Chapter 6

第六章

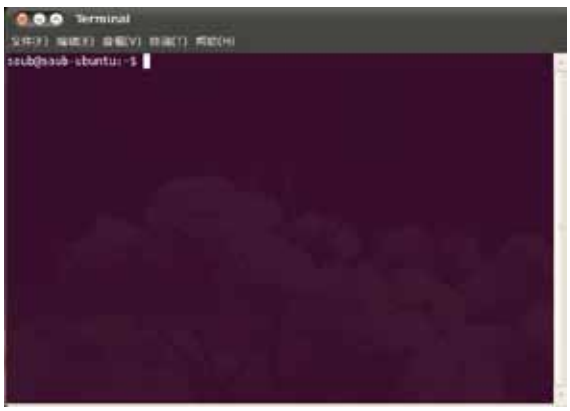
shell

6.1 这就是 Shell

能够有一个理解你，信任你的主人，对于一个 ubuntu 系统来说是一件非常幸福的事情。

虽然我们 linux 的图形界面已经比较先进了，绝大多数操作都完全可以用图形界面来完成。但是就像吃过麦当劳肯德基不等于吃过西餐一样，连刀叉都不知道怎么用的你也好意思说你会吃西餐？连终端都没进去过你同样也不好意思说你会用 Linux 吧？或许有人不是很赞同这个说法，不过我的主人是这么认为的。主人现在在周围的同学同事里面已经俨然是一位 linux 大牛了，能够把这个别人见都没见过的系统耍的上下翻飞（比如 3D 桌面转屏幕的时候），真的是很拉风的。可是他并没有停留在这种表面上的绚丽。如果仅仅会用图形界面，不会几行命令，那作为一个号称是会用 linux 的大牛来说是肯定很没面子的，所以主人为了进一步巩固 linux 大牛的地位，决定去学习 linux 的命令行。

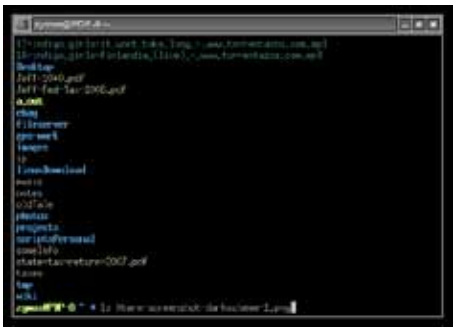
命令行并不神秘，打开应用程序 →附件 →终端，就看到了。

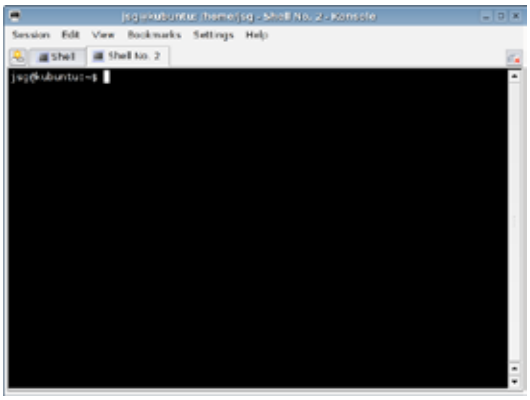


在这里，你可以近距离的跟我交流。我们操作系统是很希望用户能够和我们使用命令行来交流的，向朋友般倾诉，感受彼此的心声，而且还低碳环保，节省能源（省 cpu 和内存啊！）。用户就相当于老板，使用图形界面的用户和使用字符界面的用户是两种完全不同的老板。前者高高在上，拒人千里之外，就会比划，有事都不直说。比如他用手一指桌面上的文件，冲你“恩～”一声，你就得明白，他是想让你

选中这个文件，你就赶紧过去把文件拿起来。他要是指着文件冲你“恩，恩～”两声，你就得明白他是让你打开文件。（这里你还得注意，他“恩”的那两声之间的间隔长短，要把握好分寸。要是间隔很短，那叫“双恩”，要是间隔长，那叫“恩”两下，对应的操作是不一样的。）

他指着电灯开关的按钮冲你“恩”一下，那就是让你过去把电灯打开，他要是用手冲着桌面上的一堆东西指一圈，然后嘴里拉长声的“恩～～～～～～～～”一下，那就是让你把这些东西全都选中。给这种老板打工，需要很大忍耐力。要是遇到第二种老板就要好的多，这种老板，平易近人，没有架子，有事会跟你平等的交流：小笨啊，你列个表，看看我这目录下都有什么东西啊；小兔啊，3 分钟后下班啦，到时候记得关机啊；小笨啊，咱挺寂寞的，放个片子看看吧，就放那个 wall-E.avi 吧。这样的老板，跟你工作起来，那才是心旷神怡。用命令行的用户，就相当于这样的老板，舒服。



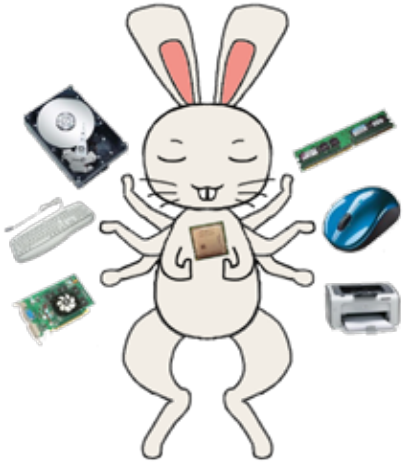


看到主人用鼠标指着终端，“恩”了一声，我就知道，该去把 gnome 终端叫起来了。gnome 终端就像查皮那里的命令提示符一样，是个虚拟终端，不过比那个可先进多了。所谓虚拟终端，是相对于真正的，黑漆漆的，按 alt-ctrl-f1 出来的那个终端来说的。虚拟终端的工作是在桌面上申请一个窗口，然后在里面模拟显示出一个黑漆漆的界面。（其实仔细看 gnome 终端还算漂亮啦，还能背景半透明呢。）除了 gnome 终端之外，还有很多用来输入命令的终端，像 konsole、rxvt、xterm 等等，他们都是虚拟终端软件。你可以多装几个虚拟终端对比对比，看哪个好用。反正无外乎都是一些黑漆漆，白惨惨，绿幽幽的窗口，在里面可以输入命令而已。

有的人可能会有顾虑：虚拟终端，那是不是里面执行的命令不是真正的 linux 命令呢？这些不一样的虚拟终端对命令的执行结果都一样么？

不一样？错！一样？还是错！

他们根本就不会执行命令，他们只是个界面而已，都是浮云。真正处理命令的不是他们，是 shell。在里面运行的命令并不归虚拟终端软件去解析，而是找到专门的命令行的程序，这个程序就叫做 bash，这个咱们待会再说。



那这个 shell 又是个什么东西呢？

shell，顾名思义，就是个壳！怎么叫壳呢？咱们慢慢说。我们 linux 是个内核，我这个内核是可以做很多事情的，整个电脑的硬件都归我管。我和硬件的关系，大概就是这样：

很威武吧。好，我现在管着所有硬件，那应该用这些硬件干点什么呢？我不知道，因为没有人给我下命令啊。这下命令的就是人类用户，就像我的主人。那么主人来了之后，大概就是鸡同鸭讲了。

哎，没办法。你们人类那么多种语言，我……，我却是一句都听不懂啊！你们人类呢？也一样，就是听不懂我说的机器语言，那怎么办？就得有个翻译，同声传译嘛。有了之后就好多了。

看，这个 shell 的作用是不是就像个壳子罩着我，让我能够和主人更好的交流呢？ shell 的名字就是这么来的。他是我这个内核用来和他们人类交流的一个外壳。

shell 的作用我们知道了，那么他的本质呢？就是个 /bin/ 目录下的一个二进制程序。比如我们 ubuntu，用户默认的 shell 是 bash，就是 /bin/bash 这个二进制文件。当主人打开终端的时候，终端程序（比如我们这里是 gnome terminal，为了省事，以后我我们就叫他 G 终端吧。）就会进入工作间，先绘制出一个界面，（当然，这需要图形部门的配合），然后就去找你这个用户的 shell 程序。那么用户的 shell 程序是什么呢？这个在 /etc/passwd 文件里有记载，G 终端找到 passwd 文件里对应当前用户的一行，比如我们这个就是这样：

```
lanwoniu:x:1000:1000:lanwoniu,,,:/home/lanwoniu:/bin/bash
```

这一行的最后一段就说明了这个用户的默认 shell 是 /bin/bash。于是 G 终端就去叫醒 bash，bash 起床，通过 G 终端来跟主人用文字交流。那么最先说的一句话大约就是“你好”，当然不会这么不专业，这句话用 bash 专业的语言说出来就是这样：

```
lanwoniu@lanwoniu-ubuntu:~$
```

这行是什么意思呢？首先最前面的，也就是 @ 之前，是当前用户名称。@ 后面是计算机名，这两个都好理解。“：”后边是当前所在目录，就是当前输入命令的人所在的位置。”~”代表用户的家目录，也就是 /home/< 用户名 > 这个位置。$ 则是命令提示符，$ 后面，就可以输入命令了。顺便说说，普通用户的提示符是 $，root 的提示符是 #，不过我们 ubuntu 不能用 root 登录，所以基本你也看不到 # 提示符了。

正说着，主人已经开始敲命令玩了。什么 ls，free，top，fdisk 等等，挨个试验。于是工

作间里也开始忙碌了起来。你可能会以为 bash 会在主人的指挥下跑来跑去，执行各种操作？呵呵，其实完全不是那么回事，bash 只是作为一个命令的传达者而已，真正干活的是那些命令们。也就是 ls，free 这些家伙。这些所谓的命令，其实都是一个一个的小程序，或者说一个个小的软件而已。就跟狐狸妹妹，OO 老先生一样，只不过比他们小巧很多。如果你愿意，也可以把 firefox 看作一个上网用的图形化界面的命令，为了好说，咱们以后管这些家伙叫做命令程序吧。当用户输入命令，比如 ls 的时候。首先 ls 这两个字符是被传给 bash 的，bash 怎么处理呢？首先 bash 要看输入的字符是不是自己的什么关键字，比如 for、history 之类的，如果是的话，就归 bash 来处理了，如果不是，那就说明主人是要找个命令程序，bash 就要负责去找到主人想要的这个程序，并且叫他起床干活。那 bash 去哪里找呢？不知道你知不知道有个叫做环境变量的东西？跟 windows 的那个环境变量差不多，其中有个环境变量叫做 PATH，这里面记录着 bash 去找程序的路径。如果你想看看 PATH 到底是什么，运行 echo $PATH 就可以了。会得到类似这样的输出：

```
lanwoniu@lanwoniu-ubuntu:~$  echo  $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games
```

当主人运行一个命令，比如 ls，bash 就对照着 PATH 里面的设置，先去 /usr/local/sbin 房间里面找，敲开门，客气的问：请问 ls 是住这屋么？ 等了半天，除了墙角的蜘蛛网上那八条腿的小家伙寂寞的弹了几下琴弦之外，再也没有活物给他任何回应，于是 bash 意识到这屋没人，赶快去下一屋。到了 /usr/local/bin，依然是很礼貌的敲开门问候，里面只有一位主人前几天安装的叫做 Maya 的软件，只听那位叽里咕噜的说了几句 2012 啥的玛雅语，bash 也听不懂，不过反正他不是 ls 就对了，赶紧去下一间。推开 /usr/sbin，这回里面很热闹，而且都是重要人物——管用户创建的 useradd、每天启动必备的 gdm、负责跟通过网络跟查皮共享文件的 smbd 和 nmbd 等等。一听 bash 来找 ls，useradd 没好气的说：“呀，ls 怎么可能在我们这里呢？我们这里都是管理级的程序，那个 ls 是谁都能运行的，怎么会在 sbin 里？你得去 bin 里面找阿。”，bash 只好客气的退了出去，继续去找 /usr/bin,/sbin/, /bin，终于在 /bin 里面找到了 ls，于是赶紧叫醒 ls，让他去干活。至此，bash 的任务也就结束了，他就回去等待主人的下一个命令就好了。等到下一个命令来了，bash 就再挨个去各个屋里找主人要的命令。

需要注意的是，bash 是严格按照 $PATH 所记录的位置去找命令程序的，一般这个 PATH 里面并不包含当前目录，也就是“./”。所以 echo 并不会在当前目录查找命令程序。有的人可能当前在 /home/bob/bin/ 目录下，这个目录下有一个 kiss 程序，于是他直接运行 kiss，结果 bash 告诉他没找到这么个命令，他就很困惑，还找来 ls 来证明：这明明就在眼前的东西，你怎么就告诉我没有呢？

其实不是没有，是你没让 bash 在眼前这个目录找嘛。那么要想运行一个程序就必须把它放在 /bin,/usr/bin 之类的这些地方么？当然也不是，当你不告诉 bash 路径的时候，bash 是去 PATH 变量里找的，如果你自己知道你要运行的程序不在 PATH 变量所记载的目录里，那就直接了当的告诉 bash 你要的程序在哪，bash 就能找到了。比如你运行 /home/bob/bin/kiss 命令，就是告诉 bash，去 /home/bob/bin/ 目录下找一个叫做 kiss 的家伙出来干活。如果你像刚才说的

似的现在就在这个目录下，就可以把刚才那很长的地址简单描述为“当前目录下的 kiss 程序”，当然你跟 bash 说这句中国话是不可能的，那么当前目录怎么表示呢？很简单——“./” 所以运行当前目录下的 kiss 程序就运行“./kiss”就好了。说了这么多，是不是不会再有人误解“./”是运行的意思了呢？（记得 ./configure 吧，编译软件的三部曲之一，就是运行“当前目录下的 configure 脚本”的意思。）

主人敲打着 ifconfig 命令来查看网络链接状况，忽然想起这个命令应该还能改变自己的 IP，于是想试试，可是又忘了具体是怎么用的了，怎么办呢？ bash 告诉他：别着急，我给你找个人问问。这个人，纯爷们！然后 bash 扭头冲着硬盘里喊：Hey man，你的出来，说说这是怎么个意思的干活。随着咔咔嚓嚓一声硬盘响，只见内存中走来一人，人高马大，膀大腰圆，扇子面的身材，胳膊跟大腿一样粗，这就是 bash 说的那个 man。这纯爷们说出话来如同打个炸雷一样：嘿！你好啊。洒家我是专职命令解说员，你有什么想知道的吗？

主人输入了命令 man ifconfig。意思就是问 man，这个 ifconfig 是怎么用法。man 仰天大笑一声：“嚯哈哈哈，要说这个 ifconfig 嘛……，不难，听我慢慢地道来！”

既然说到了 ifconfig，那就说说他的用法。这个命令是用来配置网络的。如果你想查看现在的网络链接状况，就找他就好了。只要运行 ifconfig，就可以得到类似如下的内容：

```
lanwoniu@lanwoniu-ubuntu:~$ ifconfig
eth0      Link encap:以太网  硬件地址 00:1f:d0:8b:9c:10
          inet 地址:192.168.1.100 广播:192.168.1.255 掩码:255.255.255.0
          inet6 地址: fe80::21f:d0ff:fe8b:9c10/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  跃点数:1
          接收数据包:34303 错误:0 丢弃:0 过载:0 帧数:0
          发送数据包:26207 错误:0 丢弃:0 过载:0 载波:0
          碰撞:0 发送队列长度:1000
          接收字节:40115963 (40.1 MB)  发送字节:2356235 (2.3 MB)
          中断:29 基本地址:0x4000

lo        Link encap:本地环回
          inet 地址:127.0.0.1  掩码:255.0.0.0
          inet6 地址: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  跃点数:1
          接收数据包:215 错误:0 丢弃:0 过载:0 帧数:0
          发送数据包:215 错误:0 丢弃:0 过载:0 载波:0
          碰撞:0 发送队列长度:0
     接收字节:15116 (15.1 KB)  发送字节:15116 (15.1 KB)
```

基本都是中文，也就不用解释了，如果不知道什么是 ip 地址，不知道什么是掩码的，那先补习一下网络基础。如果你想设置网卡的 IP 之类的呢？那也找 ifconfig，很方便，比如要修改 eth0 网卡的 IP：

```
lanwoniu@lanwoniu-ubuntu:~$ ifconfig eth0 192.168.1.100
```

就可以了，eth0 后面就是你要设置的 ip 了，如果同时想设置下子网掩码，就在后面加上一段，类似这样：

```
lanwoniu@lanwoniu-ubuntu:~$ ifconfig eth0 192.168.1.100 netmask 255.255.255.0
```

经常有人想修改网卡的 mac 地址，在查皮那里好像还要改什么注册表什么的，我们这里还是交给 ifconfig 就可以了，简单：

```
lanwoniu@lanwoniu-ubuntu:~$ ifconfig eth0 hw ether xx:xx:xx:xx:xx:xx
```

其中 xxx 就是你要改的 mac 地址。最后，如果你看着某个网卡不顺眼，可以停掉他，就这样：

```
lanwoniu@lanwoniu-ubuntu:~$ifconfig eth0 down
```

如果过会你又怀念他了，还可以把它找回来：

```
lanwoniu@lanwoniu-ubuntu:~$ifconfig eth0 up
```

说了这么多 ifconfig 的用法，最后还是要说明一下，ifconfig 这家伙做事还是有些毛糙，他只管立即生效的东西，所有你做的设置在系统重启之后就算都失效了。

故事外的事——bash 的历史

在最初创造 UNIX 系统的时候，Ken Thompson 大牛在 Bell 实验室写了一个简单的程序，将它作为新的 UNIX 操作系统的接口界面来让系统和人类进行交流。这个程序就叫做 shell，那时候还没正式起名字，为了让使用这个 shell 的人们不忘记作者的辛劳，大家也就管它叫 Thompson shell 了。Thompson shell 的功能很简单，用户通过它输入一些指定的命令，它负责解释为需要计算机做的操作，并去执行。另外它还能够支持一些简单的脚本，就是把一堆命令写进一个文件里依次执行。但并没有更高级的例如流程控制，分支，变量，函数之类的东西。后来这个 shell 进行了扩展……

同时，Ken Thompson 的同事，同在 Bell 实验室的 Steve Bourne 也写了一个 shell 程序，作为一个有很多重要程序要写的大牛，Steve Bourne 也没来得及给他写的 shell 起名字，于是人们同样为了不忘记作者的共享，也就管这个 shell 叫做 Bourne Shell。这个 shell 跟

Thompson shell 不同，他加入了流控制，可以写简单地函数。等到了 1970 年代晚期，每种 shell 在 Bell 实验室都有不少的人用，于是形成了两个派别。说来这两个 shell 应该是都很不错的，用熟悉了都很顺手，但是这两个 shell 是不兼容的，就如同修习了少林的内功后再去学武当的心法，多半不是很习惯。但是作为一个成熟的商业化的系统，总该有个默认的，标准的 shell 才好方便用户学习使用。于是在 bell 实验里，分别支持 Thompson shell 和 Bourne Shell 的两大帮派，进行了激烈的辩论，经过三次连续的 Unix 用户组集会上两大帮派的斗争之后，终于确立了 Bourne shell 为了 Unix 的标准 shell。

等到 1978 年，Bourne Shell 随着 Version7 Unix 一同发布，于告别了实验室，和广大用户见面了。9 年后，1987 年，一个叫做 Brian Fox 的家伙非常喜欢 Bourne Shell 并且觉得它还可以更加完善，于是就开始在 Bourne shell 的基础上进行创造，几年后成为了一个更加完整而且好用的 shell。出于对 Bourne shell 的缅怀和崇拜，他将这个 shell 命名为 Bourne Again Shell——简称 bash。现在，bash 是绝大多数 linux 系统，以及 Mac OS X v10.4 系统的默认 shell。甚至还被移植到了 Windows 系统上，什么？你没见过？那你听说过 Cygwin 吧？哈哈，那里面就是 bash 啦。

6.2 这么用 Shell

于是，主人那敲敲打打的命令行生活就这样开始了。从那以后，工作间里少见了红酒大师；看不到盒子妹妹；心有灵犀也不常来上班了；OO 老先生也难得起床了，工作间里净是些命令行的小程序在跑来跑去。唯一还经常出现的熟悉的面孔就是狐狸妹妹了，因为主人总要上网去查查命令行的相关知识，毕竟跟那个彪悍的 man 说话还是有点费劲的。

后来主人觉得玩虚拟终端还不够过瘾，现在每次都要按 ctrl-alt-f1 进入真正的终端来敲名令，看着一屏屏的字符好象很有成就感的样子，如果遇到需要查的东西再按 ctrl-alt-f7 回图形界面来查。

进入终端以后，最先被主人叫起来的是 ls，这家伙算是出镜率最高的一个命令程序了，ls 是 list 的缩写，他是专业负责清点物资的。用起来也方便，就敲 ls 就可以了，他就会把当前目录下的所有文件给你列出来。如果你觉得光看文件名不够的话，还可以这样：“ls -l ”，这样以列表方式查看，信息就更丰富了。主人运行了一下 ls，看到了当前目录下的所有文件。所谓当前目录，就是用户现在所在的目录，比如你在你家卧室发呆，那么你的当前目录就是卧室。过一会你又去客厅发呆了，那当前目录就是客厅，然后你去厕所发呆，当前目录就是厕所（怎么到哪都发呆……(⊙ _ ⊙)?）

那么主人现在的当前目录是哪个目录呢？就是他的家目录，就是图形界面的“位置”下的“主文件夹”，也就是 /home/lanwoniu 这个目录。当主人每次打开终端的时候，无论是虚拟终端还是 ctrl-alt-f1 进入的终端，刚一进去，都是在主人的这个家目录里。

不过毕竟不是什么事情都要在家目录里做的，如果你在这个目录里看够了，想出去走走，到其他的目录逛逛，这时候就需要 cd 了。

cd 不是那个光盘，而是 Change Directory（改变目录）的缩写。这个命令可以改变当前的目录，他就像出租车一样，可以让你到达你想去的任何一个目录。（当

然，前提是你得有权限）用法就是：

```
cd  < 后面写你想去的目录 >
```

那么，如何描述你想去的目录呢？一般有两种方法：绝对路径和相对路径。

绝对路径，就是无论去哪都从根上说，比如你打车，司机师傅问你去哪，你说：“我去地球，亚洲，中国，北京市，崇文区，羊肉大街，排骨胡同，376 号。”这么说就是绝对路径，无论什么地方，都从一个根本的位置上说起，（比如地球……当然，你愿意从太阳系说也行。）层层递进，最终说到最小最详细的那个地方为止。那我们这里的目录的地址当然不能从地球开始说了，我们的根目录“/”就是那个最初的，根本的位置，无论你去哪个目录，都可以从这里说起，比如 cd /usr/share/system/hidden/av，这就是说：要去根目录下的 usr 目录下的 share 目录下的 system 目录下的 hidden 目录下的 av 目录。（埋藏的真深啊！）

这么说很准确并且直接，不过有时候也比较费劲。所以，cd 还可以支持相对路径。

相对路径，就相当于你打车，司机师傅问你去哪，你说：“就前面那路口，左转，过仨红绿灯走 700 米有一精神病院，到时候我指给您，您就靠边停车就行了。”这种描述方法就是以当前所在的地点为起始地点进行描述，而不用从外太空开始说了。那么具体到我们这个系统里怎么说呢，就这样 cd abc/bcd/dca/cbd/adb/av ，这句话的意思就是：要去当前目录下的 abc 目录下的 bcd 目录下的 dca 目录下的 cbd 目录下的 adb 目录下的 av 目录。（归根结底还是 av 目录……）

主人学会了 cd 命令，兴奋的在终端里 cd 来，cd 去的，像个跑进游乐园的小孩子，到处走到处看。看着看着，忽然觉得有点迷路了。静下来想一想：我现在是在那个目录呢？是在 /usr/bin，还是 /bin，还是 /usr/local/bin 呢？其实，他看一眼那个提示符 $ 前面的内容就可以了，默认设置下这里显示的就是用户所在的目录的绝对路径。不过这个提示符的格式是可以修改的，如果修改了，这里显示的不是当前的路径了，那怎么办呢？我们早为您想到了，命令行中配有专业的引导员，告诉您您现在所在的位置，这位引导员就是 pwd。

可别一看名字就以为他是负责修改密码的阿，其实他跟 password 一点关系都没有，他是 Print Working Directory 的缩写。用法简单，就输入 pwd 就行了，他就会告诉你现在所在的目录。

忽然，主人发现开机了半天还没有联网呢，这个机器上有个无线网卡，需要链接一下家里的无线路由器才可以上网，在图形界面有网络管理器负责，在字符界面下，也同样有称职的人员为你服务，这就是 iwconfig。主人又叫来 man 来解释解释 iwconfig，纯爷们发话：“嚯哈哈哈，这个 iwconfig 嘛，就是如此……比如……你这样……对啦……然后……没错，嘿嘿，果然……就连上啦。”您问到底怎么样？您自己去问 man 吧，我在这也不好把 iwconfig 的手册翻译一遍不是。学习了一下之后，主人敲 iwconfig 来试着联网。可是，他忽然觉得，这个 iwconfig 字母有点多，不如 ls，cd 那样简短。这么多字母输入起来有点麻烦，有没有什么省事的办法呢？

很多人不喜欢键盘，不喜欢打字。其实想想，早在电脑刚刚被发明出来的时候，键盘就已经是每一台电脑所必备的输入设备。作为从那个字符界面的时代走过来的我们 Linux 系统，自然充分考虑的通过键盘操作整个系统的便捷和效率问题。直到现在，使用键盘操作 linux 都会拥有意想不到的高效率和成就感。

我以前很不明白，命令键盘可以发送上百个命令，用起来应该很方便才对，为什么人类就那么喜欢那个只能发送：向上，向下，向左，向右，左键，右键，这六个命令的鼠标呢？（当然，现在的鼠标还多了滚轮，还有的鼠标有更多的按键，但是那也比键盘少啊。）后来见多识广的 OOo 老先生给我解释，我才明白，原来是因为人类记忆力不行，没有我们软件这么可靠。记不住那么多个键，于是只好用那只能发送六个命令的鼠标了。

好了，绕的有点远，其实说起来在我们的命令行里通过键盘敲命令是很方便的，只是很多人不大熟悉如何节省时间而已，都以为用键盘和我交流跟用键盘和那个剁死系统交流一样麻烦呢。其实我已经很人性化了，就因为键盘上有个键——Tab。

> 看一个人的键盘，就可以猜测出他平时用电脑干什么。如果 W,A,S,D,U,I,J,K 严重磨损，说明这哥们玩拳皇的；如果 A，Shift，Ctrl，1，2，3，4……9，0 严重磨损，说明是个玩即时战略的，星际魔兽之类；如果 ALT,S 或 Ctrl,Enter 磨损，大概是天天聊 QQ；如果 Tab 键严重磨损，那估计就是个 Linux 高手了。

在 Linux 的命令行下，Tab 键起着命令补全的作用。比如说，你要运行 ifconfig 命令，你可以不用完全输入这 8 个字母，只要输入 ifc，然后按 Tab 键，Bash 就知道你要干什么了，因为所有可以运行的命令里面以 ifc 开头的就只有 ifconfig，所以当你按下 Tab 键的时候，他就会替你写出完整的命令：ifconfig。这都因为在你按下 Tab 键的时候，Bash 会去 PATH 变量所设置的所有目录里遍历一遍，检查了里面所有的有 x 权限的文件，结果只查到了 ifconfig 文件（命令其实就是个可执行文件）。

之所以这么快，是因为他早就把这些重要的东西缓冲进内存了，所以下次就别抱怨我们 linux 动不动就把你内存占满了哦。那如果你再少写个字母呢？比如你只写了 if，然后就按 Tab 键，Bash 遍历了一边 PATH 中的路径后发现，有 4 个命令是以 if 开头的，所以他不知道你要的是哪个命令，于是就不做任何动作。这时候如果你再按一下 Tab，他就会提示你：以 if 开头的命令有 if、ifconfig、ifup、ifdown。然后你自己看需要的是哪个，照着输入就行了，很交互是吧？这样除了减少按键次数以外，还有一个好处就是你可以不必完全记住整个命令，能够记住前几个字母就可以通过 Tab 把整个命令回忆出来。

有了 Tab，就让用户输入新命令的时候省事了不少，还有一个 history 功能，可以让用户重复以前输入过的命令的时候省心。如果你想输入上一次输入的命令，就按一下向上箭头，就看到了；如果想要再上一次的命令，就再按一下；如果想要再再上一次的命令，就再再按一下；如果想要再再再上一次的命令，就再再再按一下；如果想……如果想写作文的时候凑字数，你就跟我学。好了，总之是可以通过方向键选择以前运行过的命令，如果

想查看很久远以前的命令呢？也可以，输入 history。这是一个命令，可以显示之前运行过的 n 条命令，默认情况下 n=1000，现在图形界面越来越发达，输入命令的机会越来越少，估计 1000 条都能把去年的命令显示出来了。说起来 history 命令也没啥神奇，他之所以能够显示曾经运行过的命令，不是因为他有啥水晶球，而是负责解释主人命令的 shell 会把每一条命令记录下来，就写在 ~/.bash_history 文件中。history 只不过是把这个文件打开，显示出里面的内容罢了。

主人输入了 iwc 三个字母，然后按了一下 Tab 键，bash 赶紧给他把命令补全，iwc 变成了 iwconfig，主人直接按下回车，哈哈，果然省事了不少。只简单的运行 iwconfig 的话，只是列出现在的无线网卡的状态，要想了解具体的使用方法——对了，还得问那个爷们，于是一阵“嚯哈哈哈”的笑声又传来了。

主人经过学习，用 iwconfig 连接到了无线路由器，但这只是相当于拿个网线插到了路由器上，还需要有一个命令来配置网口的 ip 地址啥的，对！就是 ifconfig 命令，之前主人在图形界面下就用过了。那时候主人运行 ifconfig，出了好多字，滚屏滚上去了，主人拖着虚拟中端的滚动条看了一下，这个动作如此的自然，没有任何异样的感觉，然而现在到了真正的黑漆漆的终端里，主人又随手运行了 ifconfig，忽然发现一个问题——滚上去的信息怎么看啊！

没有滚动条啊！！有木有滚动条啊！！！有木有啊！！！！有木有！！！！！

好，我们让主人自己咆哮一会去，咆哮的主人我伤不起阿。

其实就是想向上滚屏嘛，很简单，有快捷键，Shift+PageUP 是向上翻页，Shift+PageDown 是向下翻页，就这样。当然，可能光滚屏还不够，有时候想把命令输出的东西翻来覆去慢慢看，有什么办法没？肯定是有的，当初的前辈们都是生活在命令行下的，没事编译个软件啥的，那输出的提示动不动就好几百行（代码写的 warning 太多 -_-b），一屏是根本显示不下的，就算 80 寸！竖着放也不够！

显示不下了，后面的内容就会把前面的内容“顶”上去，“楼主”固然是看不见了，什么“沙发”、“板凳”、”地板”、”下水道”的也一样没希望，只能看到最后那几十条。那想看前面的怎么办？虽然可以用 shift+page up 来向上翻页，但一来向上翻的页数也是有限的，二来这也麻烦，一般人都是习惯从上往下看的，倒着往上翻就别扭了。那怎么办呢？这时候 more 就该出场了。

more 的功能就是分页显示，把所有要输出的内容先显示出一屏来，等着用户按回车，之后再显示第二屏，直到显示完全部内容。当然，用户也可以不等显示完全部就中途按 q 退出。那怎么用呢？就比如刚才主人这种情况，那就运行“ifconfig | more”。“|”符号叫做管道符，就在你键盘上的”\”键。这个符号的意思就是把前面一个命令的输出内容交给后面一个命令作为数据输入。于是，ifconfig 命令输出的那些个字符就像一大堆泔水一样哗啦啦的都流倒 more 这里了（怎么偏偏是泔水……），more 就把这些东西都先整的大桶缓存起来，然后先盛出一碗来给你看，“您看有没有想吃的？”，你看完了之后，他再去盛第二碗，第三碗……直到你把整个一大桶泔水都检阅完了，more 才结束工作。（当然，你要是坚持不到最后就吐了，那就按 q 退出）有了 more 就满足了么？不，还有他的死对头，less。

less 实现的功能和 more 基本一样，也是用来分屏输出的，同行是冤家嘛。不同的是，more 只能一页一页往下看，看完了就退出。less 可以上下翻页，看过去的东西可以按向上键或者 page up 键翻回去看。是不是比 more 更人性一点？另外，都看完了之后 less 是不会自动退出的，一定要按 q 退出。顺便说一下，more 和 less 不光可以通过管道将其他命令的输出当作输入，同时也可以直接查看文件。只要“more /< 路径 >/ 文件名”或者“less /< 路径 >/ 文件名”就可以查看文件内容，当然，只能是文本文件（里面是文本就行，不一定非得以 .txt 为扩展名，我是 linux，没有某些 OS 那么傻）。

通过这两个命令，您大概可以感受到我们 Linux 系统和有点软公司的系统的不同理念。用过剁死系统的都知道里面有个 DIR 命令，和我们的 ls 一样，都是显示文件用的。当文件很多的时候，dir 命令有专门的参数可以实现分屏显示，而 ls 命令就没有，只能一下子显示出来。为什么？因为分屏显示的事情是由 more 和 less 负责的，完全可以通过“ls | more”这样的组合实现分屏显示。linux 的理念每个程序只专注于一种功能的实现，而通过多个程序的组合可以实现任何功能。试想如果没有 more 和 less，ls 要负责分屏显示的话，那 history 命令是不是也要处理分屏显示的问题呢？那么所有输出行比较多的命令都要自己负责分屏显示，这些命令的源代码中都要有负责分屏显示的部分，这是一种无谓的重复劳动，而且各自分别实现分屏显示，效果多半也不一样，可能有的命令是按空格显示下一行，有的是按 n 显示下一行等等。与其这样不如把相同的功能独立出来，成为一个统一的，单独的命令。

好了，主人终于结束了咆哮，回到图形界面找狐狸妹妹去查怎么在命令行下翻页了。

说起命令行下省事的办法，还有个事情不能不提，这就是通配符。

过去用过“剁死”系统的同学可能知道，通配符就是“*”和“?”，“*”可以代表任意多个任意的字符，“?”代表任意一个字符。不过“剁死”系统下通配符的实现是需要每一个命令对通配符都理解的。

比如说剁死系统里运行“copy *.jpg aaa”。剁死这个懒得要死的家伙肯定啥也不管，只把 copy 叫醒，然后告诉他：“用户让我告诉你：‘*.jpg aaa’快干活去吧！然后 copy 就要去理解，*.jpg 就是所有的 .jpg 为扩展名的文件，所以用户给他的任务就是拷贝（因为他只会拷贝，别的任务不会叫他来）当前目录下的所 .jpg 文件和 JPG 文件和 Jpg 文件和 jPg 文件和 jpG 文件和……（谁让剁死不区分大小写呢。）拷贝到当前目录下的 aaa 目录去。这样做有什么不好呢？不好就是没个剁死下的命令，要想支持通配符，都要自己处理，浪费资源啊。

我们 linux 下的 shell 们就不这么懒了。

就拿 bash 来说吧，也一样支持通配符，同样也是“*”代表任意个任意字符，“?”代表某一个任意字符。不过，通配符的解释都是由 bash 来做的。

比如拷贝文件，运行“cp *.jpg ./aaa ”那么，这条命令输入进去之后，先交到 bash 这里，bash 一看有通配符那就需要自己进行解释，“*”代表任意长个字符嘛，那么这条命令的本意就是“cp 11111.jpg 22222.jpg 33333.jpg 44444.jpg ./aaa ”，也就是说，bash 用当前目录下的所有 jpg 文件的文件名代替了 *.jpg 这个字段，然后再叫醒 cp，把扩展后的参数传给他，cp 看到

的就是没有星号的命令了，翻译过来就是“复制当前目录下的 11111.jpg 和 22222.jpg 和 33333.jpg 和 44444.jpg 到当前目录下的 aaa 目录下。”这样做的好处是明显的：bash 下的各种软件都不许要自己处理通配符的问题，减少了不必要的重复开发。

当然，也有的时候是不需要 bash 对特殊符号进行扩展的。比如主人可能就是有个文件叫做“*.jpg”（没错！就叫这么个特殊的名字！在我们 linux 里你爱叫啥就啥，绝对不会跳出来个警告框说你不能起这种名称之类的，搞的主人好怕怕），要想复制着一个文件而不是所有的 jpg 文件那怎么办呢？也好办，就这样：“cp *.jpg ./aaa”在“*”前面加上一个“\”就可以了，“\”的意思就是后面的字符是个普通字符，告诉 bash 不要进行处理。

故事外的事——iwconfig

这节说到 iwconfig，那么就为懒得查 man 的同学们说说 iwconfig 的用法吧。

iwconfig 小程序是用来管理无线网络的，现在 wifi 的使用是越来越广泛了，很多人家里都使用无线路由器来共享网络链接，我家主人这里也是。在图行界面下，那个 nm-applet 工作的还是不错的，只是主人不想特意为了链接网络而去一下图形界面，所以就学习起 iwconfig 来。iwconfig 的主要任务就是负责把电脑接到某个无线接入点上。经常和 iwconfig 同时来工作的还有一个家伙，叫做 iwlist，这里是用来查找无线信号的。一般，都是先运行 iwlist scan，来扫描附近可以找到的无限信号，然后再选择其中的某一个接入。iwlist 得到的结果里最主要的是那个你意中的无线接入点的 essid，有了这个 id 才好跟 iwconfig 说你要接谁，如果不说的话，你只运行 iwconfig，那么他只是帮你列出现在机器上的无线网卡的状态——多半是单身状态，也就是说没有连接到任何无线接入点中。那么要想连接怎么办呢？只要知道了 essid，就运行 iwconfig wlan0 essid “xxxxxx”就可以了，就像你知道了对方姑娘叫啥，才好找人去她家说媒是不。这其中 wlan0 是你要是用的无线网卡（因为你可能有多个无线网卡，就像你有俩儿子，你得决定一下给那个儿子说媒嘛），xxxx 是那个你要链接的无线接入点的名字，也就是 essid。这样运行了之后，iwconfig 就会去试着让你的无线网卡跟对方眉来眼去一翻，然后鸿雁传书，互诉衷肠，心驰神往，最后喜结连理——你就能上网了。不过这是对方没什么要求的情况，有的时候对方会要点彩礼——一把钥匙。别急，不是房子的钥匙，不过自行车钥匙也不行，要的是能够解开对方

甜言蜜语的钥匙——解密用的 Key。毕竟无线这东西看不见摸不着，你俩聊的卿卿我我的搞不好就隔墙有耳，所以设个密码还是需要的。密码基本有两类，wep 的和 wpa 的。wpa 又有很多种，我们暂时就不具体讨论了，因为 iwconfig 处理不了那些，如果是 wep 的话，那就好办。链接的时候就运行 iwconfig wlan0 essid “xxxxxx” key yyyy-yyyy-yy 这样的格式就可以了。那些 yyyy 就是密码，就像俩人的信物一样。密码必须是 5 个字节的 16 位数字，具体是什么，那只有你知道咯。当然，除此之外 iwconfig 还有很多功能，比如设置网卡工作频率阿，设置网卡自己作为无线接入点啦之类的，这些东西，你还是去问那个纯爷们吧。如果你也不大愿意和那家伙交流，你也可以问那个命令，就是运行 iwconfig --help，让他自己告诉你应该怎么用。不过，虽然绝大多数命令都可以用 --help 来查看使用方法，但毕竟不是全部的命令都能这样，要最详细的说明，还是得靠 Man。

6.3 多彩的 Shell

主人逐渐的开始适应了命令行的操作，于是他有个想法，在这个黑漆漆的界面中，搭建起一个可用的环境，这样以后开机就可以不进图形界面，直接用命令行的软件来做各种事情，可以更高效，更快速。

他决定，用 7 天的时间来完成这件事情。

起初，主人来到混沌漆黑的命令行。界面是黑漆漆一片，ls 还会出些菱形的方块块，主人的手指游弋在键盘上面主人说：“要有中文！”就有了中文。

主人看中文是顺眼的，有中文，有英文，没有了乱码。这是头一日主人说：“要有声音！打破寂静的黑夜。”

主人就让命令行里发出美妙的乐声，这是第二日主人说：“要有窗，看到外面的世界。”于是有了浏览器，主人可以重新看到那些网络上的奇花异草，光怪陆离。

有了通讯工具，主人可以把这些光怪陆离，异草奇花分享给朋友们，这是第三日主人说：“要有色彩，有图，才有真相！”于是就有了图。

图片为黑白的世界带来了色彩，美好的，丑陋的，思念的，怀旧的，唯美的，憧憬的，现实的，抽象的，红色的，绿色的，蓝色的，黄色的，各种的图片，主人微笑了。这是第四日。

主人说：“要动起来，要鲜活的世界。”于是，就动起来了，这是第五日。

第六，七日，主人说，我累了，要休息，于是，就没有于是了，丫没开机。

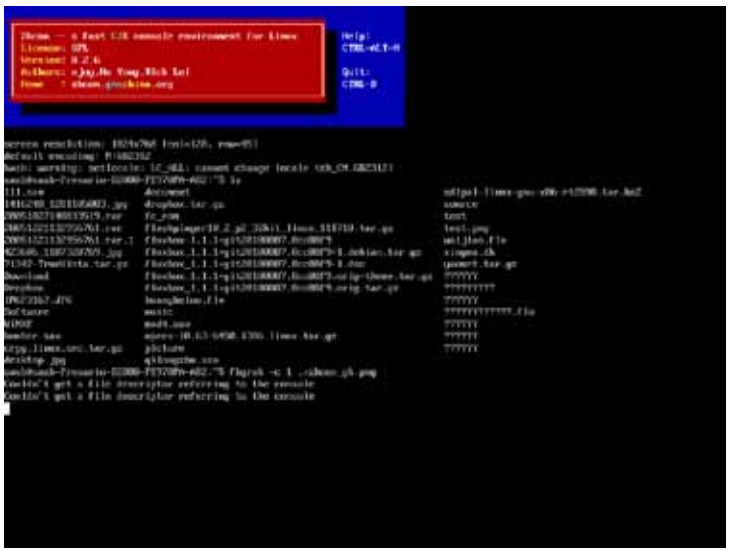
众人说：和着您主人就是传说中的上帝？？

我说：不是，上帝第七天才休息，主人第六天就休息了。

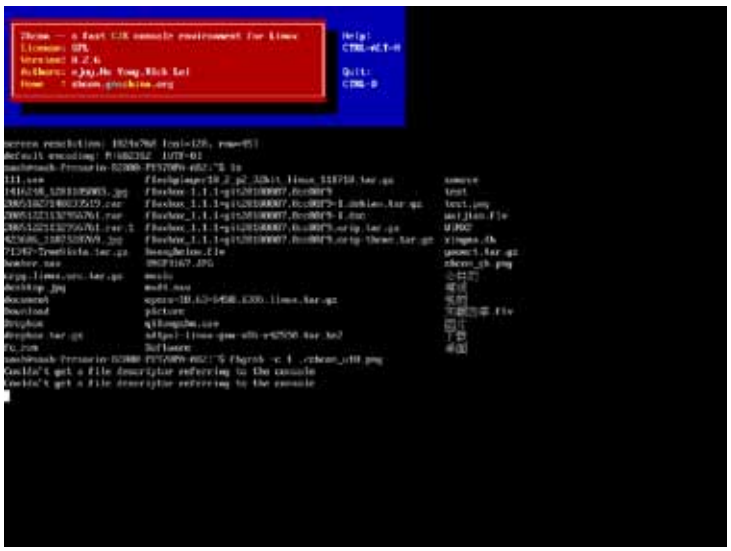
众人说：废话，那是上帝没赶上实行双休日。

咳咳，我估计这么说您还是听不懂，那我慢慢给您讲解。

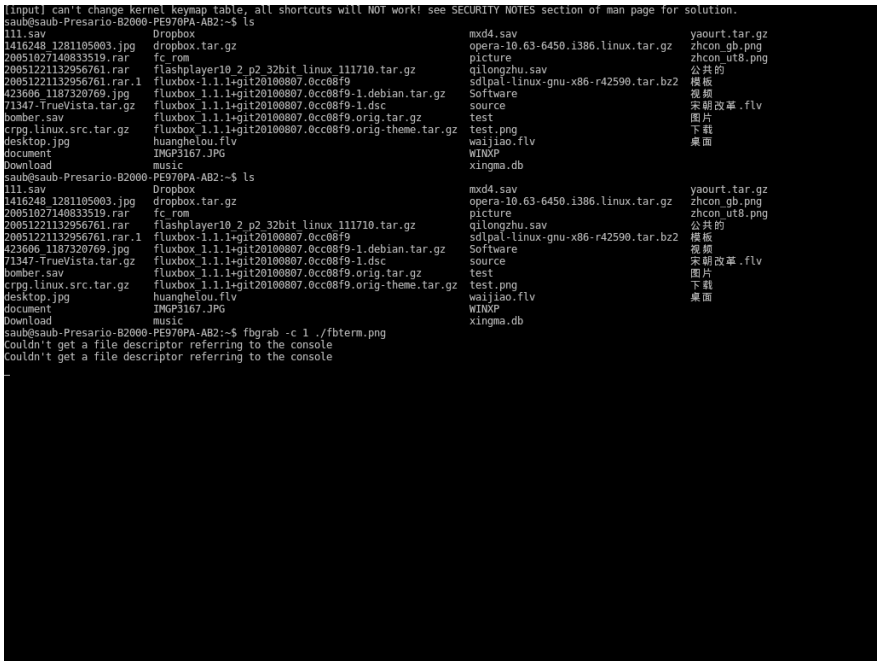
这周一，主人开始打造自己的命令行世界。各种命令对主人来说都已经很熟悉了，就算有些不大熟练的命令，也可以很轻松的找到纯爷们来讲解。但是一来到命令行里面遇到的第一个问题就是，没有中文的支持。运行下 ls，主目录下的图片啊，文档啊这些目录显示的都是一些菱形的方块。这也难怪，在伪终端下，中文环境由图形部门负责，可到了真正的终端界面中，这事情就没人管了。毕竟多数人还是要使用图形界面的嘛，像我主人这么神经质的人不多，所以我们没有更多的考虑字符界面的使用感受。不过主人现在需要字符下的中文环境，那我们也有人能够胜任，而且，还不只一个。



首先就是老牌的 zhcon，这个家伙有些岁数了，很久以前他就担当起让命令行下显示和输入中文的重任。他是一个外挂的中文环境，就像很久以前的“剁死”系统里的那个 UC-DOS 一样。要安装他很简单，找超级牛力就行。装上之后，在终端里（不能是图形界面下的伪终端哦。）输入 zhcon，就运行起来了。不过主人就这么运行了之后发现，这目录下的中文还是有问题。原来中文的地方都是菱形方块，现在倒好，改成一堆问号了。难道这 zhcon 的能力就是把方块改成问号？



想想也不是阿，于是主人运行了“zhcon --help”，问问 zhcon 这到底是怎么回事。zhcon 解释说，你要设置一下编码，我默认是使用 GB2312 的，你系统里的文件都是 UTF-8 的，那你运行的时候就要这样：“zhcon --utf8”才行。主人这才恍然大悟，心说你倒是早说啊。赶紧照着他说了运行，然后 ls 一下，果然，终于看见中文了。按下 ctrl-空格，哈哈，出现了输入法，虽然全拼不是那么好用，不过，命令行下，也凑和了。



zhcon 也是有不少后遗症的，启动了 zhcon 后，凡是一些需要有排版有格式的显示，比如 w3m 显示的网页啦，moc 显示的 mp3 播放介面啦，都会或多或少的乱掉。所以主人又找了更好的解决方案——fbterm。

fbterm 是一个基于 framebuffer 的终端。framebuffer 是一种字符界面下程序访问显卡的接口。他是能让命令行更加丰富多彩的关键的一环。通过 framebuffer，程序们可以直接向显卡的显存里面更新数据，也就是直接影响屏幕上显示的点和颜色，这就让在终端下显示图片成为了可能。不过由于程序是直接控制显卡的显存，并不经过显卡运算，所以显卡的运算能力是不能发挥的，所有的图形都是靠 CPU 运算好之后直接送到显卡显存里，所以 framebuffer 是比较消耗 CPU 的。framebuffer 以后我们还会经常看到，现在先说 fbterm。

fbterm 其实也可以算是个虚拟终端了，就跟图形界面下的那个 Gnome 终端一样，说不定还是一个学校毕业的呢。只不过 fbterm 是在终端下，用 framebuffer 来“画”出一个虚拟终端来，而 Gnome 终端是靠图形界面组的那些人来画虚拟终端。fbterm 的好处就是不像 zhcon 那样会搞乱屏幕的排版，所以主人自从用了 fbterm 之后，就再也没运行过 zhcon。不过 fbterm 只是一个能够显示中文的终端而已，他不是一个中文环境，所以，只能看到中文，输入是不行的。不过没关系，专门在命令行下的输入法，也是有的，和 fbterm 配合起来，就天衣无缝了。

关于中文输入法，首先就是来自中国的 yong，学名叫做小小输入法，这个我们的源里可没有，得去他的网站上下载并安装。装好之后，只要在运行 fbterm 的时候加上参数，指定输入法，就可以在 fbterm 中使用了，指定输入法用 -i 参数，就这样：“fbterm -i yong”。除了 yong，还有 fbterm_ucimf 可以使用，不过相对有点简陋。另外还可以使用 ibus-fbterm，看名字就知道了，是基于 ibus 的，这个相对好用些。

最终主人是选择了 fbterm，这不单是因为看上去顺眼点，更关键的是后来的境遇。

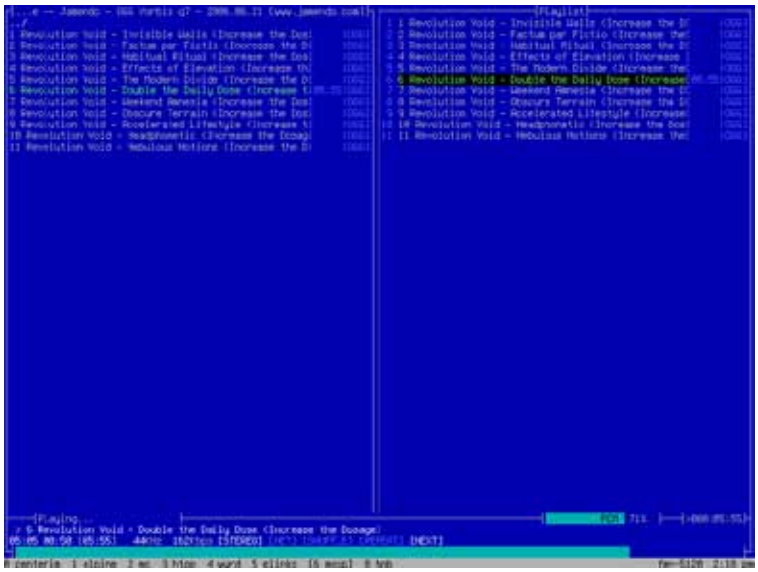
周二，主人得到了比较顺手的有中文环境的命令行之后，又开始踏上了新的寻觅旅程——寻觅音乐。

在图行界面下的时候，有很多的播放软件可以让主人浸泡在音乐的海洋里。那在字符界

面呢？想想也不应该有问题呀，放音乐又不需要图形界面是不是？字符界面下播放个音乐还是不成问题的，mplayer 就可以播放各种音频，还有个 mpg123 也可以放出声来。你知道 gnome 界面下，鼠标悬停在音乐文件上就可以播放出声音么？不知道？你 OUT 了。那个悬停播放的功能就是靠调用 mpg123 实现的。不过这俩虽然能马马虎虎弄出点动静，不过毕竟不专业，怎么也得有个播放列表，歌曲管理的功能吧？于是，主人选择了 moc。

moc 是个字符界面的音乐播放软件。有好奇而 OUT 的同学可能猜想了：字符界面播放音乐，那是不是要输入命令才播放音乐呢？比如输入 play，就播放，输入 stop 就暂停，输入 load xxxx.list 就导入播放列表，输入…… 反正干啥都得输入吧？这个，又 OUT 了吧？现在这么简介高效的年代，怎么可能又这么难用的东西呢？虽说是字符界面，但是不代表就不能有窗口！是的，你没听错，在字符界面下也能画出窗口来！！怎么画？用字符拼呗！！！

我们知道，ASCII 码中有一些特殊的字符，什么横杠阿，竖杠阿，拐弯杠阿……什么的。用这些特殊的符号，加上可控制的字符底色，可以拼接出又窗口效果的终端显示的界面。当然，如果每个程序都自己写代码去拼去肯定是很费事的，所以崇尚共享的我们 linux 系统提供了一套专门在字符界面下画窗口的函数库，叫做 ncurses，调用这个库画窗口就跟用 gtk+ 提供的接口在图形界面下画窗口类似，所以编程的人员不必在意怎么用各种字符拼出好看的窗口，只要调用 ncurses 就行了。ncurses 的前身是 curses，大名鼎鼎的 vi 就是用它实现的界面。moc 是一个基于 ncurses 的，字符界面的音乐播放软件，直接找超级牛力就可以安装，主人装了之后赶紧运行一下试试。虽然包名叫 moc，不过运行的命令是 mocp，运行起来之后就是这个样子：



左边是文件列表窗口，可以通过上下箭头选择相应目录，按回车进入，最上面的那个“..”表示上一级目录，这个都知道吧？不知道的面壁去。选到要听的 mp3 文件按 A 键将文件加入右边的播放列表。把 mp3 都加进来之后，用 TAB 键，切换左边的文件列表和右边的播放列表。按回车就可以播放了。然后还有些快捷键比较常用，左右键可以调节播放的进度，“,”“.”用来控制音量，s 键停止，b，n 跳到上一首 / 下一首歌，R 和 X 键用来控制重复播放和循环播放。（注意大小写的区别哦，所谓按 R，就是按下 shift 然后按下 r 键，也就是在终端打出大写的 R 的操作顺序。）

有人说，这播放软件把整个屏幕都占了，我还怎么干别的呀？简单，你有三个选择：

1. 不干别的了；

2. ctrl-alt-f2，换个终端；

当然，这两个方法都是比较笨笨的，呵呵。其实 mocp 是一个 server 和前端分离的软件，你可以按 q 键退出前端界面，然后该干啥干啥，音乐继续播放，想再回来就再运行 mocp。如果想彻底退出就按 Q。

好了，主人已经添加好播放列表，按下回车键，音箱里传来了悠扬的乐曲：“……老爷子 89 了，满面红光，别看就一个牙了，吃东西胃口很好，吃着吃着还塞牙了。”“就一个牙了怎么还塞牙？”“吃藕，套眼里了”……

周三这天，主人要解决一个关键的应用——上网。平时在图形界面下，最忙录的就是狐狸妹妹，基本上只要电脑开着，内存里就少不了狐狸妹妹的身影。这要是命令行下不能上网，那还不得憋屈死。不过在命令行毕竟是命令行，功能还是有限的，所以对浏览器的要求也不能太高是不是，简单的文字的网页还是没问题的，图片呢，实时的显示也不行，不过勉强可以调用别的程序来看一下，但是 flash 啦，在线的视频这类的就不用想了。好在主人比较通情达理，也就是想看看文字的东西和简单的一些图片，所以，有那么几个候选软件是可以满足主人要求的：w3m，lynx，links。

这三个都是可以直接叫超级牛力去装的，很省事。但经过主人的试用，lynx 和 links 对中文的支持都相对差点，并且他们都不能显示图片，所以被 pass 了，w3m 华丽胜出。不过 w3m 也不是天生就会显示图片，他需要一些工具，这些工具被打成一个包放在软件源里，包的名字叫做 w3m-img，主人让超级牛力给 w3m 搬来了这个工具包之后，w3m 才拥有了显示网页上图片的能力。

当然，w3m 显示图片跟 moc 显示窗口可不一样，可不是拿特殊字符拼出图片，要那样的话，整个屏幕，顶多拼个超级玛丽出来，连蘑菇都没地方显示了。要显示正经的 jpg 这样的图片，还是需要 framebuffer 的支持（当然，一些不正经的图片也一样需要 framebuffer）。基本上 framebuffer 是能够使命令行界面变得多姿多彩的最基本最重要的条件。主人要访问的网页无疑基本都是中文的，所以需要能够支持中文的终端，咱之前说过了，就是 fbterm。于是，主人就在 fbterm 下运行起了 w3m。

不过，天有不测风云日，人有无意失手时，软件也有偶尔不大正常的时候。虽然在某些系统某些机器上，fbterm 下的 w3m（前提是装了 w3m-img 哦）确实可以正常的显示出图片，不过不知道为什么在我们这里，这个 w3m 和 fbterm 配合了半天也没配合上，他们一运行起来就听着他们俩不停的吵吵：“fbterm，快点把 fb0 设备给我，我要显示网页上这个图片”“你怎么又要啊，我这边的中文还没显示好呢。”“你怎么那么多中文要显示啊？？”“你这不废话么，我显示的都是你的网页，显示多少中文还不是你所了算。”“别急别急，你显示完了赶紧的给我用。”“等等，等等，再有 3 毫秒就完了”“我倒，主人翻篇了。刚才那个图片翻过去了，白算了半天。”“好了我用完了，给你用去吧”“用什么用，他已经翻到下一页了，又一个新的图片，我先得解码看看这个图片里是什么才知道该显示什么嘛。”……

总之，这俩从来没对上过，于是主人就没看到图片。不过后来主人还是找到了解决办法，那就是，换了另一个终端，其实跟 fbterm 还有点关系，叫做 jfbterm，也一样可以支持中文。在 jfbterm 下，w3m 终于可以正常显示图片了。

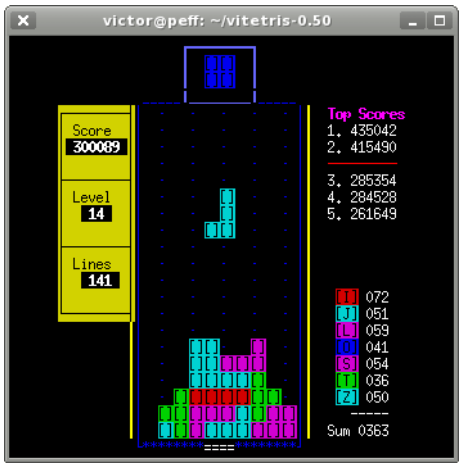
周四，主人又一次摄影归来，插上 SD 卡，运行 mount 命令挂在并且把照片 cp 到主目录中的照片文件夹。之后，就开始研究命令行下到底用什么来看照片了。要看照片无疑还是需要 framebuffer 的支持，这个咱就不说了。那么那个能够利用 framebuffer 设备来显示图片的软件是谁呢？那可是一个大名鼎鼎的家伙——FBI ！！

别激动，这个 FBI 不是某大叔的啥啥调查局，只是 linux 下的一个 Fram Buffer Image viewer，也就是基于 framebuffer 的图像查看器而已。这个查看器可以查看各种常见的图片格式，全屏显示，缩放，幻灯片模式播放，旋转，都没问题，当然更复杂的什么调节对比度、亮度啥的自然就不行了，毕竟人家只是个查看器嘛。主人拷贝好了照片后，用 cd 命令切换进入存着照片的目录，运行了“fbi ./*.JPG”，这个命令大家应该能看懂吧，就是让 fbi 去查看当前目录下的，所有的 JPG 文件的意思。顺便复习一下，* 这个通配符是由 bash 来处理的，就是说，主人输入了这个命令后，bash 先处理一下，把命令改成“fbi ./123.JPG 456.JPG GSMLY.JPG WTL.JPG CJK.JPG BDYJY.JPG FDA.JPG”等等等等，然后再按照这个命令去叫醒 fbi，告诉他要去显示这么多图片。然后 fbi 就起床，一个一个的显示这些图片，当主人按 -、+ 键的时候就对图片进行缩小，放大。按下 PageDown 键就播放下一张，偶尔遇到竖着拍的照片可能还需要按 R 或者 L 旋转一下，总之，主人在悠然轻松的欣赏照片的过程中度过了一晚上。

光看静态的照片还是有些不过瘾的，于是周五，主人拷来了一个叫 RPT.rmvb 的文件，想要看片了。那命令行下能看片么？答案依然是肯定的，当然，还是离不开 framebuffer。平时在图形界面下主人经常用来看片的播放器是 SMplayer，当初咱们介绍这个家伙的时候就说过，他只是个前端界面，真正在后面默默无闻的进行播放工作的是 Mpayer，Smplayer 能播放什么文件完全取决于 mplayer 能播放什么文件，只是因为 mplayer 总是在后台，人们都忽略了它的存在。现在，到了字符界面，mplayer 终于可以从后台来到前台了。主人直接运行了 mplayer RPT.rmvb ，于是，一个动态的画面出现在了黑漆漆的屏幕上，一个醒目的字幕写着：肉片汤的做法 ……

除了硬盘里现成的视频文件以外，mplayer 还可以播放 mms 流媒体，也就是一些在线的

视频也是可以播放的。主人找到了一些网上的电视台的流媒体地址，什么 CCAV5 阿，CCAV8 阿，这 AV 那 AV 的，反正我是不知道什么内容，但是主人看的很起劲。好像他屋里也没有那个叫做 TV 的东西，于是就拿电脑当作 TV 了。不过每次总是输入很长的 mms 地址很麻烦，所以主人就写了个脚本，把脚本命名为 tv.sh，运行的时候后面加上 CCAV5 阿 CCAV8 阿作为参数，想看电视的时候就晕行 tv.sh CCAV5，然后就可以坐在电脑前享受着没有 xorg 运行的 linux 系统带来的电视节目了。



到了周六，主人休息了，周日也没来找我们。不过后来的后来还是有故事发生的，主人竟然开始在字符界面下玩游戏了。不过为了同学们好好学习，这里就不多做介绍了，就给两个线索吧，有兴趣的同学可以研究 vitetris。

```
apt-get install bsdgame
```