

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

`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/maweqing
```

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/maweqing/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`: 递归复制目录及其内容。

```
~/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。

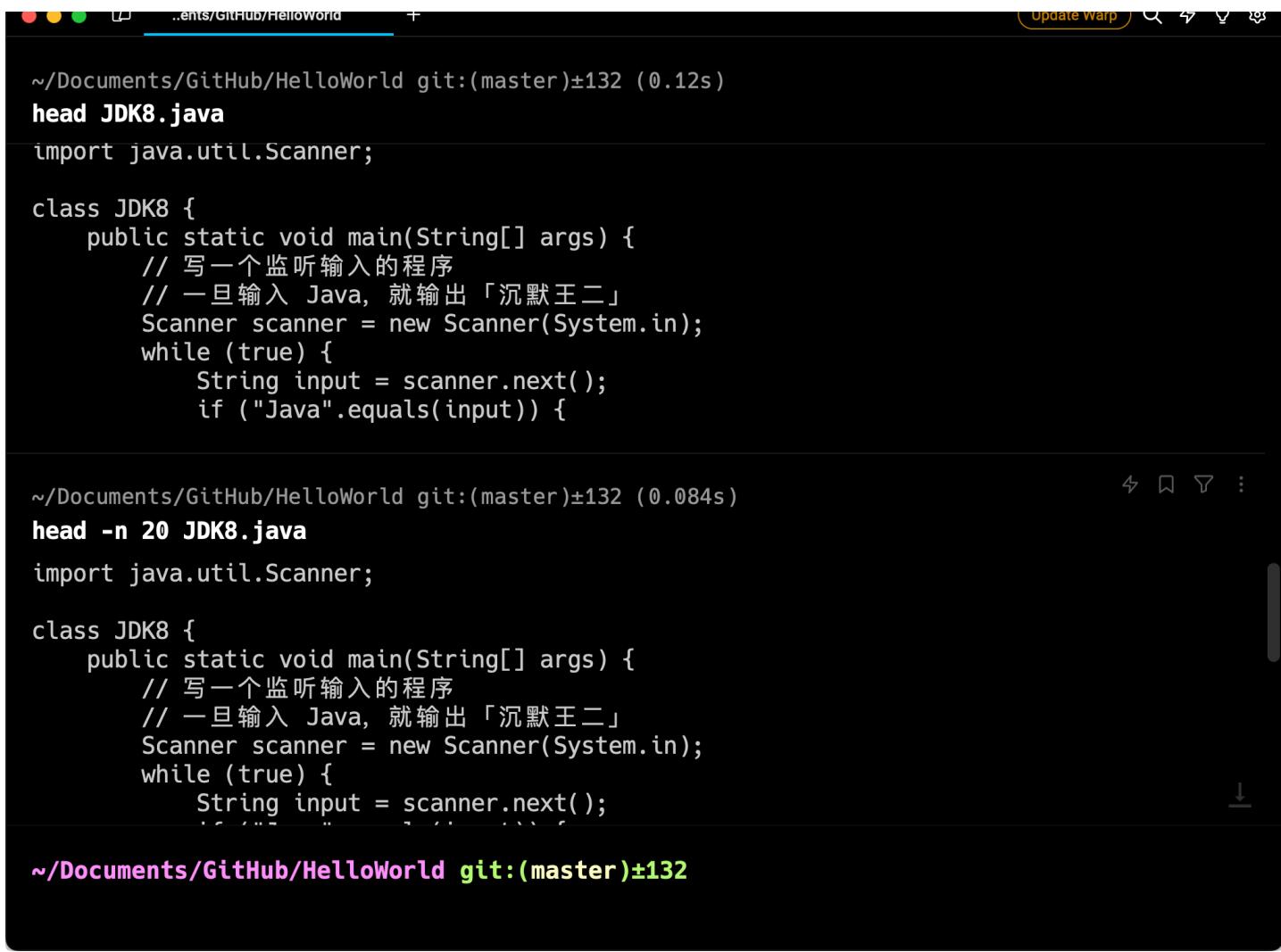
echo 输出文本

- `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。



~/Documents/GitHub/HelloWorld git:(master)±132 (0.12s)
head JDK8.java
import java.util.Scanner;

class JDK8 {
 public static void main(String[] args) {
 // 写一个监听输入的程序
 // 一旦输入 Java, 就输出「沉默王二」
 Scanner scanner = new Scanner(System.in);
 while (true) {
 String input = scanner.next();
 if ("Java".equals(input)) {

~/Documents/GitHub/HelloWorld git:(master)±132 (0.084s)
head -n 20 JDK8.java
import java.util.Scanner;

class JDK8 {
 public static void main(String[] args) {
 // 写一个监听输入的程序
 // 一旦输入 Java, 就输出「沉默王二」
 Scanner scanner = new Scanner(System.in);
 while (true) {
 String input = scanner.next();

~/Documents/GitHub>HelloWorld git:(master)±132

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。

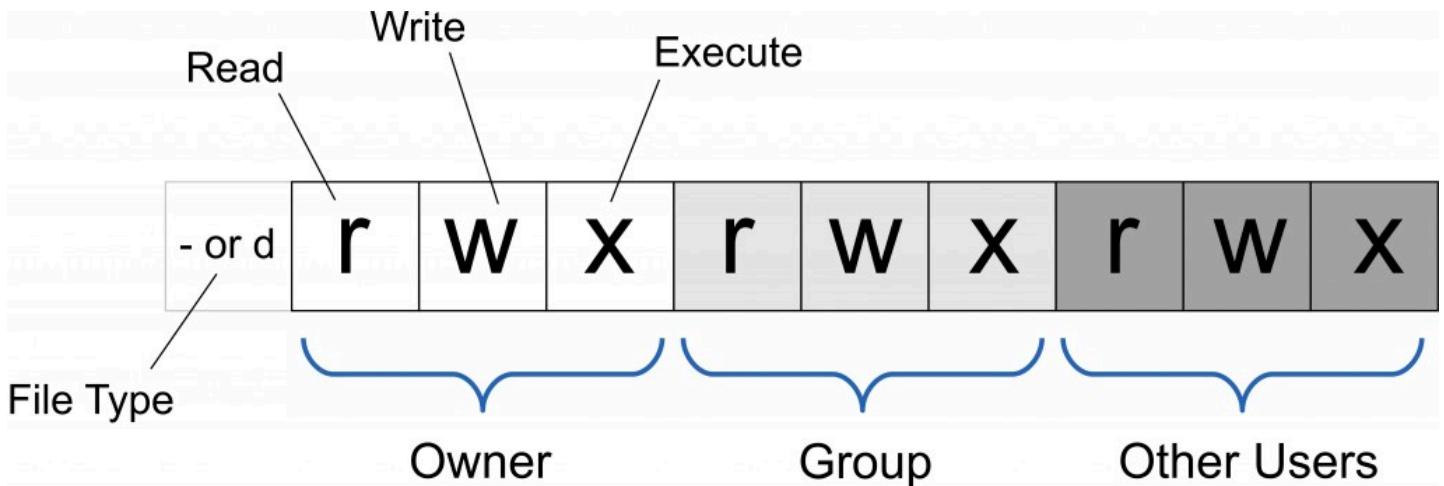
vim 是一个非常强大的文本编辑和查看工具，我这里只列出了一些最基本的操作，更多的操作可以参考[《Vim 学习路线》](#)。

chmod 修改文件权限

- `chmod 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：

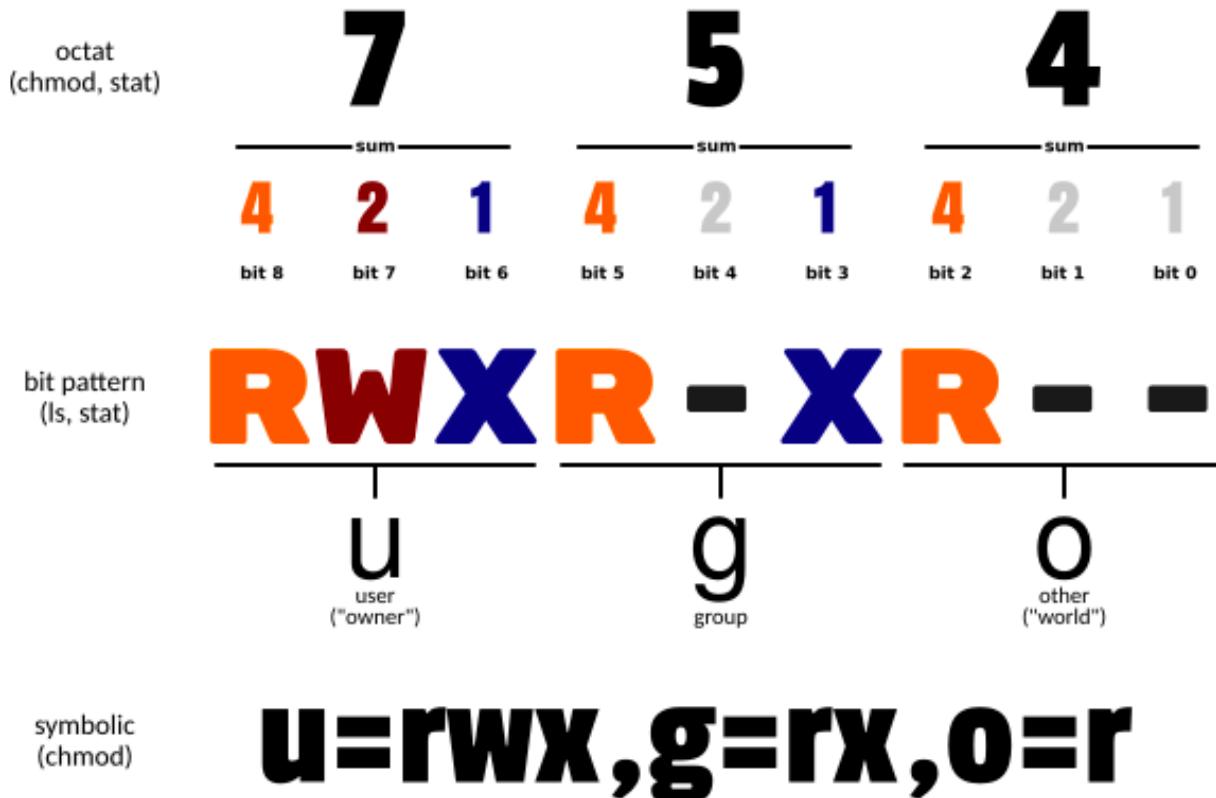
- 读 (r)
- 写 (w)
- 执行 (x)
- 所有 (a)

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。

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：

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。

zin

--r

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。

```
~/Documents/GitHub/linux-pdf git:(master) ±132 (0.171s)
tree

.
├── c
│   └── pdf
│       └── hello.wanger
├── c1
│   └── pdf
│       ├── hello.wanger
│       ├── test.md
│       └── wanger.hello
└── java
    └── hello.wanger
        └── javabetter.pdf

5 directories, 7 files

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.083s)
tree -L 2

.
├── c
│   └── pdf
├── c1
│   └── pdf
└── java
    └── hello.wanger
        └── javabetter.pdf

5 directories, 2 files

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.085s)
tree -d

.
├── c
│   └── pdf
├── c1
│   └── pdf
└── java

5 directories

~/Documents/GitHub/linux-pdf git:(master) ±132 (0.087s)
tree -a

.
├── c
│   └── pdf
│       └── hello.wanger
├── c1
│   └── pdf
│       ├── hello.wanger
│       ├── test.md
│       └── wanger.hello
└── java
    └── hello.wanger
        └── javabetter.pdf

5 directories, 7 files
```

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
```

```
lina / var/log -type f -size 10M
```

```
~/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-doc-
ker-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
-rw-r--r-- 1 maweiqing staff 48 Feb 14 16:32 javabetter.pdf
```

whereis 搜索命令

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
```

网络和进程管理

ping 测试网络连通性

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，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 2       00:50:30 java -jar codingmore-web-1.0-SNAPSHOT.jar
```

```
root 29442 1 0 2022 ? 09:50:50 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。

The screenshot shows the output of the `top` command. At the top, it displays system statistics: Processes (645 total, 2 running, 643 sleeping, 3628 threads), Load Avg (2.57, 2.86, 3.00), CPU usage (1.80% user, 3.50% sys, 94.69% idle), SharedLibs (598M resident, 85M data, 261M linkedit), MemRegions (690507 total, 12G resident, 310M private, 5002M shared), PhysMem (31G used (2932M wired), 1009M unused), VM (72T vsize, 3152M framework vsize, 0(0) swapins, 0(0) swapouts), Networks (packets: 5653461/4878M in, 4387641/2781M out), and Disks (5000887/89G read, 5200228/91G written). Below this, a detailed table lists processes by PID, command, %CPU, TIME, #TH, #WQ, #PORT, MEM, PURG, CMPRS, PGRP, and PPID.

| PID | COMMAND | %CPU | TIME | #TH | #WQ | #PORT | MEM | PURG | CMPRS | PGRP | PPID |
|-------|--------------|----------|----------|-------|------|-------|--------|-------|-------|-------|-------|
| 82137 | top | 10.9 | 00:01.28 | 1/1 | 0 | 26+ | 6172K+ | 0B | 0B | 82137 | 75805 |
| 169 | WindowServer | 3.0 | 80:35.88 | 13 | 5 | 3059 | 830M | 49M+ | 13M | 169 | 1 |
| 0 | kernel_task | 1.4 | 73:48.54 | 229/8 | 0 | 0 | 32M+ | 0B | 0B | 0 | 0 |
| 33222 | QA0Z微01.2 | 09:28.98 | 78 | 3 | 8087 | 340M | 240K | 44M | 33222 | 1 | |
| 550 | stable | 1.10 | 00:04.14 | 40 | 6 | 349+ | 219M | 28K | 0B | 550 | 1 |
| 48639 | bluetoothd | 1.0 | 02:05.30 | 9 | 4 | 272 | 4816K | 48K | 4096B | 48639 | 1 |
| 46924 | idea | 0.7 | 26:21.23 | 87 | 2 | 514 | 2769M | 0B | 430M | 46924 | 1 |
| 686 | Creative Clo | 0.6 | 03:26.86 | 24 | 3 | 387 | 44M | 0B | 14M | 686 | 1 |
| 724 | Adobe_CCXPro | 0.5 | 00:55.66 | 19 | 2 | 90 | 39M | 0B | 1056K | 724 | 1 |
| 514 | WeChat | 0.3 | 18:54.05 | 57 | 11 | 1492- | 767M- | 1072K | 128M | 514 | 1 |
| 142 | ToDesk_Servi | 0.3 | 05:50.30 | 15 | 1 | 197 | 26M | 0B | 8192B | 142 | 1 |
| 579 | UniversalCon | 0.2 | 01:17.88 | 2 | 1 | 122 | 8604K | 0B | 32K | 579 | 1 |
| 2185 | Adobe CEF He | 0.2 | 07:05.08 | 14 | 1 | 231 | 101M+ | 0B | 107M | 686 | 686 |
| 82125 | mdworker_sha | 0.2 | 00:00.03 | 4 | 2 | 57+ | 1304K+ | 0B | 0B | 82125 | 1 |
| 752 | beam.smp | 0.1 | 04:38.82 | 32 | 0 | 54 | 68M- | 0B | 26M | 675 | 675 |
| 673 | mongod | 0.1 | 03:30.14 | 33 | 0 | 56 | 20M | 0B | 8312K | 673 | 1 |
| 68884 | Google Chrom | 0.1 | 00:13.91 | 14 | 1 | 342 | 204M+ | 128K | 0B | 500 | 500 |
| 1870 | Adobe CEF He | 0.1 | 03:48.82 | 8 | 1 | 192 | 78M | 0B | 11M | 686 | 686 |
| 574 | BuhoCleanerM | 0.1 | 02:54.22 | 4 | 1 | 185 | 15M | 56K | 796K | 574 | 1 |
| 105 | FocusriteCon | 0.1 | 03:01.43 | 10 | 2 | 66 | 1516K | 0B | 4096B | 105 | 1 |
| 29343 | Electron | 0.1 | 07:43.30 | 48 | 3 | 525 | 98M | 84K | 18M | 29343 | 1 |

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。

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 |
|------------------------------------|----------|
| <code>service 服务名 status</code> | 查看某个服务状态 |
| <code>service 服务名 start</code> | 启动某个服务 |
| <code>service 服务名 stop</code> | 停止某个服务 |
| <code>service 服务名 restart</code> | 重启某个服务 |
| <code>systemctl status 服务名</code> | 查看某个服务状态 |
| <code>systemctl start 服务名</code> | 启动某个服务 |
| <code>systemctl stop 服务名</code> | 停止某个服务 |
| <code>systemctl restart 服务名</code> | 重启某个服务 |
| <code>systemctl enable 服务名</code> | 设置开机启动 |
| <code>systemctl disable 服务名</code> | 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
```

```

● 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.
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

```

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----- swan----io----system---cpu-----

```

| | | 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 | | | |

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。

| | 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:/usr/local/sbin:/Library/Frameworks/Python.framework/Versions/3.9/bin:/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr/local/share/dotnet:~/dotnet/tools:/Library/Apple/usr/bin:/Library/Frameworks/Mono.framework/Versions/Current/Commands:/Applications/Xamarin Workbooks.app/Contents/SharedSupport/path-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 | | | 02/23/2024 | | | _x86_64_ | | | (2 CPU) | | |
|---|----------|-------|------------|---------|---------|----------|--------|-------|---------|--|--|
| 03:38:55 PM | DEV | tps | rkB/s | wkB/s | areq-sz | aqu-sz | await | svctm | %util | | |
| 03:38:56 PM | dev252-0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | | |
| 03:38:56 PM | DEV | tps | rkB/s | wkB/s | areq-sz | aqu-sz | await | svctm | %util | | |
| 03:38:57 PM | dev252-0 | 2.00 | 0.00 | 40.00 | 20.00 | 0.00 | 0.00 | 0.00 | 0.00 | | |
| 03:38:57 PM | DEV | tps | rkB/s | wkB/s | areq-sz | aqu-sz | await | svctm | %util | | |
| 03:38:58 PM | dev252-0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | | |
| ^C | | | | | | | | | | | |
| Average: | DEV | tps | rkB/s | wkB/s | areq-sz | aqu-sz | await | svctm | %util | | |
| Average: | dev252-0 | 0.67 | 0.00 | 13.33 | 20.00 | 0.00 | 0.00 | 0.00 | 0.00 | | |
| root@iZ2zebrh6ffesjwecx7eq9Z:~# sar -u 1 10 | | | 02/23/2024 | | | _x86_64_ | | | (2 CPU) | | |
| 03:39:12 PM | CPU | %user | %nice | %system | %iowait | %steal | %idle | | | | |
| 03:39:13 PM | all | 0.50 | 0.00 | 0.00 | 0.00 | 0.00 | 99.50 | | | | |
| 03:39:14 PM | all | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 100.00 | | | | |
| 03:39:15 PM | all | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 100.00 | | | | |
| 03:39:16 PM | all | 0.50 | 0.00 | 0.50 | 0.00 | 0.00 | 99.00 | | | | |
| ^C | | | | | | | | | | | |
| Average: | all | 0.25 | 0.00 | 0.13 | 0.00 | 0.00 | 99.62 | | | | |

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 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>

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](#)