8. ЛИНЕЙНИ КОМАНДНИ ФАЙЛОВЕ, КОМАНДА IF

Файловете с разширение BAT са текстови файлове, съдържащи последователност от команди, които се изпълняват от командния интерпретатор. Често BAT фаловете се наричат командни файлове, файлове за пакетна обработка, а самият процес на изпълнението им пакетна обработка. Тези файлове могат да съдържат произволна последователност от вътрешни и външни команди на операционната система включително и управляващи команди. Могат да се създават с команда copy con име.bat, със системния текстов редактор EDIT или с произволен текстов редактор.

В командните файлове се налага използването на някой допълнителни команди, които нямат приложение директно в конзолата.

```
REM [коментар]
```

Командата указва, че всичко след rem е коментар и не се изпълнява от командния интерпретатор. Най-често коментарите се използват или от тези, които са разработили файла, или са ориентир за потребителите, за които е предназначен. Възможно е редове с настройки в конфигурационни файлове да са поставени в коментар. При премахването на командите rem от началото на реда настройките се активират.

```
PAUSE
```

Командата прекъсва изпълнението на пакетния файл и извежда съобщението:

Press any key to continue

Изпълнението продължава след натискане на произволен клавиш.

```
ECHO [съобщение]
```

Командата извежда текста след нея. Използва се главно в случаите, когато има нужда да се изведат подсещащи потребителя указания за следващите действия или уточнение на вече получени резултати.

Приложението на командата ECHO директно в конзолата най-често е свързано с пренасочване към файл.

Пример 1.

```
ECHO It will be added at the end of the file >> hello.txt
```

Възможно е с командата ECHO да се изведе и стойност на параметър.

Пример 2.

```
ECHO current user is %USERNAME% >> hello.txt
```

Съществува още един синтаксис на командата ECHO.

```
ECHO [ON | OFF]
```

В този вариант командата разрешава (ON) или забранява (OFF) извеждането на комадите от командния файл преди тяхното изпълнение. Стойността по подразбиране е ON. Препоръчително е в процеса на разработка на изпълнимия файл извеждането да е включено и след като файлът е завършен и тестван да се постави @ECHO OFF на първия ред. Символът „@” поставен в началото на ред, указва командния интерпретатор да не извежда командата на този ред при изпълнението.

Пример 3.

```
@ECHO off
REM My first bach file
ECHO Hello World!
PAUSE
```

Ако приемем, че горния текст е записан във файл с име TEST1.BAT в главната директория на C устройство, ще имаме следния изход от изпълнението му.

```
C:\TEST1.BAT
My first bach file
Press any key to continue
```

Условна команда IF

Тази команда ще бъде разгледана в 3 основни варианта.

Вариант 1. Файлов синтаксис.

```
IF [NOT] EXIST filename (command)
IF [NOT] EXIST filename (command) ELSE (command)
```

Двата реда представляват съкратения и пълния (с клауза ELSE) запис на командата IF.

Ако файлът съществува условието е изпълнено. Когато е необходимо отрицание на условието може да се използва клаузата NOT. Ако command се състои само от една команда не е задължително да се поставят скоби. Когато command съдържа повече от една команда е необходимо да се поставят скоби и като разделител между командите да се използва символа „§”. Следващите два примера са еквивалентни.

Пример 1.

```
IF EXIST filename.txt (Echo deleting filename.txt § Del
filename.txt) ELSE (Echo The file was not found.)
```

В този пример всичко е изписано на един ред.

Пример 2.

```
IF EXIST filename.txt (
    Echo deleting filename.txt
    Del filename.txt
 ) ELSE (
    Echo The file was not found.
 )
```

В този пример трябва да се спазва позицията на скобите. Например ако преместим първата отваряща скоба на следващия ред ще се получи грешка.

Вариант 2. Синтаксис за работа с низове.

```
IF [/I] [NOT] item1==item2 (command) ELSE (command)
IF [/I] item1 опер_за_сравнение item2 (command)
ELSE (command)
```

По подразбиране при сравнението на низове се прави разлика между главни и малки букви (т. н. case sensitive). Параметърът „I” указва да не се прави разлика между главни и малки букви (case insensitive).

Оператора за сравнение „опер_за_сравнение” е един от следните:

EQU : равено

NEQ : различно

LSS : по-малко (<)

LEQ : по-малко или равно (<=)

GTR : по-голямо (>)

GEQ : по-голямо или равно (>=)

Използването на трибуквено означение се налага поради това че знаците < и > са запазени за пренасочване.

Когато се сравняват низове се извършва лексикографско сравнение, а когато операндите са числа се третират като такива. Това се илюстрира със следните 2 примера.

Пример 1.

```
if 5 lss 44 (echo числото 5 е по-малко от числото 44)
```

Пример 2.

```
if NOT ”5” lss ”44” (echo низът 5 не е по-малък от низа 44)
```

Операндите при сравнението могат да бъдат и параметри.

Пример 3.

```
if %USERNAME%==Administrator (echo Hello Administrator)
```

Използването на параметри ще бъде уточнено по-долу.

```
IF [NOT] DEFINED променлива команда
```

Ако в условието се използва клаузата `DEFINED` може да бъде проверено наличието (дали е дефинирана) на променлива.

Вариант 3. Синтаксис за проверка на грешки

Всяка команда или програма генерира код при завършване на своята работа. Той има стойност 0 ако програмата е завършила успешно работата си или друго число ако е възникнала грешка при изпълнението. Всяка стойност отразява точно определена грешка. Като правило се използват само положителни стойности, но се срещат и изключения. Операционната система записва в променливата на средата `ERRORLEVEL` кода на завършване на последната програма. Тъй като външните команди в MS-DOS са програми те също връщат код на грешката. Вътрешните команди не връщат код на грешкка (с някои изключения като например командата „move”). Ако се налага да се провери как е завършила вътрешна команда може командата да се изпълни в нова обвивка и да се прихване нейния код. Например командата:

```
cmd /c dir /wrong_parameter
```

ще върне код на грешка 1, защото не съществува параметър с име `wrong_parameter`. Стойността на грешката записана в променливата `ERRORLEVEL` се проверява с помощта на командата:

```
IF [NOT] ERRORLEVEL номер команда
```

Този запис трябва да се разбира като „ако Errorlevel >= номер изпълни команда". Тоест условието е изпълнено ако последната команда е е завършила действието си с грешка чиито номер е по-голям или равен на числото „номер".

Когато е необходимо да се провери дали грешката има точно определена стойност се използва оператор за сравнение.

```
IF %ERRORLEVEL% equ 0 echo true if not error
```

В операционните системи на компанията Microsoft съществува специален файл за пакетна обработка, който се изпълнява автоматично по време на зареждане на системата. По подразбиране този файл се намира в главната директория на системното устройство (най-често устройство „C:"). Името на файла е абривиатура от „automatic execution" (автоматично изпълнение) и се нарича AUTOEXEC.BAT. Предназначението на файла е да се присвоят стойности на важни за системата параметри или да се стартират системни приложения.

Във версиите на ОС Windows NT, Windows Server 2000, Windows Server 2003, Windows XP еквивалентния файл се нарича Autoexec.nt и се намира в директорията %SystemRoot%\system32 , където параметътът на средата %SystemRoot% е директорията, в която е инсталирана ОС. Най-често файлът се намира в C:\Windows\System32. Autoexec.nt се изпълнява всеки път когато потребителя стартира командния интерпретатор или друга MS-DOS програма. AUTOEXEC.BAT разположен в главната директория също се стартира при зареждане на системата, но от него се изпълняват само командите PATH и SET. Параметрите, които се инициализират тук са глобални и са достъпни за всички потребители. Ако параметри на обвивката са дефинирани в Autoexec.nt, то те са достъпни само за текущия потребител.

В Windows 7 по-подразбиране е забранено изпълнението на AUTOEXEC.BAT. То може да бъде разрешено с модификация на системния регистър.

HKEY_CURRENT_USER\Software\Microsoft\WindowsNT\CurrentVersion\Winlogon

ParseAutoexec

1 = autoexec.bat се изпълнява

0 = autoexec.bat не се изпълнява

ПАРАМЕТРИ ПРЕДАВАНИ ОТ КОМАНДНИЯ РЕД НА ПАКЕТНИЯ ФАЙЛ.

Параметрите предадени от командния ред (или друг изпълним файл) могат да бъдат обработени във файла посредством служебните параметри %1 %2 %3 %4 %5 ...%255.

Параметърът %* представлява масив от параметрите %1 %2 %3 %4 %5 ...%255.

Параметърът %0 прихваща командата. Той не участва в масива %*.

Пример.

Нека файлът MyScript.bat съдържа следния код.

```
@echo off
```

```
echo Command is: %0
echo First parameter is: %1
echo Second parameter is: %2
echo Third parameter is: %3
```

Файлът може да бъде изпълнен по следния начин:

```
C:> MyScript January 1234 "Some value"
```

Изхода от изпълнението му е следния:

```
Command is: MyScript
First parameter is: January
Second parameter is: 1234
Third parameter is: "Some value"
```

Параметърът %0 се обработва по начина по който е изписан на командния ред (т. Е. MyScript а не MyScript.bat). Всички параметри са разделени от интервал. Последния параметър "Some value" е един параметър а не два защото е изписан в кавички.

Параметрите са позиционни. В посочения пример те са индексирани по следния начин:

```
 %0          %1      %2    %3
C:> MyScript January 1234 "Some value"
```

Съществува команда, която променя позицията на параметрите в командния файл.

```
SHIFT [/n]
```

Измества индекса на параметрите с индекс по-голям от n с един напред. Тук n е зело число в интервала 1 - $#. По подразбиране n=1

Пример.

При изпълнение на файла abc.bat по следния начин

```
abc f1.txt f2.txt f3.txt
```

Параметрите са индексирани както следва:

```
 %0   %1    %2     %3
abc f1.txt f2.txt f3.txt
```

След изпълнението на командата shift имат следните индекси:

```
      %0    %1     %2
abc f1.txt f2.txt f3.txt
```

Името на изпълнимия файл „abc“който е достъпен с %0 преди изпълнение на командата след това ства недостъпен. При повторно изпълнение на shift става недостъпен и параметъра „f1.txt“ Действието на командата е необратимо.

ЕТИКЕТИ В КОМАНДЕН ФАЙЛ.

Избрани редове в командните файлове могат да бъдат използвани като етикети при следния синтаксис:

```
:test_label
```

Целта на такава конструкция е когато е необходимо да се наруши естествения (последователен) ред на изпълнение на команди, да се изпълнява преход към избран етикет.

```
GOTO label
```

Командата изпълнява преход към етикет в текущия файл.

Пример.

```
set prom=%1
:One
shift
if [%1]==[] goto Two
copy %1 %prom%
goto One
:Two
set prom=
echo Happy end!
```

В този команден файл като първи параметър се задава директория а всички следващи параметри са файлове. Командата копира файловете в директорията.

```
GOTO:eof
```

Съществува предварително дефиниран етикет „:eof“ във всеки команден файл, който сочи края на файла и може да бъде използван без да се дефинира. Горния пример може да бъде написан и така:

```
:One
shift /2
if [%1]==[] goto :eof
copy %2 %1
goto One
```

```
CALL [устройство:][път]команден_файл [параметри]
```

Стартира указания файл за пакетна обработка.