**windows批处理程序中的特殊字符**

* 原文: <http://blog.csdn.net/yuanchao3333/article/details/3855415>

## @

* //隐藏命令的回显。

## ~

//在for中表示使用增强的变量扩展； 在set中表示使用扩展环境变量指定位置的字符串； 在set/a中表示按位取反。

## %

* 使用两个%包含一个字符串表示引用环境变量。比如一个%time%可以扩展到当前的系统时间;
* 单个%紧跟0-9的一个数字表示引用命令行参数；
* 用于for中表示引用循环变量；
* 连续的两个%表示执行时脱为一个%。

## ^

* //取消转义字符，即将所有转义字符的转义作用关闭。比如要在屏幕显示一些特殊的字符,比如> >> | ^等时,就可以在其前面加一个^符号来显示这个^后面的字符了,^^就是显示一个^,^|就是显示一个|字符了;
* 在set/a中是按位异；
* 在findstr/r的[]中表示不匹配指定的字符集。

## &

* //命令连接字符。比如我要在一行文本上同时执行两个命令,就可以用&命令连接这两个命令；
* 在set/a中是按位与。
* 即使第一条命令失败，后面的命令也会执行

### 实例1，第一条命令执行成功后继续执行第二条命令

|  |
| --- |
|  |

## 实例2.第一条命令执行失败后继续执行第二条命令

|  |
| --- |
|  |

## \*

* //代表任意个任意字符,就是我们通常所说的"通配符";比如想在c盘的根目录查找c盘根目录里所有的文本文件(.txt),那么就可以输入命令"dir c:/\*.txt"；
* 在set/a中是乘法。比如"set/a x=4\*2",得到的结果是8；
* 在findstr/r中表示将前一个字符多次匹配。

## ()

//命令包含或者是具有优先权的界定符吧,比如for命令要用到这个(),我们还可以在if,echo等命令中见到它的身影；

## -

* //范围表示符,比如日期的查找,for命令里的tokens操作中就可以用到这个字符；
* 在findstr/r中连接两个字符表示匹配范围；
* -跟在某些命令的/后表示取反向的开关。

## +

* //主要是在copy命令里面会用到它,表示将很多个文件合并为一个文件,就要用到这个+字符了；
* 在set/a中是加法。

## |

* //管道符。就是将上一个命令的输出,作为下一个命令的输入."dir /a/b | more"就可以逐屏的显示dir命令所输出的信息;
* 在set/a中是按位或；
* 在帮助文档中表示其前后两个开关、选项或参数是二选一的。

#### 实例1：在当前目录中查找png文件，效果如下

|  |
| --- |
|  |

#### 实例2.用netstat命令查找所有tcp协议

|  |
| --- |
|  |

## :

//标签定位符,可以接受goto命令所指向的标签。比如在批处理文件里面定义了一个":begin"标签,用"goto begin"命令就可以转到":begin"变迁后面来执行批处理命令了。

## " "

* //界定符,在表示带有空格的路径时常要用""来将路径括起来,在一些命令里面也需要" "符号；
* 在for/f中将表示它们包含的内容当作字符串分析；
* 在for/f "usebackq"表示它们包含的内容当作文件路径并分析其文件的内容；
* 在其它情况下表示其中的内容是一个完整的字符串，其中的>、>>、<、&、|、空格等不再转义。

## /

* //表示其后的字符（串）是命令的功能开关（选项）。比如"dir /s/b/a-d"表示"dir"命令指定的不同的参数；
* 在set/a中表示除法。

## >

* //命令重定向符,将其前面的命令的输出结果重新定向到其后面的设备中去，后面的设备中的内容被覆盖。比如可以用"dir > lxmxn.txt"将"dir"命令的结果输出到"lxmxn.txt"这个文本文件中去；
* 在findstr/r中表示匹配单词的右边界，需要配合转义字符/使用。

## >>

//命令重定向符。将其前面的命令的输出结果重新定向到其后面的设备中去，后面设备中的内容没有被覆盖。

## <

* //将其后面的文件的内容作为其前面命令的输入。
* 在findstr/r中表示匹配单词的左边界，需要配合转义字符/使用。

## =

* //赋值符号,用于变量的赋值。比如"set a=windows"的意思意思是将"windows"这个字符串赋给变量"a"；
* 在set/a中表示算术运算,比如"set /a x=5-6\*5"。

## /

* //这个"/"符号在有的情况下,代表的是当前路径的根目录.比如当前目录在c:/windows/system32下,那么你"dir /"的话,就相当与"dir c:/"
* 在findstr/r中表示正则转义字符。

## ''

* 在for/f中表示将它们包含的内容当作命令行执行并分析其输出；
* 在for/f "usebackq"中表示将它们包含的字符串当作字符串分析。

## .

* //
* 在路径的/后紧跟或者单独出现时：
* 一个.表示当前目录；
* 两个.表示上一级目录；
* 在路径中的文件名中出现时：
* 最后的一个.表示主文件名与扩展文件名的分隔。

## &&

### //连接两个命令,当&&前的命令成功时,才执行&&后的命令；

### 实例1：打开命令行，输入di && dir，效果如下

|  |
| --- |
|  |

### 实例2.

|  |
| --- |
|  |

## ||

### 连接两个命令,当||前的命令失败时,才执行||后的命令。

### 实例1.打开命令行，输入di || dir，效果如下

|  |
| --- |
|  |

### 实例2.当第一条命令成功执行，不会执行第二条命令

|  |
| --- |
|  |

#### 可见第二条命令没有执行

### && 和||的综合应用

#### 输入一条命令，如果成功，输出成功，如果失败输出失败

|  |
| --- |
|  |
|  |

##### 因为我的电脑上面没有j盘所以dir命令执行失败此时会执行||后面的语句输出failed。下面的例子，因为电脑的用户目录是存在的所以会执行&&后面的语句而||后面的语句不执行

## $

//在findstr命令里面表示一行的结束。

## []

* 在帮助文档表示其中的开关、选项或参数是可选的；
* 在findstr/r中表示按其中指定的字符集匹配。

## ?

* //在findstr/r中表示在此位置匹配一个任意字符；
* ?在路径中表示在此位置通配任意一个字符；
* 紧跟在/后表示获取命令的帮助文档。

## !

* //当启用变量延迟时,使用!!将变量名扩起来表示对变量值的引用；
* 在set /a中表示逻辑非。比如set /a a=!0,这时a就表示逻辑1。