**第二十章 条件测试语句和if流程控制语句的使用**

**本节所讲内容：**

**20.1 read命令键盘读取变量的值**

**20.2 流程控制语句if**

**20.3 test测试命令**

**20.4 流程控制过程中复杂条件和通配符**

**20.5 实战-3个shell脚本实战**

# 20.1 read命令键盘读取变量的值

**从键盘读取变量的值，通常用在shell脚本中与用户进行交互的场合。该命令可以一次读取多个变量的值，变量和输入的值都需要使用空格隔开。在read命令后面，如果没有指定变量名，读取的数据将被自动赋值给特定的变量REPLY**

**read从键盘读入数据，赋给变量**

**例1：**

**[root@xuegod63 ~]# read a b**

**hello world**

**[root@xuegod63 ~]# echo $a $b**

**hello world**

## 2.1.1 read常用见用法及参数

**例1：从标准输入读取一行并赋值给变量passwd**

**[root@xuegod63 ~]# read passwd**

**例2：读取多个值，从标准输入读取一行，直至遇到第一个空白符或换行符。把用户键入的第一个词存到变量first中，把该行的剩余部分保存到变量last中**

**[root@xuegod63 ~]# read firt last**

**aaaa bbbb**

**例3：read -s passwd 将你输入的东西隐藏起来，值赋给passwd。这个用户隐藏密码信息**

**[root@xuegod63 ~]# read -s passwd**

**[root@xuegod63 ~]# echo $passwd**

**123456**

**例4：输入的时间限制**

**[root@xuegod63 ~]# read -t 2 time #超过两秒没有输入，直接退出**

**例5：输入的长度限制**

**[root@xuegod63 ~]# read -n 2 test #最多只接受2个字符**

**例6：使用-r参数输，允许让输入中的内容包括：空格、/、\、 ？等特殊字符串。**

**[root@xuegod63 ~]# read -r line**

**sdf sdf / sdfs /n**

**[root@xuegod63 ~]# echo $line**

**sdf sdf / sdfs /n**

**例7：-p 用于给出提示符，在前面的例子中我们使用了echo –n “…“来给出提示符**

**方法1：**

**[root@xuegod63 ~]# read -p "please input: " pass**

**please input: 123456**

**[root@xuegod63 ~]# echo $pass**

**123456**

**方法2：**

**[root@xuegod63 ~]# echo -n "please input: " ; read pass**

**please input: 123456**

**[root@xuegod63 ~]# echo $pass**

**123456**

**例8：read 综合实例**

**[root@xuegod63 ~]# vim test-read.sh #写入以下内容**

**#!/bin/bash**

**read -p "请输入姓名：" NAME**

**read -p "请输入年龄：" AGE**

**read -p "请输入性别：" SEX**

**cat<<eof**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**你的基本信息如下：**

**姓名: $NAME**

**年龄：$AGE**

**性别：$SEX**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**eof**

**[root@xuegod63 ~]# sh test-read.sh**

**请输入姓名：xuegod**

**请输入年龄：111**

**请输入性别：man**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**你的基本信息如下：**

**姓名: xuegod**

**年龄：111**

**性别：man**

# 20.2 流程控制语句if

## 20.2.1 语法格式：

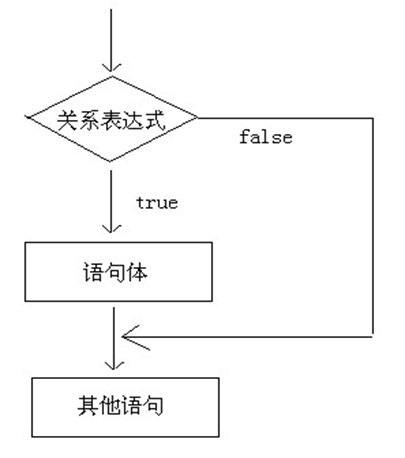
**if 条件**

**then**

**commands**

**fi**

**if语句流程图：**

****

**注：根据我们的命令退出码来进行判断(echo $? =0)，如果是0,那么就会执行then后面的命令**

**例1：**

**[root@xuegod63 ~]# vim if-1.sh**

**#!/bin/bash**

**if ls /mnt**

**then**

**echo "it's ok"**

**fi**

**[root@xuegod63 ~]# bash !$**

**bash if-1.sh**

**CentOS\_BuildTag GPL LiveOS RPM-GPG-KEY-CentOS-7**

**EFI images Packages RPM-GPG-KEY-CentOS-Testing-7**

**EULA isolinux repodata TRANS.TBL**

**it's ok**

## 20.2.2 双分支if语句

**语法格式：**

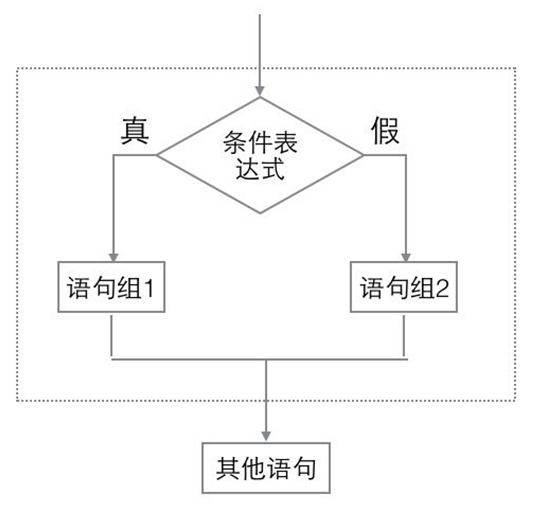
**if command ; then**

**commands**

**else**

**commands**

**fi**

****

**例2：**

**[root@xuegod63 ~]# cat if-2.sh**

**#!/bin/bash**

**if grep root /etc/passwd ; then**

**echo "it's ok"**

**else**

**"it's err"**

**fi**

**[root@xuegod63 ~]# sh if-2.sh**

**root:x:0:0:root:/root:/bin/bash**

**operator:x:11:0:operator:/root:/sbin/nologin**

**it's ok**

**例3：**

**[root@xuegod63 ~]# cat if-3.sh**

**#!/bin/bash**

**if grep xuegod /etc/passwd ;then**

**echo "it's ok"**

**else**

**echo "it's err"**

**fi**

**[root@xuegod63 ~]# sh if-3.sh**

**it's err**

## 20.2.3 多分支if语句

**语法结构：**

**if条件测试操作1 ; then**

**commands**

**elif 条件测试操作2 ; then**

**commands**

**elif 条件测试操作3 ; then**

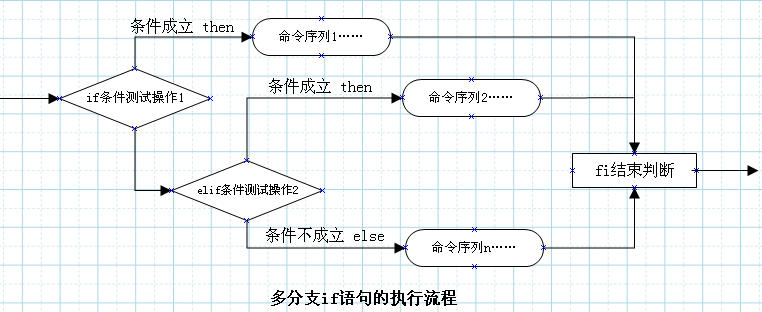
**commands**

**.......**

**else**

**commands**

**fi**

****

**例4：判断用户在系统中是否存在，是否有家目录**

**[root@xuegod63 ~]# cat if-4.sh**

**#!/bin/bash**

**read -p "input a user:" tu**

**if grep $tu /etc/passwd ; then**

**echo "the user $tu exists on this system"**

**elif ls -d /home/$tu ; then**

**echo "the user $tu not exists on this system"**

**echo "$tu has a home directory"**

**else**

**echo "the user $tu not exists on this system"**

**echo "$tu not has a direcotry"**

**fi**

**[root@xuegod63 ~]# sh if-4.sh**

**Input a user: hr**

**chrony:x:994:990::/var/lib/chrony:/sbin/nologin**

**hr:x:1000:1000:hr:/home/hr:/bin/bash**

**the user hr exists on this system**

**[root@xuegod63 ~]# sh if-4.sh**

**Input a user: xuegod**

**/home/xuegod**

**xuegod has a directory**

# 20.3 test测试命令

**Shell中的 test 命令用于检查某个条件是否成立，它可以进行数值、字符和文件三个方面的测试**

**格式：test 测试条件**

**如果结果是对的，也叫结果为真，用$?=0表示，反之为假，用非0表示**

## 20.3.1 数值比较

|  |  |  |
| --- | --- | --- |
| **参数** | **说明** | **示例** |
| **-eq** | **等于则为真** | **[ “$a” -eq “$b” ]** |
| **-ne** | **不等于则为真** | **[ “$a” -ne “$b” ]** |
| **-gt** | **大于则为真** | **[ “$a” -gt “$b” ]** |
| **-ge** | **大于等于则为真** | **[ “$a” -ge “$b” ]** |
| **-lt** | **小于则为真** | **[ “$a” -lt “$b” ]** |
| **-le** | **小于等于则为真** | **[ “$a” -le “$b” ]** |

**例1：比较大小**

**[root@xuegod63 ~]# cat test1.sh**

**#!/bin/bash**

**if test 2 -eq 1 ; then**

**echo ok**

**else**

**echo err**

**fi**

**if [ 2 -eq 2 ] ; then**

**echo ok**

**else**

**echo err**

**fi**

**例2： 比较整数大小**

**[root@xuegod63 ~]# cat test2.sh**

**#!/bin/bash**

**read -p "input var1 var2:" var1 var2**

**if [ $var1 -gt $var2 ] ; then**

**echo "$var1 > $var2"**

**elif [ $var1 -lt $var2 ] ; then**

**echo "$var1 < $var2"**

**else**

**echo "$var1 = $var2"**

**fi**

**注意：在做数值比较时，只能用整数**

## 20.3.2 字符串比较

|  |  |  |
| --- | --- | --- |
| **参数.** | **说明** | **示例** |
| **==** | **等于则为真** | **[ “$a” == “$b” ]** |
| **!=** | **不相等则为真** | **[ “$a” != “$b” ]** |
| **-z 字符串.** | **字符串的长度为零则为真** | **[ -z “$a” ]** |
| **-n 字符串** | **字符串的长度不为空则为真** | **[ -n “$a” ]** |
| **str1 > str2** | **str1大于str2为真** | **[ str1 \> str2 ]** |
| **str1 < str2** | **str1小于str2为真** | **[ str1 \< str2 ]** |

**例1：根据用户名判断是否是超级管理员**

**[root@xuegod63 ~]# cat test3.sh**

**#!/bin/bash**

**read -p "input your name: " name**

**if [ $name == "root" ] ; then**

**echo "you are super administrator"**

**else**

**echo "You are a general user"**

**fi**

**[root@xuegod63 ~]# bash test3.sh**

**input your name: root**

**you are super administrator**

**[root@xuegod63 ~]# bash test3.sh**

**input your name: mk**

**You are a general usero "You are a general user"**

**例2：在做字符串大小比较的时候，注意字符串的顺序**

* **大于号和小于号必须转义，要不然SHELL会把它当成重定向符号**
* **大于和小于它们的顺序和sort排序是不一样的**
* **在test比较测试中，它使用的是ASCII顺序，大写字母是小于小写字母的；sort刚好相反**

**扩展： ASCII（American Standard Code for Information Interchange，美国信息交换标准代码）是基于拉丁字母的一套电脑编码系统，主要用于显示现代英语和其他西欧语言。它是现今最通用的单字节编码系统，并等同于国际标准ISO/IEC 646。**



**[root@xuegod63 ~]# cat test4.sh**

**#!/bin/bash**

**var1=test**

**var2=Test**

**if [ $var1 \> $var2 ] ; then**

**echo "$var1 > $var2"**

**else**

**echo "$var1 < $var2"**

**fi**

**[root@xuegod63 ~]# bash test4.sh**

**test > Test**

## 20.3.3 文件比较

|  |  |  |
| --- | --- | --- |
| **参数** | **.说明** | **示例** |
| **-e 文件名.** | **如果文件或目录存在则为真** | **[ -e file ]** |
| **-r 文件名.** | **如果文件存在且可读则为真** | **[ -r file ]** |
| **-w 文件名.** | **如果文件存在且可写则为真** | **[ -w file ]** |
| **-x 文件名.** | **如果文件存在且可执行则为真** | **[ -x file ]** |
| **-s 文件名** | **如果文件存在且至少有一个字符则为真** | **[ -s file ]** |
| **-d 文件名.** | **如果文件存在且为目录则为真** | **[ -d file ]** |
| **-f 文件名.** | **如果文件存在且为普通文件则为真** | **[ -f file ]** |
| **-c 文件名.** | **如果文件存在且为字符型文件则为真** | **[ -c file ]** |
| **-b 文件名..** | **如果文件存在且为块特殊文件则为真** | **[ -b file ]** |
| **file1 -nt fle2** | **检查file1是否比file2新** | **[ file1 -nt file2 ]** |
| **file1 -ot file2** | **检查file1是否比file2旧** | **[ file1 -ot file2 ]** |

**例1：**

**[root@xuegod63 ~]# vim test5.sh**

**#!/bin/bash**

**if [ -e /etc/passwd ] ; then**

**echo ok**

**else**

**echo err**

**fi**

**[root@xuegod63 ~]# bash test5.sh**

**ok**

**例2：**

**[root@xuegod63 ~]# test -e /etc/aaa.txt && echo ok || echo err**

**err**

**[root@xuegod63 ~]# test -e /etc/passwd && echo ok || echo err**

**ok**

**[root@xuegod63 ~]# test -e /etc && echo ok || echo err**

**ok**

**例：清空日志目录**

**[root@xuegod63 ~]# cat clear-log.sh**

**#!/bin/bash**

**# clear /var/log/messages**

**#确定当前是root用户**

**if [ $USER != "root" ];then**

**echo "你必须使用root用户才能执行脚本"**

**exit 10 #直接退出，并返回10**

**fi**

**#判断文件是否存在**

**if [ ! -f /var/log/messages ];then**

**echo "文件不存在"**

**exit 12**

**fi**

**#保留最近100行的日志内容**

**tail -100 /var/log/messages > /var/log/mesg.tmp**

**#日志清理**

**>/var/log/messages**

**#cat /dev/null > /var/log/messages**

**mv /var/log/mesg.tmp /var/log/messages**

**echo "Logs clean up"**

**注：退出码 exit ,取值范围是0-255**

**例： exit 退出bash，并返回一个值**

**[root@xuegod63 ~]# ssh 192.168.1.63**

**root@192.168.1.63's password: 123456**

**Last login: Mon May 28 20:37:41 2018 from xuegod63.cn**

**[root@xuegod63 ~]#**

**[root@xuegod63 ~]# exit 10**

**登出**

**Connection to 192.168.1.63 closed.**

**[root@xuegod63 ~]# echo $?**

**10**

# 20.4 流程控制过程中复杂条件和通配符

## 20.4.1 判断第一种：两个条件都为真或有一个为真就执行

**if [ 条件判断一 ] && (||) [ 条件判断二 ]; then**

**命令一**

**elif [ 条件判断三 ] && (||) [ 条件判断四 ]; then**

**命令二**

**else**

**执行其它**

**fi**

**判断第二种**

**if [条件判断一 -a (-o) 条件判断二 -a (-o) 条件判断三]; then**

**elif [条件判断三 -a (-o) 条件判断四 ]; then**

**else**

**执行其它**

**fi**

**判断第三种**

**if [[条件判断一 && (||) 条件判断二 ]]; then**

**elif [[ 条件判断三 && (||) 条件判断四 ]]; then**

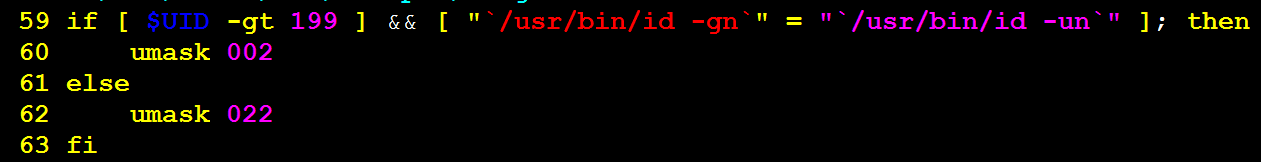
**else**

**执行其它**

**fi**

**例1：设置umask**

**参考： [root@xuegod63 ~]# vim /etc/profile**



**[root@xuegod63 ~]# vim umask.sh**

**if [ $UID -gt 199 ] && [ "`/usr/bin/id -gn`" = "`/usr/bin/id -un`" ]; then**

**echo "umask 002"**

**else**

**echo "i am root :umask 022"**

**fi**

**[root@xuegod63 ~]# bash umask.sh**

**i am root :umask 022**

**例2：[[ 。。。 ]]和[ 。。。]的区别**

**[[… ]] 运算符是[… ]运算符的扩充；[[… ]]能够支持 \*，< 、>等符号且不需要转义符**

**例1：**

**[root@xuegod63 ~]# if [[ $USER == r\* ]] ; then echo "hello,$USER" ; else echo $USER not ; fi**

**hello,root**

**注： $USER == r\*对比时， r\* 表示以r开头的任意长度字符串，这样就包括root**

**当只有一个[] 方括号时：**

**[root@xuegod63 ~]# if [ $USER == r\* ] ; then echo "hello,$USER" ; else echo $USER not ; fi**

**root not**

**#对比时r\* ，就表示两个字符串 r\***

**也可以这样写：**

**[root@xuegod63 ~]# if [[ $USER == [a-z]oot ]] ; then echo "hello,$USER" ; else echo $USER not ; fi**

**[[ 。。。 ]]和[ 。。。]的区别汇总：**

**1、所有的字符与逻辑运算符直接用“空格”分开，不能连到一起。**

**2、在[… ]表达式中，常见的> 、<需要加转义符\，大小比较**

**3、进行逻辑运算符&& 、||比较时；如果用的[ ]符号，则用在外面，如[… ] && [… ] || [ …]如果在[…]里面进行逻辑与或的比较，则用-a、-o进行表示，如[ x = y –a x < z –o x > m ]**

**4、[[… ]] 运算符只是[… ]运算符的扩充；能够支持< 、>符号运算不需要转义符；它还是以字符串比较大小。里面支持逻辑运算符 || 、 && ， 不再使用-a 、-o**

**5、[[…]] 用 && 而不是 -a 表示逻辑“与”；用 || 而不是 -o表示逻辑“或”**

**6、[[… ]]可以进行算术扩展，而[ ... ]不可以**

**7、[[...]]能用正则，而[...]不行**

**8、双括号(( ))用于数学表达式**

**9、双方括号号[[ ]]用于高级字符串处理，比如“模糊匹配”**

## 20.4.2 shell中的通配符

**shell常见通配符：**

|  |  |  |
| --- | --- | --- |
| **字符** | **含义** | **实例** |
| **\*** | **匹配 0 或多个字符** | **a\*b  a与b之间可以有任意长度的任意字符, 也可以一个也没有, 如aabcb, axyzb, a012b, ab。** |
| **?** | **匹配任意一个字符** | **a?b  a与b之间必须也只能有一个字符, 可以是任意字符, 如aab, abb, acb, a0b。** |
| **[list]** | **匹配 list 中的任意单一字符** | **a[xyz]b   a与b之间必须也只能有一个字符, 但只能是 x 或 y 或 z, 如: axb, ayb, azb。** |
| **[!list]** | **匹配 除list 中的任意单一字符** | **a[!0-9]b  a与b之间必须也只能有一个字符, 但不能是阿拉伯数字, 如axb, aab, a-b。** |
| **[c1-c2]** | **匹配 c1-c2 中的任意单一字符 如：[0-9] [a-z]** | **a[0-9]b  0与9之间必须也只能有一个字符 如a0b, a1b... a9b。** |
| **{string1,string2,...}** | **匹配 sring1 或 string2 (或更多)其一字符串** | **a{abc,xyz,123}b    a与b之间只能是abc或xyz或123这三个字符串之一。** |

**例：**

**[root@xuegod63 ~]# ls /etc/\*.conf**

**[root@xuegod63 ~]# ls /etc/???.conf**

**/etc/nfs.conf /etc/sos.conf /etc/yum.conf**

**[root@xuegod63 ~]# touch /opt/a{1,2,3}.txt**

**[root@xuegod63 ~]# ls /opt/a[123].txt**

**/opt/a1.txt /opt/a2.txt /opt/a3.txt**

**[root@xuegod63 ~]# ls /opt/a[1,2,3].txt**

**[root@xuegod63 ~]# ls /opt/a[13].txt**

**/opt/a1.txt /opt/a3.txt**

# 20.5 实战-3个shell脚本实战

## 20.5.1 实战1：编写脚本检查服务器运行状态

**[root@xuegod63 ~]# vim status.sh**

**#!/bin/bash**

**if [ $# -ge 1 ] ; then**

**systemctl status $1 > /dev/null**

**if [ $? -eq 0 ] ; then**

**echo "$1 服务正在运行"**

**else**

**systemctl start $1**

**fi**

**else**

**echo "执行脚本的格式"**

**echo "sh $0 服务名"**

**fi**

## 20.5.2 实战2:根据学生的成绩判断 学生的优劣

**[root@xuegod63 ~]# vim check\_cj.sh**

**#!/bin/bash**

**read -p "请输入你的成绩 " cj**

**if [ $cj -ge 0 ] && [ $cj -le 59 ] ;then**

**echo "补考"**

**elif [ $cj -ge 60 ] && [ $cj -le 70 ] ;then**

**echo "良好"**

**elif [ $cj -ge 71 ] && [ $cj -le 85 ] ;then**

**echo "好"**

**elif [ $cj -ge 86 ] && [ $cj -le 100 ] ;then**

**echo "优秀"**

**else**

**echo "成绩的有效范围是0-100之间"**

**fi**

## 20.5.3 实战3：每周一晚上3:00 ，备份数据库服务器上webdb库的所有数据到系统的/mysqlbak目录里，使用系统日期做备份文件名。

**[root@xuegod63 ~]# vim mysqlbak.sh**

**#!/bin/bash**

**baknamefile=`date +%Y-%m-%d`**

**bakdir=/mysqlbak**

**user=root**

**password=123**

**dbname=webdb**

**[ -e $bakdir ] || mkdir $bakdir**

**mysqldump -u$user -p$password --flush-logs $dbname > $bakdir/${baknamefile}-webdb.sql**

**因为mysql咱们还没有学，这里以/etc目录来做实验：**

**[root@xuegod63 ~]# vim etcbak.sh**

**#!/bin/bash**

**baknamefile=`date +%Y-%m-%d`**

**bakdir=/etcbak**

**srcdir=/etc**

**[ -e $bakdir ] || mkdir $bakdir**

**tar zcvf ${bakdir}/${baknamefile}-etc.tar.gz /etc/**

**echo "========================"**

**ls -lh ${bakdir}/${baknamefile}-etc.tar.gz**

**echo "back etc is ok!"**

**[root@xuegod63 ~]# chmod +x etcbak.sh**

**[root@xuegod63 ~]# crontab -e**

**0 3 \* \* \* /root/etcbak.sh 2>&1 > /dev/null**

**总结：**

**20.1 read命令键盘读取变量的值**

**20.2 流程控制语句if**

**20.3 test测试命令**

**20.4 流程控制过程中复杂条件和通配符**

**20.5 实战-3个shell脚本实战**