曾 晔

Shell编程

DAY01 Shell基础、编写第一个Shell脚本、变量

DAY02 数值运算、条件测试、if选择结构（为脚本提供智能）

DAY03 循环控制（重复操作）、系统服务脚本、中断和退出

DAY04 字符串处理（系统指标提取）、扩展技巧、正则表达式

DAY05 sed文件处理（非交互的vim编辑器）

DAY06 awk文本过滤（类似于grep--行，但是要更强大--列）

DAY07 实战脚本编写

Linux运维 高薪 编程技能（Shell --》Python --》Java/C）

能更加有效率地执行管理任务（偷懒）

Shell语言：不支持小数、多线程不方便

Python语言：支持小数运算、网络编程等高级应用，支持多线程

比如：巡检脚本查100台服务器，挨个查；多线程执行可同时查

Bash Shell环境的一些特点 ——

常用的快捷键：

Ctrl w、Esc . 、Ctrl a、Ctrl e

Tab键自动补全（命令名、路径、选项、软件名、系统服务名、……）

命令别名：

alias 别名='实际执行的命令行' ==》/etc/bashrc

历史命令的记录数量：

/etc/profile ==》适用于所有用户

~/.bash\_profile ==》适用于选中的用户

HISTSIZE=1000

重定向输入：

命令行 < 文件

重定向标准输出：

命令行 > 文件

重定向标准错误：

命令行 2> 文件

分别重定向标准输出、标准错误：

命令行 > 文件A 2> 文件B

合并重定向标准输出、标准错误：

命令行 &> 文件C

--双引号 “ ” ：允许扩展，以$引用其他变量

--单引号 ‘ ’ ： 禁用扩展，即便$也视为普通字符

--反撇号 、 、： 将命令的执行输出作为变量值

$( )与 、 、 等效，但 $( ) 方便嵌套使用

创建脚本的过程：

1.建文件

2.添加可执行语句（正常完成XX任务的命令行组合）

3.设置x权限

执行脚本的方式：

方式1：脚本程序的绝对路径，或相对路径（不能省略文件所在位置）

脚本程序独立运行

方式2：命令解释器 脚本文件

脚本程序被其他解释器加载再执行

任务需求 ——

1. 已知软件仓库位于：

http://classroom/content/rhel7.0/x86\_64/dvd/

2. 编写一个脚本，执行此脚本后能为server0快速配置yum源

清理旧配置：

rm -rf /etc/yum.repos.d/\*.repo

添加新配置：

echo '[rhel7]

name=Red hat Enterprise 7.0

baseurl=http://classroom.example.com/content/rhel7.0/x86\_64/dvd/

gpgcheck=0' > /etc/yum.repos.d/rhel7.repo

任务需求 ——

0. 前提条件：已正确配置好yum

1. 为当前主机快速搭建并启用FTP服务器

2. 当匿名访问此FTP服务器时可以下载 fstab.txt

正常部署FTP的命令行操作：

1）装包 vsftpd

yum -y install vsftpd

2）配置（准备下载资源）

cp /etc/fstab /var/ftp/fstab.txt

3）开启服务

systemctl restart vsftpd

systemctl enable vsftpd

# vim xxxx

.. .. 可执行语句

# chmod +x xxxx

编写一个脚本、编写一个好脚本的难点 ——

任何如何实现（命令行操作）？

命令行的顺序？

如何用更少的命令、更快的命令？（变量、字符传处理、控制结构）

任务需求：

1）创建一个新变量，名称为 var1，值为 CentOS

# var1=CentOS

2）显示变量var1的内容

# echo $var1

3）新建一个文件，文件名称来自于变量var1的值，扩展名为.txt

# touch ${var1}.txt

4）查看环境变量USER、HOME、PATH的值

# echo $USER $HOME $PATH

5）把 first.sh 复制到 /usr/local/bin 目录下，名称为hello

# cp first.sh /usr/local/bin

6）直接执行 hello 命令

# hello

$?：反馈前一条命令的返回状态值（0、非0）

$#：位置变量： 跟在脚本后面提供的参数

$1 第一个参数、$2 第二个参数、……

${10}

任务需求：# myuseradd 用户名

1. 编写一个添加用户账号的脚本 /usr/sbin/myuseradd

2. 执行此脚本时可以提供用户名做参数，执行完成后能够自动添加此用户，并且把登录密码设置为 1234567

#!/bin/bash

useradd $1

echo 1234567 | passwd --stdin $1

任务需求：# myuseradd 用户名 密码

1. 编写一个添加用户账号的脚本 /usr/sbin/myuseradd

2. 执行此脚本时可以提供2个参数，其中第一个参数作为用户名，第二个作为密码，添加相应的用户账号

#!/bin/bash

useradd $1

echo $2 | passwd --stdin $1

任务需求： ./location.sh a b c d e f g h j k l m n

#!/bin/bash

echo 当前的脚本路径和名称是 $0

echo 第1个参数是 $1

echo 第3个参数是 $3

echo 第10个参数是 ${10}

echo 你一共提供了$# 个参数

echo 你提供的所有参数是：

echo $\*

进程和进程间的关系

pstree

变量的生效范围：默认只在当前命令行环境

黄药师（玉箫），不给欧阳锋用，不给黄蓉

如果黄药师希望黄蓉也能使用这把玉箫

export 玉箫

Bash ( 进程 )

任务需求：控制变量的作用范围

1）在当前环境（黄药师）定义新变量 cjb=玉箫

2）新打开bash进程（黄蓉），查看变量 cjb 的值，然后exit退出

看不到

3）在原环境（黄药师）使用export输出变量cjb

# export cjb

4）再次打开bash进程（黄蓉），查看变量 cjb 的值，然后exit退出

看到

问题：欧阳锋无法使用cjb

任务需求：

1. 定义一个所有用户都能使用的变量 KLING="天王盖地虎"

2. 每个用户登录时，能看到欢迎信息"Welcome XXXX"

实现方法：在所有用户登录login之前定义

# vim /etc/profile //环境变量配置文件

.. ..

export KLING="天王盖地虎"

echo "Welcome XXXX"

# su - student

.. ..

Welcome student //显示欢迎信息

$ echo $KLING

天王盖地虎

使用 su - 用户名 命令时

read命令，提示用户从键盘输入一个字符串并且保存到指定的变量

比较时编写交互式的脚本

任务需求：（脚本改造）

# myuseradd 用户名 密码

== 改造为 ==》

# myuseradd

需要添加的用户名是:

需要设置的登录密码是:

#!/bin/bash

read -p "需要添加的用户名是: " username

read -p "需要设置的登录密码是: " password

useradd $username

echo $password | passwd --stdin $username

任务需求：

1. 编写一个模拟登录脚本 login.sh

2. 根据提示输入密码时不显示

3. 把用户提供的用户名和密码记录到一个文件/tmp/rec.txt 内

#!/bin/bash

read -p 'login: ' username

stty -echo

read -p 'pasword: ' password

stty echo

echo "你的登录名是 $username

你的密码是 $password" >> /tmp/rec.txt

命令替换：

# 命令行1 $(命令2)

获取当前的年月日的操作：

date +%F

!!!! 在crotab计划任务记录里，% 表示 注释

小结：

1）Shell基础概念

2）编写一个脚本的基本过程

3）Shell变量的使用（分类、定义/赋值/引用、read、export）

任务需求：

1. 使用tar备份 /home 目录，保存到 /opt/当前日期.tar.gz

2. 这个备份操作每3天执行一次

# vim /root/homebak.sh

tar -zcPf /opt/$(date +%F).tar.gz /home

# chmod +x /root/homebak.sh

# crontab -e

0 0 \*/3 \* \* /root/homebak.sh