IT SPA Club

Search this site

Android >

4 - Build System

More details can be found at http://pdk.android.com/online-pdk/guide/build_system.html

Subscribe to posts

PRODUCT_* variable

posted May 28, 2011, 1:58 AM by Ilya Yukhnovskiy **[updated May 28, 2011, 2:01 AM]**

- variables controls what is included into the Android platform build.

```
PRODUCT_*
```

- defines list of packages to be included into the build.

```
PRODUCT_PACKAGES
```

- this variable contains the list files (non executable & non library) which shall be the part of the platform build. Usually it is various settings files, bootstrap files or multimedia files.

```
PRODUCT_COPY_FILES
```

Android.mk structure

posted May 28, 2011, 1:49 AM by Ilya Yukhnovskiy

There is a separate Android.mk for every module in the Android framework.
Android.mk files are searched and included by build system.
Typically Android.mk start with (sets current dir and clears all other LOCAL_* variables and make them having default values)

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

0

Set required LOCAL_ variables:

- list of files to be included into the build
```
LOCAL_SRC_FILES
```
- the module name (as it will appear in the final output, should be unique through the Android.mk files)
```
LOCAL_MODULE
```
- additional folders to search for the header files.
```
LOCAL_C_INCLUDES
```
- names of the libraries to be statically linked with the current module
```
LOCAL_STATIC_LIBRARIES
```
- list the shared libraries to be used by your module when running
```
LOCAL_SHARED_LIBRARIES
```
- the types of the builds into which this module to be included (eng, development, release)
```
LOCAL_MODULE_TAGS
```
- additional flags to be passed to C/C++ compiler
```
LOCAL_CFLAGS
```
- list JAVA libraries your module is using.
```
LOCAL_JAVA_LIBRARIES
```
- name of the application (for $(BUILD_PACKAGE) target)
```
LOCAL_PACKAGE_NAME
```
- certificate name to be used for signing the application.
```
LOCAL_CERTIFICATE
```

Android helper targets – build targets

posted May 28, 2011, 1:30 AM by Ilya Yukhnovskiy

- Various targets are defined by Android to build Android Linux executable, shared dynamic library, static library, java library, Android application and etc.
- The targets are a set of required Makefile instructions based on values of LOCAL_* variables.
- You can make use of a target using
```
include $(BUILD_TARGET_NAME)
```

- Before start setting LOCAL_* variables you need to clear them from previous makefile. Use include $(CLEAR_VARS). This will clear all LOCAL_* variables. This command doesn't affect any of the TARGET_* or PRODUCT_* variables.

- builds Android APK package.

```
$(BUILD_PACKAGE)
```

- builds JAVA library.

```
$(BUILD_JAVA_LIBRARY)
```

- builds native shared library.

```
$(BUILD_SHARED_LIBRARY)
```

- builds native static library.

```
$(BUILD_STATIC_LIBRARY)
```

- builds native executable.

```
$(BUILD_EXECUTABLE)
```

- includes an application package without source codes into the platform output.

```
$(BUILD_PREBUILT)
```

- builds documentation for the JAVA library and includes it into SDK distribution.

```
$(BUILD_DROIDDOC)
```

- builds keymap from the keymap sources.

```
$(BUILD_KEY_CHAR_MAP)
```

- various targets to build various types of the Host executables or libraries.

```
$(BUILD_HOST_*)
```

Android helper targets

posted May 28, 2011, 1:26 AM by Ilya Yukhnovskiy

- returns current dir name in which you Makefile is located.

```
$(call my-dir)
```

- finds all JAVA files available in the folder <src> and all its subfolders. Usually used to form LOCAL_SRC_FILES variable – as it defines the files to be included into the build

```
$(call all-java-files-under, <src>)
```

- the same as previous but finds c files.

```
$(call all-c-files-under, <src>)
```

- the same as previous but it finds *.aidl files to be passed to AIDL tool.

```
$(call all-Iaidl-files-under, <src>)
```

- finds all makefiles available under the <folder>

```
$(call all-makefiles-under, <folder>)
```

- finds the folder where the build output will be stored

```
$(call intermediates-dir-for, <class>,
<app_name>, <host or target>, <common?> )
```

Many other target available. Check them in build/core/definitions.mk file.

Example Android.mk file

posted May 28, 2011, 1:20 AM by Ilya Yukhnovskiy

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

LOCAL_SRC_FILES:= \
…
LOCAL_CFLAGS:= -DLOG_TAG=\"SurfaceFlinger\"
LOCAL_CFLAGS += -DGL_GLEXT_PROTOTYPES
-DEGL_EGLEXT_PROTOTYPES
ifeq ($(TARGET_BOARD_PLATFORM), msm7k)
 LOCAL_CFLAGS += -DDIM_WITH_TEXTURE
Endif
# need "-lrt" on Linux simulator to pick up
clock_gettime
ifeq ($(TARGET_SIMULATOR),true)
 ifeq ($(HOST_OS),linux)
  LOCAL_LDLIBS += -lrt -lpthread
 endif
Endif

LOCAL_SHARED_LIBRARIES := \
 libcutils \
 libpixelflinger \
 libhardware \
 libutils \
 libskia \
 libEGL \
 libGLESv1_CM \
 libbinder \
 libui

LOCAL_C_INCLUDES := \
 $(call include-path-for, corecg graphics)
```

```
LOCAL_C_INCLUDES += hardware/libhardware/modules
/gralloc
LOCAL_MODULE:= libsurfaceflinger
include $(BUILD_SHARED_LIBRARY)
```

TARGET_* - those variables control low level Android build. Controls platform HW features, HAL modules configurations and how the final image is produced
PRODUCT_* - variables containing information on how Android platform source shall be built and what to be included into Android System and RamDisk images. Controls what applications to be included into the distribution, what preloaded files to be the part of the build and etc.
LOCAL_* - various variables used widely across Android.mk file which are defining various features local to the current Android.mk file
ALL_* - internal variable which are produced out of the above 3 types

Android Build System Search Rules

posted May 28, 2011, 1:15 AM by Ilya Yukhnovskiy

BoardConfig.mk & AndroidBoard.mk are looked in the source tree by the following rule:

```
vendor/*/$(TARGET_DEVICE)/BoardConfig.mk
```

TARGET_DEVICE – is your device name specified in choosecombo
Android.mk files are searched on the first 2 levels of hierarchy.
If you need to include Android.mk in the lower level – put Android.mk file in the 2nd level of hierarchy and place

```
include $(call all-subdir-makefiles)
```

Android main build flow

posted May 28, 2011, 1:09 AM by Ilya Yukhnovskiy

The main file from where build starts

```
build/core/main.mk
```

If you use your own makefile – build/core/main.mk must be included into your file
Build/core/main.mk specifies common Android source files set and required libraries

It searches for BoardConfig.mk and AndroidBoard.mk files and includes them into the build
It searches for all Android.mk files available on the 1st and 2nd level of Android platform source tree
It includes all Android.mk files into the build
Start executing the builds.

第 1-7 项，共 7 项