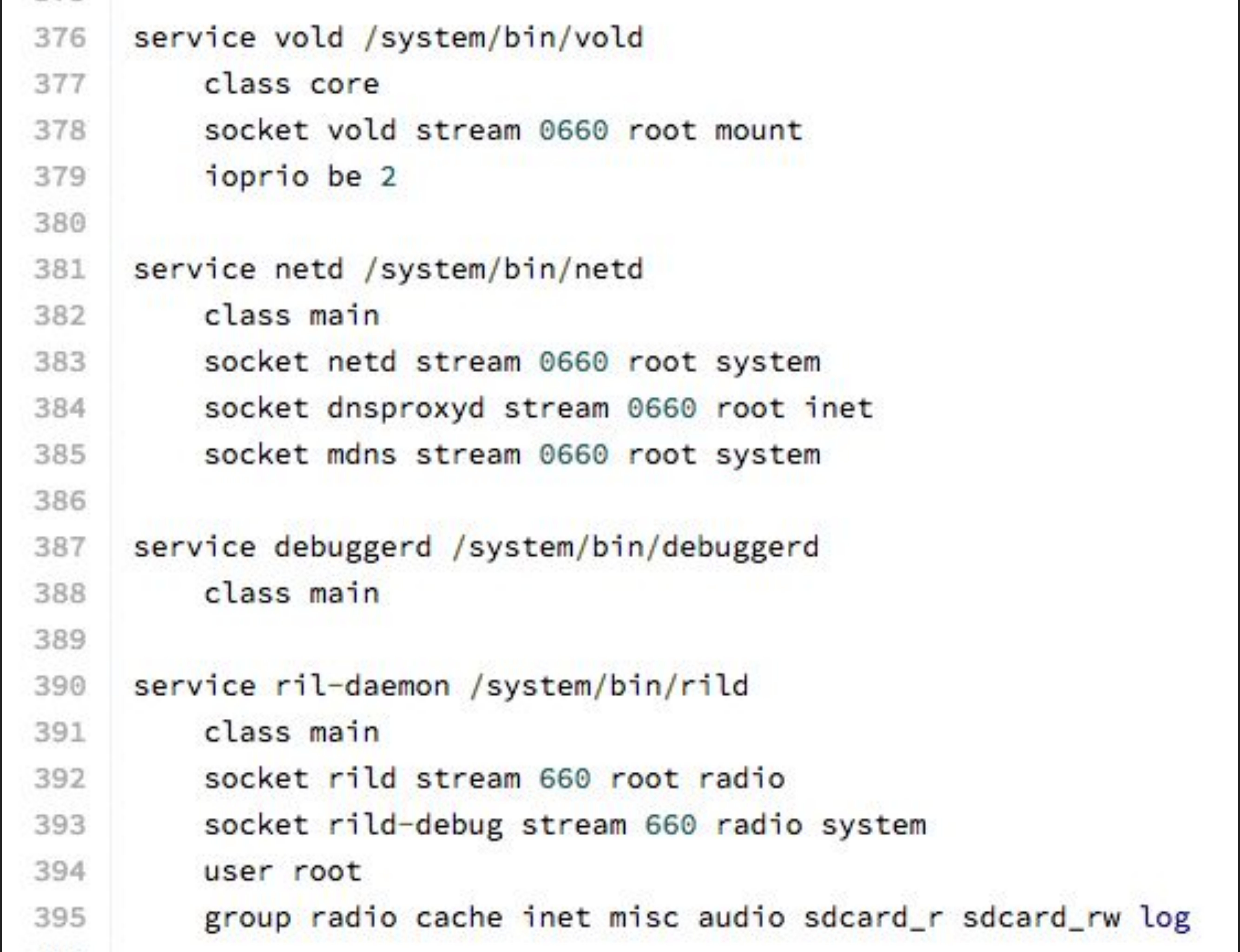
Window 1

```
flo:/ $ su
flo:/ # ps
USER      PID   PPID  VSIZE  RSS   WCHAN            PC  NAME
root      1     0     8132   1316  sys_epoll_ 000ac39c S /init
root      2     0     0      0       kthreadd 00000000 S kthreadd
root      3     2     0      0     smpboot_th 00000000 S ksoftirqd
/0
root      6     2     0      0     smpboot_th 00000000 S migration
/0
root      7     2     0      0     smpboot_th 00000000 S migration
/1
root      8     2     0      0     smpboot_th 00000000 S ksoftirqd
/1
root      10    2     0      0     __kthread_ 00000000 R migration
/2
root      11    2     0      0     __kthread_ 00000000 R ksoftirqd
```

## RUNTIME WALKTHROUGH

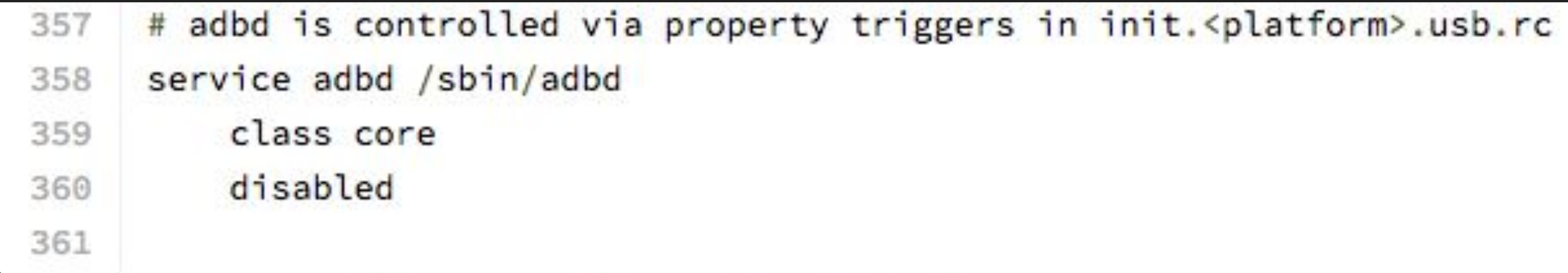
- Bootloader
- Linux Kernel
- init
- daemons



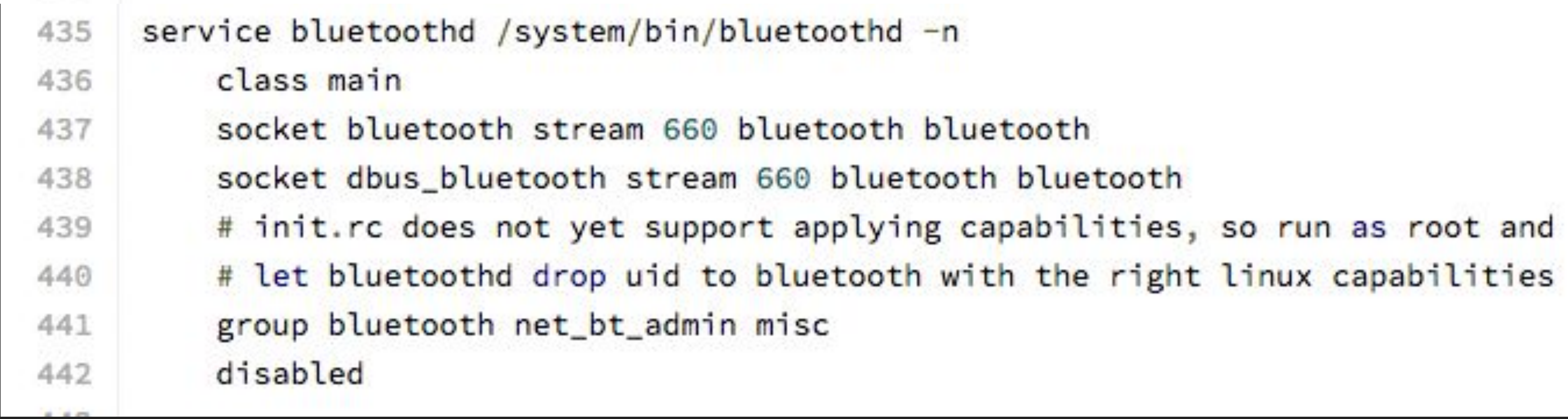


```
service vold /system/bin/vold
    class core
    socket vold stream 0660 root mount
    ioprio be 2

service netd /system/bin/netd
    class main
    socket netd stream 0660 root system
    socket dnsproxyd stream 0660 root inet
    socket mdns stream 0660 root system

service debuggerd /system/bin/debuggerd
    class main

service ril-daemon /system/bin/rild
    class main
    socket rild stream 660 root radio
    socket rild-debug stream 660 radio system
    user root
    group radio cache inet misc audio sdcard_r sdcard_rw log
```

## INIT.RC



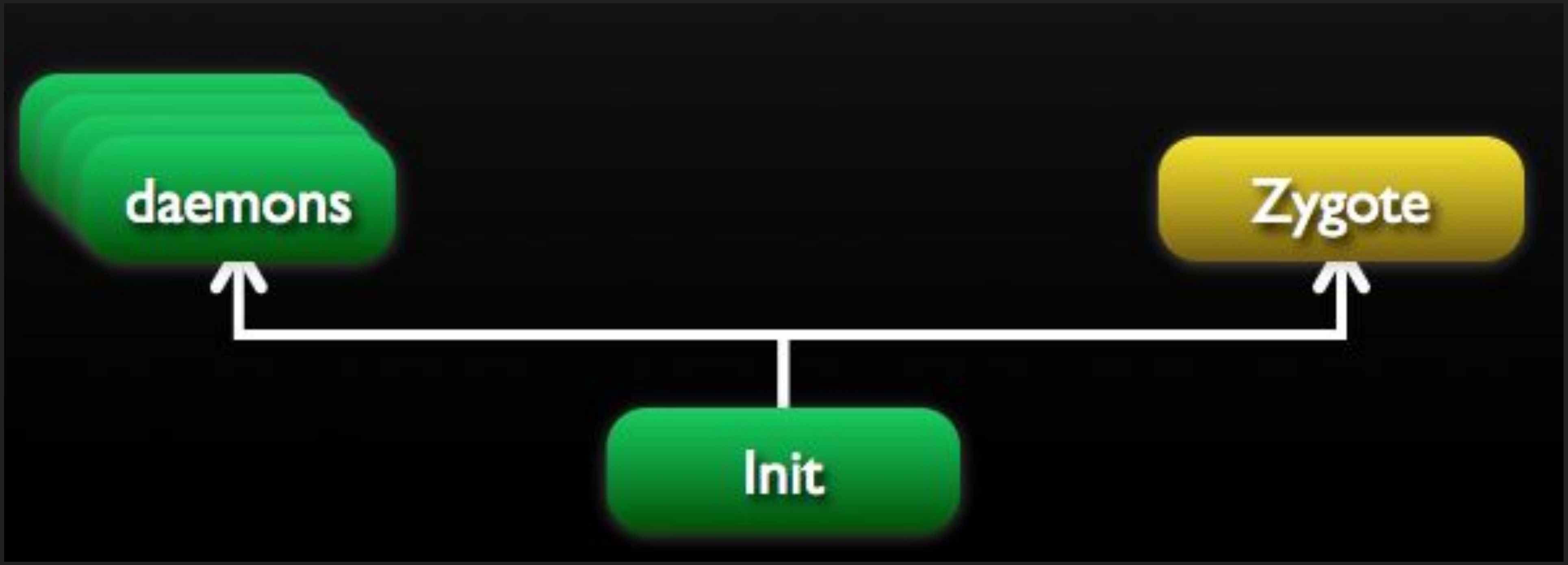
```
# adbd is controlled via property triggers in init.<platform>.usb.rc
service adbd /sbin/adbd
    class core
    disabled

```



```
service bluetoothd /system/bin/bluetoothd -n
    class main
    socket bluetooth stream 660 bluetooth bluetooth
    socket dbus_bluetooth stream 660 bluetooth bluetooth
    # init.rc does not yet support applying capabilities, so run as root and
    # let bluetoothd drop uid to bluetooth with the right linux capabilities
    group bluetooth net_bt_admin misc
    disabled
```

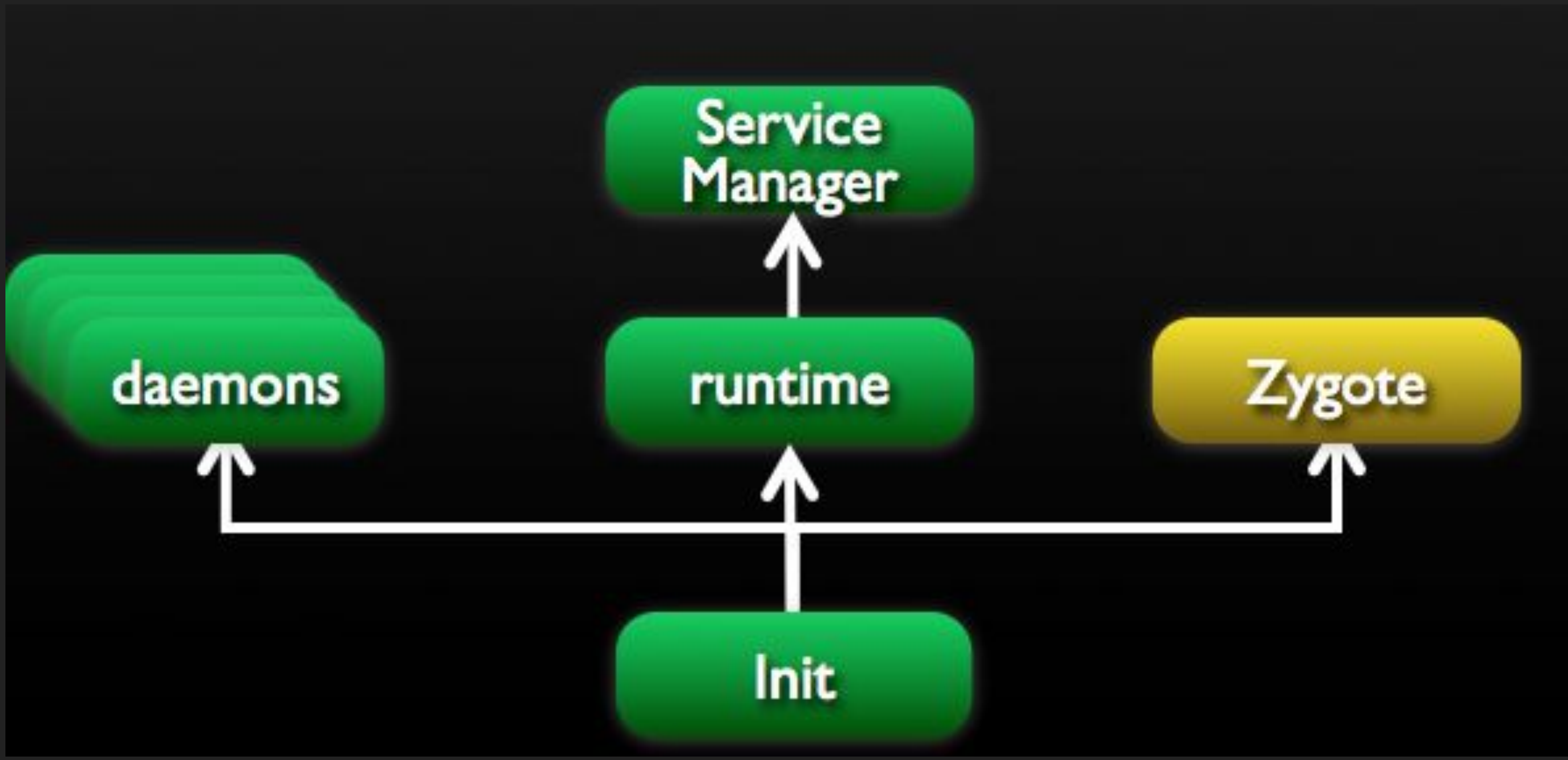
## RUNTIME WALKTHROUGH

- Bootloader
- Linux Kernel
- init
- daemons
- Zygote

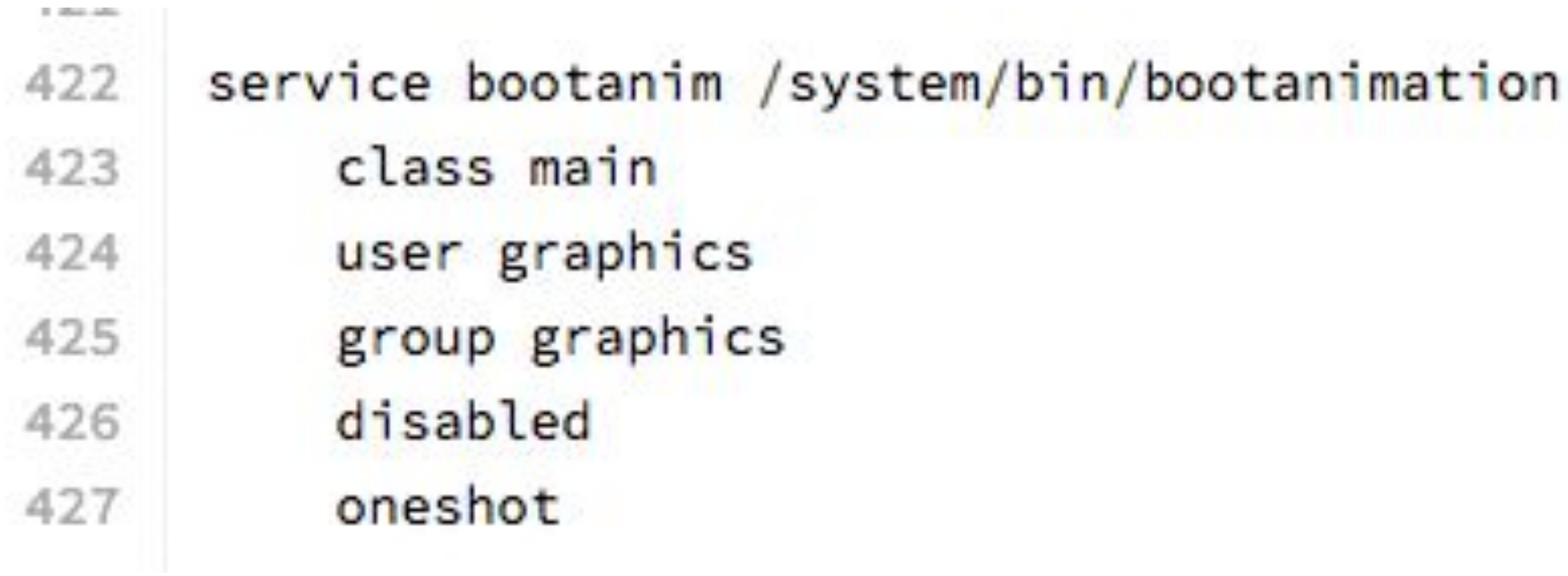


## RUNTIME WALKTHROUGH

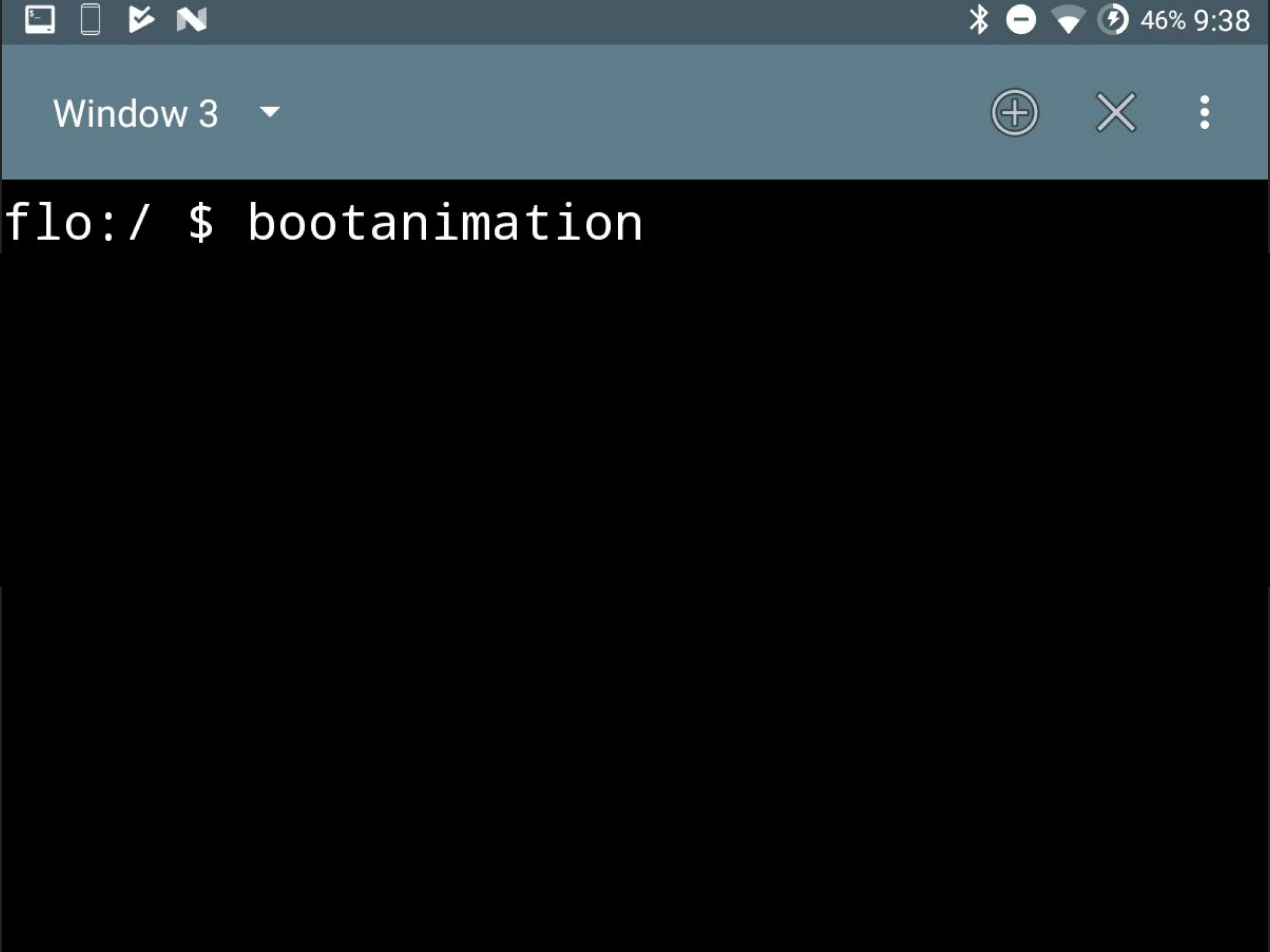
- Bootloader
- Linux Kernel
- init
- daemons
- Zygote
- Runtime (Service Managers)



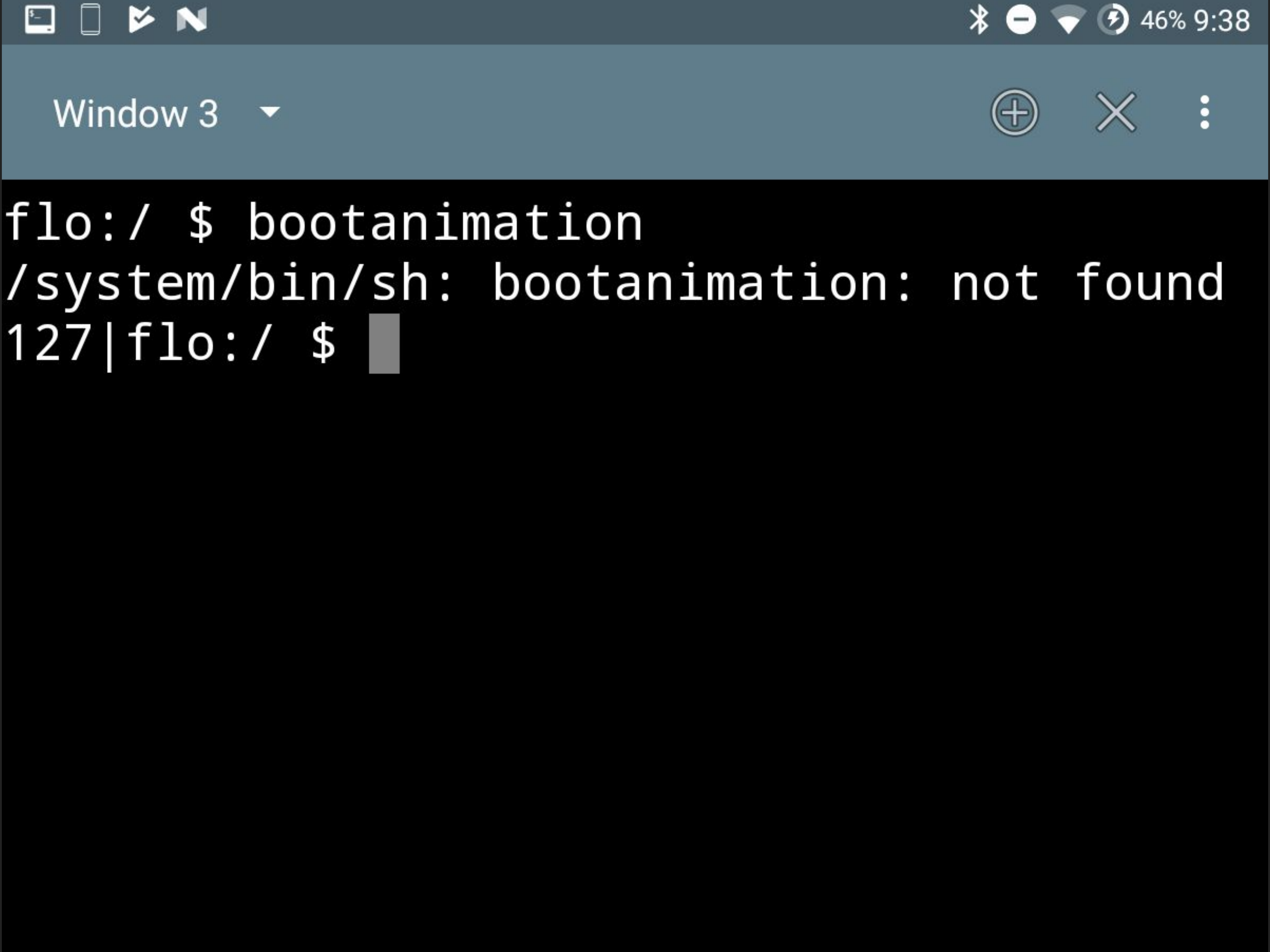
## INIT.RC



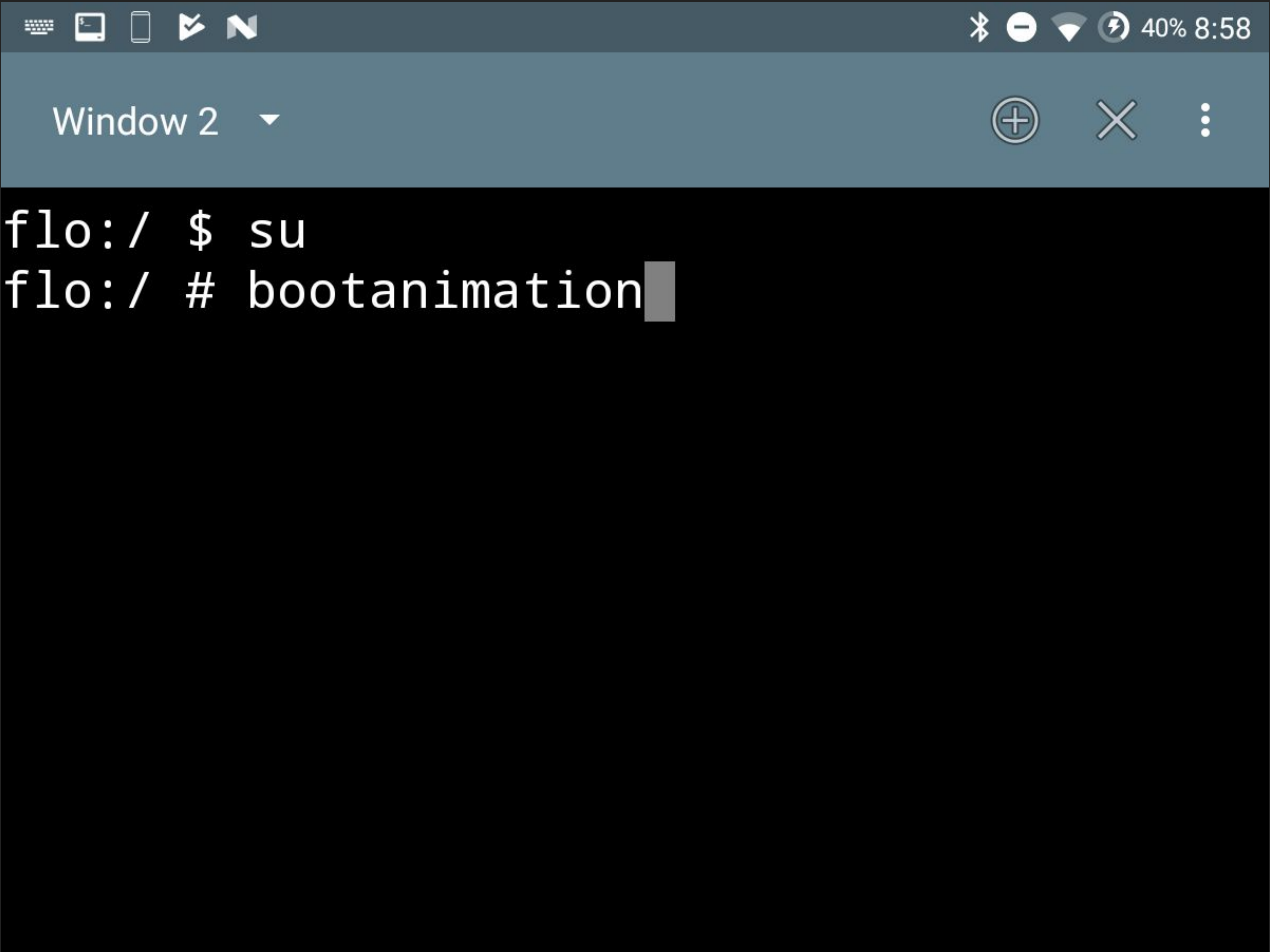
```
service bootanim /system/bin/bootanimation
    class main
    user graphics
    group graphics
    disabled
    oneshot
```



```
flo:/ $ bootanimation
```


Window 3

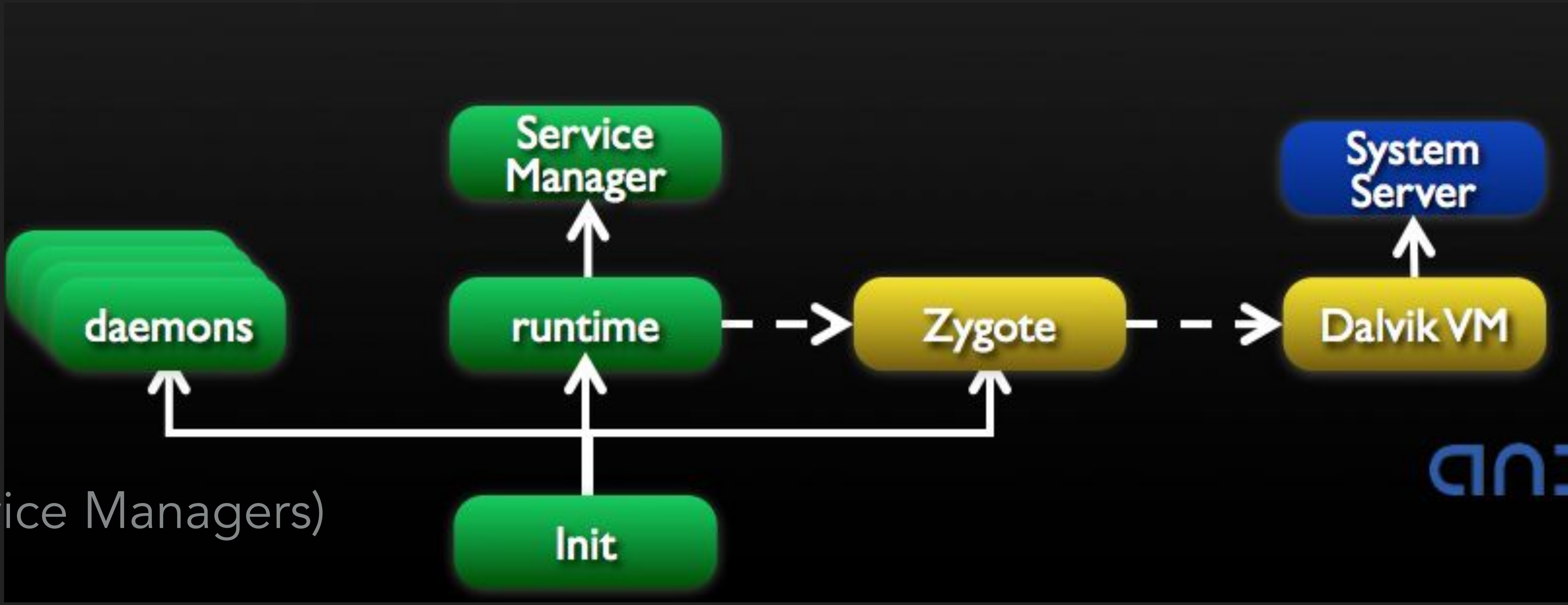
```
flo:/ $ bootanimation
/system/bin/sh: bootanimation: not found
127|flo:/ $
```



```
flo:/ $ su
flo:/ # bootanimation
```

## RUNTIME WALKTHROUGH

- Bootloader
- Linux Kernel
- init
- daemons
- Zygote
- Runtime (Service Managers)
- Dalvik VM
- System Server



```
service zygote /system/bin/app_process -Xzygote /system/bin --zygote --start-system-server
    class main
    socket zygote stream 660 root system
    onrestart write /sys/android_power/request_state wake
    onrestart write /sys/power/state on
    onrestart restart media
    onrestart restart netd
```

frameworks/base/cmds/app_process/app_main.cpp::main

```
if (strcmp(arg, "--zygote") == 0) {
        zygote = true;
        niceName = ZYGOTE_NICE_NAME;
        startSystemServer = true;
}
...
if (zygote) {
    runtime.start("com.android.internal.os.ZygoteInit", args, zygote);
```

frameworks/base/core/jni/AndroidRuntime.cpp::start

```
void AndroidRuntime::start(const char* className, const Vector<String8>& options){
...
        char* slashClassName = toSlashClassName(className);
        jclass startClass = env->FindClass(slashClassName);
        jmethodID startMeth = env->GetStaticMethodID(startClass, "main","([Ljava/lang/String;)V");
        env->CallStaticVoidMethod(startClass, startMeth, strArray);
...
}
```

frameworks/base/core/java/com/android/internal/os/ZygoteInit.java#main

```
preload();
if (argv[1].equals("start-system-server")) {
        startSystemServer();
runSelectLoop(abiList);
```

frameworks/base/core/java/com/android/internal/os/ZygoteInit.java#startSystemServer

```
String args[] = {
        "--setuid=1000",
        "--setgid=1000",
        "--setgroups=1001,1002,1003,1004,1005,1006,1007,1008,1009,1010,3001,3002,3003",
        "--capabilities=130104352,130104352",
        "--runtime-init",
        "--nice-name=system_server",
        "com.android.server.SystemServer",
    };
    ...
    try {
        parsedArgs = new ZygoteConnection.Arguments(args);
        ...
        /* Request to fork the system server process */
        pid = Zygote.forkSystemServer(
                parsedArgs.uid, parsedArgs.gid,
                parsedArgs.gids, debugFlags, null,
                parsedArgs.permittedCapabilities,
                parsedArgs.effectiveCapabilities);
```

```
 */
public static int forkSystemServer(int uid, int gid, int[] gids, int debugFlags,
        int[][] rlimits, long permittedCapabilities, long effectiveCapabilities) {
    VM_HOOKS.preFork();
    int pid = nativeForkSystemServer(
            uid, gid, gids, debugFlags, rlimits, permittedCapabilities, effectiveCapabilities);
    // Enable tracing as soon as we enter the system_server.
    if (pid == 0) {
        Trace.setTracingEnabled(true);
    }
    VM_HOOKS.postForkCommon();
    return pid;
}
```

/frameworks/base/services/java/com/android/server/SystemServer.java

```
public static void main(String[] args) {
    // The system server has to run all of the time, so it needs to be
    // as efficient as possible with its memory usage.
    VMRuntime.getRuntime().setTargetHeapUtilization(0.8f);

    System.loadLibrary("android_servers");

    init1(args);
}
```

frameworks/base/services/jni/com_a[illegible]id_server_SystemServer.cpp

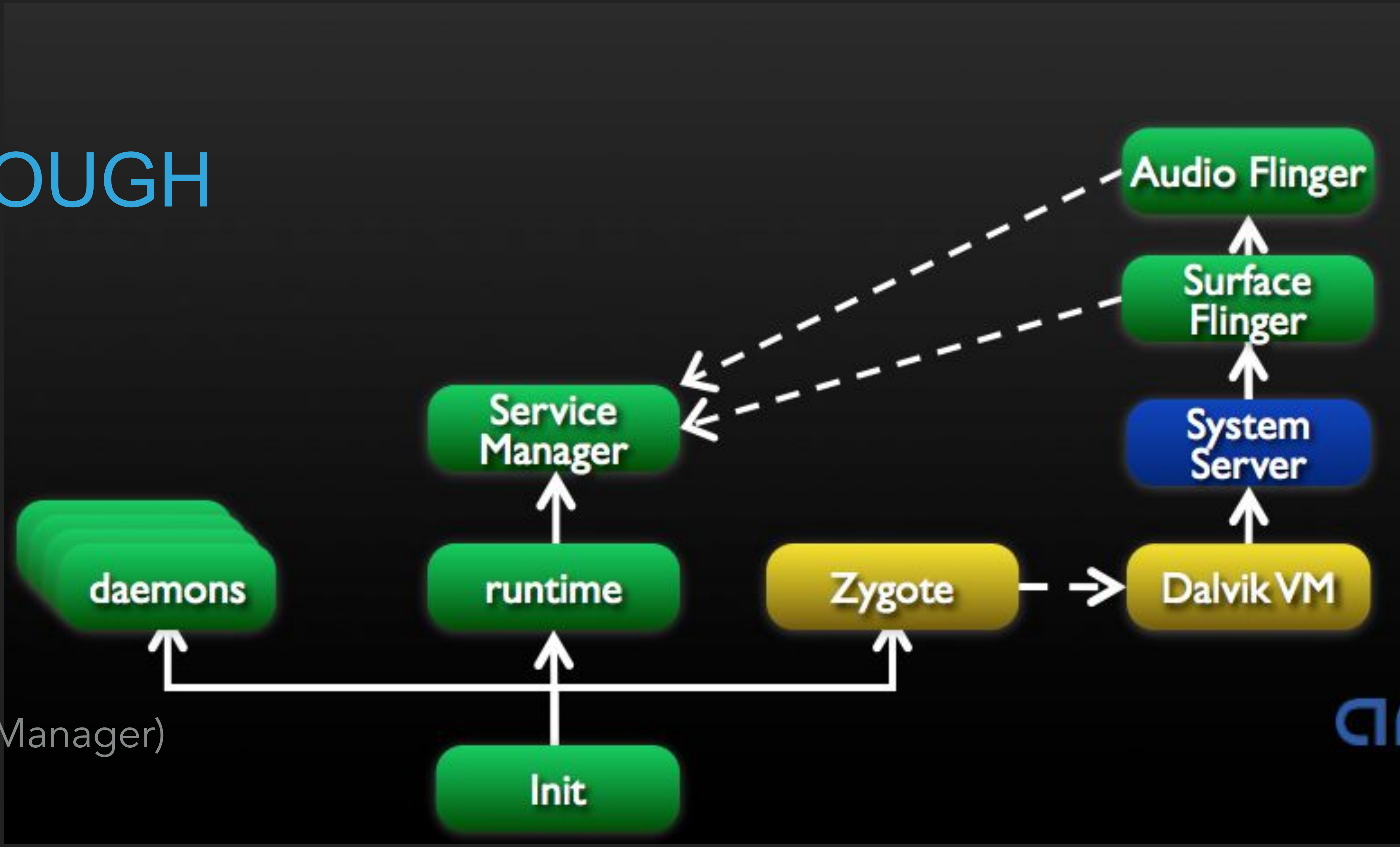
```
static void android_server_SystemServer_init1(JNIEnv* env, jobject clazz){
  system_init();
}
```

frameworks/base/cmds/system_server/library/system_init.cpp

```
extern "C" status_t system_init(){
      SurfaceFlinger::instantiate();
      SensorService::instantiate();
      jclass clazz = env->FindClass("com/android/server/SystemServer");
      if (clazz == NULL) {
            return UNKNOWN_ERROR;
      }
      jmethodID methodId = env->GetStaticMethodID(clazz, "init2", "()V");
      if (methodId == NULL) {
            return UNKNOWN_ERROR;
      }
      env->CallStaticVoidMethod(clazz, methodId);
```

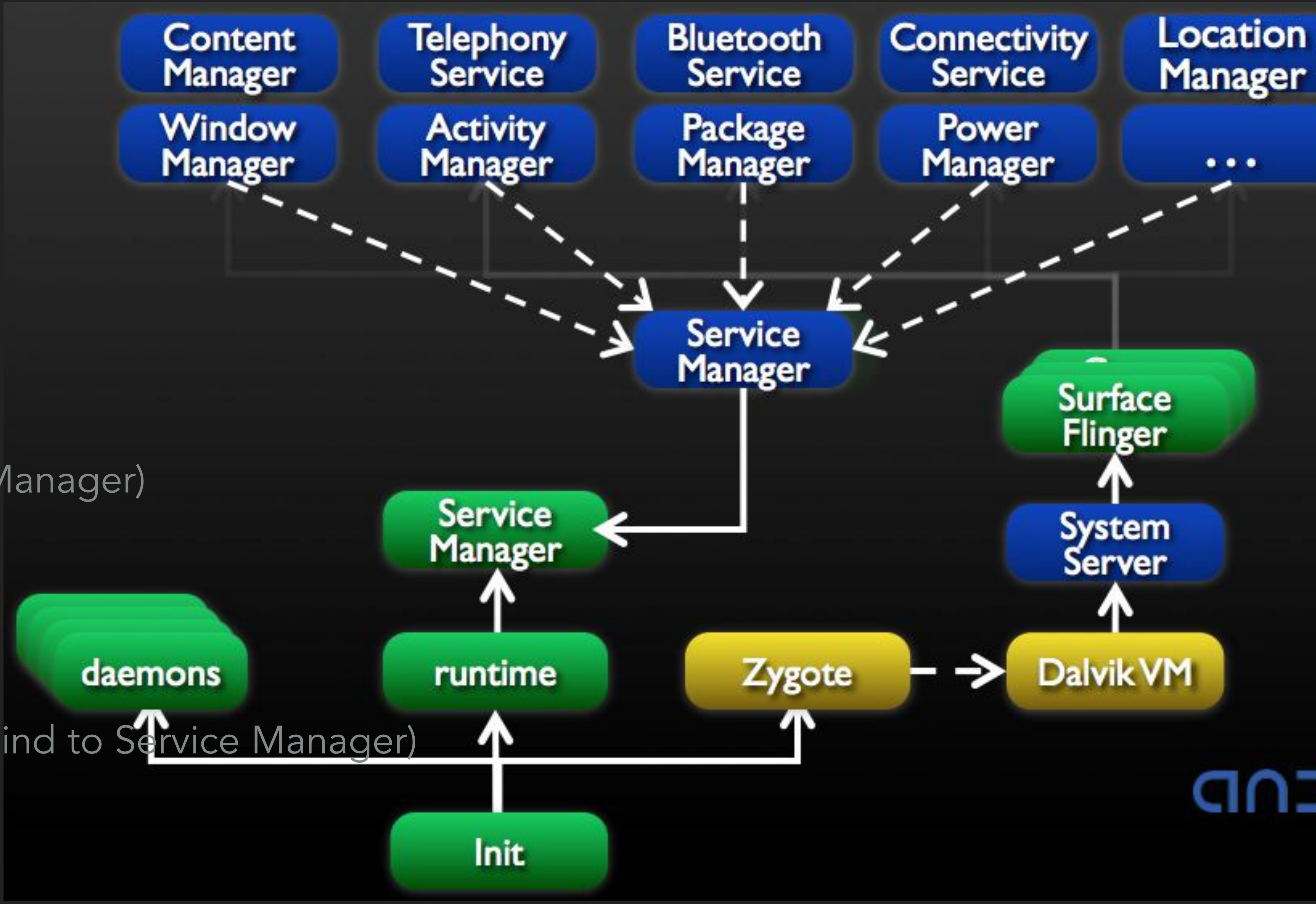
## RUNTIME WALKTHROUGH

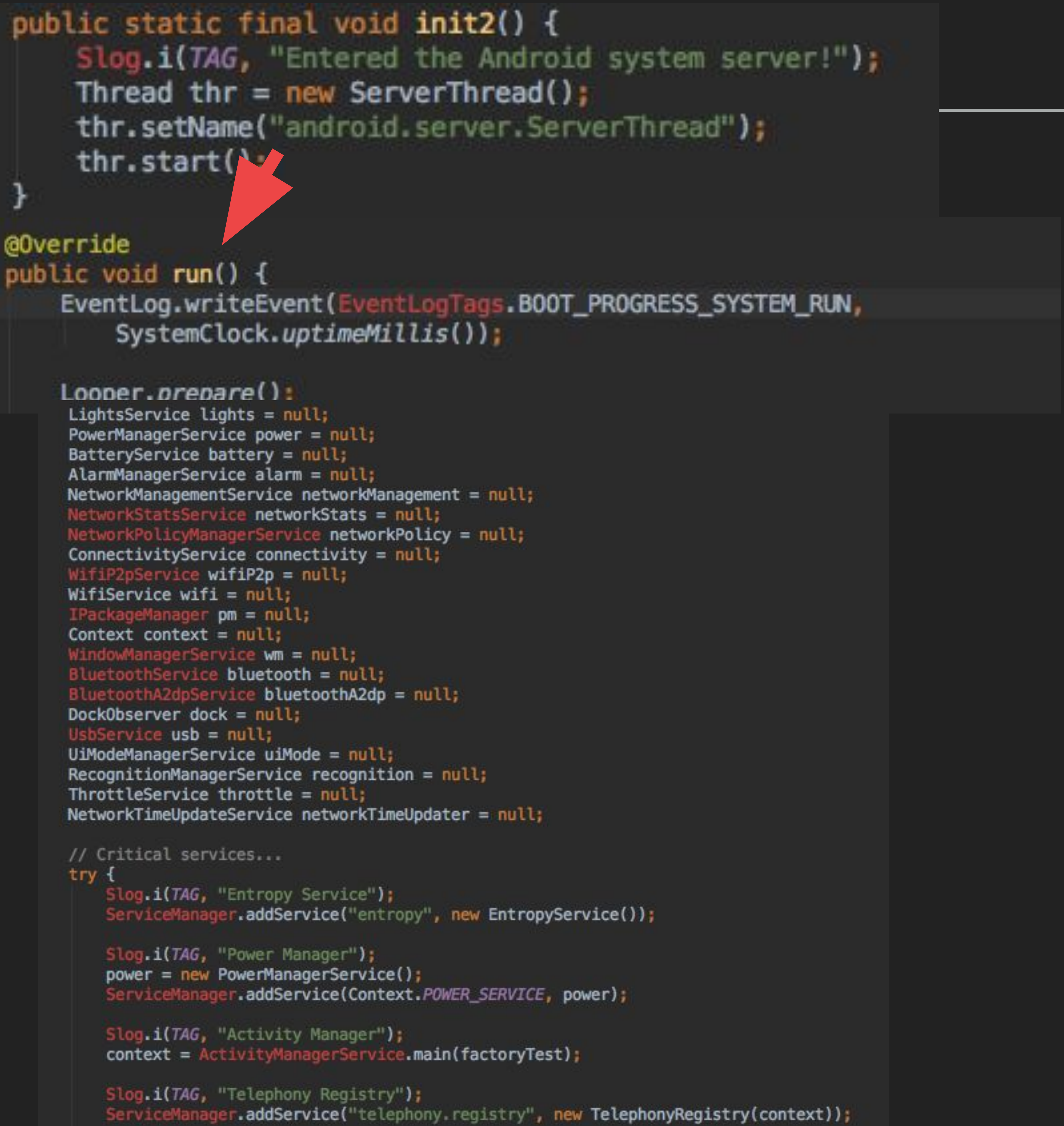
- Bootloader
- Linux Kernel
- init
- daemons
- Zygote
- Runtime (Service Manager)
- Dalvik VM
- System Server
- System Services (bind to Service Manager)



## RUNTIME WALKTHROUGH

- Bootloader
- Linux Kernel
- init
- daemons
- Zygote
- Runtime (Service Manager)
- Dalvik VM
- System Server
- System Services (bind to Service Manager)

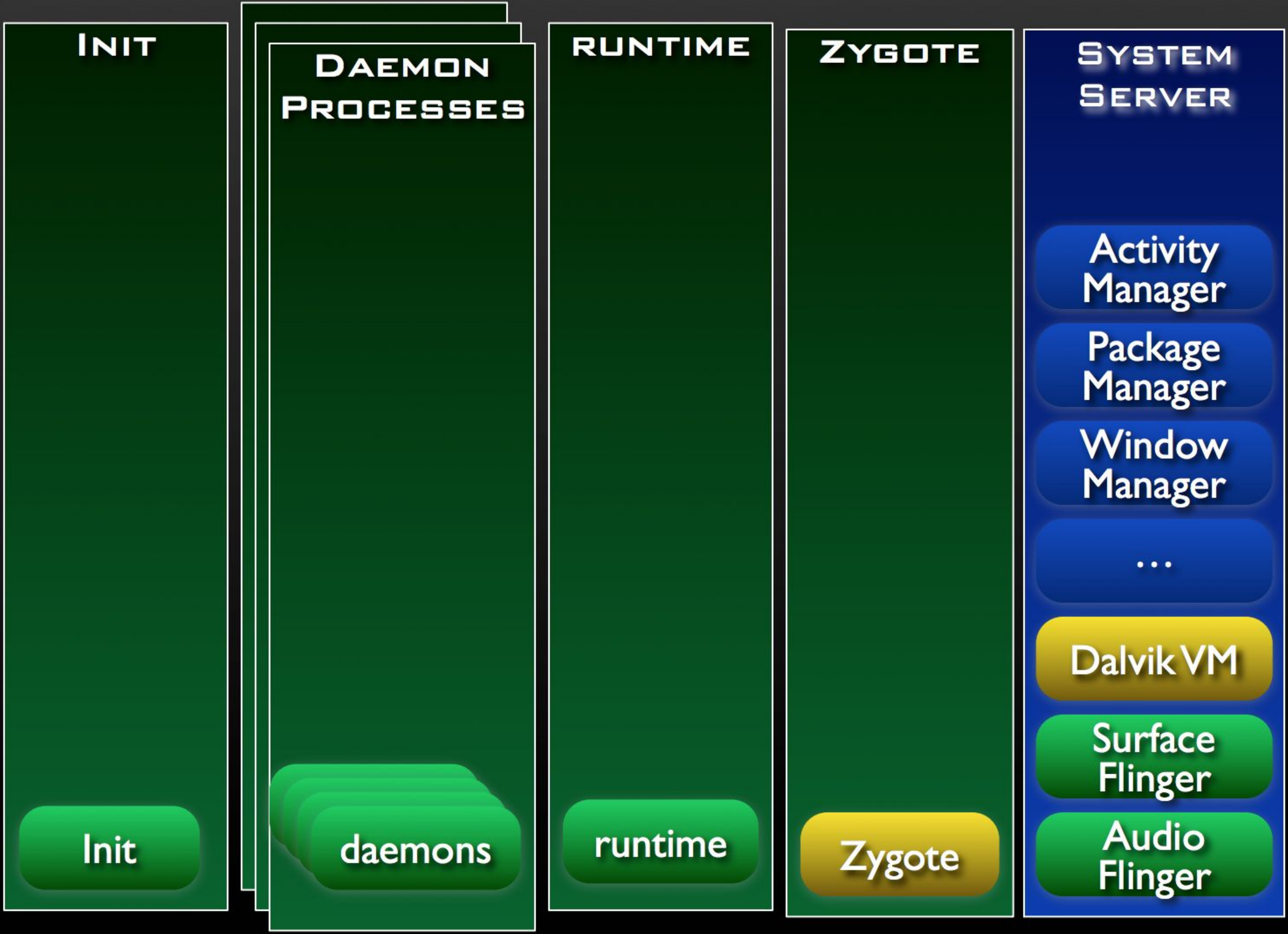


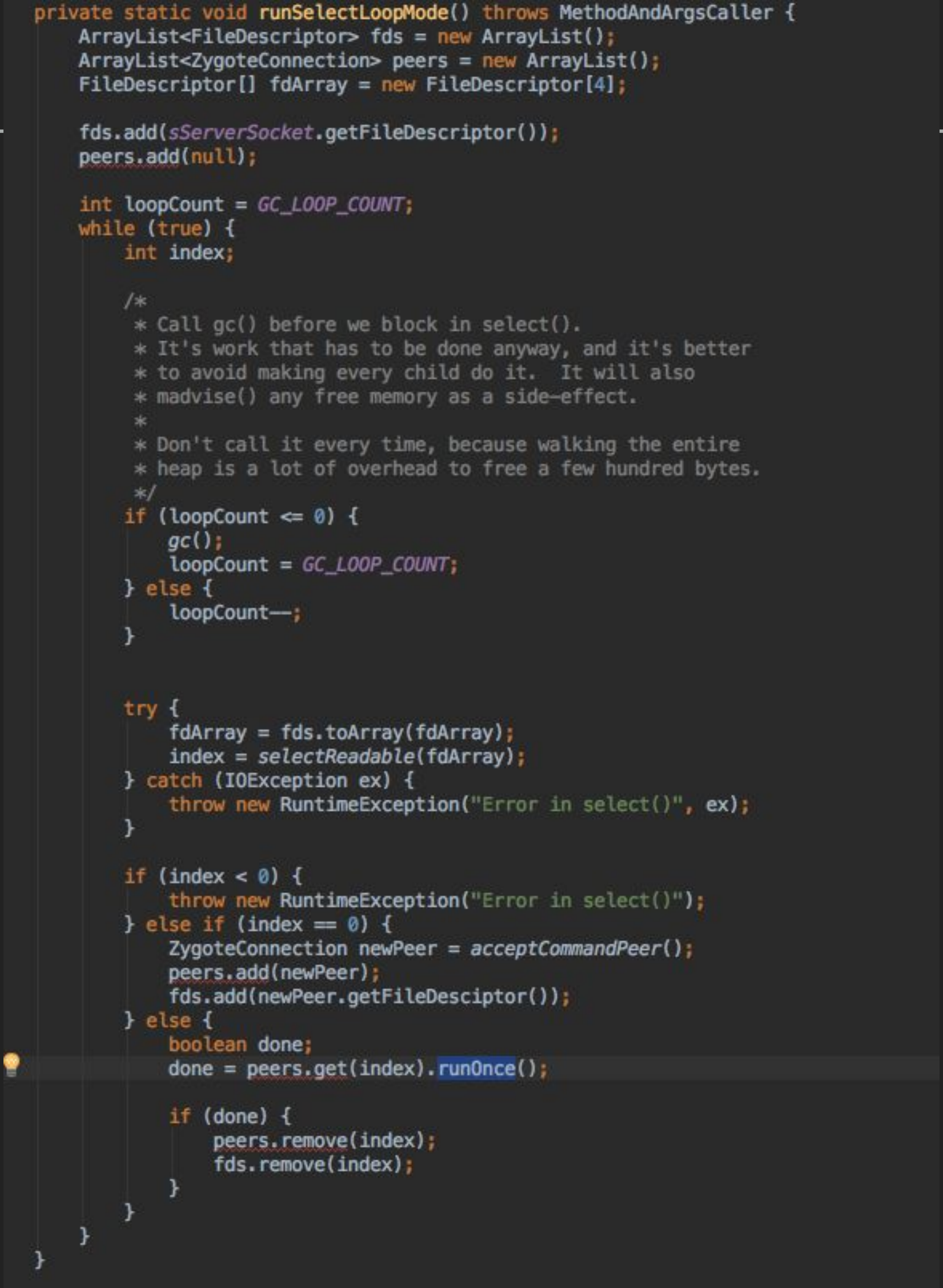


```java
public static final void init2() {
    Slog.i(TAG, "Entered the Android system server!");
    Thread thr = new ServerThread();
    thr.setName("android.server.ServerThread");
    thr.start();
}
```

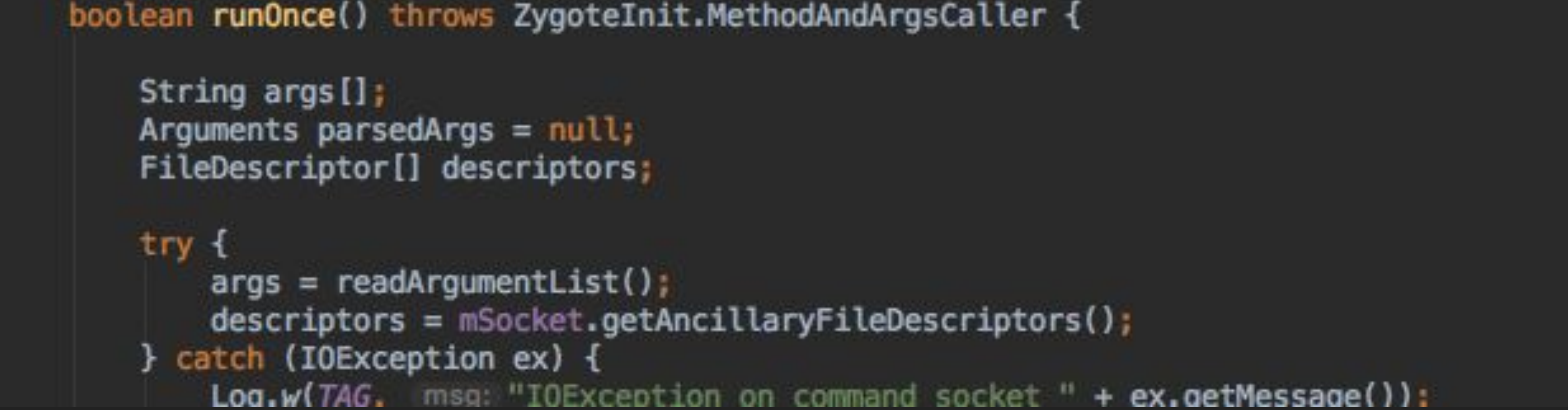
```java
@Override
public void run() {
    EventLog.writeEvent(EventLogTags.BOOT_PROGRESS_SYSTEM_RUN,
        SystemClock.uptimeMillis());

    Looper.prepare();
```

```java
    LightsService lights = null;
    PowerManagerService power = null;
    BatteryService battery = null;
    AlarmManagerService alarm = null;
    NetworkManagementService networkManagement = null;
    NetworkStatsService networkStats = null;
    NetworkPolicyManagerService networkPolicy = null;
    ConnectivityService connectivity = null;
    WifiP2pService wifiP2p = null;
    WifiService wifi = null;
    IPackageManager pm = null;
    Context context = null;
    WindowManagerService wm = null;
    BluetoothService bluetooth = null;
    BluetoothA2dpService bluetoothA2dp = null;
    DockObserver dock = null;
    UsbService usb = null;
    UiModeManagerService uiMode = null;
    RecognitionManagerService recognition = null;
    ThrottleService throttle = null;
    NetworkTimeUpdateService networkTimeUpdater = null;

    // Critical services...
    try {
        Slog.i(TAG, "Entropy Service");
        ServiceManager.addService("entropy", new EntropyService());

        Slog.i(TAG, "Power Manager");
        power = new PowerManagerService();
        ServiceManager.addService(Context.POWER_SERVICE, power);

        Slog.i(TAG, "Activity Manager");
        context = ActivityManagerService.main(factoryTest);

        Slog.i(TAG, "Telephony Registry");
        ServiceManager.addService("telephony.registry", new TelephonyRegistry(context));
```

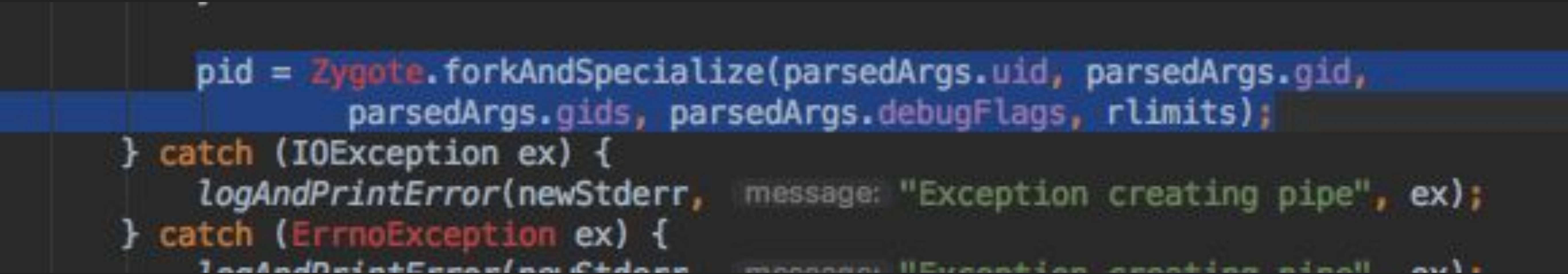

Init
Daemon Processes
runtime
Zygote
System Server
Activity Manager
Package Manager
Window Manager
...
Dalvik VM
Surface Flinger
Audio Flinger
Init
daemons
runtime
Zygote



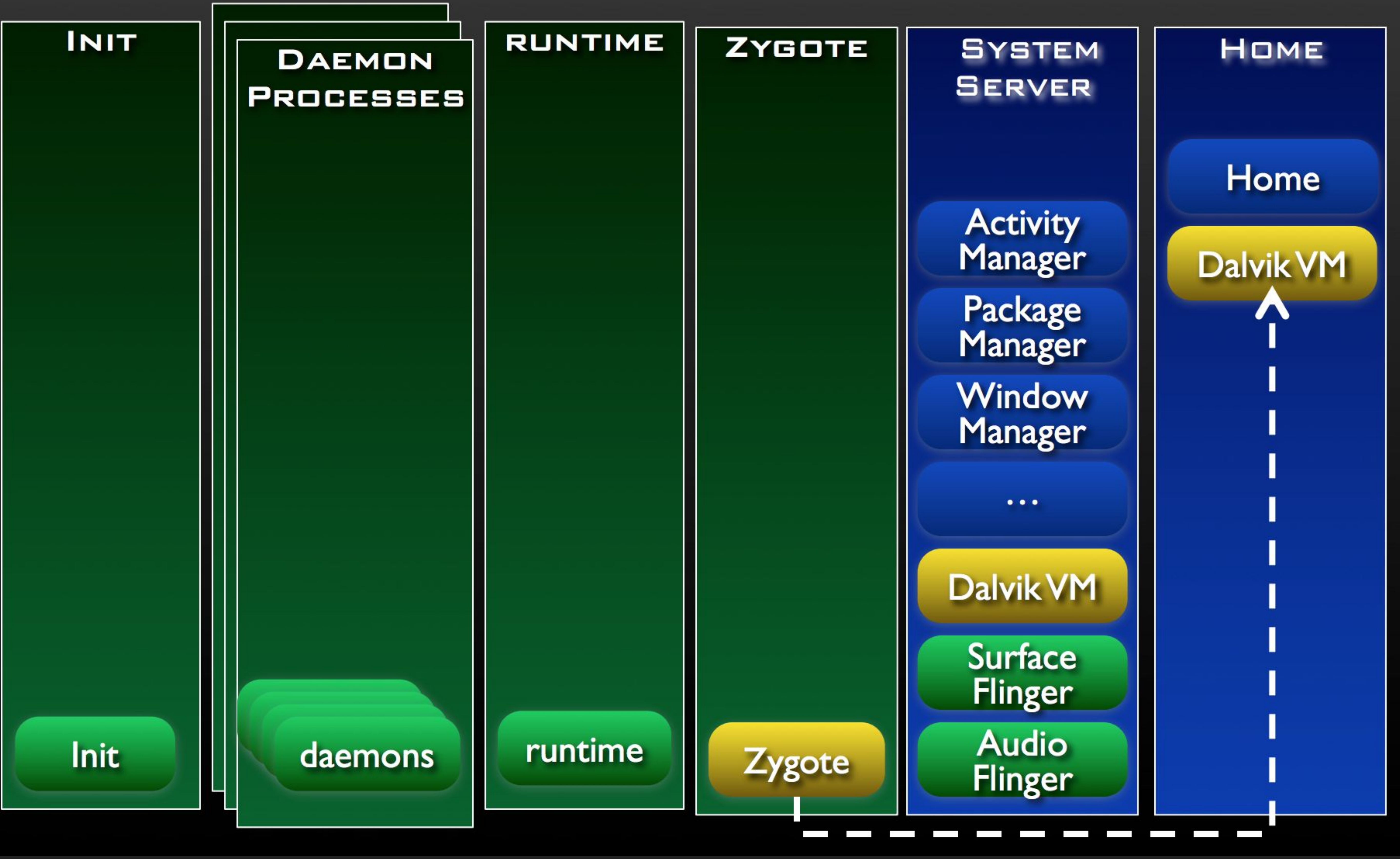
```java
private static void runSelectLoopMode() throws MethodAndArgsCaller {
    ArrayList<FileDescriptor> fds = new ArrayList();
    ArrayList<ZygoteConnection> peers = new ArrayList();
    FileDescriptor[] fdArray = new FileDescriptor[4];

    fds.add(sServerSocket.getFileDescriptor());
    peers.add(null);

    int loopCount = GC_LOOP_COUNT;
    while (true) {
        int index;

        /*
         * Call gc() before we block in select().
         * It's work that has to be done anyway, and it's better
         * to avoid making every child do it.  It will also
         * madvise() any free memory as a side-effect.
         *
         * Don't call it every time, because walking the entire
         * heap is a lot of overhead to free a few hundred bytes.
         */
        if (loopCount <= 0) {
            gc();
            loopCount = GC_LOOP_COUNT;
        } else {
            loopCount--;
        }


        try {
            fdArray = fds.toArray(fdArray);
            index = selectReadable(fdArray);
        } catch (IOException ex) {
            throw new RuntimeException("Error in select()", ex);
        }

        if (index < 0) {
            throw new RuntimeException("Error in select()");
        } else if (index == 0) {
            ZygoteConnection newPeer = acceptCommandPeer();
            peers.add(newPeer);
            fds.add(newPeer.getFileDesciptor());
        } else {
            boolean done;
            done = peers.get(index).runOnce();

            if (done) {
                peers.remove(index);
                fds.remove(index);
            }
        }
    }
}
```



```
boolean runOnce() throws ZygoteInit.MethodAndArgsCaller {

    String args[];
    Arguments parsedArgs = null;
    FileDescriptor[] descriptors;

    try {
        args = readArgumentList();
        descriptors = mSocket.getAncillaryFileDescriptors();
    } catch (IOException ex) {
        Log.w(TAG, msg: "IOException on command socket " + ex.getMessage());
```



```
        pid = Zygote.forkAndSpecialize(parsedArgs.uid, parsedArgs.gid,
                parsedArgs.gids, parsedArgs.debugFlags, rlimits);
    } catch (IOException ex) {
        logAndPrintError(newStderr, message: "Exception creating pipe", ex);
    } catch (ErrnoException ex) {
        [illegible]
```


Init
Daemon Processes
runtime
Zygote
System Server
Home
Home
Activity Manager
Dalvik VM
Package Manager
Window Manager
...
Dalvik VM
Surface Flinger
Init
daemons
runtime
Zygote
Audio Flinger

## AND WE ARE READY