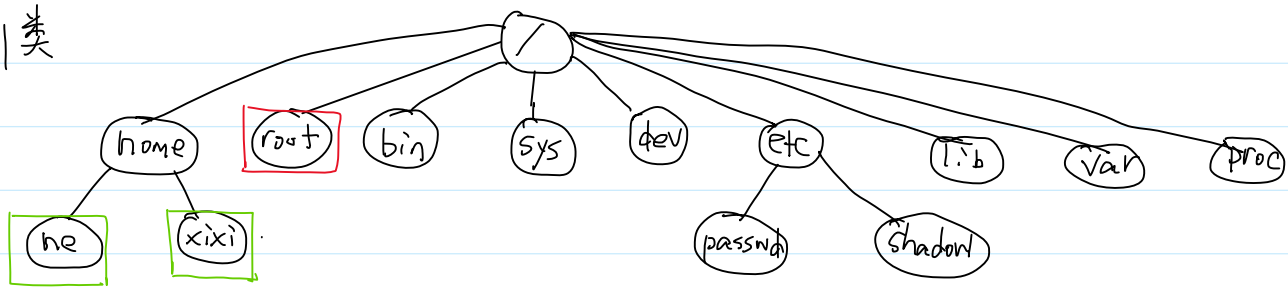
文件子系统

2024年5月14日 10:46

分门别类



通配符

2024年5月14日 11:09

*：匹配任意多个字符（包括0个）

?：匹配任意一个字符

集合（类）：

[characters] 匹配集合内任意一个字符

[!characters] 匹配集合外任意一个字符

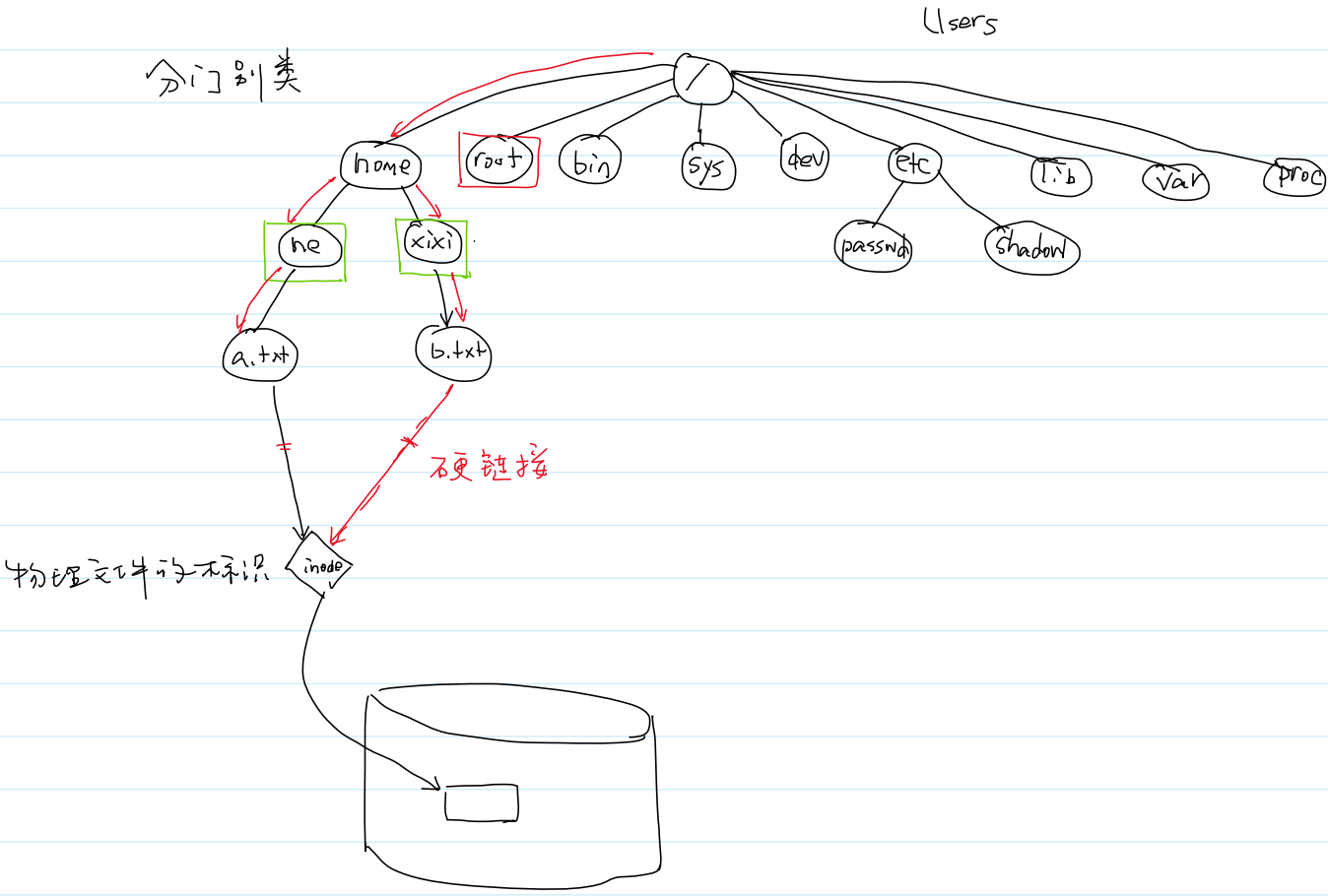
[abc] [!abc] [0-9], [a-z] [0-9A-Za-z_]

作用：匹配文件名字 （通配符）

匹配文件内容 （正则表达式）

ls

2024年5月14日 11:21



硬链接数

最近修改时间

文件大小

```
he@he-vm:~/cpp58/Linux02$ ls -l
total 12
drwxrwxr-x 2 he he 4096  5月 14 11:22 dir1
drwxrwxr-x 2 he he 4096  5月 14 11:22 dir2
drwxrwxr-x 2 he he 4096  5月 14 11:22 dir3
-rw-rw-r-- 1 he he    0  5月 14 11:22 text1
-rw-rw-r-- 1 he he    0  5月 14 11:22 text2
-rw-rw-r-- 1 he he    0  5月 14 11:22 text3
```

第一个字符：文件类型

- -：普通文件
- d (directory)：目录
- c (character)：字符设备文件（键盘、显示器、鼠标...）
- b (block)：块设备文件（硬盘、U盘...）
- p (pipe)：有名管道
- s (socket)：本地套接字
- l (link)：符号链接

剩余九个字符：权限

八进制数

剩余九个字符，权限

user 文件拥有者，(rw-) → 八进制数

group 文件所有组，rw-

others 其它人 ，r--

775 → rwx rwx r-x

664 → rw- rw- r--

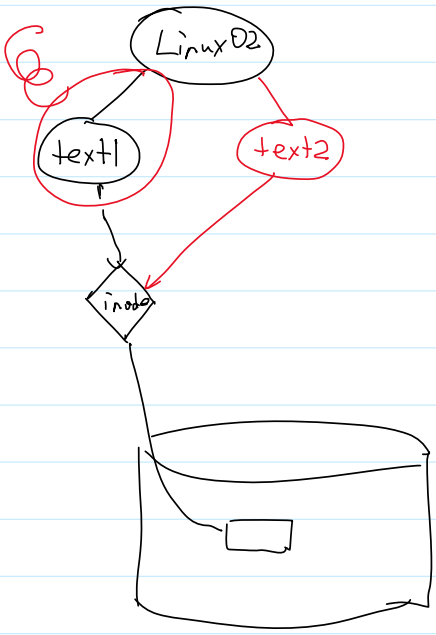
mv, cp, rm

2024年5月14日 14:28

1. mv

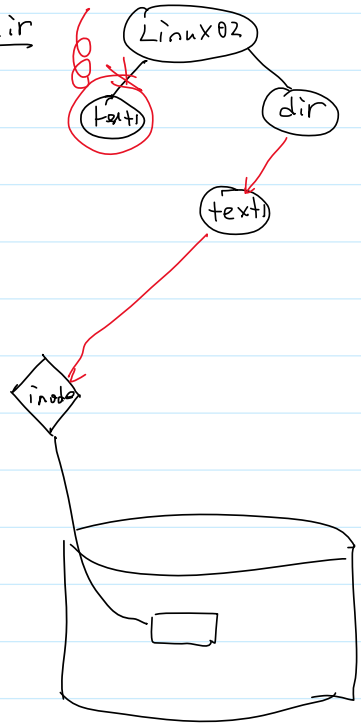
rename

mv text1 text2



mv text1 dir

move



复制文件或目录

cp(copy) 命令可以用来复制文件和目录。

```
$ man cp
    cp - copy files and directories
格式:
    cp [选项] SOURCE DEST
    cp [选项] SOURCE... DEST
    Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.
常用选项:
    -n, --no-clobber
        如果文件已存在，则不覆盖(默认会覆盖已有文件)。
    -i, --interactive
        如果文件存在，则给用户提示信息(由用户决定是否覆盖?)
    -R, -r, --recursive
        递归复制(用于copy目录)
```

常用方式:

```
$ cp text1 text2                  # 将text1复制到text2中；如果text2存在，则覆盖。
$ cp text1 text2 text3 dir        # 将text1,text2,text3复制到目录dir中；如果文件已存在，则覆盖。
$ cp -n text1 text2               # 将text1复制到text2中；如果text2存在，不覆盖。
$ cp -i text1 text2 text3 dir     # 将text1,text2,text3复制到目录dir中；如果文件已存在，则提示用户是否覆盖。
$ cp -r dir1 dir2                 # 递归地将目录dir1复制到目录dir2
```

rm(remove)命令可以删除文件和目录。

```
$ man rm
    rm - remove files or directories
格式:
    rm [选项] FILE...
常用选项:
    -f, --force
        忽略不存在的文件，永远不提示
    -i
        在每次删除前，都提示用户是否删除
    -r, -R, --recursive
        递归删除
```

常用方式:

```
$ rm text1                  # 删除文件text1
$ rm text1 text2 text3      # 删除文件text1, text2, text3
$ rm -i *.txt               # 删除当前目录下所有以.txt结尾的文件，并提示用户是否删除
$ rm -rf dir                # 递归删除目录dir，不给出任何提示
```

alias

2024年5月14日 14:53

别名

有些命令比较长，自然我们就想给这些命令起一个别名，方便以后使用，alias 命令就是用来做这事情的。

```
$ man alias
    alias - define or display aliases
格式:
    alias [命令=别名]...
```

常见用法:

```
$ alias                 # 查看别名
alias g++11='g++ -std=c++11'
alias ll='ls -alF'
alias ls='ls --color=auto'
...
$ alias h='history'      # 设置别名
```

创建文件

2024年5月14日 15:02

① 创建空文件 touch

② 创建文件并输入简短的内容。 echo

③ 编辑文件 vim

查找文件

2024年5月14日 15:09

#1. which

1. 我们可以使用 which 命令来查找可执行程序的路径：

```
$ man which
    which - locate a command
格式:
    which [-a] cmd...
选项:
    -a
        显示所有匹配的路径
```

常用方式：

```
$ which bash        # 查看bash的路径
$ which ls tree     # 查看命令ls和tree的路径
$ which -a vim      # 查看vim的所有路径（我们可能装了多个版本的vim）
```

which 是根据 PATH 环境变量中的路径依次去查找的，然后显示第一个匹配项，或者显示所有匹配项。

我们可以使用 env 命令查看环境变量：

```
$ env
...
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
...
```

```
PATH=/home/he/.local/bin:/home/he/.cargo/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:
/snap/bin:/home/he/.local/bin:/home/he/.vim/pack/minpac/start/fzf/bin:/usr/local/go/bin
```

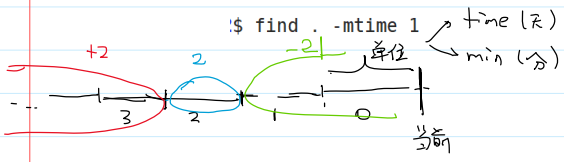
```
he@he-vm:~/cpp58/Linux02/dir1$ which ls
/usr/bin/ls
```

```
he@he-vm:~/cpp58/Linux02/dir1$ which -a ls
/usr/bin/ls
/bin/ls
```

#2. find

```
$ man find
    find - search for files in a directory hierarchy
格式:
    find [start-point...] 查找条件      # 省略start-point，默认起始点为当前工作目录
常用选项:
    -name pattern
        查找文件名符合pattern的文件
    -type c
        查找类型为c的文件:
            b(block): 块设备文件
            c(character): 字符设备文件
            d(directory): 目录
            p(named pipe): 有名管道
            f(file): 普通文件
            l(symbolic link): 符号链接
            s(socket): 套接字
    -size n[cwbkMG]
        File uses n units of space, rounding up.
            b: for 512-byte blocks (this is the default if no suffix is used)
            c: for bytes
            w: for two-byte words
            k: for Kibibytes (KiB, units of 1024 bytes)
            M: for Mebibytes (MiB, units of 1024 * 1024 = 1048576 bytes)
            G: for Gibibytes (GiB, units of 1024 * 1024 * 1024 = 1073741824 bytes)
        可以在n前面添加'+'和'-'，表示大于和小于。
    -empty
        查找空的文件或空的文件夹
```

-user username, -uid uid
    根据用户名和用户id查找
-group groupname, -gid gid
    根据组名和组id查找
-perm mode
    根据权限查找
根据时间查找：
    -amin n, -atime n, -cmin n, -ctime n, -mmin n, -mtime n
        a(access)：文件访问的时间
        c(change)：文件属性发生改变的时间
        m(modify)：文件内容发生改变的时间
        min：以分钟为单位
        time：以天为单位
    可以在n前面添加'+'和'-'，表示大于和小于。
组合查找：
    -a(and)，-o(or)，!(not)：分别表示与、或、非



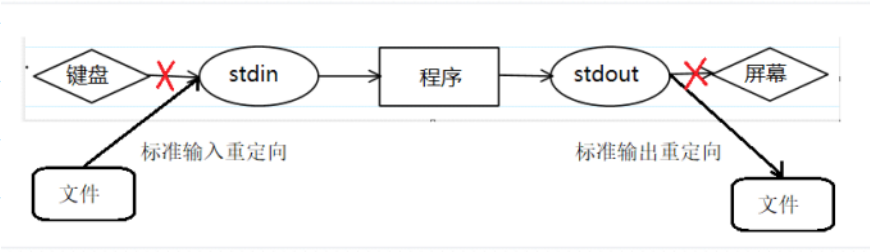

```
!$ find . -mmin 2
!$ find . -mmin +2
```

查看文件内容

2024年5月14日 16:20

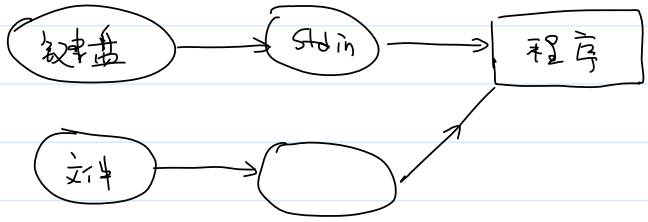
重定向

2024年5月14日 17:02

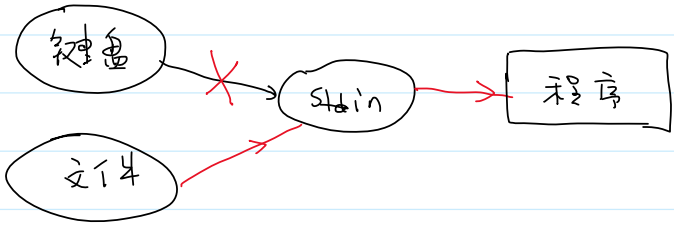


标准流	文件描述符	重定向符号
stdin	0	<
stdout	1	> 和 >>
stderr	2	2> 或 2>>

```
he@he-vm:~/cpp58/Linux02$ wc users
 2  6 59 users
```




```
he@he-vm:~/cpp58/Linux02$ wc < users
 2  6 59
```



搜索文件内容

2024年5月14日 17:17

```
$ man grep
    grep - print lines matching a pattern
格式:
    grep [选项] pattern [file]...
常见选项:
    -E, --extended-regexp
        使用扩展的正则表达式
    -i, --ignore-case
        忽略大小写
    -v, --invert-match
        显示不匹配正则表达式的行
    -n, --line-number
        显示行号
    -c, --count
        不显示匹配的行，显示匹配行的个数
```

正则表达式

2024年5月14日 17:28

正则表达式

正则表达式语法有点复杂，而且不同的工具使用的语法还不尽相同。下面我们介绍一些常用的正则表达式语法规则：

```
1. 基本单位
    基本单位主要包含：字符、转义字符、.(表示任意一个字符)、集合(比如，[abc]，[^abc])、(expr)

2. 基本操作
    操作的对象是基本单位，主要包含两个基本操作：连接和重复
    a. 连接："ab"，"[abc]x"，".txt"，"\.txt"
    b. 重复（前面的基本单位重复的次数）
        +：重复至少一次(>=1)，比如："abc+"，"[abc]+"，"(abc)+"
        ?：重复零次或一次(0|1)，比如："abc?"，"[abc]?" "(abc)?"
        *：重复任意次数(>=0)，比如："abc*"，"[abc]*"，".*"
        {m}：重复m次
        {m,n}：重复m到n次([m,n])
        {n, }：至少重复n次([>=n])

        xyz abc
3. 指定基本单位出现的位置
    ^：行首，比如："^abc"
    $：行尾，比如："xyz$"
    \<：词首
    \>：词尾
```

单词，用空白字符分割 \< word \>

Q：以f开头，以t结尾的单词的正则表达式如何写？（单词之间以空格分隔）

'\<f[^ \t\n]*t\>'