Command Grouping

Command grouping allows you to logically group a set of commands together by enclosing them in parentheses. The parentheses are similar in function to the BEGIN and END block statements in some programming languages.

There are two primary uses for command grouping. One is to execute multiple commands in a place where normally only a single command is allowed. For example, suppose you want to execute two different REN commands in all subdirectories of your hard disk. You could do it like this:

```
[c:\] global ren *.wx1 *.wxo
[c:\] global ren *.tx1 *.txo
```

But with command grouping you can do the same thing in one command (enter this on one line):

```
[c:\]global (ren *.wx1 *.wxo & ren *.tx1 *.txo)
```

The two REN commands enclosed in the parentheses appear to GLOBAL as if they were a single command, so both commands are executed for every directory, but the directories are only scanned once, not twice.

This kind of command grouping is most useful with the EXCEPT, FOR, GLOBAL, and IF commands. When you use this approach in a batch file you must either place all of the commands in the group on one line, or place the opening parenthesis at the end of a line and place the commands on subsequent lines. For example, the first two of these sequences will work properly, but the third will not:

```
for %f in (1 2 3) (echo hello %f & echo goodbye %f)
for %f in (1 2 3) (
        echo hello %f
        echo goodbye %f
)
for %f in (1 2 3) (echo hello %f
echo goodbye %f)
```

The second common use of command grouping is to redirect input or output for several commands without repeatedly using the redirection symbols. For example, consider the following batch file fragment which uses the ECHO command to create a file (with >), and to append to the file (with »):

```
echo Data files %_date > filelist
dir *.dat >> filelist
echo. >> filelist
echo Text files %_date >> filelist
dir *.txt >> filelist
```

Using command grouping, these commands can be written much more simply. Enter this example on one line:

```
(echo Data files %_date & dir *.dat & echo. & echo Text files
```

```
        %_date & dir *.txt) > filelist
```

The redirection, which appears outside the parentheses, applies to all the commands within the parentheses. Because the redirection is performed only once, the commands will run slightly faster than if each command was entered separately. The same approach can be used for input redirection and for piping.

You can also use command grouping in a batch file or at the prompt to split commands over several lines. This last example is like the redirection example above, but is entered at the prompt. Note the “More?” prompt after each incomplete line. None of the commands are executed until the command group is completed with the closing parenthesis. This example does **not** have to be entered on one line:

```
        [c:\] (echo Data files %_date
        More? dir *.dat
        More? echo.
        More? echo Text files %_date
        More? dir *.txt) > filelist
        [c:\]
```

A group of commands in parentheses is like a long command line. The total length of the group may not exceed 2,047 characters, whether the commands are entered from the prompt, an alias, or a batch file. The limit **includes** the space required to expand aliases and environment variables used within the group. In addition, each line you type at the normal prompt or the **More?** prompt, and each individual command within the line, must meet the usual length limit of 1,023 characters.

From:
https://osfree.org/doku/ - **osFree wiki**

Permanent link:
https://osfree.org/doku/doku.php?id=en:docs:cmd:other:grouping&rev=1400914581

Last update: **2014/05/24 06:56**