

# Linux 桌面玩家指南：10. 没有 GUI 的时候应该怎么玩

## 阅读目录

- 前言 ##
- 逆天的 GRUB##
- GRUB 实战 ##
- Linux 纯字符模式和 Framebuffer##
- 字符界面下联网 ##
- 使用 fbterm##
- 显示 Framebuffer 的信息 ##
- 在 Framebuffer 下载图 ##
- 在 Framebuffer 下查看图片 ##
- 在纯字符界面下上网 ##
- 视频播放 ##
- 屏保和装逼 ##
- 版权申明 ##

特别说明：要在我的随笔后写评论的小伙伴们请注意了，我的博客开启了 MathJax 数学公式支持，MathJax 使用  $\$$  标记数学公式的开始和结束。如果某条评论中出现了两个  $\$$ ，MathJax 会将两个  $\$$  之间的内容按照数学公式进行排版，从而导致评论区格式混乱。如果大家的评论中用到了  $\$$ ，但是又不是为了使用数学公式，就请使用  $\$$  转义一下，谢谢。



想从头阅读该系列吗？下面是传送门：

- [Linux 桌面玩家指南：01. 玩转 Linux 系统的方法论](#)
- [Linux 桌面玩家指南：02. 以最简洁的方式打造实用的 Vim 环境](#)
- [Linux 桌面玩家指南：03. 针对 Gnome 3 的 Linux 桌面进行美化](#)
- [Linux 桌面玩家指南：04. Linux 桌面系统字体配置要略](#)
- [Linux 桌面玩家指南：05. 发博客必备的图片处理和视频录制神器](#)
- [Linux 桌面玩家指南：06. 优雅地使用命令行及 Bash 脚本编程语言中的美学与哲学](#)
- [Linux 桌面玩家指南：07. Linux 中的 Qemu、KVM、VirtualBox、Xen 虚拟机体验](#)
- [Linux 桌面玩家指南：08. 使用 GCC 和 GNU Binutils 编写能在 x86 实模式运行的 16 位代码](#)
- [Linux 桌面玩家指南：09. X Window 的奥秘](#)

[回到顶部](#)

## 前言 ##

前一篇讲了 X Window，这一篇就来讲讲没有 X Window 的 Linux，主要是 Linux 的字符模式，甚至是在进入 Linux 之前的 Grub 命令行模式。本来想把这一篇的标题叫《字符界面怎么玩》，或者《没有图形界面怎么玩》，但是怎么想都觉得不准确。去掉桌面环境的 Linux，并不一定就不是图形界面，因为 Linux 图形界面无处不在。以前我使用 Linux 桌面的时候，总是有一个误区：认为只有 XServer 启动后，才能够访问到图形系统，否则只能访问字符界面。随着对 Linux 的认



识逐步加深，才发现即使在 XServer 启动之前，图形界面也是无处不在的。例如，Grub 的系统启动菜单，可以是图形化的，还可以通过改背景和主题进行美化。再例如在 Linux 初始化过程中，有一个 Plymouth 软件，可以直接通过内核的 DRM 模块访问图形硬件，从而显示一个图形化的启动界面和进度条，同理，Plymouth 也是可以通过更改主题进行美化的。最后，当 Linux 初始化完成后，会给我们显示一个让我们登录的图形界面，这就是 DM（Display Manager），这个 DM 既是 XServer 的父进程，负责启动 XServer，又是一个 XClient，给出图形化的登录接口。登录成功后，它又是 Gnome Shell 的父进程，负责启动 Gnome Shell。还有，即使在纯字符界面下，也是可以使用 FrameBuffer 获得图形功能的，甚至可以截图和播放视频。唯一的区别，就是在这些模式下，在没有桌面环境的情况下，我们和计算机的交互，往往只能通过 CLI 进行。所以，我给这一篇取名《没有 GUI 的时候应该怎么玩》。

[回到顶部](#)

## 逆天的 GRUB##

Linux 系统启动的过程是这样的：先由 BIOS 启动一个系统引导程序；然后系统引导程序负责把 Linux 的内核加载到内存，同时把 initrd 加载到内存，然后把控制权交给 Linux 的内核；Linux 的内核初始化完成后，将控制权交给 init 程序；init 程序负责启动各种服务。如果要启动图形桌面系统，则 init 先启动一个窗口管理器，由窗口管理器负责用户的登录和验证；用户登录和验证成功后，窗口管理器负责启动 X 服务器和客户端，进入桌面系统。如果是不需要图形桌面系统的 Linux，则 init 启动 login 程序，login 程序负责用户的登录和验证，验证成功后，启动一个 shell。

GRUB 就是目前 Linux 系统使用的系统引导程序，是计算机启动后运行的第一个程序（当然，BIOS 除外）。它在将 Linux 内核加载到内存的时候，还可以向内核传递各种参数。目前的 Linux 发行版使用的 GRUB 都已经是第 2 版了，它的功能和配置都和以前的版本不一样。网上很多文章都是基于以

前的 GRUB Legacy 版本进行的讲解，已经不能适应现在新的形势了。

GRUB 的文档在这里：

<https://www.gnu.org/software/grub/manual/grub/>。从前面的介绍可以看出，GRUB 是计算机启动后运行的第一个程序，这个时候 Linux 的内核还没有加载，其它的程序也都不可能运行。这时有人就会想了，这个 GRUB 的功能应该相当有限吧。我刚开始也是这么想的。但是当我读完前面给出的 GRUB 文档后，我的思想被彻底颠覆了。GRUB 的功能太 TM 强了，简直逆天。

那么这个一开机就启动的简单程序究竟具有哪些让人意想不到的功能呢？请看我列举几条：

- 能够访问任何设备上的数据，不管你是硬盘、软盘还是光盘；
- 能够探测到所有的内存；
- 能够识别大部分的文件系统，不管你是 FAT32、NTFS 还是 ext2/ext3/ext4；
- 能够识别文件系统中的文件，文档中说它可识别大部分可执行文件格式，ELF 什么的根本不在话下；
- 能够使用 .png、.jpg 格式的图片作为背景，说明它能够识别一些图片格式；
- 对字体的支持稍微差一点，好像只能使用 PFF2 格式的字体；
- 当然可以读取和输出硬盘上的文本文件；
- 据说还能播放乐曲；
- 支持联网，可以从网络上启动操作系统；
- 可以支持串口输入输出。



这些功能真的是已经超强了，就快赶上一个操作系统了。重要的是，它还提供了一个非常好用的命令行界面，该命令行界面的使用方法和 Linux 系统中的 Shell 极其接近，也支持编程、支持环境变量，其编程的语法也和 Bash 差不多。再加上 GRUB 提供的丰富的命令，该界面使用起来爽得不要不要的。

[回到顶部](#)

## GRUB 实战 ##

实践出真知，下面以 Ubuntu 为例开始实战。

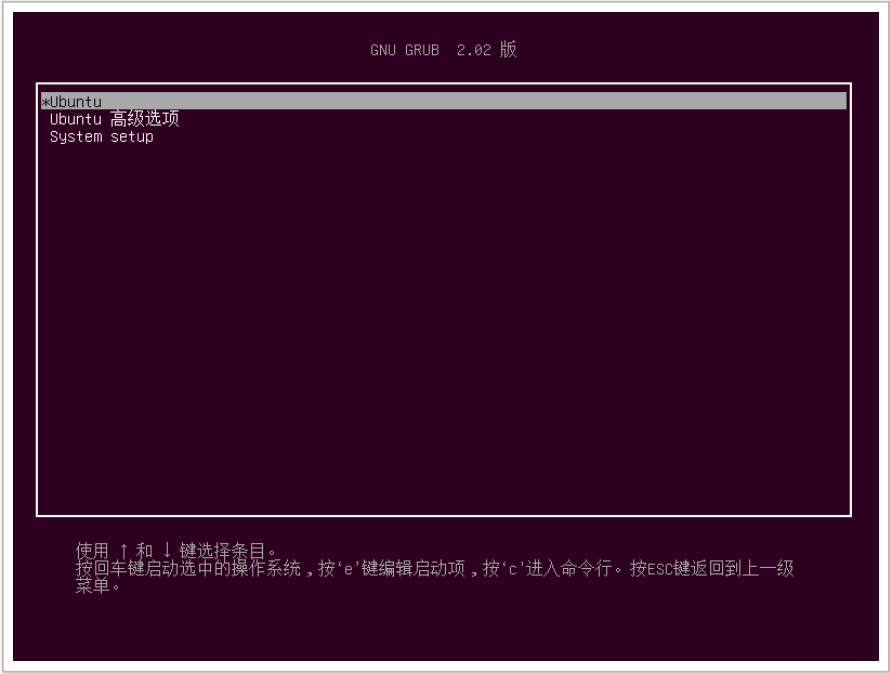
### GRUB 的界面 ###

刚安装好的 Ubuntu 启动时不显示 GRUB 界面，因为它在设置中把它隐藏了。它的启动画面是这样的：

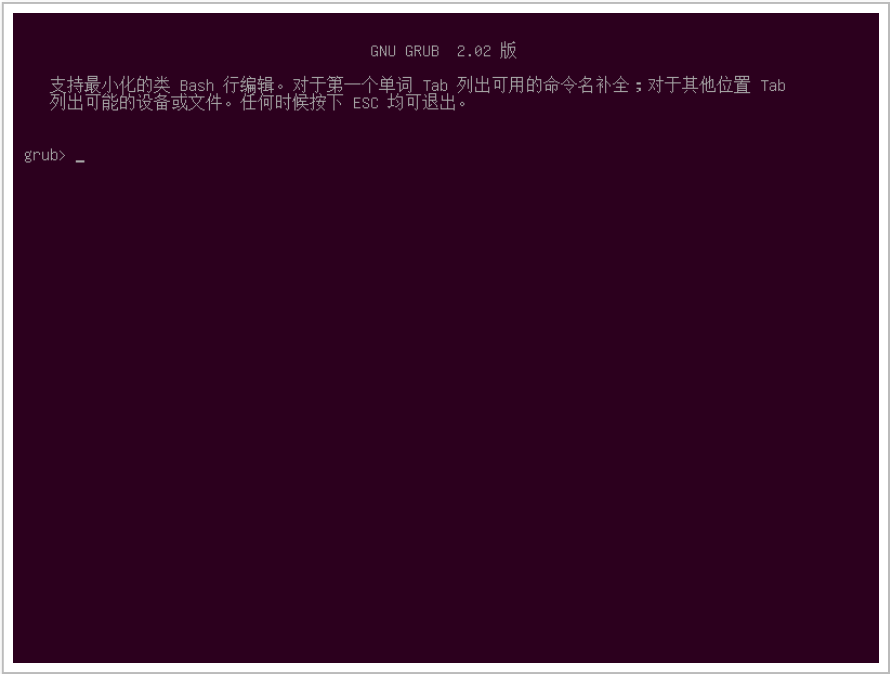


必须按一下 ESC 键，我们才能够看到 GRUB 的菜单，它是这样的：





上面这个界面想必大家已经很熟悉了。在这个界面中，如果按下 c 键，就会切换到 GRUB 的命令行界面，如下：



还有一种情况就是，如果大家在使用 Linux 过程中不小心删除了 `/boot/grub/grub.cfg`，或者配置错误，或者删除了 Linux 系统所在的硬盘分区的数据，使得 GRUB 无法正确加载 Linux 系统，也会自动进入到这个命令行界面。

## GRUB 支持的命令 ###

GRUB 的命令补全功能非常方便，只要按一下 TAB 键，就可以显示它支持的所有命令。命令之后按 TAB 键，可以自动补全文件名。下面是 GRUB 支持的命令，我按 TAB 键调出来的：



```
GNU GRUB 2.02 版

支持最小化的类 Bash 行编辑。对于第一个单词 Tab 列出可用的命令名补全；对于其他位置 Tab
列出可能的设备或文件。任何时候按下 ESC 均可退出。

grub>
可用命令：

. [ acpi all_functional_test appleloader authenticate background_color background_image
backtrace badram blocklist boot break cat cbmemc chainloader clear cmp configfile continue
coreboot_boottime cpuid crc cryptomount cutmem date distrust dump echo eval exit export
extract_entries_configfile extract_entries_source extract_legacy_entries_configfile
extract_legacy_entries_source extract_syslinux_entries_configfile
extract_syslinux_entries_source fakebios false file fix_video functional_test fwsetup gettext
gptsync halt hashsum hdparm hello help hexdump hexdump_random inb initrd initrd16 initrdefi inl
insmod inw keymap keystatus kfreebsd kfreebsd_loadenv kfreebsd_module kfreebsd_module_elf
knetbsd knetbsd_module knetbsd_module_elf kopenbsd kopenbsd_ramdisk legacy_check_password
legacy_configfile legacy_initrd legacy_initrd_nounzip legacy_kernel legacy_password
legacy_source linux linux16 linuxefi list_env list_trusted load_env loadbios loadfont loopback
ls lsacpi lscoreboot lsefi lsefimmap lsefisystab lsfonts lsmmmap lsmod lspci lssal macppcbless
mactelbless md5sum menuentry module module2 multiboot multiboot2 nativedisk net_add_addr
net_add_dns net_add_route net_bootp net_bootp6 net_del_addr net_del_dns net_del_route
net_get_dhcp_option net_ipv6_autoconf net_ls_addr net_ls_cards net_ls_dns net_ls_routes
net_nslookup normal normal_exit outb outl outw parttool password password_pbkdf2 pcidump play
probe read read_byte read_dword read_word reboot regexp return rmmmod save_env search search.file
search.fs_label search.fs_uuid serial set setparams setpci shasum sha256sum sha512sum shift
sleep source submenu syslinux_configfile syslinux_source terminal_input terminal_output terminfo
test test_blockarg testload testspeed time tr true trust unset usb verify_detached videoinfo
videotest write_byte write_dword write_word xnu_devprop_load xnu_kernel xnu_kernel164 xnu_kext
xnu_kextdir xnu_mkext xnu_ramdisk xnu_resume xnu_splash xnu_uuid zfs-bootfs zfsinfo zfskey
grub> _
```

使用 `ls` 命令可以列出目录和文件，使用 `cat` 命令可以输出文本文件的内容。在 GRUB 中，使用 `(hd0, msdos1)` 或者 `(hd0, gpt1)` 识别硬盘分区，使用 `(hd0, gptN)/boot/grub/grub.cfg` 这样的形式识别文件。由于 GRUB 能自动识别根分区，所以我下面的命令中省略掉了指定硬盘分区的部分。如下图：



```


grub> ls /
lost+found/ boot/ swapfile etc/ media/ bin/ dev/ home/ lib/ lib64/ mnt/ opt/ proc/ root/ run/
sbin/ snap/ srv/ sys/ tmp/ usr/ var/ initrd.img initrd.img.old vmlinuz cdrom/
grub> ls /boot
efi/ grub/ System.map-4.15.0-20-generic abi-4.15.0-20-generic config-4.15.0-20-generic
memtest86+.bin memtest86+.elf memtest86+_multiboot.bin retpoline-4.15.0-20-generic
vmlinuz-4.15.0-20-generic abi-4.15.0-30-generic config-4.15.0-30-generic
retpoline-4.15.0-30-generic initrd.img-4.15.0-20-generic System.map-4.15.0-30-generic
vmlinuz-4.15.0-30-generic initrd.img-4.15.0-30-generic
grub> cat /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/sda2 during installation
UUID=663d2f45-3c18-4ecb-9111-378021111111 / ext4 errors=remount-ro 0
# /boot/efi was on /dev/sda1 during installation
UUID=AA00-654A /boot/efi vfat umask=0077 0 1
# swap file
UUID=11111111-1111-1111-1111-111111111111 swap 0 0
grub> _

```

在 GRUB 的启动过程中，`cat /etc/fstab` 命令显示了我系统中硬盘分区的情况。可以看到，我使用的是 GPT 分区格式和 EFI 固件，硬盘分了三个去，第一个分区的挂载点是 `/boot/efi`，并且是 `vfat` 格式的文件系统，第二个分区的挂载点是根目录 `/`，第三个分区是 `swap` 空间。按照 GRUB 的术语，则分区 `(hd0, gpt1)` 是挂载的 `/boot/efi`，分区 `(hd0, gpt2)` 是根目录，分区 `(hd0, gpt3)` 是交换分区。可以看到，GRUB 中硬盘是从 0 开始计数的，而分区是从 1 开始计数的。

## GRUB 的环境变量 ###

我在之前的某一篇随笔中讲过，命令行参数、环境变量、配置文件是对软件进行配置的三驾马车，GRUB 也不例外，它的很多行为也受环境变量控制。下面看一个例子，当我想查看 GRUB 的启动配置文件 `/boot/grub/grub.cfg` 时，使用 `cat` 命令查看该文件的内容，但是由于该文件太长，一个屏幕显示不完，所以只能看到最后几行，如下两图：



```

grub> cat /boot/grub/grub.cfg_

```



```
    }  
  }  
  ### END /etc/grub.d/10_linux ###  
  ### BEGIN /etc/grub.d/20_linux_xen ###  
  ### END /etc/grub.d/20_linux_xen ###  
  ### BEGIN /etc/grub.d/20_memtest86+ ###  
  ### END /etc/grub.d/20_memtest86+ ###  
  ### BEGIN /etc/grub.d/30_os-prober ###  
  ### END /etc/grub.d/30_os-prober ###  
  ### BEGIN /etc/grub.d/30_uefi-firmware ###  
  menuentry 'System setup' $menuentry_id_option 'uefi-firmware' {  
    fusetup  
  }  
  ### END /etc/grub.d/30_uefi-firmware ###  
  ### BEGIN /etc/grub.d/40_custom ###  
  # This file provides an easy way to add custom menu entries. Simply type the  
  # menu entries you want to add after this comment. Be careful not to change  
  # the 'exec tail' line above.  
  ### END /etc/grub.d/40_custom ###  
  ### BEGIN /etc/grub.d/41_custom ###  
  if [ -f ${config_directory}/custom.cfg ]; then  
    source ${config_directory}/custom.cfg  
  elif [ -z "${config_directory}" -a -f $prefix/custom.cfg ]; then  
    source $prefix/custom.cfg;  
  fi  
  ### END /etc/grub.d/41_custom ###  
grub> _
```

这是非常蛋疼的，但还不是最郁闷的，毕竟 `/boot/grub/grub.cfg` 是系统中的一个文件，大不了我进 Linux 后用 `vim` 看。最蛋疼的是某些命令的输出，只能看到最后几行，又不能保存下来，真的让人捉急。就像下面这个例子，我使用 `videoinfo` 命令查看我的 GRUB 支持哪些图形分辨率：



```

grub> videoinfo_
0x12f 320 x 400 x 16 ( 640) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x130 640 x 400 x 16 (1280) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x131 640 x 400 x 16 (1280) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x132 800 x 600 x 16 (1600) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x133 1024 x 768 x 16 (2048) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x134 1152 x 864 x 16 (2304) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x13c 320 x 200 x 32 (1280) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x13d 320 x 400 x 32 (1280) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x13e 640 x 400 x 32 (2560) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x13f 640 x 400 x 32 (2560) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
* 0x140 800 x 600 x 32 (3200) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x141 1024 x 768 x 32 (4096) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x142 1152 x 864 x 32 (4608) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x156 320 x 240 x 8 ( 320) Paletted
0x157 320 x 240 x 16 ( 640) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x158 320 x 240 x 32 (1280) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x159 400 x 300 x 8 ( 416) Paletted
0x15a 400 x 300 x 16 ( 800) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x15b 400 x 300 x 32 (1600) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x15c 512 x 384 x 8 ( 512) Paletted
0x15d 512 x 384 x 16 (1024) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x15e 512 x 384 x 32 (2048) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x15f 854 x 480 x 8 ( 864) Paletted
0x160 854 x 480 x 16 (1728) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x161 854 x 480 x 32 (3424) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x16e 720 x 480 x 8 ( 736) Paletted
0x16f 720 x 480 x 16 (1440) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x170 720 x 480 x 32 (2880) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x171 720 x 576 x 8 ( 736) Paletted
0x172 720 x 576 x 16 (1440) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x173 720 x 576 x 32 (2880) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x174 800 x 480 x 8 ( 800) Paletted
0x175 800 x 480 x 16 (1600) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x176 800 x 480 x 32 (3200) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
Adapter 'VGA Video Driver':
No info available
grub> _

```

这个时候，就只能通过设置环境变量的方法来解决问题了。使用 `set pager=1` 命令设置环境变量 `pager`，让 GRUB 的输出启用分页，如下图：

```

grub> set pager=1
grub> cat /boot/grub/grub.cfg
#
# DO NOT EDIT THIS FILE
#
# It is automatically generated by grub-mkconfig using templates
# from /etc/grub.d and settings from /etc/default/grub
#

### BEGIN /etc/grub.d/00_header ###
if [ -s $prefix/grubenv ]; then
  set have_grubenv=true
  load_env
fi
if [ "${next_entry}" ]; then
  set default="${next_entry}"
  set next_entry=
  save_env next_entry
  set boot_once=true
else
  set default="0"
fi

if [ x"${feature_menuentry_id}" = xy ]; then
  menuentry_id_option="--id"
else
  menuentry_id_option=""
fi

export menuentry_id_option

if [ "${prev_saved_entry}" ]; then
  set saved_entry="${prev_saved_entry}"
  save_env saved_entry
  set prev_saved_entry=
  --MORE--

```

```

grub> set pager=1
grub> videoinfo
List of supported video modes:
Legend: mask/position=red/green/blue/reserved
Adapter 'Cirrus CLGD 5446 PCI Video Driver':
  No info available
Adapter 'Bochs PCI Video Driver':
  No info available
Adapter 'VESA BIOS Extension Video Driver':
  VBE info:   version: 2.0  OEM software rev: 2.0
             total memory: 4096 KiB
0x100 640 x 400 x 8 ( 640) Paletted
0x101 640 x 480 x 8 ( 640) Paletted
0x103 800 x 600 x 8 ( 800) Paletted
0x105 1024 x 768 x 8 (1024) Paletted
0x10e 320 x 200 x 16 ( 640) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x111 640 x 480 x 16 (1280) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x114 800 x 600 x 16 (1600) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x117 1024 x 768 x 16 (2048) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x120 320 x 200 x 8 ( 320) Paletted
0x121 320 x 400 x 8 ( 320) Paletted
0x122 640 x 400 x 8 ( 640) Paletted
0x123 640 x 480 x 8 ( 640) Paletted
0x124 800 x 600 x 8 ( 800) Paletted
0x125 1024 x 768 x 8 (1024) Paletted
0x126 1152 x 864 x 8 (1152) Paletted
0x12e 320 x 200 x 16 ( 640) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x12f 320 x 400 x 16 ( 640) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x130 640 x 400 x 16 (1280) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x131 640 x 480 x 16 (1280) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x132 800 x 600 x 16 (1600) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x133 1024 x 768 x 16 (2048) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x134 1152 x 864 x 16 (2304) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x13c 320 x 200 x 32 (1280) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x13d 320 x 400 x 32 (1280) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
--MORE--

```

我们还可以通过不带参数的 `set` 命令显示所有可用的环境变量，如下图：

```

grub> set
?=0
cmdpath=(hd0,gpt1)/EFI/ubuntu
color_highlight=black/light-gray
color_normal=light-gray/black
config_directory=(hd0,gpt1)/EFI/ubuntu
config_file=(hd0,gpt1)/EFI/ubuntu/grub.cfg
default=0
feature_200_final=y
feature_all_video_module=y
feature_chainloader_bpb=y
feature_default_font_path=y
feature_menuentry_id=y
feature_menuentry_options=y
feature_nativdisk_cmd=y
feature_ntldr=y
feature_platform_search_hint=y
feature_timeout_style=y
font=unicode
gfxmode=800x600x32
grub_cpu=x86_64
grub_platform=efi
have_grubenv=true
lang=zh_CN
linux_gfx_mode=keep
locale_dir=(hd0,gpt2)/boot/grub/locale
menu_color_highlight=black/light-gray
menu_color_normal=white/black
menuentry_id_option="--id
net_default_ip=(null)
net_default_mac=(null)
net_default_server=
pager=1
prefix=(hd0,gpt2)/boot/grub
--MORE--

```

也可以使用 `echo` 命令输出某一个环境变量，如下图：

```
grub> echo $root
hd0,gpt2
grub>
grub> ls (hd0,gpt2)/
lost+found/ boot/ swapfile etc/ media/ bin/ dev/ home/ lib/ lib64/ mnt/ opt/ proc/ root/ run/
sbin/ snap/ srv/ sys/ tmp/ usr/ var/ initrd.img initrd.img.old vmlinuz cdrom/
grub>
grub> ls (hd0,gpt1)/
efi/
grub> _
```

## 更改分辨率 ###

我们可以控制 GRUB 显示界面的分辨率，还可以通过 GRUB 控制 Linux 启动进入字符模式后的分辨率。前提条件是要看我们的 BIOS 和显卡支持哪些模式。可以通过 `videoinfo` 命令查看，如下图：

我使用的是虚拟机，因为玩 GRUB 不使用虚拟机无法截图啊。如果采取的是 EFI 固件，则其输出如下：

```
grub> videoinfo
List of supported video modes:
Legend: mask/position-red/green/blue/reserved
Adapter: 'Bochs PCI Video Driver':
  No info available
Adapter: 'Cirrus CLGD 5446 PCI Video Driver':
  No info available
Adapter: 'EFI GOP driver':
0x000 320 x 200 x 32 (1280) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x001 640 x 480 x 32 (2560) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
* 0x002 800 x 600 x 32 (3200) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x003 1024 x 768 x 32 (4096) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x004 1152 x 864 x 32 (4608) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x005 320 x 240 x 32 (1280) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x006 400 x 300 x 32 (1600) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x007 512 x 384 x 32 (2048) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x008 854 x 480 x 32 (3416) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x009 720 x 480 x 32 (2880) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x00a 720 x 576 x 32 (2880) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x00b 800 x 480 x 32 (3200) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
Adapter: 'EFI UGA driver':
  No info available
grub> _
```

如果采取的是 Legacy BIOS，则其输出如下：

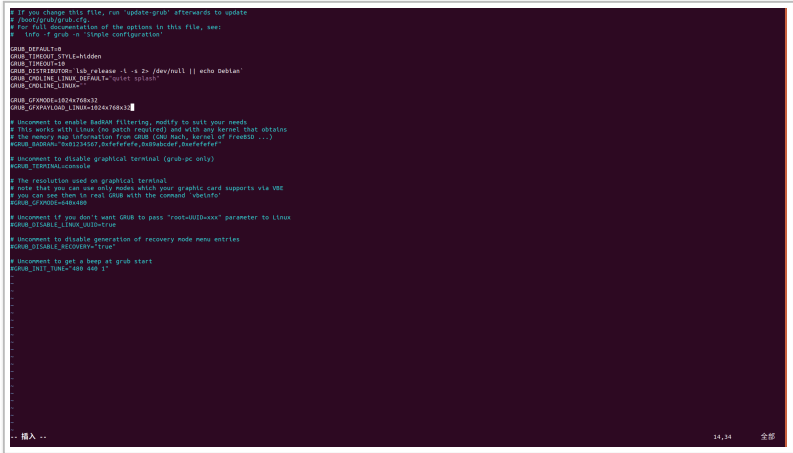
```
grub> set pager=1
grub> videoinfo
List of supported video modes:
Legend: mask/position-red/green/blue/reserved
Adapter: 'Cirrus CLGD 5446 PCI Video Driver':
  No info available
Adapter: 'Bochs PCI Video Driver':
  No info available
Adapter: 'VESA BIOS Extension Video Driver':
VBE info:  version: 2.0  OEM software rev: 2.0
           total memory: 4096 KiB
0x100 640 x 400 x 8 ( 640) Paletted
0x101 640 x 480 x 8 ( 640) Paletted
0x103 800 x 600 x 8 ( 800) Paletted
0x105 1024 x 768 x 8 (1024) Paletted
0x10e 320 x 200 x 16 ( 640) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x111 640 x 480 x 16 (1280) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x114 800 x 600 x 16 (1600) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x117 1024 x 768 x 16 (2048) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x120 320 x 200 x 8 ( 320) Paletted
0x121 320 x 400 x 8 ( 320) Paletted
0x122 640 x 400 x 8 ( 640) Paletted
0x123 640 x 480 x 8 ( 640) Paletted
0x124 800 x 600 x 8 ( 800) Paletted
0x125 1024 x 768 x 8 (1024) Paletted
0x126 1152 x 864 x 8 (1152) Paletted
0x12e 320 x 200 x 16 ( 640) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x12f 320 x 400 x 16 ( 640) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x130 640 x 400 x 16 (1280) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x131 640 x 480 x 16 (1280) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x132 800 x 600 x 16 (1600) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x133 1024 x 768 x 16 (2048) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x134 1152 x 864 x 16 (2304) Direct color, mask: 5/6/5/0 pos: 11/5/0/0
0x13c 320 x 200 x 32 (1280) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
0x13d 320 x 400 x 32 (1280) Direct color, mask: 8/8/8/8 pos: 16/8/0/24
--MORE--
```

可以看到，如果使用的是 Legacy BIOS，它的显示模式是由 'VESA BIOS Extension Video Driver' 提供支持的。如果使

用的是 EFI，则其显示模式是由'EFI GOP driver' 提供支持的。在我的虚拟机中，它们能提供的最高分辨率也只有 1152 x 864，远远达不到 1920 x 1080。但是在我的物理机中，都是可以达到 1920 x 1080。而且貌似只能进入 1920 x 1080，想改小还改不了。在物理机上，想通过改小分辨率，然后利用显示器的放大功能来放大字体的梦想是破灭了。

在虚拟机中，我要做的是把分辨率改大，至少让我完全进入 Linux 字符界面的时候有个 1024 x 768 的分辨率吧，不然字符界面用起来岂不是太憋屈。可以通过修改

`/etc/default/grub` 文件，然后调用 `sudo update-grub` 命令更新 GRUB。如下图，使用 `sudo vim /etc/default/grub` 修改配置文件：



在上面的这个文件中的注释里，也写得很明白了，要修改 GRUB 和 Linux 字符界面的分辨率，可以通过修改

`GRUB_GFXMODE` 和 `GRUB_GFXPAYLOAD_LINUX` 参数来设置，而且千万不要设置 `GRUB_TERMINAL=console`，不然就真的进入只有文字的文字模式了，没有 Graphic 的支持，还谈啥分辨率呢。

然后重启系统，可以看到我们的 GRUB 界面变大了一圈，如下两图：



下面进入 Linux 的字符界面，进入 Linux 字符界面的方式是启动进入 Linux 后，使用 `sudo systemctl set-default multi-user.target`，然后重启，在 1024 x 768 的分辨率下开一个 vim 看看，如下图：



- 装逼必备
- 省资源，服务器一般不安装图形界面
- 图形界面崩溃后紧急救援

## 进入字符界面的正确方式 ###

目前新的 Linux 发行版基本上都使用 Systemd 作为 init 程序，不再使用 SysV init 和 Upstart init。所以如果能让系统启动后直接进入字符界面，应该使用如下命令：

```
sudo systemctl set-default multi-user.target
```

反过来，要让系统启动后直接进入图形界面，应该使用如下命令：

```
sudo systemctl set-default graphical.target
```

另外，Linux 本身提供有虚拟控制台的功能，使用 `Ctrl + Alt + F1` 到 `Ctrl + Alt + F7` 进行切换，其中有一个是图形界面，剩下的是字符界面。图形界面玩崩溃了，就不得不使用 `Ctrl + Alt + F3` 切换到字符界面进行救援。



## 关于 Framebuffer###

字符界面分两种，一种是不开启 Framebuffer 的，另一种是开启 Framebuffer 的。Framebuffer 是一种图形驱动，不开启 Framebuffer 就是真的全字符，不能改变分辨率，不能显示图像，不能截图。目前最新的 Linux 发行版默认开启 Framebuffer。控制 Framebuffer 开启和关闭，以及分辨率的方法，是设置 Grub2 的参数。修改 `/etc/default/grub` 文件，添加如下参数可以设置分辨率：

```
GRUB_GFXPAYLOAD_LINUX=1024x768x32
```

然后使用如下命令更新 GRUB2 配置：

```
sudo update-grub
```

其中的分辨率必须是我们的硬件支持的。可以通过 GRUB2 命令行中的 `videoinfo` 命令查看我们的硬件支持的分辨率。

如果要关闭 Framebuffer，则这样更改 GRUB2 的配置：

```
GRUB_GFXPAYLOAD_LINUX=text
```

同样，需要：

```
sudo update-grub
```

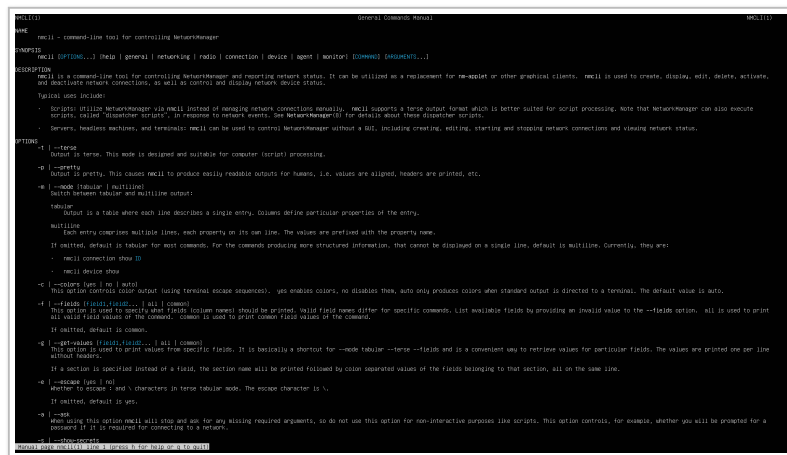
然后重启。

[回到顶部](#)

## 字符界面下联网 ##

以前在图形界面的时候，设置个网络、连接个 wifi 非常简单，玩儿似的，结果一进入纯字符界面就抓瞎。不联网，就不能下载和安装软件包，后面就玩不下去了。所以进入纯字符界

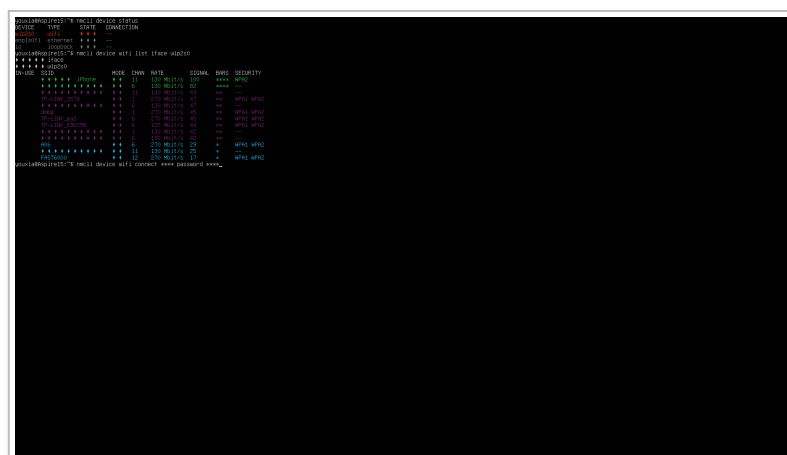
面后，要解决的第一件事就是怎么联网。说到管理网络的工具，大家可以列举一大堆，什么 `ipconfig`、`iwconfig`、`ip` 等等。但是，在最新的 Linux 发行版中，已经是使用 `NetManager` 管理网络了。通过阅读 `NetManager` 的文档，可以知道它提供一个功能很强大的命令行工具，那就是 `nmcli`。通过 `man nmcli` 可以查看该工具的用法。如下图：



使用如下命令可以查看可用的 wifi 热点以及连接 wifi:

```
nmcli device status
nmcli device wifi list iface 网卡名称 #查看可连接wifi
nmcli device wifi connect **** password **** #连接 wifi,
```

如下图:



连上了网，Ubuntu 就可以在纯字符界面下起飞了。

[回到顶部](#)

## 使用 fbterm##

通过上面的截图，发现两个问题：

- 在我的 1920x1080 的笔记本电脑屏幕上，默认的字实在太小；
- 不能显示中文，上图中 wifi 热点名称含有中文的，都显示不出来。

解决办法是使用 fbterm。一举解决字体大小问题和中文显示问题。先安装 fbterm，使用如下命令：

```
sudo apt-get install fbterm
```

先使用 `sudo fbterm` 启动 fbterm 一次，再用 `exit` 命令退出，这样，fbterm 会自动生成一个默认的配置文件的

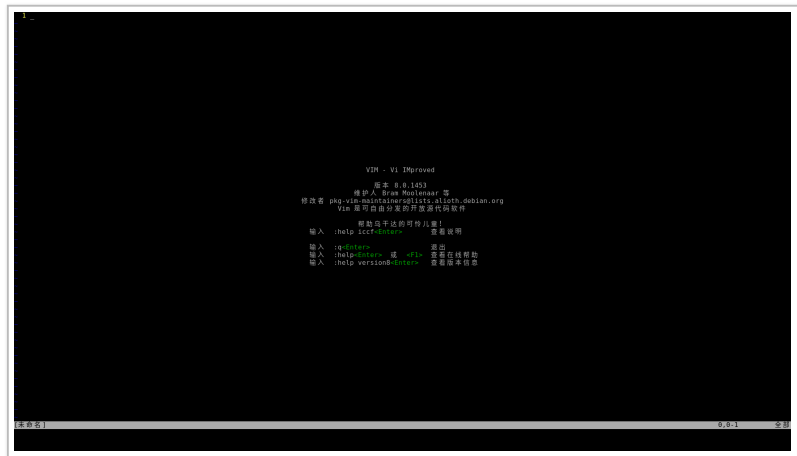
`~/.fbtermrc`，然后修改 `~/.fbtermrc` 配置文件中的两行，设置使用的字体和字体大小，如下：

```
font-names=DejaVu\ Sans\ Mono  
font-size=16
```

然后使用 `sudo fbterm` 命令启动 fbterm，就可以了。下面是看对比图，使用 fbterm 之前，Vim 的启动界面是不能显示中文的：



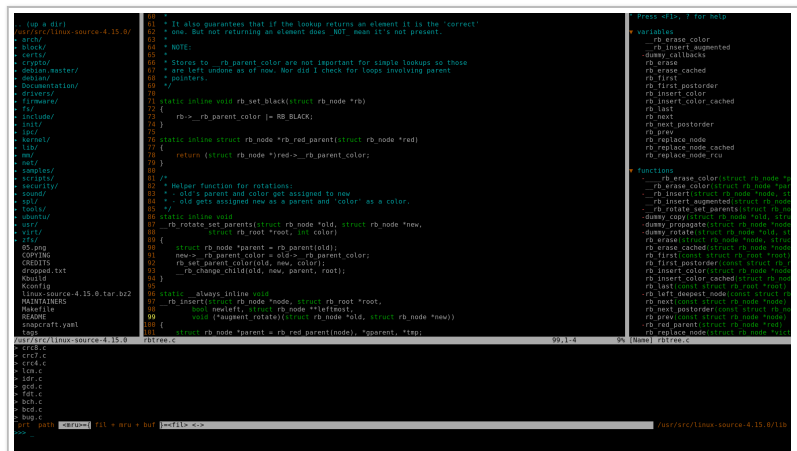
使用 fbterm 之后，中文可以正常显示：



使用 fbterm 之前，阅读代码是这样的：



字非常的小，NERDTree 和 Tagbar 里面的符号显示也有问题。使用 fbterm 之后，就很漂亮了，如下图：



这才是全高清屏该有的显示效果嘛。关于在 fbterm 下输入中文，我尝试过 fbterm-ucimf，也尝试过 fcitx-frontent-fbterm，都没有成功。后来我就不试了，反正我也没有在全字符界面下输入中文的需求。

[回到顶部](#)

## 显示 Framebuffer 的信息 ##

使用 fbset 可以查看 Framebuffer 的信息，包括 Framebuffer 是否开启，分辨率是多少，由哪个内核模块提供支持等。使用如下命令安装 fbset：

```
sudo apt-get install fbset
```

使用 `sudo fbset -i` 命令查看 Framebuffer 的信息，如下图：



```
root@hiperis:~# fbset -i
mode "1024x1000"
geometry 1024 1000 1024 1000 32
timing 0 0 0 0 0 0
hsync 0
vsync 0
rgb 0/16,0/16,0/16,0/0
endmode

Frame Buffer Device Information:
Name      : intelrnfb
Address   : 00000000
Size      : 8384000
Type      : PACKED_PIXELS
Visual    : TRUECOLOR
XResStep  : 1
YResStep  : 1
XResStep  : 1
YResStep  : 1
LineLength : 7680
Accelerator : No
root@hiperis:~#
```

[回到顶部](#)

## 在 Framebuffer 下截图 ##

使用 fbgrab 可以在 Framebuffer 下进行截图。使用如下命令安装 fbgrab：

```
sudo apt-get install fbgrab
```

使用 fbgrab 命令的方式如下：

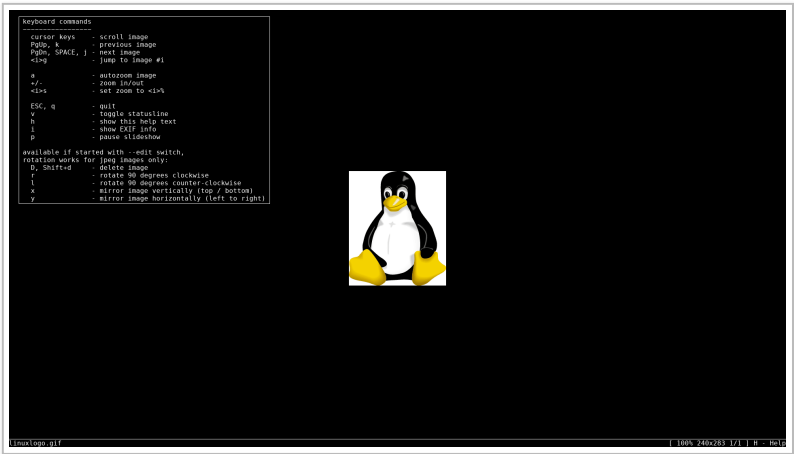
```
sudo fbgrab -c N filename.png      #对 /dev/ttyN 对应的终端设
sudo fbgrab -C N filename.png      #先跳转到 /dev/ttyN 对应的:
sudo fbgrab -s N filename.png      #先等待 N 秒, 再进行截图
```

我前面的图片都是使用 fbgrab 截的，这里就不贴图了。

[回到顶部](#)

## 在 Framebuffer 下查看图片 ##

使用 fbi 可以在 Framebuffer 下查看图片，同样使用 `sudo apt-get install fbi` 安装这个软件。在看图界面下按 H 键，还会显示帮助信息。如下图：



[回到顶部](#)

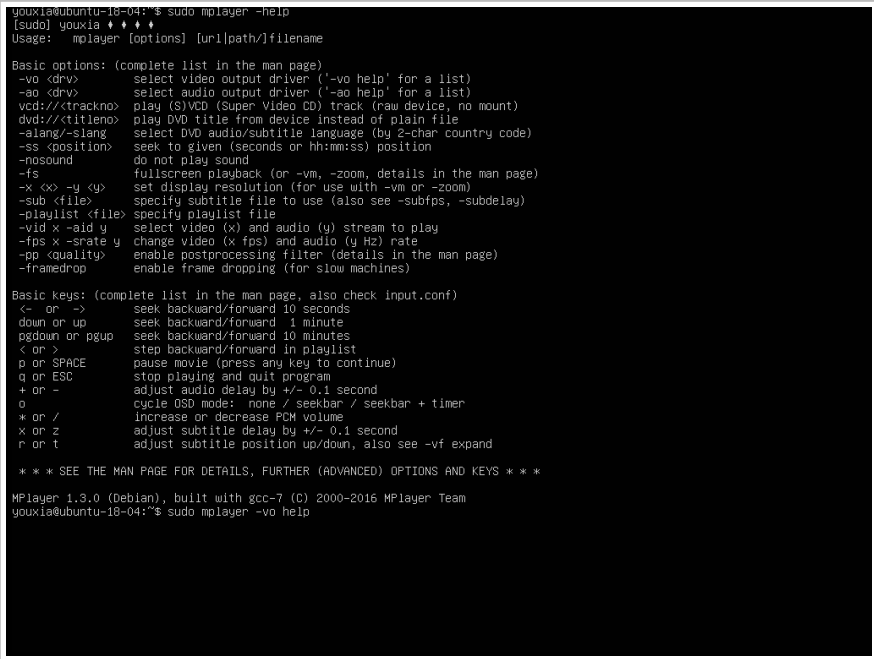
## 在纯字符界面下上网 ##

使用老牌的上网工具 w3m，安装方式 `sudo apt-get install w3m`，然后使用 `w3m https://www.cnblogs.com/` 就可以访问博客园了，只有文字哦，图片就不要想了。效果如下图：

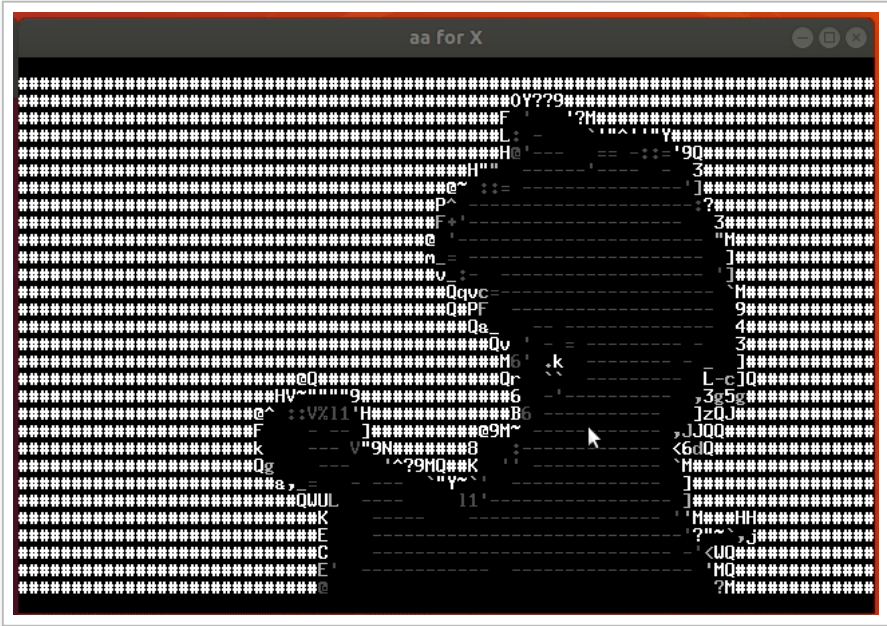


视频播放 ##

使用 mplayer 播放器可以播放视频，通过 `mplayer -vo help` 命令可以看到，mplayer 支持很多种视频驱动，而 Framebuffer 正是其中一种。使用 `sudo mplayer -vo fbdev2 badapple.mp4` 播放 Bad Apple 的 PV 视频，效果如下：



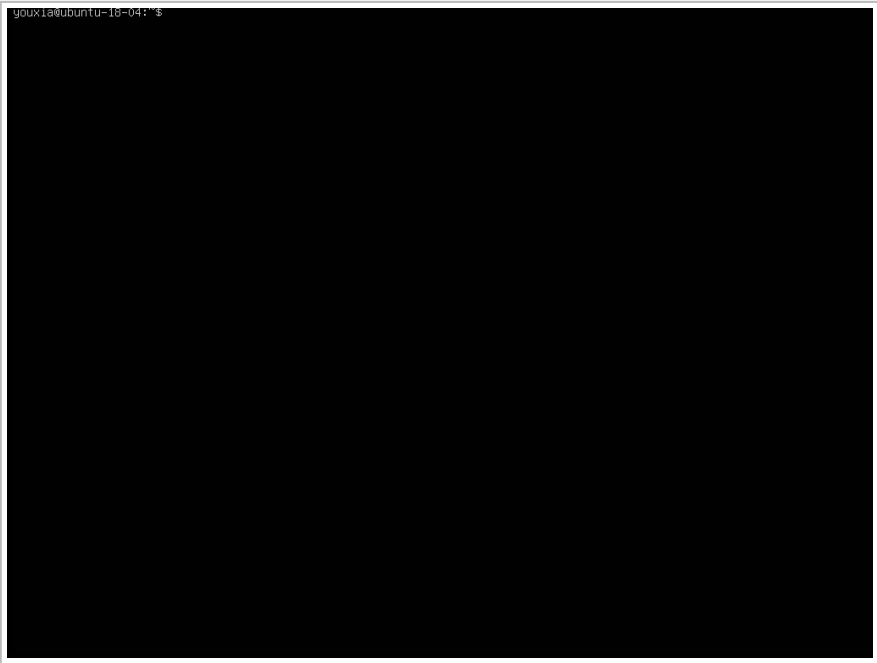
因为这里要录制 gif 动画，所以我使用了虚拟机运行 Linux，Framebuffer 的分辨率设置为 1024x768。同样，通过上面的 `mplayer -vo help` 命令，还可以看到 mplayer 支持使用 libaa 库，将视频播放为字符画。据我观察，只有在图形界面下效果才可以，纯字符界面不行。使用 `mplayer -vo aa -moniterpixelaspect 0.5 badapple.mp4` 播放，效果如下：



[回到顶部](#)

## 屏保和装逼 ##

当然是 cmatrix 啦。效果如下：





[回到顶部](#)

## 版权声明 ##

该随笔由京山游侠在 2018 年 11 月 02 日发布于博客园，引用请注明出处，转载或出版请联系博主。QQ 邮箱：  
1841079@qq.com

---

全文完

---

本文由 简悦 SimpRead 优化，用以提升阅读体验

使用了 全新的简悦词法分析引擎<sup>beta</sup>，[点击查看详细说明](#)

