一、Shell脚本应用（一）

1.shell脚本的概念

命令按顺序 > 文件 > 执行权限 ==== Shell脚本

2.Shell脚本应用场景

重复性操作
批量事务处理
自动化运维
服务运行状态
定时任务执行
减轻管理员负担
规避费时操作

3.Shell的概念

shell---特殊的应用程序（软件程序）；用户，系统内核；命令解释器“翻译官”

shell脚本----文件　　----依赖于shell解释器

1.当前系统支持shell解释器

```
[root@ns1 ~]# cat /etc/shells
/bin/sh
/bin/bash                    # linux默认使用程序；IBM AIX(Uinix)默认使用/bin/ksh
/sbin/nologin
/usr/bin/sh
/usr/bin/bash
/usr/sbin/nologin
/bin/tcsh
/bin/csh
```

2.系统默认shell解释器

```
[root@ns1 ~]# echo $SHELL
/bin/bash
```

3.切换shell解释器

```
[root@ns1 ~]# /bin/csh
```

或者[root@ns1 ~]# csh

4.切换到子shell

```
[root@ns1 ~]# bash
[root@ns1 ~]# pstree
```

Shell也有父子关系

sshd———sshd———bash———bash———pstree

注意：系统默认分配一个shell，bash是子shell

5.shell脚本模板

```
#!/bin/bash          #声明解释器，生效
```

后面的加#的信息不生效

每写一段脚本之前都要#注释

6.编写Shell脚本的格式：

1、#!/bin/bash　　表示脚本通过以/bin/bash程序来编写

2、#　表示注释信息，例如：对编写的脚本作用进行解释，每写一段脚本之前都应该用“#”来注释以下命令执行的结果

3、定义脚本中的变量

4、定义脚本中的函数

5、脚本执行语句（判断、循环、选择等语句）

6、**利用　echo　定义输出一些让人更容易看懂得信息**（可以是中文也可以是英文）

案例：小脚本

```
[root@ns1 ~]# vim baby.sh
#!/bin/bash
#the second shell i write
cd /boot
echo -n "当前路径为："　　#　-n　直接在后面显示，echo是回显信息的输出
pwd
echo　　# 单写echo，直接换行显示
echo "查看vml*开头的文件"
ls -lh vml*
```

注意：echo和命令的前后关系，即提示信息与命令的输出结果的前后关系

7.执行脚本的三种方法

第一种：依据脚本路径的执行方法，需要执行权限，打开子shell环境

1. [root@ns1 ~]# chmod u+x baby.sh
2. [root@ns1 ~]# /root/baby.sh
3. [root@ns1 ~]# ./baby.sh

第二种：不需要执行权限，打开子shell环境

1. [root@ns1 ~]# sh baby.sh
2. [root@ns1 ~]# bash baby.sh

第三种：当前shell执行

1. [root@ns1 ~]# source baby.sh
2. [root@ns1 ~]# . baby.sh

8.脚本后缀名

-sh：编写程序为shell

-pl：编写程序为Perl

-py：编写程序为Python

9.查看/etc/init.d/下的服务控制脚本

```
[root@ns1 init.d]# vim functions
[root@ns1 init.d]# cat functions
```

10.创建本地yum仓库的脚本

1.还原yum初始化环境

```
[root@ns1 ~]# cd /etc/yum.repos.d
[root@ns1 yum.repos.d]# ls
backup  epel-release-latest-7.noarch.rpm  httpd-2.4.37.tar.gz  local.repo  yum-bak
[root@ns1 yum.repos.d]# mv backup/* ./
[root@ns1 yum.repos.d]# ls
backup             CentOS-Debuginfo.repo  CentOS-Sources.repo
httpd-2.4.37.tar.gz
CentOS-Base.repo  CentOS-fasttrack.repo  CentOS-Vault.repo
local.repo
CentOS-CR.repo    CentOS-Media.repo      epel-release-latest-7.noarch.rpm  yum-bak
[root@ns1 yum.repos.d]# rm -rf backup
```

2.编写脚本

```
[root@ns1 ~]# vim yum_cr.sh
    #!/bin/bash
    #创建本地yum仓库的脚本
    #挂载光盘
    mkdir /media/cdrom
    mount /dev/sr0 /media/cdrom
    #准备repo文件
    cd /etc/yum.repos.d/
    mkdir bakup
    mv *.repo bakup
    echo '[xxx]
```

```
    name=xxx
    baseurl=file:///media/cdrom
    enabled=1
    gpgcheck=0' > local.repo
    #清理及创建缓存
    yum clean all && yum makecache fast
[root@ns1 ~]# bash yum_cr.sh
```

11.DNS正向解析的脚本

```
[root@ns1 ~]# vim DNS.sh
    #!/bin/bash
    #这是一个DNS正向解析的脚本

    #安装bind相关软件包
    echo "正在安装中..."
    yum -y install bind bind-utils bind-libs

    #修改主配置文件/etc/named.conf
    echo 'options {
          directory    "/var/named";
    };

    zone "sky.com" IN {
          type master;
          file "sky.zheng";
    };' > /etc/named.conf

    #配置正向解析区域文件
    echo '$TTL 86400
    @     IN     SOA     sky.com.     admin.sky.com.     (
                  202031712
                  3H
                  15M
                  1W
                  1D
```

```
)
        IN      NS      ns.sky.com.
ns      IN      A       192.168.200.103
www     IN      A       192.168.200.103' > /var/named/sky.zheng

chgrp named /var/named/sky.zheng

#启动服务
systemctl restart named
systemctl enable named
systemctl status named

#修改resolv.conf文件
echo "nameserver 192.168.200.103" > /etc/resolv.conf
echo

#进行测试
echo "
==========================================================================
正在测试中..."
nslookup www.sky.com
```

```
[root@ns1 ~]# bash DNS.sh
```

4.重定向和管道（修改脚本1）

1.交互式硬件设备文件

standard (input output error)

类型	设备文件	文件描述编号	默认设备
标准输入	/dev/stdin	0	键盘
标准输出	/dev/stdout	1	显示器
标准错误输出	/dev/stederr	2	显示器

标准输入----键盘

```
[root@ns1 ~]# useradd zhangsan
```

标准输出----显示器

```
[root@ns1 ~]# passwd zhangsan
更改用户 zhangsan 的密码 。
新的 密码：
无效的密码： 密码少于 8 个字符
重新输入新的 密码：
passwd：所有的身份验证令牌已经成功更新。
标准错误输出----显示器
[root@ns1 ~]# ipconfig
bash: ipconfig: 未找到命令...
```

2.重定向操作

类型	操作符	用途
重定向输入	**<**	从指定的文件读取数据，不是从键盘输入
重定向输出	**>**	覆盖文件原有内容
	>>	在原文件末尾追加
标准错误输出	**2>**	将前面命令输出的错误信息重定向到指定文件夹
	2>>	将前面命令输出的错误信息追加到指定文件夹
混合输出	**&>**	不管前面命令输出的命令是否正确都重定向到指定文件夹

示例：

```
重定向输入
[root@ns1 ~]# vim pass.txt
    123
[root@ns1 ~]# passwd --stdin zhangsan < pass.txt
重定向输出
[root@ns1 ~]# echo "hahha" > file.txt
[root@ns1 ~]# echo "heheh" > file.txt
[root@ns1 ~]# echo "hahha" >> file.txt
标准错误输出
[root@ns1 ~]# ipconfig 2> file.txt
[root@ns1 ~]# cat file.txt
    bash: ipconfig: 未找到命令...
```

```
[root@ns1 ~]# ipconfig 2>> file.txt
[root@ns1 ~]# cat file.txt
    bash: ipconfig: 未找到命令...
    bash: ipconfig: 未找到命令...
```

注意：只有当前边的命令报错，后面才会重定向

混合输出

```
[root@ns1 ~]# make &> file.txt
[root@ns1 ~]# cat file.txt
    make: *** 没有指明目标并且找不到 makefile。 停止。
```

注意：不管你的命令输出是正确还是失败都要放到指定文件里面

3.处理多行输出

更改脚本（第一次）

echo -----> cat << EOF > file.txt EOF

```
[root@ns1 ~]# vim DNS.sh
    #!/bin/bash
    #这是一个DNS正向解析的脚本
    #安装bind相关软件包
    echo "正在安装中..."
    yum -y install bind bind-utils bind-libs
    #修改主配置文件/etc/named.conf
    cat << END > /etc/named.conf
    options {
          directory    "/var/named";
    };
    zone "sky.com" IN {
          type master;
          file "sky.zheng";
    };
    END
    #配置正向解析区域文件
    cat << EOF > /var/named/sky.zheng
```

```
\$TTL 86400                                    #  注意这里的转义符号；这里注意
换行
@       IN      SOA     sky.com.        admin.sky.com.      (
                202031716
                3H
                15M
                1W
                1D
)
        IN      NS      ns.sky.com.
ns      IN      A       192.168.200.103
www     IN      A       192.168.200.103
EOF
chgrp named /var/named/sky.zheng
#启动服务
systemctl restart named
systemctl enable named
systemctl status named
#修改resolv.conf文件
cat << END > /etc/resolv.conf
nameserver 192.168.200.103
END
echo
#进行测试
echo "
------------------------------------------------
正在测试中..."
nslookup www.sky.com
```

4.管道符 |

将前面的命令的输出结果作为后面命令的参数使用

5.awk命令初体验

noh取消查找高亮显示

1.格式：

awk -F"分隔符" '/过滤文本/{print $1}' 处理文件

注意：分隔符可以是查找的文件里的任意符号

默认分隔符为空格，可以省略

过滤的时候也可以挑尾部不同的作为关键词过滤，找简单点的

2.练习

- **过滤IP**

[root@ns1 ~]# ifconfig ens32 | awk -F ' ' '/inet /{print $2}'

- **从 df -Th 命令结果中提取 / 分区的已用百分比（数字部分）**

1. df -Th | awk '/\/$/{print $6}' | awk -F% '{print $1}'
2. df -Th | awk -F'[%]+' '/\/$/{print $6}'
3. 另一个关键词过滤：df -Th | awk -F '[M]+' '/\/dev$/{print $5}'
4. df -Th | awk -F ' ' '/\/dev\/mapper\/centos-root/{print $6}' | awk -F '%' '{print $1}'

----这个过滤的关键词选复杂了，可以选/$,意思是以 / 结尾，那么就有 ^ 开头的过滤

5.Shell变量（修改脚本2）

1.常量

固定字符串；固定含义

2.变量类型

自定义变量：本地变量

定义格式：

变量名=变量值 # 等号两边没有空格

变量名：必须字母或下划线开头，无特殊字符，区分大小写

echo查看变量值，$后是变量名或者${变量名}

```
[root@ns1 ~]# name=liruifang
[root@ns1 ~]# echo $name
liruifang
```

更改脚本（第二次）

加入变量$，加强脚本灵活性

```
[root@ns1 ~]# vim DNS.sh
#!/bin/bash
#这是一个DNS正向解析的脚本
IP="192.168.200.103"
```

```
#安装bind相关软件包
echo "正在安装中..."
yum -y install bind bind-utils bind-libs
#修改主配置文件/etc/named.conf
cat << END > /etc/named.conf
options {
      directory    "/var/named";
};
zone "sky.com" IN {
      type master;
      file "sky.zheng";
};
END
#配置正向解析区域文件
cat << EOF > /var/named/sky.zheng
\$TTL 86400
@     IN      SOA      sky.com.      admin.sky.com.     (
              202031716
              3H
              15M
              1W
              1D
)
      IN    NS      ns.sky.com.
ns    IN    A       $IP
www   IN    A       $IP
EOF
chgrp named /var/named/sky.zheng
#启动服务
systemctl restart named
systemctl enable named
systemctl status named
#修改resolv.conf文件
cat << END > /etc/resolv.conf
nameserver $IP
```

```
    END
```

3.案例

```
[root@ns1 ~]# product=cloud
[root@ns1 ~]# version=2.0
[root@ns1 ~]# echo $product
    cloud
[root@ns1 ~]# echo $version
    2.0
[root@ns1 ~]# echo $product $version              # 有无空格都可以，只识别$
    cloud 2.0
[root@ns1 ~]# echo $product 3.0
    cloud 3.0
[root@ns1 ~]# echo $product3.0
    .0
[root@ns1 ~]# echo "$product"3.0
    cloud3.0
[root@ns1 ~]# echo $product"3.0"
    cloud3.0
[root@ns1 ~]# echo $product\3.0
    cloud3.0
[root@ns1 ~]# echo $product'3.0'                  # 单引号不能放在$product前面，会把它识别为普通字符输出
    cloud3.0
[root@ns1 ~]# echo ${name}
    liruifang
[root@ns1 ~]# echo ${product}3.0
    cloud3.0
```

4.特殊符号

1.双引号（" "）

可引用变量

```
[root@ns1 ~]# echo $product"3.0"
    cloud3.0
```

2.单引号（' '）

引号内视为完整字符串，不能引用变量

```
[root@ns1 ~]# echo $product'3.0'
    cloud3.0
```

3.反撇号（``）

普通操作：

```
[root@ns1 ~]# which mkdir
/usr/bin/mkdir
[root@ns1 ~]# rpm -qf /usr/bin/mkdir
coreutils-8.22-21.el7.x86_64
```

快捷操作：

```
[root@ns1 ~]# rpm -qf `which mkdir`
coreutils-8.22-21.el7.x86_64
[root@ns1 ~]# rpm -qi `rpm -qf `which mkdir``
```

问题：``不易识别，容易报错　==$()

```
rpm -qf $(which mkdir)
[root@ns1 ~]# rpm -qi $(rpm -qf $(which mkdir))
[root@ns1 ~]# ifconfig ens32 | awk '/inet /{print $2}'
```

6.read命令（修改脚本3）

交互式设置变量的方法

```
IP=$(ifconfig ens32 | awk '/inet /{print $2}')
read -p"请输入你要设置的域名(example: sky.com)" dn
domain name    ————>    dn
```

更改脚本（第三次）

自动识别本机IP，省去写IP的过程；

交互式read ，将域名变量化

进一步智能化

```
[root@ns1 ~]# vim DNS.sh

    #!/bin/bash
```

```
#这是一个DNS正向解析的脚本
IP=$(ifconfig ens32 | awk '/inet /{print $2}')
read -p"请输入你要设置的域名(example: sky.com):" dn
#安装bind相关软件包
echo "正在安装中..."
yum -y install bind bind-utils bind-libs
#修改主配置文件/etc/named.conf
cat << END > /etc/named.conf
options {
      directory    "/var/named";
};
zone "$dn" IN {
      type master;
      file "$dn.zheng";
};
END
#配置正向解析区域文件
cat << EOF > /var/named/$dn.zheng
\$TTL 86400
@     IN      SOA       $dn.      admin.$dn.     (
                202031716
                3H
                15M
                1W
                1D
)
      IN    NS      ns.$dn.
ns    IN    A       $IP
www   IN    A       $IP
EOF
chgrp named /var/named/$dn.zheng
#启动服务
systemctl restart named
systemctl enable named
systemctl status named
```

```
#修改resolv.conf文件
cat << END > /etc/resolv.conf
nameserver $IP
END
echo
#进行测试
echo "
--------------------------------------------------
正在测试中..."
nslookup www.$dn
```

模拟登陆界面的案例脚本

```
[root@ns1 ~]# vim login.sh
#!/bin/bash
cat << END
welcome ......
centos  ......
kernel  ......
END
read -p"localhost login: " user
if [ $user == "ruifang" ]
then
        echo "哈哈，欢迎小美女~"
else
        echo "哪凉快哪待着去"
```

7.设置变量生效范围

新定义的变量（局部变量，本地变量），只能在当前Shell环境中生效

export命令：将局部变量发布到全局 ------只能向子Shell发，不能向上一级发

```
[root@localhost ~]# name=zhangsan
[root@localhost ~]# echo $name
    zhangsan
[root@localhost ~]# bash
[root@localhost ~]# echo $name
```

```
[root@localhost ~]# exit
    exit
[root@localhost ~]# export name
[root@localhost ~]# bash
[root@localhost ~]# echo $name
    zhangsan
```

定义变量的同时发布全局：

```
[root@localhost ~]# export address=beijing
[root@localhost ~]# bash
[root@localhost ~]# echo $name
zhangsan
[root@localhost ~]# echo $address
beijing
```

8.数值变量的运算

1.expr命令

简单的整数运算，不支持小数运算

```
[root@localhost ~]# a=100
[root@localhost ~]# b=33
[root@localhost ~]# expr $a + $b
    133
[root@localhost ~]# expr $a - $b
    67
[root@localhost ~]# expr $a / $b
    3
[root@localhost ~]# expr $a \* $b
    3300
[root@localhost ~]# expr $a % $b
    1
```

2.shell本身的运算命令

```
[root@localhost ~]# echo $((100+33))
133
[root@localhost ~]# echo $[100+33]
```

```
    133
```

3.特殊计算器

```
[root@localhost ~]# echo "100/33" | bc
    3
[root@localhost ~]# echo "scale=2;100/33" | bc
    3.03
```

9.特殊的Shell变量

1.环境变量

env命令：查看

设置用户工作环境

/etc/profile　　:全局

.bash_profile　　：局部

```
[root@ns1 ~]# echo $name
    zhangsan
[root@ns1 ~]# vim /etc/profile
    export name=zhangsan
[root@ns1 ~]# vim .bash_profile
    export name=lisi
```

如果两个都配置，那么将显示lisi,显示最后加载的那个文件

set、 env、 export、 declare

unset　：取消变量

```
    [root@ns1 ~]# game=wangzhe
    [root@ns1 ~]# echo $game
    wangzhe
    [root@ns1 ~]# unset game
    [root@ns1 ~]# echo $game
```

2.位置变量

作用：给脚本传参

　　位置变量=位置参数　　$1、$2、$3、.....${10}

```
[root@ns1 ~]# /etc/init.d/network
用法：/etc/init.d/network {start|stop|status|restart|reload|force-reload}
```

```
[root@ns1 ~]# /etc/init.d/network start
Starting network (via systemctl):                          [  确定  ]
```

案例：

```
[root@ns1 ~]# vim sum.sh
[root@ns1 ~]# bash sum.sh 12 35
    12 + 35 = 47
[root@ns1 ~]# cat sum.sh
    #!/bin/bash
    sum=$(expr $1 + $2)
    echo "$1 + $2 = $sum"
```

修改DNS脚本：

```
[root@ns1 ~]# vim DNS.sh
#read -p"请输入你要设置的域名(example: sky.com):" dn
dn="$1"
[root@ns1 ~]# bash DNS.sh aaa.com
```

3.预定义变量----用的不频繁

```
$*          所有位置参数内容          整体
$@        所有位置参数内容          局部
$0          脚本名
$#          位置参数的个数
$?          判断上一条命令的成功与否；0正确；非0错误
$$          PID号
```

```
[root@ns1 ~]# ifconfig ens32 &> /dev/null
[root@ns1 ~]# echo $?
    0
[root@ns1 ~]# ifconfig ens33 &> /dev/null
[root@ns1 ~]# echo $?
    1
```

案例：

```
[root@ns1 ~]# vim yu.sh
#!/bin/bash
```

```
echo "当前的位置变量是： $1"
echo "当前的位置变量是： $2"
echo "当前的位置变量是： $3"
echo "当前的位置变量是： $4"
[root@ns1 ~]# bash yu.sh 11 22 33 44 55
当前的位置变量是： 11
当前的位置变量是： 22
当前的位置变量是： 33
当前的位置变量是： 44

[root@ns1 ~]# vim yu.sh
echo "当前的位置变量是： $1"
echo "当前的位置变量是： $2"
echo "当前的位置变量是： $3"
echo "当前的位置变量是： $4"
echo "所有位置参数的内容: $*"
[root@ns1 ~]# bash yu.sh 11 22 33 44 55
当前的位置变量是： 11
当前的位置变量是： 22
当前的位置变量是： 33
当前的位置变量是： 44
所有位置参数的内容: 11 22 33 44 55

 [root@ns1 ~]# vim yu.sh
#!/bin/bash
echo "当前的位置变量是： $1"
echo "当前的位置变量是： $2"
echo "当前的位置变量是： $3"
echo "当前的位置变量是： $4"
echo "所有位置参数的内容: $*"
echo "脚本名： $0"
[root@ns1 ~]# bash yu.sh 11 22 33 44 55 66
当前的位置变量是： 11
当前的位置变量是： 22
当前的位置变量是： 33
```

当前的位置变量是： 44
所有位置参数的内容: 11 22 33 44 55 66
脚本名：yu.sh
用法：yu.sh 参数1 参数2 参数3 参数4 ...

```
[root@ns1 ~]# vim yu.sh
#!/bin/bash
echo "当前的位置变量是： $1"
echo "当前的位置变量是： $2"
echo "当前的位置变量是： $3"
echo "当前的位置变量是： $4"
echo "所有位置参数的内容: $*"
echo "脚本名：$0"
echo "位置参数的个数：$#"
echo "用法：$0 参数1 参数2 参数3 参数4 ..."
[root@ns1 ~]# bash yu.sh 11 22 33 44 55 66
当前的位置变量是： 11
当前的位置变量是： 22
当前的位置变量是： 33
当前的位置变量是： 44
所有位置参数的内容: 11 22 33 44 55 66
脚本名：yu.sh
位置参数的个数：6
用法：yu.sh 参数1 参数2 参数3 参数4 ...
```

4.date命令

1.作用：备份

给文件名里嵌入时间戳

```
[root@ns1 ~]# cp /etc/named.conf /etc/named.conf-$(date +%F)
[root@ns1 ~]# ls /etc/named.conf
    /etc/named.conf
[root@ns1 ~]# ls /etc/named.conf-2020-03-18
    /etc/named.conf-2020-03-18
[root@ns1 ~]# cp /etc/named.conf /etc/named.conf-$(date +%s)
[root@ns1 ~]# cp /etc/named.conf /etc/named.conf-$(date +%s)
```

```
[root@ns1 ~]# ll /etc/named*
    -rw-r----- 1 root named  120 3月  19 2020 /etc/named.conf
    -rw-r----- 1 root root   120 3月  18 22:27 /etc/named.conf-1584541657
    -rw-r----- 1 root root   120 3月  18 22:28 /etc/named.conf-1584541732
    -rw-r----- 1 root root   120 3月  18 22:26 /etc/named.conf-2020-03-18
    -rw-r----- 1 root root   120 3月  18 22:26 /etc/named.conf.bak
    -rw-r----- 1 root named 1806 8月   8 2019 /etc/named.conf.rpmnew
    -rw-r--r-- 1 root named 3923 8月   8 2019 /etc/named.iscdlv.key
    -rw-r----- 1 root named  931 6月  21 2007 /etc/named.rfc1912.zones
    -rw-r--r-- 1 root named 1886 4月  13 2017 /etc/named.root.key
    /etc/named:
    总用量 4
    -rw-r--r-- 1 root root 268 3月  18 19:03 sky.zheng


[root@ns1 ~]# date +'%F'
2020-03-18
[root@ns1 ~]# date +'%T'
22:16:47
[root@ns1 ~]# date +'%Y'          年
2020
[root@ns1 ~]# date +'%m'          月
03
[root@ns1 ~]# date +'%d'           日
18
[root@ns1 ~]# date +'%H'          时
22
[root@ns1 ~]# date +'%M'         分
18
[root@ns1 ~]# date +'%S'         秒
25
[root@ns1 ~]# date +'%Y-%m-%d %H-%M-%S'
2020-03-18 22-20-43
```

2.1970年到当前时间的秒数:

```
[root@ns1 ~]# date +%s
    1584541338
```

3.随机数：

```
[root@ns1 ~]# date +%N
501753257
```

4.获取昨天的年月日：

```
[root@ns1 ~]# date +%F -d "-1 day"
2020-03-17
```