《常用 Linux 命令备忘手册》

- 由于时间仓促和个人能力有限，文档中难免存在错误和疏漏，还请大家批评指正。
- 该手册会持续更新，遇到好玩的命令会往里面加。如果你那也有好玩的命令想要加，可以私信我哦。
- 二哥的 Java 进阶之路：Linux 最新学习路线

最新 PDF 获取

讲个笑话，PDF 内容没办法自动更新（😂），所以只能通过下面的方式，别怪我：

> 微信搜索《**沉默王二**》或者微信扫下面的二维码，关注后回复《**Linux**》即可获取最新的 PDF 版本。

获取方式见下图（我用的 PC 端微信截图，手机端差不多）：

附其他干货笔记下载地址：

- 阮一峰 C 语言入门教程 PDF 下载
- Java 核心知识点整理 PDF 下载
- 深入浅出 Java 多线程 PDF 下载
- Pro Git 中文版 PDF 下载

- 给操作系统捋条线 PDF 下载

个人常用命令

> 收录一些我个人在工作和学习中常用的命令，见下表，觉得好用可以背下来哦。

命令	说明
`ls`	列出当前目录下的文件和目录
`cd`	切换目录
`pwd`	显示当前目录的路径
`vim`	编辑文件
`mkdir`	创建新目录
`rm -rf`	删除文件或目录
`cp source target`	复制文件或目录，`-r` 参数为目录
`mv source target`	移动文件或目录
`cat`	查看文件内容
`chomd`	修改文件权限
`zip`	压缩文件
`unzip`	解压文件
ps -ef | grep java	查找所有包含 java 的进程
`kill -9 PID`	强制杀死进程
`top`	查看系统资源使用情况
`df -h`	查看磁盘空间使用情况
`nohup java -jar app.jar > output.log 2>&1 &`	后台启动 Java 应用，并将日志保存到指定文件
`ping`	测试网络连通性
netstat -anp | grep 8080	查看端口占用情况

文件和目录

> 文件和目录是 Linux 系统中最基本的组成部分，下面是一些常用的文件和目录操作命令，一起来看下 😯。

`ls` 列出目录内容

- `ls`：列出当前目录下的文件和子目录。
- `ls -l`：以列表形式显示详细信息，包括文件权限、所有者、大小和最后修改日期。
- `ls -a`：显示所有文件，包括隐藏文件（以.开头的文件）。
- `ls -lh`：以易读的格式显示文件大小（例如，用 K、M、B 表示）。

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (0.084s)
ls
javabetter.pdf

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.088s)
ls -l
total 8
-rw-r--r--  1 maweiqing  staff  48 Feb 14 16:32 javabetter.pdf

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.085s)
ls -a
.               ..              javabetter.pdf

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.091s)
ls -lh
total 8
-rw-r--r--  1 maweiqing  staff    48B Feb 14 16:32 javabetter.pdf
```

`cd` 更改当前目录

- `cd /path/to/directory`：切换到指定的目录。
- `cd ..`：返回上一级目录。
- `cd ~`：切换到当前用户的主目录。

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (0.082s)
cd java

~/Documents/GitHub/linux-pdf/java git:(master) ±132 (0.086s)
cd ..

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.094s)
cd ~

~ git:(master) ±132 (0.083s)
pwd
/Users/maweiqing
```

`pwd` 显示当前目录

- `pwd`：显示当前工作目录的绝对路径。

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (0.084s)
pwd
/Users/maweiqing/Documents/GitHub/linux-pdf
```

`mkdir` 创建新目录

- `mkdir directory`：创建一个新目录。
- `mkdir -p directory/subdirectory`：创建一个新目录和子目录。

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (0.09s)
mkdir -p c/pdf

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.112s)
ls
c               java            javabetter.pdf

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.083s)
cd c

~/Documents/GitHub/linux-pdf/c git:(master) ±132 (0.085s)
ls
pdf
```

`rm` 删除文件或目录

- `rm file`：删除文件。
- `rm -r directory`：删除目录及其所有内容。
- `rm -f file`：强制删除文件，不提示。
- `rm -rf directory`：强制删除目录及其所有内容，不提示。

```
~/Documents/GitHub/linux-pdf/c/pdf git:(master) ±132 (0.089s)
ls
hello.wanger

~/Documents/GitHub/linux-pdf/c/pdf git:(master) ±132 (0.087s)
rm hello.wanger

~/Documents/GitHub/linux-pdf/c/pdf git:(master) ±132 (0.09s)
ls

~/Documents/GitHub/linux-pdf/c/pdf git:(master) ±132 (0.088s)
mkdir test

~/Documents/GitHub/linux-pdf/c/pdf git:(master) ±132 (0.089s)
rm -f test/
rm: test/: is a directory

~/Documents/GitHub/linux-pdf/c/pdf git:(master) ±132 (0.088s)
rm -r test

~/Documents/GitHub/linux-pdf/c/pdf git:(master) ±132 (0.087s)
ls
```

`cp` 复制文件或目录

- `cp source target`：复制文件。
- `cp -r source target`：递归复制目录极其内容。

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (0.086s)
ls
c               java            javabetter.pdf

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.089s)
cp -r c c1

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.087s)
ls
c               c1              java            javabetter.pdf

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.082s)
cd c1/pdf/

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.084s)
ls
hello.wanger

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.09s)
cp hello.wanger wanger.hello

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.086s)
ls
hello.wanger wanger.hello
```

`mv` 移动文件或目录

- `mv source target`：移动文件或目录，也可以用来重命名文件。

```
~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.085s)
ls
hello1.wanger wanger.hello

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.094s)
mv hello1.wanger /Users/maweiqing/Documents/GitHub/linux-pdf/java/hello.wanger

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.088s)
ls
wanger.hello
```

`cat` 查看文件内容

- `cat file`：显示文件内容。
- `cat file1 file2`：连接多个文件并显示内容。
- `cat > file`：创建一个新文件并输入内容，按 `Ctrl + D` 保存退出。
- `cat >> file`：追加内容到文件末尾。
- `cat file1 file2 > file3`：合并 file1 和 file2 的内容，并将合并后的内容存储到 file3 中。

```
~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±13
cat > hello.wanger
龙年暴富，二哥牛逼。

考研上岸，校招无敌。^D

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±13
cat hello.wanger
龙年暴富，二哥牛逼。

考研上岸，校招无敌。

~/Documents/GitHub/linux-pdf/c1/pdf git:
cat hello.wanger wanger.hello
龙年暴富，二哥牛逼。

考研上岸，校招无敌。
sd

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±13
cat >> hello.wanger
不错。%
```

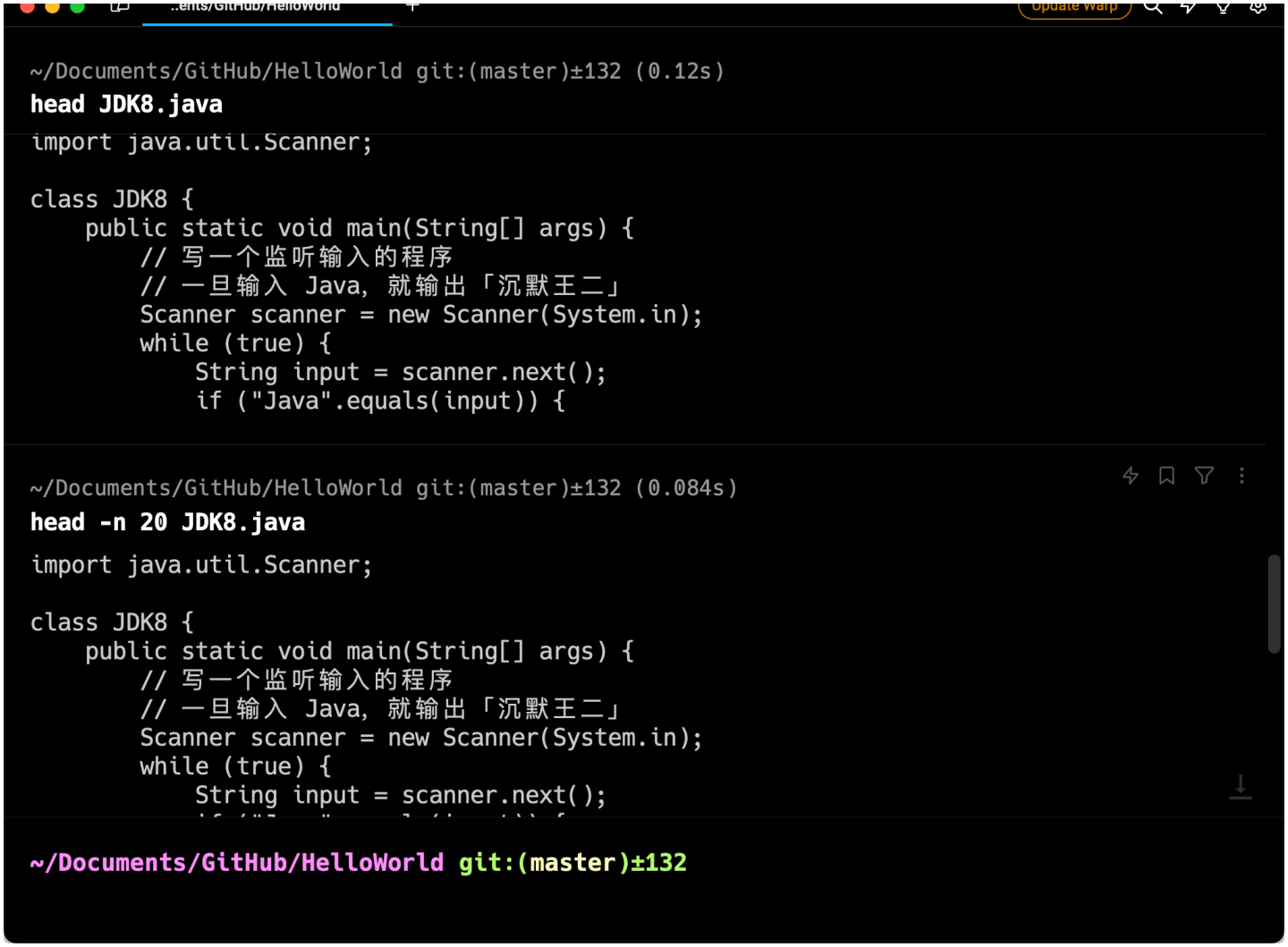
`echo` 输出文本

- `echo "Hello, World!"`：输出文本。
- `echo "Hello, World!" > file`：将输出重定向到文件。
- `echo "Hello, World!" >> file`：将输出追加到文件末尾。
- `echo -e "Hello\nWorld"`：支持转义字符。
- `echo -n "Hello, World!"`：不换行输出。
- `echo $?`：显示上一个命令的退出状态。
- `echo $SHELL`：显示当前 shell 的类型。

```
root@iZ2zebrh6ffesjwecx7eq9Z:~# echo "Hello, World!"
Hello, World!
root@iZ2zebrh6ffesjwecx7eq9Z:~# echo -e "Line 1\nLine 2"
Line 1
Line 2
root@iZ2zebrh6ffesjwecx7eq9Z:~# echo "Today is $(date)"
Today is Fri Feb 23 16:24:11 CST 2024
```

`head` 和 `tail` 查看文件头和尾

- `head file`：显示文件的前 10 行。
- `head -n 20 file`：显示文件的前 20 行。
- `tail file`：显示文件的最后 10 行。
- `tail -n 20 file`：显示文件的最后 20 行。



`vim` 编辑文件

- `vim file`：编辑文件。
- `i`：切换到插入模式。
- `Esc`：退出插入模式。
- `:w`：保存文件。
- `:q`：退出文件。
- `:wq`：保存并退出文件。
- `:q!`：强制退出文件。

vim 是一个非常强大的文本编辑和查看工具，我这里只列出了一些最基本的操作，更多的操作可以参考《Vim 学习路线》。

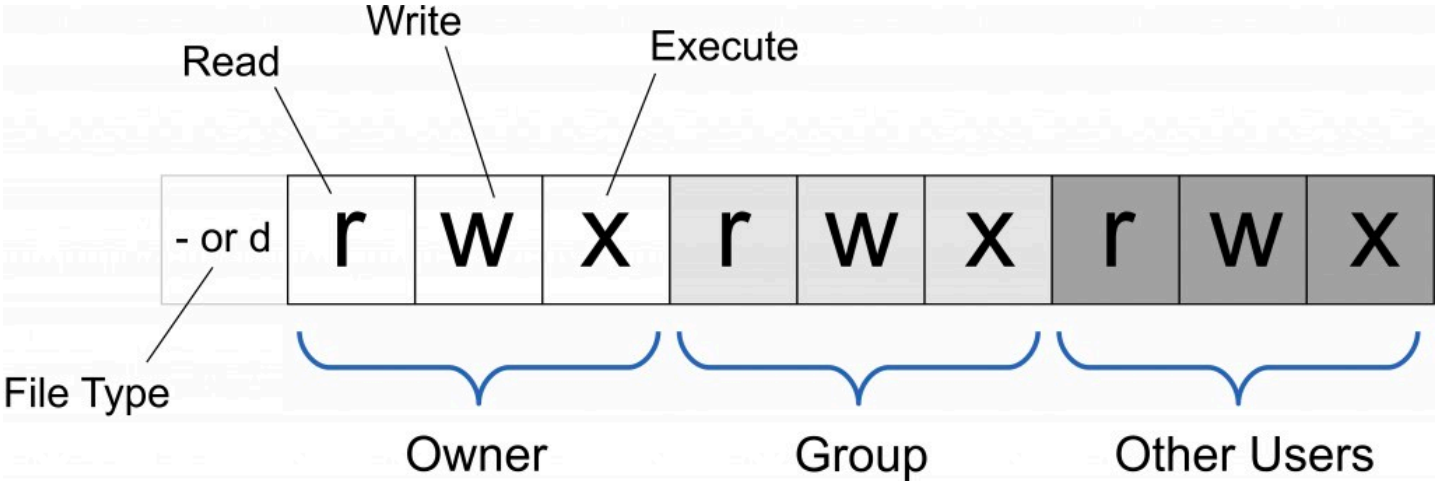
`chmod` 修改文件权限

chmod 修改文件权限

- `chmod 777 file`：赋予文件所有权限。

Linux 中的权限可以应用于三种类别的用户：

- 文件所有者（u）
- 与文件所有者同组的用户（g）
- 其他用户（o）



①、符号模式

符号模式使用字母来表示权限，如下：

- 读（r）
- 写（w）
- 执行（x）
- 所有（a）

例如：

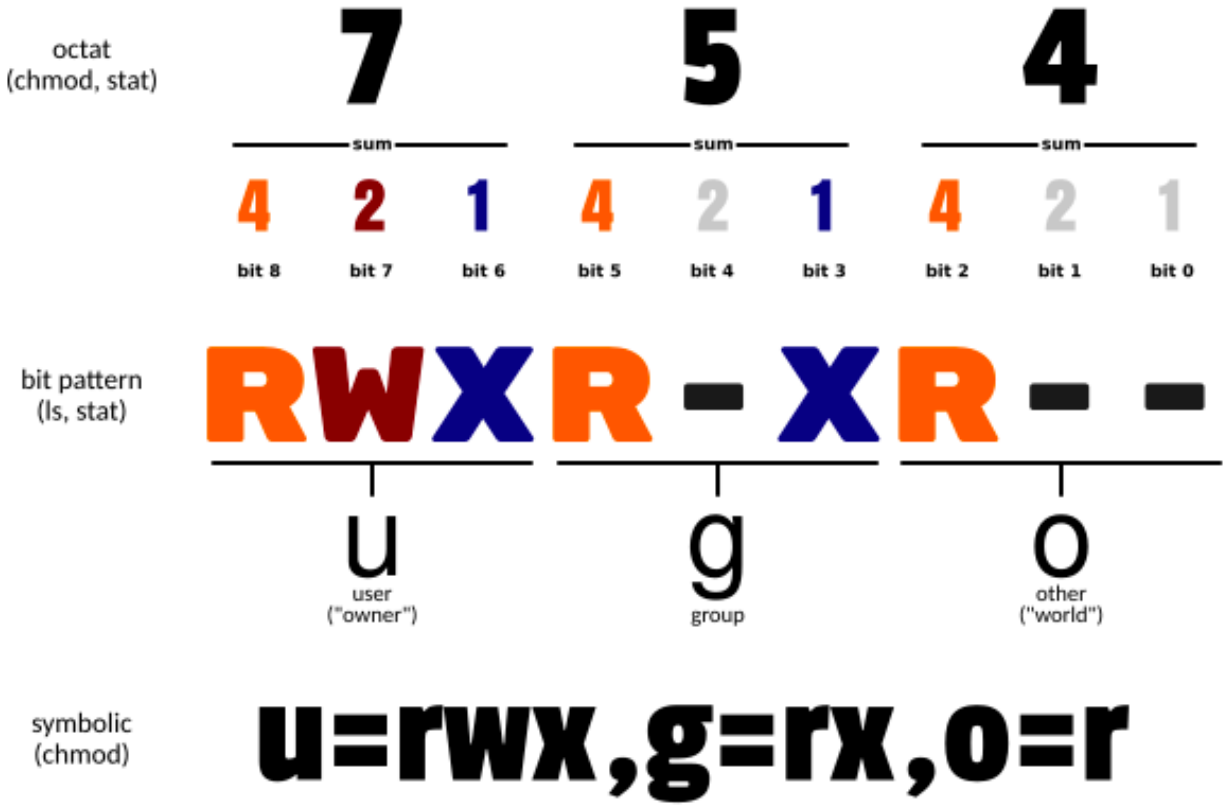
- `chmod u+w file`：给文件所有者添加写权限。
- `chmod g-r file`：移除组用户的读权限。
- `chmod o+x file`：给其他用户添加执行权限。
- `chmod u=rwx,g=rx,o=r file`：设置文件所有者具有读写执行权限，组用户具有读执行权限，其他用户具有读权限。

②、数字模式

②、数字模式

数字模式使用三位八进制数来表示权限，每位数字代表不同的用户类别（所有者、组、其他用户），数字是其各自权限值的总和：

- 读（r）= 4
- 写（w）= 2
- 执行（x）= 1



因此，权限模式可以是从 0（无权限）到 7（读写执行权限）的任何值。

- chmod 755 file：使得文件所有者有读写执行（7）权限，组用户和其他用户有读和执行（5）权限。
- chmod 644 file：使得文件所有者有读写（6）权限，而组用户和其他用户只有读（4）权限。

我之前在讲MySQL 的可执行文件的时候，也提到过这个命令。

使用 `chmod +x test.sh` 或者 `chmod 755 test.sh` 来给文件添加可执行权限。

`zip` 压缩文件

Linux 提供了多种工具来压缩和解压文件，其中最常用的包括`gzip`、`bzip2`、`tar`、和`zip`。下面是这些工具的基本用法：

gzip

gzip

1、**压缩文件**

- `gzip filename`：将文件压缩成`.gz`格式。原文件会被压缩后的文件替换。
- `gzip -k filename`：压缩文件但保留原文件。

```
~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.129s)
ls
hello.wanger test.md      wanger.hello

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.134s)
gzip hello.wanger

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.086s)
ls
hello.wanger.gz test.md         wanger.hello

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.09s)
gzip -k wanger.hello

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.086s)
ls
hello.wanger.gz test.md         wanger.hello    wanger.hello.gz
```

2、**解压文件**，`gzip -d filename.gz`：解压`.gz`格式的文件。

```
~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.086s)
ls
hello.wanger.gz test.md         wanger.hello    wanger.hello.gz

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.086s)
gzip -d hello.wanger.gz

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.087s)
ls
hello.wanger    test.md         wanger.hello    wanger.hello.gz
```

tar

`tar`是一种将多个文件和目录打包成一个文件（通常为`.tar`文件）的工具，它可以与`gzip`结合使用来进行压缩。

1、打包并压缩，`tar -czvf archive_name.tar.gz directory_or_file`：创建一个 gzip 压缩的 tar 包。

- `c`：创建一个新的 tar 包。
- `z`：使用 gzip 压缩。
- `v`：显示详细信息。
- `f`：指定文件名。

```
~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.087s)
ls
hello.wanger    test.md         wanger.hello    wanger.hello.gz

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.093s)
tar -czvf hello.wanger.tar.gz hello.wanger
a hello.wanger

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.084s)
ls
hello.wanger        hello.wanger.tar.gz test.md             wanger.hello        wanger.hello.gz
```

2、**解压**，`tar -xzf archive_name.tar.gz`：解压一个 gzip 压缩的 tar 包。

zip

1、**压缩文件或目录**，`zip -r archive_name.zip directory_or_file`：压缩文件或目录到`.zip`格式。

```
~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.084s)
ls
hello.wanger test.md      wanger.hello

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.094s)
zip wanger.hello.zip wanger.hello
  adding: wanger.hello (stored 0%)

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.083s)
ls
hello.wanger     test.md          wanger.hello     wanger.hello.zip
```

2、**解压**，`unzip archive_name.zip`：解压`.zip`格式的文件。

```
~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.087s)
ls
hello.wanger     test.md          wanger.hello.zip

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.093s)
unzip wanger.hello.zip
Archive:  wanger.hello.zip
 extracting: wanger.hello

~/Documents/GitHub/linux-pdf/c1/pdf git:(master) ±132 (0.084s)
ls
hello.wanger     test.md          wanger.hello     wanger.hello.zip
```

`tree` 显示目录结构

- `tree`：以树形结构显示目录。
- `tree -L 2`：只显示两层目录。
- `tree -d`：只显示目录。
- `tree -a`：显示所有文件和目录。

- `tree -h`：以易读的格式显示文件大小。

`find` 查找文件

find 命令用于在指定目录下递归地查找符合条件的文件和目录。可以根据文件名、类型、大小、权限、修改日期等多种条件来搜索文件。

基本用法：`find [path] [options] [expression]`

- path：指定`find`命令开始搜索的目录路径。`.`为当前目录。可以指定多个路径。
- options：定义搜索的具体条件，如文件名、文件类型、大小、权限、所有者、修改时间等。如果没有指定搜索条件，`find`命令会列出指定路径下的所有文件。
- expression：对搜索到的文件执行的操作，如打印文件名、删除文件等。如果没有指定操作，则默认操作是打印到标准输出。

常用的搜索条件有：

- **按名称搜索**，`-name 'pattern'`：按照文件名模式搜索文件。模式内可以使用通配符。
- **按类型搜索**
  - `-type f`：只搜索文件。
  - `-type d`：只搜索目录。
- **按大小搜索**
  - `-size +2M`：搜索大于 2MB 的文件。

- `-size -5k`：搜索小于 5KB 的文件。

- **按修改时间搜索**
  - `-mtime +7`：搜索最后修改时间在 7 天前的文件。
  - `-mtime -1`：搜索最后修改时间在 1 天内的文件。
- **按权限搜索**，`-perm 644`：搜索权限正好为 644 的文件。

常用操作有：

- **打印文件名**：`-print`：显示搜索结果的完整路径名。
- **执行命令**：`-exec command {} \;`：对每个搜索到的文件执行指定的命令。`{}`代表当前找到的文件名。

示例

①、查找并列出当前目录及子目录下所有的`.txt`文件：

```
find . -type f -name "*.txt"
```

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (0.09s)
find . -type f -name "*.hello"
./c1/pdf/wanger.hello
```

②、搜索`/home`目录下所有修改时间在 10 天前的`.jpg`文件：

```
find /home -type f -name "*.jpg" -mtime +10
```

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (8.005s)
find /Users/maweiqing/Documents/Github -mtime -1
/Users/maweiqing/Documents/Github
/Users/maweiqing/Documents/Github/javabetter/docs/pdf/linux-vip.md
/Users/maweiqing/Documents/Github/javabetter/docs/mysql
/Users/maweiqing/Documents/Github/javabetter/docs/mysql/column.md
/Users/maweiqing/Documents/Github/javabetter/docs/mysql/btree.md
/Users/maweiqing/Documents/Github/javabetter/.git
/Users/maweiqing/Documents/Github/javabetter/.git/ORIG_HEAD
/Users/maweiqing/Documents/Github/javabetter/.git/objects
/Users/maweiqing/Documents/Github/javabetter/.git/logs/HEAD
```

③、搜索`/var/log`目录下所有大于 50MB 的文件：

```
find /var/log -type f -size +50M
```

```
find /var/log -type f -size +50M
```

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (2.579s)
find /Users/maweiqing/Documents/Github -type f -size +10M

/Users/maweiqing/Documents/Github/checkstyle/.git/objects/pack/pack-9fd68090.pack
/Users/maweiqing/Documents/Github/codingmore-learning/springboot-docker-0.0.1-SNAPSHOT.jar
```

④、查找并对所有`.txt`文件执行`chmod`命令改变权限：

```
find . -type f -name "*.txt" -exec chmod 755 {} \;
```

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (0.088s)
ls -l

total 8
drwxr-xr-x  3 maweiqing  staff  96 Feb 14 16:44 c
drwxr-xr-x  3 maweiqing  staff  96 Feb 14 20:38 c1
drwxr-xr-x  3 maweiqing  staff  96 Feb 14 20:42 java
-rw-r--r--  1 maweiqing  staff  48 Feb 14 16:32 javabetter.pdf

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.097s)
find . -type f -name "*.pdf" -exec chmod 755 {} \;

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.092s)
ls -l

total 8
drwxr-xr-x  3 maweiqing  staff  96 Feb 14 16:44 c
drwxr-xr-x  3 maweiqing  staff  96 Feb 14 20:38 c1
drwxr-xr-x  3 maweiqing  staff  96 Feb 14 20:42 java
-rwxr-xr-x  1 maweiqing  staff  48 Feb 14 16:32 javabetter.pdf
```

`whereis`查找命令

whereis 查找命令

whereis 命令用来查找文件的位置，特别是用于查找二进制文件、源代码文件、和帮助文档的位置。它比 find 命令运行得更快，因为 whereis 不会遍历整个文件系统，而是在一组预定义的路径中查找文件，这些路径通常包括 Linux 系统上安装软件的标准位置。

基本用法：`whereis [选项] 程序名`：

- `-b`：只查找二进制文件。
- `-m`：只查找帮助手册的路径。
- `-s`：只查找源代码文件。
- `-u`：查找未分类的文件。

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (10.276s)
whereis java
2024-02-14 21:54:49.360 xcodebuild[76408:20445062] Requested but did not find extension point with identifier Xcode.IDEKit.ExtensionSentinelHostApplications for extension Xcode.DebuggerFoundation.AppExtensionHosts.watchOS of plug-in com.apple.dt.IDEWatchSupportCore
2024-02-14 21:54:49.361 xcodebuild[76408:20445062] Requested but did not find extension point with identifier Xcode.IDEKit.ExtensionPointIdentifierToBundleIdentifier for extension Xcode.DebuggerFoundation.AppExtensionToBundleIdentifierMap.watchOS of plug-in com.apple.dt.IDEWatchSupportCore
java: /usr/bin/java
```

`which` 查找命令

- `which command`：查找命令的位置，前提是该命令在用户的 PATH 环境变量所指定的目录中。

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (0.09s)
which java
/Users/maweiqing/.jenv/shims/java
```

表格总结

用一张表格再总结一下（贴心吧）：

命令	说明
`ls`	列出目录内容
`ls -l`	详细列表
`ls -a`	显示所有文件
`ls -lh`	易读格式
`cd /path/to/directory`	切换目录

`cd ..`	返回上一级目录
`cd ~`	切换到主目录
`pwd`	显示当前目录
`mkdir directory`	创建新目录
`mkdir -p directory/subdirectory`	创建新目录和子目录
`rm file`	删除文件
`rm -r directory`	删除目录及其内容
`rm -f file`	强制删除文件
`rm -rf directory`	强制删除目录及其内容
`cp source target`	复制文件或目录
`cp -r source target`	递归复制目录及其内容
`mv source target`	移动文件或目录
`cat file`	查看文件内容
`cat file1 file2`	连接多个文件并显示内容
`cat > file`	创建新文件并输入内容
`cat >> file`	追加内容到文件末尾
`cat file1 file2 > file3`	合并文件内容
`echo "Hello, World!"`	输出文本
`echo "Hello, World!" > file`	输出重定向到文件
`echo "Hello, World!" >> file`	输出追加到文件末尾
`echo -e "Hello\nWorld"`	支持转义字符
`echo -n "Hello, World!"`	不换行输出
`echo $?`	显示上一个命令的退出状态
`echo $SHELL`	显示当前 shell 的类型
`head file`	显示文件的前 10 行
`head -n 20 file`	显示文件的前 20 行

`tail file`	显示文件的最后 10 行
`tail -n 20 file`	显示文件的最后 20 行
`vim file`	编辑文件
`i`	切换到插入模式
`Esc`	退出插入模式
`:w`	保存文件
`:q`	退出文件
`:wq`	保存并退出文件
`:q!`	强制退出文件
`chmod 777 file`	修改文件权限
`chmod u+w file`	给文件所有者添加写权限
`chmod g-r file`	移除组用户的读权限
`chmod o+x file`	给其他用户添加执行权限
`chmod u=rwx,g=rx,o=r file`	设置文件所有者具有读写执行权限，组用户具有读执行权限，其他用户具有读权限
`chmod 755 file`	给文件所有者有读写执行权限，组用户和其他用户有读和执行权限
`chmod 644 file`	给文件所有者有读写权限，而组用户和其他用户只有读权限
`gzip filename`	gzip压缩文件
`gzip -k filename`	gzip压缩文件但保留原文件
`gzip -d filename.gz`	解压.gz文件
`tar -czvf archive_name.tar.gz directory_or_file`	创建一个 gzip 压缩的 tar 包
`tar -xzf archive_name.tar.gz`	解压一个 gzip 压缩的 tar 包
`zip -r archive_name.zip directory_or_file`	压缩文件或目录到.zip格式
`unzip archive_name.zip`	解压.zip格式的文件

`tree`	显示目录结构
`tree -L 2`	只显示两层目录
`tree -d`	只显示目录
`tree -a`	显示所有文件
`tree -h`	易读格式
`find . -type f -name "*.txt"`	查找并列出当前目录及子目录下所有的.txt文件
`find /home -type f -name "*.jpg" -mtime +10`	搜索/home目录下所有修改时间在10天前的.jpg文件
`find /var/log -type f -size +50M`	搜索/var/log目录下所有大于50MB的文件
`find . -type f -name "*.txt" -exec chmod 755 {} \;`	查找并对所有.txt文件执行chmod命令改变权限
`whereis command`	查找命令的位置
`which command`	查找命令的位置

网络和进程管理

`ping` 测试网络连通性

> 如果你本机的电脑连不上服务器，那结果无非两样，你本机的网络出问题了，或者服务器的网络出问题了，（假装我没说）这时候就可以使用 `ping` 命令来测试网络连通性。

- `ping domain`：测试域名的连通性。
- `ping IP`：测试 IP 的连通性。

```
~ git:(master)±132 (7.368s)
```

```
ping javabetter.cn

PING javabetter.cn (39.105.208.175): 56 data bytes
64 bytes from 39.105.208.175: icmp_seq=0 ttl=51 time=22.849 ms
64 bytes from 39.105.208.175: icmp_seq=1 ttl=51 time=22.568 ms
64 bytes from 39.105.208.175: icmp_seq=2 ttl=51 time=23.513 ms
64 bytes from 39.105.208.175: icmp_seq=3 ttl=51 time=23.077 ms
64 bytes from 39.105.208.175: icmp_seq=4 ttl=51 time=23.513 ms
64 bytes from 39.105.208.175: icmp_seq=5 ttl=51 time=23.681 ms
64 bytes from 39.105.208.175: icmp_seq=6 ttl=51 time=23.512 ms
^C
--- javabetter.cn ping statistics ---
7 packets transmitted, 7 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 22.568/23.245/23.681/0.387 ms
```

`netstat` 查看网络状态

- `netstat -a`：显示所有的输入和输出连接，包括未处于监听状态的。
- `netstat -at`：显示所有 TCP 连接。
- `netstat -au`：显示所有 UDP 连接。
- `netstat -l`：仅显示监听中的服务器端口。
- `netstat -p`：显示进程 ID 和进程名称。
- `netstat -s`：显示网络统计信息。
- `netstat -n`：显示 IP 地址和端口号。
- `netstat -tuln`：显示所有 TCP 和 UDP 连接的 IP 地址和端口号。
- `netstat -antp`：显示所有 TCP 连接的 IP 地址和端口号，以及进程 ID 和进程名称。
- `netstat -anp | grep 8080`：查看端口占用情况。

```
root@iZ2zebrh6ffesjwecx7eq9Z:~# netstat -anp | grep 443
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN      30836/nginx: master
tcp        0      0 172.25.12.84:443        42.92.124.74:2969       TIME_WAIT   -
tcp        0      0 172.25.12.84:443        36.60.52.76:24501       TIME_WAIT   -
```

`ps` 查看进程

- `ps`：显示当前用户的进程。
- `ps -ef`：显示所有进程。
- `ps -ef | grep java`：查找所有包含 java 的进程。

```
root@iZ2zebrh6ffesjwecx7eq9Z:~# ps -ef | grep java
root     29442     1  0  2022 ?        09:50:30 java -jar codingmore-web-1.0-SNAPSHOT.jar
```

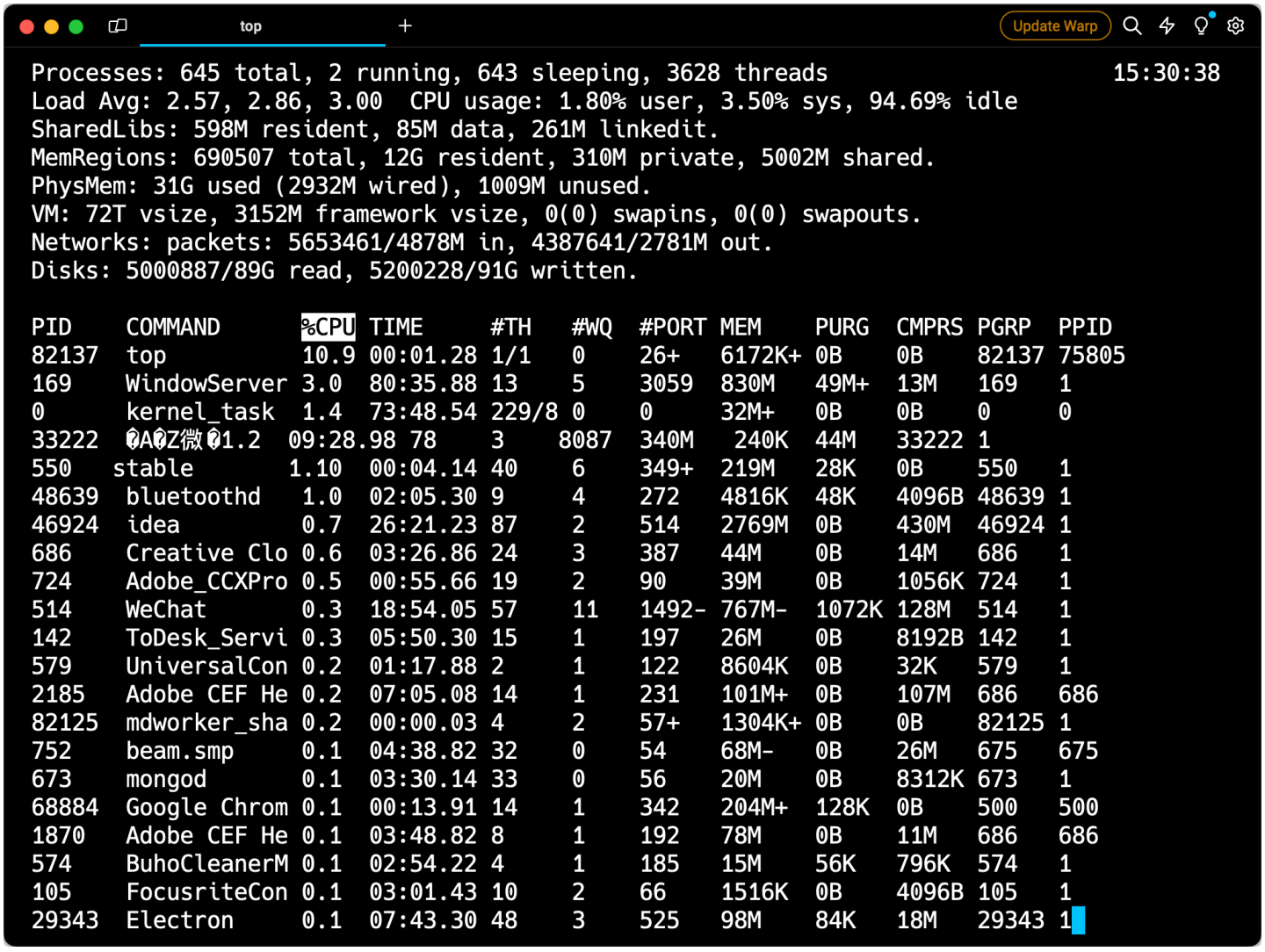
```
root     31231 30941  0 15:28 pts/1    00:00:00 grep --color=auto java
root     31747     1  0  2022 ?        10:51:43 java -jar codingmore-admin-1.0-SNAPSHOT.jar
```

`kill` 终止进程

- `kill PID`：终止指定进程。
- `kill -9 PID`：强制终止指定进程。

`top` 查看系统资源

- `top`：显示系统资源使用情况，`Ctrl + C` 退出。



`nohup` 后台启动应用

- `nohup java -jar app.jar > output.log 2>&1 &`：后台启动 Java 应用，并将日志保存到指定文件。
- `nohup command &`：后台启动命令。

表格总结

同样，我们来用一张表格总结一下（贴心到你会感动 😂）：

命令	说明

`ping domain`	测试域名的连通性
`ping IP`	测试 IP 的连通性
`netstat -a`	显示所有的输入和输出连接
`netstat -at`	显示所有 TCP 连接
`netstat -au`	显示所有 UDP 连接
`netstat -l`	仅显示监听中的服务器端口
`netstat -p`	显示进程 ID 和进程名称
`netstat -s`	显示网络统计信息
`netstat -n`	显示 IP 地址和端口号
`netstat -tuln`	显示所有 TCP 和 UDP 连接的 IP 地址和端口号
`netstat -antp`	显示所有 TCP 连接的 IP 地址和端口号，以及进程 ID 和进程名称
netstat -anp | grep 8080	查看端口占用情况
`ps`	显示当前用户的进程
`ps -ef`	显示所有进程
ps -ef | grep java	查找所有包含 java 的进程
`kill PID`	终止指定进程
`kill -9 PID`	强制终止指定进程
`top`	显示系统资源使用情况
`nohup java -jar app.jar > output.log 2>&1 &`	后台启动 Java 应用，并将日志保存到指定文件
`nohup command &`	后台启动命令

常见系统服务命令

> 我自己最常用的主要有 service 和 systemctl，见下表。

命令	说明

`service 服务名 status`	查看某个服务状态
`service 服务名 start`	启动某个服务
`service 服务名 stop`	停止某个服务
`service 服务名 restart`	重启某个服务
`systemctl status 服务名`	查看某个服务状态
`systemctl start 服务名`	启动某个服务
`systemctl stop 服务名`	停止某个服务
`systemctl restart 服务名`	重启某个服务
`systemctl enable 服务名`	设置开机启动
`systemctl disable 服务名`	取消开机启动

service 命令通常与传统的 SysVinit 系统一起使用，而 systemctl 命令是 systemd 初始化系统的一部分。

- SysVinit：是传统的 Linux 系统初始化工具，负责启动系统时的进程启动、监控和终止。SysVinit 使用脚本（位于`/etc/init.d/`目录下）来管理服务。
- systemd：是一个较新的初始化系统，旨在提供更快的启动时间和更好的依赖管理。它引入了 unit 文件的概念来管理资源，这些资源不仅限于服务，还包括挂载点、设备等。systemd 的配置文件通常位于`/etc/systemd/system/`和`/usr/lib/systemd/system/`目录下。

```
root@iZ2zebrh6ffesjwecx7eq9Z:~# service nginx status
● nginx.service - nginx - high performance web server
```

```
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2023-04-11 08:39:51 CST; 10 months 13 days ago
     Docs: http://nginx.org/en/docs/
 Main PID: 30836 (nginx)
    Tasks: 3 (limit: 4455)
   CGroup: /system.slice/nginx.service
           ├─30836 nginx: master process /usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf
           ├─31362 nginx: worker process
           └─31363 nginx: worker process

Dec 22 08:54:28 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloading nginx - high performance web server.
Dec 22 08:54:28 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloaded nginx - high performance web server.
Dec 22 09:08:48 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloading nginx - high performance web server.
Dec 22 09:08:48 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloaded nginx - high performance web server.
Dec 22 09:13:28 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloading nginx - high performance web server.
Dec 22 09:13:28 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloaded nginx - high performance web server.
Dec 22 09:14:55 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloading nginx - high performance web server.
Dec 22 09:14:55 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloaded nginx - high performance web server.
Jan 20 07:27:42 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloading nginx - high performance web server.
Jan 20 07:27:42 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloaded nginx - high performance web server.
root@iZ2zebrh6ffesjwecx7eq9Z:~# systemctl status nginx
● nginx.service - nginx - high performance web server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2023-04-11 08:39:51 CST; 10 months 13 days ago
     Docs: http://nginx.org/en/docs/
 Main PID: 30836 (nginx)
    Tasks: 3 (limit: 4455)
   CGroup: /system.slice/nginx.service
           ├─30836 nginx: master process /usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf
           ├─31362 nginx: worker process
           └─31363 nginx: worker process

Dec 22 08:54:28 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloading nginx - high performance web server.
Dec 22 08:54:28 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloaded nginx - high performance web server.
Dec 22 09:08:48 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloading nginx - high performance web server.
Dec 22 09:08:48 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloaded nginx - high performance web server.
Dec 22 09:13:28 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloading nginx - high performance web server.
Dec 22 09:13:28 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloaded nginx - high performance web server.
Dec 22 09:14:55 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloading nginx - high performance web server.
Dec 22 09:14:55 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloaded nginx - high performance web server.
Jan 20 07:27:42 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloading nginx - high performance web server.
Jan 20 07:27:42 iZ2zebrh6ffesjwecx7eq9Z systemd[1]: Reloaded nginx - high performance web server.
```

磁盘和分区管理

`df` 查看磁盘空间

- `df`：显示磁盘空间使用情况。
- `df -h`：以易读的格式显示磁盘空间使用情况，例如 GB、MB 等。

```
~/Documents/GitHub/HelloWorld git:(master)±132 (0.081s)
df -h

Filesystem       Size   Used  Avail Capacity iused      ifree %iused  Mounted on
/dev/disk2s5s1  953Gi   14Gi  247Gi     6%  502128 2590780400    0%   /
devfs           191Ki  191Ki    0Bi   100%     660          0  100%   /dev
/dev/disk2s4    953Gi  2.0Gi  247Gi     1%       2 2590780400    0%   /System/Volumes/VM
/dev/disk2s2    953Gi  413Mi  247Gi     1%    2131 2590780400    0%   /System/Volumes/Preboot
/dev/disk2s6    953Gi  3.4Mi  247Gi     1%      19 2590780400    0%   /System/Volumes/Update
/dev/disk2s1    953Gi  684Gi  247Gi    74% 4257397 2590780400    0%   /System/Volumes/Data
map auto_home     0Bi    0Bi    0Bi   100%       0          0  100%   /System/Volumes/Data/home
```

`du` 查看目录大小

- `du`：估算文件或目录的磁盘空间使用量。
- `du -ash /path/to/directory`：`-s`汇总总用量，`-h`以易读方式显示，不指定陌路默认为当前目录。
- `du -sk * | sort -n`：列出当前目录下所有文件和目录的大小，并按大小排序。
- `du -sk * | sort -rn`：列出当前目录下所有文件和目录的大小，并按大小逆序排序。

```
~/Documents/GitHub/HelloWorld git:(master)±132 (0.114s)
du -sh

6.1M    .
```

```
~/Documents/GitHub/HelloWorld git:(master)±132 (0.082s)
du

24      ./out/production/HelloWorld
24      ./out/production
24      ./out
16      ./.idea/codeStyles
0       ./.idea/ZeppelinRemoteNotebooks
56      ./.idea
8       ./src
12544   .
```

`fdisk` 分区管理

- `fdisk -l`：列出所有磁盘分区。
- `fdisk /dev/sda`：对磁盘`/dev/sda`进行分区。

```
root@iZ2zebrh6ffesjwecx7eq9Z:~# fdisk -l
```

```
Disk /dev/vda: 80 GiB, 85899345920 bytes, 167772160 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xa07257b5

Device     Boot Start       End   Sectors Size Id Type
/dev/vda1  *     2048 167772126 167770079  80G 83 Linux
```

表格总结

继续用一张表格总结一下（贴心到你会觉得自己就是个皇帝或者皇后 😂）：

命令	说明
`df`	查看磁盘空间
`df -h`	以易读的格式显示磁盘空间使用情况
`du`	查看目录大小
`du -ash /path/to/directory`	汇总总用量，以易读方式显示
du -sk * | sort -n	列出当前目录下所有文件和目录的大小，并按大小排序
du -sk * | sort -rn	列出当前目录下所有文件和目录的大小，并按大小逆序排序
`fdisk -l`	列出所有磁盘分区
`fdisk /dev/sda`	对磁盘`/dev/sda`进行分区

系统信息和性能查看

`uname` 查看系统信息

- `uname`：显示系统信息。
- `uname -a`：查看内核/OS/CPU 信息。
- `uname -r`：查看内核版本。
- `uname -m`：查看处理器架构，同`arch`命令。
- `hostname`：查看主机名。
- `who`：查看当前登录用户，同 `who am i`。
- `whoami`：查看当前用户名。

- `cat /proc/version`：查看内核版本。
- `cat /proc/cpuinfo`：查看 CPU 信息。
- `uptime`：查看系统运行时间和平均负载。

```
root@iZ2zebrh6ffesjwecx7eq9Z:~# uname -a
Linux iZ2zebrh6ffesjwecx7eq9Z 4.15.0-128-generic #131-Ubuntu SMP Wed Dec 9 06:57:35 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
root@iZ2zebrh6ffesjwecx7eq9Z:~# uname -r
4.15.0-128-generic
root@iZ2zebrh6ffesjwecx7eq9Z:~# uname -m
x86_64
```

`vmstat` 查看系统性能

- `vmstat`：显示系统性能信息。
- `vmstat 1`：每秒显示一次系统性能信息。
- `vmstat 1 10`：每秒显示一次系统性能信息，共显示 10 次。`Ctrl + C` 退出。

```
root@iZ2zebrh6ffesjwecx7eq9Z:~# vmstat 1 10
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0      0 187608 218620 1918756    0    0     0     8    0    0  1  0 99  0  0
 0  0      0 187736 218620 1918792    0    0     0     0  457  568  1  0 99  0  0
 0  0      0 187864 218620 1918804    0    0     0     0  493  579  1  0 99  0  0
 0  0      0 187992 218620 1918804    0    0     0     0  378  489  1  0 99  0  0
 0  0      0 187992 218620 1918808    0    0     0    20  234  404  0  1 100  0  0
^C
```

`free` 查看内存使用

- `free`：显示内存使用情况。
- `free -h`：以易读的格式显示内存使用情况。
- `free -m`：以 MB 为单位显示内存使用情况。

```
root@iZ2zebrh6ffesjwecx7eq9Z:~# free -m
              total        used        free      shared  buff/cache   available
Mem:           3757        1487         145           7        2125        1981
Swap:             0           0           0
```

`env` 查看环境变量

```
~/Documents/GitHub/HelloWorld git:(master)±132 (0.082s)
env
```

```
COLORTERM=truecolor
COMMAND_MODE=unix2003
HOME=/Users/maweiqing
LANG=en_US.UTF-8
LOGNAME=maweiqing
LaunchInstanceID=3E03C6D7-3FDB-4CE7-9B7D-43C6164572DB
PATH=/usr/local/scala/bin:/Users/maweiqing/.jenv/shims:/Users/maweiqing/.jenv/bin:/usr/local/bin:/u
sr/local/sbin:/Library/Frameworks/Python.framework/Versions/3.9/bin:/usr/local/bin:/usr/bin:/bin:/u
sr/sbin:/sbin:/usr/local/share/dotnet:~/.dotnet/tools:/Library/Apple/usr/bin:/Library/Frameworks/Mo
no.framework/Versions/Current/Commands:/Applications/Xamarin Workbooks.app/Contents/SharedSupport/p
ath-bin:/usr/local/bin
SECURITYSESSIONID=186a5
```

`date` 查看系统时间

```
root@iZ2zebrh6ffesjwecx7eq9Z:~# date
Fri Feb 23 16:19:36 CST 2024
```

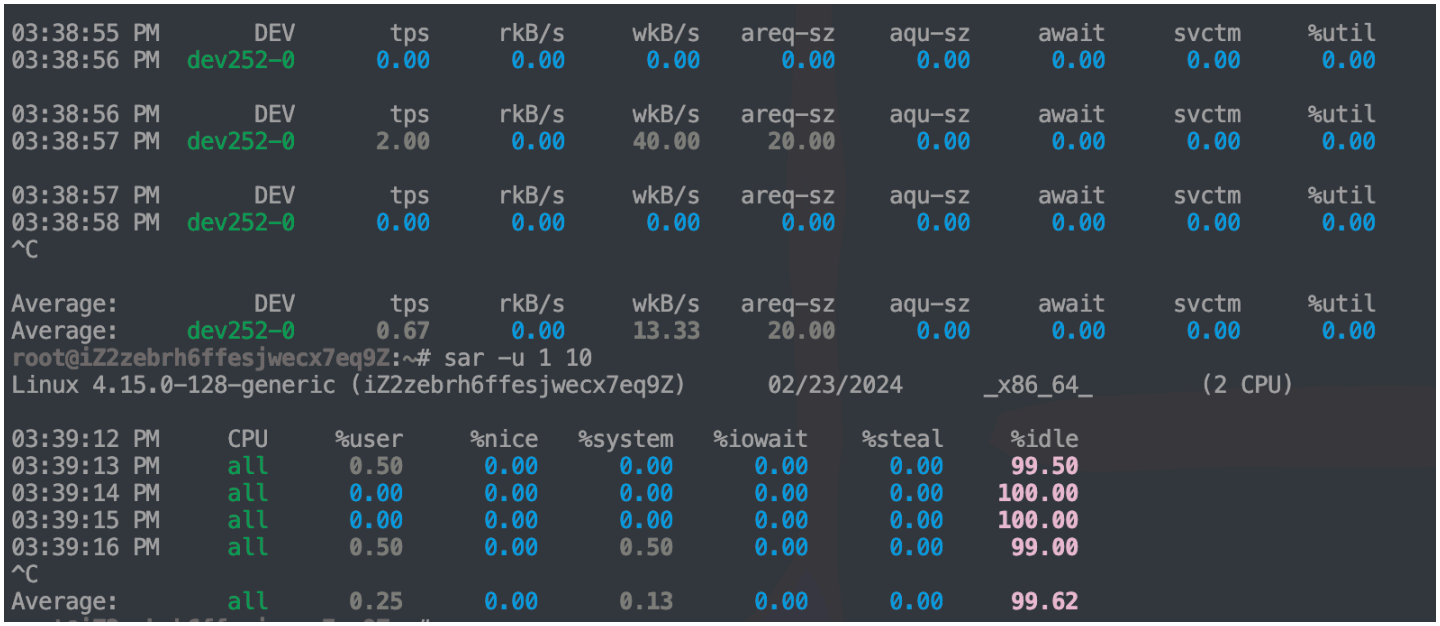
`iostat` 查看磁盘性能

```
~ git:(master)±132 (0.261s)
iostat
              disk0               disk1       cpu    load average
    KB/t  tps  MB/s     KB/t  tps  MB/s  us sy id   1m   5m   15m
   13.32   47  0.61    37.65   12  0.46   4  2 94  3.13 2.72 2.84
```

`sar` 查看系统负载

- `sar -u 1 10`：每秒显示一次 CPU 使用情况，共显示 10 次。`Ctrl + C` 退出。
- `sar -d 1 10`：每秒显示一次磁盘 I/O 使用情况，共显示 10 次。`Ctrl + C` 退出。

```
root@iZ2zebrh6ffesjwecx7eq9Z:~# sar -d 1 10
Linux 4.15.0-128-generic (iZ2zebrh6ffesjwecx7eq9Z)      02/23/2024      _x86_64_        (2 CPU)
```



表格总结

好，继续用一张表格总结一下（贴心到你会觉得自己就是个天之骄子 😂）：

命令	说明

`uname`	查看系统信息
`uname -a`	查看内核/OS/CPU 信息
`uname -r`	查看内核版本
`uname -m`	查看处理器架构
`hostname`	查看主机名
`who`	查看当前登录用户
`whoami`	查看当前用户名
`cat /proc/version`	查看内核版本
`cat /proc/cpuinfo`	查看 CPU 信息
`uptime`	查看系统运行时间和平均负载
`vmstat`	查看系统性能
`vmstat 1`	每秒显示一次系统性能信息
`vmstat 1 10`	每秒显示一次系统性能信息，共显示 10 次
`free`	查看内存使用
`free -h`	以易读的格式显示内存使用情况
`free -m`	以 MB 为单位显示内存使用情况
`env`	查看环境变量
`date`	查看系统时间
`iostat`	查看磁盘性能
`sar -u 1 10`	每秒显示一次 CPU 使用情况，共显示 10 次
`sar -d 1 10`	每秒显示一次磁盘 I/O 使用情况，共显示 10 次

包管理器

> 作用上就相当于 App Store，或者以前的 360 流氓管家，或者手机上的应用商店，用来在 Linux 系统上安装、升级、卸载软件包。

RPM 包管理

RPM（Red Hat Package Manager）是一种用于在 Red Hat、Fedora 和 CentOS 等 Linux 发行版上安装、升级、卸载软件包的工具。

- `rpm -qa`：列出所有已安装的软件包。
- `rpm -qi package`：显示软件包的详细信息。
- `rpm -ql package`：列出软件包的文件列表。
- `rpm -qf /path/to/file`：查找文件属于哪个软件包。
- `rpm -Uvh package.rpm`：升级软件包。
- `rpm -e package`：卸载软件包。

YUM 包管理

YUM（Yellowdog Updater, Modified）是一个在 Fedora 和 CentOS 等 Linux 发行版上的软件包管理器，它可以自动下载和安装软件包及其依赖的软件包。

- `yum list`：列出所有可用的软件包。
- `yum list installed`：列出所有已安装的软件包。
- `yum search keyword`：搜索软件包。
- `yum info package`：显示软件包的详细信息。
- `yum install package`：安装软件包。
- `yum update package`：升级软件包。
- `yum remove package`：卸载软件包。
- `yum clean all`：清除缓存。

APT 包管理

APT（Advanced Package Tool）是一个在 Debian 和 Ubuntu 等 Linux 发行版上的软件包管理器，它可以自动下载和安装软件包及其依赖的软件包。

- `apt list`：列出所有可用的软件包。
- `apt list --installed`：列出所有已安装的软件包。
- `apt search keyword`：搜索软件包。
- `apt show package`：显示软件包的详细信息。
- `apt install package`：安装软件包。
- `apt update`：更新软件包列表，在 `apt install` 之前执行。
- `apt upgrade`：升级软件包。

- `apt remove package`：卸载软件包。
- `apt [illegible]`：卸载不再需要的软件包

- `apt autoremove`：卸载不再需要的软件包。
- `apt clean`：清除缓存。

apt 速查表下载地址：https://opensource.com/downloads/apt-cheat-sheet

表格总结

表格不能少，绝不敷衍（贴心到你会觉得自己不配做个废柴 😂）：

命令	说明

`rpm -qa`	列出所有已安装的软件包
`rpm -qi package`	显示软件包的详细信息
`rpm -ql package`	列出软件包的文件列表
`rpm -qf /path/to/file`	查找文件属于哪个软件包
`rpm -Uvh package.rpm`	升级软件包
`rpm -e package`	卸载软件包
`yum list`	列出所有可用的软件包
`yum list installed`	列出所有已安装的软件包
`yum search keyword`	搜索软件包
`yum info package`	显示软件包的详细信息
`yum install package`	安装软件包
`yum update package`	升级软件包
`yum remove package`	卸载软件包
`yum clean all`	清除缓存
`apt list`	列出所有可用的软件包
`apt list --installed`	列出所有已安装的软件包
`apt search keyword`	搜索软件包
`apt show package`	显示软件包的详细信息
`apt install package`	安装软件包
`apt update`	更新软件包列表
`apt upgrade`	升级软件包
`apt remove package`	卸载软件包
`apt autoremove`	卸载不再需要的软件包
`apt clean`	清除缓存

花絮

该 PDF 文档是我在学习 Linux 时的一些积累，如果遇到更好玩的命令也会往里面添加。整理不易，希望能帮助到大家（❤️）。

最新版更新完成后我会放到网盘中，微信搜索《**沉默王二**》或者微信扫下面的二维码，关注后回复《**Linux**》即可获取最新的 PDF 版本。

附其他干货笔记下载地址：

- 阮一峰 C 语言入门教程 PDF 下载
- Java 核心知识点整理 PDF 下载
- 深入浅出 Java 多线程 PDF 下载
- Pro Git 中文版 PDF 下载
- 给操作系统捋条线 PDF 下载