

## Shell 和命令操作基础

### 内容提要

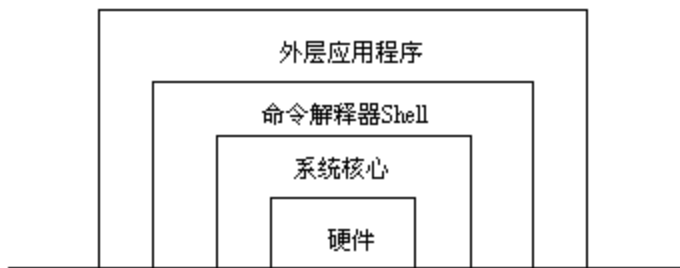
1. 熟悉 Shell 及其功能
2. 熟悉 Shell 的元字符
3. 熟悉命令行格式
4. 学会使用命令帮助

## Shell 简介

### 什么是Shell

- Shell是系统的用户界面，提供了用户与内核进行交互操作的一种接口(命令解释器)。
- Shell接收用户输入的命令并把它送入内核去执行。
- Shell起着协调用户与系统的一致性和在用户与系统之间进行交互的作用。

Shell在Linux系统中具有极其重要的地位，如图所示。



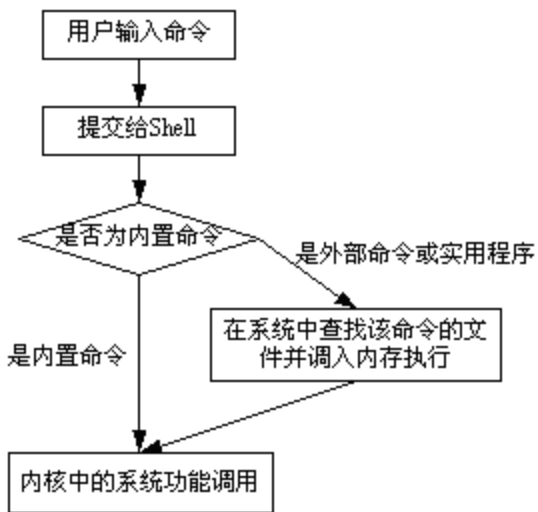
### Shell在Linux系统中的地位

### Shell的功能

Shell最重要的功能是命令解释，从这种意义上说，Shell是一个命令解释器。Linux系统中的所有可执行文件都可以作为Shell命令来执行。Linux系统上可执行文件的分类见下表。

类 别	说 明
Linux命令	存放在/bin、/sbin目录下的命令
内置命令	出于效率的考虑，将一些常用命令的解释程序构造在Shell内部
实用程序	存放在/usr/bin、/usr/sbin、/usr/share、/usr/local/bin 等目录下的实用程序或工具
用户程序	用户程序经过编译生成可执行文件后，也可作为Shell命令运行
Shell脚本	由Shell语言编写的批处理文件

下图描述了Shell是如何完成命令解释的。



### 命令解释过程

当用户提交了一个命令后，**Shell**首先判断它是否为内置命令，如果是就通过**Shell**内部的解释器将其解释为系统功能调用并转交给内核执行；若是外部命令或实用程序就试图在硬盘中查找该命令并将其调入内存，再将其解释为系统功能调用并转交给内核执行。在查找该命令时分为两种情况：

1. 用户给出了命令的路径，**Shell**就沿着用户给出的路径进行查找，若找到则调入内存，若没找到则输出提示信息。
  2. 用户没有给出命令的路径，**Shell**就在环境变量**PATH**所制定的路径中依次进行查找，若找到则调入内存，若没找到则输出提示信息。
1. 内置命令是包含在**Shell**自身当中的，在编写**Shell**的时候就已经包含在内了，当用户登录系统后就会在内存中运行一个**Shell**，由其自身负责解释内置命令。一些基本的命令如**cd**、**exit**等都是内置命令。查看内置命令的使用方法可以用 **help** 命令。
  2. 外部命令是存在于文件系统某个目录下的具体的可执行程序，如文件拷贝命令**cp**，就是在**/bin**目录下的一个可执行文件。查看外部命令的使用方法可以用 **man** 或 **info** 命令。外部命令也可以是某些商业或自由软件，如 **mozilla**等。

此外，**Shell**还具有如下的一些功能：

- 通配符
- 命令补全、别名机制、命令历史
- 重定向
- 管道
- 命令替换
- **Shell**编程语言

## Shell的主要版本

下表列出了几种常见的**Shell**版本。

版 本	说 明
Bourne Again Shell ( <b>bash</b> , <b>bsh</b> 的扩展)	<b>bash</b> 是大多数Linux 系统的默认 <b>Shell</b> 。 <b>bash</b> 与 <b>bsh</b> 完全向后兼容，并且在 <b>bsh</b> 的基础上增加和增强了很多特性。 <b>bash</b> 也包含了很多 C <b>Shell</b> 和 Korn <b>Shell</b> 中的优点。 <b>bash</b> 有很灵活和强大的编程接口，同时又有很友好的用户界面
Korn Shell ( <b>ksh</b> )	Korn <b>Shell</b> ( <b>ksh</b> ) 由 Dave Korn所写。它是UNIX系统上的标准 <b>Shell</b> 。另外，在Linux环境下有一个专门为Linux系统编写的Korn <b>Shell</b> 的扩展版本，即Public Domain Korn <b>Shell</b> ( <b>pdksh</b> )。
<b>tcsh</b> ( <b>cs</b> h 的扩展)	是C <b>Shell</b> 的扩展。 <b>tcsh</b> 与 <b>cs</b> h 完全向后兼容，但它包含了更多的使用户感觉方便的新特性，其最大的提高是在命令行编辑和历史浏览方面

在 Linux 下默认的 **Shell** 是 **bash**。

## Shell元字符

在Shell中有一些具有特殊的意义字符，称为 **Shell元字符**（**shell metacharacters**）。若不以特殊方式指明，Shell并不会把它们当做普通文字符使用。

下表简单介绍了常用的Shell元字符的意义：

Shell元字符	Shell元字符的含义
*	代表任意字符串
?	代表任意字符
/	代表根目录或作为路径间隔符使用
\	转义字符。当命令的参数要用到保留字时，要在保留字前面加上转义字符
\<Enter>	续行符。可以使用续行符将一个命令行分写在多行上
\$	变量值替换，如： <code>\$PATH</code> 表示环境变量PATH的值
'	在'...'中间的字符都会被当做文字处理，指令、文件名、保留字等都不再具有原来的意义
"	在"..."中间的字符会被当做文字处理并允许变量值替换
`	命令替换，置换`...`中命令的执行结果
<	输入重定向字符
>	输出重定向字符
	管道字符
&	后台执行字符。在一个命令之后加上字符"&"，该命令就会以后台方式执行
;	分割顺序执行的多个命令
( )	在子Shell中执行命令
{ }	在当前Shell中执行命令
!	执行命令历史记录中的命令
~	代表登录用户的宿主目录（自家目录）

## 命令操作基础

---

### 目录和文件名的命名规则

在Linux下可以使用长文件或目录名，可以给目录和文件取任何名字，但必须遵循下列的规则：

- 除了/之外，所有的字符都合法
- 有些字符最好不用，如空格符、制表符、退格符和字符：? , @ # \$ & ( ) \ | ; ` ^ " < > 等
- 避免使用+、-或.来作为普通文件名的第一个字符
- 大小写敏感
- 以"."开头的文件或目录是隐含的

### 命令基本格式

Shell命令的一般格式为：

```
cmd [options] [arguments]
```

其中：

- `cmd` 是命令名

- **options** 是选项
- **arguments** 是参数

说明:

- 最简单的**Shell**命令只有命令名, 复杂的**Shell**命令可以有多个选项和参数。
- 选项和参数都作为**Shell**命令执行时的输入, 它们之间用空格分隔开。
- 单字符参数前使用一个减号 (-), 单词参数前使用两个减号 (--).
- 多个单字符参数前可以只使用一个减号。
- 操作对象 (**arguments**) 可以是文件也可以是目录, 有些命令必须使用多个操作对象, 如**cp**命令必须指定源操作对象和目标操作对象。
- 并非所有命令的格式都遵从以上规则, 例如**dd**、**find**等。

例如:

```
$ ls
$ ls -lRa /home
$ cat abc xyz
$ ls --help
```

具有以上格式的字符串习惯地称为命令行, 命令行是用户与**Shell**之间对话的一个基本单位。

## 通配符

通配符主要用于用户方便描述目录或文件。常用的通配符如下:

- **\***: 匹配任何字符和任何数目的字符
- **?**: 匹配单一数目的任何字符
- **[ ]**: 匹配[ ]之内的任意一个字符
- **[! ]**: 匹配除了[! ]之外的任意一个字符, !表示非的意思
- **"\*"**能匹配文件或目录名中的`\"`。
- **"\*"**不能匹配首字符是`\"`的文件或目录名。

通配符在指定一系列的文件名时非常有用, 下面列举一些例子:

- **ls \*.c**: 列出当前目录下的所有C语言源文件。
- **ls /home/\*/\*.c**: 列出/home目录下所有子目录中的所有C语言源文件。
- **ls n\*.conf**: 列出当前目录下的所有以字母n开始的conf文件。
- **ls test?.dat**: 列出当前目录下的以test开始的, 随后一个字符是任意的.dat文件。
- **ls [abc]\***: 列出当前目录下的首字符是a或b或c的所有文件。
- **ls [!abc]\***: 列出当前目录下的首字符不是a或b或c的所有文件。
- **ls [a-zA-Z]\***: 列出当前目录下的首字符是字母的所有文件。

## 获得命令帮助

---

### 使用man命令获得帮助

在系统中, 用户可以非常容易的获得系统的帮助和支持, 系统发行版本中为几乎每个程序、工具、命令或系统调用编制了使用手册。要想查看某个命令的使用手册页, 只要输入**man**后跟该命令的名称即可。例如, 输入如下命令将显示如图

所示的界面。

```
$ man ls
```

```
LS(1)                                FSF                                LS(1)
NAME
  ls - list directory contents
SYNOPSIS
  ls [OPTION]... [FILE]...
DESCRIPTION
  List information about the FILES (the current directory by
  default).  Sort entries alphabetically if none of -cftuSUX
  nor --sort.

  -a, --all
      do not hide entries starting with .
  -A, --almost-all
      do not list implied . and ..
  -b, --escape
      print octal escapes for nongraphic characters
  --block-size=SIZE
      use SIZE-byte blocks
:
```

使用man获得命令帮助

在此界面中可以查看有关ls命令的详细使用说明。用户可以使用↑、↓和PgDn、PgUp键进行翻阅，按q键退出。

一般来说，命令的使用手册页中会包括如表中所示的组成信息。

- **Name** 命令的名称及简单说明
- **Synopsis** 如何使用这个命令即命令行选项
- **Description** 对这个命令及其选项的解释
- **Files** 这个命令用到的文件清单和它们存放的位置
- **See Also** 有相互联系的使用手册页的清单
- **Diagnostics** 特殊输出情况的说明
- **Bugs** 编程漏洞
- **Author** 命令程序的主要编写者和其他维护人员

另外，根据内容的不同可将手册页分为不同的类型，不同类型用一个数字（或字母）代表，各种类型的含义如表所示。

- **man1** 普通用户的可执行命令手册
- **man2** 系统调用手册，内核函数的说明
- **man3** 子程序手册，库函数的说明
- **man4** 系统设备手册，“/dev”目录中设备文件的参考说明
- **man5** 配置文件格式手册，大多为“/etc”目录下各种配置文件的格式描述
- **man6** 游戏和趣味小程序的说明手册
- **man7** 协议转换手册，也包括一些杂项
- **man8** 系统管理工具手册，这些命令只有超级用户才可以执行
- **man9** Linux 系统例程手册

手册页按照不同的类型被存放在系统不同的目录下（/usr/share/man/man[1..9]）。

例如： 使用如下命令可以查看passwd命令的使用方法

```
$ man 1 passwd
```

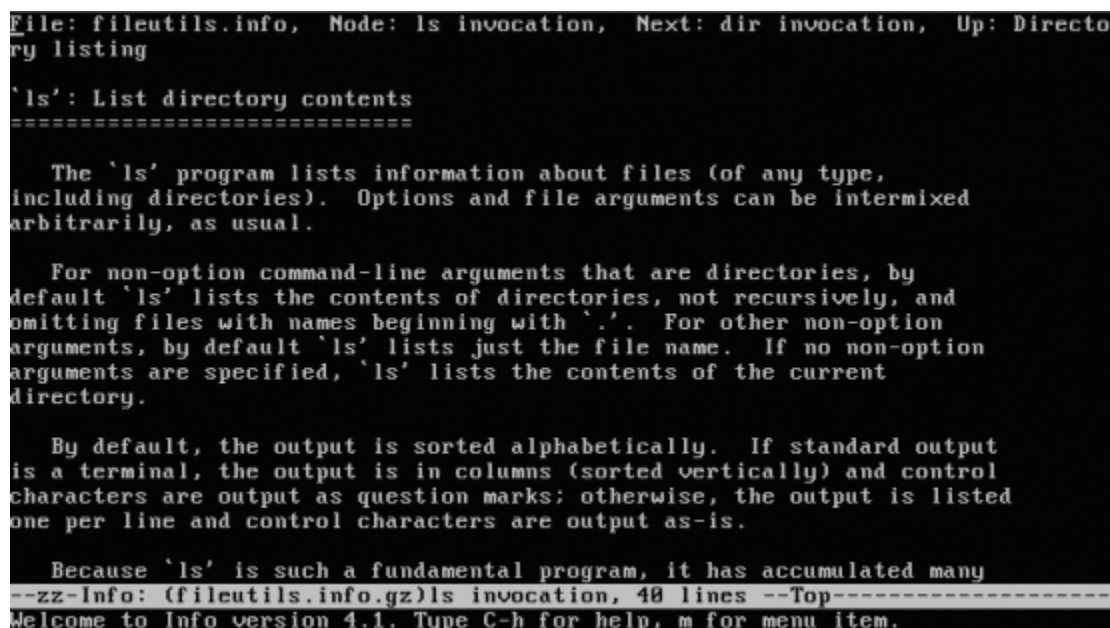
使用如下命令可以查看passwd配置文件的描述信息

```
$ man 5 passwd
```

## 使用info命令获得帮助

texinfo是Linux系统中提供的另一种格式的帮助信息，他与手册页相比具有更强的交互性，如支持连接跳转功能等。通常可以使用info命令查看texinfo格式的帮助文档（info文档存放在“/usr/share/info/”目录中）。例如，输入如下命令将显示如图所示的界面。

```
$ info ls
```



```
File: fileutils.info, Node: ls invocation, Next: dir invocation, Up: Directory listing
`ls': List directory contents
=====

The `ls' program lists information about files (of any type, including directories). Options and file arguments can be intermixed arbitrarily, as usual.

For non-option command-line arguments that are directories, by default `ls' lists the contents of directories, not recursively, and omitting files with names beginning with `.'. For other non-option arguments, by default `ls' lists just the file name. If no non-option arguments are specified, `ls' lists the contents of the current directory.

By default, the output is sorted alphabetically. If standard output is a terminal, the output is in columns (sorted vertically) and control characters are output as question marks; otherwise, the output is listed one per line and control characters are output as-is.

Because `ls' is such a fundamental program, it has accumulated many
--zz-Info: (fileutils.info.gz)ls invocation, 48 lines --Top-----
Welcome to Info version 4.1. Type C-h for help, m for menu item.
```

使用 info 获得命令帮助

用户可以使用↑、↓和PgDn、PgUp键进行翻阅，按q键退出。另外，用户可以使用Ctrl+H键进入info命令的帮助屏幕，学习info命令的更详细的使用方法。

- 显示源文件
- 登录