Unleashing the Power of Android for Vehicle Cockpit Solutions

IoTShow.in @ India Electronics Week

By

Sankalp Rajan P & Rajesh Sola
L&T Technology Services

25-11-2022

Session Agenda

Overview

- Quick intro to Vehicle Cockpit & Infotainment
- Android scope for Software Defined Vehicles
- Introduction to Android, Android Automotive, Feature listing
- Architecture Insights of Android, Android Automotive OS (AAOS)

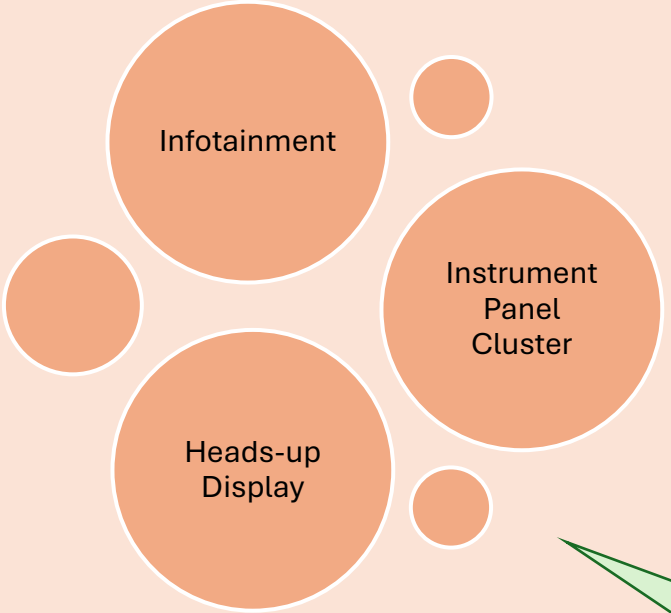
Internals

- Android Framework Components – HAL, System Services, APIs
- Automotive HAL Extensions, VHAL
- Car Services & Car API
- Consumption of Services by Car Apps

Android Auto & Other Concepts

Introduction to Infotainment

Software Defined Vehicles – Cockpit Solutions



Ref:- **https://www.arm.com/solutions/automotive/digital-cockpit**

Automotive Cockpit

Contemporary Trends:-

- ❑ Virtualization (Hypervisor)s
- ❑ Cockpit Domain Controller (CDC)

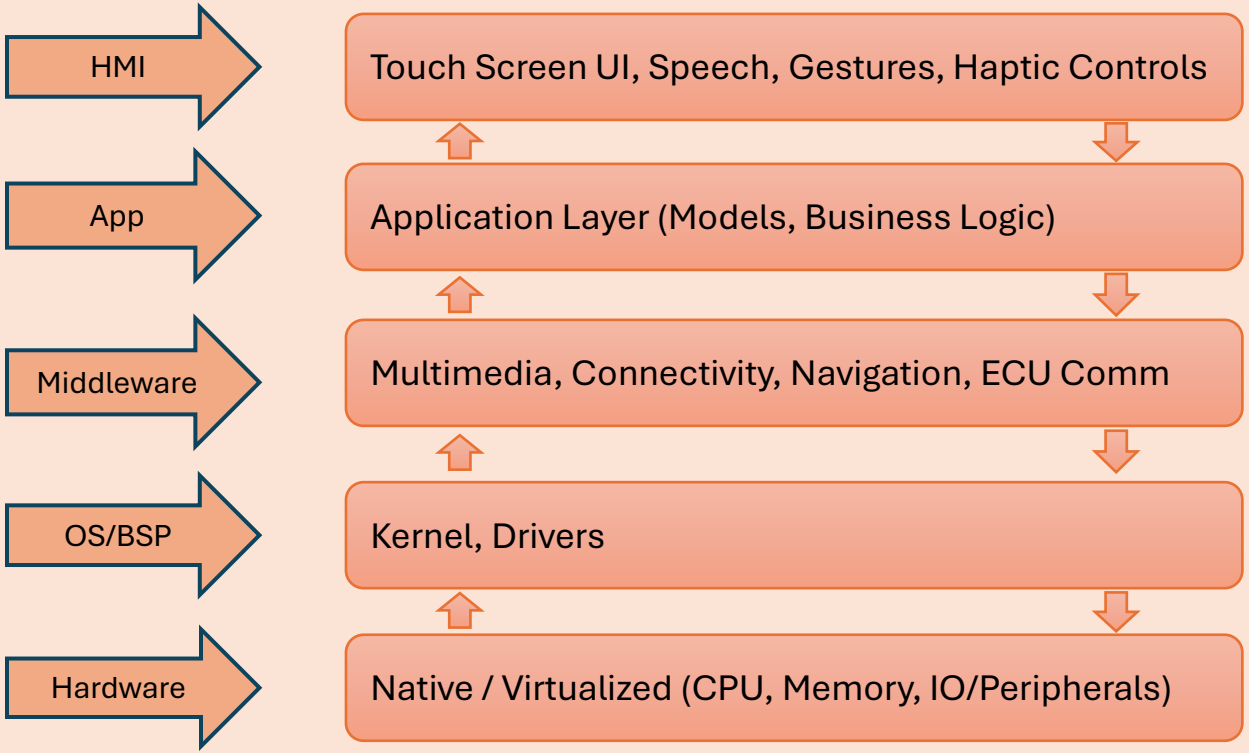
Additional Reading:-

Reshaping_Cockpit_Architecture - Infineon Technologies

Infotainment - Functionality & Services

- Multimedia – Audio, Video
- Radio Services – AM, FM, DAB (Digital - Satellite, Internet Radios)
- Navigation Services
- Weather Updates
- HVAC & Vehicular Functions
- Human Machine Interface(HMI) - Touch Screen, Voice Recognition
- Telematics Integration
- Connectivity – Wifi, Bluetooth, Ethernet, AVB
- Smart Phone Integration & Mirroring
- Personalization, e.g. AddressBook, Calendar
- RearSeat Entertainment
- OTA / FOTA

Infotainment - Layered Architecture

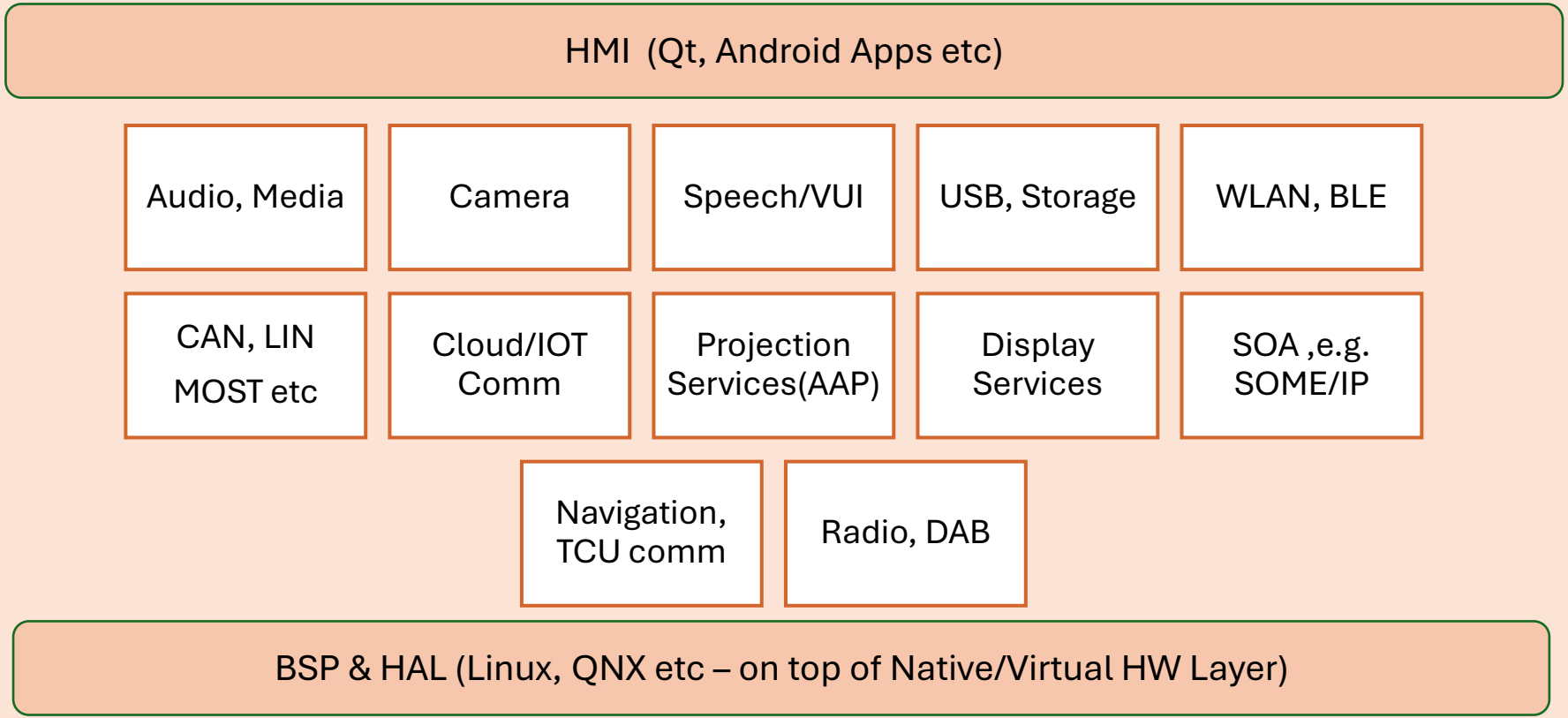


Enabling Technologies:-

- ➔ Linux/QNX as platform base + QtQML or alternative HMI
- ➔ Yocto baseline for Linux
- ➔ Android Automotive

Mapping of these layers w.r.t Android Architecture

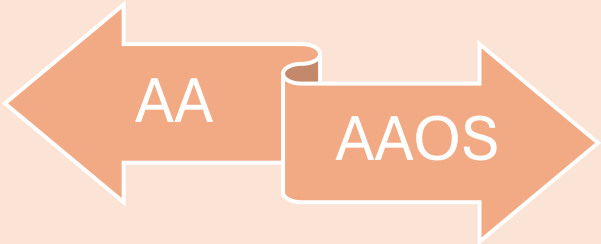
Infotainment Middleware



Introduction to Android

Scope of Android for Cars

- ❑Connected Apps [Remote Control], e.g., SmartDeviceLink [Not in scope of this session]
- ❑Mirroring/Projection – Android Auto [Small focus at end]
- ❑Complete OS for Head Unit – Android Automotive OS (AAOS) [Major focus]



Android may not meet all Safety Critical requirements

Not a choice for certain ASIL levels of ISO26262

What is Android and Why Android?

- ❑Familiar UI and Apps
- ❑Larger Developer Eco System
- ❑Consistent releases & improvements from Google
- ❑Clear Models for Chip Vendors, OEMs and Tier-x suppliers (HAL, Service Layers, Vendor Additions etc)

AA vs AAOS

Android Auto

- Projection or Mirroring Technology
- Limited/no access to Vehicular functions/information

Android Automotive

- Full OS/Platform solution for Infotainment in Head Unit
- Better access to Vehicle info, thru ECU communication

Android Automotive (AAOS)

Android Auto(AA)

Both are not same !!

Key Features/Considerations in AAOS

- Car Launcher/System UI
- Car Settings
- Vehicle Camera (EVS)
- Audio
- Connectivity
- Display and Input
- Rotary Controllers
- Location & Navigation
- Radio (AM/FM)
- Notifications, HUN
- Driver Distractions Optimization(DO)
- Voice, VUI
- Storage/Flash Wear
- Power Management
- Security

Google Automotive Services(GAS)

- Maps
- Playstore
- Assistant

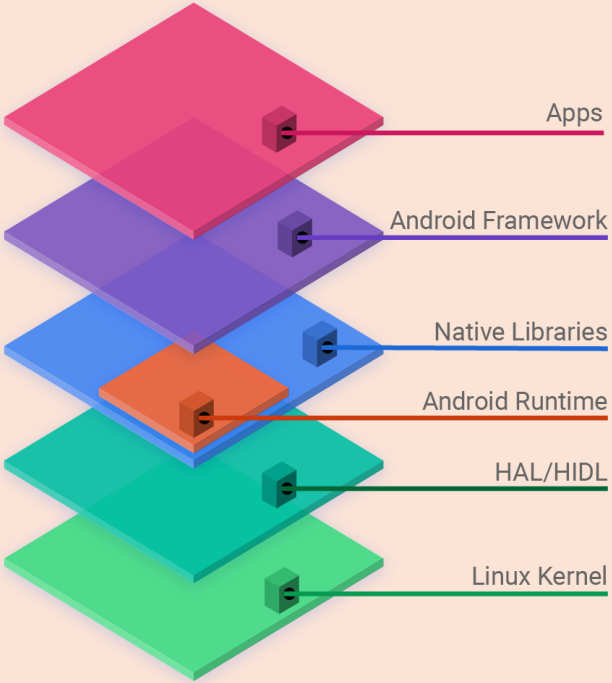
Instrument Cluster

Architecture Insights

Android Architecture – Key Layers

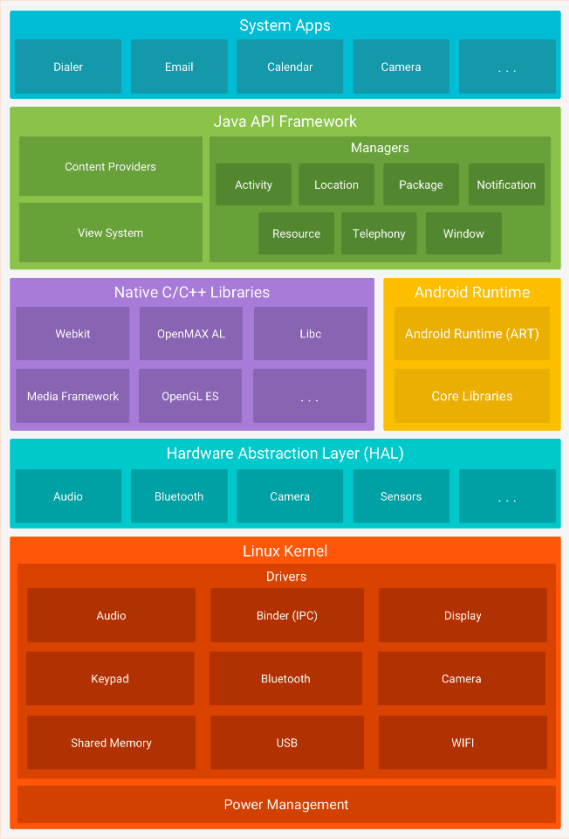
Apps	Alarm - Browser - Calculator - Calendar - Camera - Clock - Contacts - IM Dialer - Email - Home - Media Player - Photo Album - SMS/MMS - Voice Dial
Framework	Content Providers - Activity Manager - Location Manager - View System Package Manager - NotificationManager - Resource Manager Telephony Manager - Window Manager
Native Libraries	Audio Manager - LIBC - SSL - Freetype - Media OpenGL/ES - SQLite - Webkit - Surface Manager
Runtime	Core Libraries Android Runtime (ART)
HAL	Audio - Bluetooth - Camera - DRM - External Storage - Graphics - Input Media - Sensors - TV
Linux Kernel	Drivers (Audio - Binder (IPC) - Bluetooth Camera Display - Keypad Shared Memory USB Wi-fi) - Power Management
Secure Element	Trusted Execution Environment

Ref:- https://source.android.com/docs/setup/about



Ref:- https://source.android.com

Android Platform – Developer View

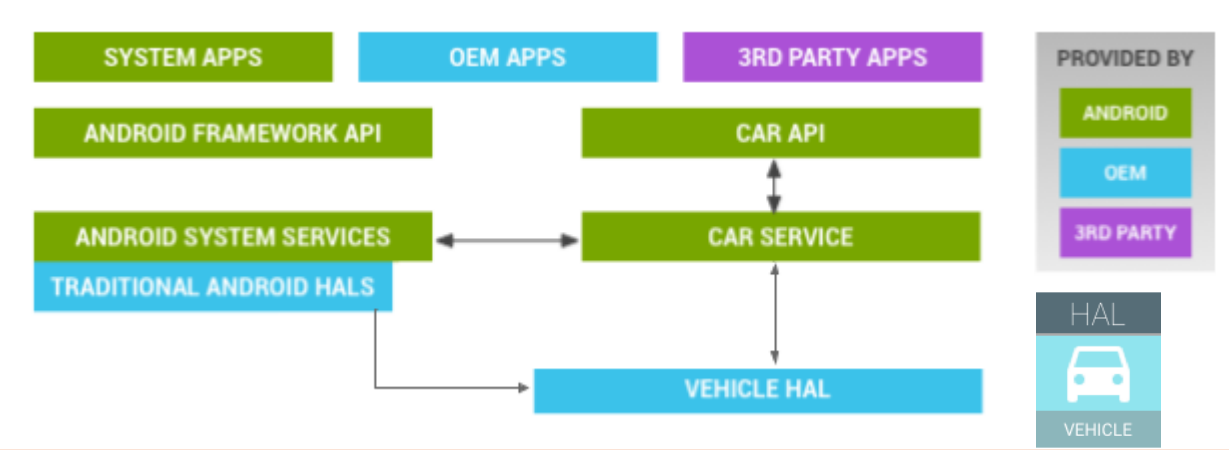


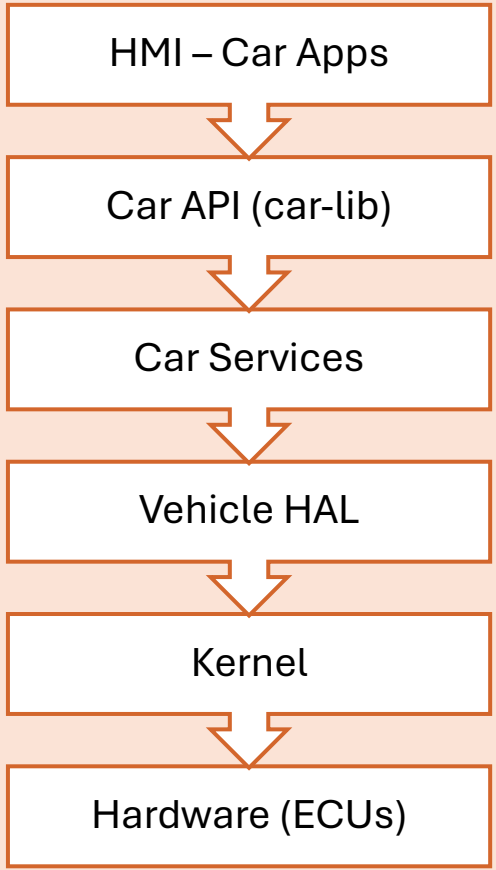
Key Points/Terminology

- ❑ Android Runtime(ART), migration from DVM from Kitkat
- ❑ Ahead of Time (AOT) model followed by ART, Dex Optimization .. Contrast to JIT model by Dalvik VM
- ❑ App Development Phases, APK & Dex Formats
- ❑ Multiplexing and Demultiplexing of services and components across layers
- ❑ System Services – Java/Native
- ❑ JNI Bindings
- ❑ Binder for IPC & RPC

Ref:- *https://developer.android.com/guide/platform*

Android Automotive Architecture





Code Pointers:-

- ❑ HAL :- hardware/interfaces/automotive/*
- ❑ Car Service:- platform/packages/services/Car/service
- ❑ Car API :- platform/packages/services/Car/car-lib

Android File System

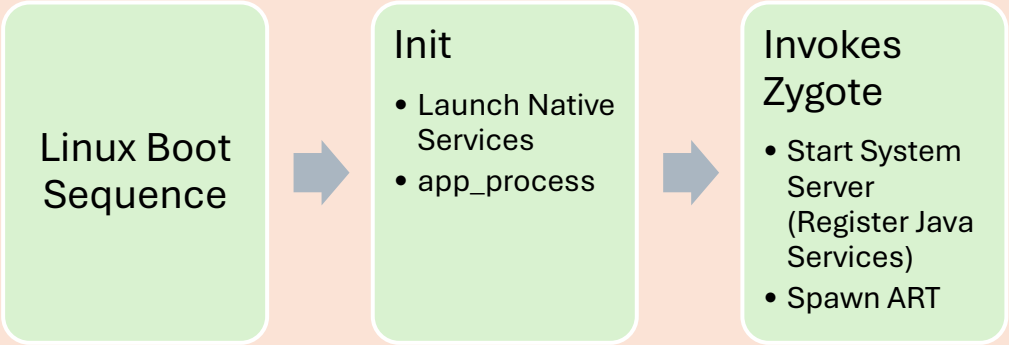
- Imp file systems and mount points in Android
  - /
  - /system (system.img)
  - /vendor (vendor.img)
  - /cache (cache.img)
  - /data (data.img)
  - /mnt/sdcard

- Imp top level directories in root (/), /system
- Mapping of AOSP top dirs. with File System Components

Pseudo File Systems
- /acct
- /dev
- /proc
- /sys
- /sys/kernel/debug

Android Boot Sequence

- ❑Std Linux Boot Sequence
  - ❑Primary bootloader
  - ❑Secondary Boot loader
  - ❑Kernel
  - ❑Init, initrc scripts
- ❑Staring Native Daemons
- ❑app_process
- ❑Zygote
- ❑System Server
- ❑Android runtime
- ❑Service Managers/Activity Manager



Self Replicate for every new Android APp

Android Kernel & ABI

Few changes/differences from std Linux kernel (Androidism)

- Binder
- ashmem
- pmem
- logger
- wakelocks
- oom handling
- alaram manager

Android Common Kernels (ACK)

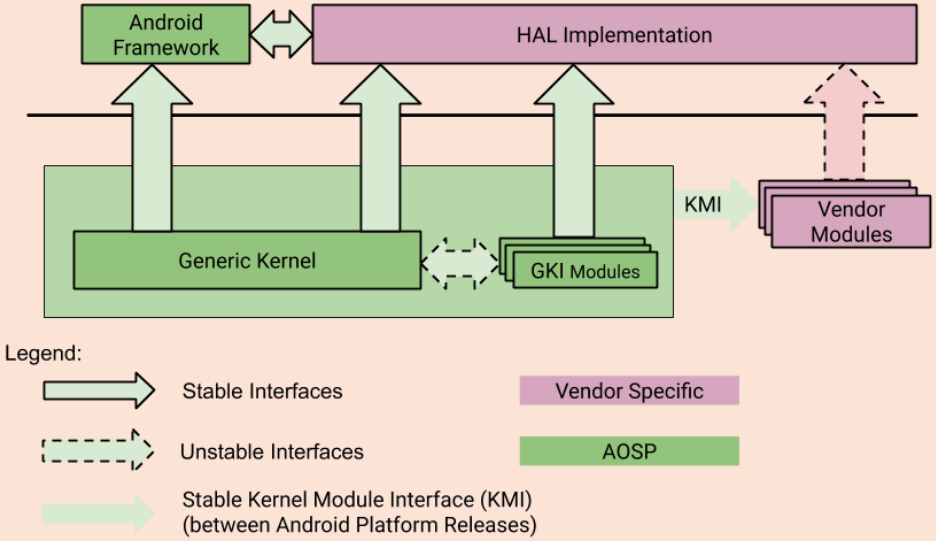
Bionic – libc implementation in Android

Toybox for shell commands

Android Kernel source is not part of AOSP. Pre-built kernel images are part of AOSP. If you wish for custom kernel, sources should be downloaded and build externally

Building Kernels | Android Open Source Project

General Kernel Image (GKI) Architecture



- Android/AOSP Common Kernel (ACK)
- General Kernel Image (GKI)
- Kernel Module Interface(KMI)
- Kernel Branches
- DLKM, Vendor Specific

Ref:-
https://source.android.com/docs/core/architecture/kernel
https://source.android.com/docs/core/architecture/kernel/generic-kernel-image

Demo/Tutorial : AOSP Build

AOSP Code base & Android Build System

- Preparing System
- Build Steps
- Build Tips/Tricks
- Build Insights

Browsing AOSP Code:-

- http://aospxref.com/
- https://cs.android.com/
- http://aosp.opersys.com/

Top level directories in AOSP and their purpose?

Android Generic System Images (GSI)

Ref:-

1) *https://source.android.com/docs/setup/create/gsi*

2) https://developer.android.com/topic/generic-system-image

3) Understand the impact of Generic System Images (GSI) (Android Dev Summit '18)

Build Steps & Tricks

- ❑ Build Steps
  - ❑ Install necessary Ubuntu packages – Click Here
  - ❑ Setting up repo client – Click Here
  - ❑ Fetching AOSP Sources
  - ❑ Choose build target – lunch menu (Car Emulator target)
  - ❑ Building AOSP – m or make –j x
  - ❑ Emulating built system image - emulator

```
repo init -u https://android.googlesource.com/platform/manifest  \
                                  -b android-11.0.0_r2 -depth=1
repo sync -c
source build/envsetup.sh
lunch aosp_car_x86_64-userdebug   # 10 or 11 or 12
m  # make -j x
emulator
```

https://source.android.com/docs/setup/build/building

Build Tricks hmm

- ❑ m
- ❑ mm
- ❑ mmm
- ❑ mma
- ❑ mmma
- ❑ godir
- ❑ croot
- ❑ clean
- ❑ cgrep, jgrep
- ❑ allmod
- ❑ gomod

Supported Targets

- ❑ Supported Architecture & Target Boards
- ❑ Flashing Devices (adb, fastboot)
  https://source.android.com/docs/setup/build/flash
  https://source.android.com/docs/setup/build/running
- ❑Emulated Target – goldfish
- ❑Emulator on cloud – cuttlefish, WebRTC Streaming
  (*https://source.android.com/docs/setup/create/cuttlefish*)
- ❑ Adding new device support
  (*https://source.android.com/docs/setup/create/new-device*)

Check via AOSP Build – lunch menu!!

Debugging - ADB

- ADB tool generated from AOSP build

  (https://source.android.com/docs/setup/build/adb)
- Some imp commands to run in ADB shell
  - adb device –l
  - logcat / adb logcat
  - dumpsys / adb dumpsys
  - lshal
  - install
  - push
  - pull
  - root

Top level dirs. Under AOSP

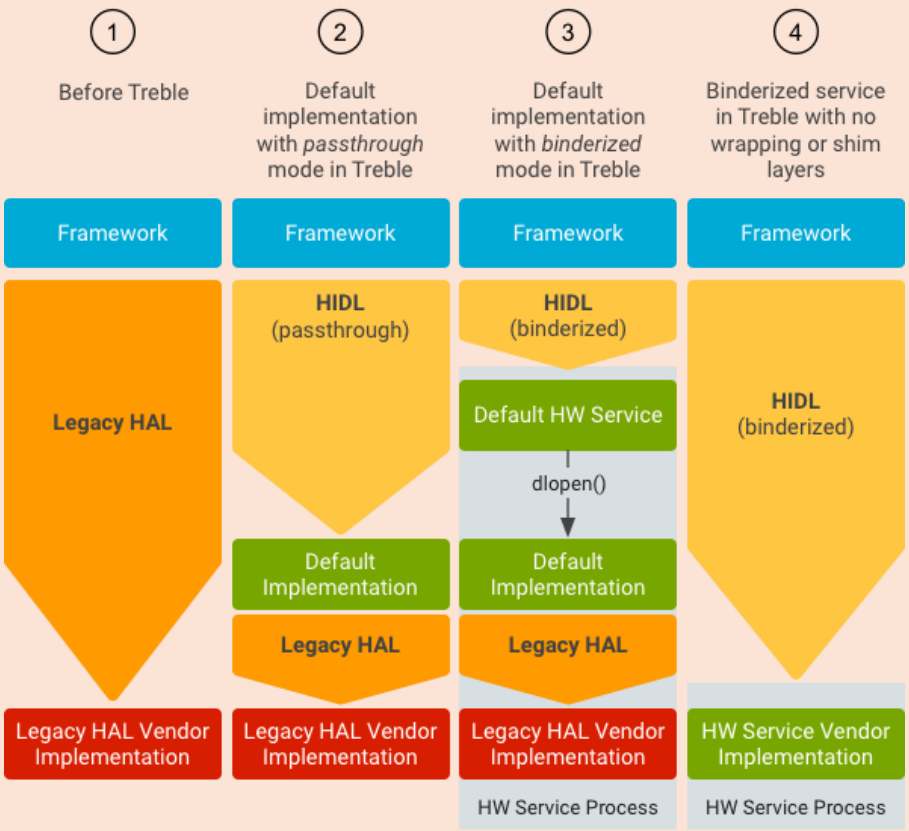
- ❑ art
- ❑ bionic
- ❑ bootable
- ❑ build
- ❑ cts
- ❑ dalvik
- ❑ development
- ❑ device
- ❑ external
- ❑ frameworks
- ❑ hardware
- ❑ kernel
- ❑ libcore
- ❑ packages
- ❑ platform_testing
- ❑ prebuilts
- ❑ sdk
- ❑ system
- ❑ test
- ❑ toolchain
- ❑ tools

Android Framework Internals : HAL

Android HAL

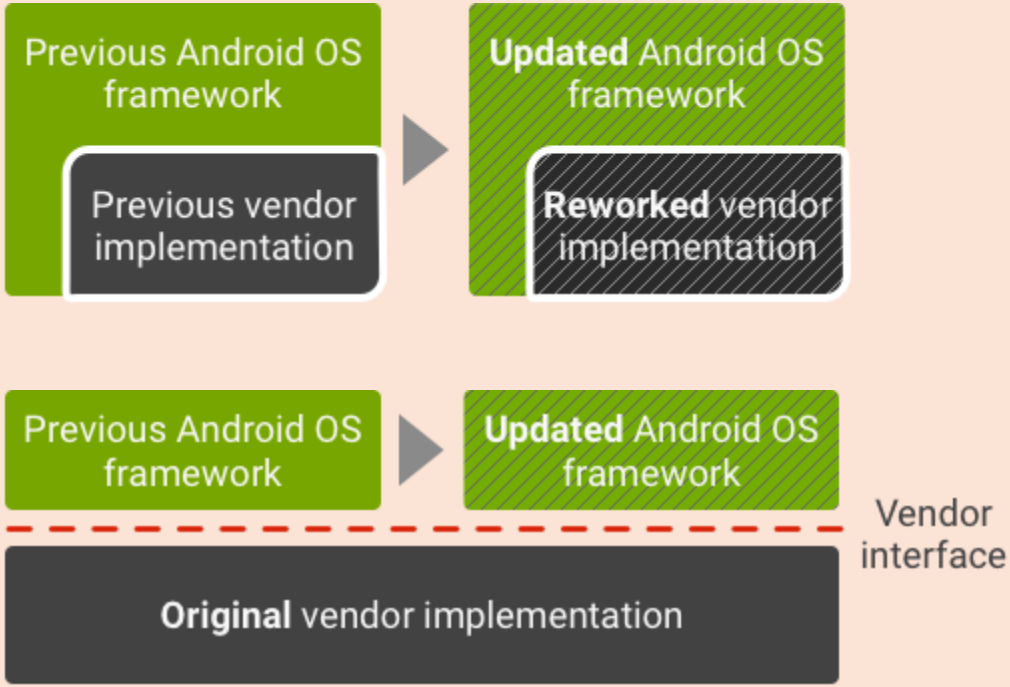
- Before Treble (8.x) – Legacy HAL
  - HAL interfaces as C Header files (hardware/libhardware/include/hardware/*.h)
  - Change of headers across versions
  - HAL libraries loaded at boot time (statically)
  - Interoperability issues across Android version upgrades
- Modern HAL, from 8.x, Project treble
  - Freezed HAL Signatures, Immutable (By Google)
  - HIDL Syntax (hardware/interfaces/*/?.?/*.hal)
  - HAL implementations by Chip Vendors
- Types of HAL – Passthrough HAL and Binderized HAL
- Some more imp concepts
  - VNDK
  - VINTF - Vendor Interface Object
  - AVB - Android Verified Boot

HAL Types



- ❑Types of HAL
  - ❑Passthrough HAL
    - ❑Implemented as shared obj files (/vendor/lib/*.so, /vendor/lib64/*.so)
    - ❑Also known as same-process HAL (sp-hal)
  - ❑Binderized HAL
    - ❑Implemented as daemon process
    - ❑Clients communicate via Binder IPC
- ❑Type of HAL specified in *device/generic/goldfish/manifest.xml*
- ❑Demo - Other code level differences
- ❑Modern HIDL wrapper to Legacy HALs

Android HAL



Project Treble!!

Android HAL – Key Concepts

HIDL Syntax

Stub Creation – C++ / Java Skelton Code

Implementation code

HIDL Data Types

HIDL Callbacks

Few examples – light, vibrator

AIDL Syntax for HAL Interfaces in recent AOSP Versions

Custom HAL Examples – Code Highlights

```
# hardware/interfaces/example/2.0/IExample.hal
package android.hardware.example@2.0;

interface IExample {
    add(int32_t op1, int32_t op2) generates (int32_t valueRet);
    multiply(int32_t op1, int32_t op2) generates (int32_t valueRet);
    compute(int32_t valueIn) generates (int32_t valueRet);
    greet(string buffer) generates (int32_t result);
};
```

```
 LOC=hardware/interfaces/example/2.0/default/
 # make hidl-gen -j64    # one time, if tool is not built
 hidl-gen -o $LOC -Lc++-impl -randroid.hardware:hardware/interfaces -
          randroid.hidl:system/libhidl/transport $PACKAGE
 hidl-gen -o $LOC -Landroidbp-impl -randroid.hardware:hardware/interfaces -
          randroid.hidl:system/libhidl/transport $PACKAGE
./hardware/interfaces/update-makefiles.sh
```

Custom HAL Examples – Code Highlights

```
Uncommented code in generated Example.h
   extern "C" IExample* HIDL_FETCH_IExample(const char* name);

Uncommented code in generated Example.cpp
   IExample* HIDL_FETCH_IExample(const char* /* name */) {
           return new Example();
   }
```

```
 device/generic/goldfish/manifest.xml
 <hal format="hidl">
           <name>android.hardware.example</name>
           <transport arch="32+64">passthrough</transport>
           <version>2.0</version>
           <interface>
           <name>IExample</name>
           <instance>default</instance>
           </interface>
</hal>
```

Custom HAL Examples – Code Highlights

```
//hardware/interfaces/example/2.0/default/Example.cpp

Return<int32_t> Example::add(int32_t op1, int32_t op2) {
    ALOGD("Example HAL -- add is invoked");
    return int32_t {op1 + op2};
}
Return<int32_t> Example::multiply(int32_t op1, int32_t op2) {
    ALOGD("Example HAL -- multiply is invoked");
    return int32_t {op1 * op2};
}
Return<int32_t> Example::compute(int32_t valueIn) {
    ALOGD("Example HAL -- compute is invoked");
    return int32_t { valueIn * valueIn};
}
Return<int32_t> Example::greet(const hidl_string& buffer) {
    ALOGD("Example HAL -- greet is invoked, received:%s\n", buffer.c_str());
    int32_t len = buffer.size();
    return int32_t {len};
}
```

Custom HAL Examples – Code Highlights

```
-   Entries to device/generic/goldfish/vendor.mk
-   Entries to build/target/product/gsi/30.txt    # current.txt
```

```
Testing HAL:-
      lshal
      ls /vendor/lib64/hw
      ls /vendor/lib/hw
```

Custom HAL Examples – Code Highlights

```
//Client Code
#include <android/hardware/example/2.0/IExample.h>
using android::hardware::example::V2_0::IExample;
//other headers and type aliased
int main() {

      android::sp<IExample> hal = IExample::getService();
      char str[]="Hello Android";
      android::hardware::Return<int32_t> ares = hal->greet(str);
      printf("Length is %d\n", ares);
      android::hardware::Return<int32_t> cres = hal->compute(10);
      printf("Square is %d\n", cres);
      int ares = hal->compute(10,20);
      printf("Square is %d\n", mres);
      int mres = hal->compute(12,15);
      printf("Square is %d\n", mres);
      return 0;
}
```

Additional Code/Changes for Binderized HAL

```
# device/generic/goldfish/manifest.xml
<hal format="hidl">
          <name>android.hardware.sample</name>
          <transport arch="32+64">hwbinder</transport>
          <version>2.0</version>
          <interface>
          <name>ISample</name>
          <instance>default</instance>
          </interface>
</hal>
```

```
# android.hardware.sample@2.0-service.rc
service samplehwserv /vendor/bin/hw/android.hardware.sample@2.0-service
   class hal
   user root
   group root
   seclabel u:r:su:s0
```

Additional Code/Changes for Binderized HAL

```
- Fetching HAL instance
- hardware/interfaces/sample/2.0/default/Sample.h

  static ISample* getInstance(void);

- hardware/interfaces/sample/2.0/default/Sample.cpp

  ISample * ISample ::getInstance(void){
      return new Sample();
  }

- HIDL_FETCH_ISample code is commented in both header & source
```

```
lshal
ls /vendor/bin/hw
ps -A | grep sample
/vendor/etc/init/    # locate rc file
sampletest
logcat | grep Sample
```

Automotive Specific HAL (VHAL)

Automotive HAL Extensions

- Vehicle HAL, VHAL Properties
- Audio HAL
- Camera HAL (EVS)
- Ocuupant Awareness
- Surround View (SV)
- SocketCAN

From Android 11

Vehicle HAL

- ❑IVehicle interface (hardware/interfaces/automotive/vehicle/2.0/IVehicle.hal)
- ❑VHAL Properties (hardware/interfaces/automotive/vehicle/2.0/types.hal)

  5000+ lines of code in Android 12 with 150+ properties
- ❑Two types of properties
  - ❑System properties by Google
  - ❑Vendor specific properties
- ❑VHAL Callbacks

VHAL Property Structure

Group *(4 bits)*	Area/Zone *(4 bits)*	Property Type *(8 bits)*	Unique ID (16 bits)

Vehicle Property Groups:-
SYSTEM
VENDOR

Property Area/Zones:-
GLOBAL
WINDOW
MIRROR
SEAT
DOOR
WHEEL

Vehicle Property Types:-
STRING, BOOLEAN
INT32, INT32_VEC
INT64, INT64_VEC
FLOAT, FLOAT_VEC
BYTES
MIXED

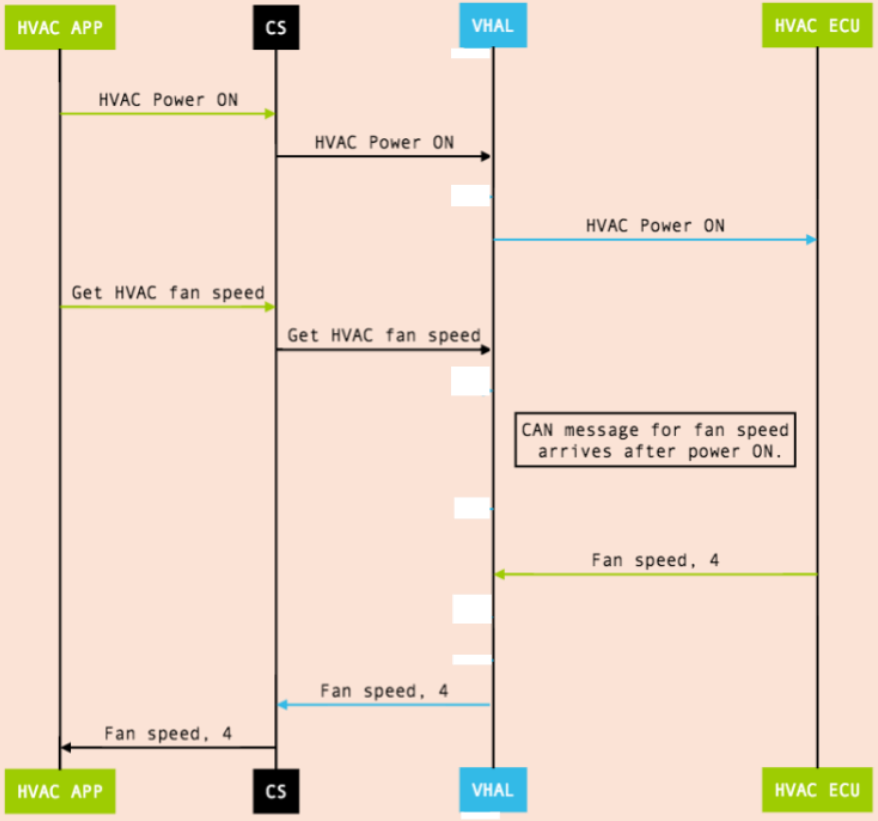
Browsing/Debugging VHAL

- ❑Debugging via lshal in ADB shell

```
lshal debug android.hardware.automotive.vehicle@2.0::IVehicle
```

- ❑Browsing VHAL Properties, via Emulator Extended Controls
- ❑Browsing VHAL Properties , via KitchenSink app

Property Flow – HVAC Example



Ref:- https://source.android.com/docs/devices/automotive/vhal/properties

CanBus HAL in Android

- Based on SocketCAN
- Key Interfaces
  - ICanBus
  - ICanController
  - ICanErrorListener
  - ICanMessageListener
  - ICloseHandle
- Types
  - struct CanMessage
  - struct CanMessageFilter
  - enum FilterFlag
  - enum Result
  - enum ErrorEvent

Android Framework Internals : Services

Java Services – Code Hints

- AIDL files (https://source.android.com/docs/core/architecture/aidl)
- AIDL ➔ framework/base/core/java/android/os/*.aidl
- Adding entries to build system (frameworks/base/Android.bp)
- Generated Stub code – Java ()
- Entries to SystemServer class

  frameworks/base/services/java/com/android/server/SystemServer.java
- API Extensions – Manager class and added entries to Context.java, SystemServerRegistry.java
- JNI Bindings to HAL code implemented in C++
- **Code Walk Through of existing Services – Lights / Vibrator**

Some Android Services (Java)

- ☑ActivityManager
- ☑PowerManager
- ☑PackageManager
- ☑AccountManager
- ☑BatteryService
- ☑LightsService
- ☑VibratorService
- ☑AudioService
- ☑AlarmManager
- ☑LocationManager
- ☑DropBox Service
- ☑NetworkManagement
- ☑ContentManager
- ☑SensorService
- ☑BluetoothService
- ☑WindowManager
- ☑ConnectivityService

Refer AOSP source, for detailed listing of services per version and few additional references

Some Android Services (Native)

- ☑ servicemanager
- ☑ logd
- ☑lmkd
- ☑Installd
- ☑vold
- ☑netd
- ☑rild
- ☑keystore

System Service (Java) – AIDL File

```
# frameworks/base/core/java/android/os/IHelloService.aidl
package android.os;
interface IHelloService {
  /**
  * {@hide}
  */
  int sayHello(String msg);
  int test(int val);
}
```

Entry to frameworks/base/Android.bp

core/java/android/os/IHelloService.aidl

System Service (Java) – Service Implementation

frameworks/base/services/core/java/com/android/server/HelloService.java as

```
package com.android.server;

import android.content.Context;
import android.os.IHelloService;
import android.util.Log;

public class HelloService extends IHelloService.Stub {
    private static final String TAG = "JHelloService";
    private Context mContext;
    private long mNativePointer;

    public HelloService(Context context) {
        super();
        mContext = context;
        Log.i(TAG, "System service initialized");
        //Log.i(TAG, "length of msg : " +
                        sayHello("abcdxyz"));
        //Log.i(TAG, " test returns " + test(10));
    }
```

```
    protected void finalize() throws Throwable {
      super.finalize();
    }

    public int sayHello(String msg)
    {
        int res = msg.length();
        Log.i(TAG, "Hello service -- sayHello invoked");
        return res;
    }

    public int test(int op1)
    {
        int res = op1 * op2;
        Log.i(TAG, "Hello service -- test invoked");
        return res;
    }
}
```

Adding and Testing Service

Add service entry to frameworks/base/services/java/com/android/server/SystemServer.java

```
traceBeginAndSlog("HelloService");
try {
     ServiceManager.addService("hello",  new HelloService(context));
     //ServiceManager.addService(Context.HELLO_SERVICE,HelloService(context));
   }catch(Throwable e) {
     reportWtf("starting Hello Service", e);
}
traceEnd();
```

```
logcat | grep -i hello
service list
dumpsys -l
service call hello  1 s16 "Hello IOT"       # sayHello
service call hello  2 i32 10
```

System Service (Java) – Manger Code

frameworks/base/core/java/android/os/HelloManager.java

```
package android.os;
import android.os.IHelloService;
public class HelloManager
{
    public int sayHello(String mString) {
            try {
                return
mService.sayHello(mString);
            } catch (RemoteException e) {
                return 0;
            }
    }
    public int test(int value) {
            try {
                return mService.test(value);
            } catch (RemoteException e) {
                return 0;
            }
    }
    public HelloManager(IHelloService service) {
        mService = service;
    }

    IHelloService mService;
}
```

Register service
frameworks/base/core/java/android/content/Context.java

```
public static final string HELLO_SERVICE="hello";

//necessary changes to
SystemServerRegistry.java/ContextImpl.java
```

sepolicy entry to
system/sepolicy/private/service_contexts

```
hello                    u:object_r:serial_service:s0
```

API Extension

Consuming Services from Android Apps

```
import android.os.ServiceManager;  // Will only work in AOSP, Stock Apps
import android.os.IHelloService;   // Interface "hidden" in SDK

IHelloService hm =
IHelloService.Stub.asInterface(getSystemService(Context.HELLO_SERVICE));

try {
  hm.sayHello("Hello Android");
  hm.test(10)
}catch(Exception e) {
     //...
}
```

Entries to Android.bp file ==> platform_apis: true,

place app code in Packages/apps/Car
and add entry to suitable mk file, under section PRODUCT_PACKAGES

Car Services & Car APIs

Car Service

- Registered Service : car_service (multiplexing service)
- Code walk through
  - packages/services/Car/car-lib/src/android/car/ICar.aidl
  - packages/services/Car/service/src/com/android/car/ICarImpl.java
- App Developer Reference
  - https://developer.android.com/reference/android/car/classes

Interfaces

- ❑ICarProperty
- ❑ICarAudioService
- ❑ICarEvsService
- ❑IAppFocus
- ❑ICarPackageManager
- ❑ICarDiagnostic
- ❑ICarPower
- ❑ICarProjection
- ❑ICarBluetooth
- ❑ICarMedia
- ❑IInstrumentClusterNavigation
- ❑IClusterHomeService
- ❑ICarTelemetryService
- ❑ICarStorageMonitoring
- ❑ICarDrivingState
- ❑ICarUxRestrictionsManager
- ❑IOccupantAwarenessManager
- ❑ICarOccupantZone
- ❑ICarBugreportService
- ❑ICarUserService
- ❑ICarWatchdogService
- ❑ICarInput
- ❑ICarDevicePolicyService
- ❑IVmsBrokerService

- CarNightService
- FixedActivityService
- GarageModeService
- CarLocationService
- PerUserCarServiceHelper
- CarUserNoticeService
- CarStatsService
- InstrumentClusterService

Underlying Services

- ❑CarPropertyService
- ❑CarAudioService
- ❑CarEvsService
- ❑AppFocusService
- ❑CarPackageManagerService
- ❑CarDiagnosticService
- ❑CarPowerManagementService
- ❑CarProjectionService
- ❑CarBluetoothService
- ❑CarMediaService
- ❑InstrumentClusterService
- ❑ClusterNavigationService
- ❑ClusterHomeService
- ❑CarTelemetryService
- ❑CarStorageMonitoringService
- ❑CarDrivingStateService
- ❑CarUxRestrictionsManagerService
- ❑OccupantAwarenessService
- ❑CarOccupantZoneService
- ❑CarBugreportManagerService
- ❑CarUserService
- ❑CarWatchdogService
- ❑CarInputService
- ❑CarDevicePolicyService
- ❑VmsBrokerService

- CarNightService
- FixedActivityService
- GarageModeService
- CarLocationService
- PerUserCarServiceHelper
- CarUserNoticeService
- CarStatsService

Case Study – VHAL Properties, Services & App Access

Example-1:-

- ❖One VHAL Property ➔ HVAC_FAN_SPEED
- ❖Car Service ➔ Car.HVAC_SERVICE
- ❖Car API ➔ CarPropertyManager, VehiclePropertyIds
- ❖Access in Android app ➔ hvac app, KitchenSink app / Own App

Example-2:-

- ❖VHAL Properties ➔ GEAR_SELECTION, PARKING_BRAKE_ON, PERF_VEHICLE_SPEED
- ❖Car Service ➔ CarDrivingStateService.java
- ❖Car API ➔ ICarDrivingState.getCurrentDrivingState, CarDrivingStateManager.java
- ❖
- ❖Access in Android App ➔ KitchenSink App / Own App

Car Apps

Android HMI & User Interaction

- ❑Stock Apps (part of AOSP) – Platform internal API
- ❑System UI (*https://source.android.com/docs/devices/automotive/hmi/system_ui*)
- ❑Car Launcher (Home Screen)
- ❑Car Settings
- ❑Unbundled Apps - Car UI Library, Dialer, Media and SMS
  (Ref:- *https://source.android.com/docs/devices/automotive/unbundled_apps*)
- ❑Notifications, Set up Notifications, Heads-up Notifications (HUN)
- ❑Resource Overlays - Build Time, Static RROs, Dynamic RROs
  (e.g. HVAC Ref is not a std Android Activity, instead it's an overlay)
  https://source.android.com/docs/devices/automotive/hmi/car_ui
- ❑Users and Accounts

Less Focus in this Session!!

https://developer.android.com/reference/android/car/Car

Accessing VHAL Properties from Android Apps

```
import android.car.VehiclePropertyIds;
import android.car.hardware.VehiclePropertyValue;
import android.car.hardware.property.CarPropertyManager;

private CarPropertyManager mCarPropertyManager;
mCarPropertyManager = (CarProprtyManager)
                  Car.createCar(this).getCarManager(Car.PROPERTY_SERVICE);
int gearSelection =
mCarPropertyManager.getIntProperty(VehiclePropertyIds.GEAR_SELECTION,0);
```

```
In AndroidManifest.xml:-
 <uses-permission android:name="android.car.permission.CAR_POWERTRAIN" />
```

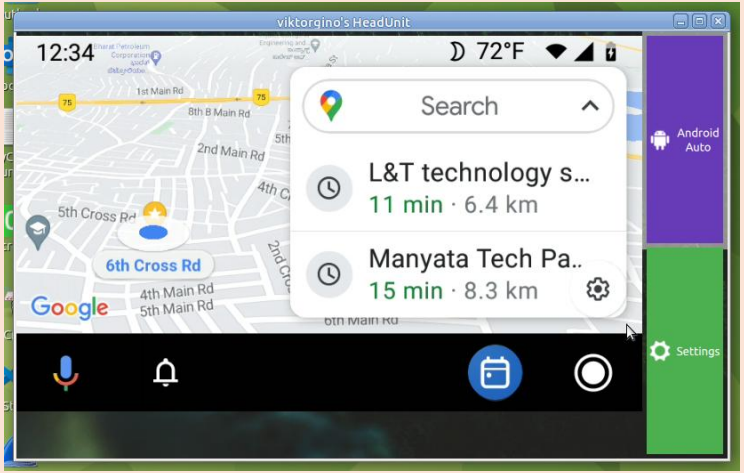
Best Ref App code with max coverage:-
packages/services/Car/tests/EmbeddedKitchenSinkApp

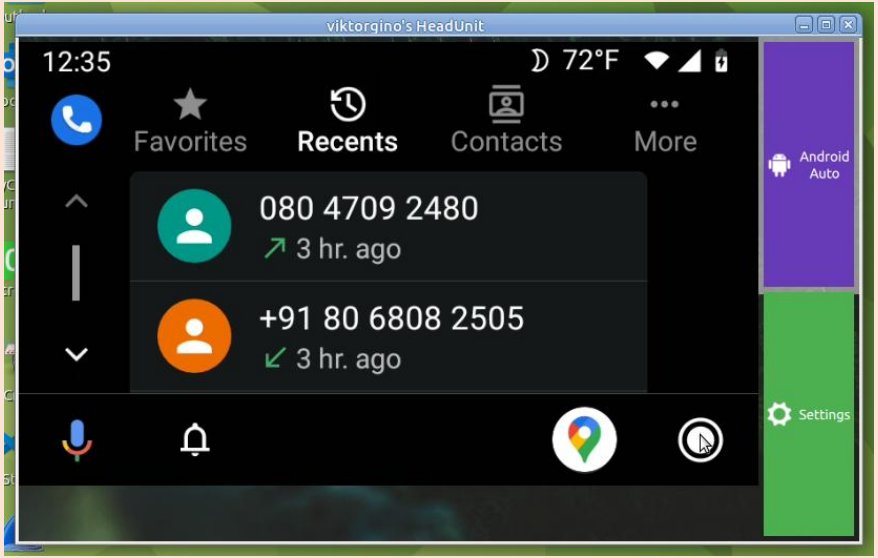
Android Auto

Android Auto – Mirroring Demo

- AA client demo on Linux,

  Ref code:- https://github.com/viktorgino/headunit-desktop





Virtualization

- Android trout target
- Ref:- https://source.android.com/docs/devices/automotive/virtualization/reference_platform
- One of talk in aaosandaaos.github.io

Testing Android Components & Compliance

- ❖CTS - https://source.android.com/docs/compatibility/cts
- ❖ VTS - https://source.android.com/docs/core/tests/vts
- ❖UI Test Automation – UI Automator, Google Mobly, Espresso (https://source.android.com/docs/devices/automotive/tools/ui-frameworks)
- ❖ Complete Automotive Tests (CATBOX) https://source.android.com/docs/devices/automotive/tools/catbox
- ❖ System Performance https://source.android.com/docs/devices/automotive/tools/sys-perf
- ❖Network Simulation https://source.android.com/docs/devices/automotive/tools/network-simulation
- ❖ Fuzz Testing – Enabling Fuzzers (lib Fuzzer) https://source.android.com/docs/devices/automotive/tools/fuzz
- ❖ Spectatio – Automotive Test Framework https://source.android.com/docs/devices/automotive/tools/spectatio

THANK YOU