if语句

if在脚本开发中用的特别多，最频繁的语句，让超哥带你起飞吧！

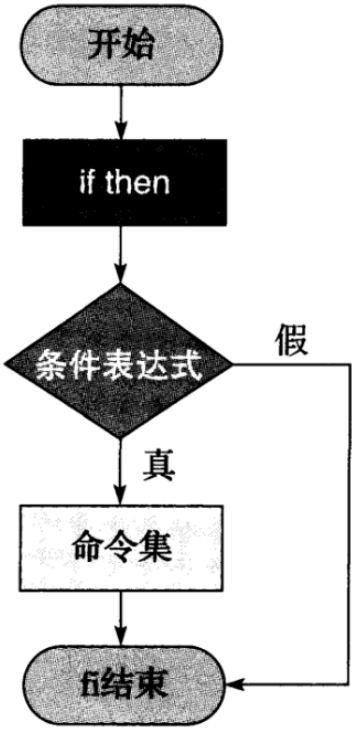
语法

```
if <条件表达式>
then
代码
fi
```

简写

```
if <条件表达式>;then
代码
fi
```

条件表达式，可以是超哥所教的 `[] test [[]] (())` 都可以。



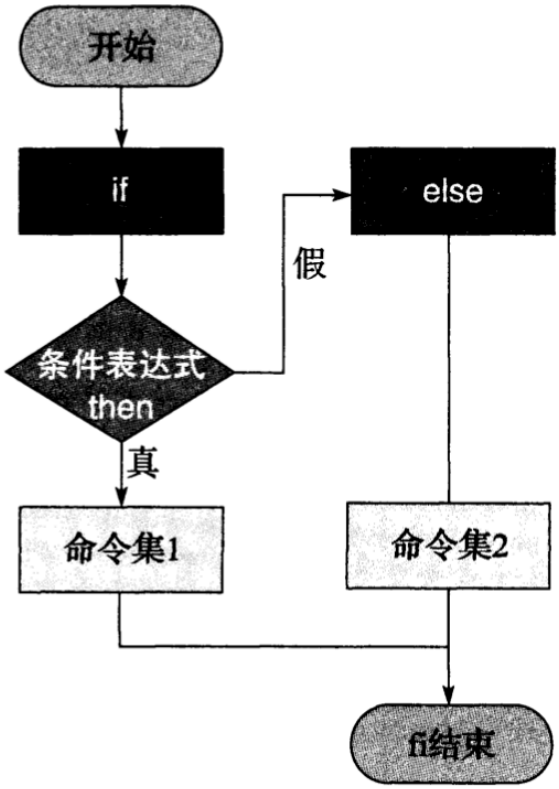
双分支

```
if  <条件表达式>
  then
    if  <条件表达式>
      then
        指令
    fi
fi
```

尽量用注释，看起来美观点

> 双分支嵌套结构
>
> 语法
>
> if 我有房
>
> 那么

```
if  <条件表达式>
  then
    代码1
else
    代码2
fi
```



多个分支

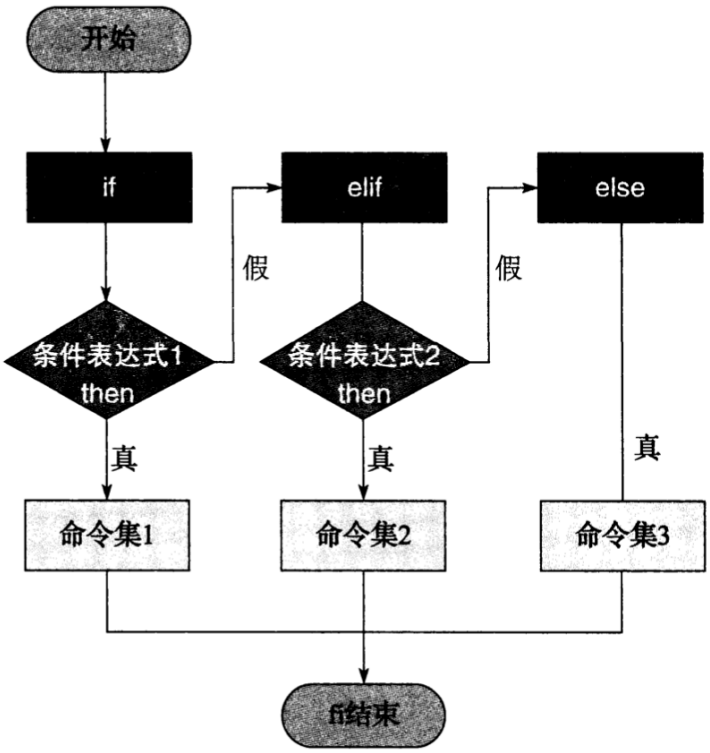
多个分支，就是当你需要多次逻辑判断，就会用到

你可以理解为，这个人有纠结选择困难症

```
if <条件表达式>
  then
    代码1
elif  <条件表达式2>
  then
    代码2
else
  代码3
fi
```

多个if,elif

```
if  <条件表达式>
  then
      代码1
elif  <条件表达式>
  then
    代码2
elif  <条件表达式>
  then
    代码3
else
  代码3
fi
```



单分支实践

> 1.将超哥之前教的条件测试语句，改造为if条件语句

```
[root@chaogelinux ~]# [ -f /etc/hosts ] && echo yes
yes
[root@chaogelinux ~]# [[ -f /etc/hosts ]] && echo yes
yes
[root@chaogelinux ~]# test -f /etc/hosts && echo yes
yes
```

改造脚本

```
[root@chaogelinux shell_program]# cat if_1.sh
#!/bin/bash

if [ -f /etc/hosts ]
```

```
    then
        echo "[ ] ok "
fi


if [[ -f /etc/hosts ]]
    then
       echo "[[ ]] ok"
fi


if test -f /etc/hosts
    then
        echo "test ok"
fi
```

执行

```
[root@chaogelinux shell_program]# bash if_1.sh
[ ] ok
[[ ]] ok
test ok
```

开发系统监控脚本

> 分析需求，开发shell脚本，检测内存剩余，可用内存小于100M的时候，发邮件报警给管理员，且加入crontab，每三分钟检查一次内存情况。

1. 获取当前内存
2. 配置邮件报警
3. 判断内存值是否小于100M，if判断
4. 开发shell脚本

5. 脚本加入crontab

开发过程

```
1.获取内存
total 系统总的可用物理内存大小
used 已被使用的物理内存大小
free 还有多少物理内存可用
shared 被共享使用的物理内存大小
buff/cache 被 buffer 和 cache 使用的物理内存大小
available 还可以被 应用程序 使用的物理内存大小

# 命令
[root@chaogelinux shell_program]# free -m
              total        used        free      shared  buff/cache   available
Mem:           1838        1315          78          16         444         328
Swap:             0           0           0

注意，不同电脑看到的结果不一样
超哥这里的电脑
# awk NR==行号，$NF最后一个字段

# 获取可用内存
[root@chaogelinux shell_program]# free -m| awk 'NR==2 {print $NF}'
328

# 脚本开发
# 这里需要提前配置好mail发邮件的设置，超哥就不演示了
[root@chaogelinux shell_program]# cat free_1.sh
#!/bin/bash
```

```
FreeMem=`free -m|awk 'NR==2 {print $NF}'`
CHARS="Current memory is $FReeMem."

if [ "$FreeMem" -lt 100 ]
    then
        echo $CHARS|tee /tmp/messages.txt
        mail -s "`date +%F-%T`$CHARS" yc_uuu@163.com < /tmp/messages.txt
fi


# 脚本加入定时任务
[root@chaogelinux shell_program]# crontab -l
*/3 * * * * /bin/bash /shell_program/free_1.sh &>/dev/null

```

读数比较大小

> 单分支脚本

```
[root@chaogelinux shell_program]# cat if_read.sh
#!/bin/bash
a=$1
b=$2
if [ $a -lt $b ];then
    echo "yes,$a less than $b"
    exit 0
fi

if [ $a -eq $b ];then
    echo "yes,$a equal $b"
```

```
    exit 0
fi

if [ $a -gt $b ];then
    echo "yes,$a grather than $b"
    exit 0
fi

```

执行

```
[root@chaogelinux shell_program]# bash if_read.sh 4 3
yes,4 grather than 3
[root@chaogelinux shell_program]# bash if_read.sh 1 3
yes,1 less than 3
```

> 多分支脚本

```
[root@chaogelinux shell_program]# cat if_read2.sh
#!/bin/bash

a=$1
b=$2
if [ $a -lt $b ];then
    echo "yes, $a less than $b"
elif [ $a -eq $b ];then
    echo "yes, $a equal $b"
else [ $a -gt $b ]
    echo "yes,$a greater than $b"
fi
```

执行

```
[root@chaogelinux shell_program]# bash if_read2.sh 3 4
yes, 3 less than 4
[root@chaogelinux shell_program]# bash if_read2.sh 3 3
yes, 3 equal 3
[root@chaogelinux shell_program]# bash if_read2.sh 3 1
yes,3 greater than 1
```

if实战开发

开发nginx以及mysql监控脚本

> 监控服务的理念

端口监控	1）在服务器本地监控服务端口的常见命令有 netstat、ss、lsof 2）从远端监控服务器本地端口的命令有 telnet、nmap、nc
监控服务进程或进程数	此方法适合本地服务器，注意，过滤的是进程的名字。命令为： ps -ef|grep nginx|wc -l ps -ef|grep mysql|wc -l
在客户端模拟用户访问	使用 wget 或 curl 命令进行测试（如果监测数据库，则需要转为通过 Web 服务器去访问数据库），并对测试结果做三种判断： 1）利用返回值（echo $?）进行判断 2）获取特殊字符串以进行判断（需要事先开发好程序） 3）根据 HTTP 响应 header 的情况进行判断
登录 MySQL 数据库判断	通过 MySQL 客户端连接数据库，根据返回值或返回内容判断。例如： mysql -uroot -poldboy123 -e "select version();" &>/dev/null; echo $?

监控mysql的思路

- 端口netstat，ss，lsof监控

```
[root@chaogelinux ~]# netstat -tunlp|grep 3380 |wc -l
1
```

```
3
4  # 通过名字找更为合适
5  [root@chaogelinux ~]# netstat -tunlp|grep mysql |wc -l
6  1
7
8  # ss
9  [root@chaogelinux ~]# ss -tunlp|grep mysql | wc -l
10 1
11
12 # lsof命令
13 [root@chaogelinux ~]# lsof -i tcp:3380
14 COMMAND   PID  USER   FD   TYPE  DEVICE SIZE/OFF NODE NAME
15 mysqld  20327 mysql   61u  IPv6 3356432      0t0  TCP *:sns-
   channels (LISTEN)
16
17
```

> 远程监测**mysql**的端口方法

```
1  [root@chaogelinux ~]# yum install telnet nmap nc -y
2  # nmap
3  [root@chaogelinux ~]# nmap 127.0.0.1 -p 3380 |grep open |wc -l
4  1
5
6  #telnet
7  [root@chaogelinux ~]# echo -e "\n" |telnet 127.0.0.1 3380
   2>/dev/null|grep Connected |wc -l
8  1
9
```

> 检查进程

```
[root@chaogelinux ~]# ps -ef|grep mysql|grep -v grep |wc -l
1
```

开发代码连接数据库

> 通过访问应用程序接口，读取数据库，查看是否正确读取

> 通过编写php，python代码，尝试连接数据库

前提，搞好linux的数据库依赖环境

php

```
yum remove php-mysql
yum install php-mysqlnd php



[root@chaogelinux shell_program]# cat mysql_test.php
<?php
$link_id=mysql_connect('localhost','root','chaoge888') or
mysql_error();
if ($link_id){
    echo "mysql successful,chaoge 666~~!";
}else{
    echo mysql_error();
}
?>

# 运行

```

```
[root@chaogelinux shell_program]# php mysql_test.php
mysql successful,chaoge 666~~![root@chaogelinux
shell_program]#
```

python

```
1.安装依赖
[root@chaogelinux shell_program]# yum install python3 python3-
devel python3-pip

[root@chaogelinux shell_program]# pip3 install PyMySQL



2.开发代码，注意python代码，缩进关系要明确
[root@chaogelinux shell_program]# cat test_python_mysql.py
import pymysql

db = pymysql.connect(host='localhost',
                     port=3380,
                     user='root',
                     password='chaoge888',
                     db='mysql',
                     charset='utf8')

cursor=db.cursor()
cursor.execute("Select version()")

data=cursor.fetchone()

```

```
print("数据库连接正确，数据库版本是：%s"%data)
db.close()

```

Shell监控mysql脚本开发

这里小于老师给出多种答案，提供大家参考学习

```
#!/bin/bash
echo "-----方法1----"
if [ `netstat -tunlp|grep mysql |wc -l` = "1" ]
  then
    echo "MySQL is running。"
else
  echo "MySQL is Stopped。"
  #systemctl start mariadb
fi


echo "-----方法2----"
if [  `ss -tunlp|grep mysql | wc -l` -eq "1" ]
  then
    echo "MySQL is running。"
else
  echo "MySQL is Stopped。"
  #systemctl start mariadb
fi


echo "-----方法3----"
if [  `lsof -i tcp:3380|wc -l` -gt 0 ]
  then
    echo "MySQL is running。"
```

```
else
  echo "MySQL is Stopped。"
  #systemctl start mariadb
fi

echo "-----方法4----"
python3 /shell_program/test_python_mysql.py

if [ "$?"  -eq 0 ]
  then
    echo "MySQL is running。"
else
  echo "MySQL is Stopped。"
  #systemctl start mariadb
fi

echo "-----方法5----"
php /shell_program/mysql_test.php

if [ "$?"  -eq 0 ]
  then
    echo "MySQL is running。"
else
  echo "MySQL is Stopped。"
  #systemctl start mariadb
fi
```

rsync启动脚本开发

> 1.rsync基础环境监察

```
[root@chaogelinux shell_program]# rpm -qa rsync
```

```
rsync-3.1.2-6.el7_6.1.x86_64

[root@chaogelinux shell_program]# ls -l  /etc/rsyncd.conf
-rw-r--r-- 1 root root 458 4月  26 2019 /etc/rsyncd.conf

# 启动服务
[root@chaogelinux shell_program]# /usr/bin/rsync --daemon

# 检查服务
[root@chaogelinux shell_program]# netstat -tunlp|grep 873
tcp        0      0 0.0.0.0:873             0.0.0.0:*
      LISTEN      1809/rsync
tcp6       0      0 :::873                  :::*
     LISTEN      1809/rsync

# 停止rsync
[root@chaogelinux shell_program]# pkill rsync
[root@chaogelinux shell_program]# netstat -tunlp|grep 873


```

> 完整rsync脚本，可以模仿/etc/init.d/network开发

```
#!/bin/bash
# author: yuchao
# -ne 不等于  $# 传递给脚本或函数的参数个数。
# $0 脚本名
if [ $# -ne 1 ]
  then
    echo "Usage: $0 {start|stop|restart}"
    exit 1
fi

```

```
if [ "$1" = "start" ]
  then
    /usr/bin/rsync --daemon
    sleep 2
    if [ `netstat -tunlp|grep rsync|wc -l` -ge 1 ]
      then
        echo "Rsync is started."
        exit 0
    fi
elif [  "$1" = "stop" ]
  then
    killall rsync &>/dev/null
    sleep 2
    if [ `netstat -tunlp|grep rsync|wc -l` -eq 0 ]
      then
        echo "Rsync is stopped."
        exit 0
    fi
elif  [ "$1" = "restart" ]
  then
    killall rsync
    sleep 1
    killpro=`netstat -tunlp|grep rsync|wc -l`
    /usr/bin/rsync --daemon
    sleep 1
    startpro=`netstat -tunlp|grep rsync|wc -l`
    if [ $killpro -eq 0 -a $startpro -ge 1 ]
      then
        echo "rsyncd is restarted."
        exit 0
    fi
else
  echo "Usage: $0 {start|stop|restart}"
    exit 1
```

```
45 fi
46
47
48 # 添加开机自启脚本
49 chkconfig --list rsyncd
50
```

作业练习

> 同理，大家也可以自行扩展，例如**nginx**，**redis**服务的状态监控