Android OS Internals

Use case :

Creating a custom system service

Kushtrim Pacaj
Lead Android Engineer @Solaborate

What is Android ?

This guy!

I FIXED IT!

- AOSP - > Android Open Source Project

- Licensed with Apache 2.0

- Core is an (android-*ized*) Linux Kernel (GPL v2)

- Apps are developed in Java || Kotlin
- Android framework is Java/C++
- Bit of XML for UI specs
- Apps are executed Dalvik/ART Virtual Machine

Android OS simplified SW stack overview

Why create a system service instead of app service?

- When you need to do something that is shared between all apps
- When something needs to be always running
- When you need to have elevated privileges

  etc…

How to create one ?

- “Basic” steps :
    - Write the interface (AIDL - Binder (Proxy/Stub))
    - Write the ***manager*** part (use the proxy)
    - Write the ***service*** code (extends the stub)
    - Register it with the system

Example

- Audio Firmware use-case: ((()))
    - IAudioFirmwareService.aidl -> framework/base/core/java/android/app/**
    - AudioFirmwareManager*.java* -> framework/base/core/java/android/app/**
    - AudioFirmwareService*.java* -> *framework/base/core/java/android/server/***
    - Add new files to Android.mk makefile -> *framework/base/*

Manager / Service

```java
public class AudioFirmwareManager {

    private final IAudioFirmwareService mService;

    /**
     * @hide
     */
    public AudioFirmwareManager(IAudioFirmwareService service) { mService = service; }
```

(app process)*

Binder IPC

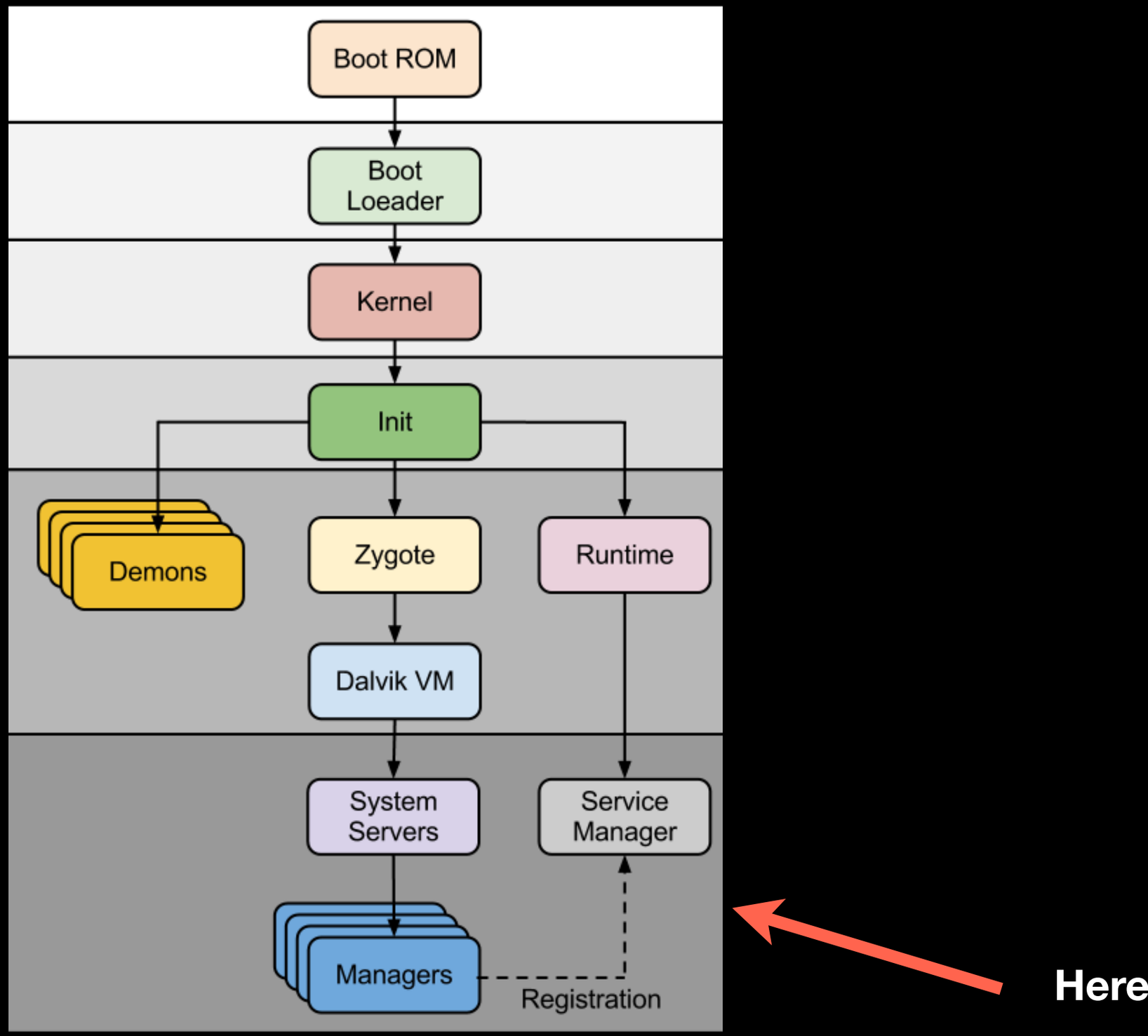
```java
public class AudioFirmwareService extends IAudioFirmwareService.Stub {
    private final Context mContext;


    public AudioFirmwareService(Context context) {
        super();
        mContext = context;
    }

    @Override
    public void run
```

(system_process)

Now how to start it



Registering it

```
registerService(Context.JOB_SCHEDULER_SERVICE, JobScheduler.class,
        (StaticServiceFetcher) () -> {
        IBinder b = ServiceManager.getService(Context.JOB_SCHEDULER_SERVICE);
        return new JobSchedulerImpl(IJobScheduler.Stub.asInterface(b));
    });

registerService(Context.AUDIO_FIRMWARE_SERVICE, AudioFirmwareManager.class,
        (StaticServiceFetcher) () -> {
        IBinder b = ServiceManager.getService(Context.AUDIO_FIRMWARE_SERVICE);
        return new AudioFirmwareManager(IAudioFirmwareService.Stub.asInterface(b));
    });

registerService(Context.PERSISTENT_DATA_BLOCK_SERVICE, PersistentDataBlockManager.class,
```

Manager -> SystemServiceRegistry.java

```
            */
            mSystemServiceManager.startService(MOUNT_SERVICE_CLASS);
            mountService = IMountService.Stub.asInterface(
                    ServiceManager.getService( name: "mount"));
        } catch (Throwable e) {
            reportWtf( msg: "starting Mount Service", e);
        }
    }

    try {
        Slog.i(TAG, msg: "Starting " + Context.AUDIO_FIRMWARE_SERVICE);
        ServiceManager.addService(Context.AUDIO_FIRMWARE_SERVICE, new AudioFirmwareService(context));
    } catch (Throwable e) {
        Slog.e(TAG, msg: "Failure starting AudioFirmwareService Service", e);
    }
}
Slog.i(TAG, msg: "start UiModeManagerService.");
```

Service -> Start it from SystemServer.java

Using it from apps

- Standard API: context.getSystemService(……)

Using it from apps

- Standard API: context.getSystemService(……)

```
AudioFirmwareManager audioFirmwareManager = (AudioFirmwareManager) getSystemService(Context.AUDIO_FIRMWARE_SERVICE);
```

Cannot resolve symbol 'AudioFirmwareManager'

```
}
```

Using it from apps

- Class available in runtime!
- Class not found when trying to compile app, since not part of standard SDK distribution

  ===== Solution ======

- Generate new SDK from AOSP source

The end!

Any Q/A is welcomed during session breaks!

Kushtrim Pacaj
Lead Android Engineer @Solaborate

https://github.com/KushtrimPacaj