参考：<http://www.cnblogs.com/advocate/archive/2010/11/23/1885279.html>

[**Bat语法**](http://www.cnblogs.com/advocate/archive/2010/11/23/1885279.html)

1. REM   
   REM是个注释命令一般是用来给程序加上注解的，该命令后的内容在程序执行的时候将不会被显示和执行。   
   例：   
   REM 你现在看到的就是注解，这一句将不会被执行。
2. ECHO   
   ECHO是一个回显命令主要参数有OFF和 ON,一般用ECHO message来显示一个特定的消息（最好不要用中文，可能是某些命令不能被识别）。例：   
     
   Echo OFF   
   Rem 以上代表关闭回显即不显示所执行的命令(即之后的所有命令(直到碰到ECHO ON)的执行都不会显示执行的命令，如果ECHO ON则相当于在控制台每次在该路径下执行命令因而能够看到命令)

如：echo.bat

ECHO ON

ECHO 123

ECHO 234

ECHO OFF

ECHO 123

ECHO 234

执行输出：

E:\deleting\bat\_study\codes>ECHO 123

123

E:\deleting\bat\_study\codes>ECHO 234

234

E:\deleting\bat\_study\codes>ECHO OFF

123

234

Echo 这个就是消息。   
Rem 以上代表显示“这就是消息”这列字符   
  
执行结果：   
C:\>ECHO.BAT   
这个就是消息。

1. GOTO
2. CALL   
   CALL命令可以在批处理执行过程中调用另一个批处理，当另一个批处理执行完后再继续执行原来的批处理。

例：   
批处理2.BAT内容如下：   
ECHO 这就是2的内容   
  
批处理1.BAT内容如下：   
ECHO 这是1的内容   
CALL 2.BAT   
ECHO 1和2的内容全部显示完成   
  
执行结果如下：   
C:\>1.BAT   
这是1的内容   
这就是2的内容   
1和2的内容全部显示完成   
  
5、PAUSE   
PAUSE停止系统命令的执行并显示下面的内容。例：   
C:\> PAUSE   
请按任意键继续 . . .   
  
6、 IF   
IF条件判断语句，语法格式如下：

IF [NOT] ERRORLEVEL number command   
IF [NOT] string1==string2 command   
IF [NOT] EXIST filename command   
  
说明：   
[NOT] 将返回的结果取反值即“如果没有”的意思。   
ERRORLEVEL 是命令执行完成后返回的退出值   
Number 退出值的数字取值范围0~255。判断时值的排列顺序应该由大到小。   
返回的值大于或等于指定的值时条件成立。   
  
string1==string2   
string1和string2都为字符的数据，英文字符的大小写将看做不同，这个条件中的等于号必须是2个（绝对相等），条件想等后即执行后面的 command   
  
EXIST filename 为文件或目录存在的意思。   
IF ERRORLEVEL 这条语句必须放在某一个命令后面。执行命令后由IF ERRORLEVEL来判断命令的返回值（类似linux的执行命令后也有返回值）。   
  
例：   
1、 IF [NOT] ERRORLEVEL number command   
检测命令执行完后的返回值做出判断。   
echo off   
dir z:   
rem 如果退出代码为1（不成功）就跳至标题1处执行   
IF ERRORLEVEL 1 goto 1   
rem 如果退出代码为0（成功）就跳至标题0处执行   
IF ERRORLEVEL 0 goto 0   
:0   
echo 命令执行成功！   
Rem 程序执行完毕跳至标题exit处退出   
goto exit   
:1   
echo 命令执行失败！   
Rem 程序执行完毕跳至标题exit处退出   
goto exit   
:exit   
Rem 这里是程序的出口 

ECHO OFF

ECHO begin to run

ECHO Demo for IF

dir n:

REM 如果退出代码为1则跳至标题1出执行

IF ERRORLEVEL 1 GOTO 1

REM 如果退出代码为0则跳至标题0出执行

IF ERRORLEVEL 0 GOTO 0

:0

ECHO success

GOTO exit

:1

ECHO failed

GOTO exit

:exit

REM exit

ECHO exit

2、 IF string1==string2 command   
检测当前变量的值做出判断   
ECHO OFF   
IF not %1==2 goto no   
Echo 变量相等！   
Goto exit   
:no   
echo 变量不相等   
goto exit   
:exit   
大家可以这样看效果 C:\>test.bat 3   
这里%1是命令行参数，%1＝3 

ECHO OFF

ECHO run

ECHO demo2 for IF

IF %1 == a GOTO yes

IF NOT %1 == a GOTO no

:yes

ECHO %1 equal to a

GOTO exit

:no

ECHO %1 not equal to a

GOTO exit

:exit

REM 这是程序出口

ECHO exit

3、 IF [NOT] EXIST filename command   
发现特定的文件做出判断   
  
echo off   
IF not EXIST autoexec.bat goto 1   
echo 文件存在成功！   
goto exit   
:1   
echo 文件不存在失败！   
goto exit   
:exit 

ECHO OFF

ECHO run

ECHO demo3 for if

IF EXIST 1.bat GOTO yes

:yes

ECHO 1.bat is exists

GOTO exit

ECHO 1.bat is not exists

:exit

REM this is the exit

ECHO exit

GOTO 即为跳转的意思。在批处理中允许以“：XXX”来构建一个标号,然后用GOTO :标号，直接执行标号后的命令。例：   
  
:LABEL   
REM 上面就是名为LABEL的标号。   
DIR C:\   
DIR D:\   
GOTO LABEL   
REM 以上程序跳转标号LABEL处继续执行。   
  
7、 FOR   
FOR这个命令比较特殊是一个循环执行命令的命令，同时FOR的循环里面还可以套用FOR在进行循环。在批处理中FOR的命令如下：   
  
FOR [%%c] IN (set) DO [command] [arguments]   
  
在命令行中命令如下：   
FOR [%c] IN (set) DO [command] [arguments]   
  
例：建立C:\TEST7.BAT，文件内容如下：   
  
@ECHO OFF   
FOR %%C IN (\*.BAT \*.TXT \*.SYS) DO TYPE %%C   
  
运行：   
C:\>TEST7   
  
  
执行以后，屏幕上会将C:盘根目录下所有以BAT、TXT、SYS为扩展名的文件内容显示出来（不包括隐藏文件）。   
  
常用参数：   
/L 该集表示以增量形式从开始到结束的一个数字序列。因此，(1,1,5) (开始步长结束)将产生序列 1 2 3 4 5，(5,-1,1) 将产生序列 (5 4 3 2 1)。   
/D 如果集中包含通配符，则指定与目录名匹配，而不与文件名匹配。   
  
例：   
ECHO OFF   
FOR /L %%F IN (1,1,5) DO DEL %%F.TXT   
或   
FOR %%F IN (1,2,3,4,5) DO DEL %%F.TXT   
以上2条命令执行的结果都是一样的如下：   
C:\>DEL 1.TXT   
C:\>DEL 2.TXT   
C:\>DEL 3.TXT   
C:\>DEL 4.TXT   
C:\>DEL 5.TXT 

ECHO OFF

FOR %%n IN (1,2,3,4,5) DO ECHO %%n

FOR /L %%n IN (0,2,20) DO ECHO %%n

8、 SETLOCAL   
开始批处理文件中环境改动的本地化操作。在执行 SETLOCAL 之后   
所做的环境改动只限于批处理文件。要还原原先的设置，必须执   
行 ENDLOCAL。达到批处理文件结尾时，对于该批处理文件的每个   
尚未执行的 SETLOCAL 命令，都会有一个隐含的 ENDLOCAL 被执行。   
  
例：   
@ECHO OFF   
SET PATH 　　/\*察看环境变量PATH   
PAUSE   
SETLOCAL   
SET PATH=E:\TOOLS /\*重新设置环境变量PATH   
SET PATH   
PAUSE   
ENDLOCAL   
SET PATH   
PAUSE   
  
从上例我们可以看到环境变量PATH第1次被显示得时候是系统默认路径。被设置成了E:\TOOLS后显示为E:\TOOLS但当ENDLOCAL后我们可以看到他又被还原成了系统的默认路径。但这个设置只在该批处理运行的时候有作用。当批处理运行完成后环境变量PATH将会还原。 

ECHO OFF

ECHO run

ECHO demo for SETLOCAL

REM 查看环境变量

SET PATH

PAUSE

SETLOCAL

REM 重新设置环境变量为当前目录

SET PATH=D:\software

SET PATH

PAUSE

ENDLOCAL /\*还原原始设置

SET PATH

PAUSE

9、 SHIFT   
SHIFT命令可以让在命令行上的命令使用超过10个（%0~%9）以上的可替代参数,第一个参数%0为当前执行脚本名

例：   
ECHO OFF   
ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9   
SHIFT   
ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9   
SHIFT   
ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9   
  
执行结果如下：   
C::\>SHIFT.BAT 1 2 3 4 5 6 7 8 9 10 11   
1 2 3 4 5 6 7 8 9   
2 3 4 5 6 7 8 9 10   
3 4 5 6 7 8 9 10 11   
以上就是基于WIN2000下的9个批处理命令。 

ECHO OFF

ECHO %0

ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9

SHIFT

ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9

SHIFT

ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9

SHIFT

ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9

SHIFT

ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9

SHIFT

ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9

SHIFT

ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9

SHIFT

ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9

SHIFT

ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9

SHIFT

ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9

SHIFT

特殊的符号与批处理   
  
在命令行下有些符号是不允许使用的但有些符号却有着特殊的意义。   
  
1、符号(@)   
@在批处理中的意思是关闭当前行的回显。我们从上面知道用命令echo off可以关掉整个批处理的命令回显但却不能不显示echo off这个命令。现在我们在这个命令前加上@这样echo off这一命令就被@关闭了回显从而达到所有命令均不回显得要求   
  
2、符号(>)   
>的意思是传递并覆盖。他所起的作用是将运行后的回显结果传递到后面的范围（后面可是文件也可是默认的系统控制台）例：   
文件1.txt的文件内容为：   
1+1   
  
使用命令c:\>dir \*.txt >1.txt   
  
这时候1.txt的内容如下:   
  
驱动器 C 中的卷没有标签。   
卷的序列号是 301A-1508   
C:\ 的目录   
2003-03-11 14:04 1,005 FRUNLOG.TXT   
2003-04-04 16:38 18,598,494 log.txt   
2003-04-04 17:02 5 1.txt   
2003-03-12 11:43 0 aierrorlog.txt   
2003-03-30 00:35 30,571 202.108.txt   
5 个文件 18,630,070 字节   
0 个目录 1,191,542,784 可用字节   
  
>将命令执行的结果覆盖了原始的文件内容。   
在传递给控制台的时候程序将不会有任何回显（注意：这里的回显跟echo off关掉的回显不是同一概念。Echo off关掉的是输入命令的回显，   
这里的回显是程序执行中或后的回显）例：   
C:\>dir \*.txt >nul   
程序将没有任何显示也不会产生任何痕迹。   
  
3、符号(>>)   
符号>>的作用与符号>相似，但他们的区别在于>>是传递并在文件末尾追加>>也可将回显传递给控制台（用法同上）例：   
文件1.txt内同为：   
1+1   
使用命令c:\>dir \*.txt >>1.txt   
这时候1.txt的内容如下:   
  
1+1   
驱动器 C 中的卷没有标签。   
卷的序列号是 301A-1508   
C:\ 的目录   
2003-03-11 14:04 1,005 FRUNLOG.TXT   
2003-04-04 16:38 18,598,494 log.txt   
2003-04-04 17:02 5 1.txt   
2003-03-12 11:43 0 aierrorlog.txt   
2003-03-30 00:35 30,571 202.108.txt   
5 个文件 18,630,070 字节   
0 个目录 1,191,542,784 可用字节   
  
>>将命令执行的结果覆加在了原始的文件内容后面。   
  
4、符号(|)   
|是一个管道传输命令意思是将上一命令执行的结果传递给下一命令去处理。   
例：   
  
C:\>dir c:\|find "12"   
2007-03-12 11:33 <DIR> WINDOWS   
2007-03-12 11:58 <DIR> Documents and Settings   
2007-03-12 12:16 <DIR> Program Files   
2007-03-12 12:20 0 CONFIG.SYS   
2007-03-12 12:20 0 AUTOEXEC.BAT   
2007-03-12 14:25 <DIR> chenhu2   
2007-03-12 20:19 <DIR>mysql   
2007-03-12 22:32 <DIR> java   
2007-03-12 22:37 <DIR> jdk1.5   
  
C:\>   
  
以上命令的意思为:   
在dir c:\的结果中查找包含“12”的字符串。   
  
Find的用法请用 find /?自行查看!   
  
在不使用format的自动格式化参数的时候我是这样来自动格式化盘片的   
echo y|fornat a: /s /q /v:system   
  
用过format命令的人都知道format有一个交互对话过程，要使用者输入y来确定当前的命令是否被执行。在这个命令前加上echo y并用管道传输符|将echo执行的结果y传递给format从而达到手工输入y的目的（这条命令有危害性，测试的时候请谨慎）   
  
5、符号(^)   
^ 是对特殊符号>、<、&、的前导字符。在命令中他将以上的3个符号的特殊动能去掉仅仅只吧他们当成符号而不使用他们的特殊意义。例：   
  
c:\>echo test ^> 1.txt   
test > 1.txt   
从上面可以看出并没有把test写入文件1.txt而是将test >1.txt 当字符串显示了出来。这个符号在远程构建批处理的时候很有效果（即通过批处理构建批处理）。   
  
6、符号(&)   
&符号允许在一行中使用2个以上不同的命令，当第一个命令执行失败将不影响第2个命令的执行。   
  
例：   
c:\>dir z:\ &dir y:\ &dir c:\   
以上的命令将会连续显示z: y: c:盘内的内容不理会该盘符是否存在。   
  
7、符号(&&)   
&&符号也是允许在一行中使用2个以上不同的命令，当第一个命令执行失败后后续的命令将不会再被执行。   
  
例：   
c:\>dir z:\ &&dir y:\ &&dir c:\   
  
以上的命令将会提示检查是否存在z:盘如果存在则执行，如果不存在则停止执行所有的后续命令   
  
8、符号(" ")   
" "符号允许在字符串中包含空格。进入一个特殊的目录可以用如下方法例:   
c:\>cd “Program Files”   
c:\>cd progra~1   
c:\>cd pro\*   
以上方法都可以进入Program Files目录 （不加括号的话也能进入）  
  
9、符号（,）   
,符号相当于空格。在某些特殊的情况下可以用,来代替空格使用。例：   
c:\>dir,c:\   
  
10、符号(;)   
;符号当命令相同的时候可以将不同的目标用;隔离开来但执行效果不变。如执行过程中发生错误则只返回错误报告但程序还是会继续执行。   
  
例：   
DIR C:\;D:\;E:\F:\   
以上的命令相当于   
DIR C:\   
DIR D:\   
DIR E:\   
DIR F:\   
当然还有些特殊的符号但他们的使用范围很小我就不再这里一一的说明了。   
  
批处理与变量   
  
在批处理中适当的引用变量将会使你所编制的程序应用面更广。批处理每次能处理的变量从%0~%9共10个。   
其中%0默认给批处理的文件名使用。除非在使用SHIFT命令后%0才能被%1所替代。引用shift命令的例子,如果把%1前面多加上一个%0那么结果如下：   
  
C:\>SHIFT.BAT 1 2 3 4 5 6 7 8 9 10 11   
SHIFT.BAT 1 2 3 4 5 6 7 8 9   
1 2 3 4 5 6 7 8 9 10   
2 3 4 5 6 7 8 9 10 11   
  
系统是如何区分每个变量的呢，系统区分变量的规则为字符串中间的空格，即只要发现空格就把空格前面的字符当作一个变量而空格后面的字符则作为另一个变量。如果你的变量是一个当中包含空格的长目录名这时候你需要用上一节特殊符号中所讲到的引号将他圈起来。   
  
例：   
批处理内容为：   
ECHO %1   
ECHO %2   
ECHO %3   
输入命令：   
C:\>TEST "Program Files" Program Files   
Program Files   
Program   
Files   
  
在一个复杂的批处理中又可能同时使用的变量会超过10个这时候会和系统的规则相冲突那么这个问题怎么解决呢？   
  
在系统中还有一种变量称之为环境变量（使用SET命令可以查看当前系统的环境变量）如当前系统目录是%windir%或%SystemRoot%等。   
  
当同时使用的参数超过10个的时候，我们可以把某些在后面的程序中还要调用的变量保存为环境变量。   
  
具体用法如 SET A=%1（类似定义变量，这样的变量还可以赋值给其他的变量）这样我们就命名了一个新的环境变量A 在调用变量A的时候要%A%这样调用，环境变量不受SHIFT命令影响。如果要改变一个环境变量需要重新对其设置才能改变。当然也可以进行变量与变量之间的传递来达到目的。下面我们来看一个例子。   
  
批处理如下：   
ECHO OFF   
SET PASS=%1   
SHIFT   
SET PASS1=%1   
SHIFT   
ECHO %PASS% %PASS1% %1 %2 %3 %4 %5 %6 %7 %8 %9   
SHIFT   
ECHO %PASS% %PASS1% %9   
SET PASS=%PASS1% 变量的传递   
SET PASS1=%9   
SHIFT   
ECHO %PASS% %PASS1% %9   
  
使用命令：C:\>TEST A B 3 4 5 6 7 8 9 10 K L   
A B 3 4 5 6 7 8 9 10 K //注意：这一行显示了11个变量   
A B L //在使用了3次SHIFT之后%9变成了L   
B L //变量的传递后的结果   
  
咳。。。做人真累啊。半夜回到家还想写点东西。不知道有多少人会看呢？不管了，别人不把他当成宝贝，咱自己留着当宝贝总是行的。   
大一的时候看了篇关于如何书写bat文件的文章。那时候看得很仔细，大概都看懂了。不过都２年了，基本上没有碰过bat文件，所以忘的都差不多了。今天是离家前的倒数第二个晚上了，突然想重温下bat文件的语法，格式，所以网上找了些资料来，研究研究，把我看懂的东西尽量写的白话一点,人人能看懂.希望能和看这篇文章的人一起学习进步．   
大家知不知道默认共享这回事？这东西用不着的地方可就不是好东西了．所以就要删掉．但这东西是每次系统重起后都会重新创建的．所以每次都要重新打开cmd重新删掉一下．极为麻烦．但有了批处理文件就不一样了，先把命令输入到批处理文件中，然后加入到启动项中，每次启动就会自动运行，免去了每次输入命令的麻烦．至于怎么写这个批处理文件嘛，到文章的结尾我会告诉大家的，但不知道在我写完之前会不会睡着呢？～   
如何创建批处理文件？   
不要听了批处理文件就感到很神气，其实这东西很简单的．你用过记事本没有？用过？　好的．将记事本打开，什么都不用写，然后选择文件，保存．保存类型选择所有文件，文件名则命名为　＊．bat　这个＊代表是文件名，你可以随便的起．保存好之后，看看你保存的地方，会出现一个白色窗口里有个黄色齿轮的图标．这东西就是你创建的批处理文件，双击他就可以运行，但他现在由于里面没有输入任何命令，所以他运行了并不会做任何事情．当我们想往这个＊.bat文件中添加东西时，只要右键选择他，然后选择编辑，就可以打开记事本往内输入命令了．   
批处理文件中的命令是什么？   
批处理文件中的命令暂时先可以理解为dos命令，等稍后深入理解了以后再进行解释．批处理顾名思义就是一大堆东西堆在一起处理．换句话说就是往里面写一条条dos命令，然后按顺序挨个执行，效果跟你在cmd里敲dos命令是一个效果．只不过用批处理写好之后，要运行只要双击下就可以运行了．而不用再一遍一遍的重复的往里面打命令．这就是批处理文件的好处．   
除了运行dos命令之外，他还可以支持选择结构if，循环结构for，goto等，与c有点类似，但远没有c全面，并且编写语言十分不规范．   
批处理语法：   
先讲个最基本的东西　@echo off   
echo的意思是回旋，这里的意思就是回显，echo　off意思就是关闭回显．前面的@表示echo off 这一行也不会回显你可以试着去掉@和整行去掉．@的另一个功能是在执行完批处理文件时自动恢复命令回显。如果第一句用的是ECHO OFF，那么在执行完批处理文件以后不会显示命令提示符。   
举个例子：如果我们先创建一个１．bat文件，往里面输入：   
dir然后保存在c：＼下．然后我们运行cmd，进入c盘根目录下，输入１．bat，则显示：   
C:>dir   
驱动器 C 中的卷没有标签。   
卷的序列号是 0C5D-07FF   
C: 的目录   
2004-08-25 00:45 <DIR> WINDOWS   
2004-08-25 00:51 <DIR> Documents and Settings   
...........   
C:   
如果将１．bat内容修改成   
echo off   
dir   
然后再在cmd里输入１．bat，则会显示   
C:>echo off　　　　　　　//因为运行了echo　off，所以没有显示dir命令，直接显示了结果   
驱动器 C 中的卷没有标签。   
卷的序列号是 0C5D-07FF   
C: 的目录   
2004-08-25 00:45 <DIR> WINDOWS   
2004-08-25 00:51 <DIR> Documents and Settings   
．．．．．   
C:   
如果将１．bat文件修改成：   
@echo off   
dir   
则显示成：   
C:>1.bat //与前面不同，并没有显示echo　off，原因是加了＠，所以不显示＠后面的内容．又因为加   
//了echo　off，所以后面的命令不显示，直接显示结果   
驱动器 C 中的卷没有标签。   
卷的序列号是 0C5D-07FF   
C: 的目录   
2004-08-25 00:45 <DIR> WINDOWS   
2004-08-25 00:51 <DIR> Documents and Settings   
．．．．   
C:   
通过以上的比较，相信你对echo　off命令已经充分掌握了．现在已经１点了．．．写得累啊！！！洗澡睡觉去了   
－－－－－－－－－－－－－－－－－－－－－－   
早晨６点就被拉起来了．．．我命苦啊，那么继续写吧   
－－－－－－－－－－－－－－－－－－－－－－   
接下来讲的是call命令：   
call是打电话的call，而不是＂喔靠＂的靠：）．call的意思为调用．假如有２个批处理文件a.bat和b.bat．如果我想运行 a.bat的当中运行b.bat．如何运行呢？其实很简单，只要在a.bat文件中输入call命令，则可以在a.bat运行的当中，运行b.bat，等 b.bat运行完之后，继续执行a.bat   
call命令格式：   
CALL [drive:][path]filename [batch-parameters]   
batch-parameters 指定批处理程序所需的命令行信息。   
举例如下，我们在c盘根目录下创建a．bat文件，内容为:   
echo　this is a.bat   
call d:b.bat   
echo done   
然后在d盘根目录下创建b.bat，内容为：   
echo this is b.bat   
保存后，打开cmd，进入c盘根目录，然后输入1.bat，显示如下：   
C:>a.bat   
C:>echo this is a.bat   
this is a.bat   
C:>call d:b.bat   
C:>echo this is b.bat   
this is b.bat   
C:>echo done   
done   
由例子很容易看出来，是先运行a.bat的内容，直到遇到callb.bat后，则调用b.bat，运行完b.bat后，再返回a.bat接着运行call　b.bat后面的echo　done语句，直至将a.bat的所有批处理命令运行完．   
注：里面有个[batch-parameters]里面的所指的参数是什么，知道的朋友能否告知，不胜感激．   
PAUSE 命令   
  
暂停批处理程序的执行并显示一条消息，提示用户按任意键继续执行。只能在批处理程序中使用该命令。   
rem命令：   
表示此命令后的字符为解释行（注释），不执行，只是给自己今后参考用的（相当于程序中的注释）．   
同时你可以用两个冒号来替代rem．如：　::等同于一个rem．但他们有个区别，就是用::作注释的话他是不回显的，即使你打ech