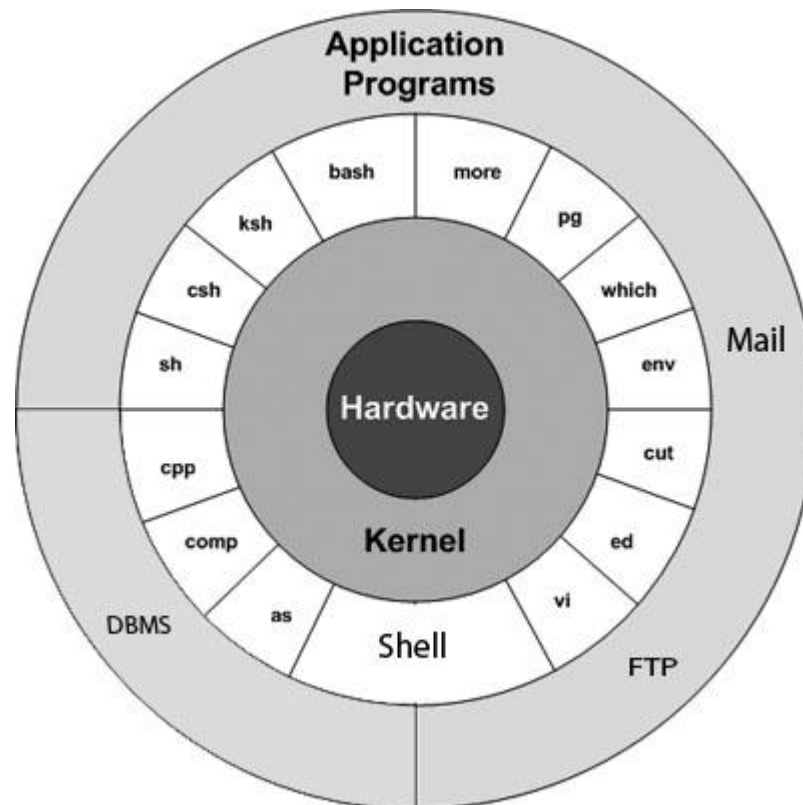


Shell脚本编程基础



内容

①

● Shell概述

- Shell的概念
- Shell的特点
- Shell版本
- Shell程序示例
- Shell程序执行方法

● Linux命令帮手

- 命令行编辑
- screen工具

②

● Shell特殊字符

- 通配符
- 正则表达式
- 引号
- 转义符与路径符
- 输入输出重定向
- 注释和后台命令
- 命令组合符
- 成组命令

③

● Shell变量

- 用户自定义变量
- 位置变量
- Shell预定义变量
- 环境变量

● 算术运算

● 控制结构

● 函数

④

● 作业控制

● Shell内置命令

● Shell脚本调试

● Shell脚本示例

⑤

Shell概述

- Shell的概念
- Shell的特点
- Shell版本
- Shell程序示例
- Shell程序的执行

Shell概念

- Linux shell是连接Linux内核与用户之间的桥梁，是Linux系统的用户界面及Linux命令的解释程序。
- Shell也是一种高级程序设计语言，有变量、关键字、控制结构和函数等程序设计语言要素。



Shell的特点

- 利用控制结构（如判断、循环等）和管道将Linux命令组合起来
- 使用正则表达式
- 可以直接使用shell内置命令，而无须创建新的进程
- 灵活使用数据流，提供管道和输入/输出重定向机制，方便数据处理
- 提供后台执行命令方式
- 用户可以配置环境

Shell版本

- **Bourne shell (sh)**
 - Steven Bourne (AT&T Bell)
- **C shell (csh)**
 - Bill Joy (美国加州大学伯克利分校)
- **Korn shell (ksh)**
 - David Korn (AT&T Bell)
- **Bourne Again shell (bash)**
 - GNU
- **Debian Almquist Shell (dash)**
 - 简化版的bash, Ubuntu默认的shell

Shell程序示例

\$ cat shell_example.sh #显示脚本内容

mkdir animal

cd animal

mkdir bird

touch fox

\$ sh shell_example.sh #运行脚本

Shell程序的执行

4种方式（shell脚本文件名为example.sh）：

- sh <example.sh 或 bash <example.sh
- sh example.sh 或 bash example.sh
- path/example.sh （需有执行权限并在脚本文件第一行指定解释器的路径，如：#!/bin/bash）
- . example.sh 或 source example.sh

CentOS中，sh是bash的符号链接；
Ubuntu中，sh是dash的符号链接

Linux命令帮手： 命令行编辑

- 命令行自动补齐
- 命令历史
- 光标跳转
- 命令编辑

自动补齐

➤ 命令自动补齐

\$ history #输入到his时按Tab键，看看会发生什么

➤ 文件名自动补齐

\$ cat hello.c #输入到cat h时按Tab键，看看会发生什么。如果没有什么发生，就再按一次Tab键看看！

命令历史

- 显示历史命令:

- history [num]
- ↑/ctrl+p, ↓/ctrl+n
- ctrl+r

- 执行历史命令:

- !!或!-1 重复上一条命令
- !n 重新执行第n条历史命令
- !-n 重新执行倒数第n条历史命令
- !string 重新执行以字符串string开头的最近的历史命令
- !?string? 重新执行包含string的最近的历史命令

光标跳转

- Ctrl + a 跳转至命令行首 (Ahead of line)
- Ctrl + e 跳转至命令行尾 (End of line)
- Ctrl + f 向前移动一个字符 (Forward)
- Ctrl + b 向后移动一个字符 (Backward)
- Alt + f 向前跳转到下一个字的第一个字符
- Alt + b 向后跳转到下一个字的第一个字符

命令编辑

- Ctrl + w 删除一个词
- Ctrl + u 删除光标前面的所有字符
- Ctrl + k 删除光标所在位置及其后的所有字符
- Ctrl + d 删除光标所在位置的字符
- Ctrl + c 取消本次命令输入
- Alt + d 删除从光标当前位置，到当前词的结尾字符
- Ctrl + y 粘贴最后一次删除的内容

Linux命令帮手：screen工具

- 我们常需要SSH 或者telnet 远程登录到Linux 服务器，运行一些需要很长时间才能完成的任务，在此期间不能关掉窗口或者断开连接，否则这个任务就会被杀掉，半途而废。
- 我们可以用screen命令解决这个问题。screen命令可以实现当前窗口与任务分离，即使我们离线了，服务器仍在后台运行任务。当我们重新登录服务器，可以读取窗口线程，重新连接任务窗口。

screen的会话与会话窗口



```
$ screen -ls
```

```
There are screens on:
```

```
166. tty1.DESKTOP-QONARUO
```

```
(11/14/22 09:07:58)
```

```
(Attached)
```

```
157. tty1.DESKTOP-QONARUO
```

```
(11/14/22 09:07:45)
```

```
(Detached)
```

```
2 Sockets in /run/screen/S-xiezy.
```

screen的使用方法

会话操作:

- 创建screen会话: `screen`
- 列出screen会话: `screen -ls`
- 恢复screen会话: `screen -r [n]`
- 退出screen会话: `screen -d [n]` #在会话外脱离已绑定（attached）的会话
- 脱离screen会话: `Ctrl+a, d` #在会话中脱离（detach）该会话

会话中的窗口操作:

- 创建会话窗口: `Ctrl+a, c`
- 会话窗口切换:
 - `Ctrl+a, w` 显示所有窗口列表
 - `Ctrl+a, n` 切换到下一个窗口
 - `Ctrl+a, p` 切换到前一个窗口
 - `Ctrl+a, 0..9` 切换到第0..9个窗口
 - `Ctrl+a, [Space]` 由窗口0循序切换到窗口9
 - `Ctrl+a, Ctrl+a` 在两个最近使用的窗口间切换
- 退出会话窗口: `exit`

未完待续
LOADING