Android Booting

The bootup of an Android system consists of several phases, which are outlined here.

**Contents**

 [hide]

* [1Key bootup components](https://elinux.org/Android_Booting#Key_bootup_components)
  + [1.1Bootloader](https://elinux.org/Android_Booting#Bootloader)
  + [1.2'init'](https://elinux.org/Android_Booting#.27init.27)
    - [1.2.1'init' resources](https://elinux.org/Android_Booting#.27init.27_resources)
* [2Sequence of boot steps on ADP1](https://elinux.org/Android_Booting#Sequence_of_boot_steps_on_ADP1)
  + [2.1firmware](https://elinux.org/Android_Booting#firmware)
  + [2.2kernel](https://elinux.org/Android_Booting#kernel)
  + [2.3user space](https://elinux.org/Android_Booting#user_space)
* [3Tools for analyzing Android Bootup](https://elinux.org/Android_Booting#Tools_for_analyzing_Android_Bootup)
* [4Notes on the Android startup procedure](https://elinux.org/Android_Booting#Notes_on_the_Android_startup_procedure)
  + [4.1Overview](https://elinux.org/Android_Booting#Overview)
  + [4.2strace](https://elinux.org/Android_Booting#strace)
  + [4.3Interaction of different processes on application initialization](https://elinux.org/Android_Booting#Interaction_of_different_processes_on_application_initialization)
  + [4.4Improving Bootup Time presentation](https://elinux.org/Android_Booting#Improving_Bootup_Time_presentation)
* [5News](https://elinux.org/Android_Booting#News)
  + [5.11-second boot of Android](https://elinux.org/Android_Booting#1-second_boot_of_Android)

Key bootup components

**Bootloader**

The first program which runs on any Android system is the bootloader. Technically, the bootloader is outside the realm of Android itself, and is used to do very low-level system initialization, before loading the Linux kernel. The kernel then does the bulk of hardware, driver and file system initialization, before starting up the user-space programs and applications that make up Android.

Often, the first-stage bootloader will provide support for loading recovery images to the system flash, or performing other recovery, update, or debugging tasks.

The bootloader on the ADP1 detects certain keypresses, which can be used to make it load a 'recovery' image (second instance of the kernel and system), or put the phone into a mode where the developer can perform development tasks ('fastboot' mode), such as re-writing flash images, directly downloading and executing an alternate kernel image, etc.

**'init'**

A key component of the Android bootup sequence is the program 'init', which is a specialized program for initializing elements of the Android system. Unlike other Linux systems (embedded or otherwise), Android uses its own initialization program. (Linux desktop systems have historically used some combination of /etc/inittab and sysV init levels - e.g. /etc/rc.d/init.d with symlinks in /etc/rc.d/rc.[2345]). Some embedded Linux systems use simplified forms of these -- such as the init program included in busybox, which processes a limited form of /etc/inittab, or a direct invocation of a shell script or small program to do fixed initialization steps.

The Android 'init' program processes two files, executing the commands it finds in them, called 'init.rc' and 'init.<machine\_name>.rc', where <machine\_name> is the name of the hardware that Android is running on. (Usually, this is a code word. The name of the HTC1 hardware for the ADP1 is 'trout', and the name of the emulator is 'goldfish'.

The 'init.rc' file is intended to provide the generic initialization instructions, while the 'init.<machine\_name>.rc' file is intended to provide the machine-specific initialization instructions.

**'init' resources**

The syntax for these .rc files is documented in a readme file in the source tree. See the [Android init language reference](https://android.googlesource.com/platform/system/core/+/master/init/README.md)

Or, see also: [kandroid copy of old Android PDK](http://www.kandroid.org/online-pdk/guide/bring_up.html)

* Note that the old PDK has been retracted from where it used to found below <http://source.android.com/porting>
  + You may be able to reconstruct it though:
    - You need a local copy of the AOSP sourcetree, and run the usual *build/envsetup.sh* preparation
    - Use *repo init -b* to check out the AOSP sourcetree with a tag around 2.3, then *make sdk sdk\_all*
    - This worked for me, though I used some tag around 4.0.4. Some links had to be fixed in the resulting html output

See also <http://www.androidenea.com/2009/08/init-process-and-initrc.html>

Sequence of boot steps on ADP1

**firmware**

* first-stage bootloader runs
  + it detects if a special key is held, and can launch the recovery image, or the 'fastboot' bootloader
* eventually, a kernel is loaded into RAM (usually with an initrd)
  + normally, this will be the kernel from the 'boot' flash partition.

**kernel**

* the kernel boots
  + core kernel initialization
    - memory and I/O areas are initialized
    - interrupts are started, and the process table is initialized
  + driver initialization
  + kernel daemons (threads) are started
  + root file system is mounted
  + the first user-space process is started
    - usually /init (note that other Linux systems usually start /sbin/init)

**user space**

* the kernel runs /init
  + /init processes /init.rc and /init.<machine\_name>.rc
  + dalvik VM is started (zygote). See [Android Zygote Startup](https://elinux.org/Android_Zygote_Startup)
  + several daemons are started:
    - rild - radio interface link daemon
    - vold - volume daemon (media volumes, as in file systems - nothing to do with audio volume)
* the system\_server starts, and initializes several core services
  + See <http://www.androidenea.com/2009/07/system-server-in-android.html>
  + initalization is done in 2 steps:
    - 1) a library is loaded to initialize interfaces to native services, then
    - 2) java-based core services are initialized in ServerThread::run() in [SystemServer.java](http://android.git.kernel.org/?p=platform/frameworks/base.git;a=blob;f=services/java/com/android/server/SystemServer.java)
* the activity manager starts core applications (which are themselves dalvik applications)
  + com.android.phone - phone application
  + android.process.acore - home (desktop) and a few core apps.
* other processes are also started by /init, somewhere in there:
  + adb
  + mediaserver
  + dbus-daemon
  + akmd

Tools for analyzing Android Bootup

* logcat - see [Android Logging System](https://elinux.org/Android_Logging_System)
  + try this command: 'adb logcat -d -b events | grep "boot"

01-01 00:00:08.396 I/boot\_progress\_start( 754): 12559

01-01 00:00:13.716 I/boot\_progress\_preload\_start( 754): 17879

01-01 00:00:24.380 I/boot\_progress\_preload\_end( 754): 28546

01-01 00:00:25.068 I/boot\_progress\_system\_run( 768): 29230

01-01 00:00:25.536 I/boot\_progress\_pms\_start( 768): 29697

01-01 00:00:25.958 I/boot\_progress\_pms\_system\_scan\_start( 768): 30117

01-01 00:00:40.005 I/boot\_progress\_pms\_data\_scan\_start( 768): 44171

01-01 00:00:45.841 I/boot\_progress\_pms\_scan\_end( 768): 50006

01-01 00:00:46.341 I/boot\_progress\_pms\_ready( 768): 50505

01-01 00:00:49.005 I/boot\_progress\_ams\_ready( 768): 53166

01-01 00:00:52.630 I/boot\_progress\_enable\_screen( 768): 56793

* + or this: 'adb logcat -d | grep preload'

10-15 00:00:17.748 I/Zygote ( 535): ...preloaded 1873 classes in 2438ms.

10-15 00:00:17.764 I/Zygote ( 535): ...preloaded 0 resources in 0ms.

10-15 00:00:17.772 I/Zygote ( 535): ...preloaded 15 resources in 7ms.

* Bootchart - see [Using Bootchart on Android](https://elinux.org/Using_Bootchart_on_Android)
* strace is pretty handy also, to see the timings for system calls from a process as it runs
  + You can use strace as a wrapper for a program in init.rc, and save the results to a file
  + Use -f to follow sub-processes
  + Use -tt to get detailed timestamps for syscalls
  + Use -o to output the data to a file

Here is an example of using strace to follow the startup of zygote, and the apps that are forked from it.

Replace:

service zygote /system/bin/app\_process -Xzygote /system/bin --zygote --start-system-server

with

service zygote /system/xbin/strace -f -tt -o /cache/debug/boot.strace /system/bin/app\_process -Xzygote /system/bin --zygote --start-system-server

Here is some sample data:

$ head boot.strace

571 00:00:11.389939 execve("/system/bin/app\_process", ["/system/bin/app\_process", "-Xzygote", "/system/bin", "--zygote", "--start-system-server"], [/\* 15 vars \*/]) = 0

571 00:00:11.658878 brk(0) = 0x804b000

571 00:00:11.659048 mmap2(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x77f9a000

571 00:00:11.659169 readlink("/proc/self/exe", "/system/bin/app\_process", 4096) = 23

571 00:00:11.659339 access("/etc/ld.so.preload", R\_OK) = 0

571 00:00:11.659440 open("/etc/ld.so.preload", O\_RDONLY) = 3

571 00:00:11.659548 fstat64(0x3, 0x7fa76650) = 0

571 00:00:11.659887 mmap2(NULL, 36, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE, 3, 0) = 0x77f99000

571 00:00:11.659970 close(3) = 0

571 00:00:11.660071 open("/lib/libc\_sse.so", O\_RDONLY) = 3

Please note that writing the strace data takes extra time. For long sequences of very fast syscalls (such as when the timezone file is being read) the overhead of strace itself exaggerates the timings in the trace. Use the timing information with caution.

Notes on the Android startup procedure

**Overview**

See "Android Initialization Process" at: <http://blog.chinaunix.net/u2/85805/showart_1421736.html> (this address is not work), using <http://blog.chinaunix.net/space.php?uid=7788581&do=blog&id=2558375> instead.

**strace**

<http://benno.id.au/blog/2007/11/18/android-runtime-strace>

**Interaction of different processes on application initialization**

Talking about Android Process - <http://blog.csdn.net/mawl2002/archive/2009/06/24/4295905.aspx>

**Improving Bootup Time presentation**

See [Improving Android Boot Time](https://elinux.org/Improving_Android_Boot_Time) - notes and material for a talk at LinuxCon North America, 2010 by Tim Bird

News

**1-second boot of Android**

Ubiquitous Corporation has announced boot of ARM-based Android system in 1 second. Actually, it's more like a suspend and resume than a boot. See <http://www.linuxfordevices.com/c/a/News/Ubiquitous-QuickBoot/?kc=LNXDEVNL032410> [March, 2010]

[Category](https://elinux.org/Special:Categories):

* [Android](https://elinux.org/Category:Android)

Navigation menu

* [Log in](https://elinux.org/index.php?title=Special:UserLogin&returnto=Android+Booting)
* [Request account](https://elinux.org/Special:RequestAccount)
* [Page](https://elinux.org/Android_Booting)
* [Discussion](https://elinux.org/index.php?title=Talk:Android_Booting&action=edit&redlink=1)
* [Read](https://elinux.org/Android_Booting)
* [View source](https://elinux.org/index.php?title=Android_Booting&action=edit)
* [View history](https://elinux.org/index.php?title=Android_Booting&action=history)

**Search**

Top of Form



Bottom of Form

* [Main Page](https://elinux.org/Main_Page)
* [Community portal](https://elinux.org/eLinux.org:Community_portal)
* [Current events](https://elinux.org/eLinux.org:Current_events)
* [Recent changes](https://elinux.org/Special:RecentChanges)
* [Help](https://www.mediawiki.org/wiki/Special:MyLanguage/Help:Contents)
* [Volunteering](https://elinux.org/Sitesupport-url)
* [Bug Tracker](http://bugs.elinux.org/)

Where else to find us

* [Google+ Community](https://plus.google.com/u/0/communities/113852495250648539361)
* [Twitter (@elinux)](https://twitter.com/elinux)
* [#elinux on Freenode](NULL#elinux@irc.freenode.net)
* [Facebook (@elinux.org)](https://www.facebook.com/elinuxorg-750739005020534/)
* [Mailing Lists](http://www.elinux.org/ELinuxWiki:Mailing_List)

Tools

* [What links here](https://elinux.org/Special:WhatLinksHere/Android_Booting)
* [Related changes](https://elinux.org/Special:RecentChangesLinked/Android_Booting)
* [Special pages](https://elinux.org/Special:SpecialPages)
* [Printable version](https://elinux.org/index.php?title=Android_Booting&printable=yes)
* [Permanent link](https://elinux.org/index.php?title=Android_Booting&oldid=459151)
* [Page information](https://elinux.org/index.php?title=Android_Booting&action=info)
* This page was last edited on 28 December 2017, at 01:53.
* Content is available under [a Creative Commons Attribution-ShareAlike 3.0 Unported License](https://creativecommons.org/licenses/by-sa/3.0/) unless otherwise noted.
* [Privacy policy](https://elinux.org/eLinux.org:Privacy_policy)
* [About eLinux.org](https://elinux.org/eLinux.org:About)
* [Disclaimers](https://elinux.org/eLinux.org:General_disclaimer)
* [a Creative Commons Attribution-ShareAlike 3.0 Unported License](https://creativecommons.org/licenses/by-sa/3.0/)
* [Powered by MediaWiki](https://www.mediawiki.org/)

v