写在前面：这是菜鸟作为新手的新手向教程，也算是正儿八经地写的第一篇博文，主要是跟新接触Linux的同学们交流与分享。由于自己接触Linux的时间也不长，纰漏与错误是在所难免的，还请各位大腿们指正。

入门篇

就从最简单的开始吧

cd、ls和man三剑客

刚刚使用Linux，最先接触到的自然是`cd`、`ls`和`man`。

- `cd`
  打开terminal，进入的是你的主目录路径，通过cd命令，可以进入其他目录。格式为`cd [dir]`，dir是目录的路径，可以是绝对路径也可以是相对路径，其中`./`代表当前文件夹，`../`代表上一目录，也就是父文件夹。如果没有参数，回到主目录（也就是`~/`）。
- `ls`
  现在有了`cd`你可以进入你想进入的目录的。可是，如果不知道当前目录下到底有什么，完蛋了_(:3」∠)_。别担心，`ls`来帮你。
  键入`ls [option] [dir]`，你会得到整个dir目录下的未隐藏文件（夹）
  如果你用的是bash的话应该可以用`l`代替`ls`，至于为什么，后面再讲。option是参数选项，如使用`ls -a ./`显示当前目录所有文件，`-a`显示所有文件，`-l`以详细列表显示，`-R`以递归的方式列出指定目录下的所有文件（也就是子目录下的文件也会列出，如果你没有勇气或者神器的话请勿轻易尝试）
- `man`
  好了，如果想知道一个命令更多的信息怎么办，一般又三种方法：
  1. 首先是试一下命令有没有自带的help参数，一般是`--help`、`-h`；
  2. 其次就是`man`，使用命令`man [command]`来打开一个命令的文档（如果有），通过回车或者方向上下就可以读到整个文档了，退出的话按`q`；
  3. 最后，如果以上两种方法失败或者文档根本看不下去（一般是后者），出门右转百度。
     其实如果你忘记了一个命令的参数，`man`或者`--help`往往是很快捷的。

mkdir、touch和cat

`mkdir dir`用来生成一个文件夹，文件夹所在的路径应当存在。
`touch`命令一般用来生成一个文件，`touch path`会生成一个路径为path的空白文件。
`cat`命令会以文本方式读取一个文件，然后把文件中的内容输出到标准输出中，使用cat可以快捷的查看一个文件的内容。

“通行证”sudo

Linux下有些命令或者执行某些动作是需要权限的，这个有点类似于Windows下的管理员权限。如果你得到一个**Permission denied**，那么试试`sudo`命令。
`sudo [command]`会让你输入用户密码，然后以超级用户的权限执行command命令（**superuser do**）。
注意： 这里输入密码时，是不会有任何回显的，不要认为没有成功输入。

一言不合就apt

有时候你在terminal输入一个命令，然后得到这个结果，这就比较尴尬

```
The program 'xx' is currently not installed.
```

这时候就要使用apt系列命令了。如果你使用的不是debian的发行版而是redhat或者centOS之类的，请出门左转百度rpm和yum。

> apt (Advanced Packaging Tools)是Debian及其派生发行版的软件包管理器。APT可以自动下载，配置，安装二进制或者源代码格式的软件包，因此简化了Unix系统上管理软件的过程。[1]

apt系列命令常用的主要有这几个：`apt`、`apt-get`、`apt-cache`和`add-apt-repository`。`apt`和`apt-get`是比较类似的。这些命令运行一般需要权限，`sudo`一下就能解决。
常用的有：

- 使用`apt-get install [package]`或者`apt install [package]`安装一个deb包，这个包必须在本地软件包列表内。
- 使用`apt-get remove [package]`或者`apt remove [package]`卸载软件包
- `apt list`列出所有的软件包
- `apt-get autoremove`或者`apt autoremove`会卸载所有无依赖关系的包（慎用，小概率可能会卸载掉一些必要的软件包，甚至是系统运行必要文件）
- `apt-cache search [regex]` 搜索匹配正则表达式的软件包
- `add-apt-repository`添加一个ppa源
- `apt update`或者`apt-get update`更新本地的软件包列表，一般是在更新ppa源或者其他更新操作之后使用
  更多详情请使用`man`或者`command --help`获得(_^_)

执行可执行文件

对于有执行权限的文件，只需要输入文件的路径就可以执行，其中，如果使用相对路径，当前目录下的可执行文件前面要加上**./**，不然会把它当做命令执行，去**/bin/**、**/usr/bin**、**/usr/local/bin/**等文件夹下查找可执行文件。假如有如下目录：

```
user@computer:~/MyBin$ tree
.
├── a.out
├── ls
└── SubDir
    └── a.out

1 directory, 3 files
```

（注意到我正在MyBin目录下）那么执行`./a.out`执行目录下的a.out文件，执行`SubDir/a.out`或者`./SubDir/a.out`执行目录下SubDir下的a.out文件。如果命令是`./ls`，执行的是MyBin文件夹下的ls文件，而`ls`是执行**/bin**下的ls文件，也就是我们上面提到的`ls`命令。

- 类似的，只要把可执行文件放入**/usr/bin/**、**/usr/local/bin/**目录下，就可以使用文件名直接运行可执行文件。

其他常用命令

其他常用命令有如`mv`（移动文件，重命名文件等）、`cp`（复制文件或者文件夹到指定路径）、`rm`（删除文件、文件夹等）、`rmdir`（删除空文件夹）等等，道理是相似的，就不再一一赘述。

入门进阶篇

gcc、g++和gdb —— C/C++一条龙

gcc

虽然背负GNU Compile Collection之名gcc并不仅仅可以编译C语言，但是这里我们就把它当做C语言的编译器好了。g++以及clang和clang++都是类似的，就不重复了。
gcc/g++ 有些参数是比较常见的：

- -c 只做编译，不做链接
- -o 生成目标，后面的路径会生成目标文件
- -std=c[++]**|gnu** 规定ISO标准，如-std=c99就是使用c99标准编译文件，默认的话C语言是c89，C++是c++98/03
- -g 插入调试信息，请配合gdb食用
- -W** 警告某种规则，常见是-Wall，打开所有警告，可用值可以使用可以百度或者查看官方文档
- -Werror 把警告视为错误
- -Wno-** 关闭某种警告，可用值与上面相同
- -I 将路径添加到预处理默认路径
- -L 将路径添加到链接库默认路径

- -l 链接的时候链接库默认路径下的库文件，如-lgtest会链接**/usr/lib**或者**/usr/local/lib**下的libgtest.a文件或许libgtest.so文件（静态库或者动态库）
- -pthread 多线程代码使用

gdb

如果你编译代码的时候有加-g参数，那么你就可以使用gdb调试你的代码了
使用gdb <program>可以对可执行文件进行调试。
常用命令列表有（括号后面是简短代替）：

- `list(l)`： 后面可以加入函数名、行号、文件名:行号、起始行号, 结束行号等参数， 打印出源代码。
- `breakpoint(b)`：后面跟行号，函数名添加断点。可以使用`clear`、`delete`、`disable`和`enable`后加断点号分别清除、删除、停用`启用相应的断电。
- `run(r)`：使用run开始运行你的程序，后面可以加参数，这些参数就是命令行参数
- `continue(c)`：回复被断点中断的程序，直至下一个断点，后面也可以跟一个数字表示跳过相应个数的断点。
- `step(s)`：逐步调试，遇到函数会进入函数内部，后面可以跟数字表示重复step次数
- `next(n)`：与step相似，但是不会进入函数
- `print(p)`：打印变量，或者一个可以确定值的表达式，后面跟变量名或者表达式
- `watch(w)`：后面跟变量名，监视变量，变量一旦变化打印新旧值
- `examine(x)`：查看内存

gdb的调试跟图形化界面调试的方法还是比较相似的，大家也可以做一些类比帮助理解。

- 关于gdb的内容，主要参考了这篇博客（http://blog.csdn.net/lyjtynet/article/details/4057723），内容比较长，但是很详细。感谢作者**lyjtynet**。

make大法好

如果有一个很大的项目的代码需要编译，make是节省时间的好工具，关于makefile的使用方法请参阅这篇Makefile的使用博客。感谢作者**DaddyTrap**

grep、locate和find查找三神器

- 如果想查找文档中的某些信息，使用grep绝对是个好选择
`grep -i regex file`会将file中匹配了正则表达式的**行**输出，请注意这里是**行**为单位的
- find可以在特定文件夹中搜索匹配规则的文件，规则比较复杂，常用的是`-name`参数，可以查找名字匹配正则表达式的文件（包括文件夹，毕竟Unix一切皆文件嘛，文件夹也不例外）
- locate命令可能需要通过`sudo apt install locate`安装，通过`locate name`可以定位文件位置，也就是查找文件啦

不如chmod

Linux下的文件是有权限控制的，使用`ls -l`可以看到每行最前面有**-rwxrwxr-x**之类的，这是描述用于文件的属性与权限。第一个字符代表文件是否为目录，如果是则为**d**，否为**-**；接下来九个字符每三个为一组，分别代表用户，组，和其他人的权限，三个字符分别代表读写执行的权限，即**r**、**w**和**x**。一个二进制文件必须要有可执行权限才能运行。
为了改变文件的属性，可以使用命令`chmod [option] mode ... file ...`来更改文件权限。
mode是指文件的权限更改模式，主要有以下两种类型：

1. 一个三位的八进制数。
   这个八进制数从高位到低位分别代表用户、组和其他人的权限模式。每个位是1、2、4或者它们的和，1代表执行，2代表写，4代表读。如**7 = 1 + 2 + 4**，所以**7**代表有读写执行的权限，而**5 = 1 + 4**，所以**5**代表有读和执行，但是没有写的权限。如果将这个三位八进制数写成二进制，每一位**0**正好对应前文说道的后九位的**-**，而每一位**1**对应字符。如数字0775写成二进制是111 111 101，对应的就是rwx rwx r-x。正所谓一言不合777，`chmod 777 path`将文件的所有权限给了所有人，是很不安全的行为。~~（但是我喜欢你管不着）~~
2. **[a|u|g|o][+|=|-][rwx]**的格式。
   其中a代表全部，u代表用户，g代表用户组，o代表其他，+是加上权限，=是权限改为，-是除去权限。如`chmod u+x a+r a.out`是使得用户可执行，全部可读文件a.out

---

入门进阶进阶篇

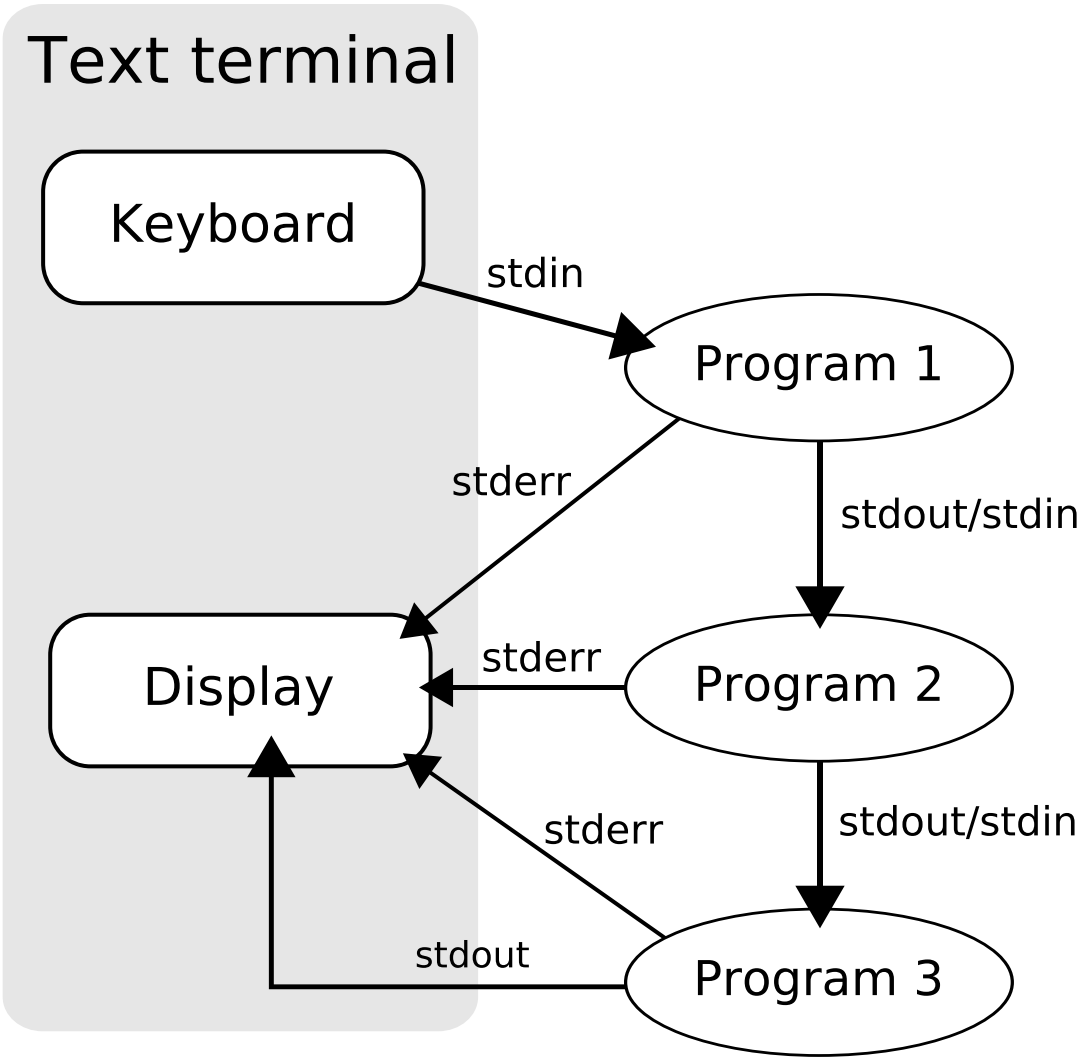
到这里为止，terminal的使用已经可以满足平时的很多需求了，但是其实还是有很多神器没有以正确的方式使用

Linux管道与（真）grep

先说管道。中文维基上是这样将的

> 一个由标准输入输出链接起来的进程集合，所以每一个进程的输出（stdout）被直接作为下一个进程的输入（stdin）

也可由一张图简单认识 [^2]



使用管道操作符 `|` 将多个命令连接起来，将前一个命令的输出作为当前命令的输入，就像管子一样字符串在命令之间流动。

介绍完管道正式介绍grep的正确使用*姿势*
先设想这样的情况，文件夹下面有数百个文件，你只想看某些特定文件的详细情况，结果使用 `ls -l` 出来好几十页，终端跑得~~比某些记者还~~快，还得慢慢翻，有点蛋疼。问题怎么办，神器grep登场。
前面介绍到grep可以匹配正则表达式，只要把标准输出的内容输入到grep，就可以只打印出包含你想要的信息的函数了。同样的，如果要快速的查找文件中是否包含某写信息，那么使用cat + grep可以快速的实现。这几个是我常用的命令。

```
apt list | grep ${package}\ \\[installed\\]   #快速查找软件包是否安装
cat file | grep regx   #查找文件中匹配正则表达式的行，快，但是缺点是只能匹配一行，其实可以使用sed
ll -R dir | grep regx #确定目录下（包括所有子目录）是否含有匹配正则表达式的文件，个人觉得比find稍快，但是不能定位路径
```

使用管道+grep，可以极大减少无用信息的输出，极大地提高你的工作效率
当然，还有 `less` 命令可以使得长长的输出逐页输出，翻页随心所欲。

bash的初始化脚本 [^3]

严格意义上说这是题外话，但是我觉得跟大家分享一下还是有必要的
一般来说有两个文件是在bash启动的过程中比较重要的，一个是/etc/bash.bashrc，另一个是~/.bashrc，主要都是bash的配置命令。
文件/etc/bash.bashrc会在系统启动时执行，而~/.bashrc会在用户打开bash的时候执行。如果你在linux或者Windows中装过JVM或者JDK，一定忘不了要设置环境变量，而在Linux下，一般是在写在/etc/bash.bashrc中，这样开机的时候就会把
JDK或者JVM的目录添加到$PATH中.

~~彩蛋——给你的同学小小的恶作剧~~

有一对命令`alias`和`unalias`，用于生成命令的宏，比如`alias l="ls"`会使得在终端的命令`l`（不管存不存在）变成`ls`。
~~好了，文件.bashrc的权限是 rw r r ，也就是说不需要`sudo`就可以修改。（简写的笑脸）~~
~~给你的同学的.bashrc最后面加上`alias ls=""`，他就没有办法用命令`ls`了，哈哈哈。多加几句，把主要命令都给废了，然后保存，完美。~~
我什么都没说，你什么都没听到。

---

1. 摘自中文维基百科zh.wikipedia.org
   [^2]: 图源自中文维基百科zh.wikipedia.org
   [^3]: 参考了这篇博客（http://www.cnblogs.com/lege/p/4235663.html）↩