Building My Product on Android Open Source Project

Android Builders Summit 2015

Rafael Coutinho - Software Engineer

Phi Innovations

Agenda

- Motivation
- Build System Overview
- Simple Build
- Product Customization Structure
- Create My Own Products
- Summary
- Q&A

PhiInnovations

Motivation

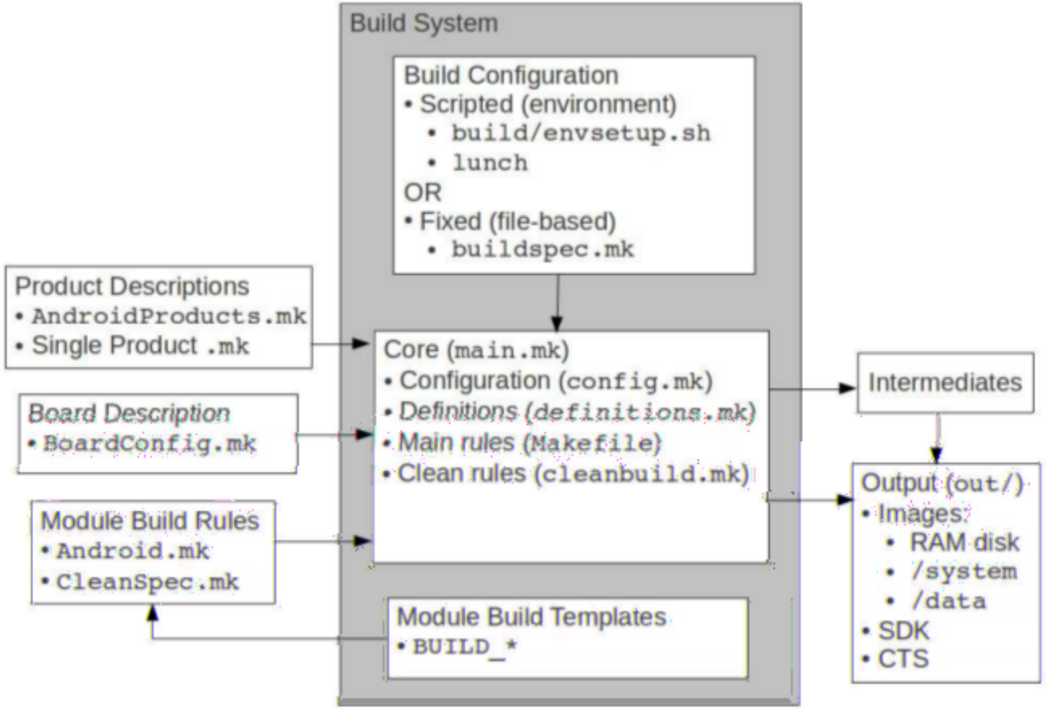
- Looking for AOSP build process documentation we have found it is scarce and what is available is old or cached versions
  - build/core/build-system.html - Starts with "**Status:** *Draft* (as of May 18, 2006)"
  - KAndroid website with cached old version of the Android build
  - Embedded Android book from Karim Yaghmour
  - Free electrons training
- Some ABS previous presentations
  - Usually deep and complete but also complex

[illegible]

Phi Innovations

[illegible]

Android Build System Architecture



Originals at: www.opersys.com/training/embedded-android

PhiInnovations

Simple build

```
$ source build/envsetup.sh
$ lunch
You're building on Linux
Lunch menu... pick a combo:
     1. aosp_arm-eng
     2. aosp_arm64-eng
     3. aosp_mips-eng
Which would you like? [aosp_arm-eng]
$ make -j16
```

Wait…

Simple build… envsetup

envsetup.sh

This script is for setting up the build environment on the current shell

- adding macros
  - type *hmm* to list all macros created
    - godir - move to the directory containing a file
    - m, mm, mmm - macros to start a build with different args
    - cgrep - alias to execute grep on c/c++ files
    - jgrep - alias to execute grep on java files

Simple build… lunch

lunch

- It lists all the combos available in the current environment to be built
  - By following all *vendor/** and *device/** folders looking for the vendorsetup.sh files.
  - vendorsetup.sh files actually executes the add_lunch_combo with parameters

Simple build… combos

- A build combo are combination of a product to build and the variant to use.
  - product (TARGET_PRODUCT)
    - A product defines how the final Android image is, selecting it’s services, initialization, applications to install etc. For example aosp - for emulators.
  - build variant (TARGET_BUILD_VARIANT) select the purpose of this build. The options are:
    - user: Includes modules tagged user, usually used for final release.
    - userdebug: Includes modules tagged user or debug. Usually for platform testing.
    - eng: Includes modules tagged user, debug or eng. Usually for development phase.

Simple build… env variables

lunch sets env variables used by the build.

PATH	$ANDROID_JAVA_TOOLCHAIN:$PATH:$ANDROID_BUILD_PATHS
ANDROID_EABI_TOOLCHAIN	*aosp-root*/prebuilt/linux-x86/toolchain/arm-eabi-4.4.3/bin
ANDROID_TOOLCHAIN	$ANDROID_EABI_TOOLCHAIN
ANDROID_BUILD_TOP	*aosp-root*
ANDROID_PRODUCT_OUT	*aosp-root*/out/target/product/*generic* (has an alias OUT)
TARGET_BUILD_VARIANT	*eng,user,userdebug*
TARGET_BUILD_TYPE	*debug or release*

Simple build… output

The build output is generated in the folder defined by

- ANDROID_PRODUCT_OUT usually *aosp/out*

The output is composed by modules built for the host system and target ones

- The system image is created in target folder under a directory named with the target product name
    - *aosp/out/target/product/aosp/*

Simple build… images

The following files (among others) are created:

- ramdisk.img
  - Contains the root file system of Android, including
    - init.* configuration files
    - default.prop containing the read only properties of this AOSP build
    - /system mounting point
- system.img
  - Contains the components generated by the AOSP build, including
    - framework, applications, daemons

Simple build… images

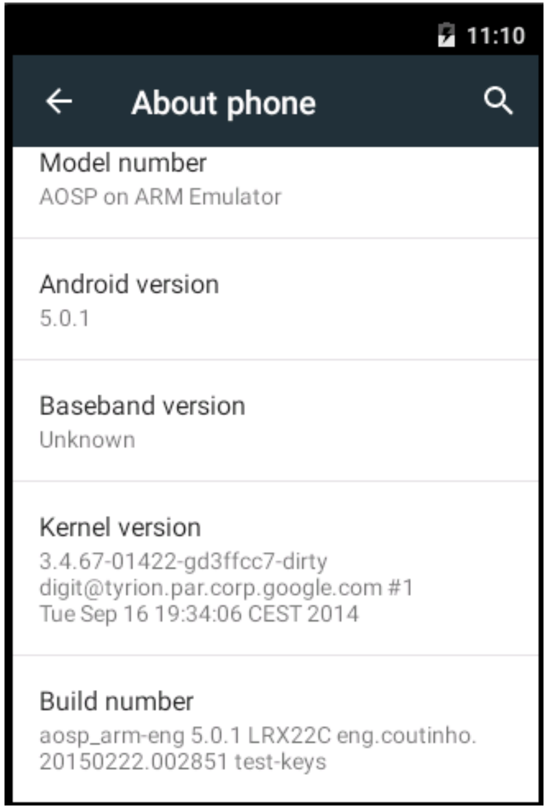
- userdata.img
  - Partition to hold the user data. Usually empty after the build
- recovery.img, ramdisk-recovery.img
  - basic image partition used to recover user data or even the actual system if anything goes wrong.

Simple build… emulator

- Open emulator for testing
  - Build has set up PATH var to point to an emulator executable.

```
emulator -show-kernel -shell
```

  - Model number
  - Build number



Model number
AOSP on ARM Emulator

Android version
5.0.1

Baseband version
Unknown

Kernel version
3.4.67-01422-gd3ffcc7-dirty
digit@tyrion.par.corp.google.com #1
Tue Sep 16 19:34:06 CEST 2014

Build number
aosp_arm-eng 5.0.1 LRX22C eng.coutinho.
20150222.002851 test-keys

[illegible]

PhiInnovations

Product customization structure

Product main makefiles:

- AndroidProducts.mk
- full_<product_name>.mk
- Android.mk
- AndroidBoard.mk
- BoardConfig.mk
- device_<board_name>.mk

PhiInnovations

Android product makefile

- A product makefile (full_<*product_name*>.mk) contains the product properties (name, version etc) and extras like modules/programs or prebuilt files to be included in the build.
- It could include/inherit from other predefined mk files from build/target/product/
- It must define its boards makefile
  - device_<*board_name*>.mk

As reference check build/target/product/

Android product makefile

- Product properties
  - PRODUCT_NAME := aosp_arm
    - This is the name that will appear in the lunch combo option. This must match this product folder under devices folder.
  - PRODUCT_DEVICE := generic
    - This must match the device’s sub directory. TARGET_DEVICE derives from this variable.
  - PRODUCT_MODEL := AOSP on ARM Emulator
    - The end-user-visible name for the end product.

Android product makefile

- Modules to be included

```
PRODUCT_PACKAGES += \
my_own_service_module \
CustomGallery \
lib4mywifi
```

  - Defines which modules, besides any inherited (due to the ‘+’ before the equals), we want to include on the build.
  - It could include libs/apps that are only defined under device/<my_company>/<my_product>.

Android product makefile

- Overriding frameworks/packages config/layout files

```
PRODUCT_PACKAGE_OVERLAYS :=
device/<my_company>/<my_product>/overlay
```

  - Defines a directory that will override the AOSP sources.
  - Avoid changing the *frameworks* folder directly
  - The sub folders must have the same AOSP root structure.

*device/<my_company>/<my_product>/overlay/***frameworks/base/core/res/res/values/config.xml**

Android product makefile

- Common overlayed files

  `frameworks/base/core/res/res/values/`**`config.xml`**

  - config_supportAutoRotation
    - Enables auto rotation support
  - config_longPressOnPowerBehavior
    - defines if pressing power button show a global actions menu, only power off or do nothing.
  - config_shortPressOnPowerBehavior
    - Similar to above but with other options
  - "Documented" here: https://github.com/android/platform_frameworks_base/blob/master/core/res/res/values/config.xml

Android product inheritance

- Inherit to reuse

  $(call inherit-product, $(SRC_TARGET_DIR)/product/full_base.mk)

  - Inheriting from full_base.mk would define most of the needed base configurations.
  - full_base inherits from
    - AllAudio.mk
      - Importing some audios for the system
    - locales_full.mk
      - Get lists of supported languages
    - generic_no_telephony.mk
      - Includes apps like Calendar, Music, Settings
      - Besides includes wpa_supplicant

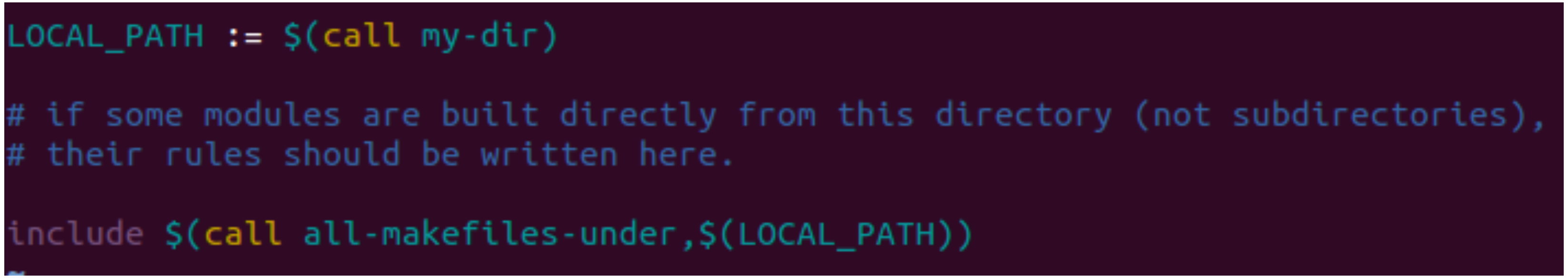
PhiInnovations

Android.mk

If there is any module is defined under

devices/<my_company>/<my_product>

folder to be built, an Android.mk file is needed to call recursively the build on the sub folders.

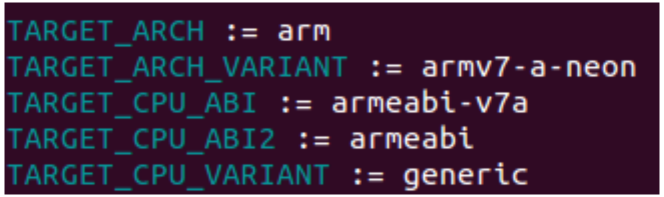


```
LOCAL_PATH := $(call my-dir)

# if some modules are built directly from this directory (not subdirectories),
# their rules should be written here.

include $(call all-makefiles-under,$(LOCAL_PATH))
```

PhiInnovations

BoardConfig.mk

Simple BoardConfig.mk



```
TARGET_ARCH := arm
TARGET_ARCH_VARIANT := armv7-a-neon
TARGET_CPU_ABI := armeabi-v7a
TARGET_CPU_ABI2 := armeabi
TARGET_CPU_VARIANT := generic
```

[redacted]

[redacted]

PhiInnovations [redacted] [redacted]

Creating my own product

- Organization “BossaNova” wants to create an Android product called “Girl Of Ipanema” that runs on the “Tom Jobim” board.
    - This product basically allows a customer to have a customized Android that has info about Girl Of Ipanema song.
- Create the organization folder under *device* folder
- Create the device folder where the product and board files are located
- Customize it

[illegible]

Creating my own product
Folders/files content

- AndroidProducts.mk

```
PRODUCT_MAKEFILES := $(LOCAL_DIR)/full_girlofipanema.mk
```

- device_tomjobim.mk

Includes Emulator's make file

```
include $(SRC_TARGET_DIR)/product/emulator.mk
```

Define this devices overlay directory (Just wallpaper replacement)

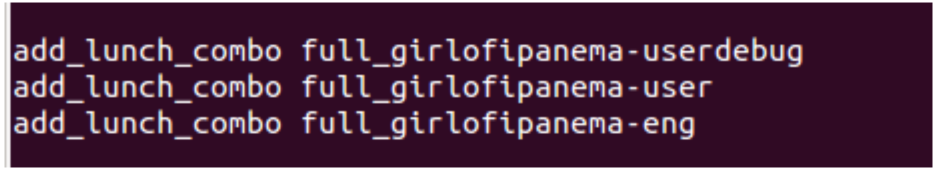
```
DEVICE_PACKAGE_OVERLAYS := device/bossanova/tomjobim/boardoverlays


frameworks/base/core/res/res/drawable-nodpi/default_wallpaper.jpg
```

Creating my own product
Folders/files content

- BoardConfig.mk
  - Pretty much the emulator’s one
  - Reducing the size of userdata partition to 256M

```
BOARD_USERDATAIMAGE_PARTITION_SIZE := 268435456
```

- vendorsetup.sh
  - Added our combos



```
add_lunch_combo full_girlofipanema-userdebug
add_lunch_combo full_girlofipanema-user
add_lunch_combo full_girlofipanema-eng
```

Creating my own product
Folders/files content

- full_girlofipanema.mk

  Define products info (model, name, device…)

  Setting this product overlay defining the launchers wallpaper

  ```
  PRODUCT_PACKAGE_OVERLAYS := device/bossanova/tomjobim/goi_overlays
  ```

  Customized config.xml overlay

  ```
  config_toastDefaultGravity=top|center_horizontal
  ```

  Set the languages to be included in the build

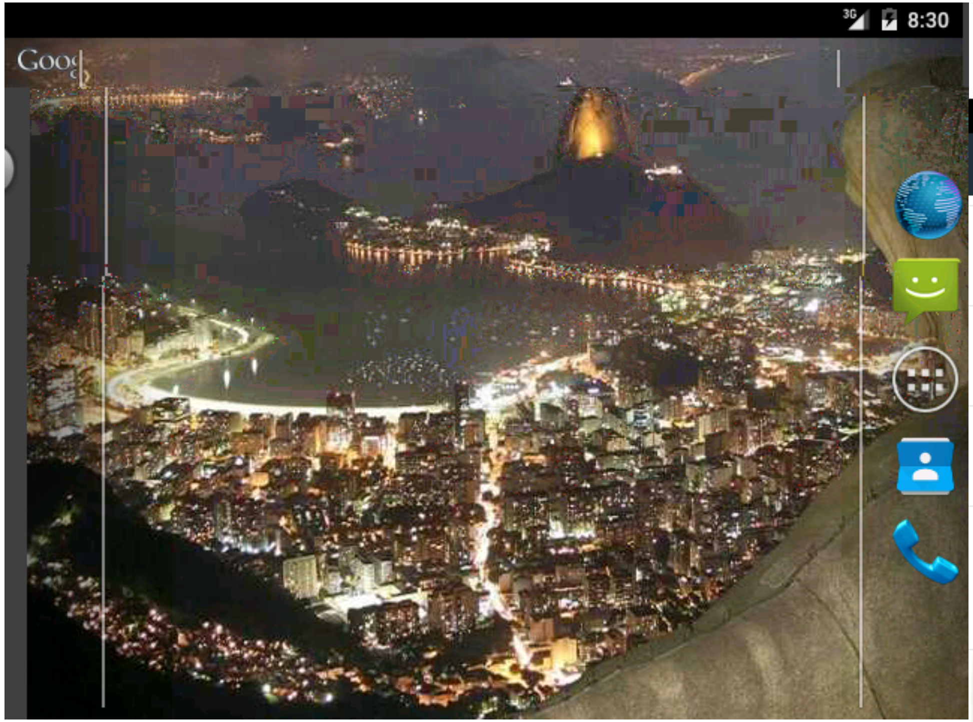
  ```
  PRODUCT_LOCALES := en_US pt_BR
  ```

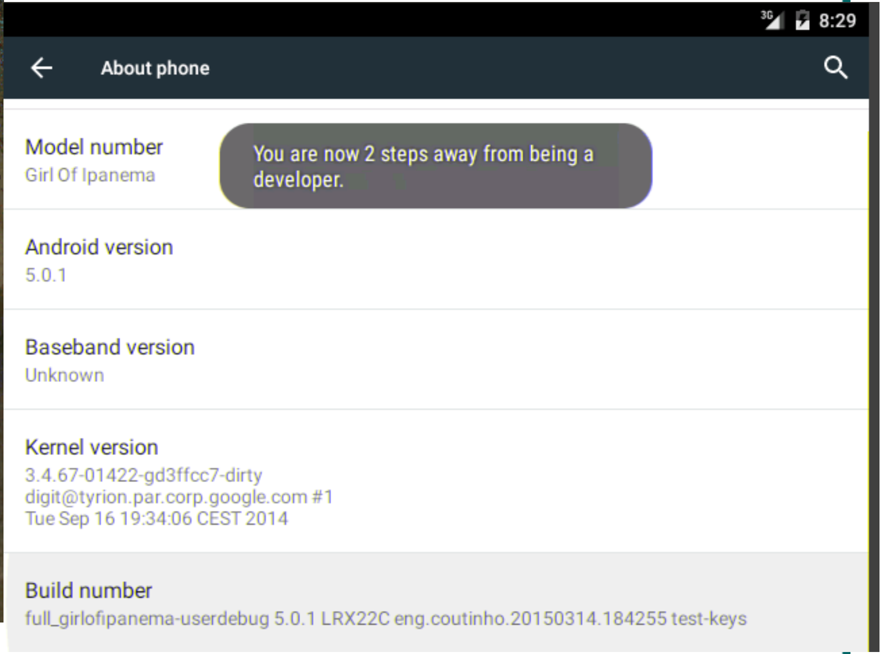
Creating my own product build

Build Girl of Ipanema's Android for Tom Jobim board

```
$ source build/envsetup.sh
$ lunch
You're building on Linux
Lunch menu... pick a combo:
    [..]
    22. full_girlofipanema-userdebug
    23. full_girlofipanema-user
    24. full_girlofipanema-eng
Which would you like? [aosp_arm-eng] 22
make -j16
```

Emulator





About phone

Model number
Girl Of Ipanema

You are now 2 steps away from being a developer.

Android version
5.0.1

Baseband version
Unknown

Kernel version
3.4.67-01422-gd3ffcc7-dirty
digit@tyrion.par.corp.google.com #1
Tue Sep 16 19:34:06 CEST 2014

Build number
full_girlofipanema-userdebug 5.0.1 LRX22C eng.coutinho.20150314.184255 test-keys

Create a second product

- Organization “BossaNova” wants to create another Android product called “One Note Samba” that runs on the “Tom Jobim” board.
- This product comes with a default prebuilt app to play One Note Samba song repeatedly
- This is not a phone but a tablet
- Target market will be Brazil (so default language is portuguese)
- Change the custom boot animation

Creating my own product
Folders/files structure

- Under bossanova
  - Create product mk files
    - full_onenotesamba.mk
  - Update the following files
    - AndroidProducts.mk
    - vendorsetup.sh
  - Create the custom app folder

Creating my own product
Folders/files content

- AndroidProducts.mk

```
PRODUCT_MAKEFILES := \
                     $(LOCAL_DIR)/full_girlofipanema.mk \
                     $(LOCAL_DIR)/full_onenotesamba.mk
```

- vendorsetup.sh

  Added our combos

```
add_lunch_combo full_onenotesamba-userdebug
add_lunch_combo full_onenotesamba-user
add_lunch_combo full_onenotesamba-eng
```

Creating my own product
Folders/files content

- full_onenotesamba.mk

  Add a new package to be included into the build, set the overlay folder and the tablet characteristic

```
PRODUCT_PACKAGES += OneNoteSambaPlayer

PRODUCT_PACKAGE_OVERLAYS := device/bossanova/tomjobim/ons_overlays

PRODUCT_CHARACTERISTICS := tablet

PRODUCT_COPY_FILES += device/bossanova/tomjobim/bootanimation.zip:
system/media/bootanimation.zip
```

Creating my own product
Folders/files content

- full_onenotesamba.mk

  Customized config.xml overlay (setting toast to be in the center)

  ```
  config_toastDefaultGravity=center_vertical|center_horizontal
  ```

  Set portuguese to be the default language

  ```
  PRODUCT_LOCALES := pt_BR en_US
  ```

Adding a prebuilt app

Android.mk

```
LOCAL_PATH := $(call my-dir)
include $(CLEAR_VARS)
# Module name should match apk name to be installed.
LOCAL_MODULE := OneNoteSambaPlayer
LOCAL_SRC_FILES := $(LOCAL_MODULE).apk
LOCAL_MODULE_CLASS := APPS
LOCAL_MODULE_SUFFIX := $(COMMON_ANDROID_PACKAGE_SUFFIX)
LOCAL_CERTIFICATE := PRESIGNED
include $(BUILD_PREBUILT)
```

Summary

- AOSP build system has standards mechanisms to allow customizations
- Following them is important to allow others to expand and customize your device
- However its documentation is mostly by reading the code

References

Embedded Android, Karim J. Yaghmour - www.opersys.com/training/embedded-android

Free electrons training - free-electrons.com/training/android

Jelly Bean Device Porting Walk through, Benjamin Zores, ABS 2013 - speakerdeck.com/gxben/jelly-bean-device-porting-walkthrough

Zip File with Device Folder - phiinnovations.com/files/bossanova.zip

Phi Innovations - www.phiinnovations.com

Q&A*