Basics of Batch File Programming

Hi Everyone, just finished a tutorial on basics of batch programming language i hope it will helpful for every learner i am also still learning it

Please Note: All the commands and batch file codes are for educational purpose only.Please keep a backup of files before attempting to use these codes.

Well, they have extensions: **.bat**(dot bat)

Now, for learning batch file programming you need to know some commands that are necessary to know.

You may try to go for help but that will give you a list of limited commands. Although to get started they are enough. As for good purposes like clearing temp files, deleting some special files, keeping a log file, creating files.....

So to get started, lets start with echo command:

1. Echo: This is the command used to print information, whether to the console(command prompt) or directed place.

This command will print everything whether u give it any value or not . i.e it will also print the command u had typed in the batch file. So to stop this and enhance the look we always turn off auto echoing by:

@echo off
Now it will print only those things which are passed to echo. Like a batch to delete all text files:

```
@echo off

echo Deleting text files.....

del /f *.txt

echo Deleted!!
```

Save this as deleter.bat and run it. Make sure to change the extension of any important text file u have in the folder because /f switch will force it to delete without asking for permission.

2. Pause: this single command pauses the operation , prints" Press any key to continue..." and waits for the user to press a key. This is usually important because batch files tend to close as soon as they complete without giving u a chance to read it. So at last its

important to put pause or u will have to run batch file from console rather than just double clicking it.

3. Rem: This command is used to save comments for the other batch file reader i.e. used for commenting(as in programming rerms). It is also used to save info to config.sys file(No need to bother about this).

4. Path: path i.e. different variables set as path . Like to go to system drive just type cd /d %windir% and u will be in system directory from any drive. /d is used to change drive with the directory. They are important because not always the system drive will be c: it can be any other depending on systems, so using path variables proves useful since this frees us from the headache of knowing system dive to get the full list, just type set at command prompt.

5. Output and Input Redirection: Generally in all programming languages there are 3 standard streams: standard Input Stream i.e. keyboard,Standard Output Stream i.e. monitor or console and Standard Error stream i.e. where error are displayed.But often we want to write to file and other places, for this we need to redirect output.

In batch file programming, we can do that with '>' operator(without Quotes). Single > will erase everything and write. Double > i.e. >> will add the text at last of the file.

Eg. you want to store the list of all files and folders(with list of all files within a folder and so on)
you can just type:

dir /s >> new.txt
The /s switch compels dir command to show all files and folder, and all files within the folder, and if any folder then all contents of that folder and so on until the chain gets finished.

Similarly, if u want to copy one files content to another file, instead of using copy command u can do following:

type file1.txt >> file2.txt -- this will copy the contents of text file1 at the end of file2.txt . We use type command because it shows all the text at once unlike more command which shows page wise and needs pressing enter for other ones.

Echo command can also be used similarly.

Now for redirecting input from other commands we use piping '|'
This redirects the output of 1 command as input of other command. like:

dir /s | more - since this command shows all the directory and its sub directories and its files ... we are only able to read
the last few folders, passing it with more will allow u read all the folders.

Note:'|' symbol can be obtained by pressing **(shift + \)**

The Redirection can also be done to other devices like Printers.

DEVICE NAME USED DEVICE

AUX Auxiliary Device (COM1)

CLOCK$ Real Time Clock

COMn Serial Port(COM1, COM2, COM3,COM4)

CON Console(Keyboard, Screen)

LPTn Parallel Port(LPT1, LPT2, LPT3)

NUL NUL Device(means Nothing)

PRN Printer

Say for example, you want to print the results of directory listings,
then you can simply give the following command:

c:\windows>dir *.* > prn
Now, nul device literally means nothing. You can direct the output of such commands which just shows work done to prevent it from showing to the user. Like coping command shows the number of file copied. If u don't want to show it or keep it just:

copy x.xx y.yy > nul
6. Parameters: We can also provide batch file, arguments at the run time. Like, passing file names for reading by user at run time.The Parameters can be obtained by using %number.

The arguments are delimited or separated by space. So 1st parameter can be obtained by:

%1

2nd parameter can be obtained by: %2
3rd parameter can be obtained by: %3
.....
9th parameter can be obtained by: %9
And if we need parameters more than 9 then just use 'shift' keyword. This will replace parameters to left by 1 i.e. 2nd

parameter will become 1 , 3rd will become 2nd ..... and the 9th is empty.
But the 1st parameter will be lost. So instead of using %2 and %3.
We can use shift each time we use the 1st Parameter.So the 2nd will become 1st and we can again use it.

7. Set command:

the variable of batch files- Set command is used to create environmental variables.
This will enable u to do 2 things:

i) Lets you to store any text,command or path address in a variable which can be expanded to the value anytime.

ii) Lets you store any text or address entered by the user at run-time. Like the choice of user.

Use-age: set variable_name= text,command or path address.

Note: for user input from keyboard use switch /P with set command and the message to display after the equals sign.

Like:

```
@echo off
echo Welcome to deleter or hider.
echo type d for deleting and h for hiding.
set /P ch=Enter Choice:
if %ch%==d goto deleter
if %ch%==h goto hide
:deleter
echo enter file name with extensions to delete
set /P file=name:
del /f %file% >> echo
goto end
:hide
```

echo enter file name with extension to hide
set /P file=name:
attrib +h +s +r %file% >> echo
:end
pause
Here the text Enter Choice: and name: will be shown and user will be asked to enter the text.

goto command takes the cmd to the label pointed by the goto. And we can create labels by putting a colon before Label name.

Like :label_name
The switch /A lets u put an expression on the right hand side of variable. I.e. the value of variable will be value of expression.

8.for command: This command lets u run any command for a number of times or for a number of files. Just like looping. Syntax: for %variable in(sets of files like *.txt for all text files and address for other folders may also be give here) Do command to be executed for each fileHere %variable means declaring any variable name like %i or %l ..

And for each loop %%i will contain the 1 file name among the set specified in the bracket.

For Eg, to rename all the .txt file to .virus just type:

for %%i in (*.txt) do ren *.txt *.vir
Although here for command was not exactly needed because directly ' ren *.txt *.vir ' would have renamed all but changing contents of dll files or rather say for corrupting files this can prove useful.

9. If command: This command is used to do comparisons or checking of parameters and user inputs and work accordingly. Moreover it can also check whether a file exists or not and also checking of variables or strings.

For File checking: **if EXIST file_name command_to_perform_if_true**

like
if EXIST c:\windows\notepad.exe EchoNotepad exists.
This command will print Notepad Exists if notepad is in directory windows.

Note: this command cannot be used for directories. I will search for directories, but now work only on files.

We can also use else with if but else has to be in the same line. And some commands

need new lines at the end to work, like del command. So better use **()** brackets and write commands in bracket like:

if exist C:\windows\notepad.exe (del /f notepad.exe) else (echo notepad doesn't exists)
For parameter checking:

```
if %1==c goto cdrive
if %1==d goto ddrive
:cdrive
copy %2 c:
exit
:ddrive
copy %2 d:
```

We can also check if parameters are passed by: if %1=="" echo No Parameter, Similarly u can check any thing u want just keep sure u use double equals sign.

For both the usage of if we can also use the NOT clause as:
if not exist c:\test echo test not created yet.

10. Choice command: Before we learn how to make use of the CHOICE command, we need to what error levels really are. Now Error levels are generated by programs to inform about the way they finished or were forced to finish their execution. For example, when we end a program by pressing CTRL+C to end a program, the error level code evaluates to 3 and if the program closes normally,then the error level evaluates to 0.

These numbers all by themselves are not useful but when used with the IF ERROR LEVEL and the CHOICE command, they become very useful. The CHOICE command takes a letter or key from the keyboard and returns the error level evaluated when the key is pressed. The general syntax of the CHOICE command is:

CHOICE "The message to user goes here " [/C:keys][/S][/N][/T:key,secs]
The string part is nothing but the string to be displayed when the CHOICE command is run.

The /C:keys defines the possible keys to be pressed. If options are not mentioned then the default Y/N keys are used instead.
For example the command:

CHOICE /C:ABCD
Defines A, B, C and D as the possible keys. During execution if the user presses a undefined key, he will hear a beep sound and the program will continue as coded.

The /S switch makes the possible keys defined by the CHOICE /c flag case sensitive. So

it means that if the /S flag is present then A and a would be different.

The /N switch, if present shows the possible keys in brackets when the program is executed. If the /N switch is missing then, the possible keys are not shown in brackets. Only the value contained in the double quotes is shown.

/T:key,secs defines the key which is taken as the default after a certain amount of time has passed.

For Example:

CHOICE "Choose Option A or B" /C:AB /T:B.5
The above command displays Choose Options and if no key is pressed for the next 5 seconds, then it chooses B. Now to truly combine the CHOICE command with the IF ERROR LEVEL command, you need to know what the CHOICE command returns. The CHOICE command is designed to return an error level according to the pressed key and its position in the /C switch. To understand this better, consider the following example:

CHOICE /C:XYZA
Now remember that the error level code value depends on the key pressed. This means that if the key X is pressed, then the error level is 1, if the key Y is pressed then the error level is 2, if Z is pressed then error level is 3 and if A is pressed then error level is 4.

Now let us see how the IF ERROR LEVEL command works. The general syntax of this command is:

IF [NOT] ERRORLEVEL number command to execute.

This statement evaluates the current error level number. If the condition is true then the command is executed.

For Example:

IF ERRORLEVEL 3 ECHO Yes
The above statement prints Yes on the screen if the current error level is 3.

The important thing to note in this statement is that the evaluation of an error level is true when the error level us equal or higher than the number compared.

For Example, in the following statement:

IF ERRORLEVEL 2 ECHO YES
The condition is true if the error level is > or = 2.
Now that you know how to use the CHOICE and ERROR LEVEL IF command together, you can now easily create menu based programs.

The following is an example of such a batch file which asks the User to choose a typing Pad:

```
@echo off
echo .
echo .
echo Welcome to typing Pad selection
echo 1.Notepad
echo 2.WordPad
echo 3.Microsoft Word
echo 4.Command prompt word
CHOICE "Choose Pad" /C:1234 /N
if ERRORLEVEL 4 edit %1
if ERRORLEVEL 3 start c:\Program Files\..i.e. path of word.exe.. %1
if ERRORLEVEL 2 start wordpad %1
if ERRORLEVEL 1. notepad %1
:END
```

Note the order of if statements, since in errorlevel comparision it accepts the value for >= so decreasing order was necessary.

NOTES:

1. To Copy the batch file itself just use %0 as the source in the copy Command. This will enable you to copy the batch file to places like **startup, startmenu, desktop,my documents** etc.

2. At many places you will find that computer is not taking the parameters, in that case use double **'%%'**, as in the batch file computer often computer often deletes one % and hence two is necessary for functioning. I guess at most places you will have to use double **%%**.

3. Before starting to use batch files and commands I recommend you to at least read the help file of command by: **Command_Name /? JUST ADD /?** after the command and it will open its help file which contains details of each switch available.

4. Work on Commands like **NET, NBTSTAT, NETSTAT, SHUTDOWN, FORMAT, CALL** etc. These Commands are useful, like to create Users, Administrators use **NET USER** COMMAND. JUST TYPE: NET USER /? and read the whole help file. If you find that help files are
too long then just save it in a file by using the Command: **NET USER/? > NET.TXT**

Happy Learning

Ajay Singh Negi