Makefile （全局的Makefile）  
bionic （Bionic 含义为仿生，这里面是一些基础的库的源代码）  
bootable （引导加载器）  
build （build 目录中的内容不是目标所用的代码，而是编译和配置所需要的脚本和工具）  
dalvik （JAVA 虚拟机）  
development （程序开发所需要的模板和工具）

external （目标机器使用的一些库）

frameworks （应用程序的框架层）  
hardware （与硬件相关的库）  
packages （Android 的各种应用程序）  
prebuilt （Android 在各种平台下编译的预置脚本）  
recovery （与目标的恢复功能相关）  
system （Android 的底层的一些库）  
out (编译完成后产生的目录，也就是我们移植文件系统需要的目录)

host 目录的结构如下所示：  
out/host/  
|-- common  
| `-- obj （JAVA 库）  
`-- linux-x86  
|-- bin （二进制程序）  
|-- framework （JAVA 库,＊.jar 文件）  
|-- lib （共享库\*.so）  
`-- obj （中间生成的目标文件）  
host 目录是一些在主机上用的工具，有一些是二进制程序,有一些是JAVA 的程序。

target 目录的结构如下所示：  
out/target/  
|-- common  
| |-- R （资源文件）  
| |-- docs  
| `-- obj （目标文件）  
`-- product  
`-- generic  
其中common 目录表示通用的内容，product 中则是针对产品的内容。  
在common 目录的obj 中，包含两个重要的目录：  
APPS 中包含了JAVA 应用程序生成的目标，每个应用程序对应其中一个子目录，将结合每个应用程序的原始文件生成Android 应用程序的APK 包。

 JAVA\_LIBRARIES 中包含了JAVA 的库，每个库对应其中一个子目录。

所以，我们提取文件系统主要是在/out/target/product/generic 目录下，我们可以看到里面有obj 目录，进入obj 目录看看，里面是android 文件系统非常重要的内容：

/obj  
APPS (文件系统下/system/apps 目录下的各种应用程序)  
SHARED\_LIBRARIES （存放所有动态库）  
STATIC\_LIBRARIES（存放所有静态库）  
EXECUTABLES （存放各种可执行文件）