# Etc

## Hosts

/system/core/rootdir/hosts

定义的域名规则解析规则

Windows 系统: C:\Windows\System32\drivers\etc\hosts

大多数Unix系统: /etc/hosts, 比如Android/Mac/iPhone/Linux系统

默认内容

127.0.0.1 localhost

::1 ip6-localhost

修改举例

127.0.0.1 localhost

18.2.3.4 www.sky123.com

表示所有域名localhost解析为127.0.0.1，[www.sky123.com解析为18.2.3.4](http://www.sky123.com解析为18.2.3.4)

注：最好是用android studio来写，不然可能文本格式不对，linux系统不能识别

## ld.config.txt

<https://source.android.google.cn/devices/architecture/vndk/linker-namespace>

So的连接器定义

# 系统全局的环境变量init.environ.rc.in

定义了系统全局的环境变量

system/core/rootdir/init.environ.rc.in

# set up the global environment  
on init  
 export ANDROID\_BOOTLOGO 1  
 export ANDROID\_ROOT /system  
 export ANDROID\_ASSETS /system/app  
 export ANDROID\_DATA /data  
 export ANDROID\_STORAGE /storage  
 export EXTERNAL\_STORAGE /sdcard  
 export ASEC\_MOUNTPOINT /mnt/asec  
 export BOOTCLASSPATH %BOOTCLASSPATH%  
 export SYSTEMSERVERCLASSPATH %SYSTEMSERVERCLASSPATH%  
 %EXPORT\_GLOBAL\_ASAN\_OPTIONS%  
 %EXPORT\_GLOBAL\_GCOV\_OPTIONS%

lrwxrwxrwx root root 1970-10-09 20:28 sdcard -> /storage/self/primar

**/storage/self** 是访问 sdcard 的关键路径

BOOTCLASSPATH

/system/framework/core.jar:   
/system/framework/conscrypt.jar:   
/system/framework/okhttp.jar:   
/system/framework/core-junit.jar:   
/system/framework/bouncycastle.jar:   
/system/framework/ext.jar:   
/system/framework/framework.jar:   
/system/framework/framework2.jar:   
/system/framework/telephony-common.jar:   
/system/framework/voip-common.jar:   
/system/framework/mms-common.jar:   
/system/framework/android.policy.jar:   
/system/framework/services.jar:   
/system/framework/apache-xml.jar:   
/system/framework/webviewchromium.jar

# init.rc

主文件

# init.usb.configfs.rc

当sys.usb.config设置为none时，设置sys.usb.ffs.ready为0.

on property:sys.usb.config=none && property:sys.usb.configfs=1  
 write /config/usb\_gadget/g1/UDC "none"  
 stop adbd  
 setprop sys.usb.ffs.ready 0

每次设置的属性有adb时，会先start adbd。  
如sys.usb.config=mtp,adb时,在init.usb.configfs.rc中有如下代码：

on property:sys.usb.config=mtp,adb && property:sys.usb.configfs=1  
 start adbd

然后在system/core/adb/daemon/main.cpp中，会调用usb\_init，进而调用system/core/adb/daemon/usb.cpp的usb\_ffs\_open\_thread()函数中，然后调用init\_functionfs()初始化该属性为1.只有sys.usb.ffs.ready=1时，才能开启adb端口。

1. sys.usb.ffs.mtp.ready  
   当设置为none时，设置sys.usb.ffs.mtp.ready为0.  
   android O新增，当功能为mtp或者ptp时，在 trySetEnableFuntction()函数中会先发送一次广播给mtpReceiver，  
   1) 给数据库添加mtp\_connected;  
   2) 调用mtpServer.configure()，最后调用mtpServer.cpp设置sys.usb.ffs.mtp.ready属性为1,才能响应on property:sys.usb.config=mtp(ptp也是).  
   然后收到kernel发送的uevent后，再发一次广播。

# .zygote

bullhead:/ # getprop | grep zygote

[init.svc.webview\_zygote32]: [running]

[init.svc.zygote]: [running]

[init.svc.zygote\_secondary]: [running]

[ro.boottime.webview\_zygote32]: [12423133466]

[ro.boottime.zygote]: [7285806289]

[ro.boottime.zygote\_secondary]: [7287155768]

[ro.zygote]: [zygote64\_32]

属性 ro.zygote 的值可为：zygote32、zygote64、zygote32\_64、zygote64\_32。

这几个脚本的主要不同：

* init.zygote32.rc：zygote 进程对应的执行程序是 app\_process (纯 32bit 模式)
* init.zygote64.rc：zygote 进程对应的执行程序是 app\_process64 (纯 64bit 模式)
* init.zygote32\_64.rc：启动两个 zygote 进程 (名为 zygote 和 zygote\_secondary)，对应的执行程序分别是 app\_process32 (主模式)、app\_process64。
* init.zygote64\_32.rc：启动两个 zygote 进程 (名为 zygote 和 zygote\_secondary)，对应的执行程序分别是 app\_process64 (主模式)、app\_process32。

# ueventd