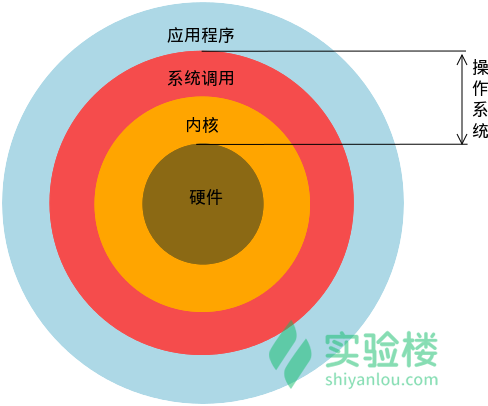
# Linux系统简介

## 1.2 知识点

* linux为何物
* linux历史简介
* linux重要人物
* linux与windows的不同
* 如何学习linux

### linux为何物

Linux 就是一个操作系统，就像你多少已经了解的 Windows（xp，7，8）和 Mac OS 。至于操作系统是什么，就不用过多解释了，如果你学习过前面的入门课程，应该会有个基本概念了，这里简单介绍一下操作系统在整个计算机系统中的角色。



我们的 Linux 主要是系统调用和内核那两层。当然直观地看，我们使用的操作系统还包含一些在其上运行的应用程序，比如文本编辑器、浏览器、电子邮件等。

### Linux历史

操作系统始于二十世纪五十年代，当时的操作系统能运行批处理程序。批处理程序不需要用户的交互，它从文件或者穿孔卡片读取数据，然后输出到另外一个文件或者打印机。

二十世纪六十年代初，交互式操作系统开始流行。它不仅仅可以交互，还能使多个用户从不同的终端同时操作主机。这样的操作系统被称作分时操作系统，它的出现对批处理操作系统是个极大的挑战。许多人尝试开发分时操作系统， 其中包括一些大学的研究项目和商业项目。当时有个项目叫做 Multics ，它的技术在当时很具有创新性。 Multics 项目的开发并不顺利，它花费了远超过预计的资金，却没有在操作系统市场上占到多少份额。而参加该项目的一个开发团体——贝尔实验室退出了这个项目。他们在退出后开发了他们自己的一个操作系统—— UNIX 。

UNIX 最初免费发布并因此在大学里受到欢迎。后来，UNIX 实现了 TCP/IP 协议栈，成为了早期工作站的操作系统的一个流行选择。

1990 年，UNIX 在服务器市场上尤其是大学校园中成为主流操作系统，许多校园都有 UNIX 主机，当然还包括一些研究它的计算机系的学生。这些学生都渴望能在自己的电脑上运行 UNIX 。不幸的是，从那时候开始，UNIX 开始变得商业化，它的价格也变得非常昂贵。而唯一低廉的选择就是 MINIX，这是一个功能有限的类似 UNIX 的操作系统，作者 Andrew Tanenbaum 开发它的目的是用于教学。

1991 年 10 月，Linus Torvalds（Linux 之父）在赫尔辛基大学接触 UNIX，他希望能在自己的电脑上运行一个类似的操作系统。可是 UNIX 的商业版本非常昂贵，于是他从 MINIX 开始入手，计划开发一个比 MINIX 性能更好的操作系统。很快他就开始了自己的开发工作。他第一次发行的版本迅速吸引了一些黑客。尽管最初的 Linux 并没有多少用处，但由于一些黑客的加入使它很快就具有了许多吸引人的特性，甚至一些对操作系统开发不感兴趣的人也开始关注它。

Linux 本身只是操作系统的内核。内核是使其它程序能够运行的基础。它实现了多任务和硬件管理，用户或者系统管理员交互运行的所有程序实际上都运行在内核之上。其中有些程序是必需的，比如说，命令行解释器（shell），它用于用户交互和编写 shell 脚本。 Linux 没有自己去开发这些应用程序，而是使用已有的自由软件。这减少了搭建开发环境所需花费的工作量。实际上，他经常改写内核，使得那些程序能够更容易地在 Linux 上运行。许多重要的软件，包括 C 编译器，都来自于自由软件基金 GNU 项目。GNU 项目开始于 1984 年，目的是为了开发一个完全类似于 UNIX 的免费操作系统。为了表扬 GNU 对 Linux 的贡献，许多人把 Linux 称为 GNU/Linux（GNU 有自己的内核）。

1992－1993 年，Linux 内核具备了挑战 UNIX 的所有本质特性，包括 TCP/IP 网络，图形界面系统（X window )，Linux 同样也吸引了许多行业的关注。一些小的公司开始开发和发行 Linux，有几十个 Linux 用户社区成立。1994 年，Linux 杂志也开始发行。

Linux 内核 1.0 在 1994 年 3 月发布，内核的发布要经历许多开发周期，直至达到一个稳定的版本。

下面列举一些 Linux 诞生大事件：

* 1965 年，Bell 实验室、MIT、GE（通用电气公司）准备开发 Multics 系统，为了同时支持 300 个终端访问主机，但是 1969 年失败了；

那时候并没有鼠标、键盘，输入设备，只有卡片机。因此，如果要测试某个程序，则需要将读卡纸插入卡片机，如果有错误，还需要重新来过；Multics：Multiplexed Information and Computing Service；

* 1969 年，Ken Thompson（C语言之父）利用汇编语言开发了 File Server System（Unics，即 UNIX 的原型）；

因为汇编语言对于硬件的依赖性，因此只能针对特定硬件； 只是为了移植一款“太空旅游”的游戏；

* 1973 年，Dennis Ritchie 和 Ken Thompson 发明了 C 语言，而后写出了 UNIX 的内核；

将 B 语言改成 C 语言，由此产生了 C 语言之父；90% 的代码是 C 语言写的，10% 的代码用汇编语言写的，因此移植时只要修改那 10% 的代码即可；

* 1977 年，Berkeley 大学的 Bill Joy 针对他的机器修改了 UNIX 源码，称为BSD（Berkeley Software Distribution）；

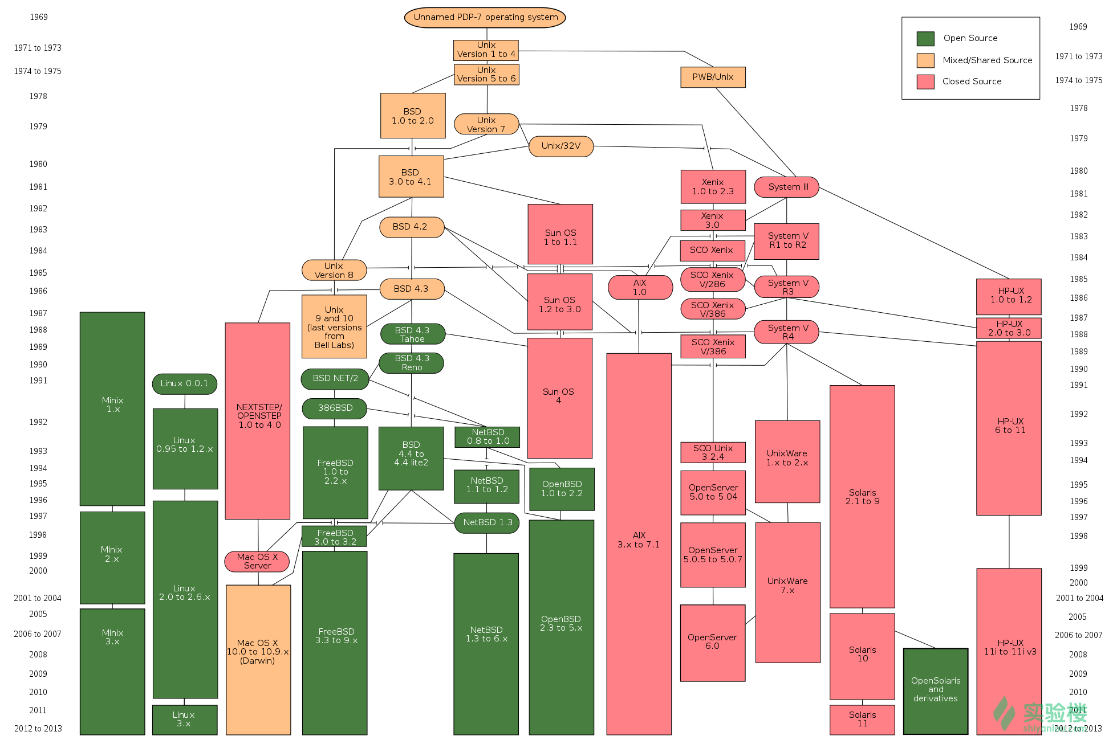
Bill Joy 是 Sun 公司的创始人；

* 1979 年，UNIX 发布 System V，用于个人计算机；
* 1984 年，因为 UNIX 规定“不能对学生提供源码”，Tanenbaum 老师自己编写兼容于 UNIX 的 Minix，用于教学；
* 1984 年，Stallman 开始 GNU（GNU's Not Unix）项目，创办 FSF（Free Software Foundation）基金会；

产品：GCC、Emacs、Bash Shell、GLIBC；倡导“自由软件”；GNU 的软件缺乏一个开放的平台运行，只能在 UNIX 上运行；自由软件指用户可以对软件做任何修改，甚至再发行，但是始终要挂着 GPL 的版权；自由软件是可以卖的，但是不能只卖软件，而是卖服务、手册等；

* 1985 年，为了避免 GNU 开发的自由软件被其他人用作专利软件，因此创建 GPL（General Public License）版权声明；
* 1988 年，MIT 为了开发 GUI，成立了研发 XFree86 的组织；
* 1991 年，芬兰赫尔辛基大学的研究生 Linus Torvalds 基于 gcc、bash 开发了针对 386 机器的 Linux 内核；
* 1994 年，Torvalds 发布 Linux-v1.0；
* 1996 年，Torvalds 发布 Linux-v2.0，确定了 Linux 的吉祥物：企鹅。

UNIX进化史（UNIX大家族族谱1969-2013）：



## Linux历史人物

### 1. Ken Thompson：C 语言之父和 UNIX 之父



### 2. Dennis Ritchie：C 语言之父和 UNIX 之父



### 3. Stallman：著名黑客，GNU 创始人，开发了 Emacs、gcc、bash shell



### 4. Bill Joy：BSD 开发者



### 5. Tanenbaum：Minix 开发者



### 6. Linus Torvalds：Linux 之父，芬兰赫尔辛基大学



## Linux和windows不同

### 1. 免费与收费

* 最新正版 Windows 10，需要付费购买；
* Linux 免费或少许费用。

### 2. 软件与支持

* Windows 平台：数量和质量的优势，不过大部分为收费软件；由微软官方提供重要支持和服务；
* Linux 平台：大都为开源自由软件，用户可以修改定制和再发布，由于基本免费没有资金支持，部分软件质量和体验欠缺；由全球所有的 Linux 开发者和自由软件社区提供支持。

### 3. 安全性

* Windows 平台：三天两头打补丁安装系统安全更新，还是会中病毒木马；
* Linux 平台：要说 Linux 没有安全问题，那当然是不可能的，这一点仁者见仁智者见智，相对来说肯定比 Windows 平台要更加安全，使用 Linux 你也不用装某杀毒、某毒霸。

### 4. 使用习惯

* Windows：普通用户基本都是纯图形界面下操作使用，依靠鼠标和键盘完成一切操作，用户上手容易，入门简单；
* Linux：兼具图形界面操作（需要使用带有桌面环境的发行版）和完全的命令行操作，可以只用键盘完成一切操作，新手入门较困难，需要一些学习和指导（这正是我们要做的事情），一旦熟练之后效率极高。

### 5. 可定制性

* Windows：这些年之前算是全封闭的，系统可定制性很差；
* Linux：你想怎么做就怎么做，Windows 能做到得它都能，Windows 做不到的，它也能。

### 6. 应用范畴

或许你之前不知道 Linux ，要知道，你之前在 Windows 使用百度、谷歌，上淘宝，聊 QQ 时，支撑这些软件和服务的，是后台成千上万的 Linux 服务器主机，它们时时刻刻都在忙碌地进行着数据处理和运算，可以说世界上大部分软件和服务都是运行在 Linux 之上的。

### 7. Windows 没有的

* 稳定的系统
* 安全性和漏洞的快速修补
* 多用户
* 用户和用户组的规划
* 相对较少的系统资源占用
* 可定制裁剪，移植到嵌入式平台（如安卓设备）
* 可选择的多种图形用户界面（如 GNOME，KDE）

### 8. Linux 没有的

* 特定的支持厂商
* 足够的游戏娱乐支持度
* 足够的专业软件支持度

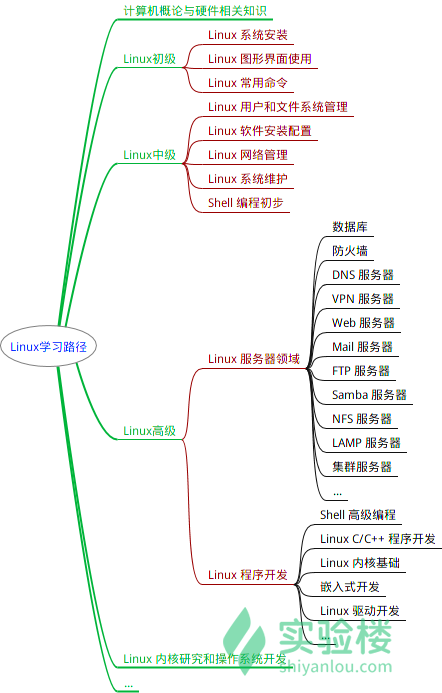
## 如何学习linux

### 1. 学习心态

* 明确目的：你是要用 Linux 来干什么，搭建服务器、做程序开发、日常办公，还是娱乐游戏；
* 面对现实：Linux 大都在命令行下操作，能否接受不用或少用图形界面；
* 是学习 Linux 操作系统本身还是某一个 Linux 发行版（[Ubuntu](http://www.ubuntu.com/" \t "_blank)，[CentOS](http://www.centos.org/" \t "_blank)，[Fedora](http://fedoraproject.org/" \t "_blank)，[OpenSUSE](http://www.opensuse.org/" \t "_blank)，[Debian](http://www.debian.org/" \t "_blank)，[Mint](http://linuxmint.com/" \t "_blank) 等等），如果你对发行版的概念或者它们之间的关系不明确的话可以参看 [Linux 发行版](http://baike.baidu.com/view/897468.htm" \t "_blank)。

### 2. 注重基础，从头开始

大致的学习路径如下：



# 基本概念和操作

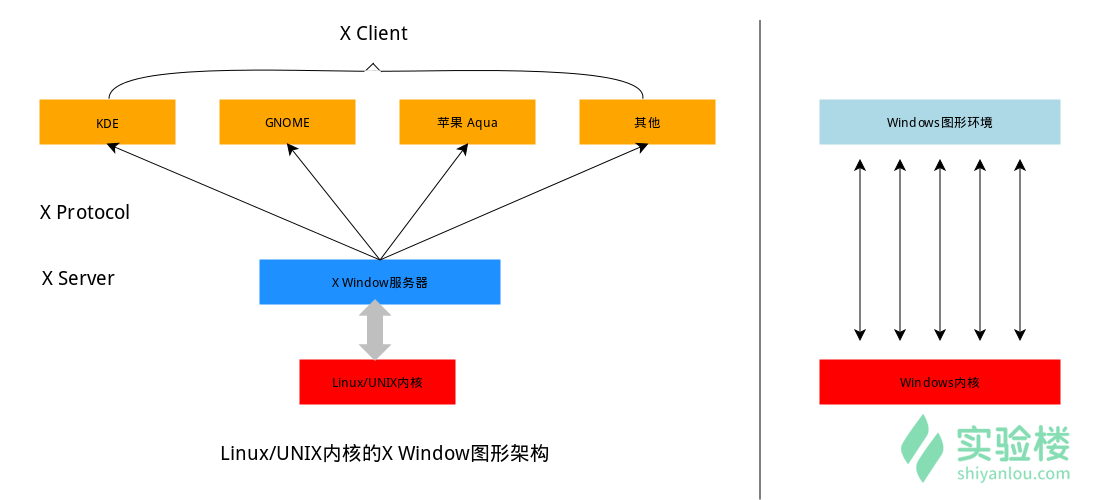
## 1.2 实验知识点

* Linux 基本命令
* 通配符的使用
* 查看帮助文档

## 桌面环境

### 1.Linux 桌面环境介绍

相对于现在的 Windows 系统，UNIX/Linux 本身是没有图形界面的，我们通常在 UNIX/Linux 发行版上看到的图形界面实际都只是运行在 Linux 系统之上的一套软件，类似 Windows95 之前的 Windows 的图形界面实则也只是运行在 DOS 环境的一套软件。而 Linux 上的这套软件以前是 XFree86，现在则是 xorg（X.Org），而这套软件又是通过 X 窗口系统（X Window System，也常被称为 X11 或 X）实现的，X 本身只是工具包及架构协议，而 xorg 便是 X 架构规范的一个实现体，也就是说它是实现了 X 协议规范的一个提供图形界面服务的服务器，就像实现了 http 协议提供 web 服务的 Apache 。如果只有服务器也是不能实现一个完整的桌面环境的，当然还需要一个客户端，我们称为 X Client，像如下几个大家熟知也最流行的实现了客户端功能的桌面环境 **KDE**，**GNOME**，**XFCE**，**LXDE** 。其中就有你看到的，实验楼目前使用的 **XFCE** 桌面环境，部分老用户可能可以回想起，实验楼之前使用的环境是 **LXDE** 。这也意味着在 Linux 上你可以自己选择安装不同的桌面环境，甚至可以定制自己的专属桌面。



这里讲这么多，主要是为了帮助你更好地理解 Linux 的桌面环境是一个怎样的概念，以及它跟 Windows 操作系统桌面的差异，实际它们之间是有着本质的差别的，希望清楚这些内容之后可以让你忽略那些体验上的差异，专心把精力放到 Linux 系统本身或者你正准备学习的某一门技术之上，这门实验课后面的内容和实验楼的其它课程便是为了帮助你实现以上两个目标。

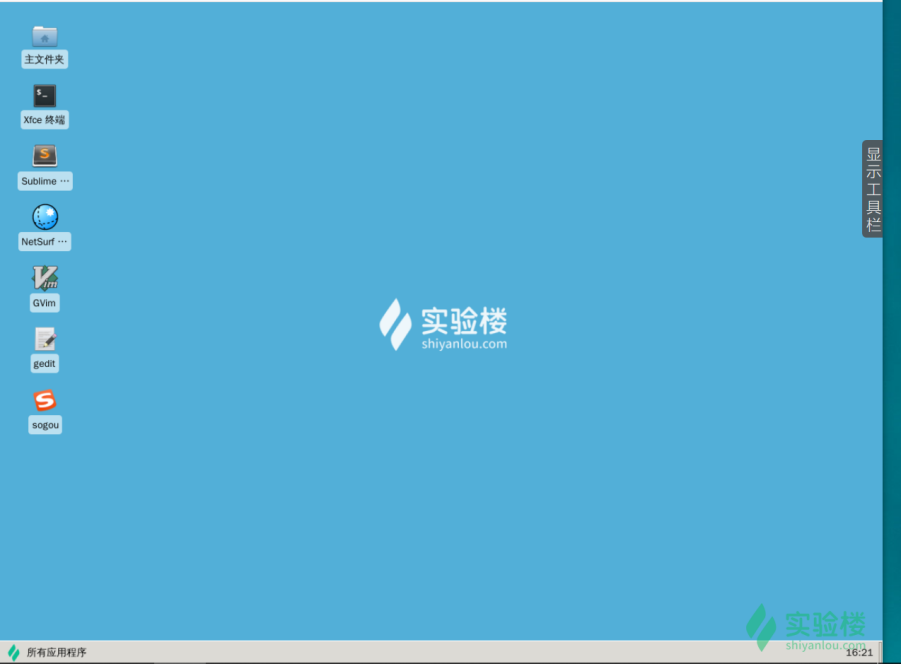
### 2.Linux 桌面环境的使用

如果你对当前实验桌面环境（xfce）的使用有任何困难，建议你先学习[实验楼入门基础课程](http://shiyanlou.com/courses/63" \t "_blank)，对于其它桌面环境的使用，不在本课程的范围之内，有兴趣的用户可以在自己的计算机中安装完整的 Linux 发行版或其 Live CD 体验。

## Linux终端

### 终端的概念

通常我们在使用 Linux 时，并不是直接与系统打交道，而是通过一个叫做 Shell 的中间程序来完成的，在图形界面下为了实现让我们在一个窗口中完成用户输入和显示输出，Linux 系统还提供了一个叫做终端模拟器的程序（Terminal）。 下面是几个比较常见的终端模拟器： gnome-terminal，Konsole，xterm，rxvt，kvt，nxterm 和 eterm 。目前我们的实验中的终端程序是 xfce 桌面环境自带的 xfce-terminal 。不过要注意的是这里所说的终端（Terminal）和控制台（Console）是有区别的。

终端本质上是对应着 Linux 上的 /dev/tty 设备，Linux 的多用户登陆就是通过不同的 /dev/tty 设备完成的，Linux 默认提供了 6 个纯命令行界面的 “terminal”（准确的说这里应该是 6 个 virtual consoles）来让用户登录。在物理机系统上你可以通过使用[Ctrl]+[Alt]+[F1]～[F6]进行切换，不过在我们的在线实验环境中可能无法切换，因为特殊功能按键会被你的主机系统劫持。当你切换到其中一个终端后想要切换回图形界面，你可以按下[Ctrl]+[Alt]+[F7]来完成。

### Shell

通常在图形界面中对实际体验带来差异的不是上述的不同发行版的各种终端模拟器，而是这个 Shell（壳）。有壳就有核，这里的核就是指 UNIX/Linux 内核，Shell 是指“提供给使用者使用界面”的软件（命令解析器），类似于 DOS 下的 command（命令行）和后来的 cmd.exe 。普通意义上的 Shell 就是可以接受用户输入命令的程序。它之所以被称作 Shell 是因为它隐藏了操作系统底层的细节。同样的 UNIX/Linux 下的图形用户界面 GNOME 和 KDE，有时也被叫做“虚拟 shell”或“图形 shell”。

UNIX/Linux 操作系统下的 Shell 既是用户交互的界面，也是控制系统的脚本语言。当然这一点也有别于 Windows 下的命令行，虽然该命令行也提供了很简单的控制语句。在 Windows 操作系统下，有些用户从来都不会直接使用 Shell，然而在 UNIX 系列操作系统下，Shell 仍然是控制系统启动、X11 启动和很多其它实用工具的脚本解释程序。

在 UNIX/Linux 中比较流行的常见的 Shell 有 bash、zsh、ksh、csh 等等，Ubuntu 终端默认使用的是 bash，默认的桌面环境是 GNOME 或者 Unity（基于 GNOME），但我们的环境中使用的分别是 zsh 和 xfce。

### 命令行

#### 写在前面

在linux中，最最重要的就是命令，这就包含了2个过程，输入和输出

* 输入：输入当然就是打开终端，然后按键盘输入，然后按回车，输入格式一般就是这类的

#创建一个名为 file 的文件，touch是一个命令

$ touch file

#进入一个目录，cd是一个命令

$ cd /etc/

#查看当前所在目录

$ pwd

* 输出：输出会返回你想要的结果，比如你要看什么文件，就会返回文件的内容。如果只是执行，执行失败会告诉你哪里错了，如果执行成功那么会没有输出，因为linux的哲学就是：没有结果就是最好的结果

#### 开始

如图，双击桌面上的 Xfce 终端 图标打开终端后系统会自动运行 Shell 程序，然后我们就可以输入命令让系统来执行了：

#### 1) 重要快捷键：

真正学习命令行之前，你先要掌握几个十分有用、必需掌握的小技巧：

##### [Tab]

使用Tab键来进行命令补全，Tab键一般是在字母Q旁边，这个技巧给你带来的最大的好处就是当你忘记某个命令的全称时可以只输入它的开头的一部分，然后按下Tab键就可以得到提示或者帮助完成：

##### [Ctrl+c]

想想你有没有遇到过这种情况，当你在 Linux 命令行中无意输入了一个不知道的命令，或者错误地使用了一个命令，导致在终端里出现了你无法预料的情况，比如，屏幕上只有光标在闪烁却无法继续输入命令，或者不停地输出一大堆你不想要的结果。你想要立即停止并恢复到你可控的状态，那该怎么办呢？这时候你就可以使用Ctrl+c键来强行终止当前程序（你可以放心它并不会使终端退出）。

尝试输入以下命令：

$ tail

然后你会发现你接下来的输入都没有任何反应了，只是将你输入的东西显示出来，现在你可以使用Ctrl+c，来中断这个你目前可能还不知道是什么的程序（在后续课程中我们会具体解释这个tail命令是什么）。

又或者输入：

$ find /

显然这不是你想的结果，可以使用Ctrl+c结束。

虽然这个按着很方便，但不要随便按，因为有时候，当你看到终端没有任何反应或提示，也不能接受你的输入时，可能只是运行的程序需要你耐心等一下，就不要急着按Ctrl+c了。

##### 其他一些常用快捷键

| **按键** | **作用** |
| --- | --- |
| Ctrl+d | 键盘输入结束或退出终端 |
| Ctrl+s | 暂停当前程序，暂停后按下任意键恢复运行 |
| Ctrl+z | 将当前程序放到后台运行，恢复到前台为命令fg |
| Ctrl+a | 将光标移至输入行头，相当于Home键 |
| Ctrl+e | 将光标移至输入行末，相当于End键 |
| Ctrl+k | 删除从光标所在位置到行末 |
| Alt+Backspace | 向前删除一个单词 |
| Shift+PgUp | 将终端显示向上滚动 |
| Shift+PgDn | 将终端显示向下滚动 |

#### 2) 学会利用历史输入命令

很简单，你可以使用键盘上的方向上键↑，恢复你之前输入过的命令，你一试便知。

#### 3) 学会使用通配符

通配符是一种特殊语句，主要有星号（\*）和问号（?），用来对字符串进行模糊匹配（比如文件名、参数名）。当查找文件夹时，可以使用它来代替一个或多个真正字符；当不知道真正字符或者懒得输入完整名字时，常常使用通配符代替一个或多个真正字符。

终端里面输入的通配符是由 Shell 处理的，不是由所涉及的命令语句处理的，它只会出现在命令的“参数值”里（它不能出现在命令名称里， 命令不记得，那就用Tab补全）。当 Shell 在“参数值”中遇到了通配符时，Shell 会将其当作路径或文件名在磁盘上搜寻可能的匹配：若符合要求的匹配存在，则进行代换（路径扩展）；否则就将该通配符作为一个普通字符传递给“命令”，然后再由命令进行处理。总之，通配符实际上就是一种 Shell 实现的路径扩展功能。在通配符被处理后， Shell 会先完成该命令的重组，然后继续处理重组后的命令，直至执行该命令。

首先回到用户家目录：

$ cd /home/shiyanlou

然后使用 touch 命令创建 2 个文件，后缀都为 txt：

$ touch asd.txt fgh.txt

可以给文件随意命名，假如过了很长时间，你已经忘了这两个文件的文件名，现在你想在一大堆文件中找到这两个文件，就可以使用通配符：

$ ls \*.txt

在创建文件的时候，如果需要一次性创建多个文件，比如：**“love\_1\_linux.txt，love\_2\_linux.txt，... love\_10\_linux.txt”**。在 Linux 中十分方便：

$ touch love\_{1..10}\_shiyanlou.txt

Shell 常用通配符：

| **字符** | **含义** |
| --- | --- |
| \* | 匹配 0 或多个字符 |
| ? | 匹配任意一个字符 |
| [list] | 匹配 list 中的任意单一字符 |
| [^list] | 匹配 除list 中的任意单一字符以外的字符 |
| [c1-c2] | 匹配 c1-c2 中的任意单一字符 如：[0-9] [a-z] |
| {string1,string2,...} | 匹配 string1 或 string2 (或更多)其一字符串 |
| {c1..c2} | 匹配 c1-c2 中全部字符 如{1..10} |

#### 4) 学会在命令行中获取帮助

在 Linux 环境中，如果你遇到困难，可以使用man命令，它是Manual pages的缩写。

Manual pages 是 UNIX 或类 UNIX 操作系统中在线软件文档的一种普遍的形式， 内容包括计算机程序（包括库和系统调用）、正式的标准和惯例，甚至是抽象的概念。用户可以通过执行man命令调用手册页。

你可以使用如下方式来获得某个命令的说明和使用方式的详细介绍：

$ man <command\_name>

比如你想查看 man 命令本身的使用方式，你可以输入：

man man

通常情况下，man 手册里面的内容都是英文的，这就要求你有一定的英文基础。man 手册的内容很多，涉及了 Linux 使用过程中的方方面面。为了便于查找，man 手册被进行了分册（分区段）处理，在 Research UNIX、BSD、OS X 和 Linux 中，手册通常被分为8个区段，安排如下：

| **区段** | **说明** |
| --- | --- |
| 1 | 一般命令 |
| 2 | 系统调用 |
| 3 | 库函数，涵盖了C标准函数库 |
| 4 | 特殊文件（通常是/dev中的设备）和驱动程序 |
| 5 | 文件格式和约定 |
| 6 | 游戏和屏保 |
| 7 | 杂项 |
| 8 | 系统管理命令和守护进程 |

要查看相应区段的内容，就在 man 后面加上相应区段的数字即可，如：

$ man 1 ls

会显示第一区段中的ls命令 man 页面。

所有的手册页遵循一个常见的布局，为了通过简单的 ASCII 文本展示而被优化，而这种情况下可能没有任何形式的高亮或字体控制。一般包括以下部分内容：

**NAME（名称）**

该命令或函数的名称，接着是一行简介。

**SYNOPSIS（概要）**

对于命令，正式的描述它如何运行，以及需要什么样的命令行参数。对于函数，介绍函数所需的参数，以及哪个头文件包含该函数的定义。

**DESCRIPTION（说明）**

命令或函数功能的文本描述。

**EXAMPLES（示例）**

常用的一些示例。

**SEE ALSO（参见）**

相关命令或函数的列表。

也可能存在其它部分内容，但这些部分没有得到跨手册页的标准化。常见的例子包括：OPTIONS（选项），EXIT STATUS（退出状态），ENVIRONMENT（环境），BUGS（程序漏洞），FILES（文件），AUTHOR（作者），REPORTING BUGS（已知漏洞），HISTORY（历史）和 COPYRIGHT（版权）。

通常 man 手册中的内容很多，你可能不太容易找到你想要的结果，不过幸运的是你可以在 man 中使用搜索/<你要搜索的关键字>，查找完毕后你可以使用n键切换到下一个关键字所在处，shift+n为上一个关键字所在处。使用Space（空格键）翻页，Enter（回车键）向下滚动一行，或者使用k,j（vim 编辑器的移动键）进行向前向后滚动一行。按下h键为显示使用帮助（因为 man 使用 less 作为阅读器，实为less工具的帮助），按下q退出。

想要获得更详细的帮助，你还可以使用info命令，不过通常使用man就足够了。如果你知道某个命令的作用，只是想快速查看一些它的某个具体参数的作用，那么你可以使用--help参数，大部分命令都会带有这个参数，如：

$ ls --help

## 作业：

本课程会在每一节实验结束部分介绍一个有趣的 Linux 命令，这一节介绍一个可以输出图形字符的命令banner，先看效果：

你可以先使用如下命令安装：

$ sudo apt-get update

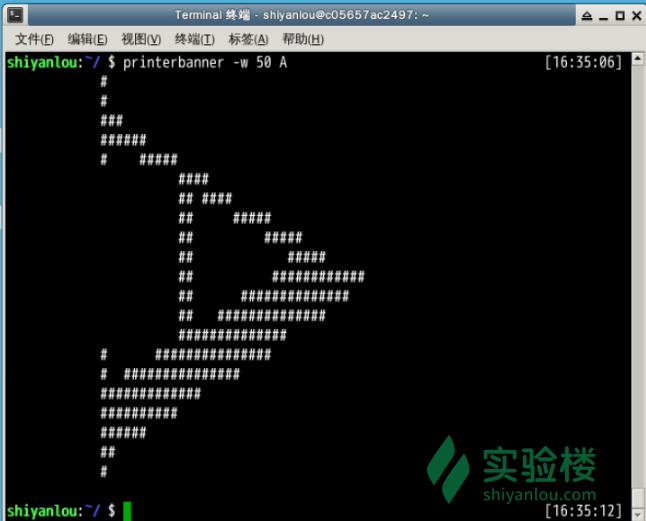
$ sudo apt-get install sysvbanner

然后：

$ banner shiyanlou

或者你觉得这字体不好看，那么你还可以使用默认已经安装的一个命令printerbanner：

$ printerbanner -w 50 A



-w参数指定打印宽度，因为我们的环境在屏幕中显示比较小，必须要加上宽度限制。

还有两个类似的命令toilet,figlet，作为作业安装试用。

# 用户及文件权限管理

#### **1.2 实验知识点**

* Linux ****用户管理****
* Linux ****权限管理****

## Linxu用户管理

通过第一节课程的学习，你应该已经知道，Linux 是一个可以实现多用户登录的操作系统，比如“李雷”和“韩梅梅”都可以同时登录同一台主机，他们共享一些主机的资源，但他们也分别有自己的用户空间，用于存放各自的文件。但实际上他们的文件都是放在同一个物理磁盘上的甚至同一个逻辑分区或者目录里，但是由于 Linux 的 ****用户管理**** 和 ****权限机制****，不同用户不可以轻易地查看、修改彼此的文件。

下面我们就来学习一下 Linux 下的账户管理的基础知识。

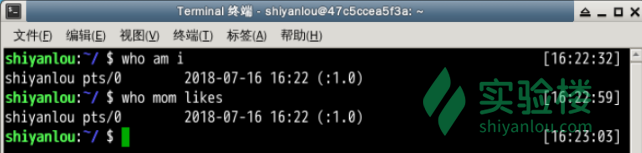
### 查看用户

请打开终端，输入命令：

$ who am i

或者

$ who mom likes

输出的第一列表示打开当前伪终端的用户的用户名（要查看当前登录用户的用户名，去掉空格直接使用 whoami 即可），第二列的 pts/0 中 pts 表示伪终端，所谓伪是相对于 /dev/tty 设备而言的，还记得上一节讲终端时的那七个使用 [Ctrl]+[Alt]+[F1]～[F7] 进行切换的 /dev/tty 设备么,这是“真终端”，伪终端就是当你在图形用户界面使用 /dev/tty7时每打开一个终端就会产生一个伪终端， pts/0 后面那个数字就表示打开的伪终端序号，你可以尝试再打开一个终端，然后在里面输入 who am i ，看第二列是不是就变成 pts/1了，第三列则表示当前伪终端的启动时间。

who 命令其它常用参数

| **参数** | **说明** |
| --- | --- |
| -a | 打印能打印的全部 |
| -d | 打印死掉的进程 |
| -m | 同am i,mom likes |
| -q | 打印当前登录用户数及用户名 |
| -u | 打印当前登录用户登录信息 |
| -r | 打印运行等级 |

### 创建用户

在 Linux 系统里， root 账户拥有整个系统至高无上的权利，比如 新建/添加 用户。

root 权限，系统权限的一种，与 SYSTEM 权限可以理解成一个概念，但高于 Administrator 权限，root 是 Linux 和 UNIX 系统中的超级管理员用户帐户，该帐户拥有整个系统至高无上的权力，所有对象他都可以操作，所以很多黑客在入侵系统的时候，都要把权限提升到 root 权限，用 Windows 的方法理解也就是将自己的非法帐户添加到 Administrators 用户组。更比如安卓操作系统中（基于 Linux 内核）获得 root 权限之后就意味着已经获得了手机的最高权限，这时候你可以对手机中的任何文件（包括系统文件）执行所有增、删、改、查的操作。

我们一般登录系统时都是以普通账户的身份登录的，要创建用户需要 root 权限，这里就要用到 sudo 这个命令了。不过使用这个命令有两个大前提，一是你要知道当前登录用户的密码，二是当前用户必须在 sudo 用户组。shiyanlou 用户也属于 sudo 用户组（稍后会介绍如何查看和添加用户组）。

#### **su，su- 与 sudo**

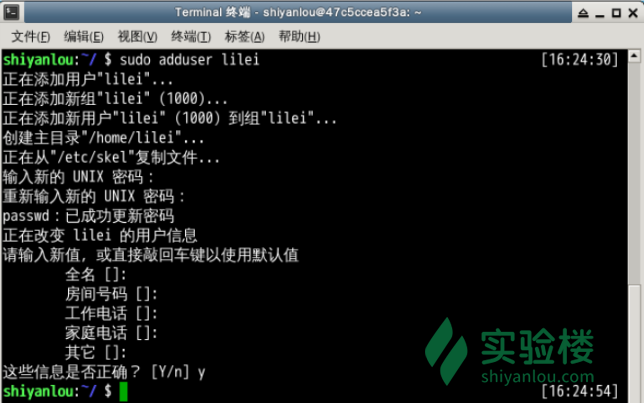
****需要注意 Linux 环境下输入密码是不会显示的。****

su <user> 可以切换到用户 user，执行时需要输入目标用户的密码，sudo <cmd> 可以以特权级别运行 cmd 命令，需要当前用户属于 sudo 组，且需要输入当前用户的密码。su - <user> 命令也是切换用户，同时环境变量也会跟着改变成目标用户的环境变量。

现在我们新建一个叫 lilei 的用户：

$ sudo adduser lilei

实验楼的环境目前设置为 shiyanlou 用户执行 sudo 不需要输入密码，通常此处需要按照提示输入 shiyanlou 密码（****Linux 下密码输入是不显示任何内容的****），**shiyanlou 用户密码可以通过 sudo passwd shiyanlou 命令进行设置**。然后是给 lilei 用户设置密码，后面的选项的一些内容你可以选择直接回车使用默认值：



这个命令不但可以添加用户到系统，同时也会默认为新用户创建 home 目录：

$ ls /home

现在你已经创建好一个用户，并且你可以使用你创建的用户登录了，使用如下命令切换登录用户：

$ su -l lilei

输入刚刚设置的 lilei 的密码，然后输入如下命令并查看输出：

$ who am i$ whoami$ pwd

退出当前用户跟退出终端一样可以使用 exit 命令或者使用快捷键 Ctrl+d。

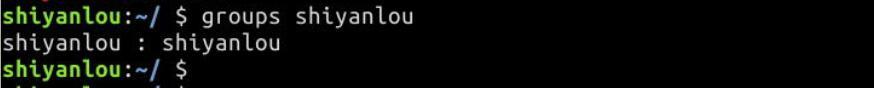
### 用户组

在 Linux 里面每个用户都有一个归属（用户组），用户组简单地理解就是一组用户的集合，它们共享一些资源和权限，同时拥有私有资源，就跟家的形式差不多，你的兄弟姐妹（不同的用户）属于同一个家（用户组），你们可以共同拥有这个家（共享资源），爸妈对待你们都一样（共享权限），你偶尔写写日记，其他人未经允许不能查看（私有资源和权限）。当然一个用户是可以属于多个用户组的，正如你既属于家庭，又属于学校或公司。

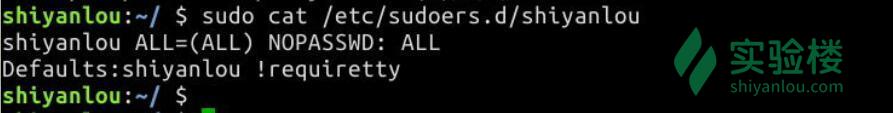
在 Linux 里面如何知道自己属于哪些用户组呢？

#### **方法一：使用 groups 命令**

$ groups shiyanlou



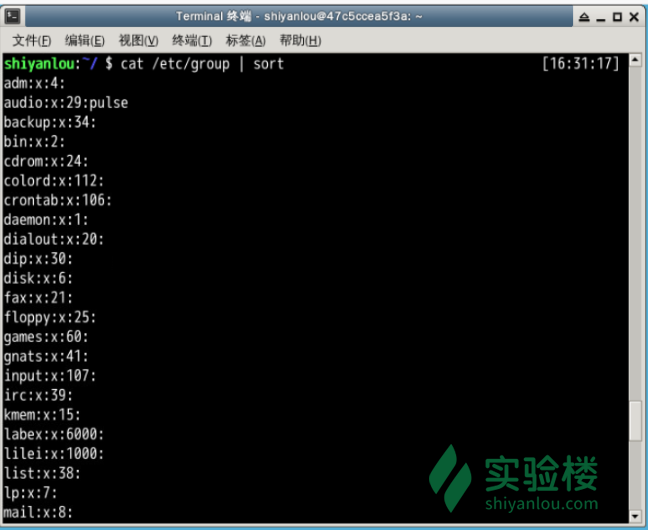
其中冒号之前表示用户，后面表示该用户所属的用户组。这里可以看到 shiyanlou 用户属于 shiyanlou 用户组，每次新建用户如果不指定用户组的话，默认会自动创建一个与用户名相同的用户组（差不多就相当于家长的意思，或者说是老总）。默认情况下在 sudo 用户组里的可以使用 sudo 命令获得 root 权限。shiyanlou 用户也可以使用 sudo 命令，为什么这里没有显示在 sudo 用户组里呢？可以查看下 /etc/sudoers.d/shiyanlou 文件，我们在 /etc/sudoers.d 目录下创建了这个文件，从而给 shiyanlou 用户赋予了 sudo 权限：



#### **方法二：查看 /etc/group 文件**

$ cat /etc/group | sort

这里 cat 命令用于读取指定文件的内容并打印到终端输出，后面会详细讲它的使用。 | sort 表示将读取的文本进行一个字典排序再输出，然后你将看到如下一堆输出，你可以在最下面看到 shiyanlou 的用户组信息：



没找到，没关系，你可以使用命令过滤掉一些你不想看到的结果：

$ cat /etc/group | grep -E "shiyanlou"

IMG_259

##### **/etc/group 文件格式说明**

/etc/group 的内容包括用户组（Group）、用户组口令、GID 及该用户组所包含的用户（User），每个用户组一条记录。格式如下：

group\_name:password:GID:user\_list

你看到上面的 password 字段为一个 x 并不是说密码就是它，只是表示密码不可见而已。

这里需要注意，如果用户的 GID 等于用户组的 GID，那么最后一个字段 user\_list 就是空的，比如 shiyanlou 用户，在 /etc/group 中的 shiyanlou 用户组后面是不会显示的。lilei 用户，在 /etc/group 中的 lilei 用户组后面是不会显示的。

##### **将其它用户加入 sudo 用户组**

默认情况下新创建的用户是不具有 root 权限的，也不在 sudo 用户组，可以让其加入 sudo 用户组从而获取 root 权限：

# 注意 Linux 上输入密码是不会显示的$ su -l lilei$ sudo ls

会提示 lilei 不在 sudoers 文件中，意思就是 lilei 不在 sudo 用户组中，至于 sudoers 文件（/etc/sudoers）你现在最好不要动它，操作不慎会导致比较麻烦的后果。

使用 usermod 命令可以为用户添加用户组，同样使用该命令你必需有 root 权限，你可以直接使用 root 用户为其它用户添加用户组，或者用其它已经在 sudo 用户组的用户使用 sudo 命令获取权限来执行该命令。

这里我用 shiyanlou 用户执行 sudo 命令将 lilei 添加到 sudo 用户组，让它也可以使用 sudo 命令获得 root 权限：

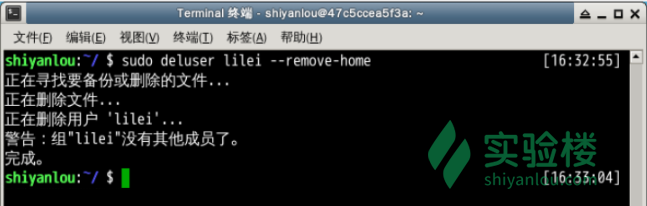
$ su shiyanlou # 此处需要输入 shiyanlou 用户密码，shiyanlou 的密码可以通过 `sudo passwd shiyanlou` 进行设置。$ groups lilei$ sudo usermod -G sudo lilei$ groups lilei

然后你再切换回 lilei 用户，现在就可以使用 sudo 获取 root 权限了。

### 删除用户

删除用户是很简单的事：

$ sudo deluser lilei --remove-home



## Linux文件权限

文件权限就是文件的访问控制权限，即哪些用户和组群可以访问文件以及可以执行什么样的操作。

Unix/Linux系统是一个典型的多用户系统，不同的用户处于不同的地位，对文件和目录有不同的访问权限。为了保护系统的安全性，Unix/Linux系统除了对用户权限作了严格的界定外，还在用户身份认证、访问控制、传输安全、文件读写权限等方面作了周密的控制。

在 Unix/Linux中的每一个文件或目录都包含有访问权限，这些访问权限决定了谁能访问和如何访问这些文件和目录。

### 查看文件权限

我们之前已经很多次用到 ls 命令了，如你所见，我们用它来列出并显示当前目录下的文件，当然这是在不带任何参数的情况下，它能做的当然不止这么多，现在我们就要用它来查看文件权限。

使用较长格式列出文件：

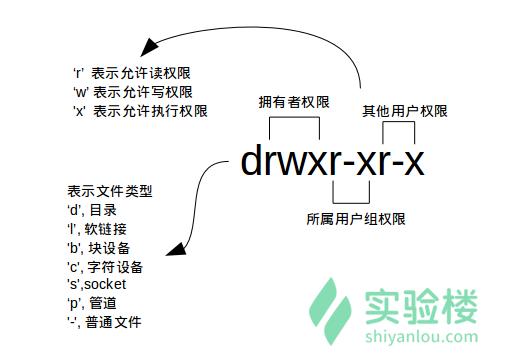
$ ls -l



你可能除了知道最后面那一项是文件名之外，其它项就不太清楚了，那么到底是什么意思呢：



可能你还是不太明白，比如第一项文件类型和权限那一堆东西具体指什么，链接又是什么，何为最后修改时间，下面一一道来：



* 文件类型

关于文件类型，这里有一点你必需时刻牢记 ****Linux 里面一切皆文件****，正因为这一点才有了设备文件（ /dev 目录下有各种设备文件，大都跟具体的硬件设备相关）这一说。 socket：网络套接字，具体是什么，感兴趣的用户可以自己去了解或期待实验楼的后续相关课程。pipe 管道，这个东西很重要，我们以后将会讨论到，这里你先知道有它的存在即可。软链接文件：链接文件是分为两种的，另一种当然是“硬链接”（硬链接不常用，具体内容不作为本课程讨论重点，而软链接等同于 Windows 上的快捷方式,你记住这一点就够了）。

* 文件权限

读权限，表示你可以使用 cat <file name> 之类的命令来读取某个文件的内容；写权限，表示你可以编辑和修改某个文件； 执行权限，通常指可以运行的二进制程序文件或者脚本文件，如同 Windows 上的 exe 后缀的文件，不过 Linux 上不是通过文件后缀名来区分文件的类型。你需要注意的一点是，****一个目录同时具有读权限和执行权限才可以打开并查看内部文件，而一个目录要有写权限才允许在其中创建其它文件****，这是因为目录文件实际保存着该目录里面的文件的列表等信息。

所有者权限，这一点相信你应该明白了，至于所属用户组权限，是指你所在的用户组中的所有其它用户对于该文件的权限，比如，你有一个 iPad，那么这个用户组权限就决定了你的兄弟姐妹有没有权限使用它破坏它和占有它。

* 链接数

链接到该文件所在的 inode 结点的文件名数目（关于这个概念涉及到 Linux 文件系统的相关概念知识，不在本课程的讨论范围，感兴趣的用户可以自己去了解）。

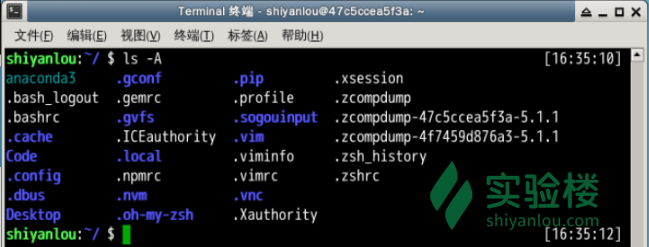
* 文件大小

以 inode 结点大小为单位来表示的文件大小，你可以给 ls 加上 -lh 参数来更直观的查看文件的大小。

明白了文件权限的一些概念，我们顺带补充一下关于 ls 命令的一些其它常用的用法：

* 显示除了 .（当前目录）和 ..（上一级目录）之外的所有文件，包括隐藏文件（Linux 下以 . 开头的文件为隐藏文件）。

$ ls -A



当然，你可以同时使用 -A 和 -l 参数：

$ ls -Al

查看某一个目录的完整属性，而不是显示目录里面的文件属性：

$ ls -dl <目录名>

* 显示所有文件大小，并以普通人类能看懂的方式呈现：

$ ls -AsSh

其中小 s 为显示文件大小，大 S 为按文件大小排序，若需要知道如何按其它方式排序，请使用“man”命令查询。

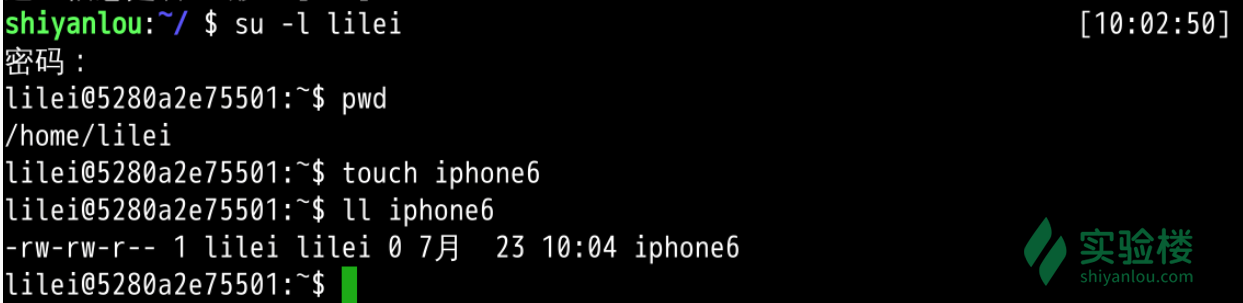
### 变更文件所有者

若前面已经执行删除lilei用户的命令，这里重新创建一下。

假设目前是 lilei 用户登录，新建一个文件，命名为 “ iphone6 ”：

# 注意当前的用户必须是 lilei# 如果是 shiyanlou 用户需要切换到 lilei（如果之前已经删除需要重新创建下）$ su lilei$ cd /home/lilei$ touch iphone6

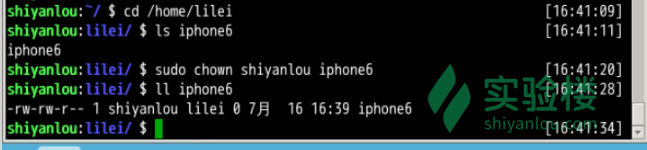
可见文件所有者是 lilei ：



现在，换回到 shiyanlou 用户身份，使用以下命令变更文件所有者为 shiyanlou ：

# 需要切换到 shiyanlou 用户执行以下操作$ cd /home/lilei$ ls iphone6$ sudo chown shiyanlou iphone6

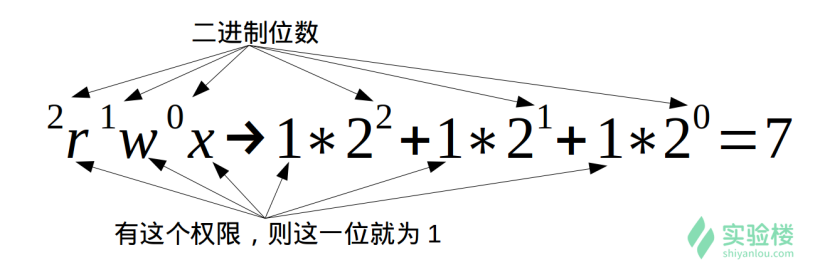
现在查看，发现 文件所有者成功修改为 shiyanlou ：



### 修改文件权限

如果你有一个自己的文件不想被其他用户读、写、执行，那么就需要对文件的权限做修改，这里有两种方式：

* 方式一：二进制数字表示



每个文件的三组权限（拥有者，所属用户组，其他用户，****记住这个顺序是一定的****）对应一个 " rwx "，也就是一个 “ 7 ” ，所以如果我要将文件“ iphone6 ”的权限改为只有我自己可以用那么就这样：

为了演示，我先在文件里加点内容：

$ echo "echo \"hello shiyanlou\"" > iphone6

然后修改权限：

$ chmod 600 iphone6

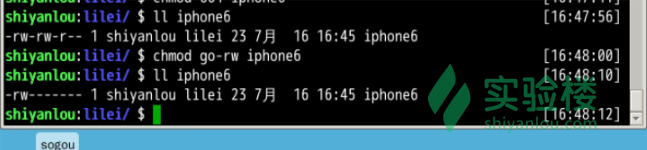
现在，其他用户已经不能读这个“ iphone6 ”文件了：



* 方式二：加减赋值操作

完成上述相同的效果，你可以：

$ chmod go-rw iphone6



g、o 还有 u 分别表示 group、others 和 user，+ 和 - 分别表示增加和去掉相应的权限。

### 其他

#### **adduser 和 useradd 的区别是什么？**

答：useradd 只创建用户，创建完了用 passwd lilei 去设置新用户的密码。adduser 会创建用户，创建目录，创建密码（提示你设置），做这一系列的操作。其实 useradd、userdel 这类操作更像是一种命令，执行完了就返回。而 adduser 更像是一种程序，需要你输入、确定等一系列操作。

Linux 还有一些关于隐藏权限和特殊权限的内容，想全面了解 Linux 权限管理这部分内容的用户可以通过其它方式学习。

### 作业

添加一个用户 loutest，使用 sudo 创建文件 /opt/forloutest，设置成用户 loutest 可以读写。截图并把操作过程写入实验报告。

****提示：**** 如何创建一个文件呢？可以考虑 touch 命令，执行 sudo touch /opt/forloutest 这个命令可以创建一个空文件，但前提是执行这个命令的 loutest 用户具备 sudo 权限。

代码

Sudo adduser loutest

Sudo usermod -G sudo loutest

Su -l loutest

Sudo touch /opt/forloutest

Sudo chmod go-rw /opt/forloutest

Sudo chmod u-w /opt/forloutest

# Linux目录结构及文件操作

#### **1.2 实验知识点**

* 每个目录的大体内容
* 文件的属性
* touch,file,rm,mv 等基本命令

## 目录结构

在讲 Linux 目录结构之前，你首先要清楚一点，那就是 Linux 的目录与 Windows 的目录的区别，或许对于一般操作上的感受来说没有多大不同，但从它们的实现机制来说是完全不同的。

一种不同是体现在目录与存储介质（磁盘，内存，DVD 等）的关系上，以往的 Windows 一直是以存储介质为主的，主要以盘符（C 盘，D 盘...）及分区来实现文件管理，然后之下才是目录，目录就显得不是那么重要，除系统文件之外的用户文件放在任何地方任何目录也是没有多大关系。所以通常 Windows 在使用一段时间后，磁盘上面的文件目录会显得杂乱无章（少数善于整理的用户除外吧）。然而 UNIX/Linux 恰好相反，UNIX 是以目录为主的，Linux 也继承了这一优良特性。 Linux 是以树形目录结构的形式来构建整个系统的，可以理解为树形目录是一个用户可操作系统的骨架。虽然本质上无论是目录结构还是操作系统内核都是存储在磁盘上的，但从逻辑上来说 Linux 的磁盘是“挂在”（挂载在）目录上的，每一个目录不仅能使用本地磁盘分区的文件系统，也可以使用网络上的文件系统。举例来说，可以利用网络文件系统（Network File System，NFS）服务器载入某特定目录等。

## FHS标准

Linux 的目录结构说复杂很复杂，说简单也很简单。复杂在于，因为系统的正常运行是以目录结构为基础的，对于初学者来说里面大部分目录都不知道其作用，重要与否，特别对于那些曾经的重度 Windows 用户，他们会纠结很长时间，关于我安装的软件在哪里这类问题。说它简单是因为，其中大部分目录结构是规定好了的（FHS 标准），是死的，当你掌握后，你在里面的一切操作都会变得井然有序。

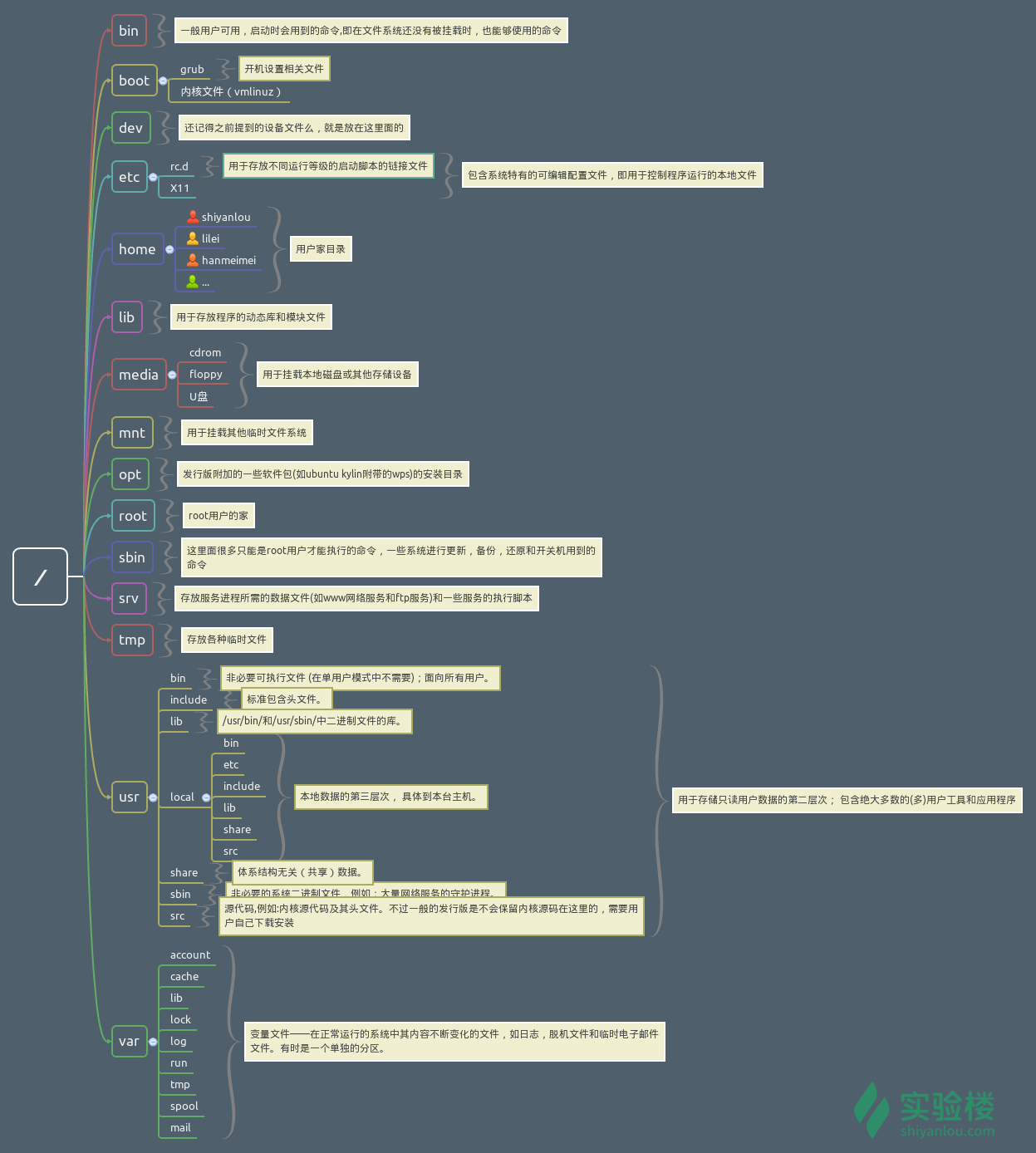
FHS（英文：Filesystem Hierarchy Standard 中文：文件系统层次结构标准），多数 Linux 版本采用这种文件组织形式，FHS 定义了系统中每个区域的用途、所需要的最小构成的文件和目录同时还给出了例外处理与矛盾处理。

FHS 定义了两层规范，第一层是， / 下面的各个目录应该要放什么文件数据，例如 /etc 应该放置设置文件，/bin 与 /sbin 则应该放置可执行文件等等。

第二层则是针对 /usr 及 /var 这两个目录的子目录来定义。例如 /var/log 放置系统日志文件，/usr/share 放置共享数据等等。

[FHS\_2.3 标准文档](http://refspecs.linuxfoundation.org/FHS_2.3/fhs-2.3.pdf" \t "https://www.shiyanlou.com/courses/1/learning/_blank)

****如果觉得图片不清晰，建议另存为到本地放大查看：****



如果你觉得看这个不明白，那么可以试试最真实最直观的方式，执行如下命令：

$ tree /

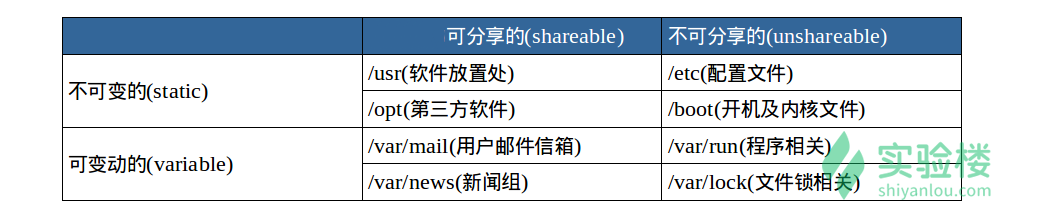
如果提示" command not found "，就先安装：

# 因为我们的环境的原因，每次新启动实验会清除系统恢复到初始状态，所以需要手动更新软件包索引，以便我们安装时能找到相应软件包的源。

sudo apt-get update

sudo apt-get install tree

关于上面提到的 FHS，这里还有个很重要的内容你一定要明白，FHS 是根据以往无数 Linux 用户和开发者的经验总结出来的，并且会维持更新，FHS 依据文件系统使用的频繁与否以及是否允许用户随意改动（注意，不是不能，学习过程中，不要怕这些），将目录定义为四种交互作用的形态，如下表所示：



## 目录路径

#### **路径**

有人可能不明白这路径是指什么，有什么用。顾名思义，路径就是你要去哪儿的路线嘛。如果你想进入某个具体的目录或者想获得某个目录的文件（目录本身也是文件）那就得用路径来找到了。

使用 cd 命令可以切换目录，在 Linux 里面使用 . 表示当前目录，.. 表示上一级目录（****注意，我们上一节介绍过的，以**.**开头的文件都是隐藏文件，所以这两个目录必然也是隐藏的，你可以使用**ls -a**命令查看隐藏文件****）, - 表示上一次所在目录，～ 通常表示当前用户的 home 目录。使用 pwd 命令可以获取当前所在路径（绝对路径）。

进入上一级目录：

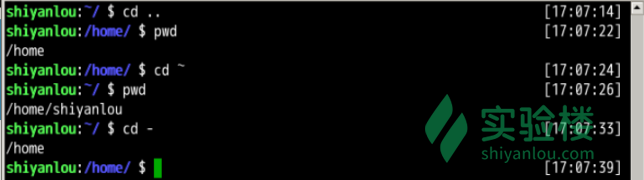
$ cd ..

进入你的 home 目录：

$ cd ~ # 或者 cd /home/<你的用户名>

使用 pwd 获取当前路径：

$ pwd



#### **绝对路径**

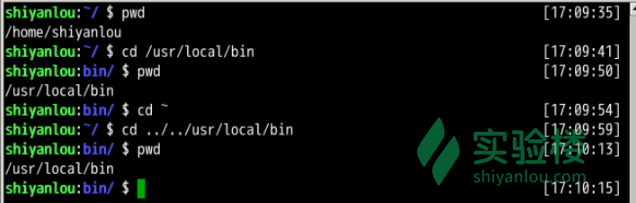
关于绝对路径，简单地说就是以根" / "目录为起点的完整路径，以你所要到的目录为终点，表现形式如：/usr/local/bin，表示根目录下的 usr 目录中的 local 目录中的 bin 目录。

#### **相对路径**

相对路径，也就是相对于你当前的目录的路径，相对路径是以当前目录 . 为起点，以你所要到的目录为终点，表现形式如： usr/local/bin （这里假设你当前目录为根目录）。你可能注意到，我们表示相对路径实际并没有加上表示当前目录的那个 . ，而是直接以目录名开头，因为这个 usr 目录为 /目录下的子目录，是可以省略这个 . 的（以后会讲到一个类似不能省略的情况）；如果是当前目录的上一级目录，则需要使用 .. ，比如你当前目录为 /home/shiyanlou 目录下，根目录就应该表示为 ../../ ，表示上一级目录（ home 目录）的上一级目录（ / 目录）。

下面我们以你的 home目录为起点，分别以绝对路径和相对路径的方式进入 /usr/local/bin 目录：

# 绝对路径$ cd /usr/local/bin# 相对路径$ cd ../../usr/local/bin



进入一个目录，可以使用绝对路径也可以使用相对路径，那我们应该在什么时候选择正确的方式进入某个目录呢。就是凭直觉嘛，你觉得怎样方便就使用哪一个，而不用特意只使用某一种。比如假设我当前在 /usr/local/bin 目录，我想进入上一级的 local 目录你说是使用 cd .. 方便还是 cd /usr/local 方便？而如果要进入的是 usr 目录，那么 cd /usr，就比 cd ../.. 方便一点了。

****提示：在进行目录切换的过程中请多使用**Tab**键自动补全，可避免输入错误，连续按两次**Tab**可以显示全部候选结果。****

## Linux文件基本操作

这一节我们主要讲解文件常用的基本操作，包括：新建、复制、删除、移动文件与文件重命名、查看文件、查看文件类型、以及编辑文件。

### 新建文件

#### **新建空白文件**

使用 touch 命令创建空白文件，关于 touch 命令，其主要作用是来更改已有文件的时间戳的（比如，最近访问时间，最近修改时间），但其在不加任何参数的情况下，只指定一个文件名，则可以创建一个指定文件名的空白文件（不会覆盖已有同名文件），当然你也可以同时指定该文件的时间戳，更多关于 touch 命令的用法，会在下一讲文件搜索中涉及。

创建名为 test 的空白文件，因为在其它目录没有权限，所以需要先 cd ~ 切换回用户的 /home/shiyanlou 目录：

$ cd /home/shiyanlou$ touch test

#### **新建目录**

使用 mkdir（make directories）命令可以创建一个空目录，也可同时指定创建目录的权限属性。

创建名为“ mydir ”的空目录：

$ mkdir mydir

使用 -p 参数，同时创建父目录（如果不存在该父目录），如下我们同时创建一个多级目录（这在安装软件、配置安装路径时非常有用）：

$ mkdir -p father/son/grandson



后面的目录路径，以绝对路径的方式表示也是可以的。

### 复制文件

#### **复制文件**

使用 cp（copy）命令复制一个文件到指定目录。

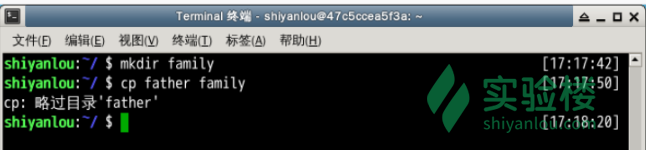
将之前创建的“ test ”文件复制到“ /home/shiyanlou/father/son/grandson ”目录中：

$ cp test father/son/grandson

是不是很方便啊，如果在图形界面则需要先在源目录复制文件，再进到目的目录粘贴文件，而命令行操作步骤就一步到位了嘛。

#### **复制目录**

如果直接使用 cp 命令复制一个目录的话，会出现如下错误：



要成功复制目录需要加上 -r 或者 -R 参数，表示递归复制，就是说有点“株连九族”的意思：

$ cd /home/shiyanlou$ mkdir family$ cp -r father family

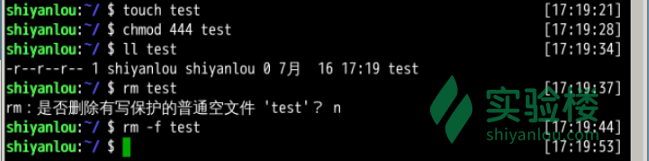
### 删除文件

#### **删除文件**

使用 rm（remove files or directories）命令删除一个文件：

$ rm test

有时候你会遇到想要删除一些为只读权限的文件，直接使用 rm 删除会显示一个提示，如下：



你如果想忽略这提示，直接删除文件，可以使用 -f 参数强制删除：

$ rm -f test

#### **删除目录**

跟复制目录一样，要删除一个目录，也需要加上 -r 或 -R 参数：

$ rm -r family

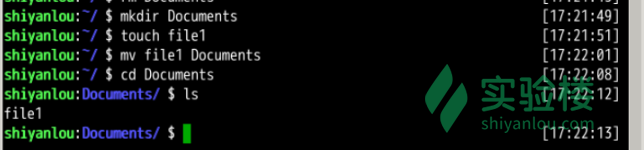
### 重命名文件/移动文件

#### **移动文件**

使用 mv（move or rename files）命令移动文件（剪切）。将文件“ file1 ”移动到 Documents 目录：

mv 源目录文件 目的目录：

$ mkdir Documents$ touch file1$ mv file1 Documents



#### **重命名文件**

将文件“ file1 ”重命名为“ myfile ”：

mv 旧的文件名 新的文件名：

$ mv file1 myfile

#### **批量重命名**

要实现批量重命名，mv 命令就有点力不从心了，我们可以使用一个看起来更专业的命令 rename 来实现。不过它要用 perl 正则表达式来作为参数，关于正则表达式我们要在后面才会介绍到，这里只做演示，你只要记得这个 rename 命令可以批量重命名就好了，以后再重新学习也不会有任何问题，毕竟你已经掌握了一个更常用的 mv 命令。

$ cd /home/shiyanlou/

# 使用通配符批量创建 5 个文件:

$ touch file{1..5}.txt

# 批量将这 5 个后缀为 .txt 的文本文件重命名为以 .c 为后缀的文件:

$ rename 's/\.txt/\.c/' \*.txt

# 批量将这 5 个文件，文件名和后缀改为大写:

$ rename 'y/a-z/A-Z/' \*.c

简单解释一下上面的命令，rename 是先使用第二个参数的通配符匹配所有后缀为 .txt 的文件，然后使用第一个参数提供的正则表达式将匹配的这些文件的 .txt 后缀替换为 .c，这一点在我们后面学习了 sed 命令后，相信你会更好地理解。

### 查看文件

#### **使用 cat，tac 和 nl 命令查看文件**

前两个命令都是用来打印文件内容到标准输出（终端），其中 cat 为正序显示，tac 为倒序显示。

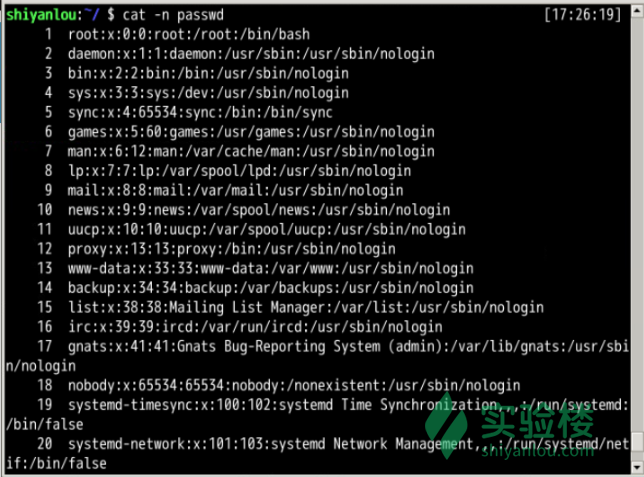
标准输入输出：当我们执行一个 shell 命令行时通常会自动打开三个标准文件，即标准输入文件（stdin），默认对应终端的键盘、标准输出文件（stdout）和标准错误输出文件（stderr），后两个文件都对应被重定向到终端的屏幕，以便我们能直接看到输出内容。进程将从标准输入文件中得到输入数据，将正常输出数据输出到标准输出文件，而将错误信息送到标准错误文件中。

比如我们要查看之前从 /etc 目录下拷贝来的 passwd 文件：

$ cd /home/shiyanlou$ cp /etc/passwd passwd$ cat passwd

可以加上 -n 参数显示行号：

$ cat -n passwd



nl 命令，添加行号并打印，这是个比 cat -n 更专业的行号打印命令。

这里简单列举它的常用的几个参数：

-b : 指定添加行号的方式，主要有两种：

-b a:表示无论是否为空行，同样列出行号("cat -n"就是这种方式)

-b t:只列出非空行的编号并列出（默认为这种方式）

-n : 设置行号的样式，主要有三种：

-n ln:在行号字段最左端显示

-n rn:在行号字段最右边显示，且不加 0

-n rz:在行号字段最右边显示，且加 0

-w : 行号字段占用的位数(默认为 6 位)



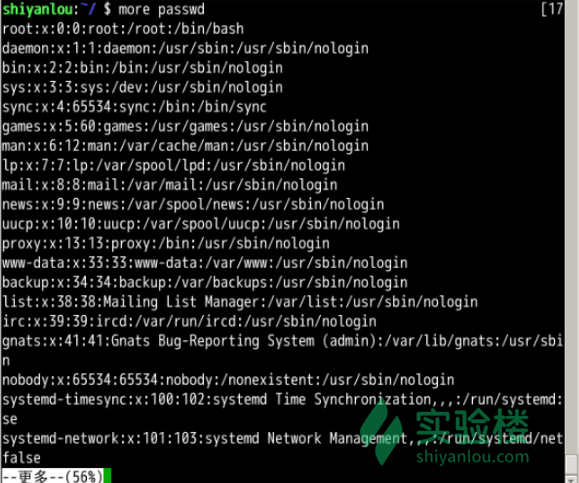
你会发现使用这几个命令，默认的终端窗口大小，一屏显示不完文本的内容，得用鼠标拖动滚动条或者滑动滚轮才能继续往下翻页，要是可以直接使用键盘操作翻页就好了，那么你就可以使用下面要介绍的命令。

#### **使用 more 和 less 命令分页查看文件**

如果说上面的 cat 是用来快速查看一个文件的内容的，那么这个 more 和 less 就是天生用来"阅读"一个文件的内容的，比如说 man 手册内部就是使用的 less 来显示内容。其中 more命令比较简单，只能向一个方向滚动，而 less 为基于 more和 vi （一个强大的编辑器，我们有单独的课程来让你学习）开发，功能更强大。less 的使用基本和 more 一致，具体使用请查看 man 手册，这里只介绍 more 命令的使用。

使用 more 命令打开 passwd 文件：

$ more passwd



打开后默认只显示一屏内容，终端底部显示当前阅读的进度。可以使用 Enter 键向下滚动一行，使用 Space 键向下滚动一屏，按下 h 显示帮助，q 退出。

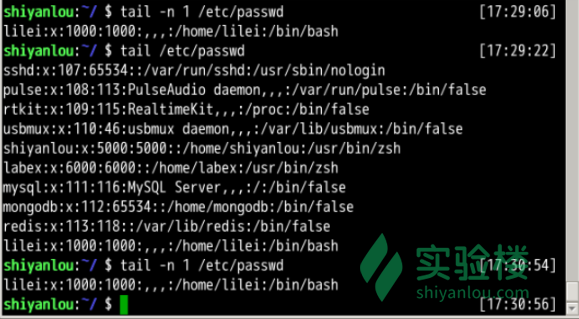
#### **使用 head 和 tail 命令查看文件**

这两个命令，那些性子比较急的人应该会喜欢，因为它们一个是只查看文件的头几行（默认为 10 行，不足 10 行则显示全部）和尾几行。还是拿 passwd 文件举例，比如当我们想要查看最近新增加的用户，那么我们可以查看这个 /etc/passwd文件，不过我们前面也看到了，这个文件里面一大堆乱糟糟的东西，看起来实在费神啊。因为系统新增加一个用户，会将用户的信息添加到 passwd 文件的最后，那么这时候我们就可以使用 tail 命令了：

$ tail /etc/passwd

甚至更直接的只看一行， 加上 -n 参数，后面紧跟行数：

$ tail -n 1 /etc/passwd



关于 tail 命令，不得不提的还有它一个很牛的参数 -f，这个参数可以实现不停地读取某个文件的内容并显示。这可以让我们动态查看日志，达到实时监视的目的。不过我不会在这门基础课程中介绍它的更多细节，感兴趣的用户可以自己去了解。

### 查看文件类型

前面我提到过，在 Linux 中文件的类型不是根据文件后缀来判断的，我们通常使用 file 命令查看文件的类型：

$ file /bin/ls



说明这是一个可执行文件，运行在 64 位平台，并使用了动态链接文件（共享库）。

### 编辑文件

在 Linux 下面编辑文件通常我们会直接使用专门的命令行编辑器比如（emacs，vim，nano），由于涉及 Linux 上的编辑器的内容比较多，且非常重要，故我们有一门单独的基础课专门介绍这中一个编辑器 vim 。****强烈希望正在学习这门 Linux 基础课的你先在这里暂停一下，去学习**[vim 编辑器](http://www.shiyanlou.com/courses/2" \t "https://www.shiyanlou.com/courses/1/learning/_blank)**的使用（至少掌握基本的操作），然后再继续本课程后面的内容，因为后面的内容会假设你已经学会了 vim 编辑器的使用。****如果你想更加快速地入门，可以直接使用 Linux 内部的 vim 学习教程，输入如下命令即可开始：

$ vimtutor

### 其他

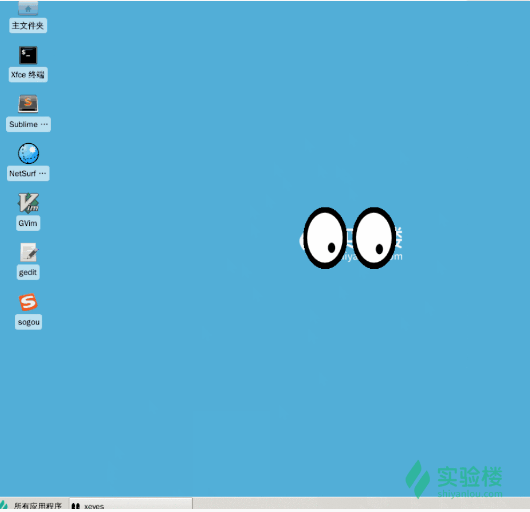
#### **轻松一下**

你是不是觉得在我们的环境中学习轻松愉快毫无压力，所以偶尔偷偷懒也是没有问题的呢？要真是这样可不太好啊，要学会给自己一点压力，严格要求自己才行。你或许会想，要是有人能监督就好了，这样能学得更快。好吧，今天就教你怎么召唤一双眼睛出来监督你：

$ xeyes

你可以使用如下命令将它放到后台运行：

$ nohup xeyes &



# 环境变量与文件查找

#### **1.2 实验知识点**

* 环境变量的设置
* 环境变量的修改

## 环境变量

#### **变量**

要解释环境变量，得先明白变量是什么，准确的说应该是 Shell 变量，所谓变量就是计算机中用于记录一个值（不一定是数值，也可以是字符或字符串）的符号，而这些符号将用于不同的运算处理中。通常变量与值是一对一的关系，可以通过表达式读取它的值并赋值给其它变量，也可以直接指定数值赋值给任意变量。为了便于运算和处理，大部分的编程语言会区分变量的类型，用于分别记录数值、字符或者字符串等等数据类型。Shell 中的变量也基本如此，有不同类型（但不用专门指定类型名），可以参与运算，有作用域限定。

变量的作用域即变量的有效范围（比如一个函数中、一个源文件中或者全局范围），在该范围内只能有一个同名变量。一旦离开则该变量无效，如同不存在这个变量一般。

在 Shell 中如何创建一个变量，如何给变量赋值和如何读取变量的值呢？这部分内容会在 [bash 脚本编程](http://www.shiyanlou.com/courses/5" \t "https://www.shiyanlou.com/courses/1/learning/_blank) 这门课中详细介绍，这里我简单举例说明一下：

使用 declare 命令创建一个变量名为 tmp 的变量：

$ declare tmp

其实也可以不用 declare 预声明一个变量，直接即用即创建，这里只是告诉你 declare 的作用，这在创建其它指定类型的变量（如数组）时会用到。

使用 = 号赋值运算符，将变量 tmp 赋值为 shiyanlou：

$ tmp=shiyanlou

读取变量的值，使用 echo 命令和 $ 符号（****$ 符号用于表示引用一个变量的值，初学者经常忘记输入****）：

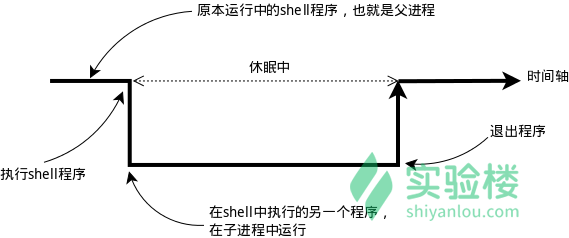
$ echo $tmp



****注意：并不是任何形式的变量名都是可用的，变量名只能是英文字母、数字或者下划线，且不能以数字作为开头。****

### 环境变量

简单理解了变量的概念，就很容易理解环境变量了。环境变量的作用域比自定义变量的要大，如 Shell 的环境变量作用于自身和它的子进程。在所有的 UNIX 和类 UNIX 系统中，每个进程都有其各自的环境变量设置，且默认情况下，当一个进程被创建时，除了创建过程中明确指定的话，它将继承其父进程的绝大部分环境设置。Shell 程序也作为一个进程运行在操作系统之上，而我们在 Shell 中运行的大部分命令都将以 Shell 的子进程的方式运行。

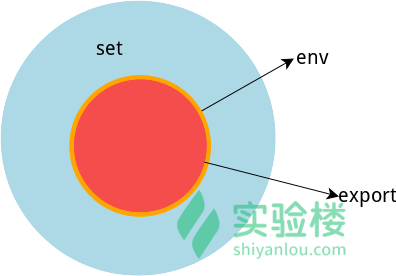


通常我们会涉及到的变量类型有三种：

* 当前 Shell 进程私有用户自定义变量，如上面我们创建的 tmp 变量，只在当前 Shell 中有效。
* Shell 本身内建的变量。
* 从自定义变量导出的环境变量。

也有三个与上述三种环境变量相关的命令：set，env，export。这三个命令很相似，都是用于打印环境变量信息，区别在于涉及的变量范围不同。详见下表：

| **命 令** | **说 明** |
| --- | --- |
| set | 显示当前 Shell 所有变量，包括其内建环境变量（与 Shell 外观等相关），用户自定义变量及导出的环境变量。 |
| env | 显示与当前用户相关的环境变量，还可以让命令在指定环境中运行。 |
| export | 显示从 Shell 中导出成环境变量的变量，也能通过它将自定义变量导出为环境变量。 |



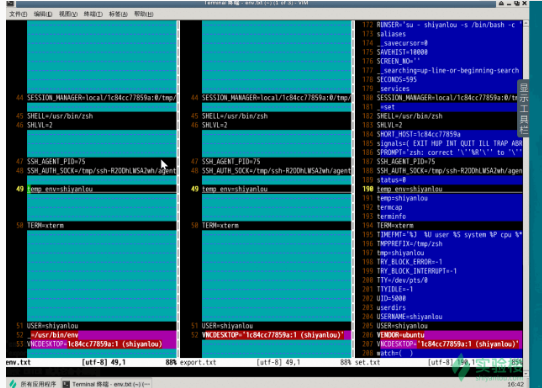
你可以更直观的使用 vimdiff 工具比较一下它们之间的差别：

$ temp=shiyanlou$ export temp\_env=shiyanlou$ env|sort>env.txt$ export|sort>export.txt$ set|sort>set.txt

上述操作将命令输出通过管道 | 使用 sort 命令排序，再重定向到对象文本文件中。

$ vimdiff env.txt export.txt set.txt

使用 vimdiff 工具比较导出的几个文件的内容：



关于哪些变量是环境变量，可以简单地理解成在当前进程的子进程有效则为环境变量，否则不是（有些人也将所有变量统称为环境变量，只是以全局环境变量和局部环境变量进行区分，我们只要理解它们的实质区别即可）。我们这里用 export 命令来体会一下，先在 Shell 中设置一个变量 temp=shiyanlou，然后再新创建一个子 Shell 查看 temp 变量的值：



****注意：为了与普通变量区分，通常我们习惯将环境变量名设为大写。****

#### **永久生效**

但是问题来了，当你关机后，或者关闭当前的 shell 之后，环境变量就没了啊。怎么才能让环境变量永久生效呢？

按变量的生存周期来划分，Linux 变量可分为两类：

永久的：需要修改配置文件，变量永久生效

临时的：用 export 命令行声明即可，变量在关闭 shell 时失效。

这里介绍两个重要文件 /etc/bashrc（有的 Linux 没有这个文件） 和 /etc/profile ，它们分别存放的是 shell 变量和环境变量。还有要注意区别的是每个用户目录下的一个隐藏文件：

.profile 可以用 ls -a 查看

cd /home/shiyanlou

ls -a

这个 .profile 只对当前用户永久生效。而写在 /etc/profile里面的是对所有用户永久生效，所以如果想要添加一个永久生效的环境变量，只需要打开 /etc/profile，在最后加上你想添加的环境变量就好啦。

### 命令的查找路径与顺序

你可能很早之前就有疑问，我们在 Shell 中输入一个命令，Shell 是怎么知道去哪找到这个命令然后执行的呢？这是通过环境变量 PATH 来进行搜索的，熟悉 Windows 的用户可能知道 Windows 中的也是有这么一个 PATH 环境变量。这个 PATH里面就保存了 Shell 中执行的命令的搜索路径。

查看 PATH 环境变量的内容：

$ echo $PATH

默认情况下你会看到如下输出：

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games

如果你还记得 Linux 目录结构那一节的内容，你就应该知道上面这些目录下放的是哪一类文件了。通常这一类目录下放的都是可执行文件，当我们在 Shell 中执行一个命令时，系统就会按照 PATH 中设定的路径按照顺序依次到目录中去查找，如果存在同名的命令，则执行先找到的那个。

下面我们将练习创建一个最简单的可执行 Shell 脚本和一个使用 C 语言创建的“ hello world ”程序，如果这两部分内容你之前没有学习过，那么你可以进行一个入门学习： [C 语言入门教程](http://www.shiyanlou.com/courses/57" \t "https://www.shiyanlou.com/courses/1/learning/_blank) [高级 Bash 脚本编程指南](http://www.shiyanlou.com/courses/5" \t "https://www.shiyanlou.com/courses/1/learning/_blank) [Linux Shell Scripting Tutorial (LSST) v2.0](http://bash.cyberciti.biz/guide/Main_Page" \t "https://www.shiyanlou.com/courses/1/learning/_blank) 。

创建一个 Shell 脚本文件：

$ cd /home/shiyanlou$ touch hello\_shell.sh$ gedit hello\_shell.sh

在脚本中添加如下内容，保存并退出（****注意不要省掉第一行，这不是注释，论坛有用户反映有语法错误，就是因为没有了第一行****）：

#!/bin/bashfor ((i=0; i<10; i++));do

echo "hello shell"done

exit 0

为文件添加可执行权限：

$ chmod 755 hello\_shell.sh

执行脚本：

$ cd /home/shiyanlou

$ ./hello\_shell.sh

创建一个 C 语言“ hello world ”程序：

$ cd /home/shiyanlou$ gedit hello\_world.c

#include <stdio.h>

int main(void){

printf("hello world!\n");

return 0;

}

保存后使用 gcc 生成可执行文件：

$ gcc -o hello\_world hello\_world.c

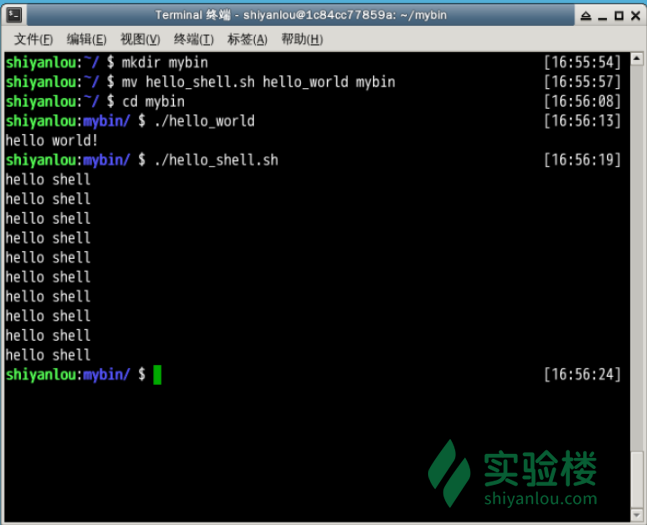
****gcc 生成二进制文件默认具有可执行权限，不需要修改****

在 /home/shiyanlou 家目录创建一个 mybin 目录，并将上述 hello\_shell.sh 和 hello\_world 文件移动到其中：

$ cd /home/shiyanlou$ mkdir mybin$ mv hello\_shell.sh hello\_world mybin/

现在你可以在 mybin 目录中分别运行你刚刚创建的两个程序：

$ cd mybin$ ./hello\_shell.sh$ ./hello\_world



回到上一级目录，也就是 shiyanlou 家目录，当再想运行那两个程序时，会发现提示命令找不到，除非加上命令的完整路径，但那样很不方便，如何做到像使用系统命令一样执行自己创建的脚本文件或者程序呢？那就要将命令所在路径添加到 PATH 环境变量了。

### 添加自定义路径到“ PATH ”环境变量

在前面我们应该注意到 PATH 里面的路径是以 : 作为分割符的，所以我们可以这样添加自定义路径：

$ PATH=$PATH:/home/shiyanlou/mybin

****注意这里一定要使用绝对路径。****

现在你就可以在任意目录执行那两个命令了（注意需要去掉前面的 ./）。你可能会意识到这样还并没有很好的解决问题，因为我给 PATH 环境变量追加了一个路径，它也只是在当前 Shell 有效，我一旦退出终端，再打开就会发现又失效了。有没有方法让添加的环境变量全局有效？或者每次启动 Shell 时自动执行上面添加自定义路径到 PATH 的命令？下面我们就来说说后一种方式——让它自动执行。

在每个用户的 home 目录中有一个 Shell 每次启动时会默认执行一个配置脚本，以初始化环境，包括添加一些用户自定义环境变量等等。zsh 的配置文件是 .zshrc，相应 Bash 的配置文件为 .bashrc 。它们在 etc 下还都有一个或多个全局的配置文件，不过我们一般只修改用户目录下的配置文件。

我们可以简单地使用下面命令直接添加内容到 .zshrc 中：

$ echo "PATH=$PATH:/home/shiyanlou/mybin" >> .zshrc

****上述命令中**>>**表示将标准输出以追加的方式重定向到一个文件中，注意前面用到的**>**是以覆盖的方式重定向到一个文件中，使用的时候一定要注意分辨。在指定文件不存在的情况下都会创建新的文件。****

### 修改和删除已有变量

#### **变量修改**

变量的修改有以下几种方式：

| **变量设置方式** | **说明** |
| --- | --- |
| ${变量名#匹配字串} | 从头向后开始匹配，删除符合匹配字串的最短数据 |
| ${变量名##匹配字串} | 从头向后开始匹配，删除符合匹配字串的最长数据 |
| ${变量名%匹配字串} | 从尾向前开始匹配，删除符合匹配字串的最短数据 |
| ${变量名%%匹配字串} | 从尾向前开始匹配，删除符合匹配字串的最长数据 |
| ${变量名/旧的字串/新的字串} | 将符合旧字串的第一个字串替换为新的字串 |
| ${变量名//旧的字串/新的字串} | 将符合旧字串的全部字串替换为新的字串 |

比如要修改我们前面添加到 PATH 的环境变量。为了避免操作失误导致命令找不到，我们先将 PATH 赋值给一个新的自定义变量 path：

$ path=$PATH

$ echo $path

$ path=${path%/home/shiyanlou/mybin}# 或使用通配符,\*表示任意多个任意字符

$ path=${path%\*/mybin}

#### **变量删除**

可以使用 unset 命令删除一个环境变量：

$ unset temp

### 环境变量立即生效

前面我们在 Shell 中修改了一个配置脚本文件之后（比如 zsh 的配置文件 home 目录下的 .zshrc），每次都要退出终端重新打开甚至重启主机之后其才能生效，很是麻烦，我们可以使用 source 命令来让其立即生效，如：

$ cd /home/shiyanlou$ source .zshrc

source 命令还有一个别名就是 .，上面的命令如果替换成 . 的方式就该是：

$ . ./.zshrc

在使用.的时候，需要注意与表示当前路径的那个点区分开。

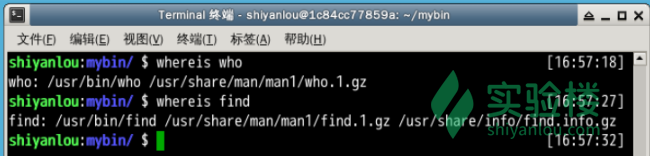
注意第一个点后面有一个空格，而且后面的文件必须指定完整的绝对或相对路径名，source 则不需要。

## 搜索文件

与搜索相关的命令常用的有 whereis，which，find 和 locate。

* **whereis**简单快速****

$ whereis who$ whereis find



你会看到 whereis find 找到了三个路径，两个可执行文件路径和一个 man 在线帮助文件所在路径，这个搜索很快，因为它并没有从硬盘中依次查找，而是直接从数据库中查询。whereis 只能搜索二进制文件(-b)，man 帮助文件(-m)和源代码文件(-s)。如果想要获得更全面的搜索结果可以使用 locate命令。

* **locate**快而全****

通过“ /var/lib/mlocate/mlocate.db ”数据库查找，不过这个数据库也不是实时更新的，系统会使用定时任务每天自动执行 updatedb 命令更新一次，所以有时候你刚添加的文件，它可能会找不到，需要手动执行一次 updatedb 命令（在我们的环境中必须先执行一次该命令）。它可以用来查找指定目录下的不同文件类型，如查找 /etc 下所有以 sh 开头的文件：

$ sudo apt-get update

$ sudo apt-get install locate

$ locate /etc/sh

****注意，它不只是在 /bin 目录下查找，还会自动递归子目录进行查找。****

查找 /usr/share/ 下所有 jpg 文件：

$ locate /usr/share/\\*.jpg

****注意要添加**\***号前面的反斜杠转义，否则会无法找到。****

如果想只统计数目可以加上 -c 参数，-i 参数可以忽略大小写进行查找，whereis 的 -b、-m、-s 同样可以使用。

* **which**小而精****

which 本身是 Shell 内建的一个命令，我们通常使用 which 来确定是否安装了某个指定的软件，因为它只从 PATH 环境变量指定的路径中去搜索命令：

$ which man

* **find**精而细****

find 应该是这几个命令中最强大的了，它不但可以通过文件类型、文件名进行查找而且可以根据文件的属性（如文件的时间戳，文件的权限等）进行搜索。find 命令强大到，要把它讲明白至少需要单独好几节课程才行，我们这里只介绍一些常用的内容。

这条命令表示去 /etc/ 目录下面 ，搜索名字叫做 interfaces 的文件或者目录。这是 find 命令最常见的格式，千万记住 find 的第一个参数是要搜索的地方：

$ sudo find /etc/ -name interfaces

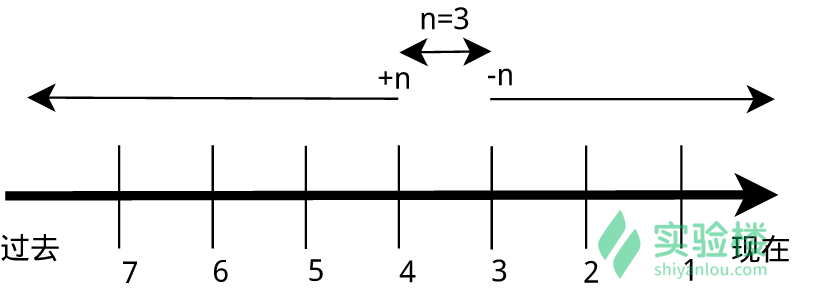
****注意 find 命令的路径是作为第一个参数的， 基本命令格式为 find [path] [option] [action] 。****

与时间相关的命令参数：

| **参数** | **说明** |
| --- | --- |
| -atime | 最后访问时间 |
| -ctime | 最后修改文件内容的时间 |
| -mtime | 最后修改文件属性的时间 |

下面以 -mtime 参数举例：

* -mtime n：n 为数字，表示为在 n 天之前的“一天之内”修改过的文件
* -mtime +n：列出在 n 天之前（不包含 n 天本身）被修改过的文件
* -mtime -n：列出在 n 天之内（包含 n 天本身）被修改过的文件
* -newer file：file 为一个已存在的文件，列出比 file 还要新的文件名



列出 home 目录中，当天（24 小时之内）有改动的文件：

$ find ~ -mtime 0

列出用户家目录下比 Code 文件夹新的文件：

$ find ~ -newer /home/shiyanlou/Code

## 其他

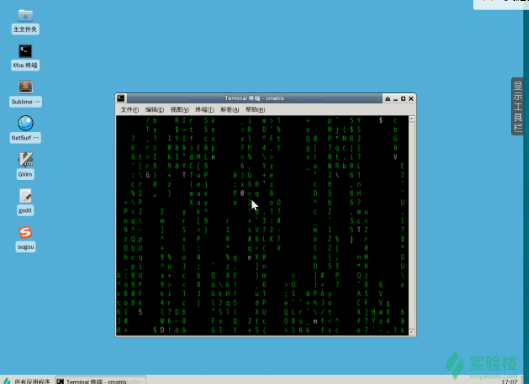
#### **轻松一下**

《黑客帝国》电影里满屏幕代码的“数字雨”，在 Linux 里面你也可以轻松实现这样的效果，你只需要一个命令 cmatrix 。

需要先安装，因为 Ubuntu 没有预装：

$ sudo apt-get update; sudo apt-get install cmatrix

装好之后先不要急着执行，为了看到更好的效果，我们需要先修改终端的主题配色，在终端上面的【选项】中修改，修改为黑底绿字：



然后按下 F11 可以全屏显示。

还可以改变代码的颜色：

cmatrix -C red

## 作业

#### **寻找文件**

找出 /etc/ 目录下的所有以 .list 结尾的文件。

Sudo find /etc/ -name \*.list