# bat脚本

--莫非

## BAT脚本与shell脚本区别

Shell脚本使用Linux/Unix下的命令，一般文件开头以#号来告诉系统这个脚本需要什么解释器来执行（如：#!/bin/bash采用bash而不是dash来执行）。而bat文件是dos下的批处理文件。批处理文件是无格式的文本文件，它包含一条或多条命令，它的文件扩展名为.bat 或.cmd。

## 常用命令1：where、cls、title、color、mode con

* 1. Where

显示符合搜索模式的文件位置： Where \R C:\ \*.dll \F \T

* 1. Cls：清屏
  2. Title

设置控制台BAT的标题

* 1. Color

设置bat显示的背景和前景色：color 02 （背景黑色，字符绿色）

* 1. MODE CON

设置bat的设置窗体大小：MODE CON COLS=200 LINES=1000

## 常用命令2：@、echo、::、rem、pause、:和goto、%以及if

* 1. @

@的作用就是让脚本在执行时不显示命令部分：@echo off

* 1. echo

echo命令的另一种用法一：可以用它来显示信息：echo Edit by CODERED

echo命令的另一种用法二：可以直接编辑文本文件：echo nbtstat -A 192.168.0.1 > a.bat

* 1. ::、rem

::和rem是注释命令: ::close echo

与rem不同的是，::后的字符行在执行时不会回显, 无论是否用echo on打开命令行回显状态，另外，rem 可以用于config.sys文件中.

* 1. pause

pause是暂停命令：pause

* 1. :和goto

goto是个跳转命令，: 是一个标签,当程序运行到goto时，将自动跳转到:定义的部分去执行了：

……

goto end

……

:end

* 1. %

%是批处理变量引导符，为命令行传递给批处理的参数（多个%一起使用的情况除外，以后还将详细介绍）:

set p=”abcdefg” & echo %p:~3,2%

* 1. If

输入判断: if "%1"=="" goto usage

存在判断:

if exist C:\Progra~1\Tencent\AD\\*.gif del C:\Progra~1\Tencent\AD\\*.gif

结果判断: if errorlevel 1 pause & edit %1.asm

## 常用命令3：call、find、for、set、shift

* 1. Call

call命令用来从一个批处理脚本中调用另一个批处理脚本：

echo %1 This is a call example.

call a.bat

* 1. find

find用来在文件中搜索特定字符串，通常也作为条件判断的铺垫程序：

type a.txt | find "7626" && echo "Congratulations! You have infected GLACIER!"

* 1. shift

改变传入的参数偏移量，如将%3移位到%2，将%4移位到%3，等等；并且不影响%0和%1：

SHIFT /2

* 1. Set

显示、设置或删除cmd.exe环境变量。

1. Set path=”D:\”
2. Set /p input=请输入： & echo 你输入的是：%input%
3. setlocal enabledelayedexpansion

Set /p input=请输入： & echo 你输入的是：!input!

* 1. For

基本格式：

FOR %%variable IN (set) DO command [command-parameters]

/D参数（匹配目录）：

for /d %%i in (c:\\*) do echo %%i

/R参数（匹配文件）

for /r [c:\] %%i in (???.exe) do if exist %%i echo %%i

/L参数（该集表示以增量形式从开始到结束的一个数字序列）

for /l %%i in (1,1,5) do @echo %%i

/F参数

使用文件解析来处理命令输出、字符串及文件内容。使用迭代变量定义要检查的内容或字符串，并使用各种options选项进一步修改解析方式。默认情况下，/F 传递每个文件每一行的第一个空白分隔符号。跳过空行。

for /f "[eol=; skip=1] delims=" %%i in ('net user') do @echo %%i

## 管道命令 |、>、>>、>、>&、<&

* 1. |

|命令的作用，就是让前一命令的输出当做后一命令的输入：

help | more，ipconfig /all | find “IPv4”

* 1. >

>会清除掉原有文件中的内容后把新的内容写入原文件: echo @echo off > a.bat

* 1. >>

>>只会另起一行追加新的内容到原文件中，而不会改动其中的原有内容：

echo exit >> a.bat

* 1. <

<是输入重定向命令，从文件中读入命令输入，而不是从键盘中读入

* 1. >&

>&将一个句柄的输出写入到另一个句柄的输入中

* 1. <&

<&刚好和>&相反，从一个句柄读取输入并将其写入到另一个句柄输出中

## 组合命令：&、&&、||

组合命令，顾名思义，就是可以把多个命令组合起来当一个命令来执行，那么这多个命令就要用这些组合命令连接起来放在同一行，因为批处理认行不认命令数目

* 1. &

&命令作用是用来连接n个DOS命令，并把这些命令按顺序执行，而不管是否有命令执行失败：

copy a.txt b.txt /y & del a.txt

* 1. &&

&&可以把它前后两个命令组合起来当一个命令来用，一旦发现有命令出错将不继续执行剩下的命令：

dir 文件://1%/www/user.mdb && copy 文件://1%/www/user.mdb e:\backup\www

if exist 文件://1%/www/user.mdb copy 文件://1%/www/user.mdb e:\backup\www

dir c:\ && dir d:\ > a.txt；

得不到预期结果必须：

dir c:\ > a.txt && dir d:\ >> a.txt

* 1. ||

||命令执行正确就退出此命令组合，不再执行剩下的命令：

dir s\*.exe || echo Didn't exist file s\*.exe & pause & exit

## 其他符号：，、；、””、()、！

* 1. ;

分号，当命令相同时，可以将不同目标用；来隔离，但执行效果不变，如执行过程中发生错误，则只返回错误报告，但程序仍会执行。

dir c:\;d:\;e:\;z:\ > 1.txt

* 1. ，

逗号相当于空格，在某些情况下“,”可以用来当做空格使用：

dir,C:\

* 1. “”

双引号允许在字符串中包含空格:

cd,"C:\ program files"

* 1. ()

小括号在批处理编程中有特殊的作用，左右括号必须成对使用，括号中可以包括多行命令，这些命令将被看成一个整体，视为一条命令行。

(echo a && echo bb) > 1.txt

* 1. !

在变量延迟问题中，用来表示变量，即%var%应该表示为!var!

Set /p input=请输入： & echo 你输入的是：!input!

## if…else…条件语句

DOS条件语句主要有以下形式

IF [NOT] ERRORLEVEL number command

IF [NOT] string1==string2 command

IF [NOT] EXIST filename command

增强用法：IF [/I] string1 compare-op string2 command

增强用法中加上/I就不区分大小写了!

增强用法中还有一些用来判断数字的符号：

EQU - 等于

NEQ - 不等于

LSS - 小于

LEQ - 小于或等于

GTR - 大于

GEQ - 大于或等于

上面的command命令都可以用小括号来使用多条命令的组合，包括else子句，组合命令中可以嵌套使用条件或循环命令。

  IF EXIST filename (

        del filename

    ) ELSE (

        echo filename missing

    )

## 循环语句

* 1. 指定次数循环

FOR /L %variable IN (start,step,end) DO command [command-parameters]

组合命令：

FOR /L %variable IN (start,step,end) DO (

Command1

Command2

……

)

* 1. 对某集合执行循环语句。

FOR %%variable IN (set) DO command [command-parameters]

  %%variable  指定一个单一字母可替换的参数。

  (set)      指定一个或一组文件。可以使用通配符。

  command   对每个文件执行的命令，可用小括号使用多条命令组合。

FOR /R [[drive:]path] %variable IN (set) DO command [command-parameters]

同前面一样，command可以用括号来组合：

FOR /R [[drive:]path] %variable IN (set) DO (

Command1

Command2

……

commandn

)

* 1. 条件循环

上面的循环结构是用for命令来实现的，for命令循环有一个缺点，就是整个循环被当作一条命令语句，涉及到变量延迟的问题。

利用goto语句和if条件判断，dos可以实现条件循环：

@echo off

set var=0

rem \*\*\*\*\*\*\*\*\*\*\*\*循环开始了

:continue

set /a var+=1

echo 第%var%次循环

if %var% lss 100 goto continue

rem \*\*\*\*\*\*\*\*\*\*\*\*循环结束了

echo 循环执行完毕

pause

## 子程序

在批处理程序中可以调用外部可运行程序，比如exe程序，也可调用其他批处理程序，批处理可以调用本程序中的一个程序段，相当于子程序，这些子程序一般放在主程序后面。

* 1. 子程序调用格式：

  CALL :label arguments

* 1. 子程序语法：

  :label

  command1

  command2

  ......

  commandn

  goto :eof

在子程序段中，参数%0指标签:label

子过程一般放在最后，并且注意在主程序最后要加上exit或跳转语句，避免错误的进入子过程。

子程序和主程序中的变量都是全局变量，其作用范围都是整个批处理程序。

传至子程序的参数在call语句中指定，在子程序中用%1、%2至%9的形式调用，而子程序返回主程序的数据只需在调用结束后直接引用就可以了，当然也可以指定返回变量。

* 1. 打印你好：

@echo off

call :sub return

echo 子程序返回值：%return%

pause

goto :eof

:sub

set %1=你好

goto :eof

* 1. 设计一个求多个整数相加的子程序：

@echo off

set sum=0

call :sub 10 20 35 40 50

echo 数据求和结果：%sum%

pause

goto :eof

:sub

rem 参数1为返回变量名称

set /a sum+=%1

shift /1

if not "%1"=="" goto sub

goto :eof

运行结果：155

## 变量嵌套与命令嵌套

和其它编程语言相比，dos功能显得相对简单，要实现比较复杂的功能，需要充分运用各种技巧，变量嵌套与命令嵌套就是此类技巧之一。

简单的说，命令嵌套首先用一条dos命令生成一个字符串，而这个字符串是另一条dos命令，用call语句调用字符串将其执行，从而得到最终结果。

用call语句实现命令嵌套：

@echo off

set str1=aaa echo ok bbb

echo 初始字符串：%str1%

echo 生成命令字符串如下：

echo %str1:~4,7%

echo 运行命令字符串生成最终结果为：

call %str1:~4,7%

pause