## Echo：

#显示当前是 echo off 状态还是 echo on 状态

echo off #从下一行开始关闭回显

@echo off #从本行开始关闭回显。一般批处理第一行都是这个

echo on #从下一行开始打开回显

echo #显示当前是 echo off 状态还是 echo on 状态

echo. #输出一个”回车换行”，空白行

#(同echo, echo; echo+ echo[ echo] echo/ echo\)

## @：关闭单行回显

## Dir：显示文件目录

dir #显示当前目录中的文件和子目录

dir /a #显示当前目录中的文件和子目录，包括隐藏文件和系统文件

dir c: /a:d #显示 C 盘当前目录中的目录

dir c: /a:-d #显示 C 盘根目录中的文件

dir c:\ /b/p #/b只显示文件名，/p分页显示

dir \*.exe /s #显示当前目录和子目录里所有的.exe文件

## cd：访问

cd\ #进入根目录

cd #显示当前目录

cd /d d:\sdk #可以同时更改盘符和目录

## rd：

rd abc #删除当前目录里的 abc 子目录，要求为空目录

rd /s/q d:\temp #删除 d:\temp 文件夹及其子文件夹和文件，/q安静模式

## del：删除

del d:\test.txt #删除指定文件，不能是隐藏、系统、只读文件

del /q/a/f d:\temp\\*.\* #删除 d:\temp 文件夹里面的所有文件，包括隐藏、只读、系统文件，不包括子目录

del /q/a/f/s d:\temp\\*.\* #删除 d:\temp 及子文件夹里面的所有文件，包括隐藏、只读、系统文件

## ren：重命名

ren d:\temp tmp #支持对文件夹的重命名

## cls：清屏

## type：显示文件内容

type c:\boot.ini #显示指定文件的内容，程序文件一般会显示乱码

type \*.txt #显示当前目录里所有.txt文件的内容

## copy：复制文件

copy c:\test.txt d:\test.bak #复制 c:\test.txt 文件到 d:\ ，并重命名为 test.bak

copy 1.txt + 2.txt 3.txt #合并 1.txt 和 2.txt 的内容，保存为 3.txt 文件。如果不指定 3.txt ，则保存到 1.txt

copy test.txt + #复制文件到自己，实际上是修改了文件日期

copy con test.txt #从屏幕上等待输入，按 Ctrl+Z 结束输入，输入内容存为test.txt文件。con代表屏幕，prn代表打印机，nul代表空设备

## title：设置cmd窗口的标题

title 新标题 #可以看到cmd窗口的标题栏变了

## ver：显示系统版本

## label 和 vol：设置卷标

vol #显示卷标

label #显示卷标，同时提示输入新卷标

label c:system #设置C盘的卷标为 system

## pause：暂停命令

## rem 和 :: 注释命令

## date 和 time：日期和时间

date #显示当前日期，并提示输入新日期，按\"回车\"略过输入

date /t #只显示当前日期，不提示输入新日期

time #显示当前时间，并提示输入新时间，按\"回车\"略过输入

time /t #只显示当前时间，不提示输入新时间

## goto 和 : 跳转命令

:label #行首为:表示该行是标签行，标签行不执行操作

goto label #跳转到指定的标签那一行

## find (外部命令)：查找命令

find \"abc\" c:\test.txt #在 c:\test.txt 文件里查找含 abc 字符串的行。如果找不到，errorlevel 返回码为1

find /i “abc” c:\test.txt #查找含 abc 的行，忽略大小写

find /c \"abc\" c:\test.txt #显示含 abc 的行的行数

## more (外部命令)：逐屏显示

more c:\test.txt #逐屏显示 c:\test.txt 的文件内容

## tree：显示目录结构

tree d:\ #显示D盘的文件目录结构

## &：顺序执行多条命令，而不管命令是否执行成功

## &&：顺序执行多条命令，当碰到执行出错的命令后将不执行后面的命令

find \"ok\" c:\test.txt && echo 成功

如果找到了\"ok\"字样，就显示\"成功\"，找不到就不显示

## ||：顺序执行多条命令，当碰到执行正确的命令后将不执行后面的命令

find \"ok\" c:\test.txt || echo 不成功

如果找不到\"ok\"字样，就显示\"不成功\"，找到了就不显示

## |：管道命令

dir \*.\* /s/a | find /c \".exe\" #管道命令表示先执行 dir 命令，对其输出的结果执行后面的 find 命令

该命令行结果：输出当前文件夹及所有子文件夹里的.exe文件的个数

type c:\test.txt|more #这个和 more c:\test.txt 的效果是一样的

## > 和 >>：输出重定向命令

> #清除文件中原有的内容后再写入

>> #追加内容到文件末尾，而不会清除原有的内容。主要将本来显示在屏幕上的内容输出到指定文件中。指

定文件如果不存在，则自动生成该文件

type c:\test.txt >prn #屏幕上不显示文件内容，转向输出到打印机

echo hello world>con #在屏幕上显示hello world，实际上所有输出都是默认 >con 的

copy c:\test.txt f: >nul #拷贝文件，并且不显示\"文件复制成功\"的提示信息，但如果f盘不存在，还是会显示出错信息

copy c:\test.txt f: >nul 2>nul #不显示”文件复制成功”的提示信息，并且f盘不存在的话，也不显示错误提示信息Echo ^^W ^> ^W>c:\test.txt

生成的文件内容为 ^W > W #^ 和 > 是控制命令，要把它们输出到文件，必须在前面加个 ^ 符号

## <：

从文件中获得输入信息，而不是从屏幕上。一般用于 date time label 等需要等待输入的命令

@echo off

echo 2005-05-01>temp.txt

date <temp.txt

del temp.txt

这样就可以不等待输入直接修改当前日期

## %0 %1 %2 %3 %4 %5 %6 %7 %8 %9 %\*：命令行传递给批处理的参数

%0 #批处理文件本身

%1 #第一个参数

%9 #第九个参数

%\* #从第一个参数开始的所有参数

批参数(%n)的替代已被增强。您可以使用以下语法:

%~1 #删除引号(\")，扩充 %1

%~f1 #将 %1 扩充到一个完全合格的路径名

%~d1 #仅将 %1 扩充到一个驱动器号

%~p1 #仅将 %1 扩充到一个路径

%~n1 #仅将 %1 扩充到一个文件名

%~x1 #仅将 %1 扩充到一个文件扩展名

%~s1 #扩充的路径指含有短名

%~a1 #将 %1 扩充到文件属性

%~t1 #将 %1 扩充到文件的日期/时间

%~z1 #将 %1 扩充到文件的大小

%~$PATH : 1 - 查找列在 PATH 环境变量的目录，并将 %1。扩充到找到的第一个完全合格的名称。如果环境变量名未被定义，或者没有找到文件，此组合键会扩充到空字符串

可以组合修定符来取得多重结果:

%~dp1 #只将 %1 扩展到驱动器号和路径

%~nx1 #只将 %1 扩展到文件名和扩展名

%~dp$PATH:1 - 在列在 PATH 环境变量中的目录里查找 %1，并扩展到找到的第一个文件的驱动器号和路径。

%~ftza1 #将 %1 扩展到类似 DIR 的输出行。

可以参照 call/? 或 for/? 看出每个参数的含意：

echo load \"%%1\" \"%%2\">c:\test.txt #生成的文件内容为 load \"%1\" \"%2\"。批处理文件里，用这个格式把命令行参数输出到文件

## if：判断命令

if \"%1\"==\"/a\" echo 第一个参数是/a

if exist c:\test.bat echo 存在c:\test.bat文件

if not exist c:\windows ( echo 不存在c:\windows文件夹 )

if exist c:\test.bat ( echo 存在c:\test.bat ) else ( echo 不存在c:\test.bat )

if /i \"%1\" equ \"/a\" echo 第一个参数是/a #/i 不区分大小写，equ 和 == 是一样的，其它运算符参见 if/?

## setlocal 和 endlocal：设置”命令扩展名”和”延缓环境变量扩充”

SETLOCAL ENABLEEXTENSIONS #启用\"命令扩展名\"

SETLOCAL DISABLEEXTENSIONS #停用\"命令扩展名\"

SETLOCAL ENABLEDELAYEDEXPANSION #启用\"延缓环境变量扩充\"

SETLOCAL DISABLEDELAYEDEXPANSION #停用\"延缓环境变量扩充\"

ENDLOCAL #恢复到使用SETLOCAL语句以前的状态

**“命令扩展名”默认为启用**

**“延缓环境变量扩充”默认为停用**

**批处理结束系统会自动恢复默认值**

可以修改注册表以禁用"命令扩展名"，详见 cmd /? 。所以用到"命令扩展名"的程序，建议在开头和结尾加上 SETLOCAL ENABLEEXTENSIONS 和 ENDLOCAL 语句，以确保程序能在其它系统上正确运行"延缓环境变量扩充"主要用于 if 和 for 的符合语句，在 set 的说明里有其实用例程

## Set：设置变量

引用变量可在变量名前后加 % ，即 %变量名%

set #显示目前所有可用的变量，包括系统变量和自定义的变量

echo %SystemDrive% #显示系统盘盘符。系统变量可以直接引用

set p #显示所有以p开头的变量，要是一个也没有就设errorlevel=1

set p=aa1bb1aa2bb2 #设置变量p，并赋值为 = 后面的字符串，即aa1bb1aa2bb2

echo %p% #显示变量p代表的字符串，即aa1bb1aa2bb2

echo %p:~6% #显示变量p中第6个字符以后的所有字符，即aa2bb2

echo %p:~6,3% #显示第6个字符以后的3个字符，即aa2

echo %p:~0,3% #显示前3个字符，即aa1

echo %p:~-2% #显示最后面的2个字符，即b2

echo %p:~0,-2% #显示除了最后2个字符以外的其它字符，即aa1bb1aa2b

echo %p:aa=c% #用c替换变量p中所有的aa，即显示c1bb1c2bb2

echo %p:aa=% #将变量p中的所有aa字符串置换为空，即显示1bb12bb2

echo %p:\*bb=c% #第一个bb及其之前的所有字符被替换为c，即显示c1aa2bb2

set p=%p:\*bb=c% #设置变量p，赋值为 %p:\*bb=c% ，即c1aa2bb2

set /a p=39 #设置p为数值型变量，值为39

set /a p=39/10 #支持运算符，有小数时用去尾法，39/10=3.9，去尾得3，p=3

set /a p=p/10 #用 /a 参数时，在 = 后面的变量可以不加%直接引用

set /a p=”1&0″ #”与”运算，要加引号。其它支持的运算符参见set/?

set p= #取消p变量

set /p p=请输入

屏幕上显示”请输入”，并会将输入的字符串赋值给变量p

注意这条可以用来取代 choice 命令

注意变量在 if 和 for 的复合语句里是一次性全部替换的，如

```

@echo off

set p=aaa

if %p%==aaa (

echo %p%

set p=bbb

echo %p%

)

```

结果将显示

```

Aaa

aaa

```

因为在读取 if 语句时已经将所有 %p% 替换为aaa

这里的"替换"，在 /? 帮助里就是指"扩充"、“环境变量扩充”

可以启用”延缓环境变量扩充”，用 ! 来引用变量，即 !变量名!

```

@echo off

SETLOCAL ENABLEDELAYEDEXPANSION

set p=aaa

if %p%==aaa (

echo %p%

set p=bbb

echo !p!

)

ENDLOCAL

```

结果将显示

```

aaa

bbb

```

还有几个动态变量，运行 set 看不到

%CD% #代表当前目录的字符串

%DATE% #当前日期

%TIME% #当前时间

%RANDOM% #随机整数，介于0~32767

%ERRORLEVEL% #当前 ERRORLEVEL 值

%CMDEXTVERSION% #当前命令处理器扩展名版本号

%CMDCMDLINE% #调用命令处理器的原始命令行

可以用echo命令查看每个变量值，如 echo %time%

注意 %time% 精确到毫秒，在批处理需要延时处理时可以用到

## start：

批处理中调用外部程序的命令，否则等外部程序完成后才继续执行剩下的指令

## call：

批处理中调用另外一个批处理的命令，否则剩下的批处理指令将不会被执行

有时有的应用程序用`start`调用出错的，也可以`call`调用

## choice (外部命令)

选择命令，让用户输入一个字符，从而选择运行不同的命令，返回码errorlevel为1234……

win98里是choice.com

win2000pro里没有，可以从win98里拷过来

win2003里是choice.exe

choice /N /C y /T 5 /D y>nul

延时5秒

## assoc 和 ftype：文件关联

assoc 设置'文件扩展名'关联，关联到'文件类型'

ftype 设置’文件类型’关联，关联到’执行程序和参数’

当你双击一个.txt文件时，windows并不是根据.txt直接判断用 notepad.exe 打开

而是先判断.txt属于 txtfile '文件类型’

再调用 txtfile 关联的命令行 txtfile=%SystemRoot%\system32\NOTEPAD.EXE %1

可以在"文件夹选项"→"文件类型"里修改这2种关联

assoc #显示所有'文件扩展名'关联

assoc .txt #显示.txt代表的'文件类型'，结果显示 .txt=txtfile

assoc .doc #显示.doc代表的'文件类型'，结果显示 .doc=Word.Document.8

assoc .exe #显示.exe代表的'文件类型'，结果显示 .exe=exefile

ftype #显示所有'文件类型'关联

ftype exefile #显示exefile类型关联的命令行，结果显示 exefile=\"%1\" %\*

assoc .txt=Word.Document.8 #设置.txt为word类型的文档，可以看到.txt文件的图标都变了

assoc .txt=txtfile #恢复.txt的正确关联

ftype exefile=\"%1\" %\* #恢复 exefile 的正确关联

如果该关联已经被破坏，可以运行 command.com ，再输入这条命令

## pushd 和 popd：切换当前目录

@echo off

c: & cd\ & md mp3 #在 C:\ 建立 mp3 文件夹

md d:\mp4 #在 D:\ 建立 mp4 文件夹

cd /d d:\mp4 #更改当前目录为 d:\mp4

pushd c:\mp3 #保存当前目录，并切换当前目录为 c:\mp3

popd #恢复当前目录为刚才保存的 d:\mp4

## for：循环命令

这个比较复杂，请对照 for/? 来看

for %%i in (c: d: e: f:) do echo %%i #依次调用小括号里的每个字符串，执行 do 后面的命令

**注意：**%%i，在批处理中 for 语句调用参数用2个%，默认的字符串分隔符是"空格键"，"Tab键"，"回车键"

for %%i in (\*.txt) do find \"abc\" %%i

#对当前目录里所有的txt文件执行 find 命令

for /r . %%i in (\*.txt) do find \"abc\" %%i

#在当前目录和子目录里所有的.txt文件中搜索包含 abc 字符串的行

for /r . %%i in (.) do echo %%~pni

#显示当前目录名和所有子目录名，包括路径，不包括盘符

for /r d:\mp3 %%i in (\*.mp3) do echo %%i>>d:\mp3.txt

#把 d:\mp3 及其子目录里的mp3文件的文件名都存到 d:\mp3.txt 里去

for /l %%i in (2,1,8) do echo %%i

#生成2345678的一串数字，2是数字序列的开头，8是结尾，1表示每次加1

for /f %%i in ('set') do echo %%i

#对 set 命令的输出结果循环调用，每行一个

for /f \"eol=P\" %%i in ('set') do echo %%i

#取 set 命令的输出结果，忽略以 P 开头的那几行

for /f %%i in (d:\mp3.txt) do echo %%i

#显示 d:\mp3.txt 里的每个文件名，每行一个，不支持带空格的名称

for /f \"delims=\" %%i in (d:\mp3.txt) do echo %%i

#显示 d:\mp3.txt 里的每个文件名，每行一个，支持带空格的名称

for /f \"skip=5 tokens=4\" %%a in ('dir') do echo %%a

#对 dir 命令的结果，跳过前面5行，余下的每行取第4列

每列之间的分隔符为默认的"空格"，可以注意到 dir 命令输出的前5行是没有文件名的

```

for /f \"tokens=1,2,3 delims=- \" %%a in ('date /t') do (

echo %%a

echo %%b

echo %%c

)

```

对 date /t 的输出结果，每行取1、2、3列

第一列对应指定的 %%a ，后面的 %%b 和 %%c 是派生出来的，对应其它列

分隔符指定为 - 和"空格"，注意 delims=- 后面有个"空格"

其中 tokens=1,2,3 若用 tokens=1-3 替换，效果是一样的

for /f \"tokens=2\* delims=- \" %%a in ('date /t') do echo %%b

取第2列给 %%a ，其后的列都给 %%b

## subst (外部命令):映射磁盘。

subst z: \server\d #这样输入z:就可以访问\server\d了

subst z: /d #取消该映射

subst #显示目前所有的映时

## xcopy (外部命令):文件拷贝

xcopy d:\mp3 e:\mp3 /s/e/i/y #复制 d:\mp3 文件夹、所有

## **参考：**

https://blog.csdn.net/qq\_43331089/article/details/124222342