

Linux 零基础知识入门到提高



深圳

October 4, 2018

① 为什么要学 Linux?

② linux 介绍

③ Linux 基本知识

④ Linux 命令行

① 为什么要学 Linux?

② linux 介绍

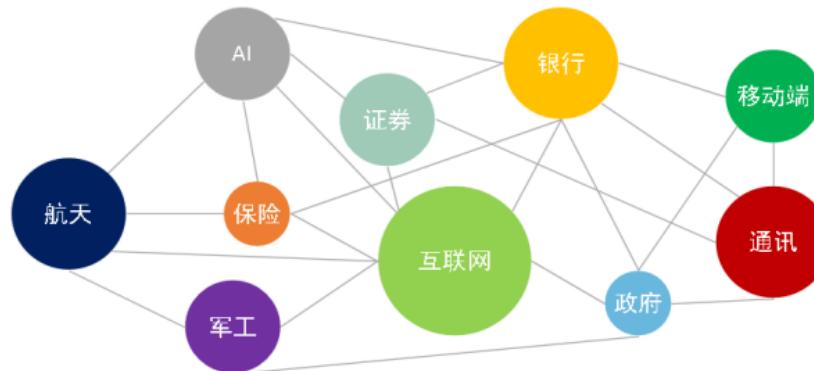
③ Linux 基本知识

④ Linux 命令行

为什么要学 Linux?



- Linux 无处不在
 - Linux 的 Android 活跃用户 >40亿
- Linux 越来越重要



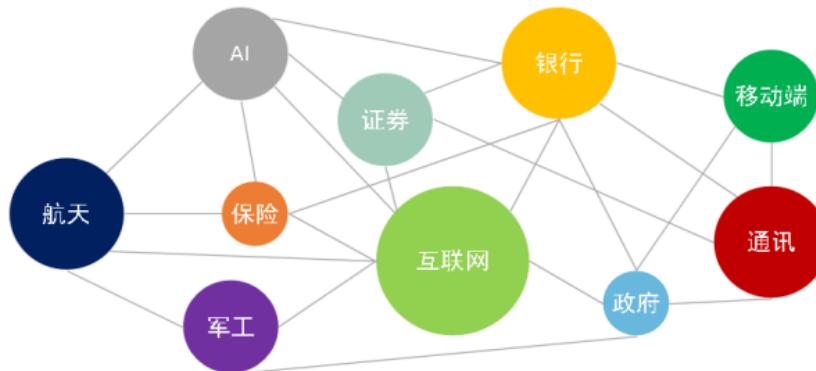
为什么要学 Linux?



- Linux 无处不在

- Linux 的 Android 活跃用户 >40亿
- Supercomputer Linux/Unix 份额 100%

- Linux 越来越重要



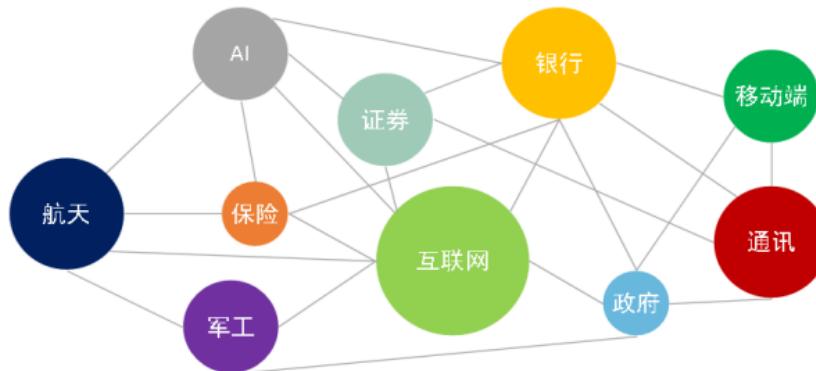
为什么要学 Linux?



- Linux 无处不在

- Linux 的 Android 活跃用户 >40亿
- Supercomputer Linux/Unix 份额 100%
- 嵌入式系统 (Embedded System)

- Linux 越来越重要



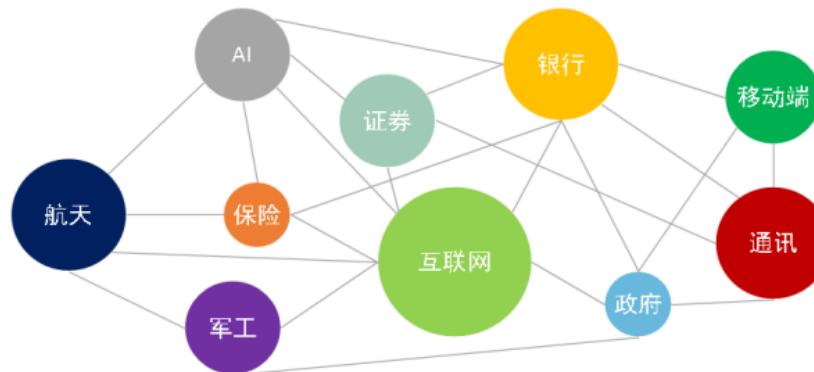
为什么要学 Linux?



- Linux 无处不在

- Linux 的 Android 活跃用户 >40亿
- Supercomputer Linux/Unix 份额 100%
- 嵌入式系统 (Embedded System)
- 服务器领域, 存储/数据库服务器

- Linux 越来越重要



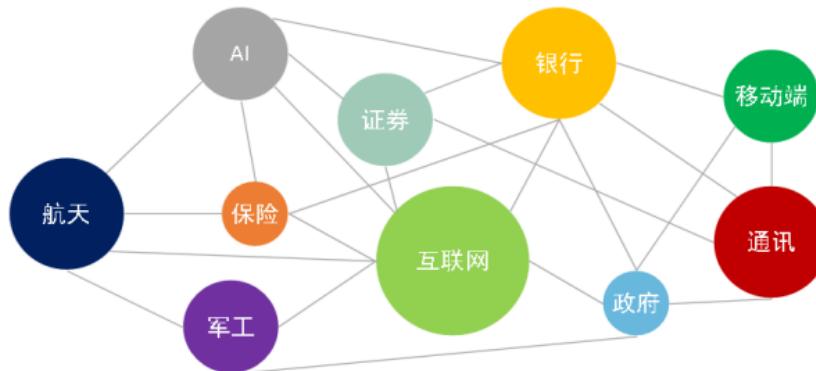
为什么要学 Linux?



- Linux 无处不在

- Linux 的 Android 活跃用户 >40亿
- Supercomputer Linux/Unix 份额 100%
- 嵌入式系统 (Embedded System)
- 服务器领域, 存储/数据库服务器
- 云计算、大数据、物联网、区块链

- Linux 越来越重要



为什么要学 Linux?

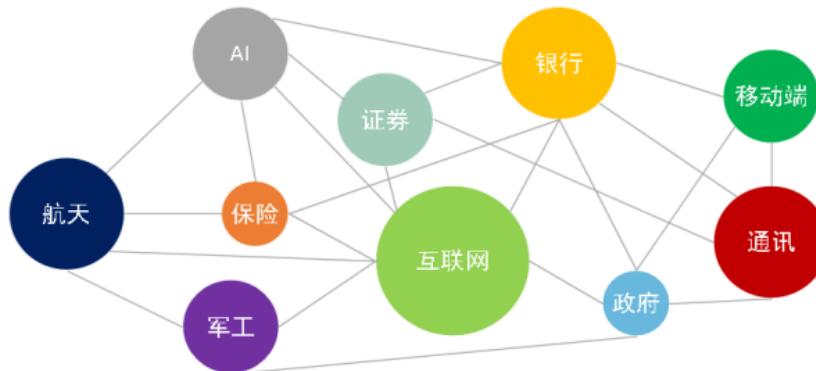


- Linux 无处不在

- Linux 的 Android 活跃用户 >40亿
- Supercomputer Linux/Unix 份额 100%
- 嵌入式系统 (Embedded System)
- 服务器领域, 存储/数据库服务器
- 云计算、大数据、物联网、区块链

- Linux 越来越重要

- 移动时代



为什么要学 Linux?

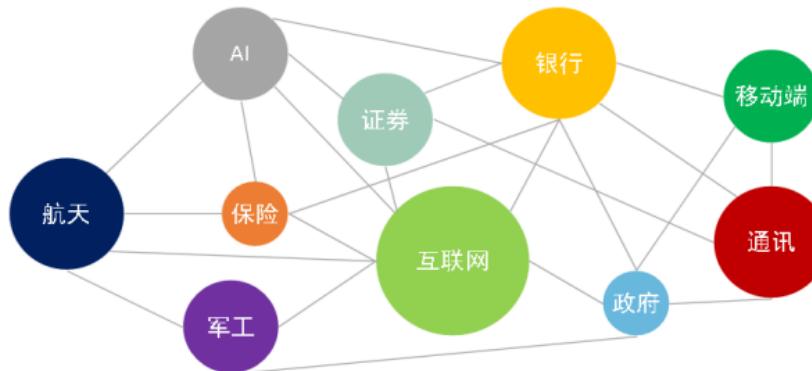


- Linux 无处不在

- Linux 的 Android 活跃用户 >40亿
- Supercomputer Linux/Unix 份额 100%
- 嵌入式系统 (Embedded System)
- 服务器领域, 存储/数据库服务器
- 云计算、大数据、物联网、区块链

- Linux 越来越重要

- 移动时代
- 智能万物互联



为什么要学 Linux?

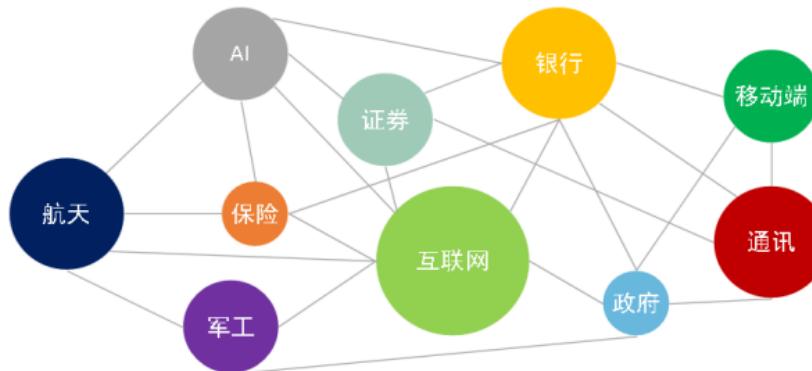


- Linux 无处不在

- Linux 的 Android 活跃用户 >40亿
- Supercomputer Linux/Unix 份额 100%
- 嵌入式系统 (Embedded System)
- 服务器领域, 存储/数据库服务器
- 云计算、大数据、物联网、区块链

- Linux 越来越重要

- 移动时代
- 智能万物互联
- 云、容器



为什么要学 Linux?

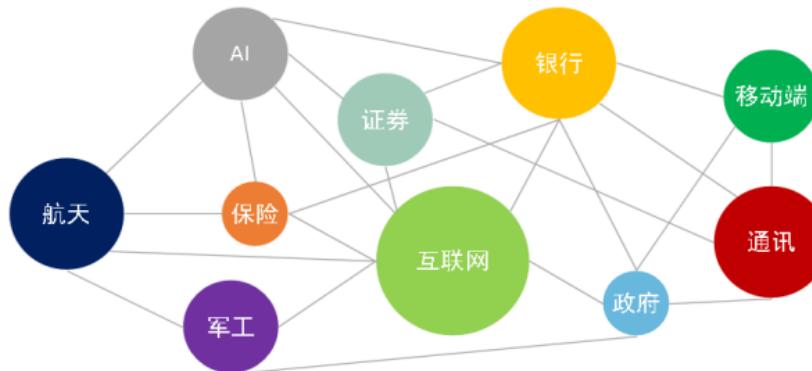


- Linux 无处不在

- Linux 的 Android 活跃用户 >40亿
- Supercomputer Linux/Unix 份额 100%
- 嵌入式系统 (Embedded System)
- 服务器领域, 存储/数据库服务器
- 云计算、大数据、物联网、区块链

- Linux 越来越重要

- 移动时代
- 智能万物互联
- 云、容器
- PC、移动、平板融合



为什么要学 Linux?

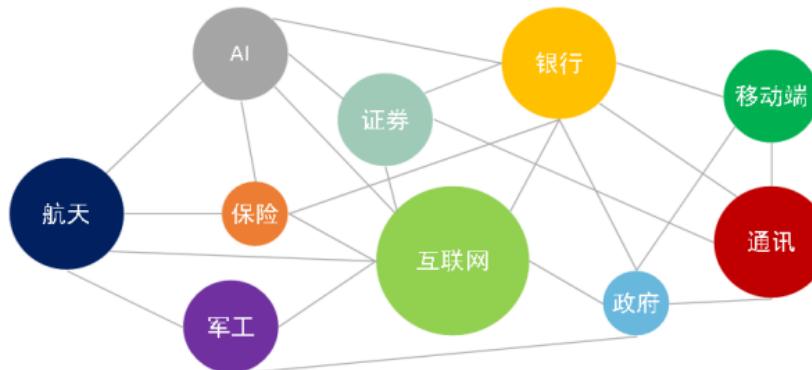


- Linux 无处不在

- Linux 的 Android 活跃用户 >40亿
- Supercomputer Linux/Unix 份额 100%
- 嵌入式系统 (Embedded System)
- 服务器领域, 存储/数据库服务器
- 云计算、大数据、物联网、区块链

- Linux 越来越重要

- 移动时代
- 智能万物互联
- 云、容器
- PC、移动、平板融合
- 新兴开发技术



为什么要学 Linux?



- Linux 自由、高效
 - 自由了解到一切，决定做什么



为什么要学 Linux?



- Linux 自由、高效

- 自由 了解到一切，决定做什么
- 高效 高效的环境，高效极致的工具



为什么要学 Linux?



- Linux 自由、高效

- **自由** 了解到一切，决定做什么
- **高效** 高效的环境，高效极致的工具
- 让你不断学习新东西，动手解决的能力



为什么要学 Linux?



- Linux 自由、高效

- **自由** 了解到一切，决定做什么
- **高效** 高效的环境，高效极致的工具
- 让你不断学习新东西，动手解决的能力
- 高度定制、环境搭建高效



为什么要学 Linux?



- Linux 自由、高效

- **自由** 了解到一切，决定做什么
- **高效** 高效的环境，高效极致的工具
- 让你不断学习新东西，动手解决的能力
- 高度定制、环境搭建高效
- 天生的编程环境



① 为什么要学 Linux?

② linux 介绍

③ Linux 基本知识

④ Linux 命令行

面向对象及培训目标?

面向对象及培训目标? .

linux 就像是一个开放的 大宝库 , 只要你想, 你随时就可以出发去 探险 ,
比 win 更加开放的格局意味着你可以接触到更多, 前提是有 兴趣 。

- 面向对 Linux 感兴趣的人员



面向对象及培训目标?

面向对象及培训目标? .

linux 就像是一个开放的 大宝库 , 只要你想, 你随时就可以出发去 探险 , 比 win 更加开放的格局意味着你可以接触到更多, 前提是有 兴趣 。

- 面向对 Linux 感兴趣的人员
- 面向需要了解 Linux 基本知识的人员



面向对象及培训目标?

面向对象及培训目标? .

linux 就像是一个开放的 大宝库 , 只要你想, 你随时就可以出发去 探险 , 比 win 更加开放的格局意味着你可以接触到更多, 前提是有 兴趣 。

- 面向对 Linux 感兴趣的人员
- 面向需要了解 Linux 基本知识的人员
- 了解到之前不了解的知识, 入门及提高方法



面向对象及培训目标?

面向对象及培训目标? .

linux 就像是一个开放的 大宝库 , 只要你想, 你随时就可以出发去 探险 , 比 win 更加开放的格局意味着你可以接触到更多, 前提是有 兴趣 。

- 面向对 Linux 感兴趣的人员
- 面向需要了解 Linux 基本知识的人员
- 了解到之前不了解的知识, 入门及提高方法
- 学习 Linux 最好的方法, 就是天天用它。



面向对象及培训目标?

面向对象及培训目标? .

linux 就像是一个开放的 大宝库 , 只要你想, 你随时就可以出发去 探险 , 比 win 更加开放的格局意味着你可以接触到更多, 前提是有 兴趣 。

- 面向对 Linux 感兴趣的人员
- 面向需要了解 Linux 基本知识的人员
- 了解到之前不了解的知识, 入门及提高方法
- 学习 Linux 最好的方法, 就是天天用它。
- 有什么 习惯 是不能改变的



面向对象及培训目标?

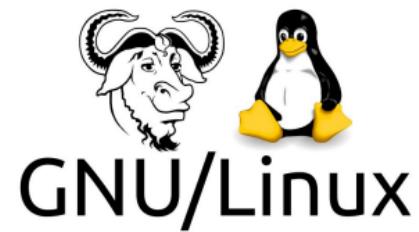
面向对象及培训目标? .

linux 就像是一个开放的 大宝库 , 只要你想, 你随时就可以出发去 探险 , 比 win 更加开放的格局意味着你可以接触到更多, 前提是有 兴趣 。

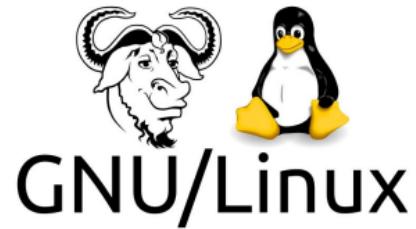
- 面向对 Linux 感兴趣的人员
- 面向需要了解 Linux 基本知识的人员
- 了解到之前不了解的知识, 入门及提高方法
- 学习 Linux 最好的方法, 就是天天用它。
- 有什么 习惯 是不能改变的
- IT 人士不了解 Linux/Unix , 对技术世界的认识是不完整的



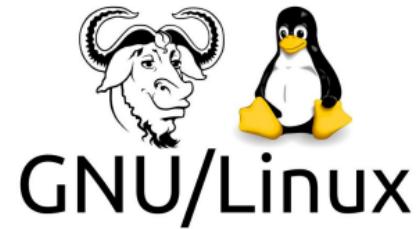
- Unix出现了，最初开源，大家很高兴



- Unix 出现了，最初开源，大家很高兴
- 大家都看到 Unix 的潜力了，于是 ATT 邪恶了。

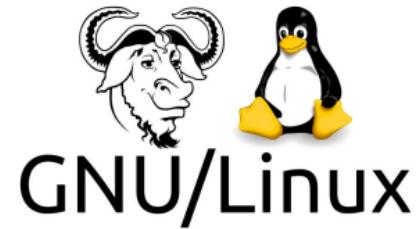


- Unix 出现了，最初开源，大家很高兴
- 大家都看到 Unix 的潜力了，于是 ATT 邪恶了。
- Unix 闭源了，大学没得用了，于是有了 Minux。

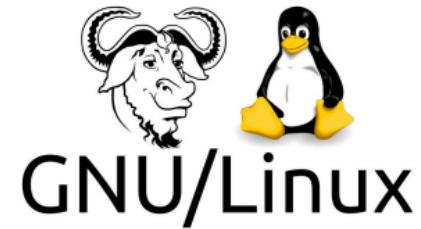


Linux 历史

- Unix 出现了，最初开源，大家很高兴
- 大家都看到 Unix 的潜力了，于是 ATT 邪恶了。
- Unix 闭源了，大学没得用了，于是有了 Minux。
- Minux 虽然开源，但是有点鸡肋。Linux 出现了

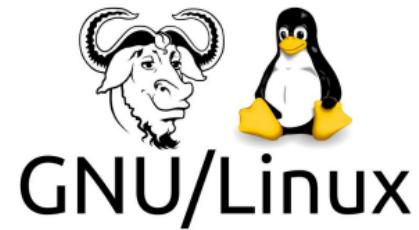


- Unix 出现了，最初开源，大家很高兴
- 大家都看到 Unix 的潜力了，于是 ATT 邪恶了。
- Unix 闭源了，大学没得用了，于是有了 Minux。
- Minux 虽然开源，但是有点鸡肋。Linux 出现了
- Linux 只是一个内核，基本上干不了啥。此时的 GNU 挺火的，但它的内核迟迟没弄出来。

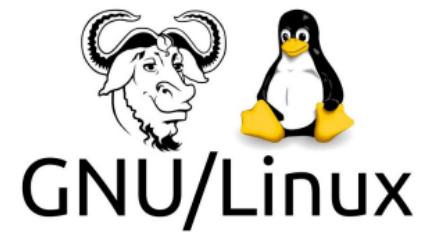


Linux 历史

- Unix 出现了，最初开源，大家很高兴
- 大家都看到 Unix 的潜力了，于是 ATT 邪恶了。
- Unix 闭源了，大学没得用了，于是有了 Minux。
- Minux 虽然开源，但是有点鸡肋。Linux 出现了
- Linux 只是一个内核，基本上干不了啥。此时的 GNU 挺火的，但它的内核迟迟没弄出来。
- 于是 Linux 和 GNU 联姻了。

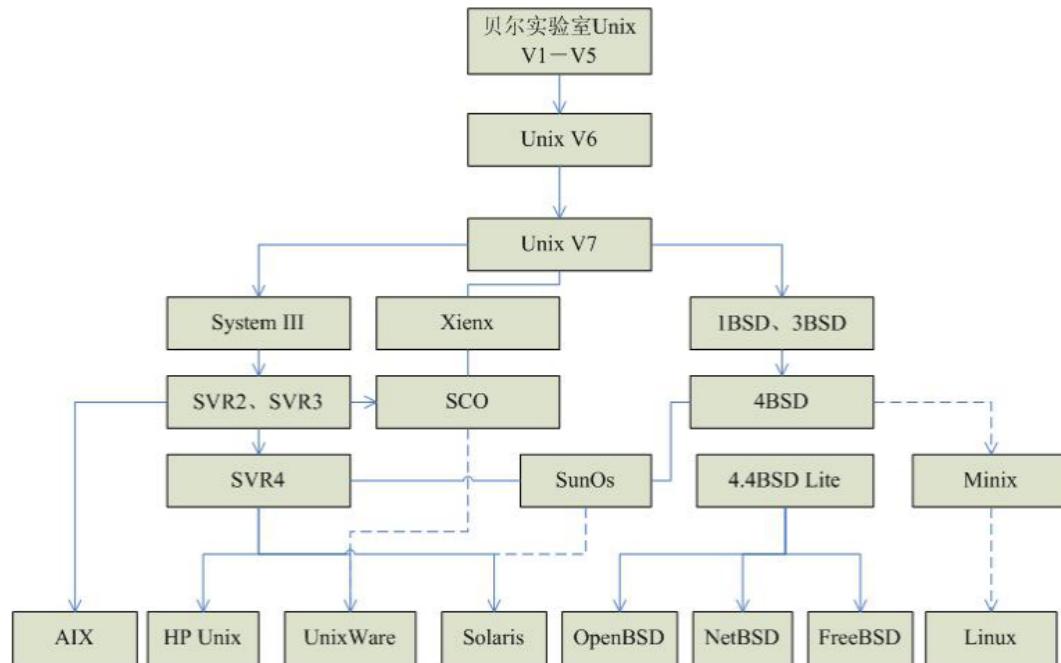


- Unix 出现了，最初开源，大家很高兴
- 大家都看到 Unix 的潜力了，于是 ATT 邪恶了。
- Unix 闭源了，大学没得用了，于是有了 Minux。
- Minux 虽然开源，但是有点鸡肋。Linux 出现了
- Linux 只是一个内核，基本上干不了啥。此时的 GNU 挺火的，但它的内核迟迟没弄出来。
- 于是 Linux 和 GNU 联姻了。
- Linux 出内核 GNU 出软件，于是就有了 GNU/Linux。



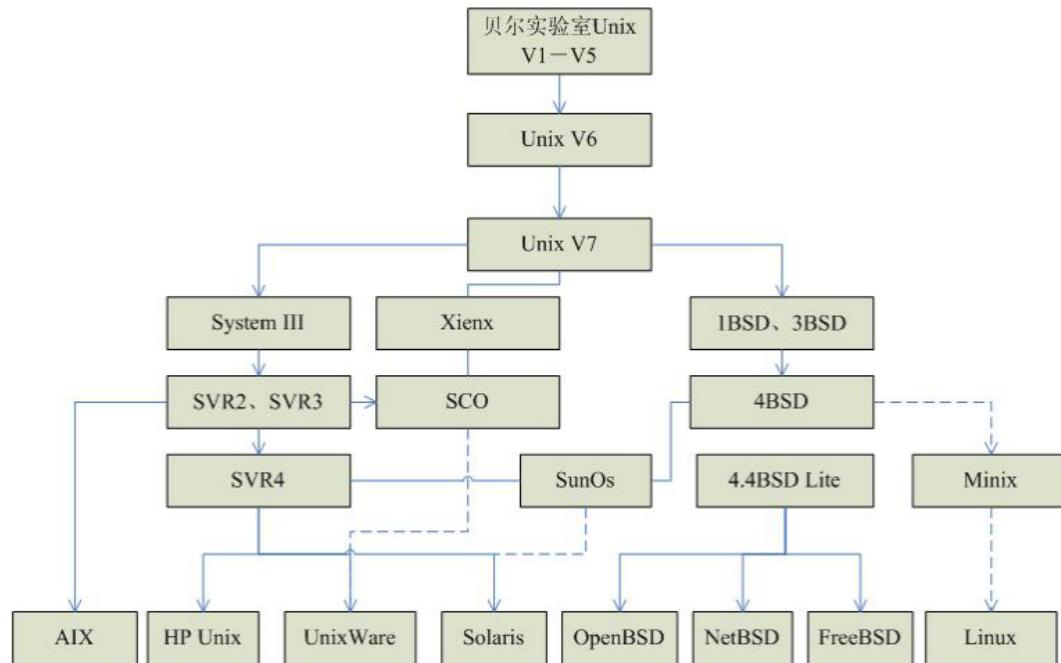
Linux 历史

● Unix 演化



Linux 历史

- Unix演化
- 诞生 1969



Linux 之父

Linus Torvalds 22 岁创造 Linux 内核，花费 2 周时间写出最牛的分布式版本控制系统**Git**。

Linux 27周年

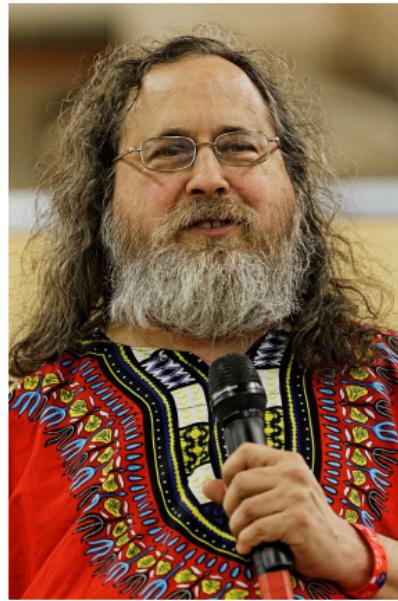
- Linux 之父，Git 之父，创造了开源世界
- 首个 Linux 版本 100% 由 Linus 完成
- 喜欢安静的写代码
- 超过 95% 的 Linux 是用 C 语言编写的
- 截至 2018, 内核已有 **20323379** 行代码



理查德·马修·斯托曼 (Richard.Matthew.Stallman, RMS)，自由软件运动的精神领袖、GNU计划以及自由软件基金会 (Free Software Foundation) 的创立者、著名黑客。他被许多人誉为当今自由软件的斗士、伟大的理想主义者。

开发的软件

- GCC

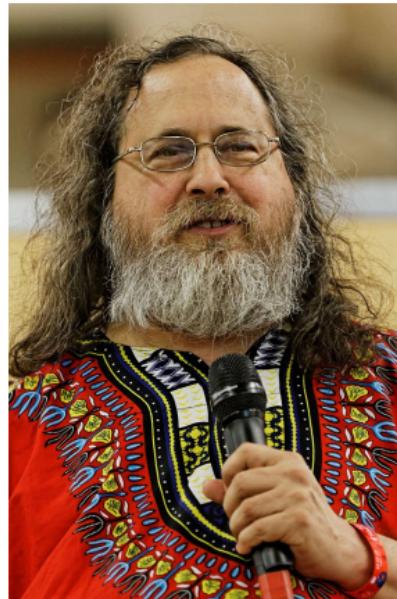


主要影响

理查德·马修·斯托曼 (Richard.Matthew.Stallman, RMS)，自由软件运动的精神领袖、GNU计划以及自由软件基金会 (Free Software Foundation) 的创立者、著名黑客。他被许多人誉为当今自由软件的斗士、伟大的理想主义者。

开发的软件

- GCC
- GNU Debugger



主要影响

理查德·马修·斯托曼 (Richard.Matthew.Stallman, RMS)，自由软件运动的精神领袖、GNU计划以及自由软件基金会 (Free Software Foundation) 的创立者、著名黑客。他被许多人誉为当今自由软件的斗士、伟大的理想主义者。

开发的软件

- GCC
- GNU Debugger
- GNU Emacs

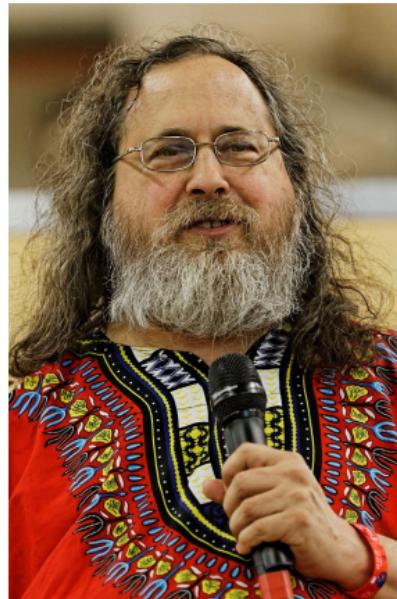


主要影响

理查德·马修·斯托曼 (Richard.Matthew.Stallman, RMS)，自由软件运动的精神领袖、GNU计划以及自由软件基金会 (Free Software Foundation) 的创立者、著名黑客。他被许多人誉为当今自由软件的斗士、伟大的理想主义者。

开发的软件

- GCC
- GNU Debugger
- GNU Emacs



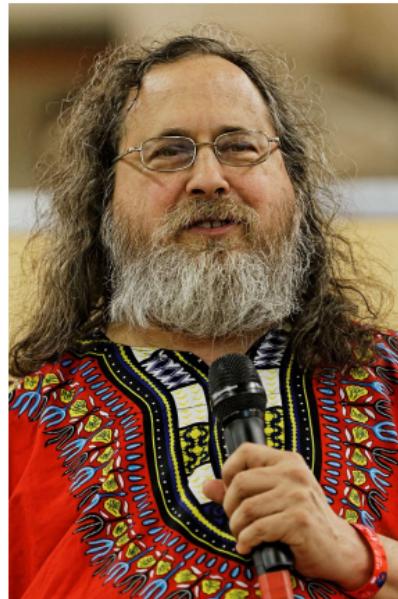
主要影响

- 神级程序员之一

理查德·马修·斯托曼 (Richard.Matthew.Stallman, RMS)，自由软件运动的精神领袖、GNU计划以及自由软件基金会 (Free Software Foundation) 的创立者、著名黑客。他被许多人誉为当今自由软件的斗士、伟大的理想主义者。

开发的软件

- GCC
- GNU Debugger
- GNU Emacs



主要影响

- 神级程序员之一
- 开源世界奠基人

理查德·马修·斯托曼 (Richard.Matthew.Stallman, RMS)，自由软件运动的精神领袖、GNU计划以及自由软件基金会 (Free Software Foundation) 的创立者、著名黑客。他被许多人誉为当今自由软件的斗士、伟大的理想主义者。

开发的软件

- GCC
- GNU Debugger
- GNU Emacs

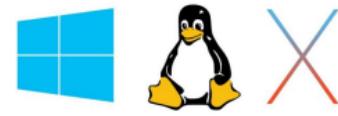
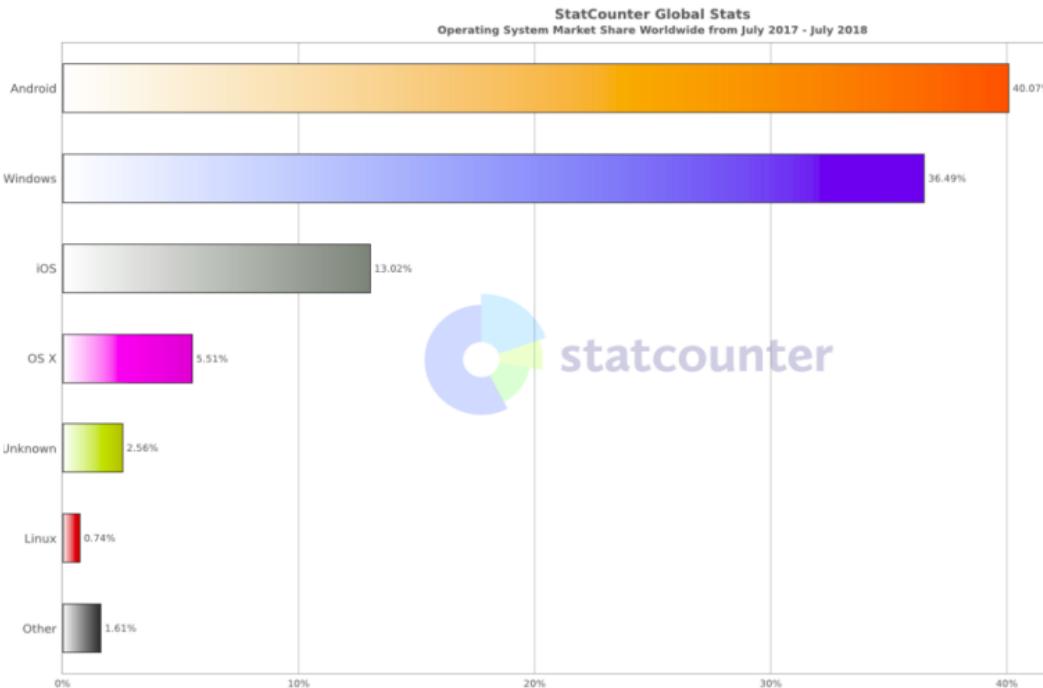


主要影响

- 神级程序员之一
- 开源世界奠基人
- GNU创立者

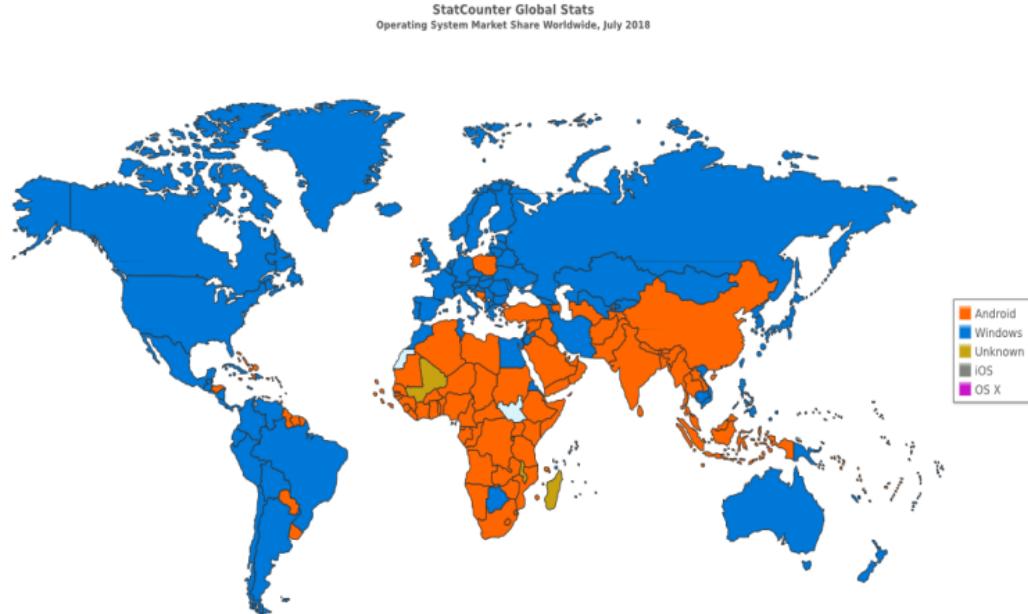
Linux 使用情况

操作系统最新使用情况，linux 只占0.74%



Linux 使用情况

从全球范围来看，操作系统最新使用情况



Linux 发行版

Linux的自由和地球的人口众多，语言五花八门决定了发行版的五花八门。



Linux 不得不说的发行版

- **Debian:** Ubuntu、Mint、Symphony、Xubuntu
- **Redhat:** CentOS、Fedora、SUSE、RHEL

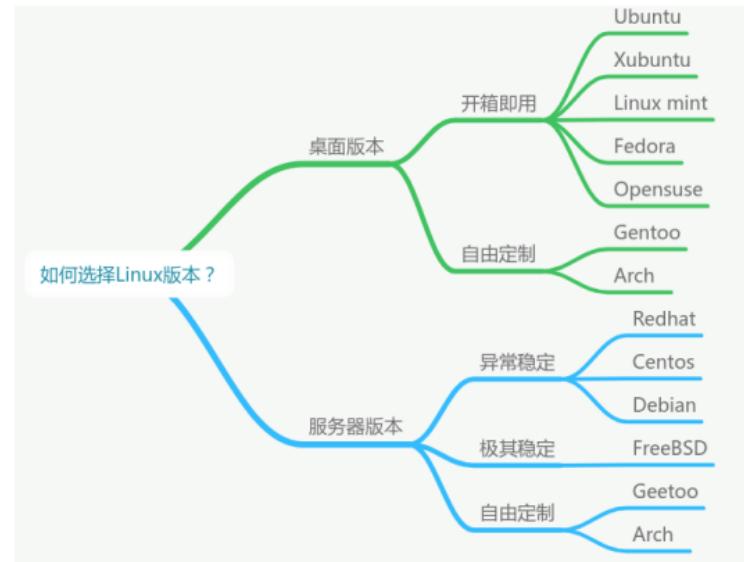
发行版太多

纷繁复杂的 Linux 发行版，**各自为阵**，不利推广。

如何选择发行版?

如何选择? 稳定易用不折腾、软件源之丰富

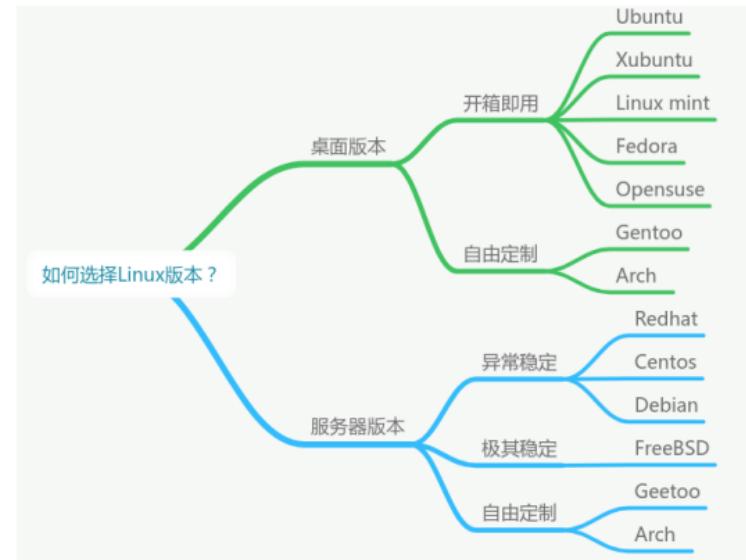
- 个人使用: ubuntu、 Mint



如何选择发行版?

如何选择? 稳定易用不折腾、软件源之丰富

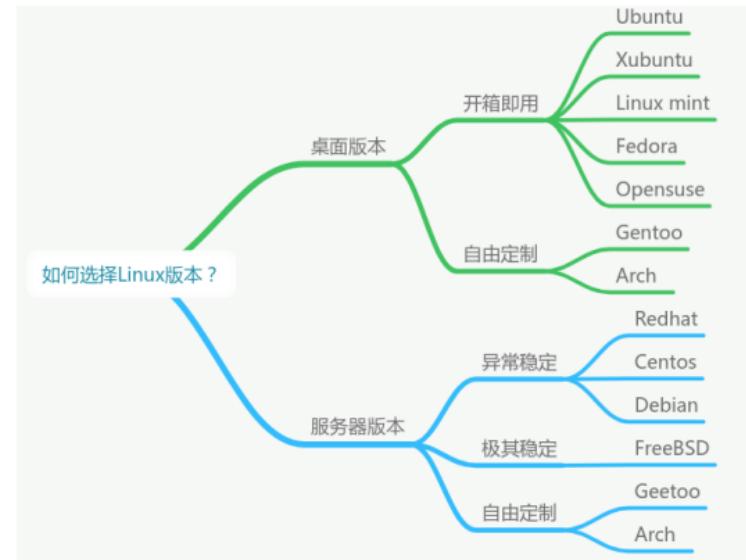
- 个人使用:ubuntu、Mint
- 程序开发:ubuntu、Mint



如何选择发行版?

如何选择? 稳定易用不折腾、软件源之丰富

- 个人使用:ubuntu、Mint
- 程序开发:ubuntu、Mint
- 服务器:Centos、RHEL



① 为什么要学 Linux?

② linux 介绍

③ Linux 基本知识

④ Linux 命令行

Linux 内核

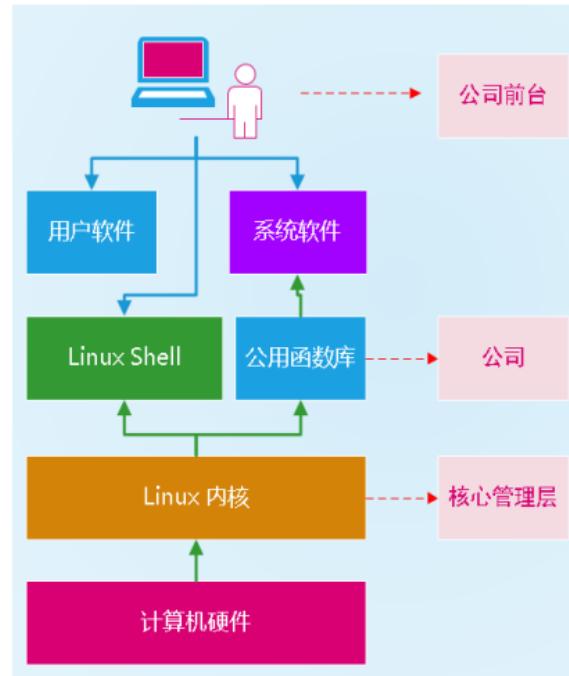
- 内核是操作系统的核心层

内核升级

Linux 内核支持热升级不用重启。

查询内核

查询内核 `uname -a`



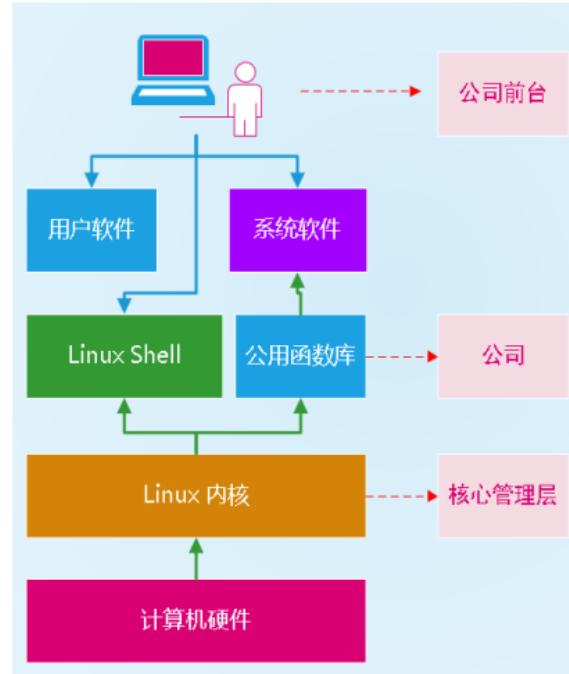
- 内核是操作系统的核心层
- 采用单文件块结构

内核升级

Linux 内核支持热升级不用重启。

查询内核

查询内核 `uname -a`



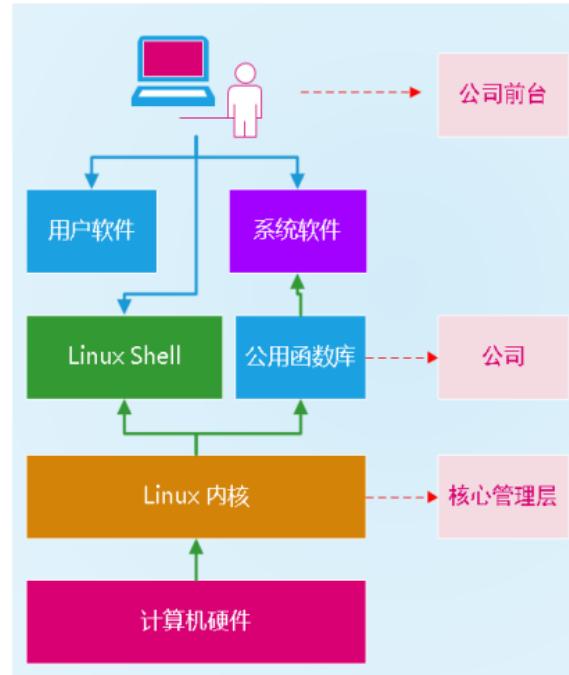
- 内核是操作系统的核心层
- 采用单文件块结构
- 最新的内核稳定版本为**4.18.5**

内核升级

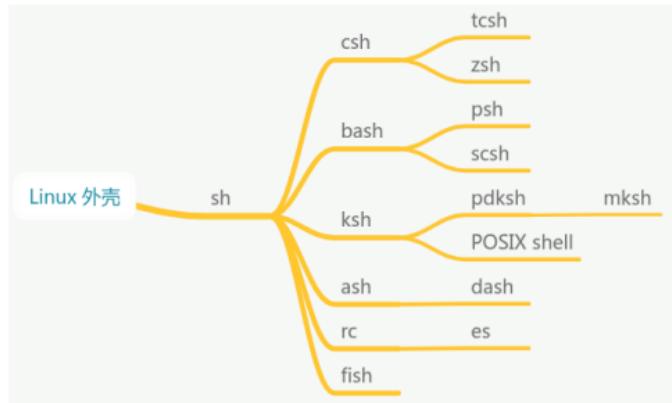
Linux 内核支持**热升级**不用重启。

查询内核

查询内核**uname -a**



- 光有个内核是干不了什么的

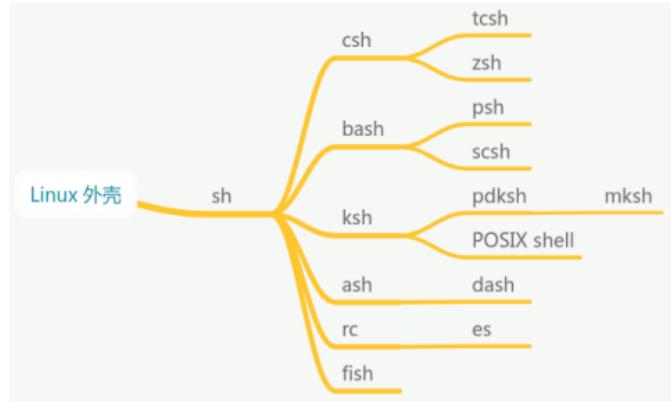


各种外壳

以开发时间从左到右依次排序，越靠后功能越强悍。

推荐的外壳

zsh、**bash**、**fish**



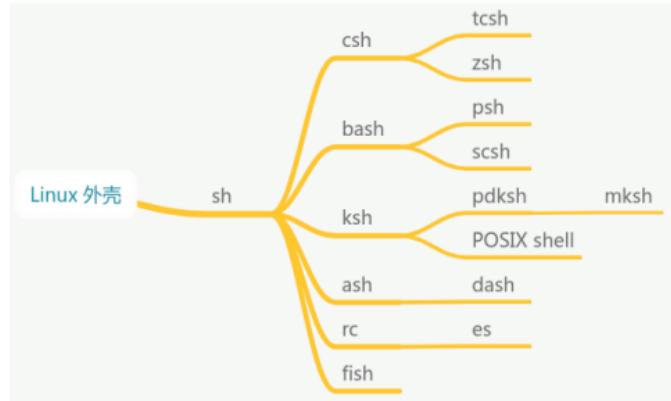
- 光有个内核是干不了什么的
- 提供访问内核的命令

各种外壳

以开发时间从左到右依次排序，越靠后功能越强悍。

推荐的外壳

zsh、**bash**、**fish**



- 光有个内核是干不了什么的
- 提供访问内核的命令
- 是内核上面的逻辑层

各种外壳

以开发时间从左到右依次排序，越靠后功能越强悍。

推荐的外壳

zsh、**bash**、**fish**

Linux 进程

- Linux 进程表示方式为 **进程号** 和 **进程名**
- 进程号从 0 开始
- Linux 的进程管理是父子制，单性繁殖

如何查进程

- 基本形式 **ps / ps -A**
- 按用户 **ps -u hls**
- 显示资源占用 **ps -aux**

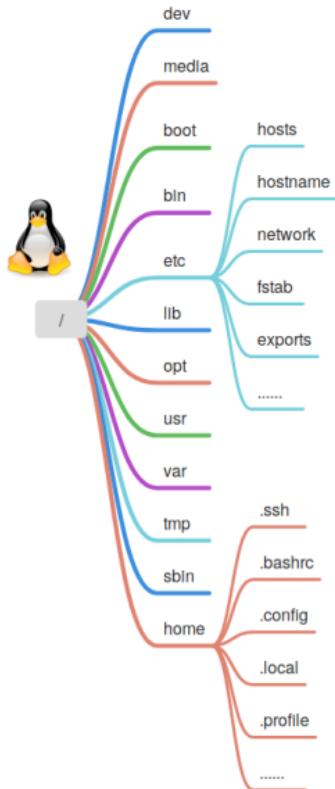
```
→ ~ ps -A | grep "chrome"
3037 tty1    00:00:54 chrome
3048 tty1    00:00:00 chrome
3052 tty1    00:00:00 chrome
3079 tty1    00:00:00 chrome
3127 tty1    00:00:00 chrome
3146 tty1    00:00:08 chrome
3177 tty1    00:00:25 chrome
3180 tty1    00:00:00 chrome
3278 tty1    00:00:00 chrome
3296 tty1    00:00:02 chrome
```

如何杀进程

- **PID:kill 5247**
- **进程名:killall -9 chrome**

```
→ ~ sudo killall -9 chrome
→ ~ ps | grep "gedit"
→ ~ ps -A | grep "gedit"
12293 ?        00:00:03 gedit
→ ~ sudo kill 12293
→ ~
```

UID	PID	PPID	C	S TIME	TTY	CMD
root	1	0	0	18:25	?	00:00:02 /sbin/
root	2	0	0	18:25	?	00:00:00 [kthre]
root	4	2	0	18:25	?	00:00:00 [kwork]
root	5	2	0	18:25	?	00:00:00 [kwork]
root	6	2	0	18:25	?	00:00:00 [mm_pe]
root	7	2	0	18:25	?	00:00:00 [ksoft]
root	8	2	0	18:25	?	00:00:00 [rcu_s]
root	9	2	0	18:25	?	00:00:00 [rcu_b]
root	10	2	0	18:25	?	00:00:00 [migrat]
root	11	2	0	18:25	?	00:00:00 [watch]
root	12	2	0	18:25	?	00:00:00 [cpuhp]
root	13	2	0	18:25	?	00:00:00 [cpuhp]
root	14	2	0	18:25	?	00:00:00 [watch]
root	15	2	0	18:25	?	00:00:00 [migrat]
root	16	2	0	18:25	?	00:00:00 [ksoft]
root	18	2	0	18:25	?	00:00:00 [kwork]
root	19	2	0	18:25	?	00:00:00 [kdevt]
root	20	2	0	18:25	?	00:00:00 [netns]
root	21	2	0	18:25	?	00:00:00 [rcu_t]
root	22	2	0	18:25	?	00:00:00 [kaudi]
root	23	2	0	18:25	?	00:00:00 [kwork]
root	24	2	0	18:25	?	00:00:00 [kwork]
root	25	2	0	18:25	?	00:00:00 [khung]
root	26	2	0	18:25	?	00:00:00 [oom_r]
root	27	2	0	18:25	?	00:00:00 [write]
root	28	2	0	18:25	?	00:00:00 [kcomp]
root	29	2	0	18:25	?	00:00:00 [ksmd]
root	30	2	0	18:25	?	00:00:00 [khuge]
root	31	2	0	18:25	?	00:00:00 [crypt]
root	32	2	0	18:25	?	00:00:00 [kinte]
root	33	2	0	18:25	?	00:00:00 [kbloc]
root	34	2	0	18:25	?	00:00:00 [kwork]



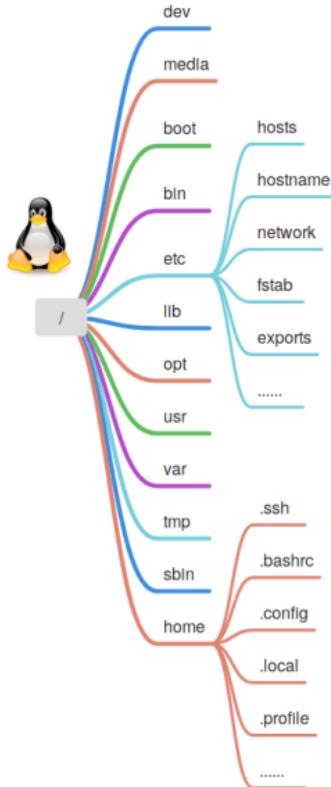
挂载点 mount point)

没有盘符、必须有**挂载点**。常见挂载点： / 、 /home 、 /usr

设备挂载方法

**mount [选项] [-source] <源> |
[-target] <目录>**

- 一切到是文件



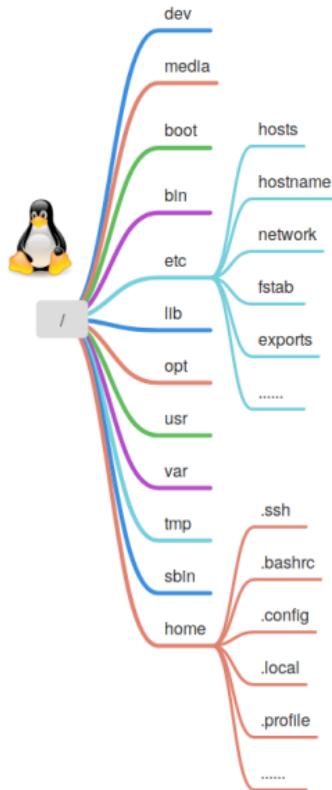
挂载点 mount point)

没有盘符、必须有**挂载点**。常见挂载点： / 、 /home 、 /usr

设备挂载方法

```
mount [选项] [-source] <源> |  
[-target] <目录>
```

- 一切到是文件
- 普通文件、目录、字符设备文件、块设备文件、符号链接文件



挂载点 mount point)

没有盘符、必须有**挂载点**。常见挂载点：/、/home、/usr

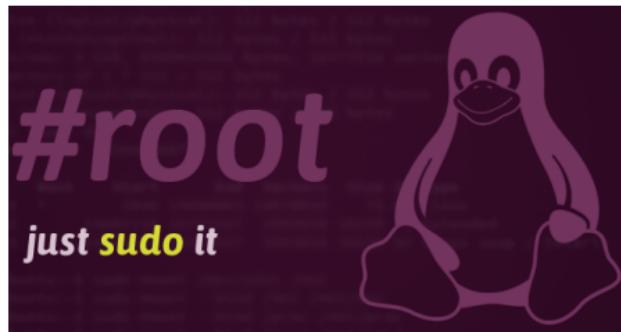
设备挂载方法

```
mount [选项] [-source] <源> |  
[-target] <目录>
```

- 一切到是文件
- 普通文件、目录、字符设备文件、块设备文件、符号链接文件
- 原生文件系统类型：**ext2 ext3 ext4**

用户和组

Linux 原生支持 **多用户多任务**。使用者 ID (User ID , 简称 **UID**)。群组 ID (Group ID , 简称 **GID**)。



关于 **root**

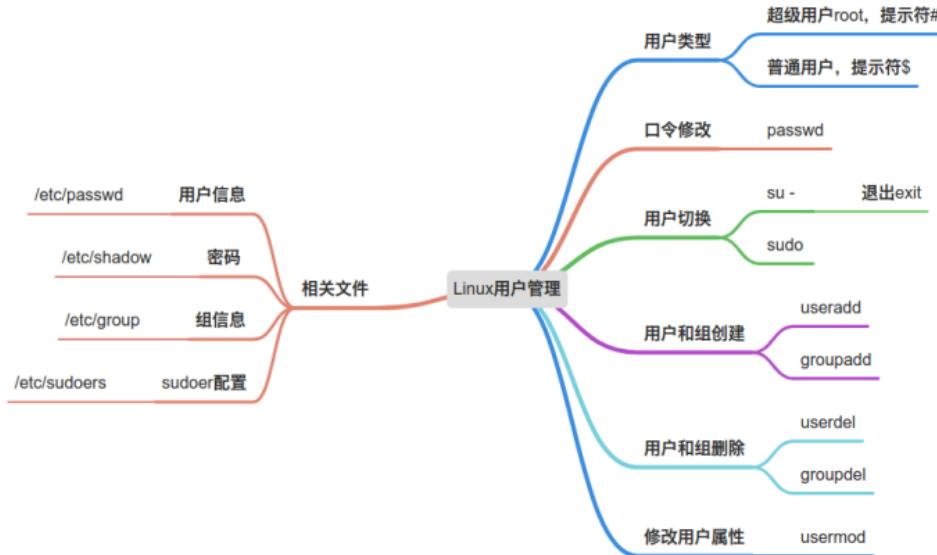
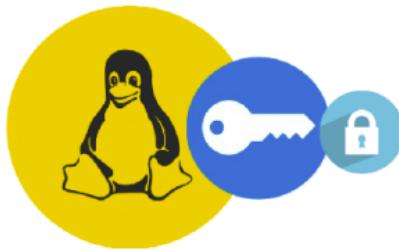
- 管理员账户，**root** 其 **UID=0**。
- 具有最高权限, 最高危险, 不建议使用。
- 身份切换：建议用 **sudo/su**

su 和 **sudo** 区别

su 需输切换用户密码，**sudo**仅需自己口令 (可配置)！

用户和组

Linux 下的用户等级实际上非常简单，就两个等级 root 和非 root。root 用户的权限是非常大，可自杀。



Linux 文件权限

Linux 文件权限

具有高效可靠的权限管理机制。

Linux 的权限

- 权限为7，代表可读、写、执行 (4+2+1)
- 权限为6，代表可读、写 (4+2)
- 权限为5，代表可读、执行 (4+1)
- 权限为4，代表可写、执行 (2+1)
- 权限为2，代表可写 (2)
- 权限为1，代表可执行 (1)



修改文件权限

修改文件权限

chmod设定不同的访问权限。**chown**更改文件或目录的所有者。**chgrp**命令更改文件或目录的用户组

```
→ ~ mkdir beamer-ppt #创建了beamer-ppt的文件夹
→ ~ ls -l | grep "ppt"
drwxr-xr-x 2 hls hls 4096 9月 9 14:45 beamer-ppt
→ ~ chmod u-x beamer-ppt #取消用户拥有者的执行权限
→ ~ ls -l | grep "ppt"
drw-r-xr-x 2 hls hls 4096 9月 9 14:45 beamer-ppt
→ ~ chmod u+x beamer-ppt #增加用户拥有者的执行权限
→ ~ ls -l | grep "ppt"
drwxr-xr-x 2 hls hls 4096 9月 9 14:45 beamer-ppt
→ ~ chmod g+wx beamer-ppt #增加用户组拥有者的执行权限
→ ~ ls -l | grep "ppt"
drwxrwxr-x 2 hls hls 4096 9月 9 14:45 beamer-ppt
→ ~ chmod -R g+wx beamer-ppt #增加-R, 代表递归子文件夹
→ ~ ls -l | grep "ppt"
drwxrwxrwx 2 hls hls 4096 9月 9 14:45 beamer-ppt
```

chmod [选项]... 模式[,模式]... 文件...

```
→ ~ chmod -R 777 beamer-ppt #修改成最大权限
→ ~ ls -l | grep "ppt"
drwxrwxrwx 2 hls hls 4096 9月 9 14:45 beamer-ppt
```

chmod [选项]... 八进制模式 文件...

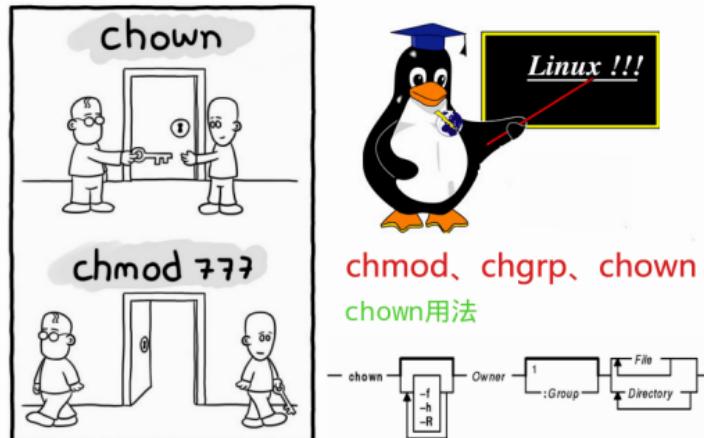
chmod使用方法

- 文字设定法
- **chmod [选项]... 模式[,模式]... 文件...**
- 数字设定法
- **chmod [选项]... 八进制模式 文件...**

修改文件所有者

修改文件权限

chown更改文件或目录的所有者。



```
→ ~ ls -l | grep "ppt"
drwxrwxrwx 2 hls hls 4096 9月 9 14:45 beamer-ppt
→ ~ chown -R root:root beamer-ppt #递归修改文件夹用户和所有者为root
chown: 正在更改 'beamer-ppt' 的所有者: 不允许的操作
→ ~ sudo chown -R root:root beamer-ppt #递归修改文件夹用户和所有者为root
```

chown 使用方法

- **chown** [选项]... [所有者] [: [组]] 文件...
- **chown** [选项]...
-reference= 参考文件文件...

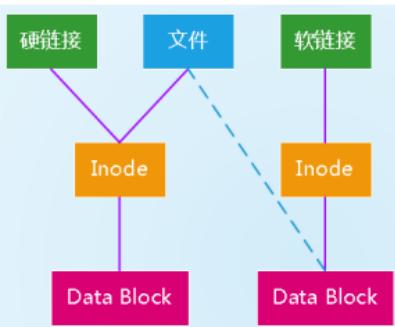
Linux 硬、软链接

Linux 的链接有两种：硬链接，软链接。简单理解为 Windows 的快捷方式。

作用：易于访问。Linux 中常用它来解决一些库版本的问题。

硬链接

- 与普通文件一样



软链接

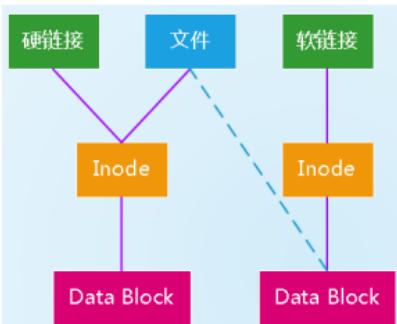
Linux 硬、软链接

Linux 的链接有两种：硬链接，软链接。简单理解为 Windows 的快捷方式。

作用：易于访问。Linux 中常用它来解决一些库版本的问题。

硬链接

- 与普通文件一样
- 指针向同一文件在硬盘区块



软链接

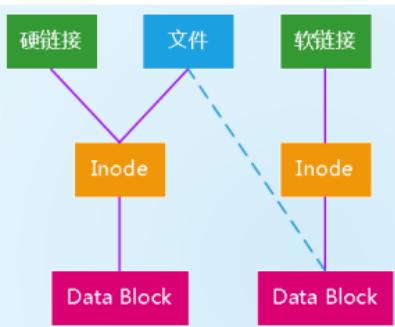
Linux 硬、软链接

Linux 的链接有两种：硬链接，软链接。简单理解为 Windows 的快捷方式。

作用：易于访问。Linux 中常用它来解决一些库版本的问题。

硬链接

- 与普通文件一样
- 指针向同一文件在硬盘区块
- inode 相同



软链接

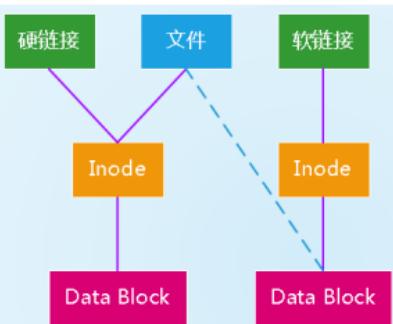
Linux 硬、软链接

Linux 的链接有两种：硬链接，软链接。简单理解为 Windows 的快捷方式。

作用：易于访问。Linux 中常用它来解决一些库版本的问题。

硬链接

- 与普通文件一样
- 指针向同一文件在硬盘区块
- inode 相同
- 不可跨文件系统



软链接

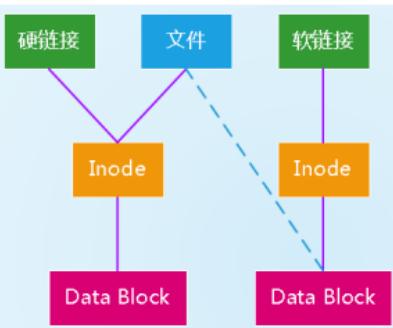
Linux 硬、软链接

Linux 的链接有两种：硬链接，软链接。简单理解为 Windows 的快捷方式。

作用：易于访问。Linux 中常用它来解决一些库版本的问题。

硬链接

- 与普通文件一样
- 指针向同一文件在硬盘区块
- inode 相同
- 不可跨文件系统



软链接

- 是另外一种不同文件

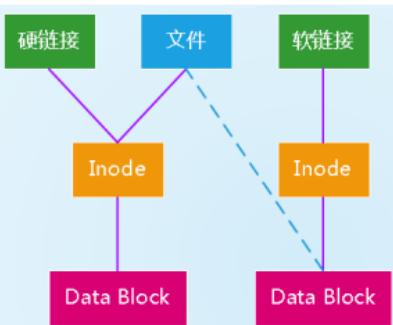
Linux 硬、软链接

Linux 的链接有两种：硬链接，软链接。简单理解为 Windows 的快捷方式。

作用：易于访问。Linux 中常用它来解决一些库版本的问题。

硬链接

- 与普通文件一样
- 指针向同一文件在硬盘区块
- inode 相同
- 不可跨文件系统



软链接

- 是另外一种不同文件
- 在硬盘上有独立的区块

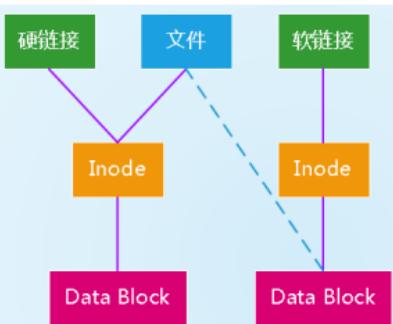
Linux 硬、软链接

Linux 的链接有两种：硬链接，软链接。简单理解为 Windows 的快捷方式。

作用：易于访问。Linux 中常用它来解决一些库版本的问题。

硬链接

- 与普通文件一样
- 指针向同一文件在硬盘区块
- inode 相同
- 不可跨文件系统



软链接

- 是另外一种不同文件
- 在硬盘上有独立的区块
- 关联了代表的文件绝对路径

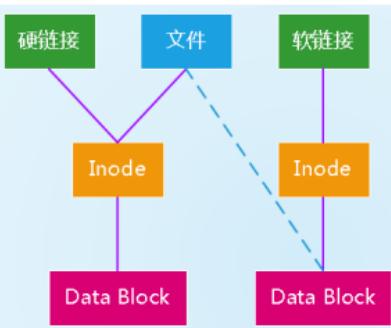
Linux 硬、软链接

Linux 的链接有两种：硬链接，软链接。简单理解为 Windows 的快捷方式。

作用：易于访问。Linux 中常用它来解决一些库版本的问题。

硬链接

- 与普通文件一样
- 指针向同一文件在硬盘区块
- inode 相同
- 不可跨文件系统



软链接

- 是另外一种不同文件
- 在硬盘上有独立的区块
- 关联了代表的文件绝对路径
- 权限和原文件权限无关系

更容易理解硬链接、软链接

```
→ ~ sudo killall -9 emacs
→ ~ touch hls-file && echo "这是一个文件" > hls-file
→ ~ cat hls-file
这是一个文件
→ ~ ln hls-file hard-link #创建一个硬链接，指向hls-file
→ ~ ls -li | ag "hls-file|hard"
849751 -rw-r--r-- 2 hls hls 19 9月 9 18:11 hard-link
849751 -rw-r--r-- 2 hls hls 19 9月 9 18:11 hls-file
```

结论：硬链接和真实文件对应的inode一样。

```
→ ~ echo "添加一个新行" >> hard-link #往硬链接添加文件
→ ~ cat hls-file #查看真实文件
这是一个文件
添加一个新行
```

结论：往硬链接写东西，实际指向真实文件

```
→ ~ ln -s hls-file soft-link #添加一个软链接，关联到hls-file
→ ~ ls -li | ag "hls-file|hard"
849751 -rw-r--r-- 2 hls hls 38 9月 9 18:21 hard-link
849751 -rw-r--r-- 2 hls hls 38 9月 9 18:21 hls-file
854829 lrwxrwxrwx 1 hls hls 8 9月 9 18:23 soft-link -> hls-file
```

软链接指向真实文件，但inode不一样

```
→ ~ rm hls-file #删除真实文件
→ ~ cat hard-link #删除后，查硬链接
这是一个文件
添加一个新行
→ ~ cat soft-link #删除后，查软链接
cat: soft-link: 没有那个文件或目录
```

删除真实文件，但硬链接文件还在。软链接指向不存在了。

- 硬链接是指针

怎么建立硬链接

- **ln [选项]... [-T] 目标链接名**
- **ln [选项]... -t 目录目标...**

怎么建立软链接

- **ln [选项]... [-T] 目标链接名**
- **ln -s... -t 目录目标...**

更容易理解硬链接、软链接

```
→ ~ sudo killall -9 emacs
→ ~ touch hls-file && echo "这是一个文件" > hls-file
→ ~ cat hls-file
这是一个文件
→ ~ ln hls-file hard-link #创建一个硬链接，指向hls-file
→ ~ ls -li | ag "hls-file|hard"
849751 -rw-r--r-- 2 hls hls 19 9月 9 18:11 hard-link
849751 -rw-r--r-- 2 hls hls 19 9月 9 18:11 hls-file
```

结论：硬链接和真实文件对应的inode一样。

```
→ ~ echo "添加一个新行" >> hard-link #往硬链接添加文件
→ ~ cat hls-file #查看真实文件
这是一个文件
添加一个新行
```

结论：往硬链接写东西，实际指向真实文件

```
→ ~ ln -s hls-file soft-link #添加一个软链接，关联到hls-file
→ ~ ls -li | ag "hls-file|hard"
849751 -rw-r--r-- 2 hls hls 38 9月 9 18:21 hard-link
849751 -rw-r--r-- 2 hls hls 38 9月 9 18:21 hls-file
854829 lrwxrwxrwx 1 hls hls 8 9月 9 18:23 soft-link -> hls-file
```

软链接指向真实文件，但inode不一样

```
→ ~ rm hls-file #删除真实文件
→ ~ cat hard-link #删除后，查硬链接
这是一个文件
添加一个新行
→ ~ cat soft-link #删除后，查软链接
cat: soft-link: 没有那个文件或目录
```

删除真实文件，但硬链接文件还在。软链接指向不存在了。

- 硬链接是指针
- 软链接是另外一种类型文件

怎么建立硬链接

- **ln [选项]... [-T] 目标链接名**
- **ln [选项]... -t 目录目标...**

怎么建立软链接

- **ln [选项]... [-T] 目标链接名**
- **ln -s... -t 目录目标...**

网络接口

Linux **kernel** 内核通常区分为两种软件的网络接口。一是物理网络接口，二是虚拟网络接口。

接口类型	接口名称	接口说明
以太网接口	ethX、wlanX、enpX	最常见的网络接口
令牌环接口	trX	少数网络环境
光纤分布式数据接口	fddiX	核心网或高速网络中
点对点协议接口	pppX	拨号网络或基于 PPTP 协议
本地回环接口	lo	并非真实存在



Linux 网络配置

查看网络接口信息

ifconfig

配置网络接口信息

ifconfig <interface>

生效方式

- 临时性网络配置：直接用命令
- 永久性网络配置：修改配置文件

启用接口

ifup <options> <ifaces...>

停用接口

ifdown <options> <ifaces...>

Linux 桌面环境

桌面环境结合 X 客户端，提供通用图形用户界面元素，如图标、工具栏、壁纸、桌面小部件。用户可以自由搭配不同桌面环境的程序，桌面环境只是提供一个完整的和方便的方法完成这项任务。

判定桌面的唯一标准：更好用、更高效

重量桌面

- GNOME



轻量桌面

Linux 桌面环境

桌面环境结合 X 客户端，提供通用图形用户界面元素，如图标、工具栏、壁纸、桌面小部件。用户可以自由搭配不同桌面环境的程序，桌面环境只是提供一个完整的和方便的方法完成这项任务。

判定桌面的唯一标准：更好用、更高效

重量桌面

- GNOME
- Unity



轻量桌面

Linux 桌面环境

桌面环境结合 X 客户端，提供通用图形用户界面元素，如图标、工具栏、壁纸、桌面小部件。用户可以自由搭配不同桌面环境的程序，桌面环境只是提供一个完整的和方便的方法完成这项任务。

判定桌面的唯一标准：更好用、更高效

重量桌面

- GNOME
- Unity
- KDE



轻量桌面

Linux 桌面环境

桌面环境结合 X 客户端，提供通用图形用户界面元素，如图标、工具栏、壁纸、桌面小部件。用户可以自由搭配不同桌面环境的程序，桌面环境只是提供一个完整的和方便的方法完成这项任务。

判定桌面的唯一标准：更好用、更高效

重量桌面

- GNOME
- Unity
- KDE
- Cinnamon



轻量桌面

Linux 桌面环境

桌面环境结合 X 客户端，提供通用图形用户界面元素，如图标、工具栏、壁纸、桌面小部件。用户可以自由搭配不同桌面环境的程序，桌面环境只是提供一个完整的和方便的方法完成这项任务。

判定桌面的唯一标准：更好用、更高效

重量桌面

- GNOME
- Unity
- KDE
- Cinnamon



轻量桌面

- Xfce

Linux 桌面环境

桌面环境结合 X 客户端，提供通用图形用户界面元素，如图标、工具栏、壁纸、桌面小部件。用户可以自由搭配不同桌面环境的程序，桌面环境只是提供一个完整的和方便的方法完成这项任务。

判定桌面的唯一标准：更好用、更高效

重量桌面

- GNOME
- Unity
- KDE
- Cinnamon



轻量桌面

- Xfce
- LXDE

Linux 桌面环境

桌面环境结合 X 客户端，提供通用图形用户界面元素，如图标、工具栏、壁纸、桌面小部件。用户可以自由搭配不同桌面环境的程序，桌面环境只是提供一个完整的和方便的方法完成这项任务。

判定桌面的唯一标准：更好用、更高效

重量桌面

- GNOME
- Unity
- KDE
- Cinnamon



轻量桌面

- Xfce
- LXDE
- MATE

Linux 桌面环境

桌面环境结合 X 客户端，提供通用图形用户界面元素，如图标、工具栏、壁纸、桌面小部件。用户可以自由搭配不同桌面环境的程序，桌面环境只是提供一个完整的和方便的方法完成这项任务。

判定桌面的唯一标准：更好用、更高效

重量桌面

- GNOME
- Unity
- KDE
- Cinnamon



轻量桌面

- Xfce
- LXDE
- MATE
- fvwm

GNU/Linux 软件世界？

Linux 软件非常丰富，满足大部分日常工作完全没有问题。Linux 软件原则之一：**small and simple**。

- 除内核外，都是以软件形式存在的

RPM 软件家族

DEB 软件家族



GNU/Linux 软件世界？

Linux 软件非常丰富，满足大部分日常工作完全没有问题。Linux 软件原则之一：**small and simple**。

- 除内核外，都是以软件形式存在的
- 源文件编译安装和二进制软件包安装

RPM 软件家族

DEB 软件家族



GNU/Linux 软件世界？

linux 软件非常丰富，满足大部分日常工作完全没有问题。Linux 软件原则之一：**small and simple**。

- 除内核外，都是以软件形式存在的
- 源文件编译安装和二进制软件包安装
- 安装时需要解决软件所有依赖的文件



RPM 软件家族

DEB 软件家族



GNU/Linux 软件世界？

Linux 软件非常丰富，满足大部分日常工作完全没有问题。Linux 软件原则之一：**small and simple**。

- 除内核外，都是以软件形式存在的
- 源文件编译安装和二进制软件包安装
- 安装时需要解决软件所有依赖的文件



RPM 软件家族

- 红帽包管理 rpm

DEB 软件家族



GNU/Linux 软件世界？

linux 软件非常丰富，满足大部分日常工作完全没有问题。Linux 软件原则之一：**small and simple**。

- 除内核外，都是以软件形式存在的
- 源文件编译安装和二进制软件包安装
- 安装时需要解决软件所有依赖的文件



RPM 软件家族

- 红帽包管理 rpm
- rpm 不能解决依赖

DEB 软件家族



GNU/Linux 软件世界？

linux 软件非常丰富，满足大部分日常工作完全没有问题。Linux 软件原则之一：**small and simple**。

- 除内核外，都是以软件形式存在的
- 源文件编译安装和二进制软件包安装
- 安装时需要解决软件所有依赖的文件



RPM 软件家族

- 红帽包管理 rpm
- rpm 不能解决依赖
- 改进版本 yum

DEB 软件家族



GNU/Linux 软件世界？

linux 软件非常丰富，满足大部分日常工作完全没有问题。Linux 软件原则之一：**small and simple**。

- 除内核外，都是以软件形式存在的
- 源文件编译安装和二进制软件包安装
- 安装时需要解决软件所有依赖的文件

RPM 软件家族

- 红帽包管理 **rpm**
- **rpm** 不能解决依赖
- 改进版本 **yum**

DEB 软件家族

- 适合 **debian** 系



GNU/Linux 软件世界？

linux 软件非常丰富，满足大部分日常工作完全没有问题。Linux 软件原则之一：**small and simple**。

- 除内核外，都是以软件形式存在的
- 源文件编译安装和二进制软件包安装
- 安装时需要解决软件所有依赖的文件

RPM 软件家族

- 红帽包管理 **rpm**
- **rpm** 不能解决依赖
- 改进版本 **yum**

DEB 软件家族

- 适合 **debian** 系
- 自动解决依赖



GNU/Linux 软件世界？

linux 软件非常丰富，满足大部分日常工作完全没有问题。Linux 软件原则之一：**small and simple**。

- 除内核外，都是以软件形式存在的
- 源文件编译安装和二进制软件包安装
- 安装时需要解决软件所有依赖的文件

RPM 软件家族

- 红帽包管理 **rpm**
- **rpm** 不能解决依赖
- 改进版本 **yum**

DEB 软件家族

- 适合 **debian** 系
- 自动解决依赖
- 丰富软件源



apt 包管理命令使用



- 根据名称列出软件包 : `apt list`

apt 包管理命令使用



- 根据名称列出软件包 : apt **list**
- 更新可用软件包列表 : apt **update**

apt 包管理命令使用



- 根据名称列出软件包 : apt **list**
- 更新可用软件包列表 : apt **update**
- 安装软件包 : apt **install xx**

apt 包管理命令使用



- 根据名称列出软件包 : apt **list**
- 更新可用软件包列表 : apt **update**
- 安装软件包 : apt **install xx**
- 搜索软件包描述 : apt **search xx**

apt 包管理命令使用



- 根据名称列出软件包 : apt **list**
- 更新可用软件包列表 : apt **update**
- 安装软件包 : apt **install xx**
- 搜索软件包描述 : apt **search xx**
- 移除软件包 : apt **remove xx**

yum 包管理命令使用

- 根据名称列出软件包 : `yum list`



yum 包管理命令使用

- 根据名称列出软件包 : `yum list`
- 更新可用软件包列表 : `yum update`



yum 包管理命令使用

- 根据名称列出软件包 : `yum list`
- 更新可用软件包列表 : `yum update`
- 安装软件包 : `yum install xx`



yum 包管理命令使用

- 根据名称列出软件包 : `yum list`
- 更新可用软件包列表 : `yum update`
- 安装软件包 : `yum install xx`
- 搜索软件包描述 : `yum search xx`



yum 包管理命令使用

- 根据名称列出软件包 : `yum list`
- 更新可用软件包列表 : `yum update`
- 安装软件包 : `yum install xx`
- 搜索软件包描述 : `yum search xx`
- 移除软件包 : `yum remove xx`



Linux 应用软件

文本编辑 *vim*, *emacs*



工具 *grep*, *ag*, *sed*,
tmux, *zsh*...



开发 *git*, *eclipse*, *Android studio*...



办公 *wps*, *chrome*, *firefox*, *gimp*, *LAT*EX...



① 为什么要学 Linux?

② linux 介绍

③ Linux 基本知识

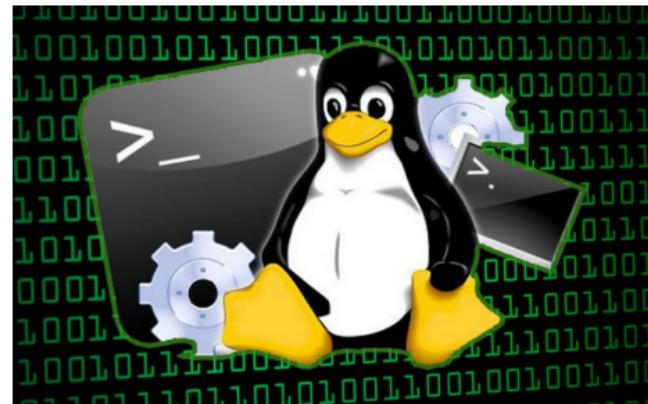
④ Linux 命令行

关于 Linux 命令行

图形用户界面让简单的任务更容易完成，而命令行界面使完成复杂的任务成为可能。一个好的命令行界面，是用来和计算机进行交流沟通的非常有效的方式，正像人类社会使用文字互通信息一样。

- 探究命令行基本语言

高效的命令行

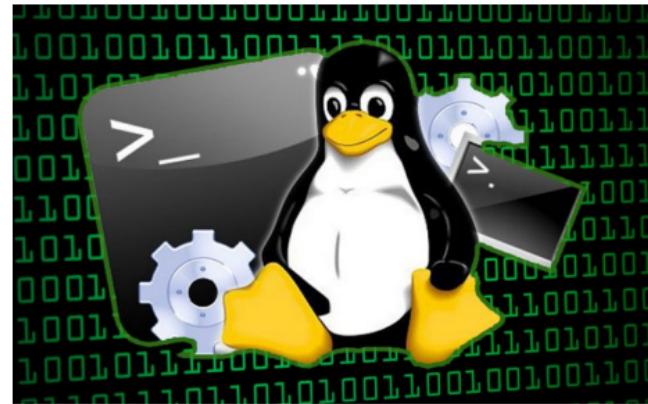


关于 Linux 命令行

图形用户界面让简单的任务更容易完成，而命令行界面使完成复杂的任务成为可能。一个好的命令行界面，是用来和计算机进行交流沟通的非常有效的方式，正像人类社会使用文字互通信息一样。

- 探究命令行基本语言
- 记住常用的命令

高效的命令行

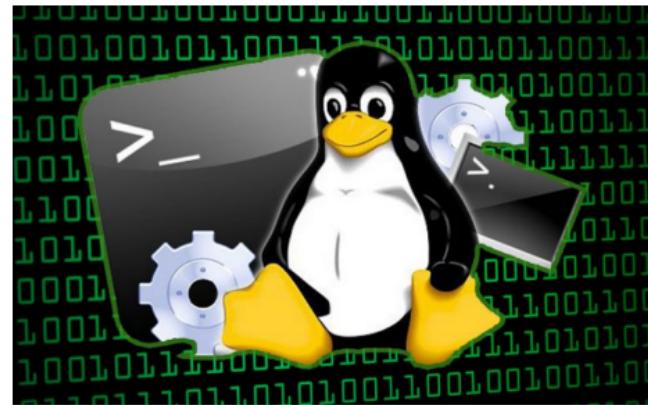


关于 Linux 命令行

图形用户界面让简单的任务更容易完成，而命令行界面使完成复杂的任务成为可能。一个好的命令行界面，是用来和计算机进行交流沟通的非常有效的方式，正像人类社会使用文字互通信息一样。

- 探究命令行基本语言
- 记住常用的命令
- 执行的普通任务

高效的命令行

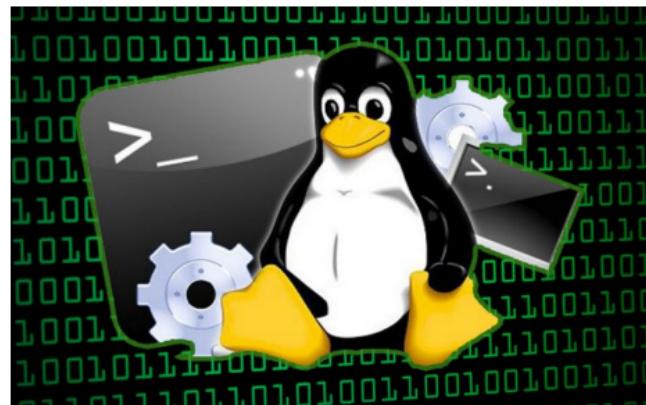


关于 Linux 命令行

图形用户界面让简单的任务更容易完成，而命令行界面使完成复杂的任务成为可能。一个好的命令行界面，是用来和计算机进行交流沟通的非常有效的方式，正像人类社会使用文字互通信息一样。

- 探究命令行基本语言
- 记住常用的命令
- 执行的普通任务
- 编写 Shell 脚本/编程

高效的命令行



关于 Linux 命令行

使用 Linux 桌面发行版，可以不懂得任何 Linux 命令。**shell** 就是一个程序，它接受从键盘输入的命令，然后把命令传递给操作系统去执行。几乎所有的 Linux 发行版都提供一个名为 **bash** 的来自 GNU 项目的 shell 程序。

最常用的一些命令



经常使用的命令行

若是了解一部分 Linux 命令，可以更好的使用 Linux，体验它的魅力。

- 查看帮助
- 目录、文件、搜索
- 输入输出
- 网络管理

学会第一个 ssh 命令

- 安全外壳协议，简称 **SSH**

使用 ssh 登录远程服务器

```
➜ ~ ssh -p 2223 hls@192.168.1.4 #登录远程的一个服务器  
Welcome to Linux Mint 18.3 Sylvia (GNU/Linux 4.4.0-119-generic x86_64)  
  
* Documentation: https://www.linuxmint.com  
Last login: Mon Sep 24 12:21:36 2018 from 192.168.1.188  
hls@gen8 ~ $
```

将远端服务器的文件下载到本地

```
➜ ~ scp -P 2223 hls@192.168.1.4:/home/hls/hls-autogit.sh ~ #从远程 cp 文件到本机  
hls-autogit.sh 100% 190 116.7KB/s 00:00  
➜ ~
```

将本地的 ssh 密钥复制到远端服务器

```
➜ ~ ssh-copy-id -p 2223 hls@192.168.1.4 #将本地 ssh key 复制到远端  
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter  
out any that are already installed  
  
/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist  
on the remote system.  
      (if you think this is a mistake, you may want to use -f option)
```

怎么 ssh 登录

- **ssh [user@]hostname**
- **ssh [-p port] [user@]hostname**

怎么传输文件

- **scp [[user@]host1:]file1 ... [[user@]host2:]file2**
- **scp [-P port] [[user@]host1:]file1 ... [[user@]host2:]file2**

学会第一个 ssh 命令

- 安全外壳协议，简称 **SSH**
- 一种 **加密** 网络传输协议

使用 ssh 登录远程服务器

```
➜ ~ ssh -p 2223 hls@192.168.1.4 #登录远程的一个服务器
Welcome to Linux Mint 18.3 Sylvia (GNU/Linux 4.4.0-119-generic x86_64)

 * Documentation: https://www.linuxmint.com
Last login: Mon Sep 24 12:21:36 2018 from 192.168.1.188
hls@gen8 ~ $
```

将远端服务器的文件下载到本地

```
➜ ~ scp -P 2223 hls@192.168.1.4:/home/hls/hls-autogit.sh . #从远程cp文件到本机
hls-autogit.sh          100% 190   116.7KB/s  00:00
➜ ~
```

将本地的 ssh 密钥复制到远端服务器

```
➜ ~ ssh-copy-id -p 2223 hls@192.168.1.4 #将本地 ssh key复制到远端
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist
on the remote system.
      (if you think this is a mistake, you may want to use -f option)
```

怎么 ssh 登录

- **ssh [user@]hostname**
- **ssh [-p port] [user@]hostname**

怎么传输文件

- **scp [[user@]host1:]file1 ... [[user@]host2:]file2**
- **scp [-P port] [[user@]host1:]file1 ... [[user@]host2:]file2**

学会第一个 ssh 命令

使用 ssh 登录远程服务器

```
➜ ~ ssh -p 2223 hls@192.168.1.4 #登录远程的一个服务器  
Welcome to Linux Mint 18.3 Sylvia (GNU/Linux 4.4.0-119-generic x86_64)  
  
* Documentation: https://www.linuxmint.com  
Last login: Mon Sep 24 12:21:36 2018 from 192.168.1.188  
hls@gen8 ~ $
```

将远端服务器的文件下载到本地

```
➜ ~ scp -P 2223 hls@192.168.1.4:/home/hls/hls-autogit.sh ~ #从远程 cp 文件到本机  
hls-autogit.sh 100% 190 116.7KB/s 00:00  
➜ ~
```

将本地的 ssh 密钥复制到远端服务器

```
➜ ~ ssh-copy-id -p 2223 hls@192.168.1.4 #将本地 ssh key 复制到远端  
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter  
out any that are already installed  
  
/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist  
on the remote system.  
      (if you think this is a mistake, you may want to use -f option)
```

- 安全外壳协议，简称 **SSH**
- 一种 **加密** 网络传输协议
- 专为远程登录会话

怎么 ssh 登录

- **ssh [user@]hostname**
- **ssh [-p port] [user@]hostname**

怎么传输文件

- **scp [[user@]host1:]file1 ... [[user@]host2:]file2**
- **scp [-P port] [[user@]host1:]file1 ... [[user@]host2:]file2**

使用 screen 管理你的远程会话

使用 screen 管理你的远程会话.

screen 的作用是，只要服务器不 shutdown，我的程序就要一直跑

- 避免因为 ssh 远程登录连接之类中断，引起的程序中断



新建会话

重新连接会话

```
查看screen列表
Administrator: ~ $ screen -ls
There is a screen on:
 21433.pts-0      (2018-09-24 15:12:00)  (Detached)
 1 Socket in /var/run/screen/S-21433.

重新连接screen会话
Administrator: ~ $ screen -r 21433
Administrator: ~ $
```

使用 screen 管理你的远程会话

使用 screen 管理你的远程会话.

screen 的作用是，只要服务器不 shutdown，我的程序就要一直跑

- 避免因为 ssh 远程登录连接之类中断，引起的程序中断

新建会话

- screen 建立 hls 会话

重新连接会话

```
查看screen列表
Administrator ~ % screen -ls
There is a screen on:
 21433.pts-0      (2018-09-24 15:12:00)  (Detached)
 1 Socket in /var/run/screen/S-21433.

重新连接screen会话
Administrator ~ % screen -r 21433
Administrator ~ % screen -ls
There is a screen on:
 21433.pts-0      (2018-09-24 15:12:00)  (Attached)
 1 Socket in /var/run/screen/S-21433.

Administrator ~ %
```

使用 screen 管理你的远程会话

使用 screen 管理你的远程会话.

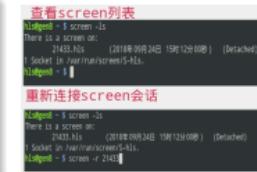
screen 的作用是，只要服务器不 **shutdown**，我的程序就要一直跑

- 避免因为 **ssh** 远程登录连接之类中断，引起的程序中断

新建会话

- screen** 建立 **hls** 会话
- screen -S hls**

重新连接会话



```
查看screen列表
Administrator: ~ $ screen -ls
There is a screen on:
 21433.pts-0      (2018-09-24 15:01:22.000)
 1 Socket in /var/run/screen/S-21433.
Administrator: ~ |
```

```
重新连接screen会话
Administrator: ~ $ screen -r
There is a screen on:
 21433.pts-0      (2018-09-24 15:01:22.000)  (Detached)
 1 Socket in /var/run/screen/S-21433.
Administrator: ~ |
```

使用 screen 管理你的远程会话

使用 screen 管理你的远程会话.

screen 的作用是，只要服务器不 **shutdown**，我的程序就要一直跑

- 避免因为 **ssh** 远程登录连接之类中断，引起的程序中断

新建会话

- screen** 建立 **hls** 会话
- screen -S hls**

重新连接会话

- 查看任务 ID: **screen -ls**



```
查看screen列表
Administrator: ~ $ screen -ls
There is a screen on:
 21433.pts-0      (2018-09-24 15:01:12+0800)  (Detached)
 1 Socket in /var/run/screen/S-21433.
Administrator: ~ $
```



```
重新连接screen会话
Administrator: ~ $ screen -r
There is a screen on:
 21433.pts-0      (2018-09-24 15:01:12+0800)  (Detached)
 1 Socket in /var/run/screen/S-21433.
Administrator: ~ $ screen -r 21433
```

使用 screen 管理你的远程会话

使用 screen 管理你的远程会话.

screen 的作用是，只要服务器不 **shutdown**，我的程序就要一直跑

- 避免因为 **ssh** 远程登录连接之类中断，引起的程序中断

新建会话

- screen** 建立 **hls** 会话
- screen -S hls**

重新连接会话

- 查看任务 ID: **screen -ls**
- 重连 **screen -r 99871**



The screenshot shows two terminal windows. The top window is titled '查看screen列表' (View screen list) and displays the command 'root@node1 ~ \$ screen -ls'. It shows one session named '21433.hls' with a status of '(Detached)'. The bottom window is titled '重新连接screen会话' (Reconnect to screen session) and displays the command 'root@node1 ~ \$ screen -r 21433'. It shows the session has been reattached.

Linux 文件操作

新建 *touch*、*mkdir*



复制 *cp*



移动或改名 *mv*



删除 *rm*



新建 touch

新建 touch

touch 用来修改文件时间戳，或者新建一个不存在的文件。

```
→ ~ cd tmp
→ tmp ls
pdf2htmlEX  poppler-0.64.0  poppler-0.64.0.tar.xz  zsh-syntax-highlighting
→ tmp touch test
→ tmp ll      创建一个空文件
总用量 1.4M
drwxr-xr-x  9 hls hls 4.0K 4月  29 17:19 pdf2htmlEX
drwxrwxr-x 12 hls hls 4.0K 4月  18 02:04 poppler-0.64.0
-rw-rw-r--  1 hls hls 1.4M 4月  29 17:25 poppler-0.64.0.tar.xz
-rw-r--r--  1 hls hls    0 9月  26 21:16 test
drwxr-xr-x  7 hls hls 4.0K 4月  30 08:07 zsh-syntax-highlighting
→ tmp
```

touch [选项]... 文件...

- -a 只更改存取时间
- -c 不建立任何文档
- -t 使用指定的日期时间

新建 mkdir

新建 `mkdir` 命令是用来创建目录的

- `mkdir [选项]... 目录...`

```
+ cmp -s /tmp/poppler-0.64.0.tar.xz /tmp/zsh-syntax-highlighting.tar.xz
+ rm /tmp/poppler-0.64.0.tar.xz /tmp/zsh-syntax-highlighting.tar.xz
+ tmp mkdir dir1 #在当前目录下创建dir1目录
+ tmp ls
dir1      poppler-0.64.0      test
pdf2htmlEX  poppler-0.64.0.tar.xz  zsh-syntax-highlighting
+ tmp dir -p dir2/dir3/dir4/dir5 #创建多层目录
+ tmp cd dir2/dir3/dir4/dir5
+ dir5 pwd
/home/hls/tmp/dir2/dir3/dir4/dir5
+ dir5 mkdir -m 700 dir6 #创建带权限的目录
+ dir5 ll
总用量 4.0K
drwx----- 2 hls hls 4.0K 9月 28 20:34 dir6
+ tmp mkdir -vp src/{lib/bin/,}doc/{info,product},logs/{info,product},deploy/{info,product}
mkdir: 已创建目录 'src'
mkdir: 已创建目录 'src/lib/'
mkdir: 已创建目录 'src/bin/'
mkdir: 已创建目录 'src/doc/'
mkdir: 已创建目录 'src/doc/info'
mkdir: 已创建目录 'src/doc/product'
mkdir: 已创建目录 'src/logs'
mkdir: 已创建目录 'src/logs/info'
mkdir: 已创建目录 'src/logs/product'
mkdir: 已创建目录 'src/deploy'
mkdir: 已创建目录 'src/deploy/info'
mkdir: 已创建目录 'src/deploy/product'
+ tmp tree src
src
├── bin
├── deploy
│   └── info
│       └── product
├── doc
│   ├── info
│   └── product
└── lib
    └── logs
        └── info
            └── product
11 directories, 0 files
```

新建 mkdir

新建 `mkdir` 命令是用来创建目录的

- `mkdir [选项]... 目录...`
- 使用 `-p` 参数可以创建多层目录

```
+ cmp -s /tmp/poppler-0.64.0.tar.xz /tmp/zsh-syntax-highlighting.tar.xz
+ tmp mkdir dir1 #在当前目录下创建dir1目录
+ tmp ls
dir1  poppler-0.64.0  test
pdf2htmlEX  poppler-0.64.0.tar.xz  zsh-syntax-highlighting
+ tmp dir2-p dir2/dir3/dir4/dir5 #创建多层目录
+ tmp cd dir2/dir3/dir4/dir5
+ tmp pwd
/home/hls/tmp/dir2/dir3/dir4/dir5
+ dir5 mkdir -m 700 dir6 #创建带权限的目录
+ dir5 ll
总用量 4.0K
drwx----- 2 hls hls 4.0K 9月 28 20:34 dir6
+ tmp mkdir -vp src/{lib,bin,doc/{info,product},logs/{info,product},deploy/{info,product}} #创建多层目录
mkdir: 已创建目录 'src'
mkdir: 已创建目录 'src/lib'
mkdir: 已创建目录 'src/bin'
mkdir: 已创建目录 'src/doc'
mkdir: 已创建目录 'src/doc/info'
mkdir: 已创建目录 'src/doc/product'
mkdir: 已创建目录 'src/logs'
mkdir: 已创建目录 'src/logs/info'
mkdir: 已创建目录 'src/logs/product'
mkdir: 已创建目录 'src/deploy'
mkdir: 已创建目录 'src/deploy/info'
mkdir: 已创建目录 'src/deploy/product'
+ tmp tree src
src
├── bin
├── deploy
│   └── info
│       └── product
├── doc
│   ├── info
│   └── product
└── lib
    └── logs
        └── info
            └── product
11 directories, 0 files
```

新建 mkdir

新建 `mkdir` 命令是用来创建目录的

- `mkdir [选项]... 目录...`
- 使用 `-p` 参数可以创建多层目录
- 加入 `-m` 参数可以指定创建目录的权限

```
+ cmp -s /tmp/ dir1 #在当前目录下创建dir1目录
+ tmp ls
dir1  poppler-0.64.0  test
pdf2htmlEX  poppler-0.64.0.tar.xz  zsh-syntax-highlighting
+ tmp dir -p dir2/dir3/dir4/dir5 #创建多层目录
+ tmp cd dir2/dir3/dir4/dir5
+ dir5 pwd
/home/hls/tmp/dir2/dir3/dir4/dir5
+ dir5 mkdir -m 700 dir6 #创建带权限的目录
+ dir5 ll
总用量 4.0K
drwx----- 2 hls hls 4.0K 9月 28 20:34 dir6
+ tmp mkdir -vp src/{lib/bin/,}doc/{info,product},logs/{info,product},deploy/{info,product}
mkdir: 已创建目录 'src'
mkdir: 已创建目录 'src/lib/'
mkdir: 已创建目录 'src/bin/'
mkdir: 已创建目录 'src/doc/'
mkdir: 已创建目录 'src/doc/info'
mkdir: 已创建目录 'src/doc/product'
mkdir: 已创建目录 'src/logs'
mkdir: 已创建目录 'src/logs/info'
mkdir: 已创建目录 'src/logs/product'
mkdir: 已创建目录 'src/deploy'
mkdir: 已创建目录 'src/deploy/info'
mkdir: 已创建目录 'src/deploy/product'
+ tmp tree src
src
├── bin
├── deploy
│   └── info
│       └── product
├── doc
│   ├── info
│   │   └── product
│   └── lib
└── logs
    └── info
        └── product

11 directories, 0 files
```

新建 mkdir

新建 `mkdir` 命令是用来创建目录的

- `mkdir [选项]... 目录...`
 - 使用 **-p** 参数可以创建多层目录
 - 加入 **-m** 参数可以指定创建目录的权限
 - **-v** 参数可以输出命令执行结果

```
tmp mkdir dir1 #在当前目录下创建dir1目录
tmp ls
dir1      poppler-0.64.0      test
pdf2htmlEX poppler-0.64.0.tar.xz zsh-syntax-highlighting
tmp mkdir -p dir2/dir3/dir4/dir5 #创建多级目录
tmp cd dir2/dir3/dir4/dir5
dir5 pwd
/home/hls/tmp/dir2/dir3/dir4/dir5
tmp mkdir -m 700 dir6 #创建带权限的目录
tmp ll
总用量 4.0K
drwx----- 2 hls hls 4.0K 9月 28 20:34 dir6
tmp chmod -vp src/{lib,bin,doc/info_product},logs/{info_product},deploy/{info_product}
skkdir 已创建目录 'src'
skkdir 已创建目录 'src/lib'
skkdir 已创建目录 'src/bin'
skkdir 已创建目录 'src/doc'
skkdir 已创建目录 'src/doc/info'
skkdir 已创建目录 'src/doc/product'
skkdir 已创建目录 'src/logs'
skkdir 已创建目录 'src/logs/info'
skkdir 已创建目录 'src/logs/product'
kdir 已创建目录 'src/deploy'
kdir 已创建目录 'src/deploy/info'
kdir 已创建目录 'src/deploy/product'
tmp tree sec
sec
├── bin
│   └── deploy
│       ├── info
│       │   └── product
│       └── doc
│           ├── info
│           │   └── product
│           └── lib
│               └── logs
│                   ├── info
│                   │   └── product
11 directories, 0 files
```

新建 mkdir

新建 `mkdir` 命令是用来创建目录的

- `mkdir [选项]... 目录...`
 - 使用 **-p** 参数可以创建多层目录
 - 加入 **-m** 参数可以指定创建目录的权限
 - **-v** 参数可以输出命令执行结果
 - 使用大括号 `{}` 可以创建目录树结构

```
tmp mkdir dir1 #在当前目录下创建dir1目录
tmp ls
dir1      poppler-0.64.0      test
cdf2htmlEX poppler-0.64.0.tar.xz zsh-syntax-highlighting
+ tmp mkdir -p dir2/dir3/dir4/dir5 #创建多层目录
+ tmp cd dir2/dir3/dir4/dir5
+ dir5 pwd
/home/hls/tmp/dir2/dir3/dir4/dir5
+ dir5 mkdir -m 700 dir6 #创建带权限的目录
+ dir5 ll
总用量 4.0K
drwx----- 2 hls hls 4.0K 9月 28 20:34 dir6
+ tmp mkdir -vp src/lib/bin/doc/{info,product},logs/{info,product},deploy/{info,product} -- 则需要此命令
mkdir: 已创建目录 'src'
mkdir: 已创建目录 'src/lib'
mkdir: 已创建目录 'src/bin'
mkdir: 已创建目录 'src/doc'
mkdir: 已创建目录 'src/doc/info'
mkdir: 已创建目录 'src/doc/product'
mkdir: 已创建目录 'src/logs'
mkdir: 已创建目录 'src/logs/info'
mkdir: 已创建目录 'src/logs/product'
mkdir: 已创建目录 'src/deploy'
mkdir: 已创建目录 'src/deploy/info'
mkdir: 已创建目录 'src/deploy/product'
+ tmp tree src
src
├── bin
│   └── deploy
│       ├── info
│       │   └── product
│       └── doc
│           ├── info
│           │   └── product
│           └── lib
│               └── logs
│                   ├── info
│                   │   └── product
│                   └── product
└── logs
    └── info
        └── product

11 directories, 0 files
```

复制 cp

主要用来复制文件、文件夹等。主要用到的 `cp` 命令是 `cp` 或者 `cp -r`，其他的都不常用。

`cp 源文件 目标目录`

```
→ cp ls  
destDir file1  
→ cp cp file1 ./destDir  
→ cp cd destDir  
→ destDir ls  
file1  
→ destDir
```

复制文件

`cp -r 源目录 目标目录`

```
→ cp ls  
destDir file1 fromDir  
→ cp cp -r fromDir destDir  
→ cp cd destDir  
→ destDir ls  
file1 fromDir
```

复制文件夹

参数`i`, 覆盖既有文件之前先询问用户

```
→ cp cp -ir fromDir2 destDir  
cp: 是否覆盖 'destDir/fromDir2/file2'? y  
cp: 是否覆盖 'destDir/fromDir2/file1'? y  
→ cp |
```

覆盖选项

移动或改名 mv

移动或改名 mv命令与 linux cp 命令用法基本一致, 区别是 mv 使源文件状态改变, 而cp不改变源文件

mv基本用法: mv 原文件 改名文件

```
→ tmp ls  
destDir fromDir fromDir2 test.log  
→ tmp mv test.log test.txt  
→ tmp ll  
总用量 12K  
drwxr-xr-x 4 hls hls 4.0K 10月 2 09:29 destDir  
drwxr-xr-x 2 hls hls 4.0K 10月 2 09:01 fromDir  
drwxr-xr-x 2 hls hls 4.0K 10月 2 09:28 fromDir2  
-rw-r--r-- 1 hls hls 0 10月 2 10:31 test.txt
```

mv常用用法举例

```
→ tmp touch 1.log 2.log 3.log
```

```
→ tmp ll mv移动多个文件
```

```
总用量 12K  
-rw-r--r-- 1 hls hls 0 10月 2 12:33 1.log  
-rw-r--r-- 1 hls hls 0 10月 2 12:33 2.log  
-rw-r--r-- 1 hls hls 0 10月 2 12:33 3.log  
drwxr-xr-x 4 hls hls 4.0K 10月 2 12:31 destDir  
drwxr-xr-x 2 hls hls 4.0K 10月 2 09:01 fromDir  
drwxr-xr-x 2 hls hls 4.0K 10月 2 09:28 fromDir2  
+ tmp mv 1.log 2.log 3.log destDir  
+ tmp cd destDir  
+ destDir ll  
总用量 8.0K
```

mv文件移动: mv 原文件.. 目标目录

```
→ tmp mv test.txt destDir  
→ tmp ll  
总用量 12K  
drwxr-xr-x 4 hls hls 4.0K 10月 2 12:31 destDir  
drwxr-xr-x 2 hls hls 4.0K 10月 2 09:01 fromDir  
drwxr-xr-x 2 hls hls 4.0K 10月 2 09:28 fromDir2  
→ tmp cd destDir  
→ destDir ll  
总用量 8.0K  
-rw-r--r-- 1 hls hls 0 10月 2 08:59 file1  
-rw-r--r-- 1 hls hls 0 10月 2 08:59 file2  
drwxr-xr-x 2 hls hls 4.0K 10月 2 09:27 fromDir  
drwxr-xr-x 2 hls hls 4.0K 10月 2 09:31 fromDir2  
-rw-r--r-- 1 hls hls 0 10月 2 10:31 test.txt
```

```
→ tmp ls  
destDir fromDir fromDir2
```

```
→ tmp mv fromDir destDir  
→ tmp ll
```

```
mv移动目录: mv 原目录 目标目录
```

```
总用量 8.0K  
drwxr-xr-x 3 hls hls 4.0K 10月 2 12:36 destDir  
drwxr-xr-x 2 hls hls 4.0K 10月 2 09:28 fromDir2  
→ tmp cd destDir  
→ destDir ll  
总用量 4.0K  
-rw-r--r-- 1 hls hls 0 10月 2 12:33 1.log
```

删除 rm

递归删除r，强制删除f

```
→ destDir ll
总用量 4.0K
-rw-r--r-- 1 hls hls    0 10月  2 14:41 1.log
-rw-r--r-- 1 hls hls    0 10月  2 14:41 2.log
-rw-r--r-- 1 hls hls    0 10月  2 14:41 3.log
-rw-r--r-- 1 hls hls    0 10月  2 08:59 file2
drwxr-xr-x 2 hls hls 4.0K 10月  2 14:42 fromDir
-rw-r--r-- 1 hls hls    0 10月  2 10:31 test.txt
→ destDir rm 1.log
→ destDir rm -i 2.log
rm: 是否删除普通空文件 '2.log'? y
→ destDir rm -rf file1 fromDir
```

危险的命令

rm -rf 目录 此命令非常危险，谨慎

rm -rf /

rm -rf * .log



rm [选项]... [文件]...

— rm 用法

——rm 谨慎操作的命令

linux 磁盘管理

磁盘与目录的容量

`df`查看磁盘分区使用状况，`du`统计磁盘上的文件大小

磁盘挂载与卸载

linux 桌面一般会自动挂载，手工挂载`mount`及卸载命令`umount`

分区管理

磁盘分区：`fdisk`

配置启动挂载

`/etc/fstab` 可列出系统启动时要挂载的设备

linux 磁盘管理相关命令



df 查看磁盘分区使用状况

查看磁盘分区容量情况 df

检查磁盘空间占用情况。硬盘被占用了多少空间，还剩余空间等信息。

```
显示系统整体文件系统的磁盘使用情况
Last login: Mon Sep 24 15:12:14 2018 from 192.168.1.188
hls@gen8 ~ $ df
文件系统      1K-块    已用    可用  已用% 挂载点
udev          8107472     0  8107472   0% /dev
tmpfs         1625984  26060 1599924   2% /run
/dev/sdd1    460305340 365878820 71021320  84% /
tmpfs         8129904    600  8129304   1% /dev/shm
tmpfs          5120      0   5120   0% /run/lock
tmpfs         8129904     0  8129904   0% /sys/fs/cgroup
/dev/sda1    3845577736 2925439344 724771176  81% /home/hls/nas1
/dev/sdb1    3845577736 3281159400 369051120  90% /home/hls/bak1
cgtrfs        100       0    100   0% /run/cgmanager/fs
tmpfs         1625984     32  1625952   1% /run/user/1000
hls@gen8 ~ $
```

```
显示可读格式的数据
hls@gen8 ~ $ df -h
文件系统 容量  已用  可用  已用% 挂载点
udev      7,8G  0    7,8G  0% /dev
tmpfs     1,6G  26M 1,6G  2% /run
/dev/sdd1 439G 349G 692G 84% /
tmpfs     7,8G  600K 7,8G  1% /dev/shm
tmpfs     5,0M  0    5,0M  0% /run/lock
tmpfs     7,8G  0    7,8G  0% /sys/fs/cgroup
/dev/sda1 3,6T  2,8T 692G 81% /home/hls/nas1
/dev/sdb1 3,6T  3,1T 352G 90% /home/hls/bak1
cgtrfs    100K  0    100K 0% /run/cgmanager/fs
tmpfs     1,6G  32K 1,6G  1% /run/user/1000
hls@gen8 ~ $
```

```
指定文件夹, 查看该文件夹所在磁盘使用情况
hls@gen8 ~ $ df -h /home/hls
文件系统 容量  已用  可用  已用% 挂载点
/dev/sdd1 439G 349G 692G 84% /
hls@gen8 ~ $ df -h /home/hls/nas1/
hls@gen8 ~ $
```

df 用法

df [选项]... 文件

- **-l** 仅显示本地磁盘
- **-a** 显示所有文件系统
- **-h** 使用人类可读的格式
- **-H** 类似 h, 1000 进制
- **-t** 显示指定类型文件系统
- **-T** 显示文件系统的形式
- **-x** 不显示指定文件系统

du 统计磁盘上的文件大小

显示当前目录下面的子目录的
当前目录的总的大小

```
hls@gen8 ~/clonezilla $ du
7596 ./EFI/images
3088 ./EFI/boot/x86_64-efi
7868 ./EFI/boot
15468 ./EFI
460 ./utils/linux
252 ./utils/win32
8 ./utils/mbr
256 ./utils/win64
984 ./utils
252796 ./live
8 ./boot/grub
12 ./boot
8 ./disk
600 ./syslinux
269896 ./live

hls@gen8 ~/clonezilla $ du live
252796 live
hls@gen8 ~/clonezilla $ du ./EFI/images/efiboot.img
7592 ./EFI/images/efiboot.img
hls@gen8 ~/clonezilla $ du -s
269896 .
hls@gen8 ~/clonezilla $ du -h
7,5M ./EFI/images
3,1M ./EFI/boot/x86_64-efi
7,7M ./EFI/boot
16M ./EFI
460K ./utils/linux
252K ./utils/win32
8,0K ./utils/mbr
256K ./utils/win64
```

查看某目录大小
只显示总和的大小
易读格式

- 显示目录或文件大小

选项与参数

- a 列出所有的文件与目录容量
- h 使用人类可读的格式
- s 仅仅列出总量
- max-depth= <目录层数>

du 和 df 命令区别

- df 通过文件系统磁盘块进行计算
- du 直接统计各文件各目录的大小
- df 的返回值可能会因此大于 du

du 统计磁盘上的文件大小

显示当前目录下面的子目录的
当前目录的总的大小

```
hls@gen8 ~/clonezilla $ du
7596 ./EFI/images
3088 ./EFI/boot/x86_64-efi
7868 ./EFI/boot
15468 ./EFI
460 ./utils/linux
252 ./utils/win32
8 ./utils/mbr
256 ./utils/win64
984 ./utils
252796 ./live
8 ./boot/grub
12 ./boot
8 ./disk
600 ./syslinux
269896 ./.

hls@gen8 ~/clonezilla $ du live
252796 live
hls@gen8 ~/clonezilla $ du ./EFI/images/efiboot.img
7592 ./EFI/images/efiboot.img
hls@gen8 ~/clonezilla $ du -s
269896 .
hls@gen8 ~/clonezilla $ du -h
7,5M ./EFI/images
3,1M ./EFI/boot/x86_64-efi
7,7M ./EFI/boot
16M ./EFI
460K ./utils/linux
252K ./utils/win32
8,0K ./utils/mbr
256K ./utils/win64
```

查看某目录大
只显示总和的大小
易读格式

- 显示目录或文件大小
- 用法: **du [选项].. 文件**

选项与参数

- **-a** 列出所有的文件与目录容量
- **-h** 使用人类可读的格式
- **-s** 仅仅列出总量
- **-max-depth= < 目录层数 >**

du 和 df 命令区别

- **df** 通过文件系统磁盘块进行计算
- **du** 直接统计各文件各目录的大小
- **df** 的返回值可能会因此大于**du**

手工挂载 mount

通常用于挂载 Linux 系统外的文件系统。

要作为挂载点的目录，应该都是空目录。

mount 设备

- **mount /dev/sdd2 ~/tmp**

mount 虚拟设备

手工挂载 mount

通常用于挂载 Linux 系统外的文件系统。

要作为挂载点的目录，应该都是空目录。

mount 设备

- **mount /dev/sdd2 ~/tmp**
- **mount -o ro /dev/sdd2 ~/tmp**

mount 虚拟设备

手工挂载 mount

通常用于挂载 Linux 系统外的文件系统。

要作为挂载点的目录，应该都是空目录。

mount 设备

- **mount** /dev/sdd2 ~/tmp
- **mount** -o ro /dev/sdd2 ~/tmp

mount 虚拟设备

- **mount** -o loop live.iso ~/cdrom

卸载命令 umount

卸载命令 umount

将已挂载的文件系统给它卸除！卸载之后，可使用 **df** 或 **mount -l** 确认。

```
hls@gen8 ~/clonezilla $ umount --help

Usage:
umount [-hV]
umount -a [options]
umount [options] <source> | <directory>

Unmount filesystems.

选项:
-a, --all           unmount all filesystems
-A, --all-targets   unmount all mountpoints for the given device in the
                   current namespace
-c, --no-canonicalize don't canonicalize paths
-d, --detach-loop   if mounted loop device, also free this loop device
--fake              dry run; skip the umount(2) syscall
-f, --force          force umount (in case of an unreachable NFS system)
-i, --internal-only don't call the umount.<type> helpers
-n, --no-mtab        don't write to /etc/mtab
-l, --lazy           detach the filesystem now, clean up things later
-o, --test-opts <list> limit the set of filesystems (use with -a)
-R, --recursive     recursively umount a target with all its children
-r, --read-only      in case unmounting fails, try to remount read-only
-t, --types <list>    limit the set of filesystem types
-v, --verbose        say what is being done

-h, --help           display this help and exit
-V, --version         output version information and exit

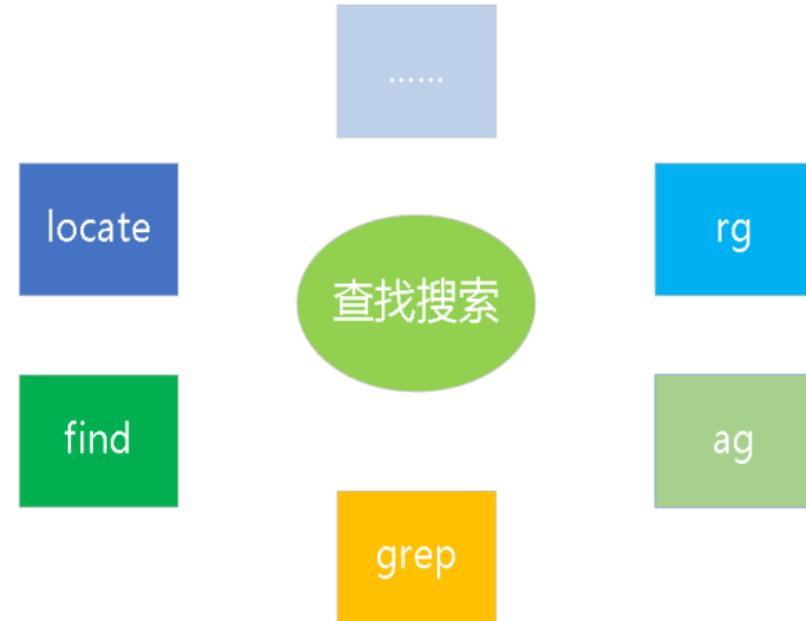
For more details see umount(8).
hls@gen8 ~/clonezilla $ umount /dev/sdd2
```

umount 使用方法

- **umount [options]**
<source> | <directory>

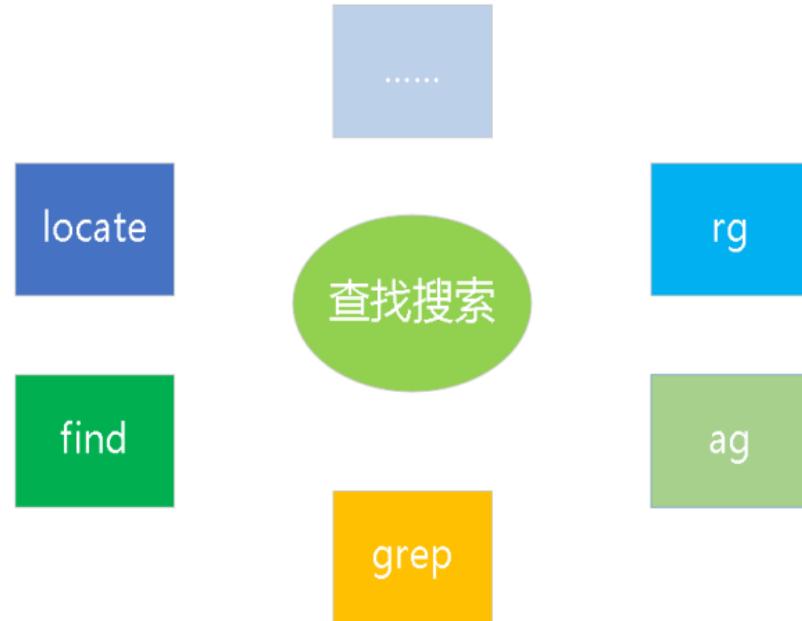
linux 各种搜索命令 掌握好搜索命令，能够大大提高 效率。

- 默认内置: find



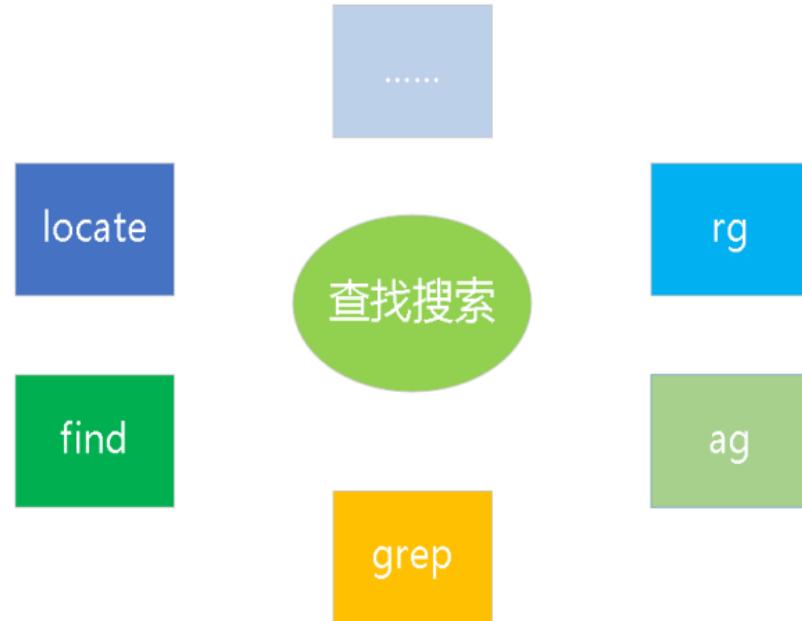
linux 各种搜索命令 掌握好搜索命令，能够大大提高 效率。

- 默认内置: find
- grep、 ag、 rg 类似



linux 各种搜索命令 掌握好搜索命令，能够大大提高 效率。

- 默认内置: find
- grep、 ag、 rg 类似
- 效率最高: rg(Ripgrep)



locate 通过名字来查找文件

locate 和 find 的不同点

locate 只能通过名字查找文件，是“find -name”的另一种写法，但速度更快。

locate用法

- 用法： locate [选项] [文件名]
- r 使用正规运算式做寻找的条件
- d 指定资料库的路径
- 查找时不区分大小写，使用 -i
- locate 的查找并不是实时的
- locate 手工更新命令 locate -u

```
hls@gen8 ~ $ locate hls-autogit.sh
/home/hls/hls-autogit.sh
/home/hls/bak1/20170111-fullback/Z01-备份 1/
h
hls@gen8 ~ $ locate -r hls-autogit.*
/home/hls/hls-autogit.sh
/home/hls/bak1/20170111-fullback/Z01-备份 1/
h
hls@gen8 ~ $ locate -ir Hls-autogit.*
```

find 查找文件的复杂方式

```
→ - find ./.emacs.d -name init.el 按照文件名查找init.el  
/home/hls/.emacs.d/init.el  
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录  
→ .emacs.d git:(master) ✘ find .. -name init.el  
./init.el  
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
./.hls-el/emmet-mode/src/init.el  
→ .emacs.d git:(master) ✘ find .. -name 'init?el'  
./init.el  
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则  
→ .emacs.d git:(master) ✘ find .. -amin -10  
./.git  
./.hls-cache/.emacs.keyfreq  
atime\ctime\mtime 文件访问\属性修改  
\内容修改时间, 按天数  
amin\cmin\mmin 文件访问\属性修改  
\内容修改时间, 按分钟  
→ hls-backup git:(master) ✘ find .. -empty 查找空文件  
.linux-bk/global/cscope/helm/.git/branches  
.linux-bk/global/cscope/helm/.git/refs/tags  
hls-backup git:(master) ✘ find .. -size +1M 根据文件大小查找  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
.linux-bk/.stardict/dic/langdao-ec-gb.dict.dz  
.linux-bk/.stardict/dic/oxford-gb.dict.dz and 或 or 表达式多条  
.linux-bk/.stardict/dic/langdao-ec-gb.idx  
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -  
-all {} ; -exec 查找后进行后处理, {}代表列表  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: find [范围] [条件] [处理]

find 查找文件的复杂方式

```
→ - find ./.emacs.d -name init.el 按照文件名查找init.el  
/home/hls/.emacs.d/init.el  
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录  
→ .emacs.d git:(master) ✘ find .. -name init.el  
./init.el  
./hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
./hls-el/emmet-mode/src/init.el  
→ .emacs.d git:(master) ✘ find .. -name 'init.el'  
./init.el  
./hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则  
→ .emacs.d git:(master) ✘ find .. -amin -10  
./.git  
./hls-cache/.emacs.keyfreq  
atime\ctime\mtime 文件访问\属性修改  
\内容修改时间, 按天数  
amin\cmin\mmin 文件访问\属性修改  
\内容修改时间, 按分钟  
→ hls-backup git:(master) ✘ find .. -empty 查找空文件  
.linux-bk/global/cscope/helm/.git/branches  
.linux-bk/global/cscope/helm/.git/refs/tags  
hls-backup git:(master) ✘ find .. -size +1M 根据文件大小查找  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
.linux-bk/.stardict/dic/langdao-ec-gb.dict.dz  
.linux-bk/.stardict/dic/oxford-gb.dict.dz and 或 or 表达式多条  
.linux-bk/.stardict/dic/langdao-ec-gb.idx  
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -  
-all {} ; -exec 查找后进行后处理, {}代表列表  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: find [范围] [条件] [处理]
- filesize:

find 查找文件的复杂方式

```
→ ~ find ./.emacs.d -name init.el 按照文件名查找init.el  
/home/hls/.emacs.d/init.el  
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录  
→ .emacs.d git:(master) ✘ find .. -name init.el  
./init.el  
.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
.hls-el/emmet-mode/src/init.el  
→ .emacs.d git:(master) ✘ find .. -name 'init.el'  
./init.el  
.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则  
→ .emacs.d git:(master) ✘ find .. -amin -10  
./.git  
./hls-cache/.emacs.keyfreq  
atime\ctime\mtime 文件访问\属性修改  
\内容修改时间, 按天数  
amin\cmin\mmin 文件访问\属性修改  
\内容修改时间, 按分钟  
→ hls-backup git:(master) ✘ find .. -empty 查找空文件  
.linux-bk/global/cscope/helm/.git/branches  
.linux-bk/global/cscope/helm/.git/refs/tags  
hls-backup git:(master) ✘ find .. -size +1M 根据文件大小查找  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
.linux-bk/.stardict/dic/langdao-ec-gb.dict.dz  
.linux-bk/.stardict/dic/oxford-gb.dict.dz and 或 or 表达式多条  
.linux-bk/.stardict/dic/langdao-ec-gb.idx  
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -  
-all {} ; -exec 查找后进行后处理, {}代表列表  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: `find [范围] [条件] [处理]`
- filesize:**
 - 50k 小于 50kb 的文件

find 查找文件的复杂方式

```
→ - find ./.emacs.d -name init.el 按照文件名查找init.el  
/home/hls/.emacs.d/init.el  
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录  
→ .emacs.d git:(master) X find . -name init.el  
.init.el  
.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
.hls-el/emmet-mode/src/init.el  
→ .emacs.d git:(master) X find . -name 'init.el'  
.init.el  
.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则  
→ .emacs.d git:(master) X find . -amin -10  
.git  
.hls-cache/.emacs.keyfreq
```

atime\ctime\mtime 文件访问\属性修改

\内容修改时间, 按天数

amin\cmin\mmin 文件访问\属性修改

\内容修改时间, 按分钟

```
→ hls-backup git:(master) X find . -empty 查找空文件  
.linux-bk/global/cscope/helm/.git/branches  
.linux-bk/global/cscope/helm/.git/refs/tags  
hls-backup git:(master) X find . -size +1M 根据文件大小查找  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
→ hls-backup git:(master) X find . -size +1M -and -mtime +120  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
.linux-bk/.stardict/dic/langdao-ec-gb.dict.dz and 或 or 表达式多条  
.linux-bk/.stardict/dic/oxford-gb.dict.dz  
→ hls-backup git:(master) X find . -type f -size +1M -and -mtime +120 -exec  
-all {} ; -exec查找后进行后处理, {}代表列表  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: find [范围] [条件] [处理]

• filesize:

- 50k 小于 50kb 的文件
- 50k 等于 50kb 的文件

find 查找文件的复杂方式

```
→ ~ find ./.emacs.d -name init.el 按照文件名查找init.el  
/home/hls/.emacs.d/init.el  
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录  
→ .emacs.d git:(master) ✘ find .. -name init.el  
./init.el  
.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
.hls-el/emmet-mode/src/init.el  
→ .emacs.d git:(master) ✘ find .. -name 'init.el'  
./init.el  
.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则  
→ .emacs.d git:(master) ✘ find .. -amin -10  
./.git  
.hls-cache/.emacs.keyfreq  
ctime\ctime\mtime 文件访问\属性修改  
\内容修改时间,按天数  
amin\cmin\mmin 文件访问\属性修改  
\内容修改时间,按分钟
```

```
→ hls-backup git:(master) ✘ find .. -empty 查找空文件  
.linux-bk/global/cscope/helm/.git/branches  
.linux-bk/global/cscope/helm/.git/refs/tags  
hls-backup git:(master) ✘ find .. -size +1M 根据文件大小查找  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
.linux-bk/.stardict/dic/langdao-ec-gb.dict.dz  
.linux-bk/.stardict/dic/oxford-gb.dict.dz and 或 or 表达式多条  
.linux-bk/.stardict/dic/langdao-ec-gb.idx  
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -  
-all {} ; -exec 查找后进行后处理, {}代表列表  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: `find [范围] [条件] [处理]`

• filesize:

- 50k 小于 50kb 的文件
- 50k 等于 50kb 的文件
- +50k 大于 50kb 的文件

find 查找文件的复杂方式

```
→ find . -name init.el 按照文件名查找init.el  
/home/hls/.emacs.d/init.el  
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录  
→ .emacs.d git:(master) X find . -name init.el  
.init.el  
.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
.hls-el/emmet-mode/src/init.el  
→ .emacs.d git:(master) X find . -name 'init.el'  
.init.el  
.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则  
→ .emacs.d git:(master) X find . -amin -10  
.git  
.hls-cache/.emacs.keyfreq  
atime\ctime\mtime 文件访问\属性修改  
\内容修改时间,按天数  
amin\cmin\mmin 文件访问\属性修改  
\内容修改时间,按分钟
```

```
→ hls-backup git:(master) X find . -empty 查找空文件  
.linux-bk/global/cscope/helm/.git/branches  
.linux-bk/global/cscope/helm/.git/refs/tags  
hls-backup git:(master) X find . -size +1M 根据文件大小查找  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
→ hls-backup git:(master) X find . -size +1M -and -mtime +120  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
.linux-bk/.stardict/dic/langdao-ec-gb.dict.dz and 或 or 表达式多条  
.linux-bk/.stardict/dic/oxford-gb.dict.dz  
→ hls-backup git:(master) X find . -type f -size +1M -and -mtime +120 -exec  
-all {} ; -exec查找后进行后处理, {}代表列表  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: `find [范围] [条件] [处理]`
- filesize:**
 - 50k 小于 50kb 的文件
 - 50k 等于 50kb 的文件
 - +50k 大于 50kb 的文件
- 避免大范围搜索, 速度慢

find 查找文件的复杂方式

```
→ ~ find ../../emacs.d -name init.el 按照文件名查找init.el
/home/hls/.emacs.d/init.el
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录
→ .emacs.d git:(master) ✘ find .. -name init.el
./init.el
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
./.hls-el/emmet-mode/src/init.el
→ .emacs.d git:(master) ✘ find .. -name 'init.el'
./init.el
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则
→ .emacs.d git:(master) ✘ find .. -amin -10
./.git
./.hls-cache/.emacs.keyfreq
atime\ctime\mtime 文件访问\属性修改
\内容修改时间,按天数
amin\cmin\mmin 文件访问\属性修改
\内容修改时间,按分钟
\根据文件大小查找
→ hls-backup git:(master) ✘ find .. -empty 查找空文件
./linux-bk/global/cscope/helm/.git/branches
./linux-bk/global/cscope/helm/.git/refs/tags
→ hls-backup git:(master) ✘ find .. -size +1M
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
./linux-bk/.stardict/dic/langdao-ec-gb.dict.dz
./linux-bk/.stardict/dic/oxford-gb.dict.dz and 或 or 表达式多条
./linux-bk/.stardict/dic/langdao-ec-gb.idx
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -exec
-all {} ; -exec 找后进行后处理, {} 代表列表
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: `find [范围] [条件] [处理]`
- filesize:**
 - 50k 小于 50kb 的文件
 - 50k 等于 50kb 的文件
 - +50k 大于 50kb 的文件
- 避免大范围搜索, 速度慢
- find 较复杂, 可参考 help

find 查找文件的复杂方式

```
→ ~ find ~/.emacs.d -name init.el 按照文件名查找init.el
/home/hls/.emacs.d/init.el
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录
→ .emacs.d git:(master) ✘ find .. -name init.el
./init.el
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
./.hls-el/emmet-mode/src/init.el
→ .emacs.d git:(master) ✘ find .. -name 'init.el'
./init.el
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则
→ .emacs.d git:(master) ✘ find .. -amin -10
./.git
./.hls-cache/.emacs.keyfreq
atime\ctime\mtime 文件访问\属性修改
\内容修改时间,按天数
amin\cmin\mmin 文件访问\属性修改
\内容修改时间,按分钟
\根据文件大小查找
→ hls-backup git:(master) ✘ find .. -empty 查找空文件
./linux-bk/global/cscope/helm/.git/branches
./linux-bk/global/cscope/helm/.git/refs/tags
→ hls-backup git:(master) ✘ find .. -size +1M
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
./linux-bk/.stardict/dic/langdao-ec-gb.dict.dz
./linux-bk/.stardict/dic/oxford-gb.dict.dz and 或 or 表达式多条
./linux-bk/.stardict/dic/langdao-ec-gb.idx
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -exec
-all {} ; -exec 找后进行后处理, {} 代表列表
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: **find [范围] [条件] [处理]**
- filesize:**
 - 50k 小于 50kb 的文件
 - 50k 等于 50kb 的文件
 - +50k 大于 50kb 的文件
- 避免大范围搜索, 速度慢
- find** 较复杂, 可参考 **help**
- find** 文件类型

find 查找文件的复杂方式

```
→ ~ find ~/.emacs.d -name init.el 按照文件名查找init.el
/home/hls/.emacs.d/init.el
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录
→ .emacs.d git:(master) ✘ find .. -name init.el
./init.el
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
./.hls-el/emmet-mode/src/init.el
→ .emacs.d git:(master) ✘ find .. -name 'init.el'
./init.el
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则
→ .emacs.d git:(master) ✘ find .. -amin -10
./.git
./.hls-cache/.emacs.keyfreq
atime\ctime\mtime 文件访问\属性修改
\内容修改时间,按天数
amin\cmin\mmin 文件访问\属性修改
\内容修改时间,按分钟
\根据文件大小查找
→ hls-backup git:(master) ✘ find .. -empty 查找空文件
./linux-bk/global/cscope/helm/.git/branches
./linux-bk/global/cscope/helm/.git/refs/tags
→ hls-backup git:(master) ✘ find .. -size +1M
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1 根据文件大小查找
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
./linux-bk/.stardict/dic/langdao-ec-gb.dict.dz and 或 or 表达式多条
./linux-bk/.stardict/dic/oxford-gb.dict.dz
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -exec
-all {} ; -exec查找后进行后处理, {}代表列表
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: **find [范围] [条件] [处理]**
- filesize:**
 - 50k 小于 50kb 的文件
 - 50k 等于 50kb 的文件
 - +50k 大于 50kb 的文件
- 避免大范围搜索, 速度慢
- find** 较复杂, 可参考 **help**
- find** 文件类型
 - type f** 文件

find 查找文件的复杂方式

```
→ ~ find ~/.emacs.d -name init.el 按照文件名查找init.el
/home/hls/.emacs.d/init.el
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录
→ .emacs.d git:(master) ✘ find .. -name init.el
./init.el
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
./.hls-el/emmet-mode/src/init.el
→ .emacs.d git:(master) ✘ find .. -name 'init.el'
./init.el
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则
→ .emacs.d git:(master) ✘ find .. -amin -10
./.git
./.hls-cache/.emacs.keyfreq
atime\ctime\mtime 文件访问\属性修改
\内容修改时间,按天数
amin\cmin\mmin 文件访问\属性修改
\内容修改时间,按分钟
\根据文件大小查找
→ hls-backup git:(master) ✘ find .. -empty 查找空文件
./linux-bk/global/cscope/helm/.git/branches
./linux-bk/global/cscope/helm/.git/refs/tags
→ hls-backup git:(master) ✘ find .. -size +1M
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120 根据文件大小查找
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
./linux-bk/.stardict/dic/langdao-ec-gb.dict.dz
./linux-bk/.stardict/dic/oxford-gb.dict.dz and 或 or 表达式多条
./linux-bk/.stardict/dic/langdao-ec-gb.idx
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -exec
-all {} ; -exec 查找后进行后处理, {}代表列表
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

● 用法: **find [范围] [条件] [处理]**

● **filesize:**

- -50k 小于 50kb 的文件
- 50k 等于 50kb 的文件
- +50k 大于 50kb 的文件

● 避免大范围搜索, 速度慢

● **find** 较复杂, 可参考 **help**

● **find** 文件类型

- **-type f** 文件
- **-type d** 目录

find 查找文件的复杂方式

```
→ find ./.emacs.d -name init.el 按照文件名查找init.el
/home/hls/.emacs.d/init.el
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录
→ .emacs.d git:(master) ✘ find .. -name init.el
./init.el
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
./.hls-el/emmet-mode/src/init.el
→ .emacs.d git:(master) ✘ find .. -name 'init.el'
./init.el
./.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则
→ .emacs.d git:(master) ✘ find .. -amin -10
./.git
./.hls-cache/.emacs.keyfreq
atime\ctime\mtime 文件访问\属性修改
\内容修改时间,按天数
amin\cmin\mmin 文件访问\属性修改
\内容修改时间,按分钟
\根据文件大小查找
→ hls-backup git:(master) ✘ find .. -empty 查找空文件
./linux-bk/global/cscope/helm/.git/branches
./linux-bk/global/cscope/helm/.git/refs/tags
→ hls-backup git:(master) ✘ find .. -size +1M
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
./linux-bk/.stardict/dic/langdao-ec-gb.dict.dz and 或 or 表达式多条
./linux-bk/.stardict/dic/oxford-gb.dict.dz
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -exec
查找后进行后处理, {}代表列表
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: **find [范围] [条件] [处理]**
- filesize:**
 - 50k 小于 50kb 的文件
 - 50k 等于 50kb 的文件
 - +50k 大于 50kb 的文件
- 避免大范围搜索, 速度慢
- find 较复杂, 可参考 help
- find 文件类型**
 - type f** 文件
 - type d** 目录
- find 命令的逻辑操作符**

find 查找文件的复杂方式

```
→ find ../../emacs.d -name init.el 按照文件名查找init.el
/home/hls/.emacs.d/init.el
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录
→ .emacs.d git:(master) ✘ find .. -name init.el
./init.el
./hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el
./hls-el/emmet-mode/src/init.el
→ .emacs.d git:(master) ✘ find .. -name 'init.el'
./init.el
./hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则
→ .emacs.d git:(master) ✘ find .. -amin -10
./.git
./hls-cache/.emacs.keyfreq
atime\ctime\mtime 文件访问\属性修改
\内容修改时间,按天数
amin\cmin\mmin 文件访问\属性修改
\内容修改时间,按分钟
\根据文件大小查找
→ hls-backup git:(master) ✘ find .. -empty 查找空文件
./linux-bk/global/cscope/helm/.git/branches
./linux-bk/global/cscope/helm/.git/refs/tags
→ hls-backup git:(master) ✘ find .. -size +1M
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120 根据文件大小查找
./linux-bk/phpunit.phar.2
./linux-bk/phpunit.phar.1
./linux-bk/.stardict/dic/langdao-ec-gb.dict.dz and 或 or 表达式多条
./linux-bk/.stardict/dic/oxford-gb.dict.dz
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -exec
-all {} ; -exec查找后进行后处理, {}代表列表
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

- 用法: **find [范围] [条件] [处理]**
- filesize:**
 - 50k 小于 50kb 的文件
 - 50k 等于 50kb 的文件
 - +50k 大于 50kb 的文件
- 避免大范围搜索, 速度慢
- find** 较复杂, 可参考 **help**
- find** 文件类型
 - type f** 文件
 - type d** 目录
- find** 命令的逻辑操作符
 - and** 并

find 查找文件的复杂方式

```
→ find ./.emacs.d -name init.el 按照文件名查找init.el  
/home/hls/.emacs.d/init.el  
/home/hls/.emacs.d/hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
/home/hls/.emacs.d/hls-el/emmet-mode/src/init.el 要查找的目录  
→ .emacs.d git:(master) ✘ find .. -name init.el  
./init.el  
.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el  
.hls-el/emmet-mode/src/init.el  
→ .emacs.d git:(master) ✘ find .. -name 'init.el'  
./init.el  
.hls-el/COSCUP2013_org-mode/examples/dot-emacs/init.el 正则  
→ .emacs.d git:(master) ✘ find .. -amin -10  
./.git  
.hls-cache/.emacs.keyfreq  
  
atime\ctime\mtime 文件访问\属性修改  
\内容修改时间, 按天数  
amin\cmin\mmin 文件访问\属性修改  
\内容修改时间, 按分钟  
→ hls-backup git:(master) ✘ find .. -empty 查找空文件  
.linux-bk/global/cscope/helm/.git/branches  
.linux-bk/global/cscope/helm/.git/refs/tags  
hls-backup git:(master) ✘ find .. -size +1M 根据文件大小查找  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
→ hls-backup git:(master) ✘ find .. -size +1M -and -mtime +120  
.linux-bk/phpunit.phar.2  
.linux-bk/phpunit.phar.1  
.linux-bk/.stardict/dic/langdao-ec-gb.dict.dz  
.linux-bk/.stardict/dic/oxford-gb.dict.dz and 或 or 表达式多条  
.linux-bk/.stardict/dic/langdao-ec-gb.idx  
→ hls-backup git:(master) ✘ find .. -type f -size +1M -and -mtime +120 -  
-all {} ; -exec 查找后进行后处理, {}代表列表  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.2  
-rwxrwxr-x 1 hls hls 3144819 4月 28 20:30 ./linux-bk/phpunit.phar.1
```

● 用法: **find [范围] [条件] [处理]**

● **filesize:**

- -50k 小于 50kb 的文件
- 50k 等于 50kb 的文件
- +50k 大于 50kb 的文件

● 避免大范围搜索, 速度慢

● **find** 较复杂, 可参考 **help**

● **find** 文件类型

- **-type f** 文件
- **-type d** 目录

● **find** 命令的逻辑操作符

- **-and** 并
- **-or** 或

grep 强大的文本搜索工具

- 全面搜索正则并把行打印出来

grep 命令与 find 命令的区别

- grep命令：搜索符字符串
- find命令：搜索文件名

grep 变种

- grep -E 相当于 egrep
- grep -F 相当于 fgrep
- grep -r 相当于 rgrep

```
→ hls-backup git:(master) ✘ grep hls /etc/passwd 搜索包括hls字符的行
hls:x:1000:1000:hls,,,:/home/hls:/usr/bin/zsh
→ hls-backup git:(master) ✘ sudo grep hls /etc/passwd /etc/shadow 搜索多文件
/etc/passwd:hls:x:1000:1000:hls,,,:/home/hls:/usr/bin/zsh
/etc/shadow:hls:$6$GhF10Fd$6TKHHRgmaAal2FgiuM2zg4hZumVeJ7SNPv0PP0DzzkM1DQ/wSV
→ hls-backup git:(master) ✘ sudo grep -l hls /etc/passwd /etc/shadow
/etc/passwd
/etc/shadow 使用 -l参数列出包含指定模式的文件的文件名
→ hls-backup git:(master) ✘ sudo grep -n hls /etc/passwd 显示行号
40 hls:x:1000:1000:hls,,,:/home/hls:/usr/bin/zsh
→ hls-backup git:(master) ✘ sudo grep -v hls /etc/passwd
root:x:0:0:root:/root:/bin/zsh 反向匹配
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
→ hls-backup git:(master) ✘ sudo grep ^root /etc/passwd
root:x:0:0:root:/root:/bin/zsh 搜索root开头的字符
→ hls-backup git:(master) ✘ grep -e "hls" -e "root" /etc/passwd
root:x:0:0:root:/root:/bin/zsh 使用 -e 参数查找多个模式
hls:x:1000:1000:hls,,,:/home/hls:/usr/bin/zsh
→ hls-backup git:(master) ✘ sudo grep -r hls /etc
/etc/subuid:hls:100000:65536 递归目录，等同rgrep
/etc/hostname:hls
→ hls-backup git:(master) ✘ grep -e "hls" -e "root" /etc/passwd
root:x:0:0:root:/root:/bin/zsh 使用 -e 参数查找多个模式
hls:x:1000:1000:hls,,,:/home/hls:/usr/bin/zsh
将需要搜索的字符串写入到一个单独文件grep_pattern中
→ hls-backup git:(master) ✘ grep -f grep_pattern /etc/passwd
```

grep 强大的文本搜索工具

- 全面搜索正则并把行打印出来
- 用法: grep [选项].. PATTERN [FILE]..

grep 命令与 find 命令的区别

- grep命令: 搜索符字符串
- find命令: 搜索文件名

grep 变种

- grep -E 相当于 egrep
- grep -F 相当于 fgrep
- grep -r 相当于 rgrep

```
→ hls-backup git:(master) ✘ grep hls /etc/passwd 搜索包括hls字符的行
hls:x:1000:1000:hls,,,:/home/hls:/usr/bin/zsh
→ hls-backup git:(master) ✘ sudo grep hls /etc/passwd /etc/shadow 搜索多文件
/etc/passwd:hls:x:1000:1000:hls,,,:/home/hls:/usr/bin/zsh
/etc/shadow:hls:$6$GhF10Fd$6TKHHRgmaAal2FgiuM2zg4hZumVeJ7SNPv0PP0DzzkM1DQ/wSV
→ hls-backup git:(master) ✘ sudo grep -l hls /etc/passwd /etc/shadow 使用-l参数列出包含指定模式的文件的文件名
/etc/passwd
/etc/shadow
→ hls-backup git:(master) ✘ sudo grep -n hls /etc/passwd 显示行号
40:hls:x:1000:1000:hls,,,:/home/hls:/usr/bin/zsh
→ hls-backup git:(master) ✘ sudo grep -v hls /etc/passwd 反向匹配
root:x:0:0:root:/root:/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
→ hls-backup git:(master) ✘ sudo grep ^root /etc/passwd
root:x:0:0:root:/root:/bin/zsh 搜索root开头的字符
→ hls-backup git:(master) ✘ grep -e "hls" -e "root" /etc/passwd
root:x:0:0:root:/root:/bin/zsh 使用-e参数查找多个模式
hls:x:1000:1000:hls,,,:/home/hls:/usr/bin/zsh
→ hls-backup git:(master) ✘ sudo grep -r hls /etc
/etc/subuid:hls:100000:65536 递归目录, 等同rgrep
/etc/hostname:hls
→ hls-backup git:(master) ✘ grep -e "hls" -e "root" /etc/passwd
root:x:0:0:root:/root:/bin/zsh 使用-e参数查找多个模式
hls:x:1000:1000:hls,,,:/home/hls:/usr/bin/zsh
将需要搜索的字符串写入到一个单独文件grep_pattern中
→ hls-backup git:(master) ✘ grep -f grep_pattern /etc/passwd
```

比 grep 更快的 ag 命令

- 速度对比 ag>ack>grep
- 搜索的总数据量较大优势明显
- ag 一般单独安装

用法和 grep 基本一致

默认颜色高亮

默认 递归搜索

默认忽略某些文件

```
→ .emacs.d git:(master) ✘ grep -r "projectile-bookmarks.eld"  
hls-el/projectile/projectile.el: (expand-file-name "projectile-bookmarks.eld"  
→ .emacs.d git:(master) ✘ ag "projectile-bookmarks.eld"  
hls-el/projectile/projectile.el  
⑨8 (expand-file-name 'projectile-bookmarks.el)
```

常用参数

-g 文件名匹配

-php 只搜索 php 文件

-ignore-dir 忽略目录

-l 只列出文件名

-i 忽略大小写

Usage: ag [FILE-TYPE] [OPTIONS] PATTERN [PATH]

Recursively search for PATTERN in PATH.



The silver searcher

#最常见的用法
ag "字符串"

指定目录搜索
ag "字符串" dir

只搜索php类型的文件
ag -G ".+\\.php" "字符串" dir

反向匹配
ag -v

忽略目录
ag --ignore-dir logdir --php "字符串" /www

只打印满足的文件名
ag -l

比 ag 更快的 rg 命令

rg 一些特性

- 递归搜索、高亮匹配、自动忽略
- 具备 grep、ag 大部分功能

使用方法



- `rg [OPTIONS] PATTERN [PATH ...]`
- `rg [OPTIONS] [-e PATTERN .] [-f FILE .] [PATH .]`
- `rg [OPTIONS] --files [PATH ...]`
- `rg [OPTIONS] --type-list`

- 更快的搜索速度

比 ag 更快的 rg 命令

rg 一些特性

- 递归搜索、高亮匹配、自动忽略
- 具备 grep、ag 大部分功能

使用方法



- `rg [OPTIONS] PATTERN [PATH ...]`
- `rg [OPTIONS] [-e PATTERN .] [-f FILE .] [PATH .]`
- `rg [OPTIONS] --files [PATH ...]`
- `rg [OPTIONS] --type-list`

- 更快的搜索速度
- 比其它类似工具在搜索速度上快上 N 倍

比 ag 更快的 rg 命令

rg 一些特性

- 递归搜索、高亮匹配、自动忽略
- 具备 grep、ag 大部分功能

使用方法



- `rg [OPTIONS] PATTERN [PATH ...]`
- `rg [OPTIONS] [-e PATTERN .] [-f FILE .] [PATH .]`
- `rg [OPTIONS] --files [PATH ...]`
- `rg [OPTIONS] --type-list`

- 更快的搜索速度
- 比其它类似工具在搜索速度上快上 N 倍
- 新生搜索工具

归档和备份

常见的归档文件： *.tar, *.tar.gz, *.tgz, *.gz, *.Z, *.bz2

最大好处就是压缩过的文件容量变小了。

压缩打包

- tar



解压文件

归档和备份

常见的归档文件： *.tar, *.tar.gz, *.tgz, *.gz, *.Z, *.bz2

最大好处就是压缩过的文件容量变小了。

压缩打包

- tar
- rar



解压文件

归档和备份

常见的归档文件： *.tar, *.tar.gz, *.tgz, *.gz, *.Z, *.bz2

最大好处就是压缩过的文件容量变小了。

压缩打包

- tar
- rar
- zip



解压文件

归档和备份

常见的归档文件： *.tar, *.tar.gz, *.tgz, *.gz, *.Z, *.bz2

最大好处就是压缩过的文件容量变小了。

压缩打包

- tar
- rar
- zip



解压文件

- tar

归档和备份

常见的归档文件： *.tar, *.tar.gz, *.tgz, *.gz, *.Z, *.bz2

最大好处就是压缩过的文件容量变小了。

压缩打包

- tar
- rar
- zip



解压文件

- tar
- rar

归档和备份

常见的归档文件： *.tar, *.tar.gz, *.tgz, *.gz, *.Z, *.bz2

最大好处就是压缩过的文件容量变小了。

压缩打包

- tar
- rar
- zip



解压文件

- tar
- rar
- unzip

tar 命令

- 打包 : `tar -zcvf log.tar.gz demo.log`



tar 命令

- 打包 : `tar -zcvf log.tar.gz demo.log`
- 打包压缩 : `tar -jcv -f demo.tar.bz2`



tar 命令

- 打包 : `tar -zcvf log.tar.gz demo.log`
- 打包压缩 : `tar -jcv -f demo.tar.bz2`
- 解压 bz2 : `tar -jxv -f demo.tar.bz2 -C dir`



tar 命令

- 打包 : `tar -zcvf log.tar.gz demo.log`
- 打包压缩 : `tar -jcv -f demo.tar.bz2`
- 解压 bz2 : `tar -jxv -f demo.tar.bz2 -C dir`
- 解压 gz : `tar -zxvf demo.tar.gz`



tar 命令



- 打包 : **tar -zcvf log.tar.gz demo.log**
- 打包压缩 : **tar -jcv -f demo.tar.bz2**
- 解压 bz2 : **tar -jxv -f demo.tar.bz2 -C dir**
- 解压 gz : **tar -zxvf demo.tar.gz**
- 压缩 xz : **tar -cJf demo.tar.xz demo.mp4**

tar 命令



- 打包 : **tar -zcvf log.tar.gz demo.log**
- 打包压缩 : **tar -jcv -f demo.tar.bz2**
- 解压 bz2 : **tar -jxv -f demo.tar.bz2 -C dir**
- 解压 gz : **tar -zxvf demo.tar.gz**
- 压缩 xz : **tar -cJf demo.tar.xz demo.mp4**
- 解压 xz : **tar -cJf demo.tar.xz demo**

tar 命令



- 打包 : **tar -zcvf log.tar.gz demo.log**
- 打包压缩 : **tar -jcv -f demo.tar.bz2**
- 解压 bz2 : **tar -jxv -f demo.tar.bz2 -C dir**
- 解压 gz : **tar -zxvf demo.tar.gz**
- 压缩 xz : **tar -cJf demo.tar.xz demo.mp4**
- 解压 xz : **tar -cJf demo.tar.xz demo**
- 查看 : **tar -jtv -f demo.tar.bz2**

tar 命令



- 打包 : **tar -zcvf log.tar.gz demo.log**
- 打包压缩 : **tar -jcv -f demo.tar.bz2**
- 解压 bz2 : **tar -jxv -f demo.tar.bz2 -C dir**
- 解压 gz : **tar -zxvf demo.tar.gz**
- 压缩 xz : **tar -cJf demo.tar.xz demo.mp4**
- 解压 xz : **tar -cJf demo.tar.xz demo**
- 查看 : **tar -jtv -f demo.tar.bz2**
- 帮助 : **tar --help**

zip 命令

关于 zip

zip 用于压缩，unzip 用于解压缩，生成文件格式是.zip。古老的压缩方式，压缩比较低，基本上操作系统都会默认支持。zip 很少在 Linux 下使用。

- zip 命令跟 tar 稍有不同

怎么用 zip 压缩



重要参数：

-r 递归目录

-e 加密

-S 含系统和隐藏文件
-d 从压缩档内删文件

```
→ tmp zip destDir destDir
   adding: destDir/ (stored 0%)
→ tmp zip -r destDir.zip destDir
   updating: destDir/ (stored 0%)
   adding: destDir/3.log (stored 0%)
   adding: destDir/test.txt (stored 0%)
   adding: destDir/file2 (stored 0%)
→ tmp zip test.zip test.txt
   adding: test.txt (stored 0%)
```

zip 命令

关于 zip

zip 用于压缩，unzip 用于解压缩，生成文件格式是.zip。古老的压缩方式，压缩比较低，基本上操作系统都会默认支持。zip 很少在 Linux 下使用。

- zip 命令跟 tar 稍有不同
- zip 流行 Windows, Linux 一般 gzip。

怎么用 zip 压缩



重要参数：

-r 递归目录

-e 加密

-S 含系统和隐藏文件
-d 从压缩档内删文件

```
→ tmp zip destDir destDir
   adding: destDir/ (stored 0%)
→ tmp zip -r destDir.zip destDir
   updating: destDir/ (stored 0%)
   adding: destDir/3.log (stored 0%)
   adding: destDir/test.txt (stored 0%)
   adding: destDir/file2 (stored 0%)
→ tmp zip test.zip test.txt
   adding: test.txt (stored 0%)
```

zip 命令

关于 zip

zip 用于压缩，unzip 用于解压缩，生成文件格式是.zip。古老的压缩方式，压缩比较低，基本上操作系统都会默认支持。zip 很少在 Linux 下使用。

- zip 命令跟 tar 稍有不同
- zip 流行 Windows，Linux 一般 gzip。
- 文件名在 linux 易乱码 (非 Unicode)

怎么用 zip 压缩



重要参数：

-r 递归目录

-e 加密

-S 含系统和隐藏文件
d 从压缩档内删文件

```
→ tmp zip destDir destDir
   adding: destDir/ (stored 0%)
→ tmp zip -r destDir.zip destDir
   updating: destDir/ (stored 0%)
   adding: destDir/3.log (stored 0%)
   adding: destDir/test.txt (stored 0%)
   adding: destDir/file2 (stored 0%)
→ tmp zip test.zip test.txt
   adding: test.txt (stored 0%)
```

unzip 命令

关于 `unzip` 命令

`unzip` 命令用于解压缩由 `zip` 命令压缩的 “`.zip`” 压缩包。在 Linux 下经常遇到 `zip` 乱码的情况，可使用 `unzip` 的 `-O` 参数。

怎么使用 `unzip` 命令

```
→ tmp unzip test.zip
Archive: test.zip
replace test.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
extracting: test.txt
→ tmp unzip test.zip -d demo
Archive: test.zip
extracting: demo/test.txt
→ tmp unzip -o test.zip -d demo
Archive: test.zip
extracting: demo/test.txt
```



unzip 用法：

- 解压到当前位置: `unzip z`
- 加-d, 解压到指定文件夹
- 加o, 按windows编码解压

unzip 常用参数

可以 `unzip -help` 了解全面的用法。

- `-d` 指定解压后要存目录
- `-O` 指定字符编码为 DOS
- `-I` 指定字符编码为 UNIX
- `-n` 不要覆盖原文件

rar 命令

- rar 是 windows 特有的

```
→ tmp rar a demo.rar destDir
RAR 5.50 Copyright (c) 1993-2017 Alexander Roshal 11 Aug 2017
Trial version Type 'rar -?' for help

Evaluation copy. Please register.

Creating archive demo.rar

Adding destDir/3.log OK
Adding destDir/test.txt OK
Adding destDir/file2 OK
Adding destDir/fromDir2/file1 OK
Adding destDir/fromDir2/file2 OK
Adding destDir/fromDir2 OK
Adding destDir OK
Done OK

→ tmp unrar x demo.rar demo
UNRAR 5.50 freeware Copyright (c) 1993-2017 Alexander Roshal


rar和unrar用法:
- [命令]不带"-"-号, [选项]带"-"-号
- rar a :压缩文件夹
- unrar x 解压到文件夹
```

rar用法

- **rar** [命令] -[选项] .. [压缩档案] [文件]..
- 命令 **a**, 添加文件到压缩文件

unrar用法

- **unrar** [命令] -[选项] .. [压缩档案] [文件]..
- **x** 解压文件到完整路径

rar 命令

```
→ tmp rar a demo.rar destDir
RAR 5.50 Copyright (c) 1993-2017 Alexander Roshal 11 Aug 2017
Trial version Type 'rar -?' for help

Evaluation copy. Please register.

Creating archive demo.rar

Adding destDir/3.log OK
Adding destDir/test.txt OK
Adding destDir/file2 OK
Adding destDir/fromDir2/file1 OK
Adding destDir/fromDir2/file2 OK
Adding destDir/fromDir2 OK
Adding destDir OK
Done OK

→ tmp unrar x demo.rar demo
UNRAR 5.50 freeware Copyright (c) 1993-2017 Alexander Roshal


rar和unrar用法:
- [命令]不带"->"号, [选项]带"->"号
- rar a :压缩文件夹
- unrar x 解压到文件夹
```

- rar 是 windows 特有的
- linux 需安装rar/unrar

rar用法

- **rar** [命令] -[选项] .. [压缩档案] [文件]..
- 命令**a**, 添加文件到压缩文件

unrar用法

- **unrar** [命令] -[选项] .. [压缩档案] [文件]..
- **x** 解压文件到完整路径

rar 命令

```
→ tmp rar a demo.rar destDir
RAR 5.50 Copyright (c) 1993-2017 Alexander Roshal 11 Aug 2017
Trial version Type 'rar -?' for help

Evaluation copy. Please register.

Creating archive demo.rar

Adding destDir/3.log OK
Adding destDir/test.txt OK
Adding destDir/file2 OK
Adding destDir/fromDir2/file1 OK
Adding destDir/fromDir2/file2 OK
Adding destDir/fromDir2 OK
Adding destDir OK
Done OK

→ tmp unrar x demo.rar demo
UNRAR 5.50 freeware Copyright (c) 1993-2017 Alexander Roshal


rar和unrar用法:
- [命令]不带"->"号, [选项]带"->"号
- rar a :压缩文件夹
- unrar x 解压到文件夹
```

- rar 是 windows 特有的
- linux 需安装rar/unrar
- 无法通过 tar 命令解压

rar用法

- **rar** [命令] -[选项] .. [压缩档案] [文件]..
- 命令 **a**, 添加文件到压缩文件

unrar用法

- **unrar** [命令] -[选项] .. [压缩档案] [文件]..
- **x** 解压文件到完整路径

⑤ 感谢及提问

感谢观看，欢迎提问！

