04.Shell流程控制

- 04.Shell流程控制
    - 1.流程控制语句if
    - 2.流程控制语句case
    - 3.交互脚本expect

> 徐亮伟, 江湖人称标杆徐。多年互联网运维工作经验，曾负责过大规模集群架构自动化运维管理工作。擅长Web集群架构与自动化运维，曾负责国内某大型电商运维工作。
> 个人博客"徐亮伟架构师之路"累计受益数万人。
> 笔者Q:552408925、572891887
> 架构师群:471443208

1.流程控制语句if

单分支结构

```
if [ 如果你有房 ];then
    我就嫁给你
fi
```

双分支结构

```
if [ 如果你有房 ];then
        我就嫁给你
    else
        再见
fi
```

多分支结构

```
if [ 如果你有房 ];then
        我就嫁给你
elif [ 你有车 ];then
        我就嫁给你
elif [ 你有钱 ];then
        我就嫁给你
    else
```

```
        再见
fi
```

1.实例, 安装 `Nginx`

```
#!/usr/bin/bash

# Install Nginx
# By xuliangwei 2018-05-16

####1.判断网络
ping -c1 www.baidu.com &>/dev/null
if [ $? -ne 0 ];then
        echo "请检查你的网络......"
        exit 1
fi

#2.yum仓库

yum_repo=$(yum repolist|grep nginx|wc -l)

if [ $yum_repo -eq 0 ];then
        cat >/etc/yum.repos.d/nginx.repo <<-EOF
        [nginx]
        name=nginx repo
        baseurl=http://nginx.org/packages/centos/7/x86_64/
        gpgcheck=0
        enabled=1
        EOF
        yum makecache

elif [ $yum_repo -eq 1 ];then
        yum install nginx -y &>/dev/null
        Install_nginx=$(rpm -q nginx|wc -l)
        if [ $Install_nginx -eq 1 ];then
                echo "Nginx已经安装"
        fi
        Nginx_Status=$(systemctl status nginx|grep Active|awk '{print $1 $3}')
        systemctl start nginx &>/dev/null
        if [ $? -eq 0 ];then
                        echo "Nginx已经启动完毕"
                        echo "Nginx当前状态是: $Nginx_Status"
                else
                        echo "Nginx启动失败 $Nginx_Status"
                        pkill -9 httpd &>/dev/null
                        pkill -9 nginx  &>/dev/null
                        systemctl start  nginx
                        if [  $? -eq 0 ];then
                                echo -e "Nginx重新启动成功"
                                Nginx_Status=$(systemctl status nginx|grep Active|a
wk '{print $1 $3}')

                                echo -e "\033[32m $Nginx_Status \033[0m"
                        fi
        fi

else
        echo "不知道什么错误，请手动检查下"
fi
```

2.根据不同的系统安装不同的`yum`源

```
 #!/usr/bin/bash

os_name=$(cat /etc/redhat-release)
os_version=$(cat /etc/redhat-release |awk '{print $4}'|awk  -F '.' '{print $1}')

if [ $os_version = "(Final)" ];then
        os_version=$(cat /etc/redhat-release |awk '{print $3}'|awk  -F '.' '{print
$1}')

fi

if [ $os_version -eq 7 ];then
        mkdir -p /etc/yum.repos.d/backup
        mv /etc/yum.repos.d/*.repo /etc/yum.repos.d/backup
        cat >/etc/yum.repos.d/base.repo<<-EOF
        [base]
        name=Local Base Yum Source
        baseurl=ftp://192.168.56.1/base/7/x86_64
        enable=1
        gpgcheck=0
        EOF
        echo "$os_name 系统已经配置好yum仓库"

elif [  $os_version -eq 6 ];then
        mkdir -p /etc/yum.repos.d/backup
        mv /etc/yum.repos.d/*.repo /etc/yum.repos.d/backup
        wget -O /etc/yum.repos.d/CentOS-Base.repo http://mirrors.aliyun.com/repo/Ce
ntos-6.repo &>/dev/null
        echo "$os_name 系统已经配置好yum仓库"

elif [  $os_version -eq 5 ];then
        mkdir -p /etc/yum.repos.d/backup
        mv /etc/yum.repos.d/*.repo /etc/yum.repos.d/backup
        curl -o /etc/yum.repos.d/CentOS-Base.repo http://mirrors.aliyun.com/repo/Ce
ntos-5.repo &>/dev/null
        echo "$os_name 系统已经配置好yum仓库"
else
        echo "无法检测当前系统版本,请检查/etc/redhat-release"
fi
```

3.安装不同版本的 `PHP`

```
#!/usr/bin/bash
#install php
install_php56() {
    echo "install php5.6......"
}
install_php70() {
    echo "install php7.0......"
}
install_php71() {
    echo "install php7.1......"
}
while :
do
    echo "##################################"
    echo -e "\t1 php-5.6"
    echo -e "\t2 php-7.0"
    echo -e "\t3 php-7.1"
    echo -e "\tq exit"
    echo "##################################"

    read -p "version[1-3]: " version
    if [ "$version" = "1" ];then
        install_php56
    elif [ "$version" = "2" ];then
        install_php70
    elif [ "$version" = "3" ];then
        install_php71
    elif [ "$version" = "q" ];then
        exit
    else
        echo "error"
    fi
done
```

2.流程控制语句case

`case` 语句

```
case 变量 in
模式 1)
    命令序列 1;;
模式 2)
    命令序列 2;;
模式 3)
    命令序列 3 ;;
*)
    无匹配后命令序列
esac
```

1.批量删除用户

```
#!/usr/bin/bash
read -p "请输入需要删除的用户前缀，以及用户的位数: " delname delnum

echo "你将要删除如下账户
        用户前缀是: $delname
        用户的个数: $delnum
"
read -p "你确定要删除吗[y|Y|Yes|n|N|NO]?" reday

for i in $(seq $delnum);do
userfull=$delname$i
        case $reday in
                y|Y|YES)
                        id $userfull &>/dev/null
                        if [ $? -eq 0 ];then
                                userdel $userfull &>/dev/null
                                echo "userdel is ok $userfull...."
                        else
                                echo "$userfull" no such user
                        fi
                        ;;
                n|N|no|NO|No)
                        exit 1
                        ;;
                *)
                read -p "你确定要删除吗[y|Y|Yes|n|N|NO]?" reday

        esac
done
```

2.系统管理工具箱

```
Command action
h 显示命令帮助
f 显示磁盘分区
d 显示磁盘挂载
m 查看内存使用
u 查看系统负载
q 退出程序

#!/usr/bin/bash
caidan(){
        cat <<-EOF
        ===================
        h 显示命令帮助
        f 显示磁盘分区
        d 显示磁盘挂载
        m 查看内存使用
        u 查看系统负载
        q 退出程序
        =====================
        EOF
}

while true
do
        read -p "请输入你想查看系统状态对应码[d/m/u/q]: " sys
        case "$sys" in
                h)
                        clear
                        caidan
                        ;;
                f)
                        clear
                        lsblk
                        ;;
                d)
                        clear
                        df -h
                        ;;
                m)
                        clear
                        free -m
                        ;;
                u)
                        clear
                        uptime
```

```
				;;
			q)
				break
				;;
			*)
				echo "error"
				exit 1;
		esac
done
```

3.实现简单的 `JumpServer`

```
#!/usr/bin/bash

#jumpServer

Mysql_master=192.168.70.160
Mysql_slave1=192.168.70.161
Mysql_slave2=192.168.70.162
Nginx_Up=192.168.70.150
Nginx_WEB1=192.168.70.151
Nginx_WEB2=192.168.56.11

meminfo(){
	cat <<-EOF
	-------------------------------
	|	1) mysql-master		|
	|	2) mysql-slave1		|
	|	3) mysql-slave2		|
	|	4) Nginx-Upstream	|
	|	5) Nginx-WebNode1	|
	|	6) Nginx-WebNode2	|
	|	h) help			|
	---------------------------------
	EOF
}
	#调用函数打印菜单
	meminfo
	#控制不让输入ctrl+c,z
	trap "" HUP INT TSTP
while true
do
	read -p "请输入要连接的主机编号： " num
	case $num in
		1|mysql-master)
			ssh root@$Mysql_master
			;;
```

```
                2|Mysql_slave1)
                         ssh root@$Mysql_slave1
                        ;;
                3|Mysql_slave2)
                        ssh root@$Mysql_slave2
                        ;;
                h|help)
                        clear
                        meminfo
                        ;;
                #退出脚本后门，不要让其他人知道
                exec)
                        break
                        ;;
        esac
done

//无论使用登陆式shell或非登陆式shell都会执行该脚本，前提root用户不允许登陆
[root@Shell day03]# cat /home/alex/.bashrc
sh /home/alex/jumpserver.sh
```

5.使用`case`编写服务启动与停止脚本

```
#!/usr/bin/bash
# manager Nginx start stop restart reload
source  /etc/init.d/functions

act=$1
te(){
if [ $? -eq 0 ];then
                action "Nginx Is $act" /bin/true
        else
                action "Nginx Is $act" /bin/false
fi
}

start(){
        /usr/sbin/nginx &>/dev/null
        te


}
stop(){
        /usr/sbin/nginx -s stop &>/dev/null
        te
}

reload(){
        /usr/sbin/nginx -s reload
        te
}

status(){
        Ngx_status=$(ps aux|grep "[n]ginx"|egrep -v "vi|sh"|grep master|awk '{print
 $2}')
        Nginx_Status_Port=$(netstat -lntp|grep nginx|awk '{print $4}')
        echo "Nginx_status_Pid: $Ngx_status"
        echo "Nginx_status_Port: $Nginx_Status_Port"
}

case $1 in
        start)
                start
                ;;
        stop)
                stop
                ;;
        restart)
                stop
                sleep 1
                start
                ;;
        reload)
                reload
                ;;
        status)
                status
                ;;
        *)
                echo "Usage: $0 {start|stop|status|restart|reload|}"
esac
```

6.使用 `case` 实现多级菜单

3.交互脚本expect

1. `expect` 实现简单的交互登陆

```
#!/usr/bin/expect
spawn ssh root@192.168.70.161

expect {
    "yes/no" { send "yes\r"; exp_continue }
    "password:" { send "centos\r" };
}
interact
```

2. `expect` 定义变量实现交互方式

```
#!/usr/bin/expect
set ip 192.168.70.161
set user root
set password centos
set timeout 5

spawn ssh $user@$ip

expect {
    "yes/no" { send "yes\r"; exp_continue }
    "password:" { send "$password\r" };
}
#交互方式
interact
```

3. `expect` 进行参数传递，执行命令或其他操作

```
#!/usr/bin/expect
#位置传参
set ip [lindex $argv 0]
set user root
set password centos
set timeout 5

spawn ssh $user@$ip

expect {
    "yes/no" { send "yes\r"; exp_continue }
    "password:" { send "$password\r" };
}

#当出现#号符执行如下命令
expect "#"
send "useradd bgx\r"
send "pwd\r"
send "exit\r"
expect eof
```

4.批量获取在线主机, 进行秘钥批量分发

```
cat for_ip.sh
#!/usr/bin/bash

#setup1 拿到IP地址
>ip.txt
for i in {160..162}
do
        ip=192.168.70.$i
        {
        ping -c1 -W1 $ip &>/dev/null
        if [ $? -eq 0 ];then
                echo "$ip" >> ip.txt
        fi
        }&
done
#2.生成对应的密钥
        if [ ! -f ~/.ssh/id_rsa ];then
                ssh-keygen -P "" -f ~/.ssh/id_rsa
        fi

#3.批量分发密钥
        while read line
        do
                /usr/bin/expect <<-EOF
                        set pass 1
                        set timeout 2
                        spawn ssh-copy-id  $line -f
                        expect {
                                "yes/no" { send "yes\r"; exp_continue}
                                "password:" { send "1\r"}
                        }
                        expect eof
                EOF
        done<ip.txt
```