准备工作

```
# 切换目录到 /export/ 且 清空内容
cd /export/ && rm -rf /export/* && tree

# 新增目录 且 切换目录 且 新增文件
mkdir -p /export/aaa/bbb/ccc/ddd/eee/
touch /export/aaa/bbb/ccc/ddd/eee/abc.txt
touch /export/aaa/123.txt
touch /export/aaa/312.txt

# 查看 /export 目录中的内容
tree /export
```

搜索命令

1.find 搜索

1.1 目标

- 通过`find`命令 **在特定目录下(包含它的后代目录)** 搜索 **符合条件**的文件

1.2 路径

- 第一步: 搜索指定目录下, 文件是 abc.txt的文件
- 第二步: 搜索指定目录下, 文件名 包含 `1` 的文件
- 第三步: 搜索指定目录下,所有以`.txt` 为扩展名的文件
- 第四步: 搜索指定目录下, 以数字`1`开头的文件

1.3 实现 : 在指定目录中 根据名称 搜索

- **命令格式**

序号	命令格式	作用
01	find [路径] -name '*.txt'	查找指定路径下扩展名是 .txt的文件, 包括子目录

- 如果省略路径, 表示在当前文件夹下查找
- 之前学习的通配符, 在使用find命令时同时可用

- **第一步: 搜索指定目录下, 文件是 abc.txt的文件**

```
# ===================实现目标============================
# 方式一: 指定全目录
find /export/ -name 'abc.txt'
# 方式二: 当前目录
find . -name 'abc.txt'
# 方式三: 当前目录 可以 省略 不写
find -name 'abc.txt'
```

- **第二步: 搜索指定目录下, 文件名 包含 `1` 的文件**

```
# =================准备工作===============
# 1 创建测试文件
touch /export/12.txt /export/616.txt /export/321.txt

# ================实现目录===============
find /export/ -name "*1*"
```

- **第三步: 搜索指定目录下,所有以 `.txt` 为扩展名的文件**

```
find /export/ -name "*.txt"
```

- **第四步: 搜索指定目录下, 以数字 `1` 开头的文件**

```
find /export -name "1*"
```

1.3 小结

- 通过 `find [path] -name "*1*"` 完成根据名称搜索文件

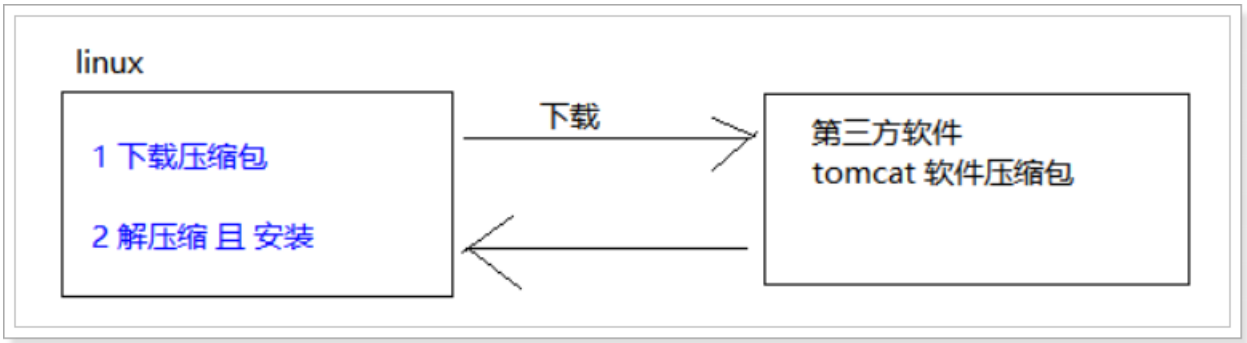
解压缩命令

- 准备工作

```
# 清空指定目录内容
cd /export/ && rm -rf *

# 创建测试文件
touch 1.txt 2.txt 3.txt

# 创建有内容的测试目录
mkdir -p /export/aaa/
touch /export/aaa/4.txt /export/aaa/5.txt

# 查看结果
tree /export
```

1.目标

- 从第三方 **下载** 压缩包, **解压缩后** 安装到 服务器上



- 通过 **打包压缩** 备份文件

jar : java rar java项目的压缩包

war : web rar web项目的压缩包

2 路径

- **2.1 第一步: 打包 和 解包**
- **2.2 第二步: 打包压缩 和 解包解压缩**

3.实现

3.1 第一步: 打包 和 解包

3.1.1 打包

- **类似将 冬天的衣服 放到 袋**
- 打包之后的大文件 需要以 `.tar` 结尾.

`tar` **打包**命令格式

```
# 将 一系列文件 打包成 一个大文件
tar -cvf 打包名.tar 被打包的目录
tar -cvf 打包名.tar 被打包的文件1 被打包的文件2 被打包的文件3
```

`tar` 选项说明

命令	英文	含义
c	create	生成档案文件, 创建打包文件
v	verbosely(啰嗦的)	像 '唐僧' 一样报告进度
f	file	指定档案的文件名称, f后面一定是 `.tar` 文件, 所以必须放到左后

```
练习1：将1.txt、2.txt、3.txt 打包成 123.tar文件

练习2：将有内容的aaa目录 打包成 aaa.tar 文件
```

3.1.2 解包

- **类似将 冬天的衣服 从 袋子里取出来**

`tar` **解包**命令格式

```
# 将一个打包后的 分解成 一系列小文件，分解位置为 当前目录
tar -xvf 打包名.tar

# 将一个打包后的 分解成 一系列小文件，分解位置为 指定目录
tar -xvf 打包名.tar -C 解包路径位置
```

命令	英文	含义
x	extract (提取)	解包
C (大写C)	directory (目录)	默认保存到当前目录, 通过`-C`更改解压目录, 注意: 解压目录必须存在

```
练习1：将 123.tar 解压到 当前目录中

练习2：将 aaa.tar 解包到 /export/test/a1/b1/c1/ 目录中
```

小结

```
打包：tar -cvf 打包之后的文件名.tar    被打包的目录或文件名

解包：tar -xvf 打包之后的文件名.tar  [ -C  指定解包位置 ]
```

- 准备工作

```
# 清空指定目录内容
cd /export/ && rm -rf *

# 创建测试文件
touch 1.txt 2.txt 3.txt

# 创建有内容的测试目录
mkdir -p /export/aaa/
touch /export/aaa/4.txt /export/aaa/5.txt

# 查看结果
tree /export
```

3.2 第二步: 使用gzip格式 压缩 和 解压缩

- **打包** 和 **压缩** 是两件事
- **类似与 先将冬天衣服放到压缩袋, 再抽取里面的空气**
- 在 `Linux` 中, 最常用的压缩文件格式是 `xxx.tar.gz`
- 在 `tar` 命令中有一个选项 -z 可以调用 `gzip` , 从而可以方便的实现压缩和解压缩的功能

命令格式如下

```
# 压缩文件
tar -zcvf 打包压缩文件名.tar.gz 被压缩的文件/目录

# 解压缩文件(记忆敲门: 至孝潍坊)
tar -zxvf 打包文件.tar.gz

# 解压缩到指定路径
tar -zxvf 打包文件.tar.gz -C 目录路径
```

`tar` 的选项说明

命令	英文	含义
z	gzip	使用gzip压缩和解压缩
j	bzip2	使用bzip2压缩和解压缩

```
练习1: 将1.txt、2.txt、3.txt 打包压缩成 123.tar.gz文件(gzip压缩格式)

练习2: 将有内容的aaa目录 打包成 aaa.tar.gz 文件(gzip压缩格式)

练习3: 将 123.tar.gz 解压到 当前目录中(gzip压缩格式)

练习4: 将 aaa.tar.gz 解包到 /export/bbb 目录中(gzip压缩格式)
```

小结

```
打包压缩: tar -zcvf 打包之后的文件名.tar.gz   被打包压缩的目录或文件名

解包解压缩: tar -zxvf 打包之后的文件名.tar.gz  [ -C  指定解包位置 ]
```

- 准备工作

```
# 清空指定目录内容
cd /export/ && rm -rf *

# 创建测试文件
touch 1.txt 2.txt 3.txt

# 创建有内容的测试目录
mkdir -p /export/aaa/
touch /export/aaa/4.txt /export/aaa/5.txt

# 查看结果
tree /export
```

3.3 第三步: 使用 bzip2 格式 压缩 和 解压缩

- `bzip` 是压缩的第二种方式
- **类似与 先将冬天衣服放到压缩袋, 再抽取里面的空气**
- 在 `Linux` 中, bzip2 压缩文件格式是 `xxx.tar.bz2`
- 在 `tar` 命令中有一个选项 -j 可以调用 `bzip2` , 从而可以方便的实现压缩和解压缩的功能

命令格式如下

```
# 压缩文件
tar -jcvf 打包压缩文件名.tar.bz2 被压缩的文件/目录

# 解压缩文件
tar -jxvf 打包文件.tar.bz2

# 解压缩到指定路径
tar -jxvf 打包文件.tar.bz2 -C 目录路径
```

`tar` 的选项说明

命令	英文	含义
z	gzip	使用gzip压缩和解压缩
j	bzip2	使用bzip2压缩和解压缩

```
练习1：将1.txt、2.txt、3.txt 打包压缩成 123.tar.bz2文件(bzip2压缩格式)

练习2：将有内容的aaa目录 打包成 aaa.tar.bz2 文件(bzip2压缩格式)

练习3：将 123.tar.bz2 解压到 当前目录中(bzip2压缩格式)

练习4：将 aaa.tar.bz2 解包到 /export/bbb 目录中(bzip2压缩格式)
```

小结

```
打包压缩：tar -jcvf 打包之后的文件名.tar.bz2   被打包压缩的目录或文件名

解包解压缩：tar -jxvf 打包之后的文件名.tar.bz2  [ -C  指定解包位置 ]
```

VI编辑器_终端编辑器

目标

1. vi简介
2. 打开和新建文件
3. 三种工作模式
4. 常用命令查询

1 简介

1.1 学习vi的目的

- 在工作中, 要对 服务器上的 **文件**进行 简单 的修改, 可以使用 ssh 登录到远程服务器上, 并且使用 vi编辑器 进行简单的编辑即可
- 需要修改的文件包括
  - 源代码
  - 配置文件
    - 例如: tomcat服务器的配置文件
    - 例如: 网卡信息的配置文件

  在没有图形界面的环境下, 要编辑文件, vi是最佳选择 每一个使用linux的程序员,都应该或多或少的学习一些vi的常用命令

1.2 vi 和 vim

- 在很多linux发行版中, 直接把vi做成vim的 软连接

1.2.1 VI

- vi是 `visual interface` 的简称, 是linux中**最经典**的文本编辑器
- vi 的**核心设计思想**
  - **让程序员的手指始终保持在键盘的 核心区域, 就能完成所有的编辑操作**

  -
- vi的特点
  - 只能是编辑 **文本内容**, 不能对 字体 段落进行排版
  - **不支持鼠标操作**
  - **没有菜单**
  - **只有命令**
- vi编辑器在 **系统管理 服务器管理** 编辑文件时, **其功能永远不是图形界面的编辑器能比拟的**

1.2.2 VIM

vim = vi improved viM 是从vi发展出来的文本编辑器, 支持**代码补全、编译 及 错误跳转**等方便编程的功能提别丰富, 在程序员中被广泛使用, 被称为**编辑器之神**

2 打开和新建文件

- 在终端中输入vim在后面跟上 文件名 即可

```
vim 文件名
```

- 如果文件已经存在, 会直接打开该文件
- 如果文件不存在, 保存且退出时 就会新建一个文件

```
1 vim 没有的文件名
2 编辑内容
    2.1 编写类
    2.2 输出一行语句
    2.3 复制粘贴 2.2内容 19次
    2.4 保存且退出
3 查看
```

注意: 本节重点是 打开 和 新建文件, 其他命令后面会一一讲解

2.1 打开文件并且定位行

- 在日常工作中, 有可能会遇到打开一个文件, 并定位到指定行的情况
- 例如: 在开发是, 知道某一行代码有错误, 可以 快速定位 到出错误代码的位置
- 这个时候, 可以使用以下命令打开文件

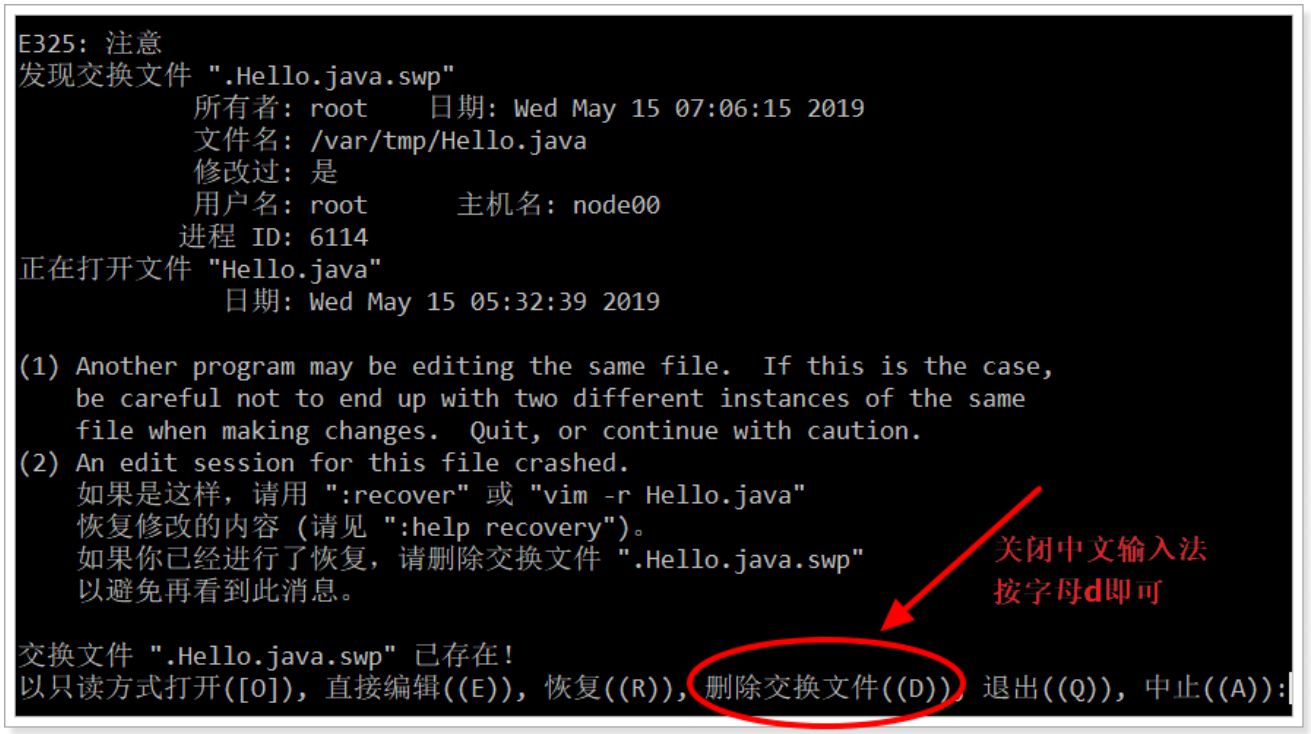
```
vim 文件名 +行数
```

> 提示: 如果只带上+ 而不指定行号, 会直接定位到文件末尾

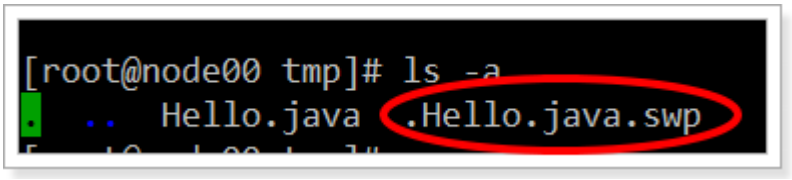
2.2 异常处理

- 如果 vi 异常退出, 在磁盘上可能会保存有 交换文件
- 下次再使用 vi 编辑文件时, 会看到以下屏幕信息, 按下字母 d 删除交换文件即可

> 提示: 按下键盘时, 注意关闭输入法



- 如何产生
  - 编辑文件, 没有退出, 关闭回话窗口

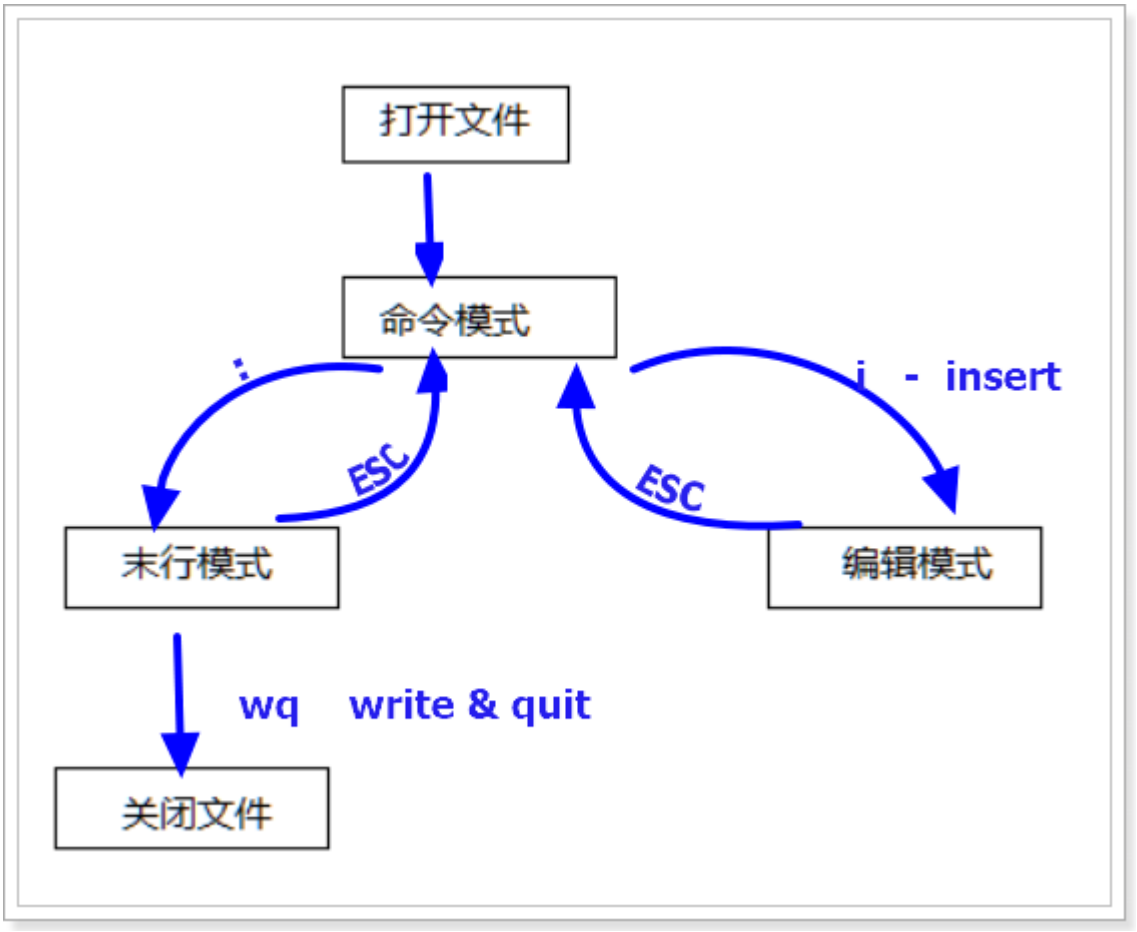


2.3 VI三种工作模式

- VI有三种工作模式

1. 命令模式
   - **打开文件首先进入命令模式**, 是使用vi的**入口**
   - 通过 **命令** 对文件进行常规的编辑操作, 例如 **定位 翻页 复制 粘贴 删除** ... ...
   - 在其他图形编辑器下, 通过 快捷键 或者 鼠标 实现的操作, 都在 **命令模式** 下实现
2. 末行模式 -- 执行 **保存 退出**等操作
   - 要退出 vi 返回到控制台, 需要在莫行模式下输入命令
   - **末行模式** 是 vi 的**出口**
3. **编辑模式** -- 正常的编辑文字

工作模式切换



2.4 末行模式命令

命令	英文	功能
w	write	保存
q	quit	退出,如果没有保存,不允许退出
q!	quit	强行退出,不保存退出
wq	write & quit	保存且退出
x		保存并退出
ZZ		保存并退出

4 常用命令

命令线路图

1. 重复次数

- 在命令模式下, **先输入一个数字, 再加上一个命令**, 可以让该命令 重复执行n次, 如: 19p

1. 移动和选择(**多练**) vi之所以快, 关键在于 **能够快速定位到要编辑的代码行 移动命令** 能够和 **编辑操作** 组合使用

2. 编辑操作
    - **删除 复制 粘贴 替换 缩排**
3. 撤销和重复
4. 查到替换
5. 编辑

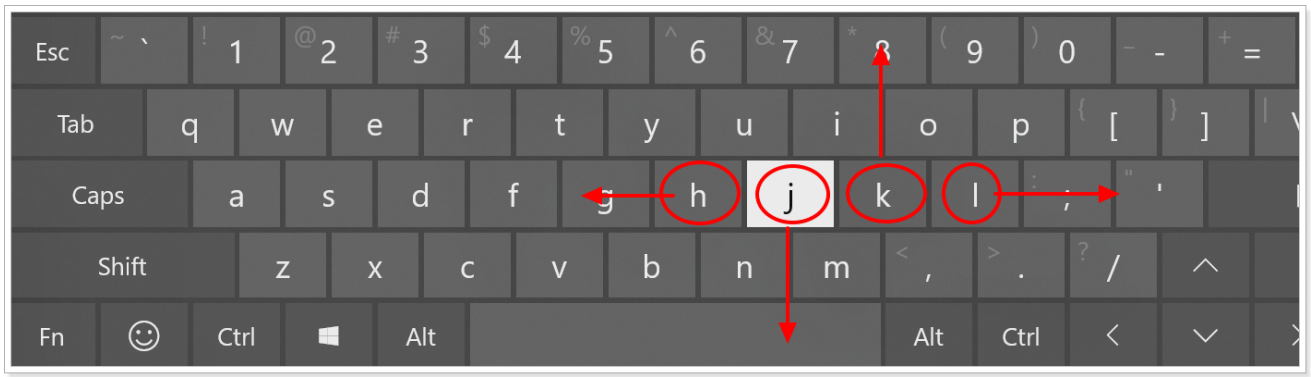
学习提示

1. vi命令较多, **不要期望一下子全部都记住**, 个别命令忘记了, 只是会影响编辑速度而已
   . 在使用vi命令时, **注意 关闭中文输入法**

4.1 移动

要熟练使用vi, 首先应该学会怎么在 命令模式下 快速移动光标 编辑操作命令 能够和移动命令一起使用

1) 上 下 左 有



命令	功能	手指
h	向左	食指
j	向下	食指
k	向上	中指
l	向右	无名指

2) 行内移动

命令	英文	功能
w	word	向后移动一个单词
b	back	向前移动一个单词
0		行首
^		行首, 第一个不是空白字符的位置
$		行尾

3) 行数移动

命令	英文	功能
gg	go	文件顶部
G	go	文件末尾
数字gg	go	移动到 数字 对应行数
数字G	go	移动到 数字 对应行数
: 数字		移动到数字对应的 行数

4) 屏幕移动

命令	英文	功能
Ctrl + b	back	向上翻页
Ctrl + f	forward	向下翻页
H	Head	屏幕顶部
M	Middle	屏幕中间
L	Low	L部

4.2 移动(程序)

1) 段落移动

- vim中使用 空行 来区分段落
- 程序开发时, 通常 **一段功能相关的代码会写在一起** -- 之间没有空行

命令	功能
{	上一段
}	下一段

2) 括号切换

- 在程序世界中, **()、[]、{}**使用率很高, 而且 **都是成对出现的.**

命令	功能
%	括号匹配及切换

3) 标记

- 在开发时, **某一块代码可能需要稍后处理**, 例如: 编辑、查看
- 此时先试用 m 增加一个标记, 这样可以 **在需要时快速的跳转回来** 或者 **执行其他编辑操作**
- **标记名称** 可以是 a~z 或者 A~Z 之间的任意 一个 字母
- 添加了标记的 行**如果删除**, **标记同时被删除**
- 如果 **在其他行添加了相同名称的标记, 之前添加的标记也会被替换掉**

命令	英文	功能
mx	mark	添加编辑x, x是a~z 或者 A~Z的任意一个字母
'x		直接定位到标记x所在的位置

4.3 选中文本(可视模式)

- 学习 复制 命令前, 应该先学会 **怎么样选中 要复制的代码**
- 在 vi 中要选择文本, 需要显示 visual 命令切换到 **可视模式**
- vi 中提供了 **三种** 可视模式, 可以方便程序员的选择 **选中文本的方式**
- 按 ESC 可以放弃选中, 返回到 **命令模式**

命令	模式	功能
v	可视模式	从光标位置开始按照正常模式选择文本
V	可视化模式	选中光标经过的完整行
Ctrl + v	可是块模式	垂直方向选中文本

- 可视模式下, 可以和 移动命令 连用, 例如 ggVG 能够 选中所有的内容

4.4 撤销和恢复撤销(保命指令)

在学习编辑命令之前,先要知道怎样撤销之前一次 错误的 编辑操作

命令	英文	功能
u	undo	撤销上次的命令(ctrl + z)
Ctrl + r	uredo	恢复撤销的命令

4.5 删除文本

命令	英文	功能
x	cut	删除光标所在的字符,或者选中的文字
d(移动命令)	delete	删除移动命令对应的内容
dd	delete	删除光标所在行, 可以ndd删除多行
D	delete	删除至行尾

提示: 如果使用 可视模式 已经选中了一段文本, 那么无论使用 d 还是 x, 都可以删除选中文本

删除命令可以和移动命令连用, 以下是常见的组合命令:

命令	作用
dw	从光标位置删除到单词末尾
d0	从光标位置删除到一行的起始位置
d}	从光标位置删除到段落末尾
ndd	从光标位置向下连续删除 n 行
d'a	从光标所在行 删除到 标记a 之间的所有代码

4.6 复制和剪切

- vi 中提供有一个 被复制文本的缓冲区
  - 复制 命令会将选中的文字保存在缓冲区
  - 删除 命令删除的文字会被保存在缓冲区
  - 在需要的位置, 使用 粘贴 命令可以将缓冲对的文字插入到光标所在的位置

命令	英文	功能
y(复制命令)	copy	复制
yy	copy	复制一行,可以nyy复制多行
d(剪切命令)	delete	剪切
dd(剪切)	delete	剪切一行, 可以 ndd 剪切n行
p	paste	粘贴

提示:

- 命令 d 、x 类似于图形界面的 **剪切操作** -- ctrl + x
- 命令 y 类似于 图形界面的 **复制操作** -- Ctrl + C
- 命令 p 类似于图形界面的 **粘贴操作** -- Ctrl + v
- vi中的文本缓冲区只有一个,如果后续做过 复制、剪切操作, 之前缓冲区中的内容会被替换.

注意

- vi中的 **文本缓冲区** 和 系统的 **剪切板** 不是同一个
- 所以在其他软件中使用 `Ctrl + C` 复制的内容, 不能再 `vi` 中通过 `p` 命令粘贴
- 可以在 **编辑模式** 下使用 **鼠标右键粘贴**

4.7 替换

命令	英文	功能	工作模式
r	replace	替换当前字符	命令模式
R	replace	替换当前行光标后的字符	替换模式

- `R` 命令可以进入 **替换模式**, 替换完成后, 按下 `ESC` , 按下 `ESC` 可以回到 **命令模式**
- **替换命令** 的作用就是不用进入 **编辑模式**, 对文件进行 **轻量级的修改**

4.8 缩排和重复执行

命令	功能
>>	向右增加缩进
<<	向左减少缩进
.	重复上次命令

- **缩进命令** 在开发程序时, **统一增加代码的缩进** 比较有用!
    - 一次性 **在选中代码前增加 4 个空格,** 就叫做 **增加缩进**
    - 一次性 **在选中代码前删除 4 个空格**, 就叫做 **较少缩进**

- 在 **可视模式** 下, 缩排命令 主需要使用 一个 `>` 或者 `<`

在程序中, **缩进** 通常用来表示代码的归属关系

- **前面空格越少, 代码的级别越高**
- **前面空格越多, 代码的级别越低**

4.9 查找

常规查找

命令	功能
/str	查找str

- 查找到指定内容之后, 使用 `Next` 查找下一个出现的位置
  - `n` : 查找下一个
  - `N` : 查找上一个
- 如果不想看到高亮显示, 可以随便查找一个文件中不存在的内容即可

- 单词快速匹配

命令	功能
*	向后查找当前光标所在单词
#	向前查找当前光标所在单词

- 在开发中, 通过单词快速匹配, 可以快速看到这个单词在其他位置使用过

4.10 查找并替换

- 在 `vi` 中查找和替换命令需要在 **莫行模式** 下执行
- 记忆命令格式

```
:%s///g
```

1) 全局替换

- 一次向 替换文件中的 所有出现的旧文本
- 命令格式如下

```
:%s/旧文本/新文本/g
```

2) 可视区域替换

- **先选中** 要替换文字的 **范围**
- 命令格式如下

```
:s/旧文本/新文本/g
```

3) 确认替换

c confirm 确认

- 如果把末尾的 `g` 改成 `gc` 在替换的时候, 会有提示! 推荐使用
- 命令格式如下

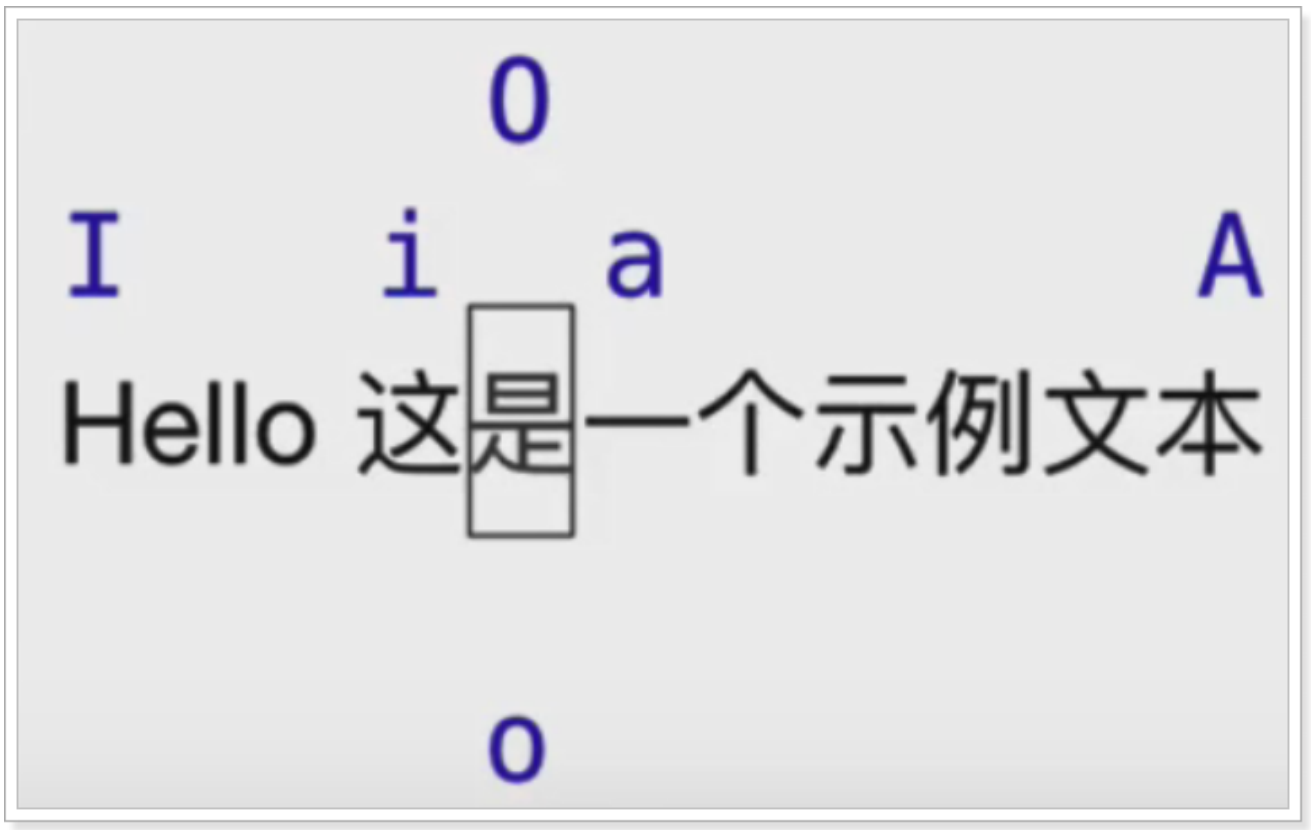
  ```
  :%s/旧文本/新文本/gc
  ```

1. `y` - `yes` 替换
2. `n` - `no` 不替换
3. `a` - `all` 替换所有
4. `q` - `quit` 退出替换
5. `l` - `last` 最后一个, 并把光标移动到行首
6. `^E` 向下滚屏
7. `^Y` 向上滚屏

4.11 插入命令(重要)

- 在 vi 中除了常用 `i` 进入**编辑模式** 外, 还提供了一下命令同样可以进入编辑模式

命令	英文	功能	常用
i	insert	在当前字符前插入文本	常用
I	insert	在行首插入文本	较常用
a	append	在当前字符后添加文本	
A	append	在行末添加文本	较常用
o		在当前行后面插入一空行	常用
O		在当前行前面插入一空行	常用



4.12 练习

演练1 -- 编辑命令 和 数字连用

- 在开发中, 可能会遇到连续输入 N 个同样的字符
- 例如: `**********` 连续10个星号

要实现这个效果可以在 **命令模式** 下

1. 输入 `10`, 表示要重复10次
2. 输入 `i` 进入 **编辑模式**
3. 输入 `*` 也就是重复的文字
4. 按下 `ESC` 返回 **命令模式**, 返回之后 `vi` 就会把 第 `2、3` 两步的操作重复 `10` 次

提示： 正常开发时， 在 **进入编辑模式之前，不要按数字**

演练2 -- 利用 可视块 给多行代码增加注释

- 在开发中， 可能会遇到一次向给多行代码 **增加注释** 的情况

> 在java中， 要给代码增加注释， 可以在代码
>
> 前增加一个 `//`

要实现这个效果可以在 **命令模式** 下

1. 移动到要添加注释的 **第1行代码**, 按 `^` 来到行首
2. 按 `Ctrl + v` 进入 **可视化** 模式
3. 使用 `j` 向下连续选中要添加的代码行

4. 输入 `I` **进入编辑模式** , 并在 **行首插入**, 注意: 一定要使用 I
5. 输入 `//` 也就是 注释符号
6. 按下 `ESC` 返回到 命令模式, 返回之后 `vi` 会在之前选中的每一行代码 前 插入 `//`

演练3: 坦克大战案例

- 已经学习过的 **莫行命令**

命令	英文	功能
:w 文件		另存为
:w	write	保存(ctrl + s)
:q	quit	退出, 如果没有保存,不允许退出
:q!	quit	强行退出, 不保存退出
:wq	write & quit	保存并退出
:x		保存并退出

在实际开发中, 可以使用 `w` 命令 **阶段性的备份代码**

用户权限相关命令

目标

- 理解 **用户** 和 **权限** 的基本概念
- **用户管理** 终端命令
- **组管理** 终端命令
- **修改权限** 终端命令

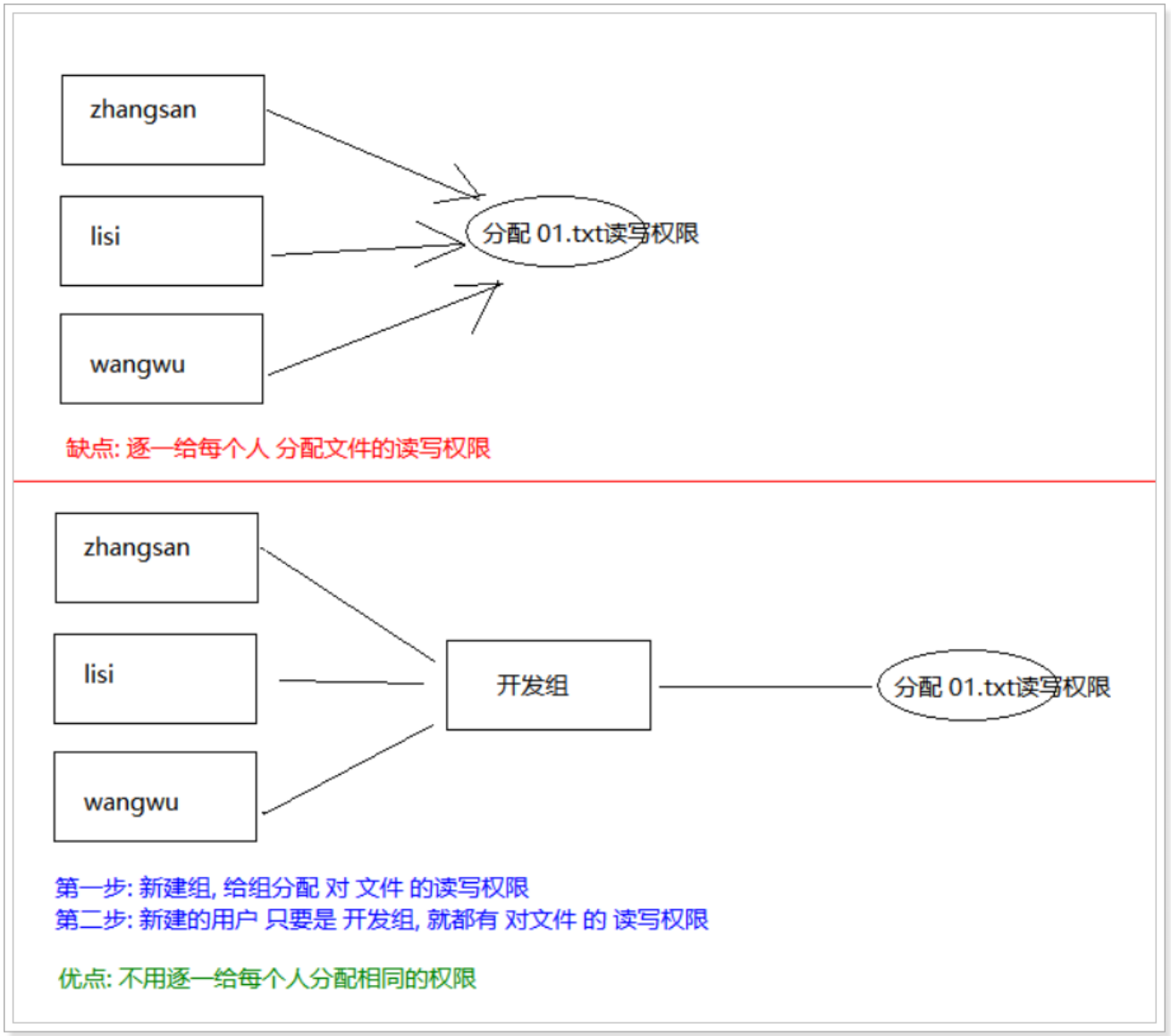
01.用户和权限的基本概念

1.1 基本概念

- **用户** 是Linux系统工作中重要的一环, 用户管理包括 **用户** 与 **组** 管理
- 在Linux系统中, 不论是由本级或是远程登录系统, 每个系统都**必须拥有一个账号**, 并且**对于不同的系统资源拥有不同的使用权限**
- 对 **文件 / 目录** 的权限包括:

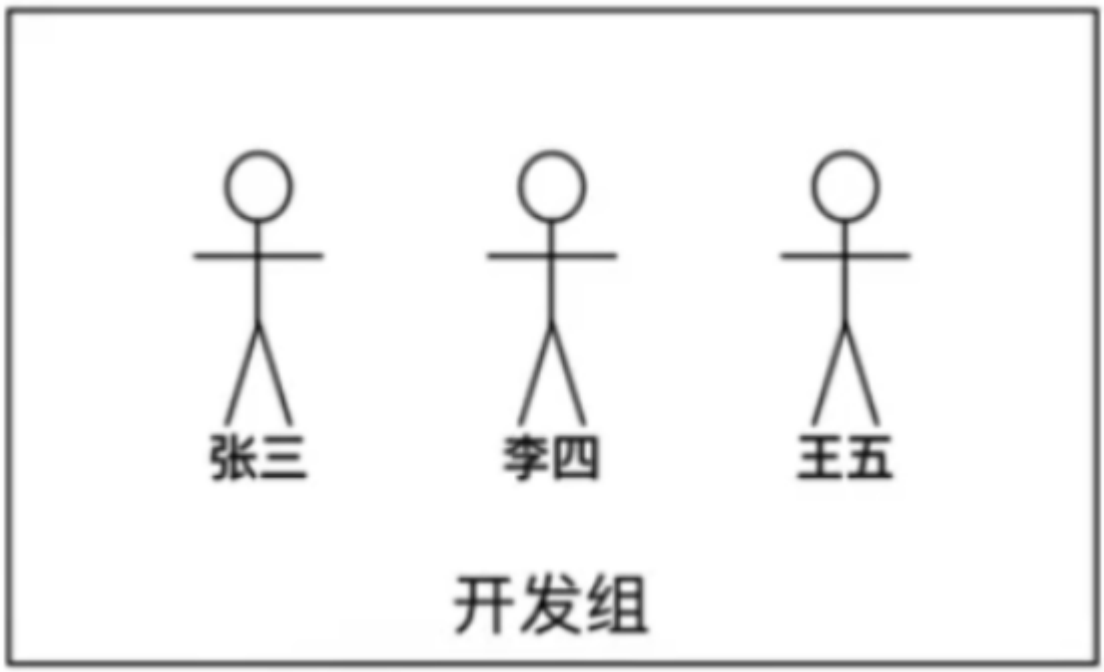
序号	权限	英文	缩写	数字序号
01	读	read	r	4
02	写	write	w	2
03	执行	execute	x	1
04	无权限		-	0

- 在 Linux中 ,可以指定 **每一个用户** 针对 **不同的文件或者目录** 的 **不同权限**



1.2 组

- 为了方便用户管理, 提出了 **组** 的概念, 如下图所示

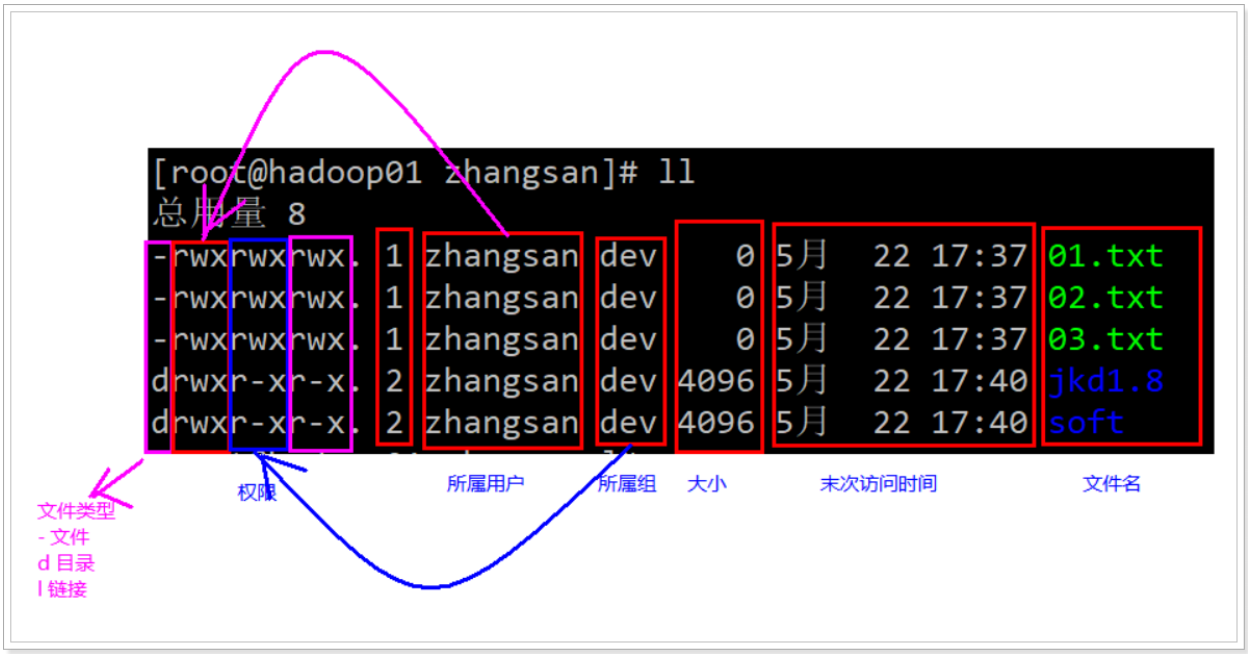


1.3 ls -l 扩展

```
[root@node00 tmp]# ll
总用量 48
-rw-r--r--. 1 root root  7378 5月  17 00:43 01.txt
drwxr-xr-x. 3 root root  4096 5月  17 00:27 aaa
```

- `ls -l` 可以查看文件夹下文件的详细信息, 从左到右 依次是:
  - **权限**, 第一个字符如果是 `d` 表示目录
  - **硬链接数**, 通俗的讲就是有多少种方式, 可以访问当前目录和文件
  - **拥有者**, 家目录下 文件 / 木兰路 的拥有者通常都是 当前用户
  - **组**, 在linux中, 很多时候, 会出现组名 和 用户名 相同的情况, 后续会讲
  - **大小**
  - **时间**
  - **名称**

	目录	拥有者权限			组权限			其他用户权限		
文件权限示例	-	r	w	-	r	w	-	r	-	-
目录权限示例	d	r	w	x	r	w	x	r	-	x



02.组管理 终端命令

本质: 给 同一类型用户 分配权限

序号	命令	作用
01	groupadd 组名	添加组
02	groupdel 组名	删除组
03	cat /etc/group	确认组信息
04	chgrp 组名 文件/目录名	修改文件/目录的所属组

> 提示:
>
> - 组信息保存在 `/etc/group` 文件中
> - `/etc` 目录是专门用来保存 **系统配置信息** 的目录

- 在实际应用中, 可以预先针对 **组** 设置好权限, 然后 **将不同的用户添加到对应的组中**, 从而**不用依次为每一个用户设置权限**

演练目标

1. 在 指定目录下 创建 `aaa` 目录
2. 新建 `dev` 组
3. 将 `aaa` 目录的组修改为 `dev`

03.用户管理 终端命令

3.1 创建用户 / 设置密码 / 删除用户

命令	作用	说明
useradd -m -g 组 新建用户名	添加新用户	`-m` 自动建立用户家目录 `-g` 指定用户所在的组, 否则会建立一个和同名的组
passwd 用户名	设置用户密码	如果是普通用户, 直接用 passwd可以修改自己的账号密码
userdel -r 用户名	删除用户	`-r` 选项会自动删除用户家目录
cat /etc/passwd | grep 用户名	确认用户信息	新建用户后,用户信息会保存在 `/etc/passwd` 文件夹中

> 提示:
>
> - 创建用时, 如果忘记添加 `-m` 选项指定新用户的家目录 -- 最简单的方式就是**删除用户,重新创建**
> - 创建用户时, 默认会创建一个和**用户名**同名的**组名**
> - 用户信息保存在 **/etc/passwd** 文件中

`/etc/passwd` 文件存放的是用户的信息, 由6个分好组成的7个信息, 分别是

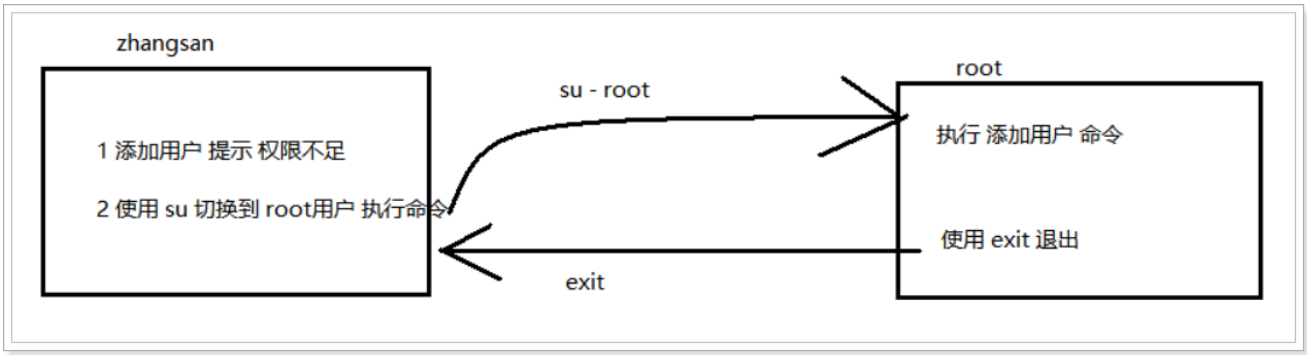
```
[root@node00 home]# cat /etc/passwd | grep --color zhangsan
zhangsan:x:500:500::/home/zhangsan:/bin/bash
```

1. 用户名
2. 密码 (x, 表示加密的密码)
3. UID (用户标志)
4. GID(组标志)
5. 用户全名或本地账号
6. 家目录
7. 登录使用的Shell, 就是登录之后, 使用的终端命令

3.2 查看用户信息

序号	命令	作用
01	id [用户名]	查看用户UID 和 GID 信息
02	who	查看当前所有登录的用户列表
03	whoami	查看当前登录用户的账户名

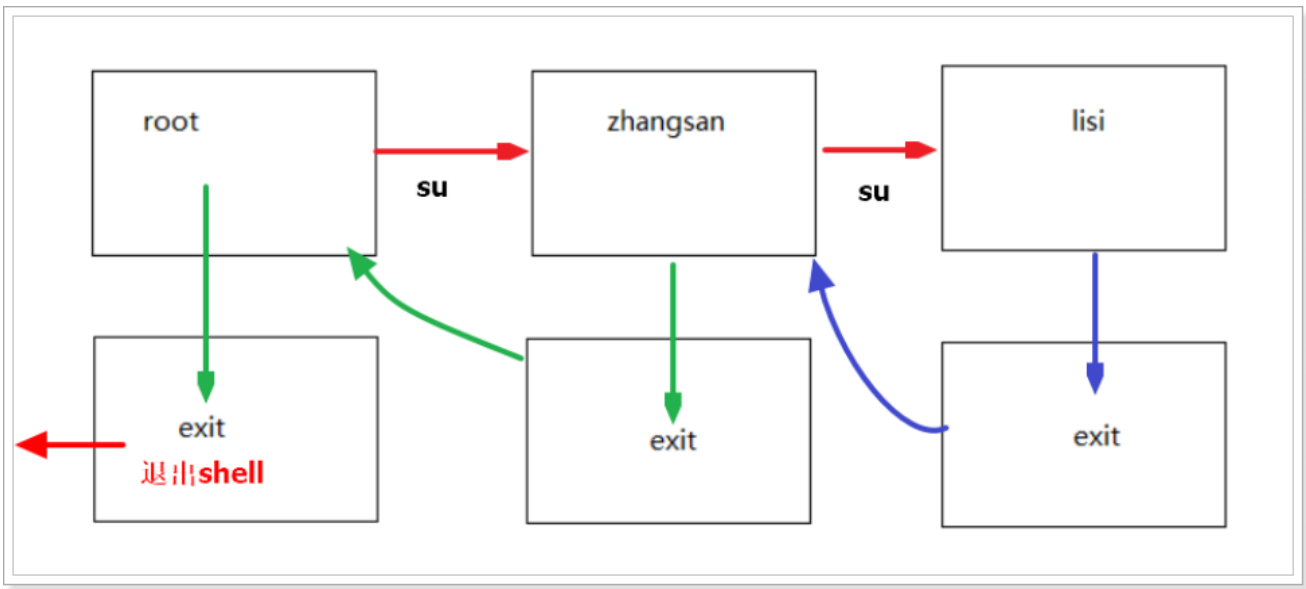
3.3 `su` 切换用户



- **因为 普通用户 不能使用某些权限, 所以需要 切换用户**

序号	命令	作用	说明
01	su `-` 用户名	切换用户, 并且企划目录	`-` 可以切换到用户家目录, 否则保持位置不变
02	exit	退出当前登录用户	

- `su` 不接用户名, 可以切换到 `root` , 但是不推荐使用, 因为不安全
- `exit` 示意图如下:



3.4 sudo

- **虽然** 通过 `su -u root` 可以切换到 **root用户**, **但是** 存在严重的 **安全隐患**
- linux系统中的 `root` 账号通常 **用于系统的维护和管理**, 对操作系统的所有资源 **具有访问权限**
- 如果不小心使用 `rm -rf ... ...` , 就可能将系统搞瘫痪
- 在大多数版本的linux中, 都不推荐 **直接只用 root 账号登录系统**

- `sudo` 命令用来以其他身份来执行命令, 预设的身份为 `root`
- 用户使用 `sudo` 时, 必须先输入密码, 之后**5分钟的有效期限**, 超过期限则必须重新输入密码

> 提示: 若其未经授权的用户企图使用 sudo, 则会发出警告邮件给管理员

3.4.1 给 指定用户 授予 权限

- **本质: 通知 服务器 给 特定用户 分配 临时管理员权限**

vim /etc/sudoer

```
# 默认存在: root用户 具备所有的权限
root    ALL=(ALL)       ALL

# 授予 zhangsan 用户 所有的权限
zhangsan    ALL=(ALL)       ALL
```

3.4.2 使用 用户 `zhangsan` 登录, 操作管理员命令

- 本质: **使用临时管理员权限**

```
# 不切换root用户, 也可以完成 添加用户的功能
sudo useradd -m -g dev zhaoliu
```

准备工作: 使用root用户操作

```
# 清空目录中的内容
rm -rf /export/*

# 创建测试目录
mkdir -p /export/aaa/

# 创建测试文件
touch /export/aaa/01.txt /export/aaa/02.txt

# 查看指定目录内容
tree /export
```

04.修改用户权限

序号	命令	作用
01	chmod	修改权限

4.1 方式一: 修改用户权限

- `chmod` 可以修改 **用户/组** 对 **文件/目录** 的权限
- 命令格式如下:

```
chmod +/- rwx 文件名|目录名
```

> 提示: 已上方式会一次向修改 `拥有者` / `组` 权限

目标演练:

```
# 1. 使用 root 删除目录的可读 可写 可执行权限

# 2. 使用 其他用户如 zhangsan 无法切换到 这个目录

# 3. 使用 root 增加目录的执行权限，再次 使用 zhangsan 切换到目录试试
```

4.2 方式二

- 虽然 方式一 直接修改**文件|目录**的 **读|写|执行** 权限, 但是不能精确到 **拥有者|组|其他** 权限
- 命令格式如下:(`u` 表示所属用户 / `g` 表示所属组 / `o` 表示其他)

```
chmod -R u=rwx,g=rx,o=rwx 文件|目录
```

序号	权限	英文	缩写	数字序号
01	读	read	r	4
02	写	write	w	2
03	执行	execute	x	1
04	无权限		-	0

目标演练:

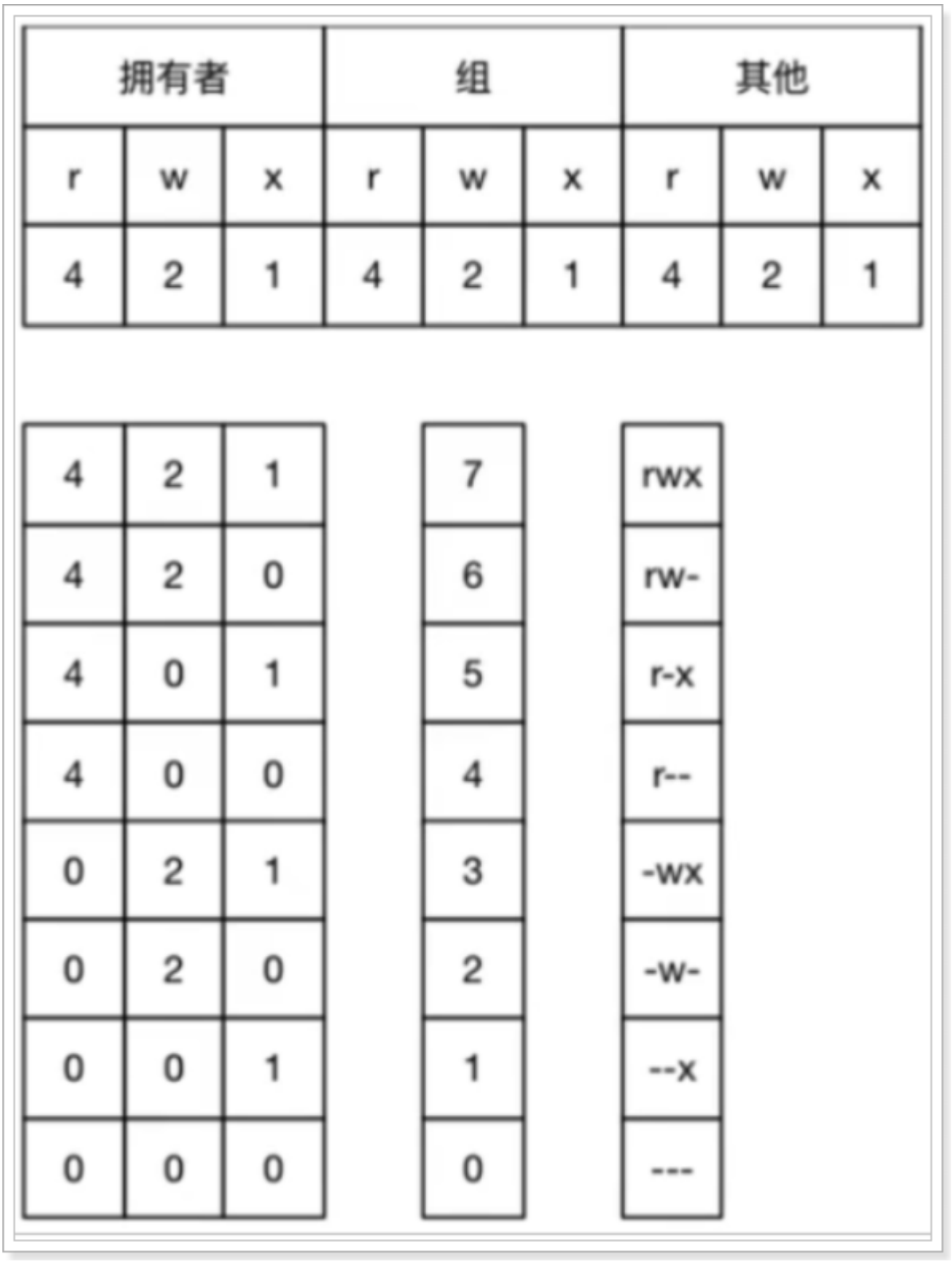
```
# 1 使用root用户给 所属用户分配 可读、可写、可执行 权限，
    所属组 分配 可读、可执行 权限，
    其他人 分配 可读、可执行 权限
```

4.3 方式三: 简化方式二

- 命令格式如下:

```
chmod -R 755 文件|目录
```

- 说明 **第一个数字** 是 **拥有者权限**, **第二个数字** 是 **组权限**, **第三个数字** 是 **其他用户权限**



拥有者			组			其他		
r	w	x	r	w	x	r	w	x
4	2	1	4	2	1	4	2	1

4	2	1	7	rwx
4	2	0	6	rw-
4	0	1	5	r-x
4	0	0	4	r--
0	2	1	3	-wx
0	2	0	2	-w-
0	0	1	1	--x
0	0	0	0	---

常用数字组合有(`u` 表示用户 / `g` 表示组 / `o` 表示其他)

- `777` ===> `u=rwx, g=rwx, o=rwx`
- `755` ===> `u=rwx, g=rx, o=rx`
- `644` ===> `u=rw, g=r, o=r`

目标演练:

```
# 1 使用root用户给 所属用户分配 可读、可写、可执行 权限,
    所属组 分配 可读、可执行 权限,
    其他人 分配 可读、可执行 权限
```

系统信息相关命令

- 本节内容主要是为了方便通过远程终端维护服务器时, 查看服务器上当前 **系统日期和时间 / 磁盘空间占用情况 /程序执行情况**
- 本小结学习终端命令都是查询命令, 通过这些命令对系统资源的使用情况有个了解

目标

- 时间和日期
  - `date`
  - `cal`
- 磁盘和目录空间
  - `df`
  - `du`
- 进程信息
  - `ps`
  - `top`
  - `kill`

1.时间和日期

1.1 date 时间

命令	作用
date	查看系统时间(**默认**)
date +"%Y-%m-%d %H:%M:%S"	查看系统时间(**指定格式**)
date -s "时间字符串"	设置系统时间

第一步: 显示当前时间

```
# 显示时间
date

# 按照指定格式显示时间
date +"%Y-%m-%d %H:%M:%S"
```

第二步: 设置系统时间

```
date -s "时间字符串"
```

1.2 cal 日历

序号	命令	作用
01	cal	查看当前月的日历
02	cal -y	查看当前年的日历
03	cal 2020	查看2020年的日历
04	cal 10 2020	查看2020年10月的日历

02.磁盘信息

序号	命令	作用
01	df -h	disk free 显示磁盘剩余空间
02	du -h [目录名]	disk usage 显示目录下的目录大小

- 选项说明

参数	含义
-h	以人性化的方式显示文件的大小

03.进程信息

- 所谓 **进程**, 通俗的说就是 **当前正在执行的一个进程**

序号	命令	作用
01	ps aux	process status 查看进程的详细情况
02	top	动态显示运行中进程并且排序
03	kill [-9] 进程代号	终止指定代号的进程 -9 表示强行终止

> `ps` 默认只会显示当前用户通过终端启动的应用程序

- `ps` 选项说明功能

选项	含义
a	显示终端上的所有进程,包括其他用户的进程
u	显示进程的详细状态
x	显示没有控制终端的进程

> 提示: 使用`kill`命令时, 最好只终止由当前用户开启的进程, 而不要终止`root`身份开启的进程, 否则可能导致系统崩溃

- 要退出 `top` 可以直接输入 `q`