Article: How to write a Batch File to loop through files
Date: 17/02/2012
Posted by: HeelpBook Staff
Source: Link
Permalink: Link

DOS – How to write a batch file to loop through files

I used to write **DOS** batch files like this all the time, but after a while you start to forget the **DOS** syntax and the tricks. So this post is almost for as much for me as it is for anyone searching for how to do this.

The goal is to create a **DOS** batch file that can loop through a set of files and call another batch file or executable and pass the filename as the argument.

In this example, I'm going to use two (2) batch files, but in your real world solution, the second batch file might be an executable **(.exe)**.

Let's examine the syntax of the DOS "**for**" command:

```
FOR %variable IN (set) DO command [command-parameters]

%variable  Specifies a single letter replaceable
           parameter.
(set)      Specifies a set of one or more files.
           Wildcards may be used.
command    Specifies the command to carry out for each
           file.
command-parameters
           Specifies parameters or switches for the
           specified command.
```

In addition, you can use "**FOR /D**", "**FOR /R**", and "**FOR /F**". The **/d** instructs the command to only return directories, the **/r** indicates recursive, and **/f** only returns files (in case a directory name and filename both meet the set pattern matching).

*So we'll create our first **DOS** batch file (dubbed "doit.bat") with the below syntax:*

```
for /f %%a IN ('dir /b *.txt') do call runner.bat %%a
```

In this for statement, we use the /f flag to only return filenames. Our variable name will be **"%%a"** and our set command will be **"dir /b *.txt"**. This set command returns a bare directory listing (no filesize, attributes, etc. just filenames) that match the ***.txt** pattern. You could also include the full directory path here, just in case this **DOS** batch file isn't located in the same directory as your txt files. Finally we do a "do call" to tell the program to call our second batch file (*runner.bat*).

One neat trick is that you can parse out the filename without the extension and the extension as two (2) separate variables by changing the command to something like this:

```
for /f %%a IN ('dir /b *.txt') do call runner.bat %%~na %%~xa
```

The above loop passes the filename (without extension) as argument 1 and the file extension (**".txt"** in this case) as argument 2. This is done by using the **"~n"** and **"~x"** variable enhancements.

Now, in **runner.bat** (our second file), we can read in the arguments by using **%1** and **%2** where %1 is the first argument (only argument if we use the first example) and **%2** as the second argument.

To test, we can write out the data such as:

```
echo %1-test%2
```

This would return the filename with the "**-test**" string inserted before the file extension and after the filename.

You can learn more by just opening up your command window and typing "help for".