

## BATCH Special characters

- ◆ Remember: < > : " / \ | ? \*
  - Consumed by CMD
  - Used for **NAME** separation or redirection or expansion
- ◆ Meet: , ; & && || ^ ( )
  - Consumed by Commands
  - Used for COMMAND parameters or sequence or group
  - , ; → parameter separators
  - & && || → command sequences
  - ^ → set & | ^<> ( ) back to character
  - ^ → "eat" the following <EOL>
  - ( ) → multi-line command(s)

## BATCH Multiple commands

- ◆ Type `echo AA & echo BB`
  - Character "&" is a BATCH command separator
  - Type `md CC/C & MD DD` → fails because first fails
- ◆ Type `md CC/C && md DD`
  - Keyword "&&" is a BATCH command separator
  - Will only execute 2nd command on success of 1st
- ◆ Type `md CC/C || md DD`
  - Keyword "||" is a BATCH command separator
  - Will only execute 2nd command on failure of 1st

## BATCH Commands on multiple lines

- ◆ Type `(md AA`
  - Will not execute command
  - Asks for more commands until we type )
  - May be nested
- ◆ To split a command on multiple lines
  - Escape final <EOL> with ^
  - Type `(md ^` ← beware the space separator
  - Type `AA,^`
  - Type `BB^`
  - Type `cc)` ← beware the space separator

## BATCH Var Content modification

- ◆ Use `%VAR:str1=str2% substitution pattern`
  - `str1` may start with "\*"
  - `str2` may be empty
- ◆ Use `%VAR:~index,qty% reduction pattern`
  - `index` & `qty` are decimals (i.e. %PATH:~50,100%)
  - `index` starts at 0 (~0 means beginning)
  - `qty` may be empty (rest of the string)
  - -3 means (`VAR_length` - 3)

## BATCH Several numeric bases

- ◆ Default is decimal
  - Type `SET /A TEN=10`
  - Displays 10
- ◆ Hexadecimal with 0x prefix
  - Type `SET /A TEN=0x10`
  - Displays 16 (...the Hex 10)
- ◆ Octal with 0 prefix
  - Type `SET /A TEN=010`
  - Displays 8 (...the Octal 10)
  - Beware 08 and 09 are not allowed

## BATCH Script argument filters

- ◆ Pathnames filtering expressions
  - %~f is full pathname
  - %~d is drive letter
  - %~p is folder pathname
  - %~f is file name
  - %~x is file extension
- Example:
  - %~dpnx0 or %~f0 is full name of the script file

## BATCH IF command

```
IF condition (
    Commands
) ELSE (
    Commands
)
```

### ◆ Condition keywords

- NOT inverse the boolean result
- EXIST files & folders
- DEFINED variables
- ERRORLEVEL errors in commands
- A==B identical values (!strings or numbers)

### ◆ Extended comparison operators

- EQU =
- NEQ !=
- LSS <
- LEQ <=
- GTR >
- GEQ >=

## BATCH IF and VARIABLES

### ◆ Fill adv2.bat

```
@echo off
setlocal enabledelayedexpansion
set /a num1=%1
set /a num2=%2

if %num1% LEQ %num2% (
    set /a offset=!num2!-!num1!
) else (
    set /a offset=!num1!-!num2!
)
echo offset is %offset%
```

Use ! in place of %

### ◆ Variable EnabledDelayedExpansion

- Variables evaluation inside <command> of IF
- Modify the behavior of the CMD interpreter (second pass)
- Code must be !VAR! in place of %VAR%

## BATCH FOR command options

FOR [options] each-item IN collection DO command

### ◆ Command structure

- /D to filter folder names (i.e. not file names)
- /R to explore recursively a following root folder (i.e. /R root)
- /L to iterate on numbers, collection is (start, step, stop)
- /F to set the collection as a file or a command output

### ◆ modifiers

- /A to generate each-item as a number
- /I to make processing NOT case sensitive
- /Q no error messages (i.e. Quiet)
- /T introduces a delay at each iteration
- /C command is a string containing spaces

## BATCH FOR /F command filters

FOR /F [filters] each-item IN collection DO command

### ◆ Filter expressions

- eol=c eliminate characters after « c » in the line
- skip=n eliminate n first lines
- delims=, defines several items (%%a, %%b, %%c ...)
- usebackq allow folder/files names with spaces  
→ use back-quotes

## BATCH FOR and VARIABLES

### ◆ Fill adv1.bat

```
@echo off
setlocal EnableDelayedExpansion
set count=0
for %i in (1 2 3) do (
    set /a count+=1
    set /a var=%i
    echo loop -%i- : count is %count%, var is %var%
    echo delayed: count is !count!, var is !var!
    echo.
)
echo Outside loop: count is %count%, var is %var%
endlocal
```

Use ! in place of %

unchanged for  
Set and %%

### ◆ Variable EnabledDelayedExpansion

- Variables evaluation inside <command> of FOR
- Modify the behavior of the CMD interpreter (second pass)
- Code must be !VAR! in place of %VAR%

## BATCH Command FINDSTR

### ◆ Fill adv4.bat

```
@echo off
Echo foo > text.txt
Echo Bar >> text.txt
Echo FooBar >> text.txt
type text.txt | findstr /^[fF]
```

Lines starting with f or F

### ◆ FINDSTR accepts metacharacters

- . One of Any character
- \* Zero or any number of any character
- ^ Start of line
- \$ End of line
- [class] Any character in the group i.e. [Aez12]
- [^class] Any character NOT in the group [^ezEZ]
- [x-y] Any character in the interval [0-9]
- \x escape for literal usage of metachar 'x'
- \<xyz « xyz » at start of word
- xyz> « xyz » at end of word