**Daily Note**

**Introduction:**

This is a note used to record mostly everything that I encounter in my daily work, I hope this can eventually improve myself, and help me keep a good habit.

**Draft date:**

July 21, 2014.

**Author:**

Derek ZHUO, intern from USTC.

**Rule:**

Information of the file is intended get into details.

**Other information:**

Content

[1. Perl 4](#_Toc394044469)

[1.1 解释与特性 4](#_Toc394044470)

[2. Shell 5](#_Toc394044471)

[2.1 解释与特性 5](#_Toc394044473)

[3. 正则表达式 6](#_Toc394044474)

[3.1 解释 6](#_Toc394044478)

[3.2 其他名称 6](#_Toc394044479)

[3.3 应用 6](#_Toc394044480)

[4. Sed 7](#_Toc394044481)

[5. grep 8](#_Toc394044483)

[5.1 解释 8](#_Toc394044489)

[5.2 定位 8](#_Toc394044490)

[5.3 应用 8](#_Toc394044491)

[5.4 语法 8](#_Toc394044492)

[6. Git 9](#_Toc394044493)

[8.1 解释 9](#_Toc394044500)

[8.2 分支 9](#_Toc394044501)

[8.3 切换分支 9](#_Toc394044502)

[8.4 删除分支 9](#_Toc394044503)

[8.5 衍合 9](#_Toc394044504)

[7. Linux链接 10](#_Toc394044505)

[9.1 文件存储 10](#_Toc394044507)

[9.2 操作系统读取文件 10](#_Toc394044508)

[9.3 用什么来描述并索引文件 10](#_Toc394044509)

[9.4 Index node的大小 10](#_Toc394044510)

[9.5 目录文件 11](#_Toc394044511)

[9.6 Inode的特殊作用 11](#_Toc394044512)

[9.7 硬链接 11](#_Toc394044513)

[9.8 软链接 11](#_Toc394044514)

[9.9 硬链接与软件链接比较 11](#_Toc394044515)

[9.9.1 硬链接两个限制 11](#_Toc394044516)

[8. tee 12](#_Toc394044517)

[9. xargs 12](#_Toc394044518)

[10. wc 12](#_Toc394044519)

[11. nm 12](#_Toc394044520)

[12. dd 12](#_Toc394044521)

[13. df 12](#_Toc394044522)

[14. 作业控制 12](#_Toc394044523)

[15. Top 12](#_Toc394044524)

[16. Kill -9 12](#_Toc394044525)

[17. Echo中单引号与双引号的区别 12](#_Toc394044526)

[18. Export 12](#_Toc394044527)

[19. Source 13](#_Toc394044528)

[20. Vim 13](#_Toc394044529)

[21. 文件压缩及解压 14](#_Toc394044530)

[22. The UNIX file system structure 15](#_Toc394044531)

[23. Linux目录结构 15](#_Toc394044532)

[24. 上一级目录及上一个当前目录 16](#_Toc394044533)

[1) 返回上一级目录 16](#_Toc394044534)

[2) 返加上一个当前目录 16](#_Toc394044535)

[25. Repo 16](#_Toc394044536)

[26. 文件系统 17](#_Toc394044537)

[26.1 存储价质 17](#_Toc394044555)

[26.2 磁盘的分区 17](#_Toc394044556)

[26.3 挂载 17](#_Toc394044557)

[26.4 文件系统的建立 17](#_Toc394044558)

[26.5 文件系统结构 17](#_Toc394044559)

[26.6 Linux常见的文件系统格式介绍 17](#_Toc394044560)

[26.6.1 Minix 17](#_Toc394044561)

[26.6.2 Ext1 17](#_Toc394044562)

[26.6.3 Ext2 17](#_Toc394044563)

[26.6.4 Ext3 17](#_Toc394044564)

[26.6.5 Ext4 17](#_Toc394044565)

[26.6.6 Swap 17](#_Toc394044566)

[26.7 文件系统工作原理 18](#_Toc394044567)

[27. Vmlinuz 19](#_Toc394044568)

[27.1 解释 19](#_Toc394044570)

[27.2 Vmlinuz的两种建立方式 19](#_Toc394044571)

[28. Image文件 19](#_Toc394044572)

[29. OSD 20](#_Toc394044573)

[30. Putty 20](#_Toc394044574)

1. Perl
   1. 解释与特性

被称之为“一种拥有各种语言功能的梦幻脚本语言”、“Unix中的王牌工具”。

借取了C、sed、shell scripting以前很多其他程序语言的特性，其中最重要的特性是它内部集成了正则表达式的功能，以及巨大的第三方代码库CPAN。

像C一样强大，像awk、sed等脚本描述语言一样方便。

剩下的部分看书！！！

1. Shell
2. 1. 解释与特性
3. 正则表达式

6. 1. 解释

使用单个字符串来描述，匹配一系列符合某个句法规则的字符串；

正则表达式是对字符串操作的**一种逻辑公式**，就是用事先定义好的一些特定字符及这些特定字符的组合，组成一个“规则字符串”，这个规则字符器用来表达对字符串的一种过滤逻辑。

* 1. 其他名称

正规表示法、常规表示法（英文全称是regular expression，在代码中简写为regex，regexp或RE）；

* 1. 应用

很多文本编辑器使用正则表达式来检索、替换那些符合某个模式的文本。

1. Sed
2. 1. 解释
3. 一种在线编辑器，一次处理一行内容；
4. 处理时，把当前处理的行存储在临时缓冲区中，称为“模式空间”，接着用sed命令处理缓冲区中的内容，处理完成后把缓冲区的内容送往屏幕；
5. 接着处理下一行，直到文件结尾。
6. **文件内容并没有改变，除非重定向存储输出；**
7. 主要用来自动编辑一个或者多个文件，**简化对文件的反复操作；**
8. 编写文本转换程序等。
9. grep



14. 1. 解释

global search regular expression and print out the line（全局搜索正则表达式并且打印出来）

* 1. 定位

一种强大的文本搜索工具，而不是文件名搜索工具

* 1. 应用

能够使用正则表达式搜索文本，并且把匹配的行打印出来

* 1. 语法

grep <pattern> <files>

1. Git




7. 1. 解释

一个版本控制系统

* 1. 分支

分支实质上是一个包含所指对象的检验和的（40个字符长度SHA-l字串）文件，所以创建和销毁一个文件就变得很廉价。即，新建一个分支就是向一个文件写入41字节加一个换行符；

Git鼓励频繁使用分支；

* 1. 切换分支

Git会把工作目录的内容恢复为检出某分支时它所指向的那个commit的快照，它会自动添加、删除和修改文件以确保目录的内容和你上次提交时完全一样。

* 1. 删除分支

git branch –d name\_of\_branch

* 1. 衍合

把一个分支合并到另一个分支有两种方法：合并（merge）与衍合（rebase）。

合并把两个分支的最新快照及其公共祖先，共计三方进行合并，其中git可以计算出最合适的共同祖先。

衍合是采用类似打补丁的方式，可以把一个分支的改变在另一个分支重放一遍。

1. Linux链接
2. 1. 文件存储

文件存储在硬盘上，硬盘上最小存储单位叫做“扇区”，每个区存储512个字节，即0.5KB 。

* 1. 操作系统读取文件

一次性连续读取多个扇区，即一次性一个块（由多个扇区组成），是文件存取的最小单位。块大小最常见的是4KB，即连续8个扇区组成一个block。一个文件比较大时，会存放在多个块中。

* 1. 用什么来描述并索引文件

Linux使用inode，即index node来描述一个文件，每个文件均有一个惟一的inode，但是可以有多个文件名同时指向一个inode，即硬链接。

Inode的具体内容如下：

1. 文件的字节数；
2. 文件拥有者的User ID
3. 文件的Group ID
4. 文件的读、写、执行权限
5. 文件的时间戳，共三个，包括ctime（inode上一次变动时间）、mtime（文件上一次变动时间）、atime（文件上一次打开时间）；
6. 链接数，即有多少个文件名指向这个inode
7. 文件数据block的位置

总之，除了文件名以外的所有文件信息，都存放在于index node之中。

* 1. Index node的大小

会消耗硬盘空间，硬盘格式化时，操作系统自动将硬盘格式化为两个区域，分别是：

1. 数据区，用于存放文件数据；
2. Inode区（inode table），存放inode所包含的信息。

每个inode节点的大小，一般是128字节或者256字节，节点总数在格式化的时候就给定，即总数是一定的。

每个inode都有一个号码，操作系统用inode号码来识别不同的文件。

* 1. 目录文件

该文件的content是一系列的目录项，每个目录项由两个部分组成：

1. 所包含文件的文件名
2. 文件名对应的inode号码
   1. Inode的特殊作用

Inode与文件名分离，在Unix/Linux系统有特殊现象

* 有时文件名包含特殊字符，无法正常删除，可以直接删除inode节点，就可以直接删除文件；
* 移动或者重命名文件，只是改变文件名，不影响inode号码；

打开一个文件以后，系统就以inode号码来识别这个文件，不再考虑文件名。因此，通常来说，系统无法从inode号码来得知文件名。

* 1. 硬链接

一般情况下，文件名与inode号码是一一对应的关系，但是Unix/Linux系统允许多个文件名指向同一个inode号码，即可以用不同的文件名访问同样的内容，对文件内容做修改，会影响到所有文件名，但是删除一个文件名，不会影响另一个文件名的访问，即为硬链接。

Ln命令可以创建硬链接，命令格式如下：

Ln 源文件名 目标文件名

执行完命令后，创建了一个“目标文件名”，但是两个文件名均指向同一个 inode号码。

* 1. 软链接

属于特殊情况，但是在Linux是比较经常使用的情况。

文件A与文件B的inode号码虽然不一样，但是文件A的内容是文件B的路径。读取文件A时，系统会自动将访问者导向文件B。因此，无论打开哪一个文件，最终读取的都是文件B，这时，文件A就称为文件B的软链接。这种情况下，文件A依赖于文件B存在，如果删除文件B，访问文件A就会出错。

* 1. 硬链接与软件链接比较
     1. 硬链接两个限制

1. 不允许给目录创建硬链接
2. 只有在同一文件系统中的文件之间才能创建链接
3. 其他：

硬链接指向的是index node，是已存在的文件的另一个名字，修改其中的一个，与其连接的文件同时被修改；

软链接指向的是路径，又叫做符号链接，这个文件包含了另一个文件的路径，可以是任意文件或者目录，还可以链接不同文件系统的文件，可以有效克服硬链接的不足。

1. tee
2. xargs
3. wc
4. nm
5. dd
6. df
7. 作业控制
8. Top
9. Kill -9
10. Echo中单引号与双引号的区别

Echo “you are logged as $USER”发生替换

Echo ‘you are logged as $USER’不发生替换

1. Export
   1. 功能说明

设备或者显示环境变量

在shell中执行程序时，shell会提供一组环境变量。Export可以新增、修改或者删除环境变量，供后续的程序使用，export的效力仅限于该次登录操作。

一个变量创建时，它不会自动被在它之后创建的shell进程所知，而命令export可以向后面的shell传递变量的值。当一个shell脚本调用并执行时，它不会自动得到原来脚本里定义的变量的访问权，除非这些变量已经被显式地设置为可用。Export可以用于传递一个或多个变量的值到任何后继脚本。

1. Source
   1. 解释

Source filename

在当前bash环境中读取并执行指定文件中的指令。通常可以直接用“.”来替代。

1. Vim
2. .bashrc与.vimrc

Linux中，以rc结尾的文件一般是某程序的脚本文件，rc表示“run command”，即运行命令，这些脚本通常在程序启动时被调用。

上面.bashrc是在shell启动时调用的，而.vimrc则是在vim启动时调用的。

该文件一般用来配置程序的相关参数。

1. 文件压缩及解压
   1. ZIP名称的由来

ZIP是一种计算机文件的压缩算法。

* 1. Gzip

GNU zip压缩工具，会创建.gz文件，性能一般。

* 1. Bzip2

比gzip更新、更有效的压缩工具，创建文件.bz2，效率通常是gzip的120-125%

* 1. Lzma

比bzip2更高的压缩率，可加入命令行选项。

* 1. Tar

Abbreviation是tape archive

tar cvf <archive> <files or directories>

--c: create

--v: verbose

--f : file

tar tvf

--t: test

tar xvf <archive>

tar xvf <archive> <files or directories>

* 1. Linux中的同步文件
  2. 差异比较工具diff
  3. 补丁工具patch

1. The UNIX file system structure

It is define by the File system Hierarchy Standard (FHS).

<http://www.pathname.com/fhs/>

1. Linux目录结构

原则：系统并不强制指定，每个发行版的不同安装都可以不一样。但是总体上遵循FHS的标准。

常见的文件系统目录如下：

/ --根目录

/bin/ --基本、必备的系统指令

/boot/ --内核镜像、inittd（init ram disk）及相关配置文件（启动系统所需要的一些核心文件，包括一些链接及镜像文件）

/dev/ --设备文件（外部设备文件）

/etc/ --系统配置文件（存放所有的系统所需要的配置文件及子目录）

/home/ --用户目录

/lib/ --存放系统最基本的动态链接共享库，几乎所有的应用程序都需要用到这些共享库

/lost+found --一般情况下是空的，但是当非正常关机后会有一些文件，以供快速恢复

/mnt --这里面有四个目录，系统提供这些目录是为了让用户临时挂载别的文件系统的，我们可以将光驱挂载在/mnt/cdrom上，然后进入该目录即可以看到光驱里面的内容

/proc --一个虚拟的目录，是系统内存的映射，我们可以通过直接访问这个目录来获取系统信息，这个目录的内容不在硬盘上，而是在内存中，也可以直接修改里面的某些文件。

/root --目录为系统管理员所使用，也称为超级权限的用户主目录

/sbin --存放系统管理员所使用的系统管理程序

/tmp --用来存放一些临时文件

我们日常需要使用的很多应用程序及文件都存放在usr目录下面，如下：

/usr/games

/usr/bin --

/usr/sbin --存放root超级用户使用的管理程序

/usr/doc --linux技术文档

/usr/include --开发和编译应用程序所需要的头文件

/usr/lib --存放一些常用的动态链接共享库及静态档案库

/usr/local --提供给一般的用户，用户可以在这里安装普通的应用软件

/usr/man --帮助文档所存放的目录

/usr/src --Linux开放的源代码就存在这个目录当中

1. 上一级目录及上一个当前目录
2. 返回上一级目录

cd ..

1. 返加上一个当前目录

cd –

1. Repo

在Linux操作系统中，repo=repository，表示安装文件的位置，同时，repo也是Linux环境中开发android主机环境需要的工具包名称，用于进行版本控制。

Repo是google用python脚本写的调用git的脚本，主要用来下载、管理anroid项目的软件苍库。

1. 文件系统















18. 1. 存储价质

硬盘、光盘、软盘、Flash盘、磁带、网络存储设备等

* 1. 磁盘的分区

主要针对大容量的存储设备，主要是指硬盘，需要合理地规划

* 1. 挂载

文件系统只有挂载才能使用

* 1. 文件系统的建立

存储设备建立文件系统的过程，一般也称为格式化或初始化。

* 1. 文件系统结构

文件系统是用来组织与排列文件存取的，所以是可见的，在Linux中文件系统都是树形结构的，例如操作系统安装在一个文件系统中，他表现为由“/”起始的树形结构。

* 1. Linux常见的文件系统格式介绍
     1. Minix

第一个受Linux支持的文件系统，有严重的性能问题，因此出现了第一个扩展文件系统ext1。

* + 1. Ext1

92年4月引入Linux，是第一个使用VFS的文件系统，支持最大的文件系统为2G。

* + 1. Ext2

93年1月引入Linux，支持最大的文件系统是2TB。

* + 1. Ext3

2001年11月引入Linux，是Linux文件系统的重大改进，在某些方面不如对手，但是引入了日志概念，以在系统突然停止时提高文件系统的可靠性，支持从ext2的系统进行就地升级。

* + 1. Ext4

2.6.28是首个稳定的ext4文件系统，在性能、伸缩性及可靠性方面进行了大量的改进，支持1EB的文件系统。

* + 1. Swap

是Linux中一种专门用于交换分区的swap文件系统，Linux是使用这一整个分区作为交换空间，一般这个swap格式的交换分区是主内存的2倍，在内存不够时，系统会将部分数据写到交换分区上。

* 1. 文件系统工作原理

文件系统的工作与操作系统的文件数据有关。现在的操作系统的文件数据除了文件实际内容外，通常含有非常多的属性，例如文件权限与文件属性等。

文件系统通常会把这两部分的数据分别存放在不同的区块：

1. 权限与属性放到inode中；
2. 数据则放到block区块中。

超级区块（super block）会记录整个文件系统的整体信息，包括inode与block的总量、使用量、剩余量等。

Superblock：记录此file system的整体信息，包括inode及block的总量、使用量、剩余量，以及文件系统的格式与相关信息等。

Inode：记录文件的属性，一个文件占用一个inode，同时记录此文件的数据所在的block号码。

Block：实际记录文件的数据内容，若文件太大时，会占用多个block。

Linux采用的是索引式文件系统，根据inode就可以一次性读取多个数据块。而windows的文件系统没有inode，fat把每一个block号码记录在前一个block当中，因此，读到速度会比Linux慢。

1. Vmlinuz
2. 1. 解释

Vmlinuz是可引导的、压缩的内核。“vm”代表“virtual memory”。

Linux支持虚拟内存，能够使用硬盘空间作为虚拟内存，因此得名“vm”。Vmlinuz是可执行的Linux内核，位于/boot/vmlinuz，它一般是一个软链接。

* 1. Vmlinuz的两种建立方式

一是编译内核时通过“make zImage”，然后通过：

“cp /usr/src/linux-2.4/arch/i386/linux/boot/zImage /boot/vmlinuz”产生。

zImage适用于小[内核](http://baike.baidu.com/view/1366.htm" \t "_blank)的情况，它的存在是为了向后的兼容性。

二是[内核](http://baike.baidu.com/view/1366.htm" \t "_blank)编译时通过命令make bzImage创建，然后通过：

“cp /usr/src/linux-2.4/arch/i386/linux/boot/bzImage /boot/vmlinuz”产生。

bzImage是压缩的[内核](http://baike.baidu.com/view/1366.htm" \t "_blank)映像，需要注意，bzImage不是用bzip2压缩的，bzImage中的bz容易引起误解，bz表示“big zImage”。 bzImage中的b是“big”意思。

zImage（vmlinuz）与bzImage（vmlinuz）都是用gzip压缩的，它们不公是一个压缩文件，而且在这两个文件的开头部分都有gzip解压缩代码。所以不能用gunzip或gzip –dc解包vmlinuz。

内核中包含一个微型的gzip用于解压缩内核并引导它。两者的不同之处在于，老的zImage解压缩内核到低端内存，bzImage解压缩内核到高端内存。如果内核比较小，采用两种方式都可以，但如果是大内核，则采用bzImage，不能采用zImage。

Vmlinux是vmlinuz的未压缩版本。

1. Image文件

基本上是去除了多余段的vmlinux ELF文件，同时也去掉了可能已经存在的调试符号。创建该文件的命令如下：

1. xscale\_be-objcopy -O binary -R .note -R .comment -S \
2. vmlinux arch/arm/boot/Image
3. OSD

The abbreviation of on-screen display。即屏幕菜单式调节方式，一般是按Menu键后屏幕弹出的显示器各项调节项目信息的矩形菜单，可通过该菜单对显示器各项工作指标包括色彩、模式、几何形状等进行调整，从而达到最佳的使用状态。

1. Putty

是一个telnet、ssh、rlogin、纯TCP以及串行接口连接软件，也是一个开放源代码软件。

1. 干掉android原生的浏览器

要点：android采用了环境变量在build系统中控制系统安装的原生浏览器，不同的版本有不同的做法。

在2.3（gingerbread）版本中，两个主要的变量是：

1. GRANDFATHERD\_USER\_MODULES
2. PRODUCT\_PACKAGES

前者基本上是固定的，一般会使用后者来进行增减。可以在*build/core/user\_tags.mk*里面或者其他一系列的*.mk*文件中对以上两个变量设置。

在4.2以后，不再采用以上方法，使用变量是：

1. GRANDFATHERED\_ALL\_PREBUILT
2. PRODUCT\_PACKAGES

PRODUCT\_PACKAGES存在于build/target/product/core.mk中，相关代码snippet如下：

PRODUCT\_PACKAGES += \

BasicDreams \

Browser \

DocumentsUI \

DownloadProviderUi \

ExternalStorageProvider \

KeyChain \

PicoTts \

PacProcessor \

ProxyHandler \

SharedStorageBackup \

VpnDialogs

ifneq ($(strip $(TARGET\_LOW\_MEMORY\_DEVICE)), true)

PRODUCT\_PACKAGES += \

Contacts

endif

$(call inherit-product, $(SRC\_TARGET\_DIR)/product/core\_base.mk)

1. 屏蔽android连接功能
2. android红外遥控器键值添加
3. Android编译系统的初始化
4. source build/envsetup.sh
   1. 在device目录中查找并加载那些名为verdorsetup.sh的文件
   2. 加载sdk/bash\_completion目录下的adb.bash文件，用来实现adb的bash completion功能
5. Lunch
   1. 人机交互，根据所选择的版本来初始化编译环境的变量
6. 总体上来说，编译环境初始化后，获得了以下三样东西
   * 1. vendor和device目录下的vendorsetup.sh文件加载到了当前终端
     2. 新增了lunch，mmm，mm，m等命令
     3. 通过执行lunch设置好了TARGET\_PRODUCT、TARGET\_BUILD\_VARIANT、TARGET\_BUILD\_TYPE和TARGET\_BUILD\_APPS等环境变量